Automated Essay Scoring

Federico Rausa – progetto di Machine Learning

0.Abstract

Il task AES è il problema di automazione della valutazione dei temi scritti dagli studenti. Per risolverlo bisogna convertire il testo del tema in predittori numerici, mediante metodi di text mining, e utilizzarli per prevedere la variabile voto, o punteggio, che è una variabile discreta ordinata, per cui rappresenta un task di ranking. Dopo aver operato il preprocessig del dataset attraverso la DTM, la SVD e il DBSCAN, si stima la variabile risposta attraverso algoritmi di apprendimento supervisionato (tra cui KNN, SVM, regressione Lasso, alberi di decisione, algoritmi di boosting e regressione logistica cumulata) e se ne confrontano i risultati, in particolare rispetto agli approcci di ranking.

Indice

1. Introduzione: Automated Essay Scoring

Il task scelto proviene da una competizione di Kaggle proposta dall'Università di Vanderbilt congiuntamente al Learning Agency Lab[1], un'organizzazione no-profit statunitense nata nel 2019 avente come obiettivo quello di offrire efficaci e gratuite risorse di apprendimento per gli studenti più svantaggiati dal punto di vista dell'offerta formativa scolastica, per ragioni storiche o economiche.
Il dataset e la descrizione della competizione sono disponibili al seguente link[2].
Nel 2012, sempre su Kaggle, la Hewlett Foundation aveva proposto una competizione molto simile[3].

Scopo della competizione è quello di costruire un algoritmo in grado di assegnare una valutazione discreta da 1 a 6 a un tema scritto in inglese da uno studente della fascia d'età compresa fra i 10 e i 14 anni. Si tratta di un problema di AWE (Automated Writing Evaluation), o AES (Automated Essay Scoring), tipicamente affrontato quado si vogliono costruire strumenti di valutazione automatica, di supporto per studenti e insegnanti, grazie ai quali possono ricevere feedback sul proprio lavoro, e che si rivelano utili specialmente nelle aree geografiche meno servite a livello di scuole e istruzione.
I criteri di valutazione del voto sulla scala da 1 a 6 riflettono quelli che sono tenuti a seguire gli insegnanti in alcune scuole americane, e un'accurata descrizione degli stessi è disponibile al link[4].

Il dataset AES si compone di 17307 osservazioni e di due variabili:
-score: il voto, valore intero da 1 a 6, assegnato allo studente che ha scritto il tema
-full_text: il testo completo del tema, in formato stringa
L'analisi della variabile full_text rappresenta quindi un problema di text mining.

La previsione della variabile score rappresenta invece un problema di supervised learning, in particolare di ranking.

2. Metodi di text mining per il preprocessing

2.1. EDA sulla DTM

Esistono molti modi per convertire dei dati di testo in valori numerici, e la maggior parte si basa sulla costruzione della DTM[5] (matrice documento-termine / Document-Term Matrix) sulle cui righe vi sono gli indici dei documenti, come istanze del dataset, e sulle colonne dei tokens di testo, generalmente costituite dalle parole di un dizionario, me che possono anche essere simboli, numeri o elementi di punteggiatura.
Ogni cella contiene il numero di volte che un dato token compare in un testo, per cui la DTM è una matrice di interi positivi. È facile che il numero di colonne del dataset superi il numero di righe, per cui la DTM viene spesso convertita a sua volta in un gruppo ridotto di variabili, mantenendo lo stesso numero di osservazioni. Tra i metodi di riduzione della dimensionalità della DTM si ricordano il

[1] https://the-learning-agency-lab.com/
[2] https://www.kaggle.com/competitions/learning-agency-lab-automated-essay-scoring-2
[3] https://www.kaggle.com/competitions/asap-aes/overview
[4] https://storage.googleapis.com/kaggle-forum-message-attachments/2733927/20538/Rubric_%20Holistic%20Essay%20Scoring.pdf
[5] Radovanović, M., & Ivanović, M. (2008). Text mining: Approaches and applications. *Novi Sad J. Math*, *38*(3), 227-234.

metodo della TF-IDF[6], il metodo word2vec[7], la LSA con SVD[8], e infine la LDA[9] (ultime due utilizzate maggiormente per task di topic modeling, ovvero di assegnazione di tokens a macro-temi specifici).

Sulle colonne della DTM compaiono in genere due gruppi di parole: le stopwords (come le preposizioni, i pronomi e gli articoli) trasversali agli argomenti, e le parole che possono invece rappresentare indizi sul tema di discussione (come i nomi, i verbi e gli aggettivi). Di solito un primo filtro per ridurre il numero di tokens può essere quello di rimuovere le parole con una frequenza molto ridotta su diversi documenti, ovvero che non supera una determinata soglia arbitraria (il trimming). Un altro metodo può essere quello di fare la lemmatizzazione o lo stemming del testo, per unire in un'unica variabile parole aventi la stessa radice.

Nell'ambito del topic modeling spesso si rimuovono le stopwords[10], in quanto non aiutano a identificare l'argomento del discorso, ma in un task di analisi di ranking delle competenze linguistiche, come quello qui affrontato, possono rivelarsi utili. Nella DTM della variabile full_text del dataset AES i primi tokens, in ordine di frequenza osservata, sono naturalmente stopwords e punteggiatura:

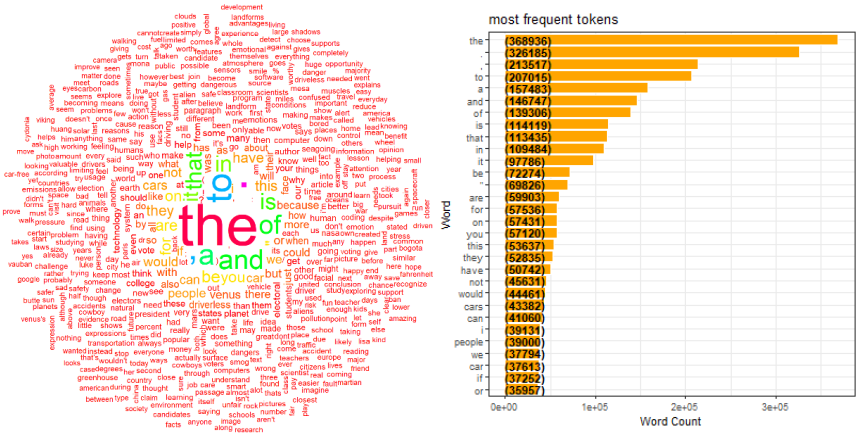

Figura 1 parole a frequenza maggiore nella dtm

Gli Ultimi, invece, hanno frequenza 1, e sono per la maggior parte parole storpiate ed errori grammaticali:

[6] Yun-tao, Z., Ling, G., & Yong-cheng, W. (2005). An improved TF-IDF approach for text classification. *Journal of Zhejiang University-Science A*, *6*(1), 49-55.

[7] Rei, M., & Briscoe, T. (2014, June). Looking for hyponyms in vector space. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning* (pp. 68-77).

[8] Antonellis, I., & Gallopoulos, E. (2006). Exploring term-document matrices from matrix models in text mining. *arXiv preprint cs/0602076*.

[9] Radovanović, M., & Ivanović, M. (2008). Text mining: Approaches and applications. *Novi Sad J. Math*, *38*(3), 227-234.

[10] Schofield, A., Magnusson, M., & Mimno, D. (2017, April). Pulling out the stops: Rethinking stopword removal for topic models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, short papers* (pp. 432-436).

```
> colnames(dtm)[(nc-100):nc]
  [1] "represeted"                "commerance"                "coulumbia"                 "officaly"
  [5] "expasive"                  "dete"                      "safte"                     "problams"
  [9] "intier"                    "soarching"                 "charaterisics"             "acutaly"
 [13] "radtion"                   "smoothing"                 "purley"                    "ctreated"
 [17] "revisted"                  "sacks"                     "ficition"                  "sicence"
 [21] "vrroooom"                  "vroooom"                   "tduring"                   "rosanthal's"
 [25] "globay"                    "lisened"                   "hisk"                      "ntural"
 [29] "mailin"                    "wb"                        "life-risking"              "exploning"
 [33] "striker"                   "plceses"                   "advoiding"                 "likwise"
 [37] "ransportation"             "femaninans"                "erition"                   "becasu"
 [41] "rocksa"                    "gernal"                    "somewhe"                   "sensorys"
 [45] "partially-driverless"      "entartain"                 "foreal"                    "recogizes"
 [49] "tremeandosly"              "i-feel-like-i-know-everyone" "engadged"                "transfromed"
 [53] "shwoing"                   "worldside"                 "stopps"                    "self-diven"
 [57] "defineatley"               "sovereignty"               "noneffective"              "redundent"
 [61] "actuallythe"               "coummicate"                "sussceful"                 "alsomany"
 [65] "immortalization"           "coomunication"             "even-license"              "field-tested"
 [69] "manueverable"              "reavive"                   "touck"                     "thpought"
 [73] "klnows"                    "ridculous"                 "contuines"                 "freesdom"
 [77] "alol"                      "in2012"                    "istitute"                  "memebrs"
 [81] "psychlogists"              "movoments"                 "communuication"            "paragarh"
 [85] "particapaed"               "basicallt"                 "celibrate"                 "exced"
 [89] "seagoping"                 "underpreciated"            "convincement"              "difficujlt"
 [93] "visbility"                 "dificulities"              "exploreation"              "siser"
 [97] "passcode"                  "peforming"                 "jin"                       "cloudds"
```

Figura 2 alcune parole a frequenza 1 nella DTM di full_text

Dopo aver rimosso i token a bassissima frequenza (in questo caso si decide di utilizzare una soglia pari a 20) il numero di colonne della DTM passa da 70167 a 6553. Le elaborazioni della DTM vengono svolte in R tramite il pacchetto quanteda[11].

2.3. Riduzione della dimensionalità con SVD

Un metodo molto ricorrente nel text mining è la decomposizione SVD[12], nella quale si decompone la DTM come il prodotto di 3 matrici, con componenti ordinate rispetto alla percentuale di varianza spiegata:

$$A = UDV^T$$

Dove A è la matrice che si vuole ridurre (il dataset o la DTM), D è una matrice diagonale contenente gli autovalori, indicanti l'importanza di ciascuna componente, U la matrice che lega le righe di A (le istanze) alle componenti e V la matrice che lega le colonne di A (le variabili) alle componenti.
Nella decomposizione della DTM, una matrice di interi positivi, spesso con distribuzioni asimmetriche, il metodo della SVD è più indicato rispetto a quello della PCA.
In questa analisi, si avrebbe a disposizione una DTM di 6553 colonne e 17307 righe. Tale dimensione, per una comune CPU, risulta temporalmente insostenibile nel calcolo della SVD, per cui si decide di tenere in considerazione solo i primi 2000 tokens in ordine di frequenza, e di buttare via gli altri.

Il numero di componenti principali K da estrarre è una scelta che dipende dalla quota di variabilità complessiva spiegata dalle stesse.

$$gained\ variance\% = \frac{\sum_{i=1}^{K} D_i}{\sum_{i=1}^{S} D_i}$$

Dove D_i è l'i-esimo elemento della diagonale di D e S è il numero di elementi sulla diagonale di D. Per scegliere K, ci si può affidare a tre criteri principali[13]. Il primo è la soglia minima percentuale di varianza che si vuole spiegare sulla varianza complessiva. Il secondo è il massimo numero di componenti che si è disposti ad accettare. Il terzo è la prassi non rigorosa del metodo del gomito.

[11] https://quanteda.io/
[12] Yang, J., & Watada, J. (2011, June). Decomposition of term-document matrix representation for clustering analysis. In *2011 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2011)* (pp. 976-983). IEEE.
[13] Falini, A. (2022). A review on the selection criteria for the truncated SVD in Data Science applications. *Journal of Computational Mathematics and Data Science*, *5*, 100064.

Con il metodo del gomito, data una misura di costo decrescente, e date diverse opzioni intere positive da 1 a S, dove S è la lunghezza del vettore ordinato contenente tale misura, l'opzione migliore viene scelta fra quelle più vicine all'angolo della curva formata da K (sulle ascisse) e dalla metrica di costo (sulle ordinate). Tale metodo viene utilizzato in tutti i casi in cui si vogliono minimizzare contemporaneamente K e la metrica di costo, attraverso una soluzione di equilibrio.

Un metodo analitico per identificare il punto sull'angolo della curva può consistere nel calcolo della lunghezza delle rette passanti tra i punti e la retta, ad essere ortogonale, passante per il primo e l'ultimo punto della sequenza. Il punto che massimizza la distanza dalla corda dell'arco, ovvero corrispondente alla retta ortogonale alla corda di lunghezza massima, è un buon candidato (non per forza il migliore) rispetto alla scelta dell'opzione K.

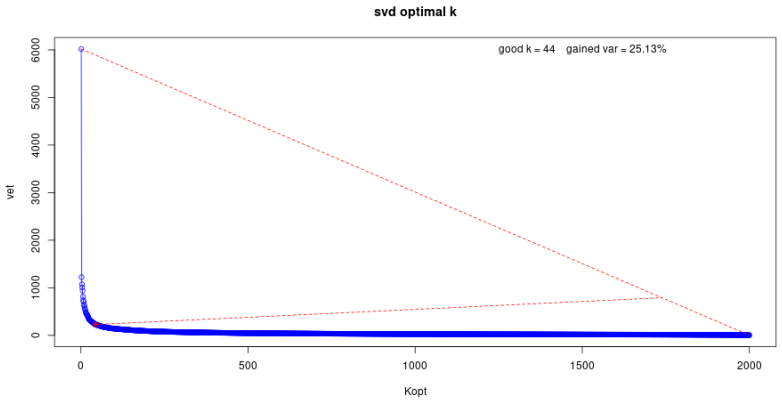

Figura 3: scelta del numero di componenti principali attraverso il metodo del gomito, con max-K pari al numero di colonne del dataset. Le due rette tratteggiate sono ortogonali, ma le scale degli assi differiscono. Si vede che il punto più vicino all'angolo corrisponde all'utilizzo delle prime 44 componenti principali.

L'interpretazione del risultato del metodo, applicato considerando tutte le possibili componenti, è che il valore di K dovrebbe essere al massimo 44. In realtà, tale valore è comunque arbitrario, ma la scelta conveniente tende a non superare tale soglia. Si vede che con sole 25 variabili si spiega il 20% della variabilità complessiva, per cui questo sarà il K adottato.

K	1	2	3	4	5	6	7	8	9	10	11
CumVar	5.89	7.09	8.14	9.13	10.05	10.85	11.57	12.26	12.9	13.51	14.08
K	12	13	14	15	16	17	18	19	20	21	22
CumVar	14.62	15.15	15.63	16.11	16.55	16.99	17.42	17.83	18.23	18.61	18.99
K	23	24	25	26	27	28	29	30	31	32	33
CumVar	19.33	19.67	20	20.32	20.63	20.94	21.24	21.54	21.83	22.11	22.39
K	34	35	36	37	38	39	40	41	42	43	44
CumVar	22.66	22.93	23.19	23.45	23.7	23.94	24.18	24.42	24.65	24.88	25.1

Figura 4 Percentuale della varianza complessiva spiegata per ciascun numero K di componenti SVD.

Il metodo del gomito, oltre che nella scelta del numero di componenti della SVD, può essere adottato in una varietà di contesti, come nell'ambito della PCA e della scelta del numero di cluster K nel caso del K-means e degli altri algoritmi di clustering rappresentativo. La decomposizione della DTM tramite la SVD risulta costosa dal punto di vista computazionale, ma opera una compressione ottimale in termini di variabilità spiegata.

2.4 dataset adottato e train-test splitting

Il dataset utilizzato per prevedere la variabile score comprende le prime 25 componenti SVD della DTM. Con Il DBSCAN[14] verranno rimosse tutte le osservazioni anomale.
Per quanto riguarda l'analisi di apprendimento supervisionato, delle 17307 osservazioni presenti nel dataset, circa 760 saranno rimosse come noise points tramite il DBSCAN. Circa 4000 saranno in via preliminare adottate come validation-set per svolgere l'ottimizzazione di alcuni modelli, ovvero per fissarne gli iper-parametri ottimi. Le restanti 12000 verranno poi utilizzate per svolgere dei confronti tra i modelli supervisionati (con iper-parametri già ottimizzati), in una fase delle analisi nella quale, via simulazione bootstrap, 5000 saranno ripetutamente assegnate a una fase di training e 7000 a una fase di testing.

La variabile score presenta la seguente distribuzione percentuale nei 4 dataset (la divisione tra validation-set e test-set è stata eseguita rispettando la distribuzione delle classi):

	1	2	3	4	5	6	N
scoreMainDist	7.23	27.29	36.29	22.68	5.6	0.9	17307
scoreValidationDist	6.97	28.04	37.01	22.61	4.82	0.55	4002
scoreTrainTestDist	7.15	27.86	37.02	22.61	4.84	0.53	12551
scoreNoiseDist	10.08	13.79	20.29	24.27	22.55	9.02	754

Figura 5 distribuzione percentuale delle categorie (1-6) della variabile score nell'intero dataset, nel validation-set, nel train/test-set e nel dataset dei nosy points, con N numero di istanze

4.1 Rimozione del rumore e degli outlier con DBSCAN

Applicando il DBSCAN sul dataset delle 25 componenti SVD (standardizzate con lo Z-score), è possibile identificare i noise e gli outlier points, che possono produrre distorsione nei modelli supervisionati. Aggiustiamo gli iperparametri eps (raggio intorno ai punti vicini) e minPts (numero minimo di punti vicini di un core point). Per eps= 5 e minPts = 5 si identificano 760 outlier e noisy points, su 17307 osservazioni. Il costo in termini di informazione persa sembra quindi sostenibile.

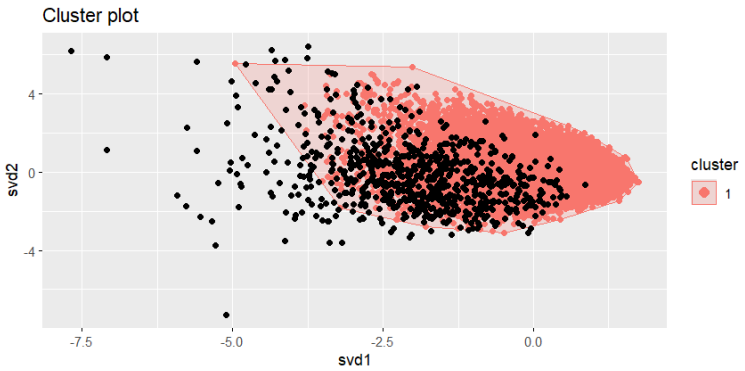

Figura 6 filtraggio dei noisy points con DBSCAN, osservato sulle prime due componenti principali della SVD

[14] Akbari, Z., & Unland, R. (2016). Automated determination of the input parameter of DBSCAN based on outlier detection. In *Artificial Intelligence Applications and Innovations: 12th IFIP WG 12.5 International Conference and Workshops, AIAI 2016, Thessaloniki, Greece, September 16-18, 2016, Proceedings 12* (pp. 280-291). Springer International Publishing.

3. Il problema del ranking e le metriche di validazione

La variabile obiettivo presenta la seguente distribuzione:

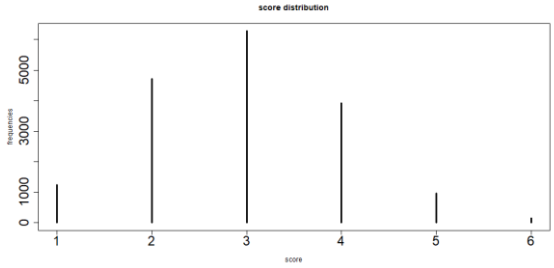

Figura 7 distribuzione della variabile risposta ordinale score

La natura del problema è di ranking, ma non si discosta molto né da un problema di regressione né da un task di classificazione. La distribuzione dei voti a campana ricorda una variabile latente continua normalmente distribuita, e sarebbe possibile stimarla con modelli di regressione per poi arrotondare le previsioni all'intero più vicino. Il fatto che si abbiano delle categorie discrete rende naturalmente possibile l'adozione di algoritmi di classificazione. È da notare come le valutazioni sulle code (1, 6 e 5) siano molto più rare, e quindi molto più difficili da prevedere.
Le metriche di validazione scelte sono dunque le seguenti:

-Accuracy: rappresenta la percentuale assoluta di previsioni corrette

-MAE: rappresenta la media dell'errore, e per valori minori di 1, in questo contesto, si considera accettabile

-Kappa Pesata di Cohen con pesi quadratici[15] (QWK o quadratitic weighted kappa): è una metrica di validazione adatta al task di ranking, in quanto pesa maggiormente gli errori che maggiormente si discostano dalla diagonale nella matrice di confusione. È anche la metrica adottata nella classifica della competizione AES di Kaggle. La formula della Kappa pesata di Cohen è la seguente:

$$K_W = 1 - \frac{\sum W_{ij} O_{ij}}{\sum W_{ij} E_{ij}} \quad \text{con pesi quadratici } W_{ij} = \frac{(i-j)^2}{(K-1)^2}$$

dove i e j sono rispettivamente l'indice della riga e della colonna della matrice di confusione, K è il numero di classi, O_ij sono le relative frequenze osservate e E_ij sono le frequenze attese (calcolate come prodotto delle frequenze relative marginali dei reali e dei previsti).

-la F1 per la classificazione delle categorie 1 e 6: il F1-score è la media geometrica di Precision e Recall, ed è quindi calcolabile per problemi di classificazione binaria, oppure individualmente per ciascuna categoria di un problema di classificazione multinomiale. Precision e Recall presentano al numeratore entrambi i TP (veri positivi). Pertanto, se F1=0, non si è fatta nessuna previsione positiva corretta dei valori della categoria osservata. Uno strumento di AES è utile solo se è in grado di assegnare tutti i voti della scala da 1 a 6, e andrebbero scartati quei modelli che non sono in grado di assegnare i punteggi in una delle code. Non si ha bisogno delle F1 delle altre categorie, in quanto, se si possono fare previsioni sugli eventi estremi nelle code, è molto probabile che si possano fare anche per tutte le categorie intermedie.

$$F1 = \frac{2 * precision * recall}{precision + recall} \quad \text{con} \quad precision = \frac{TP}{TP+FP} \qquad recall = \frac{TP}{TP+FN}$$

[15] Vaughn, D., & Justice, D. (2015). On the direct maximization of quadratic weighted kappa. *arXiv preprint arXiv:1509.07107*.

5 Scelta e ottimizzazione dei modelli supervisionati

Al fine di prevedere la variabile score, si sperimentano alcuni tra i modelli più comuni e ritenuti idonei al task. La scelta finale riguarderà un modello tra alcuni di regressione (Lasso, KNN per la regressione, GBM per la regressione e SVM per la regressione), alcuni di classificazione (Alberi di decisione, GBM per la classificazione) e alcuni modelli di ranking (la regressione logistica ordinale e un metodo di ensemble learning applicato a un gruppo di modelli di classificazione binaria presentato nella sezione 5.2.1).

Dei modelli sperimentati, si anticipa che la regressione logistica ordinale sembra raggiungere la combinazione di performance preferibile, ma anche che molti modelli mostrano performance simili in termini di Accuracy, MAE e QWK.

5.1 modelli di regressione e classificazione adottati

In questa sezione si introducono i modelli supervisionati scelti e se ne ottimizzano, con grid-search o con metodi di ottimizzazione bayesiana, i relativi iper-parametri. Il confronto fra i modelli ottimizzati è riportato nella sezione 5.1.6.

5.1.1 KNN

Il KNN è un modello di base applicabile a qualsiasi tipo di task. Il suo unico iper-parametro K (il numero di punti più vicini a una nuova istanza da considerare per calcolare la moda o la media della variabile risposta) viene qui ottimizzato con grid-search, utilizzando la 3-fold cross validation. Lo spazio parametrico fissato per K va da 2 a 100, e l'ottimizzazione viene eseguita sul train-set. La metrica di costo adottata è il MAE (per cui si decide di considerare un problema di regressione), ma la decisione finale viene presa in base al risultato del metodo elbow applicato al RMSE (meno interpretabile ma più affidabile). I valori stimati di MAE e RMSE rappresentano la media degli stessi calcolati nei 3 test-set prodotti con la 3-fold cross-validation. Il K ottimale individuato in base a questi criteri risulta essere pari a 13.

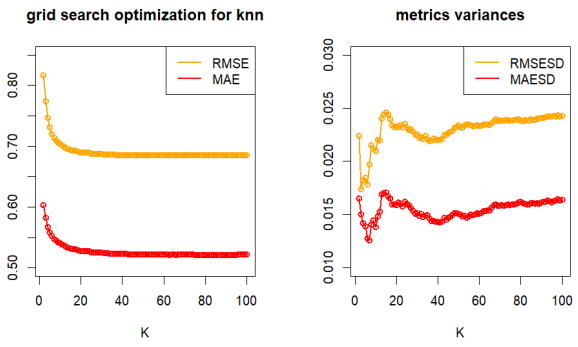

Figura 8 calcolo del RMSE e del MAE del KNN per K che va da 2 a 100

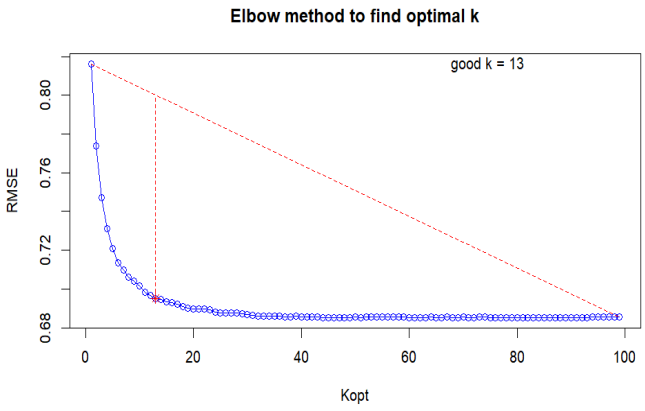

Figura 9 scelta di K per KNN con il metodo elbow applicato sul RMSE

5.1.2 Lasso

Dato l'elevato numero di regressori (25) un modello di regressione lineare potrebbe non risultare ottimale per risolvere il problema di previsione con metodi di regressione. Si può quindi considerare l'utilizzo di un modello Lasso[16], con il solo iper-parametro lambda, al fine di ponderare l'impatto di ciascun regressore con la variabile risposta per il loro grado di correlazione. Il modello lasso cerca infatti di minimizzare la funzione obiettivo:

$$RSS + \lambda \sum_{j=1}^{p} |\beta_j|$$

per cui aggiunge una penalizzazione L1 sui coefficienti dei regressori che non contribuiscono a migliorare significativamente le previsioni della variabile risposta. I modelli Ridge e ElasticNet differiscono dal modello Lasso per la forma della penalizzazione.

L'iper-parametro lambda qui si ottimizza con la 10-fold cross-validation minimizzando il MSE, e risulta sostanzialmente pari a 0 (per la precisione 0.00038) per cui si potrebbe considerare equivalente a un modello di regressione lineare[17].

5.1.3 Alberi di decisione

Se si vuole trattare la previsione della variabile score come un problema di classificazione, si può considerare l'utilizzo di un albero di decisione di tipo CART. Le performance di questo modello sono vicine, anche se quasi sempre inferiori, a quelle degli altri, ma il costo computazionale è notevolmente più basso.
Si può dimostrare, nel task di previsione della variabile score, che le performance raggiunte da un singolo albero di decisione sono praticamente equivalenti a quelle mostrate da un modello RF, ovvero dall'applicazione del bagging su più alberi. Utilizzando il parametro minsplit=20 (numero minimo di osservazioni presenti in una foglia affinché sia consentito eseguire lo split) l'albero si arresta dopo soli tre nodi. Il principale problema di questo modello è l'incapacità di prevedere classi a bassa frequenza.

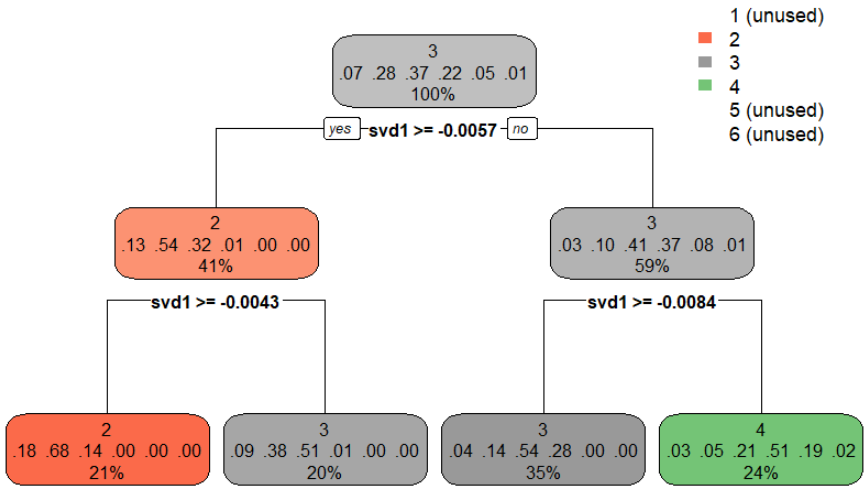

Figura 10 struttura dell'albero di decisione: si vede che vengono previste solo le classi più frequenti, utilizzando la prima componente svd come unico predittore.

[16] Genovese, C. R., Jin, J., Wasserman, L., & Yao, Z. (2012). A comparison of the lasso and marginal regression. *The Journal of Machine Learning Research*, *13*(1), 2107-2143.
[17] https://wavedatalab.github.io/machinelearningwithr/post4.html

5.1.4 GBM e XGB

Le GBM[18] sono modelli di ensemble learning che applicano il metodo del boosting sugli alberi di decisione, e possono configurarsi sia come modelli di regressione sia come modelli di classificazione. Di seguito i risultati dell'ottimizzazione bayesiana[19] degli iper-parametri n.tree, bag.fraction, shrinkage, nel caso della GBM per la classificazione e per la regressione:

iper-parametri	descrizione	regression BayesOpt	classification BayesOpt
shrinkage	learning rate del modello (decimale tra 0 e 1)	0,055	0,05
n.trees	Numero di alberi di boosting, ovvero di iterazioni (intero)	434	517
bag.fraction	frazione delle osservazioni del train-set selezionate casualmente per addestrare ciascun albero (decimale)	0,42	0,42

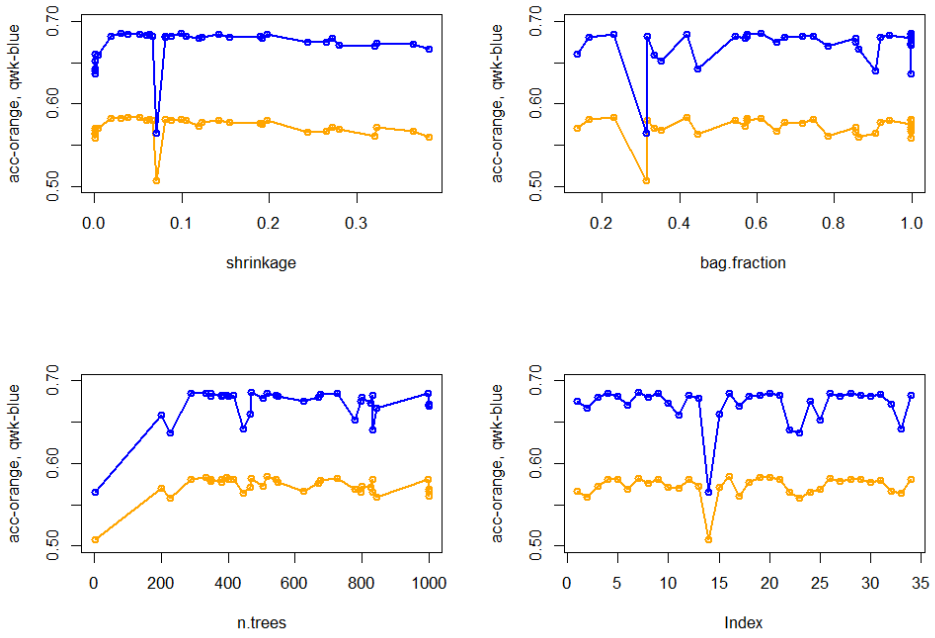

Figura 11 ottimizzazione bayesiana dei parametri della GBM per la classificazione, con accuracy e QWK (40 iterazioni). L'ultimo grafico presenta l'evoluzione delle performance nel corso delle iterazioni.

[18] Fafalios, S., Charonyktakis, P., & Tsamardinos, I. (2020). Gradient boosting trees. *Gnosis Data Analysis PC*, *1*.
[19] Wu, J., Chen, X. Y., Zhang, H., Xiong, L. D., Lei, H., & Deng, S. H. (2019). Hyperparameter optimization for machine learning models based on Bayesian optimization. *Journal of Electronic Science and Technology*, *17*(1), 26-40.

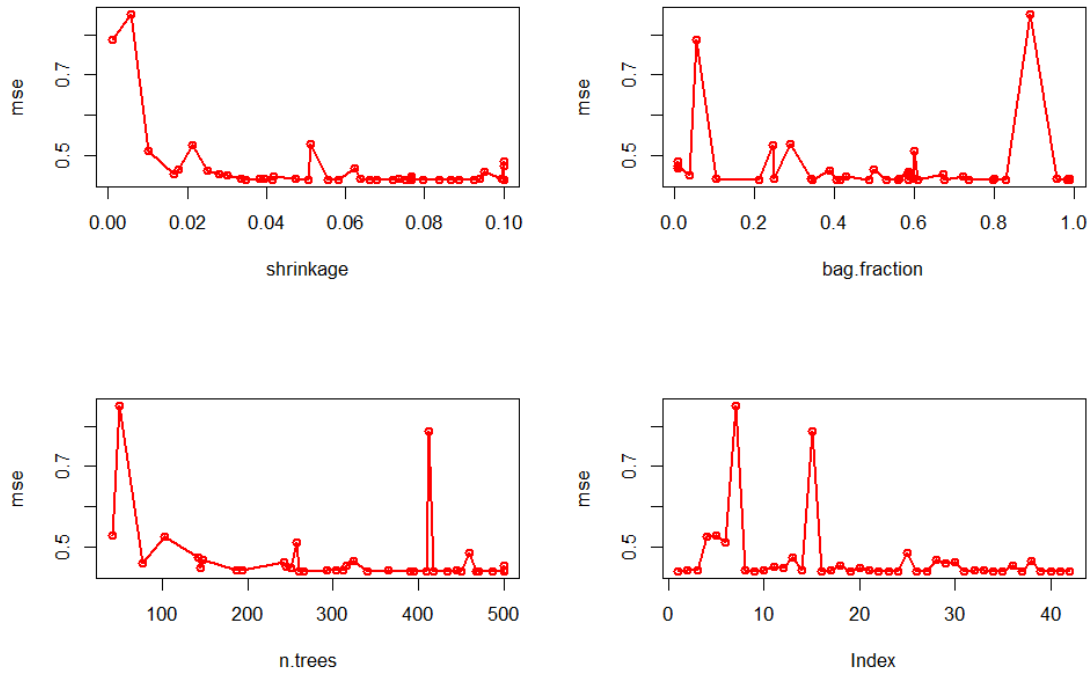

Figura 12 ottimizzazione bayesiana della GBM per la regressione, con MSE come metrica di costo (40 iterazioni)

Alternative alle GBM sono le XGB, comprensive di operazioni come la regolarizzazione (che le rende meno inclini all'overfitting) e la parallelizzazione (che le rende più efficienti dal punto di vista computazionale).

iper-parametri	descrizione	regression BayesOpt	classification BayesOpt
booster	Metodo di boosting tra gbtree, dart e gblinear. (gbtree e dart utilizzano alberi di decisione mentre gblinear usa funzioni lineari)	gblinear	gbtree
nrounds	massimo numero di iterazioni del boosting (intero)	47	38
eta (gbtree)	learning rate (decimale in (0,1))	(inutilizzato)	0,471
alpha (gblinear)	peso degli errori di tipo L1 (decimale)	2,22	(inutilizzato)
gamma (gbtree)	minima riduzione della funzione di perdita richiesta per aggiungere un albero a una foglia.	(inutilizzato)	3,6
lambda (gblinear)	peso degli errori di tipo L2 (decimale)	1,27	(inutilizzato)

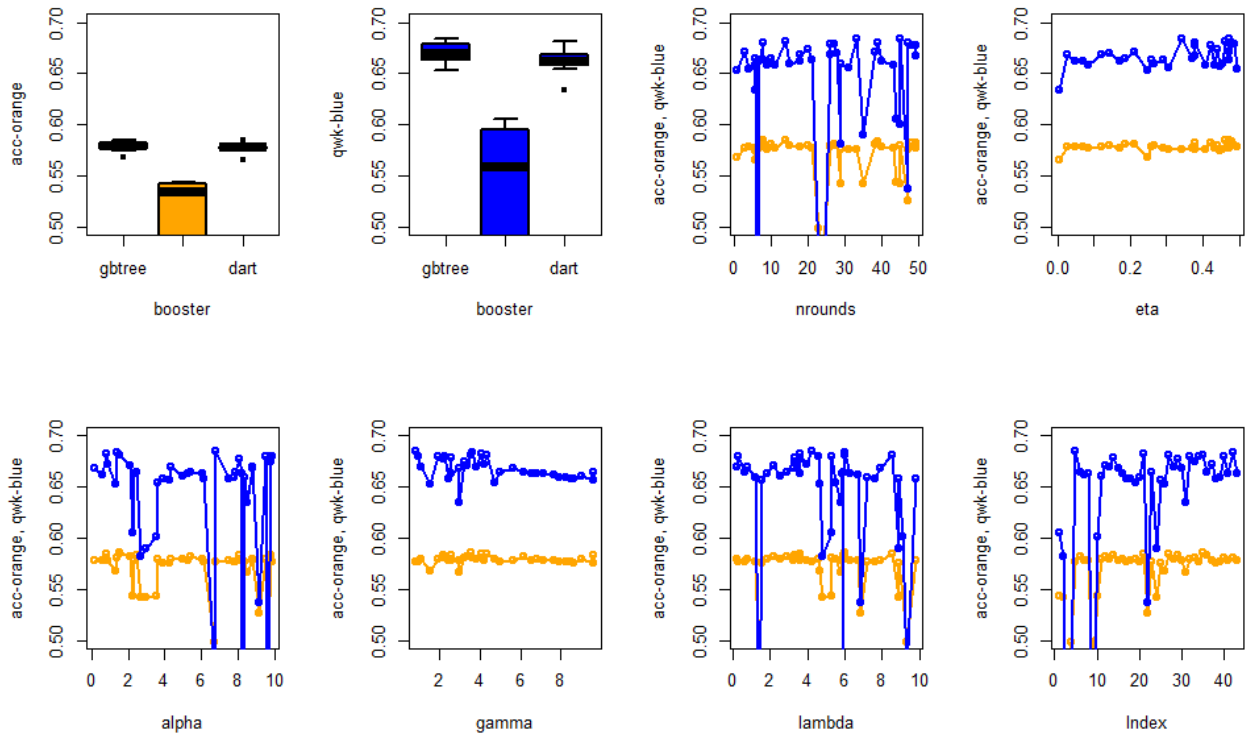

Figura 13 ottimizzazione bayesiana del XGB per la classificazione

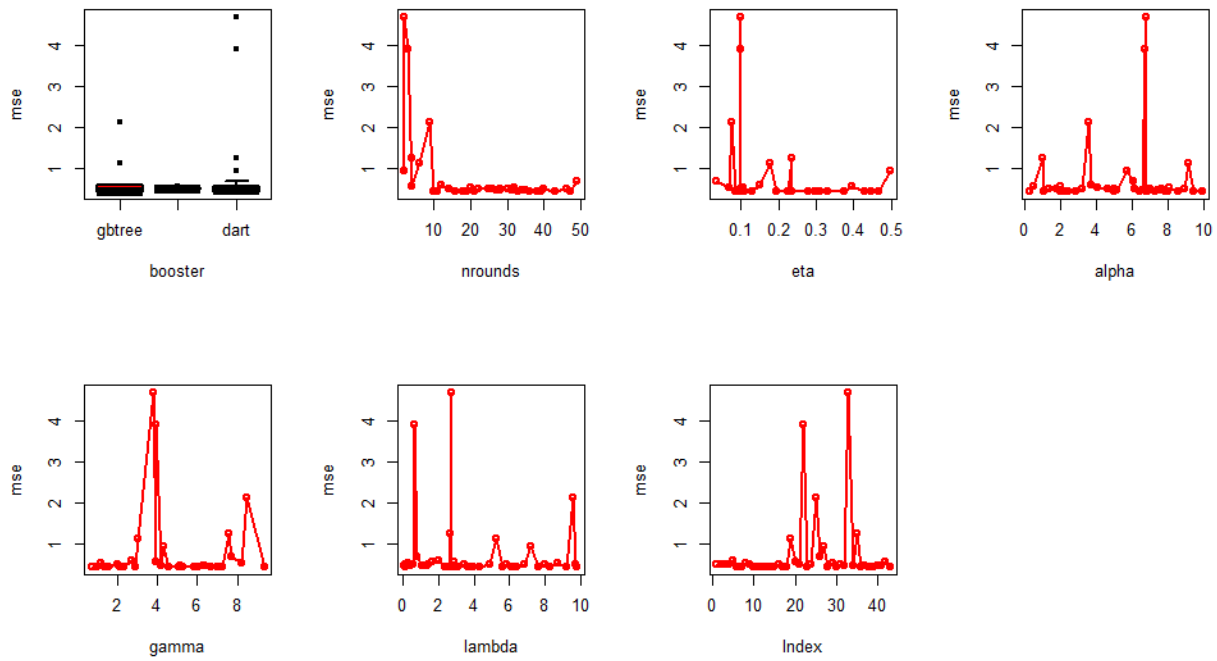

Figura 14 ottimizzazione bayesiana del XGB per la regressione

5.1.5 SVM

Anche le SVM[20] possono configurarsi come strumenti di regressione o di classificazione. In questa sede si ottimizzano separatamente entrambe le configurazioni (rispettivamente con accuracy e mse). Sembra che in entrambi i casi l'ottimizzazione bayesiana conduca a miglioramenti significativi.

iper-parametri	descrizione	regression BayesOpt	classification BayesOpt
kernel	funzione kernel polinomiale o radiale	radial	radial
cost	costante C nella formula di Lagrange	0,75	24
gamma	parametro utilizzato nel kernel	0,093	0,01

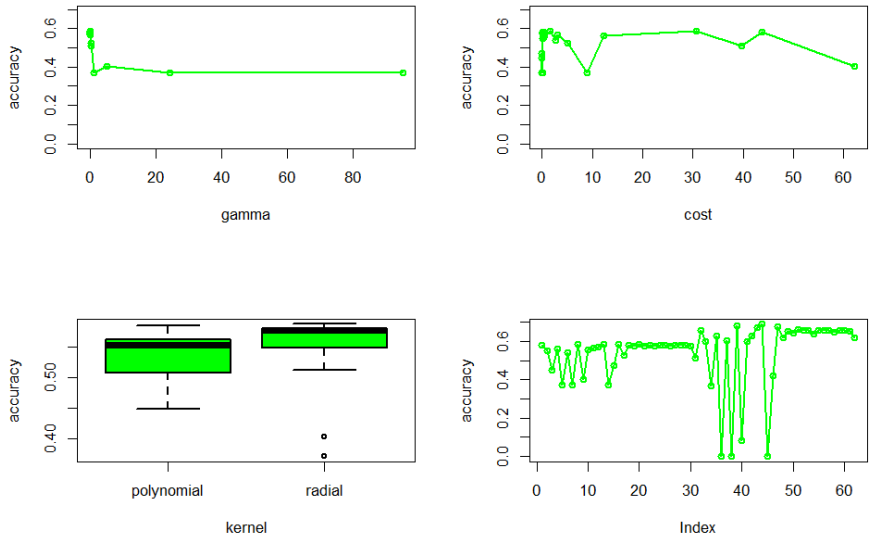

Figura 15 ottimizzazione bayesiana degli iper-parametri della SVM per la classificazione (30 iterazioni)

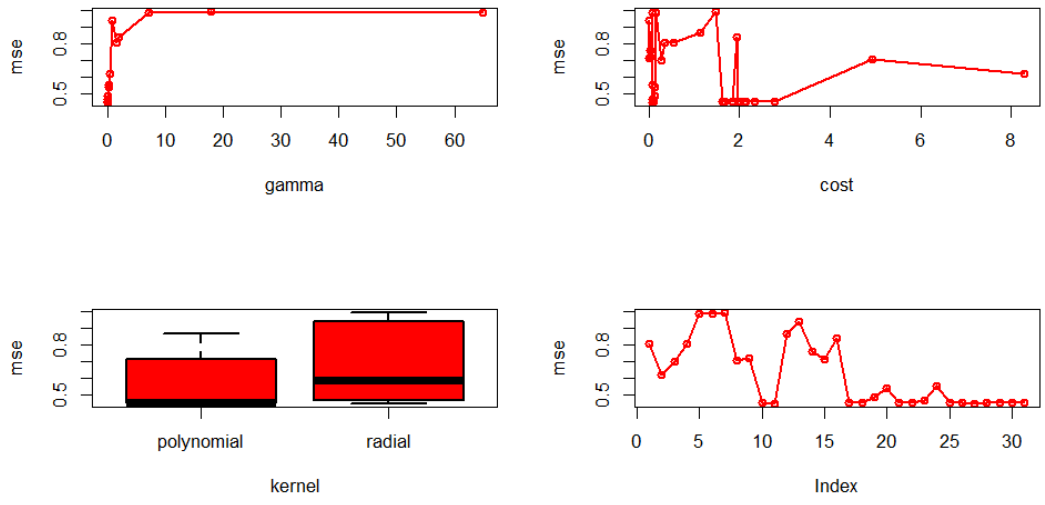

Figura 16 ottimizzazione bayesiana degli iper-parametri della SVM per la regressione (30 iterazioni)

[20] Meyer, D., & Wien, F. T. (2001). Support vector machines. *R News*, *1*(3), 23-26.

5.1.6 primi risultati e confronti

Al fine di mettere a confronto i diversi modelli, con i relativi iper-parametri già ottimizzati, si eseguono 10 ricampionamenti bootstrap sul test-set[21] (di circa 12000 osservazioni), con il 50% osservazioni per la fase di training e l'altro 50% per la fase di testing, ottenendo quindi 10 stime delle metriche di performance di ciascun modello. In questo modo possiamo identificare i più performanti per ciascuna metrica. Elapsed misura congiuntamente il tempo di training e testing, in secondi, per una iterazione.

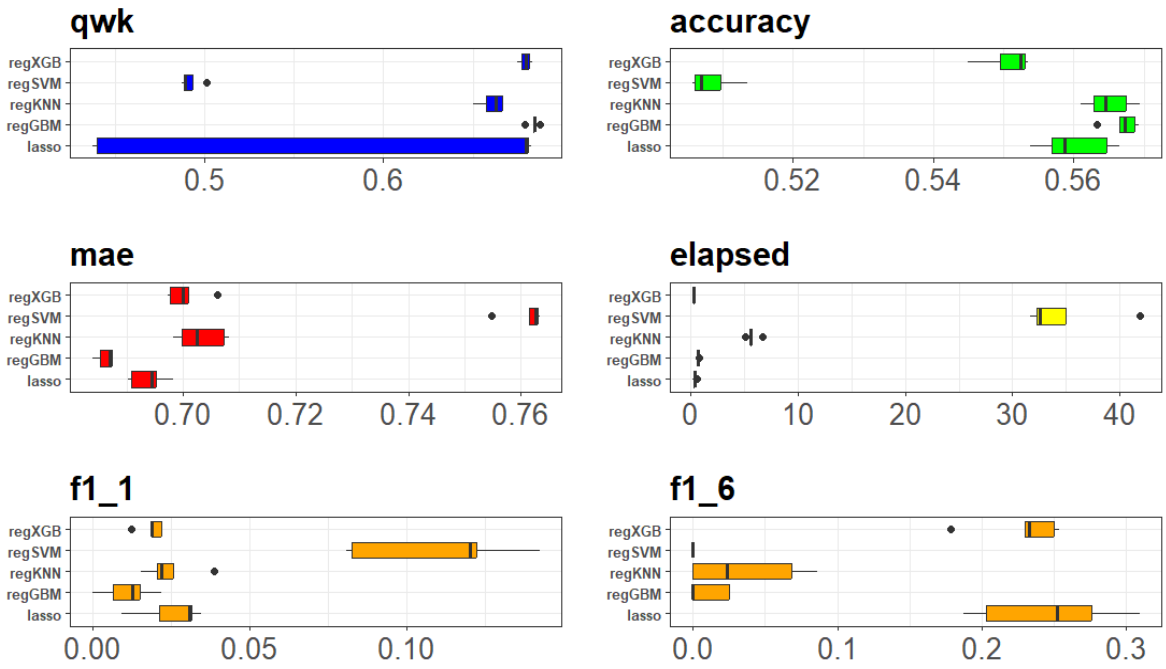

Figura 17 confronto bootstrap delle performance dei modelli di regressione.

[21] Kohavi, R. (1995, August). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai* (Vol. 14, No. 2, pp. 1137-1145).

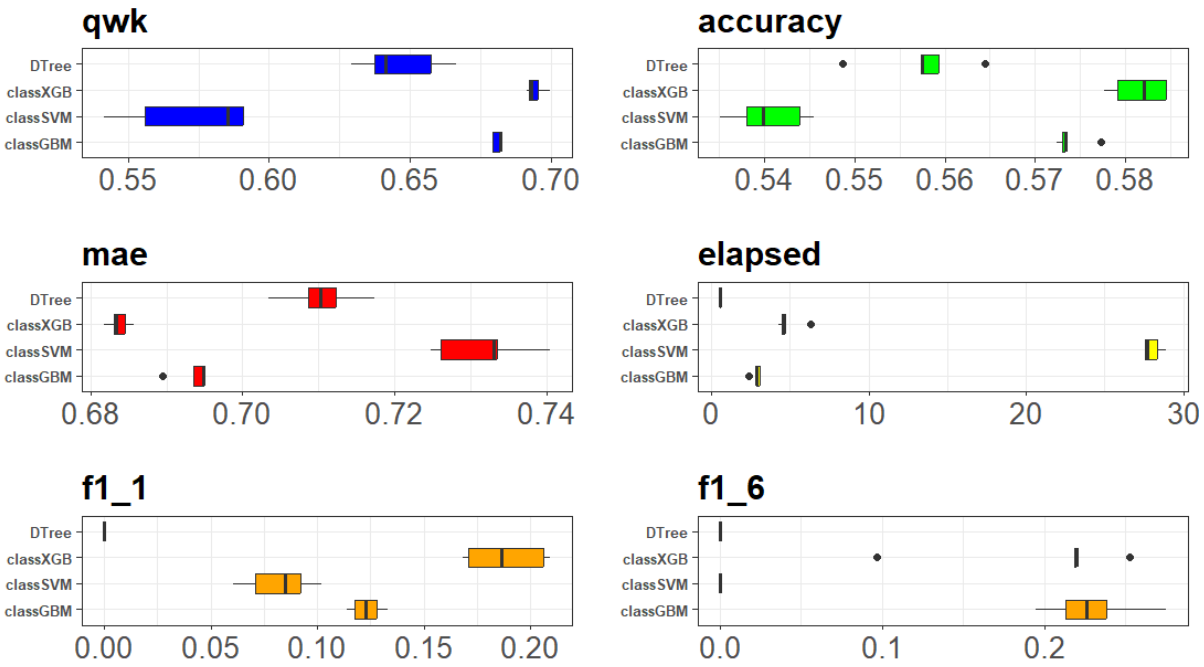

Figura 18 boostrap delle performance dei modelli di classificazione

Si nota la superiorità dei modelli di boosting sulla maggior parte delle metriche, nonché l'inadeguatezza delle SVM rispetto al problema affrontato.

5.2 Modelli di ranking adottati

5.2.1 Regressione Logistica Ordinale

Il modello di regressione logistica ordinale[22] (anche definita logit cumulata o proportional odds) è costruito appositamente per la risoluzione di problemi di ranking e non prevede il fissaggio di alcun iper-parametro. Nonostante l'aggiustamento dei suoi parametri venga svolto attraverso metodi numerici, il suo costo computazionale è tra i più bassi tra quelli dei modelli considerati.

Data y_i una variable risposta discreta ordinata, riferita all'individui i-esimo, con c classi, dato α_j l'intercetta class specific per la classe j-esima, $j = \{1, \dots, c\}$, dato il vettore di regressori x_i con lunghezza pari al relativo vettore di coefficienti β, comune per tutte le classi, allora si ha che

$$logit\big(P(y_i \leq j)\big) = ln\left(\frac{P(y_i \leq j)}{P(y_i > j)}\right) = \alpha_j + \beta x_i$$

$$P(y_i \leq j) = \frac{exp(\alpha_j + \beta x_i)}{1 + exp(\alpha_j + \beta x_i)}$$

$$P(y_i = j) = P(y_i \leq j) - P(y_i \leq j - 1)$$

I coefficienti β e α_j si ricavano massimizzando la funzione di verosimiglianza proposta da McCullagh[23]. La stima della probabilità di una singola classe come differenza tra la cdf stimata della classe

[22] Agresti, A. (2010). Modeling ordinal categorical data. *Anal Ordinal Categ Data*, *75*(10.1002), 9780470594001.
[23] McCullagh, P. (1980). Regression models for ordinal data. *Journal of the Royal Statistical Society: Series B (Methodological)*, *42*(2), 109-127.

medesima e della precedente verrà ripresa come strategia di ensemble learning per la trattazione di problemi di ranking nella sezione 5.3.1.

5.2.2 XGB per il ranking

Se si vuole costruire un modello XGB per la classificazione ordinale, è possibile utilizzarlo per minimizzare la pariwise logistic loss, una misura di costo adatta a un problema di ranking[24]. La pairwise logistic loss function si definisce come segue[25]:

Dati una serie di vettori di predittori $\overline{\boldsymbol{x}} = \{\boldsymbol{x}_i\}_{i=1}^{N},\ \boldsymbol{x}_i \in X,$ e di osservazioni di una variabile risposta $\overline{y} = \{\boldsymbol{y}_i\}_{i=1}^{N}$ con $y_i \in Y = \{1,..,c\}$ dove c è il numero di classi ordinate, $1 <...< c$, che vengono stimate tramite la funzione $f: X \rightarrow Y$, si ha che

$$pairwiseLoss = L(\overline{\boldsymbol{x}}, \overline{y}, f) = \sum_{(i,j)\in Pairs} loss_{ij}$$

Dove, per ogni possibile coppia di osservazioni (in tutto $\binom{N}{2}$),

$$loss_{ij} = -\Big(z_{ij} * ln\big(p_{ij}\big) + \big(1 - z_{ij}\big) * ln\big(1 - p_{ij}\big)\Big)$$

Con

$$\Delta_{\mathrm{ij}} = f(x_i) - f\big(x_j\big)$$

$$p_{ij} = \frac{1}{\Big(1 + exp\big(-\Delta_{ij}\big)\Big)}$$

$$z_{ij} = \begin{cases} 1 & y_i > y_j \\ 0 & y_i < y_j \end{cases}$$

Dove si escludono le iterazioni per cui vale $y_i = y_j$. Il nome "logistic" deriva dall'all'applicazione della funzione logit per trasformare la differenza tra i ranghi previsti Δ_{ij} in un valore di probabilità (per cui la minimizzazione della pairwise loss con questa funzione corrisponde alla massimizzazione di una funzione di verosimiglianza). Altri modelli di ranking utilizzano le funzioni hinge-max ed esponenziale al posto della logit[26].

5.2.3 performance dei modelli di ranking

Si può vedere la superiorità dei due modelli di ranking proposti su tutti quelli di classificazione e regressione finora sperimentati. Si nota che la regressione logistica ordinale, e la maggior parte dei modelli di boosting, prevalgono sulla XGB ordinale in termini di MAE e Accuracy, mentre la XGB ordinale si mostra la migliore nella QWK (l'unica metrica di ranking) e in parte nella previsione delle code. La regressione logistica ordinale è comunque il modello di ranking più efficiente in termini di costo temporale.

[24] https://xgboost.readthedocs.io/en/stable/tutorials/learning_to_rank.html#training-with-the-pairwise-objective
[25] https://pykeen.readthedocs.io/en/v1.5.0/reference/losses.html
[26] Chen, W., Liu, T. Y., Lan, Y., Ma, Z. M., & Li, H. (2009). Ranking measures and loss functions in learning to rank. *Advances in Neural Information Processing Systems*, *22*.

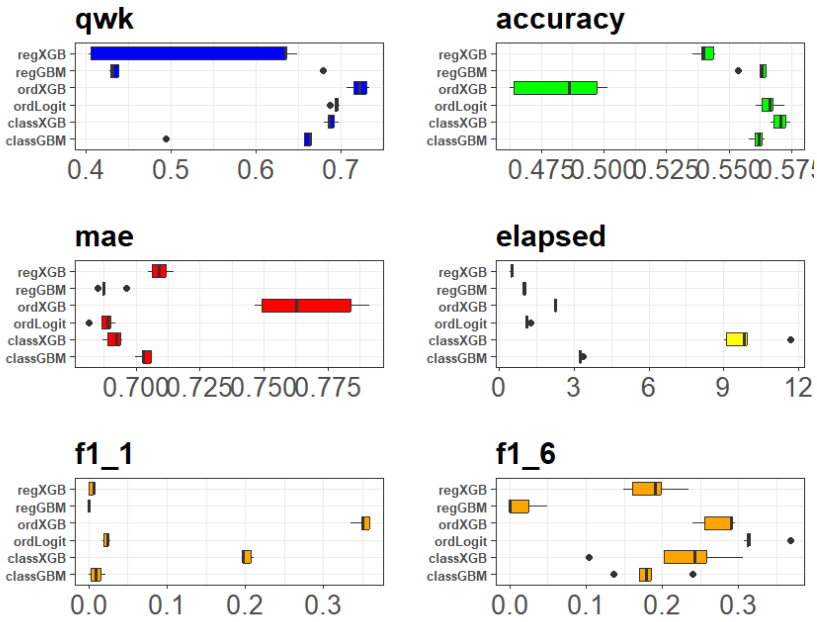

Figura 19 confronto tra modelli di ranking e modelli di boosting per classificazione e regressione

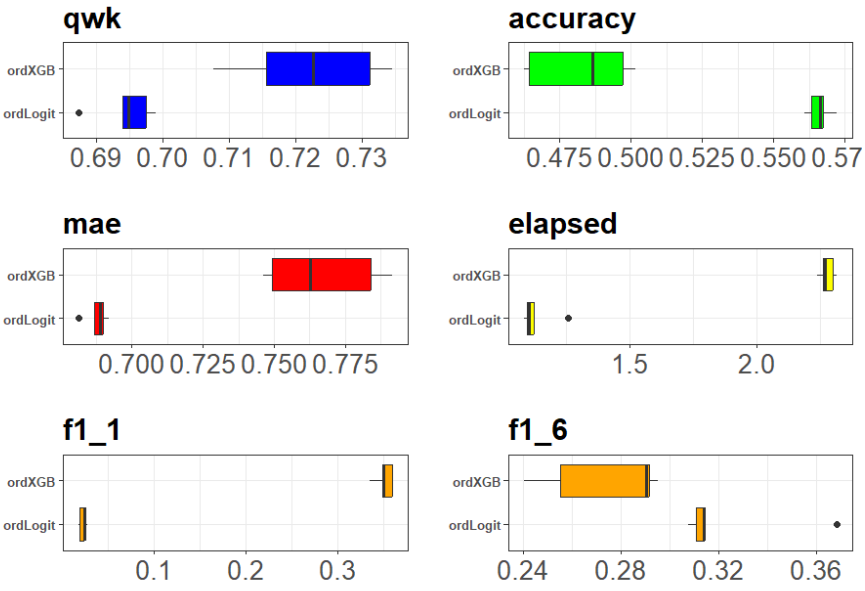

Figura 20 confronto boostrap tra regressione logitica cumulata e XGB con pairwise logistic loss

5.2.4 Ensemble learning per il ranking

La regressione logistica cumulata basa le sue previsioni sulla differenza tra la cdf stimata di una classe e della precedente.

$$P(y_i = j) = P(y_i \leq j) - P(y_i \leq j - 1)$$

È allora possibile basare un metodo di ensemble learning per la previsione di K classi ordinate utilizzando K-1 modelli di classificazione binaria per stimare le cdf di ciascuna classe (K-1 perché la classe K-esima presenta cdf $= P(y_i \leq K) = 1$). Se infatti si dispone di una variabile risposta ordinale one-hot encoded, ovvero con una struttura tale per cui:

$$y_{ij} = \begin{cases} 1 & y_i = j \\ 0 & y_i \neq j \end{cases}$$

Allora è facile ricavare K-1 variabili risposta dummy indicanti se l'istanza corrente presenta una classe maggiore o uguale alla classe j-esima:

$$y_{ij} = \begin{cases} 1 & y_i \geq j \\ 0 & y_i < j \end{cases}$$

$$P(y_i = j) = \big(1 - P(y_i \geq j)\big) - \big(1 - P(y_i \geq j - 1)\big) = P(y_i \geq j - 1) \; - \; P(y_i \geq j)$$

j=1	j=2	j=3	j=4	j=5	j=6		j<=1	j<=2	j<=3	j<=4	j<=5	j<=6
0	0	1	0	0	0		0	0	1	1	1	1
0	1	0	0	0	0		0	1	1	1	1	1
0	0	0	1	0	0		0	0	0	1	1	1
0	0	1	0	0	0		0	0	1	1	1	1
0	0	0	1	0	0		0	0	0	1	1	1
0	1	0	0	0	0		0	1	1	1	1	1

Figura 21 codifica della variabile score one-hot encoded in una serie di variabili risposta per un gruppo di classificatori binari per il ranking

Per sperimentare questo metodo, si utilizzano un gruppo di 5 alberi di decisione, un gruppo di 5 KNN, un gruppo di 5 SVM e un gruppo di 5 GBM, tutti classificatori binari che restituiscono la probabilità della cdf associata a ciascuna classe, utilizzando i medesimi predittori. Al fine di garantire la condizione per cui $P(y_i \leq j) \; > P(y_i \leq j - 1)$ sempre, è necessario che tutti i modelli nel gruppo di ranking condividano gli stessi iper-parametri. Si vede che i modelli di ensemble learning per il ranking presentano quasi sempre performance migliori rispetto ai singoli modelli di classificazione e regressione:

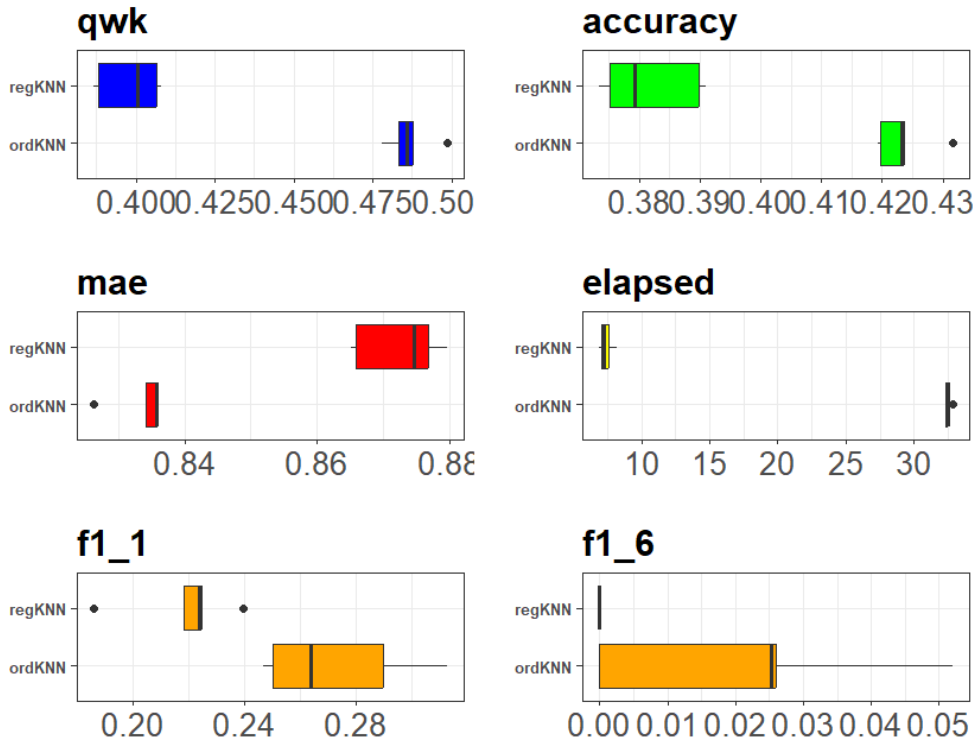

Figura 22 confronto fra KNN per la regressione e ensemble learning di 5 KNN binari per il ranking

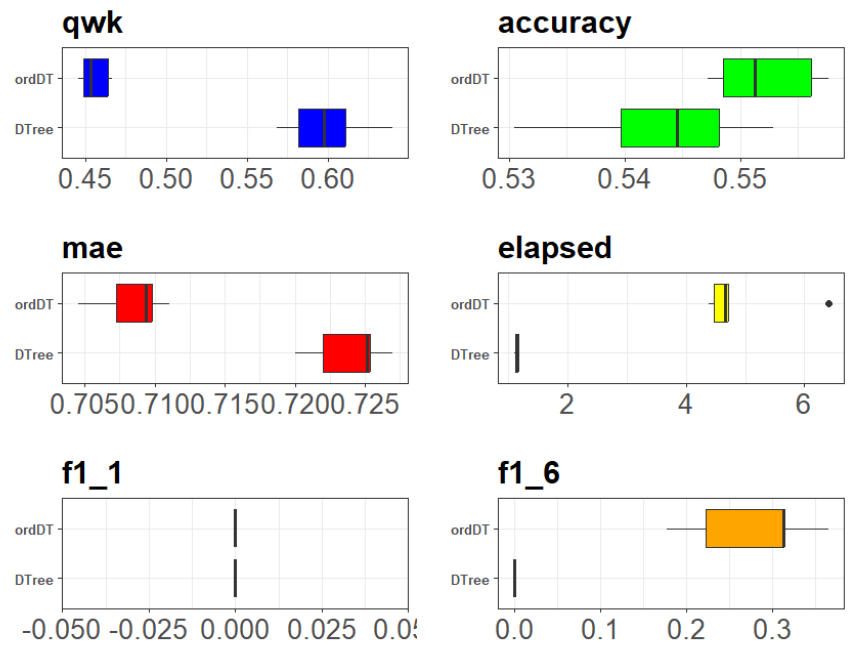

Figura 23 confronto fra un albero di decisione e 5 alberi binari per il ranking

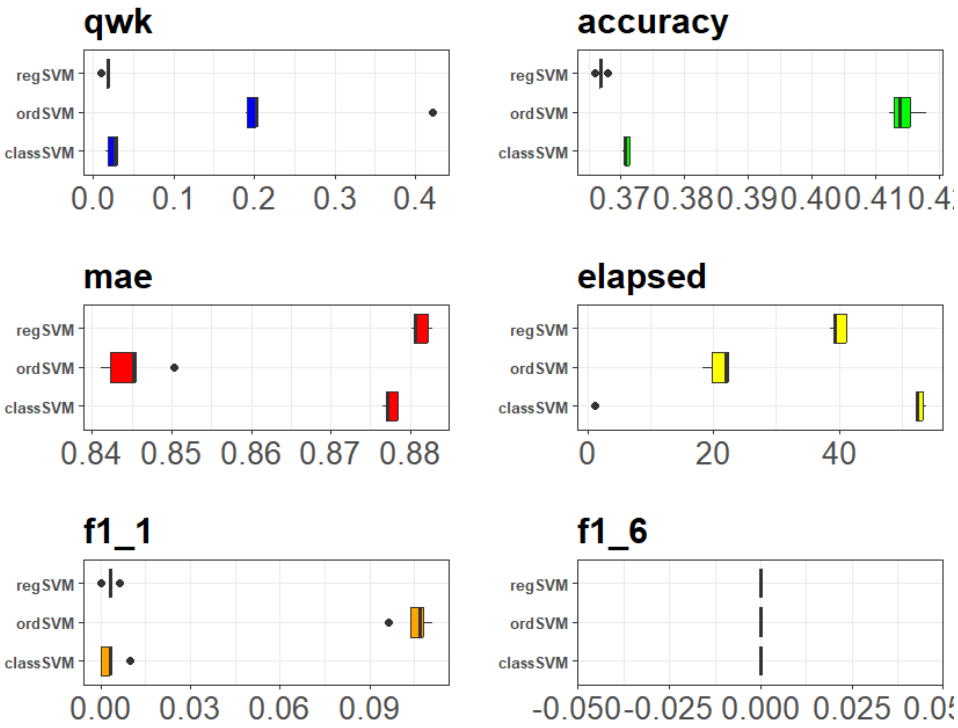

Figura 24 confronto fra SVM per regressione e classificazione e 5 SVM binari per il ranking

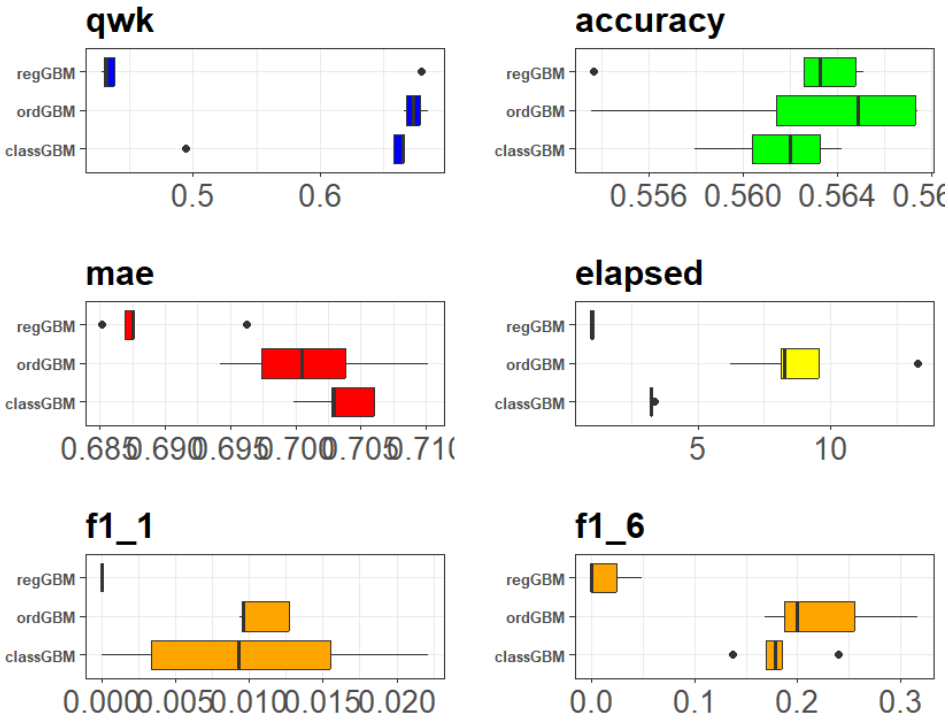

Figura 25 confronto bootstrap tra GBM di regressione e classificazione e gruppi di GBM binari per i ranking

Tra i modelli ordinali, il modello più performante in termini di QWK rimane il XGB con pairwise loss, seguito dalla regressione logistica ordinale e dal gruppo di GBM binari:

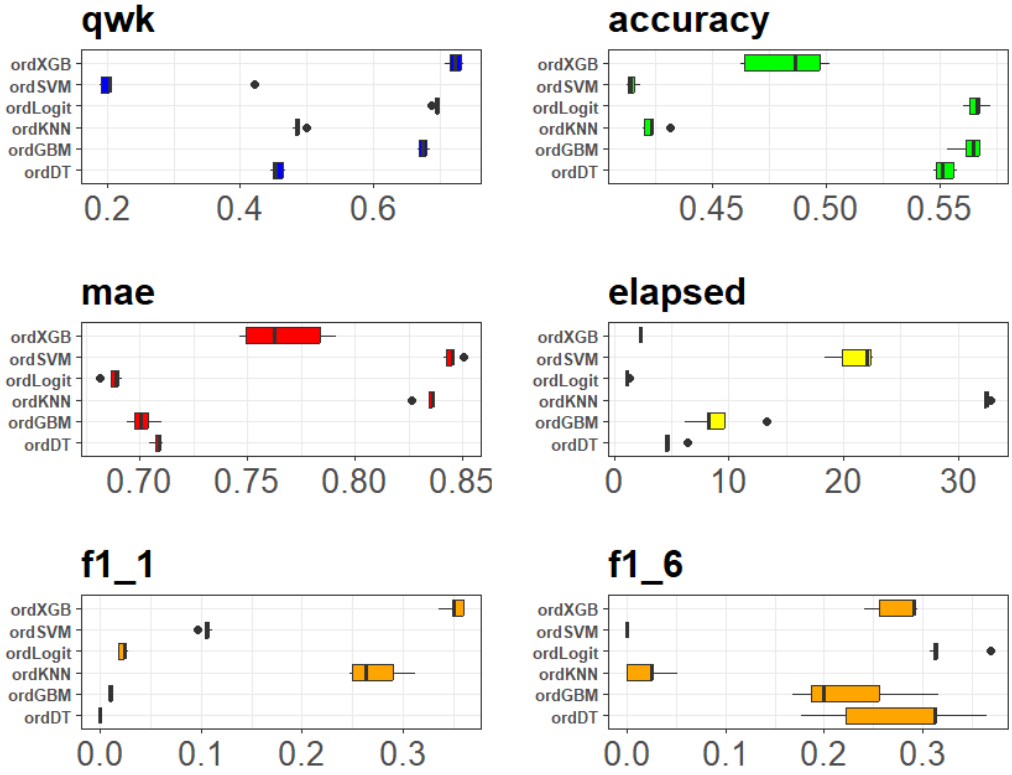

Figura 26 confronto bootstrap tra i modelli di ranking

6. Conclusioni

Le replicazioni bootstrap delle fasi di training e testing di ciascun modello supervisionato mostrano l'effettiva superiorità dei modelli di ranking sui modelli di regressione e classificazione, e riflettono la natura ordinale della variabile score. L'utilizzo delle metriche di valutazione F1 sulle code è essenziale per poter valutare l'effettiva utilità pratica di un modello supervisionato, che in un contesto reale dovrebbe essere in grado, come un qualsiasi insegnante, di assegnare tutti i voti agli studenti, e non solo quelli intermedi e più frequenti. Si nota un trade-off tra le performance sulla Kappa Quadratica di Cohen e le performance di Accuracy e MAE, particolarmente evidente nei risultati del modello XGB con pairwise loss. In generale, sembra che modelli più performanti in termini di QWK presentino anche delle F1 accettabili sulle code.
Accuracy e MAE sono strumenti utili per fornire interpretabilità ai risultati, ma in un task di ranking un modello dovrebbe essere ottimizzato secondo metriche di valutazione o di costo basate sulla qualità dell'ordinamento.
I modelli migliori presentano mediamente un'accuracy intorno al 55%, un MAE pari a circa 0.7, e una QWK pari o superiore al 71%. La competizione AES di Kaggle è basata interamente sulle performance in termini di QWK, e la più alta raggiunta dal primo classificato è pari all' 84%.
Vi sono molti miglioramenti e strade alternative a quelle qui proposte per affrontare il problema.

In primo luogo, il preprocessing dei dati può essere svolto con metodi più articolati rispetto a quello della DTM, come il word2vec o la TF-IDF. La decomposizione in componenti principali, se effettuata, può adottare il t-SNE al posto della SVD, o considerare l'estrazione di un numero più elevato di componenti.

In secondo luogo, molti dei partecipanti che hanno raggiunto elevati livelli di QWK hanno fatto ricorso a modelli a elevatissimo costo computazionale, come transformer e DEBERTA, utilizzando quindi modelli per il NLU (natural language understanding) che richiederebbero elevati costi in termini di risorse computazionali. Una QWK di 10 punti inferiore sembra quindi accettabile, se ricavata con modelli a bassissimo costo computazionale come la regressione logistica ordinale o il modello XGB.

Infine, va evidenziato come la natura del task sia estremamente soggettiva, e come non possa esistere un valutatore perfetto e universale, umano o statistico, capace di valutare nello stesso modo gli infiniti temi che sarebbe possibile scrivere. Di conseguenza la presenza di differenze fra una stima del modello e una effettiva osservazione nel dataset potrebbe sempre considerarsi come un errore dell'insegnante umano, piuttosto che del modello. D'altro canto, bisogna ricordare che i modelli qui presentati basano le proprie stime sui conteggi delle parole inserite nel testo, senza considerare minimamente né il loro ordinamento, né il significato logico del contenuto.

7. Bibliografia

[1] https://the-learning-agency-lab.com/

[2]https://www.kaggle.com/competitions/learning-agency-lab-automated-essay-scoring-2

[3] https://www.kaggle.com/competitions/asap-aes/overview

[4] https://storage.googleapis.com/kaggle-forum-message-attachments/2733927/20538/Rubric_%20Holistic%20Essay%20Scoring.pdf

[5] Radovanović, M., & Ivanović, M. (2008). Text mining: Approaches and applications. *Novi Sad J. Math*, *38*(3), 227-234.

[6] Yun-tao, Z., Ling, G., & Yong-cheng, W. (2005). An improved TF-IDF approach for text classification. *Journal of Zhejiang University-Science A*, *6*(1), 49-55.

[7] Rei, M., & Briscoe, T. (2014, June). Looking for hyponyms in vector space. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning* (pp. 68-77).

[8] Antonellis, I., & Gallopoulos, E. (2006). Exploring term-document matrices from matrix models in text mining. *arXiv preprint cs/0602076*.

[9] Radovanović, M., & Ivanović, M. (2008). Text mining: Approaches and applications. *Novi Sad J. Math*, *38*(3), 227-234.

[10] Schofield, A., Magnusson, M., & Mimno, D. (2017, April). Pulling out the stops: Rethinking stopword removal for topic models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, short papers* (pp. 432-436).

[11] https://quanteda.io/

[12] Yang, J., & Watada, J. (2011, June). Decomposition of term-document matrix representation for clustering analysis. In *2011 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2011)* (pp. 976-983). IEEE.

[13] Falini, A. (2022). A review on the selection criteria for the truncated SVD in Data Science applications. *Journal of Computational Mathematics and Data Science*, *5*, 100064.

[14] Akbari, Z., & Unland, R. (2016). Automated determination of the input parameter of DBSCAN based on outlier detection. In *Artificial Intelligence Applications and Innovations: 12th IFIP WG 12.5 International Conference and Workshops, AIAI 2016, Thessaloniki, Greece, September 16-18, 2016, Proceedings 12* (pp. 280-291). Springer International Publishing.

[15] Vaughn, D., & Justice, D. (2015). On the direct maximization of quadratic weighted kappa. *arXiv preprint arXiv:1509.07107*.

[16] Genovese, C. R., Jin, J., Wasserman, L., & Yao, Z. (2012). A comparison of the lasso and marginal regression. *The Journal of Machine Learning Research*, *13*(1), 2107-2143.

[17] https://wavedatalab.github.io/machinelearningwithr/post4.html

[18] Fafalios, S., Charonyktakis, P., & Tsamardinos, I. (2020). Gradient boosting trees. *Gnosis Data Analysis PC*, *1*.

[19] Wu, J., Chen, X. Y., Zhang, H., Xiong, L. D., Lei, H., & Deng, S. H. (2019). Hyperparameter optimization for machine learning models based on Bayesian optimization. *Journal of Electronic Science and Technology*, *17*(1), 26-40.

[20] Meyer, D., & Wien, F. T. (2001). Support vector machines. *R News*, *1*(3), 23-26.

[21] Kohavi, R. (1995, August). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai* (Vol. 14, No. 2, pp. 1137-1145).

[22] Agresti, A. (2010). Modeling ordinal categorical data. *Anal Ordinal Categ Data*, *75*(10.1002), 9780470594001.

[23] McCullagh, P. (1980). Regression models for ordinal data. *Journal of the Royal Statistical Society: Series B (Methodological)*, *42*(2), 109-127.

[24] https://xgboost.readthedocs.io/en/stable/tutorials/learning_to_rank.html#training-with-the-pairwise-objective

[25] https://pykeen.readthedocs.io/en/v1.5.0/reference/losses.html

[26] Chen, W., Liu, T. Y., Lan, Y., Ma, Z. M., & Li, H. (2009). Ranking measures and loss functions in learning to rank. *Advances in Neural Information Processing Systems*, *22*.