# Dislexia-iCon

# Un Sistema di Machine Learning per la Predizione e il supporto alla dislessia

**Gruppo di lavoro:**

- Russo Nicola, 737220, n.russo26@studenti.uniba.it
- Squarcella-Gorgoglione Francesco Pio, 758467, f.squarcellagorgo@studenti.uniba.it
- Troiano Francesco, 779566, f.troiano7@studenti.uniba.it

**Link GitHub:**

https://github.com/Russo888/Dislexia-iCon-Un-Sistema-di-Machine-Learning-per-la-Predizione-e-il-Supporto-alla-Dislessia.git

AA 2024-2025

**INDICE:**

## INTRODUZIONE

Questo progetto si propone di supportare la diagnosi di dislessia attraverso l'applicazione di diversi algoritmi di apprendimento automatico. La dislessia è un disturbo dell'apprendimento che influisce sulla capacità di leggere e scrivere correttamente.

L'identificazione precoce della dislessia è fondamentale per intervenire tempestivamente e supportare al meglio le persone che ne sono affette. Tuttavia, i metodi tradizionali di diagnosi possono essere lunghi e costosi.

Per questo motivo, abbiamo sviluppato un sistema che utilizza un dataset preso su kaggle.com riguardante la dislessia per addestrare diversi modelli di apprendimento, tra cui K-nearest neighbors (KNN), Random Forest, Support Vector Machine (SVM), Neural Network e K-means. L'obiettivo è valutare le prestazioni di ciascun algoritmo nel distinguere tra persone con e senza dislessia, al fine di individuare il modello più efficace per uno screening efficiente e accessibile.

# 1. DATASET

Presentiamo un set di dati relativo allo screening della dislessia, raccolto da un esperimento condotto su un gruppo eterogeneo di soggetti. Questo dataset contiene caratteristiche influenti utilizzabili per l'analisi, in particolare per identificare tratti di dislessia e migliorare la classificazione dei casi. Il set di dati registra una serie di metriche comportamentali e cognitive che si sono dimostrate efficaci nel rilevare la dislessia.

## 1.1. Descrizione dataset (dominio)

Questo dataset si basa su una raccolta di 3644 soggetti. A ciascun partecipante è stato somministrato un test strutturato in 32 attività, ciascuna caratterizzata da specifiche misurazioni quantitative delle performance.

Il test include parametri come:

- **Clicks**: Numero di risposte fornite.
- **Hits**: Numero di risposte corrette.
- **Misses**: Numero di errori.
- **Score**: Punteggio ottenuto.
- **Accuracy**: Accuratezza delle risposte.
- **Missrate**: Tasso di errore.

Oltre a queste metriche, vengono raccolte informazioni individuali come **sesso, età, lingua madre e altre lingue parlate**.

Se un soggetto presenta determinate caratteristiche nei punteggi ottenuti durante il test, vi è un'alta probabilità che manifesti tratti riconducibili alla dislessia. Il campo "Dyslexia" del dataset indica la presenza o assenza di diagnosi di dislessia per ciascun individuo. Questo set di dati rappresenta una risorsa utile per lo studio dei pattern cognitivi e per migliorare gli strumenti di diagnosi della dislessia.

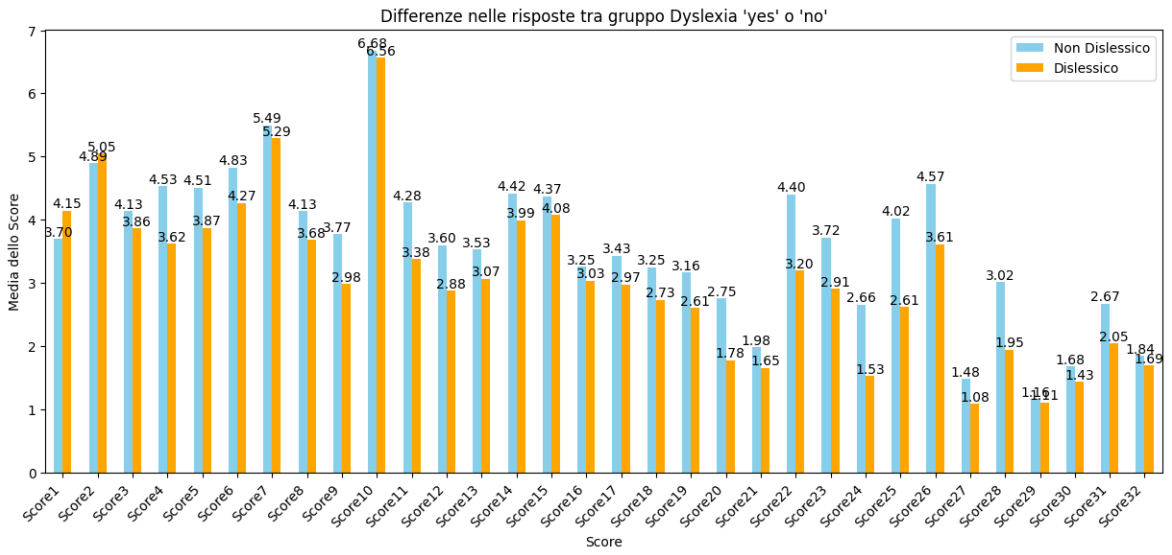
| Feature | Type | Description |
|---|---|---|
| Gender | Object | Genere del soggetto: Maschio o Femmina |
| Nativelang | Object | Lingua madre del soggetto |
| Otherlang | Object | Se il soggetto parla altre lingue (Sì/No) |
| Age | Integer | Età del soggetto (in anni) |
| Clicks1 | Integer | Numero di clic effettuati nel test 1 |
| Hits1 | Integer | Numero di colpi andati a segno nel test 1 |
| Misses1 | Integer | Numero di mancate nel test 1 |
| Score1 | Integer | Punteggio totale del test 1 |
| Accuracy1 | Float | Accuratezza nel test 1 (percentuale) |
| Missrate1 | Float | Percentuale di errori nel test 1 |
| Clicks2 | Integer | Numero di clic effettuati nel test 2 |
| Hits2 | Integer | Numero di colpi andati a segno nel test 2 |
| Misses2 | Integer | Numero di mancate nel test 2 |
| Score2 | Integer | Punteggio totale del test 2 |
| Accuracy2 | Float | Accuratezza nel test 2 (percentuale) |
| Missrate2 | Float | Percentuale di errori nel test 2 |
| ... | ... | ... |
| Dyslexia | Object | Indica se il soggetto è dislessico (Sì/No) |

Il dataset contiene numerosi test e misurazioni simili, ciascuno rappresentato da variabili come "Clicks", "Hits", "Misses", "Score", "Accuracy" e "Missrate". Ogni serie numerica corrisponde a un test specifico (es. 1, 2, 3, ecc.).

## 1.2 Osservazione grafica dei dati

Abbiamo svolto delle osservazioni grafiche che ci permettessero di valutare la correlazione dei dati e soprattutto per capire in che modo la diagnosi fosse influenzata da essi. Qui di seguito riportiamo un grafico che abbiamo realizzato in linguaggio python:

- Il grafico descrive la differenza nella media dello score tra chi è risultato dislessico e chi no.



# 2. ONTOLOGIA

Un'ontologia è un modello di conoscenza che consente di rappresentare in modo univoco le informazioni relative ad un determinato dominio. Durante l'analisi del dominio si indentificano i concetti principali, le relazioni e le proprietà che caratterizzano il dominio.

Successivamente questi concetti, relazioni e proprietà possono essere formalizzate mediante un linguaggio di rappresentazione formale come, ad esempio, OWL (Ontology Web Language).

## 2.1 Analisi Dominio

Dato il set di dati utilizzato e le informazioni sugli attributi, abbiamo deciso di dividere gli attributi del dataset in:

- *"ciò che deve essere rappresentato"*: gli attributi fondamentali e cruciali che devono essere rappresentati per catturare le informazioni essenziali dal dataset. Nel contesto della dislessia, include: persone, dislessia, difficoltà e prestazioni
- *"ciò che caratterizza ciò che deve essere rappresentato"*: le proprietà delle entità rappresentate, (nel caso delle persone ad esempio possono essere età, sesso...)
- *"ciò che può essere ricavato"*
- *"ciò che può essere scartato"*.

### 2.1.1 Classi:

Analizzando il dominio abbiamo deciso di rappresentare come classi dell'ontologia:

• ***Persona***: La classe Paziente, rappresenta l'insieme dei pazienti sottoposti al test. A questa classe abbiamo associato diverse proprietà:

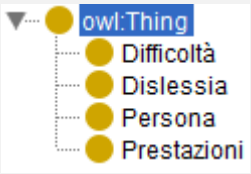
- età: rappresenta l'età del paziente al momento del test.
- genere: rappresenta il genere del paziente.
- linguaMadre: vera se paziente è inglese, falso altrimenti.
- conosceAltraLingua: vera se il paziente conosce altre lingue, falso altrimenti.

• ***Prestazioni:*** La classe Prestazioni, rappresenta l'insieme delle prestazioni sulla base dei test svolti dai pazieni. A questa classe sono associati cinque proprietà:

- valoreAccuracy: Indica la percentuale di previsioni corrette rispetto al totale.
- valoreHits: Numero di previsioni corrette (classificazioni giuste).
- valoreMisses: Numero di previsioni errate (classificazioni sbagliate).
- valoreMissrate: Percentuale di errori rispetto al totale delle previsioni.
- valoreScore: Punteggio complessivo del modello basato su una metrica specifica.

- ***Dislessia:*** rappresenta l'insieme dei pazienti che sono affetti dal disturbo della dislessia.

- ***Difficoltà:*** rappresenta l'insieme delle difficoltà che una persona può avere

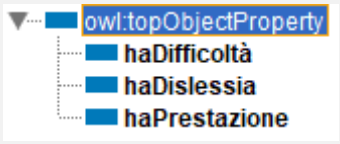



**2.1.2 Object property:**

Una object property permette di mettere in relazione due individui, siano essi di classi distinte o della stessa classe.

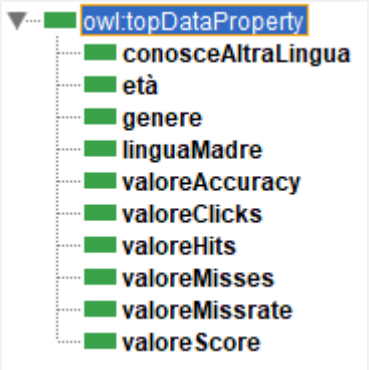
Tra le classi abbiamo definito delle relazioni che rappresentano come queste interagiscono tra di loro. Le relazioni definite sono:

- ***haDifficoltà(Paziente) -> Difficoltà*** : permette di individuare le difficoltà svolte da un paziente.
- ***haPrestazione(Persona) ->Prestazione***: permette di individuare a prestazione di un paziente sulla base dei test effettuati.
- ***haDislessia(Test) ->Domanda***:Permette di capire se un paziente è affetto dal disturbo della dislessia o meno.



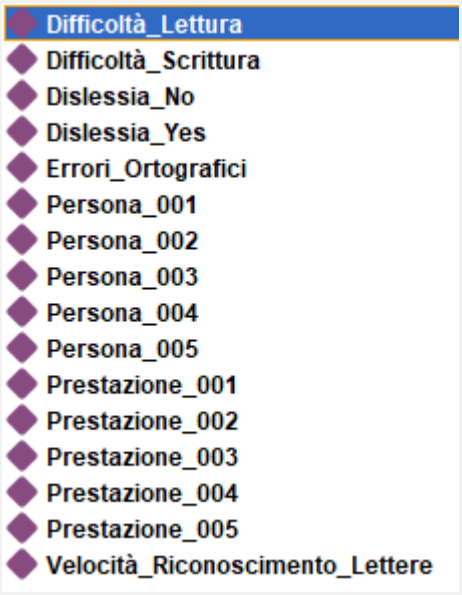

**2.1.3 Data property:**

Una data property permette di mettere in relazione un individuo con un valore di tipo primitivo.



### 2.1.4 Individuals:

Per alcune entità si sono create delle istanze, ad esempio per individuare istanze della classe “persona” a cui attribuire una prestazione e le relazioni con essa. Individui inseriti nella nostra ontologia:



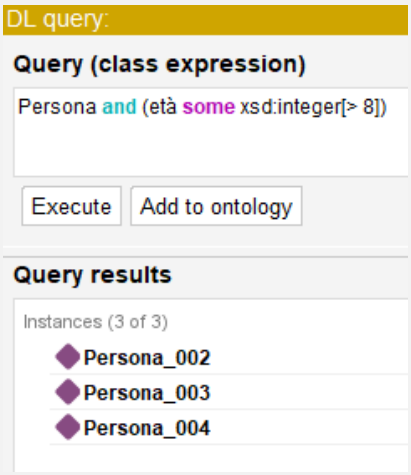

## 2.2 Software per la realizzazione dell’Ontologia

L’ontologia è stata create mediante il software Protege, che è uno strumento per creare e gestire ontologie e che permette di definire concetti, classi e relazioni in ontologie basate su standard come OWL.

# 3. QUERY

Successivamente sono state formulate delle query per interrogare l'ontologia.

## 3.1 DL Query




Una query che restituisce tutti i pazienti con età maggiore di 8 anni, scritta in linguaggio DL (Descriptions logics), che è un linguaggio formale utilizzato principalmente per la modellazione concettuale e l'ontologia nelle discipline dell'intelligenza artificiale e della rappresentazione della conoscenza.

## 3.2 OwlReady

È stato poi creato il file "query.py" con il quale è possibile consultare l'ontologia direttamente in Python grazie alla libreria OwlReady2 per la manipolazione di ontologie e il ragionamento.

- Con il seguente codice è quindi possibile estrarre dall'ontologia la lista delle classi, delle object property, dei data property e degli individui che appartengono alle relative classi.

```
# Stampa le classi, proprietà e individui
print("----------------------Classi in ontologia:--------------------------\n")
print(list(onto.classes()), "\n")

print("----------------------Proprietà oggetto:--------------------------\n")
print(list(onto.object_properties()), "\n")

print("----------------------Proprietà dati:--------------------------\n")
print(list(onto.data_properties()), "\n")

# Cerca e stampa individui della classe Persona
print("----------------------Lista delle persone:--------------------------\n")
persone = onto.search(is_a = onto.Persona)
print(persone, "\n")

# Cerca e stampa individui della classe Prestazioni
print("----------------------Lista delle prestazioni:--------------------------\n")
prestazioni = onto.search(is_a = onto.Prestazioni)
print(prestazioni, "\n")
```

Il codice precedente produce i seguenti risultati:

```
----------Classi in ontologia:----------

[ontologia_dislessia.owx.Difficoltà, ontologia_dislessia.owx.Dislessia, ontologia_dislessia.owx.Persona, ontologia_dislessia.owx.Prestazioni]

----------Proprietà oggetto:----------

[ontologia_dislessia.owx.diagnosi, ontologia_dislessia.owx.haDifficoltà, ontologia_dislessia.owx.haDislessia, ontologia_dislessia.owx.haPrestazione]

----------Proprietà dati:----------

[ontologia_dislessia.owx.conosceAltraLingua, ontologia_dislessia.owx.età, ontologia_dislessia.owx.genere, ontologia_dislessia.owx.linguaMadre, ontologia_dislessia.owx.valoreAccuracy, ontologia_dislessia.owx.valoreClicks, ontologia_dislessia.owx.valoreHits, ontologia_dislessia.owx.valoreMisses, ontologia_dislessia.owx.valoreMissrate, ontologia_dislessia.owx.valoreScore]

----------Lista delle persone:----------

[ontologia_dislessia.owx.Persona, ontologia_dislessia.owx.Persona_001, ontologia_dislessia.owx.Persona_002, ontologia_dislessia.owx.Persona_003, ontologia_dislessia.owx.Persona_004, ontologia_dislessia.owx.Persona_005]

----------Lista delle prestazioni:----------

[ontologia_dislessia.owx.Prestazioni, ontologia_dislessia.owx.Prestazione_001, ontologia_dislessia.owx.Prestazione_002, ontologia_dislessia.owx.Prestazione_003, ontologia_dislessia.owx.Prestazione_004, ontologia_dislessia.owx.Prestazione_005]
```

- Attraverso il seguente codice è stato invece possibile interrogare l'ontologia.

```python
# QUERY----------------------------------------------------------------------
print("_____________________________QUERY_____________________________________\n")

# Trova le persone che hanno effettuato una prestazione
personeConPrestazioni = [p for p in onto.Persona.instances() if p.haPrestazione]

print("- Persone che hanno effettuato una prestazione:\n")
for persona in personeConPrestazioni:
    print(persona.name)

# Trova le persone che sono dislessiche
personeDislessiche = onto.search(is_a = onto.Persona, haDislessia=onto.Dislessia_Yes)

print("\n\n- Persone dislessiche:\n")
for persona in personeDislessiche:
    print(persona.name)
```

Il codice precedente restituisce come risultato la lista delle persone che hanno effettuato una prenotazione e la lista delle persone dislessiche

```
_____________________________QUERY_____________________________________

- Persone che hanno effettuato una prestazione:

Persona_001
Persona_002
Persona_003
Persona_004
Persona_005


- Persone dislessiche:

Persona_003
```

# 4. APPRENDIMENTO SUPERVISIONATO

L'apprendimento supervisionato è una tecnica di apprendimento automatico che mira a istruire un sistema informatico in modo da consentirgli di elaborare automaticamente previsioni sui valori di uscita di un sistema rispetto ad un input sulla base di una serie di esempi ideali, costituiti da coppie di input e di output, che gli vengono inizialmente forniti.

## 4.1 Decisioni di progetto

In questo progetto, l'obiettivo dell'apprendimento supervisionato è risolvere un problema di classificazione, in particolare utilizzando la colonna 'Dyslexia' come variabile target da predire basandosi sulle altre caratteristiche relative ai dati dei pazienti.

Inizialmente è stato necessario decidere quali modelli utilizzare. Sono stati scelti gli algoritmi con le migliori prestazioni, in quanto la sensibilità del tema necessitava una grande precisione nelle valutazioni. Per questo motivo è stato scelto ad esempio random forest piuttosto che decision tree, in quanto è stata preferita la precisione del random forest alla minore complessità del decision tree, che potrebbe garantire prestazioni inferiori nel caso, ad esempio, in cui l'albero diventa più profondo e complesso.

Sono stati scelti 4 altri modelli (implementati con le librerie *'sklearn'* e *'keras'*) e ne è stata valutata l'efficacia e l'efficienza:

- **K-nearest-neighbors (KNN)**
- **Random Forest**
- **Support Vector Machine (SVM)**
- **Neural Network**

## 4.2 Metriche di valutazione

Andiamo adesso ad analizzare le metriche con cui sono stati valutati i modelli ed i risultati. Per ogni algoritmo implementato sono statati prodotti 4 tipologie di grafici differenti:

- **ROC Curve (Receiver Operating Characteristic Curve):** La curva ROC è un grafico in cui l'asse delle ordinate rappresenta il tasso di veri positivi (True Positive Rate), mentre l'asse delle ascisse rappresenta il tasso di falsi positivi (False Positive Rate).
- **Precision-Recall Curve:** Questa curva rappresenta il trade-off tra la precisione e il recall di un modello di classificazione binaria.
- **Bar Chart di Varianza e Deviazione Standard:** Questo tipo di grafico a barre rappresenta la varianza e la deviazione standard dei punteggi di cross-validation. La varianza e la deviazione standard

aiutano a comprendere quanto i risultati siano consistenti o variabili su diverse iterazioni della cross-validation.

- **Matrice di Confusione:** La matrice di confusione è una tabella che mostra il numero di previsioni corrette e errate fatte da un modello di classificazione in termini di veri positivi (TP), falsi positivi (FP), veri negativi (TN) e falsi negativi (FN). .

La **cross-validation** è un'altra tecnica utilizzata per valutare le prestazioni dei modelli in modo accurato e affidabile. Abbiamo scelto di dividere i dati in 5 "fold" uguali, in modo da addestrare e testare il modello 5 volte, ognuna delle quali usa una fold diversa come set di test e l'unione delle rimanenti come set di addestramento.

## 4.3 Selezione delle Feature

Nella selezione delle feature abbiamo scelto di selezionare:

- Metriche di performance nei test ('Score1', 'Score2', ..., 'Score32'): Questi punteggi rappresentano la performance complessiva nei vari test di lettura e comprensione e sono le nostre feature principali, in quanto riflettono direttamente la presenza di dislessia.
- Età ('Age') e Genere ('Gender'): L'età e il genere possono influenzare la prevalenza e la manifestazione della dislessia, quindi sono state incluse come feature potenzialmente rilevanti.
- Accuratezza e frequenza di errori ('Accuracy1', 'Missrate1', ..., 'Accuracy32', 'Missrate32'): Queste metriche possono fornire ulteriori informazioni sulle difficoltà specifiche di lettura e comprensione.
- Lingua nativa ('Nativelang'): La lingua nativa potrebbe influire sulla difficoltà di lettura e comprensione, specialmente se diversa dalla lingua del test.
- Dislessia ('Dyslexia'): Questa è la variabile target che indica la presenza o assenza di dislessia.

Feature non incluse:

- Lingue aggiuntive ('Otherlang'): Anche se potrebbe influire sulla performance, non è stato ritenuto un fattore rilevante al punto da considerarlo come feature principale.
- Metriche secondarie di interazione ('Clicks1', 'Hits1', 'Misses1', ..., 'Clicks32', 'Hits32', 'Misses32'): Questi dati sono stati esclusi per ridurre la complessità del modello, nonostante possano fornire ulteriori dettagli sul comportamento del soggetto.

Questa selezione delle feature si basa sull'importanza percepita e sulla loro correlazione con la presenza di dislessia.

### 4.4 Preprocessing del dataset

L'addestramento di ogni modello inizia con il preprocessing dei dati. L'obiettivo è adattare il dataset per gli algoritmi di apprendimento supervisionato, eliminando i valori nulli e gestendo le feature categoriche.

- **Conversione delle colonne in stringhe:**

```
def clean_data(data):
    """Converte le colonne stringa in numerico e mappa i valori categorici."""
    for col in data.columns:
        data[col] = data[col].astype('string')
```

Questo garantisce che tutti i valori siano trattati come stringhe, prevenendo errori di tipo.

- **Tentativo di conversione in numerico:**

```
        data[col] = data[col].astype('float', errors='ignore')
```

Dopo la conversione in stringa, il codice tenta di trasformare i valori in numeri (float). Se la conversione fallisce, come nel caso di valori testuali, l'errore viene ignorato.

- **Mappatura di valori categorici in numeri:**

```
    data['Gender'] = data['Gender'].map({'Male': 1, 'Female': 2})
    data['Dyslexia'] = data['Dyslexia'].map({'No': 0, 'Yes': 1})
    data['Nativelang'] = data['Nativelang'].map({'No': 0, 'Yes': 1})
    data['Otherlang'] = data['Otherlang'].map({'No': 0, 'Yes': 1})
```

Le variabili categoriche (sesso, dislessia, madrelingua, altre lingue) vengono convertite in valori numerici per facilitare l'utilizzo nei modelli di machine learning.

Abbiamo evitato l'utilizzo di one-hot encoding poiché le colonne categoriche contengono solo due valori ciascuna. L'applicazione di questa tecnica avrebbe comportato la creazione non necessaria di quattro nuove colonne binarie.

Data la natura del dataset, che presenta uno sbilanciamento significativo tra le classi, abbiamo implementato una strategia di bilanciamento utilizzando **SMOTE** (Synthetic Minority Over-sampling Technique). Questa tecnica avanzata genera esempi sintetici della classe minoritaria, garantendo una maggiore equità nella distribuzione dei dati e migliorando l'affidabilità delle previsioni.

## 4.5 Divisione dei Dati

Per il nostro dataset, abbiamo scelto una suddivisione del 80% per l'addestramento e 20% per il test. Questa proporzione rappresenta un equilibrio ottimale tra i dati necessari per l'addestramento e quelli richiesti per una valutazione accurata delle prestazioni del modello.

```
# Divisione dei dati in training e test set
X_train, X_test, y_train, y_test = train_test_split(X, y, train_size = 0.80, random_state = 13)
```

Una percentuale maggiore del test set ridurrebbe i dati disponibili per l'addestramento, compromettendo la capacità di apprendimento del modello. D'altra parte, aumentare la percentuale del training set potrebbe causare overfitting, limitando la capacità di generalizzazione su nuovi dati.

## 4.6 K-Nearest Neighbors (KNN)

---

Il KNN è un algoritmo di classificazione che mira a determinare la classe di appartenenza di un dato di input cercando tra tutti gli esempi di addestramento quello più vicino al dato di input in base alla metrica desiderata, come ad esempio la distanza euclidea.

### 4.6.1 Decisioni di progetto

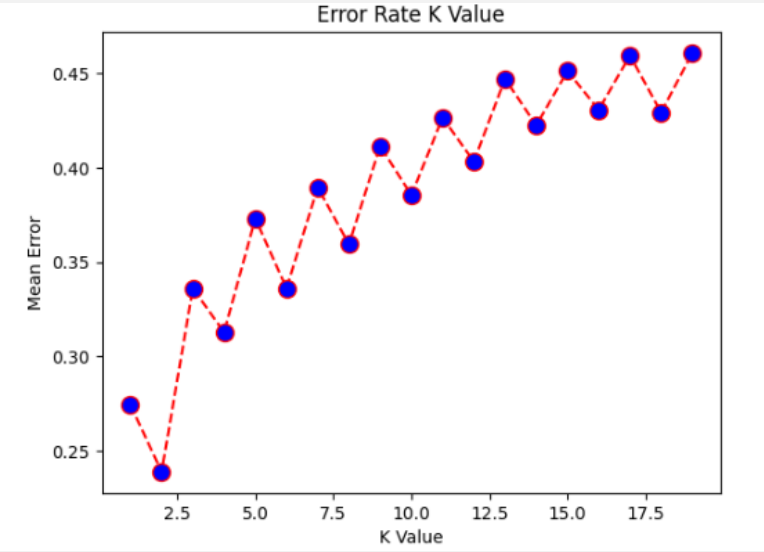
Questo modello è stato scelto perché semplice da implementare, versatile, in quanto funziona bene con dataset di qualsiasi dimensione, e robusto anche in presenza di dati rumorosi o valori anomali. Dato che il modello si basa sui vicini più prossimi, un singolo valore anomalo avrà meno impatto sull'output finale.

### 4.6.2 Ottimizzazione numero di vicini

L'obiettivo di questo codice è quello di valutare come l'errore varia al variare del numero di vicini utilizzati in un range di valutazione da 1 a 20.

```
error = []
for i in range(1, 20):
    knn = KNeighborsClassifier(n_neighbors=i)
    knn.fit(X1, y1)
    pred_i = knn.predict(X_test)
    error.append(np.mean(pred_i != y_test))
```

Si sta cercando il valore ottimale di numero di vicini che minimizza l'errore di previsione e migliora la capacità di generalizzazione del modello. L'obiettivo è scegliere il valore di k che equilibra la complessità del modello con la sua capacità di generalizzazione.



Nel nostro caso p**er k piccoli (1-3), l'errore è basso**, il che suggerisce che il modello si adatta bene ai dati di training, ma potrebbe soffrire di overfitting. **All'aumentare di k, l'errore cresce progressivamente**, il che significa che il modello sta diventando troppo generalista e perde capacità predittiva. Basandoci sui risultati ottenuti in precedenza, **k=6 ci è sembrato un buon compromesso tra accuratezza e generalizzazione**.

### 4.6.3 Addestramento e predizione

Scelto il numero di vicini, inizia l'addestramento del modello. Durante questo processo, il modello regola i suoi parametri in modo che le previsioni siano il più vicine possibile ai valori reali.

```
# Addestramento del modello con il valore ottimale di K
neigh = KNeighborsClassifier(n_neighbors = 6)
knn = neigh.fit(X1, y1)
```

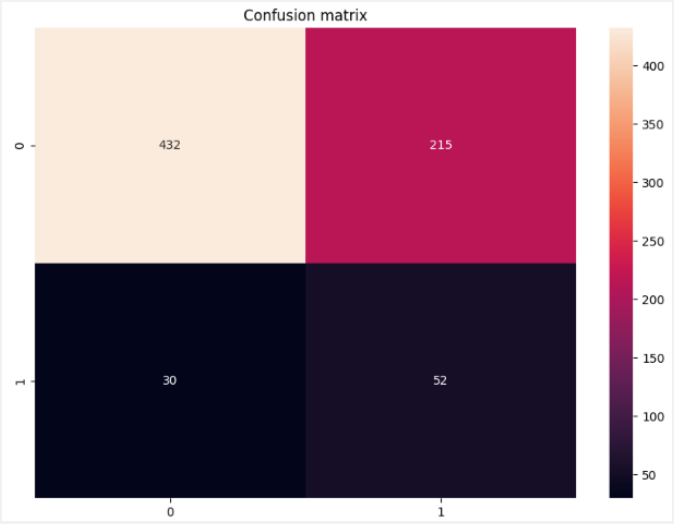
Il modello viene addestrato cercando di trovare la relazione tra le feature e le etichette di classe. Il codice crea quindi un modello di classificazione addestrato pronto per essere utilizzato per fare previsioni su nuovi dati, cercando di assegnare loro l'etichetta di classe appropriata in base alle sue vicinanze rispetto ai dati di addestramento.

```
# Effettua previsioni sul test set
prediction = knn.predict(X_test)
accuracy = accuracy_score(prediction, y_test)
print(f"accuracy_score: {accuracy:.2f}")
```

Il classificatore k-NN addestrato viene quindi usato per effettuare previsioni sul set di dati di test. Il metodo predict restituisce un array contenente i valori delle previsioni, che vengono confrontate con le etichette di classe effettive per calcolare il valore dell'accuracy.

### 4.6.4 Valutazione Finale

**Confusion Matrix:**



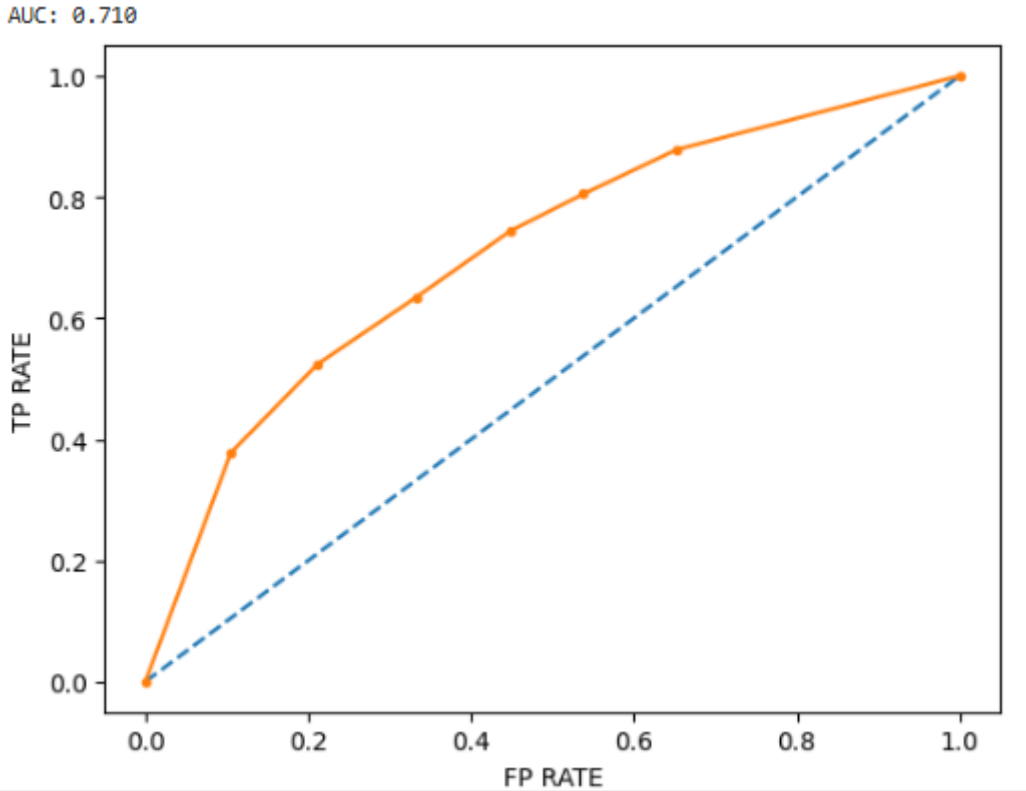

- 215 falsi positivi → il modello assegna erroneamente molti "1" ai campioni della classe 0.
- 30 falsi negativi → il modello classifica 30 veri "1" come "0".
- 52 veri positivi → correttamente identificati come "1".
- 432 veri negativi → correttamente identificati come "0"

**Cross-Validation Scores:**

```
cv_scores mean:0.8905055849500293

cv_score variance:9.944083873000027e-06

cv_score dev standard:0.0031534241505068784
```

- Media: 0.8905 → Il modello si comporta in modo stabile su diverse suddivisioni del dataset.
- Varianza: 9.94e-06 → Varianza molto bassa, indica che il modello è consistente nei diversi test.
- Deviazione standard: 0.0031 → Bassa fluttuazione nei risultati, il modello è robusto.
- Il modello ha una buona stabilità.
- Anche se l'accuratezza del test è 0.67, il modello ha una performance media vicina a 0.89 in validazione, quindi potrebbe essere un segno che il set di test è particolarmente difficile.

**AUC Score: 0.709:**



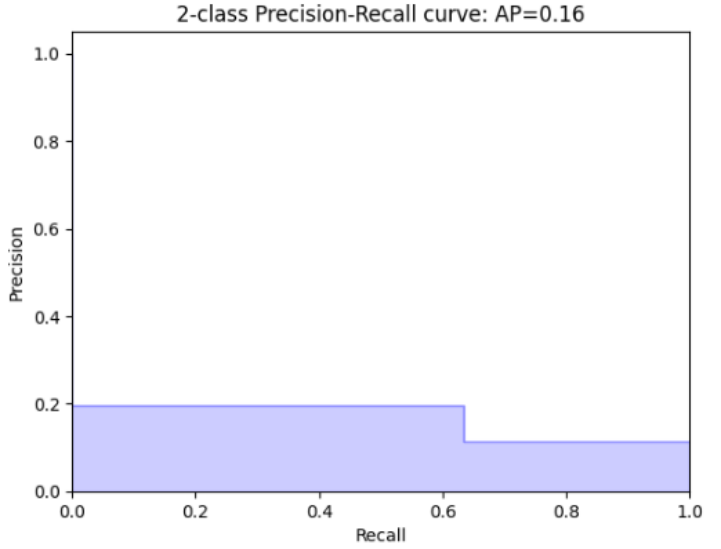

L'AUC (Area Under the Curve ROC) misura la capacità del modello di distinguere tra le due classi.

- 0.709 significa che il modello ha una discreta capacità

di separare le due classi. Un valore vicino a 0.7 indica un modello moderatamente buono.

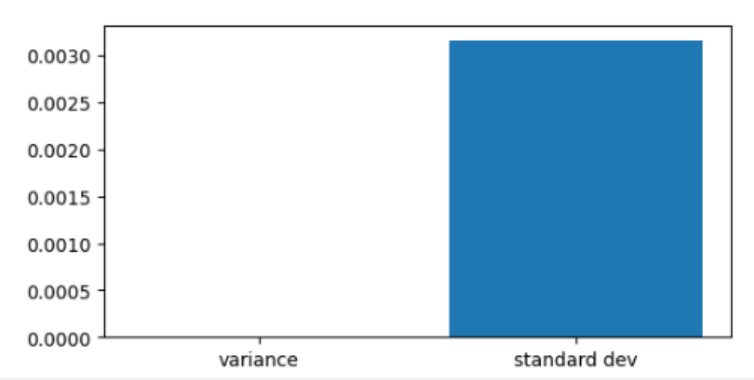
- Se l'AUC fosse vicino a 0.5, il modello sarebbe casuale.
- Per problemi di classificazione sbilanciata, l'AUC è un indicatore più affidabile dell'accuratezza.
- Il modello ha una buona capacità discriminante (migliore rispetto ai test con k più alti).
- Tuttavia, l'errore sulla classe minoritaria (1) è ancora alto, quindi si potrebbe provare a migliorare la selezione delle feature o il bilanciamento.

**Precision-Recall Curve:**




- Il valore di Average Precision (AP) = 0.16 è molto basso, indicando che il modello ha una scarsa capacità di distinguere correttamente la classe minoritaria.
- La precisione rimane molto bassa (~0.2) anche per valori elevati di recall, il che significa che molti dei positivi previsti sono in realtà falsi positivi.
- Questo indica che il modello non è efficace nella classificazione della classe 1, il che è coerente con i risultati precedenti sulla bassa precisione.

**Bar Chart di Varianza e Deviazione Standard:**




- La varianza è molto bassa, indicando che il modello è abbastanza stabile nelle sue previsioni e non mostra una grande variabilità tra le diverse esecuzioni.
- La deviazione standard (~0.003) è piccola, suggerendo che la cross-validation ha dato risultati consistenti.
- Tuttavia, la bassa varianza non significa che il modello sia accurato, ma solo che è coerente nei suoi risultati.

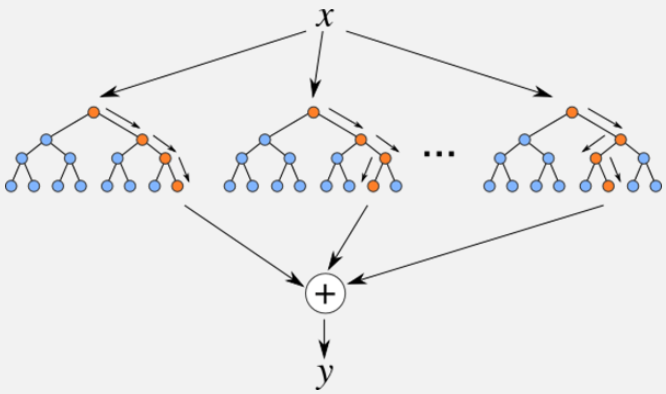
**Conclusione Finale**

Il modello KNN con **k=6** offre un compromesso tra accuratezza e capacità di distinguere le classi, ma soffre ancora nel classificare correttamente la classe minoritaria. Per migliorare ulteriormente, è necessario sperimentare tecniche di bilanciamento più avanzate o considerare modelli alternativi più adatti a dataset sbilanciati.

## 4.7 Random Forest

---

L'algoritmo **Random Forest** è un modello di apprendimento supervisionato che combina molteplici **alberi decisionali** per migliorare la precisione delle previsioni. Mentre un singolo albero decisionale può essere soggetto a overfitting, la combinazione di più alberi riduce questo rischio e migliora la stabilità del modello.




### 4.7.1 Decisioni di progetto

L'algoritmo Random Forest è stato scelto poiché tende ad avere una buona precisione e stabilità nei problemi di classificazione. Inoltre, la sua capacità di addestrare gli alberi in modo parallelizzato consente una rapida esecuzione anche con risorse computazionali limitate.

- **n_estimators=20**: si è scelto di impostare a 20 questo parametro, che indica il numero di alberi decisionali che vengono creati nell'ensemble, in modo da trovare un equilibrio tra prestazioni e efficienza computazionale, dato che un numero maggiore di alberi può portare a prestazioni migliori, ma aumenta anche il tempo di addestramento.
- **max_depth=30**: questo parametro indica la profondità massima degli alberi decisionali all'interno dell'ensemble. È stato scelto 30 in quanto è un valore che permette agli alberi di adattarsi ai dati diminuendo il rischio di overfitting che si avrebbe avuto con una maggiore profondità.
- **random_state=0**: questo parametro controlla la generazione dei numeri casuali all'interno del modello. Fornendo un valore specifico ci si assicura che l'addestramento e la previsione del modello siano riproducibili.
- **class_weight**: Dopo diverse prove, abbiamo bilanciato il peso delle classi per ridurre il numero di falsi positivi e minimizzare gli errori nella diagnosi della dislessia.

```
rf_clf = RandomForestClassifier(n_estimators=200, max_depth=6, class_weight=weights, random_state=42)
rf_clf.fit(X_train, y_train)
```

### 4.7.2 Bilanciamento delle classi con SMOTE

Poiché il dataset originale era sbilanciato, è stato applicato il metodo SMOTE (Synthetic Minority Over-sampling Technique) per bilanciare le classi:

**Distribuzione dopo SMOTE:**

```
Class distribution after SMOTE: Dyslexia
0    2603
1    2603
```

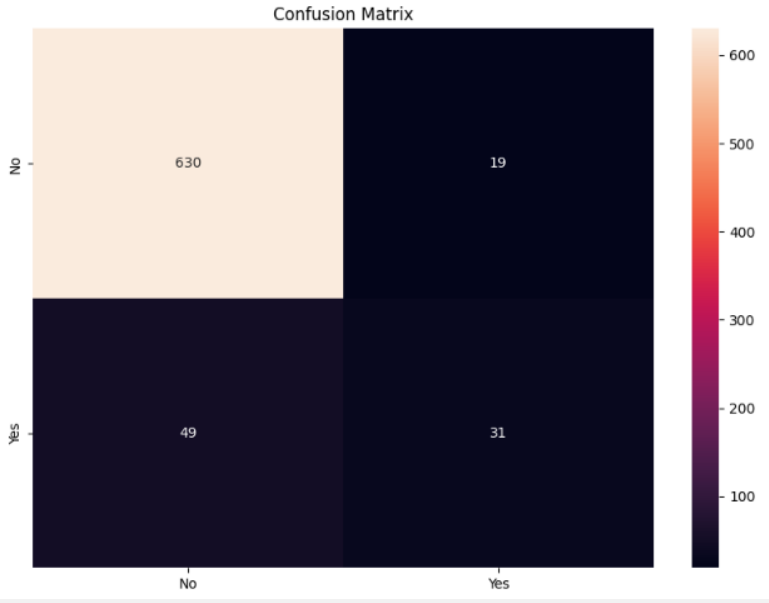
### 4.7.3 Valutazione Finale

Il modello è stato testato su un set di dati di validazione, producendo i seguenti risultati:

**Classification Report**

```
Random Forest Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.97      0.95       649
           1       0.62      0.39      0.48        80

    accuracy                           0.91       729
   macro avg       0.77      0.68      0.71       729
weighted avg       0.89      0.91      0.90       729
```

**Confusion Matrix:**

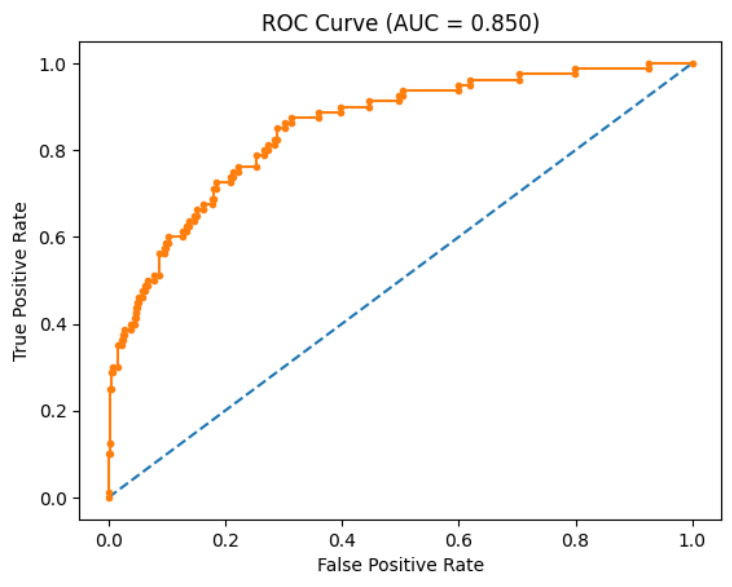


Interpretazione della matrice di confusione:

- 630 veri negativi (corretta classificazione della classe 0)
- 31 veri positivi (corretta classificazione della classe 1)
- 19 falsi positivi (classificati erroneamente come 1)
- 49 falsi negativi (classificati erroneamente come 0)

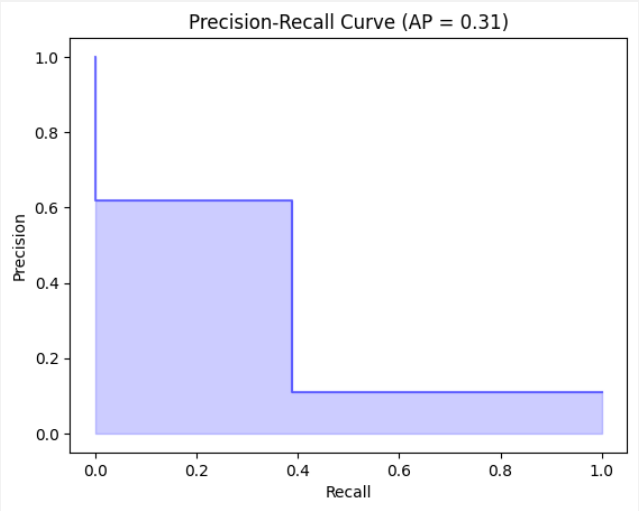
L'F1-score della classe 1 è 0.48, indicando che il modello ha ancora margini di miglioramento nella classificazione della dislessia.

**Roc Curve:**



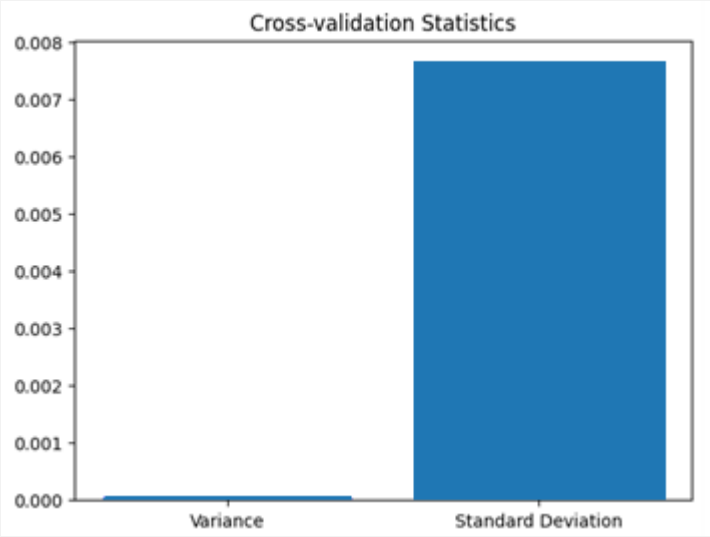

La curva ROC con AUC pari a 0.850 mostra che il modello ha una buona capacità discriminativa, con un rapido incremento iniziale che indica alta sensibilità e pochi falsi negativi. La parte piatta vicino al punto (1,1) suggerisce che il modello classifica correttamente la maggior parte delle istanze positive senza generare troppi falsi positivi.

**Precision-Recall Curve:**




La Precision-Recall Curve con AP di 0.31 evidenzia un bilanciamento non ottimale tra precisione e recall. La precisione è elevata solo per bassi valori di recall, suggerendo che il modello identifica con buona affidabilità alcuni casi positivi ma ne perde molti altri, indicando margini di miglioramento.

**Bar Chart di Varianza e Deviazione Standard:**




Il grafico della varianza e della deviazione standard nella validazione incrociata mostra una bassa variabilità delle prestazioni tra i diversi fold. La bassa varianza indica un modello stabile, mentre la deviazione standard contenuta conferma che i risultati ottenuti sono consistenti su diversi sottoinsiemi del dataset.

**Cross-Validation:**

Per valutare la generalizzazione del modello, è stata eseguita una cross-validation:

```
Cross-validation mean: 0.887
Cross-validation variance: 0.000
Cross-validation standard deviation: 0.008
```

- **Accuracy media**: 88.7%
- **Varianza**: 0.000 (bassa variabilità)
- **Deviazione standard**: 0.008 (stabilità tra i diversi fold)

Questi risultati indicano che il modello è stabile e generalizza bene sui dati di test.

**Conclusione**

Il modello Random Forest ha raggiunto un'accuracy del 91%, dimostrando una buona capacità complessiva di classificare correttamente i dati. Questo risultato è stato ulteriormente confermato dalla cross-validation, che ha evidenziato la stabilità delle prestazioni attraverso diverse suddivisioni dei dati.

Tuttavia, l'analisi più approfondita delle metriche per le singole classi ha rivelato che l'F1-score per la classe 1 (0.48) è inferiore rispetto a quello della classe maggioritaria. Questo punteggio indica che il modello ha ancora margini di miglioramento nel riconoscimento dei casi di dislessia, suggerendo una difficoltà nel distinguere correttamente i soggetti dislessici.

Il bilanciamento delle classi applicato al dataset ha contribuito a ridurre il numero di falsi positivi, mantenendo al contempo una buona capacità di distinzione tra soggetti con e senza dislessia. Tuttavia, la sfida di migliorare la sensibilità del modello verso la classe minoritaria rimane.

## 4.8 Support Vector Machines (SVM)

---

L’algoritmo Support Vector Machines (SVM) è un modello di classificazione che si basa sull’idea di trovare un iperpiano che separi al meglio un set di dati in due classi. I **Support Vector** sono i punti dati più vicini all'iperpiano e rappresentano gli elementi più influenti nella definizione del margine di separazione.

### 4.8.1 Decisioni di progetto

Questo modello è stato scelto per la sua capacità di gestire problemi di classificazione binaria, in particolare quando le classi sono ben separate. Inoltre, è adatto per dataset con un numero moderato di feature e può gestire relazioni non lineari attraverso l'uso di kernel appropriati.

Per questo motivo, abbiamo scelto di utilizzare il **kernel RBF (Radial Basis Function)**, in quanto consente di modellare strutture complesse e non lineari nei dati. Questo kernel permette di migliorare le probabilità di ottenere un modello di classificazione più accurato e generalizzabile rispetto a un semplice modello lineare.

### 4.8.2 Ottimizzazione degli iperparametri con Grid Search

L’ottimizzazione degli iperparametri è stata effettuata utilizzando la tecnica della **Grid Search** in combinazione con la **cross-validation**. Questa strategia ha permesso di individuare i migliori valori per i parametri **C**, **gamma** e **kernel**, migliorando così le prestazioni del modello.

```
[16] from sklearn.model_selection import GridSearchCV

     # Definizione della griglia di parametri da ottimizzare per Grid Search
     param_grid = {
         'C': [0.1, 1, 10, 100],              # Parametro di regolarizzazione
         'gamma': [0.001, 0.01, 0.1, 1],      # Parametro del kernel
         'kernel': ['rbf', 'linear']          # Tipi di kernel
     }
```

La ricerca è stata condotta su una griglia di valori, testando diverse combinazioni:

- **Parametro C**: valori testati nell'intervallo [0.1, 1, 10, 100] per analizzare il trade-off tra complessità del modello e generalizzazione.
    - Valori bassi di C favoriscono un margine più ampio e un modello più semplice, tollerante agli errori.
    - Valori alti di C cercano di minimizzare gli errori di classificazione, ma possono portare a overfitting.

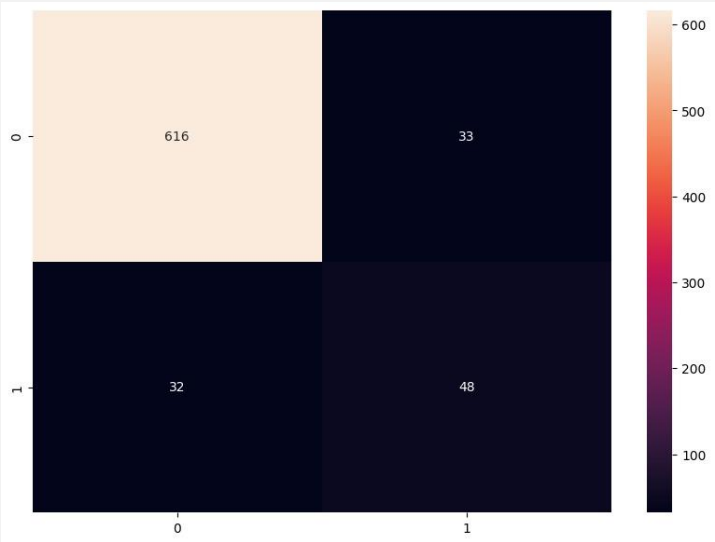
- **Parametro gamma**: valori testati nell'intervallo [0.0001, 0.001, 0.01, 0.1, 1, 10, 100] per controllare l'influenza di un singolo esempio di addestramento sulla funzione decisionale.
    - Valori bassi implicano un'influenza più ampia di ogni punto dati, portando a decisioni più generali.
    - Valori alti rendono il modello più sensibile ai dati di addestramento, con il rischio di overfitting.
- **Kernel**: sono stati testati sia **lineare** che **RBF**.
    - Il kernel lineare è utile per dati separabili linearmente.
    - Il kernel RBF permette di modellare relazioni più complesse tra le feature.

Dopo l'esecuzione della **Grid Search**, il miglior modello è stato selezionato in base alle prestazioni ottenute sui dati di validazione, utilizzando il metodo grid_search.best_estimator_.

### 4.8.3 Valutazione Finale

La valutazione del modello finale è stata condotta utilizzando diverse metriche di performance per comprendere la sua efficacia nella classificazione.

**Matrice di Confusione**



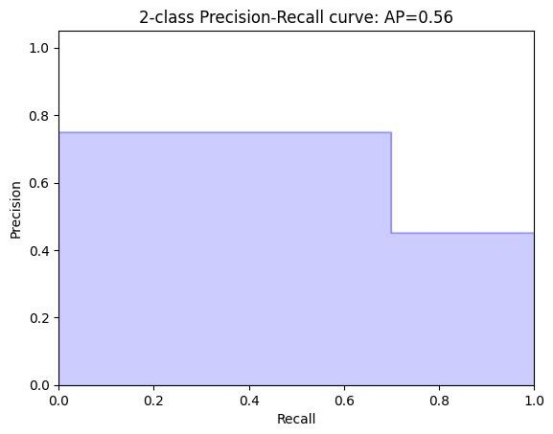

mostra la distribuzione delle previsioni corrette ed errate. Nonostante alcuni errori di classificazione, il modello ha ottenuto un buon livello di accuratezza.

**Classification Report**:

```
SVM Classification Report:
              precision    recall  f1-score   support

           0       0.85      0.95      0.90       649
           1       0.75      0.60      0.67        80

    accuracy                           0.84       729
   macro avg       0.80      0.78      0.78       729
weighted avg       0.84      0.84      0.84       729

Confusion Matrix:
[[616  33]
 [ 32  48]]
```

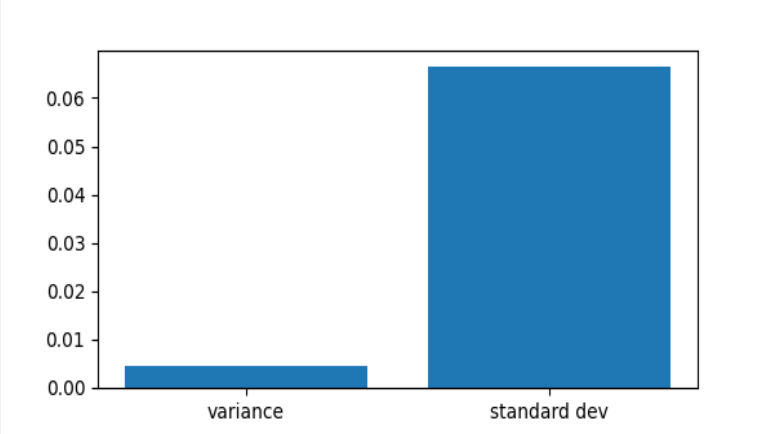
Accuratezza Globale: Il modello nel suo complesso è accurato nell'84% dei casi.
Classe 0 (più frequente): Predice molto bene questa classe (precisione 85%, recall 95%).
Classe 1 (meno frequente): Ha più difficoltà a riconoscere questa classe (precisione 75%, recall 60%).

**Precision-Recall Curve**




Analizza il bilanciamento tra precisione e recall, confermando l'affidabilità delle previsioni.

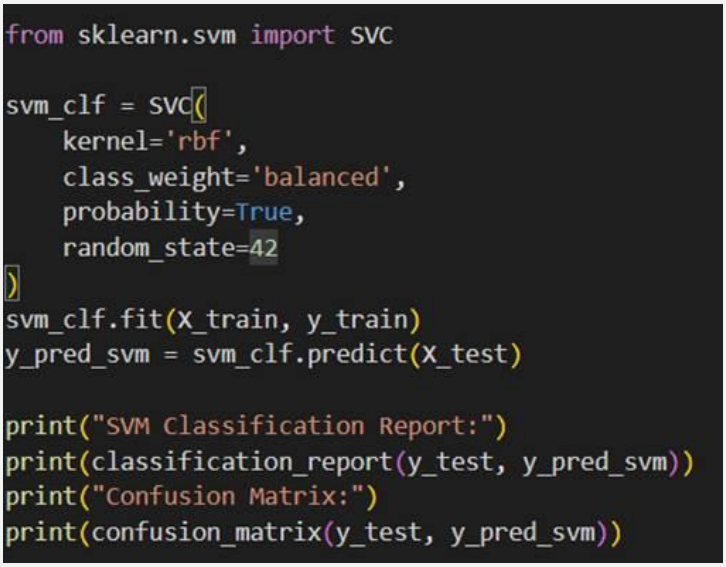
**Bar Chart di Varianza e Deviazione Standard:**



La bassa varianza indica che il modello è stabile e coerente nelle sue predizioni. Una leggera deviazione standard evidenzia variazioni tra i diversi fold della cross-validation, indicando che il modello ha una certa sensibilità ai dati.

**Conclusione**

Il modello SVM con kernel RBF si è dimostrato una scelta adeguata al problema affrontato, riuscendo a cogliere pattern non lineari nei dati e migliorando la capacità di generalizzazione rispetto a un modello lineare. L'ottimizzazione degli iperparametri tramite Grid Search ha permesso di ottenere un modello bilanciato, riducendo il rischio di overfitting e migliorando la performance globale della classificazione.



```
from sklearn.svm import SVC

svm_clf = SVC(
    kernel='rbf',
    class_weight='balanced',
    probability=True,
    random_state=42
)
svm_clf.fit(X_train, y_train)
y_pred_svm = svm_clf.predict(X_test)

print("SVM Classification Report:")
print(classification_report(y_test, y_pred_svm))
print("Confusion Matrix:")
print(confusion_matrix(y_test, y_pred_svm))
```

## 4.9 Neural Network

---

Le reti neurali artificiali simulano il funzionamento dei neuroni biologici attraverso neuroni artificiali collegati tra loro. Questi neuroni elaborano gli input assegnando loro pesi, e la loro attivazione dipende dal superamento di una soglia prestabilita.

- I pesi determinano l'importanza di ciascun input nel processo decisionale.
- L'attivazione avviene se il valore elaborato supera una soglia definita.

Questo modello è stato scelto per la sua capacità di affrontare problemi complessi di apprendimento e per l'elevata accuratezza nelle previsioni.

### 4.9.1 Decisioni di Progetto

Il modello di rete neurale artificiale è stato scelto per la sua capacità di affrontare problemi complessi di apprendimento e per l'elevata accuratezza nelle previsioni. Abbiamo optato per una rete neurale sequenziale con due livelli principali:

- **Primo livello (Hidden Layer):** 30 neuroni nascosti con funzione di attivazione ReLU, che introduce non-linearità e migliora la capacità di apprendimento del modello senza causare overfitting.
- **Secondo livello (Output Layer):** un singolo neurone con funzione di attivazione sigmoide, che permette di ottenere un output compreso tra 0 e 1 per la classificazione binaria.

Questa configurazione consente al modello di apprendere in maniera efficace, distinguendo con precisione tra le due classi del dataset.

### 4.9.2 Creazione Modello e Addestramento

Il modello di rete neurale è stato implementato utilizzando il framework Keras.

Il codice seguente mostra la definizione della rete:

```python
from keras.models import Sequential
from keras.layers import Dense
#----------------------------------------------------------------------------------------
# Creazione modello della rete neurale

def create_model(input_dim):
    network = Sequential()
    network.add(Dense(30, input_dim=input_dim, activation="relu"))
    network.add(Dense(1, activation='sigmoid'))
    network.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
    return network

#----------------------------------------------------------------------------------------
```

**Spiegazione del codice:**

- **Sequential()**: definisce una rete neurale sequenziale in Keras.
- **Dense(30, activation="relu")**: aggiunge un livello completamente connesso con 30 neuroni e attivazione ReLU.
- **Dense(1, activation='sigmoid')**: aggiunge un'unità di output con attivazione sigmoide per la classificazione binaria.
- **Compilazione del modello**: utilizza la funzione di perdita binary_crossentropy, ideale per la classificazione binaria, e l'ottimizzatore adam, scelto per la sua efficienza in vari scenari.

Una volta definito il modello, si procede con l'addestramento:

```
# Creazione e addestramento del modello
model = create_model(X_train.shape[1])
print(model)
model.fit(X_train, y_train, epochs=30, batch_size=64)
```

- **Epochs = 30**: il numero di epoche stabilisce quante volte il modello analizzerà l'intero dataset di training.
- **Batch size = 64**: definisce il numero di istanze processate prima dell'aggiornamento dei pesi, bilanciando precisione ed efficienza computazionale.

### 4.9.3 Valutazione Finale

Dopo l'addestramento, il modello è stato valutato sui dati di test e validazione.

- **Validation Accuracy:** 88.45%, con una loss di 0.3047.
- **Test Accuracy:** 86.04%, con performance stabili rispetto alla validazione.

**Metriche di Classificazione:**

Clasification report:

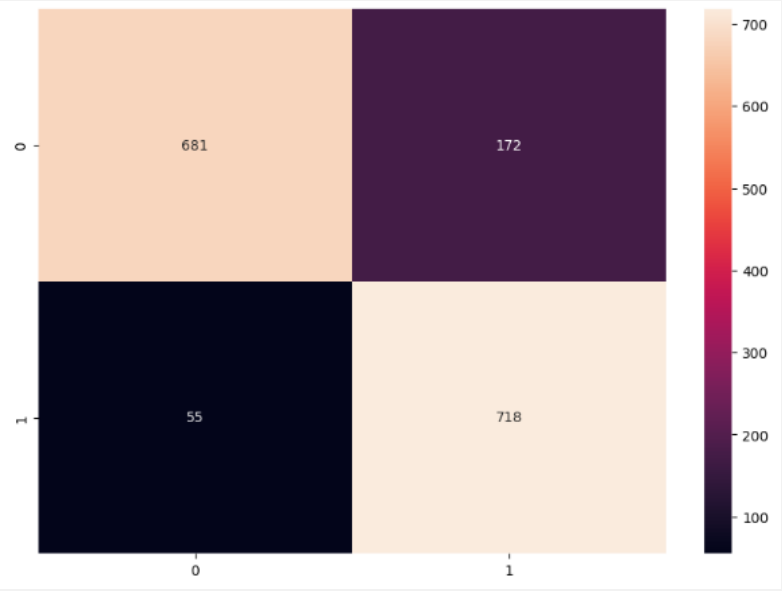
| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.93 | 0.80 | 0.86 | 853 |
| 1 | 0.81 | 0.93 | 0.86 | 773 |
| accuracy | | | 0.86 | 1626 |
| macro avg | 0.87 | 0.86 | 0.86 | 1626 |
| weighted avg | 0.87 | 0.86 | 0.86 | 1626 |

**Interpretazione:**

- **Precisione Classe 0 (0.93)**: pochi falsi positivi per la classe 0.
- **Recall Classe 1 (0.93)**: buona identificazione della classe 1 con pochi falsi negativi.
- **F1-score medio (0.86)**: equilibrio ottimale tra precisione e recall.

## Grafici di Valutazione

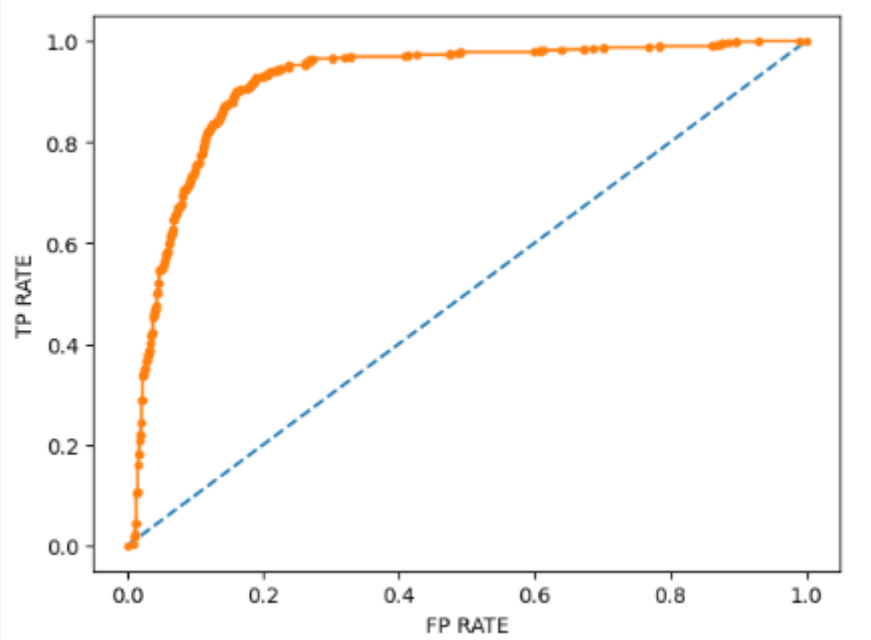
**Confusion Matrix:**



- **681 veri positivi** (classe 0 correttamente classificata).
- **718 veri negativi** (classe 1 correttamente classificata).
- **172 falsi positivi** (erroneamente classificati come classe 1).
- **55 falsi negativi** (erroneamente classificati come classe 0).

Il modello mostra una leggera difficoltà nella distinzione della classe 0, evidenziata da un numero maggiore di falsi positivi rispetto ai falsi negativi.

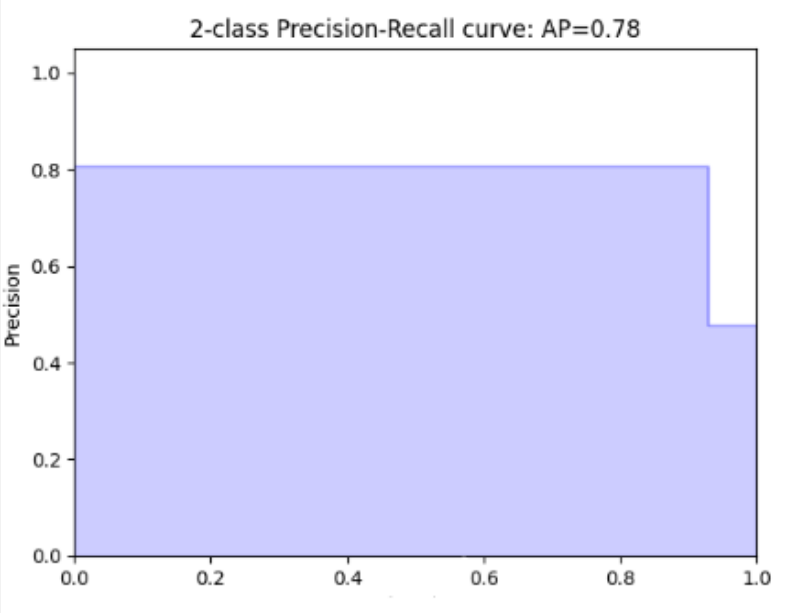
**ROC Curve**:




- La curva ROC è **ben al di sopra della diagonale** (baseline), indicando che il modello ha una **buona capacità predittiva**.
- **Il rapido aumento iniziale** suggerisce un modello con un'alta sensibilità e pochi falsi negativi.
- La curva **si appiattisce vicino al punto (1,1)**, indicando che il modello è in grado di classificare correttamente quasi tutte le istanze positive senza molti falsi positivi.

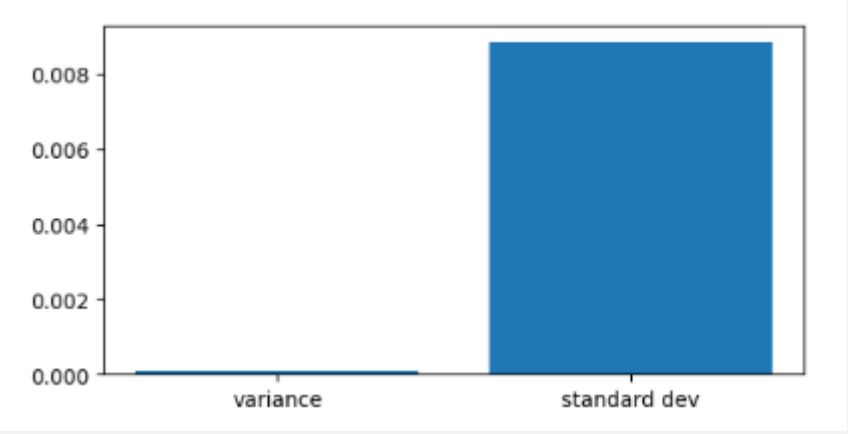
Il grafico della **ROC Curve** mostra quindi che il modello possiede **un'ottima capacità di discriminazione** tra le due classi, bilanciando **sensibilità (recall) e specificità**, e riducendo al minimo il numero di classificazioni errate.

**Precision-Recall Curve**:




- La deviazione standard è significativamente più elevata rispetto alla varianza.
- Questo indica che i dati sono distribuiti in modo ampio rispetto alla media, ma la varianza complessiva è molto bassa, suggerendo una stabilità relativa del modello.
- Il modello può avere una capacità predittiva consistente, ma con una distribuzione dei dati che si estende oltre il valore centrale.

**Bar Chart di Varianza e Deviazione Standard:**

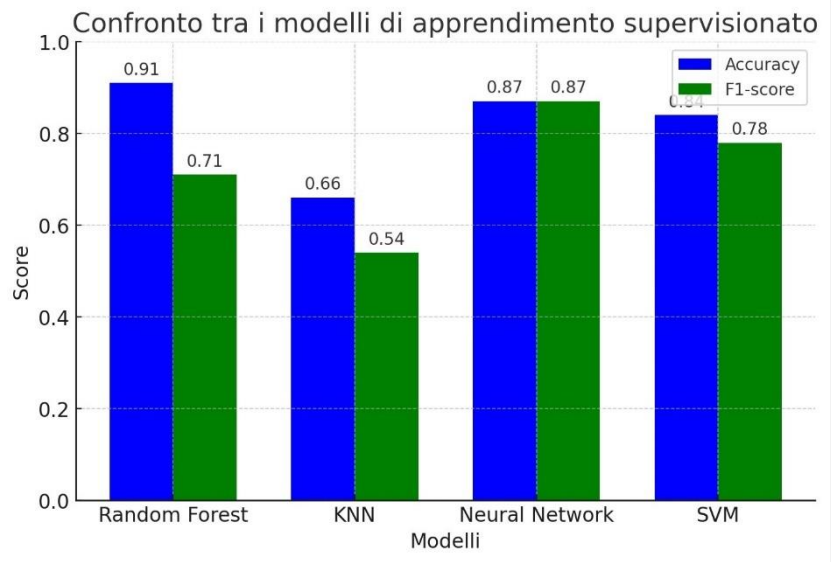



- La deviazione standard è significativamente più elevata rispetto alla varianza.
- Questo indica che i dati sono distribuiti in modo ampio rispetto alla media, ma la varianza complessiva è molto bassa, suggerendo una stabilità relativa del modello.
- Il modello può avere una capacità predittiva consistente, ma con una distribuzione dei dati che si estende oltre il valore centrale.

## 4.10 Conclusioni

Abbiamo applicato una **cross-validation a 5 fold** per ottenere metriche più affidabili. I risultati ottenuti sono stati:

| Modello | Accuratezza Media |
|---|---|
| KNN | 66% |
| Random Forest | 91% |
| SVM | 84% |
| Neural Network | 87% |




**Random Forest: Ottima Accuracy, ma Potenziale Squilibrio**

- Accuracy: 0.91 (la più alta tra tutti i modelli)
- F1-score: 0.71 (più basso rispetto a Neural Network e SVM)

**Interpretazione:**

- L'alta accuracy suggerisce che il modello classifica bene la maggior parte dei dati.
- Tuttavia, il F1-score inferiore indica un possibile squilibrio: Random Forest classifica molto bene la classe maggioritaria, ma ha difficoltà con la classe minoritaria.
- Il report di classificazione conferma che la precision per la classe 1 è bassa (0.62), suggerendo che il modello fatica a individuare correttamente i falsi negativi.

---

**KNN: Scarse Prestazioni, Sensibile ai Dati**

- Accuracy: 0.66 (la più bassa tra tutti i modelli)
- F1-score: 0.54 (indicando prestazioni mediocri)

**Interpretazione:**

- KNN non riesce a generalizzare bene sui dati. Questo è prevedibile perché KNN è molto sensibile al rumore e alla dimensionalità dei dati.
- La recall per la classe 1 è relativamente alta (0.63), ma la precision è solo 0.19, indicando che classifica molti falsi positivi.

---

**Neural Network: Modello Più Bilanciato**

- Accuracy: 0.87
- F1-score: 0.87 (equilibrato e alto)

**Interpretazione:**

- Il modello più bilanciato tra tutti, con punteggi elevati sia in accuracy che in F1-score.
- Funziona bene per entrambi le classi, con un buon compromesso tra precision e recall.
- Suggerisce che il dataset beneficia di una modellazione più complessa e che i pattern nei dati sono non-lineari.

---

**SVM: Buon Compromesso tra Accuratezza e Generalizzazione**

- Accuracy: 0.84
- F1-score: 0.78

**Interpretazione:**

- Un buon compromesso tra accuratezza e generalizzazione.
- La precision per la classe 1 è **0.75** e la recall **0.60**, suggerendo che SVM è meno incline a falsi negativi rispetto a Random Forest.
- Tuttavia, è meno performante rispetto alla Neural Network.

---

**Conclusioni Generali e Scelta del Modello**

Dai risultati, possiamo concludere che:

1. Neural Network è il modello migliore in termini di bilanciamento tra accuratezza e F1-score, rendendolo la scelta più robusta.
2. Random Forest è molto forte in accuracy, ma potrebbe soffrire di squilibrio tra le classi.
3. SVM è una scelta solida, con buoni compromessi e margini di miglioramento.
4. KNN è il meno adatto, avendo difficoltà a generalizzare bene.

Se l'obiettivo è ottenere un modello altamente performante e bilanciato, Neural Network è la scelta migliore. Se invece vogliamo un modello interpretabile e robusto, possiamo considerare Random Forest o SVM, con le giuste ottimizzazioni.

# 5. CLUSTERING

Dopo aver esplorato i metodi di apprendimento supervisionato, abbiamo ritenuto che l'applicazione del clustering potesse offrire un ulteriore approccio per comprendere meglio la struttura intrinseca dei dati sulla dislessia. Questa tecnica ci consente di identificare naturali raggruppamenti nei dati, anche senza etichette di classe, e ci permette di rivelare relazioni sottostanti tra le prestazioni nei test cognitivi e comportamentali, fornendo una prospettiva diversa sulla distribuzione dei dati. Il clustering può essere di due tipi:

- **Clustering rigido (Hard clustering)**: assegna ciascun esempio a un unico cluster specifico.
- **Clustering morbido (Soft clustering)**: utilizza distribuzioni di probabilità per assegnare le classi associate a ciascun esempio.

## 5.1 Decisioni di progetto

Abbiamo scelto il clustering rigido, in particolare il k-means, per raggruppare i soggetti in categorie basate sulle caratteristiche presenti nel file Dyslexia_dataset.csv. L'obiettivo è individuare dei cluster con centroidi calcolati automaticamente, creando una nuova colonna nel dataset con l'assegnazione dei cluster. Il nuovo dataset, Dyslexia_Dataset_Clustered.csv, potrebbe aprire nuove prospettive sulla dislessia e la sua identificazione.ù

## 5.2 K-means

K-means è un algoritmo di clustering che suddivide un insieme di dati in **K cluster**, dove K è un valore predefinito. Inizia posizionando casualmente K centroidi nel dataset e assegna ciascun punto al centroide più vicino. Successivamente, aggiorna la posizione dei centroidi in base alla media dei punti assegnati e ripete il processo fino a convergenza. L'obiettivo è minimizzare la somma dei quadrati delle distanze tra i punti e i centroidi assegnati, creando cluster di dati simili.

Dato che parte delle caratteristiche nel dataset sono di natura categorica e l'algoritmo k-means accetta solo caratteristiche numeriche, è stato necessario convertirle. A tal fine, abbiamo scelto di utilizzare la tecnica di **one-hot encoding**, che crea una nuova variabile binaria univoca per ogni categoria.

Una volta codificate le variabili categoriche, abbiamo dovuto determinare il numero ottimale di cluster. Abbiamo utilizzato la tecnica del **metodo del gomito (elbow method)**, in cui sull'asse delle ordinate sono rappresentati i valori delle somme dei quadrati intra-cluster (**WCSS**), mentre sull'asse delle ascisse sono rappresentati i K cluster.

Osservando il grafico del gomito, abbiamo notato che il punto di svolta netto si trova intorno a **K=4**, suggerendo che questo sia il numero ottimale di cluster.

Abbiamo testato diversi valori di **K**:

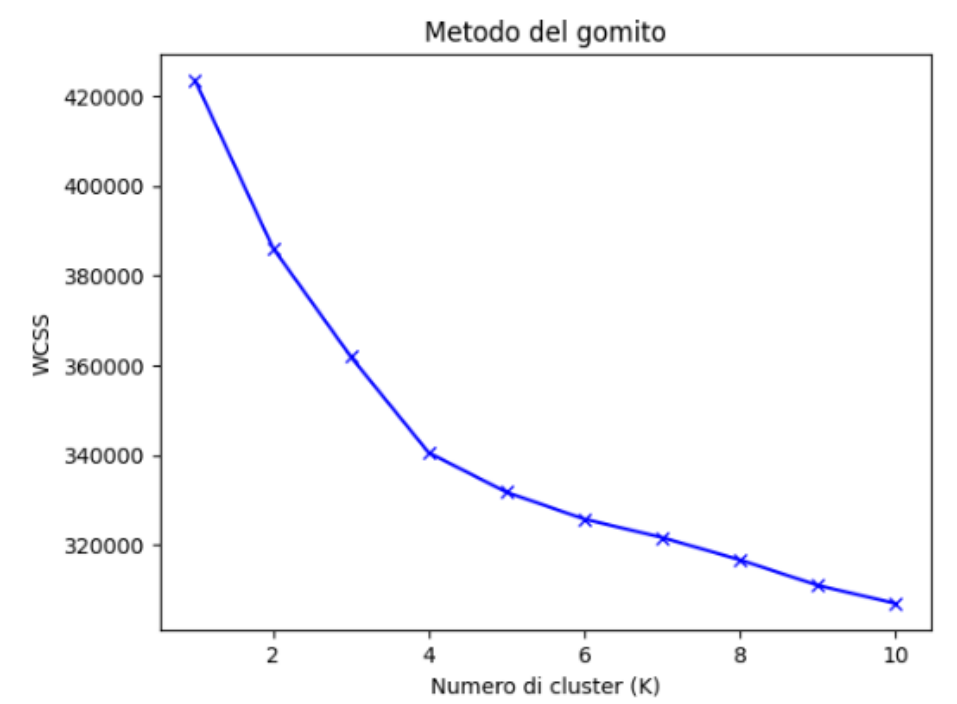
- Con **K=3**, il **Silhouette Score** ottenuto è stato **0.102**.
- Con **K=4**, il **Silhouette Score** è salito a **0.115**, indicando una migliore separazione tra i cluster.
- Con **K=5**, il **Silhouette Score** è sceso a **0.082**, suggerendo una minore qualità dei cluster.

Poiché **K=4** ha ottenuto il miglior punteggio in termini di separazione dei cluster, è stato scelto come valore ottimale per il clustering.

Le metriche scelte per la configurazione del modello sono state:

- **n_clusters=4**: il numero di cluster in cui il dataset deve essere diviso.
- **random_state=42**: questo parametro controlla la riproducibilità dei risultati. Fissando un valore per random_state, il modello fornirà sempre gli stessi risultati quando viene addestrato con gli stessi dati.
- **n_init=10**: specifica il numero di volte che l'algoritmo viene eseguito con diverse inizializzazioni casuali dei centroidi. L'algoritmo seleziona la soluzione migliore tra i tentativi basandosi sulla somma dei quadrati delle distanze. È stato scelto un valore non troppo alto, ma sufficiente per massimizzare la qualità dell'output.

**Grafico:**



## 5.3 Valutazione Finale

Sono state valutate le prestazioni del modello utilizzando due metriche:

- **WCSS (Within-Cluster Sum of Squares)**: calcola la somma dei quadrati delle distanze di ogni punto rispetto al proprio centroide di cluster. Un valore più basso di WCSS indica una maggiore coesione all'interno dei cluster.
- **Silhouette Score**: valuta la coesione all'interno dei cluster e la separazione tra i cluster. È calcolato per ciascun punto e rappresenta il rapporto tra la distanza media al proprio cluster e la distanza media ai cluster più vicini. Un punteggio più alto del Silhouette Score indica una migliore separazione dei cluster.

Nel nostro caso, il **Silhouette Score ottenuto è pari a 0.115**, il che suggerisce che i cluster identificati sono ben separati e possono fornire indicazioni utili sulla struttura dei dati.

## 5. CONCLUSIONI

In questo progetto, abbiamo affrontato il problema della dislessia applicando tecniche di intelligenza artificiale per analizzare e segmentare i dati derivanti da test cognitivi e comportamentali. Lo scopo principale era identificare pattern nascosti e ottenere una comprensione più approfondita della dislessia, con l'obiettivo finale di migliorare il supporto diagnostico.

Attraverso l'applicazione del **clustering rigido (K-Means)**, il dataset è stato suddiviso in quattro cluster significativi. L'analisi ha rivelato che specifiche caratteristiche cognitive e linguistiche influenzano la categorizzazione dei soggetti, suggerendo che esistono gruppi distinti con differenti profili di prestazioni. Sebbene il **Silhouette Score** indichi una separazione non perfetta tra i cluster, i risultati mostrano il potenziale dell'algoritmo nel rilevare strutture latenti nei dati complessi legati ai disturbi dell'apprendimento.

L'analisi dei modelli di apprendimento supervisionato ha evidenziato che:

- La **Neural Network** si è dimostrata la scelta migliore in termini di bilanciamento tra accuratezza e F1-score, offrendo una classificazione robusta ed equilibrata tra le classi.
- **Random Forest** ha raggiunto la massima accuratezza (91%), ma ha mostrato squilibri tra le classi.
- **SVM** ha fornito un buon compromesso tra accuratezza e generalizzazione, mentre **KNN** ha dimostrato prestazioni inferiori, evidenziando difficoltà nel generalizzare i dati complessi.

Questi risultati indicano che i dati sulla dislessia presentano pattern complessi e non lineari, beneficiando quindi di modelli più sofisticati.

## 6.1 Sviluppi futuri

Il lavoro svolto offre molteplici spunti per futuri sviluppi e miglioramenti:

1. **Clustering morbido:** L'integrazione di tecniche di clustering morbido (soft clustering) potrebbe migliorare l'identificazione delle sovrapposizioni tra gruppi, offrendo una rappresentazione più accurata delle differenze individuali.
2. **Combinazione con modelli supervisionati:** Una direzione promettente sarebbe combinare i risultati del clustering con modelli di apprendimento supervisionato, utilizzando i pattern individuati per migliorare la predizione della dislessia.
3. **Espansione del dataset:** Ampliando il dataset con nuove variabili o integrando dati provenienti da altre fonti (ad esempio, dati genetici, neurofisiologici o ambientali), si potrebbero migliorare ulteriormente le prestazioni e l'accuratezza delle analisi.

4. **Applicazione ad altri disturbi dell’apprendimento:** La metodologia sviluppata potrebbe essere estesa per analizzare altri disturbi dell’apprendimento, come la **discalculia** o il **disturbo da deficit di attenzione e iperattività (ADHD)**, contribuendo alla creazione di nuovi strumenti diagnostici basati sull'intelligenza artificiale.

Questi sviluppi futuri potrebbero portare a sistemi diagnostici più precisi, personalizzati e automatizzati, fornendo un contributo significativo al miglioramento delle diagnosi e del supporto per i disturbi dell'apprendimento.