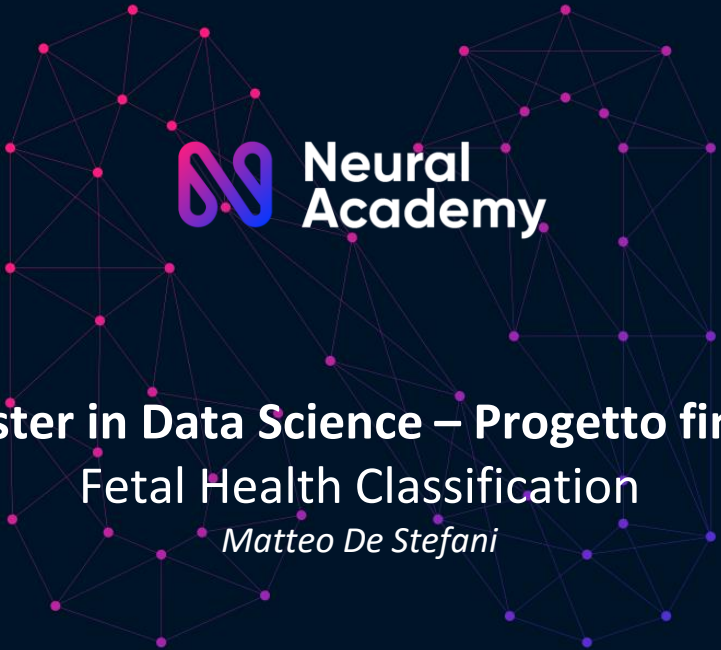


Master in Data Science – Progetto finale

Fetal Health Classification

Matteo De Stefani

Analisi del problema e obiettivi

- Il dataset oggetto dell'analisi è composto da una serie di parametri medici ottenuti mediante un esame di **Cardiotocografia (CTG)**, esame che si effettua dalla 27° settimana di gestazione per verificare lo stato di salute del feto con lo scopo di ridurre la mortalità perinatale
- Il dataset è composto da **2126 osservazioni** e da **21 features**, di cui alcune di carattere prettamente medico (ad es. battito cardiaco, contrazioni uterine) e altre di carattere più statistico (ad es. media, moda e mediana dell'istogramma dell'esame)

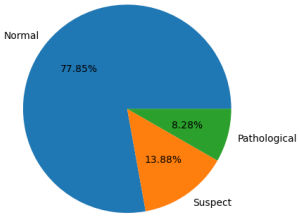
La variabile target è rappresentata dallo stato di salute del feto che può essere classificato in uno dei tre seguenti:

1. **Normal**
2. **Suspect**
3. **Pathological**

Obiettivo dell'analisi è quello di costruire un **modello di classificazione** in grado di predire correttamente lo stato di salute del feto in base alle features prese in input

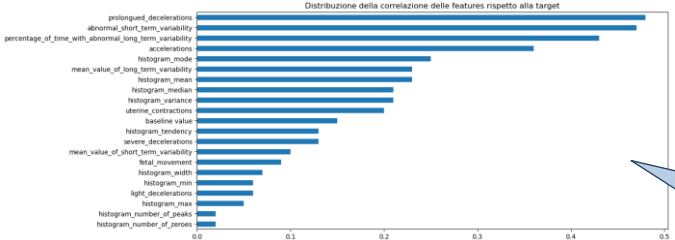
Exploratory Data Analysis

Dopo un iniziale controllo del dataset (presenza valori nulli, verifica e pulizia dei dati..), inizio la fase di esplorazione analizzando la distribuzione della **variabile target** all'interno del campione:



Noto subito un **punto di attenzione**: il dataset è fortemente **sbilanciato** verso la classe *Normal*, aspetto che potrebbe incidere negativamente sulla capacità del modello di predire correttamente le classi di minoranza avendo poche osservazioni su cui allenarsi

Passo ora ad analizzare le **features**, verificando innanzitutto la **correlazione** di ciascuna con la variabile target:



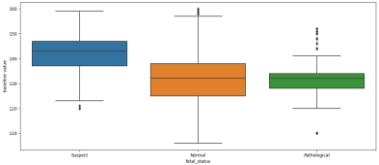
Visualizzo in valore assoluto le correlazioni per identificare quelle maggiormente correlate alla target (indipendentemente dal fatto che siano correlate positivamente o negativamente)

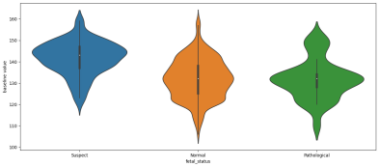
Focalizzo l'analisi sul dato relativo al battito cardiaco (*baseline value*) e su alcune delle features con correlazione maggiore alla target

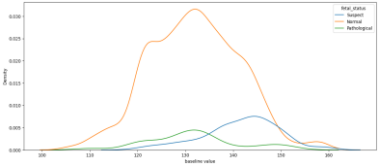
Exploratory Data Analysis

Neural Academy

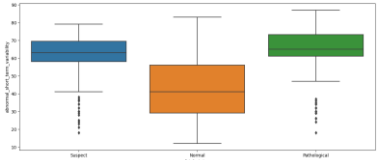
Baseline value (corr 0,14)

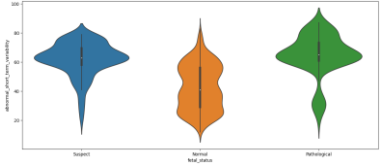


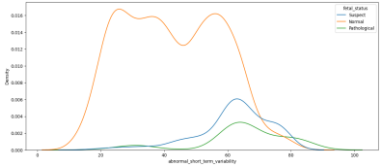




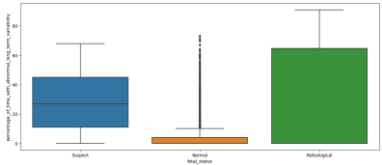
Abn. short term variability (corr 0,47)

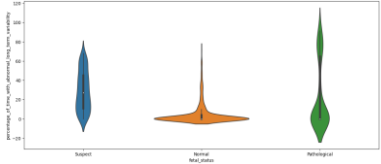


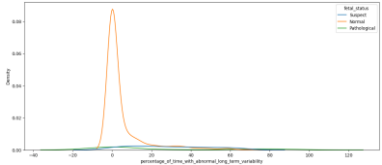




% of time Abn. Long term var. (corr 0,43)







Mentre la distribuzione del battito cardiaco per le tre classi target risulta essere poco esplicativa, l'analisi sulle altre due features (entrambe con elevata correlazione verso la target) evidenzia maggiori peculiarità della classe *Normal* rispetto alle altre due

Applicazione dei modelli di classificazione

- Dopo alcuni perfezionamenti al dataset (rimuovo alcune features molto correlate tra loro), procedo allo split del campione tra train set (1700 osservazioni) e test set (426 osservazioni) e alla standardizzazione dei dati
- A questo punto applico i seguenti modelli di classificazione: **Logistic Regression, KNN, Random Forest e XGBoost**
- Logistic Regression e KNN sono modelli piuttosto semplici per cui mi aspetto di ottenere le migliori performance con Random Forest e XGBoost

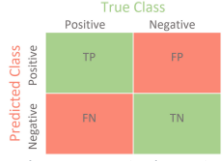
Quando un dataset risulta essere molto sbilanciato verso una classe piuttosto che le altre (come nel nostro caso in cui circa il 78% delle osservazioni appartengono alla classe *Normal*), valutare le performance di un modello basandosi solo sull'*Accuracy*, che misura la % di predizioni corrette, può essere fuorviante
Per questo motivo è necessario utilizzare anche altre **metriche di valutazione**

Un modello *dummy* predicendo sempre che i feti appartengono alla classe *Normal* avrebbe, infatti, una *Accuracy* del 78%

La ***Confusion Matrix*** consente di verificare le previsioni del modello per ciascuna classe identificando su quali è più accurato e su quali, invece, commette il maggior numero di errori

Dalla Confusion Matrix si originano poi ulteriori metriche:

- ***Precision***: Indica la percentuale delle previsioni positive corrette (TP) sul totale delle previsioni positive del modello (TP+FP)
- ***Recall***: Indica la percentuale delle previsioni positive corrette (TP) sul totale delle istanze positive (TP+FN)
- ***F1 Score***: Combina i risultati di *Precision* e *Recall* calcolandone la media armonica

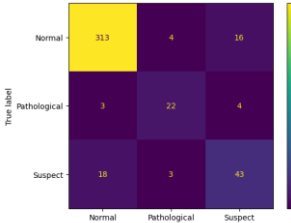


Confusion Matrix per classificazione binaria

Applicazione dei modelli di classificazione

Neural Academy

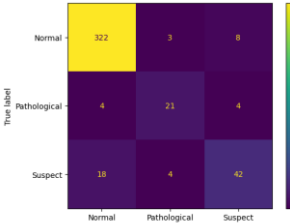
Logistic Regression



Predizioni corrette: ***378***
Predizioni errate: ***48***
Accuracy: ***89%***

	precision	recall	f1-score	support
Normal	0.94	0.94	0.94	333
Pathological	0.76	0.76	0.76	29
Suspect	0.68	0.67	0.68	64
accuracy			0.89	426
macro avg	0.79	0.79	0.79	426
weighted avg	0.89	0.89	0.89	426

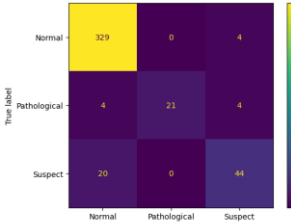
KNN



Predizioni corrette: ***385***
Predizioni errate: ***41***
Accuracy: ***90%***

	precision	recall	f1-score	support
Normal	0.94	0.97	0.95	333
Pathological	0.75	0.72	0.74	29
Suspect	0.78	0.66	0.71	64
accuracy			0.90	426
macro avg	0.82	0.78	0.80	426
weighted avg	0.90	0.90	0.90	426

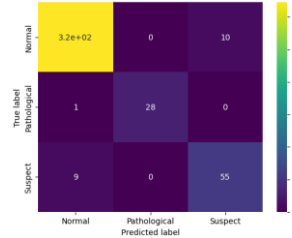
Random Forest



Predizioni corrette: ***394***
Predizioni errate: ***32***
Accuracy: ***92%***

	precision	recall	f1-score	support
Normal	0.93	0.99	0.96	333
Pathological	1.00	0.72	0.84	29
Suspect	0.85	0.69	0.76	64
accuracy			0.92	426
macro avg	0.93	0.80	0.85	426
weighted avg	0.92	0.92	0.92	426

XGBoost



Predizioni corrette: ***406***
Predizioni errate: ***20***
Accuracy: ***95%***

	precision	recall	f1-score	support
Normal	0.97	0.97	0.97	333
Pathological	1.00	0.97	0.98	29
Suspect	0.85	0.86	0.85	64
accuracy			0.95	426
macro avg	0.94	0.93	0.94	426
weighted avg	0.95	0.95	0.95	426

Tutti i modelli ottengono buoni risultati, nonostante alcune difficoltà a classificare correttamente le classi di minoranza (cfr. recall)

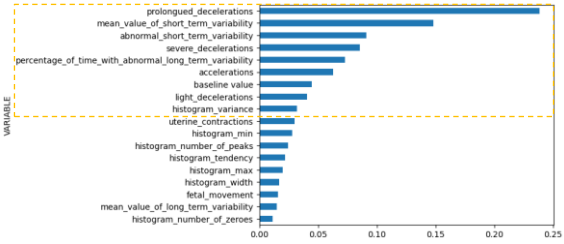
XGBoost è il modello che ottiene i migliori risultati su queste classi (in particolare su Pathological) raggiungendo l'accuracy complessiva maggiore

Fine tuning del modello

Individuato il modello che garantisce le migliori performance, provo ad effettuare un fine tuning per renderlo ancora più accurato applicando alcune possibili modalità di ottimizzazione:

Analisi Feature Importance

- La Feature Importance mi consente di identificare le features che hanno contribuito maggiormente all'allenamento del modello
- Questo mi permette di selezionare un subset di features su cui poter ri-eseguire il modello senza intaccarne le performance



La Confusion matrix e le metriche di valutazione confermano le performance del XGBoost anche con questo subset di features

```
[[324   2   7]
 [  0  29   0]
 [ 13   0  51]]
```

	precision	recall	f1-score	support
Normal	0.96	0.97	0.97	333
Pathological	0.94	1.00	0.97	29
Suspect	0.88	0.80	0.84	64
accuracy			0.95	426
macro avg	0.93	0.92	0.92	426
weighted avg	0.95	0.95	0.95	426

Analisi degli Iper-Parametri con Grid Search

- Una volta costruito il modello posso ottimizzarlo attraverso la regolazione dei suoi iper-parametri
- Seleziono alcuni iper-parametri del mio XGBoost e applico una Grid Search per identificare la migliore combinazione possibile di valore degli iper-parametri
- In questo caso decido di regolare i seguenti iper-parametri:
  - *max_depth: profondità massima dell'albero*
  - *n_estimators: numero dei boosting round*
  - *learning_rate: tasso di apprendimento del boosting*

Una volta ottimizzati gli iper-parametri e identificata la migliore combinazione posso verificare tramite la Confusion matrix e le metriche di valutazione come si comporta il modello

```
[[323   0  10]
 [  2  27   0]
 [ 12   0  52]]
```

	precision	recall	f1-score	support
Normal	0.96	0.97	0.96	333
Pathological	1.00	0.93	0.96	29
Suspect	0.84	0.81	0.83	64
accuracy			0.94	426
macro avg	0.93	0.90	0.92	426
weighted avg	0.94	0.94	0.94	426

Ho testato la Grid Search direttamente sul test set ma sarebbe più corretto splittare nuovamente il dataset tra training, validation e test per evitare il rischio di overfitting

Fine tuning del modello

Per perfezionare il modello ho anche altre possibilità:

Resample dei dati

Come visto in precedenza, il dataset è fortemente sbilanciato verso una classe rispetto alle altre. Per scongiurare il problema posso fare un resample dei dati tramite **sotto campionamento** della classe di maggioranza o **sovra campionamento** di quelle di minoranza

Il sotto campionamento non sembra indicato per il nostro dataset non essendo composto da un numero particolarmente elevato di osservazioni

	precision	recall	f1-score	support
Normal	0.98	0.86	0.92	333
Pathological	0.67	1.00	0.81	29
Suspect	0.63	0.91	0.74	64
accuracy			0.88	426
macro avg	0.76	0.92	0.82	426
weighted avg	0.91	0.88	0.88	426

Applicando il sotto campionamento le classi si bilanciano ma le osservazioni all'interno del train set si riducono da 1700 a 441 e le performance del modello peggiorano

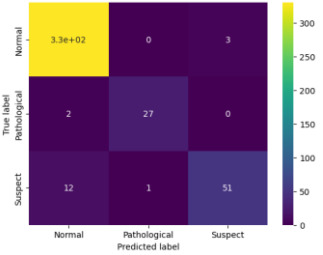
Il sovra campionamento delle classi di minoranza può rappresentare, invece, una migliore soluzione per il modello

	precision	recall	f1-score	support
Normal	0.97	0.96	0.97	333
Pathological	1.00	0.93	0.96	29
Suspect	0.80	0.88	0.84	64
accuracy			0.95	426
macro avg	0.92	0.92	0.92	426
weighted avg	0.95	0.95	0.95	426

Applicando il sovra campionamento le classi si bilanciano aumentando il train set fino a quasi 4000 osservazioni e e le performance del modello rimangono pressoché uguali al XGBoost standard

Applicare un metodo Ensemble

- I metodi Ensemble consentono di combinare insieme più modelli al fine di ottenere un unico modello che massimizzi le performance
- Decido di combinare insieme i tre modelli che hanno ottenuto gli score migliori – KNN, Random Forest e XGBoost



	precision	recall	f1-score	support
Normal	0.96	0.99	0.97	333
Pathological	0.96	0.93	0.95	29
Suspect	0.94	0.80	0.86	64
accuracy			0.96	426
macro avg	0.96	0.91	0.93	426
weighted avg	0.96	0.96	0.96	426

Applicando il metodo Ensemble sui tre modelli riesco ad ottenere uno score leggermente migliore rispetto a quello del solo XGBoost soprattutto grazie alla migliore capacità del modello ensemble di ridurre al minimo i FP della categoria Suspect

CONCLUSIONI

Neural Academy

I modelli sviluppati hanno dimostrato buoni score, anche se lo sbilanciamento del dataset ha influito soprattutto sulle capacità predittive nei confronti della classe *Suspect*

L'esplorazione dei dati e l'analisi dei modelli hanno consentito di identificare alcune variabili molto esplicative per discriminare tra un feto sano e un feto con patologie

Un metodo per perfezionare il modello potrebbe essere quello di utilizzare una cross-validation per lo split tra training, validation e test set e per l'ottimizzazione degli iper-parametri


Grazie per
l’attenzione!