UNIVERSITA' DEGLI STUDI DI CAGLIARI
INFORMATICA APPLICATA E DATA ANALYTICS
CORSO DI MACHINE LEARNING

# Progetto Corso di Machine Learning

Alice Zonca: 60/79/00013, Simone Cocco: 60/79/00024, Samuele Felice Corrias: 60/79/00090

## Indice

# Abstract

Nel 2023, la comunicazione interpersonale è ormai digitale. Nonostante gli articoli continuino ad essere stampati in giornali è riviste, la maggior parte delle notizie vengono lette via internet. In questo mondo digitale, la diffusione di una notizia avviene quindi per popolarità, ovvero per numero di condivisioni nei social media dell'articolo che contiene la notizia in questione. Risulta quindi necessario poter valutare la popolarità di una nuova notizia, per capire se questa verrà condivisa e si diffonderà oppure se resterà nell'ombra.

Per questo motivo, il task di questo progetto è quello di addestrare un modello di Machine Learning a predire il numero di condivisioni, e quindi la popolarità di una determinata notizia, e valutare così le sue prestazioni, per riuscire a trovare il modello ottimale.

# 1 - Introduzione

Il dataset per il task è fornito dal sito dell'Università di Irvine, in California, UCI Machine Learning Repository al link:

Online News Popularity

Esso contiene i dati di moltissimi articoli pubblicati sulla pagina internet Mashable ed è completo di etichette, ovvero il numero di condivisioni di quel determinato articolo sui social media. Esso è stato acquisito all'inizio del 2015, quindi contiene articoli vecchi di pochi anni.

Essendo un dataset con etichette con valori discreti, che risultano essere potenzialmente in un range compreso fra (0, +∞), non è possibile trattarlo come un task di classificazione, in quanto non si possono assegnare ai nuovi record, delle etichette specifiche. Si è scelto, quindi, di utilizzare dei modelli di regressione che possono meglio predire dei valori compresi in un range infinito.

Il codice implementato per la creazione, il pre-processing, l'addestramento e la valutazione dei modelli, viene quindi diviso in diversi file, i quali vengono poi importati nel file main.py.

Si utilizza inoltre un menù di selezione, che permette di semplificare la scelta delle tecniche, dei modelli e la visualizzazione dei risultati.

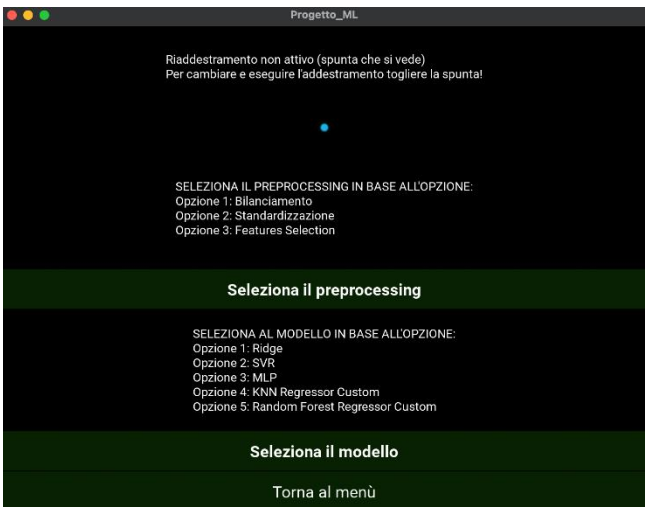
# 2 - Menù

Il menù viene implementato grazie all'utilizzo della libreria Kivy, che permette di scrivere codice in linguaggio Python e che contiene delle funzioni utili per la costruzione di un'applicazione desktop adatta all'interazione con l'utente.

Il menù è implementato nel file main.py, il quale si occupa solo di avviare l'applicazione.


Una volta lanciata l'esecuzione del file main.py, viene avviata la schermata iniziale del menù, contenente il titolo, le matricole dei membri del gruppo che hanno partecipato al progetto e tre tasti di selezione:

- Analisi dei dati: permette di eseguire l'analisi del dataset scaricato e stampare il risultato in un'altra finestra
- Modelli: permette di passare alla finestra di selezione della tecnica di pre-processing e del modello desiderati
- Miglior combinazione: permette di stampare la tabella dei valori delle metriche della miglior combinazione 'Tecnica di Pre-Processing + Modello'



Una volta selezionato il bottone ‘MODELLI’, l’applicazione apre una nuova finestra contenente un tasto di selezione e due tendine a scorrimento:

- Riaddestramento non attivo: se selezionato, carica semplicemente l’immagine dei valori delle metriche dell’addestramento effettuato in fase di test, per ogni combinazione scelta. Se non selezionato, riaddestra il modello con la combinazione scelta e stampa l’immagine dei valori delle metriche ottenuta
- Selezione del pre-processing: permette di selezionare la tecnica di pre-processing desiderata, fra le tre disponibili
- Selezione del modello: permette di selezionare il modello desiderato, fra i cinque disponibili

Una volta fatta una qualsiasi selezione, è presente, nella finestra successiva, un bottone che permette di ritornare alla pagina principale del menù, tranne che nelle immagini ottenute dalla selezione all’interno del menù ‘Modelli’, dove basta cliccare sull’immagine per tornare al menù iniziale.

# 3 - Analisi dei dati

Il dataset di riferimento per l'addestramento dei modelli risulta essere di grandi dimensioni, con 60 attributi più le etichette e 39797 record.

Si esegue quindi l'analisi del dataset, per la valutazione dei dati e degli attributi in esso contenuti.

```
Columns: Index(['url', ' timedelta', ' n_tokens_title', ' n_tokens_content',
       ' n_unique_tokens', ' n_non_stop_words', ' n_non_stop_unique_tokens',
       ' num_hrefs', ' num_self_hrefs', ' num_imgs', ' num_videos',
       ' average_token_length', ' num_keywords', ' data_channel_is_lifestyle',
       ' data_channel_is_entertainment', ' data_channel_is_bus',
       ' data_channel_is_socmed', ' data_channel_is_tech',
       ' data_channel_is_world', ' kw_min_min', ' kw_max_min', ' kw_avg_min',
       ' kw_min_max', ' kw_max_max', ' kw_avg_max', ' kw_min_avg',
       ' kw_max_avg', ' kw_avg_avg', ' self_reference_min_shares',
       ' self_reference_max_shares', ' self_reference_avg_sharess',
       ' weekday_is_monday', ' weekday_is_tuesday', ' weekday_is_wednesday',
       ' weekday_is_thursday', ' weekday_is_friday', ' weekday_is_saturday',
       ' weekday_is_sunday', ' is_weekend', ' LDA_00', ' LDA_01', ' LDA_02',
       ' LDA_03', ' LDA_04', ' global_subjectivity',
       ' global_sentiment_polarity', ' global_rate_positive_words',
       ' global_rate_negative_words', ' rate_positive_words',
       ' rate_negative_words', ' avg_positive_polarity',
       ' min_positive_polarity', ' max_positive_polarity',
       ' avg_negative_polarity', ' min_negative_polarity',
       ' max_negative_polarity', ' title_subjectivity',
       ' title_sentiment_polarity', ' abs_title_subjectivity',
       ' abs_title_sentiment_polarity', ' shares'],
      dtype='object')

Numero valori mancanti
Result: None

First six:     n_tokens_title   n_tokens_content   n_unique_tokens  ...   abs_title_subjectivity   abs_title_sentiment_polarity   shares
0              12.0              219.0          0.663594  ...                 0.000000                       0.187500              593
1               9.0              255.0          0.604743  ...                 0.500000                       0.000000              711
2               9.0              211.0          0.575130  ...                 0.500000                       0.000000             1500
3               9.0              531.0          0.503788  ...                 0.500000                       0.000000             1200
4              13.0             1072.0          0.415646  ...                 0.045455                       0.136364              505
5              10.0              370.0          0.559889  ...                 0.142857                       0.214286              855

[6 rows x 59 columns]

Last six:         n_tokens_title   n_tokens_content   n_unique_tokens  ...   abs_title_subjectivity   abs_title_sentiment_polarity   shares
39638              11.0              223.0          0.653153  ...                 0.500000                       0.000000             1200
39639              11.0              346.0          0.529052  ...                 0.400000                       0.000000             1800
39640              12.0              328.0          0.696296  ...                 0.200000                       1.000000             1900
39641              10.0              442.0          0.516355  ...                 0.045455                       0.136364             1900
39642               6.0              682.0          0.539493  ...                 0.500000                       0.000000             1100
39643              10.0              157.0          0.701987  ...                 0.166667                       0.250000             1300

[6 rows x 59 columns]

Data describe:         n_tokens_title   n_tokens_content   n_unique_tokens  ...   abs_title_subjectivity   abs_title_sentiment_polarity         shares
count     39644.000000       39644.000000       39644.000000  ...            39644.000000                   39644.000000   39644.000000
mean         10.398749         546.514731          0.548216  ...                0.341843                       0.156064    3395.380184
std           2.114037         471.107508          3.520708  ...                0.188791                       0.226294   11626.950749
min           2.000000           0.000000          0.000000  ...                0.000000                       0.000000       1.000000
25%           9.000000         246.000000          0.470870  ...                0.166667                       0.000000     946.000000
50%          10.000000         409.000000          0.539226  ...                0.500000                       0.000000    1400.000000
75%          12.000000         716.000000          0.608696  ...                0.500000                       0.250000    2800.000000
max          23.000000        8474.000000        701.000000  ...                0.500000                       1.000000  843300.000000

[8 rows x 59 columns]

Dim train: (29733, 58)

Dim test: (9911, 58)
```

Il dataset in questione contiene due attributi non utili per la predizione di nuove etichette: 'url' e 'timedelta', la prima che indica l'indirizzo url dell'articolo e la seconda che contiene il numero di giorni intercorsi tra la pubblicazione dell'articolo e il suo inserimento nel dataset. Si sceglie quindi di eliminare tali attributi, riducendo il numero di features, escluse le etichette, a 58.

Si nota inoltre, dal valore della variabile 'Result' che nessun attributo del dataset contiene valori mancanti. Tutti i valori degli attributi sono di tipo float64, mentre le etichette assumono valori di tipo int32.

Grazie alla funzione descrive() si possono visualizzare delle informazioni utili riguardo il dataset, nell'immagine mostrate sotto la dicitura 'Data describe'. Grazie alla media degli attributi, si capisce che molti valori sono compresi o uguali tra 0 e 1, quindi si suppone che essi siano attributi binari (0 o 1) memorizzati come float o valori all'interno del range. Dai nomi di attributi, infatti, si può vedere che alcune colonne contengono valori che rappresentano unicamente la presenza o meno di un determinato genere (lifestyle, entertainment, bus, socmed, tech e world) oppure se l'articolo è stato pubblicato o meno in un determinato giorno della settimana.

Si nota inoltre, che il 50esimo percentile corrisponde all'etichetta 1400, quindi si può considerare questo valore come uno split tra articoli popolari e non popolari.

Inoltre, ogni articolo viene condiviso almeno una volta, in quanto il minimo delle etichette è pari a 1. Questo vuol dire che nelle etichette non ci sono valori minori o uguali a 0.

La funzione per l'analisi dei dati riceve in input il dataset diviso in Train Set e Test Set, quindi si può vedere dalle loro dimensioni che il Test Set è composto dal 25% dei record del dataset originale.

# 4 - Implementazione

Per valutare la possibilità di utilizzare dei modelli di regressione di Machine Learning per il task assegnato insieme al dataset, si utilizzano tre tecniche di pre-processing, cinque modelli con relativo tuning degli iperparametri per tre di essi e due metriche.

## 4.1 - Tecniche di Pre-Processing

Come tecniche di pre-processing si è scelto di utilizzare:

- Bilanciamento
    - Undersampling Near Miss versione 2
    - Random Oversampling
- Standardizzazione
    - Standard Scaler
    - MinMax Scaler
- Features Selection
    - Mediante indici di correlazione
    - Con algoritmo PCA

### 4.1.1 - Bilanciamento

Data la grande quantità di record presenti nel dataset, con etichette tutte diverse fra loro, si è scelto di utilizzare la tecnica Undersampling Near Miss versione 2, fornito dalla libreria Scikit-learn, che elimina alcuni record, diminuendo la dimensione del dataset, preservando però i record più vicini ai più lontani. Questo perché si può supporre che articoli molto simili abbiano un range di interesse simile e quindi anche il numero delle condivisioni tende ad essere analogo. In questo modo, si diminuisce il numero dei record del dataset, dividendolo poi in Train Set e Test Set in modo da mantenere un Test Set con il 25% dei record del dataset bilanciato.

Si è tentato anche di utilizzare un bilanciamento mediante Oversampling, in quando le etichette dei record sono tutte diverse fra loro, quindi si è supposto che il regressore potesse avere più semplicità nell'apprendimento con un dataset con record più vicini. Si utilizza quindi un Random Oversampling, sempre fornito dalla libreria Scikit-learn per aumentare il numero di record del dataset.

Si è notato, però, che l'aumento delle prestazioni del modello avviene solo quando il bilanciamento con Oversampling produce una quantità grandissima di record (circa 2 milioni) e quindi questo tende a risultare troppo complesso, in quanto non ottimale come tempi di esecuzione.

### 4.1.2 - Standardizzazione

Si è scelto di utilizzare due tecniche diverse di standardizzazione, per verificare se, modificando i dati con due sistemi differenti, il regressore avrebbe potuto avere prestazioni migliori.

La prima tecnica di standardizzazione che si è scelto di utilizzare è lo Standard Scaler, un metodo che permette di impostare la media dei dati a 0 e la loro varianza a 1.

La seconda tecnica di standardizzazione utilizzata è il MinMax Scaler che permette di modificare il valore dei dati entro un certo range, solitamente [0, 1].

Purtroppo, si è notato che non ci sono variazioni significative rispetto all'addestramento dei modelli con il dataset originale.

### 4.1.3 - Features Selection

L'ultima tecnica di pre-processing che si è utilizzata per tentare di migliorare le prestazioni dei modelli di regressione è la Features Selection, che permette di ridurre il numero di attributi. Questo tipo di tecniche avrebbe potuto risultare efficace per migliorare la qualità del dataset, in quanto quest'ultimo ha un alto numero di features (58 più le etichette).

Si è quindi implementata una Feature Selection mediante valutazione degli indici di correlazione fra gli attributi. Tuttavia, gli attributi del dataset hanno una correlazione molto bassa, quindi si sono potute eliminare solo cinque features, senza ottenere miglioramenti dopo l'addestramento dei modelli.

È stata implementata anche una Features Selection mediante algoritmo PCA (Principal Component Analysis), che permette di eliminare alcune features tramite controllo sulla varianza dei dati. L'algoritmo ha permesso di eliminare un buon numero di attributi ma le prestazioni non solo non sono migliorate, ma in alcuni casi sono peggiorate.

Questo permette di capire che gli attributi del dataset potrebbero essere essenziali per permettere ai regressori di allenarsi correttamente e quindi aumentare la precisione di predizione.

## 4.2 - Ottimizzazione degli iperparametri

Prima di implementare e addestrare i modelli della libreria Scikit-learn, si è voluto controllare quali parametri delle classi dei regressori possono produrre il miglior risultato.

Si è quindi utilizzata la classe GridSearch per la valutazione dei parametri dei modelli Ridge, SVR e MLP.

```
Migliori risultati per Ridge:
Ridge(alpha=0.9)
```

Per il modello di regressione lineare Ridge si è scelto di ottimizzare solo il parametro alpha, che rappresenta il valore del termine di regolarizzazione della funzione di loss del regressore.

```
Migliori risultati per SVR:
SVR(C=100, epsilon=2, kernel='poly')
```

Per quanto riguarda il modello SVR, si è scelto di ottimizzare i parametri C, che rappresenta il valore del parametro di regolarizzazione, epsilon, che rappresenta il valore della penalità della funzione di loss e il kernel (tra rbg, poly e sigmoid).

```
Migliori risultati per MLP:
{'estimator': MLPClassifier(alpha=0.001, early_stopping=True, hidden_layer_sizes=(50, 30, 10),
              learning_rate_init=0.2, max_iter=100, random_state=0,
              solver='sgd'), 'estimator__alpha': 0.001, 'estimator__hidden_layer_sizes': (50, 30, 10), 'scaler': MinMaxScaler()}
```

L'ultimo modello per il quale si sono ottimizzati i parametri è il modello MLP. I parametri ottimizzati in questo caso sono il numero di step del modello, la dimensione di tali livelli, il valore di learnig_rate iniziale e la standardizzazione da utilizzare.

## 4.3 - Modelli

Come modelli di regressione si è scelto di utilizzare:

- Regressore lineare Ridge
- SVR (Support Vector Machine Regressor)
- MLP (Multi-layer Perceptron)
- kNN Regressor Custom
- Random Forest Regressor Custom

### 4.3.1 - Ridge

$$L(\boldsymbol{w}) = \frac{1}{2n}\|\mathbf{X}\boldsymbol{w} - \boldsymbol{y}\|^2 + \lambda\|\boldsymbol{w}\|^2$$

Il modello di regressione lineare Ridge utilizza il Mean Squared Error e la norma l2 per minimizzare la funzione di loss.

Il modello è stato implementato a partire dalla classe Ridge() della libreria Scikit-learn. Avendo eseguito l'ottimizzazione degli iperparametri, è stato impostato alpha=0.9. Successivamente il modello stato addestrato sia sul dataset originale che sui dati preprocessati.

**Modello Ridge con e senza tecniche di bilanciamento**

| Metrica\Tecnica | Non Bilanciato | Undersampling NearMiss | Random Oversampling |
|---|---|---|---|
| R2 Score | -0.13 | 0.17 | 0.14 |
| MSE | 81691038.0 | 4090807500.0 | 2527960832.0 |

**Modello Ridge con e senza tecniche di standardizzazione**

| Metrica\Tecnica | Non Bilanciato | Standard Scaler | MinMax Scaler |
|---|---|---|---|
| R2 Score | -0.13 | -0.13 | -0.13 |
| MSE | 81691038.0 | 81691038.0 | 81691038.0 |

**Modello Ridge con e senza tecniche di feature selection**

| Metrica\Tecnica | Non Bilanciato | Sel. su indici di corr. | PCA |
|---|---|---|---|
| R2 Score | -0.13 | 0.02 | -0.92 |
| MSE | 81691038.0 | 71167226.0 | 138763110.0 |

Il modello risulta avere prestazioni migliori con dei dati preprocessati tramite Undersampling Near Miss.

### 4.3.2 - SVR (Support Vector Machine Regressor)

$$y = \sum_{i=1}^{N} (\alpha_i - \alpha_i^*) \cdot K(x_i, x) + b$$

Il SVR utilizza una funzione, lineare o non a seconda del kernel, per dividere lo spazio delle features e così predire le etichette per i nuovi record. Il modello utilizzato è un SVR non lineare in quanto il kernel utilizzato è 'poly' perché restituito dall'ottimizzazione degli iperparametri.

**Modello SVR con e senza tecnica di undersampling**

| Metrica\Tecnica | Non Bilanciato | Undersampling NearMiss |
|---|---|---|
| R2 Score | -0.04 | 0.08 |
| MSE | 75400601.0 | 4550125111.0 |

**Modello SVR con e senza tecniche di standardizzazione**

| Metrica\Tecnica | Non Bilanciato | Standard Scaler | MinMax Scaler |
|---|---|---|---|
| R2 Score | -0.04 | -0.04 | -0.04 |
| MSE | 75400601.0 | 75400601.0 | 75400601.0 |

**Modello SVR con e senza tecniche di feature selection**

| Metrica\Tecnica | Non Bilanciato | Sel. su indici di corr. | PCA |
|---|---|---|---|
| R2 Score | -0.04 | -0.04 | -0.05 |
| MSE | 75400601.0 | 75400756.0 | 75722200.0 |

Il modello SVR risulta avere prestazioni migliori eseguendo l'addestramento con dataset bilanciato mediante Undersampling Near Miss. Tuttavia, il modello non sembra ottimale per questo tipo di dataset.

### 4.3.3 - MLP (Multi-layer Perceptron)

Il Multi-layer Perceptron è una rete neurale fortemente connessa, con un numero di livelli interni definiti dai parametri della classe. Ogni neurone (percettrone) di un determinato livello della rete è connesso a tutti i percettroni dei livelli immediatamente precedenti e successivi. Ad ogni step di addestramento vengono aggiornati i persi e si calcola la funzione di loss, che produrre un valore di loss il più piccolo possibile.

Nel modello implementato, l'addestramento si blocca se non ci sono grandi variazioni della loss e delle metriche del Validation Set per almeno 10 step. Inoltre, si utilizza la funzione di attivazione dei percettroni sigmoid, che restituisce dei valori nel range (0, 1).

**Modello MLP con e senza tecniche di bilanciamento**

| Metrica\Tecnica | Non Bilanciato | Undersampling NearMiss | Random Oversampling |
|---|---|---|---|
| R2 Score | 0.0 | 0.0 | 0.0 |
| MSE | 77221476.0 | 5607866511.0 | 2932084798.0 |

**Modello MLP con e senza tecniche di standardizzazione**

| Metrica\Tecnica | Non Bilanciato | Standard Scaler | MinMax Scaler |
|---|---|---|---|
| R2 Score | 0.0 | 0.0 | 0.0 |
| MSE | 77221476.0 | 77221476.0 | 77221476.0 |

**Modello MLP con e senza tecniche di feature selection**

| Metrica\Tecnica | Non Bilanciato | Sel. su indici di corr. | PCA |
|---|---|---|---|
| R2 Score | 0.0 | 0.0 | 0.0 |
| MSE | 77221476.0 | 77221476.0 | 77221476.0 |

Le prestazioni del modello non variano anche con dati preprocessati e non sono ottimali rispetti ai modelli visti precedentemente.

### 4.3.4 - kNN Regressor Custom

Il quarto modello implementato è un modello di regressione kNN custom. Per la definizione della classe e dei suoi parametri si sono considerati il numero dei record più vicini da valutare per la distanza e la funzione della distanza da utilizzare. Nella classe è possibile calcolare la distanza fra due record mediante distanza Euclidea e distanza di Manhattan.

```
DISTANZA EUCLIDEA
p=1 (-0.20923319195547574, 87606705.4497306)
DISTANZA DI MANATTHAN
p=2 (-0.23103193955074564, 89185984.34519625)
```

Metriche: $R^2$ e MSE

Si è però notato che le prestazioni risultano migliori all'aumentare dei vicini considerati e con l'utilizzo della distanza Euclidea per il calcolo.

La classe contiene anche una funzione fit() per l'addestramento del modello sul Train Set e una funzione predict() per la predizione delle etichette di ogni record del Test Set.

**Modello KNN Regressor Custom con e senza tecnica di undesampling**

| Metrica\Tecnica | Non Bilanciato | Undersampling NearMiss |
|---|---|---|
| R2 Score | -0.07 | 0.11 |
| MSE | 77698043.0 | 4424330939.0 |

**Modello KNN Regressor Custom con e senza tecniche di standardizzazione**

| Metrica\Tecnica | Non Bilanciato | Standard Scaler | MinMax Scaler |
|---|---|---|---|
| R2 Score | -0.07 | -0.07 | -0.07 |
| MSE | 77698043.0 | 77698043.0 | 77698043.0 |

**Modello KNN Regressor Custom con e senza tecniche di feature selection**

| Metrica\Tecnica | Non Bilanciato | Sel. su indici di corr. | PCA |
|---|---|---|---|
| R2 Score | -0.07 | -0.07 | -0.07 |
| MSE | 77698043.0 | 77695909.0 | 77698043.0 |

Il modello così costruito risulta avere buoni risultati con un dataset bilanciato mediante Undersampling Near Miss.

### 4.3.5 - Random Forest Regressor Custom

L'ultimo modello implementato è un regressore multiplo custom. Viene definita la classe Ensemble che riceve i pesi, nel caso in cui esistano. Questo regressore multiplo è stato composto mediante diversi modelli DecisionRegressorTree(). Sono stati utilizzati tre Alberi di Regressione in quanto si è notato che il modello impiegava troppo tempo per l'addestramento.

La classe contiene una funzione fit che addestra il modello sul Train Set e permette l'addestramento di ogni Albero di Regressione solo una parte del Train Set. Contiene inoltre una funzione predict() che, per ogni record del Test Set, predice un'etichetta sulla base della media delle predizioni di ogni regressore singolo che compone il modello.

**Modello Random Forest Regressor Custom con e senza tecniche di bilanciamento**

| Metrica\Tecnica | Non Bilanciato | Undersampling NearMiss | Random Oversampling |
|---|---|---|---|
| R2 Score | -0.02 | -0.07 | -0.07 |
| MSE | 74016573.0 | 5283215048.0 | 3126280811.0 |

**Modello Random Forest Regressor Custom con e senza tecniche di standardizzazione**

| Metrica\Tecnica | Non Bilanciato | Standard Scaler | MinMax Scaler |
|---|---|---|---|
| R2 Score | -0.02 | -0.02 | -0.02 |
| MSE | 73984811.0 | 73534996.0 | 73725070.0 |

**Modello Random Forest Regressor Custom con e senza tecniche di feature selection**

| Metrica\Tecnica | Non Bilanciato | Sel. su indici di corr. | PCA |
|---|---|---|---|
| R2 Score | -0.02 | -0.02 | -0.01 |
| MSE | 73994294.0 | 73578964.0 | 73152439.0 |

Il modello non risulta avere buone prestazioni. Si può ipotizzare con un l'aumento dei regressori singoli che lo compongono, potrebbe migliorare le predizioni, tuttavia risulta troppo complesso per l'esecuzione.

## 4.4 - Metriche

Per la valutazione del modello si sono utilizzate due metriche per i regressori:

- $R^2$ Score
- MSE (Mean Squared Error)

### 4.4.1 - $R^2$ Score

$$R2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})},$$

Viene usata per valutare quanto variano le predizioni del modello sul Test Set rispetto alle etichette reali. Maggiore è il valore della metrica, più i due valori sono correlati.

### 4.4.2 - MSE

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

Viene utilizzata per valutare la media della radice quadrata dell'errore della predizione sul Test Set. Più è alta, più la predizione si discosta dal valore reale dell'etichetta.

## Conclusioni

Dai test effettuati, la miglior combinazione tra tecniche di pre-processing e modelli è quella del modello Ridge addestrato su un dataset bilanciato mediante Undesampling Near Miss.

**Combinazione migliore: Undersampling + Ridge**

| Combinazione\Metrica | R2 Score | MSE |
|---|---|---|
| Undersampling + Ridge | 0.17 | 4090807500.0 |

Tuttavia il valore di $R^2$ Score risulta troppo basso, in quanto il suo limite massimo è 1, mentre il valore del MSE risulta comunque troppo alto.

Si suppone, quindi, che questo tipo di dataset possa essere trattato meglio attraverso l'implementazione di reti neurali più complesse.