TwoTheRoot

https://github.com/BiBooBap/TwoTheRoot

Laurea di Informatica L-31 all'Università di Salerno

Corso di Machine Learning

Created by:
Antonio Maiorano
Silvana De Martino

Supervised by:
Prof. Giuseppe Polese
Prof. Loredana Caruccio

Indice

CHAPTER 1

Definizione del problema

1.1 Introduzione

TwoTheRoot è un progetto che si propone come un supporto per gli specialisti in ambito psichiatrico e psicologico; il suo scopo è individuare possibili trigger che potrebbero scatenare un attacco di panico in un soggetto.

1.1.1 MachineLearning e Psicologia

Sulla base dell'esperienza acquisita dal modello si vogliono individuare pattern, all'apparenza nascosti, che possano permettere di definire una possibile causa scatenante con una precisione discreta, in modo da poter aiutare specialisti e pazienti a risolvere il problema alla radice. Gli attacchi di panico sono un problema che in qualche modo, al giorno d'oggi, riguarda tutti, dai più piccoli fino agli anziani. Dunque, riuscire a riconoscerne le cause è un modo per evitare che tali problematiche diventino invalidanti nella vita quotidiana di chi ne soffre; questo potrebbe essere possibile cercando di limitare l'impiego di farmaci tramite l'apporto di un forte sostegno alla professione medica dedicata a tale materia. Lo scopo del modello non è quello di andare a sostituire la figura dello specialista, bensì di offrire un saldo supporto in fase di diagnosi ed analisi della problematica, garantendo allo specialista stesso la possibilità di avere un riscontro immediato, ma continuando a rispettare tutte le norme a tutela del paziente relative alla privacy e alla correttezza deontologica del medico stesso. Tramite un'analisi delle sintomatologie e di altri aspetti chiave, ci si pone lo scopo di individuare un pattern ricorrente nei soggetti in esame e di riconoscere la radice del problema. Il modello potrà essere impiegato solamente da figure specializzate nel campo, evitando in questo modo di finire nelle mani di persone non esperte in materia. Le previsioni ottenute saranno unicamente un supporto per gli esperti e avranno sempre bisogno di una diagnosi di conferma da parte di un professionista. Per questo motivo, come appellativo, è stato scelto il gioco di parole TWO invece di TO, in modo da sottolineare ulteriormente che il modello si propone come assistente e non come specialista lui stesso; infatti, un uso improprio potrebbe causare delle auto-diagnosi improprie ed inesatte, come accade spesso con i più comuni mezzi di ricerca.

1.1.2 Attacchi di panico

Gli attacchi di panico sono un problema che ad oggi affligge un numero sempre maggiore di persone, soprattutto adolescenti e giovani adulti. Le cause di questa crescita si possono trovare in vari aspetti della società moderna, dalla presenza dei social media nella vita quotidiana e di conseguenza con il costante confronto con dei modelli inarrivabili, al deterioramento delle posizioni socio-lavorative o, ancora, all'incertezza del futuro economico e ambientale che pone dei seri punti di domanda alle nuove generazioni sul loro futuro. In questo grande marasma i giovani, così come gli adulti, si sentono sempre più disorientati e oppressi, e questo sta portando ad un rapido calo nella qualità delle loro condizioni psicologiche rispetto alle generazioni precedenti, come quella dei "Boomer". Un attacco di panico è generalmente definito come un episodio improvviso di intensa paura o disagio, accompagnato da una serie di sintomi fisici e psicologici. Questi episodi possono verificarsi apparentemente senza un motivo definito o, al contrario, essere scatenati da specifiche situazioni o pensieri.

1.1.3 La soggettività dei dati

Il cervello umano ha una struttura estremamente complessa e ancora in gran parte sconosciuta. Per quanto la psicologia moderna stia facendo passi da gigante, molti dei farmaci prescritti in ambito psichiatrico non risultano sempre adatti a quello scopo, o per lo meno, non per tutti si ha ancora perfettamente coscienza di come operino in maniera specifica sui recettori, nella maggior parte dei casi sono stati semplicemente osservati risposte positive dei pazienti all'assunzione, tralasciando le implicazioni negative. Inoltre, le cause che possono portare ad un attacco di panico, così come allo sviluppo di malattie mentali ben più gravi, sono estremamente soggettive e variano in base alle esperienze di vita del singolo soggetto. Questo rende ancor più complesso interfacciarsi all'argomento, e mette gli sviluppatori e i progettisti nella posizione di dover individuare dei pattern che si possano considerare quanto più oggettivi possibile. Lo sviluppo di progetti informatici in questo campo è quindi reso complesso soprattutto dalla necessità di considerare sempre il fattore soggettivo che potrebbe rendere una previsione all'apparenza perfetta, nell'effettivo sbagliata. I dati raccolti, essendo di matrice esclusivamente umana, pur se estratti in ambito medico, portano con sé un

enorme bias, ovvero quello della poca conoscenza medica del cervello umano e di ciò che lo controlla.

1.1.4 Soluzione del problema

Sulla base dell'esperienza del modello si ha lo scopo di individuare pattern all'apparenza nascosti che possano permettere di definire una possibile causa generante con una precisione discreta in modo da poter aiutare specialisti e pazienti a risolvere il problema alla radice. Gli attacchi di panico sono un problema che in qualche modo, al giorno d'oggi, riguardano una crescente fetta della popolazione, e riuscire a riconoscerne le cause è un modo per evitare che tali problematiche diventino invalidanti nella vita quotidiana di chi ne soffre. La limitazione dell'impiego di farmaci potrebbe essere una felice conseguenza del sostegno fornito dall' AI alle professioni mediche dedicate. Chiaramente lo scopo del modello non sarà quello di andare a sostituire la figura dello specialista, ma bensì quello di offrire un saldo supporto in fase di diagnosi ed analisi della problematica, garantendo allo stesso la possibilità di avere un confronto immediato, continuando però a rispettare tutte le norme a tutela del paziente relative alla privacy e alla correttezza deontologica del medico stesso. Tramite un'analisi delle sintomatologie e di altri aspetti chiave quindi ci si pone lo scopo di individuare un pattern nei soggetti affetti e di riconoscere la radice del problema. Il modello si propone però di essere impiegato solamente da figure specializzate nel campo, evitando così di finire nelle mani di persone meno esperte. Le previsioni ottenute avranno chiaramente sempre bisogno della conferma della diagnosi da parte di un esperto, ed è per questo che è stato scelto il gioco di parole TWO invece di TO, in quanto il modello si propone come assistente e non come specialista lui stesso, un uso improprio potrebbe, come capita spesso con i più comuni mezzi di ricerca, causare delle auto-diagnosi improprie ed insatte.

CHAPTER 2

Data Understanding & Data Preparation

2.1 Data Gathering

Il dataset è stato individuato tra una serie di dataset disponibili online, ed è stato scelto sulla base del confronto tra altri riguardanti lo stesso ambito,considerando le valutazioni fatte da altri utenti, il numero di download e la chiarezza delle feature oltre che una considerazione sull'impiego delle stesse in ambito pratico e di addestramento. Il nome del dataset è : ***Panic Attack Dataset : A dataset capturing key factors related to panic attacks, including demographics***. Il dataset si presenta in formato csv scaricabile, presenta un punteggio di usabilità del 10.00 e la licenza è fornita dal **MIT**. Un altro punto a favore di questo dataset è il fatto che la sua creazione sia estremamente recente: 01/17/2024 [1]. Il numero totale di features presentate è di **21**, il numero di samples è di **1199**.

2.2 Data Examination

Si dedicherà un breve spazio alla descrizione delle feature presenti all'interno del dataset in questione per rendere chiare ed evidenti le considerazioni che verranno fatte di seguito. Le feature presentano tipi diversi: continui che sono in totale il 47% della totalità delle feature, il che è un aspetto positivo ma verrà richiesta una sostanziale normalizzazione in futuro, categorici, il 14%, binari il 38%. Le variabili binarie presentano come valore assumibile Yes/No, e si possono considerare binarie traducendo i valori in 0/Yes e 1/No, o possibilmente anche in booleani. Le feature sono le seguenti:

1. **ID** – Identificatore univoco del sample, è un valore continuo;
2. **Age** – Età del paziente di cui si sono raccolti i dati del sample;
3. **Gender** – Non è indicatore del sesso biologico, bensì del genere, tiene conto di tre categorie : Male, Female, Non-binary. Il valore Non-binary è una categoria inserita per definire della rappresentazione di genere meno specifica per consentire una maggiore integrazione.

[1] si sottolinea che tale documento ha visto l'inizio della sua stesura in data 01/24/2025 per dare contesto alla considerazione.

4. **Panic_Attack_Frequency** – Numero di attacchi di panico per mese. Il range di valori si estende tra lo 0 e il 9, lo 0 non indica una totale assenza di attacchi di panico, ma un soggetto che non soffre di questa problematicità in maniera costante o clinica, che può aver presentato l'attacco di panico non perchè legato ad un disturbo ma piuttosto per un evento specifiche.

5. **Duration_Minutes** - Durata media dell'attacco di panico nel soggetto. Il range dei valori si estende da 5 a 44. La feature è di tipo continuo

6. **Trigger** – Motivo principale che porta allo scatenarsi dell'attacco di panico nel soggetto (e.g., stress, social anxiety, phobia, PTSD, caffeine, unknown), la feature è categorica;

7. **Heart_Rate** – Battito cadiaco registrato durante l'attacco di panico(bpm). Il range dei valori assumibili dalla variabile della feature va da 80 a 159;

8. **Sweating** – Indica la presenza di sudorazione durante l'attacco di panico. La feature è binaria;

9. **Shortness_of_Breath** – Indica la mancanza di fiato, quindi difficoltà respiratorie, durante l'attacco di panico. La feature è binaria;

10. **Dizziness** – Indica la presenza di vertigini durante l'attacco di panico. La feature è binaria;

11. **Chest_Pain** – Indica la presenza di dolori al petto, sintomo che viene accostato all'attacco cardiaco durante l'attacco di panico. La feature è binaria;

12. **Trembling** – Indica la presenza di tremori durante l'attacco di panico. La feature è binaria;

13. **Medical_History** – Condizioni pre-esistenti nella storia medica del paziente (e.g., Anxiety, Depression, PTSD)

14. **Medication** – Indica l'assunzione regolare o meno da parte del paziente di farmaci per l'ansia, presupponendo una possibile presenza di condizioni pre-esistenti. La feature è binaria;

15. **Caffeine_Intake** – Numero di tazze di the o caffè nel corso della giornata. Il valore della variabile associata alla feature è continuo e il range va dal valore di 0 a 5;

16. **Exercise_Frequency** – Giorni in cui il soggetto pratica esercizio nel corso della settimana. Il valore della variabile associata alla feaure è continuo e il range va da un valore di 0 a 6;

17. **Sleep_Hours** – Media delle ore di sonno per notte del paziente. La variabile associata alla feature è continua e il range di valori assumibili va da 4 a 9;

18. **Alcohol_Consumption** – Numero di alcolici consumati nel corso di una settimana. La variabile associata alla feature è di tipo continuo e il range di valori assubili va da 0 a 9;

19. **Smoking** – Indica se il soggetto è fumatore. La variabile associata alla feature è di tipo binario.

20. **Therapy** – Indica se il soggetto va in terapia. La variabile associata alla feature è di tipo binario.

21. **Panic_Score** – Definisce la gravità dell'attacco di panico. La variabile associata alla feature è di tipo continuo e il range di valori assumibili va da 1 a 10.

Feature	Data Type
Age	int64
Gender	object
Panic_Attack_Frequency	int64
Duration_Minutes	int64
Trigger	object
Heart_Rate	int64
Sweating	bool
Shortness_of_Breath	bool
Dizziness	bool
Chest_Pain	bool
Trembling	bool
Medical_History	object
Medication	bool
Caffeine_Intake	int64
Exercise_Frequency	int64
Sleep_Hours	float64
Alcohol_Consumption	int64
Smoking	bool
Therapy	bool
Panic_Score	int64

Table 2.1: Features e i loro data types

2.3 Data Cleaning

La fase di Data Cleaning è una parte estremamente importante del processo, che consente di iniziare ad individuare eventuali problematicità nel set di dati scelto al fine di addestrare il modello. Richiede diversi passaggi, a partire dall'individuazione e analisi fino alla risoluzione di eventuali problematicità ed errori riscontrati. Lo scopo finale è quello di migliorare la qualità del dataset.

2.3.1 Feature superflue

Nel dataset è presente una sola feature superflua, la feature ***ID***, che non apporta nessuna informazione rilevante ai fini della risoluzione del problema. Per questa stessa ragione, si è deciso di eliminarla dal dataset, prima di partire con la fase di pulizia.

2.3.2 Valori duplicati

Nel dataset non sono presenti dati duplicati. Nonostante questo, grazie all'impiego di un codice apposito, è stato possibile rimuovere eventuali righe che presentassero valori delle feature tutti identici tra loro. Il numero di righe eliminate è stato pari a **0**.

2.3.3 Valori nulli

I valori nulli identificati sono in totale **328**, appartenenti unicamente alle feature ***Trigger*** e ***Medical_History***.
Per quanto riguarda i valori nulli in ***Trigger***, questi sono indicati da "**Unknown**", che potrebbe di fatto indicare tanto una mancanza di informazione, così quanto la totale assenza di causa scatenante.
Invece i valori nulli in ***Medical_History*** sono rappresentati dal valore vuoto. Questo tipo di valore è chiaramente un errore, in quanto il valore per un referto vuoto (volontariamente creato come tale) esiste, ed e' "**None**" (parliamo in questo caso di valori **Missing Completely at Random (MCAR)**).

Feature	Null Values	Non-Null Values	Data Type
Age	0	1200	int
Gender	0	1200	object
Panic_Attack_Frequency	0	1200	int64
Duration_Minutes	0	1200	int64
Trigger	206	994	object
Heart_Rate	0	1200	int64
Sweating	0	1200	object
Shortness_of_Breath	0	1200	object
Dizziness	0	1200	object
Chest_Pain	0	1200	object
Trembling	0	1200	object
Medical_History	122	1078	object
Medication	0	1200	object
Caffeine_Intake	0	1200	int64
Exercise_Frequency	0	1200	int64
Sleep_Hours	0	1200	float64
Alcohol_Consumption	0	1200	int64
Smoking	0	1200	object
Therapy	0	1200	object
Panic_Score	0	1200	int64

Table 2.2: Tabella dei valori nulli, non nulli e tipi di dati

2.3.4 Imputazione e gestione dei valori nulli

La feature interessate a questa fase dello sviluppo sono le colonne ***Trigger*** e ***Medical_History***. Per la feature ***Trigger*** si è deciso di rimuovere i sample che presentassero il valore "Unknown". Un dato così incerto sulla variabile indipendente avrebbe rischiato di causare cali di prestazione troppo alti e avrebbe rischiato di non produrre i risultati attesi in fase di predizione del modello.

Feature	**Null Values**	**Non-Null Values**	**Data Type**
Age	0	994	int64
Gender	0	994	object
Panic_Attack_Frequency	0	994	int64
Duration_Minutes	0	994	int64
Trigger	0	994	object
Heart_Rate	0	994	int64
Sweating	0	994	object
Shortness_of_Breath	0	994	object
Dizziness	0	994	object
Chest_Pain	0	994	object
Trembling	0	994	object
Medical_History	97	897	object
Medication	0	994	object
Caffeine_Intake	0	994	int64
Exercise_Frequency	0	994	int64
Sleep_Hours	0	994	float64
Alcohol_Consumption	0	994	int64
Smoking	0	994	object
Therapy	0	994	object
Panic_Score	0	994	int64

Table 2.3: Risultato dell'eliminazione dei valori nulli di Trigger

Per ***Medical_History*** si è scelto di proseguire con l'imputazione dei valori mancanti, precisamente è stata usata una imputazione basata sul KNN Clustering (la **KNN Imputation**): essa cerca tra i valori delle varie features e della feature in analisi per trovare istanze simili a quella in risoluzione (i "neighbors") usando la **Distanza Euclidea** come metrica di similitudine, e sulla base di essi imputare i valori mancanti della feature. Lo scopo del suo utilizzo sta nel voler ottenere un valore dall'alto grado di affidabilità all'interno del dataset imputato. In un ambito così delicato non è sembrato opportuno utilizzare metodologie come la moda dei valori.
Il risultato finale di queste fasi di eliminazione e imputazione di valori nulli e controllo della possibile presenza di sample duplicati ha portato ad ottenere:

Valori nulli nel dataset: 0

Numero di righe del dataset : 994

2.3.5 Oversampling

Questa fase è stata creata per rendere il dataset il piu' "fair and square" possibile.
Per bilanciare la distribuzione delle classi, è stato utilizzato **SMOTE**, che genera nuovi campioni sintetici per le classi meno rappresentate basandosi sui K-Nearest Neighbors.

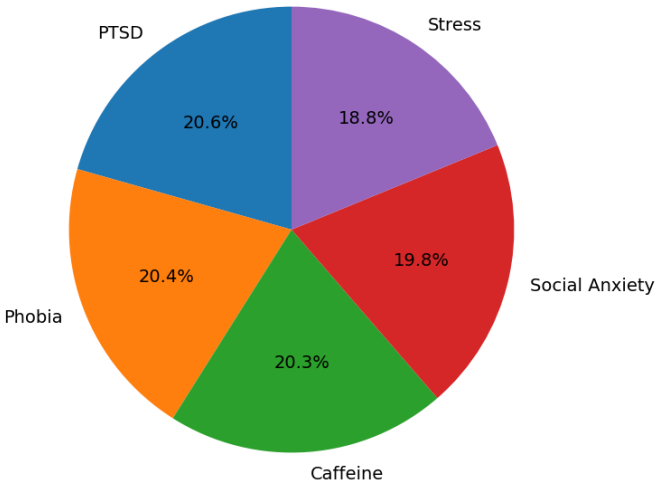

Figure 2.1: Prima dell'oversampling SMOTE

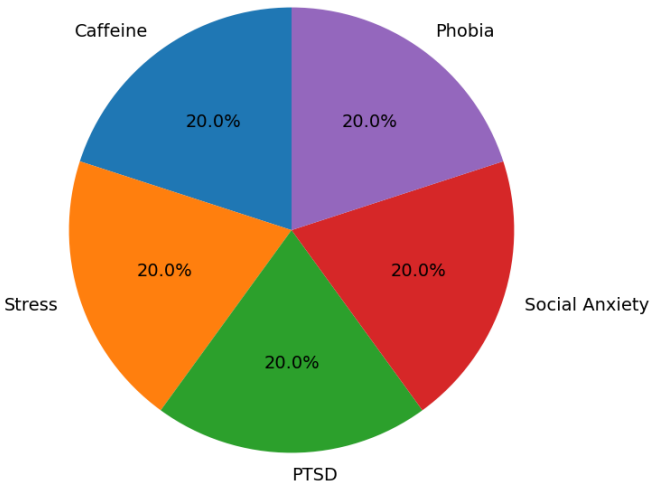

Figure 2.2: Dopo dell'oversampling SMOTE

2.3.6 Codifica dei dati categorici

Per ottimizzare l'addestramento del modello e garantire una più efficiente elaborazione dei dati, è fondamentale rappresentare le variabili categoriche in una forma numerica. Questa trasformazione consente di preservare l'informazione contenuta nei dati senza introdurre complessità computazionali eccessive. A tal fine, si ricorre a tecniche di codifica che permettono di mappare le categorie in valori numerici, mantenendo inalterata la loro struttura informativa. Tra questi metodi, l'encoding rappresenta una strategia consolidata per garantire la coerenza tra la rappresentazione originale e quella trasformata, consentendo al modello di apprendimento automatico di elaborare i dati in modo efficace.

2.3.7 Normalizzazione dei dati numerici

Nel dataset sono presenti diverse features numeriche, ed ognuna ha un proprio dominio di valori. Per allenare il modello, ci serve avere una scala unica per tutte le features con valori numerici, così che i valori non siano legati al contesto della feature stessa.
Per raggiungere questo scopo, andiamo a usare la funzione di normalizzazione **Min-Max**, definita come segue:

$$x' = \frac{x - min(x)}{max(x) - min(x)}$$

Dove x' è il nuovo valore, x il valore attuale, $min(x)$ e $max(x)$ i valori minimi e massimi della feature.
Questa normalizzazione è utilizzabile in questo contesto per via del fatto che non ci sono outliers, in caso contrario si avrebbero avuti risultati ambigui con valori normalizzati che non rispecchiavano i valori originali, dovuti appunto a questi ultimi.

2.4 Variabile dipendente

La variabile dipendente individuata come scopo della predizione del modello è la variabile categorica ***Trigger***. Questo aspetto era stato già definito in termini meno tecnici in fasi precedenti della trattazione del dataset e del problema in generale. Trigger è stata ritenuta la più opportuna poiché è la variabile che si presta meglio al raggiungimento dello scopo finale che ci si è proposti.

2.4.1 Analisi della variabile dipendente

La variabile dipendente, target, come dir si voglia, è di tipo categorico, e può assumere diversi valori quali: **PTSD**, **Caffeine**, **Phobia**, **Social Anxiety** e **Stress**. In principio, come visto, presentava anche il valore Unknown, che si è deciso di rimuovere in fase di pulizia dei dati al fine di evitare di generare rumore e previsioni errate.

2.4.2 Distribuzione della variabile dipendente

Le feature individuate come campione per studiare la distribuzione della variabile target sono le seguenti tre: ***Medical_History***, ***Gender*** ed ***Age***. Tramite varie valutazioni empiriche si è ritenuto che la corretta distribuzione di queste feature fosse di particolare importanza e quindi si è deciso di concentrarsi su di esse. Quello che ha portato l'attenzione sulla feature Gender è stata la presenza del gender "not binary", difatti questa feature può assumere tre valori categorici: Male, Female e Non binary. La percentuale di sample per quest'ultima categoria è estremamente più bassa rispetto alle altre due : 10%. Questo comporta un forte calo anche nell'incidenza totale che i vari tipi di trigger hanno sulla categoria non binary. Male e Female risultano però essere ben bilanciate tra loro non creando problemi di sorta all'algoritmo.

Per quanto riguarda Medical_History, il collegamento tra una possibile presenza di una diagnosi antecedente all'attacco di panico potrebbe, con grande probabilità, avere delle correlazioni con la causa scatenante. Chiaramente il valore maggiormente presente, quindi la diagnosi più comune tra i referti medici disponibili, è l'ansia.

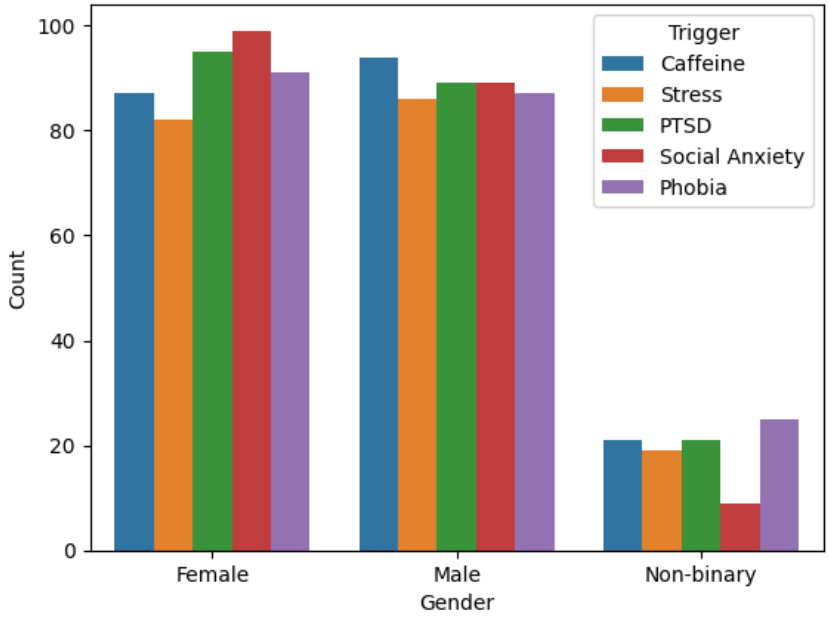

Figure 2.3: Distribuzione del Gender rispetto Trigger

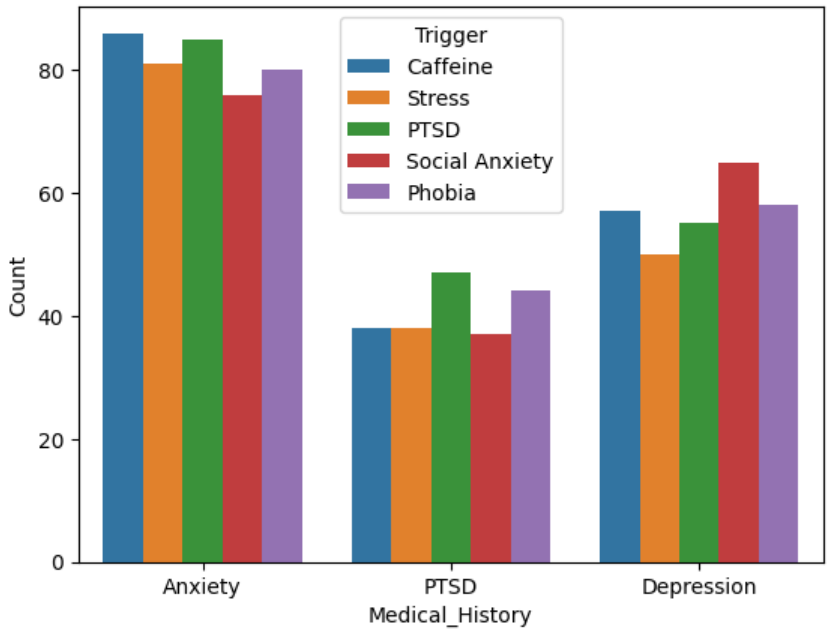

Figure 2.4: Distribuzione del Medical History rispetto Trigger

L'ultimo aspetto che si è ritenuto di analizzare in fase di verifica delle distribuzioni è stata l'età del soggetto, e dal grafico è possibile notare che Phobia e Stress vedono il loro picco d'incidenza nei casi di soggetti con un'età compresa tra i 40 e 60 anni, mentre caffeina e ansia sociale sono le maggiori cause scatenanti nelle persone con età compresa tra i 30 e i 50, mentre l'unico valore che vede un'incidenza maggiore nell'età adolescenziale è il PTSD.

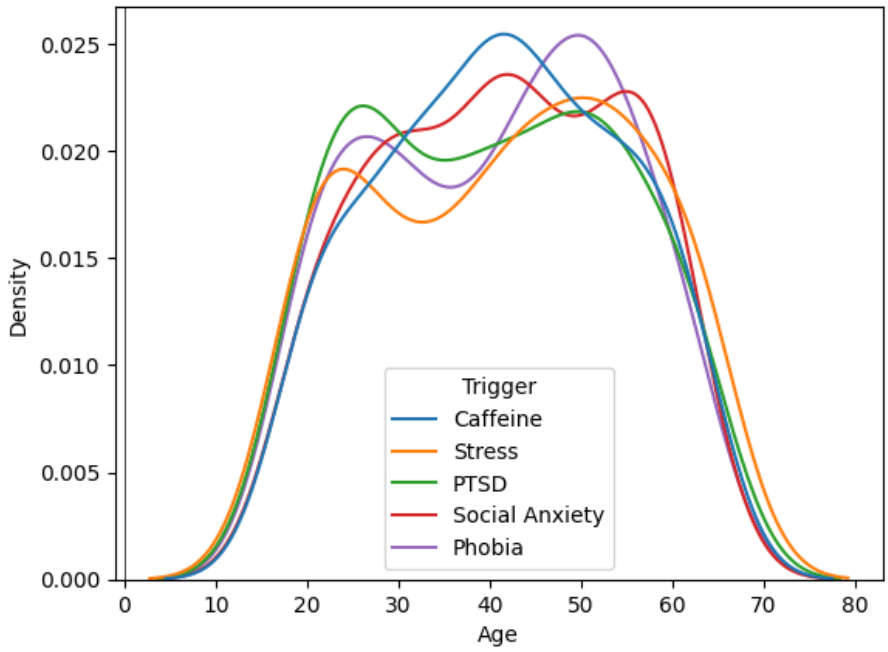

Figure 2.5: Distribuzione di Age rispetto Trigger

2.5 Data Splitting

Come già accennato in precedenza, per la valutazione e training del modello si è deciso di impiegare l'approccio ***K-Fold N-Times Cross Validation***, che permette di valutare le prestazioni di un modello riducendo il rischio di overfitting ottenendo una stima più affidabile del modello, evitando anche di cadere nel problema del Data Leaking che una suddivisione manuale, ad esempio basata sul principio di Pareto del 80-20, o su un'osservazione empirica del dataset, avrebbe potuto causare. Questo tipo di tecnica è stata facilmente impiegabile per il dataset Panic Attack poiché esso non presenta pattern temporali, quindi un mescolamento causale e divisione casuale non avrebbe causato perdita di informazioni importanti come sarebbe potuto accadere in altri contesti. Per implementarlo è stata utilizzata la ormai nota libreria sklearn, con particolare riferimento alla model_selection. La classe che si è importata per la suddivisione ai fini della validazione incrociata è il **RepeatedKFold**.

CHAPTER 3

Data Modeling & Evaluation

3.1 Introduzione

Avendo deciso di procedere con un tipo di organizzazione del progetto TDSP, questa parte del documento tratterà non solo la fase di modeling e quindi selezione dell'algoritmo, ma anche di valutazione dei risultati ottenuti, permettendo di lavorare in maniera sincrona ad entrambi gli aspetti, raffinando l'algoritmo e prendendo le scelte più appropriate in maniera fluida, senza necessariamente dover passare da una fase all'altra della progettazione e permettendo di risparmiare sulle tempistiche.

3.2 Data Modeling

La selezione dell'algoritmo più adatto allo sviluppo del modello è stata il primo passo di questa fase. Dopo un'analisi delle possibili alternative, è stato scelto il **Random Forest**, in quanto fornisce un buon compromesso tra accuratezza predittiva, interpretabilità e gestione della varianza nei dati.
Le reti neurali sono state prese in considerazione, ma, nonostante il loro potenziale in termini di prestazioni, avrebbero introdotto una complessità aggiuntiva non necessaria per il contesto del progetto. L'obiettivo principale era ottenere un modello efficace senza complicare inutilmente il processo di sviluppo e ottimizzazione.
Il **Random Forest**, consente di ottenere risultati affidabili grazie alla sua natura collaborativa. Inoltre, facilita il tuning degli iperparametri e l'analisi dell'importanza delle feature, rendendolo una scelta efficace sia dal punto di vista predittivo che interpretativo.

3.2.1 Algoritmo impiegato

Il **Random Forest** è un algoritmo di apprendimento supervisionato basato sulla combinazione di molteplici alberi decisionali. Ogni albero viene addestrato su un sottoinsieme casuale delle feature e dei dati, riducendo così il rischio di overfitting e migliorando la capacità di generalizzazione del modello.
Questo approccio fa parte della categoria degli algoritmi **Ensemble**, che combinano più algoritmi per ottenere una previsione finale più robusta. Il metodo utilizzato per l'aggregazione delle previsioni è la selezione della classe più votata tra i diversi alberi,

rendendo il modello resiliente rispetto al rumore nei dati.
Per l'implementazione, è stato utilizzato il classificatore *RandomForestClassifier* della libreria **scikit-learn**. Il tuning degli iperparametri ha riguardato in particolare:

- **n_estimators**: definisce il numero di alberi nella foresta; un valore più alto può migliorare la stabilità del modello, ma aumenta i tempi di addestramento;
- **random_state**: imposta un seme casuale per garantire riproducibilità nei risultati;
- **max_depth**: rappresenta la profondità massima di ogni albero; limitare la profondità aiuta a prevenire l'overfitting;
- **min_samples_split**: specifica il numero minimo di campioni richiesti per dividere un nodo interno;
- **min_samples_leaf**: specifica il numero minimo di campioni richiesti per essere in un nodo foglia.

Il criterio decisionale del random forest per arrivare alla soluzione in questo caso è la **Gini Impurity**. Nel contesto specifico del dataset in esame, è stato osservato che oltre una certa soglia di alberi decisionali, i miglioramenti nelle metriche di valutazione erano trascurabili. Pertanto, si è scelto un valore di *n_estimators* bilanciato tra qualità delle predizioni e efficienza computazionale.

3.2.2 Implementazione

Il modello è stato sviluppato utilizzando il linguaggio di programmazione **Python**, e sono stati usati come ambienti di sviluppo **Visual Studio Code** e **Google Colab**. Quest'ultimo si è rivelato particolarmente utile per la computazione nel cloud offerta da Google, e per l'esecuzione interattiva del codice senza la necessità di installare librerie localmente.
Le principali librerie utilizzate nel processo di implementazione includono:

- **pandas**: per il caricamento e la gestione dei dati;
- **matplotlib** e **seaborn**: per la visualizzazione di grafici e analisi esplorativa;
- **numpy**: per operazioni numeriche e gestione di array;

- **scikit-learn** e **imblearn**: per la modellazione, il preprocessing e tecniche di bilanciamento come SMOTE.

3.3 Training del modello

Il processo di Data Modeling culmina nella fase di training e valutazione del modello di machine learning. Questa fase è cruciale per determinare l'efficacia del modello nel generalizzare a nuovi dati e per ottimizzarne le prestazioni.

3.3.1 Step di training

Il training del modello rappresenta il cuore del processo di machine learning. Durante ciascuna iterazione della cross-validation:

- I dati vengono suddivisi in training e test set, garantendo che la distribuzione delle classi sia rispettata.
- Viene addestrato un modello Random Forest utilizzando parametri scelti empiricamente (ad esempio, `n_estimators` = 500, `max_depth` = 8, `min_samples_split` = 6, e `min_samples_leaf` = 3). Questi parametri sono stati ottimizzati per bilanciare la capacità predittiva e la complessità del modello, riducendo il rischio di overfitting.
- Viene effettuato il fitting del modello sul training set, durante il quale il modello apprende le caratteristiche discriminanti del dataset.
- Al termine del training, il modello viene testato sul set di validazione per generare predizioni e calcolare le principali metriche di valutazione (Accuracy, Precision, Recall, F1-score).

3.3.2 Data Splitting and Cross-Validation

Per valutare le prestazioni del modello Random Forest, è stata utilizzata la tecnica di cross-validation **RepeatedKFold**. Questa tecnica prevede la suddivisione del dataset in K folds, ripetendo il processo per N ripetizioni. In questo caso, sono stati utilizzati **n_splits = 6** folds e **n_repeats = 3** ripetizioni, risultando in un totale di 18 iterazioni di training e test. Questo approccio permette di ottenere una stima più robusta delle prestazioni del modello rispetto a una singola suddivisione training/test.
Un cambiamento nel valore di questi parametri porterebbe a tempi di addestramento più

lunghi ma idealmente anche a una qualità di predizione più alta. Nel caso del dataset in questione, il numero di fold (n_splits) e ripetizioni (n_repeats) è impostato per bilanciare accuratezza e tempo di calcolo. RepeatedKFold effettua uno shuffle dei dati ad ogni ripetizione della cross-validation, garantendo una distribuzione più equa tra i fold. È stato definito un seme (random_state) per garantire la riproducibilità della soluzione. Infine, vengono restituiti gli indici che indicano quali dati usare per l'addestramento e quali per i test in ogni iterazione, che poi andranno impiegati per suddividere il dataset. L'aspetto negativo di questa tecnica risiede nelle tempistiche. Dovendo effettuare l'addestramento per un numero di volte pari a n_splits * n_repeats, il training risulta rallentato. Tuttavia, essendo il progetto svolto in un ambiente accademico, senza la necessità di rifarsi a tempi di mercato stretti, le tempistiche leggermente dilatate non sono considerate una problematica troppo impattante.

Inoltre, per ogni fold, le metriche sono salvate e aggregate tramite la costruzione di confusion matrix, che vengono poi utilizzate per l'analisi per classe. Questa strategia consente non solo di stimare le performance globali, ma anche di evidenziare le criticità nella classificazione di specifiche categorie, in particolare in un contesto multi-classe.

3.3.3 Salvataggio del modello

Una volta completato il processo di training, il modello finale viene salvato utilizzando la libreria `pickle`. Il salvataggio del modello permette di effettuare future predizioni su nuovi dati senza dover ri-addestrare il modello, garantendo al contempo una maggiore efficienza computazionale.

Esempio di salvataggio: `pickle.dump(rf_model, "random_forest_model.pkl")`

3.3.4 Ottimizzazione dei Parametri e Pipeline di Training

Sebbene l'attuale configurazione dei parametri del Random Forest sia stata scelta per ottenere un buon compromesso tra complessità e performance, il processo di training includerebbe naturalmente ulteriori fasi di ottimizzazione, come:

- Trovare la combinazione ottimale di iperparametri;
- Feature selection e engineering per migliorare la capacità del modello di distinguere tra le classi;

- Validazione incrociata più dettagliata per analizzare la variabilità delle prestazioni sui diversi fold.

Tale pipeline di training e ottimizzazione assicura che il modello sviluppato sia non solo accurato, ma anche robusto nella generalizzazione, tenendo conto della natura multi-classe del dataset e degli squilibri presenti tra le categorie.
La forte integrazione tra la fase di training, la gestione della cross-validation e l'analisi dettagliata delle confusion matrix, costituisce la base per il successivo step di valutazione e perfezionamento del modello.
Per motivi di tempistiche, nonostante siano state provate delle ottimizzazioni, non siamo riusciti a integrarle a dovere.

3.4 Valutazione del modello

Per una visione più dettagliata delle prestazioni del modello, è stata calcolata la matrice di confusione. Questa matrice permette di distinguere tra veri positivi, falsi positivi, veri negativi e falsi negativi, evidenziando eventuali criticità nella classificazione di specifiche categorie. L'uso combinato di queste metriche ha permesso di ottenere un quadro completo dell'efficacia del modello sia in termini globali che per ciascuna classe.
Per valutare le prestazioni del modello, sono state utilizzate diverse metriche di classificazione, ognuna con un ruolo specifico nell'analisi della qualità delle predizioni. L'**Accuracy** misura la proporzione di predizioni corrette rispetto al totale delle istanze, ma in presenza di un dataset sbilanciato può risultare fuorviante, motivo per cui è stata affiancata da altre metriche più specifiche. La **Precision** indica la frazione di istanze classificate come positive che sono effettivamente corrette, risultando particolarmente utile per ridurre il numero di falsi positivi. La **Recall**, o sensibilità, misura la capacità del modello di identificare correttamente le istanze positive. Il **F1-score**, che rappresenta la media armonica di Precision e Recall, fornisce un equilibrio tra la capacità del modello di identificare correttamente le istanze positive e la riduzione dei falsi positivi.

3.4.1 Matrici di confusione

Le matrici di confusione, essendo state realizzate per ogni Fold di ogni ripetizione N, per ogni classe della variabile dipendente, ne abbiamo in totale 18, con valori come di seguito:

Fold 1

	TP	FP	FN	TN
Caffeine	3	7	37	117
PTSD	6	29	29	100
Phobia	3	31	18	112
Social Anxiety	8	22	36	98
Stress	8	47	16	93

Fold 2

	TP	FP	FN	TN
Caffeine	7	19	27	111
PTSD	8	38	19	99
Phobia	10	29	19	106
Social Anxiety	8	20	24	112
Stress	10	15	32	107

Fold 3

	TP	FP	FN	TN
Caffeine	9	24	17	114
PTSD	12	19	23	110
Phobia	3	14	37	110
Social Anxiety	11	36	13	104
Stress	12	24	27	101

Fold 4

	TP	FP	FN	TN
Caffeine	6	18	25	115
PTSD	11	27	26	100
Phobia	3	27	29	105
Social Anxiety	6	28	23	107
Stress	14	24	21	105

Fold 5

	TP	FP	FN	TN
Caffeine	4	23	27	109
PTSD	11	31	16	105
Phobia	6	16	33	108
Social Anxiety	3	19	36	105
Stress	13	37	14	99

Fold 6

	TP	FP	FN	TN
Caffeine	7	36	28	92
PTSD	5	20	35	103
Phobia	9	21	25	108
Social Anxiety	10	21	17	115
Stress	10	24	17	112

Figure 3.1: Ripetizione 1, folds da 1 a 6

Fold 1

	TP	FP	FN	TN
Caffeine	11	19	18	116
PTSD	13	38	16	97
Phobia	5	15	31	113
Social Anxiety	11	21	25	107
Stress	8	23	26	107

Fold 2

	TP	FP	FN	TN
Caffeine	5	14	38	107
PTSD	7	20	33	104
Phobia	6	21	23	114
Social Anxiety	5	23	19	117
Stress	11	52	17	84

Fold 3

	TP	FP	FN	TN
Caffeine	4	27	26	107
PTSD	7	21	30	106
Phobia	6	25	21	112
Social Anxiety	9	19	28	108
Stress	10	36	23	95

Fold 4

	TP	FP	FN	TN
Caffeine	5	23	24	112
PTSD	10	31	20	103
Phobia	4	21	34	105
Social Anxiety	6	35	26	97
Stress	7	22	28	107

Fold 5

	TP	FP	FN	TN
Caffeine	7	21	29	106
PTSD	10	27	23	103
Phobia	5	19	30	109
Social Anxiety	9	33	19	102
Stress	14	18	17	114

Fold 6

	TP	FP	FN	TN
Caffeine	8	26	22	107
PTSD	7	25	25	106
Phobia	4	20	26	113
Social Anxiety	10	22	28	103
Stress	13	28	20	102

Figure 3.2: Ripetizione 2, folds da 7 a 12

Fold 1

	TP	FP	FN	TN
Caffeine	9	22	25	108
PTSD	9	29	24	102
Phobia	8	16	27	113
Social Anxiety	11	37	13	103
Stress	9	14	29	112

Fold 2

	TP	FP	FN	TN
Caffeine	6	21	22	115
PTSD	10	32	26	96
Phobia	4	17	32	111
Social Anxiety	5	25	31	103
Stress	12	32	16	104

Fold 3

	TP	FP	FN	TN
Caffeine	9	19	23	113
PTSD	6	19	32	107
Phobia	9	26	21	108
Social Anxiety	9	31	18	106
Stress	11	25	26	102

Fold 4

	TP	FP	FN	TN
Caffeine	5	25	21	113
PTSD	10	21	26	107
Phobia	9	20	24	111
Social Anxiety	7	20	35	102
Stress	10	37	17	100

Fold 5

	TP	FP	FN	TN
Caffeine	6	30	26	101
PTSD	8	23	30	102
Phobia	4	23	24	112
Social Anxiety	8	20	28	107
Stress	11	30	18	104

Fold 6

	TP	FP	FN	TN
Caffeine	2	9	43	109
PTSD	10	45	10	98
Phobia	9	12	24	118
Social Anxiety	10	21	20	112
Stress	13	32	22	96

Figure 3.3: Ripetizione 3, folds da 13 a 18

Dai risultati delle matrici di confusione emerge che per alcune classi vi è un basso numero di True Positives accompagnato da un elevato numero di True Negatives. Questo accade perché nella classificazione multi-classe gli esempi che non appartengono a una specifica classe sono contati come True Negatives. Di conseguenza, sebbene il modello mostri un'alta accuratezza globale, per le classi target potrebbe non riuscire a riconoscere adeguatamente gli esempi, evidenziando la necessità di analizzare per classe metriche come Precision, Recall e F1-Score per una valutazione più accurata delle performance.

3.4.2 Metriche finali

Le metriche impiegate per la valutazione del modello hanno prodotto i seguenti risultati medi:

1. Precision: **0.251079452211499**
2. Accuracy: **0.2450911932432208**
3. Recall: **0.2450911932432208**
4. F1-score: **0.23908805478749015**

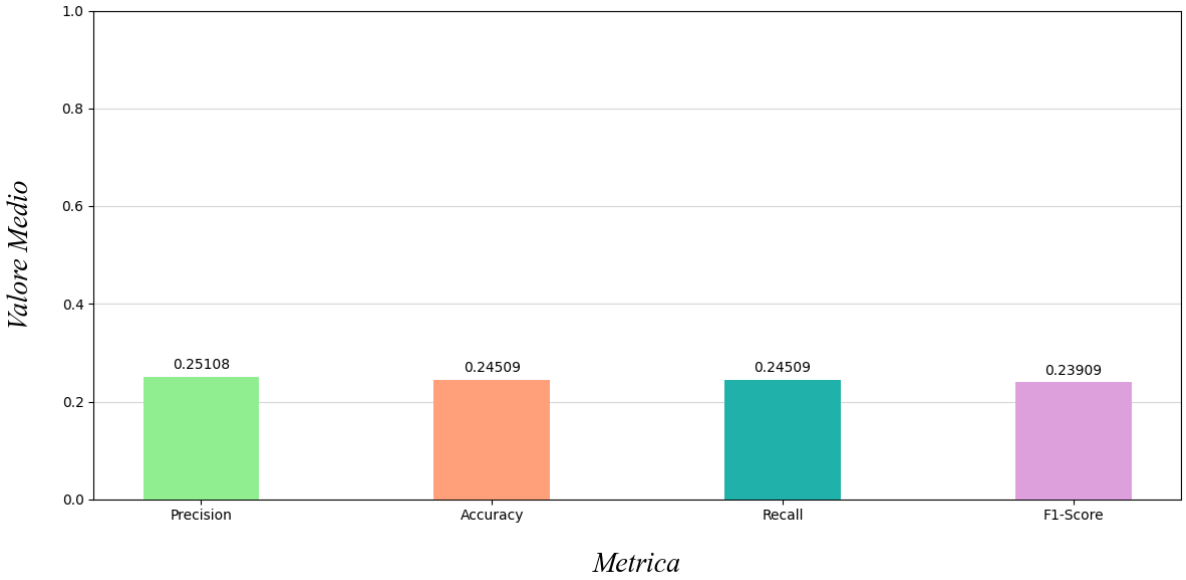

Figure 3.4: Valori medi delle metriche calcolate sul modello

Possiamo comprendere che, nonostante alcuni indicatori possano dare l'impressione di una performance decente, il modello presenta una capacità predittiva che si avvicina a quella di una classificazione randomica. Questo risultato si deve, in parte, alla complessità intrinseca del problema, all'eterogeneità dei dati e alla presenza di classi sbilanciate. Tuttavia, è importante sottolineare che una bassa performance nelle metriche aggregate potrebbe celare delle peculiarità nascoste nei singoli gruppi, motivo per cui si è proceduto ad una valutazione per classe.

Precision per-class

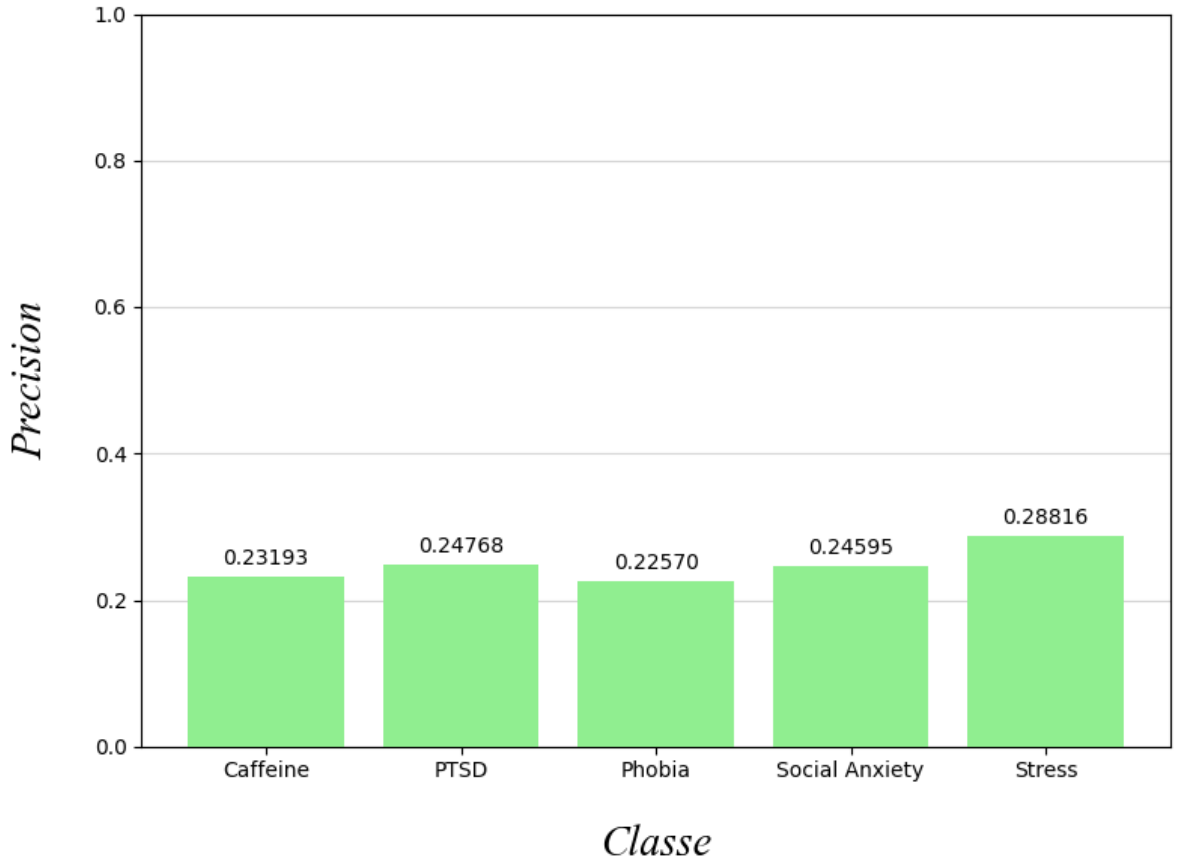

Figure 3.5: Valori Precision per-class

I valori di Precision per ciascuna classe risultano relativamente bassi (intorno a 0.3), il che indica che, pur essendo il modello in grado di escludere numerosi falsi positivi (False Positives) in maniera globale, risulta difficile identificare correttamente le istanze positive all'interno di ciascun gruppo. Questo può essere interpretato come un segnale del fatto che il modello tende ad attribuire erroneamente la classe positiva a casi ambigui, generando sia un numero elevato di errori di classificazione sia una bassa proporzione tra predizioni corrette e totali predizioni positive.

Accuracy per-class

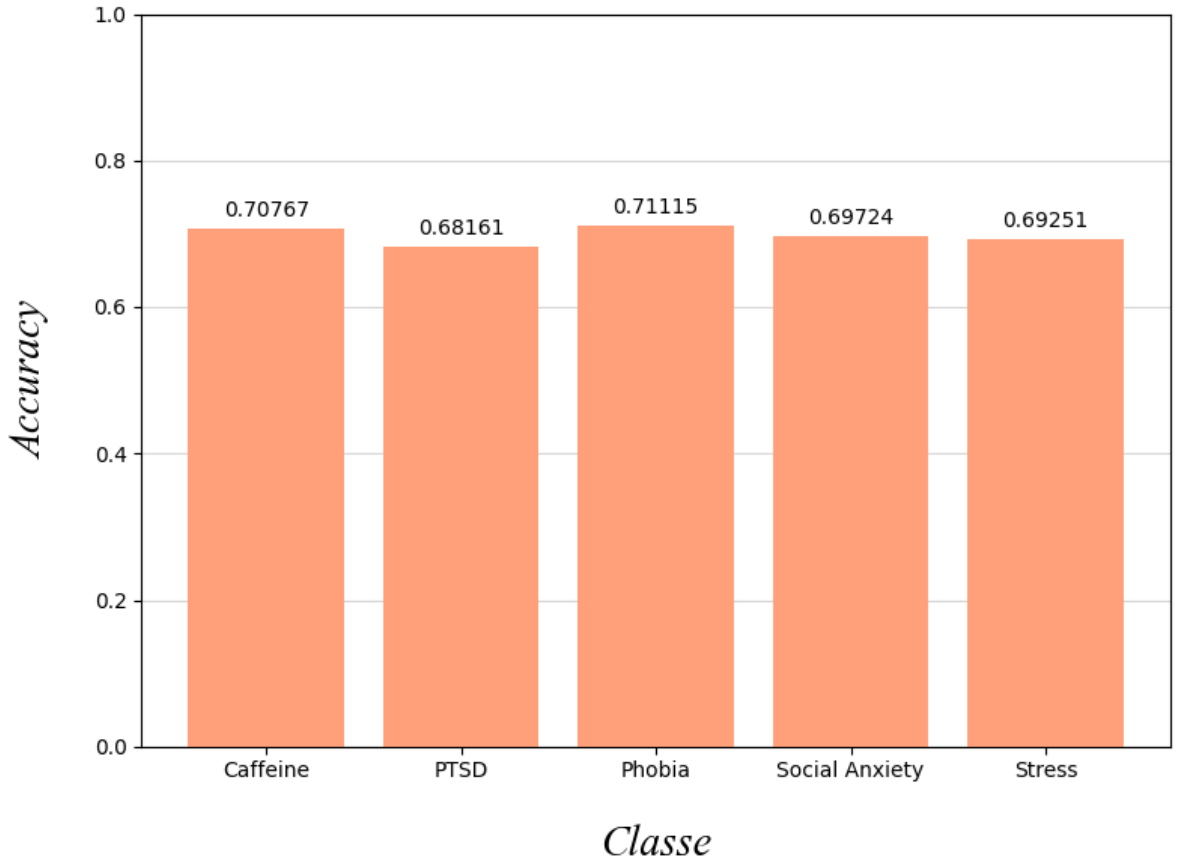

Figure 3.6: Valori Accuracy per-class

I valori di Accuracy per ogni classe risultano elevati (circa 0.7), rispetto al valore medio finale. Tale fenomeno si verifica perché, in un contesto multi-classe, la metrica Accuracy tende a beneficiare dell'influenza dei numerosi True Negatives derivanti da tutte le altre classi. Questo effetto porta la metrica ad apparire più favorevole, anche se la capacità predittiva per le istanze positive specifiche risulta bassa. In sostanza, l'Accuracy complessiva può non riflettere appieno le criticità del modello in contesti con sbilanciamento tra le classi.

Recall per-class

Figure 3.7: Valori Recall per-class

Il Recall per ciascuna classe si attesta su valori modesti (circa 0.3), evidenziando la scarsa capacità del modello di individuare tutte le istanze positive corrette. Questo basso valore è indicativo di un'elevata incidenza di False Negatives, che rappresentano i casi in cui il modello non è riuscito a riconoscere esplicitamente la presenza della classe target. Questa caratteristica è tipicamente preoccupante e può avere conseguenze rilevanti, suggerendo la necessità di ulteriori interventi di miglioramento o di applicazione di tecniche di data augmentation.

F1-score per-class

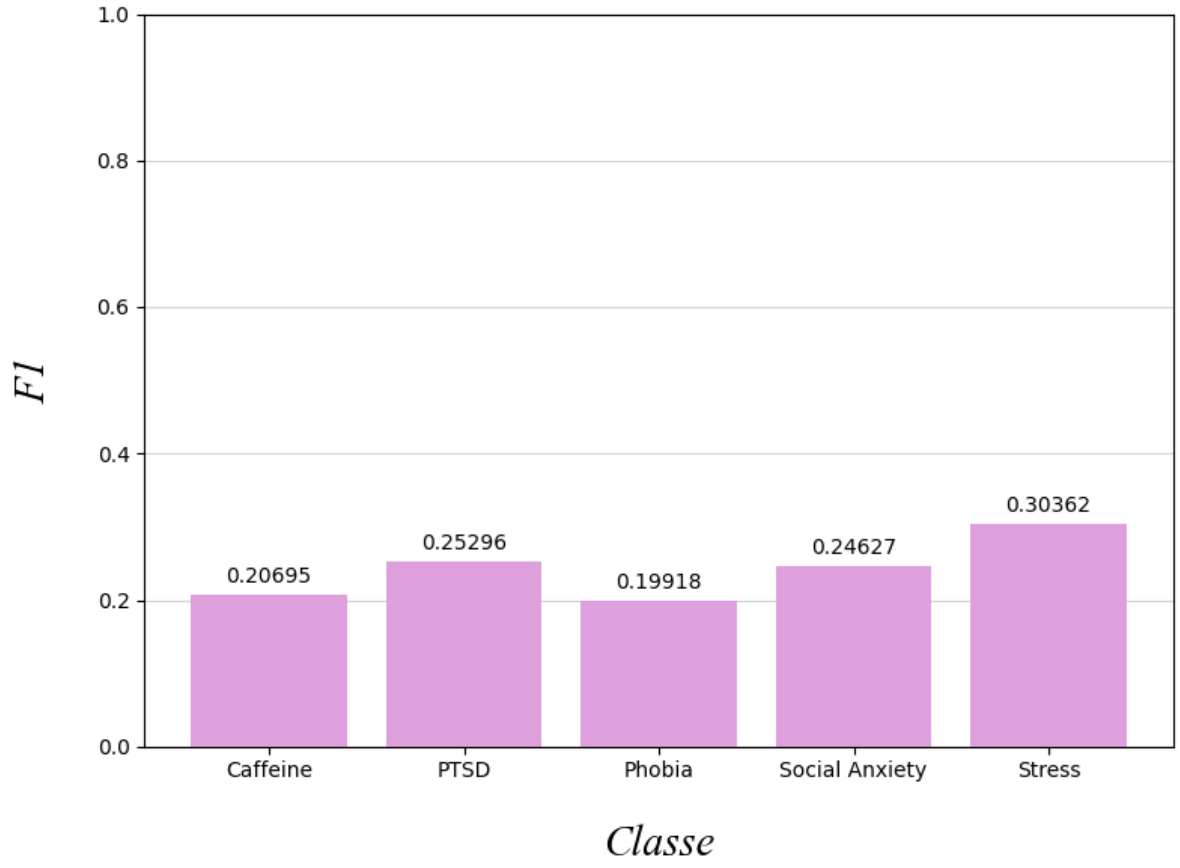

Figure 3.8: Valori F1-score per-class

L'F1-score, che rappresenta la media armonica tra Precision e Recall, mostra anch'esso valori bassi (circa 0.3), confermando la problematica principale del modello: un insufficiente equilibrio tra la capacità di evitare falsi positivi e quella di ridurre i falsi negativi. Questa metrica sintetica evidenzia come, nonostante una discreta accuratezza complessiva, il modello fatica a fornire una performance robusta nelle classi positive. Tali risultati suggeriscono l'opportunità di rivedere sia le tecniche di preprocessing sia gli approcci di modellazione, oppure di considerare metodi di trasferimento dell'apprendimento per migliorare la rilevazione delle istanze critiche.

ROC Curve

La Receiver Operating Characteristic (ROC) curve rappresenta un'analisi fondamentale nella valutazione delle prestazioni di un classificatore multi-classe. Per ogni classe, la ROC curve è ottenuta tracciando la True Positive Rate (TPR), anche noto come recall o sensibilità, contro il False Positive Rate (FPR) al variare della soglia decisionale del modello. L'area sottesa dalla ROC curve (Area Under the Curve, AUC) fornisce un indicatore sintetico della capacità discriminativa del modello per quella specifica classe. Un valore di AUC = 1.0 indica un classificatore perfetto, mentre un valore AUC = 0.5 denota un modello privo di capacità predittiva, equivalente a una scelta casuale. Nel codice, le curve ROC sono calcolate e visualizzate per ogni classe presente nella variabile target "Trigger".

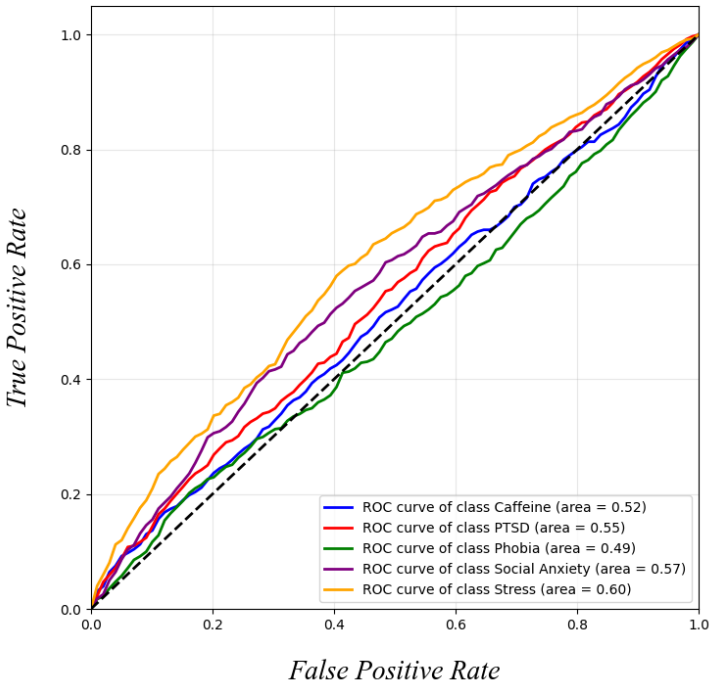

Figure 3.9: ROC Curves per-class

Da come possiamo vedere dal grafico, le ROC Curves sono molto vicine ad una predizione totalmente casuale, ma cio' è dovuto anche al fatto che la ROC Curve è calcolata sui Positivi, e come precedentemente abbiamo detto il modello predice in modo giusto particolarmente le istanze Negative.

3.5 Valutazione finale

Dall'analisi complessiva dei risultati emerge che il modello presenta una capacità di generalizzazione mediamente bassa. Le metriche aggregate, con valori di Accuracy, Precision, Recall e F1-score intorno al 0.25-0.30, insieme alle ROC Curves che si comportano in maniera simile a predizioni casuali, evidenziano che il modello fatica a distinguere correttamente le istanze positive dalle negative. Questo andamento, in parte, è influenzato dal fatto che il classificatore beneficia notevolmente dalla forte presenza dei True Negatives, mentre la rilevazione degli elementi positivi risulta sostanzialmente carente.

In sintesi, sebbene il modello riesca a riconoscere adeguatamente le istanze negative, la difficoltà nel classificare correttamente quelle positive denota una generalizzazione limitata. Miglioramenti potrebbero passare da una revisione delle tecniche di preprocessamento e da un approfondimento nell'analisi delle feature (ad esempio mediante feature engineering o tecniche di oversampling/undersampling per affrontare eventuali squilibri), fino all'esplorazione di metodologie alternative o ensemble per incrementare la capacità discriminativa complessiva del modello. Una continua iterazione e sperimentazione saranno fondamentali per rafforzare la sua efficacia in scenari di predizione reali.

CHAPTER 4

Deployment

4.1 Deployment del Modello Random Forest

Il deployment del modello Random Forest avviene in una serie di fasi che integrano il processo di data preparation, la validazione e l'implementazione in produzione tramite un'applicazione web. Di seguito, si descrivono in forma sintetica gli step prodotti dalla fase.

4.1.1 Preparazione al Deployment

Il modello, precedentemente addestrato, è stato salvato in formato **pickle**, garantendo la conservazione dello stato del modello post-training. Viene, dunque, caricato all'interno di un'applicazione web sviluppata con Flask, che si occupa di gestire le richieste di inferenza. L'applicazione carica anche gli strumenti di preprocessing (encoder e scaler) per assicurare che i dati in ingresso siano trasformati nello stesso modo in cui lo sono stati durante il training.

4.1.2 Inferenza e Gestione dell'Incertezza

Per l'inferenza, l'utente fornisce i dati tramite una form web. I dati vengono processati e riordinati per rispettare l'ordine originario delle feature. Il modello, dopo aver effettuato la predizione, restituisce una probabilità associata ad ognuna delle classi. Se la probabilità massima è inferiore a una soglia stabilita (nel nostro caso **0.3**, già a questo valore il modello è leggermente inaffidabile), il sistema comunica che il Trigger è stato determinato, ma non in maniera affidabile, restituendo il messaggio:

```
Valore predizione + "(low probability, uncertain prediction)"
```

Questo approccio consente di far gestire ai medici in modo prudente i casi di bassa confidenza, senza eliminare totalmente il supporto che il modello fornisce.
In sintesi, il deployment integra una fase di preprocessing coerente con il training, un salvataggio accurato del modello e un'interfaccia web interattiva, garantendo una transizione fluida dalla fase di ricerca a un utilizzo pratico in ambiente di produzione.

CHAPTER 5

Conclusioni

5.1 Conclusioni

Questo studio ha analizzato il processo di sviluppo di un modello di machine learning per l'individuazione dei trigger che possono precedere un attacco di panico, coprendo tutte le fasi dalla preparazione del dataset fino alla valutazione delle prestazioni del modello.

La pulizia e la pre-elaborazione dei dati hanno svolto un ruolo cruciale, consentendo di gestire valori mancanti, identificare correlazioni rilevanti e ridurre il rumore nei dati. Tecniche di selezione delle feature e metodi di imputazione hanno migliorato la qualità del dataset, aumentando la capacità predittiva del modello.

Tuttavia, alcune limitazioni devono essere considerate. Il dataset utilizzato, composto da 1199 campioni, sebbene sufficiente per una prima analisi, potrebbe non essere rappresentativo di un'ampia varietà di soggetti. In particolare, la distribuzione delle classi potrebbe non riflettere con precisione la popolazione generale, riducendo la capacità del modello di generalizzare su nuovi dati.

Per migliorare la robustezza del modello, sarebbe utile ampliare il dataset includendo un numero maggiore di soggetti con caratteristiche più diversificate, ad esempio integrando dati provenienti da diverse fasce d'età, contesti socio-culturali differenti e condizioni cliniche più varie. Inoltre, l'aggiunta di dati fisiologici raccolti da dispositivi *wearable*, come frequenza cardiaca, variabilità della pressione sanguigna e attività elettrodermica, potrebbe fornire informazioni più oggettive sui trigger degli attacchi di panico, riducendo il rischio di bias nei dati raccolti tramite auto-valutazione.

Un altro passo fondamentale sarà testare il modello su un dataset più ampio e diversificato per valutarne la capacità di generalizzazione. L'adozione di tecniche avanzate come modelli di deep learning o metodi di apprendimento semi-supervisionato potrebbe migliorare ulteriormente la qualità delle previsioni, specialmente se combinata con una maggiore quantità di dati.

In sintesi, il lavoro svolto rappresenta un primo passo verso l'uso dell'intelligenza artificiale come supporto agli specialisti nella comprensione e gestione degli attacchi di panico. Con un dataset più ricco e diversificato e l'integrazione di nuove tecnologie, il modello potrebbe contribuire in modo ancora più significativo alla prevenzione e all'intervento tempestivo in ambito clinico.

5.2 References

1. "Panick Attack" Dataset, *Akshay Choudhary*

2. "TwoTheRoot" Github Repository, *Antonio Maiorano e Silvana De Martino*