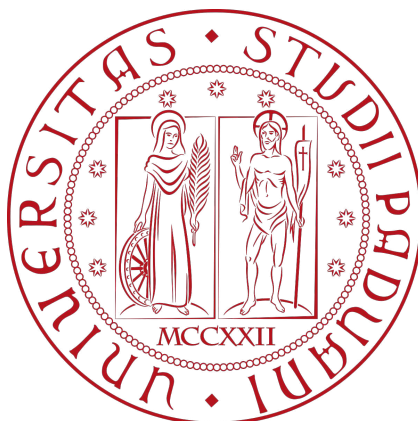


**UNIVERSITÀ DEGLI STUDI DI PADOVA**

DIPARTIMENTO DI BIOLOGIA

Corso di Laurea in Biotecnologie



**ELABORATO DI LAUREA**

Machine Learning to Predict Melting Temperatures of  
Deep Eutectic Solvents from Molecular Descriptors

Tutor

*Dott. Sergio Rampino*

*Dipartimento di Scienze Chimiche*

Laureanda: *Luna Liviero*

ANNO ACCADEMICO 2024/2025

# Contents

<b>1</b>	<b>State of the art</b>	<b>1</b>
1.1	Deep Eutectic Solvents . . . . .	2
1.2	Machine-Learning Basics . . . . .	3
<b>2</b>	<b>Methods</b>	<b>5</b>
2.1	Software environment and tools . . . . .	5
2.2	Codebase . . . . .	5
2.2.1	Feature engineering . . . . .	6
2.2.2	Datasets . . . . .	7
2.2.3	Evaluation Metrics . . . . .	8
2.3	Machine Learning Models . . . . .	9
2.3.1	Training and Testing . . . . .	9
2.3.2	Hyperparameter optimization . . . . .	9
2.3.3	Models . . . . .	10
2.4	Workflow . . . . .	12
<b>3</b>	<b>Results</b>	<b>14</b>
<b>4</b>	<b>Conclusion and discussion</b>	<b>18</b>
	<b>Bibliography</b>	<b>22</b>

## Abstract

Deep Eutectic Solvents (DESs) emerged as promising alternatives to conventional solvents for their peculiar characteristics. Among these, melting temperature is a critical parameter that deeply influences their industrial applicability. It is possible to use machine learning (ML) models to predict the melting temperatures of DES mixtures based on their molecular features, to reduce time-consuming and costly experimental work. The approach involves training and testing a series of existing learning models, using the code developed by a separate research group, on an expanded dataset curated by a colleague. Moreover the possibility of modifying the molecular descriptors originally used as input for prediction will be explored. The impact of the changes implemented in the original code will be assessed using statistical metrics, which also allows to compare each model's performance. The results should provide valuable insight into the influence of feature selection and dataset size on prediction accuracy, and confirm the most effective model for melting temperature prediction.

# 1 State of the art

In the last decades, the field of artificial intelligence (AI) has progressed at a blistering pace. One reason for this growth is the innumerable time- and cost-saving applications in all sectors.[1] The development of machine learning (ML) algorithms and techniques has recently yielded impactful results. Most people’s understanding of these advancements begins and ends with chatbots, planted in everyday search engines and apps, considered by the public as revolutionary, while barely understanding what machine learning truly enables, quietly, in fields most have never even heard of. Among the most astonishing of these specialized advancements are those revolutionizing scientific research particularly in biology and drug discovery. Notable examples include the development of AlphaFold-3 to predict protein structure and the Insilico AI platform[2], implemented to fully design a drug that entered human clinical trials. In chemistry, efforts are being made towards full in-silico laboratories, though it lies beyond our grasp for now. Nonetheless, advances have been pursued in exploring chemical space for de novo molecular design, synthesis pathway prediction, and molecular property prediction. The possible applications range from the medical to the industrial field; from new antibiotics to environmentally friendly solvents.

The ML algorithms employed in these endeavours are various, but can be categorized in two main groups: predictive and generative. Predictive models estimate an output after discerning a pattern from a large amount of input data. Generative models create new data based on patterns learned from training sets. This thesis focuses on the former, specifically various regression models and one neural network (NN) and their applicability in chemistry, in particular in relation to mixtures called Deep Eutectic Solvents (DES). These compounds have been of rising interest in the chemistry field, as an environmentally safe alternative in various industrial processes.[1]

The complexity and variability of DES mixtures, combined with the increasing demand for sustainable solvents, motivates the emerging interest in exploiting ML models to design, optimize and predict their properties. While

many experimental studies on DES exist, only a few recent papers include relevant computational approaches. Two research groups focused on two different aspects of DES in their investigation. Odegova et al.[3] aimed at computationally predicting DES properties to lessen the costly experimental work associated with these compounds, whose complexity stems from the presence of multiple components, requiring extensive trial and error empirical work to find the molar ratio that yields the lowest melting temperature, while also considering viscosity and density, that are influenced by the temperature itself. The research group of Luu et al.[4] addressed the need to discover novel mixtures with eutectic properties by developing generative models. They employed the latest algorithms and techniques, such as Diffusion models. Molecule generation is complex in itself, and even more so when dealing with binary or tertiary compounds. Luu et al.'s research is sophisticated and ambitious, though it results in complex and challenging model output evaluation.

In this thesis, the study by Odegova et al.[3] will serve as the foundational base for further investigation and refinement. The research group explored various predictive ML models, each one based on different algorithms, affecting the level of complexity that the models can capture. The key elements of datasize and feature selection of the ML pipeline will be further investigated using Odegova et al.[3]'s codebase. Luu et al.[4]'s research, in particular their data collection, will also partially serve the purpose of this work.

Predictive ML is still at its dawn. Research and development of algorithms and techniques in various fields will provide continuously improved and eventually reliable AI, as much as data collection will grant valid and enhanced training.

## 1.1 Deep Eutectic Solvents

Deep Eutectic Solvents are mixtures of two or more compounds. Their defining characteristic, responsible for the name itself, lies in the fact that at a specific molar ratio of the components, the melting point is lower compared to that of the components taken singularly. The nature of this behavior is given by the hydrogen bonds that form. Differently from Ionic Liquids, composed of ionic

compounds, DES are composed of a hydrogen bond donor and a hydrogen bond acceptor, varying in nature, composition, and structure. Due to their diversity, depending on their physicochemical composition, they are classified into five different groups, from type I to type V. Type I, II and III DES combine a quaternary ammonium salt with, respectively, a metal, a metal hydrate, and an organic compound. Type IV includes a metal halides and a hydrogen bond donor; while type V consists of non-ionic substances.[5] Type III and V DES are the most extensively investigated for being the most industrially relevant mixtures. DES are preferred to conventional solvents for properties such as biodegradability and low toxicity, while maintaining high extraction capabilities.[6]

## 1.2 Machine-Learning Basics

A common underlying pipeline underlies the implementation of machine learning algorithms. They are employed to predict a value after identifying a pattern that relates input data, known as features, to the target variable defined as label. The performance of an ML model in predicting the target variable, is highly dependable on the selected features that describe each sample, a selection defined as feature engineering. A well-structured dataset with sufficient, accurate, and appropriate information is needed. ML models are sensitive to the nature of data, therefore pre-processing is necessary to ensure compatibility. In the field of chemistry, this involves the correct translation of molecules and the atoms composing them into their defining properties. A molecule’s physicochemical properties have to be transformed into data, usually vectors, that can be encoded by a machine. This process produces data points, which are then decoded from vectoral value back to the comprehensible target feature.

Standard molecular representations fall into two categories: 2D and 3D features. 2D representation derive from molecule graphs with atoms as nodes and bonds as edges. SMILES (Simplified Molecular Input Line Entry System) are commonly used strings that encode the molecule’s atoms and bonds infor-

mation. 3D representations are derived from atomic coordinates and encoded into machine-readable input.[1]

The models need to interpret and relate the input variable, before making a prediction. These stages are known as training and testing sessions. During training ML models build connections between the variables and store them into a decision function that is employed in testing. Depending on how the dataset is learned they can be differently categorized. In supervised learning the model is trained on labeled data. In unsupervised learning the data is not labeled. Reinforcement learning uses a feedback mechanism to improve over time.

The models implemented by Odegova et al.[3] are based on supervised learning. During training the model looks for the pattern that better relate features to label, adjusting the function after each sample based on the discrepancy between the prediction and the actual label value. This process is referred to as model fitting. In the following testing phase, the model can only access the feature data and predicts the label with the learned pattern. If the model is unable to capture the complexity of the data, both training and testing metrics will be underperforming and the model is defined as underfitting. In the case of solely high training scores the model could be overfitting, meaning it knows the training data too well but it has low generalization capability, so it cannot predict unseen data. There needs to be a balance between the bias present during training and the variance observed in unseen data.

To improve model complexity some techniques are introduced, such as regularization, boosting and bagging. Regularization implies the introduction of a bias or penalty, to reduce overfitting and improve generalization. Boosting consists of combining simple models sequentially, each learning from the error of the previous one. Bagging applies the same logic, but the learning occurs in parallel and the final predictions are aggregated.[7]

Another important aspect to consider in machine learning is the configuration of the models settings, defined as hyperparameters. Lastly, the model's prediction capability has to be correctly evaluated as much during training to tune the parameters to their best combination, as for final model performance

assessment. The correct implementation of all the mentioned phases facilitates obtaining a right fitting model. [1]

## 2 Methods

### 2.1 Software environment and tools

The computational operations were conducted using a Linux-based operating system, employing the Bash shell in the command line. A specific Conda environment was created to manage dependencies and ensure functionality of Python packages.

The analysis was implemented in Python, selected for its widespread adoption among the scientific community, particularly in biology. Part of the development was executed in Jupyter Notebook, for interactive visualization and initial engagement with coding. This was crucial in the early stages of the work to gradually gain a certain level of proficiency and confidence.

The online platform GitHub was used throughout the project to facilitate tracking of code changes, to share progress with collaborators and to ensure having a backup of the work at all times.

### 2.2 Codebase

Odegova et al.[3] developed the open platform DESignSolvents in 2023 comprising nine machine learning models trained on an exhaustive DES dataset. Their comprehensive work included data collection, data analysis, feature selection, descriptor generation, splitting into train and test sets, feature normalization, model training and its evaluation.

The dataset was manually curated including all minimal and experimentally relevant data. The data analysis performed examined the DES class distribution, revealing an imbalanced dataset given by type III and V prevalence. The melting temperature distribution for each class was also examined and it did not follow a Gaussian pattern. The molecular descriptors for each component were obtained from their SMILES representation and include molecular

weight, potential toxicity, hydrogen bond donor count, and nine descriptors quantifying specific functional groups in the molecule. The dataset was split into training and testing ensuring that each data point was confined a single subset. Training and testing were iterated five times for each model, rotating sequentially the test set to perform cross-validation, fundamental for statistical relevance of the findings. Preprocessing involved a scaling step, for features to have values between 0 and 1, necessary to ensure robust performance across all models. Each model was first trained on every possible combination of hyperparameters, and with the generated information were assessed the results on model performance using statistical metrics.

To compensate for any limitations, the research group evaluated the results accordingly, as will be explained in more detail for the most critical elements of model training.[3]

### **2.2.1 Feature engineering**

The performance of machine learning algorithms in predicting the target variable is highly dependable on the selected features that describe each sample. Odegova et al.[3] carefully considered this when compiling the dataframe, implementing a series of measures to avoid miscalculations.

Each component was converted into a series of molecular descriptors derived from its SMILES representation and encoded into numerical features able to capture their physicochemical properties. This transformation was possible using RDKit, a cheminformatics toolkit. Publicly available descriptors were used to allow for more accessible reproducibility of results.

From hundreds of generated descriptors, only 20 were maintained following feature selection based on correlation matrices. To avoid redundancy, descriptors with high mutual correlation, that therefore contribute in the same way to the pattern estimation, were excluded. Their presence could prevent the models from capturing complex relationships and introduce unnecessary computational weight.

Following model training and testing the researchers retrieved and analyzed featured importance employing the framework SHAP (SHapley Additive

exPlanations)[8]. This method was proposed to aid in the correct interpretation of prediction by machine learning models. It is possible to obtain a quantitative importance value associated with each feature, revealing which ones contributed most significantly to the prediction. These results were presented averaged for all models, visually available in the article, and singularly for the best model, accessible in the specific notebook repository.

### 2.2.2 Datasets

All the available data was obtained from two data collections. The first, curated by Odegova et al.[3] contains 2,305 entries collected from articles published between 2003 and 2022. The other, compiled experimentally by Luu et al.[4] includes 402 entries. Both are available as CSV files and include the minimal information for each mixture: the molecular structures of the individual components in SMILES format, their respective molar ratio, singular component's melting temperature and system's melting temperature. Exclusively Odegova et al.[3] includes the chemical classification of the mixture, defined as "type of DES". An integrated dataset was compiled by Dalla Pozza[9]. The colleague removed repeated and invalid entries, obtaining a final dataset of 2,658 mixtures. The originating data entries were merged, including only shared attributes to avoid obtaining an incomplete dataset.

Odegova's original dataset and Dalla Pozza's expanded dataset were further manipulated for the models training. The minimal information in the datasets were expanded to include the molecular features from the SMILES of each component, obtaining dataframes complete with all selected descriptors contributing to model prediction. Conversely, redundant and obsolete columns were removed during preprocessing. The main difference lies in the "Type of DES" variable, which is present exclusively in Odegova et al.[3]'s original collection. For this variable, part of the pre-processing consisted in making this categorical variable compatible with machine learning algorithms. One-hot encoding was applied for the conversion, obtaining individual binary columns for each DES class, including the entry 'IL' for Ionic Liquids.

### 2.2.3 Evaluation Metrics

The performance of the models is evaluated through two metrics:  $R^2$  (Coefficient of Determination) and RMSE (Root Mean Squared Error).  $R^2$  measures how much better the model performs compared to simply predicting the mean value. It indicates the proportion of the variance in the target that can be explained by the input features, meaning it gives an idea of how well the model was able to capture the underlying relationships. RMSE quantifies model efficiency by measuring the average deviation of predicted versus actual values. Lower RMSEs correspond to higher prediction accuracy, as this reflects smaller errors in the model's output.

The model is trained on 80% of the data, validated on the remaining 20% and five-cross fold validation is performed. The metrics are calculated separately for the test set of data for each cross-validation fold (CV) and the train set of data for each fold (Train). The values reported as results are an integration of the metrics: the mean  $R^2$  and RMSE across all five folds. This repeated evaluation ensures statistical reliability, while maximizing the use of available data. The separation into five groups is efficiently performed using the "mixtures out" method to avoid data leakage from train to test. Since the mixtures include multiple components that could be repeated across entries, an indexing system is implemented to assign the same value to all mixtures with a common component. The data is then split ensuring that entries with the same component are in the same set. This strategy avoids overlapping between training and validation data, ensuring fair and reliable assessment of the models prediction accuracy. Through these measures, the results are consistent and more accurately reflect the models' true predictive capabilities.

The best-performing model is not selected on the sole best metric output. The researchers accounted for the imbalanced dataset and evaluated performance across all classifications of DES, choosing the model that performs consistently well across all categories, not just the frequent ones.

## 2.3 Machine Learning Models

The models compared by Odegova et al.[3] are nine. The first eight are various regression models of increasing complexity, while the last one is a neural network (NN). The python library SciKit-learn is mainly implemented. It contains various models with pre-written functions for training and testing.[7]

### 2.3.1 Training and Testing

Models using supervised learning are subjected to a training session, where the model has access to features and labels and it infers the pattern that best relates them. During this process of model fitting the formula can be adjusted after each iteration based on the error observed in the previous one. In SciKit-learn for each model the method `.fit` is used for training using as arguments the feature and label data. The parameter "sample weights" can optionally be included as an argument. It allows to determine the contribution of each sample to model learning, which is otherwise equal. Its implementation can be useful to correct imbalanced datasets. In the following session of testing, the model can only access the feature data and predicts the label with the pattern captured during the training. In SciKit-learn, `.predict` is used to output the predicted variable from solely features as input. Since some models can be sensitive to value range, it is good practice to normalize all data before training and testing, and subsequently reconvert the output. This is considered part of the preprocessing of data and the function `MinMaxScaler` from SciKit-learn is used. [7]

### 2.3.2 Hyperparameter optimization

Machine learning algorithms are composed of various modular parts. The tunable parameters, known as hyperparameters, can be considered as the settings of learning for a model. Depending on the model structure and components, the hyperparameters will be different. In the codebase, for each model two distinct blocks of code were executed: one for hyperparameter optimization and the other for performance evaluation. To identify the best combinations of hy-

perparameters a grid was created including all their possible combinations. For each of them training and testing was performed, including cross-validation. The best configuration yielding the highest average  $R^2$  value on the test set was selected and subsequently used for performance assessment. The final metric evaluations for both the training and testing sets were computed separately and presented in tabular form. A plot of predicted versus actual values including the regression line was also generated for visual support in the interpretation of performance. From the repository of Odegova et al.[3] can be inferred that hyperparameter optimization is not to be unquestionably followed, although it is not clear what criterion guided this group in the ultimate hyperparameter selection.

### 2.3.3 Models

A basic understanding of each model is crucial for an insightful interpretation of performance results.

The first model, Ridge Regression, is implemented to act as baseline, it is a variation of linear regression (LR), but it is referred to as LR nonetheless. The model introduces L2 regularization to linear regression, adding a penalty to avoid overfitting and improve generalization. The linearity constraint it is able to capture remains, therefore it is not really applicable for the more complex data being related.[7]

The next models are increasingly complex decision tree-based algorithms. Decision Tree models use flowchart-like structures to infer non-linear relationships between variables. They obtain a set of decision rules from the feature data and then follow the learned splits and paths to predict the label. In Decision Tree Regression (DTR) each leaf is a numeric value and it represents a cluster of observed data. It is prone to overfitting and unstable, meaning small changes in data can result in completely different trees. To overcome this limitation ensembles are implemented where the predictions of several base models are combined. Gradient Boosting Regression (GBR) combines the models sequentially, so a train and test error is produced at each iteration and implemented by the following decision tree for learning. The number of trees and contribu-

tion of each one are some of the tunable hyperparameters. Further developed versions of this model are Cat Boosting Regression (CBR) and Extreme Boosting Regression (XGB). The former supports categorical features and ordered boosting, training each model only on preceding data points to avoid label information leakage. The latter introduces regularization in the boosting process, introducing a penalty to the error of each sequential model being trained. Random Forest Regression (RFR) introduces the bagging method, it trains trees in parallel and aggregates their final predictions. The introduced randomness derives from bootstrap sampling, building the trees on a subsample that is randomly selected from the training dataset; and feature subsampling, considering a random subset of features at each tree split. [7]

The Support Vector Machine Regression model (SVM) builds the decision function based on a subset of training points, given by the ones excluded from the acceptable margin of error, so those whose prediction is far off. By not considering good predictions for the cost function, it avoids overfitting and is robust to noise.

K-nearest Neighbor Regression (KNN) is an instance-based learning model. It stores instances from data and predicts the label on a new point based on the mean of the labels of its nearest neighbors. Parameters that can be tuned are on the way to compute the distance between neighboring points and the number of neighbors the algorithm considers.

The last model is the only one based on more recent Deep Learning (DL) techniques. Multilayer Perceptron (MLP) is a neural network with some hidden layers, between an input and output layer. The input layer has a set of neurons representing the features, the hidden layers use activation functions to convey non-linear relationships, and the output layer returns a continuous value. To enhance prediction accuracy, backpropagation is implemented, with iterative training where errors from each round are used for improvement to adjust the model for the next iteration. The number of neurons, hidden layers and iterations, together with the type of activation function are modifiable hyperparameters. [7]

The novelty or complexity of a model should not be misinterpreted as a

synonym for improved prediction. The best algorithms depend on the data available and its nature, just as the predictive function can convey certain, but not all, associations between variables and labels. The yield relies on the correctness of the hyperparameters selection. There is not one model that outperforms the others, only one that demonstrates itself to be the optimal fit given the structure and content of the features and labels.

## 2.4 Workflow

In this thesis all modifications, analysis and model training is performed using the code deposited by the research group of Odegova et al.[3]. In the primary source the predicted labels include melting temperature, viscosity and density, all industrially important properties of DES mixtures. Nonetheless in this study the focus is solely on the prediction of melting temperature.

The aim of this work is to train and test the predefined machine learning models on the amplified dataset, evaluate their performance and compare it to the results obtained with the more limited original dataset. Therefore a series of modifications were performed on the original code, the original dataset and the novel dataset.

Firstly, the code was correctly executed on the original dataset, after proper adaptations to newer versions of dependencies to ensure compatibility. In addition to code maintenance, a file "Requirements" including all packages and versions needed to run it, was compiled. Finally, to facilitate result interpretation and comparison, the evaluation metrics from all models were integrated into a single DataFrame and exported as a CSV file.

Secondly, for the implementation of the various models on the novel dataset the adjustments consisted in dataset cleaning, dataframe adaptation, hyperparameters optimization and debugging.

**Dataset cleaning** - One of the files used to obtain an augmented dataset, contained five empty entries regarding the single components melting temperatures, returning a "Not a Number" error for most of the models. Since this feature is very relevant to the training, the empty cells

were manually filled with the correct values sourced from a separate file containing the same entries, but different dataframe, that included the missing values.

**Dataframe adaptation** - The larger dataset derived from integration lacks some columns, since only some of the features were present in both the smaller datasets used to build it. Most of the missing features were removed and not used in the models, with the exception of DES classification. All references to non-existing columns were removed to avoid errors. In particular, it was critical to eliminate the function that stored the results partitioned by DES type as well as the final visual plots generated based on this categorization.

**Hyperparameters optimization** - After execution of the code that outputs the optimal combination of hyperparameters, the corresponding input parameters were applied for the final evaluation of the models and plotting of the predicted versus actual values. Given the unawareness of an alternative criterion, no further critical adjustments were applied and the automated optimization outputs were implemented without modification.

Finally, the original code and dataset were further refined to allow for more precise comparison of the results. Since the amplified dataset lacked a feature, it seemed proper to also remove it from the original dataset, allowing for a more objective comparison of the results and further insight into the relevance of this particular feature to the models' training.

A substantial part of the computational work consisted in the conversion from a Jupyter Notebook format to Python scripts. For easier visualization of the complexity and significance of the lengthy code, each model was modularized into a function, including data preprocessing and graph visualization. In the module file `desmol.py`, all libraries are imported and each code block is stored as a function. In the file `main.py`, the module is imported and the needed functions are called in the desired order and the output plots, originally shown in jupyter notebook, are saved as PNG files, one for each fold for

each model with the optimized hyperparameters, along with a CSV table of  $R^2$  and RMSE outputs for both training and testing for each model. Through this implementation, the code can be executed as a whole, differently from the cell division structure of Notebooks. Moreover, optimized hyperparameters are optional arguments of the functions, facilitating their integration once the best combination has been identified. For completeness the more compact code for Data Analysis was converted, obtaining a complete scripted Python version of the code. The functions were added to the same module file and later called and executed from a new python script for both the original and amplified dataset.

### 3 Results

The evaluation metrics used to assess model performance are  $R^2$  and RMSE, consistent with the approach employed by Odegova et al.[3]. The evaluation encompasses training and testing of all models in three different dataset scenarios: the original dataset, the original dataset after removal of a feature, and the amplified dataset.

While cross-validation serves a good basis for statistical validation, additional reliability was ensured by repeating full model training and testing five times for each dataset configuration. This approach allows for calculation of mean ( $\mu$ ) and standard deviation ( $\sigma$ ) values derived from multiple runs. Variation across runs is expected, due to machine learning models including algorithm-specific randomness. These models are generally non-deterministic unless explicitly configured. In this implementation, randomness was partially contrasted using a fixed random seed in group-based data splitting, ensuring reproducible folds as long as the input remains constant.

RMSE (CV and Train) is reported for completeness, however in assessing model performance, the focus will be placed on  $R^2$  of the cross-validation testing set(CV). Metrics derived from the training set (Train) are not considered for comparative evaluation, but will be taken into account qualitatively to ensure that no overfitting occurs.

Firstly, the original dataset of 2,305 entries is used to validate the results of Odegova et al.[3] and their reproducibility.

Table 1: Original Performance Metrics

Name ML	R <sup>2</sup> CV	RMSE CV	R <sup>2</sup> Train	RMSE Train
LR	-1257.43	1735.46	0.543651	52.6733
DTR	0.580097	47.4616	0.703426	42.4362
RFR	0.729735	38.2291	0.918986	22.1485
GBR	0.738335	37.5928	0.956714	16.2013
CBR	0.756928	36.1843	0.953196	16.8572
XGB	0.707312	39.8644	0.999877	0.854109
SVM	0.704009	39.5265	0.788522	35.7411
KNN	0.66637	41.1073	0.921516	21.8243
MLP	0.617088	45.1355	0.757535	38.1508

Secondly, the same dataset is applied excluding the molecular feature Of DES classification.

Table 2: Original Performance Metrics Excluding Type of DES

Name ML	R <sup>2</sup> CV	RMSE CV	R <sup>2</sup> Train	RMSE Train
LR	-1226.46	1698.11	0.498506	55.2121
DTR	0.583633	47.3243	0.703946	42.3969
RFR	0.713469	39.311	0.918783	22.1787
GBR	0.733936	37.9431	0.947041	17.9158
CBR	0.757018	36.2145	0.970909	13.2903
XGB	0.733484	38.0284	0.996108	4.86269
SVM	0.629013	43.7242	0.73385	40.0113
KNN	0.642345	43.0502	0.8974	24.9557
MLP	0.469012	52.9626	0.606542	48.2267

Lastly, the amplified dataset including 2,658 entries by Dalla Pozza [9] is used to assess the models' susceptibility to datasize.

Table 3: Performance Metrics on the Amplified Dataset

Name ML	R <sup>2</sup> CV	RMSE CV	R <sup>2</sup> Train	RMSE Train
LR	0.449161	54.4788	0.568353	50.0968
DTR	0.652694	47.7102	0.74987	37.9342
RFR	0.785704	37.278	0.930924	19.9236
GBR	0.788667	36.6524	0.956177	15.8575
CBR	0.823381	33.8333	0.96524	14.1091
XGB	0.803071	35.6826	0.998849	2.57332
SVM	0.812928	34.8591	0.865566	27.7883
KNN	0.80552	35.3635	0.953392	16.3674
MLP	0.771632	38.4747	0.831733	31.0895

Table 4 summarizes the performance metrics across five repeated runs, by reporting the mean ( $\mu$ ) and standard deviation ( $\sigma$ ) of R<sup>2</sup> CV for each model, providing insight on average performance and consistency under repeated execution.

Table 4: Mean R<sup>2</sup> ( $\mu$ ) and Standard Deviation R<sup>2</sup> ( $\sigma$ ) across datasets

Model	Original		No Type		Amplified	
	$\mu$	$\sigma$	$\mu$	$\sigma$	$\mu$	$\sigma$
DTR	0.5836	0.0000	0.5836	0.0000	0.6527	0.0000
RFR	0.7314	0.0021	0.7153	0.0016	0.7873	0.0019
GBR	0.7383	0.0000	0.7243	0.0000	0.7887	0.0000
CBR	0.7343	0.0000	0.7468	0.0000	0.8234	0.0000
XGB	0.7073	0.0000	0.7328	0.0000	0.8031	0.0000
SVM	0.7040	0.0000	0.6290	0.0000	0.8129	0.0000
KNN	0.6664	0.0000	0.6241	0.0000	0.8055	0.0000
MLP	0.5961	0.0162	0.4192	0.0216	0.6565	0.0747

The highest variability is observed in the performance of the MLP model across runs. The model showed testing R<sup>2</sup> values ranging from 0.5665 to 0.8230 in the Amplified Dataset. The ample variation suggests higher sensitivity compared to the other models.

By comparing the model’s performance in the presence and absence of a feature, we are able to better grasp the importance of type of DES in the prediction by the models.

Tabella 1: Average  $R^2$  – Baseline vs. Removed feature

Model	Original	no Type of DES	$\Delta R^2$
LR	-1257.43	-1226.46	-30.9719
DTR	0.583633	0.583633	0.000000
RFR	0.731417	0.715286	-0.016131
GBR	0.738335	0.724271	-0.014064
CBR	0.734335	0.746789	0.012454
XGB	0.707312	0.732827	0.025515
SVM	0.704009	0.629013	-0.074996
KNR	0.66637	0.624124	-0.042246
MPR	0.596118	0.419198	-0.17692

The results show that MLP and SVM are the most influenced by this feature, MLP exceptionally.

As the ample dataset inherently lacks the DES classification feature, to assess the models performance we compare it with the results obtained from the original dataset with the same feature removed. This allows for a more objective evaluation of whether a benefit is brought by the increased training data available.

Tabella 2: Average  $R^2$  – Removed feature vs. Amplified Dataset

Model	no Type of DES	Amplified	$\Delta R^2$
LR	-1226.46	0.449161	1226.9
DTR	0.583633	0.652694	0.069061
RFR	0.715286	0.787328	0.072042
GBR	0.724271	0.788667	0.064396
CBR	0.746789	0.823381	0.076592
XGB	0.732827	0.803071	0.070244
SVM	0.629013	0.812928	0.183915
KNN	0.624124	0.80552	0.181396
MLP	0.419198	0.656549	0.237351

The results show an overall improvement for all models. SVM, KNN and MLP are the most noticeably improved.

For a final comparison across all datasets, we focus on the models that demonstrated the most distinctive behavior, excluding those that did not benefit nor were affected by the data manipulation.

Linear Regression (LR) and Decision Tree Regression (DTR) show poor performance, as they are not able to reproduce complex, non-linear relationships.

The models RFR, GBR, CBR perform well and demonstrate robustness across changes. Only CBR is included in the final comparison as it is the top performer among them. XGB will be excluded, even if it appears well-performing, due to a high concern of overfitting. SVM, KNN and MLP are included for their high sensitivity to changes in the data.

Table 7: Average  $R^2$  – Baseline vs. Removed feature vs. Amplified Dataset

Model	Original	No Type of DES	Amplified
LR	-1257.43	-1226.46	0.449161
CBR	0.734335	0.746789	0.823381
SVM	0.704009	0.629013	0.812928
KNN	0.66637	0.624124	0.80552
MLP	0.596118	0.419198	0.656549

RL is used as baseline, allowing us to appreciate the other models results. CBR retains the best metric values and consistently strong  $R^2$  that is stable across datasets. SVM, KNN are responsive to dataset changes and are subject to meaningful improvement with amplification. MLP is the most variable, most sensitive to feature removal and amplification.

## 4 Conclusion and discussion

The comprehensive work carried out by Odegova et al.[3] provided a solid and well-structured foundation for this study. The group accounted for all critical aspects of machine learning, from descriptor selection and dataset curation to rigorous preprocessing and evaluation. The robust codebase offered a reliable basis for subsequent targeted modifications and analysis, enabling this work to focus on and expand specific elements.

This study concentrated on two key aspects: the contribution of the single feature DES type to model performance, and the impact of dataset size on predictive accuracy.

The results indicate that the relevance of the DES type feature differs depending on the model. The kernel based models, SVM, KNN and MLP, were the most sensitive to this feature removal with a notable drop in performance.

On the other hand, CBR, the model identified as the top performing by Odegova et al.[3], is the least affected with an almost unchanged output. When performing the contribution estimation with SHAP values averaged across models, DES classification was not identified as a significant contributor in Odegova et al.[3]’s results. In particular when the same SHAP analysis was confined to the CBR model, Type V DES appeared in eighth position. While this might suggest some influence, its actual SHAP value was low, indicating limited impact despite ranking in the top ten most contributing features.

Changing the focus to the dataset size, the results confirm the expected improved performance across all models. Also in this case, the kernel models are deeply affected in their performance, more than with the feature removal. In the field of machine learning data abundance generally enhances model accuracy. Quantity is important for an increasingly better model’s performance, nonetheless quality is even more crucial. Elements such as proper data structure, complete and accurate information, thoughtful feature selection, and correct separation into training and testing subsets are all imperative for effective model training and reliable output.

Despite the expanded dataset, concerns about uneven data distribution persist. Although the additional entries lacked the DES class information, it is assumed that their inclusion did not attenuate the imbalance, leaving type III and V disproportionately represented. Classification being removed as a feature does not dismiss the concern. Each DES type is defined by chemical characteristics, so the information is indirectly encoded into the molecular descriptors. It cannot be excluded that the seemingly strong model performance is driven primarily by the overrepresented classes, while the model may struggle to capture relationships within underrepresented DES types. This highlights, once again, the importance of the dataset quality. A perfectly balanced dataset is not always achievable, therefore it is essential to apply pondered measures that account for the imbalance.

Although removing the DES type feature from both datasets allowed for a more objective comparison, the inability to analyze which DES classes contributed most to the positive metrics remains a significant limitation. Odegova et al.[3]

accounted for the dataset imbalance through retrospective analysis that cannot be replicated due to lack of data, so a direct comparison is not feasible. It is not possible to guarantee that the prediction is improved across all DES categories. The possibility remains that overrepresented classes drive the observed improvement.

The top performing model remains Cat Boosting Regression, even after removal of a feature and with an augmented dataset.

Table 8: Cat Boosting Regression

Dataset	R <sup>2</sup> CV	RMSE CV	R <sup>2</sup> Train	RMSE Train
Original	0.734335	37.922182	0.950760	17.290322
Removed Feature	0.746789	36.998356	0.961418	15.280578
Amplified	0.823381	33.833303	0.965240	14.109087

In all conditions, it yielded the highest average R<sup>2</sup> on the test subset and as this value rose sharply, the corresponding training increased mildly. This suggests that the model performance improved on unseen data points and has therefore good generalization capabilities. SVM and KNN are also worth mentioning for their great potential in future predictions with more training data available. The same cannot be said for Multilayer Perceptron despite its similarly better results. This model exhibited unstable behavior across runs, therefore its outputs are not considered robust enough for reliable conclusions. They may serve as a guideline, but further assessment and greater reproducibility is needed for validation.

Future contribution in the application of ML in the study of Deep Eutectic Solvents is possible. These include the analysis of other features to explore the significance of their contribution to the prediction, and obtaining even larger datasets to enhance further model precision. Completing the dataset with DES classification would be relevant for data analysis and to fit the models most affected by this feature. It would be insightful to address the dataset imbalance during training, possibly by implementing sample weights, rather than only subsequently in result analysis.

A solid foundation for ML implementation in DES has been laid both in the prediction and generative direction. Advancements in AI development, im-

proved molecule representation, and increased data collection will serve great progress in this field in time to come.

## Bibliography

- [1] Yun-Fei Shi et al. “Machine Learning for Chemistry: Basics and Applications”. In: *Engineering* 27 (2023). DOI: <https://doi.org/10.1016/j.eng.2023.04.013>.
- [2] F. Ren et al. “A small-molecule TNIK inhibitor targets fibrosis in pre-clinical and clinical models”. In: *Nature Biotechnology* 43 (2025). DOI: 10.1038/s41587-024-02143-0.
- [3] Valeria Odegova et al. “DESIGNsolvents: an open platform for the search and prediction of the physicochemical properties of deep eutectic solvents”. In: *Green Chem.* 26 (2024). DOI: 10.1039/D3GC04533A.
- [4] Rachel K. Luu, Marcin Wysokowski, and Markus J. Buehler. “Generative discovery of de novo chemical designs using diffusion modeling and transformer deep neural networks with application to deep eutectic solvents”. In: *Applied Physics Letters* 122 (2023). DOI: 10.1063/5.0155890.
- [5] Yuying Gao et al. “Deep eutectic solvent: Synthesis, classification, properties and application in macromolecular substances”. In: *International Journal of Biological Macromolecules* 278 (2024). DOI: <https://doi.org/10.1016/j.ijbiomac.2024.134593>.
- [6] Justyna Płotka-Wasyłka et al. “Deep eutectic solvents vs ionic liquids: Similarities and differences”. In: *Microchemical Journal* 159 (2020). DOI: <https://doi.org/10.1016/j.microc.2020.105539>.
- [7] Fabian Pedregosa et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830. DOI: 10.5555/1953048.2078195.
- [8] Scott M Lundberg and Su-In Lee. *A Unified Approach to Interpreting Model Predictions*. Vol. 30. Curran Associates, Inc., 2017.
- [9] Pietro Dalla Pozza. “Deep Eutectic Solvents e Intelligenza Artificiale: Approcci di Deep Learning per la Progettazione e la Caratterizzazione”. Supervised by Dr. Rampino Sergio, unpublished. Bachelor’s thesis. Università degli Studi di Padova, 2025.