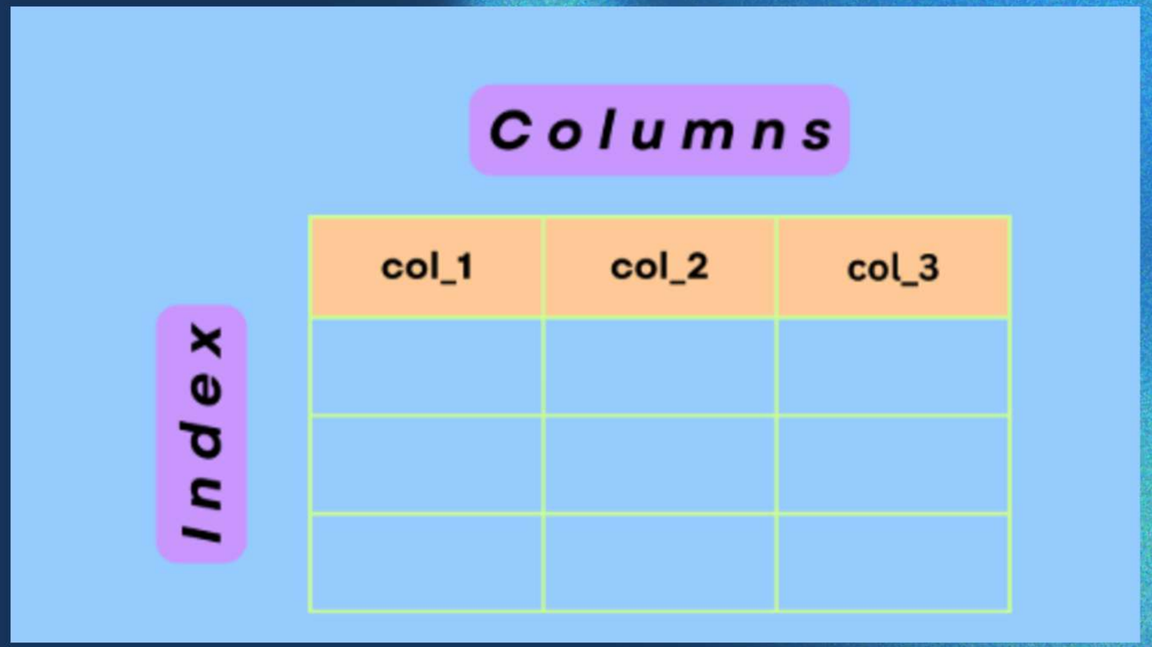
Esempio applicazione di machine learning

https://drive.google.com/file/d/1e-RhL70NPn5WRuzzzgKpyz1pcOKf2v83/view?usp=drive_link

Si inizia con l'importazione del dataset "Wine recognition dataset" presente nella libreria Scikit learn. Per una maggiore leggibilità e "manovrabilità" converto il dataset in Pandas Dataframe e tramite i suoi attributi ne visulaizzo le informazioni principali (numero di istanze, features e classi dei vini).



Eseguendo alcune operazioni di analisi osservo i seguenti dettagli:

- sono presenti sbilanciamenti nella distribuzione delle classi
- alcune features denotano una maggiore suddivisione delle classi, come “lavanoids” e “color_intensity”
- la feature “proline” ha un range numerico molto maggiore rispetto alle altre features

La prima domanda che mi pongo è: quale modello potrebbe predire le classi dei vini nel migliore dei modi?

A questo proposito procedo nel seguente metodo:

IMPORTAZIONE DEI MODELLI

Importo i modelli di cui saranno valutate le prestazioni, tra cui un modello semplice (GaussianNB), il quale servirà come termine di confronto per capire se avvalersi di modelli più complessi porterà a risultati migliori

SUDDIVISIONE DEL DATASET

Una volta estrapolate le variabili X e y dal nostro dataset divido quest'ultimo ulteriormente in "train" e "test"

CALCOLO DELLE PRESTAZIONI

Per ogni modello calcolo la media del punteggio di f1-score su ogni porzione di dataset (train set), usando un metodo di cross-validation

STAMPA DEI RISULTATI

Salvo i risultati ottenuti da ogni modello e li stampo; ne emerge che il miglior modello risultante è il RandomForest

Tuttavia la scelta del modello non basta da sola ad assicurarci un analisi con buone capacità rappresentative o evitare l'overfitting. In questi casi si va a indagare sui suoi iperparametri, nel caso specifico ne prendo in considerazione due:

- n_estimarors: rappresenta il numero di alberi che comporranno la foresta, molti alberi garantiscono stabilità e buona generalizzazione, tuttavia se troppo elevato comporta alti costi computazionali
- max_depht: rappresenta la profondità di ogni albero, se troppo elevato comporta il fenomeno di overfitting che cattura anche il grado di caos dei pattern

Fasi di valutazione iperparametri

SCELTA DEI VALORI

Per ogni iperparametro scelgo 4 valori. Questi valori rappresentano livelli graduali di complessità; lo scopo è determinare il valore ottimale che garantisca il giusto compromesso fra efficacia predittiva e costo computazionale

SCELTA DELLA METRICA

In questo caso il metodo di valutazione scelto è l' "F-1 score"; questa metrica di valutazione bilancia precision e recall, quindi è utile quando vogliamo penalizzare sia i falsi positivi che i falsi negativi.

APPLICAZIONE GRID-SEARCH

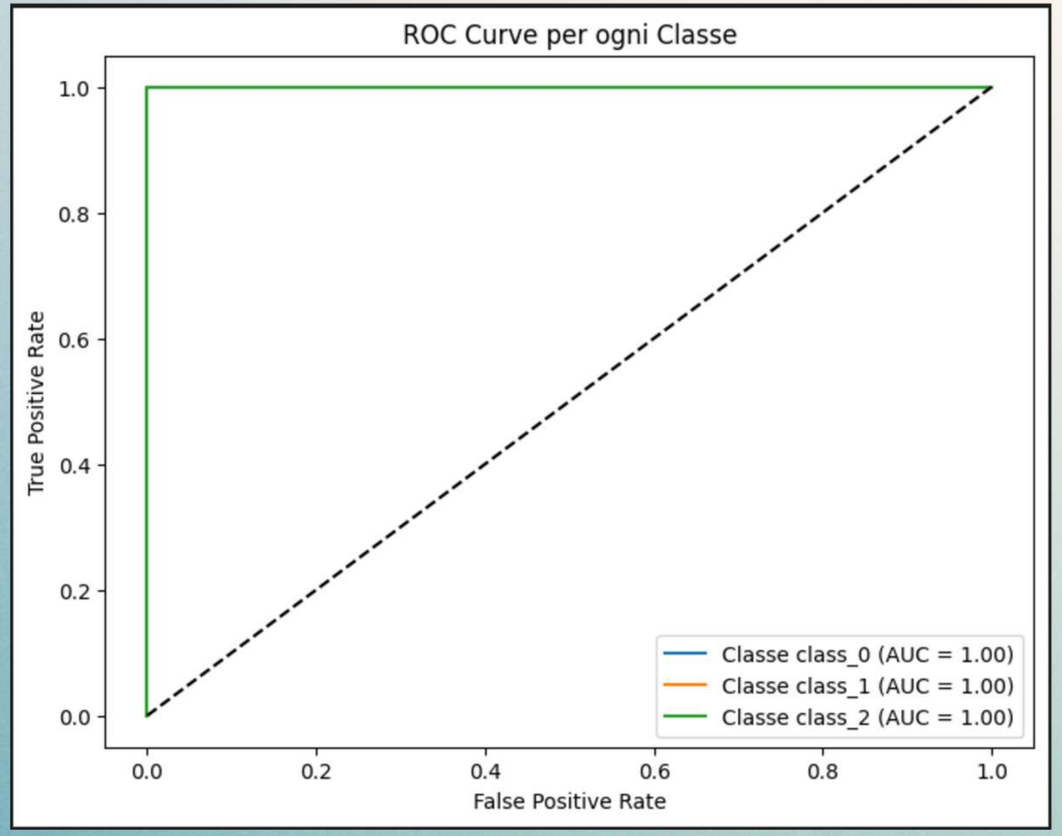
La gridsearch è una classe che permette di determinare la combinazione di uno o più iperparametri (griglia di iperparametri) che meglio predice un target, una volta stabilita una metrica di valutazione (accuracy in questo caso), un metodo di cross-validation (k-fold) e un modello (RF)

Ma una volta scelto il modello con i giusti iperparametri come faccio a determinare quanto bene sarà in grado di prevedere la classe dei vini? Un metodo può essere quello dell'AUC-ROC: è una metrica utilizzata per valutare le prestazioni di un modello di classificazione binaria. Quindi in questo caso andrò a:

- convertire la mia y di test (3 classi) in una matrice binaria
- il mio modello selezionato (best_model) estrapolerà una matrice y_proba (valori compresi tra 0 a 1) che descriverà la possibilità di ogni campione di ricadere in quella classe (ogni colonna rappresenta una classe)
- la funzione "roc_auc_score" estrapolerà un valore confrontando quanto le probabilità calcolate corrispondono alla realtà, in questo caso ho optato per il parametro "One versus rest", ovvero ad ogni iterazione una classe viene consiedarata "vera" e le altre "false" e così via.

Per una migliore visualizzazione del risultato atteso ho tradotto i risultati in un grafico grazie alla funzione “roc_curve” e la libreria matplotlib:

- La curva ROC (la linea verde) descrive il compromesso tra la Sensibilità (Recall) e il Tasso di Falsi Positivi (FPR) per diversi valori di soglia di classificazione
- L'AUC è l'area sotto la curva ROC e misura quanto bene il modello separa le classi, un AUC uguale a 1 rappresenta un modello perfetto, un AUC uguale 0.5 un modello casuale, sotto questa cifra il modello non ha funzionato



Grazie per l’attenzione!