



Valuta la tua casa

De Giglio Pietro - Mat: 779409
p.degiglio5@studenti.uniba.it
Franco Andrea - Mat: 774248
a.franco56@studenti.uniba.it

A.A. 2024/25

INDICE DEI CONTENUTI

01. INTRODUZIONE

02. INTRODUZIONE ALLE SCELTE PROGETTUALI

03. SCELTE PROGETTUALI

04. GUIDA ALL'UTILIZZO

01. INTRODUZIONE

Valuta la tua casa è un progetto software che si pone come obiettivo la predizione dei prezzi delle case, in questo caso specifico, visto il dataset utilizzato, di quelle americane. Date in input le caratteristiche della casa, il programma fornisce una predizione del prezzo e una predizione della probabilità di finire in un dato range di prezzo, tramite forme di apprendimento supervisionato e non supervisionato, e che il prezzo sia presente in una fascia di prezzi tramite apprendimento probabilistico.





0.2 INTRODUZIONE ALLE SCELTE PROGETTUALI

Sviluppato in Python 3.11, abbiamo utilizzato le seguenti librerie



Pandas

Gestisce e analizza dati strutturati (come tavelle). Permette manipolazioni avanzate di dati con DataFrame e Series.



Numpy

Fornisce array multidimensionali e operazioni efficienti su di essi, essenziale per calcoli numerici ad alte prestazioni.



Scikit-learn (Sklearn)

Implementa algoritmi di machine learning e strumenti per analisi di dati (classificazione, regressione, clustering).



Matplotlib

Crea grafici e visualizzazioni personalizzabili per dati (2D e 3D).



Tqdm

Mostra barre di progresso per cicli e processi iterativi, migliorando la comprensione del tempo rimanente.



Tkinter

Crea interfacce grafiche (GUI) per applicazioni desktop Python.

DATASET

Si è utilizzato un dataset estrapolato dal sito kaggle.com, chiamato **“data.csv”**, il file contiene i campi **‘date’** (data di registrazione), **‘price’** (prezzo), **‘bedrooms’** (numero camere da letto), **‘bathrooms’** (numero bagni), **‘sqft_living’** (piedi quadrati vivibili), **‘sqft_lot’** (piedi quadrati totali), **‘floors’** (piani), **‘waterfront’** (facciata su mare), **‘condition’** (condizioni), **‘sqft_above’** (piedi quadrati calpestabili), **‘sqft_basement’** (piedi quadrati seminterrato), **‘yr_builtin’** (anno di costruzione), **‘yr_renovated’** (anno di ristrutturazione), **‘street’** (via), **‘city’** (città), **‘statezip’** (CAP), **‘country’** (nazione). Al dataset sono stati uniti i 2 campi, **‘bedrooms’** e **‘bathrooms’** in un unico campo **‘rooms’** (numero delle stanze), per diminuire il numero delle features e incrementare il peso delle stanze a livello di predizione.



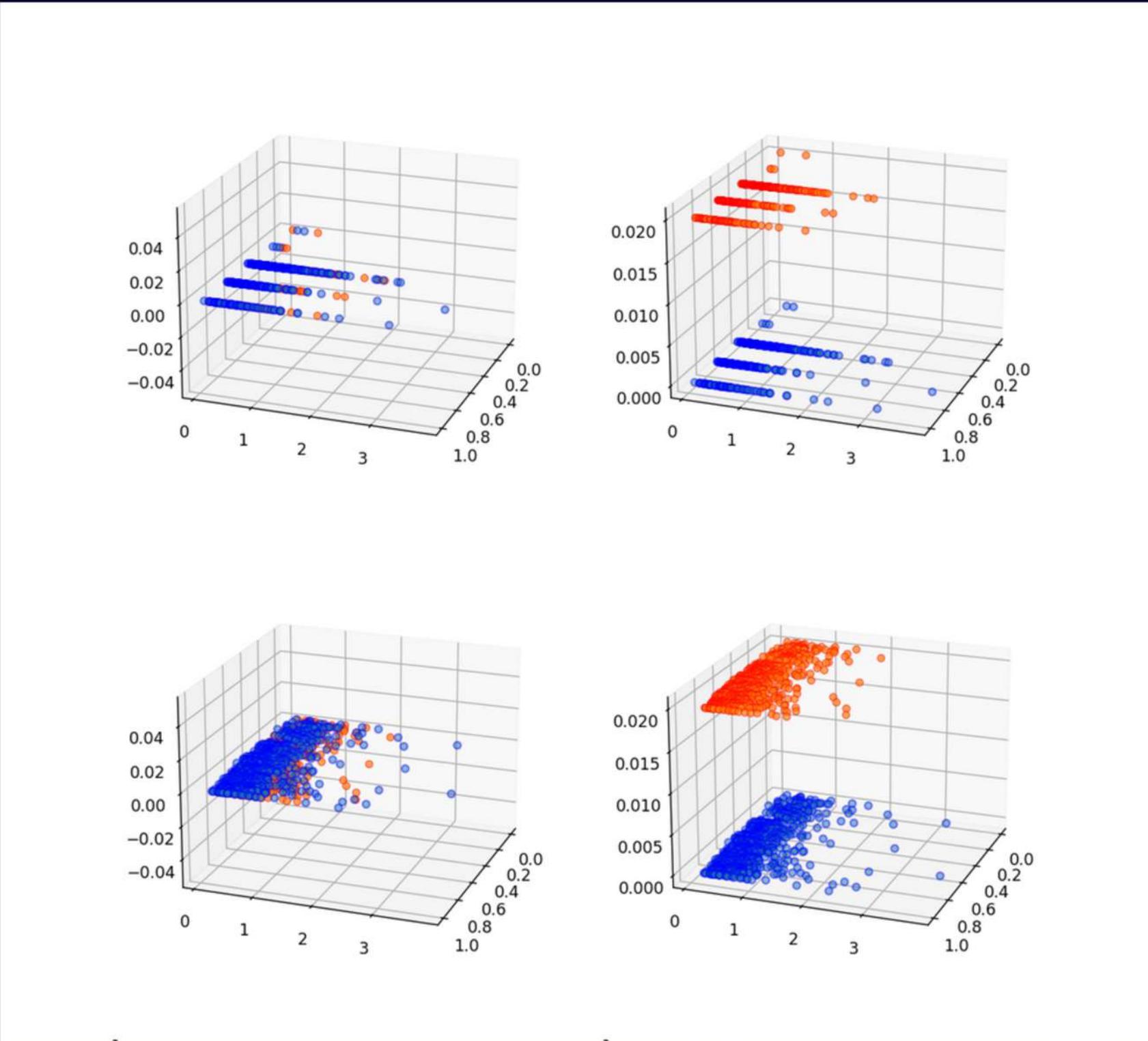
COME FUNZIONA

INGEGNERIA DELLA CONOSCENZA A.A.2024/25

Il software genera un file “**house.csv**” che contiene il dataset risultante dalle operazioni appena descritte. Le istruzioni che creano tale file **sono commentate**, in quanto **non più utili** una volta che si crea il file. Tornerebbero utili in caso di aggiornamenti dei dataset utilizzati. Per avere **dati più accurati** abbiamo deciso di **eliminare i duplicati** e i **campi con i valori nulli** dal csv, inoltre abbiamo deciso di **convertire i ‘piedi quadri’ in ‘metri quadri’** poiché il sistema metrico decimale in Italia è quello dei metri. Per ottimizzare la predizione del prezzo abbiamo scalato i valori del dataframe tra 0 e 1 e prima dell’**apprendimento supervisionato**, viene utilizzato l’**apprendimento non supervisionato** con operazioni di **clustering**, inoltre, in aggiunta viene utilizzata l’**apprendimento probabilistico** per predire la probabilità che il prezzo di una determinata casa possa essere presente in una determinata fascia di prezzo. I **modelli** (di tipo **random forest regressor**) utilizzano come **training-set** una porzione **(80%)** del dataset, l’altra porzione **(20%)** è usata come **test-set**. La scelta di non utilizzare tutta la base di conoscenza come training-set è atta ad **evitare l’overfitting** e a **valutare i modelli** appresi utilizzando dati (test-set) non utilizzati per l’apprendimento. Invece per l’**apprendimento probabilistico** viene utilizzato il **modello SGD** (Stochastic Gradient Descent) **Classifier** per la sua accuratezza e affidabilità migliore. Successivamente vi è l’utilizzo del **K-fold**, per suddividere il training set in k fold e addestrare/valutare il modello su ogni fold, nei 2 modelli di Grid Search utilizzati appunto per vedere quali sono le variabili migliori da utilizzare per entrambi i modelli. La **valutazione** della qualità **del modello** del Random Forrest si basa sul calcolo del punteggio medio **MAE (Mean Absolute Error ‘110438.91’)** calcolato tra i prezzi predetti per le case del test-set e quelli reali, sul **MSE (Mean Square Error ‘34133152382.54’)** calcolato sul test-set e sul calcolo del **r2-score** (**‘0.75’ = 75%**) in grado di calcolarci la percentuale di accuratezza delle predizioni. Inoltre, per la valutazione della qualità del modello probabilistico è basato sulla percentuale dell’**accuracy del SGD Classifier (75,28%)**. Le features selezionate sono i campi **‘sqft_living’, ‘sqft_lot’, ‘floors’, ‘waterfront’, ‘view’, ‘condition’, ‘sqft_above’, ‘sqft_basement’, ‘yr_built’, ‘yr_renovated’, ‘street’, ‘city’, ‘country’, ‘rooms’**, i campi **‘date’ e ‘statezip’** si sono mostrati irrilevanti per la qualità della predizione di Price.

ACCURACY

Per mostrare l'accuratezza della predizione per **ogni singolo feature** abbiamo inserito la stampa di grafici che dimostrano tale accuratezza (in blu i valori reali delle features e in rosso la loro predizione), le istruzioni che creano tale grafico sono commentate, in quanto non più utili una volta che si crea il grafico, di seguito qualche esempio rispettivamente delle seguenti features (**sqft_living, floors**):



03. SCELTE PROGETTUALI

Scelta di “IndexState = 2167”

Grazie l'utilizzo di un ciclo che partiva da un indexstate = 0 e finiva a un indexstate = 10.000 abbiamo cercato l'indexstate che ci restituisse il valore migliore a livello di predizione nel RandomForest.

Scelta dei range da utilizzare nell'apprendimento probabilistico

Sono stati utilizzati le fasce di prezzo presenti nel codice a discrezione nostra dopo aver letto tutto il file house.csv e aver visto quali sono i prezzi più ricorrenti.

Uso del get_dummies e del LabelEncoder

Il get_dummies viene utilizzato sulla feature "city" per la creazione di nuove colonne dove ogni colonna rappresenterà ogni singola città e sarà valorizzata con 0 o 1 in base al valore effettivo di quella città (es. "city_Seattle=1" | "tutte le altre città=0"); invece, LabelEncoder viene utilizzato sulle features "street, country" così da riuscire a trasformare le stringhe in valori labels (es. "street=18810 Densmore Ave N" => "street = 42"), in modo tale che il modello operi solo su numeri piuttosto che su stringhe.

03. SCELTE PROGETTUALI

INGEGNERIA DELLA CONOSCENZA A.A.2024/25

Normalizzazione su tutte le features tranne "price"

"price" non viene normalizzata poiché ci servirà il valore numerico intero per effettuare le varie predizioni, essa verrà discostata dal nostro modello essendo il nostro target.

Train/Test split

La scelta di splittare il training set e il test set con la variabile "test_size=0.2" è stata fatta dopo svariati tentativi e sapendo che è idealmente più corretto riservare 2/3 al training set e 1/3 al test set.

Train/Test split

La scelta del SGD Classifier è stata fatta dopo aver tentato di effettuare un apprendimento probabilistico tramite il Gaussian model e il Multinomial model ed aver avuto risultati poco attendibili poiché, il primo è un ottimo classificatore, ma non un buon stimatore di probabilità e invece il secondo ci dava semplicemente un Accuracy inferiore.

Apprendimento non supervisionato

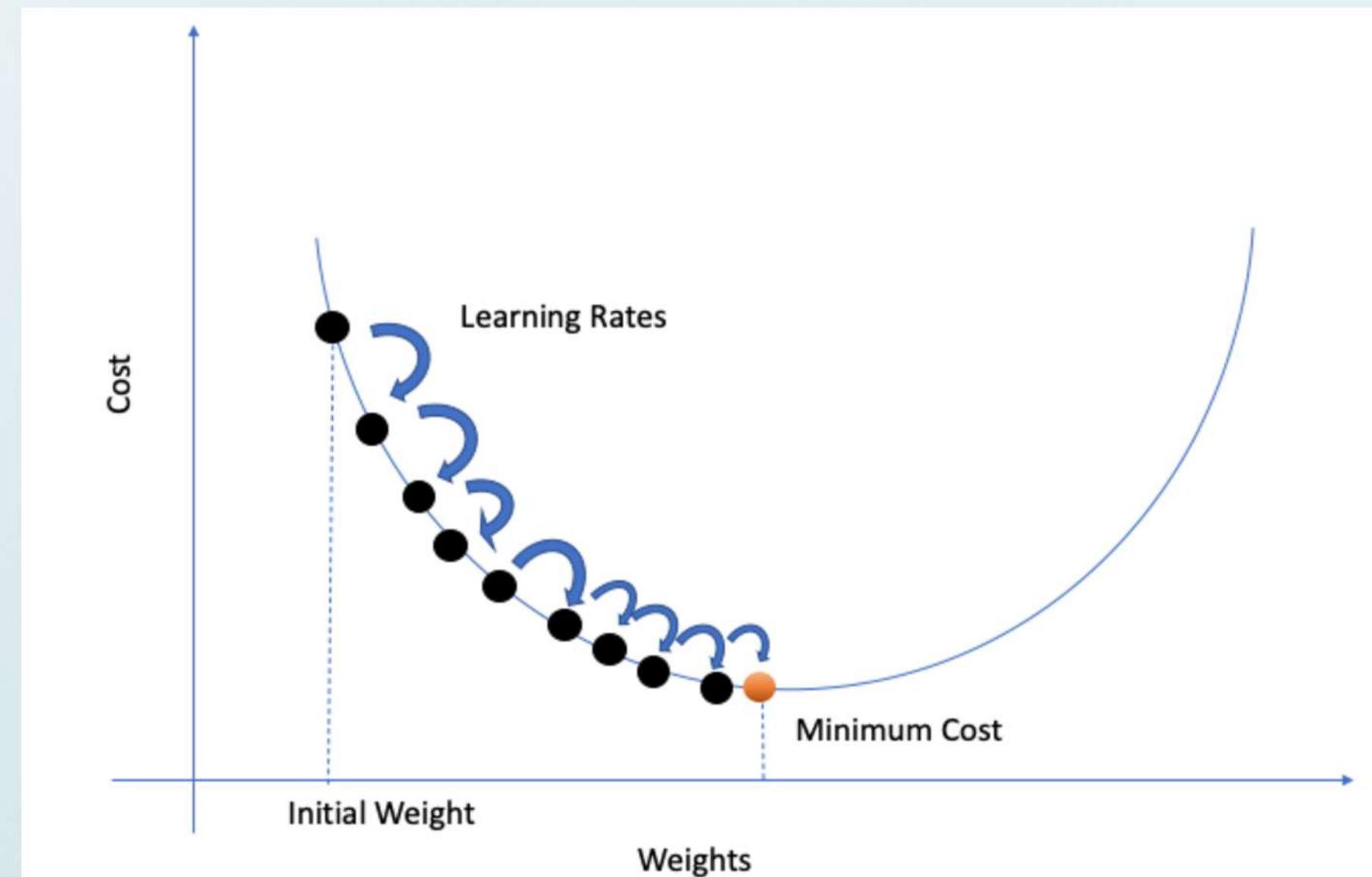
DBSCAN è un algoritmo di clustering che suddivide i dati in gruppi sulla base della loro densità, identificando punti densi come nuclei dei cluster e punti meno densi come rumore o outliers. Qui viene creato un oggetto "clusters" utilizzando l'algoritmo DBSCAN con i parametri `eps=0.9` (raggio della vicinanza) e `min_samples=3` (numero minimo di punti nel raggio per formare un cluster), i valori sono quelli di default. L'oggetto "clusters" viene addestrato sui dati di "prices_x" e dopo l'addestramento con DBSCAN vi è la restituzione dell'etichetta di clustering assegnata a ciascuna istanza dei dati in "prices_x". Successivamente viene aggiunta una nuova colonna chiamata "noise" al dataframe "prices_x" e al dataframe "prices_y" e vengono assegnate le etichette di clustering ottenute dall'algoritmo DBSCAN. Vengono selezionate solo le righe che hanno un valore di "noise" maggiore di -1. Questa operazione filtra le righe che non sono state assegnate a nessun cluster, quindi rimuove il rumore. E infine, La colonna "noise" viene rimossa dal dataframe "prices_x" utilizzando il metodo "drop". Riguardo al clustering, DBSCAN è un algoritmo di clustering "soft" perché assegna i punti a cluster o al rumore.

ARCHITETTURA:

INGEGNERIA DELLA CONOSCENZA A.A.2024/25

Lo SGDClassifier si basa sull'algoritmo di ottimizzazione Stochastic Gradient Descent (SGD). Durante l'addestramento, esso regola iterativamente i parametri del modello (pesi e bias) usando mini-batch di dati di addestramento (Dataset) per minimizzare una funzione di perdita. Questo processo mira a trovare i parametri ottimali e a minimizzare il costo finale (Loss). L'obiettivo dell'SGD è raggiungere il punto più basso, rappresentato da un pallino arancione nel grafico iniziando da un punto (peso) totalmente randomico (Initial Weight). Successivamente, tramite un parametro denominato Learning Rate, questi pesi si spostano verso la ricerca del minimo globale. Questo parametro (rappresentato dalle frecce blu) è di vitale importanza, dato che se impostato ad un valore troppo alto, rischia di farci saltare completamente il punto più basso del costo, mentre se impostato ad un valore troppo basso, rischia di non farci mai raggiungere quel punto, muovendoci troppo lentamente.

L'SGDClassifier, di natura, applica come modello base o l'SVM (Support Vector Machine) o la Logistic Regression. Abbiamo utilizzato il secondo modello, dato che prevedeva l'applicazione di una funzione di decisione basata su probabilità.



Funzione di decisione



Il classificatore calcola una funzione di decisione per ogni istanza nel dataset (sample). Questa funzione è una combinazione lineare dei valori delle features e dei loro pesi corrispondenti (weights).

Matematicamente, per un problema di classificazione multiclass, può essere rappresentata come: $w_1 * x_1 + w_2 * x_2 + \dots + w_n * x_n + b_i$ dove w_1, w_2, \dots, w_n sono i pesi, x_1, x_2, \dots, x_n sono i valori delle varie features e b_i è il bias di quella specifica classe.

Calcolo delle probabilità di classe:



Le probabilità di classe vengono calcolate utilizzando la funzione di decisione, citata pocanzi. Queste probabilità vengono ottenute trasformando la funzione di decisione in un valore di probabilità usando la funzione di attivazione chiamata Softmax (data la classificazione multiclass) che ci restituisce un valore compreso tra -1 e 1 che può essere interpretato come la probabilità del sample di appartenere a quella specifica classe. La funzione softmax è definita come:

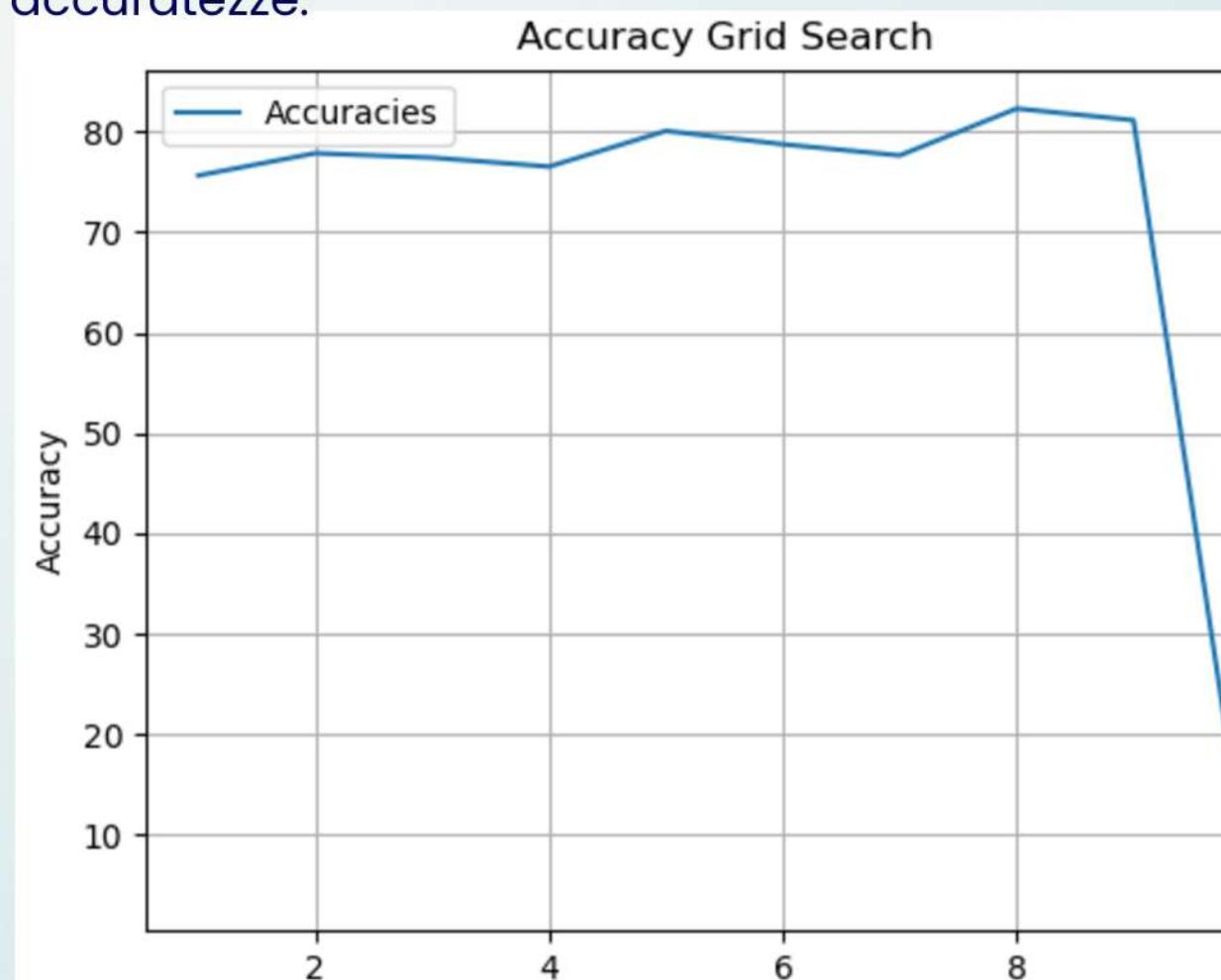
$$P(y=i|x) = \exp(score_i) / \sum(\exp(score_j) \text{ for } j \text{ in classes})$$

Dove:

- $P(y=i|x)$ è la probabilità che l'istanza appartenga alla classe i dato l'input x .
- $\exp(x)$ è la funzione esponenziale.
- $score_i$ è il punteggio (funzione di decisione) per la classe i .
- $classes$ rappresenta l'insieme di tutte le classi.

Uso di predict_proba()

Quando si chiama il metodo `predict_proba()` su un sample o un insieme di sample, il classificatore calcola le probabilità di classe in base ai loro valori delle feature e ai parametri del modello appresi (pesi). Il metodo restituisce un array in cui ogni riga corrisponde a un'istanza, e ogni colonna rappresenta le probabilità (tra -1 e 1) che quel sample appartenga a quella specifica classe. Nel nostro caso, avremmo 6 colonne, date le nostre 6 classi. Inoltre, nel SGD è stato scelto il valore dello split n.8 "alpha=0.0001" dopo aver cercato tramite il grid search il miglior parametro da utilizzare. Di seguito il grafico indica il punto migliore che ci indica l'accuratezza migliore riguardante i valori da utilizzare e, in seguito, la stampa su terminale dei vari parametri con i vari split e accuratezze:



INGEGNERIA DELLA CONOSCENZA A.A.2024/25

```
Fitting 10 folds for each of 10 candidates, totalling 100 fits
Split n.1 | Best hyper-parameters per SGD: {'alpha': 0.0001, 'fit_intercept': True}
Accuracy split n.1: 75.61%
Fitting 10 folds for each of 10 candidates, totalling 100 fits
Split n.2 | Best hyper-parameters per SGD: {'alpha': 0.0001, 'fit_intercept': True}
Accuracy split n.2: 77.83%
Fitting 10 folds for each of 10 candidates, totalling 100 fits
Split n.3 | Best hyper-parameters per SGD: {'alpha': 0.0001, 'fit_intercept': False}
Accuracy split n.3: 77.38%
Fitting 10 folds for each of 10 candidates, totalling 100 fits
Split n.4 | Best hyper-parameters per SGD: {'alpha': 0.0001, 'fit_intercept': True}
Accuracy split n.4: 76.50%
Fitting 10 folds for each of 10 candidates, totalling 100 fits
Split n.5 | Best hyper-parameters per SGD: {'alpha': 0.0001, 'fit_intercept': True}
Accuracy split n.5: 80.04%
Fitting 10 folds for each of 10 candidates, totalling 100 fits
Split n.6 | Best hyper-parameters per SGD: {'alpha': 0.0001, 'fit_intercept': True}
Accuracy split n.6: 78.71%
Fitting 10 folds for each of 10 candidates, totalling 100 fits
Split n.7 | Best hyper-parameters per SGD: {'alpha': 0.0001, 'fit_intercept': True}
Accuracy split n.7: 77.61%
Fitting 10 folds for each of 10 candidates, totalling 100 fits
Split n.8 | Best hyper-parameters per SGD: {'alpha': 0.0001, 'fit_intercept': True}
Accuracy split n.8: 82.26%
Fitting 10 folds for each of 10 candidates, totalling 100 fits
Split n.9 | Best hyper-parameters per SGD: {'alpha': 0.0001, 'fit_intercept': True}
Accuracy split n.9: 81.11%
Fitting 10 folds for each of 10 candidates, totalling 100 fits
Split n.10 | Best hyper-parameters per SGD: {'alpha': 0.0001, 'fit_intercept': True}
Accuracy split n.10: 4.44%
```

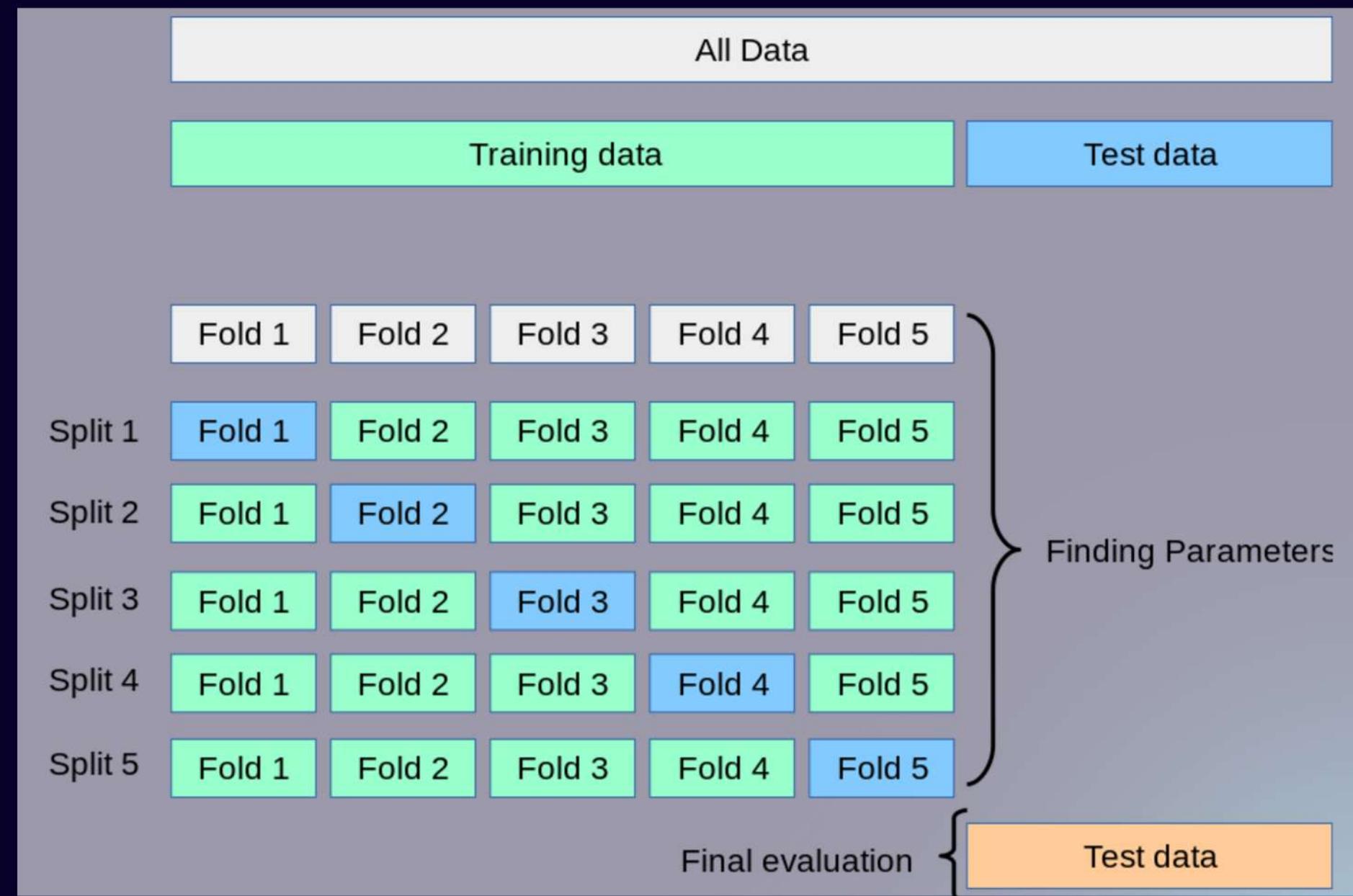
Grid search

INGEGNERIA DELLA CONOSCENZA A.A.2024/25

Nei grid search utilizzati sia per il SGD che per il RandomForest è stato scelto un KFold con 10 split poiché, nonostante di default sono impostati 5 split, abbiamo ritenuto più efficiente dividerlo in più Fold così da avere più precisione. Nello stesso algoritmo abbiamo impostato i valori sempre a nostra discrezione dopo aver letto le documentazioni della funzione. Di seguito un esempio di suddivisione in k-fold.

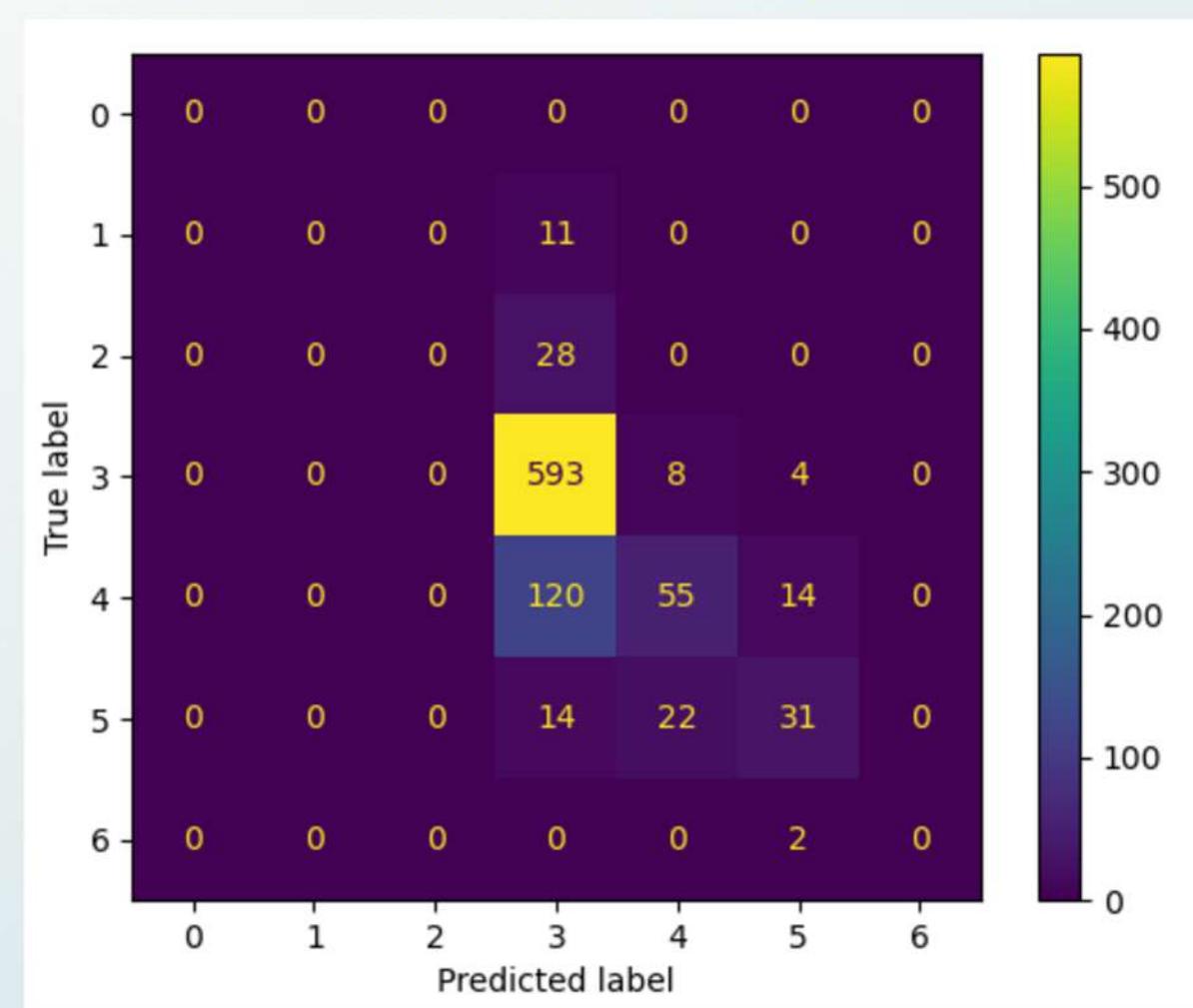
Come funziona il grid search?

Partendo dal training set si divide quest'ultimo in k parti (fold) e, analogamente, in k splits. Per ogni split viene utilizzato una parte di fold diversa come test set, in modo tale da trovare i migliori parametri per il modello. Infine, una volta trovati questi parametri, si procede alla classificazione con il test set originale. Per quanto riguarda la Grid Search sul SGD è stato impostato un valore di 'alpha':np.arange(0.0001,0.0006,0.0001) dove viene aumentato di 0.0001, per ogni split, il valore di alpha così da effettuare 100 fits; per quanto riguarda il fit intercept lo abbiamo impostato come una lista [True, False] in modo tale da provare il valore True e False per ogni valore di alpha. La stessa cosa è stata applicata per il Grid Search del RandomForest con valori diversi ma con la stessa metodologia.



Accuracy del SGD

Per verificare l'accuratezza del SGD lo abbiamo allenato sul traing set e abbiamo lanciato l'algoritmo che ci restituisse le probabilità sul test set. Dato che i valori Groundtruth del test set corrispondevano ad un determinato range (es. "1,3,3,2,1,5,4"), abbiamo ricavato l'indice della probabilità maggiore (np.argmax) sulle predizioni di cui abbiamo parlato all'inizio. Infine, tramite un semplice confronto abbiamo controllato se gli indici predetti corrispondessero ai valori Groundtruth e popolato, di conseguenza, la variabile "correct". Per calcolarci l'accuracy finale del modello abbiamo diviso correct per il numero totale di sample presenti nel test set ($((\text{correct}/\text{prices_y_test.size})*100)$). Inoltre, abbiamo inserito una stampa del Classification Report con i valori di Precision($\text{tp}/(\text{tp}+\text{fp})$), Recall($\text{tp}/(\text{tp}+\text{fn})$), F1-score ($2 * (\text{precision} * \text{recall}) / (\text{precision} + \text{recall})$), Support (sample per ogni classe) calcolati per ogni classe.



INGEGNERIA DELLA CONOSCENZA A.A.2024/25

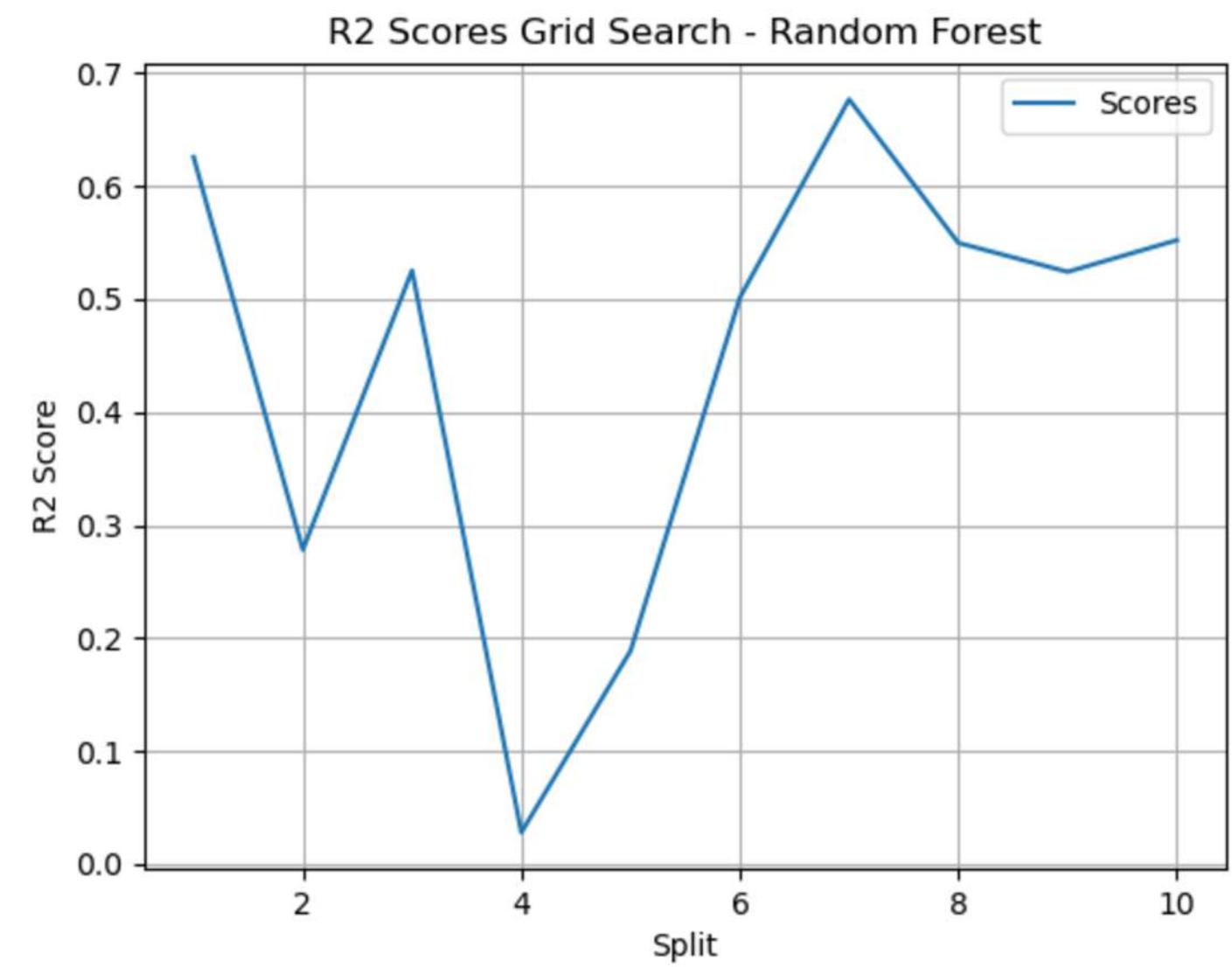
Classification report for SGD:

	precision	recall	f1-score	support
0	0.000	0.000	0.000	0
1	0.000	0.000	0.000	11
2	0.000	0.000	0.000	28
3	0.774	0.980	0.865	605
4	0.647	0.291	0.401	189
5	0.608	0.463	0.525	67
6	0.000	0.000	0.000	2

Come tesi a supporto della Classification report abbiamo inserito la stampa del Confusion matrix che ci mostra quanti sample sono stati catalogati in quale classe, ad es. si può notare come 11 sample della classe 1 vengono classificati nella classe 3, oppure 14 sample della classe 4 vengono classificati come classe 5.

Random Forest

Dopo aver provato vari modelli come il KNN o il LinearRegressor abbiamo accorto che il ForestModel era quello che più facesse al caso nostro, e dopo aver applicato anche su di esso il grid search abbiamo trovato i migliori parametri da utilizzare per la sua implementazione. Come mostrato dal grafico seguente i migliori parametri coincidevano con l'R2-score corrispondente con lo split n.7:



04. GUIDA ALL'UTILIZZO

INGEGNERIA DELLA CONOSCENZA A.A.2024/25

COMANDI DI ESECUZIONE

E' possibile eseguire il programma attraverso un qualsiasi IDE che supporti Python, in particolare i membri del gruppo hanno utilizzato Visual Studio Code. Per lanciare il programma è sufficiente fare il run del file 'ValutaCasa.py'.

Se si vuole eseguire il file 'ValutaCasa.py' dal terminale, occorre scaricare la cartella del progetto da GitHub e successivamente digitare sul terminale i seguenti comandi:

1. cd *percorso_cartella/ICON_24-25_De_Giglio-Franco*
2. python ValutaCasa.py

N.B.:

Le librerie utilizzate nell'implementazione sono le seguenti: pandas, Numpy, scikit-learn (sklearn), Matplotlib, tqdm, tkinter. Se le librerie non sono installate sulla propria macchina, prima di eseguire il programma occorre installarle dal prompt dei comandi tramite il seguente comando:
pip install nome_libreria.

Avvio dell'app

INGEGNERIA DELLA CONOSCENZA A.A.2024/25

Dopo aver lanciato il programma, si aprirà la schermata sulla destra.

L'applicazione offre 14 parametri , che si possono dividere in due grandi categorie, per poter selezionare le caratteristiche della vostra casa ideale.

Dacci qualche informazione in più

Nazione:	USA
Città:	Algonia
Via:	1 View Ln NE
Metri quadri Vivibili:	34
Metri quadri Lotto:	59.27
Metri quadri Seminterrato:	0.0
Metri quadri Calpestabili:	34.37
Anno di costruzione:	2014
Anno di restauro:	2014
Piani:	0.0
Affaccio sul mare:	0
Stanze:	1.00
Vista:	0
Condizione:	0
Scegli il modello di predizione:	Random Forest

Avvia Predizione

VALUTA LA TUA CASA IN POCHI
SECONDI CON IL NOSTRO VALUTATORE
DE GIGLIO & FRANCO
IMMOBILIARE



Parametri di selezione

INGEGNERIA DELLA CONOSCENZA A.A.2024/25

PARAMETRI GEOGRAFICI

La prima categoria di parametri rappresenta le caratteristiche geografiche della proprietà e sono i seguenti :

- **Nazione**: Indica la nazione di appartenenza della proprietà. È un valore selezionato dall'utente che influenza i dati geografici e di mercato.
- **Città** : Specifica la città in cui si trova l'immobile. Serve per filtrare le informazioni relative alle strade e altre caratteristiche locali.
- **Via** : Indica la via specifica della proprietà. Utilizzata per identificare in modo univoco l'immobile in combinazione con città e nazione.

PARAMETRI COSTRUTTIVI E ABITATIVI

La seconda categoria ,invece, rappresenta le caratteristiche costruttive e abitative della proprietà, e sono i seguenti :

- **Metri quadri Vivibili** : La superficie totale abitabile della casa, espressa in metri quadrati.
- **Metri quadri Lotto** : Rappresenta l'area totale del lotto su cui si trova la casa, espressa in metri quadrati.
- **Metri quadri Seminterrato** : Indica la superficie del seminterrato della casa, espressa in metri quadrati.
- **Metri quadri Calpestabili** : La superficie calpestabile della casa, che esclude il seminterrato.
- **Anno di costruzione** : L'anno in cui è stata costruita la casa.
- **Anno di restauro** : L'anno in cui l'immobile è stato ristrutturato.
- **Piani** : Il numero totale di piani presenti nella casa.
- **Affaccio sul mare** : Indica se la casa ha una vista sul mare o si trova vicino a una costa .
- **Stanze** : Il numero di stanze totali nell'immobile, esclusi bagni e altre aree non abitative.
- **Vista** : Una valutazione della qualità della vista dalla proprietà, su una scala numerica (ad esempio, da 0 a 4).
- **Condizione** : Un'indicazione dello stato generale della casa (da 1 a 5, dove 5 rappresenta una condizione eccellente).

Selezione Parametri

INGEGNERIA DELLA CONOSCENZA A.A.2024/25

Nell'interfaccia mostrata nell'immagine, i parametri possono essere selezionati utilizzando metodi diversi a seconda della loro tipologia. Per i campi come la nazione, la città e l'indirizzo, è presente un elenco preimpostato a discesa (menu a tendina) da cui l'utente può scegliere un'opzione. Tuttavia, per alcuni campi come l'indirizzo, è anche possibile digitare manualmente un valore specifico. Per i parametri numerici, come i metri quadri vivibili, il lotto, il seminterrato e altri, l'utente può inserire il valore direttamente digitandolo nei campi corrispondenti. Questo sistema offre una combinazione di flessibilità e facilità d'uso, permettendo di selezionare opzioni già definite oppure di personalizzare i valori inseriti.

Dacci qualche informazione in più

Nazione:	<input type="text" value="USA"/>
Città:	<input type="text" value="Algona"/>
Via:	<input type="text" value="1009 S 229th PI"/>
Metri quadri Vivibili:	<input type="text" value="71"/>
Metri quadri Lotto:	<input type="text" value="59.27"/>
Metri quadri Seminterrato:	<input type="text"/>
Metri quadri Calpestabili:	<input type="text" value="34.37"/>
Anno di costruzione:	<input type="text" value="2014"/>
Anno di restauro:	<input type="text" value="2014"/>
Piani:	<input type="text" value="2.0"/>
Affaccio sul mare:	<input type="text" value="1"/>
Stanze:	<input type="text" value="3.00"/>
Vista:	<input type="text" value="2.0"/>
Condizione:	<input type="text" value="5.0"/>
Scegli il modello di predizione:	<input type="text" value="SGD"/>

Avvia Predizione

MODELLO DI PREVISIONE

INGEGNERIA DELLA CONOSCENZA A.A.2024/25

DIFFERENZA TRA RANDOM FOREST E SGD

Random Forest:

- Output: Valore numerico continuo (es. prezzo stimato: 433038,77).
- Applicazione: Adatto per problemi di regressione.
- Vantaggi: Alta precisione, robustezza e gestione di relazioni non lineari.

Scegli il modello di previsione:

Il prezzo predetto è: 433038.77

Scegli il modello di previsione:

Questo sample ha probabilità 90.77%
di rientrare nella fascia (200000, 650000).

SGD (Stochastic Gradient Descent):

- Output: Probabilità di appartenenza a una fascia di valori (es. 90,77% nella fascia 200000-650000).
- Applicazione: Ideale per problemi di classificazione.
- Vantaggi: Leggero e veloce, adatto a grandi dataset.

 BACK

INGEGNERIA DELLA CONOSCENZA A.A.2024/25

Il nostro Team



Andrea Franco

774248

a.franco56@studenti.uniba.it



Pietro De Giglio

779409

p.degiglio5@studenti.uniba.it

Grazie per
l'attenzione.

