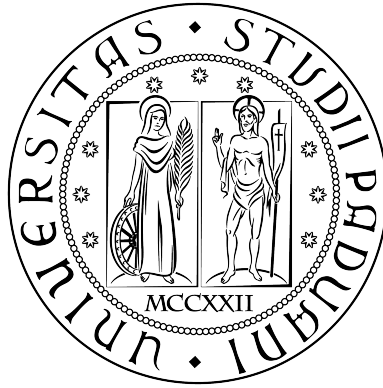




UNIVERSITÀ DEGLI STUDI DI PADOVA
Dipartimento di Tecnica e Gestione dei Sistemi Industriali

Corso di Laurea Magistrale in Ingegneria Gestionale

**A Machine Learning Approach to
Lead Time Prediction for Custom Components
in Yacht Interior Outfitting**

Supervisor:
Ch.mo Prof. Luigi Salmaso

Candidate: Matteo Danieli
Matricola: 2157906

Co-supervisor:
Ch.ma Prof.ssa Marta Disegna

Academic Year 2025/2026

a Muma e Papi

Ringraziamenti

Desidero ringraziare il Prof. Luigi Salmaso, relatore di questo lavoro, per aver accolto con favore la mia proposta di tesi.

Desidero ringraziare in modo particolare la Prof.ssa Marta Disegna, correlatrice, che ha seguito il mio lavoro passo dopo passo con competenza, pazienza e una disponibilità mai scontata. I suoi consigli sono stati preziosi in ogni fase, dalla definizione del problema alla lettura dei risultati, e le sono grato per la fiducia che mi ha sempre dimostrato.

Desidero ringraziare Paolo, il mio tutor aziendale, per avermi dato la possibilità di svolgere questo lavoro di tesi e per aver stimolato la mia creatività.

Infine, desidero ringraziare la mia famiglia e i miei amici per l'affetto e il sostegno con cui mi hanno accompagnato in questi anni.

Vicenza, 6 ottobre 2026

Matteo Danieli

Abstract

In the outfitting of luxury yacht interiors every piece is made to a drawing for a single vessel, and planning relies on processing durations fixed by rule or left to experience. This thesis attempts to estimate with machine learning the finishing lead time, the time a piece spends in painting, from the movements that the production management portal of a firm in the sector records for each piece. The data cover four commissions, three new builds and a refit, and 23 months of entries into painting, from which 14,245 finishing cycles are rebuilt. Eight algorithms are compared with four baselines, simple rules that require no training, over 17 consecutive months, with the models trained only on the cycles already closed at the date of prediction, and the differences are submitted to a statistical test.

At entry into painting the chosen model, a random forest, errs by 14.95 days on average, against 15.24 for the family median, the median duration by material family, wood or lacquer, and the difference cannot be told apart from zero. Grouping the pieces before prediction, by domain criteria or with an unsupervised algorithm, does not improve on the single model in an established way.

The most solid result is the estimate updated during processing. When the first pieces of the batch a piece travels with return, the updated estimate for the pieces still in painting cuts the error of the family median by 27.9%. For three pieces out of four, in fact, the recorded exit date is that of the batch, and the decision of the department on which batch to process and when does not appear in the data.

For a firm that records movements at batch level, the updated estimate and a date to promise computed as the prediction plus a margin, which leaves 34.2% of the pieces late as the family median does, against 66.8% for the rule in use, can already feed a planning tool. Recording the start and the end of the processing of each piece would make observable its duration and the order in which the department processes the batches. The model would then have information specific to each piece and many more independent observations to learn from, with the possibility of predicting better.

Table of Contents

Abstract

Introduction	1
1 Problem Setting and Industrial Context	3
1.1 Objectives and research questions	3
1.1.1 The problem: planning without knowing how long processing takes	3
1.1.2 Defining lead time	3
1.1.3 The main objective	4
1.1.4 The operational objectives	5
1.1.5 The research questions	5
1.1.6 The experimental design on segmentation	6
1.2 The industrial context	7
1.2.1 Supply chain management in Engineer-to-Order production	8
1.2.2 Product complexity in bespoke interiors	8
1.2.3 Sector positioning and commission portfolio	9
1.2.4 Geographical presence and site logic	10
1.2.5 The production process and the material flow	11
1.2.6 The role of the project manager	12
1.2.7 The design office and the drawings	12
1.2.8 A sector that tolerates craft work	13
1.2.9 The digital tracking infrastructure	13
1.2.10 The observed commissions and the data base	15
2 Machine Learning	17
2.1 Learning paradigms	17
2.1.1 Supervised learning	18
2.1.2 Unsupervised learning	18
2.2 The machine learning process	19
2.2.1 Preprocessing the features	20
2.3 Supervised models for prediction	21
2.3.1 Linear and regularised models	21
2.3.2 k -nearest neighbours	22
2.3.3 Support vector regression	22
2.3.4 Decision trees	23

2.3.5	Ensemble methods	25
2.3.6	Artificial neural networks	26
2.4	Generalisation and data splitting	27
2.4.1	Underfitting and overfitting	27
2.4.2	The bias-variance trade-off	28
2.4.3	Parameters and hyperparameters	29
2.4.4	Data splitting and cross-validation	30
2.5	Model evaluation	31
2.5.1	Loss function and evaluation metric	31
2.5.2	Error measures	31
2.5.3	The baseline to measure against	32
2.5.4	Evaluating an unsupervised model	33
2.5.5	Interpretability	33
2.6	From prediction to decision	34
2.6.1	Two routes to a lead time	34
2.6.2	Sequencing and date quotation	35
3	Literature Review	37
3.1	The corpus	37
3.2	The classification framework	38
3.2.1	Data classes	39
3.2.2	Data origin	40
3.2.3	Methods	41
3.3	Data preparation and the recorded quantity	41
3.4	Predicting a duration or classifying a delay	42
3.5	Segmenting before predicting	43
3.6	Formulations that use the process sequence	44
3.7	Validation, references and tests	44
3.8	Interpretability	46
3.9	Position of this work	47
4	Data Processing and Feature Engineering	49
4.1	The exports and the fields used	49
4.2	The data in the framework of the literature	50
4.2.1	Why the indirect route is not viable	50
4.3	The portal vocabulary and the change of regime	51
4.4	The boundaries of finishing	51
4.5	Building the cycle	53
4.6	The target and the observation window	54
4.7	Reducing the dataset	56
4.7.1	A process that cannot be measured	57

4.8	The finish dictionary	58
4.9	Exploratory analysis	58
4.9.1	The distribution	58
4.9.2	The commissions	59
4.9.3	The trend over time	60
4.9.4	Material families and day of entry	61
4.10	The finish as a predictor	61
4.11	The other recorded fields	62
4.12	The candidate features	63
5	Model Development and Evaluation	65
5.1	How the predictions are judged	65
5.1.1	The baselines	65
5.1.2	Month-by-month evaluation	66
5.1.3	The hyperparameter search	67
5.1.4	When a difference is real	68
5.2	The choice of algorithm	69
5.3	The prediction at entry	71
5.3.1	Wood and lacquer	73
5.3.2	Robustness checks	74
5.3.3	After the first year of history	74
5.4	How much the protocol matters	75
5.4.1	Training on the cycles still open	76
5.4.2	A single split in time	77
5.5	Segmentation	78
5.6	Transferability across commissions	79
5.7	The updated prediction	81
5.8	Where the model errs	83
5.8.1	Why the model underestimates	85
5.8.2	Remedies for selection by closure	87
5.8.3	What the remedies depend on	89
5.9	What the model uses	91
5.10	From prediction to the date to promise	92
5.11	Other interventions tried	93
5.12	The limit of batch-level recording	95
6	Limitations and Future Developments	97
6.1	Limitations	97
6.1.1	Limitations of the data	97
6.1.2	Limitations of the model and of the evaluation	100
6.2	Future developments	102

6.2.1	Recording the processing of the piece	103
6.2.2	Harmonising the master data	105
6.2.3	More history	106
6.2.4	Developments of the model	106
6.2.5	A tool built on what is already measured	108
6.2.6	Dates for the piece, the item and the cabin	111
Conclusions		115
A Source Code Excerpts		119
A.1	Building the finishing cycle	119
A.2	The monthly folds of the honest protocol	121
A.3	The baselines and the model in each month	121
A.4	The updated prediction	122
A.5	Paired comparison and Holm correction	123
A.6	The date to promise	124
References		125

List of Figures

Figure 1.1	Product hierarchy, from the commission down to the individual piece.	9
Figure 1.2	Material flow of a piece, from design to installation. The highlighted steps make up the finishing lead time.	11
Figure 1.3	The three label formats in use, shown with fictitious data.	13
Figure 2.1	Taxonomy of machine learning paradigms.	17
Figure 2.2	The machine learning process: step zero, the six subsequent steps and the training, validation and test phase, specific to the supervised case.	19
Figure 2.3	Support vector regression: insensitivity tube and support vectors.	23
Figure 2.4	Illustrative example of a regression tree.	24
Figure 2.5	Bagging and random forests: parallel construction and averaging of predictions.	26
Figure 2.6	Gradient boosting: sequential construction on the residuals.	26
Figure 2.7	Illustrative example of a feedforward neural network.	27
Figure 2.8	Trade-off between bias and variance as model complexity varies.	29
Figure 2.9	Illustrative example of three data splitting protocols.	30
Figure 3.1	The three dimensions of the classification framework, adapted from Burggräf et al. (2020).	39
Figure 3.2	Share of publications using each data class. Data from Burggräf et al. (2020).	39
Figure 4.1	Map of the tracking phases, grouped by state of the piece.	53
Figure 4.2	How a finishing cycle is read from the movements of a piece.	53
Figure 4.3	Share of the physical pieces with each field filled, by commission.	58
Figure 4.4	Distribution of the finishing lead time on linear and logarithmic scales.	59
Figure 4.5	Finishing lead time by commission.	60
Figure 4.6	Monthly median of the finishing lead time, with its interquartile range, and cycles opened in each month by commission.	61
Figure 4.7	Finishing lead time of the most frequent finish codes, within the two families. Each code is shown with its commission.	61
Figure 4.8	Median finishing lead time of white lacquer and of all lacquer, by commission.	62

Figure 5.1 Mean absolute error of the eight algorithms on the honest protocol, with the family median and the standard rule. 69

Figure 5.2 Mean absolute error in each test month: the random forest, the family median and the standard rule. 71

Figure 5.3 (a) The same comparison, the random forest against the family median, under the leaking and the honest protocol. (b) Median eventual duration of the cycles of the last entry month before each of the 18 test months of the leaking protocol, still open at the boundary or already closed. 76

Figure 5.4 Change in mean absolute error of each segmentation configuration against the single global model A, mean over the 17 test months. 79

Figure 5.5 Mean absolute error on the pieces still in painting when the first pieces of their batch return: the updated model, the rule without learning, the refitted model, the static prediction made at entry and the baselines. 81

Figure 5.6 (a) Mean absolute error by band of actual duration, all test pieces. (b) Calibration: mean actual duration against mean prediction, by decile of the prediction. 83

Figure 5.7 (a) Permutation importance on the test months: increase in mean absolute error when a feature is permuted. (b) Mean absolute SHAP value on the log scale of the duration, over the test rows. 91

Figure 5.8 Mean absolute error of four oracles, each knowing in advance information not available at prediction time. 96

Figure 6.1 Illustrative timeline of three pieces of the same batch. (a) Today only the entry and the exit of the batch document are recorded, and every piece gets the same duration. (b) Two scans of each piece at the rack separate the waiting before processing, the processing and the waiting for the rest of the batch. Invented times. 103

Figure 6.2 Prototype of the batch view: the batches of NC-1 in painting, most urgent first, with the time already spent in painting, the range given at entry, the date to promise, the updated prediction of a batch whose first pieces have returned and a batch beyond the range. Historical data, anonymised, as of 16 July 2026. 108

Figure 6.3 Detail of batch B-17 of Figure 6.2: two pieces back on 13 July and nine still in painting, whose estimate has switched to the updated prediction, the 13 days to the first return plus the median remaining time of lacquer pieces. The estimate given at entry stays visible. Historical data, anonymised, as of 16 July 2026. 109

Figure 6.4 Illustrative scenario with piece-level recording: exploded view of a generic sideboard of an invented commission, NC-X, with the phase of each 112

piece and, for the selected piece, its finish, date to promise and range. The item is ready by the latest date to promise among the pieces still in painting. Invented data.

Figure 6.5 Illustrative scenario for a cabin of NC-X: plan with the week in 113 which each element is expected to be ready and, beside it, planned installation against the predicted ready-by date, with the elements ready after the planned start. Invented data.

List of Tables

Table 1.1	Hierarchy of the lead time definitions adopted in this thesis.	4
Table 1.2	Experimental design for assessing the effect of segmentation on finishing lead time prediction.	6
Table 1.3	Comparison between new build and refit programmes.	10
Table 1.4	The three fields describing a movement.	14
Table 1.5	Composition of the data base used for modelling.	15
Table 2.1	The steps of the machine learning process, with evaluation distinguished between the supervised and the unsupervised case.	19
Table 3.1	Applied studies reviewed, by sector, data used and predicted quantity.	38
Table 3.2	Data split, choice of hyperparameters, metrics and reference in the studies on recorded data.	45
Table 4.1	Size of the two exports, by commission.	49
Table 4.2	Fields used from the two exports and their role.	50
Table 4.3	Share of the movements recorded in the regime after the change.	51
Table 4.4	Openings without a recorded closing on 1 September 2026, by commission.	55
Table 4.5	From the exports to the modelling dataset.	56
Table 4.6	Pieces reaching the family filter, by dictionary family, with the share that has at least one closed cycle.	57
Table 4.7	Tracking of upholstery on the leather pieces.	57
Table 4.8	Descriptive statistics of the finishing lead time by commission, in calendar days.	59
Table 4.9	Candidate features, by group, and the moment at which they become known.	63
Table 5.1	Search spaces of the hyperparameters, by algorithm. Ranges marked “log” are explored on a logarithmic scale.	67
Table 5.2	The eight algorithms on the honest protocol: mean absolute error (mean over months) and difference from the random forest, positive when the algorithm errs less, with the p-value of the paired test and the one corrected over the seven comparisons with the forest.	69

Table 5.3	The prediction at entry against the baselines: reduction of the mean absolute error in the three averages, p-value of the paired test, Holm p-value over the four baselines and months won. The comparison with the family median is one of the two main questions and is judged on the nominal p-value.	71
Table 5.4	The prediction at entry with the forest and the two algorithms with the lowest error: mean absolute error and reduction of the error of the two main baselines (mean over months), on all pieces, on wood and on lacquer, with the p-value of the paired test. Regularised linear regression and CatBoost are rerun with the same hyperparameter search as in Table 5.2.	71
Table 5.5	The prediction at entry over the 17 test months, from October 2025, with at least a year of history in training, and from October 2025 without June and July: reduction of the error of the baseline (mean over months), p-value of the paired test and months won. The readings from October 2025 were decided after the results and are descriptive.	74
Table 5.6	One commission held out at a time: mean absolute error of the model and of the baselines on the held-out commission (days), and share of its rows whose finish code, taken without the commission, never appears in training.	80
Table 5.7	The updated prediction on the pieces still in painting at the first return of their batch: reduction of the error against the baselines (mean over months), p-value, Holm p-value over the four baselines and months won. The comparison with the family median is one of the two main questions and is judged on the nominal p-value.	81
Table 5.8	Error measures of the prediction at entry, mean over months. The mean error is prediction minus observed duration; the shares count the pieces underestimated or overestimated by more than seven days.	83
Table 5.9	Mean absolute error of the prediction at entry by commission and by material family: mean over the months in which the group has test rows and, in brackets, mean by piece. The mean error of the model is prediction minus observed duration.	83
Table 5.10	Diagnosis of the underestimation: mean absolute error, mean error and difference in error from the official model (positive when the variant errs less), mean over months.	85
Table 5.11	Remedies for selection by closure, mean over months. The difference from the family median is positive when the remedy errs less, and the Holm p-value is computed within the family of nine comparisons, the eight remedies and the uncapped accelerated failure time model. The main remedy was fixed before the results and is judged on the nominal p-value; the last two were added after them.	88

Table 5.12 The promised date: share of late pieces, mean delay among the late 92
pieces and mean earliness among the others, by piece. The margin of the model
is the stated quantile of the errors of the previous months.

Table 5.13 Interventions tried during development: difference in mean absolute 93
error from the model without the intervention, positive when the intervention
errs less, mean over months and p-value of the paired test. Dataset before
the corrections of Section 4.5 (13,384 cycles), leaking protocol, 18 test months,
random forest.

Table 5.14 Interventions measured on the honest protocol: difference in mean 94
absolute error from the official model, positive when the intervention errs less,
mean over months, with the p-value of the paired test and the one corrected
over the three feature groups.

Table 6.1 Limitations of the work and the developments that address them. 103

Introduction

In firms that design and produce to order, known as Engineer-to-Order, the product is designed for a customer after the sale and is not repeated (Wortmann, 1992; Gosling and Naim, 2009). The outfitting of luxury yacht interiors is an extreme case. A commission comprises from a few thousand to tens of thousands of pieces, made to drawing for a single vessel with materials and finishes chosen by the client, and every piece must arrive on board within the installation window set by the shipyard. Between rough production and installation, panels and cabinetry go through painting, where they stay for a time that varies from one piece to another and that planners know as a standard value or from experience. Without an estimate, a delay in this phase often emerges only when the piece is missing at installation.

The systematic review by Burggräf *et al.* (2020) distinguishes an indirect route, which derives the duration from a plan of the resources, from a direct one, which predicts it from the durations observed in the past (Section 2.6), and places feedback data from real production, combined with material attributes, among the areas of greatest potential. According to the scoping review by Koblasa and Urban (2026), covering 173 works, the error of machine learning predictions lies around 10%, and at 20% for activities with a strong human component, and applications outside highly automated sectors are few. Among the recent applied studies some concern the naval domain, none the interior outfitting of yachts (Section 3.9).

In evaluation, among the studies on recorded data a single one splits the observations in time order (Talonpoika, 2026), and just two test whether the difference between models can be told apart from chance (Steinberg *et al.*, 2022; Canedo Rosa, Gonçalves and Barbalho, 2025) (Section 3.7). Two other studies also measure the gain against a rule without learning, the mean of the training durations, and find it already more accurate than the estimate in use (Bender, Trat and Ovtcharova, 2022; Talonpoika, 2026). Recording errors, where they appear, are noise to be filtered or a limit to be declared (Bender, Trat and Ovtcharova, 2022; Flores-Huamán *et al.*, 2025), and none of the studies reviewed checks whether the recorded durations actually observe the unit the model has to predict.

This thesis attempts to estimate the finishing lead time, the time a piece spends in painting, with machine learning models trained on the movements of the pieces as recorded by the production management portal of an Italian manufacturer of yacht interiors, where the internship took place. The practical aim is to give the project manager an estimate of the date on which a piece will be available again,

and with it the possibility of acting before a delay reaches installation. The research questions, formulated in Chapter 1, concern the accuracy of the prediction at the moment of entry into painting compared with rules that require no training, the information that determines the duration and whether the recorded data observe it for the individual piece, the value of grouping the pieces before prediction, and the improvement of the estimate updated during processing, once part of the batch the piece travels with has returned.

The work follows the direct route. The data are the exports of the portal for four commissions, three new builds and a refit, with 23 months of entries into painting, from October 2024 to August 2026. From the event log 14,245 finishing cycles are rebuilt, each from the entry into painting to the first return, and their duration in calendar days is the target variable. Eight supervised learning algorithms are compared with four baselines that require no training, three medians and the estimate in use. The evaluation reproduces the use of the model. In each of 17 test months, training uses the cycles that entered before its first day and had already closed by then, and the model predicts the duration of those that enter painting during the month. The monthly differences between a model and a baseline are submitted to a statistical test, corrected when several comparisons answer the same question. How much the verdict depends on the design of the evaluation is measured as well, together with the transferability of the models from one commission to another and the use of the prediction to set a date to promise. The six chapters follow the order of the work.

1. Chapter 1 defines the problem, the finishing lead time and the research questions, and describes the industrial context and the tracking infrastructure from which the data come.
2. Chapter 2 introduces the machine learning fundamentals used in the work, from the models to their evaluation and to the step from prediction to decision.
3. Chapter 3 reviews the literature on lead time prediction in Engineer-to-Order production, from the data to the validation of the models, and positions the work against the studies reviewed.
4. Chapter 4 describes how the exports of the portal become the target variable: the boundaries of finishing, the construction of the cycle, the observation window, the filters, the exploratory analysis and the candidate features.
5. Chapter 5 develops and evaluates the models and answers the research questions.
6. Chapter 6 discusses the limitations of the work and the future developments, among them the recording of the individual piece and the use of the predictions in a planning tool.

The Conclusions summarise the answers and the contribution of the thesis, and Appendix A reports excerpts of the code.

Chapter 1

Problem Setting and Industrial Context

1.1 Objectives and research questions

1.1.1 The problem: planning without knowing how long processing takes

In yacht interior outfitting, planning rests on durations that nobody has ever measured. A painting cycle is assumed to take a typical number of days, and that figure sets the shipping dates, the finishing sequence and the on-board installation schedule. The assumption is practical but fragile, because it ignores the variability the product itself introduces. Two pieces identical on the drawing can spend weeks more or less in painting, depending on the material, the finish required and the queue they find in the department.

Delays therefore reach the project manager after they have already propagated. A piece that returns late from painting delays the installation of a cabin, and the window agreed with the shipyard is consumed by a problem that could, in principle, have been foreseen. The literature on lead time prediction in Engineer-to-Order settings locates the value of a reliable estimate in the possibility of identifying deviations from agreed delivery dates at an early production stage, while counter-measures can still be set (Burggräf *et al.*, 2020).

This thesis attempts to estimate those durations rather than assume them, using machine learning models trained on the recorded movement history of the pieces.

1.1.2 Defining lead time

The term lead time carries different meanings in the literature, and its scope has to be fixed before any model is built. In its general definition it is the number of minutes, hours or days that must elapse for an operation or process to be completed, before the desired action takes place (Burggräf *et al.*, 2020). In manufacturing the definition

narrows to the production lead time, the total time required to manufacture an item, covering order preparation, queue time, setup, run time, move time, inspection and put-away; for make-to-order products it runs from the release of the production order to shipment (Burggräf *et al.*, 2020).

At a finer level the lead time of an order breaks down into its operations, and the lead time of each operation divides into interoperation time and operation time. The first covers the wait after the previous operation, the transport and the queue before the next one, while the second comprises setup and processing proper (Burggräf *et al.*, 2020). In the data used here the two components cannot be told apart. The firm's production management portal records when a piece enters painting and when it leaves, no event marks the start of work on the piece, and every duration measured is therefore the sum of waiting and processing.

On this basis the thesis defines its object of study, the *finishing lead time*: the total time a piece spends in painting, from its dispatch for processing to its return, finished and ready for installation. Table 1.1 summarises the hierarchy of definitions adopted.

Level	Definition adopted
Lead time	Time that must elapse for an operation or process to be completed (Burggräf <i>et al.</i> , 2020).
Production lead time	Total time to manufacture an item, from order preparation to put-away; for make-to-order products, from order release to shipment (Burggräf <i>et al.</i> , 2020).
Order lead time	Time between the start of the first operation and the end of the last one (Burggräf <i>et al.</i> , 2020).
Operation lead time	Time of a single operation, the sum of interoperation time (wait, transport, wait) and operation time (setup, processing) (Burggräf <i>et al.</i> , 2020).
Finishing lead time	Object of this thesis. Total time a piece spends in painting, from being sent for processing to returning ready for installation.

Table 1.1: Hierarchy of the lead time definitions adopted in this thesis.

Upholstery, the other finishing operation, is left out of the analysis for the reasons discussed in Chapter 4.

1.1.3 The main objective

The main objective of the thesis is to attempt to build a tool that predicts the finishing lead time of the pieces of a commission, using machine learning models trained on the history of their movements. The motivation is operational before it is methodological. A project manager who knew, with reasonable approximation,

when a piece with a given finish would be available again could build a plan closer to reality and act before a delay propagated downstream.

It is an industrial case study, carried out during a five-month internship at a firm in the sector, on four commissions and 23 months of entries into painting, from October 2024 to August 2026. The models use the movements as the portal records them, and operators were not asked to change the way they record. This is what allows the thesis to measure how much a recording practice of this kind makes it possible to predict.

1.1.4 The operational objectives

From the main objective follow the operational objectives listed below.

1. Build a model that estimates, at the moment a piece is sent to painting, how many days will pass before it returns, using only the information available at that instant.
2. Identify which attributes of the piece and of its context explain the observed variability, among finish code, material family, element and functional family, destination deck and cabin, and period of dispatch. Knowing which of them matter serves process decisions as well as prediction.
3. Establish whether pieces should be grouped into homogeneous sets before predicting and, if so, whether the grouping should be defined a priori from domain knowledge or discovered by an unsupervised algorithm.
4. Turn predictions into decision lead time, making deviations from agreed dates visible at an early stage of the process, so that those responsible for production can act before the delay reaches installation (Burggräf *et al.*, 2020).

1.1.5 The research questions

Each operational objective translates into one research question, in the same order, which the following chapters answer with experimental evidence.

1. How accurately can the finishing lead time of a piece be predicted at the moment it is sent to painting, using only the information available at that instant, and how does that prediction compare with reference estimates that involve no learning?
2. What information determines the duration of finishing, and do the recorded data observe it at the level of the individual piece?

3. Does grouping the pieces before prediction, by domain criteria or by an unsupervised algorithm, improve on a single global model?
4. By how much does the estimate improve if it is updated during processing, once part of the batch the piece travels with has returned?

The transferability of the models across commissions, and in particular between new build and refit, is analysed in Chapter 5 without being the subject of a separate question.

1.1.6 The experimental design on segmentation

Grouping observations into homogeneous sets before training recurs in the studies reviewed in Chapter 3, in different forms. The grouping can be statistical, with a separate model for each group of jobs or operations with similar characteristics (Koblasa and Urban, 2026); it can cluster the categorical attributes of the units before training, as for ship blocks (Li *et al.*, 2024); or it can precede the estimate as a classification of the pieces into design categories (Masoumi and Bond, 2025). None of the three works reports a comparison, on the same data, between a grouping defined from domain knowledge and one found by an algorithm.

The thesis compares on the same data, and under the same evaluation protocol, the five configurations of Table 1.2, designed to separate the effect of grouping from that of its origin.

Comparing A with the other configurations measures the benefit of grouping as such. Setting B and C beside D and E shows whether domain knowledge yields better partitions than those found in the data, while the comparison within each

Config.	Description
A	Single global model, without segmentation. It is the reference against which every other configuration is measured.
B	Global model with the segment defined by the domain expert, the combination of functional family and material family, added as a feature.
C	Separate models for groups defined by domain criteria, in two variants: one per material family and one per commission.
D	Unsupervised clustering of the pieces, fitted on the training data alone at each step of the evaluation, with 2, 4 and 8 groups, followed by a separate model for each group.
E	Global model with cluster membership as a feature, either as a label or as the membership degrees of a fuzzy clustering.

Table 1.2: Experimental design for assessing the effect of segmentation on finishing lead time prediction.

pair indicates whether it is preferable to train separate models or to pass the group information as a feature to a single model.

1.2 The industrial context

The organisation hosting this work is a small and medium enterprise in a segment of Italian manufacturing that grew around the country's shipbuilding districts, the subcontractors specialised in outfitting the interiors of luxury yachts. Behind the large shipyards operates a network of such firms, supplying what the yards do not produce in-house. If the hull is the structure, interior architecture is what turns a floating platform into a residence at sea: furniture and cabinetry, panelling, ceiling systems, floors, doors, upholstered elements and bespoke furnishings, each component made to drawing and each delivery scheduled against the shipyard's installation window.

The size of the firm conditions the resources available for digitalisation, and any method introduced has to be sustainable with the existing skills, a condition the literature describes as typical of small and medium manufacturing enterprises (Bianchini *et al.*, 2024).

In this segment the unit of production is the yacht. Each commission, the contract for outfitting a specific vessel, comprises from a few thousand to tens of thousands of pieces, which have in common only their destination. Two commissions may share suppliers, finishes and even geometries, yet the link between a piece and its place on board holds only for the yacht it was made for. The delivery constraint also comes from outside. The shipyard sets the launch date, and every supplier in the chain has to meet it with a coordinated flow of components ready for installation, so neither design nor timing can be separated from the individual commission.

The result is the operating regime that the Engineer-to-Order literature has documented in shipbuilding, in plant construction and in other one-of-a-kind settings, characterised by high product variety, no repetition of units, long lead times and strong interdependence among suppliers (Wortmann, 1992; Hicks, McGovern and Earl, 2000; Gosling and Naim, 2009). The products of a yacht interior manufacturer consist of a large number of components designed individually to achieve a tailor-made solution for each customer (Burggräf *et al.*, 2020).

1.2.1 Supply chain management in Engineer-to-Order production

Manufacturing strategies can be ordered by the position of the customer order decoupling point, the stage of the flow at which a product becomes associated with a specific customer order (Amaro, Hendry and Kingsman, 1999). Engineer-to-Order sits at the far end of this spectrum. The order arrives before the product has been designed, and engineering, procurement, production and assembly all start from a unique specification (Wortmann, 1992; Amaro, Hendry and Kingsman, 1999). Luxury yacht interior outfitting is a typical case, since every commission is designed around a vessel, the bill of materials is defined progressively and no commission repeats the same set of pieces.

Under such a regime production control is exercised over a network of internal and external stages whose duration varies from piece to piece, because the object produced is new every time and the stable cycles of repetitive manufacturing are absent (Bertrand and Muntslag, 1993). The supply chain resembles a set of parallel flows that have to converge on a date chosen by others. In shipbuilding in particular, the decoupling point and the coordination it requires change with the degree of customisation and with the distinction between new build and refit (Semini *et al.*, 2014), the same two programme types that structure this case.

The firm operates in this regime by coordinating rough production, finishing and on-board installation across several sites, so that a flow of customised components reaches the vessel in the right sequence and by the launch date. Painting, one of the stages most sensitive for quality, is entrusted to a single specialised supplier, which works in its own departments inside the firm's plants and uses its external sites only when a piece is urgent.

In practice painting is therefore an internal operation, although the order in which pieces are processed is set by the department, which also works on pieces of other commissions. The lead time of an individual piece is the outcome of this network of decisions.

1.2.2 Product complexity in bespoke interiors

Product complexity shows in the hierarchical structure represented in Figure 1.1. The vessel divides into decks, each deck into cabins, each cabin into functional areas, each area into elements and each element into the pieces physically produced and installed. In the data the functional area of a piece is recorded as its functional family. A large commission unfolds this hierarchy into several thousand pieces across dozens of cabins and multiple decks.

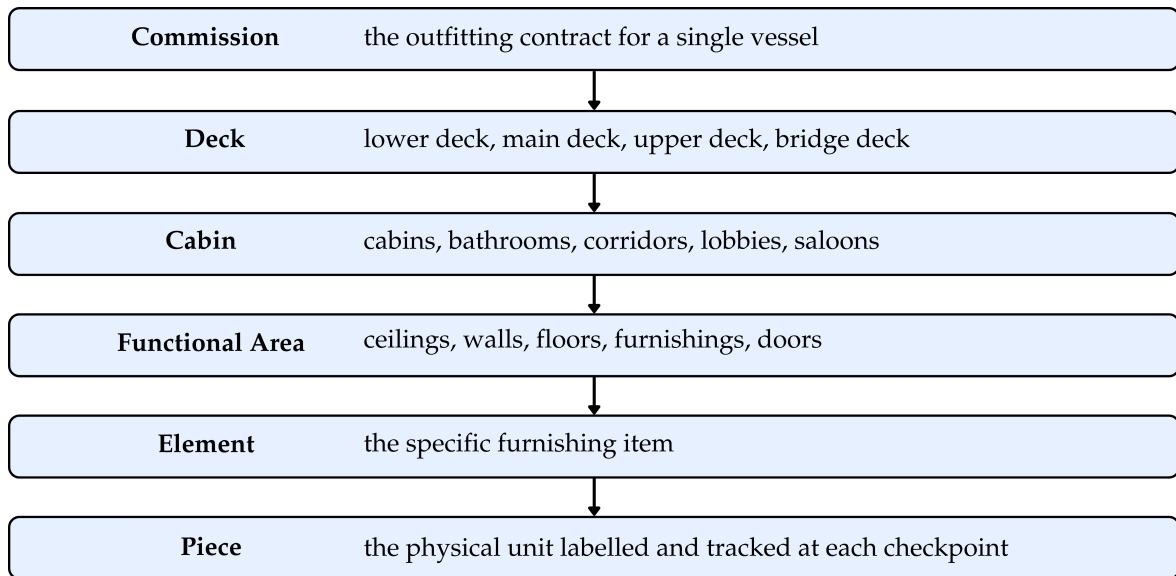


Figure 1.1: Product hierarchy, from the commission down to the individual piece.

Customisation is total, since every piece is made to a drawing that applies to one vessel only, with materials and finishes chosen by the client. The refinement of veneers, leathers, lacquers and stone is the value being sold, and it is also what the time needed to finish a piece is expected to depend on; the search for features starts here.

The workload alternates quiet periods with phases of saturation driven by the pre-season launch calendar. The firm also depends on parties it does not control. The owner requests changes that redefine the bill of materials while work is in progress, the shipyard sets the installation window and can move it, and the suppliers of electrical and mechanical components constrain what the outfitting has to accommodate. In such an interdependent environment a delay accumulates along the chain and surfaces at installation, where the margin to absorb it is smallest.

A last layer of complexity concerns information. In the portal data, element and piece names follow no fixed standard, different nomenclatures are recorded under the same element, and there is no notion of an assembly grouping a set of pieces, so that a single functional element ends up scattered across records that share only the cabin. Chapter 4 examines this fragmentation, the first obstacle for any data-driven method.

1.2.3 Sector positioning and commission portfolio

The firm operates in the segment as an outfitting specialist, for new builds and for refits alike. The two programme types share the same physical activities but place

	New build	Refit
Design	Known months in advance, stable bills of materials, drawings frozen at defined milestones	The vessel is already in the yard, the initial inventory is incomplete and the scope widens as surveys proceed
Drawings	Available before production starts	Arrive later, sometimes once production has begun
Typical risk	Missed sequencing: the piece is ready too late for its window	Overlooked dependencies: the piece is produced without information on the existing interior
Commercial rhythm	Concentrated before the cruising season	Concentrated out of season, when yachts return
Piece volume	High, in the order of thousands per commission	Lower, with more compressed priorities

Table 1.3: Comparison between new build and refit programmes.

different demands on the logistics flow and, as the data in Chapter 4 show, produce distinct lead time distributions. Table 1.3 summarises the comparison.

The two portfolios spread the workload across the year instead of competing for the same resources, with direct effects on the workforce and on pre-assembly capacity. For the predictive work, seasonality thus becomes a plausible feature, and the difference between the two regimes makes it necessary to verify how far a model trained on one programme type transfers to the other, an analysis reported in Chapter 5.

1.2.4 Geographical presence and site logic

Operationally the firm is distributed across four sites. The head office concentrates the design office, part of the production capacity and the main logistics function. Two further sites act as pre-assembly and staging hubs, where rough components arrive, sub-assemblies are made and shipments to the vessels are consolidated. The fourth site, in another region, is dedicated to refits and serves as a local logistics hub for programmes berthed at nearby yards.

The configuration follows the choice of keeping finishing close to the place of installation. During outfitting the vessels do not move, and bringing finishing close to the ship reduces the risk of transporting parts that are already finished. Painted panels and upholstered elements are easily damaged in transit, and a finish rejected on delivery costs weeks of rework in the final part of the programme, when there is no margin left to absorb them.

Geography bears directly on the variable the thesis sets out to predict. A piece may leave the head office, pass through the painting department of one of the plants,

or through an external site of the painter, return to a hub for sub-assembly and finally reach the vessel. Every transfer adds transport and waiting time, and the path followed may itself influence the finishing lead time.

1.2.5 The production process and the material flow

Activities at the firm fall into functional flows, each with its own physical presence and its own tracking events: production, logistics and on-board installation.

Production covers rough fabrication and finishing. Panels and cabinetry start as sheet material cut to the dimensions and shapes given in the design office drawings, and the rough piece goes through sanding, priming and painting in the departments the painter runs inside the plants, before returning for surface inspection. Upholstered pieces follow a similar path, with the rough structure assembled internally and the fabric or leather covering applied by the upholsterer. Bespoke furnishings, ceiling elements and doors follow variants of the same sequence.

Every transfer between sites and every shipment to and from suppliers is a tracking event. Logistics volumes are considerable, a single commission generating tens of thousands of records over its production, and the timing of each transfer depends on the availability of the receiving site. The painter cannot start before the shipment has arrived, nor return the piece before the drying cycle is complete.

On-board installation, finally, assembles the finished pieces on the vessel and is carried out by dedicated teams working at the shipyard. The installation window is shared with the other subcontractors working on the hull, the systems and the decks, each competing for the time available on board under the coordination of the shipyard's project office.

Figure 1.2 shows the overall path of a piece.

The flow is drawn as a line, and in practice it branches precisely at finishing. A piece can return for a second or third pass, a rough piece can be test-fitted on board and sent back for adjustment, a piece can be rerouted between sites when a drawing

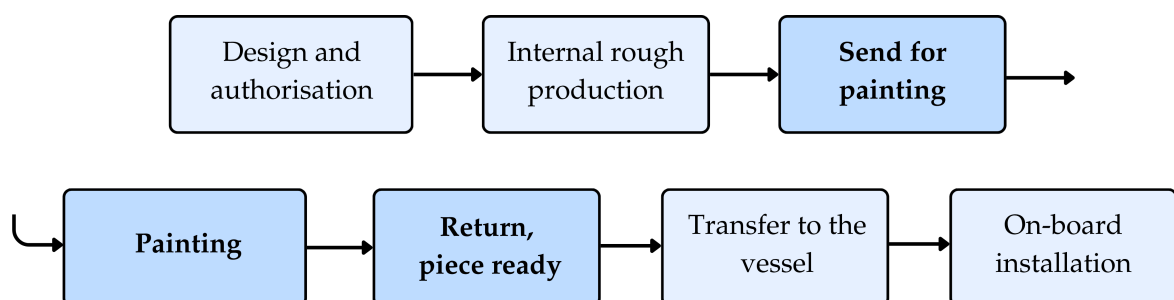


Figure 1.2: Material flow of a piece, from design to installation. The highlighted steps make up the finishing lead time.

changes. These returns belong to the craft iteration the sector sells. For prediction they mean that the number of movements a piece accumulates is neither fixed nor known in advance.

1.2.6 The role of the project manager

Coordination across sites rests on a few roles. The logistics function organises transfers and shipments to and from suppliers, a warehouse manager at each staging site handles physical inventory and outbound consignments, and an installation team leader receives the pieces on board and sequences their fitting. Above them all, the project manager holds sole responsibility for the commission, from design freeze to the installation of the last piece.

That responsibility spans classic project management activities, such as scope and schedule control and procurement, and operational tasks specific to the sector: agreeing with the painting supplier which pieces come first, deciding which pieces to consolidate into a shipment, reconciling the physical inventory of the hubs with the approved drawings, arbitrating between competing installation priorities.

Every week the project manager reconstructs the state of the commission from the portal data, transfers it into a tracking sheet and completes it with calls to the hubs and to the suppliers, and the decisions of the following days rest on that reconstruction. The work takes the better part of a day for each active commission.

The reconstruction shows where the pieces have been. When they will be ready again is left to the project manager's experience, and this is the quantity the models of the following chapters attempt to estimate on a statistical basis.

1.2.7 The design office and the drawings

Upstream of production, the design office issues the drawings from which each piece is made and maintains the bill of materials linking every piece to its destination on board. The drawings organise the interiors according to the same hierarchy as Figure 1.1.

Client change requests also enter the process through the drawings. A change of finish, a variation to a piece of cabinetry or an additional feature starts as a drawing revision and propagates through the bill of materials down to production. The frequency of these changes is one of the reasons the sector resists full industrialisation, since every commission carries a continuous backlog of open modifications and part of the project manager's daily work consists in deciding which can still be absorbed by the current plan and which have to be renegotiated with the client. For prediction,

a drawing revision arriving once production has started is one of the plausible causes of a longer finishing time.

1.2.8 A sector that tolerates craft work

Luxury interior outfitting is only partly industrialised, and this weighs on the distribution of the target variable. The finish quality the client expects is judged by eye, by touch and often in person at delivery, and no standard inspection is enough to certify it.

Industrial-scale planning therefore coexists with craft-scale iteration. A piece may complete the finishing sequence in a few weeks, or come back several times for corrections and take much longer overall, and neither trajectory is an anomaly. The distribution of the finishing lead time has a long right tail, described in Chapter 4, and the choice of the metric used to evaluate the models has to take it into account.

1.2.9 The digital tracking infrastructure

Every checkpoint a piece passes is recorded in the portal, the firm's reference system for the state of a commission. The portal keeps every piece, every label and every movement as a log of individual events, with no layer that aggregates them, and this log is the raw material of the models developed in the thesis.

A piece acquires its identity when production is authorised and it is assigned a label carrying a unique, optically readable code, which stays with it for its whole life cycle and is read at every checkpoint. The label encodes the commission code, the number and name of the cabin, the names of the element and of the piece, the finish code and an alphanumeric identifier. A single scan therefore tells which piece is being handled, which point of the vessel it belongs to and which finish it is to be produced with. Figure 1.3 shows the three label formats in use.

Movements follow a different logic. Pieces travel grouped on checkpoint documents, compiled by hand by the staff managing the warehouse and the movements from the pieces that arrive from rough production and have to go to finishing. The grouping often follows the element, but the same document frequently collects elements from the same area of a cabin, several ceiling elements, or elements from different cabins with a similar shape and number of pieces. Each document carries a general QR code, and scanning it records the movement of all the pieces it contains. In this thesis the set of pieces travelling on the same document is called a batch.

In the painting department, as could be observed directly, the pieces of a document enter together on a rack to which the document is attached. They are taken one at

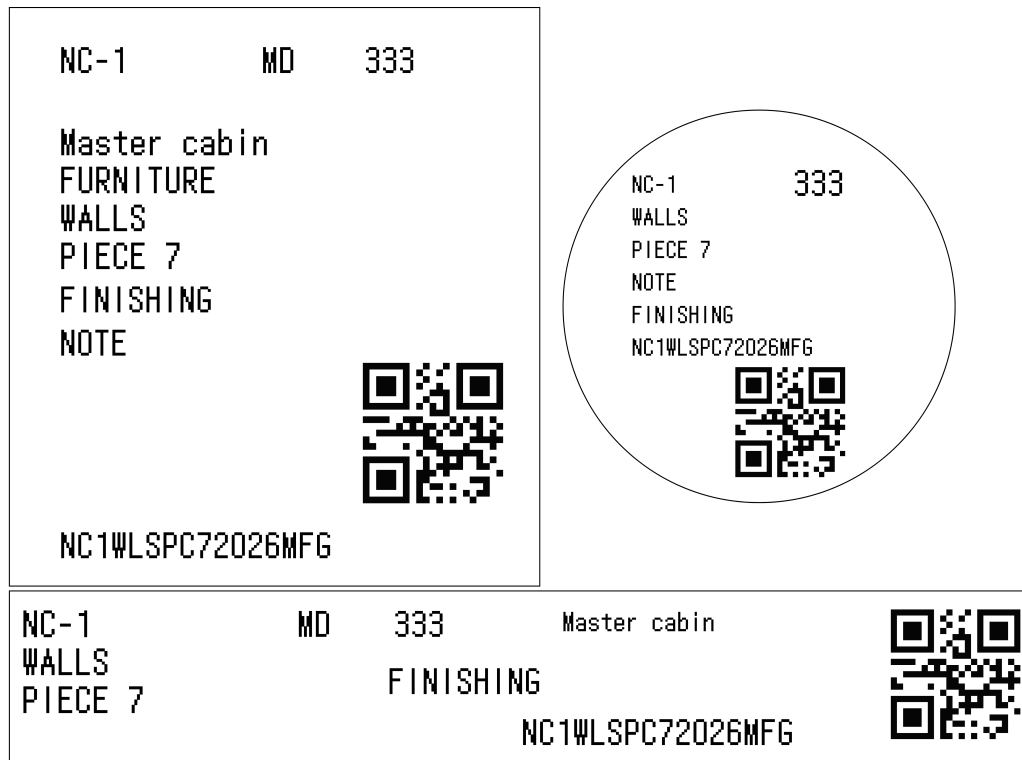


Figure 1.3: The three label formats in use, shown with fictitious data.

a time, processed each with its own finish and its own timing, and put back on the rack. When all or nearly all of them are finished, the rack leaves the department.

Each movement is described by three fields that have to be read together, listed in Table 1.4.

Field	Meaning
Site	The site where the piece is at that moment.
Department	The functional unit performing the action.
Phase	The action itself, whether a processing step or a transfer.

Table 1.4: The three fields describing a movement.

A piece recorded as received, for example, says nothing until site and department reveal whether it came back from finishing, arrived at the shipyard or entered the warehouse. The expressive unit of the portal is therefore the combination of the three fields. In the new build commission analysed, the history uses 49 distinct phase strings which, combined with sites and departments, give 155 triples, so the phase alone distinguishes fewer than a third of the situations the triple describes. Interpreting the triple is one of the main preprocessing problems addressed in Chapter 4.

1.2.10 The observed commissions and the data base

The commissions followed during the observation period form the empirical base of the work. The first is a new build programme for a motor yacht in the upper range of the segment, with several thousand pieces distributed over the whole vessel, across multiple decks and dozens of cabins. The second is the refit of a mid-size vessel returned for a complete interior renewal, with fewer pieces but with the added complexity of coexisting with the existing interiors and of an accelerated painting sequence.

Together the two commissions stress the process in complementary ways, the new build with sequencing at high volume, the refit with dependencies under compressed timing. Two commissions alone form a limited statistical base, however, especially for validating the models across different programmes, and the dataset is therefore extended with two further large new build commissions. The dataset thus covers four programmes, with a substantial increase both in the finishing cycles observed and in the coverage of finish and functional family combinations. Table 1.5 reports its composition.

Commission	Type	Role in modelling
NC-1	New build	Commission observed directly; main training base for volume and depth of history.
RF-1	Refit	Commission observed directly; test of transferability towards a regime with compressed timing.
NC-2	New build	Extension of the data base; increases the coverage of finish and functional family combinations.
NC-3	New build	Extension of the data base; enables validation across homogeneous commissions.

Table 1.5: Composition of the data base used for modelling.

The operating profile described in this section, with production distributed across several sites, checkpoint-based tracking, a mixed portfolio of new builds and refits, coordination led by the project manager and tolerance for craft work, is that of the Engineer-to-Order regime recalled at its start. For this reason the conclusions of the thesis on data recording may also concern other firms that track production in the same way.

Chapter 2

Machine Learning

2.1 Learning paradigms

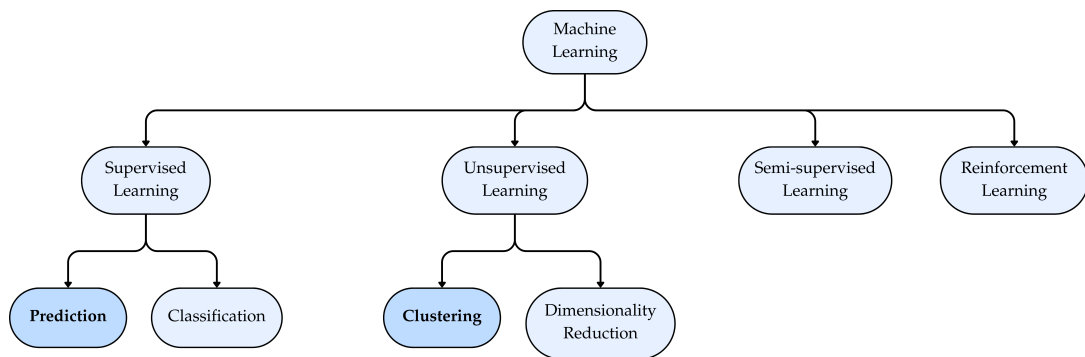


Figure 2.1: Taxonomy of machine learning paradigms.

Figure 2.1 arranges machine learning methods by the kind of supervision available to them (Bishop, 2006; Hastie, Tibshirani and Friedman, 2009). Machine learning is a sub-field of artificial intelligence that gathers methods for building a model directly from data. Instead of encoding the relationship between inputs and output as an explicit set of rules, an algorithm infers it from a set of examples (Alpaydin, 2020) and adjusts its parameters until it reproduces the regularity present in the data, including on cases it has never seen. This ability to extend beyond the observed cases is called generalisation, and it is what separates learning from memorisation.

The two branches on the right of the figure complete the taxonomy and play no part in this work. Semi-supervised learning uses labelled and unlabelled data together and is useful when labels are expensive while raw observations are abundant (Bishop, 2006); reinforcement learning concerns an agent that learns a policy of actions by interacting with an environment and collecting rewards (Goodfellow, Bengio and Courville, 2016). The problem addressed here has a label for every observation, the duration of the finishing cycle.

2.1.1 Supervised learning

In supervised learning each training example carries both the values of its explanatory variables, called *features*, and the correct value of the output, and the algorithm learns a function mapping the former to the latter by minimising an error measure computed on the known examples. The aim is for the estimated function to hold for cases it has not seen.

The type of output separates two tasks. When the output is a continuous quantity, such as a duration in days, the task is the prediction of a numerical value; when it is a category, it is classification (James *et al.*, 2013). A continuous quantity can always be reduced to classes and treated as a classification problem, at the cost of losing information, while the reverse is impossible, and the choice depends on the form in which the answer is needed. Planning the finishing stage needs a number of days.

2.1.2 Unsupervised learning

Unsupervised learning has no target variable. The algorithm observes only the features and looks for structure in them, either grouping similar observations, in clustering, or compressing the feature space with the smallest possible loss of information, in dimensionality reduction (Hastie, Tibshirani and Friedman, 2009). This thesis uses clustering only.

Clustering can precede a supervised task. The observations are divided into homogeneous groups, which either receive separate models or enter a single model through their label, used as a feature, on the hypothesis that the relationship between features and response changes from one group to another. This is the strategy that the experimental design of Chapter 1 puts to the test.

A widely used algorithm is k -means (Lloyd, 1982), which divides the observations into k groups by minimising the sum of squared distances from the centroids:

$$\min_{C_1, \dots, C_k} \sum_{m=1}^k \sum_{\mathbf{x}_i \in C_m} \|\mathbf{x}_i - \boldsymbol{\mu}_m\|^2 \quad (2.1)$$

where $\boldsymbol{\mu}_m$ is the centroid of the m -th group. The procedure alternates two steps until it stabilises: each observation is assigned to the nearest centroid, then the centroids are recomputed as the means of the resulting groups. The number k is fixed in advance, and because the result depends on the starting point the procedure is repeated from different initialisations.

In the fuzzy variant of k -means every observation belongs to all groups, with a degree between zero and one for each (Bezdek, 1981), and the membership degrees

can enter a supervised model as continuous features. How a partition is judged when there is no correct answer to compare it with is discussed in Section 2.5.4.

2.2 The machine learning process

Whatever the algorithm, a machine learning study follows a recurring sequence of steps, shown in Figure 2.2 and described in Table 2.1. It is preceded by a preliminary stage, often called step zero, in which the problem, its objective and the variable to be predicted are understood (Alpaydin, 2020). This step determines which algorithms are suitable, which performance measures make sense and how much effort to devote to tuning. An error made here cannot be recovered downstream, because a model that is accurate with respect to a badly posed objective remains useless.

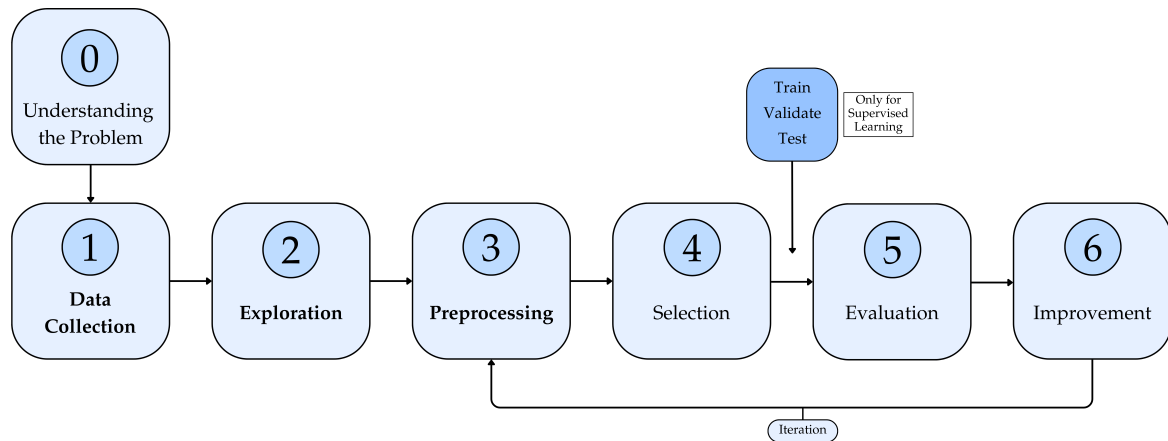


Figure 2.2: The machine learning process: step zero, the six subsequent steps and the training, validation and test phase, specific to the supervised case.

Comparing models is meaningful only in the supervised case, since only there does a correct answer exist against which the error of each can be measured. In the unsupervised case the validity indices measure how compact and separated a partition is, a property that no correct answer can confirm. The training, validation and test phase and the two branches of evaluation are developed in Section 2.4.4 and Section 2.5.

The sequence is travelled more than once, because the results of evaluation often send the analyst back to preprocessing and to the construction of the features. In this work, as often happens in applied studies, data collection and preprocessing absorbed the largest share of the effort, and Chapter 4 is devoted to them.

Step	Content
0. Understanding the problem	Objective, variable to be predicted, constraints of use.
1. Data collection	Sourcing the data and building a single, coherent dataset.
2. Exploration	Descriptive analysis and visualisation of what is available.
3. Preprocessing	Cleaning, treatment of missing values, construction of the features.
4. Model selection	Choice of the algorithm families to be compared.
Training, validation and test	Supervised case only, between selection and evaluation. Parameters estimated on the training data, hyperparameters chosen on a validation portion, final error estimated on test data never used before.
5. Evaluation, supervised case	Validation of the final model on the test data, with error measures and a reference that involves no learning.
5. Evaluation, unsupervised case	No correct answer to compare with: the partition is judged by internal validity indices.
6. Improvement	Comparison of performance between different models, in the supervised case only, and strategies to improve it: tuning, enrichment of the features, combination of models.

Table 2.1: The steps of the machine learning process, with evaluation distinguished between the supervised and the unsupervised case.

2.2.1 Preprocessing the features

Most algorithms work on numerical features, so categorical features have to be encoded. The simplest scheme creates one binary feature per level, equal to one when the observation takes that value and zero otherwise, and is known as one-hot encoding (James *et al.*, 2013). It works while the levels are few. With dozens or hundreds of levels it produces a very wide space and unstable estimates, and it is then better to replace each category with the mean of the target variable observed for it in the training data, shrunk towards the overall average when it occurs only a few times, a technique known as target encoding with smoothing (Micci-Barreca, 2001).

Numerical features measured on different scales are brought to a common one, by subtracting the mean and dividing by the standard deviation or by mapping them to the interval between zero and one. The step matters for methods based on distances or on gradients, such as nearest neighbours, support vector regression and neural networks, while trees are unaffected.

Encodings and scales are estimated on the training data alone and then applied, unchanged, to the validation and test sets. A mean computed on the test data as well would bring into the model information that does not exist at the moment of prediction.

2.3 Supervised models for prediction

The model families described in this section are those that recur most often in the literature on lead time prediction and that Chapter 5 compares. The selection follows the taxonomy of the reference systematic review (Burggräf *et al.*, 2020), which draws in turn on a widely cited empirical comparison of supervised algorithms (Caruana and Niculescu-Mizil, 2006), with the addition of gradient boosting, absent from that 2006 classification but widespread in the applied literature that followed. All models are presented in the form that produces a continuous output.

2.3.1 Linear and regularised models

Linear regression expresses the output as a weighted sum of the features plus an intercept:

$$y_i = \beta_0 + \sum_{j=1}^p \beta_j x_{ij} + \varepsilon_i \quad (2.2)$$

The coefficients are estimated by minimising the residual sum of squares,

$$\text{RSS}(\beta) = \sum_{i=1}^n \left(y_i - \beta_0 - \sum_{j=1}^p \beta_j x_{ij} \right)^2 \quad (2.3)$$

which has the closed-form solution $\hat{\beta} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}$ when the matrix $\mathbf{X}^T \mathbf{X}$ is invertible.

The strength of this family is transparency, since each coefficient quantifies the effect of one feature with the others held constant and its sign can be checked against domain knowledge. Its weakness is rigidity. The functional form is imposed a priori and interactions have to be specified by hand, one at a time, which is why linear models often serve as the term of comparison for measuring the gain of more flexible methods.

Regularisation controls the complexity of the model by adding to the loss function a term that penalises large coefficients. Ridge regression (Hoerl and Kennard, 1970) penalises the squared Euclidean norm of the coefficients,

$$\hat{\beta}^{\text{ridge}} = \arg \min_{\beta} \left\{ \text{RSS}(\beta) + \lambda \sum_{j=1}^p \beta_j^2 \right\} \quad (2.4)$$

whereas the lasso (Tibshirani, 1996) penalises their absolute value:

$$\hat{\beta}^{\text{lasso}} = \arg \min_{\beta} \left\{ \text{RSS}(\beta) + \lambda \sum_{j=1}^p |\beta_j| \right\} \quad (2.5)$$

Ridge shrinks the coefficients without cancelling them; the lasso can set them exactly to zero and so selects features. The elastic net (Zou and Hastie, 2005) combines the two penalties and is preferable when the features are strongly correlated, because in that case the lasso tends to pick one of them arbitrarily. In every variant the parameter λ sets the strength of the penalty. With $\lambda = 0$ the estimator returns to ordinary least squares, and as λ grows the model becomes progressively more rigid.

2.3.2 k -nearest neighbours

The k -nearest neighbours method builds no explicit model. It stores the training set and, to predict, finds the k most similar observations and returns their mean (Cover and Hart, 1967):

$$\hat{f}(\mathbf{x}_0) = \frac{1}{k} \sum_{i \in N_k(\mathbf{x}_0)} y_i \quad (2.6)$$

where $N_k(\mathbf{x}_0)$ is the set of the indices of the k nearest observations under a chosen metric, usually the Minkowski distance

$$d(\mathbf{a}, \mathbf{b}) = \left(\sum_{j=1}^p |a_j - b_j|^q \right)^{\frac{1}{q}} \quad (2.7)$$

which for $q = 2$ coincides with the Euclidean distance.

The hyperparameter k governs the bias-variance trade-off of Section 2.4.2 directly. Small values give estimates with little bias but high variance, large ones the opposite. The features have to be brought onto comparable scales (Section 2.2.1), otherwise those with the widest range dominate the distance, and the computational cost falls on prediction, since every new estimate requires the distances from all stored observations.

2.3.3 Support vector regression

Support vector regression extends to continuous outputs the maximum-margin idea developed for classification (Cortes and Vapnik, 1995; Drucker *et al.*, 1997). It looks for a function that departs from the observed values by no more than a tolerance ε , and penalises only the deviations beyond that threshold through the so-called ε -insensitive loss:

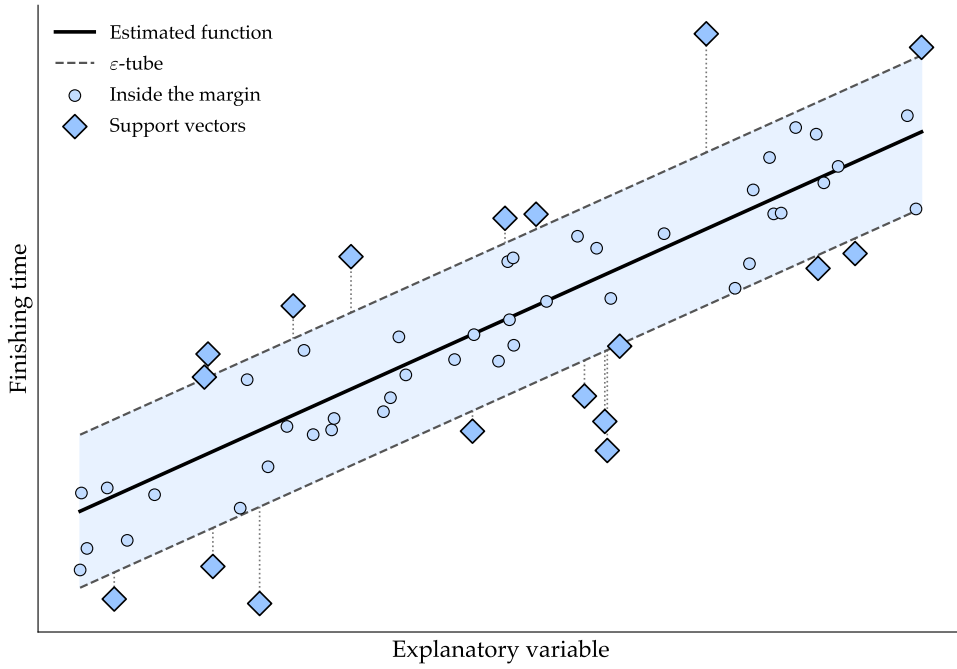


Figure 2.3: Support vector regression: insensitivity tube and support vectors.

$$L_{\varepsilon}(y, \hat{y}) = \begin{cases} 0 & \text{if } |y - \hat{y}| \leq \varepsilon \\ |y - \hat{y}| - \varepsilon & \text{otherwise} \end{cases} \quad (2.8)$$

In the linear formulation with slack variables ξ_i, ξ_i^* the optimisation problem becomes

$$\min_{w, b, \xi} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n (\xi_i + \xi_i^*) \quad (2.9)$$

subject to each observation falling within the tube of width ε up to the slack terms; the parameter C sets the trade-off between the flatness of the function and the tolerance of errors.

Flexibility comes from replacing the inner product with a kernel function, usually Gaussian, which represents non-linear relationships without constructing the transformed space explicitly. The method requires standardised features (Section 2.2.1) and careful tuning of ε , C and the kernel parameters.

2.3.4 Decision trees

A regression tree divides the feature space recursively into disjoint regions and assigns a constant value to each (Breiman *et al.*, 1984):

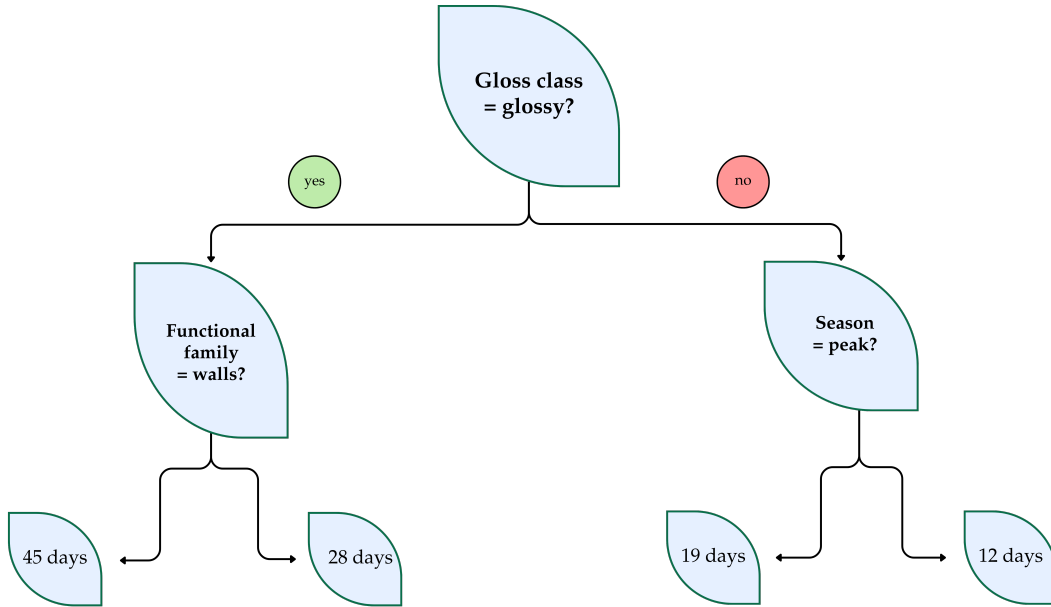
$$\hat{f}(x) = \sum_{m=1}^M c_m \cdot \mathbb{1}(x \in R_m) \quad (2.10)$$

where $R_1, \dots, R_M$ are the terminal regions and c_m is the mean of the observations falling in R_m . Construction proceeds by successive binary splits, choosing at each node the feature and the cut point that minimise the residual sum of squares in the two resulting branches:

$$\min_{j,s} \left[\min_{c_1} \sum_{x_i \in R_1(j,s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j,s)} (y_i - c_2)^2 \right] \quad (2.11)$$

Left to grow without constraint, a tree reaches leaves holding a single observation and reproduces the training set perfectly, the extreme case of the overfitting described in Section 2.4.1. Growth is therefore limited by stopping rules, such as a maximum depth or a minimum number of observations per leaf, or by pruning after the fact.

Trees handle categorical features natively, find interactions on their own and have a structure that can be read. They are unstable, however, because a small change in the data can alter the whole structure, and this fragility is what motivates ensemble methods.



Leaf values are purely illustrative.

Figure 2.4: Illustrative example of a regression tree.

2.3.5 Ensemble methods

Ensemble methods combine several models to obtain a prediction more stable and more accurate than that of any single component, and they differ in whether the components are built in parallel or in sequence.

Bagging, from bootstrap aggregation, trains B models on as many samples drawn with replacement from the original set and averages their predictions (Breiman, 1996):

$$\hat{f}_{\text{bag}}(\mathbf{x}) = \frac{1}{B} \sum_{b=1}^B \hat{f}^{(b)}(\mathbf{x}) \quad (2.12)$$

Since the mean of weakly correlated estimates has lower variance than each of them, the procedure reduces the instability of trees without increasing their bias.

Applied to trees, bagging has a limitation. If one feature is highly informative, every tree uses it for the first split and the trees end up resembling one another, which reduces the benefit of averaging. Random forests (Breiman, 2001) add a second source of randomness by restricting the choice of split at each node to a random subset of $m < p$ features; the trees become less correlated and the estimate improves. The method needs little tuning and copes well with a large number of features.

Boosting builds the models in sequence instead, each correcting the errors of the previous one (Freund and Schapire, 1997). In the gradient formulation (Friedman, 2001) the model is built additively,

$$\hat{f}_m(\mathbf{x}) = \hat{f}_{m-1}(\mathbf{x}) + \nu \cdot h_m(\mathbf{x}) \quad (2.13)$$

where h_m is a tree fitted to the negative gradient of the loss function at the current model and $\nu \in (0, 1]$ is a learning rate that slows convergence to improve generalisation. Modern implementations (Chen and Guestrin, 2016; Ke *et al.*, 2017) add explicit regularisation, native handling of missing values and computational refinements that have extended their use to large datasets. CatBoost (Prokhorenkova *et al.*, 2018) starts from a bias of the standard procedure: the gradients on which each new tree is fitted are computed by a model already trained on the same examples, so they are shifted with respect to those of new data, a form of target leakage. In its ordered mode the gradient of each example comes from a model trained only on the examples that precede it in a random permutation. Categorical features are encoded on the same principle, each category being replaced by a statistic of the target computed on the earlier rows that share it. Its trees are oblivious: every node of a level applies the same split. Bagging lowers variance and leaves bias largely unchanged; boosting progressively corrects the latter, at the price of greater sensitivity to noise and more delicate tuning of the number of stages.

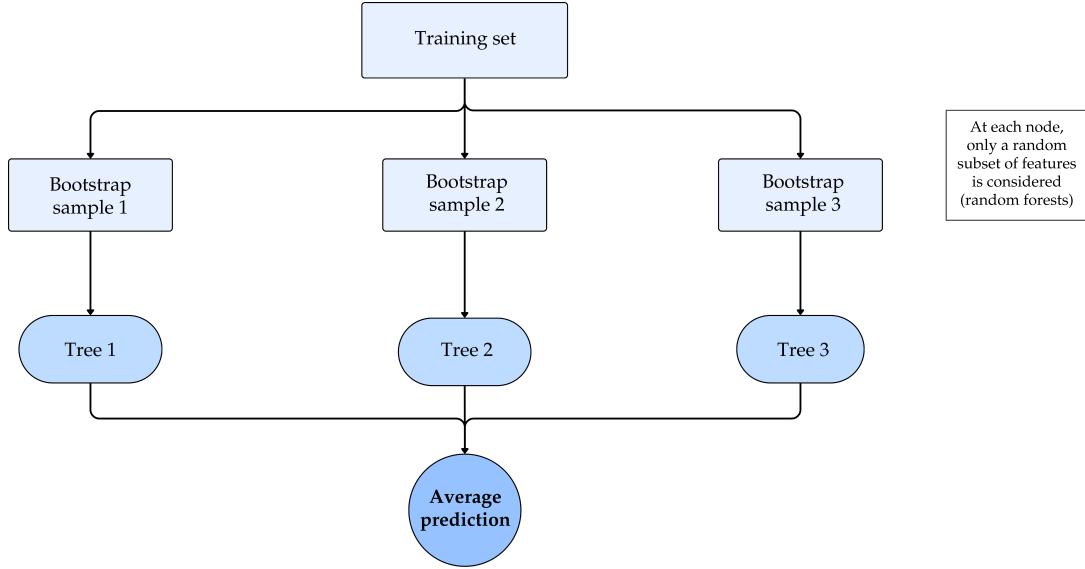


Figure 2.5: Bagging and random forests: parallel construction and averaging of predictions.

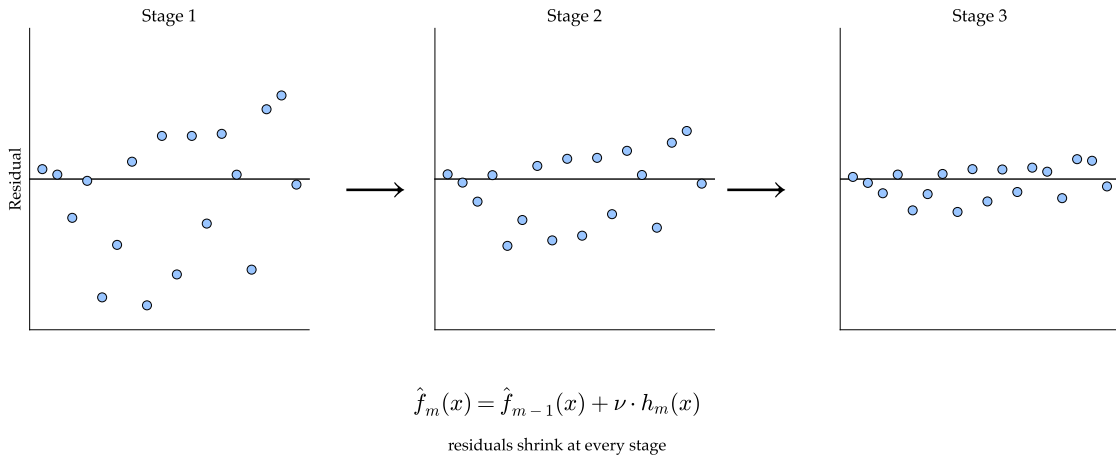


Figure 2.6: Gradient boosting: sequential construction on the residuals.

2.3.6 Artificial neural networks

A feedforward neural network composes affine transformations and non-linear activation functions organised in layers (Goodfellow, Bengio and Courville, 2016). With one hidden layer the prediction takes the form

$$\hat{y} = \mathbf{w}^{(2)T} \sigma(\mathbf{W}^{(1)} \mathbf{x} + \mathbf{b}^{(1)}) + b^{(2)} \quad (2.14)$$

where $\sigma(\cdot)$ is an activation function applied element by element, today usually the rectifier $\sigma(z) = \max(0, z)$, while for a continuous output the final layer is linear. The parameters are estimated by minimising a loss function through gradient descent,

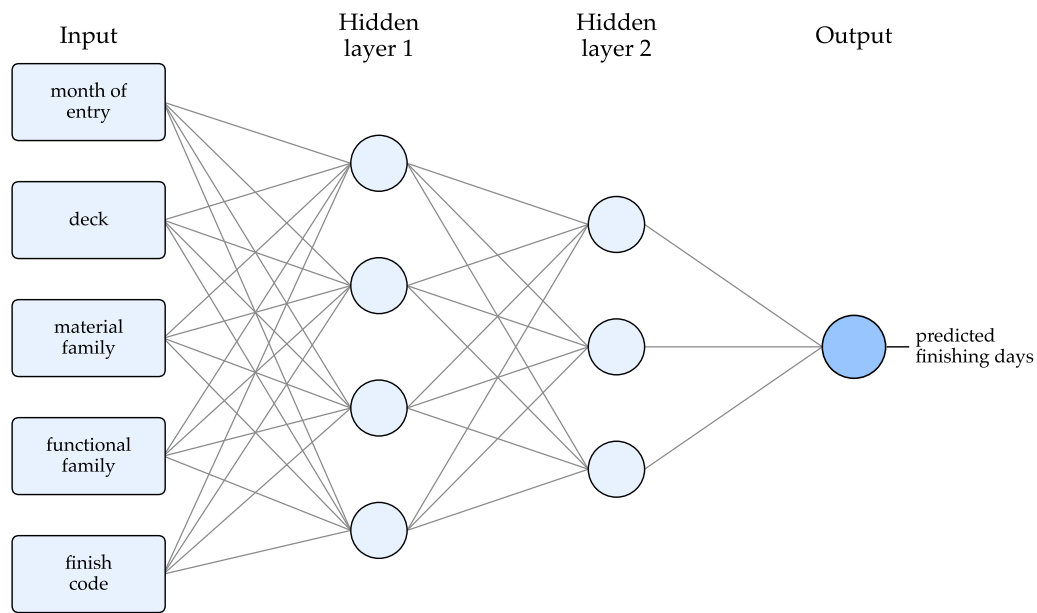


Figure 2.7: Illustrative example of a feedforward neural network.

with the gradients computed by backpropagation (Rumelhart, Hinton and Williams, 1986).

Neural networks represent complex non-linear relationships without requiring them to be specified, and they pay for it. They need more data than the other methods and more laborious tuning over a larger number of hyperparameters, and they return a model that is hard to interpret directly and needs the tools of Section 2.5.5.

2.4 Generalisation and data splitting

2.4.1 Underfitting and overfitting

A model can fail in two opposite ways. In underfitting it is too rigid to represent the structure present in the data, and the error stays high on both the training and the test set. In overfitting it is flexible enough to reproduce the random fluctuations of the sample as well, so the training error becomes very low while that on new data worsens (Hastie, Tibshirani and Friedman, 2009).

Overfitting is the more insidious of the two, because it presents itself as an apparent success. It becomes more likely with categorical features that have many levels, which offer many ways to isolate individual observations, and with correlated data,

such as repeated measurements on the same unit, from which the model can learn to recognise their source instead of the general relationship.

In methods trained through successive iterations, such as boosting and neural networks, overfitting appears during training itself. As iterations proceed, the error on the training data keeps falling, while on a validation portion it stops decreasing and then rises. Interrupting the fit once the validation error has not improved for a set number of iterations, called early stopping, is a form of complexity control (Goodfellow, Bengio and Courville, 2016).

2.4.2 The bias-variance trade-off

The phenomenon is formalised by decomposing the expected error. If the data are generated by $y = f(\mathbf{x}) + \varepsilon$ with $\mathbb{E}[\varepsilon] = 0$ and $\text{Var}(\varepsilon) = \sigma^2$, the expected squared error of an estimate $\hat{f}$ at a point $\mathbf{x}_0$ decomposes as follows (Hastie, Tibshirani and Friedman, 2009):

$$\begin{aligned} \mathbb{E}\left[\left(y - \hat{f}(\mathbf{x}_0)\right)^2\right] &= \underbrace{\left(\mathbb{E}[\hat{f}(\mathbf{x}_0)] - f(\mathbf{x}_0)\right)^2}_{\text{bias}^2} \\ &\quad + \underbrace{\mathbb{E}\left[\left(\hat{f}(\mathbf{x}_0) - \mathbb{E}[\hat{f}(\mathbf{x}_0)]\right)^2\right]}_{\text{variance}} + \underbrace{\sigma^2}_{\text{irreducible}} \end{aligned} \quad (2.15)$$

Bias measures how far the estimated function departs systematically from the true one, and it is high for rigid models. Variance measures how much the estimate would change if the model were trained on a different sample, and it is high for flexible models. The last term depends on the intrinsic noise of the phenomenon and no model can reduce it, so it fixes the lower bound of the attainable error.

As complexity varies the first two terms move in opposite directions (Figure 2.8). Greater flexibility lowers bias and raises variance, so the total error has a minimum at an intermediate complexity, which tuning has to find.

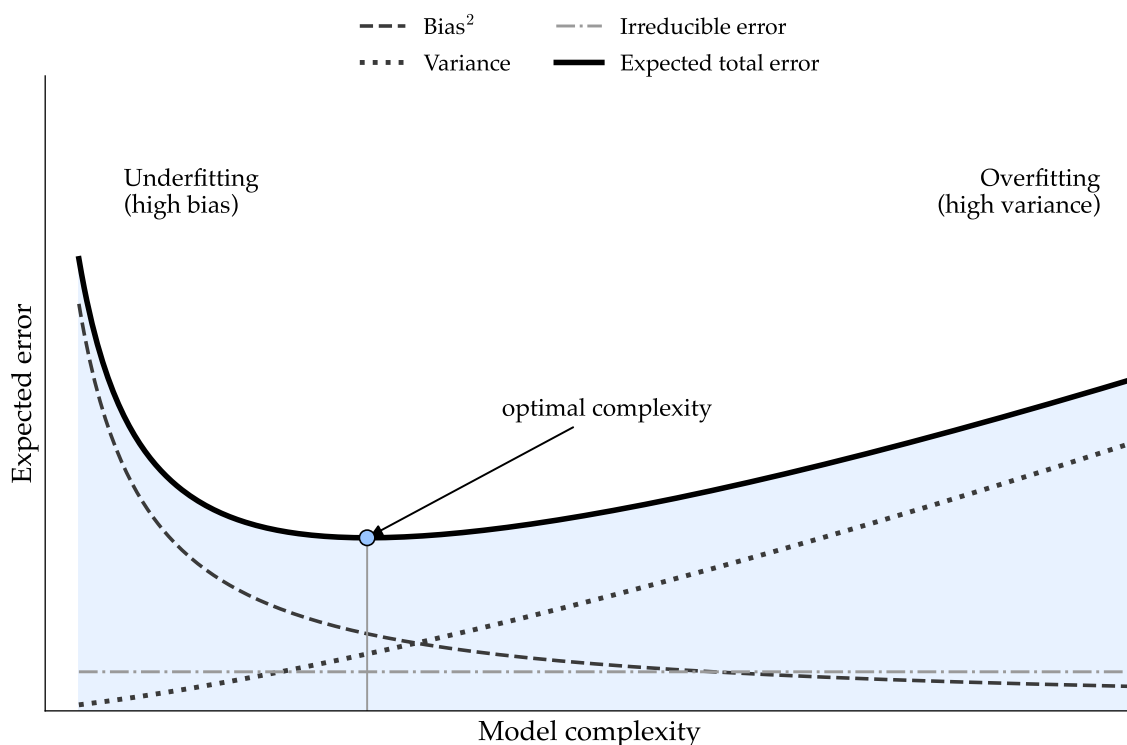


Figure 2.8: Trade-off between bias and variance as model complexity varies.

2.4.3 Parameters and hyperparameters

Parameters are the quantities the algorithm estimates from the data during training: the coefficients of a linear model, the weights of a neural network, the values assigned to the leaves of a tree. They are determined by minimising the loss function, and nobody chooses them.

Hyperparameters are fixed before training and govern it, like the coefficient λ of a penalty, the maximum depth of a tree or the number of neighbours. Choosing them by the error on the training data itself would systematically reward the most flexible model, the one that overfits, so observations distinct from those used to estimate the parameters are needed. This requirement is the reason for the second of the three portions in the training, validation and test phase (Table 2.1).

The choice proceeds on two levels. The model is trained for every plausible combination of hyperparameters and its validation error is measured, and the best setting is kept at the end. Grid search tries every combination of a fixed set of values for each hyperparameter, at a cost that grows multiplicatively with their number. Sequential searches instead choose each new combination on the basis of the results of those already tried, and concentrate the trials in the most promising regions (Bergstra *et*

al., 2011). When the data allow it, validation itself is carried out by cross-validation within the training set, and the test set is still used only once.

2.4.4 Data splitting and cross-validation

Evaluating a model on the same data used to train it gives an optimistic and useless estimate. Standard practice therefore divides the data into a training set, a validation set for the choice of hyperparameters and a final test set, used only once (Hastie, Tibshirani and Friedman, 2009).

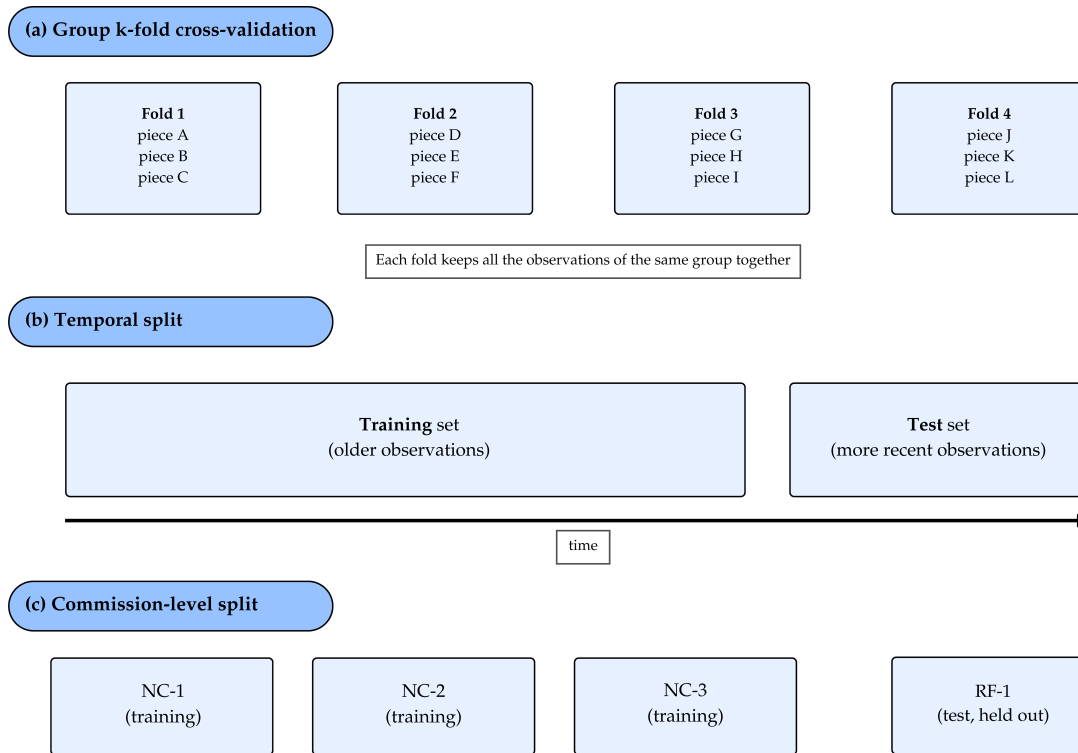


Figure 2.9: Illustrative example of three data splitting protocols.

When data are scarce, k -fold cross-validation uses them more efficiently. The set is divided into k parts, the model is trained k times leaving out a different part each time, and the estimated error is the mean of the k errors (Kohavi, 1995):

$$CV(k) = \frac{1}{k} \sum_{m=1}^k L(y^{(m)}, \hat{f}^{-m}(x^{(m)})) \quad (2.16)$$

where $\hat{f}^{-m}$ is the model trained without the m -th fold.

Random splitting assumes that the observations are independent of one another. When they are not, the estimate turns out optimistic and variants that respect the structure of the data are needed, illustrated in Figure 2.9.

1. When several observations belong to the same unit, grouped splitting keeps all of them in the same fold, so that the model cannot learn the identity of the group instead of the general relationship.
2. When the data have a time dimension, temporal splitting trains on earlier observations and tests on later ones, reproducing the conditions of use in which the future is predicted from the past. Repeating the cut forward, one period at a time, gives rolling-origin evaluation (Tashman, 2000). A random split would instead let the model use future information.
3. When transferability to an unobserved context is of interest, the model is trained on some groups and one group at a time is held out for testing, a stricter criterion than ordinary cross-validation.

2.5 Model evaluation

2.5.1 Loss function and evaluation metric

The loss function is the quantity the algorithm minimises during training, and it needs properties suited to optimisation, usually differentiability and convexity. The evaluation metric is the quantity by which the model is judged afterwards. It need not be differentiable and is chosen according to what matters to whoever uses the predictions. The two need not coincide. A model trained by minimising the squared error, for example, can be evaluated on the absolute error, if the latter describes the practical cost of a mistake better.

The loss function also determines which quantity the model estimates. Given the features, the squared loss leads to the mean of the response, the absolute loss to its median and the quantile loss to the desired quantile (Koenker and Bassett, 1978). For a level τ the quantile loss weights underestimates by τ and overestimates by $1 - \tau$, and it is minimised by the τ -quantile of the conditional distribution. A high quantile gives a value that the response exceeds only with low probability, which is the quantity needed when a date has to be promised (Section 2.6).

2.5.2 Error measures

For a prediction task the measures in use compare the observed values y_i with the predicted ones $\hat{y}_i$ on data not used for training. The mean absolute error measures the typical deviation in the units of the variable,

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i| \quad (2.17)$$

while the root mean squared error weights large deviations more heavily:

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2} \quad (2.18)$$

The mean absolute percentage error expresses the deviation in relative terms and makes groups with different typical durations comparable,

$$\text{MAPE} = \frac{100}{n} \sum_{i=1}^n \left| \frac{y_i - \hat{y}_i}{y_i} \right| \quad (2.19)$$

and the coefficient of determination gives the share of variability explained:

$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2} \quad (2.20)$$

The choice of measure has to be justified by the distribution of the response. On skewed distributions the squared error gives heavy weight to a few observations in the tail, whereas the absolute error describes typical performance better, and the two can rank models differently. The percentage error has to be used with care when the response takes small values, because a tiny denominator inflates it, and it is undefined at zero.

2.5.3 The baseline to measure against

An error measure on its own does not say whether a model is useful. A mean absolute error of 10 days can be an excellent result or an unacceptable one, depending on the variability of the phenomenon and on what could be obtained without a model.

A baseline is therefore needed, an elementary prediction rule against which the gain is measured. The most common choices are the mean or the median of the target variable, possibly computed by subgroup, or the rule already in use by whoever would adopt the model, which shows the gain over current practice. When evaluation is repeated over several folds, model and baseline are compared fold by fold, and Chapter 5 describes how the significance of the difference is judged.

2.5.4 Evaluating an unsupervised model

Clustering has no correct answer to compare with, and the quality of a partition is judged by internal validity indices. Among the most widely used is the silhouette (Rousseeuw, 1987), which for each observation compares the mean distance from the observations of its own group with that from the nearest other one:

$$s_i = \frac{b_i - a_i}{\max(a_i, b_i)} \quad (2.21)$$

where a_i is the mean distance within the observation's own group and b_i that from the nearest other one. The index ranges between -1 and 1 . Values close to 1 indicate a clean assignment and negative ones a likely misassignment, and the mean over all observations allows partitions with different k to be compared and the value that separates the groups best to be chosen.

Internal indices measure how compact and separated the groups are, a geometric property of the partition. When clustering serves a predictive task, this property need not coincide with usefulness. A geometrically good partition may fail to improve prediction, the reverse can also happen, and the usefulness of a grouping for prediction can only be measured by feeding it into a supervised model and evaluating the error.

2.5.5 Interpretability

A model meant to support a human decision has to be able to expose the reasons for its predictions. Tree-based methods provide feature importance measures that quantify how much each feature contributes to reducing the overall error, and that describe the average behaviour of the model.

Permutation importance measures how much the error on the test data grows when the values of one feature are shuffled at random across observations, which breaks its link with the response (Breiman, 2001). It needs no retraining and applies to any model family. A more direct and more expensive approach removes a feature, or a group of features, retrains the model and compares its error with that of the full version, and the difference measures what the removed features add to everything else.

SHAP (SHapley Additive exPlanations) attributes to each feature its own contribution to an individual prediction, through an additive decomposition based on the Shapley values of cooperative game theory (Lundberg and Lee, 2017). The prediction for an observation is written as the average prediction of the model, φ_0 , plus the sum of the contributions of the p features,

$$\hat{f}(\mathbf{x}) = \varphi_0 + \sum_{j=1}^p \varphi_j \quad (2.22)$$

and the contribution of feature j is the weighted mean, over all subsets S of the other features, of how much the prediction changes when j is added to S :

$$\varphi_j = \sum_{S \subseteq F \setminus \{j\}} \frac{|S|! (p - |S| - 1)!}{p!} [f_{S \cup \{j\}}(\mathbf{x}_{S \cup \{j\}}) - f_S(\mathbf{x}_S)] \quad (2.23)$$

where F is the set of features. The result makes it possible to check that the factors the model relies on are consistent with domain knowledge, and to accompany each estimate with its justification.

2.6 From prediction to decision

A prediction becomes useful when it guides a choice, such as which job to start first or which date to communicate. Such choices are the object of operations research, the discipline that applies mathematical modelling and optimisation to decisions on the use of limited resources (Hillier and Lieberman, 2015).

2.6.1 Two routes to a lead time

The reference systematic review distinguishes two routes to a lead time (Burggräf *et al.*, 2020). The indirect route builds a production schedule and derives the duration as a consequence of the sequence and of the expected waiting times, with tools such as queueing theory, characteristic curves and scheduling methods, and it requires knowledge of capacities, resource availability and batching constraints. The direct route predicts the overall duration from historical data and treats the production system as a black box, requiring a sufficiently long history of observed durations and no explicit knowledge of its internal parameters. The choice between the two depends on which information can be observed. The indirect route explains more and allows alternative scenarios to be simulated, but its results are only as good as the parameters that feed it, and when these are assumed rather than measured the model is formally coherent and impossible to verify. This thesis follows the direct route, for the data availability reasons discussed in Chapter 4.

2.6.2 Sequencing and date quotation

When several jobs wait to be started on the same resource, the release order determines the overall throughput time, and on a single machine sequencing them by increasing duration minimises the mean completion time (Graham *et al.*, 1979; Pinedo, 2016). The rule assumes durations known in advance, however, and its value depends on the prediction that feeds it.

The second problem concerns the date to be promised. Communicating a date that is too early exposes the promise to the risk of delay, while one that is too cautious ties up resources and weakens the credibility of the plan. Denoting by c_r the unit cost of lateness and by c_a that of earliness, the optimal date corresponds to the quantile of the predicted distribution at level

$$\tau^* = \frac{c_r}{c_r + c_a} \quad (2.24)$$

(Hillier and Lieberman, 2015), which the quantile loss described in Section 2.5 estimates directly. When lateness costs more than earliness, the optimal quantile lies above the median and the communicated date includes an explicit safety margin.

Chapter 3

Literature Review

3.1 The corpus

The reference systematic review on lead time prediction in Engineer-to-Order settings is the one by Burggräf *et al.* (2020), which covers only the direct route described in Section 2.6. Starting from 18,697 publications, the authors keep 42 after successive filters and a forward and backward search of citations, and only three of them predate 2000.

Koblasa and Urban (2026) widen the scope to production times of every kind, beyond Engineer-to-Order, in a scoping review of 173 works published between 2000 and 2024 and classified by type of predicted time, method, manufacturing system and reported accuracy.

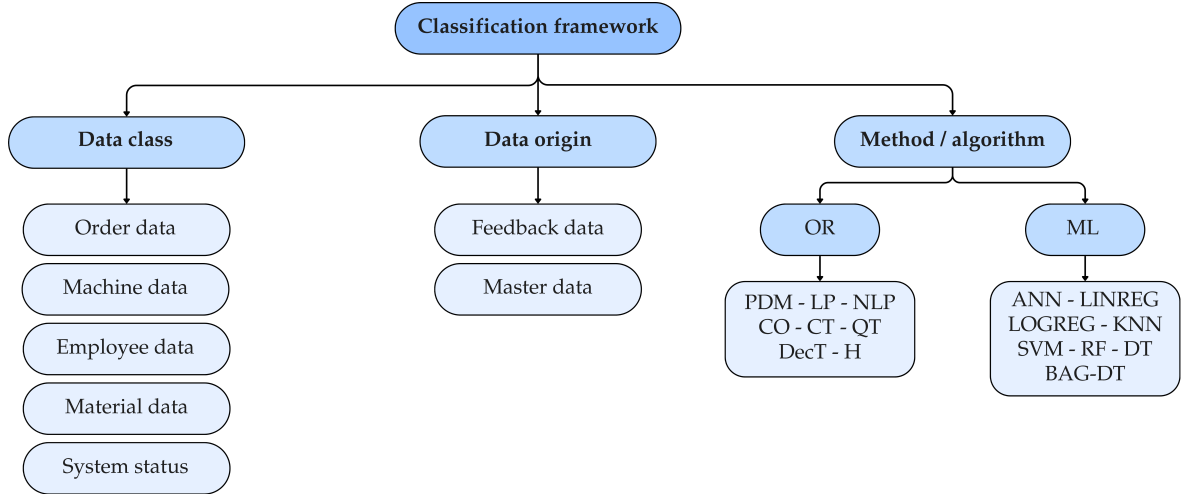
Since the systematic review stops in 2020, the picture it gives is extended here with the applied studies of Table 3.1, almost all of them more recent. Bender and Ovtcharova (2021) train their models on data generated by a simulation and Li and Lafond (2023) on a synthetic dataset, so their results do not concern durations recorded in production. The study by Talonpoika (2026) is a master's thesis, not peer-reviewed, and is included as an example of recent practice.

Study	Sector	Data	Predicted quantity
Lingitz et al. (2018)	Semiconductors	MES records: order, system status	Lead time of process steps
Burggräf et al. (2021)	Machinery and plant	Order, system status	Assembly start delay and lead times
Welsing et al. (2021)	Heavy engines	Event log, bill of materials	Lead time of the process and its sub-processes
Bender and Ovtcharova (2021)	Make-to-order	Simulated orders and factory state	Lead time
Bender, Trat and Ovtcharova (2022)	Make-to-order, two firms	ERP: order, load indicators	Duration of single steps
Steinberg et al. (2022)	Machinery and plant	Order, system status, material	Assembly start delay
Li and Lafond (2023)	Ship refit, deck painting	Synthetic, with expert estimates	Duration class
Li et al. (2024)	Hull blocks	Categorical block attributes	Construction time
Masoumi and Bond (2025)	Furniture	Part images, dimensions, time study	Processing time
Canedo Rosa, Gonçalves and Barbalho (2025)	Prototypes and Engineer-to-Order products	Geometry, material	Manufacturing time
Flores-Huamán et al. (2025)	Wind towers	Order, status, upstream prediction	Lead time of two operations
Lee et al. (2026)	Ship piping	Event log, static attributes	Procurement lead time
Talonpoika (2026)	Electric motors	ERP: production orders	Production lead time

Table 3.1: Applied studies reviewed, by sector, data used and predicted quantity.

3.2 The classification framework

Burggräf *et al.* (2020) organise the literature along the three dimensions of Figure 3.1: which data feed the model, where they come from, and which method processes them.



OR, operations research; ML, machine learning; PDM, precedence diagram method; LP, linear programming; NLP, nonlinear programming; CO, combinatorial optimisation; CT, control theory; QT, queueing theory; DecT, decision theory; H, heuristics; ANN, artificial neural networks; LINREG, linear regression; LOGREG, logistic regression; KNN, k -nearest neighbours; SVM, support vector machines; RF, random forest; DT, decision trees; BAG-DT, bagged decision trees.

Figure 3.1: The three dimensions of the classification framework, adapted from Burggräf et al. (2020).

3.2.1 Data classes

Order data comprise dates, quantities and operation-specific times. *Machine data* describe the resources that carry out the operations, from the identifier to fault messages, and *employee data* concern the operators. *Material data* gather the characteristics of the product to be manufactured, such as geometric specifications, weight and the material itself, while the *system status* covers information such as the level of intermediate stock or the capacity utilisation of the machines (Burggräf et al., 2020).

The use of the five classes is heavily unbalanced (Figure 3.2). Order data appear in 95% of the publications and the system status in 62%; machine data stop at 21% and material data at 5%, and employee data are never used (Burggräf et al., 2020).

The authors describe material data as an almost unexplored area, a *white spot*, and attribute their absence to the systems that hold them, the CAD system, drawings or the material master records of the ERP, which are often not linked to the order data. For employee data they point to privacy restrictions (Burggräf et al., 2020). They discard the hypothesis that material data are simply unrelated to lead time, since the few works that use them obtain good results, and in an Engineer-to-Order setting, where each product is designed individually and materials vary widely from one component to the next, they see a high potential in them.

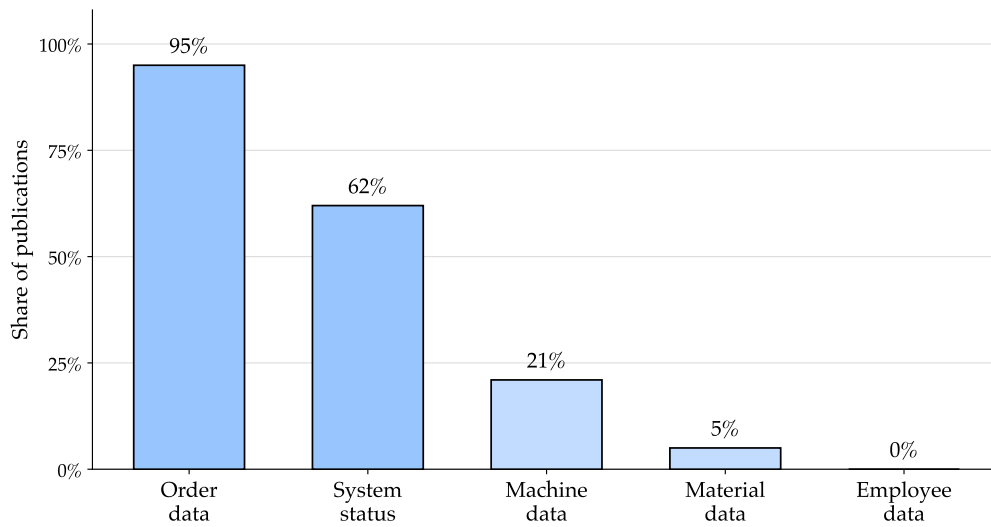


Figure 3.2: Share of publications using each data class. Data from Burggräf et al. (2020).

Steinberg *et al.* (2022) put the hypothesis to the test at a machinery and plant manufacturer with 12 models, six algorithms trained with and without material data (here gross weight and component name), and find a positive effect on model quality. Canedo Rosa, Gonçalves and Barbalho (2025) predict the manufacturing time of prototypes and Engineer-to-Order products at an aerospace and medical equipment company from the geometry and material of the parts, and report for the best models an accuracy above 87% and a mean absolute error below one day.

3.2.2 Data origin

The second dimension separates *feedback data*, recorded in a real production environment during the process, from *master data*, used for planning without feedback from the shop floor; the authors count among the latter the data generated by simulation and those whose origin the publication does not describe. The two origins split the publications evenly (Burggräf *et al.*, 2020).

Crossed with the number of data classes, the proportion shifts. When a model uses a single class, almost 70% of the publications rely on feedback data, and with three classes the share falls to 33%. Reading the number of classes as an indicator of model complexity, the authors conclude that data observed in production are used less as models grow more complex. They name the combination of many classes with feedback data as the area of greatest potential for further research (Burggräf *et al.*, 2020).

3.2.3 Methods

The third dimension separates operations research from machine learning. Machine learning accounts for 55% of the publications and operations research for 45%, but 18 of the 23 machine learning works appeared after 2010, while the number of operations research studies stays almost constant (Burggräf *et al.*, 2020). Among machine learning methods, artificial neural networks appear in 43% of the works, linear and logistic regression in 30%, decision trees in 26% and random forests in 22%, whereas support vector machines remain marginal.

Gradient boosting enters the field after 2020. In the multi-level model of Burggräf *et al.* (2021) a gradient boosting classifier is the best of 24 models, and XGBoost and LightGBM appear in the comparisons of Flores-Huamán *et al.* (2025) and Talonpoika (2026). When a study compares several model families, tree ensembles almost always reach the lowest error: random forests in Lingitz *et al.* (2018), random forests and bagging in Welsing *et al.* (2021), tree-based pipelines in Bender, Trat and Ovtcharova (2022), tuned XGBoost in Talonpoika. The exception is the furniture study of Masoumi and Bond (2025), where the neural network has the lowest error. The model families are described in Section 2.3.

Crossing method and data origin reveals a specialisation. In the works based on feedback data machine learning is used in 63% of cases, mostly with neural networks and regression models, whereas operations research prevails in those relying on master data, with 55% (Burggräf *et al.*, 2020). The authors attribute the pattern to the volume of feedback data, abundant and suited to learning, while producing master data is manual and costly work that suits operations research better.

In ship outfitting the indirect route of Section 2.6 is still in use. Liu *et al.* (2025) address production scheduling and material ordering for outfitting together, with a two-stage stochastic programming model that treats decision-dependent uncertainty as endogenous, and evaluate it on a shipyard case.

3.3 Data preparation and the recorded quantity

Lingitz *et al.* (2018) compare 11 methods on the manufacturing execution system records of a semiconductor manufacturer. Tree ensembles give the lowest errors, yet most of the work goes into the features: the authors derive 41 of them from the time stamps, measure their importance and build the final model on eight, the most important of which is the work in progress in the analysed steps.

Welsing *et al.* (2021) see the lack of defined processes and the poor quality of input data as a major obstacle to machine learning in real operations. In their case, the

assembly, testing and painting of heavy engines, the process model mined from the unfiltered event log is a tangle of paths from which it is unclear which lead time should be predicted. Only after filtering it, with heuristic methods and with the feedback of process engineers and shop floor staff, does the log yield defined start and end events on which models can be trained. A comparison with models trained without this step was not possible, because before it the data lacked consistent features.

Comparing three automated machine learning tools on the records of two firms, Bender, Trat and Ovtcharova (2022) find that they handle algorithm selection, training, hyperparameter optimisation and benchmarking well, and give little support to data understanding, preprocessing and feature construction, the activities that take most of the work. An earlier prototype built on one of these tools and trained on simulated data had been proposed by Bender and Ovtcharova (2021), so that small make-to-order firms could train models without in-house specialists.

The quality of the recorded duration comes up in two of the studies. In the data of Bender, Trat and Ovtcharova some steps that take minutes or hours are recorded as lasting several days, and manual operations such as sawing as almost instantaneous; the authors attribute these values to barcode scans not performed correctly and remove them as outliers (Bender, Trat and Ovtcharova, 2022). Flores-Huamán *et al.* (2025) describe a plant where operators have to leave their workstation to record that a part is complete, and therefore sometimes forget to do so or postpone it until the end of the shift, and they name the lead time as the variable most exposed to errors of manual recording. In both studies the recording error is a defect to be filtered or declared, and neither measures how far the way data are entered changes the quantity the model has to predict.

3.4 Predicting a duration or classifying a delay

Most studies predict the lead time as a continuous quantity. A minority estimates instead whether a component will be late, and Steinberg *et al.* (2022) identify through binary classification the components that will delay the start of assembly.

Burggräf *et al.* (2021) compare the two formulations on the same problem, with 24 models on four levels of detail, six per level. At the most aggregated level the delay is classified directly, whereas at the three finer levels it is derived from the predicted lead times of components, orders or operations. The best result comes from the most aggregated level with a gradient boosting classifier. Since delays are a minority of cases, both studies judge their classifiers by the Matthews correlation coefficient and

the F-score instead of accuracy, which would reward even a model unable to predict a delay.

Li and Lafond (2023) choose classification for another reason. They study the duration of deck painting in ship refit, where many tasks recur rarely and little history exists, and split it into three classes (short, average, long) because an expert finds it easier to give a category than a number. Their hybrid model, which combines expert estimates with 50 historical observations, raises classification accuracy by 5 to 10% over the history alone. Both the data and the experts' errors are generated artificially, so the result concerns how to combine the two sources, and how much the combination is worth on a real refit remains to be measured. It is the only study reviewed that concerns painting in the naval sector.

This thesis predicts a number of days, for the reason given in Section 2.1.1.

3.5 Segmenting before predicting

Koblasa and Urban (2026) devote a section of their review to grouping jobs before prediction, with k -means, fuzzy c -means or self-organising maps, followed either by separate models per cluster or by a single model that takes the group label as a feature, the two forms described in Section 2.1.2. They also document a variant, post-classification, which groups jobs by their prediction error and feeds the group back to the model as a feature. The results improve, but the authors observe that a group built in this way follows the error rather than the similarity between jobs, and can therefore act like overfitting.

Li *et al.* (2024) apply grouping to hull blocks. They cluster the categorical attributes of the blocks with distributed centroids. On 1,000 blocks of types with a long history, each group reaches a lower percentage error than the unclustered set, whose error on the test data is 9.5%; the errors of the groups and that of the whole sample, however, are computed on different subsets. The network trained on the data-rich types is then transferred to a type with only 80 samples, freezing part of its parameters and fine-tuning the rest, and the mean absolute percentage error falls from 14.97% without transfer to 8.62%.

In a furniture plant, Masoumi and Bond (2025) classify the parts before estimating their processing time. A convolutional network assigns the image of each part to a shape category (plain, curved in two or in three dimensions), and this label enters the models as a feature together with length and volume. The times come from a time study of 366 parts. All models exceed an R^2 of 0.90, and the neural network

reaches a mean absolute percentage error of 1.63% against 10.26% for the random forest and 11.15% for gradient boosting.

3.6 Formulations that use the process sequence

Production data are also a sequence of dated events, and some studies use those that have already occurred as information.

Lee *et al.* (2026) predict the procurement lead time of pipe spools at a large South Korean shipyard, from the release of raw material to the receipt of the finished spool, including transport, waiting, re-inspection and rework at subcontractors. Previous studies predicted it from static physical attributes such as diameter and weight, while records of the process begin to accumulate inside the shipyard much earlier. The authors extract from the event log the sequence of activities and their temporal attributes, process it with a sequential neural network and combine it with the static features processed by a multilayer perceptron. On three tasks, the production, post-processing and procurement lead times, they report a reduction in mean absolute error between 22.6% and 50.4% over the best approaches based on static attributes alone.

Flores-Huamán *et al.* (2025) use the sequence in a different way, in wind tower manufacturing. The predicted lead time of the upstream bending enters with its error among the features of the model for the downstream operation, longitudinal welding. Against the engineering estimates in use, the integrated welding model reduces the mean absolute error from 11.36 to 2.03 hours and the root mean squared error from 12.01 to 3.13, while the bending predictions remain of moderate quality. After reconstructing the process, Welsing *et al.* (2021) predict both the overall lead time and the times of the sub-processes between one event and the next.

The fourth research question of Chapter 1 applies the same idea to the single piece, updating the estimate once part of the batch it travels with has returned.

3.7 Validation, references and tests

Koblasa and Urban (2026) count the variety of validation procedures and metrics among the methodological gaps of the field, together with inconsistent definitions of production times. Table 3.2 reports, for the studies on recorded data, how the observations are split, how the hyperparameters are chosen, which metrics measure the error and against what.

Study	Split	Hyperparameters	Metrics	Reference
Lingitz et al. (2018)	10-fold cross-validation	partly, by trial	MAE, MAPE, MSE, RMSE, NRMSE	other algorithms
Burggräf et al. (2021)	80/20, by assembly order	grid search with cross-validation	MCC, F-score	other models and levels
Steinberg et al. (2022)	80/20	grid search with cross-validation	MCC, F-score, paired t -test	same models without material data
Bender, Trat and Ovtcharova (2022)	80/20, random	automated	MAE, RMSE, MSLE, R^2	manual estimates, training mean
Li et al. (2024)	70/30, random, 100 repetitions	particle swarm, transfer	MAPE, adjusted R^2	same model without clusters or transfer
Masoumi and Bond (2025)	80/20	not reported	R^2 , MSE, MAPE	other algorithms
Canedo Rosa, Gonçalves and Barbalho (2025)	10-fold cross-validation	configurations compared in cross-validation	accuracy, MAE, MSE, Friedman and Nemenyi tests	other algorithms
Flores-Huamán et al. (2025)	80/20, random, 5 inner folds	random search	MAE, RMSE	engineering estimates
Lee et al. (2026)	70/10/20	not reported	MAE, RMSE, MAPE	static-attribute models
Talonpoika (2026)	80/20, chronological, expanding folds	TPE, time-series cross-validation	MAE, RMSE, R^2 , MAPE	ERP planned times, training mean

Table 3.2: Data split, choice of hyperparameters, metrics and reference in the studies on recorded data.

Of the studies that describe how they split the data, only one splits them in time order. Talonpoika (2026) trains on the older orders, tests on the most recent 20% and tunes the hyperparameters with a cross-validation that respects the chronological order. The others divide the data at random, by shuffled folds or without stating the criterion, and Burggräf *et al.* (2021) are the only ones to keep all the components of an assembly order on the same side of the split. Temporal and grouped splits are described in Section 2.4.4.

The choice of hyperparameters (Section 2.4.3) ranges from grid search with cross-validation (Burggräf *et al.*, 2021; Steinberg *et al.*, 2022) to the random search of Flores-Huamán *et al.* (2025), which the authors prefer to grid search because it avoids exploring the whole space and to Bayesian optimisation because it costs less, and on to the tree-structured Parzen estimator used by Talonpoika and the automated search of the tools compared by Bender, Trat and Ovtcharova, which optimise the

mean absolute error because it is immediately readable. Flores-Huamán et al. also compare tuned models with others left at their default values.

The reference differs across studies, and with it the meaning of the reported gain. Flores-Huamán et al. compare their model with the engineering estimates in use, Li et al. with the same approach without transfer, Lee et al. with approaches based on static attributes alone. The studies of Talonpoika and of Bender, Trat and Ovtcharova are the only ones that also measure the models against a rule without learning, the mean of the training durations, besides the estimate currently used by the firm, and in both this average proves more accurate than the current practice. In Talonpoika the mean has a mean absolute error of 7.89 days against 8.12 for the planned times of the ERP. In the study of Bender, Trat and Ovtcharova all data-driven models, the mean included, outperform the manual predictions. For the second of the two firms the best automated model errs by 75.50 against 76.71 for the mean. Its predictions, however, fall in the same narrow range as those of that baseline, and the authors conclude that the tools failed to produce a better model.

Almost everywhere models are compared on the point values of the metrics. Steinberg *et al.* (2022) test with two paired *t*-tests whether the effect of material data on the Matthews coefficient and the F-score can be told apart from chance, and Canedo Rosa, Gonçalves and Barbalho (2025) apply the Friedman and Nemenyi procedures to their comparisons between algorithms; in the other studies of Table 3.2 the difference between two models is not tested.

The reported values also call for caution, because the predicted durations differ in unit and order of magnitude. A mean absolute error below one day on prototypes (Canedo Rosa, Gonçalves and Barbalho, 2025) and one of 2.03 hours on a welding operation (Flores-Huamán *et al.*, 2025) are not comparable quantities, and relative measures reduce the problem without removing it, since they remain sensitive to the typical duration of the task (Section 2.5). As an overall calibration, the scoping review places the error of machine learning lead time prediction at about 10%, that of activities with a strong human component at 20%, and that of machining operations estimated with deterministic methods at 6% (Koblasi and Urban, 2026).

3.8 Interpretability

Flores-Huamán *et al.* (2025) accompany their model with SHAP (Section 2.5.5) and identify sheet thickness, personnel experience and the quality of the upstream process, including the bending prediction built into it, as the determining factors. They also note that the short execution time of the model allows it to feed real-time scheduling adjustments. Talonpoika (2026) applies SHAP to the three tree ensembles

compared in the thesis, to identify the product characteristics that weigh most on the duration. Lingitz *et al.* (2018) note that an ensemble does not let the relation between features and output be read, and quantify the importance of the features by permutation and by removal: removing the work in progress from the final model increases the mean squared error by 5.3%. Welsing *et al.* (2021) choose a single cross-validated decision tree as their final model, although random forests and bagging perform best, because its logic can be followed by everyone and makes the predictions easier to accept.

3.9 Position of this work

None of the studies reviewed concerns the interior outfitting of yachts. The closest are those in the naval domain, on hull blocks (Li *et al.*, 2024), pipe spools (Lee *et al.*, 2026), outfitting scheduling (Liu *et al.*, 2025) and deck painting in refit (Li and Lafond, 2023), the last of them on synthetic data. Koblasa and Urban (2026) observe that applications outside highly automated sectors are few, particularly for tasks with a strong human component, and the craft finishing described in Section 1.2 is one such case.

This thesis uses the movements of individual pieces as the firm's portal records them, feedback data from a real production environment, together with material and finish attributes, a combination that Burggräf *et al.* (2020) place among the areas of greatest potential. It splits the data in time order and measures every model against rules without learning, testing the difference (Chapter 5), whereas among the studies reviewed the temporal split appears once and the test twice.

Where the studies describe recording errors, these remain noise to be filtered or a limit to be declared (Section 3.3). None of the studies reviewed checks whether the recorded durations actually observe the unit the model has to predict, and this is the second research question of the thesis.

Chapter 4

Data Processing and Feature Engineering

4.1 The exports and the fields used

The data come from two exports per commission, downloaded from the firm’s production management portal on 1 September 2026. The *piece register* has one row per piece and 37 columns, and describes the current state of each piece, as captured at the moment of extraction, with the attributes fixed at design stage: deck, cabin, element, functional family, finish code and finish description. The *movement history* has one row per event and 27 columns, and records every passage of a piece through a checkpoint with its date, phase, department, location and document. The register describes what a piece is, the history what happened to it. Duration is a property of the trajectory and is therefore built from the history, while the register supplies the attributes that may explain it, starting with the finish, which appears only there.

Commission	Type	Pieces	Movements
NC-1	New build	10,147	53,868
NC-2	New build	15,423	89,818
NC-3	New build	14,356	77,211
RF-1	Refit	1,781	8,871
Total		41,707	229,768

Table 4.1: Size of the two exports, by commission.

The observed entries into painting run from 1 October 2024 to 7 August 2026, and the exits up to the day of extraction.

Of the 64 columns, 20 are needed, and Table 4.2 lists them with their role. The others are empty, constant, duplicates of a coded field, or unusable although filled in.

Export	Field	Role
Piece register	piece_id	Join key
Piece register	commission_id	Commission
Piece register	deck, cabin, element	Attributes of the piece
Piece register	functional_family	Attribute of the piece
Piece register	piece_description	Filters on the pieces
Piece register	finish_code, finish_description	Finish and dictionary
Movements	piece_id, event_date	Construction of the cycle
Movements	phase, legacy_phase	Direction and meaning of the movement
Movements	department, location	Recognition of the phase and of the site
Movements	document_id	Checkpoint document
Movements	supplier	Recognition of the supplier
Movements	requested_date	Date requested by the department
Movements	rework_flag	Error analysis only

Table 4.2: Fields used from the two exports and their role.

4.2 The data in the framework of the literature

In the classification framework of Section 3.2, the exports contain feedback data, events recorded during production, and a single planned value, the requested date discussed in Section 4.11, which falls under master data. By class, they include order data, since every movement is a dated event referring to a work phase, and material data, through the finish code and description, the material family and the element. The system status, which the portal does not record as such, can be reconstructed from the movements by counting the pieces that have entered a department and not yet left it at a given date, and this is the route taken for the shop-state features of Section 4.12. Machine data are absent, and so are employee data, because the resource field identifies who performs the scan rather than whoever works on the piece.

4.2.1 Why the indirect route is not viable

The indirect route of Section 2.6 requires the capacity of the departments, the availability of resources and the rules by which queues form. Capacity and availability do not appear in the data, and the load can only be approximated after the fact from the movement dates. A scheduling model would therefore have to assume unmeasured parameters, and it would be internally consistent but impossible to verify.

4.3 The portal vocabulary and the change of regime

Until mid-June 2026 every movement is described by a string that names the origin or destination of the piece explicitly, with between 45 and 54 distinct strings in the three new builds and 25 in the refit. After the change the phase field takes only three values, marking a dispatch, a receipt or the final shipment, which give the direction of the movement without its object, and the meaning moves to the department and location fields. The previous phase column remains in the export, but on the rows after the change it carries a copy of the new value. The reading rule therefore depends on the regime. Where the old column still carries the extended string, it is read directly; elsewhere the movement is rebuilt from direction and department.

Commission	Movements	Later regime
NC-1	53,868	11.5%
NC-2	89,818	0.3%
NC-3	77,211	42.9%
RF-1	8,871	3.8%
Total	229,768	17.4%

Table 4.3: Share of the movements recorded in the regime after the change.

The switch is sharp and simultaneous in the four commissions, with no movement of May in the new regime, 60.2% of those of June and all of them from July.

4.4 The boundaries of finishing

Figure 4.1 maps the observed phases, grouped by state of the piece. Of the eight states, from the creation of the label to installation on board, only one corresponds to the quantity the thesis tries to estimate, the piece in processing while it receives its finish. The figure therefore also makes the interval to be measured easy to identify, from entry into finishing to exit from it.

4.5 Building the cycle

A *finishing cycle* opens when the piece is dispatched to painting and closes with its first return. In the earlier regime the opening is a phase that names both the dispatch and painting, and the closing a receipt from the same shop. After the change both are recognised from the direction of the movement and the department. In the refit the painting was carried out at the painter's external site, and the cycle runs from the dispatch to the supplier to the receipt from it. The finishing phases are identified by the substring common to all the strings that name painting, never by the site where they occur.

Figure 4.2 shows the pairing rules. The earliest entry opens the cycle and the first receipt from painting closes it. A further entry is ignored while a cycle is open, since the same dispatch is often scanned more than once, unless the piece has been recorded elsewhere in the meantime; in that case the pending opening is abandoned and a new one starts. In the refit this check does not apply, because there the piece is legitimately away for the whole cycle. A dispatch towards the exit department closes the cycle only if no receipt follows, and an entry that never receives a closing is abandoned.

Further corrections were added after inspecting the durations that the first rules produced. The acknowledgement of receipt, recorded in the new regime by the painting department itself, was read as an exit and produced zero-day cycles. The 430 rows involved fall at a median distance of zero days from the opening, and they no longer close. The pieces concerned return with their true cycle, and 362 are recovered. As a declared choice, a receipt from the painting entry department no longer counts as a closing in the earlier regime either; of the 54 rows it touches, 19 fall inside a cycle, at a median distance of 17 days from its opening, and applied on its own the rule removes three pieces and changes the duration of 16 others.

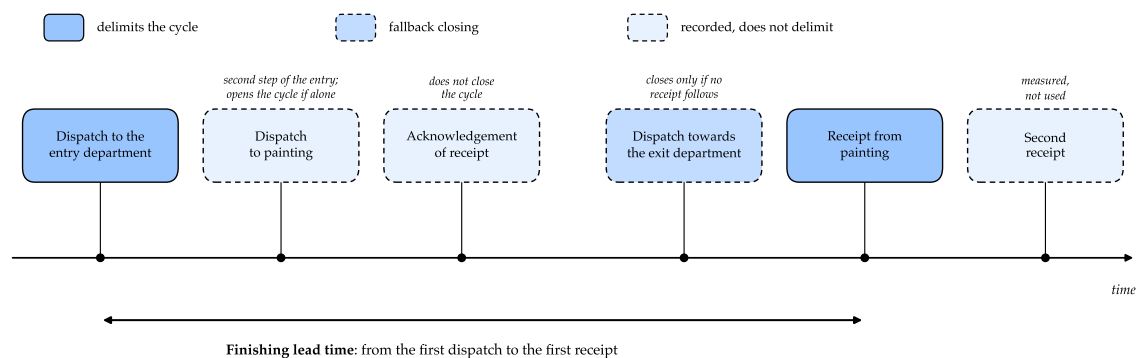


Figure 4.2: How a finishing cycle is read from the movements of a piece.

In the three new builds a dispatch to the external painter now opens a cycle and its return closes it; those rows were previously neutral, and 507 pieces enter the dataset. Finally, a dispatch to a supplier other than the painter abandons the open cycle, because the piece left painting before shipment and the date of its exit is unknown; five pieces leave the dataset. Together the corrections bring 869 pieces in, remove 8 and change the duration of 77.

Other variants were measured and not adopted. Turning the dispatch towards the exit department into a closing in its own right changes no cycle, because the fallback rule already covers it. In 919 cycles a second receipt follows the closing, with no dispatch in between and, in 84.5% of cases, no event at all. The criterion proposed to decide which of the two to keep did not separate the cases cleanly, and the first one is kept; Chapter 5 returns to this second receipt as a robustness check.

In 25.5% of cycles the piece enters in two steps, first sent to the painting entry department and then dispatched into the shop, a median of three days apart. The cycle starts at the first step, as a declared choice; starting at the second would lower the median of those cycles from 40 to 30 days. Leaving painting is almost always a single step, since 99.97% of cycles close with a receipt and 0.03% with the fallback rule, and only 0.95% have a dispatch towards the exit department before the closing, a median of two days earlier.

A piece can pass through finishing more than once, and 970 pieces do. The second pass begins a median of 91 days after the end of the first. Its duration, 11 days at the median, marks it as a later operation distinct from the first. The first cycle of each piece is kept, and the 999 later cycles leave the dataset. So do the 98 cycles of zero or one day that remain after the corrections, 62 of them in NC-1.

The pieces sent to the external painter form a recognisable population. The final table holds 538 of their cycles, 509 of them in NC-2, entered between July 2025 and March 2026; within NC-2 they last a median of 25 days, against 42 for the rest of the commission. The figure is consistent with the practice described in Section 1.2. The destination is known at entry, because the dispatch itself opens the cycle.

4.6 The target and the observation window

The target is the duration of the cycle in calendar days, weekends and company shutdowns included, a choice made because a piece stays in painting even when the department is closed. A working-day variant, net of weekends and of the August and Christmas shutdowns, has a correlation of 0.984 with the calendar version. The

Commission	Openings	Median age	Over 180 days	With a later event
NC-1	420	220 d	61.2%	81.9%
NC-2	1,299	363 d	81.1%	93.8%
NC-3	1,169	81 d	13.6%	89.7%
RF-1	254	111 d	0.0%	94.5%
Total	3,142	166 d	46.8%	90.8%

Table 4.4: Openings without a recorded closing on 1 September 2026, by commission.

shutdown periods are declared in advance, since inferring them from days without events would bring into the target information later than the moment of prediction.

A cycle that opened shortly before the extraction is observed only if it has already closed, so the long cycles of the last months are still open and missing. The longest observed duration falls to 47 days for entries in July 2026 and to 29 for those in August. On 1 September, 384 cycles of the July entries and 76 of the August ones were still open. Overall, 4.06% of cycles have an observation window shorter than the 90th percentile of the duration.

Openings without a closing (Table 4.4) have no single reading. In NC-3 the median age is under three months, consistent with a commission still in progress. In NC-2 it exceeds a year, and in 93.8% of cases the piece has a later event, such as a shipment to the shipyard or an entry into the warehouse; the piece has left the department, and its exit from painting was not recorded. That event cannot replace the missing exit, because it measures painting, transport and storage together, which Figure 4.1 keeps apart. Openings without a closing stay out of the dataset, since they have no target.

Between 60 and 100 days fall 1,590 cycles, and whether durations this long are real was checked directly. The check was made on the version of the dataset that preceded the corrections of Section 4.5, and a dedicated file was built for it with the complete movement history of the 942 pieces of the band that have at least one event between entry and exit, since for the others no intermediate movement allows the time stamps to be checked. Pieces that travel on the same document and enter and leave painting within three days of each other were gathered together, and one representative was chosen for each such set, with the number of pieces it stands for and the range of their durations. This reduced the reading to 140 representatives, whose sequences of movements were then examined one by one. The time stamps proved correct, and the pieces had entered painting and presumably been set aside for other priorities. The recorded duration is real, and the reason for the wait does not appear in the data.

4.7 Reducing the dataset

Table 4.5 reports every step from the 41,707 pieces of the exports to the 14,245 cycles on which the models are trained and evaluated. The first filters remove records that do not represent physical pieces: rows attached to a generic cabin, elements named after a supplier or a category of goods, technical documents, mock-ups and cabins filled in with the name of an element. Then come the pieces without a finish code, which cannot be linked to the dictionary, and the pieces of families other than wood and lacquer.

Step	Removed	Remaining
Pieces in the exports		41,707
Generic cabin	950	40,757
Element that is not an element	144	40,613
Piece that is a document	1,207	39,406
Mock-up	319	39,087
Cabin filled with an element name	30	39,057
No finish code	4,821	34,236
Family other than wood and lacquer	14,222	20,014
Closed cycles of the retained pieces		15,342
Duration of zero or one day	98	15,244
Cycles after the first, per piece	999	14,245

Table 4.5: From the exports to the modelling dataset.

The family filter is the largest, and Table 4.6 shows that most families either never reach painting or go through it for a different operation. Unfinished pieces receive only a primer coat on the components that need one, and leather gets in painting only the preparation of its support, while the operation that sets its timing takes place at the upholsterer. Metal and textile go through painting in about a third of cases, but they amount to a few hundred pieces. The 27 pieces classified as both lacquer and wood are excluded because their family is not defined.

Family	Pieces	With a cycle	Outcome
Wood	10,483	76.9%	retained
Lacquer	9,531	65.4%	retained
Unfinished	6,771	12.1%	excluded
Other	3,042	6.0%	excluded
Leather	2,511	33.6%	excluded
Laminate	577	2.8%	excluded
Stone	436	0.0%	excluded
Textile	420	31.0%	excluded
Metal	347	31.1%	excluded
Glass	57	0.0%	excluded
Mirror	34	23.5%	excluded
Lacquer and wood	27	96.3%	excluded
Total	34,236		

Table 4.6: Pieces reaching the family filter, by dictionary family, with the share that has at least one closed cycle.

4.7.1 A process that cannot be measured

Upholstery, the other finishing operation, has no measurable target in these exports. The dispatch of leather pieces to an external upholsterer is recorded to very different extents from one commission to another (Table 4.7), and the return almost always appears as a receipt from transport, without the name of the supplier. In the refit, where 90.5% of dispatches are tracked, the cycle can be rebuilt by closing it at the next receipt, which gives 208 cycles with a median of 36 days. In NC-3, with 1.5% tracked, it practically does not appear in the data.

Commission	Leather pieces	Sent to an upholsterer	Sent to internal upholstery	Upholsterers
NC-1	353	57.2%	2.8%	3
NC-2	793	38.0%	17.8%	5
NC-3	1,100	1.5%	4.8%	3
RF-1	274	90.5%	2.6%	2

Table 4.7: Tracking of upholstery on the leather pieces.

Where every site follows the same path, as in painting, the checkpoints give valid time boundaries; where it changes with the supplier, as in upholstery, they cannot be rebuilt. Leather pieces therefore stay out of the dataset.

4.8 The finish dictionary

The finish is the material datum recorded in every commission, and material data are one of the classes that the studies reviewed in Section 3.2 adopt least and, where they do, find useful (Steinberg *et al.*, 2022). It is filled on 87.64% of the physical pieces, from 80.25% in NC-2 to 98.17% in RF-1 (Figure 4.3). Using it requires a normalisation step, because the finish code is valid only within its commission. It is defined commission by commission together with the interior design, and the same string can denote different finishes in two commissions. The correct key is therefore the commission-code pair, and treating the bare code as a global identifier would introduce a systematic error.

The textual description of the finish is filled on 51.19% of the physical pieces; among the retained pieces it differs from the bare code on 80.91%, and on only 35.77% in NC-1. Automatic rules applied to the text cannot classify the finishes reliably, and the dictionary was therefore compiled by hand, with domain knowledge. It assigns a material family and a colour class to each of the 197 commission-code pairs in the exports. It is the only step that requires knowledge not derivable from the data, and it is also what makes the family filter possible.

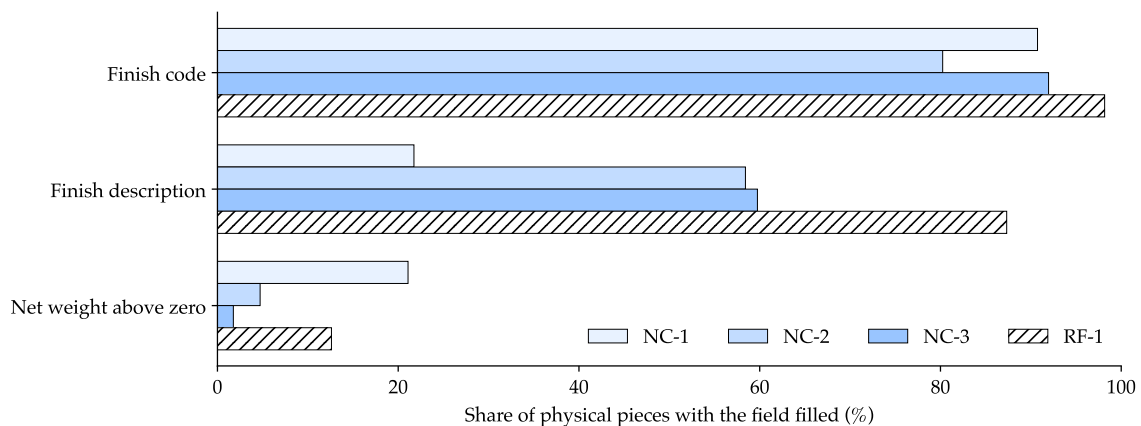


Figure 4.3: Share of the physical pieces with each field filled, by commission.

4.9 Exploratory analysis

4.9.1 The distribution

With a median of 30 days and a mean of 34.9, the duration is strongly right-skewed (Figure 4.4); the first quartile falls at 20, the third at 48 and the 90th percentile at 63,

while the maximum is 276. Of the cycles, 11.1% exceed 60 days and 1.6% go beyond 90, and that 1.6% accounts for 21.5% of the total variance. This skewness bears on the choice of metric discussed in Section 2.5.

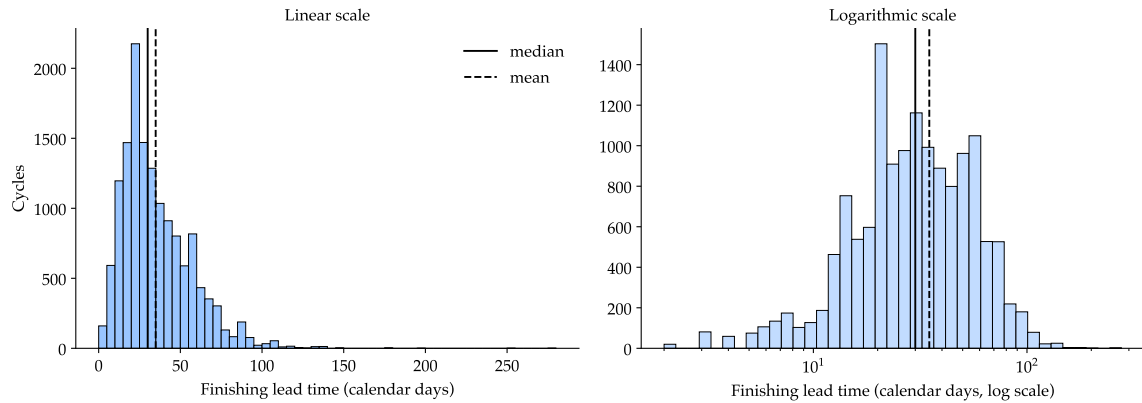


Figure 4.4: Distribution of the finishing lead time on linear and logarithmic scales.

4.9.2 The commissions

The four commissions behave differently (Table 4.8 and Figure 4.5). NC-2 has a median of 39 days against 22 for NC-3 and the widest dispersion, NC-1 sits in between, and RF-1 has the most compact distribution, with a standard deviation of 8.3.

Commission	n	Mean	Std. dev.	p25	Median	p90
NC-1	4,542	33.6	19.5	20	30	60
NC-2	4,648	42.5	23.1	26	39	71
NC-3	4,257	29.3	20.0	16	22	56
RF-1	798	27.8	8.3	22	26	40
Total	14,245	34.9	21.2	20	30	63

Table 4.8: Descriptive statistics of the finishing lead time by commission, in calendar days.

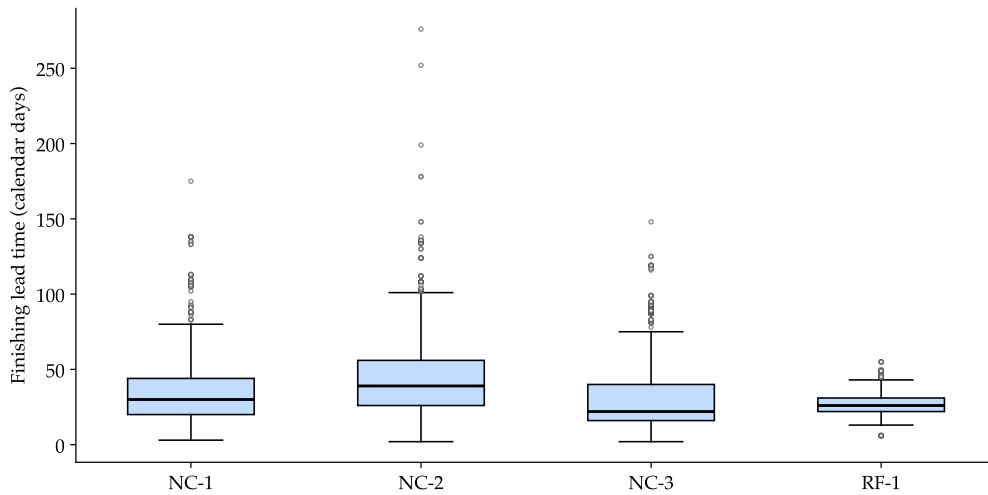


Figure 4.5: Finishing lead time by commission.

4.9.3 The trend over time

Figure 4.6 reports the monthly median and the number of cycles opened in each month. The median moves between 18 and 64 days over the period, and the active commissions change with it. In the first months almost only NC-2 enters, from February 2026 NC-3 prevails, and RF-1 occupies April and May 2026. Two months have fewer than 100 cycles, February 2025 with 95 and August 2026 with 25. In the final months the median has to be read together with the censoring described in Section 4.6.

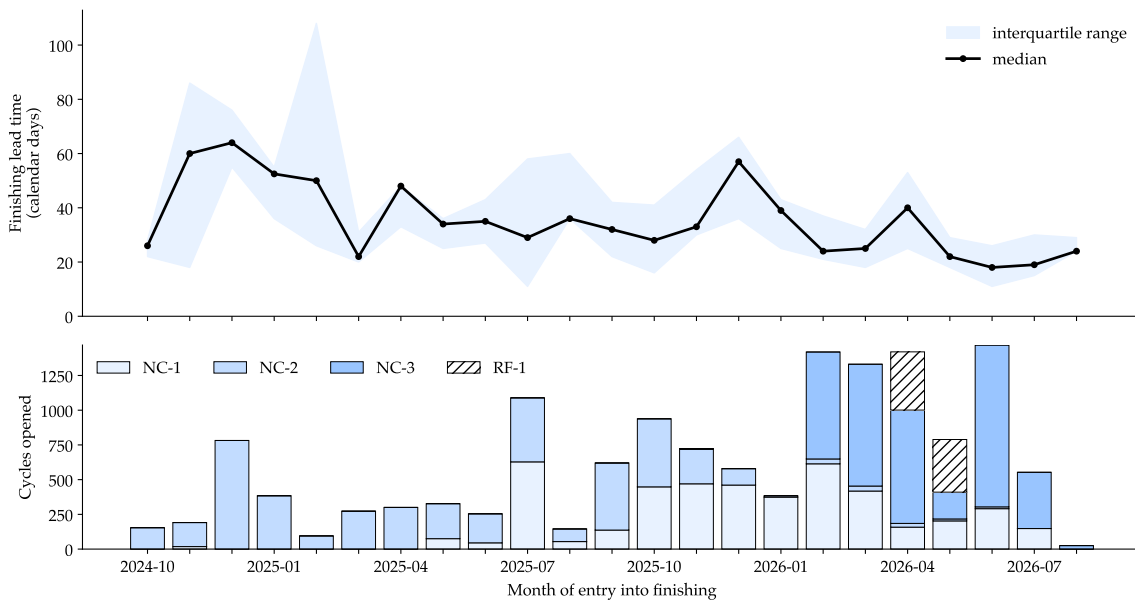


Figure 4.6: Monthly median of the finishing lead time, with its interquartile range, and cycles opened in each month by commission.

4.9.4 Material families and day of entry

After the filter, wood accounts for 56.3% of cycles, with a median of 36 days, and lacquer for 43.7%, at 24. What the next chapter models is therefore, in practice, wood and lacquer, and the 12-day gap between the two medians is the first sign that the family carries information.

The day of entry also has a visible effect, with a median of 27 days for cycles opened on a Wednesday and 32 for those starting on a Thursday or Friday. The 524 cycles opened on a Saturday have a median of 24 days.

4.10 The finish as a predictor

Within each family the finish code explains 29.4% of the variance of the duration for wood and 20.9% for lacquer, measured as the share between codes. Figure 4.7 shows the most frequent codes of the two families. Between two wood finishes of the same commission the median nearly doubles, so treating the family as a single category would lose information.

The same figure shows the limit. Finish codes are defined per commission and overlap little: NC-1 uses 13, 69.2% of which appear nowhere else; NC-2 has 24, none shared; NC-3 has 15, 73.3% exclusive; RF-1 has 3, all present in other commissions but with a different meaning. LQ01, for example, denotes a lacquer of another colour in each of NC-1, NC-3 and RF-1, with different medians. The most informative predictor is therefore usable within a commission and unusable on a new one, where every code appears as an unseen category. The problem also arises within a single

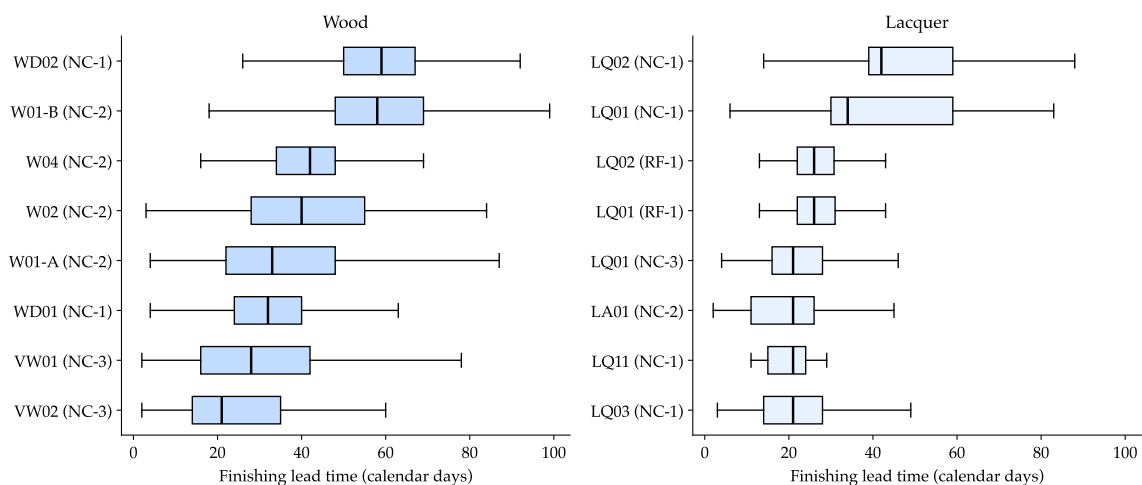


Figure 4.7: Finishing lead time of the most frequent finish codes, within the two families. Each code is shown with its commission.

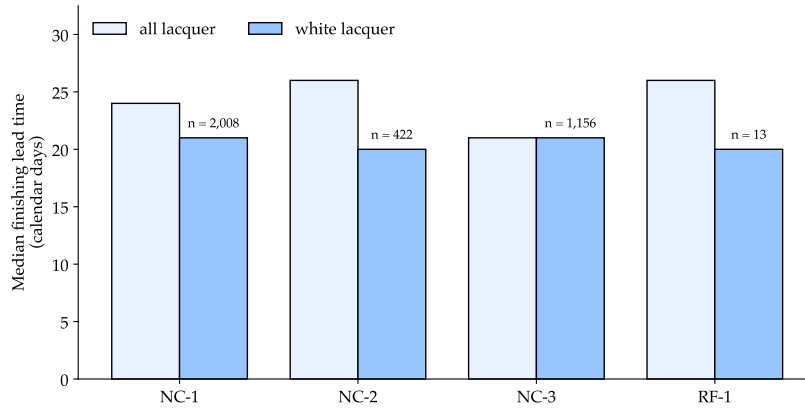


Figure 4.8: Median finishing lead time of white lacquer and of all lacquer, by commission.

commission, because each commission introduces new elements, cabins and finishes as it progresses. On the monthly folds of the protocol of Chapter 5, the share of test rows with at least one of element, cabin and finish code never seen in training is 31.3% in NC-1, 32.5% in NC-2, 46.3% in NC-3 and 69.4% in RF-1. A model trained once and then frozen therefore ages by construction, and the protocol retrains every month.

Something of the finish does carry over between commissions, however. For white lacquer, present in all four commissions under different codes, the median stays between 20 and 21 days, while that of all lacquered pieces ranges from 21 to 26 (Figure 4.8; RF-1 has only 13 such pieces). The transferable information might therefore lie in an abstraction of the finish code, the colour class, rather than in the code itself. It is an observation on four commissions only, and Chapter 5 submits it to a direct test.

4.11 The other recorded fields

Every movement carries the number of the checkpoint document on which the piece travels (Section 1.2.9). In the final dataset there are 939 entry documents; a document carries 15.2 pieces on average, 9 at the median and 36 at the 90th percentile, with a maximum of 222, and 66.2% of pieces enter painting on a document of at least 21 pieces. The entry document is known at the moment of prediction, and Chapter 5 tests its composition.

The *requested date* is the day by which the department asks for the piece to be ready, and it underlies the estimate currently used there. On the entry row it is filled for 49.5% of cycles, 40.7% for wood and 60.8% for lacquer. When present, it falls exactly

21 days after entry in 80.8% of cases, and in 5.3% it is the same day, a value treated as missing.

The field is not a frozen record, however. For 49.6% of the pieces with a date it differs across movement rows, by a median of 48 days, and whether the entry row holds what was known at that moment cannot be established with certainty.

Rework is recorded on 7.8% of cycles. It is related to duration, but of the rework events on the pieces of the dataset only 24.3% precede the opening of the cycle, while 30.4% fall within it and 45.3% after the closing. Used as a feature, it would give the model information later than the moment of prediction, and it therefore remains a variable for the analysis of errors.

The *supplier* appears in its own field and, where that is empty, in the phase string. For the external painter it is recorded on the dispatch row that opens the cycle, and is therefore known at entry; read from that row, it is present in 83.3% of these cases, and in the rest only the phase string names it.

4.12 The candidate features

Only information available when the piece enters painting, the moment of prediction, may enter the features. Table 4.9 lists them by group, with the moment at which each becomes known. Categorical features with many levels are encoded as described in Section 2.2.1, and the finish code always enters as the commission-code pair.

Group	Features	Known
Piece	deck, cabin, cabin type, element, functional family	from design
Finish	material family, finish code (commission and code), colour class	from design
Calendar	month, week and weekday of entry	at entry
Shop state	concurrent load; backlog, also of the same family; maximum and median age of the queue; entries and exits over 7 and 28 days; median of the durations closed over 30 and 60 days	from the events up to entry
Document	number of pieces, share of wood, number of elements, cabins and codes, prevailing family	when the document is created
Commission	commission, commission type, recording regime	at entry
Other	supplier, project phase, requested date	at entry, with the limits of Section 4.11

Table 4.9: Candidate features, by group, and the moment at which they become known.

The *concurrent load* counts the physical pieces of any material that are in painting at the same site on the day of entry, the piece itself excluded. A piece is present from the day it enters until the first event showing that it has left: the closing, a dispatch towards the exit, a record elsewhere or a new entry. For pieces sent to the external painter the site is its own location, and a piece stops counting at its return or when it is first recorded somewhere else. The start of the interval and each of these events are observable when they happen. An entry that is never followed by any of them, because the exit of the piece was not recorded, ends at the piece's last event, which is known only afterwards; Chapter 5 measures what this choice changes.

Other quantities cannot be built. The age of the piece at entry has no reliable starting point. The event that would date it, the creation of the label, was logged only by a single site and marks the authorisation of production (Section 1.2.9) rather than the start of work on the piece. After the change of regime (Section 4.3) the phase field admits only dispatches, receipts and final shipments, and the creation of the label no longer appears at all. The gloss class of the finish is missing precisely for the most numerous lacquer codes. Weight and dimensions, which Steinberg *et al.* (2022) and Canedo Rosa, Gonçalves and Barbalho (2025) use as material data, are too sparsely filled (Figure 4.3). From the candidates, Chapter 5 selects the set used by the model.

Chapter 5

Model Development and Evaluation

5.1 How the predictions are judged

At the moment a piece enters painting, the model estimates how many calendar days will pass before its first return, the target built in Section 4.6. The observations are the 14,245 cycles of Section 4.7. The duration has the strong right skew described in Section 4.9, with a few very long cycles, and so that these do not dominate the fit, the model works on the logarithm, $y = \ln(1 + d)$, with d in days. The prediction is brought back to days with $d = e^y - 1$. Predictions are judged by their mean absolute error in days, which on a duration this skewed describes typical performance better than the squared error, dominated by the few longest cycles (Section 2.5).

Each cycle carries the 13 features of the model, all known at entry. They describe the aspects on which the duration may depend: the piece (deck, cabin, element, functional family), the finish (material family and finish code, the latter as the commission-code pair), the commission (commission, commission type, recording regime), the calendar (month and week of entry, weekday) and the department (concurrent load). They were fixed in the first phase of development, the exploratory work that preceded the evaluation of this chapter, one group per aspect, from the candidates of Table 4.9. Two candidates of the piece and finish groups stay out, the cabin type and the colour class. Within a commission they are functions of the cabin and of the finish code, which the model already has, and they could add information only on a commission never seen, where cabin and code are unknown. The remaining candidates were tried by adding them to the 13, one group at a time (Section 5.11).

5.1.1 The baselines

To measure what learning adds, four baselines were built, simple rules that require no training (Section 2.5). Three of the four are medians, because the median is the constant that minimises the absolute error; with a skewed target a mean would err

more and make the comparison too easy. They are recomputed every month on the cycles the model sees in training, so that both start from the same information.

The main baseline is the *family median*, the median by material family, with one value for wood and one for lacquer. It simulates someone who, having to estimate the duration, knows whether the piece is wood or lacquer. Since the medians of the two families differ by 12 days (Section 4.9), this distinction alone carries information, and it makes the family median the reference opponent.

Beside it, the *standard rule* reproduces the estimate in use: the requested date where it is filled in, and 21 days from entry otherwise. It measures how much the model would improve current practice. When present, the requested date falls 21 days after entry in 80.8% of cases (Section 4.11). The median duration is 30 days, however, so the standard rule underestimates systematically.

The other two are simpler medians. The *segment median* divides the pieces by functional family and material family into 11 cells, and tests whether a finer subdivision is enough to do better. Some cells hold few pieces, and their value is unstable. The *global median* is a single value, computed on all pieces without distinction. It is the weakest opponent.

5.1.2 Month-by-month evaluation

A model has to be tested in the conditions in which it would be used, learning from the past and predicting the future, and the evaluation therefore follows the rolling origin of Section 2.4.4. For each test month the first day acts as the *boundary*. Training uses the cycles that entered before that day, testing those of the month itself. The boundary then moves to the next month. Each month evaluated in this way forms a *fold*, which repeats the division into training, validation and test of Section 2.2. The validation used to choose the hyperparameters takes place inside the training data. At least 500 cycles are needed to train and 100 to test, because with fewer the error depends on a handful of pieces. For January 2025 there are not enough training cycles, February has fewer than 100 to test, and August 2026, the last month of the data, has 25, so the tests run from March 2025 to July 2026, 17 months. Training grows from 1,122 to 12,564 cycles, while the test sets range from 145 to 1,469 rows, 12,614 in all, a median of 13.8% of the fold.

The boundary raises a problem that affects the whole chapter. On the first day of a test month many pieces that entered in the previous weeks are still in painting. Summed over the 17 folds, 78.7% of the cycles that arrived in the month before the boundary are still open, and at that date their duration does not yet exist. Training on them as well, reading their duration from the September 2026 extraction, gives the

model information that nobody had at the moment of prediction. A protocol built in this way is called *leaking* here, because it lets information pass from the future into training. The *honest* protocol instead trains only on the cycles that entered before the boundary and had already closed by then, those a planner would have known. The figures of this chapter come from the honest protocol unless stated otherwise. The leaking protocol is nonetheless the most immediate way to split these data in time, because it trains on all the pieces that entered before the boundary without asking whether they have finished, and the work had started with it; for this reason the difference between the two is measured.

Even the honest protocol has a cost. Of the months just before the boundary only the pieces that have already finished are visible, the fastest, while the long ones enter training later, when they close. This *selection by closure* makes the training sample faster than reality in its last months.

5.1.3 The hyperparameter search

The hyperparameters of each algorithm are searched on the training rows of the fold alone, because choosing them by looking at later months would bring the future into the model by another route. Each search explores 50 combinations in the spaces of

Algorithm	Hyperparameters and ranges	Encoding
Regularised linear regression	L1, L2 or elastic net penalty; strength from 10^{-3} to 10^3 (log); L1 share from 0.1 to 0.9 for the elastic net	target
Random forest	200 to 1,000 trees; maximum depth 4, 8, 12, 16, 24, 32 or unlimited; minimum rows per leaf from 1 to 32; features per split: square root, logarithm or a fraction from 0.3 to 1	target or one-hot
XGBoost, LightGBM	learning rate from 0.005 to 0.3 (log); depth from 3 to 12; row subsampling from 0.6 to 1; column subsampling from 0.4 to 1; L2 penalty from 10^{-3} to 10 (log); up to 3,000 trees, stopped by early stopping	target or one-hot
CatBoost	learning rate from 0.005 to 0.3 (log); depth from 3 to 10; row subsampling from 0.6 to 1; column subsampling from 0.4 to 1; L2 penalty from 10^{-3} to 10 (log); up to 3,000 trees, stopped by early stopping	target or one-hot
Support vector regression (RBF kernel)	C from 0.1 to 1,000 (log); epsilon from 0.01 to 1; gamma automatic or from 10^{-4} to 10^{-1} (log)	target
k -nearest neighbours	k from 3 to 50; uniform or distance weights; Manhattan or Euclidean distance	target
Neural network (MLP)	1 to 3 layers of 32-256 units; L2 penalty from 10^{-5} to 10^{-1} (log); initial learning rate from 10^{-4} to 10^{-2} (log); early stopping, at most 400 iterations	target

Table 5.1: Search spaces of the hyperparameters, by algorithm. Ranges marked “log” are explored on a logarithmic scale.

Table 5.1, with the sequential method of Section 2.4.3, and is repeated every three test months, a compromise between computing cost and updating. Each combination is evaluated by a cross-validation on three blocks of the training rows (Section 2.4.4), which keeps in the same block the pieces that entered the same site on the same day. Those pieces travel on the same checkpoint documents (Section 1.2.9) and share the conditions of the department, and splitting them across blocks would put into validation pieces almost identical to ones already learned. Once the combination is chosen, the model is refitted on all the training rows of the month. In the following two months it is retrained on the updated data with the same combination.

For the boosting algorithms the number of trees is set by the early stopping of Section 2.4.1, measured on a further 15% of the site-day groups held out of the fit. Categorical features with many levels are encoded as in Section 2.2.1. For the tree ensembles the choice between target and one-hot encoding is part of the search, while the other methods always use target encoding, because in a matrix of about 300 columns of zeros and ones distances lose their meaning.

5.1.4 When a difference is real

Each test month gives one mean absolute error for the model and one for the baseline, so a comparison produces 17 differences. Their mean says how much the model gains. The paired *t*-test establishes whether that mean can be told apart from chance. The paired version is used because model and baseline are measured on the same months, and the level of each month weighs on both. The outcome of the test is the p-value, the probability of observing a mean difference at least as large if model and baseline in fact had the same expected error. With p below 0.05 the difference is called *established*; above it, *not distinguishable from zero*, whatever its size.

Neighbouring months share most of their training data, so the 17 differences are not entirely independent, although the test treats them as such. Nadeau and Bengio (2003) show that when error estimates obtained on different tests share data, treating them as independent underestimates their variance and leads the test to declare a difference more often than it should. The correction they propose is designed for repeated random splits of the same data, and it does not apply as it stands to a monthly evaluation, in which the test months do not overlap. The direction of the effect remains, however. The p-values of this chapter may be smaller than they should be, and a result close to the threshold has to be read with caution.

The reference figure is the mean over months, in which each month weighs one, fixed before the results because it is the unit the test works on. Two other averages are reported beside it. In the mean by piece every piece weighs one, so months with more cycles count more. In the mean by group each month weighs the number of

combinations of site and day of entry it contains, and for the reason seen above these are the units closest to independent observations.

Many comparisons appear in the chapter, a consequence of the number of experiments carried out: eight algorithms, 12 segmentation configurations, groups of features, variants of the model. When the tests are many, some p-values below 0.05 can appear by chance even if no difference is real. Comparisons that answer the same question form a *family*, for example the seven algorithms compared with the random forest, and within it the p-values are corrected with the method of Holm (1979). With n comparisons, the method orders the p-values from the smallest to the largest and compares the first with $0.05/n$, the second with $0.05/(n - 1)$, and so on, stopping at the earliest comparison that exceeds its threshold. That comparison and all the following ones are not established. Holm proves that in this way the probability of declaring established even a single non-existent difference does not exceed 0.05, whatever the true differences in the family. The tables report the corrected p-value, obtained by multiplying each nominal one by the number of comparisons still in play, never letting it fall below the previous one; compared with 0.05, it gives the same decision as the procedure. A result is called established only if its corrected p-value stays below 0.05. One that passes the threshold only before the correction is *nominally significant*.

The main questions of the chapter, which correspond to the first and the fourth research question, are instead judged on the nominal p-value: whether the model at entry errs less than the family median, and whether the updated prediction does, the estimate revised during processing once the earliest pieces of the batch have returned (Section 5.7). The correction within families was adopted after the results had been seen, and it is the more cautious choice.

5.2 The choice of algorithm

All eight algorithms of Chapter 2 were tried, each with its own hyperparameter search (Table 5.1).

Table 5.2 compares each algorithm with the random forest, the model used for the analyses of this chapter. The forest was chosen during development, on the leaking protocol, when the best four algorithms could not be told apart, with a tie-break rule fixed at that moment: with errors within 0.2 days of each other, the one that varies least across months wins.

On the honest protocol the forest is no longer in the lead. Regularised linear regression has the lowest error, 14.04 days against 14.95 for the forest, and does better in

Algorithm	MAE (d)	Against the forest (d)	p	Holm p
Regularised linear regression	14.04	+0.91	0.310	1.000
CatBoost	14.31	+0.64	0.283	1.000
Support vector regression	14.36	+0.59	0.524	1.000
Random forest	14.95			
<i>k</i> -nearest neighbours	15.21	−0.26	0.728	1.000
XGBoost	15.34	−0.39	0.434	1.000
LightGBM	15.39	−0.44	0.558	1.000
Neural network (MLP)	18.19	−3.24	0.035	0.247

Table 5.2: The eight algorithms on the honest protocol: mean absolute error (mean over months) and difference from the random forest, positive when the algorithm errs less, with the p-value of the paired test and the one corrected over the seven comparisons with the forest.

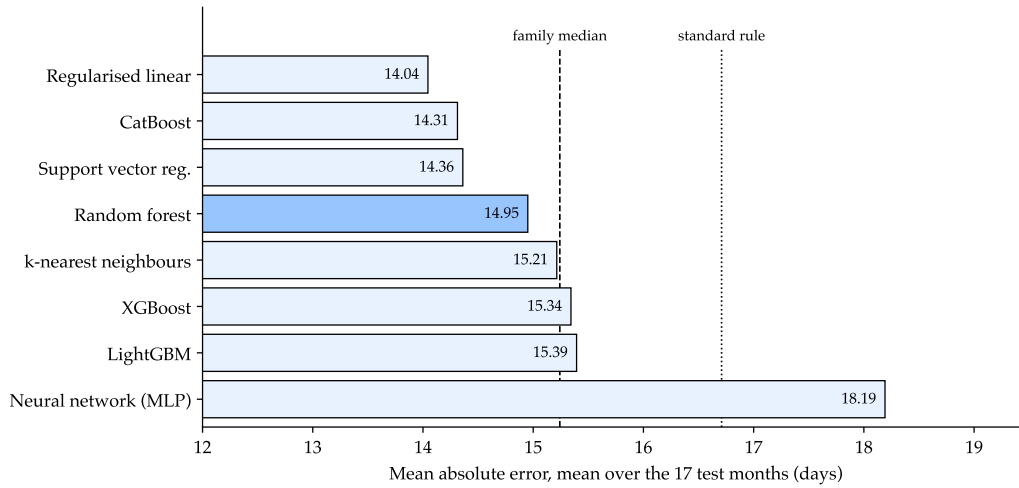


Figure 5.1: Mean absolute error of the eight algorithms on the honest protocol, with the family median and the standard rule.

11 months out of 17. It also varies less from one month to the next, with a standard deviation of the monthly error of 4.79 against 5.40. The difference cannot be told apart from chance, however (p 0.310), and with the correction for the seven comparisons its p-value is 1.000 (Figure 5.1).

CatBoost and support vector regression come next. CatBoost was tried both like the other algorithms, with the features already encoded and the standard scheme, and with the categorical features passed as such and ordered boosting, the two devices described in Chapter 2. The two versions err by 14.31 and 14.40 days, a difference not distinguishable from zero (p 0.834), and the table reports the first. The neural network is the only algorithm distinguishably worse than the forest before the correction, by 3.24 days (p 0.035).

This result has a reading consistent with the bias-variance trade-off of Section 2.4.2. The honest protocol trains on fewer rows and, in the last months before the boundary, only on the fast pieces; under these conditions the more flexible models pay for their variance, and a linear model with a regularisation term matches the tree ensembles. The neural network, with more parameters than these data can estimate, comes out worst of all. Grinsztajn, Oyallon and Varoquaux (2022) observe the same order on 45 tabular datasets of medium size, about 10,000 rows, where trees remain the best and neural networks suffer from uninformative features and irregular functions. Here the trees show no such advantage over the linear model.

The forest remains the model of the chapter, because its difference from linear regression cannot be told apart from chance and the choice had been fixed beforehand. Since linear regression and CatBoost are the two algorithms with the lowest error, they were also used in other experiments of the chapter, beside the forest and with the same hyperparameter search.

5.3 The prediction at entry

The first research question asks how accurately the duration of a piece can be predicted at the moment it enters painting, compared with the baselines. At entry the model, a random forest, reduces the error of the family median by 1.9% and that of the standard rule by 10.5%, and neither difference is distinguishable from zero. Table 5.3 reports the comparisons with the four baselines, and Figure 5.2 the course of the error over the 17 months.

On average the model errs by 14.95 days in absolute value, against 15.24 for the family median and 16.71 for the standard rule. This is about half the duration of a cycle, which has a median of 30 days. Figure 5.2 shows where the verdict comes from. The error of the model ranges from 7.9 days in May 2026 to 25.2 in April 2025, and

Baseline	Mean over months	By piece	By group	p	Holm p	Months won
Family median	1.91%	5.27%	5.48%	0.835	1.000	10/17
Standard rule	10.52%	6.23%	6.93%	0.124	0.496	10/17
Segment median	2.02%	4.07%	4.57%	0.785	1.000	11/17
Global median	11.80%	11.31%	11.98%	0.186	0.558	12/17

Table 5.3: The prediction at entry against the baselines: reduction of the mean absolute error in the three averages, p-value of the paired test, Holm p-value over the four baselines and months won. The comparison with the family median is one of the two main questions and is judged on the nominal p-value.

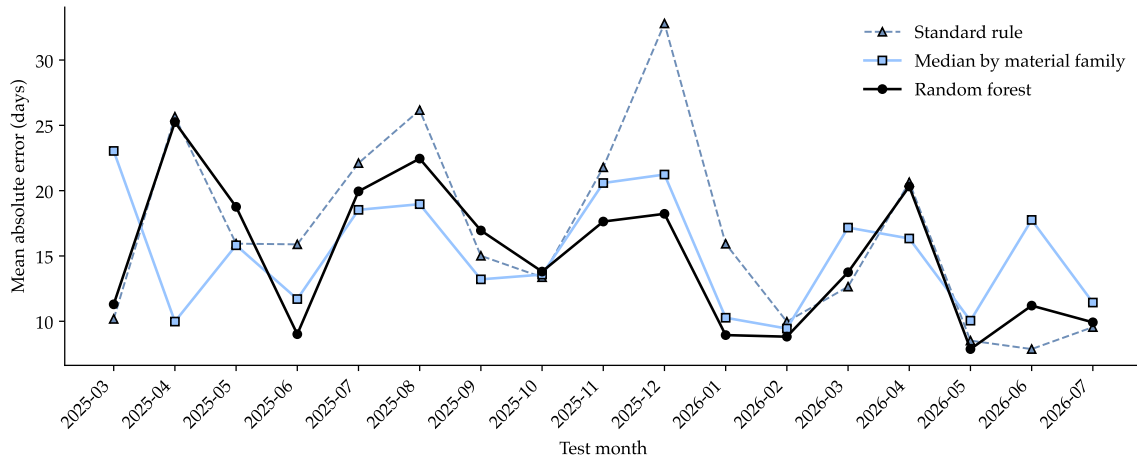


Figure 5.2: Mean absolute error in each test month: the random forest, the family median and the standard rule.

the family median rises and falls with it. Compared with this swing the difference between the two is small, and the model does better in 10 months out of 17.

The two simpler medians measure what the material family is worth. Splitting the pieces further, into the 11 cells of the segment median, adds nothing, since it errs by 15.26 days, as much as the family median. The global median, which ignores the family, errs by 16.95 days, so knowing whether a piece is wood or lacquer reduces the error by 10.1%.

Since the error changes so much across months, a baseline that follows the recent level might do as well. This was checked with a *rolling median*, recomputed every month on the cycles that entered in the previous three months and had closed

Pieces and baseline	Forest	p	Linear	p	CatBoost	p
All, MAE (d)	14.95		14.04		14.31	
All, family median	1.91%	0.835	7.85%	0.174	6.09%	0.461
All, standard rule	10.52%	0.124	15.94%	0.028	14.34%	0.062
Wood, MAE (d)	16.45		14.50		16.49	
Wood, family median	4.42%	0.689	15.78%	0.046	4.19%	0.697
Wood, standard rule	18.19%	0.041	27.91%	0.003	17.99%	0.034
Lacquer, MAE (d)	13.00		13.56		11.83	
Lacquer, family median	-6.68%	0.378	-11.29%	0.216	2.86%	0.783
Lacquer, standard rule	-3.48%	0.646	-7.94%	0.325	5.78%	0.586

Table 5.4: The prediction at entry with the forest and the two algorithms with the lowest error: mean absolute error and reduction of the error of the two main baselines (mean over months), on all pieces, on wood and on lacquer, with the p-value of the paired test. Regularised linear regression and CatBoost are rerun with the same hyperparameter search as in Table 5.2.

by the boundary. The model errs 7.98% less, with $p = 0.382$, so the difference is not distinguishable from zero.

The three averages give different percentages because the gain of the model is not the same in large and in small months. With the months ranked by their number of test rows, the model reduces the error of the family median by 8.18% in the eight largest and increases it by 1.16% in as many of the smallest, so weighting by piece raises its advantage to 5.27%. Against the standard rule the opposite happens, and weighting by piece lowers the gain to 6.23%.

With the hyperparameters found once during development, searched on the leaking protocol, the advantage over the standard rule would have been 13.50% ($p = 0.042$). The search on the folds of the honest protocol, adopted after both versions had been seen, is the less favourable to the model, and the advantage falls to 10.52% ($p = 0.124$).

Regularised linear regression and CatBoost give the same verdict (Table 5.4). Although they reduce the error of the family median by 7.85% ($p = 0.174$) and 6.09% ($p = 0.461$), against 1.91% for the forest, none of the three differences can be told apart from zero. Against the standard rule linear regression reaches 15.94% with $p = 0.028$, the only value below 0.05, which the correction over the eight algorithms raises to 0.223, while CatBoost gets to 14.34% ($p = 0.062$).

5.3.1 Wood and lacquer

The value of the model is concentrated on wood. The comparisons were also read separately on the wood and on the lacquer pieces, the two branches of the analysis. On the wood pieces the model reduces the error of the family median by 4.42% ($p = 0.689$) and that of the standard rule by 18.19% ($p = 0.041$, 10 months out of 17). The result on wood is nominally significant, because among the four comparisons of the two branches the correction raises its p -value to 0.164. On lacquer its error exceeds that of both baselines, by 6.68% and 3.48%, without the difference being distinguishable from zero.

With linear regression the advantage on wood is wider, 15.78% over the family median, nominally significant, and 27.91% over the standard rule, which holds up under the correction between the two alternative algorithms. On lacquer linear regression errs more than both baselines, by 11.29% and 7.94%, while CatBoost is the only one of the three whose error stays below theirs, by 2.86% and 5.78%. None of these differences on lacquer can be told apart from zero.

A compatible reading is that wood, with the widest dispersion and the most heterogeneous finish codes of Section 4.10, leaves room for piece-level information, whereas lacquer is already described well by a median.

5.3.2 Robustness checks

In the last two folds, June and July 2026, many of the pieces that entered were still in painting at the extraction of 1 September, so the data contain mostly the fast pieces of those months (Section 4.6). Without these two folds no comparison changes its verdict. Against the standard rule the advantage becomes 12.60% (p 0.075), and 20.78% on wood (p 0.023). Two estimates that are not distinguishable from zero change sign, the advantage over the family median, which becomes -1.36% (p 0.890), and that over the segment median, -0.26% (p 0.974).

The second check concerns the receipt that closes the cycle. In 919 cycles the first receipt is followed by another with no dispatch in between, and the data do not say which of the two really ends finishing (Section 4.5). The whole evaluation was therefore repeated closing these cycles at the second receipt, which comes a median of seven days later. The error of the model rises from 14.95 to 15.27 days, the baselines worsen more, and no verdict changes. The gain remains 0.82 days over the family median (p 0.513) and 2.49 over the standard rule (p 0.084).

A last check concerns the concurrent load. For a piece whose exit was never recorded, the count keeps it in the department until its last event, which is known only afterwards (Section 4.12). The load was recomputed in two ways that use only what is known at the date: keeping those pieces in the department until an exit event appears, or for at most 60 days from entry. With the first variant the load grows by a median of 161 pieces per row, and with neither does the error change distinguishably, $+0.06$ days (p 0.693) and -0.05 (p 0.830).

5.3.3 After the first year of history

A firm that adopted the model would train it on all the history it has recorded, which in these data covers 23 months of entries. In the rolling-origin evaluation, by contrast, the first test months have little history behind them. In the first, training holds five months of entries, and until September 2025 fewer than twelve, so the month of the year being predicted had never appeared in it.

To come closer to the position of a firm that uses the model with a longer record, the comparisons were also read over the ten test months from October 2025, the first with at least a year of entries in training, and over the same period without June and July (Table 5.5). This reading was decided after the results had been seen, and its figures are descriptive.

In the first seven test months the model does better than the family median only twice, and from October 2025 in eight of the ten, while its advantage in the monthly mean rises from 1.9% to 11.7% (p 0.076). By piece it is 11.2% and by group 10.8%.

Model and baseline	17 months	From October 2025	From October 2025, without June and July
Forest, family median	1.91% p 0.835, 10/17	11.75% p 0.076, 8/10	7.83% p 0.219, 6/8
Forest, standard rule	10.52% p 0.124, 10/17	14.76% p 0.201, 6/10	19.38% p 0.122, 6/8
Linear regression, family median	7.85% p 0.174, 11/17	12.82% p 0.080, 7/10	7.40% p 0.236, 5/8
CatBoost, family median	6.09% p 0.461, 10/17	15.97% p 0.057, 7/10	12.13% p 0.174, 5/8
Forest on wood, family median	4.42% p 0.689, 11/17	15.62% p 0.138, 8/10	10.55% p 0.352, 6/8
Forest on lacquer, family median	-6.68% p 0.378, 7/17	5.73% p 0.364, 5/10	6.04% p 0.419, 4/8

Table 5.5: The prediction at entry over the 17 test months, from October 2025, with at least a year of history in training, and from October 2025 without June and July: reduction of the error of the baseline (mean over months), p-value of the paired test and months won. The readings from October 2025 were decided after the results and are descriptive.

From October 2025 every comparison in the table improves. Regularised linear regression and CatBoost reach 12.8% and 16.0%, and the advantage of the forest appears on lacquer as well, where over the 17 months it was negative. None of these differences can be told apart from zero, however.

Without June and July, the two test months made mostly of fast pieces, the advantage of the forest over the family median falls to 7.8% (p 0.219), and so do those of linear regression, CatBoost and wood. The advantage over the standard rule rises instead, to 19.4%, because on pieces this fast the 21 days that the rule almost always assigns err less than the model.

The advantage also depends on when the hyperparameters are searched. The forest searches them every three months from March 2025, so October and November use those found in September. Rerun with the searches starting in October 2025, as a firm beginning to use it with a year of history would do, it reduces the error of the family median by 8.0% in the mean over months (p 0.138), and does better in nine of the ten.

5.4 How much the protocol matters

The choice of protocol reverses the verdict. With the leaking protocol, and the hyperparameters searched on it once during development, the model beats the family

median by 14.56%, an established result (p 0.031). With the honest one the advantage falls to 1.91% (p 0.835). The leaking protocol has 18 test months, one more, because without the condition on closure January 2025 also has enough cycles for training.

5.4.1 Training on the cycles still open

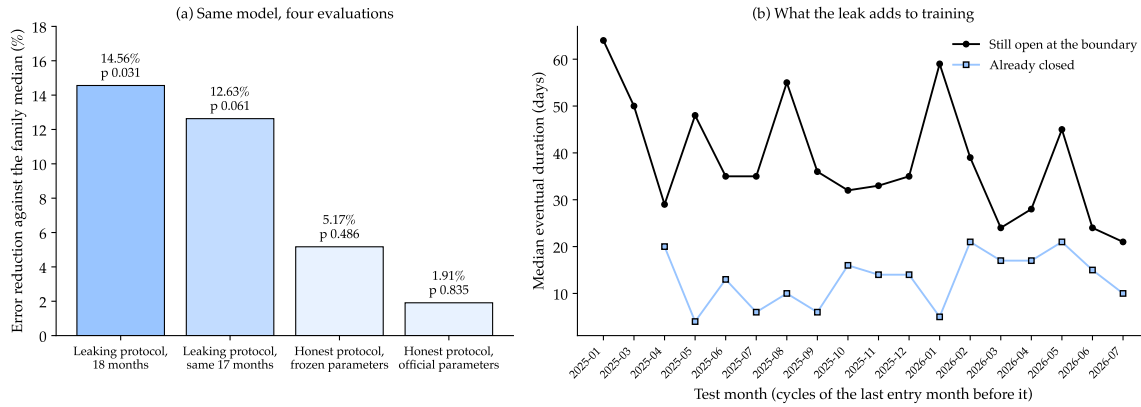


Figure 5.3: (a) The same comparison, the random forest against the family median, under the leaking and the honest protocol. (b) Median eventual duration of the cycles of the last entry month before each of the 18 test months of the leaking protocol, still open at the boundary or already closed.

To measure the effect of the protocol alone, the two have to be compared with the same hyperparameters. With those found during development, on the same 17 months, the leaking protocol gives an advantage of 12.63% (p 0.061) and the honest one of 5.17% (p 0.486), so the construction of the training data alone is worth 7.46 percentage points (Figure 5.3, panel a). With the hyperparameters used in the rest of the chapter, searched on the folds of the honest protocol, it is worth 8.54, because the advantage rises from 1.91% to 10.45% (p 0.172). In the direct comparison between the two versions, the leaking one errs 1.20 days less than the other, 8.0%, and does better in 10 months out of 17, but with p 0.163 the difference is not distinguishable from zero. The mechanism shows in the counts over the 18 folds of the leaking protocol. The cycles still open at the boundary (Section 5.1) are the longest, with a median eventual duration of 35 days against 14 for those already closed (panel b). Summed over those folds, the leaking protocol adds 14,480 rows out of 106,357 to training, 13.6%, and these are the very cycles that tell the model which pieces of the month just ended will be slow.

Regularised linear regression, with the hyperparameters found by its own search on the honest protocol and applied to both, shows no leak at all. Over the 17 common months its leaking version errs 0.05 days more (p 0.933), and even with its own

search on that protocol the advantage over the family median stays close to the honest one, 8.39% (p 0.070) against 7.85%.

None of the studies reviewed in Chapter 3 checks this aspect, and among those on recorded data only one splits in time, with a single cut that leaves the most recent 20% of the orders for testing (Section 3.7).

5.4.2 A single split in time

A *single split* in time also gives a different answer from the rolling origin. The model is trained once on the cycles that entered before the cut, and, as in the honest protocol, only on those already closed at that moment; it is then tested on the following months without being retrained. With the cut on 1 March 2026, the same hyperparameter search and testing on the 3,540 pieces that entered up to May, each with the same weight, the forest reduces the error of the family median by 13.0%. With a single test period, the uncertainty is measured by resampling the site-day groups of the quarter 2,000 times and recomputing the difference each time. Of the values obtained, 95% fall between 0.64 and 3.51 days, and on that quarter the advantage would count as established, as with the leaking protocol.

The same resampling gives the rolling-origin model an advantage of 1.1% on the same pieces, with an interval from -2.17 to $+2.80$ days. Part of the difference comes from the family median of the single split, which is frozen at February and errs more, 15.66 days against 15.25. Against the family median of the rolling origin, recomputed at every boundary from the cycles already closed, the forest of the single split would have an advantage of 10.6%, and the rest of the gap comes from the model. The resampling, however, describes only how much the pieces of that quarter vary. The monthly test used for the rolling origin can also be applied to the three months of the single split, and it does not tell the difference apart from zero. In the mean over months the advantage is 12.5% over the median of the single split (p 0.135) and 8.9% over the one recomputed at every boundary (p 0.397).

In March the two evaluations coincide, because the cut and the hyperparameters are the same, and the difference arises in April, when the pieces take 40.6 days on average. Retrained at the end of March, the rolling-origin model knows of that month's entries only those that have already come out, which are the shortest, and it predicts 23.4 days on average, with an error of 20.32. The model frozen at February predicts 32.9 days and errs by 15.08 in April. The order of the algorithms changes as well, since over the three months the forest errs least with the single split and linear regression with the rolling origin.

Extended to the entries up to July, 5,562 pieces each with the same weight, the model errs more with the single split than with the rolling origin, 14.72 days against 13.54, 8.7% more. From June the data contain mostly fast cycles (Section 5.3). In June the model frozen at February predicts 35 days on average for pieces that last 19, whereas with the rolling origin the forest, retrained every month on recent cycles that are fast as well, estimates shorter durations.

Against its own family median, also frozen at February and even less accurate (17.29 compared with 15.53), the forest of the single split keeps an advantage of 14.9% when weighting by piece, with an interval from 1.20 to 4.10 days that excludes zero. This time the monthly test also tells it apart from zero, with an advantage of 14.5% in the mean over the five months, present in each of them (p 0.016). With the rolling origin the model gains 12.8% over its own baseline when weighting by piece and 13.3% in the mean over months, with p 0.322. Against the same family median as the rolling origin, recomputed at every boundary, the advantage of the forest with the single split falls to 3.6% in the mean over months (p 0.561). Of its 2.38 days of advantage over its own baseline, 1.86 therefore come from the frozen median erring more.

In windows of three consecutive months, moreover, the advantage of the forest with the rolling origin over the family median ranges from -41.4% to $+26.1\%$ in the mean of each window (Figure 5.2), so a single quarter can give almost any result. A single split measures one period only, and its interval describes how much the pieces falling in it vary, leaving out the variation between months, which in these data is the largest. Tashman (2000) observes that forecasts made from a single origin are exposed to what happens only in that period. The rolling origin uses the same data, but repeats the cut at every boundary, so an atypical month weighs one in 17 instead of one in three.

The two measurements show that, with the same algorithm, the design of the evaluation can reverse the conclusion of a lead time study.

5.5 Segmentation

The third research question asks whether grouping the pieces before prediction, by domain criteria or with an unsupervised algorithm, improves on a single global model. None of the configurations tried improves on it distinguishably. The configurations of Table 1.2 were evaluated on the honest protocol, each with the hyperparameters found in the same fold for the single model A, so that the difference measures the grouping alone (Figure 5.4).

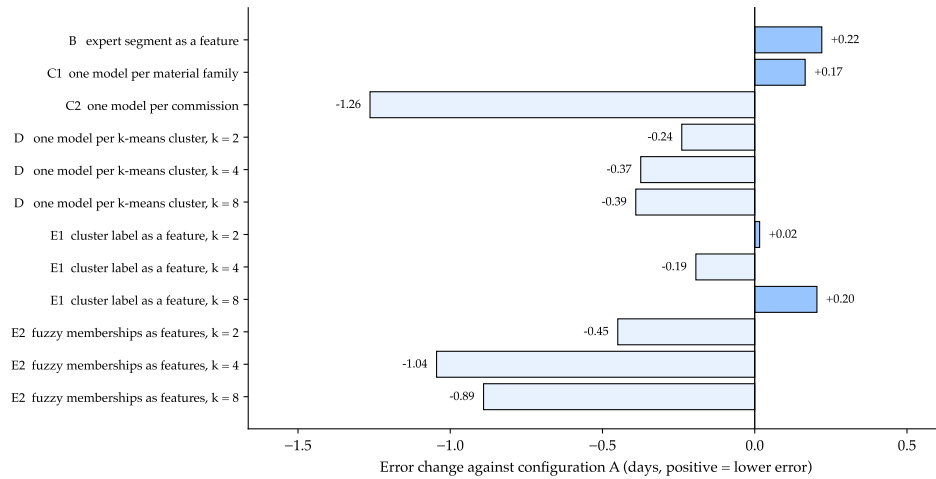


Figure 5.4: Change in mean absolute error of each segmentation configuration against the single global model A, mean over the 17 test months.

An expert segmentation, by functional family and material family, used as a feature (B) gains 0.22 days, and one model per material family (C1) 0.17. Separating the commissions (C2) costs 1.26 days, partly because in the months in which a commission has no closed cycles yet its model does not exist, and for 1,195 rows out of 12,614 the configuration falls back on A. The groups found by k -means, with one model each (D), lose between 0.24 and 0.39 days. The group label used as a feature (E1) oscillates around zero, and the membership degrees of a fuzzy clustering (E2) worsen the error by up to 1.04. Two p-values are nominally below 0.05, that of the gain of B (0.041) and that of E2 with four groups, which worsens by 1.04 days (0.012). With 12 configurations values like these can appear by chance, and the Holm correction raises them to 0.454 and 0.144.

The groupings found in the data are also weak as partitions. The silhouette of Section 2.5.4, measured on the training rows, lies between 0.28 and 0.35 for k -means and between 0.23 and 0.34 for the fuzzy clustering, and the pieces do not form clear groups in the feature space. The result is consistent with the reading of Section 5.2, based on bias and variance. With few independent observations, dividing the sample raises the variance more than each group adds in information.

5.6 Transferability across commissions

Trained only on the other commissions, the model does better than the family median on each of the three held-out new builds, and worse on the refit (Table 5.6). The widest gain is on NC-3, where the model errs 23.0% less than the median, 13.65 days against 17.73. This analysis uses the split of Section 2.4.4 that holds out one group

at a time, and here the groups are the commissions. Training includes all the cycles of the other commissions, from 67% to 94% of the table depending on which one is missing, and testing all those of the held-out commission; the hyperparameters are chosen with the same cross-validation on three blocks. The model therefore sees all the pieces of the training commissions, including those later than the test period, and measures how far what it learns transfers to a new commission.

The finish code, the most informative feature within a commission (Section 4.10), helps little here. It is defined commission by commission (Section 4.8), so the commission-code pair never appears in training. Even the code alone, taken without the commission, is missing from training for 46.9%, 100% and 68.8% of the rows of the three new builds. Cabin names are specific to each commission, and in 97-100% of the rows the cabin of the held-out commission does not appear in training. For this reason the cabins were also mapped to types common to all commissions, the cabin type of Table 4.9, which is not among the 13 features of the model (Section 5.1). In the held-out commission the cabin type almost always appears in training as well, in all the rows of NC-1, NC-2 and RF-1 and in 95.5% of those of NC-3. Across commissions the attributes defined in the same way everywhere remain comparable, such as the material family, the functional family and the cabin type.

On the refit the model loses, 7.05 days against 6.45 for the family median. RF-1 has 798 cycles, compressed timing and a much narrower distribution than the new builds (Table 4.8), and a compatible reading is that what the model learns on three new builds does not hold for a commission that works in a different way. The result is consistent with the distinction between the two types of commission of Chapter 1, and suggests training a model for refits on refit data.

Trained in the same way with its own hyperparameter search, regularised linear regression transfers worse. It errs more than the family median on NC-1 (14.35 days against 13.84), on NC-2 (19.50 and 18.45) and on the refit (7.25 and 6.45), and does better only on NC-3 (15.47 and 17.73).

Held out	Model	Family median	Standard rule	Segment median	Code never seen, without commission
NC-1	13.19	13.84	16.69	12.97	46.9%
NC-2	17.17	18.45	23.87	17.20	100.0%
NC-3	13.65	17.73	14.26	16.55	68.8%
RF-1	7.05	6.45	7.20	7.00	0.0%

Table 5.6: One commission held out at a time: mean absolute error of the model and of the baselines on the held-out commission (days), and share of its rows whose finish code, taken without the commission, never appears in training.

The colour class of Section 4.10, added to a model that uses only the material family, with one commission held out at a time, reduces the error in all four commissions, by 0.44 days on average (p 0.069). The cabin type, tried in the same way, increases it in all four, by 0.74 days on average (p 0.200). Added to the 13 features, the cabin type does not change the error distinguishably, 0.05 days more on average (p 0.860), worse on NC-1 and NC-2 and better on NC-3 and RF-1. With four commissions the paired test has little power, and these comparisons have to be read commission by commission.

5.7 The updated prediction

The fourth research question asks by how much the estimate improves if it is updated during processing, once part of the batch the piece travels with has returned. Updating the prediction for the pieces still in painting cuts the error of the family median by 27.9% (p 0.005, 15 months out of 17), and the result holds up under the Holm correction against all four baselines. It is the most solid result of the work, and a rule without learning achieves it too.

Only the pieces still in painting on the day the first piece of their entry document returns are updated, 4,844 of the 12,614 test rows, 38.4%. For the others the first return is their own, and there is nothing to update. The updated model is the prediction made at entry, raised to at least the days already elapsed. The rule without learning adds to the days already elapsed the median remaining time for the material family of the piece, measured on the training pieces that were still in painting when the first piece of their batch returned. All baselines are recomputed on the same population (Table 5.7 and Figure 5.5).

Baseline	Reduction	p	Holm p	Months won
Family median	27.86%	0.005	0.005	15 / 17
Standard rule	45.29%	0.0003	0.001	16 / 17
Segment median	29.96%	0.0006	0.002	16 / 17
Global median	30.23%	0.003	0.005	15 / 17

Table 5.7: The updated prediction on the pieces still in painting at the first return of their batch: reduction of the error against the baselines (mean over months), p-value, Holm p-value over the four baselines and months won. The comparison with the family median is one of the two main questions and is judged on the nominal p-value.

The rule without learning errs by 12.83 days and the updated model by 12.67, a difference of 0.16 that is not distinguishable from zero (p 0.893). Refitting a model

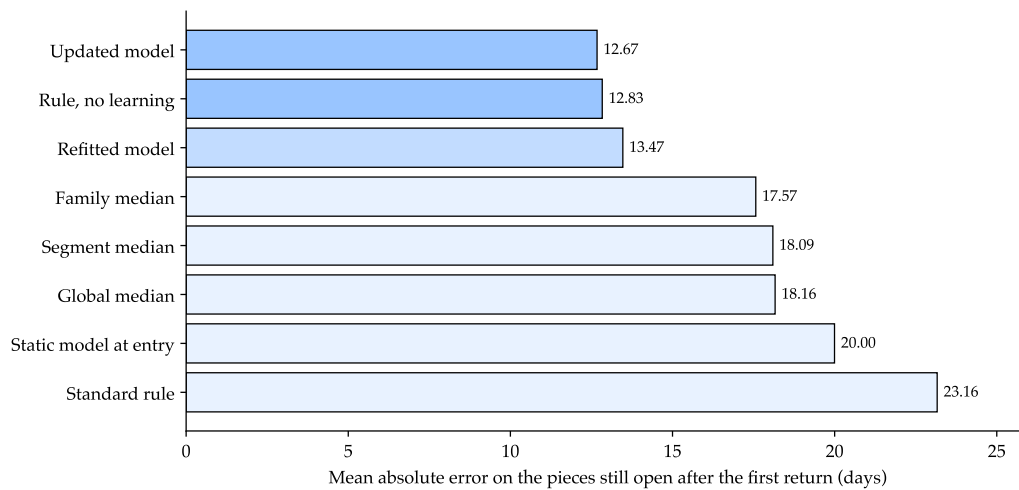


Figure 5.5: Mean absolute error on the pieces still in painting when the first pieces of their batch return: the updated model, the rule without learning, the refitted model, the static prediction made at entry and the baselines.

specifically on this population, with the elapsed days among the features, worsens the error by 0.79 relative to the updated prediction (p 0.344). The same prediction made at entry and left without update errs by 20.00 days, and the rule beats it by 35.8% (p 0.005). The gain therefore comes from knowing that the batch has started to come out and from the days already elapsed, and any rule that uses this information obtains it.

Nor does it depend on the protocol. With the leaking protocol the updated prediction gives 27.93%, almost the same figure, because at the moment of the update the decisive information has already been observed.

From October 2025, with at least a year of history in training (Section 5.3), the updated prediction reduces the error of the family median by 35.0% in the mean over months (p 0.005), doing better in each of the ten.

The other two algorithms do not change the result. Updated in the same way, the prediction of linear regression reduces the error of the family median by 25.71% (p 0.010) and that of CatBoost by 26.89% (p 0.004), against 27.86% for the forest, and neither can be told apart from the rule without learning (p 0.849 and 0.993). In the mean over months the forest has the lowest error, 12.67 days, against 12.84 for CatBoost and 13.05 for linear regression. Weighting each piece reverses the order, with linear regression at 11.00, CatBoost at 11.29 and the forest at 11.46, and in both averages the gaps between the three stay below 0.5 days. Neither of the two differences of the forest, from linear regression and from CatBoost, is distinguishable from zero (p 0.671 and 0.661).

5.8 Where the model errs

The model underestimates systematically, by eight days on average. Table 5.8 reports the error measures of Section 2.5 for the model and the baselines.

Of the four predictors, the model has the lowest absolute and squared errors, but it underestimates 45.4% of the pieces by more than a week. The standard rule underestimates even more, by 13.36 days, for the reason seen in Section 5.1. The family median, by contrast, overestimates.

The percentage error rewards the standard rule, because its short predictions err little in proportion on fast pieces, which is why Chapter 2 treats this measure with caution. The R^2 of each fold is negative for all predictors, baselines included. The mean of the month, which nobody can know in advance, beats every prediction, and the R^2 measures how much the level changes between months more than the quality of a model.

Figure 5.6 shows where the error concentrates. On the pieces that last up to 40 days the model errs less than the family median (12.12 and 19.21 between 0 and 20, 7.57

Measure	Model	Family median	Standard rule	Global median
MAE (d)	14.95	15.24	16.71	16.95
RMSE (d)	19.90	20.20	22.56	20.88
MAPE	57.0%	71.9%	51.8%	88.9%
Mean error (d)	-7.99	+2.92	-13.36	+6.57
Underestimated by over 7 d	45.4%	22.5%	54.6%	19.8%
Overestimated by over 7 d	20.5%	42.2%	9.1%	55.1%

Table 5.8: Error measures of the prediction at entry, mean over months. The mean error is prediction minus observed duration; the shares count the pieces underestimated or overestimated by more than seven days.

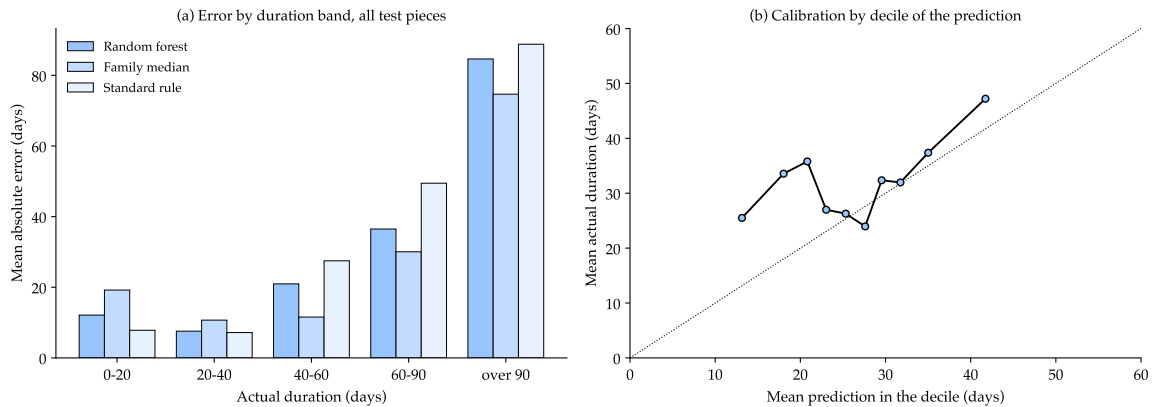


Figure 5.6: (a) Mean absolute error by band of actual duration, all test pieces. (b) Calibration: mean actual duration against mean prediction, by decile of the prediction.

Group	Rows	Months	Model	Family median	Standard rule	Mean error of the model
NC-1	4,524	15	13.70 (13.87)	13.05 (13.82)	14.89 (16.74)	−8.80 (−6.62)
NC-2	3,060	16	15.64 (15.89)	17.28 (16.46)	14.61 (17.04)	−0.88 (−8.45)
NC-3	4,232	7	22.03 (15.60)	25.51 (17.52)	22.64 (14.31)	−12.95 (−2.90)
RF-1	798	2	5.83 (5.89)	6.34 (6.45)	7.11 (7.20)	−1.27 (−1.37)
Wood	6,547	17	16.45 (15.91)	17.21 (17.79)	20.11 (18.12)	−9.85 (−7.00)
Lacquer	6,067	17	13.00 (12.85)	12.18 (12.49)	12.56 (12.45)	−6.12 (−3.85)

Table 5.9: Mean absolute error of the prediction at entry by commission and by material family: mean over the months in which the group has test rows and, in brackets, mean by piece. The mean error of the model is prediction minus observed duration.

and 10.70 between 20 and 40). Between 40 and 60 it errs more (20.96 and 11.57), and between 60 and 90 as well (36.49 and 30.04), because its predictions are compressed towards the centre. Over all the test pieces the standard deviation of the predictions is 0.425 times that of the durations (0.32 in the mean over months, the measure used in the following sections), the correlation between prediction and duration is 0.23, and the highest estimate is 55.5 days. No predictor estimates more than 60 days for a piece that takes longer, and the tail of the distribution is out of reach for all of them. The lowest decile of the predictions, which promises 13.1 days on average, gathers pieces that take 25.5.

In the groups of Table 5.9 the model errs less than the family median on NC-2, NC-3 and RF-1; on NC-1 the two are equivalent by piece, while in the mean over months the baseline does better. On wood the model does better than the baseline and on lacquer worse, as in the branches of Section 5.3. For these groups a description is enough, because RF-1 is tested in only two months and NC-3 in seven, too few for a paired test. On RF-1, in the months in which the commission is under test, the model errs less than the median (5.89 against 6.45 days by piece), whereas trained on the new builds alone it loses (Section 5.6). In the second of those months training already includes the first closed cycles of the refit. On NC-3 the two averages diverge, 22.03 days by month and 15.60 by piece, because some of its seven months have few pieces.

Rework cannot enter the features, because in 75.7% of cases it is recorded during the cycle or after it (Section 4.11). It can still indicate where the model errs, and the error turns out not to depend on it. The 670 test rows with a rework carry 4.98% of the absolute error of the model, and the cycles that contain one last a median of 27 days, against 28 for the others.

5.8.1 Why the model underestimates

The main cause of the underestimation is the selection by closure of Section 5.1, through which, in the last months before the boundary, the model sees only the fast pieces. A diagnosis with five variants shows it, all with the hyperparameters used in each month by the forest of Section 5.3, the *official model* from here on, so that the difference measures the variant alone (Table 5.10).

Removing from training the last two months of entries before the boundary, those in which only the fast pieces have closed, moves the mean error from -8.0 to -1.0 days. The absolute error falls by 1.48, a difference not distinguishable from zero (p 0.212). The gain comes from the level and the spread of the predictions. In the mean over months their standard deviation goes from 0.32 to 0.43 times that of the durations, while the correlation between prediction and duration within the month barely changes, from 0.23 to 0.26. Over all the test pieces together it rises from 0.23 to 0.36, a difference that comes from the level of the months, which the variant follows more closely. The error therefore falls because the predictions sit closer to the level of the durations and spread more around it, while the order of the pieces within a month stays about the same.

The logarithm of the target explains a smaller part of the underestimation. Brought back to days, a prediction made on the logarithm is close to the median duration, which with a distribution stretched to the right lies below the mean. The correction of Duan (1983) multiplies the prediction by the mean of the exponentials of the errors observed on the logarithmic scale. For the forest they are measured on a fifth of the training rows, held out of the fit, because on the data it has learned the algorithm errs very little. The correction moves the mean error from -7.99 to -7.10 days, and

Variant	MAE (d)	Mean error (d)	Against the official model (d)	p
Official model	14.95	-7.99		
Without the calendar features	14.51	-3.03	+0.44	0.688
With the Duan correction on the logarithm	14.86	-7.10	+0.09	0.369
Trained on days instead of the logarithm	14.75	-3.83	+0.20	0.640
Without the last month before the boundary	14.09	-4.48	+0.86	0.137
Without the last two months before the boundary	13.47	-0.95	+1.48	0.212

Table 5.10: Diagnosis of the underestimation: mean absolute error, mean error and difference in error from the official model (positive when the variant errs less), mean over months.

training directly on the duration reduces it to -3.83 , while neither variant changes the absolute error distinguishably.

Without the calendar the mean error shrinks by more than half, from -8.0 to -3.0 days. To see how the calendar shifts the prediction, each test piece was predicted again with the month and week of entry taken from a training row drawn at random, leaving the other features unchanged and averaging 20 draws. When the drawn row entered at least half a year before the boundary, in periods already almost entirely closed, the mean error goes from -7.43 to -1.72 days (p 0.009, 12 months out of 15) and the share of pieces underestimated by more than a week from 43.8% to 35.0% (p 0.018). The absolute error does not change distinguishably (p 0.793). This comparison uses the 15 months in which training rows that old exist.

When the drawn row instead entered in the two months before the boundary, of which only the fast pieces have closed, the mean error stays close to the actual one, -8.36 days against -7.99 (p 0.751). Taken from periods already closed, the calendar therefore removes 5.72 of the 7.43 days of underestimation.

The test months were then divided according to whether their month of the year had already appeared in the training data. It is the division of Section 5.3 between the first year of history and the months after it, decided after these results had been seen. The effect is concentrated in the first seven, from March to September 2025, when that month had never appeared. There the calendar of periods already closed reduces the underestimation by 11.40 days (p 0.036, in the five of those months in which rows that old exist). The calendar of the two months before the boundary lowers it too, by 2.56 days (p 0.045). From October 2025, when the same month had appeared a year earlier, the two substitutions shift it by 2.87 and -2.40 days, without the differences being distinguishable from zero. The underestimation that the calendar brings arises where the month of the test piece has no precedent in training.

The leak of the protocol and the underestimation are two sides of the same phenomenon. Training on the pieces still open reads their duration from the future, and discarding them leaves only the fast ones in view. A piece still open at the boundary carries legitimate information, however. It has been in painting for a known number of days, and its duration will be at least that; in survival analysis it is called a *right-censored* observation. An accelerated failure time model uses it without reading the future (Barnwal, Cho and Hocking, 2020), because in its likelihood a closed cycle counts through the density of its duration and an open one through the probability of lasting longer than the days already elapsed. Tried with XGBoost on the same 17 folds, adding to training the cycles open at each boundary, 13,563 in all, it does not improve the prediction (p 0.332). In the last month the prediction for a piece that took 17 days diverges, growing to values with no physical meaning, and on its own makes the mean over months unusable. The defect goes beyond that piece, because

the model errs more than the official one in 11 months out of 17, and even without the last two folds its error remains higher by 3.81 days ($p = 0.042$). The finish code of that piece appears in 35 training rows, 34 of them censored, and where almost all the observations of a group are open the likelihood pushes the prediction upwards without a counterweight.

5.8.2 Remedies for selection by closure

Correcting the selection by closure brings the mean underestimation close to zero, or halves it, while the absolute error does not fall distinguishably below that of the family median (Table 5.11). Eight remedies were evaluated on the same 17 folds, each with its own hyperparameter search. The main remedy trains the model only on the *mature months*, defined as those in which at least 90% of the cycles that entered have already closed at the boundary. The threshold was fixed before the results, by looking at the mechanism and not at the outcome, and on these data it excludes the last two months in 14 folds out of 17. The share was counted among the cycles that close by the extraction. Counting also the openings that never have a closing in the data, the rule would exclude further months in 16 folds out of 17. For this reason two fixed windows, without the last two or three months, are reported beside the main remedy, since they do not depend on that count.

The next three rows of the table drop the calendar features from the main remedy, weight each closed cycle by the inverse of the probability of observing its closing without them (Vock *et al.*, 2016), and cap the prediction of the accelerated failure time model of Section 5.8 at the longest observed duration.

Tried after the results of the others had been seen, the last two remedies in the table let the model know that at the boundary many recently entered cycles are still in painting. The first adds them to training, from which the honest protocol excludes them, with an estimated duration equal to the days already elapsed plus the median remaining time of the closed cycles of the same family that had lasted at least as long, taken from the mature months. It is the logic of the rule without learning of Section 5.7, applied to training. The second trains on the closed cycles alone, like the official model, and gives each row one more feature, the *observation window*: the time the cycle has had to close, in days from entry to the boundary, capped at 120. Every test piece receives 120, as if it had had all the time it needed to close, and the model can thus learn that rows with a short window are fast because, among the recent entries, only the quickest pieces have come out.

None of the eight remedies reads the future. Of each cycle they use only what is known at the boundary, its entry date, whether it had already closed and, if so, its duration. The first new remedy uses, of a cycle still in painting, only the days it

Model	MAE (d)	Mean error (d)	Under- estimated by over 7 d	Against the family median (d and %)	p	Holm p	Correlation
Official model	14.95	−7.99	45.4%	+0.29 (1.9%)	0.835		0.23
Mature months (main)	14.41	−0.72	31.7%	+0.83 (5.4%)	0.238		0.19
Without the last two months	14.16	−1.26	32.4%	+1.08 (7.1%)	0.117	0.885	0.19
Without the last three months	13.89	+2.55	23.6%	+1.35 (8.8%)	0.115	0.885	0.19
Mature months, without calendar	15.33	+2.44	26.8%	−0.09 (−0.6%)	0.933	0.933	0.23
Inverse weighting, without calendar	14.10	−1.64	32.4%	+1.14 (7.5%)	0.185	0.885	0.25
Accelerated failure time, capped prediction	20.42	−1.84	42.7%	−5.18 (−34.0%)	0.077	0.697	0.21
Open cycles with an estimated duration	14.29	+2.66	26.2%	+0.96 (6.3%)	0.170	0.885	0.19
Observation window as a feature	13.52	−4.32	38.4%	+1.72 (11.3%)	0.111	0.885	0.26

Table 5.11: Remedies for selection by closure, mean over months. The difference from the family median is positive when the remedy errs less, and the Holm p-value is computed within the family of nine comparisons, the eight remedies and the uncapped accelerated failure time model. The main remedy was fixed before the results and is judged on the nominal p-value; the last two were added after them.

has already spent there, and the second does not use the open cycles at all, because its window depends on the dates of entry and of the boundary alone. A check confirms it, since moving the exit dates later than the boundary at random leaves their predictions identical.

Trained on the mature months, the model has a mean error of −0.72 days, against −7.99 for the official one, and the share of pieces underestimated by more than a week falls from 45.4% to 31.7%. The mean absolute error goes from 14.95 to 14.41 days (p 0.642), and against the family median the advantage is 0.83, 5.4%, in 11 months out of 17, with p 0.238, a difference not distinguishable from zero. Computed on the mature months alone, the family median also raises its level but errs more (15.88 days against 15.24), whereas the forest trained on the same data has a lower error.

The first of the two new remedies overcorrects. The estimated durations exceed those the open cycles later had in two cases out of three, 48 days against 41 in the median. The model moves from an underestimation to a mean overestimation of 2.66 days. Against the official model its error falls from 14.95 to 14.29 days in the mean

over months, but counting the pieces it rises from 14.44 to 14.92. The observation window has the lowest error among the remedies, 13.52 days, 11.3% less than the family median (p 0.111). Compared with the official model it errs 1.43 days less and does better in 12 months out of 17, a nominally significant difference (p 0.029) in a comparison for which no Holm family had been fixed. With the same hyperparameters as the official model, which isolate the effect of the feature alone, the gain rises to 1.69 days (p 0.034, 0.069 with the Holm correction that includes it together with the other new remedy). Unlike the other remedies, it halves the underestimation, to -4.32 days.

For none of the eight remedies does the comparison with the family median hold up under the Holm correction, applied to the nine comparisons that also include the uncapped accelerated failure time model. The lowest corrected p-value of the family, against any baseline, is 0.088.

All the remedies widen the predictions. For almost all of them the standard deviation by month goes from 0.32 to between 0.45 and 0.55 times that of the durations (0.90 for the capped accelerated failure time model), and the correlation between prediction and duration stays between 0.19 and 0.25. The observation window is an exception, since it widens them less, to 0.38, and raises the correlation to 0.26, against 0.23 for the official model. Correcting the selection brings the level of the prediction near the right one, while the ability to rank the pieces, measured by the correlation, remains close to that of the official model.

For whoever uses the prediction the result has practical value. Trained only on periods already almost closed, the model keeps the same absolute error and loses the systematic underestimation, and a date computed in this way is late by more than a week for 31.7% of the pieces instead of 45.4%.

5.8.3 What the remedies depend on

The main remedy follows a principle similar to that of the single split of Section 5.4, which does not train on the most recent months, and applies it at every boundary. Removing those months raises the predictions, because only their fast pieces are visible, and the effect on the error depends on the month to be predicted. In April 2026, a slow month, the remedy errs by 14.41 days, by chance the same as its mean over the 17 months, against 15.08 for the forest with the single split and 20.32 for the official model. In March, when the forest of the single split coincides with the official model and keeps the fast pieces of January and February, the remedy that removes them overestimates and errs more (15.51 days against 13.75), and so it does in May, a month of short cycles (10.04 and 7.87). Over the 17 months the remedy beats the official model in 8 and loses in the other 9.

On the pieces of Section 5.4, each with the same weight, the remedy errs by 13.85 days from March to May, compared with 13.64 for the forest with the single split and 15.08 for the official model, and by 14.59 up to July, against 14.72 and 13.54. The two ways of not learning from the most recent months give almost the same error.

Without June and July, whose test pieces are mostly fast (Section 5.3), the main remedy reduces the error of the family median by 7.0% in the mean over months (p 0.150), against -1.36% for the official model, and errs 1.28 days less than the latter (p 0.287). Those two months favour the official model, which predicts shorter durations, through the same selection that penalises it in April.

The other two algorithms show the same picture without any remedy. Linear regression underestimates by 2.52 days on average, against 7.99 for the forest, and its pieces underestimated by more than a week are 33.7% instead of 45.4%. Its predictions are more dispersed, with a standard deviation of 0.48 times that of the durations (0.32 for the forest), but the correlation between prediction and duration is lower, 0.19 against 0.23. CatBoost sits in between, with -6.31 days and a correlation of 0.21. The lower error of linear regression at entry (Section 5.3) therefore comes from the level of its predictions more than from the order of the pieces, as for the remedies. In the updated prediction (Section 5.7) the floor at the days already elapsed applies more often for the forest, which underestimates more from the start, in 51.7% of the pieces to be updated against 36.8% for linear regression and 41.2% for CatBoost in the mean over months. On the rows concerned, however, the forest and linear regression differ by only 0.19 days (p 0.812).

On the other two algorithms the new remedies help less. Linear regression worsens with both, from 14.04 to 14.95 and 14.52 days, and CatBoost with the first, from 14.31 to 14.81. With the observation window CatBoost instead falls to 13.54 days, 11.2% less than the family median (p 0.050, 0.200 with the Holm correction over the four comparisons of the two algorithms).

The hyperparameter search alone can move a verdict over 17 months. The variant without the last two months in Table 5.10 beat the family median by 1.77 days with the hyperparameters of the official model (p 0.014, 0.070 with Holm) and by 1.08 with its own search (p 0.117). The two new remedies behave in the same way. Computed with the hyperparameters of the official model in each fold instead of their own search, the open cycles with an estimated duration beat the family median by 8.3% (p 0.021, 15 months out of 17) and the observation window by 13.0% (p 0.031, 13 out of 17), against 6.3% and 11.3% in the table, and for both the p-value falls below 0.05. The comparisons in the table, each remedy with its own search as had been fixed in advance, remain the main ones; those with the hyperparameters of the official model were decided after the results had been seen and are reported only to show this effect.

5.9 What the model uses

The finish is the information that weighs most in the model, and it is the only group of features whose removal raises the error with a nominally significant difference. Three measures from Section 2.5.5 answer different questions and have to be read together (Figure 5.7). Permutation importance measures how much the error on the test months rises when a feature loses its link with the piece. The most useful features are the material family (0.28 days), the functional family (0.16), the finish code (0.15), the concurrent load (0.12) and the week of entry (0.11). Permuting the cabin, the weekday and the commission lowers the error instead; the model uses them, but on new months they lead it astray. The importance of the month of entry is zero by construction, because each test fold contains only one.

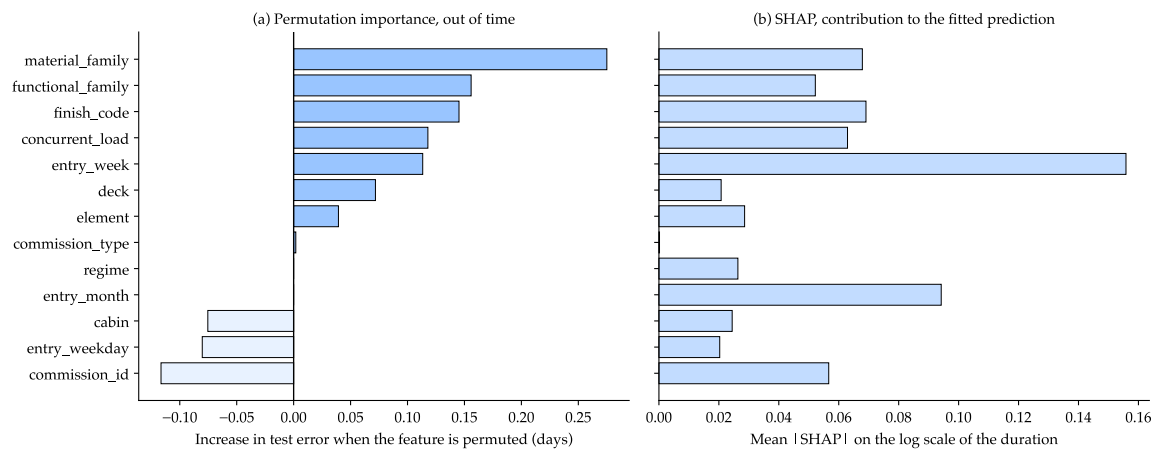


Figure 5.7: (a) Permutation importance on the test months: increase in mean absolute error when a feature is permuted. (b) Mean absolute SHAP value on the log scale of the duration, over the test rows.

Ablation removes a group of features and retrains with the same hyperparameters. Without the finish group, finish code and material family, the error rises by 0.60 days, the only deterioration visible before the correction (p 0.029, 0.146 with Holm). Without the commission the error falls by 0.61 days (p 0.069), without the calendar by 0.44 (p 0.688), and the piece features and the department feature change nothing distinguishably.

SHAP values describe instead how the model, as fitted, builds each prediction. They put the week and month of entry at the top, then the finish code, the material family and the load. The model relies on the calendar because in the training data it explains the level of the period. On new months that level shifts and the calendar no longer helps, so the three measures are compatible with one another. It is the same conclusion as the ablation, seen from inside the model.

Unlike the forest, regularised linear regression makes little use of the calendar. The mean SHAP value of the week of entry is 0.006, against 0.156 for the forest, and at the top are the finish code (0.121), the element (0.101) and the cabin (0.079). To check whether the leak of Section 5.4, absent for linear regression, passes through the calendar, the forest was measured again on the two protocols without month, week and weekday of entry, with the same hyperparameters. The leaking version errs 0.34 days less than the honest one (p 0.689), against 1.20 with the calendar (p 0.163). Neither difference is distinguishable from zero, and the test does not support the idea that the leak passes through the calendar.

This leaves the second research question, which asks what information determines the duration of finishing and whether the recorded data observe it for the individual piece. Taken together, the three measures answer its first part. Among the recorded information, the one that determines the duration is the finish, through the code within a commission and the material family across commissions.

5.10 From prediction to the date to promise

In Section 2.6 the date to promise is a quantile of the distribution of the duration. Here the date is the prediction plus a margin, equal to a quantile of the errors of the previous months computed only on the cycles already closed at the boundary (Table 5.12).

Rule	Late	Mean delay (d)	Mean earliness (d)
Standard rule	66.8%	20.1	6.3
Family median	35.4%	16.8	14.1
Model, margin at 50%	47.0%	17.2	14.2
Model, margin at 60%	34.2%	18.0	15.9
Model, margin at 80%	20.7%	16.8	23.7
Model, margin at 90%	14.8%	17.7	27.9

Table 5.12: The promised date: share of late pieces, mean delay among the late pieces and mean earliness among the others, by piece. The margin of the model is the stated quantile of the errors of the previous months.

With the standard rule 66.8% of the pieces are late, by 20 days on average. Using the family median as the date halves the late pieces, to 35.4%, with a mean earliness of 14 days. With a margin at 60% the model reaches the same share of late pieces as the median and a similar earliness, 15.9 days. Wider margins buy fewer delays with more days of waiting, a trade-off that depends on how much a late piece costs

compared with one ready too early. The practical gain at entry comes mostly from moving from a constant to a family median, which halves the delays.

In the ten months with at least a year of history (Section 5.3) the model pulls ahead, leaving 26.4% of the pieces late with a margin at 60%, 8.3 percentage points fewer than the family median (34.7%), while the standard rule leaves 64.6%.

Prediction intervals built from the same errors, measured on the logarithm of the duration, cover less than they declare. At 80% they contain the true duration in 76-79% of cases and at 90% in 83-86%, with a mean width of about 47 and 64 days. Built on the errors in days, those at 80% fall to 70.4% on wood. The errors of the past do not describe the following period well, because the level shifts, and it is the same reason why the R^2 of each month is negative. The first test month has no intervals, because at its boundary no error could yet be observed.

With regularised linear regression the same intervals cover almost what they declare, 81-82% of cases at 80% and 91-92% at 90%, with a larger mean width, about 50-54 and 76-81 days.

5.11 Other interventions tried

The interventions of this section, tried to improve the model at entry, gave no distinguishable gain. They include the candidates left out of the 13 features (Section 5.1), added one group at a time; the reverse path, which removes the groups one by one, is the ablation of Section 5.9. Those in Table 5.13 were measured during development, on the version of the dataset that preceded the corrections of Section 4.5, with 13,384 cycles, and with the leaking protocol, each in its own run. Under those conditions the advantage of the model was inflated, so an intervention that brought no improvement even then is unlikely to help on the honest protocol. Those in Table 5.14 were measured with the official model on the honest protocol, because in development they had no recorded figure or had one built in a flawed way.

The requested date changes nothing, because in 80.8% of cases it is 21 days (Section 4.11). The simple average gives the same weight to the predictions of the forest, XGBoost and LightGBM. The blend combines the prediction of the model with the family median, with a weight that is constant or depends on the load of the month, estimated on the earlier months only. It brings the predictions closer to that median by construction, and the small gain it shows cannot be told apart from chance. Isotonic calibration corrects each prediction with an increasing function, estimated within training on the predictions made for rows held out of the fit. It

Intervention	Difference (d)	p	Outcome
Requested date	−0.07 to −0.27	0.40 to 0.63	no effect
Simple average of the algorithms	−0.18	0.525	no gain
Blend with a baseline, constant or load-dependent	+0.20; +0.37	0.527; 0.233	not distinguishable
Isotonic calibration	−2.19	0.021	worse
Median of a quantile forest	−0.62	0.143	not distinguishable
Weighting by batch	−0.18	0.696	no effect
Weighting by recency, half-life from 3 to 24 months	−0.04 to −0.21	0.075 to 0.645	no effect
Drift correction, three forms	−0.54; −2.49; −0.92	0.306; 0.193; 0.193	not distinguishable
Two-stage prediction	−0.41 to −1.60	0.113 to 0.562	not distinguishable

Table 5.13: Interventions tried during development: difference in mean absolute error from the model without the intervention, positive when the intervention errs less, mean over months and p-value of the paired test. Dataset before the corrections of Section 4.5 (13,384 cycles), leaking protocol, 18 test months, random forest.

worsens the absolute error by 2.19 days (p 0.021) and turns the underestimation into an overestimation. The mean error goes from −1.21 to +5.80 days.

In the quantile forest the prediction is the median of the training durations that fall in the same leaves as the piece, instead of their mean. Weighting by batch gives each row a weight inverse to the size of its document, so that 40 pieces that entered together count as much as one piece on its own. Weighting by recency halves the importance of a cycle every 3, 6, 12 or 24 months of age. The drift corrections remove from the duration the trend of the monthly level estimated on the training data and put it back into the prediction. They extend it to the test month, hold it at the level of the last period observed, or estimate it only on the months far enough from the boundary to have closed. The two-stage prediction, which first predicts the document and then the deviation of the piece from the duration of the document, errs more in every version tried.

Intervention	Difference (d)	p	Holm p	Outcome
Document features	+0.48	0.478	0.478	not distinguishable
Shop-state features	+0.97	0.136	0.273	not distinguishable
Supplier and project phase	−2.54	0.003	0.008	worse
A quarter, half, three quarters of the training rows	−0.15; −0.34; −0.23	0.833; 0.464; 0.362		no effect

Table 5.14: Interventions measured on the honest protocol: difference in mean absolute error from the official model, positive when the intervention errs less, mean over months, with the p-value of the paired test and the one corrected over the three feature groups.

The document features describe the batch the piece enters with (the pieces it carries, the share of wood, the number of distinct elements, cabins and codes), counted over all the pieces entered up to that moment, including those that will never close, and they do not improve the model (+0.48 days, Holm p 0.478). The shop-state features (queue, arrivals, exits and recent durations) reduce the error by 0.97 days without the difference being distinguishable from chance (Holm p 0.273), and the reduction comes from the level of the predictions, because the correlation between prediction and duration falls from 0.23 to 0.18. Supplier and project phase worsen the error by 2.54 days, an established result (Holm p 0.008), as they already did in development. Training the forest on a quarter, half or three quarters of the rows does not change its error distinguishably, so for this algorithm the limit is not the quantity of data. For regularised linear regression the curve rises instead: with a quarter of the rows it errs 2.17 days more (p 0.006), with half 0.86 (p 0.029), with three quarters 0.39 (p 0.023). For a simpler model the quantity of data matters, even though with all the rows linear regression cannot be told apart from the forest (Section 5.2).

During development, where it was adopted, the logarithm of the target lowered the error by 1.06 days compared with training on the raw duration (p 0.069). On the honest protocol the difference does not appear, 0.20 in favour of the raw duration with p 0.640 (Table 5.10). The grouping by day and site inside the search is a design choice instead, motivated in Section 5.1, with no figure that measures it.

5.12 The limit of batch-level recording

For three pieces out of four, the recorded exit date is the day the batch left. Chapter 1 describes how the pieces of a document enter painting together on a rack, are processed one by one with their own finishes, and how the rack leaves when all or nearly all of them are finished (Section 1.2.9). What the data record is mostly this exit.

Of all pieces, 75.8% leave painting on the day on which most of their document comes out. Their distance from the median exit date of the document has a standard deviation of 1.48 days, and within each document there is no variability to explain, because all these pieces leave together. For the remaining 24.2%, the pieces recorded separately, the same distance has a standard deviation of 19.11 days, and there the piece becomes visible. Within the document the finish code explains 7.29% of the variability of that distance and the element 5.56%, against a share expected by chance of 0.95% for the code. These shares are descriptive and computed on a population selected for having deviated, but they show that where the recording lets the piece through, piece-level information is there.

The ladder of oracles in Figure 5.8 measures how much the information the model lacks is worth. It is computed on the 18 test months of the leaking protocol, because an oracle is not trained and can also evaluate January 2025. A predictor that knew in advance the level of each month would err by 11.93 days. By month and family it would err by 10.55, by site and week by 10.14, and with the exit date of its own batch by 2.59. Between the week and the batch lie 7.55 days, which presumably correspond to the decision on which batch to process and when, taken in the department after entry and not recorded.

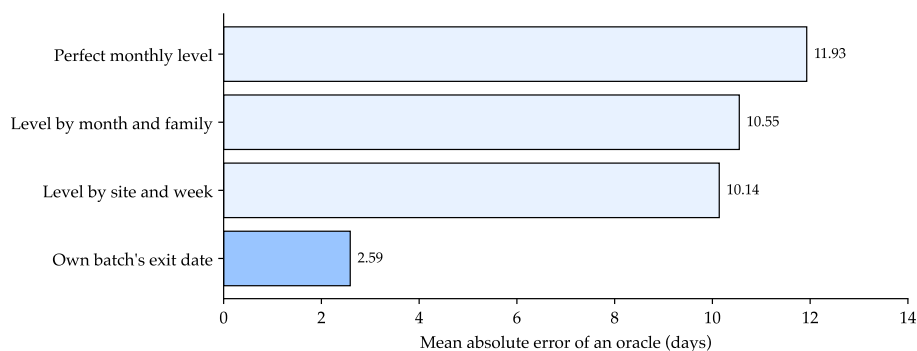


Figure 5.8: Mean absolute error of four oracles, each knowing in advance information not available at prediction time.

From this follows the conclusion that ties the chapter together and answers the part of the second research question on recording. At entry, the quantity to be predicted is mostly the moment at which the batch will be processed and sent out. Of the 10.14 days of error left to whoever knows site and week, 7.55 presumably correspond to that decision, which the data do not contain, and within the batch the duration of the individual piece is not recorded for three pieces out of four. No algorithm can learn what is not recorded, and the results of the previous sections are consistent with this limit. As soon as the batch starts to come out, by contrast, the information arrives and the error falls by 28%. The value lies in the recording, and Chapter 6 draws the implications for a firm in this position.

Chapter 6

Limitations and Future Developments

6.1 Limitations

Some limitations of the work depend only on the nature of the data, and no choice of method could have avoided them. Others concern the model and the way it was evaluated.

6.1.1 Limitations of the data

As described in Section 1.2.9, in painting the firm's production management portal records the entry and the exit of the document the batch travels with, and no event marks when work on an individual piece starts or ends, so its processing is never observed. For three pieces out of four, moreover, the recorded exit date is that of the batch (Section 5.12), and the duration of the piece is in effect that of its document. The information that tells a piece apart from the others, such as its name, the element it belongs to and its finish code, is flattened onto the common time of the document. Within a batch it produces no difference in duration, and a model can use it only to tell one document from another.

The independent observations are therefore far fewer than the 14,245 cycles. The pieces of a document enter and leave together, and there are 939 entry documents. Grouped by site and day of entry, the unit the evaluation keeps together (Section 5.1), the cycles form 420 groups of 34 pieces on average. Durations within a group are very similar, and the intraclass correlation, the share of the variance of the duration due to differences between groups, is 0.656 by site and day and 0.833 by document. For estimating a mean, the 14,245 cycles grouped in this way are worth about 630 independent observations, or 1,110 if the group is the document. A sample of this size weighs on the whole work. A reading consistent with the results is that the more flexible algorithms pay for their variance, to the point that regularised linear regression matches the error of the tree ensembles (Section 5.2), and that splitting the pieces into groups takes from each model more observations than it gives infor-

mation (Section 5.5). Categories with few pieces, such as many finish codes, rest on an even smaller number of batches.

The evaluation takes this dependence into account without being able to remove it. In the hyperparameter search the pieces that entered a site on the same day stay in a single block, so validation contains almost no copies of pieces already seen in training (Section 5.1). The test between months works on 17 monthly errors, and the number of pieces does not inflate its confidence. The dependence, however, lies in the data. The durations of pieces that left together remain linked under any evaluation design, and the model itself is trained on the pieces as if they were independent. It follows that only a different way of recording would reduce it (Section 6.2.1).

In the Engineer-to-Order regime of luxury yachting (Section 1.2) every vessel is a one-of-a-kind project, and the workload alternates quiet periods with phases of saturation driven by the pre-season launch calendar. For an activity of this kind, 23 months of history are few. Each calendar month appears at most twice, and with fewer than two annual cycles a seasonal pattern cannot be separated from the other differences between one year and the next. In the first seven test months, from March to September 2025, the month of the year being predicted had never appeared in training. The calendar features therefore learn the level of few periods, and their effect on the underestimation is concentrated precisely in those months (Section 5.8). Over the following ten, the first with at least a year of history behind them, the model reduces the error of the family median by 11.7% in the mean over months (p 0.076), against 1.9% over the whole test, and does better in eight of them. For the updated prediction the reduction rises from 27.9% to 35.0% (p 0.005, Section 5.3, Section 5.7). This reading was decided after the results and is descriptive, and with the hyperparameter searches starting in October 2025 the advantage at entry falls to 8.0% (p 0.138).

With only 17 test months, the evaluation shows how much the error changes from one month to the next, and cannot tell apart differences of a day or two. On the honest protocol, the forest trained without the last two months before the boundary errs 1.48 days less than the 14.95 of the official model, and the difference cannot be told apart from chance (p 0.212, Section 5.8).

Commissions, too, are few and different from one another. The data come from a single firm and four commissions, three new builds and a refit. The refit has 798 cycles and is under test in only two months, and trained on the new builds the model errs more on it than the family median (Section 5.6). With a single refit it cannot be established whether the difference depends on the type of commission or on that commission in particular.

Master data follow no standard common to the commissions. The finish code is defined commission by commission, and the commissions share few codes. None of

the 24 codes of NC-2 appears elsewhere, and the same code can denote a different finish in two commissions (Section 4.10). The textual description of the finish is filled on 51.19% of the physical pieces and does not lend itself to automatic rules, which is why the dictionary that assigns a material family and a colour class to each of the 197 commission-code pairs was compiled by hand (Section 4.8). The gloss class and the surface effect of the finish, such as brushed or striped, can be derived for only a small part of the codes, and the former is missing precisely for the most frequent lacquers, while the weight and dimensions of the piece are fields of the exports filled too rarely to be used (Section 4.12). The names of elements and pieces follow no fixed scheme, those of cabins are specific to each commission, and the data contain no assembly grouping the pieces of the same item of furniture (Section 1.2.2). A model therefore learns the meaning of a code or a name only within the commission that uses it.

Over time, moreover, the data change a great deal, which makes the task of any model that learns from the past harder. In the three new builds the decks enter painting months apart, and new cabins and elements appear until the end of the observed period. On the monthly folds, the share of test rows with at least one element, cabin or finish code never seen in training ranges from 31.3% in NC-1 to 69.4% in RF-1 (Section 4.10). The level of the durations shifts too, and the monthly error of the model ranges from 7.9 to 25.2 days (Section 5.3). The mean of the month, which nobody knows in advance, has a lower squared error than any prediction, and for this reason the R^2 is negative in every fold and for every predictor (Section 5.8). In mid-June 2026 the vocabulary of the portal changed. Since then the phase says only whether a movement is a dispatch, a receipt or the final shipment, and the cycle is rebuilt from the direction of the movement and the department, with rules different from those used before (Section 4.3, Section 4.5). The weeks recorded in the new way are few and fall at the end of the observed period, where the long cycles are still open, so any effect of the change on the durations would be confounded with their absence. Close to the extraction, cycles that entered in the last months appear only if they have already closed, and on 1 September 384 of the July entries and 76 of the August ones were still open (Section 4.6).

Some information on which the duration may depend appears in a form that cannot be used, or is not recorded at all. The requested date is not a frozen field. For 49.6% of the pieces that carry it, it changes from one movement row to another, by a median of 48 days, and where it is present it falls three weeks after entry in 80.8% of cases (Section 4.11). Rework is recorded during the cycle or after its closing in three cases out of four, and in this thesis it served only for the analysis of the errors (Section 5.8). The concurrent load counts the pieces of the four commissions, while the department also works on pieces of other commissions (Section 1.2), which do not appear in the exports. Finally, the painting station, the operator, the capacity of the department

and the moment at which it is decided which batch to process are not present in the data.

Upholstery, the other finishing operation, cannot be measured with these data. The dispatch of leather pieces to the upholsterers is tracked to very different extents from one commission to another, from 1.5% in NC-3 to 90.5% in RF-1, and the return almost always appears as a receipt from transport, without the name of the supplier (Section 4.7.1).

6.1.2 Limitations of the model and of the evaluation

At entry, none of the interventions tried improved the model in an established way. Eight algorithms were compared, with errors between 14.04 and 15.39 days if the neural network is excluded, together with 12 segmentation configurations. The interventions tried also include the requested date, the document and shop-state features, the supplier and the project phase, the average of several algorithms, the blend with the family median, isotonic calibration, the quantile forest, weighting by batch or by recency, the drift corrections, the two-stage prediction, the remedies for selection by closure and a model for censored data (Section 5.8.2, Section 5.11). Some worsen the error. Among those that reduce it is a remedy tried after the results, the observation window, a feature that tells the model how many days each cycle has had to close before the boundary. With this feature the forest errs 1.43 days less than the official model, a nominally significant difference (p 0.029) in a comparison for which no Holm family had been fixed (Section 5.8.2). Outcomes this similar, obtained with such different methods, indicate that the limit lies in the information available at entry more than in the algorithm. According to the oracles, even a predictor that knew in advance the mean duration of each site and week would still err by 10.14 days, and only by knowing the exit of its own batch would it fall to 2.59 (Section 5.12).

The honest protocol does not read the future, and it has a cost. In the last months before the boundary training sees only the fast pieces as closed, and the model underestimates by eight days on average. The remedies of Section 5.8.2 bring the predictions, on average, close to the observed durations, with a mean error of -0.72 days for the main remedy against -7.99 . None of them, however, errs less than the family median in an established way, and in the mean over months the correlation between prediction and duration stays between 0.19 and 0.26, while that of the official model is 0.23. They correct the level of the predictions more than the order of the pieces. Giving the cycles still open an estimated duration overcorrects, because the estimate exceeds the true one in two cases out of three. The model for censored data, which would use those cycles without reading their duration, breaks down

instead where almost all the observations of a group are open. The cost arises from the little time the recent cycles have had to close, and as long as training uses them only once they have closed, the slow pieces of the months just ended stay out of it.

As a result, the predictions are compressed towards the centre. In the mean over months their standard deviation is 0.32 times that of the durations, and the highest estimate is 55.5 days. No predictor estimates more than 60 days for a piece that takes longer. The lowest decile of the predictions promises 13.1 days on average to pieces that take 25.5 (Section 5.8). The model distinguishes fast pieces from slow ones poorly, and this limits any use that requires ranking them, such as choosing the sequence of processing.

Uncertainty, too, is described only in part. Intervals built from the errors of the previous months cover less than they declare, 76-79% of cases for those at 80% and 83-86% for those at 90%, because the level shifts from one month to the next (Section 5.10). On wood, when built on the errors in days, those at 80% fall to 70.4%. With regularised linear regression they cover almost what they declare, and are wider. No interval was measured for the updated prediction.

The updated prediction, the most solid result, concerns 38.4% of the test pieces, those still in painting when the first piece of their batch returns (Section 5.7). For the others the first return is their own, and there is nothing left to update. A rule without learning obtains the same gain, because it comes from knowing that the batch has started to come out and from the days already elapsed. The estimate also changes only once, and what happens afterwards, such as a second group of pieces returning, does not modify it.

A known effect concerns target encoding. To use categorical features with many levels, such as the cabin, the element and the finish code, the model replaces each category with the mean duration of the training pieces that carry it, drawn towards that of the whole set when those pieces are few (Section 2.2.1). The same mean is then used to train the model on the same rows, and for a training piece the encoded value already contains its own duration and that of the pieces of its batch. The model may therefore rely on these features more than their value on new pieces warrants. Prokhorenkova *et al.* (2018) describe this shift between training and use and avoid it by computing the statistic of each row only on the rows that precede it in a random order. The evaluation remains correct, because no information from the test months enters the encoding, and the size of the effect appears small. CatBoost, when it receives the categories as they are and applies this device, errs by 14.40 days, against 14.31 with the common encoding (p 0.834, Section 5.2).

Finally, the evaluation has limitations, declared where they arise. The algorithm, the 13 features and the logarithm of the target were fixed during development, with the leaking protocol, on the same months later used for testing. The choices made after

seeing the results are declared. The Holm correction and the hyperparameter search on the folds of the honest protocol go in the more cautious direction, because they make it harder to call an advantage established. The two remedies tried after the results are judged within the Holm family fixed beforehand, which becomes stricter with them, and the reading of the months from October 2025 is descriptive. Over 17 months the hyperparameter search alone moves verdicts close to the threshold, as two comparisons decided after the results show. With the hyperparameters of the official model in every fold, instead of searching their own, the open cycles with an estimated duration reduce the error of the family median by 8.3% (p 0.021) and the observation window by 13.0% (p 0.031), against 6.3% and 11.3% in Table 5.11. The p -values may also be smaller than they should be, because neighbouring months share most of their training data (Section 5.1). The leak of the protocol that reads the future is itself measured with uncertainty. In the direct comparison between the two versions of the forest the difference, 1.20 days, cannot be told apart from zero (p 0.163), and with regularised linear regression the leak does not appear (Section 5.4). The answer also depends on the design of the evaluation. On one quarter, with the single split, the advantage of the forest over the family median, 13.0%, looks established, while on the same pieces with the rolling origin it is 1.1%. Part of the difference comes from the baseline of the single split, frozen at February, which errs more. With the test extended to July, the model errs more with the single split than with the rolling origin, 14.72 days against 13.54 (Section 5.4).

6.2 Future developments

The limitations of Section 6.1 suggest the developments that follow, and Table 6.1 links them. The oracles of Section 5.12 indicate that the largest gain would come from the data, and the developments therefore begin with recording.

Limitation	Where it is measured	Development
Duration of the piece not observed, batch-level recording	Section 5.12	Recording the processing of the piece (Section 6.2.1)
Few independent observations	Section 6.1.1	Recording the processing of the piece
Department capacity and batch decision not recorded	Section 4.2.1, Section 5.12	Recording the processing of the piece
Master data not harmonised, unseen categories	Section 4.10	Harmonising the master data (Section 6.2.2)
Requested date not frozen, upholstery not measurable	Section 4.11, Section 4.7.1	Recording and master data
23 months, four commissions, a single refit	Section 5.3, Section 5.6	More history (Section 6.2.3)
Underestimation from selection by closure	Section 5.8, Section 5.8.2	Observation window, models for censored data (Section 6.2.4)
Intervals covering less than declared	Section 5.10	Quantile loss, margin on recent months
Updated prediction revised only once	Section 5.7	Remaining time updated at every event
Transfer to the refit	Section 5.6	Starting from other commissions, refit data
Weak ranking of the pieces	Section 5.8	Processing sequence after piece-level recording

Table 6.1: Limitations of the work and the developments that address them.

6.2.1 Recording the processing of the piece

A firm that records movements at batch level could have the label of the piece, which already carries a unique optically readable code (Section 1.2.9), scanned at two moments: when the operator takes the piece from the rack to process it, and when it is put back once finished. A convenient reader near the racks, fixed or handheld, would be enough. The entry and the exit of the batch would still be recorded with the document, as they are today, and the two scans would be added to the existing movements (Figure 6.1).

With the two scans, the duration of each piece splits into the wait between the entry of the batch and the pick-up, the processing up to its return, and the time spent waiting for the other pieces until the exit. Waiting is thus separated from processing, two components that the definition of Chapter 1 cannot tell apart, and the quantity to be predicted can become the duration of the individual piece. The information that the batch flattens today would begin to count. Where the recording already lets the

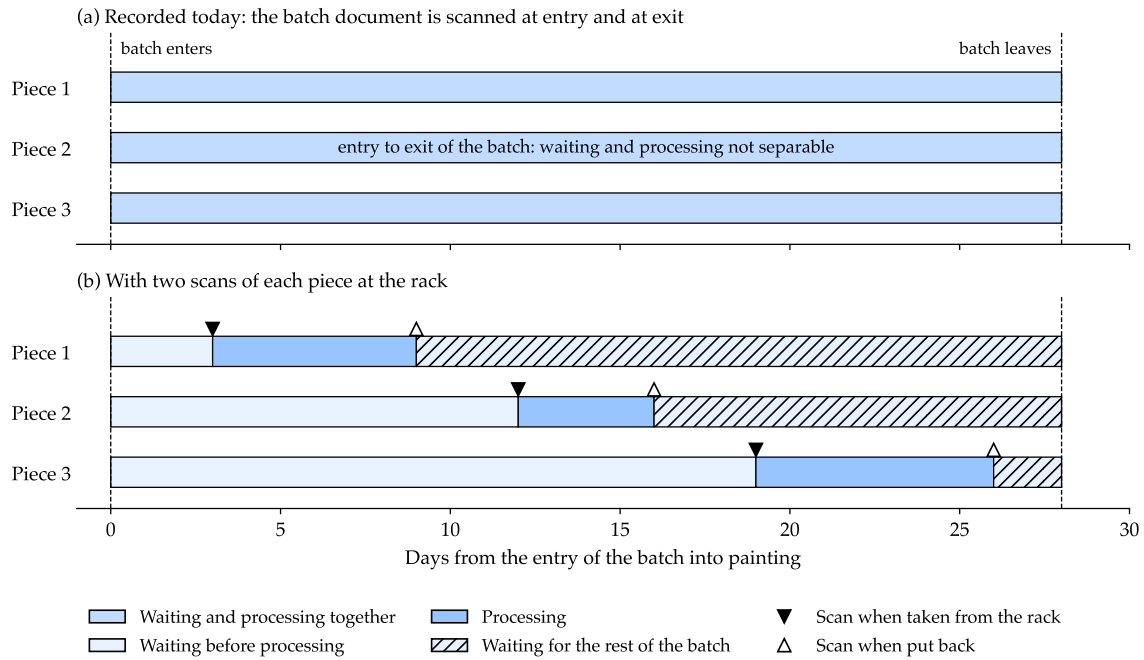


Figure 6.1: Illustrative timeline of three pieces of the same batch. (a) Today only the entry and the exit of the batch document are recorded, and every piece gets the same duration. (b) Two scans of each piece at the rack separate the waiting before processing, the processing and the waiting for the rest of the batch. Invented times.

piece through, the finish code explains 7.29% of the variability of its exit within the document, against 0.95% expected by chance, and the element 5.56% (Section 5.12). Each piece would carry its own duration, and the independent information would approach the number of pieces rather than that of the 939 documents. Pieces processed on the same days would still remain linked, because they wait in the same department.

The department would also become measurable as a queue. Each day, the pieces waiting on the racks and those arriving would be counted. Little's law (Little, 1961) links these quantities: in a stable system the average number of pieces in the queue equals the arrival rate multiplied by the mean wait. From the number of pieces taken and returned each day an estimate of the capacity of the department, missing today, could be derived. The indirect route of Section 2.6, ruled out in Section 4.2.1 for this very lack, would then potentially become viable, and a queueing model could estimate the wait of a piece from how many precede it.

How much this information is worth is already measured. The difference between the two oracles of Section 6.1.2, 7.55 days, presumably corresponds to the decision of the department on which batch to process and when (Section 5.12). The two scans would make this decision observable after the fact, through the order in which the pieces are taken, and with a few months of data a model could learn which pieces the

department tends to process first, for example by finish, requested date or expediting requests.

The main risk of such a system lies in the errors of manual recording described in Section 3.3, scans skipped or postponed to the end of the shift. Each scan must therefore cost the operator very little, and the system can flag inconsistent cases, such as a piece returned without having been taken or a batch leaving with pieces never scanned. The same scheme, one event at dispatch and one at return with the name of the supplier, would make upholstery measurable as well.

6.2.2 Harmonising the master data

The scans would measure how long each piece takes, and the master data would say what the piece is. A firm in this position could describe every finish with attributes common to all commissions and always filled in, at least the colour class, for example white or black, the gloss and the surface effect. It could also give uniform names to cabins, elements and pieces, and link each piece to the item of furniture it belongs to. Much of this information already exists upstream, because the design office maintains the bill of materials that links every piece to its destination on board (Section 1.2), and weight and dimensions are fields of the exports, rarely filled today (Section 4.12).

For a model, a common attribute is worth more than a code. A finish described as glossy white lacquer is recognised in any commission, and the history of one commission would become usable for the next. Within a commission, too, unseen categories would weigh less, because a new element would still have a known type, dimensions and finish. The results of Chapter 5 give an indication. Trained on the other commissions, the model does better than the family median on each of the three new builds. Added to a model that uses only the material family, the colour class, an attribute defined uniformly across commissions, reduces the error in each of the four held out, by 0.44 days on average (p 0.069). The cabin type, tried in the same way, increases it by 0.74 (p 0.200), so which attributes transfer has to be measured one by one (Section 5.6). Attributes of shape and size also appear in the literature. Masoumi and Bond (2025) classify the pieces of a furniture maker by shape and use their length and volume, while Canedo Rosa, Gonçalves and Barbalho (2025) start from geometry and material; in both studies these data explain the processing time of the piece, which the recording of Section 6.2.1 would make observable here too.

The requested date should be kept as it was when the piece was dispatched, in a field that does not change, because only then can it be used as a feature or compared with the actual duration. With uniform attributes at the source, finally, the dictionaries

that today have to be compiled by hand for every new commission would no longer be needed.

6.2.3 More history

Time alone resolves part of the limitations. With a few more years of history, every test period would find the same month of the year in training several times, whereas from October 2025 it finds it only once (Section 6.1.1), and the calendar features would have more examples of each season. With a longer period of evaluation, the paired test between months could also tell apart differences of a day or two. Other commissions, and refits in particular, would make it possible to train and test a model built specifically on refit data, which a single refit does not allow (Section 5.6).

More rows would help some algorithms more than others. The error of the forest does not change when it is trained on part of the rows, while that of regularised linear regression falls as they grow, by 2.17 days from a quarter to all of them (Section 5.11), and with a longer history the comparison between the two could turn out differently.

6.2.4 Developments of the model

Some developments of method respond to the limitations of the model, each starting from a measured result.

The first concerns the observation window of Section 6.1.2, which among the remedies for selection by closure has the lowest error, 13.52 days (Section 5.8.2). Against the family median its advantage, 11.3%, cannot be told apart from chance ($p = 0.111$), and the underestimation is halved without disappearing, to -4.32 days. With CatBoost the same feature gives an almost identical error, 13.54 days, while regularised linear regression worsens. A longer history (Section 6.2.3) would show whether the advantage over the official model is real.

A cycle still open at the date of prediction carries legitimate information, because its duration will be at least the time already elapsed. Survival analysis, which calls such observations right-censored (Section 5.8), estimates the distribution of durations from the observed length of the closed cycles and, for the open ones, only from the fact that they last longer than the time already elapsed (Klein and Moeschberger, 2003). The two ways of using this information that were tried, the accelerated failure time model (Barnwal, Cho and Hocking, 2020) and the estimated duration of the open cycles, have the limitations seen in Section 6.1.2. Models that pool information across groups, drawing towards the mean the estimates of those with few observed

closings, could use the cycles still in painting even where almost none has closed, and reduce the underestimation while keeping all the months in training.

In Section 2.6 the date to promise, communicated to whoever waits for the piece, is a quantile of the distribution of the duration. In Section 5.10 it is computed as the prediction plus a margin, equal to a quantile of the errors of the previous months. Intervals built in the same way cover less than they declare (Section 6.1.2). The quantile loss of Section 2.5 weighs overestimates and underestimates differently, and a model trained with it estimates directly the date that balances the costs of lateness and earliness (Koenker and Bassett, 1978). A margin computed on the most recent months only could instead follow the shift of the level. Both should be compared with the current margin on the same protocol, in terms of late pieces and days of earliness.

Today the updated prediction changes only once, at the first return of the batch. A model of the remaining time could update the estimate at every recorded event, as Lee *et al.* (2026) do with the sequence of activities of a piece, or as Flores-Huamán *et al.* (2025) use the prediction of an upstream operation for the next one. At present the useful events are few, the returns of pieces of the same batch, and with the scans of Section 6.2.1 they would become many more.

On new commissions the transfer worked for the three new builds, while on the refit the model loses (Section 5.6). Li *et al.* (2024) transfer a network trained on hull block types with a long history to a type with few examples, freezing part of it and adapting the rest. In the same way a model could start from the other commissions and adapt with the first closed cycles of the new one, giving more weight to its data as they accumulate. A refit would still require other data of the same type.

The automatic choice of the whole pipeline remains. Automated machine learning tools choose the data preparation, the algorithm and its hyperparameters by themselves. Tried by Bender, Trat and Ovtcharova (2022) in two firms, they chose algorithm and hyperparameters well but helped little in understanding the data and building the features, which in this thesis took the largest part of the work. Tested on the same honest protocol, they would show whether a pipeline chosen automatically does better than the eight algorithms compared. The oracles of Section 6.1.2, however, lead one to expect a small gain, because even a predictor that knew in advance the mean duration of site and week remains far from the error of one that knows the exit of the batch, information that no algorithm has at entry. The comparison would therefore serve mainly as a further check that no algorithm has been overlooked.

The use of the prediction to decide the order of processing depends instead on the ability to rank the pieces. Processing the shortest jobs first reduces the mean completion time (Pinedo, 2016), and requires their durations to be known in advance

(Section 2.6). At entry the correlation between prediction and duration is 0.23 (Section 5.8), and with a link this weak an order suggested by the model would often be wrong. The rule would become useful if piece-level recording improved the ability to rank.

6.2.5 A tool built on what is already measured

During the internship a tool was developed to support the project manager in tracking the pieces, a web application that acts as a control tower for the commissions. The application imports the two exports of the portal, maps the recorded phases onto a few states, such as in painting, in the warehouse or on board, and shows where each piece is, organising the pieces by commission, deck, cabin, element and finish. It is meant to gather in a single interface the weekly reconstruction of the state of the commission described in Section 1.2. The application shows where the pieces are, and the question from which this thesis starts is when they will be available again.

Even without changing the recording, two results of Chapter 5 are ready to answer it, and they are the two estimates that the batch view of Figure 6.2 displays. The updated prediction reduces the error of the family median by 27.9% on the pieces still in painting at the first return of their batch (Section 5.7). At entry, the date to promise with a margin equal to the 60% quantile of the past errors leaves 34.2% of the pieces late, as the family median does (35.4%), against 66.8% for the standard rule. In the ten months with at least a year of history the late pieces fall to 26.4%, against 34.7% for the family median (Section 5.10).

Since the pieces mostly move as batches, the main view is organised by entry document. Figure 6.2 shows a prototype built on the historical data of the thesis, with the anonymous codes used throughout it, as it would have appeared on 16 July 2026 for NC-1. Each row is a batch in painting, and the batches are ordered from the most urgent. The dark bar shows the time already spent in painting, the light band the range given at entry and the diamond the date to promise. The range, built at 80%, is described as containing the duration about eight times out of ten, because it covers slightly less (Section 6.1.2), and the point estimate stays in the detail, since on average it is early. Beside the bar the status of the batch can be read: on track, past the date to promise, beyond the range, or updated when some of its pieces have already returned. At the top, the boxes report the pieces and batches in painting, those expected in the next seven days and those past the date to promise or beyond the range.

For the project manager the view orders the urgencies. It shows at once the batches that have passed the date to promise and need expediting, those expected shortly, which can enter the planning of transport and installation, and those still on track.

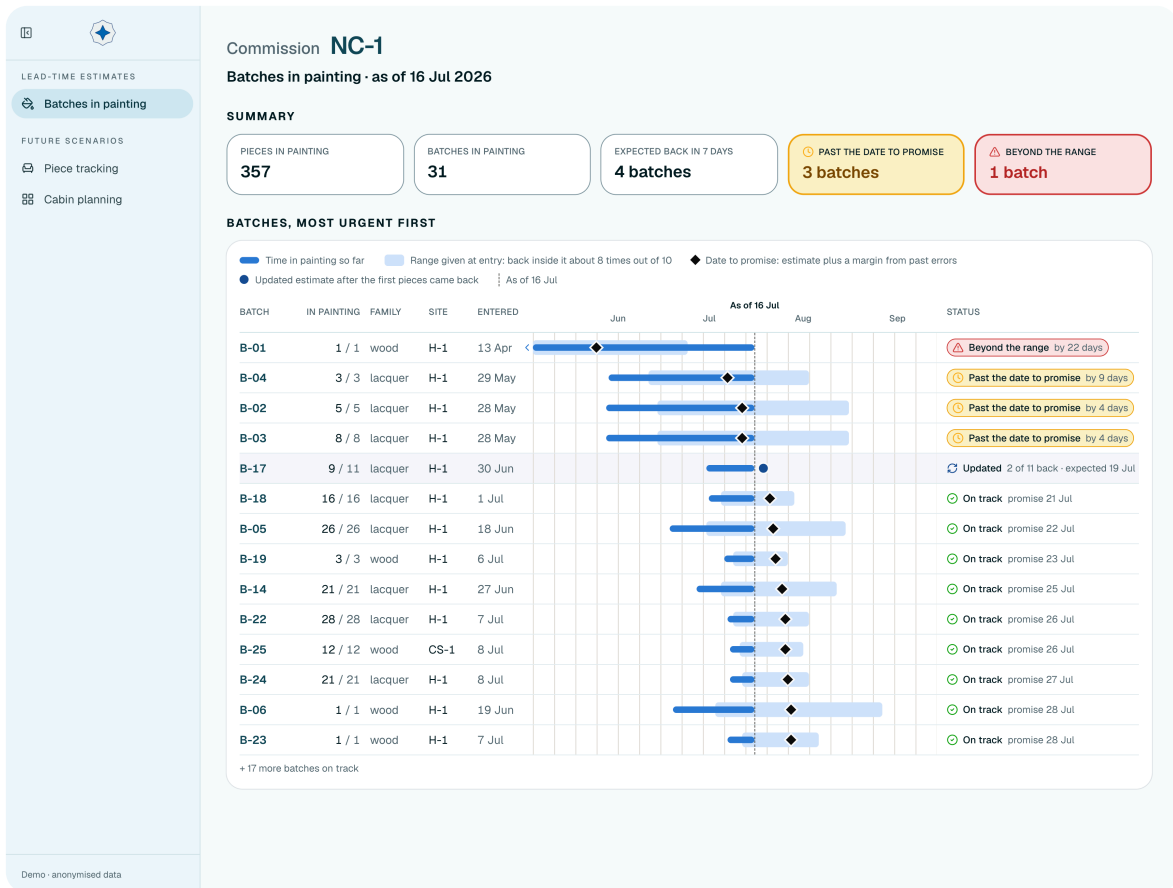


Figure 6.2: Prototype of the batch view: the batches of NC-1 in painting, most urgent first, with the time already spent in painting, the range given at entry, the date to promise, the updated prediction of a batch whose first pieces have returned and a batch beyond the range. Historical data, anonymised, as of 16 July 2026.

Figure 6.3 shows the detail of a batch taken as an example, B-17, which entered painting on 30 June with 11 lacquer pieces. On 13 July two of them came back, and the estimate for the nine still inside switched to the rule of the updated prediction. The median remaining time of the lacquer pieces closed by the end of June that were still in painting when their batch began to come out, 6 days, is added to the 13 elapsed until the first return, and the pieces are expected on 19 July. The estimate given at entry stays visible beside the updated one, for transparency. The update carries no range, because its dispersion has not been measured.

The design for integrating the machine learning model into the web application keeps training and prediction outside it, and the interface only displays their results. After each new export from the portal, a separate scheduled job reuses the data preparation and training evaluated in Chapter 5, rebuilds the finishing cycles, computes an estimate for every piece in painting and writes it to a table that the application only reads. The application stays light, and the estimates come from the same code produced by the thesis. The model is retrained once a month on the cycles

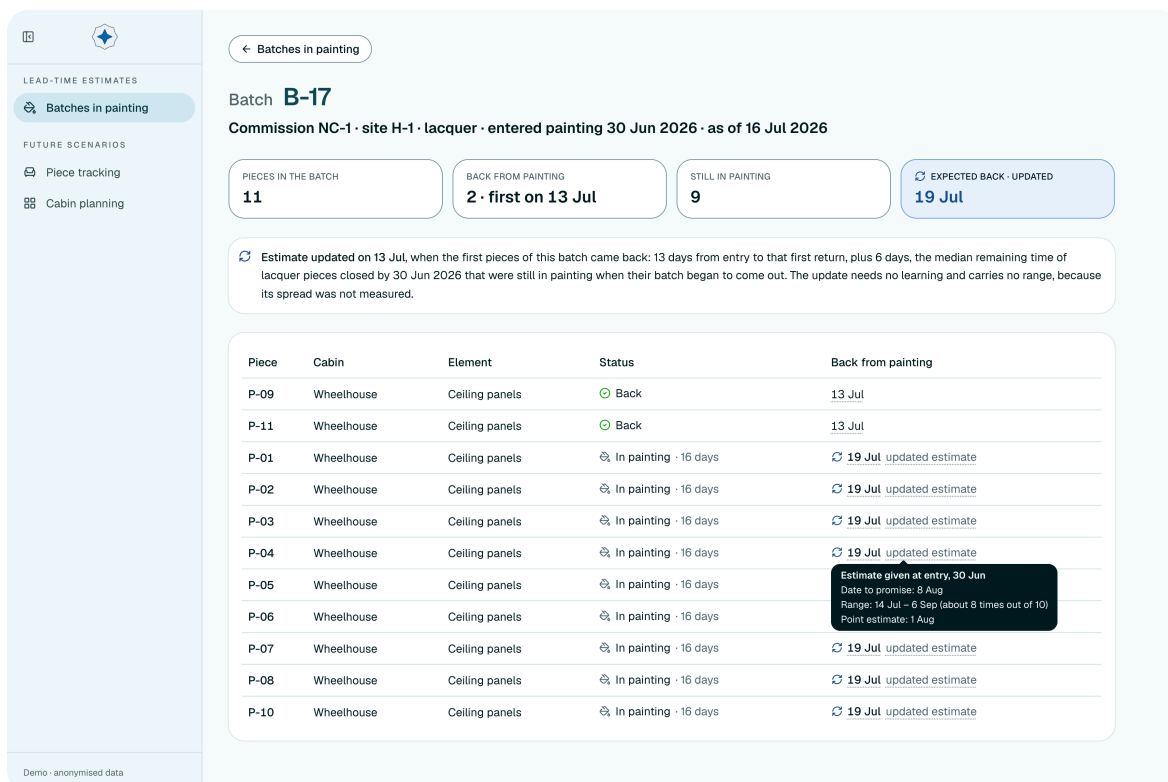


Figure 6.3: Detail of batch B-17 of Figure 6.2: two pieces back on 13 July and nine still in painting, whose estimate has switched to the updated prediction, the 13 days to the first return plus the median remaining time of lacquer pieces. The estimate given at entry stays visible. Historical data, anonymised, as of 16 July 2026.

already closed, as in the honest protocol, with fixed hyperparameters and a saved version. Every estimate carries the day it was computed, the date of the export it used and the version of the model, so that whoever reads it knows which situation it refers to. A monitoring page compares every month the predictions with the observed durations, with the family median and with the standard rule. Continuous use also requires the original exports to be kept, the finish and cabin dictionaries to be updated for every new commission, and a person responsible for the process.

The monitoring page also shows how far the tool changes what it measures. Among the tasks of the project manager is agreeing with the painting supplier which pieces come first (Section 1.2), and during the internship urgent pieces were observed to be expedited often. A tool that makes delays visible would presumably lead to expediting earlier and more often, and the expedited pieces would come out sooner, while the others, with the same capacity in the department, might wait longer. A model trained on the past would learn times that no longer hold, and the monthly comparison with the observed durations would reveal the drift. Recording each expediting request as an event, with its date and piece, would allow its effect to be measured and taken into account by the model, for example as a feature known at the moment of the updated prediction.

6.2.6 Dates for the piece, the item and the cabin

With the recording of Section 6.2.1 and the master data of Section 6.2.2, the same interface could give a date to every piece, and from there to every unit the project manager plans, the item of furniture and the cabin. Figure 6.4 and Figure 6.5 show this scenario on an invented commission, NC-X. They are illustrative screens, because the piece-level data that would feed them do not exist today, and the dates they show are invented.

For an individual piece, the page reports the phase, the finish, the site, the day of entry into painting, the date to promise, the range and the point estimate. It is the information the tool gives today by batch, brought down to the piece, and it lets the project manager know when each piece returns even when the pieces of the same batch leave on different days.

For an item of furniture, the three-dimensional exploded view shows its pieces apart, distinguished by phase, in painting or already returned (Figure 6.4). The item is ready by the latest date to promise among the pieces still in painting. The date to promise of each piece is needed, because the latest of many point estimates falls earlier than the day the last piece actually returns, and it takes only one piece exceeding its estimate for the whole to be ready later.

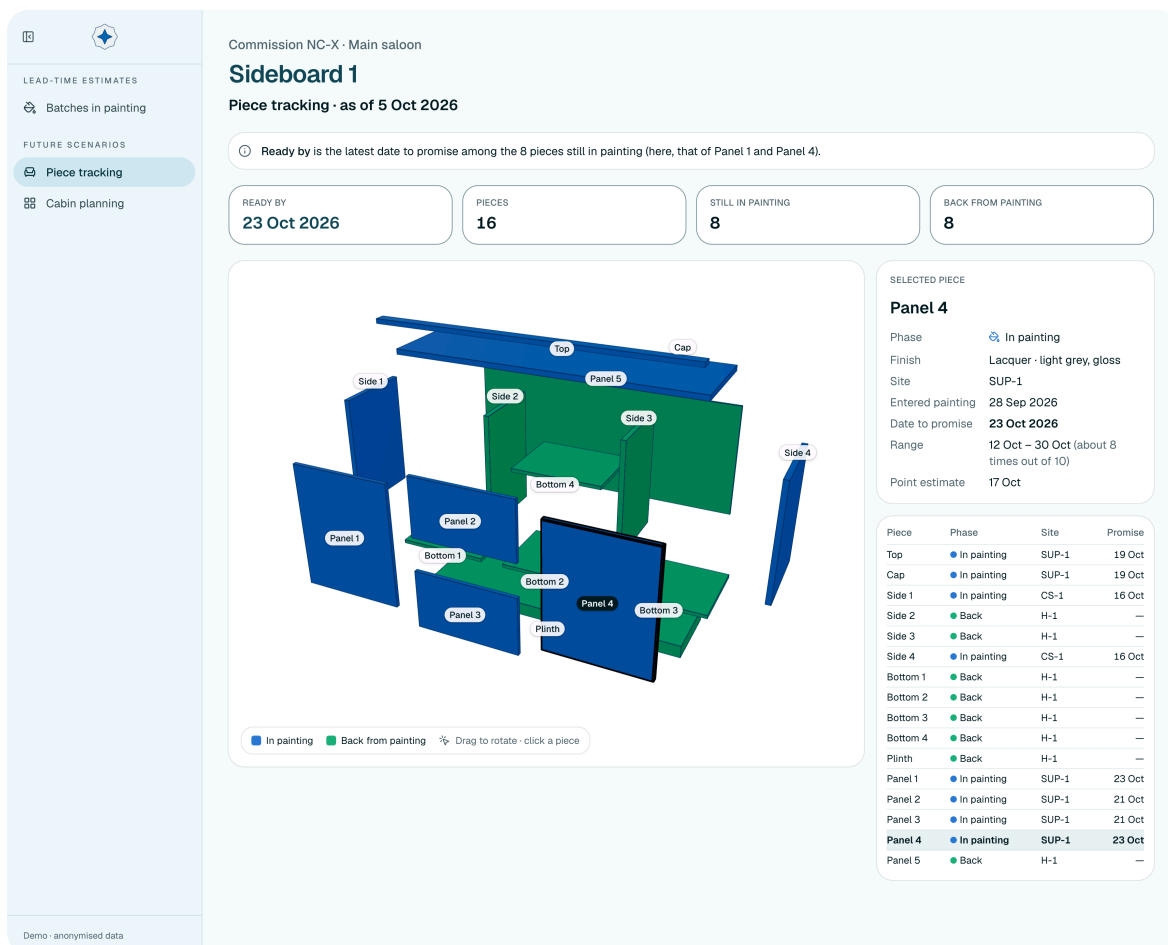


Figure 6.4: Illustrative scenario with piece-level recording: exploded view of a generic sideboard of an invented commission, NC-X, with the phase of each piece and, for the selected piece, its finish, date to promise and range. The item is ready by the latest date to promise among the pieces still in painting. Invented data.

The exploded view can cross, for each piece, the finish with its structural characteristics, shape, dimensions and role in assembly. The pieces of an item do not all matter equally for assembly. In an item like the one in the figure, the sides, the bottoms and the plinth form the structure and are presumably mounted before the panels and the top. Crossing these data with those of painting, the project manager can decide which piece to expedite according to its role, and start pre-assembly with the structural pieces already returned while the finishing ones are still in the department. A targeted request on one piece presumably weighs less on the department than expediting the whole batch, and leaves it free to process the rest in its own order.

For a cabin, the plan shows the week in which each element will be ready, and beside it a Gantt chart compares the planned installation with the predicted date of availability, row by row (Figure 6.5). An element is ready by the latest date to promise among its pieces still in painting, and the cabin when all its elements are.

Until an element has entered the department the cabin has no date, and the page says so.

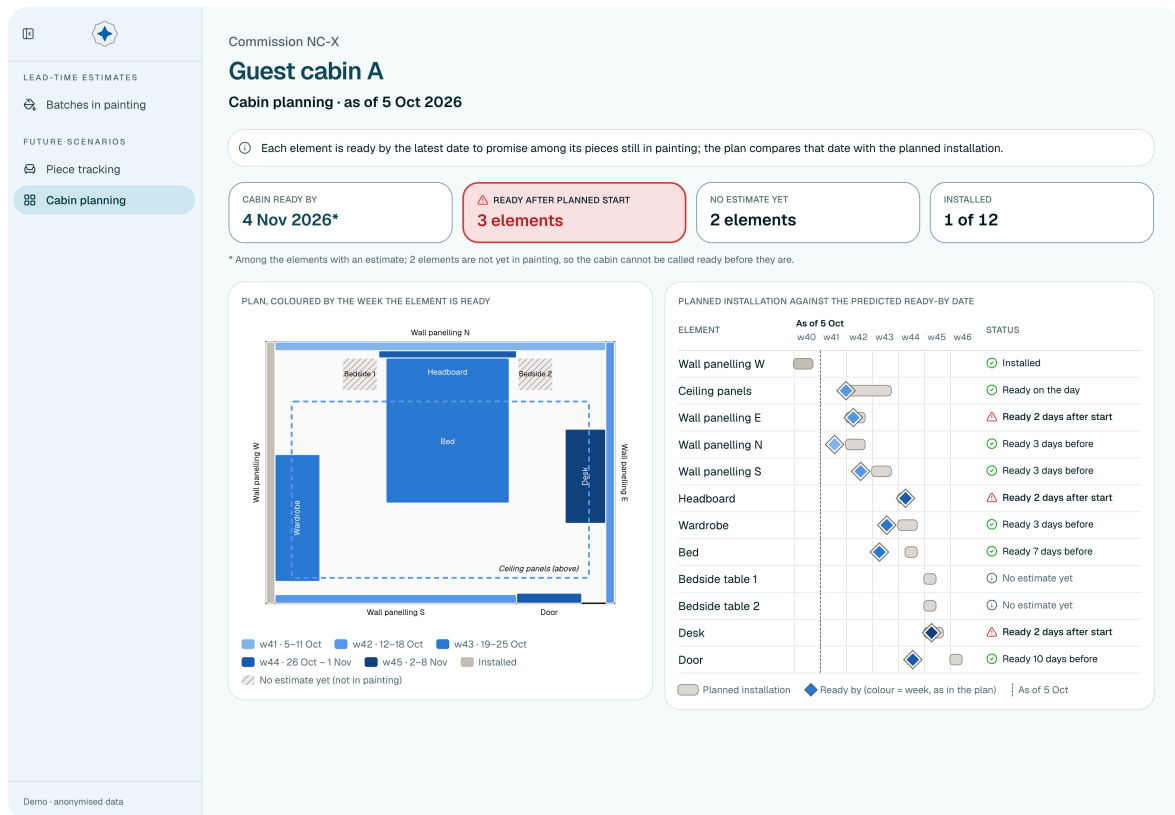


Figure 6.5: Illustrative scenario for a cabin of NC-X: plan with the week in which each element is expected to be ready and, beside it, planned installation against the predicted ready-by date, with the elements ready after the planned start. Invented data.

The advantage for the project manager is to see weeks in advance the conflicts between the installation plan and the availability of the pieces. They can expedite earlier the elements that would arrive after the planned start of work on board, reorder the fitting starting from those that are ready, agree in good time with the shipyard on moving an installation window and give the fitters a realistic sequence.

In Chapter 1 the main objective of the thesis was motivated by its operational use: a project manager who knew, with reasonable approximation, when a piece with a given finish would be available again could build a plan closer to reality and act before a delay propagated downstream. An effective prediction model, combined with an interface like that of the web application, would allow them to do so. With what is already measured, the application could give them this information for the batches, through the updated prediction and the date to promise, and with the two scans for every piece, item and cabin.

Conclusions

The work has attempted to estimate with machine learning the finishing lead time of the pieces of yacht interiors, from the movements recorded by the portal of a firm in the sector, on 14,245 cycles of four commissions. The models were evaluated over 17 consecutive months in the conditions of use, trained only on what was already known at the date of prediction, and compared through a statistical test with base-lines that require no training.

The first research question concerns the prediction at entry into painting. The model errs by 14.95 days, against 15.24 for the family median and 16.71 for the standard rule, and neither difference can be told apart from zero (p 0.835 and 0.124). The error is about half the median duration of a cycle, 30 days. The advantage is concentrated on wood, where the model reduces the error of the standard rule by 18.19% (p 0.041, nominally significant). None of the other seven algorithms differs from the forest in an established way, and regularised linear regression, at 14.04 days, is the one that errs least.

On average the model underestimates by eight days, mainly because in the last months before the prediction it sees only the fast pieces as closed. Trained on the mature months only, it underestimates by less than a day, with an absolute error that does not change distinguishably. As in Talonpoika (2026) and Bender, Trat and Ovtcharova (2022), where the mean of the training durations beats the estimate in use, here too a simple baseline does better than current practice, and at entry the useful gain comes mostly from moving from a constant to a median. Used as the date to promise, the family median and the prediction with a margin at 60% leave 35.4% and 34.2% of the pieces late, against 66.8% for the rule in use. In the ten months from October 2025, the first with at least a year of history in training, the advantage over the family median rises to 11.7% (p 0.076) and the late pieces with the date to promise fall to 26.4%, figures read after the results and therefore descriptive.

Among the recorded information, the duration depends above all on the finish, through its code within a commission and the material family across commissions. The finish is the only group of features whose removal worsens the model with a nominally significant difference (p 0.029). Whether the data observe this information for the individual piece is the other part of the second question, and for three pieces out of four the answer is negative, because they leave painting on the same day as their document. A predictor that knew in advance when its own batch leaves would err by 2.59 days, against 10.14 for one given the mean duration of site and week, and

the difference, 7.55, presumably corresponds to the decision of the department on which batch to process and when. Where the recording lets the piece through, the finish code explains 7.29% of the variability of its exit within the document, against 0.95% expected by chance.

None of the 12 segmentation configurations improves on the single model in an established way, whether the groups come from domain knowledge or from an algorithm. Those found in the data are also weak as partitions, with a silhouette between 0.23 and 0.35. A compatible reading is that, with few independent observations, splitting the sample raises the variance more than each group adds in information.

Trained on the other commissions, the model does better than the family median on each of the three new builds and loses on the refit, 7.05 days against 6.45, in line with the difference between the two types of commission described in Chapter 1.

Updated at the first return of the batch, the estimate for the pieces still in painting cuts the error of the family median by 27.9% ($p = 0.005$, in 15 months out of 17), a result that holds up under the Holm correction against all four baselines, answers the fourth question and is the most solid of the work. A rule without learning, which adds the median remaining time of the material family to the days already elapsed, obtains the same gain, with an error of 12.83 against 12.67 ($p = 0.893$). The decisive information is that the batch has started to come out, together with the time already spent.

Compared with the studies reviewed in Chapter 3, the first contribution of the thesis concerns evaluation. Trained also on the cycles still open at the date of prediction, the same algorithm beats the family median by 14.56% in an established way, and with the honest protocol the advantage falls to 1.91%. With the same hyperparameters, the construction of the training data alone is worth 7.46 percentage points. With a single split in time, too, an advantage looks established that the rolling origin does not confirm on the same pieces. Among the studies on recorded data only one splits the observations in time, and none checks whether the training cycles had already closed at the date of prediction. In these data, with the same algorithm, that check reverses the verdict.

The second contribution is the value of the first return of the batch. Lee *et al.* (2026) and Flores-Huamán *et al.* (2025) use as information the events that have already occurred in the process, the sequence of activities of a piece or the prediction of the upstream operation. Here the event is the return of the first piece of the same batch, which concerns 38.4% of the test pieces, and the gain it brings does not require a model.

The third concerns recording. In the studies reviewed, wrong or missing records are noise to be filtered or a limit to be declared (Bender, Trat and Ovtcharova, 2022;

Flores-Huamán *et al.*, 2025), whereas in these data it is batch-level recording that establishes what can be predicted. No algorithm learns what is not recorded, and at entry the quantity to be predicted is mostly the moment at which the batch will be processed and sent out, a decision taken in the department and absent from the data.

Upholstery, the other finishing operation, was not modelled, because dispatches to the upholsterers are recorded to very different extents from one commission to another and the return almost always appears without the name of the supplier (Section 4.7.1). The indirect route, which would derive the duration from a plan of the department, was not viable, because capacities and resource availability do not appear in the data (Section 4.2.1). The future developments are discussed in Chapter 6. The one that promises the largest gain concerns the recording of the start and the end of the processing of the individual piece. A planning tool, by contrast, can already use the updated prediction and the date to promise with a margin. Open questions for the model include the observation window, the feature that tells it how long each cycle has had to close, which has the lowest error among the remedies and awaits confirmation on a longer history, training with the quantile loss, models for censored data that pool information across groups, and updating the prediction at every recorded event.

Appendix A

Source Code Excerpts

This appendix reports excerpts of the Python code that produces the results of Chapter 4 and Chapter 5, in the order in which the steps are carried out. Comments and docstrings are removed, and no name of the firm, of its sites or of its suppliers appears. Where lines that only count intermediate cases, or that build variants not discussed here, are left out, the caption marks the listing as abridged.

A.1 Building the finishing cycle

The function reads the events of each piece in time order and pairs an entry into painting with the first receipt from the department that follows it, applying the rules of Section 4.5 and Figure 4.2. Each event has already been classified as an entry (OPEN), a receipt (CLOSE), a dispatch towards the exit department (FALLBACK) or a record elsewhere (ELSEWHERE). A dispatch towards the exit closes the cycle only if no receipt follows, and in the refit a record elsewhere does not abandon the opening.

Listing A.1: Pairing of entries and exits into finishing cycles (data_prep.py, abridged)

```
1 def pair_cycles(events: pd.DataFrame, refit_exempt: bool = True) -> pd.DataFrame:
2     columns = ["code", "piece_id", "event_date", "kind", "regime", "site",
3               "document"]
4     code, piece, date, kind, regime, site, document = (
5         events[c].to_numpy() for c in columns)
6
7     rows = []
8     boundaries = np.flatnonzero(
9         (piece[1:] != piece[:-1]) | (code[1:] != code[:-1])) + 1
10    for start, stop in zip(np.r_[0, boundaries],
11                          np.r_[boundaries, len(events)]):
12        exempt = refit_exempt and COMMISSION_TYPE[code[start]] == "refit"
13        open_at = fallback_at = open_regime = open_site = open_document = None
14        seen_elsewhere = False
15
16    def emit(close_index, by_fallback):
17        rows.append((code[start], piece[start], open_at, date[close_index],
18                    open_regime, regime[close_index], by_fallback, open_site,
19                    open_document))
```

```

20
21     for i in range(start, stop):
22         k = kind[i]
23         if k == OPEN:
24             if open_at is None:
25                 open_at, open_regime, open_site = date[i], regime[i], site[i]
26                 open_document = document[i]
27                 fallback_at, seen_elsewhere = None, False
28             elif seen_elsewhere and not exempt:
29                 if fallback_at is not None:
30                     emit(fallback_at, True)
31                 open_at, open_regime, open_site = date[i], regime[i], site[i]
32                 open_document = document[i]
33                 fallback_at, seen_elsewhere = None, False
34         elif k == CLOSE:
35             if open_at is not None:
36                 emit(i, False)
37                 open_at = fallback_at = open_regime = open_site = None
38                 open_document = None
39                 seen_elsewhere = False
40         elif k == FALLBACK:
41             if open_at is not None and fallback_at is None:
42                 fallback_at = i
43         elif k == ELSEWHERE:
44             if open_at is not None:
45                 seen_elsewhere = True
46
47     if open_at is not None and fallback_at is not None:
48         emit(fallback_at, True)
49
50     cycles = pd.DataFrame(rows, columns=[
51         "code", "piece_id", "open_date", "close_date", "open_regime",
52         "close_regime", "closed_by_fallback", "site", "entry_document"])
53     cycles["duration_days"] = (cycles["close_date"] - cycles["open_date"]).dt.days
54     return cycles

```

A.2 The monthly folds of the honest protocol

For every test month with enough data the function yields one fold, training on the cycles that entered before it and had already closed by its first day and testing on those that enter during it (Section 5.1). In the code `boundary` names the last month of training; the boundary of Section 5.1 is the day after its end.

Listing A.2: Monthly folds of the honest protocol (`honest.py`)

```
1 MIN_TRAIN, MIN_TEST = 500, 100
2
3
4 def honest_folds(t):
5     months = sorted(t["entry_date"].dt.to_period("M").unique())
6     per = t["entry_date"].dt.to_period("M")
7     for boundary in months[:-1]:
8         cutoff = boundary.to_timestamp(how="end")
9         train = t.index[(per <= boundary) & (t["close_date"] <= cutoff)]
10        test = t.index[per == boundary + 1]
11        if len(train) >= MIN_TRAIN and len(test) >= MIN_TEST:
12            yield list(train), list(test), boundary
```

A.3 The baselines and the model in each month

The four baselines are recomputed on the training rows of every fold: the median by material family, by segment and over all pieces, and the standard rule, the requested date where it is filled in and 21 days (STANDARD) otherwise (Section 5.1). In the excerpts `T` and `L` are the modules of training and evaluation, and `T.TARGET` is the duration in days.

Listing A.3: The four baselines (`ladder.py`)

```
1 def baselines_for(train, test):
2     ov = train[T.TARGET].median()
3     fam = train.groupby("material_family")[T.TARGET].median()
4     seg = train.groupby("domain_segment")[T.TARGET].median()
5     pl = test["plan_days"].to_numpy(dtype=float)
6     return {"material": test["material_family"].map(fam).fillna(ov).to_numpy(),
7             "segment": test["domain_segment"].map(seg).fillna(ov).to_numpy(),
8             "global": np.full(len(test), ov),
9             "hybrid-standard": np.where(~np.isnan(pl),
10                                         np.nan_to_num(pl, nan=STANDARD),
11                                         float(STANDARD))}
```

In each fold the model receives the hyperparameters of its search, fits the encoders on the training rows alone and predicts the durations of the test month, and its error is set beside that of the baselines.

Listing A.4: Monthly training and prediction (stage2_rb3.py, abridged)

```

1 def static_arm(t, folds, params_for):
2     rows = []
3     for i, (tr, te, boundary) in enumerate(folds):
4         a, b = t.iloc[tr], t.iloc[te]
5         m, ek = T.suggest(L.METHOD, optuna.trial.FixedTrial(params_for(i, a)))
6         e = T.fit_encoders(a, a[T.TARGET].to_numpy(), None, ek)
7         X, _, _ = T.transform(a, e)
8         Xt, _, _ = T.transform(b, e)
9         pred = np.maximum(T.fit_predict(m, X, a[T.TARGET].to_numpy(), Xt,
10                                     groups=a["site_day"].to_numpy(),
11                                     early_stop=True), 0.0)
12         truth = b[T.TARGET].to_numpy()
13         base = L.baselines_for(a, b)
14         r = dict(fold=i + 1, month=str(boundary + 1), n=len(b),
15                 n_groups=int(b["site_day"].nunique()),
16                 mae=float(np.abs(pred - truth).mean()))
17         for k, v in base.items():
18             r[f"mae_{k}"] = float(np.abs(v - truth).mean())
19         rows.append(r)
20     return pd.DataFrame(rows)

```

A.4 The updated prediction

Updating concerns the pieces that leave after the first exit of their entry document, and for them the days elapsed until that moment are known (Section 5.7). The updated model raises the prediction made at entry to at least the elapsed days, and the rule without learning adds to them the median remaining time of the training pieces of the same material family that were in the same position.

Listing A.5: The updated prediction and the rule without learning (stage2_rb3.py, abridged)

```

1 def first_wave_on(t, static_all, folds):
2     f = t.copy()
3     f["exit_date"] = f["open_date"] + pd.to_timedelta(f[T.TARGET], unit="D")
4     g = f.groupby("entry_document")
5     first = g["exit_date"].transform("min")
6     f["still_open"] = f["exit_date"] > first
7     f["elapsed"] = (first - f["open_date"]).dt.days.astype(float)
8     f["remaining"] = f[T.TARGET] - f["elapsed"]
9     f["_static"] = static_all
10    rows = []
11    for i, (tr, te, boundary) in enumerate(folds):
12        a, b = f.iloc[tr], f.iloc[te]

```

```

13     a_pop, b_pop = a[a["still_open"]], b[b["still_open"]]
14     if len(b_pop) < L.MIN_TEST or len(a_pop) < 50:
15         continue
16     static = b_pop["_static"].to_numpy(dtype=float)
17     elapsed = b_pop["elapsed"].to_numpy()
18     floored = np.maximum(static, elapsed)
19     gap = a_pop.groupby("material_family")["remaining"].median()
20     gap_all = float(a_pop["remaining"].median())
21     sibling = (elapsed + b_pop["material_family"].map(gap)
22               .fillna(gap_all).to_numpy())
23     truth = b_pop[T.TARGET].to_numpy()
24     row = dict(fold=i + 1, month=str(boundary + 1), n=len(b_pop),
25               n_groups=int(b_pop["site_day"].nunique()), n_train=len(a_pop),
26               mae=float(np.abs(floored - truth).mean()),
27               mae_static=float(np.abs(static - truth).mean()),
28               mae_sibling=float(np.abs(sibling - truth).mean()))
29     rows.append(row)
30     return pd.DataFrame(rows)

```

A.5 Paired comparison and Holm correction

Each comparison between a model and a baseline yields one mean absolute error per test month for each. The first function computes the difference in the three averages of Section 5.1, the paired *t*-test and the months won. It also computes a sign test, which the thesis does not report. The second applies the correction of Holm to the p-values of a family.

Listing A.6: Paired statistics over the months and Holm adjustment (ladder.py, ch5.py)

```

1  def summarise(g, col, ref):
2      d = (g[ref] - g[col]).to_numpy()
3      a_, b_ = g[ref].to_numpy(), g[col].to_numpy()
4      nr, ng = g["n"].to_numpy(float), g["n_groups"].to_numpy(float)
5      w = int((d > 0).sum())
6      return {
7          "fold_mean": (float(d.mean()),
8                       float((a_.mean() - b_.mean()) / a_.mean() * 100)),
9          "row_weighted": (
10             float((d * nr).sum() / nr.sum()),
11             float(((a_ * nr).sum() - (b_ * nr).sum()) / (a_ * nr).sum() * 100)),
12          "group_weighted": (
13             float((d * ng).sum() / ng.sum()),
14             float(((a_ * ng).sum() - (b_ * ng).sum()) / (a_ * ng).sum() * 100)),
15          "sd": float(d.std(ddof=1)),
16          "t_p": float(stats.ttest_lsamp(d, 0).pvalue),
17          "wins": w, "folds": len(g), "n": int(nr.sum()),
18          "sign_p": float(stats.binomtest(w, len(d), 0.5,
19                                         alternative="two-sided").pvalue),
20          "mae_ref": float(a_.mean()), "mae_col": float(b_.mean())}
21

```

```

22
23 def holm(p):
24     p = np.asarray(p, float)
25     adj, running = np.empty(len(p)), 0.0
26     for j, idx in enumerate(np.argsort(p, kind="stable")):
27         running = max(running, min(1.0, (len(p) - j) * p[idx]))
28         adj[idx] = running
29     return adj

```

A.6 The date to promise

For each test month the margin is a quantile of the errors of the previous months, computed only on the cycles that had closed by the boundary, and the date to promise adds it to the prediction (Section 5.10). The table `R` holds, for every test piece, its fold, the prediction of the model and the actual duration, and `_promise` computes the share of late pieces and the mean delay and earliness. The first month, without earlier errors, has no margin.

Listing A.7: Margin of the date to promise (ch5.py, abridged)

```

1  for k in sorted(R["fold"].unique()):
2      cur = R[R["fold"] == k]
3      end = cur["boundary_end"].iloc[0]
4      pool = R[(R["fold"] < k) & (pd.to_datetime(R["close_date"]) <= end)]
5      if len(pool) < 30:
6          continue
7      r_day = pool["actual"] - pool["model"]
8      n_p = len(pool)
9      for q in (0.5, 0.6, 0.7, 0.8, 0.9):
10         margin = float(np.quantile(r_day, q))
11         promised = np.maximum(cur["model"] + margin, 0)
12         promise.append(_promise(cur, promised, k, f"q{int(round(q * 100))}",
13                                "model", margin, n_p))

```

References

- Alpaydin, E. (2020) *Introduction to Machine Learning*. 4th ed. MIT Press.
- Amaro, G., Hendry, L. and Kingsman, B. (1999) "Competitive advantage, customisation and a new taxonomy for non make-to-stock companies," *International Journal of Operations and Production Management*, 19(4), pp. 349–371.
- Barnwal, A., Cho, H. and Hocking, T. (2020) "Survival regression with accelerated failure time model in XGBoost," *arXiv:2006.04920* [Preprint].
- Bender, J. and Ovtcharova, J. (2021) "Prototyping machine-learning-supported lead time prediction using AutoML," *Procedia Computer Science*, 180, pp. 649–655.
- Bender, J., Trat, M. and Ovtcharova, J. (2022) "Benchmarking AutoML-supported lead time prediction," *Procedia Computer Science*, 200, pp. 482–494.
- Bergstra, J. *et al.* (2011) "Algorithms for hyper-parameter optimization," in *Advances in Neural Information Processing Systems* 24, pp. 2546–2554.
- Bertrand, J.W.M. and Muntslag, D.R. (1993) "Production control in engineer-to-order firms," *International Journal of Production Economics*, 30–31, pp. 3–22.
- Bezdek, J.C. (1981) *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum Press.
- Bianchini, A. *et al.* (2024) "Manufacturing execution system application within manufacturing small–medium enterprises towards key performance indicators development and their implementation in the production line," *Sustainability*, 16(7), p. 2974.
- Bishop, C.M. (2006) *Pattern Recognition and Machine Learning*. Springer.
- Breiman, L. (1996) "Bagging predictors," *Machine Learning*, 24(2), pp. 123–140.
- Breiman, L. (2001) "Random forests," *Machine Learning*, 45(1), pp. 5–32.
- Breiman, L. *et al.* (1984) *Classification and Regression Trees*. Wadsworth.
- Burggräf, P. *et al.* (2021) "Machine learning-based prediction of missing components for assembly – a case study at an engineer-to-order manufacturer," *IEEE Access*, 9, pp. 105926–105938.
- Burggräf, P. *et al.* (2020) "Approaches for the prediction of lead times in an engineer to order environment—A systematic review," *IEEE Access*, 8, pp. 142434–142445.
- Canedo Rosa, R., Gonçalves, M.C. and Barbalho, S.C.M. (2025) "Machine learning applied to forecasting the manufacturing time of new products prototypes and ETO

products: An exploratory study," *International Journal of Production Economics*, 287. Available at: <https://doi.org/10.1016/j.ijpe.2025.109688>.

Caruana, R. and Niculescu-Mizil, A. (2006) "An empirical comparison of supervised learning algorithms," in *Proceedings of the 23rd International Conference on Machine Learning*. ACM, pp. 161–168.

Chen, T. and Guestrin, C. (2016) "XGBoost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, pp. 785–794.

Cortes, C. and Vapnik, V. (1995) "Support-vector networks," *Machine Learning*, 20(3), pp. 273–297.

Cover, T.M. and Hart, P.E. (1967) "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, 13(1), pp. 21–27.

Drucker, H. *et al.* (1997) "Support vector regression machines," in *Advances in Neural Information Processing Systems 9*, pp. 155–161.

Duan, N. (1983) "Smearing estimate: A nonparametric retransformation method," *Journal of the American Statistical Association*, 78(383), pp. 605–610.

Flores-Huamán, K.-J. *et al.* (2025) "Enhancing process lead time forecasting with machine learning and upstream process data: A case study in wind tower manufacturing," *Computers & Industrial Engineering*, 209. Available at: <https://doi.org/10.1016/j.cie.2025.111410>.

Freund, Y. and Schapire, R.E. (1997) "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, 55(1), pp. 119–139.

Friedman, J.H. (2001) "Greedy function approximation: A gradient boosting machine," *The Annals of Statistics*, 29(5), pp. 1189–1232.

Goodfellow, I., Bengio, Y. and Courville, A. (2016) *Deep Learning*. MIT Press.

Gosling, J. and Naim, M.M. (2009) "Engineer-to-order supply chain management: A literature review and research agenda," *International Journal of Production Economics*, 122(2), pp. 741–754.

Graham, R.L. *et al.* (1979) "Optimization and approximation in deterministic sequencing and scheduling: A survey," *Annals of Discrete Mathematics*, 5, pp. 287–326.

Grinsztajn, L., Oyallon, E. and Varoquaux, G. (2022) "Why do tree-based models still outperform deep learning on typical tabular data?," in *Advances in Neural Information Processing Systems 35, Datasets and Benchmarks Track*.

Hastie, T., Tibshirani, R. and Friedman, J. (2009) *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed. Springer.

- Hicks, C., McGovern, T. and Earl, C.F. (2000) "Supply chain management: A strategic issue in engineer to order manufacturing," *International Journal of Production Economics*, 65(2), pp. 179–190.
- Hillier, F.S. and Lieberman, G.J. (2015) *Introduction to Operations Research*. 10th ed. McGraw-Hill.
- Hoerl, A.E. and Kennard, R.W. (1970) "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, 12(1), pp. 55–67.
- Holm, S. (1979) "A simple sequentially rejective multiple test procedure," *Scandinavian Journal of Statistics*, 6(2), pp. 65–70.
- James, G. et al. (2013) *An Introduction to Statistical Learning with Applications in R*. Springer.
- Ke, G. et al. (2017) "LightGBM: A highly efficient gradient boosting decision tree," in *Advances in Neural Information Processing Systems 30*, pp. 3146–3154.
- Klein, J.P. and Moeschberger, M.L. (2003) *Survival Analysis: Techniques for Censored and Truncated Data*. 2nd ed. Springer.
- Koblasa, F. and Urban, M. (2026) "Time prediction methods in production – a scoping review of the 21st century so far," *Computers & Industrial Engineering*, 213. Available at: <https://doi.org/10.1016/j.cie.2025.111713>.
- Koenker, R. and Bassett, G. (1978) "Regression quantiles," *Econometrica*, 46(1), pp. 33–50.
- Kohavi, R. (1995) "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pp. 1137–1143.
- Lee, Y. et al. (2026) "Process-aware procurement lead time prediction for shipyard delay mitigation," in *Production Research (ICPR 2025), Lecture Notes in Production Engineering*. Springer, pp. 3–13.
- Li, J. and Lafond, D. (2023) "Hybrid human-AI forecasting for task duration estimation in ship refit," in *Machine Learning, Optimization, and Data Science (LOD 2022), Lecture Notes in Computer Science*. Springer, pp. 558–572.
- Li, J. et al. (2024) "Time prediction in ship block manufacturing based on transfer learning," *Journal of Marine Science and Engineering*, 12(11), p. 1977. Available at: <https://doi.org/10.3390/jmse12111977>.
- Lingitz, L. et al. (2018) "Lead time prediction using machine learning algorithms: A case study by a semiconductor manufacturer," *Procedia CIRP*, 72, pp. 1051–1056.
- Little, J.D.C. (1961) "A proof for the queuing formula: $L = \lambda W$," *Operations Research*, 9(3), pp. 383–387.

- Liu, L. *et al.* (2025) "Integrated optimization scheduling model for ship outfitting production with endogenous uncertainties," *Journal of Marine Science and Application*, 24, pp. 194–209.
- Lloyd, S.P. (1982) "Least squares quantization in PCM," *IEEE Transactions on Information Theory*, 28(2), pp. 129–137.
- Lundberg, S.M. and Lee, S.-I. (2017) "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems* 30, pp. 4765–4774.
- Masoumi, A. and Bond, B.H. (2025) "Machine learning-based prediction of processing time in furniture manufacturing to estimate lead time and pricing," *European Journal of Wood and Wood Products*, 83. Available at: <https://doi.org/10.1007/s00107-024-02177-w>.
- Micci-Barreca, D. (2001) "A preprocessing scheme for high-cardinality categorical attributes in classification and prediction problems," *ACM SIGKDD Explorations Newsletter*, 3(1), pp. 27–32.
- Nadeau, C. and Bengio, Y. (2003) "Inference for the generalization error," *Machine Learning*, 52, pp. 239–281.
- Pinedo, M.L. (2016) *Scheduling: Theory, Algorithms, and Systems*. 5th ed. Springer.
- Prokhorenkova, L. *et al.* (2018) "CatBoost: Unbiased boosting with categorical features," in *Advances in Neural Information Processing Systems* 31.
- Rousseeuw, P.J. (1987) "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, 20, pp. 53–65.
- Rumelhart, D.E., Hinton, G.E. and Williams, R.J. (1986) "Learning representations by back-propagating errors," *Nature*, 323, pp. 533–536.
- Semini, M. *et al.* (2014) "Strategies for customized shipbuilding with different customer order decoupling points," *Proceedings of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment*, 228(4), pp. 362–372.
- Steinberg, F. *et al.* (2022) "Impact of material data in assembly delay prediction: A machine learning-based case study in machinery industry," *The International Journal of Advanced Manufacturing Technology*, 120, pp. 1333–1346. Available at: <https://doi.org/10.1007/s00170-022-08767-3>.
- Talonpoika, T. (2026) *Machine Learning for Predicting Production Lead Times using ERP Data: A predictive modelling study in make-to-order manufacturing*. Master's thesis. University of Vaasa.
- Tashman, L.J. (2000) "Out-of-sample tests of forecasting accuracy: An analysis and review," *International Journal of Forecasting*, 16(4), pp. 437–450.

- Tibshirani, R. (1996) "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society, Series B*, 58(1), pp. 267–288.
- Vock, D.M. *et al.* (2016) "Adapting machine learning techniques to censored time-to-event health record data: A general-purpose approach using inverse probability of censoring weighting," *Journal of Biomedical Informatics*, 61, pp. 119–131.
- Welsing, M. *et al.* (2021) "Combining process mining and machine learning for lead time prediction in high variance processes," in *Production at the Leading Edge of Technology (WGP 2020), Lecture Notes in Production Engineering*. Springer, pp. 528–537.
- Wortmann, J.C. (1992) "Production management systems for one-of-a-kind products," *Computers in Industry*, 19(1), pp. 79–88.
- Zou, H. and Hastie, T. (2005) "Regularization and variable selection via the elastic net," *Journal of the Royal Statistical Society, Series B*, 67(2), pp. 301–320.