882679 - Rizzolo Alessandro, 881997 - Vo Duy Khiem

PROGETTO DWM

FEATURE ENGINEERING

Aggiunta delle feature

Età della casa da quando è stata rinnovata a quando è stata venduta
Bldg_Year = Year_Remod_Add - Year_Sold

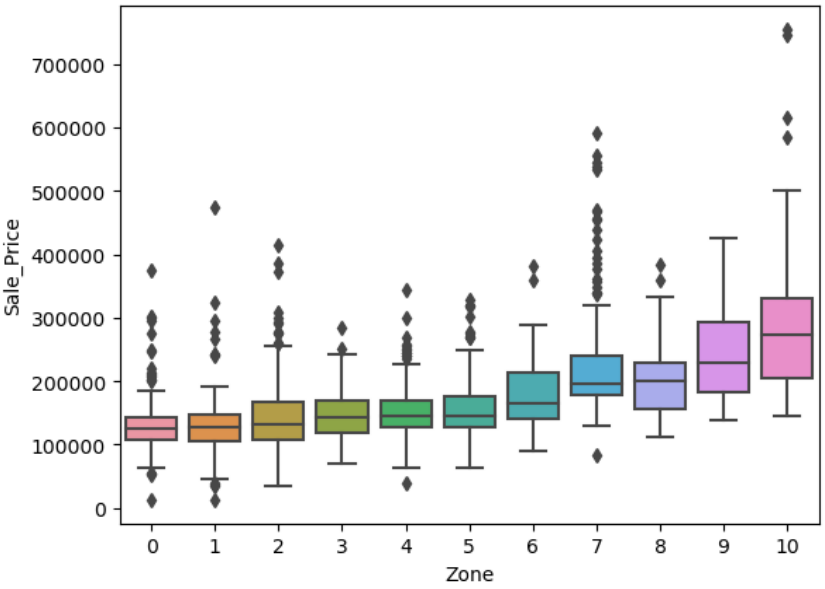
Numero di bagni indistintamente dall'half a quello completo
Tot_Bath = Half_Bath + Full_Bath + Bsmt_Half_Bath + Bsmt_Full_Bath

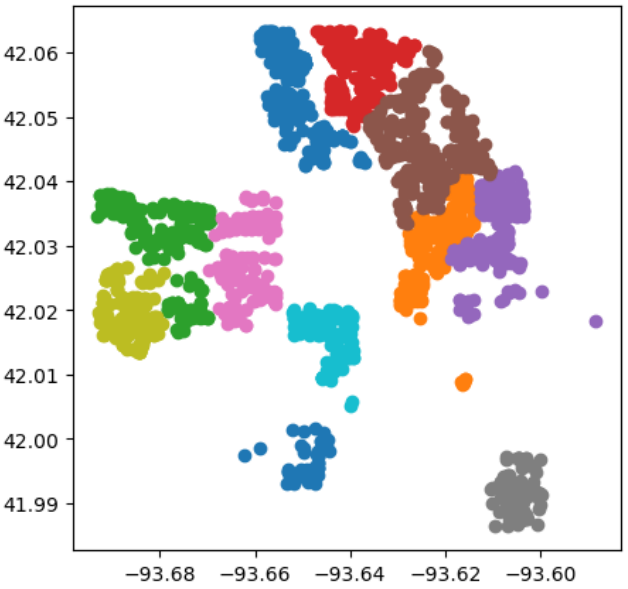
Rapporto fra il numero di auto e l'area del garage
Garage_Perc = Garage_Cars / Garage_Area
Ovviamente sostituendo con 0 quando non era presente il garage

Somma del primo e secondo piano quindi risultante un piano intero
All_Flr_SF = First_Flr_SF + Second_Flr_SF

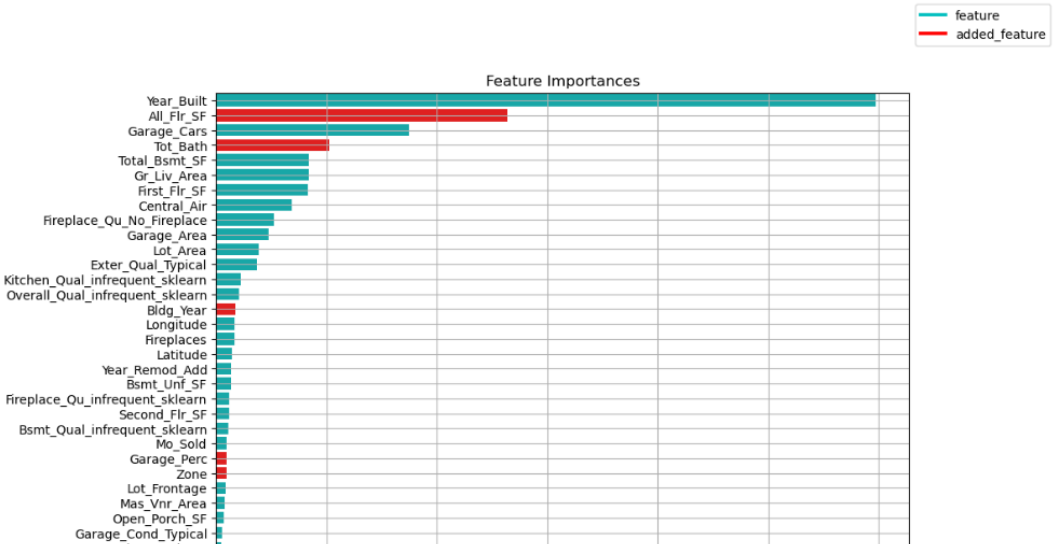
Cluster

Oltre alle feature di costruzione matematica abbiamo voluto creare un cluster basandosi sulla longitudine e latitudine costruendo così la feature "Zone", il cui valore al suo interno è la mediana del prezzo della zona in cui si trova.
Testando infatti vari algoritmi di clustering come l'agglomerative, dbscan e gaussian mixture,abbiamo valutato attraverso silhouette e alla feature selection della random forest il miglior modello e il miglior numero di cluster per il nostro dataset, ovvero agglomerative clustering con numero di cluster=11.





La risultante feature importance determinata dalla RandomForest delle prime 30 colonne è:



come si può notare alcune delle nuove feature vengono rappresentate come molto importanti, mentre altre pur essendo tra le prime 30 su 138 vengono usate meno a livello di predizione, come nell'esempio di "Zone". Essendo costruita su un cluster prendendo solo il punto di vista posizionale, ignora gli outliers presenti, rendendolo così poco ottimale.

One-hot encoding

Studiando il database originale abbiamo constatato che necessita solamente della one-hot encoding per la trasformazione di valori stringa in valori binari per i vari algoritmi successivi.
Attraverso dunque un analisi del database abbiamo effettuato sia la conversione dei "simil booleani" cioè quelle feature aventi solo variabili dicotomiche e la one-hot encoding.

Attraverso la random forest abbiamo successivamente calcolato quali valori fossero ottimali.
Abbiamo preferito dividere il database in 3 diversi campioni con cui far allenare poi i regressori.
Quello che abbiamo definito short indica il database dove attraverso l'algoritmo definito da SK-Learn come "SelectFromModel" ci ritorna un database di 29 feature, mentre la medium attraverso all'analisi dei quantili di propone il 70% del dataset. Infine il database originale senza la selezione delle feature di scarso valore.

Normalizzazione della variabile Sale_Price

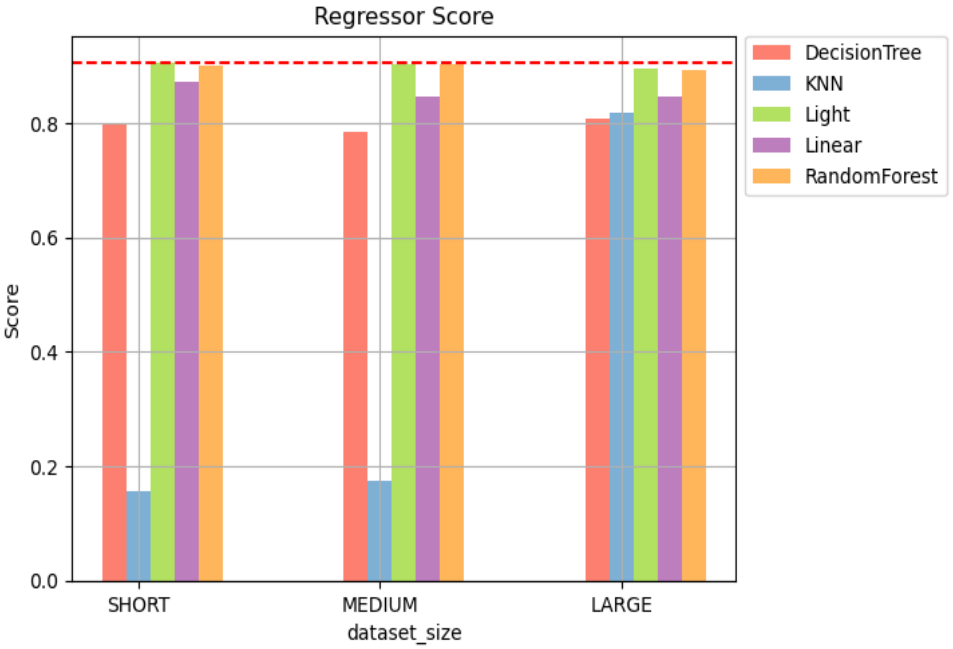
La normalizzazione dei dati ci aiuta a rendere più semplice il processo di addestramento, evitando che ci possa essere una possibile predizione negativa.
Inoltre migliora anche il grafico dei residui.
Attraverso algoritmi come la Linear Regression ci permette di trovare una relazione lineare tra le variabili indipendenti e dipendenti tra di loro.
Il modello assegna infatti un peso a ciascuna variabile indipendente per determinare l'importanza di ciascuna di predire il valore della variabile dipendente.

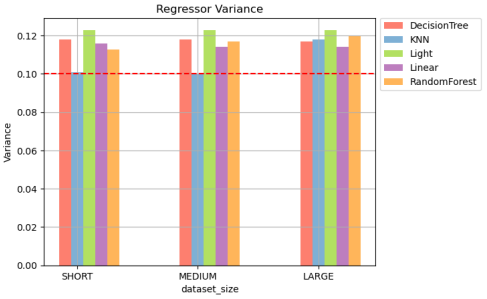
Modelli di Regressione:

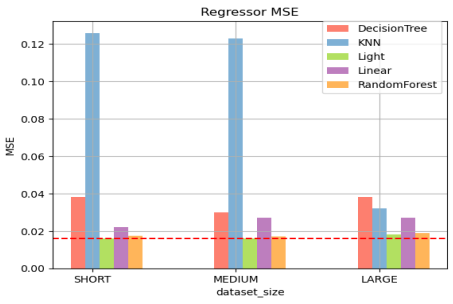
- **DecisionTreeRegressor**
- **KNN**
- **Linear Regression**
- **Random Forest**
- **LightGBM Regressor**

Miglior Score: 90.8% nel modello LGBMRegressor con lo SHORT dataset
Peggior Score: 27.8% nel modello : KNeighborsRegressor con lo SHORT dataset

Per la valutazione dei nostri regressori abbiamo optato per il confronto su 3 tipi di feature selection: la Random Forest con quantile al 0.70 (che costruisce il medium dataset(~45 feature) ed infine con nessuna eliminazione, questo per valutare al meglio quale regressore riuscisse a predirre con più precisione il nostro Sale_Price. Difatti si può notare che sia il Decision Tree che il KNN hanno prestazioni migliori (nel caso dell'ultimo citato anche di molto) in assenza di una feature selection ovvero mantenendo tutte le 138 colonne, al contrario sia la LightGBM che il Linear Regressor hanno Score più alti con un dataset ristretto ma con feature più importanti. In opposizione agli ultimi la Random Forest riscontra prestazioni elevate nella Medium.







CONCLUSIONE:

Possiamo notare nei grafici dunque che tra i tre dataset usati nei regressori, la short e medium sono pressoché simili dal punto di vista dei risultati. Prendendo in considerazione tra questi ultimi la Short e mettendolo a confronto con il database completo, dopo la pulizia dei dati, possiamo notare come sia significativa la differenza di score del KNN. Questo dato è possibile probabilmente data la quantità di dati disponibili. Infatti questo algoritmo si basa sulla distanza tra un punto predetto e tutti gli altri punti, avendone a disposizione di più otteniamo una precisione se si può dire migliore a discapito però di una varianza prettamente elevata.
Le prestazioni ottimali li abbiamo avuti infatti con la LGBMRegressor che nei rispettivi grafici ottiene un ottimo punteggio come score, una bassa varianza ma non rispetto a tutte, a discapito di un ottimo MSE cioè la bontà tra i valori previsti e quelli osservati.

Possibili miglioramenti:

- zone
- encoding
- selezione iperparametri

Per il cluster zone è possibile migliorare la valutazione estrinseca andando ad osservare meglio la distribuzione delle case ed i loro prezzi, possibilmente eseguendo, ove possibile un tuning dei parametri.

È possibile, inerente all'encoding (poiché si tratta di predirre valori di immobili), valutare personalmente anche features meno importanti e/o valori anomali che influenzano il mercato immobiliare.
Infatti sarebbe possibile costruire altre features considerando le varie caratteristiche inerenti alla vendita di una casa e dunque al suo valore e prezzo.

Inerente alla selezione degli iperparametri, è stato effettuato il tuning possibile di ogni regressore, cercando di avere il miglior score possibile. La difficoltà è stata nella scelta degli iperparametri, dato che abbiamo eseguito il tuning su tutti i tre dataset creati per poi valutare il medium il dataset quello con lo score migliore da usare su tutti i regressori (per semplicità e velocità del codice), così facendo potremo aver scartato opzioni più valide.