Arrhythmia UCI Dataset
Report per l'Esame di Fondamenti di Machine Learning

Nicola Gutierrez

146875
Ingegneria informatica - sede di Mantova
276269@studenti.unimore.it

Abstract

L'obiettivo è quello di creare un modello di classificazione che tragga le conclusioni dai dati forniti distinguendo tra l'assenza o la presenza di aritmia cardiaca e di, eventualmente, classificarla in una delle 16 classi.

1 Introduzione

I dati utilizzati nel progetto sono stati donati da H. Altay Guvenir e il dataset è disponibile nell'UCI Machine Learning Repository sotto il nome di Arrhythmia Data Set. Il dataset presenta **452** samples, ognuno dei quali rappresenta la cartella di un paziente. Per quanto riguarda le features sono **279**, di cui 206 hanno valore lineare e 73 hanno valore nominale.
La feature ***class*** è la feature target e si suddivide come segue :

Nr	Nome della classe
1	Normal (absence of arrhythmia)
2	Ischemic changes (CAD)
3	Old Anterior Myocardial Infraction
4	Old Inferior Myocardial Infraction
5	Sinus tachycardy
6	Sinus bradycardy
7	Ventricular Premature Contraction (PVC)
8	Supraventricular Premature Contraction
9	Left Boundle branch block
10	Right boundle branch block
11	1.Degree AtrioVentricular block
12	2.Degree AV block
13	3.Degree AV block
14	Left Ventricule hypertrophy
15	Atrial Fibrillation or Flutter
16	Other

Si tratta di un caso di classificazione multiclass. Le classi *1.Degree AtrioVentricular block*, *2.Degree AV block* e *3.Degree AV block* non vengono rappresentate da nessun paziente presente all'interno del dataset, di conseguenza verranno implicitamente ignorate. La classe *other* viene considerata nonostante la sua natura indefinita in modo da non ridurre ulteriormente il numero di samples; verrà considerata secondo il *worst-case concept* quindi verrà fatta ricadere sotto la macro-classe aritmia.

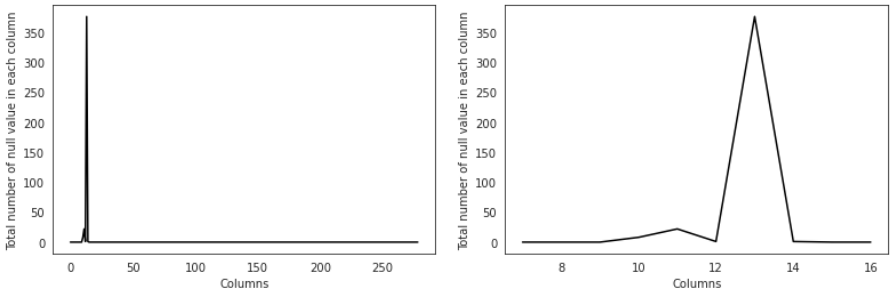
1.1 Gestione del dataset

Il dataset viene trattato in fase di Exploratory Data Analysis & Pre-processing nella sua interezza. Solo in seguito viene suddiviso in due sezioni, la prima rappresenta l'**80%** del dataset ed è dedicata alla fase di training; mentre la seconda rappresenta il rimanente **20%** ed è dedicata alla fase di testing. La suddivizione avviene in seguito ad uno *shuffle* dei dati e attraverso l'opzione *stratify* in modo da assicurarsi una distribuzione bilanciata delle classi tra i sub-set.

2 Exploratory Data Analysis & Pre-Processing

2.1 Gestione valori mancanti

Secondo le informazioni fornite insieme al dataset, i valori mancanti sono rappresentati con il simbolo '?'.



(a) visione di insieme. (b) vista dell'intorno del picco.

Figure 1: valori mancanti per ogni colonna.

Le classi che presentano valori mancanti sono *T*, *P*, *J*, *QRST* e *Heart Rate*.

Nome	**Valori mancanti**	**Soluzione**
T	8	Sostituzione attraverso tecniche di *imputation*
P	22	Sostituzione attraverso tecniche di *imputation*
J	1	Sostituzione attraverso tecniche di *imputation*
QRST	376	Feature scartata, l'**83%** dei suoi valori risulta mancante
Heart Rate	1	Sostituzione attraverso tecniche di *imputation*

In particolare, si è scelta la modalità più immediata per sostituire i valori indefiniti, cioè con la mediana dei valori della colonna, data la bassa percentuale di valori da determinare.

2.2 Gestione outliers

Dal controllo della presenza di valori non veritieri si è riscontrato che nella maggior parte del dataset i valori sono alquanto uniformi e plausibili, l'unico caso degno di nota è stato riportato di seguente, dove si può notare la presenza di alcuni valori fuori scala all'interno degli attributi *Height* e *Weight*.Gli ulteriori grafici utilizzati sono all'interno della cartella *extra/outliers*.

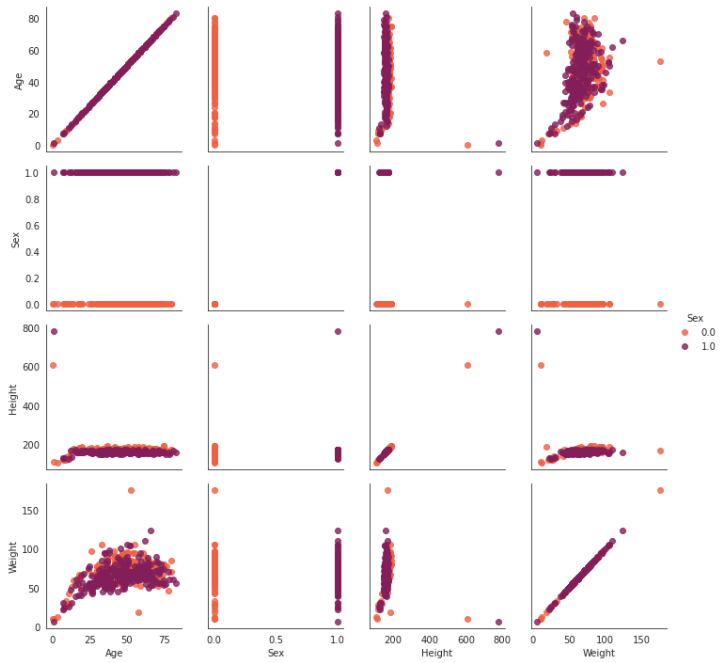


Figure 2: Controllo presenza di outliers.

2.3 Exploratory Data Analysis

Analizzando la distribuzione totale delle feature ***class*** si può notare come il dataset sia **fortemente sbilanciato** verso i casi *normali* e come le classi riferite all'*aritmia* non siano particolarmente equilibrate. In generale, la distribuzione binaria tra casi *normali* e casi di *aritmia* è piuttosto equilibrata. Con queste premesse un'ottima soluzione sarebbe quella di ridimensionare il dataset, eliminando o aggiungendo samples in modo da avere una distribuzione equa. Tuttavia, la prima soluzione non è stata presa in considerazione in quanto il dataset offre di per sé pochi samples (se considerati rispetto alle feature), mentre la seconda semplicemente non è possibile.

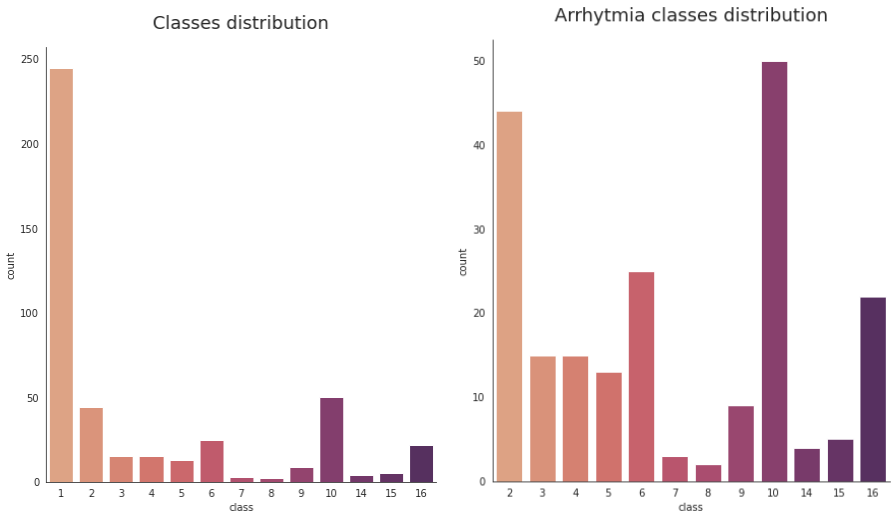


Figure 3: Distribuzione delle classi

Per quanto riguarda la correlazione tra feature :

- **185** coppie di attributi hanno una correlazione maggiore di |0.7|;
- **76** coppie di attributi hanno una correlazione maggiore di |0.8|;
- **19** coppie di attributi hanno una correlazione maggiore di |0.9|;

Sulla base di questi risultati si è deciso di rimuovere le **19** coppie con correlazione maggiore. Tra queste coppie solo la feature *AVF212* si ripropone due volte, quindi in totale vengono rimosse **18** feature, portando il numero di attributi del nuovo dataset a **261**

3 Selezione del modello

3.1 Modelli candidati

- **K-Nearest Neighbours** : scelto in quanto semplice da implementare e veloce nell'esecuzione dato il numero accessibile di samples;
- **Decision Tree Classifier** : scelto nonostante la complessità del dataset in quanto l'implementazione richiede poche risorsee risulta veloce nell'esecuzione;
- **Support Vector Machine** : scelto come riferimento in caso in cui si presentasse overfitting e perché permette di ottenere un buon riscontro in casistiche multiclass. Inoltre si prevede di utilizzare la libreria `sklearn`, semplificando la selezione degli iperparametri;
- **Softmax Regression** : scelto nonostante il numero di feature presenti nel dataset per completezza e per permettere un confronto completo;

3.2 Cross validation

Per scegliere la migliore combinazione di iperparametri per ogni modello, è stata sfruttata la *k-fold validation*, data anche la natura sbilanciata del dataset. In particolare, è stata utilizzata la classe `GridSearchCV` di `sklearn` con $cv = 7$ e $score = accuracy$.
Le configurazioni di iperparametri sono state scelte euristicamente per ogni modello.

- **K-Nearest Neighbours** : `n_neighbors` : **3**
- **Decision Tree Classifier** : `max_depth` : **10**, `criterion` : **gini**
- **Support Vector Machine** : `C` : **100**, `gamma` : **0.001**, `kernel` : **rbf**
- **Softmax Regression** : `penalty` : **l1**, `C` : **1.0**

3.3 Ensemble : Stacking Classifier

Oltre ai modelli sopra citati si è deciso di utilizzare il meta-algoritmo *stacking*, combinando i tre modelli con la *accuracy* maggiore, quindi *SVM*, *KNN* e *Decision Tree Classifier*.

3.4 Valutazione dei modelli

La valutazione è avvenuta tramite l'utilizzo della funzione `cross_validate` di `sklearn`.

Modello	Accuracy
Stacking Classifier	75%
Support Vector Machine	70%
Decision Tree Classifier	64%
K-Nearest Neighbours	60%
Softmax Regression	58%

4 Fase di testing

4.1 ToDo

Aggiungere considerazioni finali ed eventuali confusion matrix.