Università degli Studi di Padova

Dipartimento di Biologia

Corso di Laurea Magistrale in Biotecnologie Industriali

Approcci di machine learning per predire profili di instabilità genomica in campioni tumorali

Relatore: Prof.ssa Chiara Romualdi

Dipartimento di Biologia

Controrelatore: Prof.ssa Laura Treu

Dipartimento di Biologia

Laureando: Marco Rota Negroni

Anno Accademico 2022/2023

Indice

Introduzione

Il cancro rimane una sfida critica per la salute globale essendo caratterizzato da una complessità molecolare che richiede un'approfondita analisi per svelarne gli intricati meccanismi sottostanti. L'instabilità cromosomica (CIN), una condizione in cui le cellule accumulano alterazioni strutturali e numeriche nei cromosomi, gioca un ruolo chiave nella genesi e nella progressione di molte forme di cancro. Questa instabilità cromosomica può agire da promotore, iniziando la tumorigenesi, promuovendo la diversità genetica e accelerando l'accumulo di mutazioni oncogeniche, ma allo stesso tempo i tumori possono indurre la CIN come risposta adattativa alle pressioni selettive del microambiente tumorale, agevolando l'eterogeneità tumorale e la resistenza ai farmaci. Essendo la CIN una caratteristica fondamentale del cancro, comprendere le sue basi molecolari nei tumori è essenziale per chiarire i meccanismi sottostanti del cancro ed esplorare possibili strategie terapeutiche.

L'idea di questo elaborato parte da un lavoro che è stato pubblicato lo scorso anno [3] dove vengono utilizzati dati di copy number di 33 tipi tumorali presi dal TCGA(The Cancer Genome Atlas) per andare a rilevare le CIN e come risultato ottengono 17 signature di instabilità genomica, ognuna con la sua prevalenza pan-cancer, la sua presunta causa e la confidenza con cui associano signature e cause e le caratteristiche. Questo lavoro di tesi adotta un approccio innovativo sfruttando il potere del machine learning per predire le signature identificate nell'articolo, utilizzando però i profili di metilazione ed espressione genica come fonti di dati alternative per migliorare la nostra comprensione della CIN nel cancro. L'obiettivo centrale è determinare se le firme CIN, precedentemente elucidate utilizzando dati di copy number, possa-

no essere replicate e validate utilizzando le alterazioni molecolari. La ricerca inizia proprio con la compilazione di un ricco dataset comprendente profili di metilazione ed espressione genica da diverse coorti di pazienti oncologici, riflettendo l'approccio pan-cancer, su 33 tipi tumorali del TCGA.
Per riuscire a soddisfare l'obbiettivo precedentemente menzionato sono stati utilizzati diversi modelli statistici per predire le signature di instabilità genomica (output del modello), e per farlo sfrutto il dataset con i profili di mediazione e di espressione genica (input del modello). Tutto il lavoro è stato svolto utilizzando il linguaggio di programmazione R.

Al fine di illustrare al meglio il lavoro svolto, questa tesi è stata strutturata come segue.
Nel primo capitolo viene data una panoramica sul machine learning e sui principali modelli predittivi di regressione che sono stati utilizzati.
Nel secondo capitolo è illustrato nel dettaglio tutto il lavoro di pre-processing dei dati di partenza che ha permesso di produrre il dataset finale che viene successivamente utilizzato per i modelli predittivi. Questo passaggio è stato fatto per garantire la qualità e l'idoneità del dato per le nostre analisi.
Nel terzo capitolo viene descritto come sono stati addestrati i modelli e l'ottimizzazione degli iperparametri.
Nel quarto capitolo vengono analizzati i risultati che ottengo dai modelli predittivi e anche le relative problematiche che si sono verificate, è il corpo principale del lavoro svolto. Le analisi vengono svolte solo su una signature.
Nel quinto capitolo sono analizzati invece i risultati delle ulteriori signature che sono state prese in considerazione.
Infine, dopo un confronto tra i risultati delle signature analizzate, è illustrato il risultato finale e i possibili sviluppi futuri per cercare di migliorare i modelli predittivi utilizzati.

Il nostro obiettivo finale è avere una nella comprensione migliore dei fattori molecolari associati all'instabilità cromosomica, aprendo potenzialmente la strada a strumenti diagnostici più accurati e strategie di trattamento personalizzate in grado di migliorare le prognosi dei pazienti.

Capitolo 1

Machine learning

L'obiettivo di questo capitolo è fornire una panoramica generale dei metodi del machine learning, concentrandosi principalmente sugli argomenti correlati al lavoro di tesi in esame. [1], [2]

1.1 Panoramica

Il machine learning, traducibile in italiano come "apprendimento automatico", è una branca dell'intelligenza artificiale (IA) che si concentra sullo sviluppo di algoritmi e modelli statistici che conferiscono ai computer la capacità di apprendere e migliorare le loro prestazioni su determinati compiti senza essere esplicitamente programmati per farlo. In altre parole, il machine learning si occupa dello sviluppo di algoritmi che consentono ai sistemi di apprendere informazioni dai dati disponibili e di predire nuove informazioni basate su tali apprendimenti senza la necessità di dipendere da istruzioni di programmazione tradizionali, in modo che la macchina riesca ad adattarsi a seconda dell'input che riceve.
Gli algoritmi di Machine Learning possono essere divisi in base al tipo di esperienza a cui sono sottoposti durante il processo di apprendimento. Questa divisione è fondamentale per comprendere come un algoritmo di machine learning impara dai dati e quale tipo di problemi è in grado di affrontare. Le principali categorie di apprendimento includono:

- Apprendimento Supervisionato: l'algoritmo viene addestrato utilizzando un insieme di dati di addestramento che includono sia le variabili di input che le corrispondenti etichette di output desiderate. L'obiettivo è far sì che l'algoritmo impari a mappare gli input alle etichette di output. Una volta addestrato, l'algoritmo può essere utilizzato per fare previsioni su nuovi dati.
- Apprendimento Non Supervisionato: In questo caso, l'algoritmo viene allenato su dati di addestramento che non includono etichette di output. L'obiettivo principale è scoprire schemi, strutture o cluster nei dati di input al fine di riprodurli o prevederli. Questo tipo di apprendimento è spesso utilizzato nell'analisi esplorativa dei dati.
- Apprendimento Semi-Supervisionato: In alcuni casi, è possibile che solo una parte dei dati di addestramento abbia una classe nota, mentre il resto è no. L'apprendimento semi-supervisionato cerca di combinare elementi degli approcci di apprendimento supervisionato e non supervisionato per sfruttare al meglio tutte le informazioni disponibili. Questo è utile quando l'annotazione dei dati può essere costosa o laboriosa.
- Apprendimento con Rinforzo: Questa categoria di apprendimento coinvolge un agente che prende decisioni in un ambiente per massimizzare una ricompensa cumulativa. L'agente apprende a compiere azioni attraverso la sperimentazione e l'interazione con l'ambiente. L'obiettivo è costruire un sistema che, attraverso le interazioni con l'ambiente, migliori le proprie performance. È spesso utilizzato in applicazioni di intelligenza artificiale per giochi, robotica e ottimizzazione.

Ogni tipo di apprendimento ha le sue applicazioni specifiche e i propri algoritmi associati. La scelta del tipo di apprendimento dipende dal problema che si sta cercando di risolvere e dalla disponibilità dei dati.

Nel contesto di questa tesi, ci concentriamo sulla prima categoria di algoritmi di machine learning, nota come apprendimento supervisionato. L'apprendimento supervisionato prende il nome dalla sua caratteristica principale: gli algoritmi apprendono sotto la guida e la supervisione dei dati di addestramento, i quali forniscono informazioni esplicite sulle risposte corrette associate a

ciascun input. In questo contesto, il modello di apprendimento supervisionato richiede un insieme di dati di addestramento come input, in cui la variabile di output nota è indicata come "y", mentre le variabili di input sono rappresentate come "x". L'obiettivo fondamentale è quello di addestrare il modello in modo che possa predire correttamente il valore di "y" in risposta alla sottomissione di nuovi dati di input.

A seconda del tipo di output atteso è possibile applicare un'ulteriore distinzione all'interno dei modelli di apprendimento supervisionato:

- **Classificazione**: l'output è discreto o categorico, ovvero assegnano un'etichetta di classe o una categoria a ciascun dato di input.
- **Regressione**: l'output è continuo o numerico.
- **clustering**: essi non producono una classificazione o una previsione specifica, ma invece raggruppano i dati in cluster o gruppi in base alla loro somiglianza. Questo tipo di output è utile per identificare pattern nascosti nei dati.

Come si può notare dalla differenza nel tipo di output prodotto, questi tre approcci vengono impiegati per scopi distinti nel contesto dell'apprendimento automatico. Nell'ambito della classificazione, gli output vengono categorizzati in due classi (binario) o più (multiclasse), e il sistema di apprendimento mira a sviluppare un modello in grado di assegnare gli input non osservati a una o più di queste classi. D'altra parte, nell'ambito della regressione, l'output è rappresentato da una variabile continua (dipendente), mentre input è rappresentato da variabili predittive (indipendenti). L'obiettivo principale è individuare e comprendere eventuali relazioni matematiche tra la variabile dipendente e le variabili indipendenti. Questo permette di effettuare previsioni precise su nuovi dati in cui la variabile di output è una quantità numerica continua.

Per quanto riguarda il nostro lavoro di ricerca, si è optato per l'utilizzo dei modelli di regressione in quanto l'output da prevedere è una signature numerica, e questo approccio risulta più adatto a soddisfare le specifiche esigenze del problema affrontato.

1.2 Modelli di regressione utilizzati

Come illustrato nel capitolo dedicato al machine learning (capitolo 1), il nostro approccio si basa sull'utilizzo di algoritmi di machine learning supervisionato, nello specifico modelli di regressione. La scelta dell'algoritmo di regressione è strettamente legata alle caratteristiche del dataset in esame e alle specifiche esigenze del problema. Nel nostro contesto, ci troviamo di fronte a un dataset, descritto dettagliatamente nel Capitolo 2.3.1, in cui il numero delle variabili esplicative è significativamente più elevato rispetto alle variabili di output. Pertanto, un modello lineare classico non risulta adeguato. Al suo posto, si è optato per l'utilizzo di due modelli lineari generalizzati (GLM), nello specifico Ridge e Lasso, insieme all'impiego di Random Forest, Stochastic Gradient Boosting (SGD Boosting) e, infine, Support Vector Machine (SVM). Dato che stiamo trattando modelli distinti, è essenziale condurre un'analisi dei dati e condurre test per determinare quale algoritmo di regressione offra le prestazioni migliori per ciascun compito specifico. In particolare, ciascuno dei quattro tipi di modelli selezionati è stato valutato per la previsione di ciascuna variabile di output, al fine di identificare quale di essi risulti più performante in ciascun caso. [5]

1.2.1 General Linear Model: lasso e ridge

Quando ci si trova di fronte a un elevato numero di variabili esplicative, i tradizionali modelli di regressione lineare possono diventare vulnerabili all'overfitting, un fenomeno in cui il modello si adatta eccessivamente ai dati di addestramento, perdendo la sua capacità di generalizzare su nuovi dati. In questo contesto, modelli come Ridge e Lasso, che appartengono alla famiglia della regressione lineare regolarizzata, emergono come soluzioni efficaci per mitigare questo problema. Ridge e Lasso introducono una penalizzazione sui coefficienti delle variabili indipendenti, limitando la loro crescita e riducendo la complessità del modello. Questo meccanismo aiuta a controllare l'overfitting. In particolare, Ridge implementa una penalizzazione L2, che mantiene tutte le variabili nel modello ma con coefficienti più ridotti. D'altra parte, Lasso

utilizza una penalizzazione L1, che può portare alla selezione automatica delle variabili più rilevanti, contribuendo alla "sparsità" del modello. Ciò significa che Lasso tende a forzare alcuni coefficienti dei predittori a diventare esattamente zero, eliminando così le variabili esplicative che non contribuiscono significativamente alla predizione della variabile target. Entrambi questi modelli consentono di regolare l'intensità della penalizzazione tramite il parametro di regolarizzazione, offrendo quindi un controllo flessibile sulla complessità del modello. Aumentando il valore del parametro di regolarizzazione, è possibile ottenere una maggiore riduzione dei coefficienti, conseguentemente riducendo la complessità del modello.
In sintesi, la scelta di utilizzare modelli di regressione Ridge e Lasso in un contesto con un elevato numero di variabili esplicative è giustificata dalla loro capacità di gestire problemi di overfitting, selezionare automaticamente le feature più rilevanti e stabilizzare i coefficienti del modello, contribuendo a migliorare le prestazioni di previsione su dataset complessi.

1.2.2 Random Forest

Il Random Forest rappresenta uno dei più diffusi e versatili algoritmi di machine learning di tipo supervisionato, essendo in grado di affrontare con successo sia problemi di classificazione che di regressione. Poiché l'algoritmo esegue la selezione delle feature in modo casuale in ogni albero e aggrega le previsioni da molti alberi, è in grado di gestire dati ad alta dimensionalità e di catturare relazioni non lineari complesse tra le variabili di input e l'output. Inoltre, è intrinsecamente robusto all'overfitting rispetto a un singolo albero di regressione. La natura casuale dell'aggregazione degli alberi e la media delle previsioni contribuiscono a ridurre la varianza complessiva del modello, evitando così un adattamento eccessivo ai dati di addestramento. Queste caratteristiche lo rendono un'opzione attraente per problemi di regressione in contesti con alta dimensionalità come nel nostro caso.

1.2.3 Stochastic Gradient Boosting, SGD Boosting

I principali vantaggi del Stochastic Gradient Boosting per la regressione includono la sua capacità di adattarsi in modo efficiente a dataset complessi caratterizzati da elevata dimensionalità. Il Boosting è una tecnica di ensemble learning in cui si combinano diversi modelli di machine learning più deboli (spesso chiamati "weak learners") per creare un modello forte. L'idea principale è addestrare iterativamente una sequenza di modelli deboli, assegnando loro pesi in modo che i modelli successivi si concentrino sul catturare gli errori residui del modello precedente. Questo processo iterativo consente di migliorare progressivamente le previsioni, riducendo l'errore residuo complessivo. Alla fine, i modelli deboli vengono combinati in un modello forte mediante una combinazione ponderata dei loro risultati. Questa modalità operativa permette al modello di contrastare l'overfitting, fornendo previsioni più robuste.
Anche se Stochastic Gradient Boosting non è specificamente progettato per la selezione delle feature, durante il processo di addestramento tende a attribuire maggiore peso alle feature più informative. Questo risultato conduce a una sorta di selezione delle feature "incorporata" nel modello, con le feature meno informative che ricevono un peso minore. Infine, anche da un punto di vista computazionale, la natura stocastica del processo di addestramento di SGD Boosting può renderlo più efficiente rispetto ad altri modelli basati su alberi decisionali, rendendolo particolarmente adatto per dataset di grandi dimensioni.
Tutte queste caratteristiche motivano la scelta nell'utilizzare il modello SGD Boosting per la regressione in ambienti con una vasta dimensionalità dei dati come nel caso in analisi.

1.2.4 Support Vector Machine, SVM

Anche se gli SVM sono generalmente associati alla classificazione, è importante notare che possono essere efficacemente impiegati anche per la regressione. Questi modelli mostrano una notevole robustezza nei confronti del rumore e sono in grado di gestire dataset caratterizzati da un elevato numero di variabili

esplicative. In termini generali, gli SVM cercano di trovare l'iperpiano migliore che permette di separare i dati sfruttando una funzione matematica. Nel mio studio, ho esplorato due tipologie di SVM: il modello lineare e quello radiale (RBF). Entrambi sono stati utilizzati per catturare relazioni intricate tra le variabili di input e l'output. Un SVM lineare rappresenta una solida opzione quando si lavora con dati più semplici o quando è importante ottenere un modello interpretabile. Al contrario, un SVM RBF si dimostra più adatto per dati caratterizzati da relazioni complesse o quando si desidera massimizzare la flessibilità del modello. Quest'ultima variante è incredibilmente flessibile e può acquisire una profonda comprensione delle relazioni complesse tra variabili esplicative e variabile target. Inoltre, il SVM RBF è in grado di gestire dataset sbilanciati, come nel caso della mia analisi, in cui il numero di campioni per la variabile y è limitato rispetto alle variabili esplicative.

Complessivamente, la decisione di impiegare gli SVM per la regressione in un contesto caratterizzato da un elevato numero di variabili esplicative è ampiamente giustificata dalla loro abilità nella gestione di dataset complessi e ad alta dimensionalità, dalla loro capacità di controllo dell'overfitting e dalla loro robustezza in situazioni con dati complessi.

Poiché non era noto a priori quale dei due modelli fosse più adatto al dataset in esame sono stati utilizzati entrambi, anche tenendo conto della loro efficienza computazionale.

Capitolo 2

Pre-process dei dati

Nell'era digitale in cui viviamo, la disponibilità di dati è diventata una risorsa inestimabile e in questo scenario, l'analisi dei dati e l'apprendimento automatico si sono affermati come strumenti di fondamentale importanza per estrarre conoscenze utili e prendere decisioni informate. Tuttavia, la possibilità di attingere delle informazioni significative da questa grande quantità di dati è condizionata dalla necessità di trattare e preparare correttamente tali dati. Per questo motivo, il pre-processamento dei dati svolge un ruolo cruciale. Esso rappresenta il punto di partenza su cui si costruisce il successo di qualsiasi progetto che coinvolga l'utilizzo di dati. Possiamo pensare ai dati come la materia prima su cui costruiamo i nostri modelli, e come si può ben intuire, la qualità della materia prima influisce direttamente sulla qualità del prodotto finito. Nel contesto dell'apprendimento automatico, questo significa che il pre-processamento dei dati è essenziale per garantire che i dati siano adeguati allo scopo previsto e che i modelli addestrati su questi dati siano in grado di produrre risultati affidabili e generalizzabili.

Una delle ragioni fondamentali per cui il pre-processamento dei dati è fondamentale è la presenza di rumore. Nel mondo reale, i dati raccolti possono essere soggetti a errori di misurazione, mancanza di coerenza o addirittura possono contenere dati fuorvianti o danneggiati. Eliminare il rumore e gli errori è una priorità durante la fase di pre-processamento, poiché questi elementi possono gravemente compromettere l'accuratezza e la validità delle analisi e

dei modelli di machine learning.

Un altro aspetto cruciale è la gestione dei dati mancanti, una sfida comune in molte fonti di dati. La mancanza di informazioni può portare a distorsioni nei risultati dell'analisi o nell'addestramento dei modelli. Nel pre-processamento dei dati, vengono adottate diverse strategie per affrontare questa problematica, tra cui l'imputazione dei valori mancanti o l'esclusione delle osservazioni incomplete, assicurando così una base dati completa e affidabile. Inoltre, bisogna ricordarsi che i diversi algoritmi di machine learning hanno requisiti specifici per quanto riguarda la forma e le caratteristiche dei dati in ingresso. Ad esempio, alcuni algoritmi richiedono dati normalizzati o standardizzati per rendere i dati comparabili tra loro e facilitare l'interpretazione dei coefficienti nei modelli, mentre altri possono richiedere che le features abbiano scale simili. Soddisfare tali requisiti consente l'applicazione efficace degli algoritmi di machine learning oltre che ottimizzare le loro performance.

Un ulteriore punto di riflessione riguarda la dimensione del dato di partenza. Spesso, i dati possono presentare un'elevata dimensionalità, ossia un gran numero di features, alcune delle quali possono risultare irrilevanti o ridondanti per il problema in analisi. L'addestramento di modelli su dati ad alta dimensionalità può comportare notevoli costi computazionali e può generare problemi di overfitting. Di conseguenza, il pre-processamento dei dati si occupa anche di selezionare solo le features più informative, contribuendo a ridurre la dimensionalità e a migliorare la capacità predittiva dei modelli. In questo modo riesco sia a ridurre i tempi di calcolo del modello che aumentarne le performance.

In conclusione, il pre-processamento dei dati è una fase imprescindibile nell'ambito del machine learning. La sua importanza risiede nel fatto che incide direttamente sulla qualità e sull'affidabilità dei risultati ottenuti. Un pre-processamento accurato dei dati non solo migliora la precisione dei modelli, ma rende anche l'intero processo più efficiente, interpretabile e adatto a supportare decisioni informate. In mancanza di una corretta pulizia e preparazione dei dati, i modelli di machine learning rischiano di essere negativamente influenzati da imperfezioni o da dati di scarsa qualità, compromettendo il valore delle analisi effettuate. Pertanto, dedicare tempo ed energia al pre-processamento dei

dati è un investimento fondamentale per chiunque desideri ottenere risultati affidabili e significativi nell'analisi dei dati e nell'apprendimento automatico.

2.1 Acquisizione dei dati

I dati di metilazione e di espressione sono stati entrambi scaricati dal portale TCGA grazie al pacchetto *curatedTCGAData*. Per quanto riguarda l'espressione i dati provengono da esperimenti di RNASeq2GeneNorm e presentano le seguenti dimensioni:

Cancer type	n°campioni	Cancer type	n°campioni	Cancer type	n°campioni
ACC	79	KIRC	606	PRAD	550
BLCA	427	KIRP	326	READ	105
BRCA	1212	LAML	173	SARC	265
CESC	309	LGG	530	SKCM	473
CHOL	45	LIHC	423	STAD	450
COAD	326	LUAD	576	TGCT	139
DLBC	48	LUSC	552	THCA	568
ESCA	196	MESO	87	THYM	122
GBM	166	OV	307	UCEC	380
HNSC	566	PAAD	183	UCS	57
KICH	91	PCPG	187	UVM	80

Tabella 2.1: dati espressione

Tutti presentano 20501 geni con i relativi valori di espressione per un totale di 10601 campioni.

Per quanto riguarda i dati di metilazione invece alcuni tipi tumorali presentavano due tipi di file: uno relativo alla piattaforma di metilazione 450 e l'altro relativo alla piattaforma di metilazione 27. La scelta di un file rispetto all'altro è stata fatta considerando il maggior numero di campioni. In particolare è stata scelta la piattaforma 450 per i tipi tumorali colorati in rosso nella tabella 2.2, mentre in blu sono i tipi tumorali in cui è stata scelta la piattaforma 27. Per tutti i restanti non è specificata quale delle due, ma era presente un unico file per la metilazione.

Cancer type	n°campioni	Cancer type	n°campioni	Cancer type	n°campioni
ACC	80	KIRC	480	PRAD	549
BLCA	434	KIRP	321	READ	106
BRCA	885	LAML	194	SARC	269
CESC	312	LGG	530	SKCM	475
CHOL	45	LIHC	429	STAD	397
COAD	333	LUAD	492	TGCT	139
DLBC	48	LUSC	412	THCA	567
ESCA	202	MESO	87	THYM	126
GBM	285	OV	591	UCEC	466
HNSC	580	PAAD	195	UCS	57
KICH	66	PCPG	187	UVM	80

Tabella 2.2: dati metilazione

Nel caso dei dati di metilazione tutti i campioni presentano 24526 geni ad eccezione dei due tipi tumorali in cui è stato scelto il file della piattaforma 27 dove ho solamente 13683 geni, per un totale di 10419 campioni.

Per quanto riguarda invece la matrice con i dati relativi alle signature che hanno ottenuto nel lavoro di riferimento [3] sono stati scaricarti da https://github.com/markowetzlab/Drews2022_CIN_Compendium seguendo la seguente directory: Section 7 Identification of putative signature aetiologies/Section 7.2 Heatmap activity by cancer type/input. Qui si trova anche l'informazione relativa al metadata.

2.2 Filtraggio dei dati

Dopo aver scaricato i dati inizia la fase di pre-processamento. La prima cosa che è stata fatta è il filtraggio. Questa fase di filtraggio dei dati mira a tenere solo i campioni che hanno come *tissue source* site il codice identificativo 01, ovvero siano un tumore solido primario. Questa operazione è stata fatta per tipo tumorale individualmente, sia per la matrice di espressione che di metilazione, ed entrambe sono state trasposte in modo da avere il nome del gene sulle colonne e l'identificativo dei campioni sulle righe. Effettuando questa

operazione, è stato completamente escluso un tipo di tumore dall'analisi, la leucemia mieloide acuta (LAML). Ciò è dovuto al fatto che tutti i campioni relativi a questa tipologia presentavano il codice identificativo 03, indicando che si tratta di campioni di cancro primario derivati dal sangue o di campioni di sangue periferico e noi siamo interessati esclusivamente ai campioni di tumore solido primario.
Sulla matrice di espressione è stato fatto il log delle conte per avere un dato con un range di valori meno ampio e quindi di conseguenza più trattabile. Mentre sulla matrice di metilazione sono state ridotte le colonne (geni) in modo da avere tutte le variabili uniformi per ogni campione tumorale. Ciò è stato fatto basandomi sul numero di variabili inferiore che le matrici dei tipi tumorali GBM e OV presentano. In seguito, sono stati tenuti solo i campioni che hanno sia il dato di espressione che di metilazione. Alla fine, ho una matrice con 8292 campioni e 34184 geni (20501 quantificati per espressione e 13683 per metilazione).
Dopo aver ordinato le righe della matrice in ordine alfabetico posso andare a filtrarle per le righe che sono presenti nella matrice con le signature del lavoro di riferimento [3] in modo da avere solamente i campioni che hanno la signature da predire. In questo modo riduco i miei campioni a 4593 e la matrice risultante è 4593 x 34184. Infine, vado a rendere unici i nomi delle colonne per poterli distinguere a posteriori se si tratta di un gene che corrisponde all'espressione o alla metilazione, in particolare i geni di metilazione che sono presenti anche come geni di espressione sono seguiti da ".1".
I sottocapitoli che seguono entrano nello specifico dei passaggi di pre-processamento che sono stati effettuati sulla matrice che è stata sopra descritta.

2.2.1 Scaling

Scalare i dati provenienti da diverse fonti prima di confrontarli è una pratica importante per garantire che i confronti siano accurati, significativi e affidabili, riducendo al contempo l'effetto di potenziali bias e differenze nelle scale di misurazione come nel nostro caso. Infatti, nella matrice ho due tipologie di

dati differenti: dati di espressione e dati di metilazione. I valori di espressione sono valori di TPM che hanno range ampi, mentre per i dati di metilazione ho il relativo valore β, dove ho un range da 0-1. Si capisce subito quindi che è necessario andare a scalare i dati per averli entrambi sulla stessa scala. Per fare ciò mi basta fornire la matrice alla funzione "*scale*", dove vado a settare "center= FALSE", per evitare di avere più della metà dei miei valori negativi.

2.2.2 Coefficiente di variazione

Il coefficiente di variazione (CV) è una misura statistica che esprime la variabilità relativa di un insieme di dati rispetto alla loro media. È definito come il rapporto tra la deviazione standard e la media dei dati ed è spesso espresso come percentuale per facilitarne la comprensione anche se nel nostro caso non andremo ad esprimerla in percentuale. La formula del coefficiente di variazione è la seguente ed è definita solo per $\mu > 0$:

$$CV = \left(\frac{\sigma}{\mu}\right) \times 100$$

Dove:

- CV è il coefficiente di variazione.
- σ rappresenta la deviazione standard dei dati.
- μ è la media dei dati.

L'importanza del coefficiente di variazione durante il pre-processamento dei dati è legata alla sua capacità di misurare la variabilità relativa dei dati. Ovvero il CV è utile per valutare quanto i dati siano dispersi intorno alla loro media. Un CV elevato indica una maggiore variabilità relativa alla media, mentre un CV basso indica una minore variabilità relativa alla media. Questa informazione è preziosa nel comprendere la distribuzione dei dati e può essere utile per identificare dati che potrebbero essere molto eterogenei o presentare discrepanze significative. In questo passaggio il mio scopo è quello di andare a ridurre il numero di features con l'idea che ciò possa aiutare a semplificare il modello e a migliorare la sua capacità predittiva.

Visto che la matrice di partenza ottenuta in 2.2.1 ha due tipologie di dati diversi, è stato prima calcolato il CV su tutti i geni, sono stati rimossi eventuali NA e dopodiché ho separato i CV relativi all'espressione e alla metilazione. Nelle immagini A.1 e A.2, possiamo osservare le distribuzioni dei CV relativi all'espressione e alla metilazione. È importante notare che questi CV si trovano su scale diverse. Il CV dell'espressione varia da 0 a 60, mentre il CV della metilazione ha un range da 0 a 2. Pertanto, è fondamentale rimuovere il 20% dei CV più bassi separatamente per l'espressione e la metilazione. La linea nera tratteggiata, presente in entrambe le immagini A.1 e A.2, permette di visualizzare la porzione di dati che sono stati rimossi, i quali corrispondono alla parte destra della divisione. Cosi facendo vado a rimuovere:

- 4070 geni di espressione.
- 2737 geni di metilazione.
- 152 NA rimossi.

Con i dati ottenuti vado a filtrare le colonne della matrice di partenza separatamente per espressione e metilazione in modo da tenere solo i geni che soddisfano il cutoff che ho attuato sul CV. Così facendo ottengo una matrice 27225 x 4593, ho ridotto la dimensionalità della matrice di circa 7 mila geni.

2.2.3 Predittori con varianza vicino allo zero, NZV

La rimozione delle variabili con bassa varianza, spesso indicate come "near zero variance" (NZV), durante il pre-processamento dei dati è una pratica utile in molte situazioni. Ad esempio, la riduzione della complessità è un obiettivo chiave, in quanto le variabili con una varianza molto bassa tendono a contenere un potere predittivo limitato poiché mostrano cambiamenti minimi o addirittura nulli tra i campioni. Pertanto, mantenerle potrebbe risultare superfluo o addirittura dannoso per l'efficacia dell'analisi. Inoltre, la semplificazione della complessità computazionale è particolarmente preziosa in contesti in cui affrontiamo dataset di grandi dimensioni, consentendo di risparmiare notevolmente tempo e risorse nell'addestramento del modello.
Infine, l'eliminazione delle variabili a varianza molto bassa può contribuire

a migliorare le prestazioni del modello di machine learning, consentendo a quest'ultimo di concentrarsi su caratteristiche più informative e rilevanti per ottenere previsioni accurate.

Per identificare questi tipi di predittori è stata utilizzata la funzione "*nearZeroVar*" del pacchetto *caret* sulla matrice ottenuta in 2.2.2, la quale permette di calcolare due metriche:

1. Rapporto di frequenza –> La frequenza del valore più prevalente rispetto al secondo valore più frequente, che sarebbe vicino a uno per i predittori buoni e molto grande per i dati altamente sbilanciati.
2. Percentuale di valori univoci –> Il numero di valori univoci diviso per il numero totale di campioni (*100) che si avvicina a zero all'aumentare della granularità dei dati.

Noi siamo interessati al secondo e vado a settare come cutoff 40% di valori unici, quindi vado a rimuovere tutti i geni che presentano più del 40% di valori che sono zero. Nell'immagine 2.1 possiamo vedere l'output che ci restituisce la funzione e quelli che soddisfano la condizione vengono dati come "TRUE" nella quarta colonna. 2315 geni soddisfano il requisito impostato e controllando la presenza di NA ne trovo 474. Vado così a rimuovere 2789 geni ottenendo una matrice 24436 x 4593.

	freqRatio	percentUnique	zeroVar	nzv
A1BG	1.000000	99.804050	FALSE	FALSE
A1CF	949.000000	37.470063	FALSE	TRUE
A2BP1	463.750000	58.110168	FALSE	FALSE
A2LD1	1.000000	99.956455	FALSE	FALSE
A2ML1	310.666667	78.684955	FALSE	FALSE
A2M	1.000000	100.000000	FALSE	FALSE
A4GALT	1.000000	99.891139	FALSE	FALSE
A4GNT	423.500000	59.372959	FALSE	FALSE
AAA1	1084.666667	27.824951	FALSE	TRUE
AACSL	457.500000	57.827128	FALSE	FALSE
AACS	1.000000	99.912911	FALSE	FALSE
AADACL2	1116.000000	26.518615	FALSE	TRUE
AADACL3	943.000000	17.483126	FALSE	TRUE

Figura 2.1: nearZeroVar output

2.2.4 Predittori correlati

L'individuazione dei predittori correlati riveste un ruolo fondamentale nell'ottimizzazione della qualità dei dati. Tale processo consente l'eliminazione di informazioni duplicate o ridondanti, entrambe potenzialmente dannose per l'addestramento dei modelli. In particolare, la presenza di dati duplicati può comportare un aumento della complessità del modello, con conseguenti complicazioni nella sua interpretazione e nell'efficienza computazionale. Inoltre, questa categoria di dati può contribuire all'insorgere dell'overfitting del modello. Pertanto, l'identificazione e la successiva eliminazione dei predittori correlati si pone come un tassello essenziale nel processo di miglioramento complessivo della capacità predittiva e della robustezza del modello.

Per identificare questi tipi di predittori è stata utilizzata la funzione "*findCorrelation*" del pacchetto *caret*, la quale permette di identificare i predittori correlati con un determinato cutoff, nel nostro caso tutti i predittori con una correlazione superiore al 90%, sia negativa che positiva. La funzione richiede la matrice di correlazione, la quale è ottenuta utilizzando la funzione "*cor*" sulla matrice che ho ottenuto nel 2.2.3. Così facendo trovo 2647 geni che hanno una correlazione maggiore del 90% in valore assoluto, in questo modo ottengo una matrice finale 21789 x 4593.

2.3 Dataset finale

Dopo aver esaminato attentamente tutti i passaggi di pre-processamento descritti nel capitolo 2.2, siamo stati in grado di ridurre significativamente il numero di predittori nella matrice iniziale. Questa ottimizzazione ha reso la matrice più adatta dal punto di vista computazionale per i modelli di machine learning che verranno applicati in seguito. In particolare, sono stati esclusi i seguenti predittori:

- 6959 con il CV.
- 2789 con la NZV.
- 2647 con la correlazione.

Per un totale di 12395 geni e se andiamo a vedere nello specifico tra espressione e metilazione sono stati eliminati:

- 6725 geni di espressione.
- 5670 geni di metilazione.

2.3.1 Inclusione di variabili cliniche nel dataset

Nel nostro studio, abbiamo riconosciuto l'importanza di arricchire il dataset con variabili cliniche al fine di potenziare l'efficacia dei modelli di machine learning. Questo arricchimento del dataset con informazioni cliniche può contribuire in modo significativo all'eliminazione di possibili bias o confondenti che potrebbero influenzare negativamente le previsioni dei modelli. In particolare, sono state introdotte tre variabili cliniche chiave: l'età del paziente, il tipo tumorale e la purezza del campione. Per purezza del campione tumorale ci si riferisce alla percentuale di cellule tumorali rispetto al totale delle cellule nel campione.

Per incorporare la purezza del campione e il tipo tumorale nel dataset, sono stati recuperati i dati dall'informazione del metadata precedentemente scaricata 2.1. Questi dati sono stati aggiunti come colonne all'inizio della nostra matrice dei dati 2.2.4. Successivamente, è stata condotta una verifica per assicurarsi di avere le informazioni complete per tutti i 4593 campioni iniziali. Sono stati identificati solo 3 valori mancanti (NA), che rappresentano una percentuale trascurabile, e quindi sono state rimosse le corrispondenti righe, portando il numero totale di campioni a 4590.

Per acquisire l'informazione sull'età dei pazienti, è stato utilizzato il pacchetto *TCGAutils* e, in particolare, la funzione "*getClinicalNames*". Questa funzione permette di ottenere informazioni cliniche, inclusa l'età espressa in "years to birth". Insieme è stata anche recuperata l'informazione del barcode del paziente, che consente di associare correttamente l'età a ciascun paziente. Come per la purezza del campione, è stata condotta una verifica per assicurarsi che tutte le 4590 osservazioni contenessero l'informazione sull'età e sono stati riscontrati 153 NA. In questo caso prima di rimuoverli tutti è stato valutato

a che tipo tumorale appartengono, perché si vuole evitare di perdere delle informazioni significative per i modelli di machine learning. Di questi 153 NA trovo che:

Cancer type	n°campioni	Cancer type	n°campioni	Cancer type	n°campioni
BLCA	1	LGG	1	PRAD	5
BRCA	10	LUAD	12	SARC	1
CESC	2	LUSC	6	STAD	6
KIRP	2	OV	8	UCEC	96
LIHC	3				

Tabella 2.3: Tipi tumorali degli NA

Analizzando attentamente la Tabella 2.3, emerge chiaramente che l'eliminazione di tutti e 153 i dati mancanti (NA) comporterebbe una significativa perdita di informazioni, soprattutto legata al tipo tumorale UCEC, mettendo a rischio l'integrità dell'analisi pan-cancer. Per limitare tale problema, si è cercato di recuperare le informazioni relative ai 96 campioni UCEC, sfruttando il pacchetto *TCGAbiolinks*. Attraverso il codice di seguito riportato, è possibile recuperare tutte le informazioni cliniche supplementari specifiche per il tipo tumorale UCEC. Tra le informazioni presenti, sono state estratte: "birth_days_to" e il "bcr_patient_barcode". Per ottenere l'età in anni, viene effettuata un'apposita trasformazione, calcolando il valore assoluto dei giorni dalla nascita (che è negativo) e dividendo il risultato per 365, arrotondando il valore a un numero intero.

```
query <- GDCquery(
  project = "TCGA-UCEC",
  data.category = "Clinical",
  data.type = "Clinical Supplement",
  data.format = "BCR Biotab")
GDCdownload(query)
clinical_tab_all <- GDCprepare(query)
```

In questo modo, è stato possibile recuperare le informazioni per tutti i 96 campioni, procedendo successivamente all'eliminazione delle righe corrispondenti

ai rimanenti 57 valori mancanti relativi ad altri tipi tumorali. Tale procedura ha permesso di limitare significativamente la perdita di informazioni che si sarebbe verificata rimuovendo tutti e 157 i campioni associati a dati mancanti. Complessivamente, eliminando i campioni privi di dati relativi alla purezza e all'età, ho rimosso 60 campioni dalla mia analisi. Di conseguenza, la matrice risultante ha dimensioni 4533 x 21789. Infine, ho applicato la funzione "*scale*" a questa matrice, con l'opzione "center=FALSE", adottando lo stesso ragionamento esposto in precedenza in 2.2.1. Per eseguire questa operazione, inizialmente escludo la colonna relativa al tipo tumorale e la raggiungo successivamente dopo aver eseguito la scalatura dei dati.

2.4 Analisi signature

Nel dataset così preparato, il passo successivo consiste nell'aggiungere la variabile da predire, la "y" dei modelli e questa variabile viene inserita come prima colonna per comodità nell'analisi. Tuttavia, prima di procedere, è necessario determinare quale delle 17 signature identificate nel lavoro [3] siano interessanti da predire. I dati relativi a queste signature sono stati acquisiti come descritto nel Capitolo 2.1, risultando in un totale di 6335 campioni, che superano il numero di campioni del nostro dataset. Di conseguenza, è stata eseguita un'operazione di filtraggio per ottenere una matrice contenente tutte e 17 le signature (come colonne) e nelle righe solamente i campioni presenti nel dataset. La matrice così ottenuta può essere analizzata per determinare quali delle 17 signature siano rilevanti per gli obiettivi di questa tesi. Ciascuna signature ha un valore compreso tra 0 e 1 e rappresenta l'attività delle signature.

Un primo approccio consiste nell'effettuare una rappresentazione grafica sotto forma di heatmap della matrice, che consente di visualizzare in modo immediato quali signature presentino valori elevati o bassi. Nella Figura 2.2, è possibile osservare il l'heatmap risultante, dove i toni più scuri di blu indicano valori prossimi a uno, mentre i colori più chiari denotano valori prossimi allo zero. È immediatamente evidente come la maggior parte delle signature mo-

strano valori che si avvicinano allo zero, in particolare quelle situate nel centro della heatmap. Tuttavia, emerge una situazione differente per la signature CX1, posizionata più a destra, la quale mostra valori tendenti verso uno ed è la più attiva tra le 17 signature considerate. È interessante notare che anche le signature CX3, CX5 e CX2 presentano un numero significativo di valori vicini a uno in confronto alle altre.

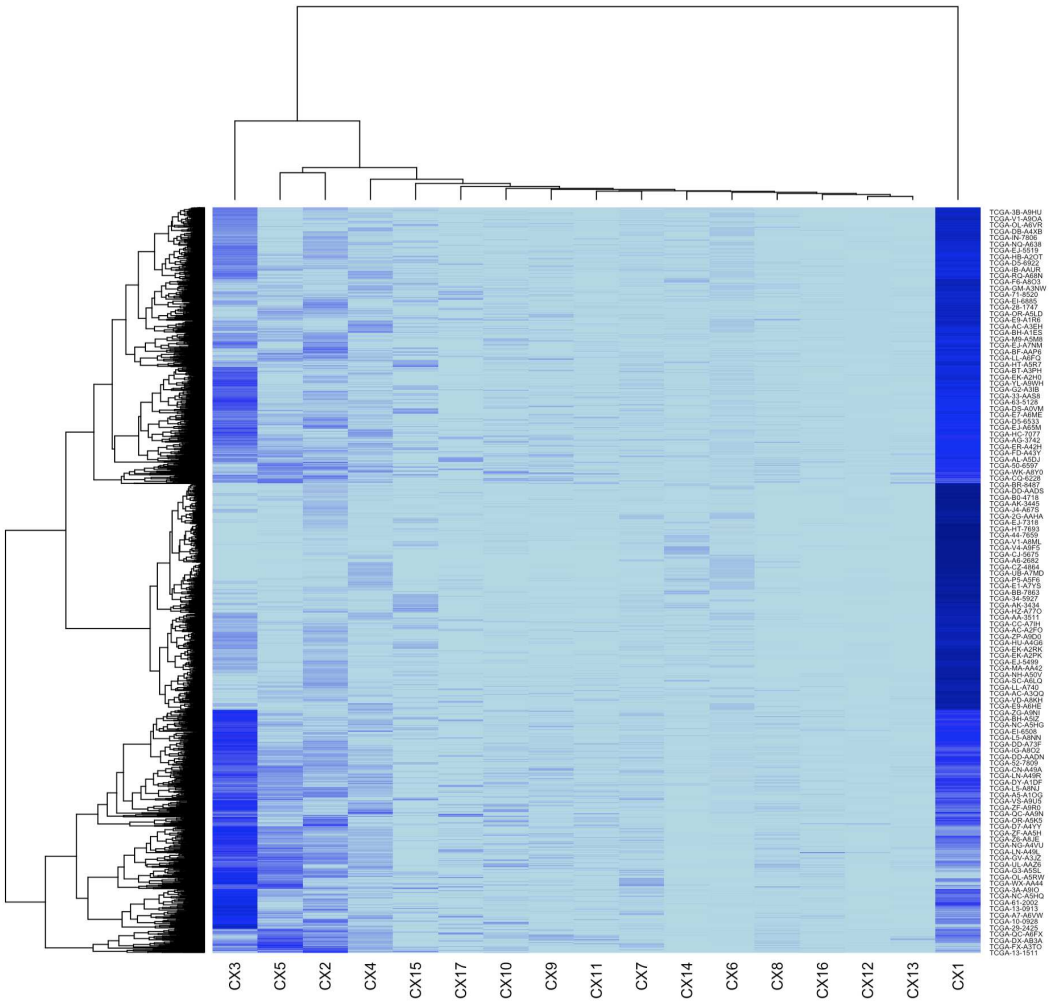


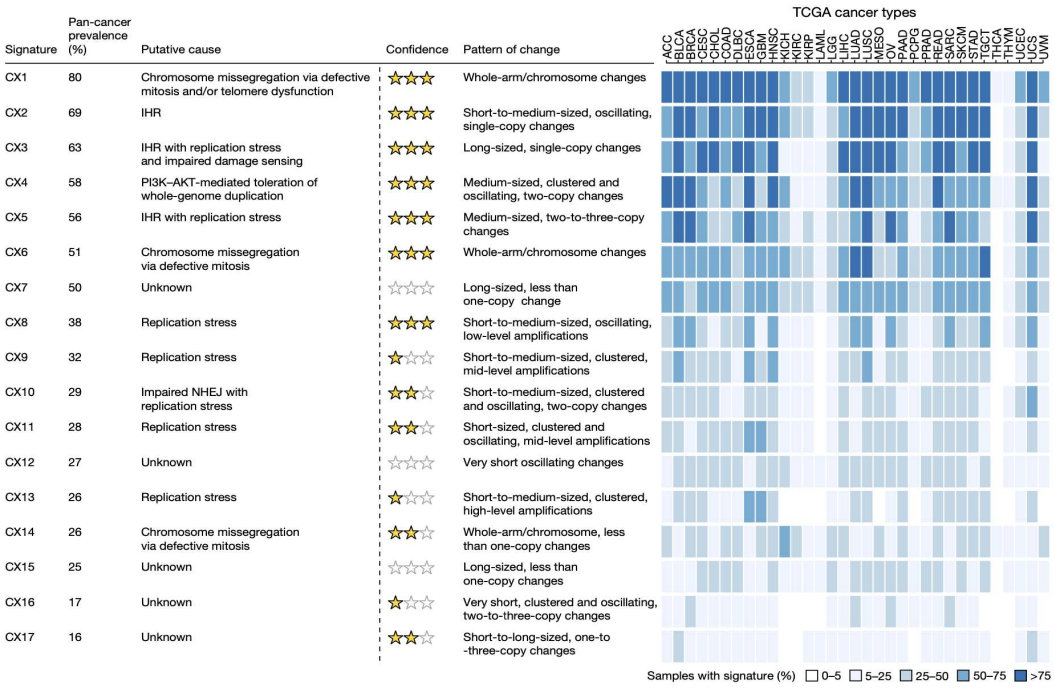
Figura 2.2: Signature Heatmap

Un'analisi più approfondita dell'asse x della heatmap rivela la presenza di due rami principali: il primo è costituito solo dalla CX1, mentre il secondo si suddivide ulteriormente, con CX3 come una delle sue ramificazione e l'altra ramificazione si divide a sua volta con un ramo che comprende CX5 e CX2, mentre l'altro ramo contiene tutte le altre signature con ulteriori suddivisioni successive. Questo schema di divisione suggerisce che la CX1 sia nettamente separata dalle altre signature, mentre CX3, CX5 e CX2 siano interconnesse tra

loro e separate dalle altre signature. Questo schema di divisione risulta coerente con le presunte cause associate alle signature menzionate nel lavoro di riferimento [3]. Infatti, per la CX1, si ipotizza la misegregazione cromosomica tramite mitosi difettosa e/o disfunzione dei telomeri, mentre per CX3, CX5 e CX2 si ipotizza una ricombinazione omologa compromessa (IHR). Pertanto, è ragionevole aspettarsi che queste signature siano correlate.

Inoltre, è importante sottolineare che nel lavoro di riferimento le signature sono numerate in base alla loro prevalenza pan-cancer, il che suggerisce che queste quattro signature (CX1, CX3, CX5 e CX2) siano particolarmente rilevanti per lo studio pan-cancer che è l'obiettivo principale di questa ricerca.

Per quanto riguarda le restanti signature, l'immagine 2.3 del lavoro di riferimento rappresenta le caratteristiche di tutte e 17 le signature.



Signature	Pan-cancer prevalence (%)	Putative cause	Confidence	Pattern of change
CX1	80	Chromosome missegregation via defective mitosis and/or telomere dysfunction	★★★	Whole-arm/chromosome changes
CX2	69	IHR	★★★	Short-to-medium-sized, oscillating, single-copy changes
CX3	63	IHR with replication stress and impaired damage sensing	★★★	Long-sized, single-copy changes
CX4	58	PI3K–AKT-mediated toleration of whole-genome duplication	★★★	Medium-sized, clustered and oscillating, two-copy changes
CX5	56	IHR with replication stress	★★★	Medium-sized, two-to-three-copy changes
CX6	51	Chromosome missegregation via defective mitosis	★★★	Whole-arm/chromosome changes
CX7	50	Unknown	☆☆☆	Long-sized, less than one-copy change
CX8	38	Replication stress	★★★	Short-to-medium-sized, oscillating, low-level amplifications
CX9	32	Replication stress	★☆☆	Short-to-medium-sized, clustered, mid-level amplifications
CX10	29	Impaired NHEJ with replication stress	★★☆	Short-to-medium-sized, clustered and oscillating, two-copy changes
CX11	28	Replication stress	★★☆	Short-sized, clustered and oscillating, mid-level amplifications
CX12	27	Unknown	☆☆☆	Very short oscillating changes
CX13	26	Replication stress	★☆☆	Short-to-medium-sized, clustered, high-level amplifications
CX14	26	Chromosome missegregation via defective mitosis	★★☆	Whole-arm/chromosome, less than one-copy changes
CX15	25	Unknown	☆☆☆	Long-sized, less than one-copy changes
CX16	17	Unknown	★☆☆	Very short, clustered and oscillating, two-to-three-copy changes
CX17	16	Unknown	★★☆	Short-to-long-sized, one-to-three-copy changes

Figura 2.3: Eziologie proposte e prevalenza delle signature di copy number

Per enfatizzare ulteriormente la validità della scelta di utilizzare queste quattro signature anziché tutte le altre, si è condotta un'analisi approfondita della distribuzione dei valori delle signature in tutti i campioni. Mentre l'heatmap fornisce un'informazione visiva e di relazione nei dati, questa analisi mira a ottenere una comprensione più approfondita della distribuzione effettiva

dei valori. Per raggiungere questo obiettivo, si è sfruttata la funzione "*densityHeatmap*" del pacchetto *ComplexHeatmap*, la quale genera una mappa di densità dei valori per ciascuna signature. L'immagine in figura 2.4 rappresenta l'output di questa analisi sulla matrice delle signature. In questa rappresentazione, i toni più vicini al rosso indicano una maggiore densità, che si riferisce alla quantità di campioni che presentano valori simili. Una considerazione immediata è che la CX1 e la CX3, rispettivamente, evidenziano una distribuzione dei valori con una densità inferiore e più uniformemente distribuita rispetto alle altre signature. La CX5 anche se mostra una notevole densità di valori prossimi allo zero, ha una distribuzione dei suoi valori migliore rispetto alle altre signature. Tra le quattro signature identificate in precedenza tramite l'heatmap, rimane ora solo la CX2 da vedere. È interessante notare che essa presenta una densità significativamente elevata di valori nell'intervallo compreso tra 0 e 0.2, pur restando comunque la quarta migliore distribuzione rispetto alle altre signature.

Per quanto riguarda tutte le restanti signature, è evidente che mostrano una densità significativa di valori vicino allo zero. Questo solleva dubbi sull'idoneità di queste signature per l'applicazione nei nostri modelli di machine learning e sulla loro effettiva rilevanza nel contesto dell'identificazione dell'instabilità cromosomica nei tumori. Questa questione verrà esplorata in dettaglio e discussa nelle conclusioni della tesi.

Questo approfondimento con la mappa di densità sottolinea ulteriormente la validità delle nostre scelte nel selezionare queste quattro signature come punti focali per la nostra analisi, poiché la loro distribuzione dei valori offre una base solida per l'applicazione dei modelli di machine learning descritti nel capitolo 1.

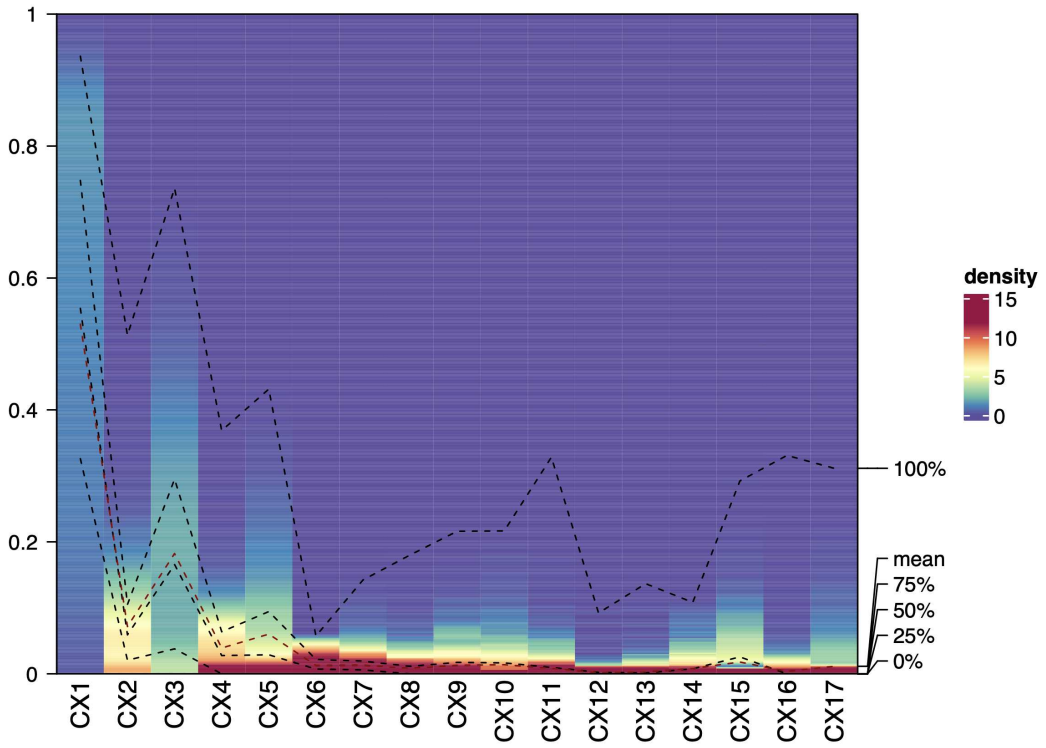


Figura 2.4: Signature density Heatmap

Tra le quattro signature selezionate (CX1, CX3, CX5 e CX2), è stato essenziale designarne una come punto di partenza per un'analisi più approfondita e per comprendere il comportamento dei modelli di machine learning. La scelta è ricaduta sulla CX1 dato che presenta una migliore distribuzione dei valori, presenta meno zeri ed è quella con una prevalenza pan-cancer maggiore rispetto alle altre secondo il lavoro di rifermento [3]. Pertanto, ci aspettiamo che funga da punto di partenza ideale per raffinare i modelli di machine learning e ottenere conclusioni più accurate. Di conseguenza, l'intero capitolo 4 è dedicato all'analisi delle performance dei modelli sulla CX1, mentre le altre tre signature (CX3, CX5 e CX2) verranno esaminate nel capitolo 5, basandosi su quanto appreso dall'analisi della CX1.

Capitolo 3

Addestramento dei modelli di regressione e tuning dei parametri

Prima di procedere con l'addestramento dei modelli di regressione, è essenziale eseguire la pratica del "data splitting". Questo processo comporta la suddivisione del dataset originale in due sottoinsiemi distinti: il set di addestramento e il set di validazione. Tale suddivisione dei dati riveste una fondamentale importanza poiché fornisce un set indipendente di dati per valutare le prestazioni del modello in modo obiettivo e imparziale. Questo significa che è possibile utilizzare misure di valutazione delle prestazioni, come l'errore quadratico medio (RMSE) o il coefficiente di determinazione (R^2), al fine di determinare quanto accuratamente il modello si adatta ai dati di test. Inoltre, il data splitting permette di confrontare diversi modelli o diverse configurazioni dello stesso modello. È possibile addestrare più modelli e valutarli utilizzando lo stesso set di test per stabilire quale di essi è il più adatto per risolvere specificamente il problema di regressione considerato. Questo processo di confronto dei modelli verrà dettagliatamente esaminato nel capitolo 4.2, in cui sarà condotta un'analisi approfondita per identificare quale tra i modelli selezionati nel capitolo 1.2 offre le prestazioni migliori per la signature in esame. Da non trascurare è il ruolo cruciale del data splitting nell'individuare eventuali segni di overfitting del modello. Esso si verifica quando il modello si adatta troppo ai dati di addestramento, catturando anche il rumore nei dati anziché solo i modelli

sottostanti. È possibile accorgersi di overfitting nel modello quando è presente una differenza significativa tra le prestazioni sul set di addestramento e quelle sul set di test, tendenzialmente con le prestazioni sul secondo che risultano inferiori.

3.1 Divisione per cancer type del training e del validation set

La divisione dei dati attuata è stata guidata sulla signature da predire e, considerando la dimensione del dataset ottenuto nel capitolo 2.3.1, si è scelto di eseguire una divisione 80/20%, risultando in 3626 campioni nel training set e 907 campioni nel validation set. Il pacchetto *caret* offre la possibilità di fare questa divisione casuale sfruttano la funzione "*createDataPartiton*". Tuttavia, dato che il nostro studio è di tipo pan-cancer e il numero di campioni tumorali è sbilanciato tra i diversi tipi di tumore, era importante evitare che un tipo tumorale fosse rappresentato esclusivamente nel training set o nel validation set. Allo stesso tempo, si vuole evitare che un tipo tumorale con un elevato numero di campioni sia sovra rappresentato nel validation set rispetto al training set. Per affrontare questa sfida, è stato utilizzato il seguente codice in R anziché la funzione "*createDataPartiton*":

```
set.seed(123)
strat_sample <- mat_finale %>%
  group_by(cancer_type) %>%
  sample_frac(size=.80)

Training <- data.frame(strat_sample, row.names = 1)
ordi_train<-as.data.frame(Training[order(row.names(Training)), ])
flag<-mat_finale[!rownames(mat_finale) %in% rownames(ordi_train),]
Testing<- data.frame(flag, row.names = 1)
ordi_test<-as.data.frame(Testing[order(row.names(Testing)), ])
```

ordi_train è il dataset con 3626 campioni per l'allenamento, mentre *ordi_test* è il dataset con 907 campioni per la validazione.
Questo approccio ha permesso una divisione casuale dei campioni, ma ha tenuto conto del tipo di tumore dei campioni. In questo modo, è stato possibile ottenere una divisione 80/20% per ciascun tipo di tumore, garantendo che anche se i tipi tumorali avevano un numero di campioni diverso nei due dataset, fossero rappresentati in egual percentuale rispetto al loro totale.

3.2 Addestramento modelli

L'addestramento dei modelli viene eseguito mediante l'utilizzo della funzione "*train*" del pacchetto *caret*. Di seguito, viene rappresentato un esempio di utilizzo di questa funzione insieme ai relativi argomenti richiesti:

```
set.seed(123)
model <- train(x, y, method = " ",
                 trControl = trainControl(),
                 tuneGrid = expand.grid())
```

Questa funzione, come si può notare nell'esempio, richiede la specifica di diversi argomenti fondamentali:

- **X**: rappresenta l'insieme di tutte le colonne di input presenti nel dataset.
- **Y**: corrisponde alla colonna contenente l'output da predire nel dataset.
- **method**: è il parametro in cui si specifica il modello di regressione che si desidera applicare.
- **trControl**: richiede come argomento la funzione "*trainControl*" e ha lo scopo di gestire il processo di ricampionamento (resampling).
- **tuneGrid**: fa uso della funzione "*expand.grid*" per consentire l'ottimizzazione degli iperparametri del modello selezionato tramite "method".

Questi argomenti sono fondamentali per configurare e addestrare correttamente i modelli di regressione durante il processo di analisi. Rispetto ai primi tre argomenti, che sono di facile comprensione, è necessario approfondire ulterior-

mente gli due ultimi. Il primo verrà affrontato nei prossimi paragrafi, mentre il secondo sarà oggetto di discussione nel capitolo 3.3.

La funzione "*trainControl*" del pacchetto *caret* è essenziale per definire e personalizzare il processo di addestramento dei modelli di machine learning. Questa funzione consente di stabilire le regole e le configurazioni per la validazione incrociata (k-fold cross-validation, CV), il ricampionamento e altre operazioni legate all'addestramento dei modelli. Nel nostro caso, tutti i modelli sono stati addestrati utilizzando la validazione incrociata a k-fold, nello specifico una cross-validation a 10 fold.
È inoltre possibile specificare se il processo di addestramento deve essere eseguito in parallelo per accelerare il calcolo, una considerazione importante quando si lavora con dataset di grandi dimensioni come nel nostro caso. Per abilitare l'esecuzione in parallelo, si aggiunge l'argomento "$allowParallel = TRUE$".
Queste personalizzazioni sono cruciali per ottenere risultati precisi e affidabili durante l'addestramento dei modelli, ed è importante utilizzare gli stessi argomenti per garantire una comparazione equa tra i diversi modelli. Di seguito, viene fornito un esempio di codice che rappresenta gli argomenti utilizzati nella funzione "*trainControl*" per tutti i modelli di regressione addestrati in questo lavoro.

```
trControl <- trainControl(
  method = "cv",           # Tipo di validazione incrociata
  number = 10,             # Numero di fold nella cross-validation
  savePredictions = TRUE,  #per salvare le predizioni del modello
  allowParallel = TRUE)    #per permettere il calcolo parallelo
```

Per confrontare correttamente i diversi modelli tra di loro, oltre ad addestrarli con una validazione incrociata che utilizza gli stessi argomenti, è essenziale utilizzare la funzione "*set.seed()*" prima di eseguire la funzione "*train*". Questa funzione è utilizzata per impostare un seme casuale, rendendo così riproducibili i risultati di operazioni che coinvolgono elementi casuali, come la suddivisione casuale dei dati o la generazione di numeri casuali. Questo aspetto diventa particolarmente critico quando si confrontano modelli diversi o si

eseguono operazioni di resampling, poiché garantisce la coerenza dei risultati tra diverse esecuzioni.
Il valore inserito all'interno di "*set.seed()*" è arbitrario e può essere modificato a piacimento. Tuttavia, è fondamentale assicurarsi di utilizzare lo stesso seme casuale per garantire la coerenza dei confronti tra i modelli. Pertanto, prima di eseguire la funzione "*train*" per ciascuno dei modelli selezionati nel capitolo 1.2, è stato fondamentale impostare lo stesso seme casuale, che nel nostro caso è stato scelto come "*set.seed(123)*".

3.3 Tuning degli iperparametri

Quando si addestrano modelli di machine learning, è spesso necessario regolare gli iperparametri del modello per ottenere le migliori prestazioni, e questo processo è noto come *tuning* degli iperparametri. L'obiettivo principale di questa ottimizzazione è massimizzare la precisione del modello, ovvero la sua capacità di effettuare previsioni accurate su nuovi dati, minimizzando al contempo il costo computazionale associato. Ogni problema di regressione può richiedere una configurazione di iperparametri diversa, e l'ottimizzazione consente di adattare il modello alle specifiche caratteristiche del problema, tenendo conto di fattori come la dimensionalità dei dati, la presenza di rumore nei dati e la complessità delle relazioni tra le variabili. Oltre a migliorare le prestazioni del modello, la ricerca dei migliori iperparametri consente anche di controllare l'overfitting, rendendo il modello più robusto e in grado di generalizzare meglio su nuovi dati.

Per questa operazione, il pacchetto *caret* offre due opzioni: è possibile utilizzare la funzione "*tuneLength*", che richiede solo il numero di combinazioni casuali da testare per l'ottimizzazione, oppure utilizzare la funzione "*expand.grid*" per definire una griglia di possibili combinazioni degli iperparametri da esplorare durante il processo di ottimizzazione per un modello di machine learning. La prima opzione, *tuneLength*, è utile per una fase esplorativa iniziale su un sottoinsieme dei dati, al fine di ottenere un'idea generale delle migliori configurazioni degli iperparametri per ciascun modello. La seconda opzione,

"*expand.grid*", è più adatta per i modelli finali quando si conoscono in anticipo quali potrebbero essere le migliori configurazioni possibili.

È importante sottolineare che a seconda del modello di regressione scelto, ci sono diversi iperparametri specifici da regolare. Nei prossimi sottocapitoli, verranno analizzati in dettaglio gli iperparametri specifici per ciascun modello di regressione, con esempi pratici utilizzati per i modelli relativi alla signature CX1.

3.3.1 Lasso e Ridge

Per poter attuare i modelli ridge e lasso si deve utilizzare il metodo "*glmnet*" nella funzione "*train*" del pacchetto *caret*[6]. Questo metodo consente di implementare la regressione Lasso e Ridge in base alla configurazione degli iperparametri. "*glmnet*" ha due iperparametri principali che possono essere oggetto di ottimizzazione: alpha (percentuale di mescolamento) e lambda (parametro di regolarizzazione). L'iperparametro alpha può variare tra 0 e 1, rappresentando rispettivamente un modello puramente Ridge (alpha = 0) e un modello puramente Lasso (alpha = 1). I valori intermedi consentono di implementare un modello di regressione Elastic Net, il che significa che se ad esempio alpha = 0,05, si ha una regressione Ridge al 95% e una regressione Lasso al 5%. Per quanto riguarda l'iperparametro lambda, esso può assumere valori arbitrari, ma è consigliabile utilizzare un insieme di valori piccoli. Questo consente di identificare il valore ottimale di lambda per il modello durante l'ottimizzazione. Nel codice sottostante, è mostrato un esempio della griglia di valori utilizzati per il modello Ridge nella regressione per la signature CX1. In questo caso, sono stati testati 100 valori di lambda per determinare il miglior valore dell'iperparametro di regolarizzazione da utilizzare.

```
tuneGrid = expand.grid(alpha = 0,
                       lambda = seq(0,2,length=100))
```

3.3.2 Random forest

Caret fornisce diverse opzioni per la creazione di modelli di regressione basati su Random Forest, tra cui "*rf*", "*Rborist*", e "*ranger*". "*rf*" è il modello base utilizzato per creare modelli di Random Forest, mentre "*Rborist*" è un'implementazione ad alte prestazioni del modello "*rf*"."*ranger*", invece, è una scelta indicata per dataset di grandi dimensioni in quanto combina un'implementazione in termini di memoria con una notevole velocità di esecuzione. Nella fase di valutazione, è stato condotto un confronto tra questi tre metodi, analizzando le loro prestazioni e valutando la complessità computazionale necessaria per completare il modello. Alla luce dei risultati ottenuti, il metodo "*ranger*" si è rivelato essere la scelta più vantaggiosa in termini di performance e di risorse computazionali. Questo metodo prevede l'ottimizzazione di tre iperparametri principali:

1. **mtry**(Predittori casuali selezionati): può assumere valori da 1 al numero totale delle variabili esplicative nel dataset, ma non può superare quest'ultimo.
2. **splitrule**(Regola di divisione): Il valore di questo iperparametro regola come avvengono le divisioni nei nodi dell'albero. Le opzioni variano a seconda del tipo di problema, e per la regressione, sono state testate diverse opzioni, identificando "variance" come la più efficace.
3. **min.node.size**(Dimensione minima del nodo): definisce la dimensione minima consentita per i nodi dell'albero. Il valore specifico da utilizzare dipende dal tipo di modello: 1 per la classificazione, 5 per la regressione, 3 per i modelli di sopravvivenza e 10 per la probabilità.

Nel codice sottostante, è mostrato un esempio della griglia di valori utilizzati per l'iperparametro "*mtry*" nel modello "*ranger* " per la signature CX1. Gli altri due iperparametri hanno valori fissi definiti in base al tipo di modello.

```
tuneGrid = expand.grid(mtry = seq(500, 21789, length =20),
                       splitrule = c("variance"),
                       min.node.size = 5),
importance = "permutation"
```

Nel codice, è importante notare l'inclusione di una voce aggiuntiva denominata "importance." Questa voce deve essere necessariamente specificata all'interno della funzione "train" per il modello di regressione Random Forest. La ragione di questa inclusione è fondamentale perché ci consente di estrarre le variabili importanti dal modello. Questo passaggio riveste notevole importanza poiché ci permette di valutare le performance dei modelli, come verrà illustrato nel dettaglio nel capitolo 4.1.

3.3.3 SGD Boosting

Per eseguire il Stochastic Gradient Boosting, si utilizza il metodo "*gbm*" del pacchetto *caret*, che offre quattro iperparametri fondamentali da ottimizzare per ottenere un modello con prestazioni ottimali:

1. **n.trees**(boosting interaction): rappresenta il numero di iterazioni o alberi da addestrare nel modello. Aumentando il valore di "*n.trees*", si riduce l'errore sul set di addestramento, tuttavia, un valore eccessivamente alto potrebbe portare a overfitting. D'altra parte, un valore troppo basso potrebbe produrre un modello sottodimensionato. Risulta essenziale trovare il numero ottimale di alberi per ottenere il modello più performante.
2. **interaction.depth**(max tree depth): controlla il numero di divisioni che un albero può effettuare a partire da un singolo nodo. La scelta del valore dipende dalla complessità del problema e dalle dimensioni del dataset. In generale, un valore superiore a due è necessario per rilevare interazioni tra le variabili.
3. **shrinkage**(learning rate): permette di regolare la velocità di apprendimento del modello. Valori bassi richiedono più iterazioni ma possono portare a una migliore generalizzazione, mentre valori alti riducono il numero di iterazioni, ma potrebbero causare overfitting. È consigliabile utilizzare valori bassi all'aumentare delle dimensioni del dataset e del numero di alberi che devono crescere. Tuttavia, bisogna considerare che valori di apprendimento bassi richiedono più tempo per l'addestramento.

4. **minobsinnode**(Min. Terminal Node Size): controlla il numero minimo di osservazioni richieste in un nodo terminale dell'albero. Aumentare questo valore rende il modello meno complesso, ma potrebbe ridurne la capacità di adattamento ai dati.

Nel codice mostrato di seguito, è presentato un esempio di griglia di valori utilizzati per il modello SGD Boosting per la signature CX1. Per ottimizzare il modello, è necessario esplorare diverse combinazioni di valori per ciascun iperparametro e valutarne l'impatto sulle prestazioni del modello su un set di dati di validazione.

```
tuneGrid = expand.grid(n.trees = (1:20)*50,
                       interaction.depth = 1:10,
                       shrinkage = c(.1, .05),
                       n.minobsinnode = c(5, 10))
```

3.3.4 SVM

Caret offre diverse opzioni per la creazione di modelli di regressione Support Vector Machines, tra cui SVM with Linear Kernel e SVM with Radial Basis Function Kernel. In particolare, i metodi selezionati sono "*svmLinear*" e "*svmRadialSigma*".

Per quanto riguarda "*svmLinear*", questo modello richiede l'ottimizzazione di un singolo iperparametro, ovvero "*C*"(cost), il quale influenza la larghezza dei margini decisionali. Il valore di "C" è inversamente proporzionale alla larghezza dei margini, un valore elevato di "C" rende il modello meno permissivo, conducendo a margini più stretti e una maggiore attenzione ai dettagli dei dati. Questo può aumentare il rischio di overfitting poiché il modello si adatta troppo ai dati di addestramento. D'altra parte, un valore basso di "C" rende il modello più permissivo, con margini più ampi che consentono una maggiore generalizzazione. L'ottimizzazione di "C" è cruciale per bilanciare l'adattamento ai dati e il controllo dell'overfitting.

Per quanto riguarda "*svmRadialSigma*", oltre all'iperparametro "C" precedentemente menzionato, include anche l'iperparametro "*sigma*"(gamma), che

regola la larghezza della funzione kernel. Un valore ridotto di "sigma" allarga la funzione, consentendo una maggiore generalizzazione e una maggiore capacità del modello di adattarsi a diverse situazioni, mentre un valore elevato di "sigma" restringe la funzione, portando a un maggiore adattamento ai dati di addestramento, aumentando di conseguenza il rischio di overfitting.
Nel codice sottostante, è mostrato un esempio della griglia di valori utilizzati per i modelli SVM per la signature CX1. Rispettivamente la griglia di SVM lineare e SVM radiale:

```
tuneGrid = expand.grid(C = c(0.000001, 0.00001, 0.00005, 0.0001, 0.001,
                             0.01, 1))


tuneGrid = expand.grid(C= c(4, 6, 8, 10, 12, 16),
                       sigma =c(.088365e-05, .411635e-05, .911635e-05,
                                1.411635e-05, 1.908374e-05, 2.405114e-05))
```

Le griglie di valori per i modelli sono state create sulla base di un'analisi preliminare effettuata utilizzando la funzione "*tuneLength*" per entrambi i metodi. Questa analisi ha consentito di identificare gli intervalli di valori ottimali da utilizzare nell'ottimizzazione degli iperparametri.

Capitolo 4

Performance modelli di regressione sulla signature CX1

In questo capitolo, verranno esaminate le performance dei sei modelli utilizzati per la previsione della signature CX1. Prima di analizzare nel dettaglio i risultati, è fondamentale comprendere il processo di valutazione delle prestazioni del modello. Questo processo comporta l'estrazione delle previsioni sui dati del set di validazione e il confronto delle distribuzioni di ricampionamento tra i vari modelli.

Prima di estrarre le previsioni sui dati del validation set, è fondamentale garantire che il modello sia stato allenato correttamente, ovvero che siano stati selezionati gli iperparametri ottimali per ottenere le migliori prestazioni possibili senza cadere nell'overfitting. Per effettuare questa verifica, è stata utilizzata la funzione "*plot*" per rappresentare graficamente il modello ottenuto dalla funzione "*train*", come descritto nel capitolo 3.2. Il grafico risultante mostra l'RMSE sull'asse delle ordinate e l'iperparametro sull'asse delle ascisse. Nell'appendice B, è possibile trovare tutti i grafici relativi ai modelli di CX1. Per quanto riguarda il modello GBM, date le complesse combinazioni degli iperparametri, è stato utilizzato l'argomento "*plotType = 'level'*", il quale rende il grafico facilmente interpretabile (figura B.4). I grafici risultanti derivano da una ricerca ottimale degli iperparametri forniti al modello, con l'obiettivo di ottenere il minor RMSE possibile. È importante notare che la ricerca dell'i-

perparametro ottimale è stata condotta in modo che il miglior iperparamentro selezionato non coincida con i valori estremi testati, al fine di evitare il rischio di overfitting.

4.1 Come valutare le performance di un modello di regressione

Per valutare l'efficacia di un modello di regressione, è necessario fare ricorso a diverse metriche. In particolare, nel contesto di questa tesi:

- **Errore medio assoluto**(MAE): calcola la media degli errori assoluti tra le previsioni del modello e i valori reali. Il MAE è utile per misurare l'errore medio del modello.
- **Radice dell'errore quadratico medio**(RMSE): È la radice quadrata dell'MSE ed è utile per ottenere un'idea dell'errore medio in unità della variabile bersaglio. L'**MSE** è calcolato come la media degli errori quadrati tra le previsioni e i valori reali. Esso dà più peso agli errori più grandi e può aiutarti a identificare le previsioni che si discostano notevolmente dai valori reali.
- **Coefficiente di determinazione**(R-squared): questa metrica varia da 0 a 1 e misura quanto bene il modello si adatta ai dati. Un valore più alto indica un migliore adattamento. L'R-squared da solo potrebbe non fornire una valutazione completa delle performance del modello, ed è importante considerarlo insieme alle altre metriche.

4.1.1 Variabili Importanti

Nel contesto di un modello di regressione, le variabili svolgono un ruolo fondamentale sia nella previsione dei risultati sia nella comprensione del processo sottostante. Identificare con precisione le variabili di maggiore importanza riveste un ruolo cruciale, poiché fornisce chiarezza sulla contribuzione di ciascuna di esse alle previsioni. Questa conoscenza consente di interpretare il modello con maggiore accuratezza, poiché rivela quali variabili influiscono in

modo più significativo sulla variabile risposta.
Per condurre questa analisi, è stata utilizzata la funzione "*varImp*" fornita da *caret*. Nel contesto della valutazione dei modelli sono state selezionate le 100 variabili più importanti, rilevando così le caratteristiche di maggior rilevanza per il processo di previsione. Questa selezione mirata di variabili consente di concentrare l'attenzione sulle informazioni più influenti, semplificando l'interpretazione del modello e ottimizzando le prestazioni complessive.

4.2 Come confrontare i modelli di regressione

Nel contesto del confronto dei modelli di regressione, è essenziale esaminare attentamente due aspetti chiave: la distribuzione di ricampionamento e la capacità di predizione sui dati di validation set. Questi due elementi offrono prospettive complementari sulla qualità e l'affidabilità dei modelli, svolgendo un ruolo essenziale nel processo decisionale. Allo stesso però bisogna considerare anche un terzo aspetto chiave: la valutazione delle differenze tra i modelli.

La valutazione della distribuzione di ricampionamento riveste un ruolo di fondamentale importanza poiché fornisce una chiara indicazione della stabilità e della robustezza dei modelli. Attraverso questa analisi, è possibile esaminare come i modelli si comportano in diverse iterazioni del processo di addestramento.

D'altra parte, la capacità di predizione sui dati di validation set rappresenta un elemento critico per la valutazione delle capacità di generalizzazione dei modelli su dati non osservati in precedenza. Durante questa fase, le previsioni dei modelli vengono confrontate con i valori reali presenti nel validation set, utilizzando le metriche di valutazione precedentemente illustrate nel capitolo 4.1. Quest'analisi mette in luce quale modello dimostra la migliore capacità di effettuare previsioni precise su dati nuovi e sconosciuti.

Infine, poiché i modelli si adattano alle stesse versioni dei dati di addestramento, ha senso fare delle inferenze sulle differenze tra i modelli. In questo modo riduciamo la correlazione all'interno del ricampionamento che può esi-

stere. Possiamo calcolare le differenze e successivamente applicare un semplice t-test per valutare l'ipotesi nulla(H0) che non ci sia differenza tra i modelli. L'incrocio tra queste tre valutazioni crea una base solida per la selezione del modello più adatto al contesto specifico, permettendo di scegliere il modello che dimostra sia stabilità e robustezza nel processo di addestramento che capacità di generalizzazione su dati sconosciuti.

4.2.1 Distribuzione di ricampionamento

Per poter vedere la distribuzione di ricampionamento *caret* metta a disposizione la funzione "*resamples*". Essa calcola le metriche di valutazione per ciascuna ripetizione della validazione incrociata e restituisce un oggetto con i risultati. È possibile sottomettere a questa funzione tutti i modelli che si vuole confrontare insieme sotto forma di lista e successivamente si possono utilizzare diversi grafici per visualizzare i risultati della distribuzione di ricampionamento ottenuti, come: *bwplot(boxplot), dotplot, densityplot e parallelplot.* Tra di esse i più esemplificativi sono i prime due plot, ma anche gli altri due rappresentano una alternativa valida.

4.2.2 Capacità di predizione

Per effettuare la scelta del modello con la capacità predittiva migliore, ci basiamo principalmente sull'RMSE calcolato sui dati di validazione. Questa metrica fornisce una misura della precisione del modello e il più performante risulta essere il modello con l'RMSE più basso. Una volta individuato il miglior modello sulla base del RMSE, si passa ad una seconda fase di valutazione in cui si utilizza il coefficiente R^2 per valutare effettivamente le capacità predittive del modello selezionato.
Per effettuare questa analisi, è stata utilizzata la funzione "*extractPrediction*" fornita da *caret*. Questa funzione consente di ottenere le previsioni dei modelli. L'oggetto risultante di questa funzione può essere manipolato in modo da calcolare separatamente l'RMSE e il Rsquared per ciascun modello e per tipo di dataset. Per calcolare le due metriche sono state usate rispettivamente le

funzioni "*RMSE*" e "*R2*" di *caret*. Nel codice successivo, è possibile osservare come è stato implementato questo processo:

```
predictions %>%
  group_by(model, dataType) %>%
  dplyr::summarise(
    rmse = RMSE(pred = pred, obs = obs),
    rsq = R2(pred = pred, obs = obs))
```

L'oggetto "*predictions*" ottenuto può essere visualizzato graficamente attraverso la funzione "*plotObsVsPred*" e il grafico che si ottiene confronta le osservazioni reali (valori osservati) con le previsioni (valori predetti) ottenuti dal modello di regressione. Questo tipo di grafico è particolarmente utile per identificare eventuali errori di previsione del modello, come sovra-stima o sotto-stima, e per valutare la qualità generale delle previsioni del modello. Per ottenere una rappresentazione grafica migliore, è stata impiegata la libreria *ggplot* per generare un grafico simile a quello ottenuto con la funzione precedentemente menzionata. Questo approccio offre maggiore flessibilità e controllo sulla visualizzazione dei dati, consentendo la creazione di grafici più dettagliati e personalizzati.

4.2.3 Valutazione delle differenze tra i modelli

L'obiettivo principale di questo processo è stabilire se uno dei modelli si comporta significativamente meglio dell'altro in termini di previsioni. Se il test t mostra che le differenze sono statisticamente significative, ciò suggerisce che uno dei modelli è probabilmente superiore all'altro per il compito specifico. In sostanza, si sta cercando di determinare se c'è una differenza significativa tra i modelli basata su prove statistiche piuttosto che su casualità o fluttuazioni casuali nei dati.

Per condurre questa analisi, si utilizza la funzione "*diff*" sull'oggetto precedentemente creato con la funzione "*resamples*", come illustrato nel capitolo 4.2.1. La funzione "*summary*" permette di visualizzare le differenze tra i modelli, generando una rappresentazione a forma di matrice con una diagonale superiore e una inferiore. La diagonale superiore rappresenta le differenze stimate tra i

modelli e può essere visualizzata graficamente con la funzione "*dotplot*, mentre la diagonale inferiore fornisce i p-value associati all'ipotesi nulla (H0).

4.3 Confronto risultati tra i modelli ottenuti

Come discusso nel paragrafo 4.2, il confronto dei modelli di regressione può basarsi su due aspetti fondamentali. Da un lato, è possibile valutare quanto bene ciascun modello si adatta ai dati di addestramento. Dall'altro lato, è essenziale esaminare la capacità predittiva dei modelli sui dati di validation. Questi due aspetti verranno esaminati in dettaglio nelle successive due sezioni al fine di fornire una visione completa delle performance dei modelli di regressione ottenuti nel contesto della signature CX1.

4.3.1 Confronto ricampionamento e differenze tra i modelli

Nel contesto della visualizzazione delle differenze nel ricampionamento, si è optato per l'utilizzo del boxplot generato tramite la funzione "*bwplot*". Questa scelta si basa sulla capacità del boxplot di offrire una valutazione precisa e una rappresentazione grafica più efficace rispetto ad altre opzioni disponibili. Inoltre, rispetto a un dotplot che rappresenta la media dei valori con un intervallo di confidenza, il boxplot rappresenta la mediana, la quale è meno influenzata da valori estremi e fornisce una stima della "tipica" performance del modello. Questo aspetto rende il boxplot più robusto rispetto alla media, specialmente in presenza di outlier.

Nella figura 4.1, vengono presentate le tre metriche utilizzate per valutare il modello di regressione applicato alla signature CX1. I modelli sono stati ordinati in modo decrescente in base all'RMSE, con i modelli meno performanti posizionati nella parte superiore e i modelli migliori nella parte inferiore del grafico. Questo ordinamento è basato sul valore mediano dei 10 risultati di ricampionamento ottenuti per ciascun modello. Da questa analisi emerge che il modello SVM radiale presenta il RMSE più basso, indicando una maggiore

precisione nella previsione rispetto agli altri modelli. Inoltre, presenta il valore più alto di R2 tra tutti i modelli, confermando la sua capacità di predire i dati con maggiore accuratezza. I dettagli relativi ai valori di ciascun modello sono riportati nella tabella 4.1.

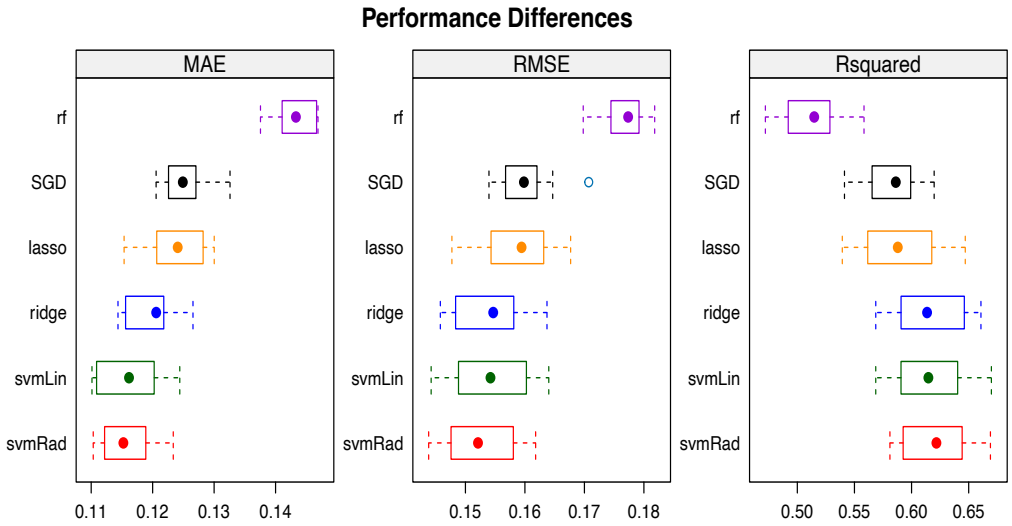


Figura 4.1: Boxplot ricampionamento

model	RMSE	Rsqaured	model	RMSE	Rsqaured
Random forest	0.177	0.515	Ridge	0.155	0.614
SGD	0.160	0.586	svmLinear	0.154	0.615
Lasso	0.159	0.588	svmRadial	0.152	0.622

Tabella 4.1: Valore mediano di ricampionamento

Un'ulteriore rappresentazione grafica che facilita la comprensione dell'andamento dei valori di RMSE durante la validazione incrociata è il grafico generato dalla funzione "*parallelplot*". Questo tipo di grafico, noto come "grafico a coordinate parallele," visualizza le prestazioni di ciascun modello come linee parallele e consente di visualizzare le differenze tra i modelli durante il processo di ricampionamento. Nella figura C.2 si può vedere il risultato ottenuto, il quale conferma la superiorità del modello SVM radial rispetto agli altri.

Per quanto riguarda le differenze tra i modelli, nella figura 4.2 è possibile esaminare i risultati ottenuti tramite la funzione "summary", come descritto precedentemente nel capitolo 4.2.3. I valori di p-value presenti nella diagonale

inferiore indicano se è possibile rifiutare l'ipotesi nulla (H0) a favore dell'ipotesi alternativa, ovvero se le differenze stimate tra i modelli nella diagonale superiore (rappresentate nella figura C.3) sono differenze significative o meno. È importante notare che alcuni valori di p-value sono uguali a 1. Un p-value pari a 1 indica l'assenza di basi statistiche per affermare che i gruppi siano diversi. Rappresenta il valore massimo possibile e suggerisce l'assenza di evidenza contro l'ipotesi nulla, pertanto, le differenze osservate sono probabilmente dovute al caso o all'aleatorietà dei dati. Questa situazione si verifica in solo due casi: quando confronto il modello SGD Boosting con il Lasso e quando confronto il modello SVM lineare con il Ridge. In tutti gli altri confronti, ad eccezione di quattro situazioni in cui il valore si avvicina solamente alla soglia di significatività statistica, si ottengono valori di p-value inferiori a 0.05, i quali indicano la presenza di differenze statisticamente significative tra i modelli.

```
p-value adjustment: bonferroni
Upper diagonal: estimates of the difference
Lower diagonal: p-value for H0: difference = 0

RMSE
       lasso     ridge      rf         SGD        svmLin     svmRad
lasso             0.0044789 -0.0177313 -0.0012897  0.0042990  0.0060757
ridge  0.035227             -0.0222101 -0.0057686 -0.0001798  0.0015969
rf     2.199e-06 1.442e-07              0.0164415  0.0220303  0.0238070
SGD    1.000000  0.021250   1.149e-06              0.0055888  0.0073654
svmLin 0.162965  1.000000   9.245e-07  0.070607               0.0017767
svmRad 0.013307  0.139480   2.579e-07  0.003701   0.103104
```

Figura 4.2: Differenze tra modelli

4.3.2 Confronto performance tra i modelli

Nella tabella riportata di seguito, è possibile esaminare i risultati ottenuti utilizzando il workflow illustrato nel capitolo 4.2.2. Come precedentemente menzionato, in questo contesto miriamo a individuare il modello con il valore più basso di RMSE sul set di test. Tale modello è rappresentato dal SVM radiale, evidenziato in rosso. Va notato che questo modello dimostra anche il miglior R^2 tra tutti quelli esaminati nel test set, evidenziato in blu.
Per stilare una sorta di classifica dei modelli, dal meno al più performante in base all'RMSE del test set, l'ordine risulta essere il seguente: random forest,

lasso, SGD, ridge, SVM lineare e infine SVM radiale. Questa analisi suggerisce che il modello SVM radiale presenta le migliori capacità predittive tra quelli considerati in questa analisi.

model	dataType	RMSE	Rsquared
Lasso	Test	0.159	0.611
Lasso	Training	0.128	0.742
Ridge	Test	0.154	0.638
Ridge	Training	0.117	0.791
Random forest	Test	0.177	0.549
Random forest	Training	0.0668	0.961
SGD	Test	0.159	0.612
SGD	Training	0.0133	0.998
svmLinear	Test	0.153	0.642
svmLinear	Training	0.113	0.797
svmRadial	Test	0.152	0.643
svmRadial	Training	0.0637	0.938

Tabella 4.2: Permormance modelli CX1

Le performance nella tabella sopra possono essere rappresentate graficamente, come illustrato in dettaglio nel capitolo 4.2.2, e possono essere osservate nella figura C.1. È interessante notare come la sparsità dei punti aumenti nel test set per tutti i modelli. Questo è un comportamento normale, poiché ci si aspetta che le prestazioni del modello siano inferiori sul test set rispetto al train set.

4.4 Modello con le performance migliori

Il modello SVM radiale risulta essere il migliore sia nel confronto del ricampionamento, sia per quanto riguarda le sue capacità predittive sul validation set e le sue metriche possono essere viste nella tabella 4.2. Identificato il modello con le performance migliori sono state identificate le top 100 variabili importanti seguendo il workflow delineato nel capitolo 4.1.1. Queste variabili sono state rappresentate graficamente nella figura 4.3. Guardando il grafico, è im-

mediatamente evidente la preponderanza delle variabili di espressione(ciano) rispetto a quelle di metilazione(rosso), con una proporzione di 91 a 9. Questo evidenzia che i dati di espressione hanno avuto un ruolo prevalente nella creazione del modello.

Figura 4.3: top 100 varImp.

Pur essendo il modello con le migliori performance, è evidente un notevole peggioramento nell'Rsquared tra il train set e il test set, con una perdita di circa il 30%. In particolare, il valore di Rsquared scende da 0.938 nel train set a 0.643 nel test set. Questa differenza è chiaramente visibile nella figura 4.4, dove si osserva un maggior numero di punti che si discostano dal valore predetto ideale rappresentato dalla linea rossa.

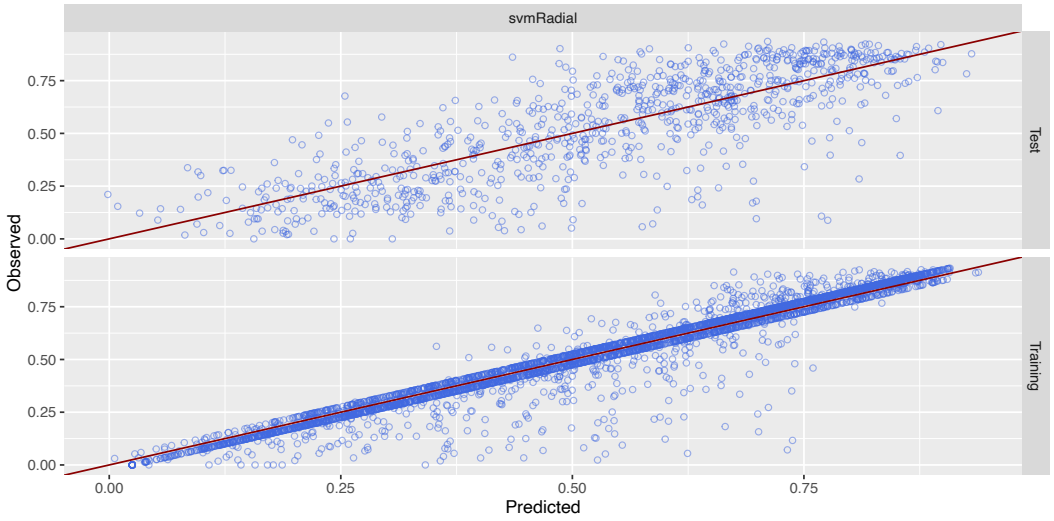


Figura 4.4: SVM radiale performance

4.4.1 Matrice di confusione

La motivazione alla base di questo sottocapitolo risiede nell'aspettativa di ottenere prestazioni migliori sul test set rispetto a quelle effettivamente osservate. Al fine di comprendere e affrontare questo peggioramento, si è deciso di costruire una matrice di confusione con l'obiettivo di identificare eventuali problematiche in specifici campioni che potrebbero influenzare negativamente l'intera analisi. Questa analisi è stata applicata sia al set di addestramento che al set di test, per ottenere una comprensione completa delle prestazioni del modello. In pratica, la signature è stata suddivisa in tre categorie basate sui suoi valori, utilizzando il primo e il terzo quartile come punti di riferimento. Questa suddivisione ha generato tre classi: valori alti, valori bassi e valori intermedi. È importante sottolineare che questa suddivisione è stata effettuata sulla matrice iniziale, prima della creazione del set di addestramento e del set di test. Inoltre, gli stessi parametri di suddivisione sono stati applicati in modo coerente sia al set di addestramento che al set di test, garantendo uniformità nel processo di categorizzazione nelle tre classi. Una volta completata la suddivisione delle classi sia per i valori reali che per quelli predetti in entrambi i dataset, possiamo procedere al confronto tra le classi dei valori reali e quelle dei

valori predetti. I risultati di questo confronto sul train set sono rappresentati come matrice di confusione (CM) nella Tabella 4.3. Come è evidente, non si verificano casi in cui un valore reale alto venga predetto come basso o viceversa. Questo indica che il modello non presenta problemi gravi di classificazione e funziona bene sul train set. Lo stesso processo è stato ripetuto anche sul test set e i risultati sono riportati nella Tabella 4.4. In questo caso, si identificano 3 campioni con valori reali bassi che vengono predetti come alti. La rimozione di questi campioni dal modello porterebbe a un guadagno in termini di Rsquared sul test set del 1.8%, aumentando da 0.643 a 0.661. Sebbene possa sembrare un miglioramento modesto, è significativo considerando l'influenza di soli tre campioni.

	Predetto		
Osservato	High	Low	Middle
High	750	0	145
Low	0	767	127
Middle	29	29	1779

Tabella 4.3: CM train set

	Predetto		
Osservato	High	Low	Middle
High	116	0	121
Low	3	90	145
Middle	35	28	369

Tabella 4.4: CM test set

Sulle due matrici di confusione è stata calcolata l'accuratezza. Questa misura rappresenta il rapporto tra il numero di previsioni corrette e il numero totale di previsioni ed è calcolata come 1 meno il tasso di errore (err.rate). Più l'accuratezza si avvicina a 1, migliori sono le prestazioni del modello. Di seguito, sono riportati i risultati ottenuti per entrambi i dataset:

	Accuracy
train	0.909
test	0.695

Tabella 4.5: Accuratezza CM

Il train set presenta ovviamente un valore di accuratezza migliore rispetto al test set come atteso. In generale il risultato che si ottiene dall'analisi di entrambi i dataset è che il modello performa bene e non ci sono problemi intrinseci nelle predizioni.

4.5 Influenza dei diversi tipi tumorali sull'analisi

Come discusso nel capitolo 4.4.1 il problema delle prestazioni del modello non sembra essere legato alle sue capacità predittive. Pertanto, si è cercato ulteriormente di individuare le cause del peggioramento delle prestazioni altrove. In questo contesto, sono state esaminate le differenze tra i diversi tipi tumorali presenti nei campioni del dataset. Notiamo che i diversi tipi tumorali non sono rappresentati in modo uniforme nel dataset, ma ciascuno ha un numero di campioni differente. Questa disparità potrebbe influenzare l'analisi complessiva, poiché alcuni tipi tumorali potrebbero avere un impatto negativo sull'intera analisi. Inoltre, è importante considerare che, nonostante le signature siano pan-cancer, potrebbe verificarsi per costruzione una maggiore affidabilità per alcuni tipi tumorali rispetto ad altri, proprio a causa delle differenze nella biologia sottostante. Per condurre questa analisi, l'oggetto "*prediction*", ottenuto nel capitolo 4.2.2, è stato arricchito con le informazioni relative al tipo tumorale di ciascun campione. Ciò consente di esaminare le performance all'interno di ciascun tipo tumorale. Una volta aggiunta questa informazione aggiuntiva è possibile calcolare il valore di RMSE e di Rsquared considerando non solo il tipo di dati, ma anche il tipo tumorale come variabile aggiuntiva. Inoltre, è stato incluso "*count = n()*" per ottenere il numero esatto di campioni per ciascun tipo tumorale.

Successivamente, è stata creata una rappresentazione grafica dei valori di Rsquared utilizzando "*ggplot*", il cui risultato è mostrato nella figura 4.5. Questa figura sarà punto di riferimento per l'intero paragrafo. Nel pannello A sono presenti i valori di Rsquared suddivisi per tipo tumorale tra il train set e il test set. Tra tutti i valori, è evidente il caso del tipo tumorale THYM nel test set, che mostra un Rsquared di 1. Un valore di Rsquared pari a 1 indica una capacità di predizione perfetta. Tuttavia, è importante notare che i valori predetti e osservati non coincidono. Questa discrepanza è dovuta al fatto che la funzione "*R2*" del pacchetto *caret* calcola la correlazione tra i valori osservati e i valori predetti, elevata al quadrato. Per affrontare questa problematica, è stato introdotto un nuovo attributo chiamato "delta", che rappresenta la

somma dei valori assoluti delle differenze tra i valori osservati e quelli predetti. Questo attributo ci consente di ottenere una visione chiara delle discrepanze effettive tra i valori predetti e quelli osservati. Nel pannello B, viene mostrato il valore del "delta", il quale è stato ordinato in base al numero di campioni per rendere la visualizzazione più significativa. È evidente che all'aumentare del numero di campioni per tipo tumorale, si osserva un aumento della somma delle differenze, il che è coerente con le aspettative.

Al fine di avere un'informazione indipendente dalla diversa numerosità dei campioni, si è introdotto un nuovo attributo chiamato "delta medio" ($\Delta.med$). Questo attributo rappresenta la media delle differenze tra i valori predetti e osservati e lo si ottiene andando a dividere il "delta" per il numero di campioni. In questo modo, si ottiene una misura della differenza media tra valore predetto e valore osservato per ciascun campione di quel tipo tumorale. Utilizzando il delta medio posso ordinare i tipi tumorali in ordine crescente. I tipi tumorali con valori più piccoli di delta medio indicano una minore differenza media tra valori predetti e osservati, mentre quelli con valori più alti mostrano una maggiore variazione. Questo risultato lo si può osservare nel pannello C, dove non si osserva una semplice linea retta, ma una variazione significativa tra i tipi tumorali. In particolare, notiamo che il tipo tumorale THYM, che ha un Rsquared di 1, non è il tipo tumorale con la minore variazione tra valori predetti e osservati. Questo dimostra che il valore di Rsquared da solo non è sufficiente per determinare quale tipo tumorale ha le migliori prestazioni.

Infine, per comprendere meglio l'influenza che ciascun tipo tumorale ha sulle prestazioni del modello, si è calcolato un ulteriore attributo chiamato "delta pesato". Questo attributo rappresenta la media ponderata delle differenze tra i valori predetti e osservati, calcolate come il delta moltiplicato per il numero di campioni di un determinato tipo tumorale e diviso per il numero totale di campioni in analisi (4533). Nel pannello D, sono mostrati i risultati del delta pesato ordinati in ordine decrescente. Maggiore è il valore del delta pesato, maggiore è il peso che la differenza tra il valore osservato e quello predetto ha sull'analisi. Notiamo che, sebbene non sia una corrispondenza perfetta, l'ordine generato segue in gran parte l'ordine della numerosità dei campioni di

ciascun tipo tumorale.

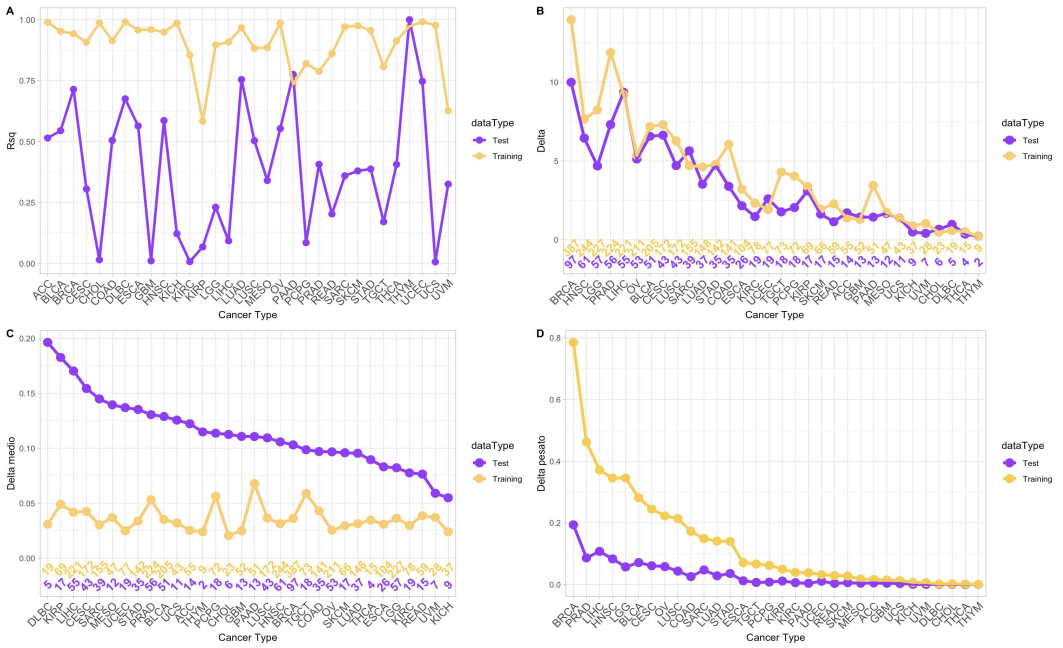


Figura 4.5: ggplot prediction cancer type. Nel pannello A, sono rappresentati i valori di R-squared per i diversi tipi tumorali. Nel pannello B, sono visualizzati i valori di delta, ordinati in modo decrescente per il numero di campioni per tipo tumorale. Nel pannello C, i valori di delta medio sono stati ordinati in modo decrescente secondo il loro valore nel test set. Infine, nel pannello D, sono riportati i valori di delta pesato, ordinati in modo decrescente in base al loro valore nel train set.

Ora non resta che andare ad associare le informazioni fornite dal delta medio e delta pesato per identificare i tipi tumorali che hanno un maggiore impatto negativo sulle performance del modello nel test set. Per ottenere questo obiettivo, è stato assegnato un rank in modo decrescente sia al delta medio che al delta pesato e sono stati rappresentati graficamente nella figura 4.6. Il quadrato blu evidenziato rappresenta le correlazioni tra i primi 10 delta medi più elevati e i primi 10 delta pesati più elevati. Questo approccio consente di identificare i tipi tumorali in cui le variazioni medie per campione influiscono maggiormente sulle performance del modello. In questo caso, si riscontrano cinque tipi tumorali che soddisfano queste condizioni: LIHC, PRAD, BLCA, CESC e SARC. Successivamente, nel capitolo 6, si analizzerà se esistono delle

corrispondenze tra i "peggiori" tipi tumorali tra le signature, in modo da comprendere meglio le caratteristiche dei diversi tipi tumorali e se esistono delle relazioni con le signature.

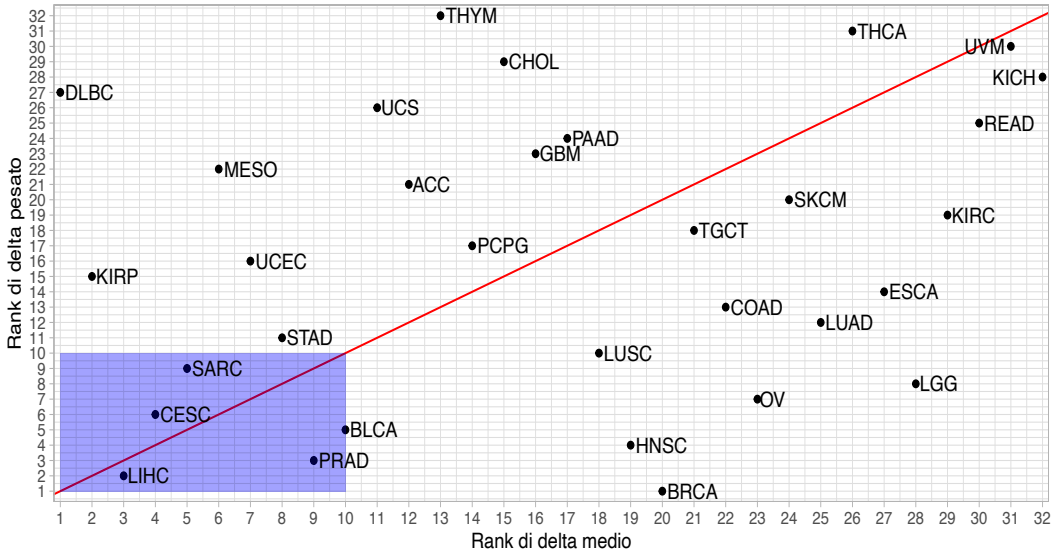


Figura 4.6: correlazione $\Delta medio$ e $\Delta pesato$. I ranghi sono in ordine decrescente di valori di $\Delta medio$ e $\Delta pesato$ da 1 a 32. Il rango 1 indica il valore più grande, mentre il rango 32 indica il valore più piccolo

4.5.1 Cancer type come variabile

Una volta identificato l'influenza del tipo tumorale sull'analisi, è stata considerata l'opzione di utilizzarlo come ulteriore variabile nel modello. Per farlo, è stato necessario convertirlo in una variabile categorica. Questo tipo di variabile è supportato solo da due dei modelli di regressione scelti: SGD Boosting e random forest. Tuttavia, solo il primo ha mostrato un minimo miglioramento delle prestazioni sul test set, ma identifica il tipo tumorale come la variabile più influente, evidenziando ulteriormente l'importanza di questo fattore nell'analisi. Sebbene il miglioramento sia molto basso, da 0.612 passo a 0.617 come Rsqaured nel test set, è importante sottolineare che questo modello non era il migliore in generale per l'analisi, come evidenziato nel Capitolo 4.3.2.

Capitolo 5

Performance modelli di regressione sulle signature: CX3, CX5 e CX2

In questo capitolo, verranno presentati i risultati ottenuti relativi alle tre signature rimanenti, seguendo il workflow precedentemente definito per l'analisi della signature CX1 nel capitolo 4. L'ordine di analisi è determinato dalla heatmap 2.2 e dalla distribuzione dei valori (figura 2.4), si inizia con la signature che mostra la migliore distribuzione dei valori e procederemo verso quella con la peggiore.

5.1 Confronto risultati tra i modelli ottenuti per la signature CX3

Nel primo confronto, ci concentriamo sull'analisi dei risultati di ricampionamento dei modelli per la signature CX3. La figura 5.1 presenta in modo chiaro i valori ottenuti dai diversi modelli, con il modello Ridge che mostra il valore più basso di RMSE. Questo indicatore suggerisce una buona capacità del modello di adattarsi ai dati e di minimizzare gli errori predittivi. Tuttavia, è rilevante notare che il modello Ridge, sebbene eccelli nell'RMSE, non è il modello con il valore più alto di R-squared. In effetti, l'SVM radiale supera gli altri modelli in termini di R-squared. Questa analisi iniziale non consente di stabilire con certezza il modello migliore per la predizione della signature

CX3. Tuttavia, identifica due candidati principali: il modello Ridge per la sua eccellente gestione dell'RMSE e l'SVM radiale per il suo elevato R-squared. Sarà fondamentale condurre ulteriori analisi utilizzando il validation set per determinare quale dei due modelli sia più adatto e comprendere meglio come ciascun modello si comporta in contesti diversi. I dettagli relativi ai valori di ciascun modello sono riportati nella tabella 5.1.

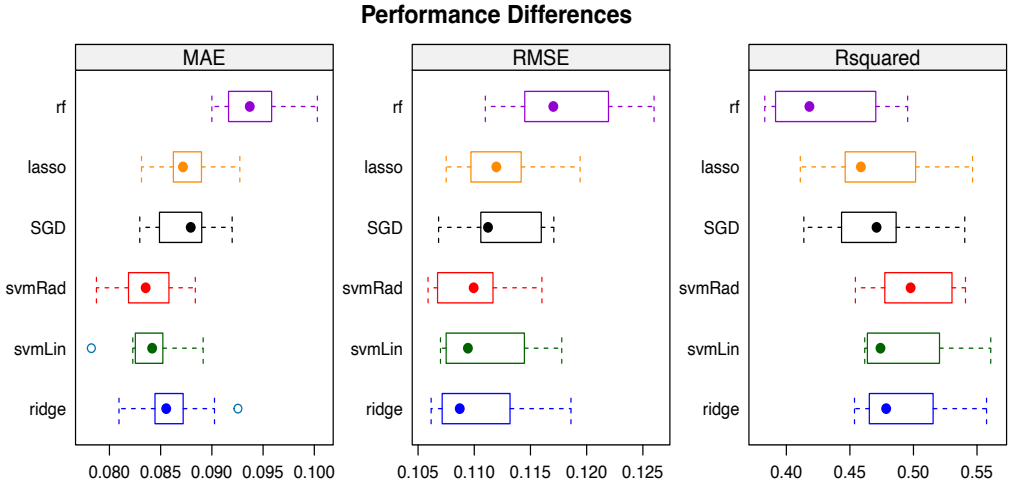


Figura 5.1: Boxplot ricampionamento

model	RMSE	Rsqaured	model	RMSE	Rsqaured
Random forest	0.117	0.418	svmRad	0.110	0.498
lasso	0.112	0.459	svmLin	0.109	0.474
SGD	0.111	0.471	ridge	0.108	0.479

Tabella 5.1: Valore mediano di ricampionamento

Prima di esaminare le prestazioni sul validation set, è importante valutare se le differenze tra i modelli sono statisticamente significative o sono semplicemente il risultato della casualità dei dati. Come evidenziato nella figura 5.2, SVM radiale e Ridge presentano un p-value di 1, suggerendo che le differenze osservate sono probabilmente dovute al caso e non riflettono differenze reali tra i modelli. Questo sottolinea ulteriormente l'importanza di utilizzare un validation set per determinare con certezza quale modello si comporta meglio per la signature CX3.

```
p-value adjustment: bonferroni
Upper diagonal: estimates of the difference
Lower diagonal: p-value for H0: difference = 0

RMSE
       lasso     ridge      rf         SGD        svmLin     svmRad
lasso             0.0020514 -0.0054248  0.0003363  0.0013190  0.0026244
ridge  0.0184056            -0.0074762 -0.0017151 -0.0007325  0.0005730
rf     0.0016407 8.776e-06              0.0057611  0.0067438  0.0080493
SGD    1.0000000 0.4157434  0.0004770               0.0009827  0.0022882
svmLin 0.2513707 0.4934078  0.0001274  1.0000000               0.0013055
svmRad 0.0146222 1.0000000  6.844e-06  0.1285882  0.5956566
```

Figura 5.2: Differenze tra modelli

5.1.1 Confronto performance

Nella tabella riportata di seguito, è possibile esaminare i risultati delle performance dei modelli. Il modello con il valore più basso di RMSE nel test set è SVM radiale, evidenziato in rosso, il quale presenta anche l' R^2 più elevato (in blu) tra tutti i modelli esaminati. Le performance illustrate nella tabella 5.2 sono rappresentate graficamente nella figura D.1.

model	dataType	RMSE	Rsquared
Lasso	Test	0.110	0.512
Lasso	Training	0.093	0.649
Ridge	Test	0.1083	0.528
Ridge	Training	0.082	0.737
Random forest	Test	0.115	0.483
Random forest	Training	0.045	0.956
SGD	Test	0.110	0.509
SGD	Training	0.042	0.939
svmLinear	Test	0.1078	0.529
svmLinear	Training	0.083	0.723
svmRadial	Test	0.106	0.541
svmRadial	Training	0.055	0.885

Tabella 5.2: Permormance modelli CX3

5.1.2 Best model e matrice di confusione

A differenza del confronto di ricampionamento, in cui sono emersi due modelli principali, nella valutazione sul validation set non ci sono incertezze. L'SVM radiale si dimostra superiore, mentre il modello Ridge si posiziona solo al terzo posto, preceduto anche dal SVM lineare, tra i sei modelli esaminati. Il modello migliore risulta quindi essere SVM radiale, e le sue 100 variabili principali possono essere visualizzate nella figura 5.3. Ancora una volta, notiamo una predominanza di variabili di espressione (ciano) rispetto a variabili di metilazione (rosso), con una proporzione di 77 a 23.

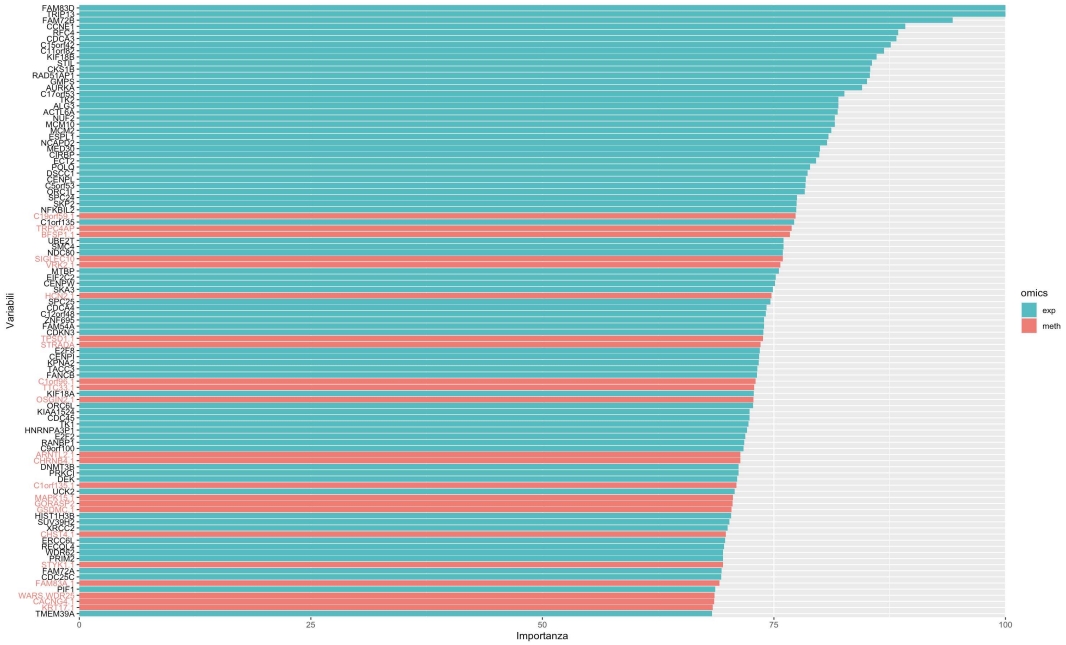


Figura 5.3: top 100 varImp.

Nonostante il modello mostri le prestazioni migliori, è evidente un significativo peggioramento dell'R-squared tra il train set e il test set, con una perdita di circa il 34%. In particolare, il valore di R-squared scende dal 0.885 nel train set a 0.541 nel test set.

Le matrici di confusione, ottenute seguendo il workflow descritto nel capitolo 4.4.1, sono presentate nelle tabelle 5.3 e 5.4, relative rispettivamente al train set e al test set. Nel train set, non si verificano casi in cui valori reali bassi vengono predetti come alti o viceversa. Nel test set, tuttavia, si riscontrano 2 campioni in cui i valori reali alti sono stati predetti come bassi. Rimuovendo

questi due campioni dal set, si otterrebbe un guadagno in termini di Rsquared pari all'0.8%, aumentando da 0.541 a 0.549.

L'accuratezza calcolata sulle due matrici di confusione risulta essere buona, con un lieve peggioramento rispetto a quanto osservato per la signature CX1. Tuttavia, questo è in linea con il peggioramento delle performance del modello. In generale, il modello continua a fornire prestazioni soddisfacenti e non si riscontrano gravi problemi nelle predizioni.

	Predetto		
Osservato	High	Low	Middle
High	704	0	219
Low	0	636	237
Middle	30	16	1784

Tabella 5.3: CM train set

	Predetto		
Osservato	High	Low	Middle
High	98	2	111
Low	0	74	186
Middle	53	15	368

Tabella 5.4: CM test set

	Accuracy
train	0.862
test	0.595

Tabella 5.5: Accuratezza CM

5.1.3 Influenza tipi tumorali

Nel pannello A della figura 5.4, è evidente che un tipo tumorale, il THCA, non presenta un valore di Rsquared. Questa situazione è il risultato di una deviazione standard pari a zero, il che porta ad avere una correlazione con un valore "NA". Allo stesso modo, come osservato per la signature CX1, il tipo tumorale THYM presenta un Rsquared di 1 nel test set. È rilevante notare che anche per la signature CX3 si osserva una variazione significativa del delta medio tra i diversi tipi tumorali, come evidenziato nel pannello C. Ancora una volta, la variazione media non dipende dalla numerosità dei campioni, sottolineando la robustezza di queste osservazioni.

Nella figura 5.5, l'analisi dell'associazione tra il delta medio e il delta pesato rivela che tra i primi 10 tipi tumorali, ben 5 di essi (LIHC, CESC, BLCA, SARC e OV) sono correlati. Questo suggerisce che questi cinque tipi tumorali

hanno un impatto negativo maggiore rispetto agli altri sulla capacità predittiva del modello.

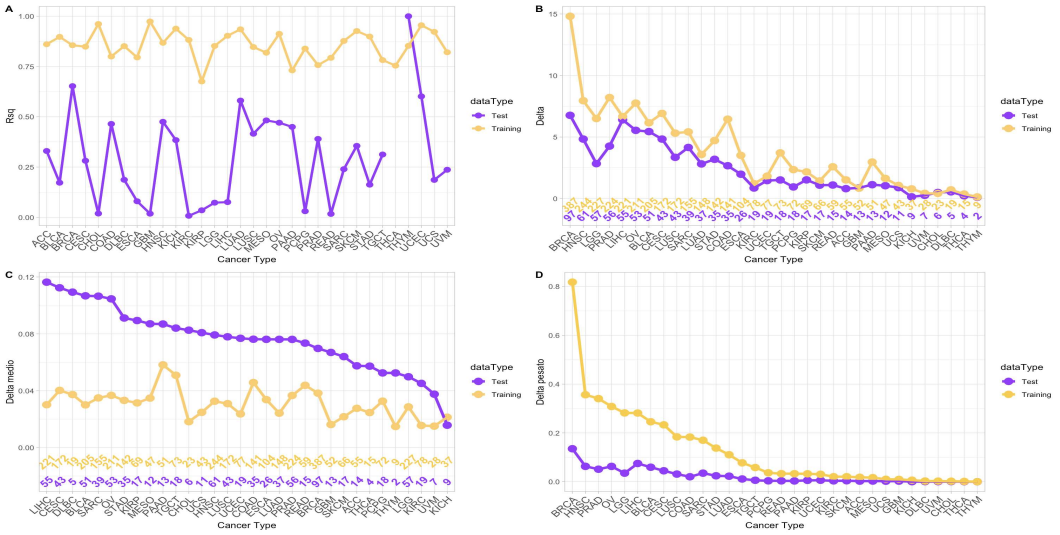


Figura 5.4: ggplot prediction cancer type. Per una descrizione dettagliata, fare riferimento alla figura 4.5.

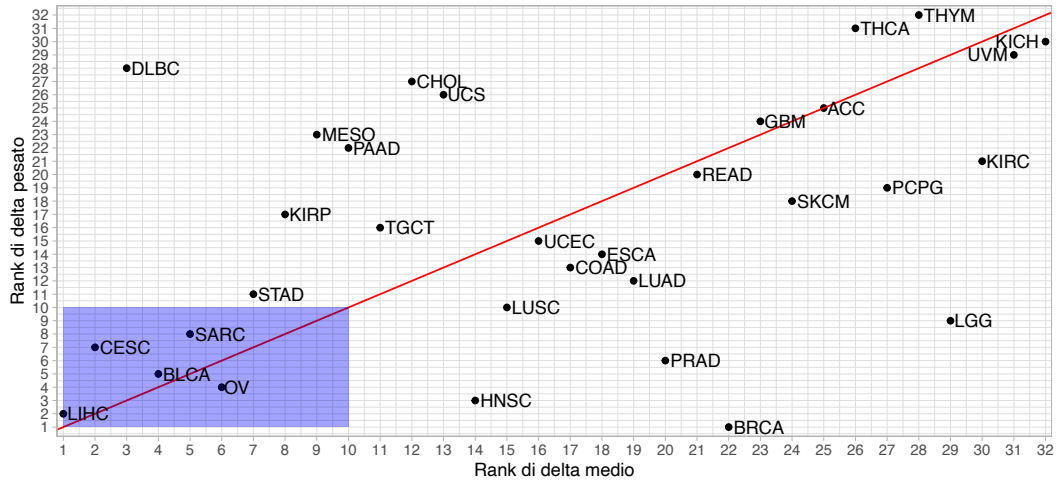


Figura 5.5: correlazione $\Delta medio$ e $\Delta pesato$. Il rango 1 indica il valore più grande, mentre il rango 32 indica il valore più piccolo.

Il modello ottenuto con SGD, utilizzando il tipo tumorale come variabile esplicativa, ha identificato il tipo tumorale come la variabile più influente, sottolineando ulteriormente l'importanza di questo fattore nell'analisi. Nonostante l'incremento dell'R-squared non sia molto elevato, passando da 0.509 a 0.521 nel test set, questo miglioramento può comunque essere significativo in termini di precisione predittiva.

5.2 Confronto risultati tra i modelli ottenuti per la signature CX5

Il confronto del ricampionamento tra modelli, rappresentato nella figura 5.6, evidenzia che il modello Ridge si distingue come il migliore per il valore di RMSE, mentre il modello SVM radiale si posiziona come il migliore per il valore di Rsquared. In questo contesto, il modello Ridge si colloca al terzo posto, superato anche dal modello SVM lineare.

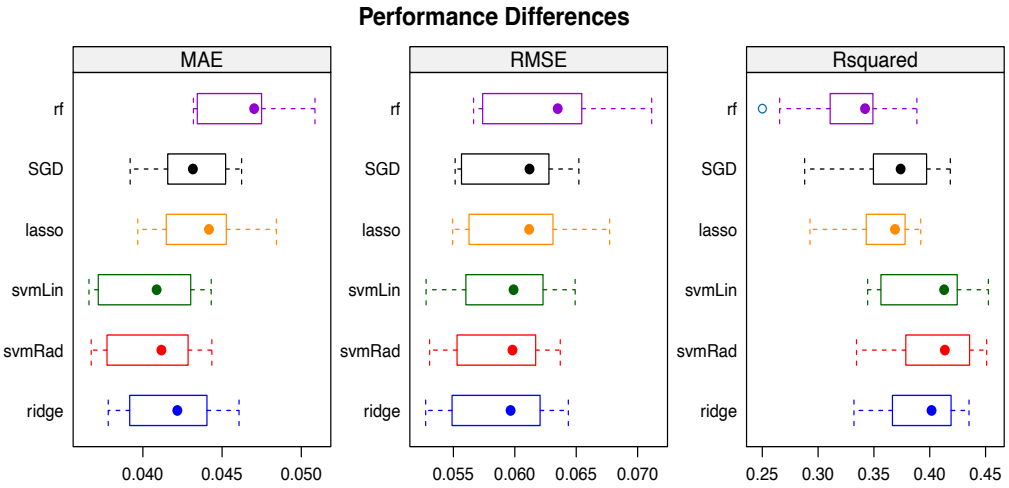


Figura 5.6: Boxplot ricampionamento

model	RMSE	Rsqaured	model	RMSE	Rsqaured
Random forest	0.063	0.342	svmLin	0.0599	0.413
SGD	0.0612	0.374	svmRad	0.0598	0.414
lasso	0.0611	0.369	ridge	0.0597	0.402

Tabella 5.6: Valore mediano di ricampionamento

Per quanto riguarda le differenze tra i modelli, nella figura 5.7 si osserva che i confronti Ridge-SVM radiale e Ridge-SVM lineare presentano un p-value di 1, mentre SVM radiale e SVM lineare presentano un p-value di 0.72. In tutti e tre i casi, il valore di p-value ottenuto suggerisce che le differenze osservate tra i modelli sono verosimilmente attribuibili al caso.

```
p-value adjustment: bonferroni
Upper diagonal: estimates of the difference
Lower diagonal: p-value for H0: difference = 0

RMSE
       lasso     ridge      rf         SGD        svmLin     svmRad
lasso            0.0017627 -0.0019890  0.0004992  0.0015256  0.0019618
ridge  0.0008481           -0.0037517 -0.0012634 -0.0002371  0.0001991
rf     0.0032640 0.0001756              0.0024882  0.0035146  0.0039508
SGD    1.0000000 0.0026443  0.0065378              0.0010263  0.0014626
svmLin 0.0725441 1.0000000  0.0027081  0.3568292              0.0004362
svmRad 0.0041754 1.0000000  0.0003265  0.0086223  0.7246455
```

Figura 5.7: Differenze tra modelli

5.2.1 Confronto performance

Il modello che evidenzia le prestazioni superiori nel test set risulta essere, anche se di poco, il Ridge. Le sue performance sono illustrate nella tabella 5.7 e sono rappresentate graficamente nella figura D.3.

model	dataType	RMSE	Rsquared
Lasso	Test	0.0617	0.390
Lasso	Training	0.0502	0.583
Ridge	Test	0.0596	0.432
Ridge	Training	0.0421	0.728
Random forest	Test	0.0650	0.341
Random forest	Training	0.0241	0.959
SGD	Test	0.0617	0.385
SGD	Training	0.0150	0.969
svmLinear	Test	0.0604	0.432
svmLinear	Training	0.0397	0.756
svmRadial	Test	0.0601	0.430
svmRadial	Training	0.0301	0.864

Tabella 5.7: Permormance modelli CX5

5.2.2 Best model e matrice di confusione

Il modello che presenta le migliori prestazioni tra tutti è il Ridge, e le sue 100 variabili principali sono illustrate nella figura 5.8. In questo contesto, si osserva una completa prevalenza di variabili di espressione e l'assenza di variabili di metilazione.

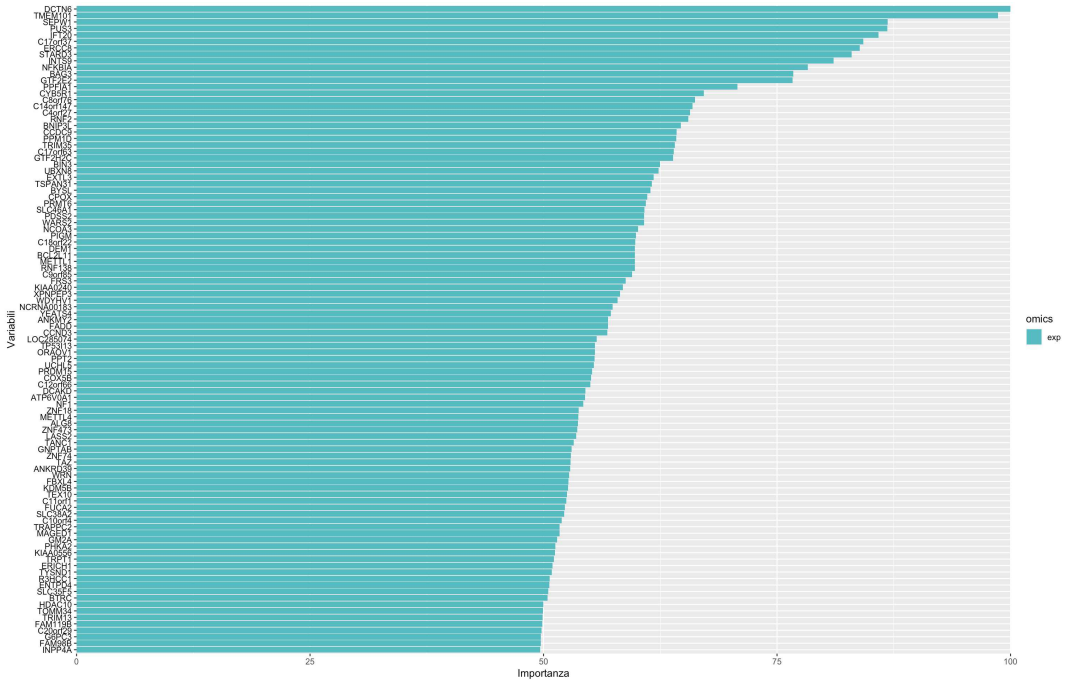


Figura 5.8: top 100 varImp.

Anche in questo caso, si osserva un significativo peggioramento delle prestazioni tra il set di addestramento e quello di test, con una perdita del 30% di Rsquared. Infatti le matrici di confusione, presentati nelle tabelle 5.13 e 5.14, evidenziano alcune notevoli discrepanze. Nel train set si verificano 4 casi in cui valori reali bassi vengono predetti come alti e 2 casi in cui valori reali alti vengono predetti bassi. Nel test set invece si riscontrano 9 campioni in cui i valori reali alti sono stati predetti come bassi e 9 campioni nella situazione opposta. Rimuovendo questi sei campioni dal train set, si otterrebbe un guadagno in termini di Rsquared pari allo 0.4%, passando da 0.728 a 0.732. Mentre rimuovendo i 18 campioni dal test set si otterrebbe un guadagno pari al 5%, aumentando da 0.432 a 0.482.

L'accuratezza calcolata sulle due matrici di confusione risulta essere accetta-

bile, con un notevole peggioramento nel train set rispetto a quanto osservato per le altre due signature.

	Predetto		
Osservato	High	Low	Middle
High	709	2	196
Low	4	465	437
Middle	194	439	1180

Tabella 5.8: CM train set

	Predetto		
Osservato	High	Low	Middle
High	153	9	65
Low	9	92	125
Middle	65	125	264

Tabella 5.9: CM test set

	Accuracy
train	0.649
test	0.561

Tabella 5.10: Accuratezza CM

5.2.3 Influenza tipi tumorali

Nel pannello A della figura 5.9, è evidente che un tipo tumorale, il THYM, non presenta un valore di Rsquared. Questa situazione è il risultato di una deviazione standard pari a zero, il che porta ad avere una correlazione con un valore "NA". È rilevante notare che anche per la signature CX5 si osserva una variazione significativa del delta medio tra i diversi tipi tumorali, come evidenziato nel pannello C e anche in questo caso la variazione media non dipende dalla numerosità dei campioni.

Nella figura 5.10, viene analizzata l'associazione tra il delta medio e il delta pesato, e emerge che tra i primi 10 tipi tumorali, solo 3 di essi sono correlati: BLCA, SARC e OV. Ciò suggerisce che questi tre tipi tumorali hanno un impatto negativo maggiore rispetto agli altri tipi tumorali sulla capacità predittiva del modello.

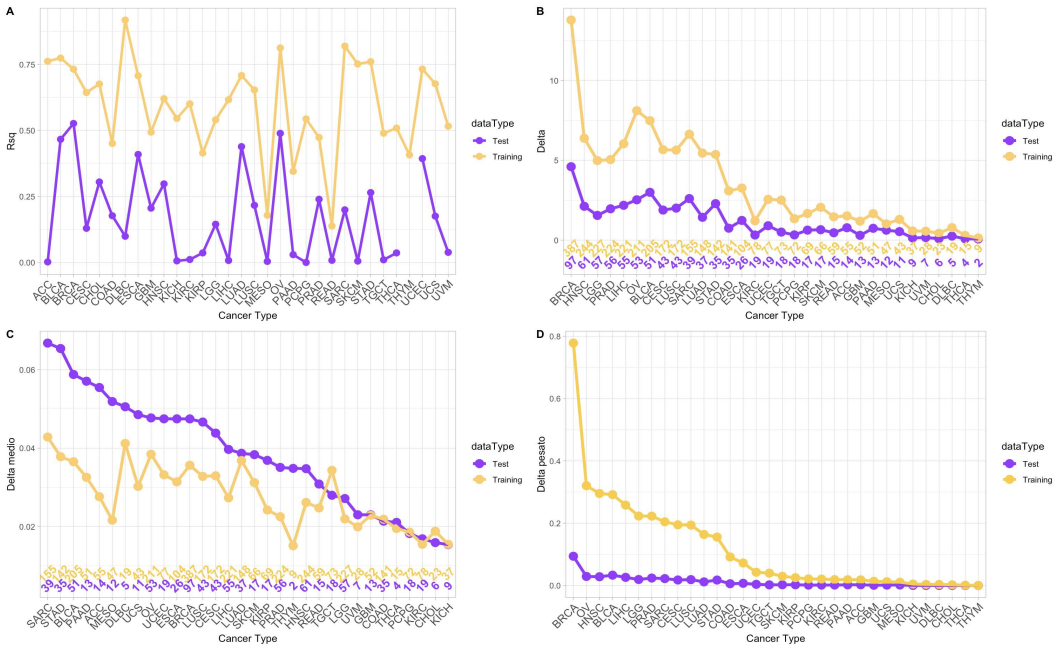


Figura 5.9: ggplot prediction cancer type. Per una descrizione dettagliata, fare riferimento alla figura 4.5.

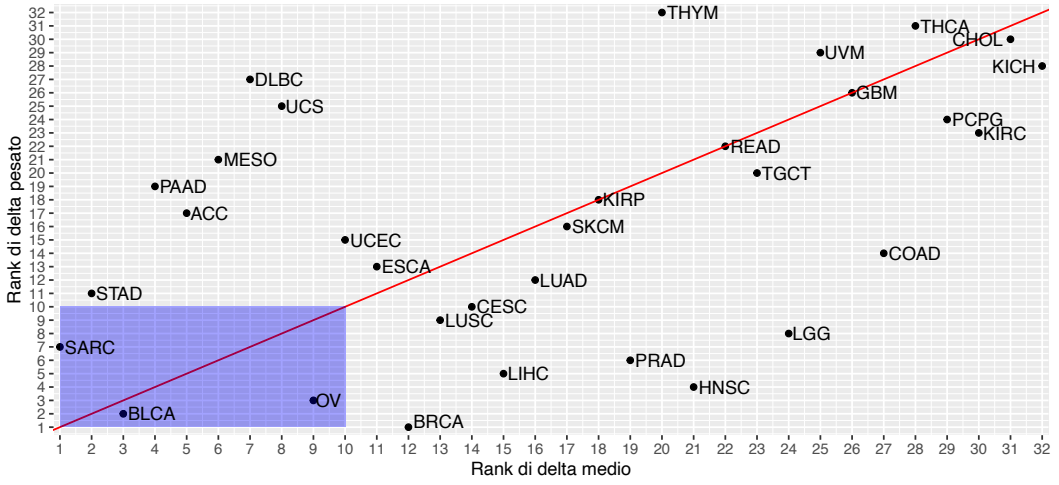


Figura 5.10: correlazione $\Delta medio$ e $\Delta pesato$. Il rango 1 indica il valore più grande, mentre il rango 32 indica il valore più piccolo.

Il modello ottenuto con SGD sfruttando il tipo tumorale come variabile esplicativa identifica il tipo tumorale come la variabile più influente, evidenziando ulteriormente l'importanza di questo fattore nell'analisi. Sebbene il miglioramento non sia molto elevato, da 0.385 passo a 0.418 come Rsqaured nel test set.

5.3 Confronto risultati tra i modelli ottenuti per la signature CX2

Il confronto del ricampionamento tra modelli, come evidenziato nella figura 5.11, dimostra che il modello lasso eccelle su tutti gli altri, sia in termini di valore di RMSE che di Rsquared.

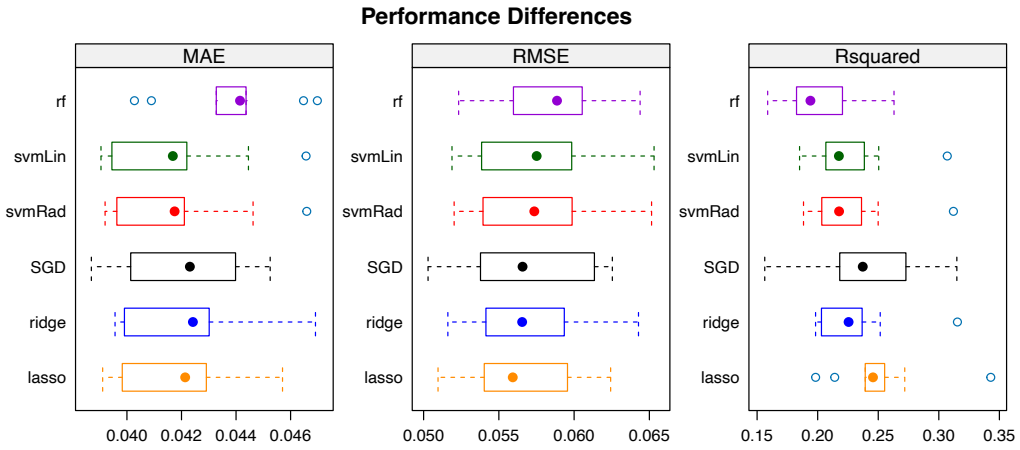


Figura 5.11: Boxplot ricampionamento

model	RMSE	Rsqaured	model	RMSE	Rsqaured
Random forest	0.0589	0.194	SGD	0.0566	0.237
svmLin	0.0575	0.218	ridge	0.0565	0.226
svmRad	0.0574	0.218	lasso	0.0559	0.246

Tabella 5.11: Valore mediano di ricampionamento

Per quanto riguarda le differenze tra i modelli, è possibile notare in figura 5.12 sei casi in cui il p-value risulta essere pari a 1, accompagnati da altri tre casi in cui il p-value è significativamente elevato. In conclusione, le divergenze osservate tra i modelli sembrano essere attribuibili al caso.

```
p-value adjustment: bonferroni
Upper diagonal: estimates of the difference
Lower diagonal: p-value for H0: difference = 0

RMSE
       lasso     ridge      rf         SGD        svmLin     svmRad
lasso            -7.371e-04 -2.013e-03 -3.844e-04 -1.171e-03 -1.189e-03
ridge  0.0744651            -1.276e-03  3.527e-04 -4.343e-04 -4.519e-04
rf     0.0004192 0.0282011               1.629e-03  8.417e-04  8.242e-04
SGD    1.0000000 1.0000000  0.0254147                -7.870e-04 -8.046e-04
svmLin 0.0728994 1.0000000  0.3907558  1.0000000               -1.754e-05
svmRad 0.0425545 0.9412754  0.4543670  1.0000000  1.0000000
```

Figura 5.12: Differenze tra modelli

5.3.1 Confronto performance

Il modello che evidenzia le prestazioni superiori nel test set risulta essere, anche se di poco, il Ridge. Le sue performance sono illustrate nella tabella 5.12 e sono rappresentate graficamente nella figura D.5.

model	dataType	RMSE	Rsquared
lasso	Test	0.0579	0.227
lasso	Training	0.0491	0.463
ridge	Test	0.0578	0.228
ridge	Training	0.0455	0.572
random forest	Test	0.0592	0.196
random forest	Training	0.0225	0.964
SGD	Test	0.0583	0.219
SGD	Training	0.00778	0.991
svmLinear	Test	0.0585	0.218
svmLinear	Training	0.0460	0.548
svmRadial	Test	0.0583	0.222
svmRadial	Training	0.0428	0.620

Tabella 5.12: Permormance modelli CX2

5.3.2 Best model e matrice di confusione

Il modello che presenta le migliori prestazioni tra tutti è il Ridge, e le sue 100 variabili principali sono illustrate nella figura 5.13. Anche in questo contesto, si osserva una completa prevalenza di variabili di espressione e l'assenza di variabili di metilazione. Tuttavia, risulta rilevante notare che la variabile clinica "purezza" si posiziona al primo posto in termini di importanza.

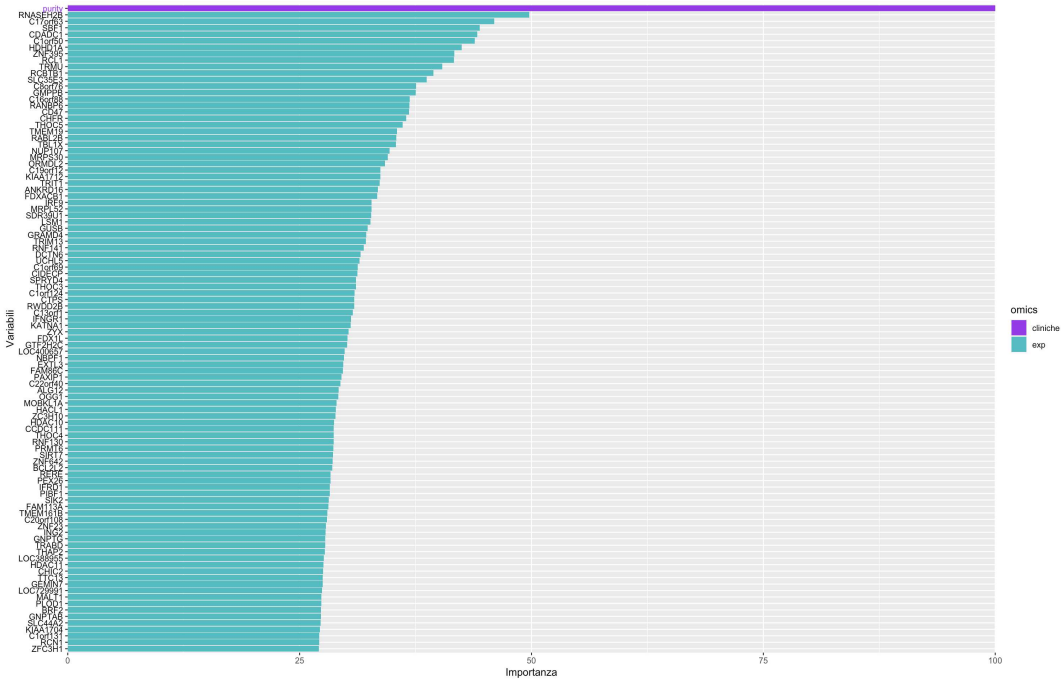


Figura 5.13: top 100 varImp.

Anche in questo caso, si osserva un significativo peggioramento delle prestazioni tra il set di addestramento e quello di test, con una perdita del 34% di Rsquared. Infatti le matrici di confusione, presentati nelle tabelle 5.13 e 5.14, evidenziano alcune notevoli discrepanze. Nel train set si verificano 12 casi in cui valori reali bassi vengono predetti come alti, mentre nel test set si riscontrano 19 campioni in cui i valori reali bassi sono stati predetti come alti. Rimuovendo questi 12 campioni dal train set, si otterrebbe un guadagno in termini di Rsquared pari allo 1.1%, passando da 0.572 a 0.583. Mentre rimuovendo i 19 campioni dal test set si otterrebbe un guadagno pari al 5%, aumentando da 0.228 a 0.278. La rimozione dei campioni non ha portato a un guadagno significativo, ma solo minimo. Questo è dovuto al fatto che il

modello incontra difficoltà nella previsione dei campioni con valori della signature bassi, tendenti allo zero. Ciò è evidente nelle due matrici di confusione, dove nella maggior parte dei casi in cui il valore reale è basso, la predizione avviene come mediano e non come basso. Questo problema sarà approfondito nel capitolo successivo.

L'accuratezza calcolata sulle due matrici di confusione risulta essere accettabile, con un notevole peggioramento nel train set rispetto a quanto osservato per le altre signature.

	Predetto		
Osservato	High	Low	Middle
High	460	0	443
Low	12	107	770
Middle	92	15	1727

Tabella 5.13: CM train set

	Predetto		
Osservato	High	Low	Middle
High	122	0	154
Low	19	1	124
Middle	78	0	409

Tabella 5.14: CM test set

	Accuracy
train	0.633
test	0.587

Tabella 5.15: Accuratezza CM

5.3.3 Influenza tipi tumorali

Nel pannello A della figura 5.14, si evidenzia un Rsquared di 1 per il tipo tumorale THYM. Tuttavia, a differenza delle altre signature, THYM presenta un delta medio tra i più piccoli (come mostrato nel pannello C), indicando che l'Rsquared elevato riflette le performance predittive specifiche per THYM. Anche per la signature CX2, nel grafico riportato nel pannello C, si osserva una notevole variazione del delta medio tra i diversi tipi tumorali nel test set. Questa variazione, come evidenziato, non dipende dalla numerosità dei campioni. Nella figura 5.15, viene esaminata l'associazione tra il delta medio e il delta pesato. Emerge che tra i primi 10 tipi tumorali, cinque di essi sono correlati: PRAD, OV, LIHC, SARC e STAD. Questo suggerisce che tali tipi tumorali hanno un impatto negativo maggiore sulla capacità predittiva del

modello rispetto agli altri tipi tumorali.

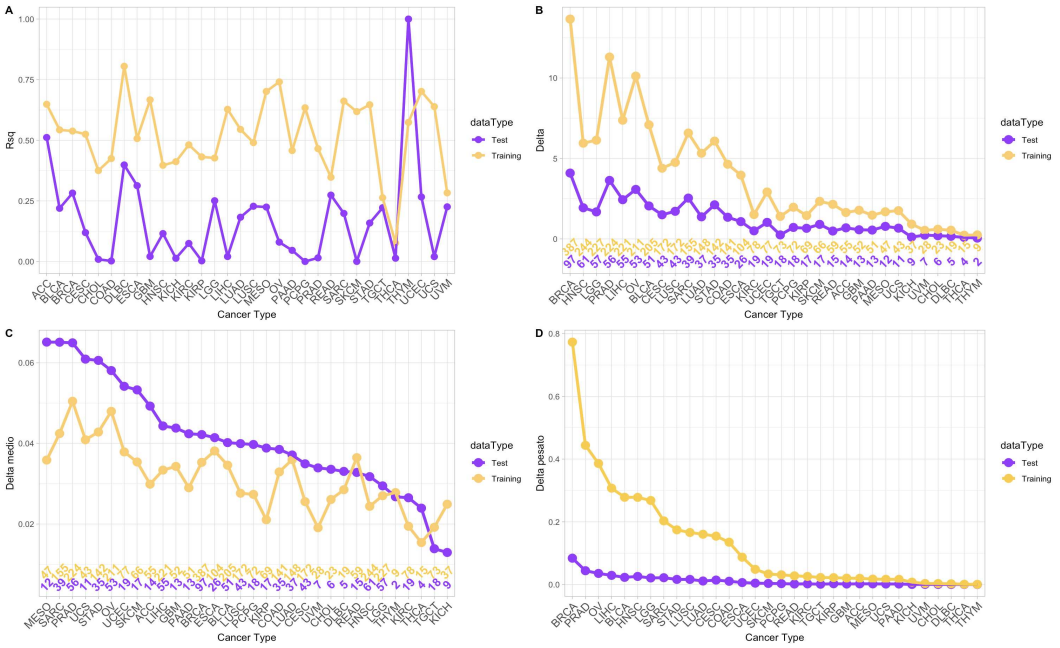


Figura 5.14: ggplot prediction cancer type. Per una descrizione dettagliata, fare riferimento alla figura 4.5.

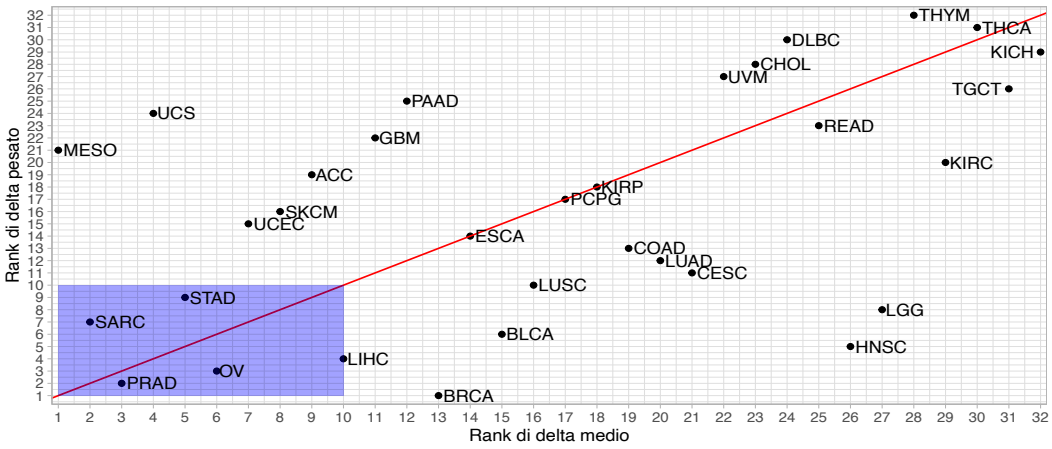


Figura 5.15: correlazione $\Delta medio$ e $\Delta pesato$. Il rango 1 indica il valore più grande, mentre il rango 32 indica il valore più piccolo.

Il modello ottenuto con SGD, utilizzando il tipo tumorale come variabile esplicativa, pone in evidenza il ruolo significativo di questa variabile nelle previsioni. Nonostante si abbia una diminuzione lieve dell'R^2, passando da 0.219 a 0.210 nel test set, la variabile tipo tumorale rimane la variabile più influente per il modello.

Capitolo 6

Confronto signature

Tra tutti i modelli testati per le diverse signature, è emerso chiaramente che il Random Forest è costantemente il modello con le performance più basse, seguito dal SGD Boosting, il quale mostra una notevole suscettibilità all'overfitting durante l'addestramento. In contrasto, Ridge e SVM sono i modelli che dimostrano le performance più elevate e, inoltre, sono caratterizzati da tempi computazionali inferiori rispetto agli altri. Il modello Ridge, ad esempio, presenta un tempo di esecuzione dell'ordine dei minuti, mentre gli SVM richiedono un tempo dell'ordine delle ore. Entrambi risultano sicuramente più efficienti rispetto a SGD e Random Forest, i quali richiedono persino giorni.

Nella tabella sottostante, si evidenzia un calo delle performance dei migliori modelli per le signature passando dalla CX1 alla CX2. Questo declino segue la distribuzione dei valori presenti nelle signature (si veda figura 2.4). Per quanto riguarda la differenza tra i set di allenamento e test, sembra rimanere costante per tutte le signature, manifestando una diminuzione delle performance quando si passa dal training al test set. È plausibile ipotizzare che all'aumentare del numero di valori della signature vicini allo zero, i modelli incontrino difficoltà nel trovare relazioni predittive tra le variabili e le signature stesse. Nella figura 6.1, sono evidenziati in blu i campioni con valore zero della signature e in rosso quelli con valore diverso da zero ma inferiore a 0.1, corrispondente al 10% di attività della signature. Si nota chiaramente come il numero di campioni con valori inferiori a 0.1 aumenti tra le signature, mostrando una

stretta correlazione con il peggioramento delle performance che si è osservato passando dalla CX1 alla CX2. In particolare, però, si nota che sia la CX5 che la CX2 presentano un numero totale simile di campioni con valori di signature inferiori a 0.1 (somma tra le due colonne). Tuttavia, le performance della CX2 sono significativamente peggiori rispetto alla CX5. Questo suggerisce che la signature CX2 potrebbe non avere un riscontro biologico reale, ma piuttosto rappresentare esclusivamente un risultato matematico.

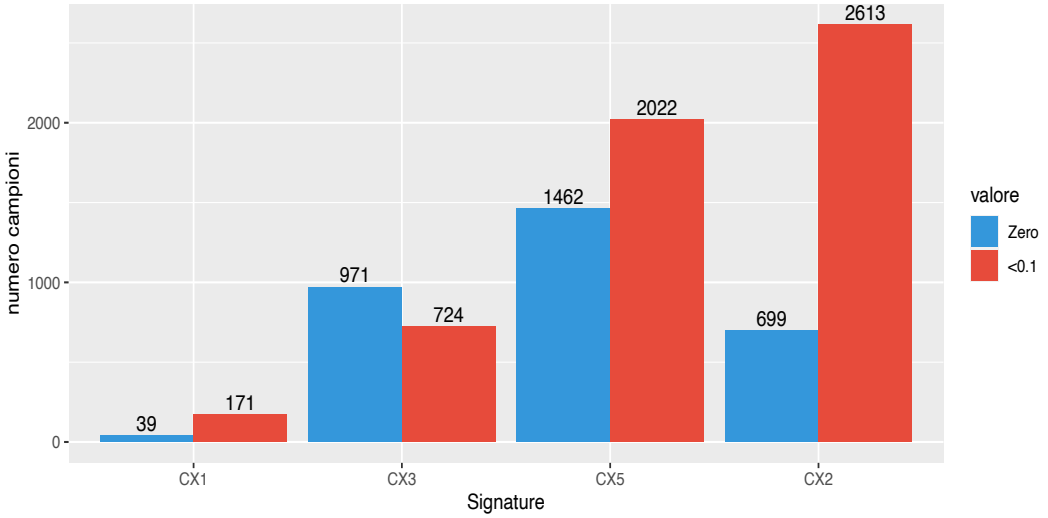


Figura 6.1: valori signature uguale a zero e inferiore a 0.1 ma diversi da 0

Questa discrepanza di performance è visivamente rappresentata nelle immagini 4.4, D.2, D.4 e D.6, dove si nota chiaramente come la linea rossa, rappresentante la perfezione nelle predizioni, si sposti sempre più verso l'asse x. Per quanto riguarda la CX2, emerge chiaramente che, nonostante sia il modello migliore, le sue performance risultano insufficienti per poter considerare le sue informazioni come valide.

Signature	Best model	dataType	Rsquared
CX1	svmRadial	Test	0.643
		Training	0.938
CX3	svmRadial	Test	0.541
		Training	0.885
CX5	ridge	Test	0.432
		Training	0.728
CX2	ridge	Test	0.228
		Training	0.572

Tabella 6.1: Performance best model

Un'altra osservazione significativa riguarda le variabili importanti identificate da ciascun modello. In tutti e quattro i modelli, emergono come predominanti le variabili legate all'espressione genica, mentre la presenza di marcatori di metilazione è scarsa, se non del tutto assente, nelle signature CX5 e CX2. Analizzando le prime 100 variabili importanti, si osserva che le signature con lo stesso miglior modello condividono alcune variabili (ad esempio, CX1 e CX3 ne condividono 73), mentre non si riscontrano variabili comuni tra modelli differenti (CX5 non ne condivide nessuna con CX1 e CX3). Questo risultato è plausibile, poiché modelli diversi operano in modi distinti e, di conseguenza, individuano differenti variabili importanti. A conferma di ciò, se consideriamo le prime 100 variabili importanti del modello SVM radiale per la signature CX5, notiamo che condivide 67 variabili con CX1 e 45 con CX3. Inoltre, analizzando la correlazione tra le signature (vedi figura 6.2), emerge che la CX1 mostra una marcata anticorrelazione con la CX3 e la CX5. Questo risultato conferma la plausibilità di trovare le stesse variabili importanti tra queste signature. È rilevante evidenziare che modelli diversi per la stessa signature invece condividono la maggior parte delle variabili importanti.

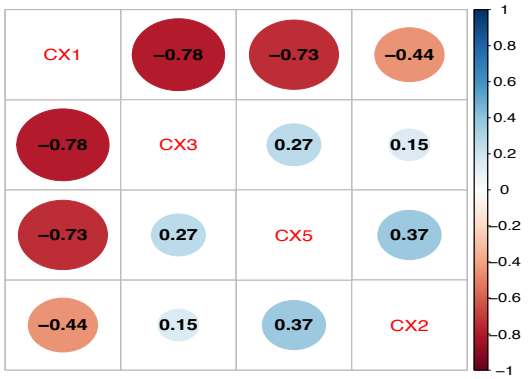


Figura 6.2: Correlazione Signature

Per confermare la relazione delle variabili importanti con le cause associate alle signature, è stato condotta un'analisi GO sulle prime 5 variabili importanti per ciascuna signature. I risultati di quest'analisi sono riportati nella tabella 6.2. Ad esempio, considerando la signature CX1, la cui causa associata è la missegregazione cromosomica tramite mitosi difettosa e/o disfunzione dei telomeri, le variabili più rilevanti includono Fam72B e Trip13. Fam72B, appartenente a una famiglia coinvolta nella modulazione dei geni del ciclo cellulare mitotico, è cruciale per la formazione di fusi centrosomiali e mitotici, e è stato identificato come potenziale biomarcatore per la prognosi del cancro [4]. Trip13, che svolge un ruolo chiave nella regolazione della ricombinazione cromosomica, può spiegare l'instabilità cromosomica (CIN) [8]. Fam83B, una famiglia di oncogeni, e CCNE1, un gene che codifica una proteina regolatrice delle kinasi CDK, sono ulteriori variabili importanti, evidenziando il ruolo critico di queste famiglie geniche nello sviluppo tumorale. In particolare, la sovraespressione di CCNE1 è stata osservata in numerosi tumori, contribuendo all'instabilità cromosomica. Infine, NUF2 è stato individuato come possibile bersaglio terapeutico, è in grado di inibire la crescita tumorale. Tutti e cinque i geni identificati svolgono ruoli correlati, evidenziando che i modelli individuano variabili con un nesso logico.

Per quanto riguarda CX3, la sua causa associata è l'IHR con stress di replicazione e rilevamento dei danni compromesso. L'instabilità genomica derivante da questa condizione contribuisce allo sviluppo dei tumori. CX3 condivide

quattro geni con CX1 nei top 5, ad eccezione di RFC4, il quale gioca un ruolo significativo nei percorsi dei checkpoint di danno al DNA [7]. Anche in questo contesto, è evidente che le variabili importanti sono in accordo con il processo di sviluppo tumorale e le cause specifiche della signature.
Per CX5 e CX2, non è stata condotta un'analisi GO a causa delle scarse performance ottenute con i modelli.

Importanza	CX1	CX3
1	FAM72B	FAM83D
2	TRIP13	TRIP13
3	FAM83D	FAM72B
4	CCNE1	CCNE1
5	NUF2	RFC4

Tabella 6.2: top 5 variabili importanti best model

6.1 Influenza tipo tumorale

È rilevante notare che il tipo tumorale influisce significativamente sull'analisi per ogni signature. Nei modelli SGD in cui il tipo tumorale è considerato come variabile esplicativa, questa emerge costantemente come la variabile più rilevante. Questo enfatizza l'importanza clinica del tipo tumorale, suggerendo che le caratteristiche molecolari specifiche di ciascun tipo possono esercitare un impatto significativo sulle signature. Questa osservazione è coerente con il lavoro di riferimento (visibile nella figura 2.3), che assegna una prevalenza pan-cancer a ciascuna signature, ma mai al 100%. Ciò sottolinea la grande eterogeneità tra i diversi tipi tumorali, nonostante l'obiettivo dell'analisi sia pan-cancer. I diversi tipi di tumori spesso presentano profili di espressione genica specifici e caratteristici che riflettono la loro origine cellulare, il loro microambiente e le vie molecolari coinvolte nella loro crescita e progressione. Questa eterogeneità molecolare rende difficile l'analisi pan-cancer, che cerca di identificare pattern comuni tra diversi tipi di tumori.

In particolare, nella nostra analisi potrebbero essere presenti alcuni tipi tumorali presenti in tutte e 4 le signature che influenzano maggiormente, in maniera negativa, le performance del modello. Nella tabella 6.3 sono elencati i tipi tumorali che hanno una maggiore influenza negativa sulle performance dei modelli. Possiamo notare che il tipo tumorale SARC (sarcoma) è presente in tutte e quattro le signature, mentre altri tre tipi tumorali appaiono in 3 su 4 signature: OV (cistoadenocarcinoma sieroso ovarico), LIHC (carcinoma epatocellulare del fegato) e BLCA (carcinoma uroteliale della vescica). Ciò sottolinea come i modelli effettivamente incontrino difficoltà nella predizione di specifici tipi tumorali, indipendentemente dal tipo di signature. Anche nel nostro caso, come nel lavoro di riferimento, siamo comunque vincolati al tipo tumorale nonostante l'obiettivo di condurre un'analisi pan-cancer.

Signature	Cancer Type				
CX1	LIHC	PRAD	BLCA	CESC	SARC
CX3	LIHC	OV	BLCA	CESC	SARC
CX5	BLCA	OV	SARC	/	/
CX2	PRAD	OV	LIHC	SARC	STAD

Tabella 6.3: Peggiori tipi tumorali

Capitolo 7

Conclusioni e sviluppi futuri

I modelli addestrati sui dati mostrano difficoltà crescenti all'aumentare della distribuzione vicino allo zero dei valori della signature da predire. In particolare, solo le prime due signature raggiungono un Rsquared superiore al 50% nel test set, mentre CX5 e soprattutto CX2 presentano valori così bassi da renderli poco affidabili. Ci sono due possibili spiegazioni che meritano considerazione: in primo luogo, potrebbe essere che i modelli faticano a elaborare efficacemente i dati di espressione e metilazione per prevedere le signature di instabilità genomica derivanti dall'analisi di copy number. Tuttavia, è importante notare che nel lavoro di riferimento, le cause associate alle signature influenzano sia il profilo di espressione che quello di metilazione dei campioni tumorali. Risulta quindi sorprendente che non vi sia un modo per predire direttamente le signature stesse da questi profili combinati di metilazione ed espressione. L'alternativa potrebbe essere che le signature identificate potrebbero non essere tutte altrettanto rilevanti come sembrano. Rifacendoci alla distribuzione dei valori delle signature rappresentata nella figura 2.4, è evidente che, ad eccezione della CX1 e della CX3, le restanti signature mostrano valori vicini allo zero per la maggior parte dei campioni. Ciò suggerisce che l'attività delle signature è, per la maggior parte dei casi, prossima allo zero. Per ottenere una risposta definitiva e comprendere appieno il problema, sarebbe necessario disporre di un dataset esterno al TCGA per testare se il problema risiede nei modelli che non performano bene e/o soffrono di overfitting, oppure se alcune delle signature

identificate sono un risultato puramente matematico senza alcuna attinenza alla biologia.

In aggiunta, è emerso chiaramente come il tipo tumorale abbia un notevole impatto sull'analisi, evidenziato dal fatto che quando utilizzato come variabile esplicativa, risulta essere la variabile più influente per tutte le signature. La presenza di una discrepanza uniforme tra le coorti di campioni per tutti i tipi tumorali potrebbe altresì influenzare la capacità predittiva dei modelli. Il passo successivo potrebbe coinvolgere analisi specifiche su determinati gruppi tumorali che condividono caratteristiche comuni. Tuttavia, è fondamentale riconoscere che questo approccio comporterebbe l'allontanamento dall'analisi pan-cancer, che era l'obiettivo principale del nostro studio. In alternativa, per mantenere l'approccio pan-cancer si potrebbe sfruttare un approccio di tipo deep learning per sviluppare modelli in grado di gestire efficacemente gli zeri, ad esempio attraverso l'implementazione di reti neurali. Questa strategia potrebbe consentire di mantenere l'ampiezza dell'analisi pan-cancer, affrontando nel contempo le sfide legate alla distribuzione vicina allo zero dei valori delle signature.

Appendice A

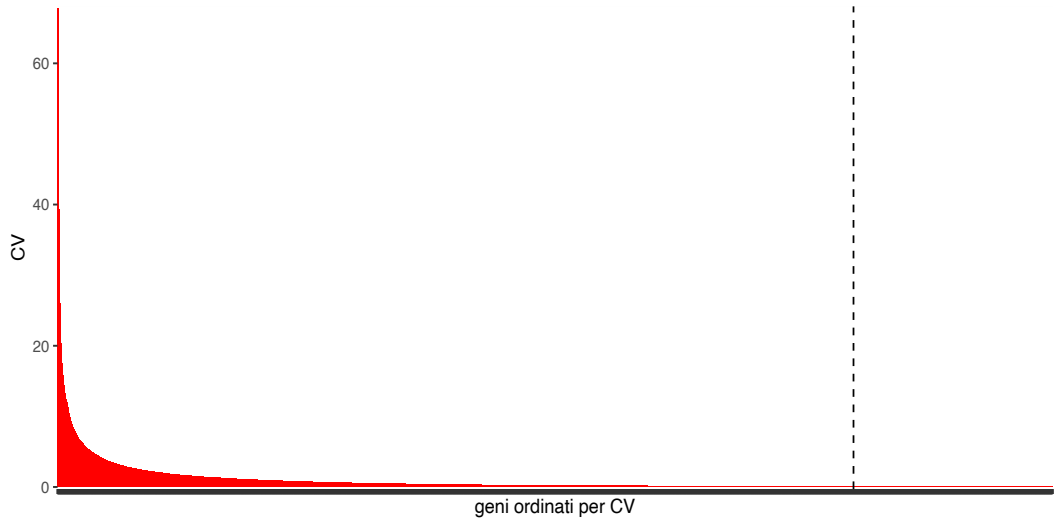


Figura A.1: CV espressione

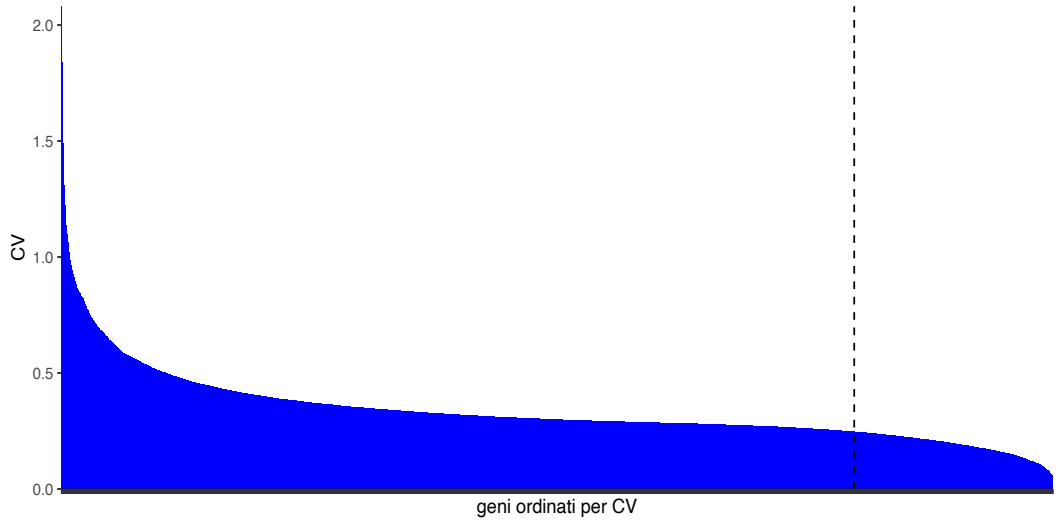


Figura A.2: CV metilazione

Appendice B

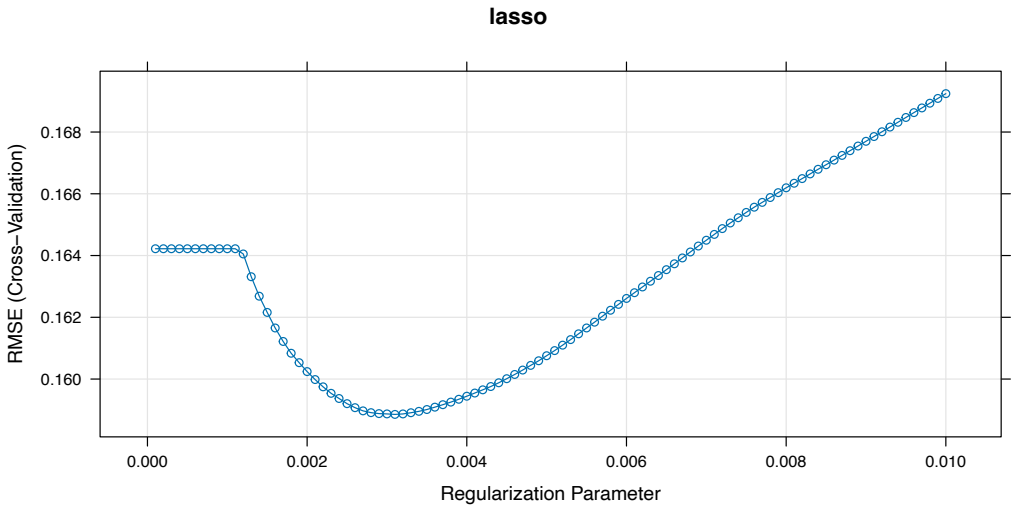


Figura B.1: lasso train model

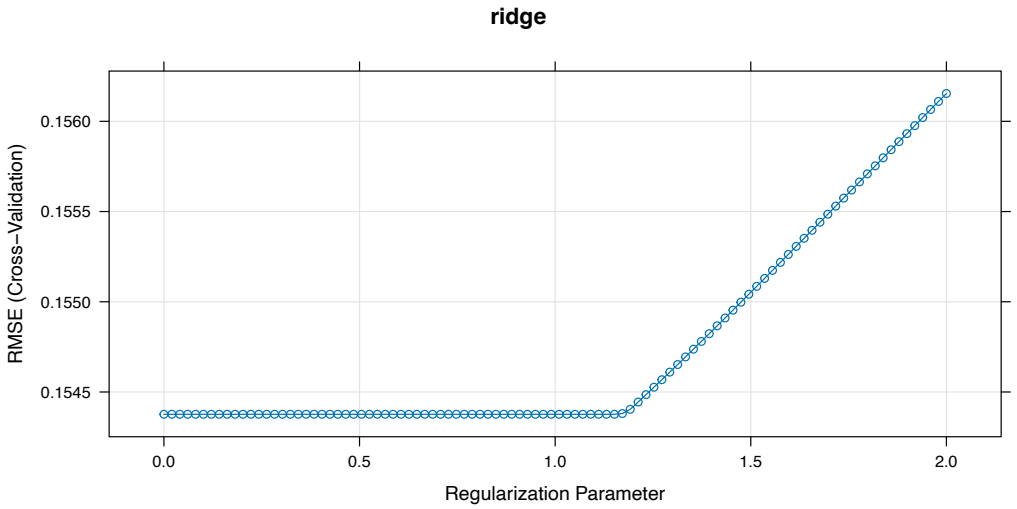


Figura B.2: ridge train model

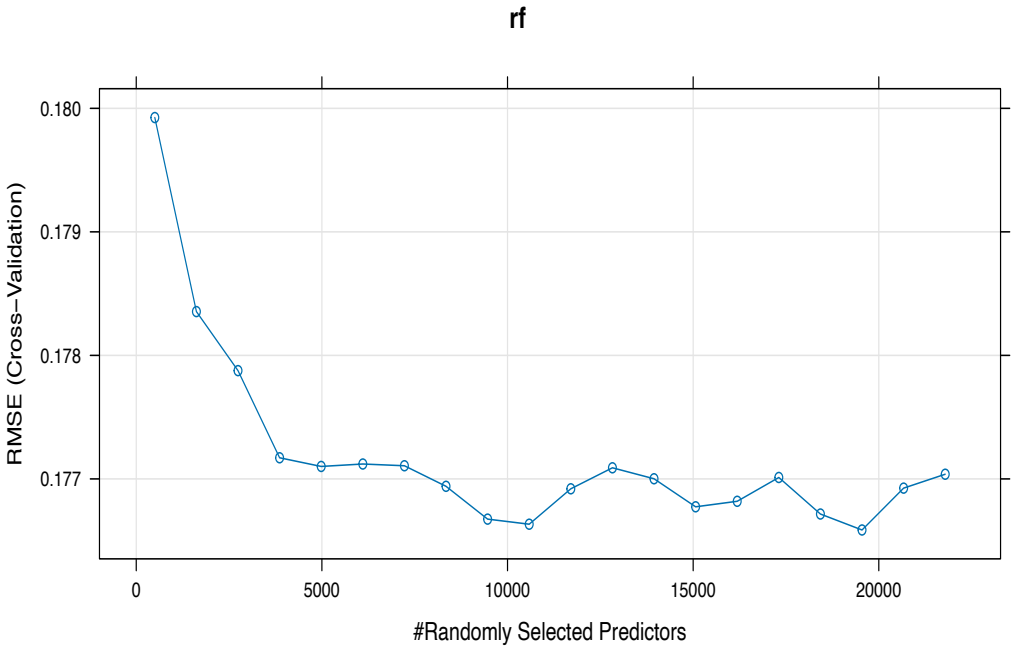


Figura B.3: rf train model

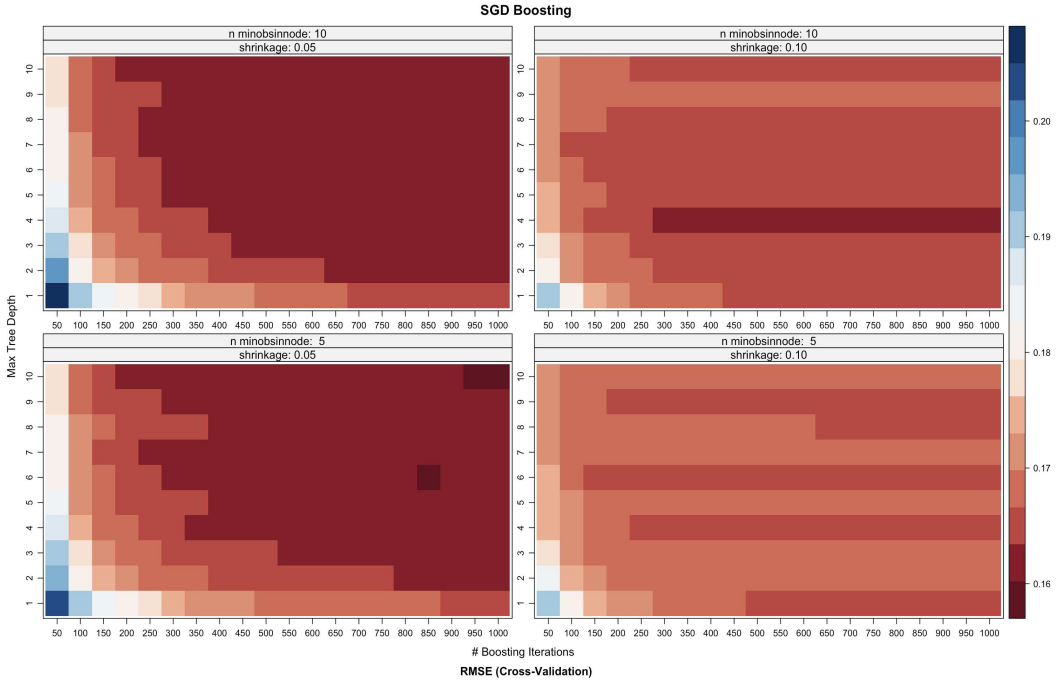


Figura B.4: SGD train model

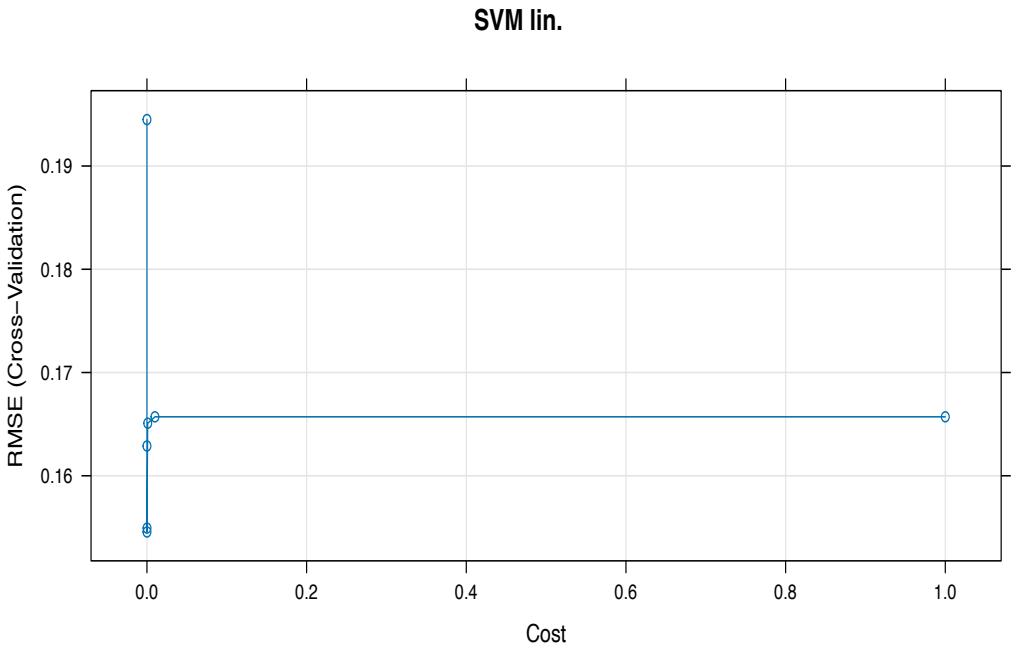


Figura B.5: SVM lin. train model

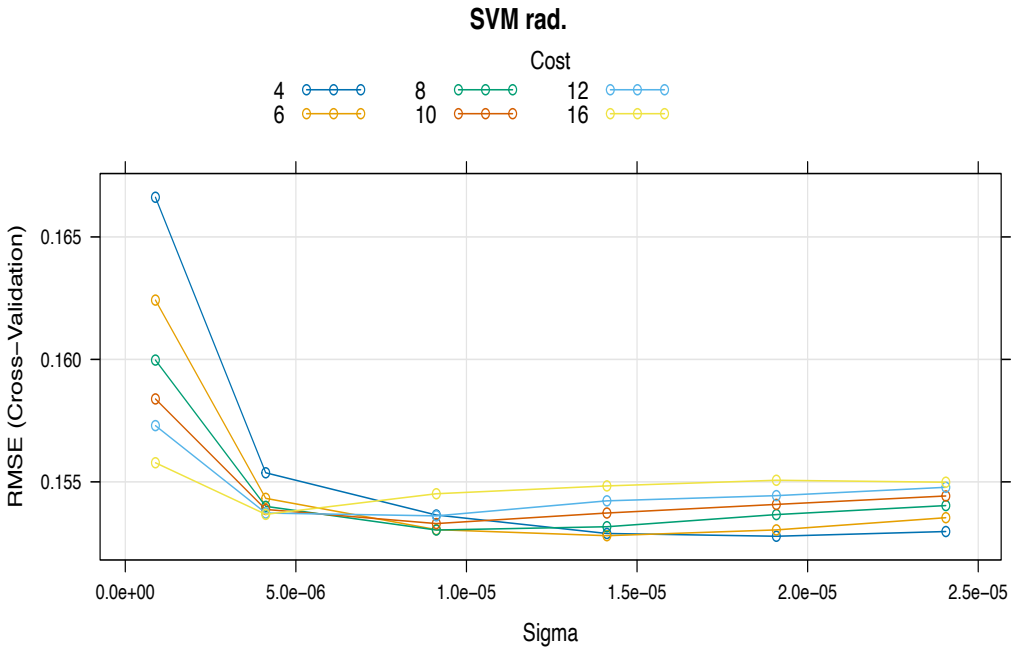


Figura B.6: SVM rad. train model

Appendice C

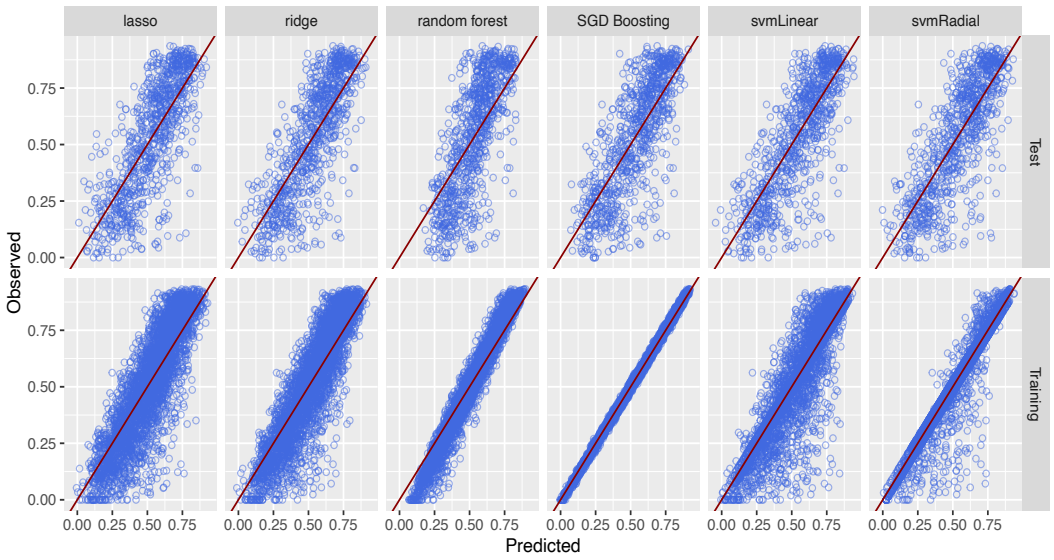


Figura C.1: Performance modelli

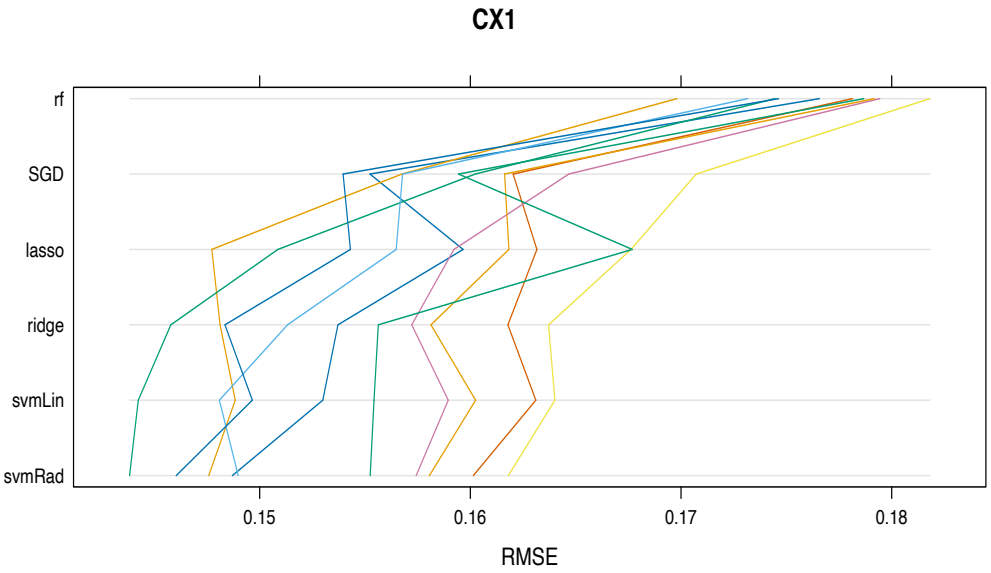


Figura C.2: Parallelplot ricampionamento

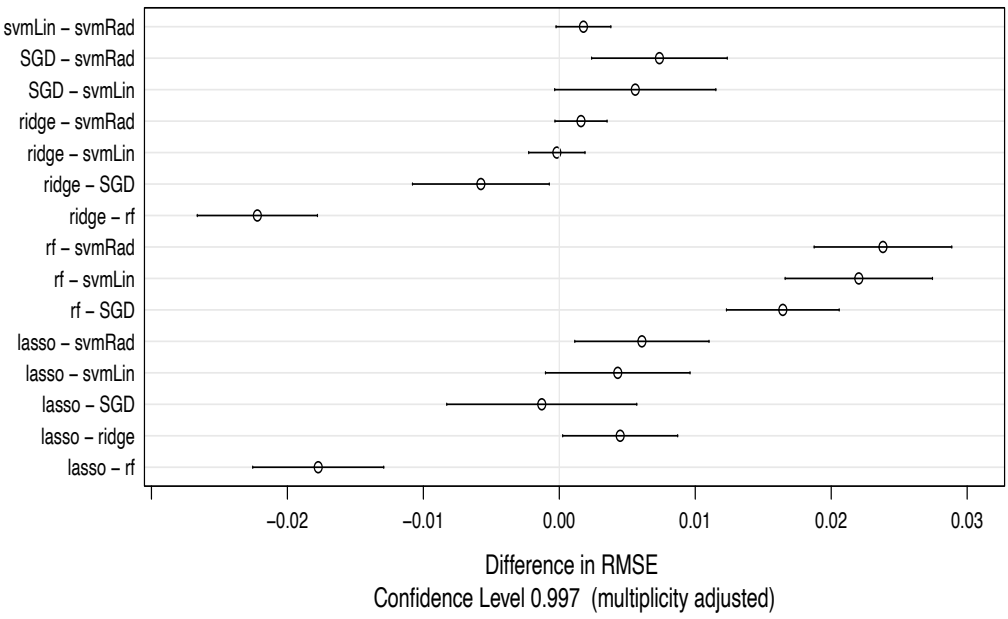


Figura C.3: dotplot differenze tra modelli

Appendice D

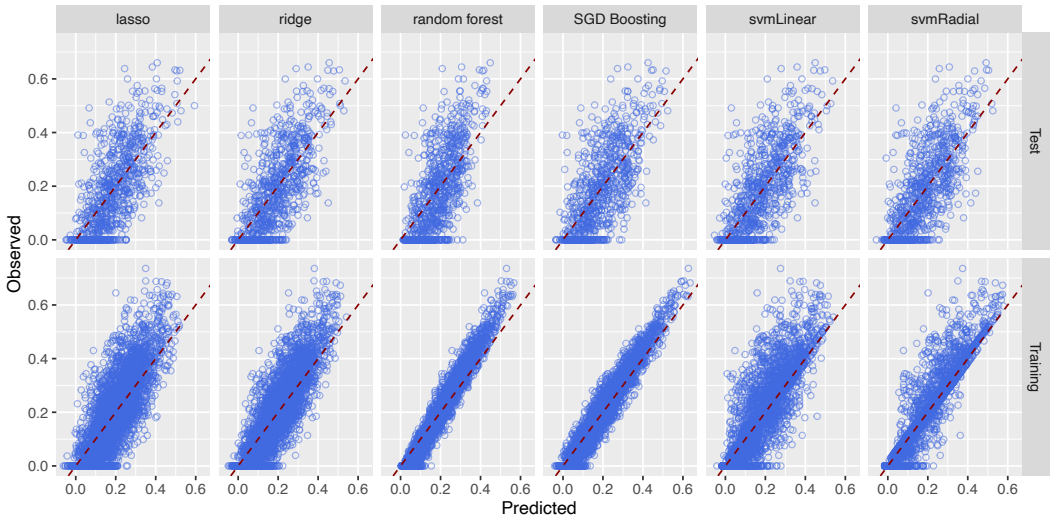


Figura D.1: Performance modelli CX3

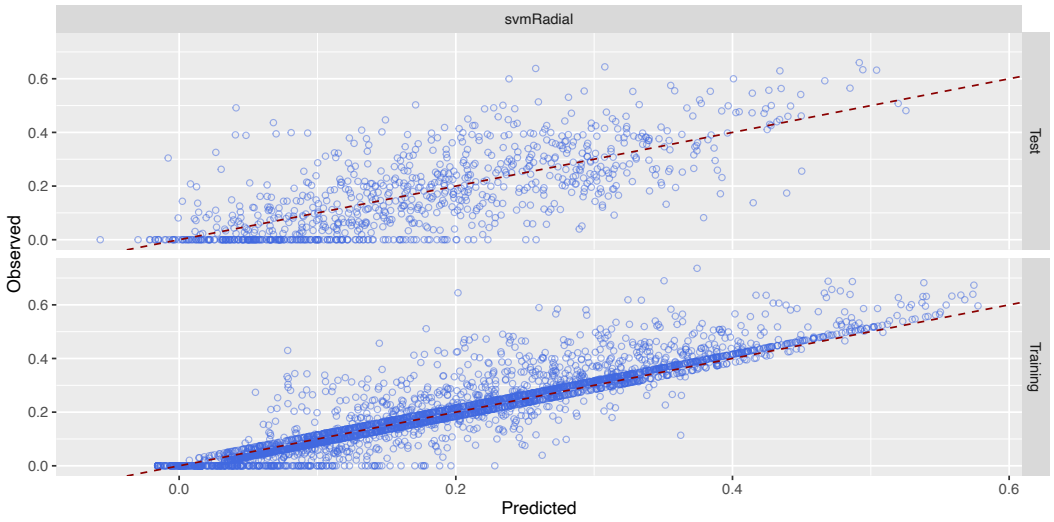


Figura D.2: Best model CX3

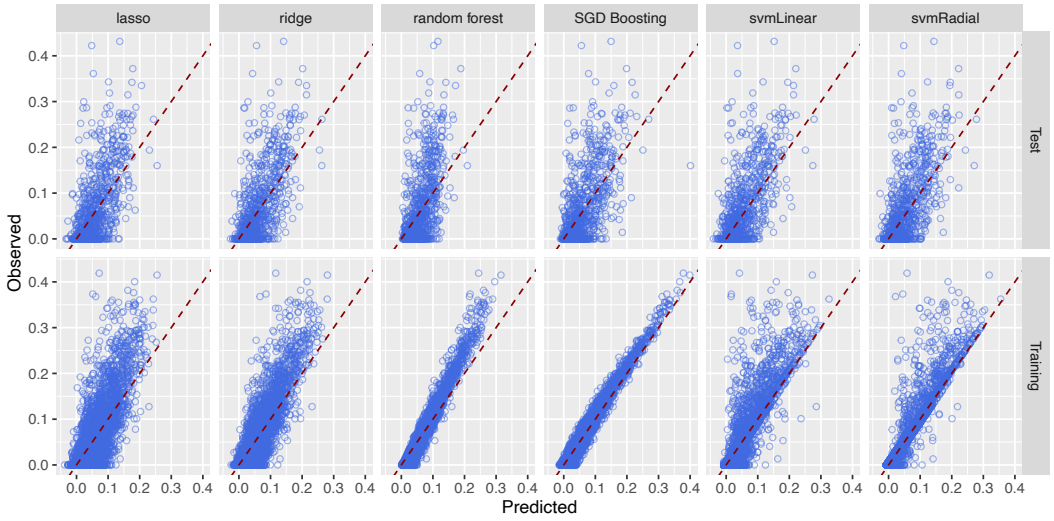


Figura D.3: Performance modelli CX5

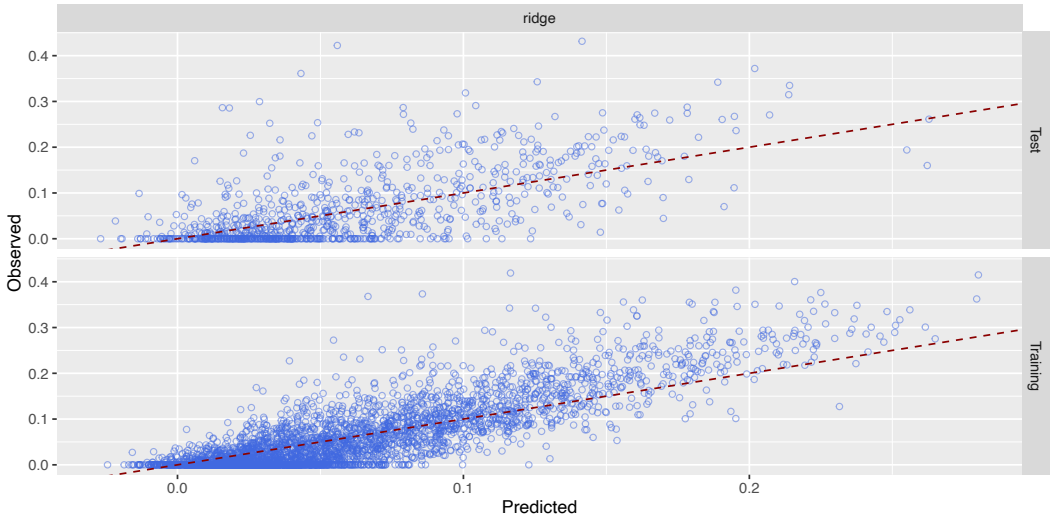


Figura D.4: Best model CX5

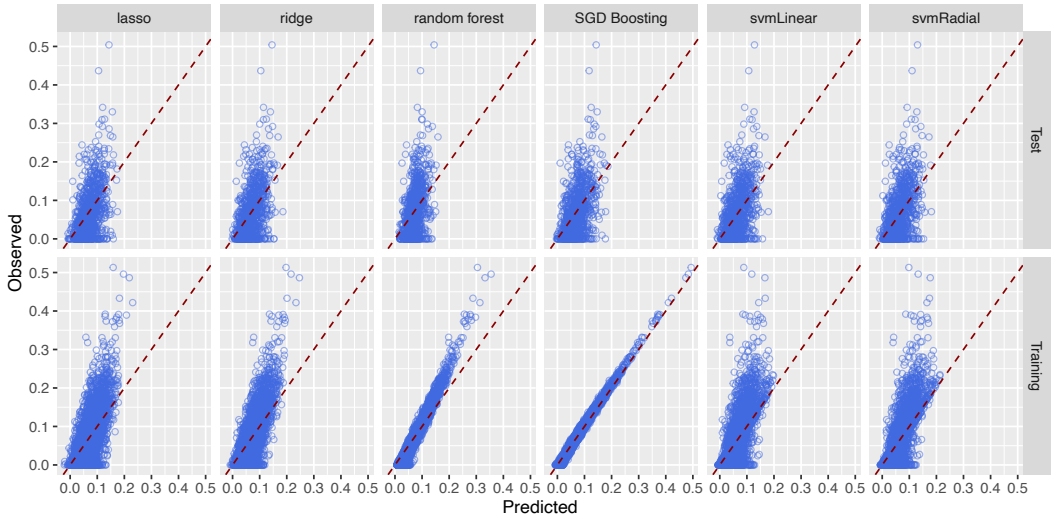


Figura D.5: Performance modelli CX2

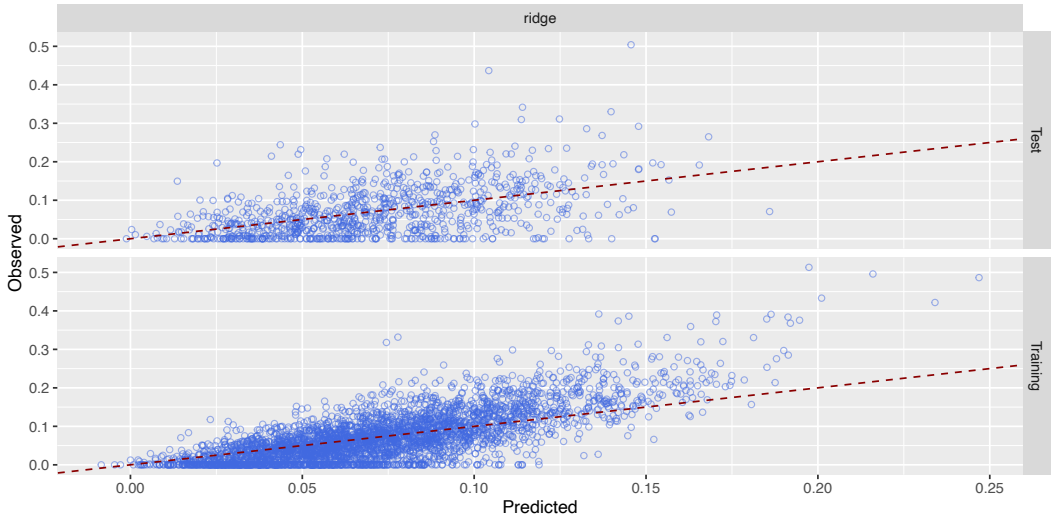


Figura D.6: Best model CX2

Bibliografia

[1] Adelchi Azzalini e Bruno Scarpa. *Analisi dei dati e data mining*. Springer Science & Business Media, 2009.

[2] Kajaree Das e Rabi Narayan Behera. «A survey on machine learning: concept, algorithms and applications». In: *International Journal of Innovative Research in Computer and Communication Engineering* 5.2 (2017), pp. 1301–1309.

[3] Ruben M Drews et al. «A pan-cancer compendium of chromosomal instability». In: *Nature* 606.7916 (2022), pp. 976–983.

[4] Yuqing Feng et al. «The FAM72 gene family promotes cancer development by disabling the base excision repair system». In: *Cancer Research* 83.7_Supplement (2023), pp. 1217–1217.

[5] Manuel Fernández-Delgado et al. «An extensive experimental survey of regression methods». In: *Neural Networks* 111 (2019), pp. 11–34.

[6] Max Kuhn et al. «Package 'caret'». In: *The R Journal* 223.7 (2020).

[7] Yanling Li et al. «Multifaceted regulation and functions of replication factor C family in human cancers». In: *American Journal of Cancer Research* 8.8 (2018), p. 1343.

[8] S Lu et al. «Insights into a crucial role of TRIP13 in human cancer». In: *Computational and structural biotechnology journal* 17 (2019), pp. 854–861.

Ringraziamenti

Desidero esprimere la mia sincera gratitudine a tutte le persone che hanno contribuito al completamento di questa tesi di laurea.

Innanzitutto, vorrei ringraziare il mio supervisore, la Prof.ssa Chiara Romualdi, per la sua guida preziosa e il supporto costante durante tutto il processo di ricerca. Le sue competenze accademiche e la dedizione al mio lavoro sono state fondamentali per il successo di questo progetto.

Un ringraziamento speciale va a Ilaria Billato, che ha condiviso la sua esperienza e ha fornito preziosi consigli tecnici che hanno arricchito il mio lavoro.

Desidero esprimere la mia gratitudine ai miei genitori e alla mia famiglia per il loro incrollabile sostegno e incoraggiamento durante tutti questi anni.

Un ringraziamento va anche ai miei amici e colleghi che hanno condiviso questo percorso con me. Le vostre discussioni stimolanti e il sostegno reciproco hanno reso questa esperienza ancora più significativa.

Grazie a tutti coloro che hanno reso possibile la realizzazione di questa tesi di laurea. Il vostro contributo è stato fondamentale e sarà sempre ricordato con gratitudine.

Il laureando, Marco Rota Negroni