COVID-19 PREDICTION BASED ON BLOOD TEST RESULT

A Machine Learning project

Gruppo:

Marchi Mattia 817587 @M-Marchi

Stoppa Miguel 820506 @mig1214

Tomasoni Lorenzo 829906 @lTomasoni

Gitlab repository:

Gitlab repository: https://gitlab.com/gruppo-pss/2020_machinelearning_project.git

Sommario

1 Introduzione

Dal dicembre 2019 il mondo sta lottando contro una pandemia dovuta al virus nominato SARS-CoV-2 (Coronavirus 2 da sindrome respiratoria acuta grave). Inizialmente il virus è stato localizzato nella città di Wuhan in Cina, per poi diffondersi in tutto il globo. I dati aggiornati al 18 Gennaio 2021 riferiscono di più di 95.1 milioni di casi rilevati in 190 paesi coinvolti e di oltre 2.03 milioni di morti causate da tale virus.

I sintomi di COVID-19 sono molto variabili, posso variare da nessun sentore al presentarsi di malattie molto gravi. Il virus si trasmette per via aerea, molto spesso tramite le goccioline respiratorie. Per limitarne la trasmissione devono essere prese precauzioni, come mantenere la distanza interpersonale di almeno 1,5 metri e tenere comportamenti corretti sul piano dell'igiene personale (lavare e disinfettare periodicamente le mani, starnutire o tossire in un fazzoletto o nell'incavo del gomito, indossare mascherine e guanti) e ambientale (rinnovare spesso l'aria negli ambienti chiusi aprendo le finestre e mantenere gli ambienti puliti).

In questo contesto di grave emergenza sanitaria risulta essere di massima importanza l'identificazione dei pazienti che necessitano di essere ricoverati in terapia intensiva, l'identificazione di quei pazienti che, a causa di insufficienze respiratorie gravi, hanno necessità di essere assistiti dalla ventilazione meccanica (VM) e infine, l'identificazione dei pazienti che hanno necessità di assistenza domiciliare, in quanto non autosufficienti.

Al fine di proporre un piccolo aiuto nell'indirizzare tali problemi, questa relazione descrive la procedura seguita per la realizzazione di un tool di machine learning che si pone l'obiettivo di classificare i pazienti, sulla base dei rispettivi risultati degli esami del sangue a cui essi sono stati sottoposti, tra casi positivi di COVID-19 e casi negativi.

L'obiettivo, come detto, è stato quello di realizzare un tool di machine learning in grado di identificare se un paziente risulta essere positivo al COVID-19, prendendo in input i risultati degli esami del sangue a cui il paziente stesso è stato sottoposto. A tal scopo sono stati addestrati tre modelli di machine learning supervisionati. Essi sono stati confrontati, con l'obiettivo di individuare il modello più efficace nell'eseguire tale classificazione. Per eseguire un confronto che fosse il più completo possibile, per ogni modello sono stati valutati, oltre all'efficacia, altri fattori, quali explainability ed efficienza. È essenziale, infatti, che un modello di classificazione, oltre ad avere un buon livello di efficacia, sia anche efficiente e che quindi converga in un numero di iterazioni ragionevole, evitando di consumare eccessivamente le risorse computazionali a disposizione. Oltre a ciò, un altro aspetto che si sta rivelando essere sempre più critico nella scelta del modello di machine learning da addestrare per effettuare predizioni e/o classificazioni è la model explainability. Un modello è associato ad un alto livello di model explainability se è possibile comprendere e conseguentemente spiegare in modo semplice per quale ragione il modello effettua una determinata classificazione.

2 Descrizione del dataset

Per realizzare l'obiettivo posto, si è deciso di utilizzare un dataset acquisito dalla nota community di data scientists kaggle[1]. Esso contiene i dati anonimizzati di pazienti dell'ospedale Israelita Albert Einstein di San Paolo, in Brasile. I soggetti a cui si riferiscono questi dati sono stati sottoposti a un'apposita analisi clinica, per mezzo della quale sono stati prelevati campioni di sangue, utili per eseguire un test molecolare (SARS-CoV-2 RT-PCR), oltre ad altri test clinici utili per individuare l'eventuale presenza del COVID-19.

Tutti i dati presenti nel dataset sono in forma anonimizzata, l'autore di tale lavoro ha seguito le più comuni best practice internazionali per tale scopo. Tutti i dati sono in forma standardizzata, con media pari a zero e deviazione standard pari ad una unità.

Il dataset è composto da 5645 istanze (una per ogni paziente esaminato) e 127 feature. Tra quest'ultime compare anche quella che è stata presa in considerazione come target (label) per i modelli di apprendimento automatico supervisionati addestrati, ovvero la feature denominata "SARS-Cov-2 exam result", che descrive l'esito del tampone molecolare effettuato per ciascun paziente, espresso mediante una stringa denominata "negative", in caso di risultato negativo; "positive" in caso di risultato positivo. Oltre a questo valore le colonne del dataset contengono informazioni relative all'età del paziente (espressa in quantili, per rispettare i criteri di anonimizzazione internazionali), al fatto che il paziente sia stato o meno ammesso al reparto di medicina regolare, a quello di terapia semi-intensiva, oppure a quello di terapia intensiva, oltre ai vari parametri sanguinei, ottenuti per mezzo dei test condotti in ospedale.

Come già affermato nella parte di introduzione di questo documento, l'obiettivo del progetto è stato quello di realizzare un tool di machine learning che permettesse di classificare un paziente tra quelli positivi al COVID-19 e quelli negativi, ricevendo in input i parametri ematici dello stesso. Il dataset utilizzato risulta essere sovrabbondante di informazioni rispetto a tale obiettivo. Esso contiene infatti numerosi ulteriori dati oltre a quelli relativi ai parametri ematici, come dati derivanti dall'analisi delle urine, positività del paziente ad altre patologie e così via. Per questo motivo, come documentato nella prossima sezione, la prima operazione compiuta è stata quella di prelevare i dati di interesse per il nostro progetto dal dataset di partenza, ignorando gli altri dati sovrabbondanti.

1: Link al dataset (https://www.kaggle.com/einsteindata4u/covid19/version/4).

3 Analisi esplorativa del dataset e Data Cleaning

Come già accennato nel paragrafo precedente, la prima operazione compiuta è stata quella di prelevare dal dataset di riferimento i parametri ematici di ciascun paziente, non considerando le informazioni del dataset che risultano essere sovrabbondanti per l'obiettivo di questo progetto, che è quello di classificare un paziente come positivo/negativo, solamente sulla base dei suoi parametri ematici. Nel dettaglio per eseguire questa operazione sono state inserite in un apposito dataframe le colonne rappresentanti i parametri sanguinei nel dataset, eseguendo il seguente comando:

df_active <- df[, c(2,3,7:20)]

Si è deciso di cambiare i nomi di alcune feature in modo tale che essi fossero più concisi e chiari, le feature mantenute sono le seguenti:

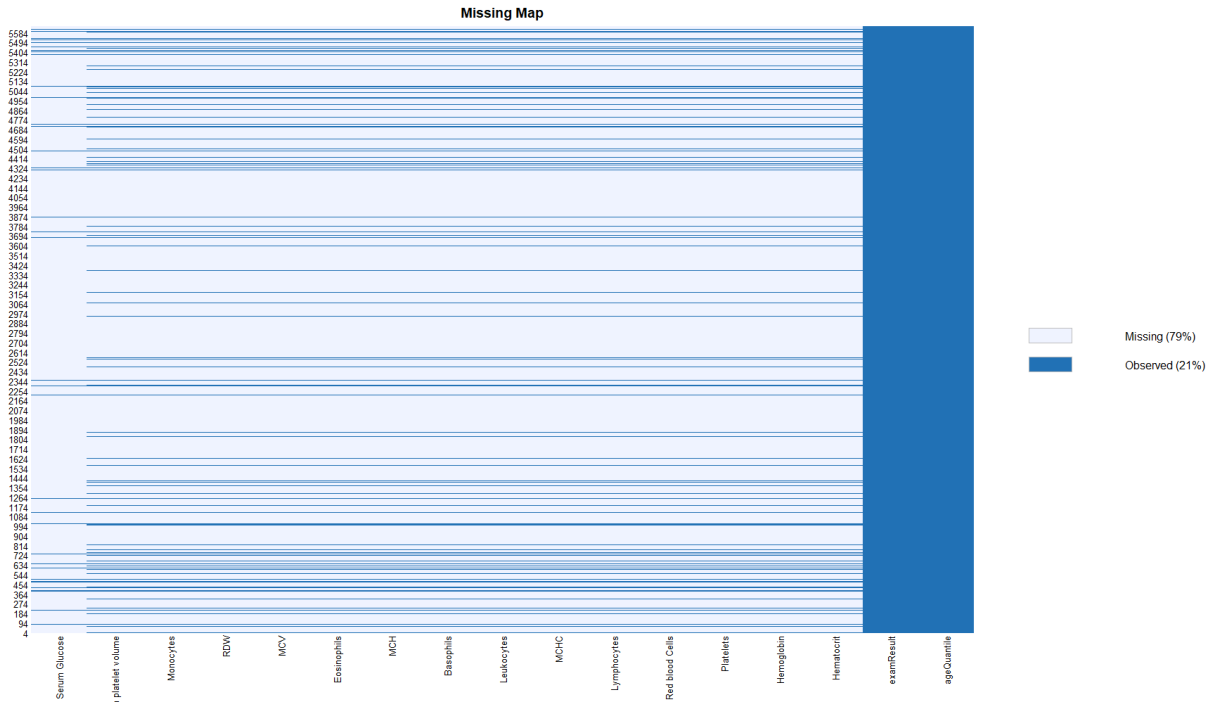
- ageQuantile: variabile numerica che rappresenta l'età espressa in quantili.
- examResult: è una variabile categorica che indica il risultato del tampone molecolare al quale ogni paziente è stato sottoposto. Rappresenta la variabile target che i modelli dovranno predire per ogni istanza.
- Hematocrit;
- Hemoglobin;
- Platelets;
- Mean Platelets;
- Red blood cells;
- Lymphcytes;
- MCHC: Mean Corpuscolar Hemoglobin Concentration;
- Leukocytes;
- Basophis;
- MCH: Mean Corpuscolar Hemoglobin;
- Eosimophils;
- MCV: Mean Corpuscolar Volume;
- Monocytes;
- RDW: Red Blood-Cell Distribution Width.

Una volta eseguita questa operazione preliminare, si è proseguito con il processo di analisi esplorativa del dataset, al fine di comprendere come esso fosse strutturato, come e di quale tipo sono le feature ed eventualmente comprendere quali relazioni intercorrono tra di esse.

Dopodiché per comprendere come i dati contenuti nel dataset sono distribuiti, sono state applicate sul dataset le funzioni head, str e summary. Da questa operazione si è notato che molti dei dati presenti nel dataset risultano essere valori NA, ovvero dati mancanti. In seguito, è stato possibile calcolare il numero di NA presenti in ciascuna colonna e il numero percentuale di NA sempre relativi a ciascuna colonna, al fine di fornire una stima numerica relativa al numero di valori mancanti presenti nel dataset in esame.

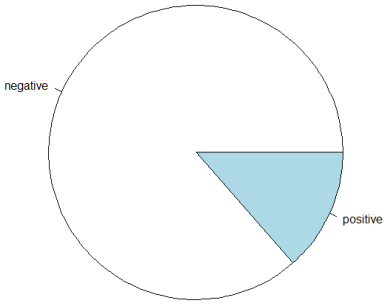
Come già potuto ipotizzare con l'esecuzione della funzione **summary**, il numero totale di NA per ogni colonna risulta essere notevole. La percentuale di valori mancanti per alcune colonne è risultato essere superiore al 90%, mentre per molte altre si attesta nell'intervallo [70%-89%]. Questo dimostra che il dataset presenta molti valori mancanti.

Per effettuare una stima visiva del numero di valori NA presenti nel dataset, è stata creata, per mezzo dell'esecuzione del comando ***missmap(df, main="Missing Map")***, una missmap, visibile nella figura sottostante.



In seguito, si è proceduto con l'eliminazione degli elementi NA, presenti nel dataset. Il numero di istanze, in seguito a questa operazione, a causa del numero elevato di valori mancanti presenti nel dataset, si è ridotto in maniera sensibile. Sono rimasti 598 valori.

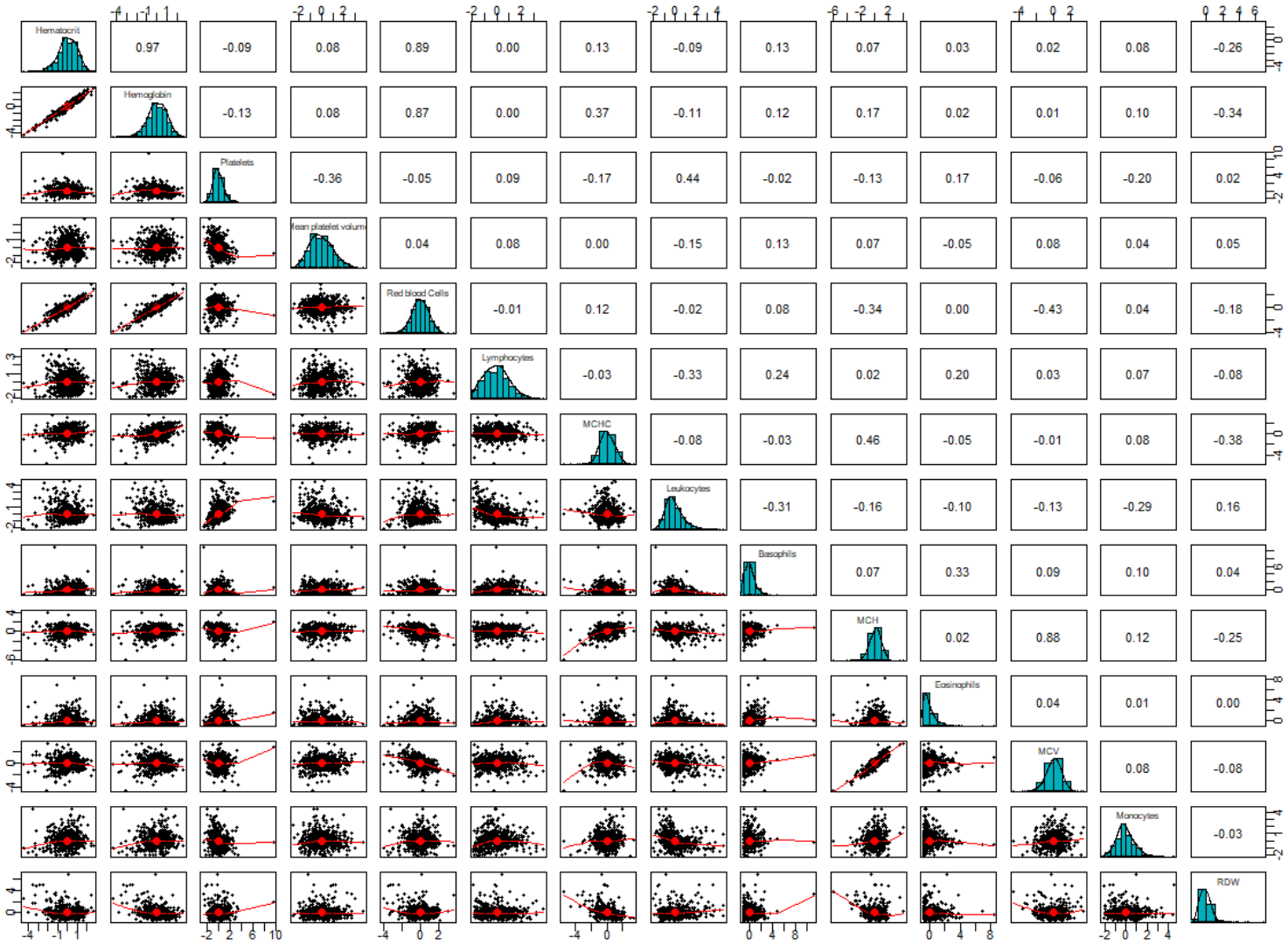
Successivamente, è stato eseguito il comando **table** per comprendere la distribuzione dei casi positivi e dei casi negativi nel dataset. Oltre a ciò, è stato realizzato un Pie chart per visualizzare questa statistica, il risultato ottenuto è visibile nella figura sottostante:



Come si può notare, il dataset risulta essere piuttosto sbilanciato; nel dettaglio, presenta 517 istanze negative e 81 positive.

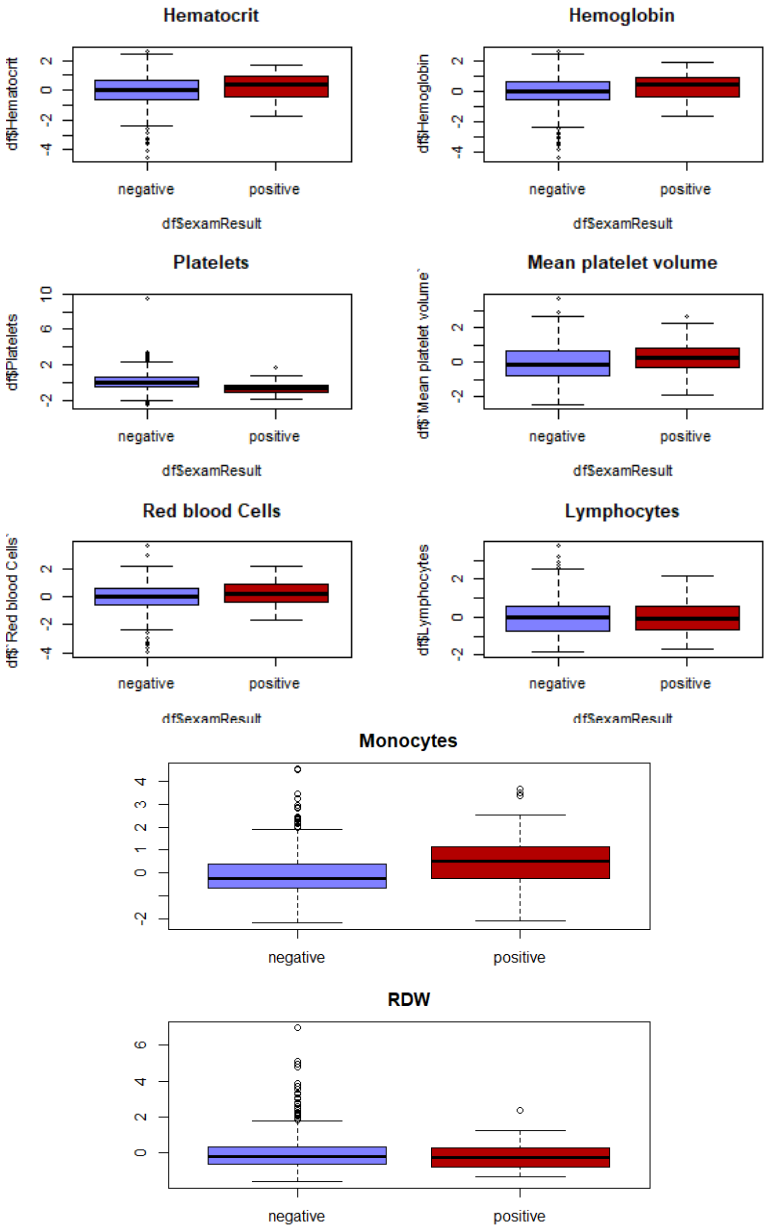
Il passo successivo è stato quello di trasformare la variabile categorica "examResult" in fattori (positive/negative), attraverso la funzione **as.factor()**.

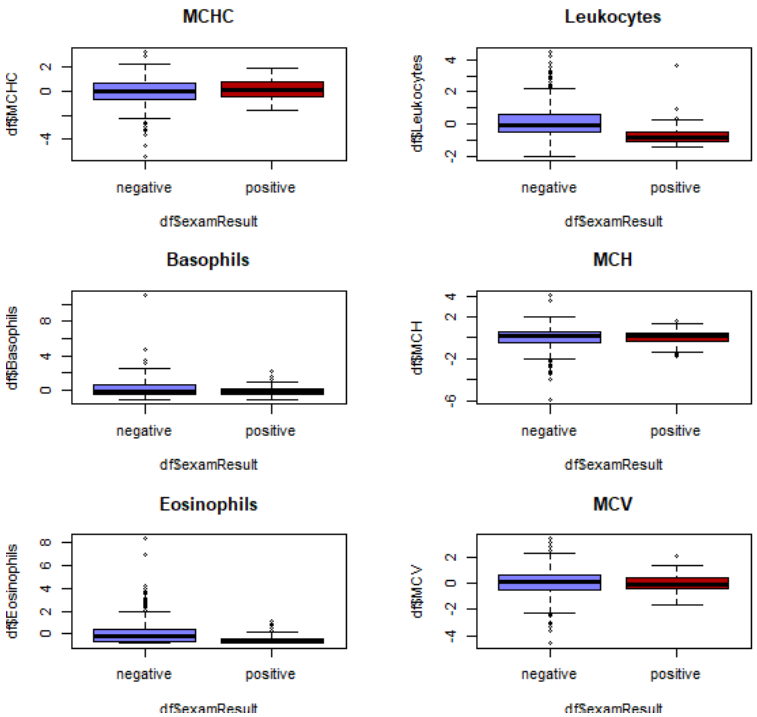
In seguito, è stato verificato il livello di correlazione tra le variabili:



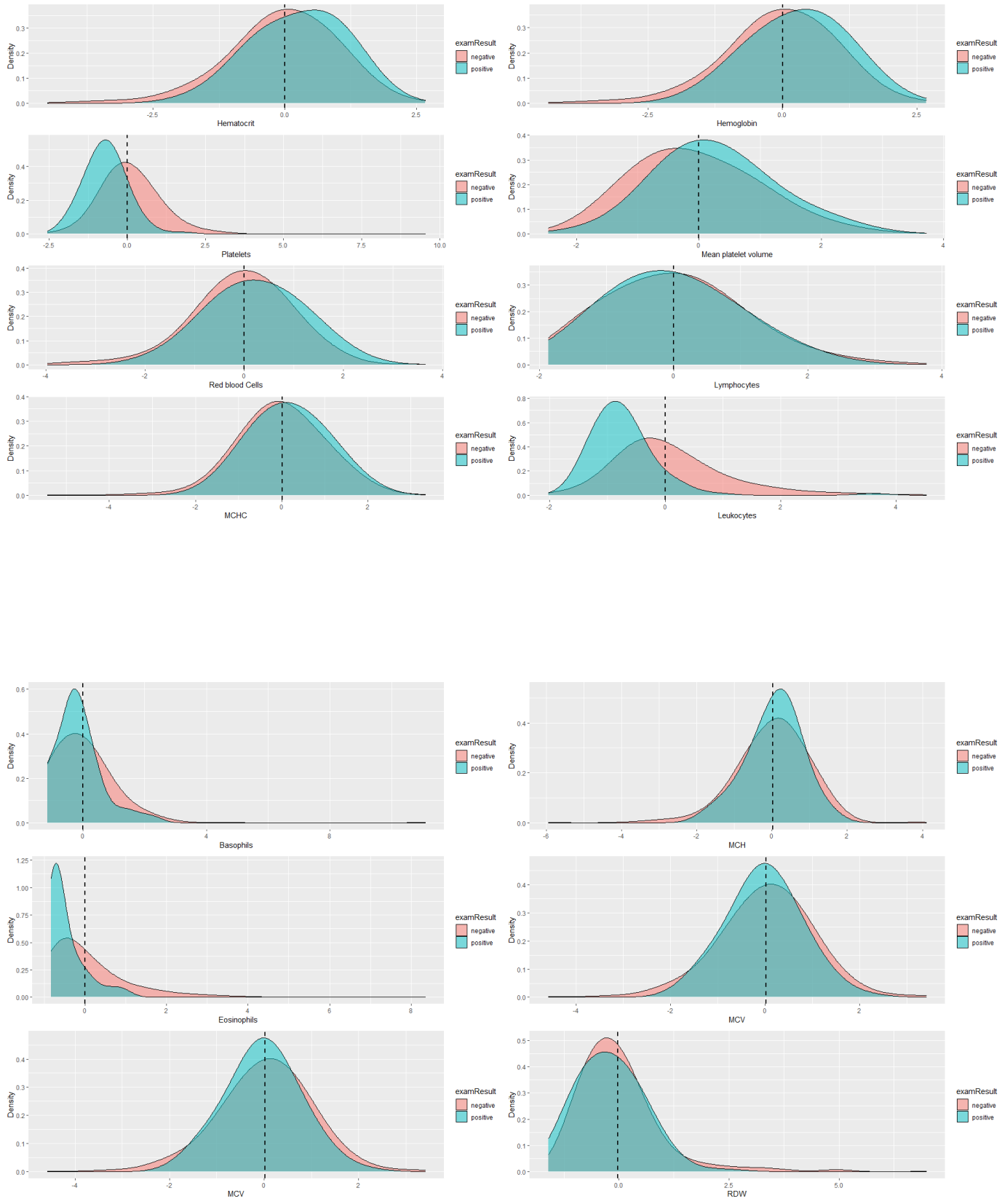
Successivamente, sono stati realizzati diversi boxplot per mettere in relazione la differenza delle distribuzioni di ciascuna variabile numerica nel caso degli individui positivi e nel caso di quelli risultati negativi. In seguito, per eseguire un'analisi più approfondita delle variabili numeriche sono stati realizzati alcuni density plot relativi, anche in questo caso, ad entrambe le categorie.

Entrambe le tipologie di visualizzazione sono rappresentate nelle figure presenti nelle pagine successive.



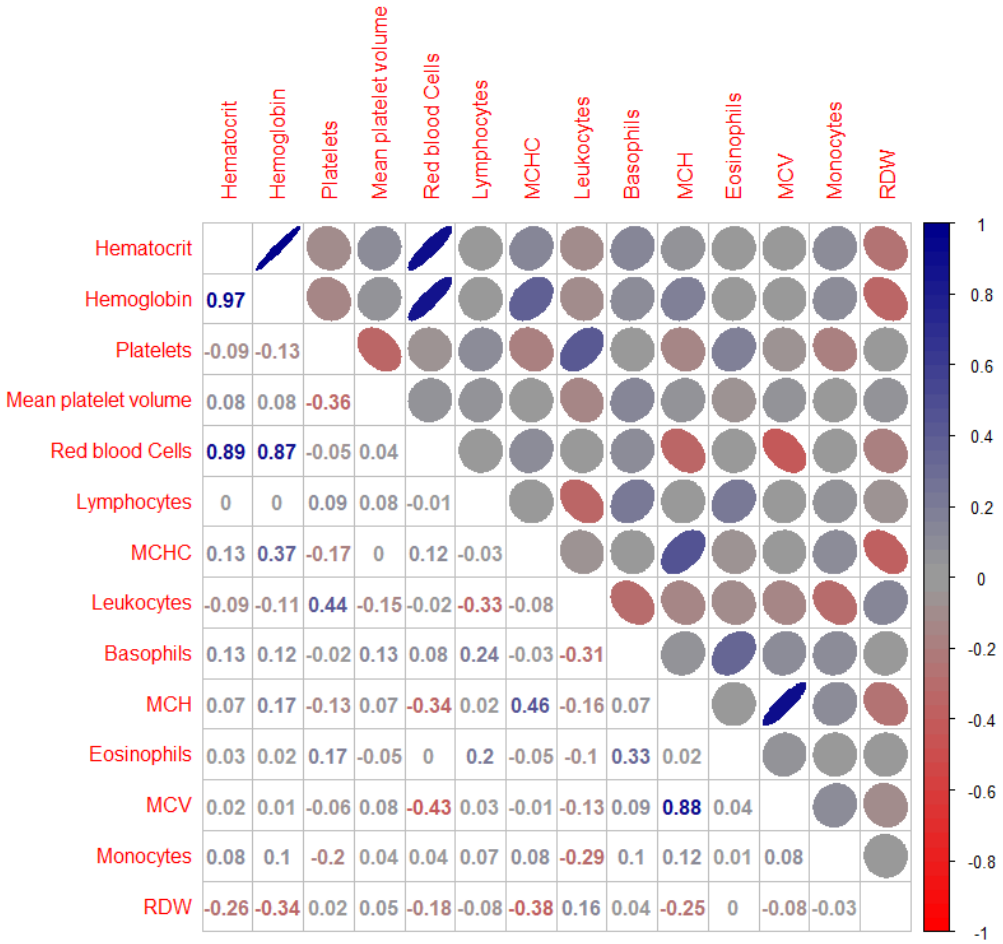


Boxplot che mettono in relazione la distribuzione delle variabili numeriche relative agli individui positivi e negativi.



Density plot

In seguito, è stata realizzata una correlation matrix per visualizzare i livelli di correlazione tra le variabili considerate. Il risultato è visibile nella figura sottostante.



Correlation matrix

4 Realizzazione dei modelli

Con l'obiettivo di realizzare un tool di machine learning che permettesse di classificare un paziente come positivo o negativo al COVID-19 sulla base dei parametri sanguinei, si è deciso di addestrare i seguenti tre modelli supervisionati:

- Decision Tree: esso è usato per la classificazione con l'obiettivo di creare un modello predittivo che prenda decisioni creando una struttura dati ad albero, dove ogni nodo è costituito da un input feature.
- Neural Network: viene creato tramite una rete di neuroni artificiali, collegati in modo orientato tra di loro, così che l'output calcolato da ciascun neurone rappresenti l'input per lo strato successivo.
- Naive Bayes: metodo probabilistico che stima l'ipotesi a posteriori più probabile.

Sono stati scelti questi modelli per le seguenti ragioni:

- Decision Tree: è facile interpretare e comprendere i risultati che questo modello fornisce anche attraverso una rappresentazione visuale. Non richiede una fase di data preprocessing particolarmente articolata ed è un modello molto veloce da addestrare.
- Neural Network: è un modello tollerante agli errori e al rumore dei dati, inoltre è facilmente aggiornabile con nuovi dati e offre una elevata capacità di generalizzazione.
- Naive Bayes: algoritmo semplice da implementare, veloce nel training e particolarmente performante su dataset non voluminosi. Oltre a ciò, esso è generalmente più performante in termini di tempo di calcolo rispetto alla Neural Network (O(pn)).

La strategia iniziale è stata quella di applicare la Principal Component Analysis (PCA), utilizzando come metodo di validazione e tuning la 10-fold cross validation. I risultati ottenuti, per ciascuno dei tre modelli, seguendo tale approccio non sono stati soddisfacenti, pertanto si è ritenuto utile applicare la tecnica di feature selection al dataset di partenza, in sostituzione alla PCA applicata nel precedente approccio. Anche in questo caso è stato eseguito tuning dei modelli tramite la 10-fold cross validation, ottenendo risultati decisamente migliori in termini di accuratezza.

Nei prossimi paragrafi verranno descritti nel dettaglio i due approcci seguiti e i relativi risultati.

4.1 PCA

Il primo approccio seguito per addestrare i tre modelli scelti ha previsto come operazione preliminare l'applicazione di una tecnica chiamata principal component analysis sul dataset di partenza.

La ragione per cui è stata condotta questa operazione risiede nel fatto che non tutti gli attributi hanno lo stesso valore predittivo, alcuni non sono per nulla significativi, o ancora peggio, potrebbero impattare in maniera negativa sul modello di apprendimento. Per questa ragione e con l'obiettivo di produrre modelli che fossero efficienti e computazionalmente non troppo onerosi, si è ritenuto necessario eseguire l'operazione di feature extraction, mediante PCA.

Più dettagliatamente, le ragioni che hanno portato alla decisione di applicare questa tecnica sono:

- Si è cercato di mantenere un rapporto tra dati di training e numero di covariate basso. Questo è un aspetto fondamentale per ottenere un modello che possa essere efficace nell'esecuzione delle classificazioni delle istanze. Siccome il numero di dati di training, in seguito all'eliminazione dei dati NA, si è ridotto a 598 istanze e il numero di covariate selezionate è pari a 15, si è ritenuto che il rapporto tra il numero di istanze e il numero di covariate fosse troppo elevato.

- Le risorse computazionali sono ovviamente limitate e uno degli obiettivi del progetto è stato quello di realizzare modelli efficienti, oltre che efficaci. Mediante la riduzione delle componenti, si è ritenuto di poter raggiungere questo obiettivo.

- Tra le 15 covariate selezionate, con ogni probabilità, parte di esse sono ridondanti e pertanto eliminabili per semplificare l'addestramento dei modelli.

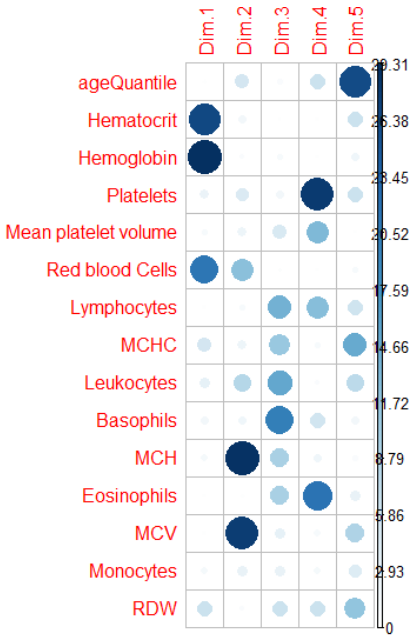
- La maledizione della dimensionalità (curse of dimensionality). Al crescere del numero di attributi, contenuti nel dataset, dovrebbe crescere in modo esponenziale anche il numero di dati per realizzare modelli efficaci.

In estrema sintesi, è possibile riassumere le ragioni per cui è stato deciso di effettuare l'operazione di feature extraction, mediante PCA, in due punti: migliorare le performance e ridurre la complessità computazionale dei modelli di classificazione, che si ha l'obiettivo di realizzare.

Tramite PCA si è cercato di passare da uno spazio di dimensione R^n ad uno spazio di dimensione R^m; con $m<n$. Per effettuare tale passaggio, la PCA permette di eseguire una trasformazione lineare, che adatta il dataset ad un nuovo sistema di coordinate in cui la varianza più significativa si trova alla prima coordinata e ogni coordinata subsequenziale è ortogonale all'ultima e ha meno varianza. In altre parole, si trasforma un insieme di variabili correlate in un insieme componenti principali non correlate.

Come già affermato nel capitolo di descrizione del dataset, tutte le feature sono già normalizzate quindi non è stato necessario svolgere questa operazione. Quindi è stata applicata la PCA ottenendo una matrice nello spazio trasformato e perciò, per tornare a quello originale mantenendo però la riduzione delle feature, è stato necessario effettuare il prodotto tra la matrice di partenza con quella ricavata tramite la proiezione ottenuta attraverso PCA.

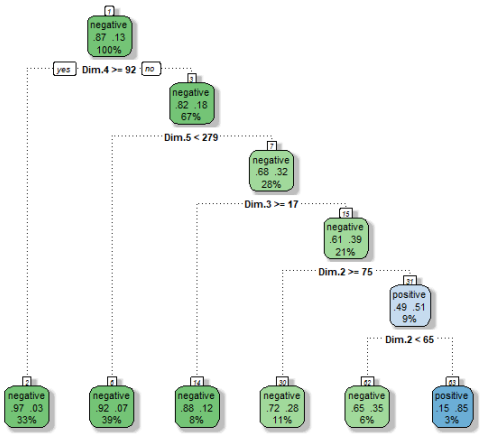
Così facendo abbiamo trasformato il dataset di partenza con 14 dimensioni a uno nuovo con sole 5 dimensioni, che insieme spiegavano più del 70% della varianza.



Dall'immagine soprastante possiamo visualizzare le feature del dataset di partenza più significative in ognuna delle 5 dimensioni che compongono il nuovo spazio.

Il dataset così ottenuto è stato diviso in trainset e testset, seguendo la proporzione 70/30.

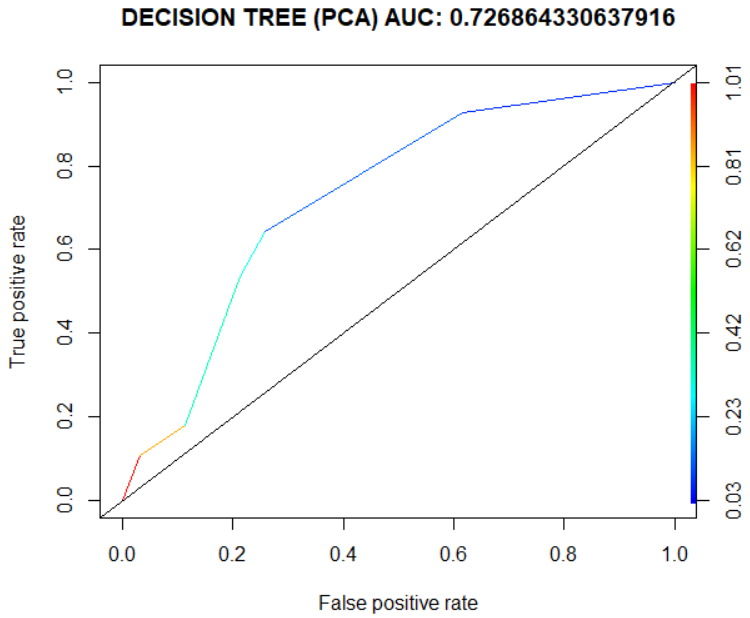
4.1.1 PCA: Decision Tree



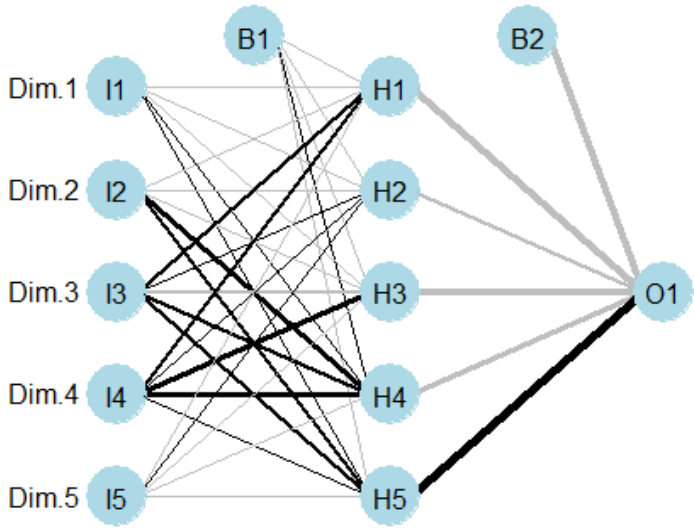

```
Confusion Matrix and Statistics

          Reference
Prediction negative positive
  negative      154       25
  positive        5        3

               Accuracy : 0.8396
                 95% CI : (0.779, 0.8891)
    No Information Rate : 0.8503
    P-Value [Acc > NIR] : 0.7018938

                  Kappa : 0.1073

 Mcnemar's Test P-Value : 0.0005226

            Sensitivity : 0.10714
            Specificity : 0.96855
         Pos Pred Value : 0.37500
         Neg Pred Value : 0.86034
             Prevalence : 0.14973
         Detection Rate : 0.01604
   Detection Prevalence : 0.04278
      Balanced Accuracy : 0.53785

       'Positive' Class : positive
```

Eseguendo la 10-fold cross validation, l'albero decisionale ottimale ottenuto è quello visibile nella figura soprastante con un livello di accuratezza pari al 83.96%, precision 37.5%, recall 10.71% e f1 16.66%, sensitività 10.71%, specificità 96.85%.

Nella seguente immagine viene mostrata la curva ROC con il relativo valore AUC pari a 72.68%



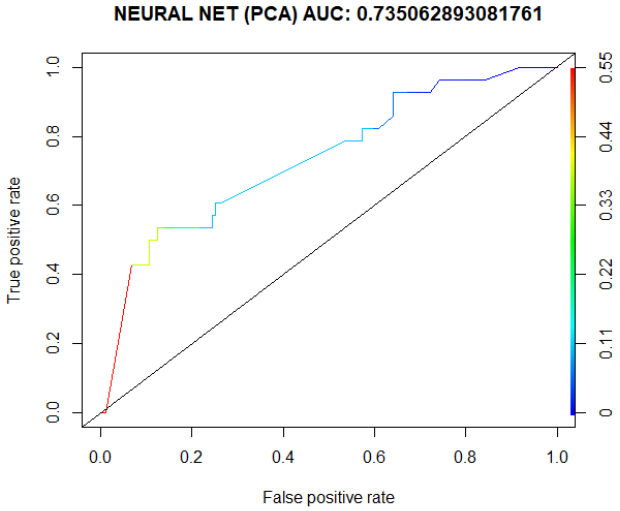
4.1.2 PCA: Neural Network



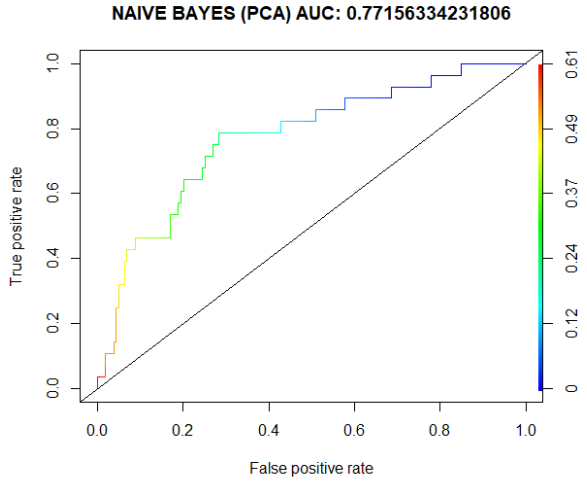

```
Confusion Matrix and Statistics

          Reference
Prediction negative positive
  negative      157       28
  positive        2        0

               Accuracy : 0.8396
                 95% CI : (0.779, 0.8891)
    No Information Rate : 0.8503
    P-Value [Acc > NIR] : 0.7019

                  Kappa : -0.0204

 Mcnemar's Test P-Value : 5.01e-06

            Sensitivity : 0.0000
            Specificity : 0.9874
         Pos Pred Value : 0.0000
         Neg Pred Value : 0.8486
             Prevalence : 0.1497
         Detection Rate : 0.0000
   Detection Prevalence : 0.0107
      Balanced Accuracy : 0.4937

       'Positive' Class : positive
```

Eseguendo la 10-fold cross validation, la rete neurale ottimale ottenuta è quella visibile nella figura soprastante con un livello di accuratezza pari al 83.96%, precision 0%, recall 0%, sensitività 0 %, specificità 96.85%. Questi valori sono causati da una classificazione quasi del tutto negativa di tutte le istanze.

Nella seguente immagine viene mostrata la curva ROC con il relativo valore AUC pari a 73.50%.



4.1.3 PCA: Naive Bayes




```
Confusion Matrix and Statistics

          Reference
Prediction negative positive
  negative      152       22
  positive        7        6

               Accuracy : 0.8449
                 95% CI : (0.7849, 0.8936)
    No Information Rate : 0.8503
    P-Value [Acc > NIR] : 0.62890

                  Kappa : 0.2185

 Mcnemar's Test P-Value : 0.00933

            Sensitivity : 0.21429
            Specificity : 0.95597
         Pos Pred Value : 0.46154
         Neg Pred Value : 0.87356
             Prevalence : 0.14973
         Detection Rate : 0.03209
   Detection Prevalence : 0.06952
      Balanced Accuracy : 0.58513

       'Positive' Class : positive
```

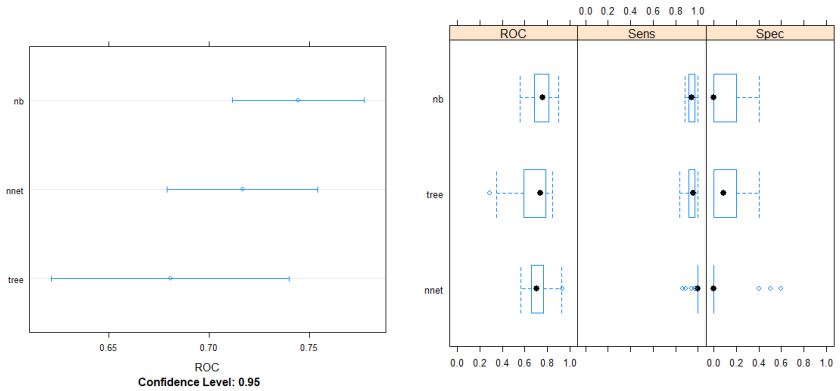
Eseguendo la 10-fold cross validation, il modello Naive Bayes ottenuto ha un livello di accuratezza pari al 84.49%, precision 46.15%, recall 21.42%, f1 29.26%, sensitività 21.42%, specificità 95.59%.

Viene mostrata sopra la curva ROC con il relativo valore AUC pari a 77.15%.

4.1.4 Confronto modelli PCA

Pur aspettandoci risultati non ottimali a causa della ridotta dimensione del dataset di partenza, utilizzando il dataset trasformato con la PCA, i tre modelli addestrati hanno ottenuto risultati non soddisfacenti. Per questo motivo si è deciso di eseguire un'ulteriore prova per verificare un eventuale miglioramento dei modelli utilizzando un approccio Leave One Out cross validation. Quest'ultimo non ha portato alcun miglioramento, peggiorando invece le performance computazionali. Per queste ragioni questo approccio è stato scartato nei modelli finali.

Comparando le curve ROC dei modelli applicati otteniamo i seguenti valori:



Dai grafici notiamo che i valori delle curve ROC dei tre modelli sono circa comparabili perché i rispettivi intervalli di confidenza si sovrappongono. La stessa cosa si può dire per la specificità e la sensitività. Per la specificità abbiamo valori molto bassi mentre per la sensitività sono molto alti a causa delle poche classificazioni positive.

Sono stati valutati anche i tempi necessari per addestrare i tre modelli e come si può notare dalla figura sottostante, il Decision Tree è quello computazionalmente più efficiente, la Neural Network è quella che è risultata più onerosa nei tempi di calcolo e Naive Bayes è risultato solo leggermente meno efficiente dell'albero di decisione.

	Everything	FinalModel	Prediction
tree	2.20	0.02	NA
nnet	11.58	0.05	NA
nb	4.42	0.00	NA

Infine, per quanto riguarda l'interpretabilità dei tre modelli il Decision Tree è il modello più facilmente interpretabile rispetto agli altri due modelli a discapito dell'efficacia.

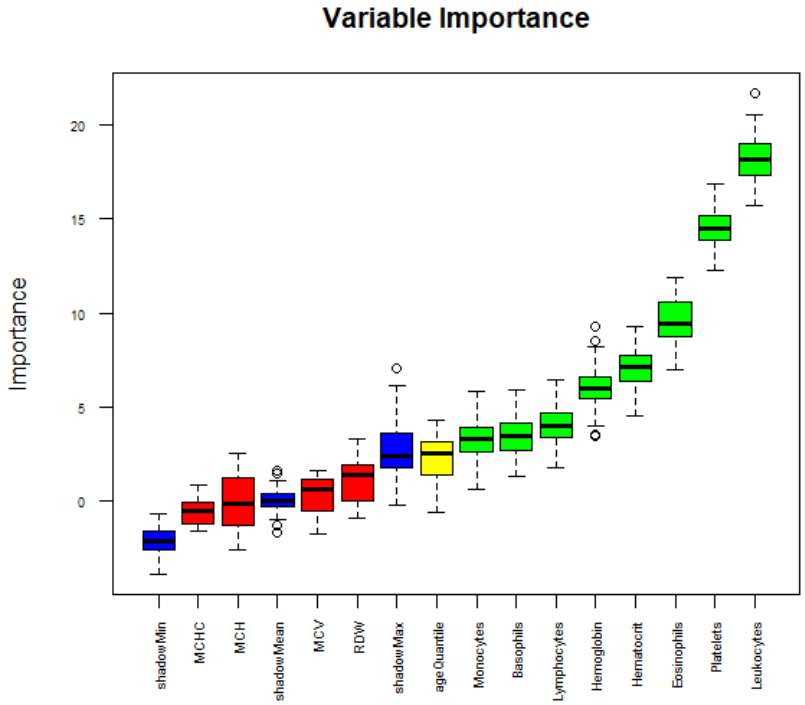
A causa degli scarsi risultati è stato quindi ritenuto opportuno, a fronte dei risultati ottenuti, utilizzare un metodo di feature selection in sostituzione della feature extraction di PCA.

4.2 Feature Selection

La seconda strategia è consistita nella realizzazione della feature selection sul dataset di partenza. Questo processo di riduzione consiste nell'analisi o nell'individuazione delle caratteristiche maggiormente significative rispetto alle altre.

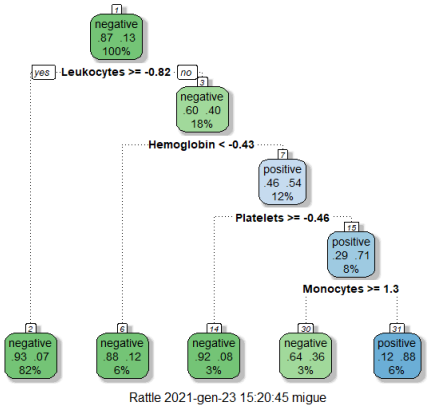
A differenza della PCA descritta precedentemente, la feature selection permette di rimuovere le feature meno rilevanti mentre, come già detto, la PCA trasforma il dataset di partenza in un nuovo spazio dimensionale.

Per eseguire tale approccio è stato utilizzato il package "Boruta" il quale compara iterativamente le importanze delle feature del dataset con le importanze degli attributi *shadow*, che sono creati mescolando quelli originali. Gli attributi che hanno un livello di importanza significativamente peggiore rispetto agli attributi shadows sono successivamente eliminati. Invece, gli attributi che hanno una importanza significativamente maggiore rispetto agli shadows vengono classificati come *confirmed*. Gli attributi shadows sono ricreati a ogni iterazione. L'algoritmo si ferma quando tutti gli attributi rimanenti sono classificati come confirmed oppure quando il numero massimo di iterazioni è stato raggiunto.



La figura soprastante mostra il risultato dell'applicazione di tale tecnica, etichettando come attributi significativamente più rilevanti: *Monocytes, Basophils, Lymphocytes, Hemoglobin, Hematocrit, Eosinophils, Platelets, Leukocytes*. Invece, i restanti attributi sono stati scartati e non considerati durante l'addestramento dei tre modelli.

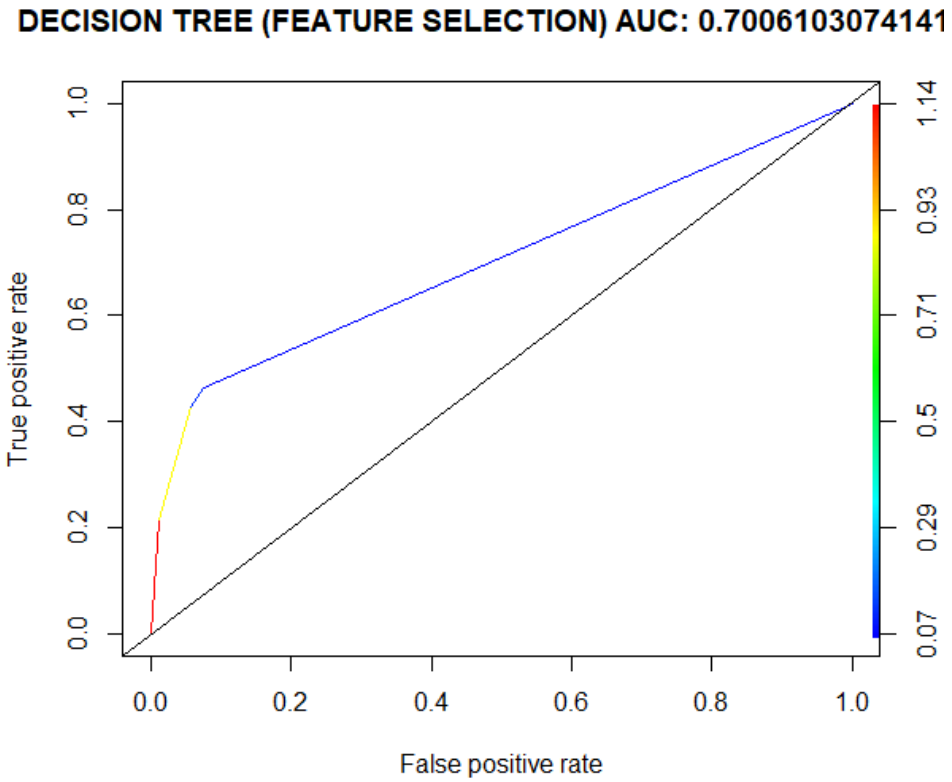
4.2.1 Feature Selection: Decision Tree



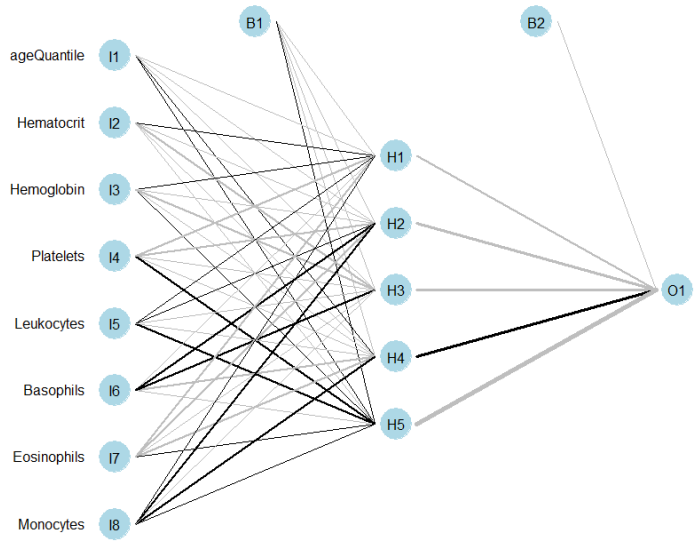

```
Confusion Matrix and Statistics

          Reference
Prediction negative positive
  negative      156       22
  positive        2        6

               Accuracy : 0.871
                 95% CI : (0.8141, 0.9155)
    No Information Rate : 0.8495
    P-Value [Acc > NIR] : 0.2401221

                  Kappa : 0.2855

 Mcnemar's Test P-Value : 0.0001052

            Sensitivity : 0.21429
            Specificity : 0.98734
         Pos Pred Value : 0.75000
         Neg Pred Value : 0.87640
             Prevalence : 0.15054
         Detection Rate : 0.03226
   Detection Prevalence : 0.04301
      Balanced Accuracy : 0.60081

       'Positive' Class : positive
```

Eseguendo la 10-fold cross validation, il Decision Tree ottimale ottenuto è quello visibile nella figura soprastante con un livello di accuratezza pari al 87.10%, precision 75%, recall 21.42%, f1 33,33%, sensitività 21,42 %, specificità 98.73%.

Nella seguente immagine viene mostrata la curva ROC con il relativo valore AUC pari a 70.06%.



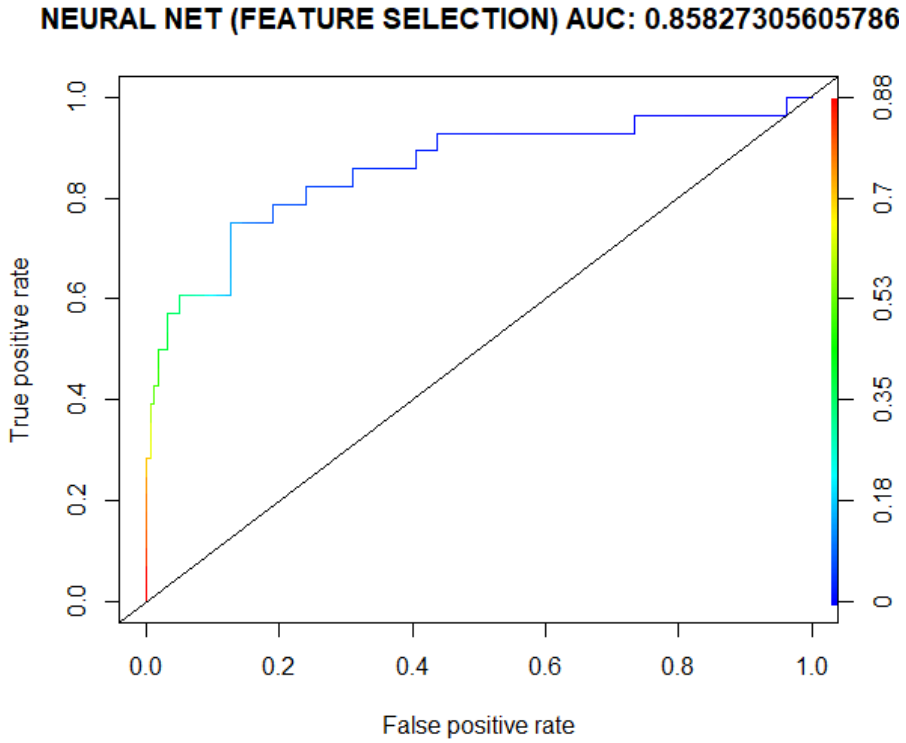
4.2.2 Feature Selection: Neural Network



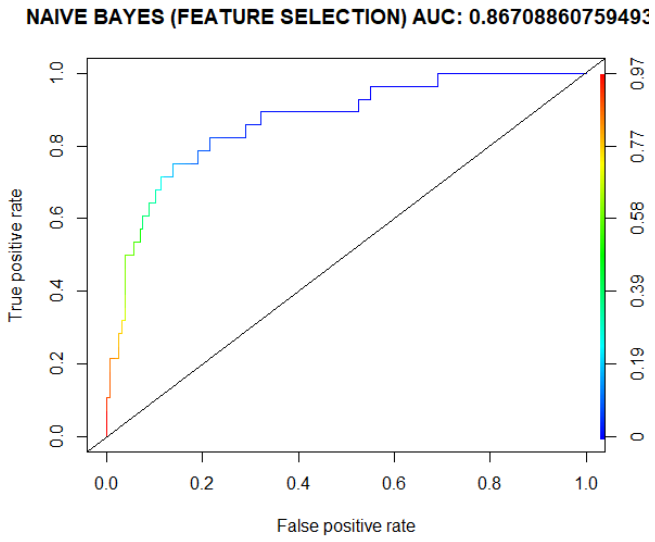

```
Confusion Matrix and Statistics

          Reference
Prediction negative positive
  negative      156       17
  positive        2       11

               Accuracy : 0.8978
                 95% CI : (0.8451, 0.9374)
    No Information Rate : 0.8495
    P-Value [Acc > NIR] : 0.035826

                  Kappa : 0.4877

 Mcnemar's Test P-Value : 0.001319

            Sensitivity : 0.39286
            Specificity : 0.98734
         Pos Pred Value : 0.84615
         Neg Pred Value : 0.90173
             Prevalence : 0.15054
         Detection Rate : 0.05914
   Detection Prevalence : 0.06989
      Balanced Accuracy : 0.69010

       'Positive' Class : positive
```

Eseguendo la 10-fold cross validation, la Neural Network ottimale ottenuta è quella visibile nella figura soprastante con un livello di accuratezza pari al 89.78%, precision 84.61%, recall 39.28%, f1 53,65%, sensitività 39.28 %, specificità 98.73%.

Nella seguente immagine viene mostrata la curva ROC con il relativo valore AUC pari a 85.82%.



4.2.3 Feature Selection: Naive Bayes



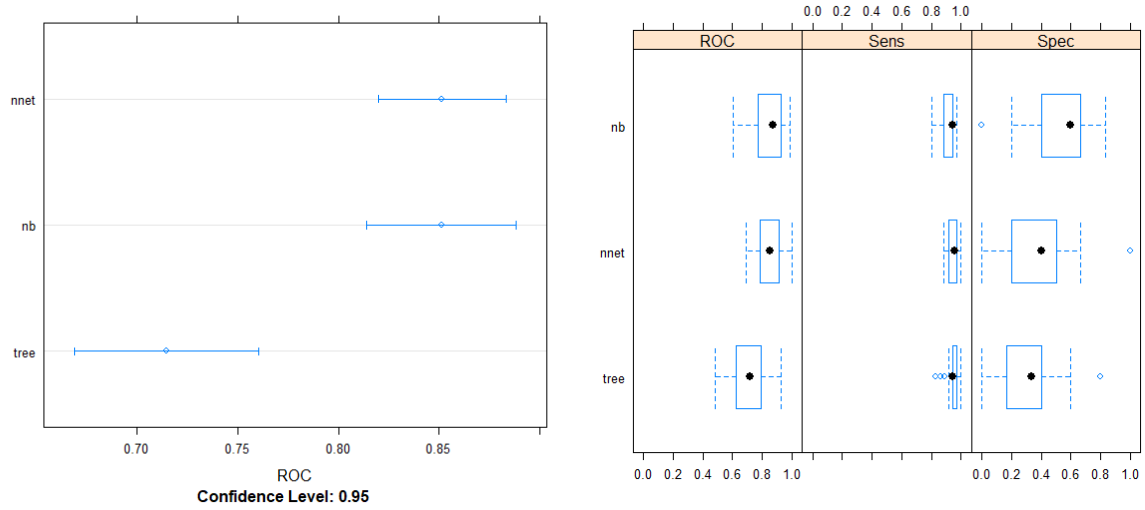

```
Confusion Matrix and Statistics

          Reference
Prediction negative positive
  negative      147       13
  positive       11       15

               Accuracy : 0.871
                 95% CI : (0.8141, 0.9155)
    No Information Rate : 0.8495
    P-Value [Acc > NIR] : 0.2401

                  Kappa : 0.4802

 Mcnemar's Test P-Value : 0.8383

            Sensitivity : 0.53571
            Specificity : 0.93038
         Pos Pred Value : 0.57692
         Neg Pred Value : 0.91875
             Prevalence : 0.15054
         Detection Rate : 0.08065
   Detection Prevalence : 0.13978
      Balanced Accuracy : 0.73305

       'Positive' Class : positive
```

Eseguendo la 10-fold cross validation, il modello Naive Bayes ottenuto è quello visibile nella figura soprastante con un livello di accuratezza pari al 87.10%, precision 57.69%, recall 53.57%, f1 55.55 %, sensitività 53.51 %, specificità 93.03%.

Nella immagine soprastante viene mostrata la curva ROC con il relativo valore AUC pari a 86.70%.

4.2.4 Confronto modelli Feature Selection

Osservando i grafici a sottostanti, è possibile trarre le seguenti conclusioni: Il modello Decision Tree è risultato essere in modo statisticamente significativo meno accurato rispetto agli altri due modelli, che invece risultano essere del tutto comparabili da questo punto di vista. Per quanto riguarda la sensitività i tre modelli hanno ottenuto valori sovrapponibili. Anche per il parametro di specificità sono stati ottenuti valori comparabili seppur più bassi rispetto alla sensitività. In termini di totalità dei parametri sia Neural Network che Naive Bayes risultano migliori che Decision Tree.



Anche in questo caso sono stati valutati i tempi necessari per addestrare i tre modelli e come si può notare dalla figura sottostante, il Decision Tree è quello computazionalmente più efficiente, la Neural Network è quella che è risultata più onerosa nei tempi di calcolo e Naive Bayes è risultato solo leggermente meno efficiente dell'albero di decisione.

	Everything	FinalModel	Prediction
tree	1.61	0.00	NA
nnet	13.17	0.04	NA
nb	4.93	0.01	NA

5 Conclusioni finali

Per trarre le conclusioni finali si è cercato di rispondere alla seguente domanda:

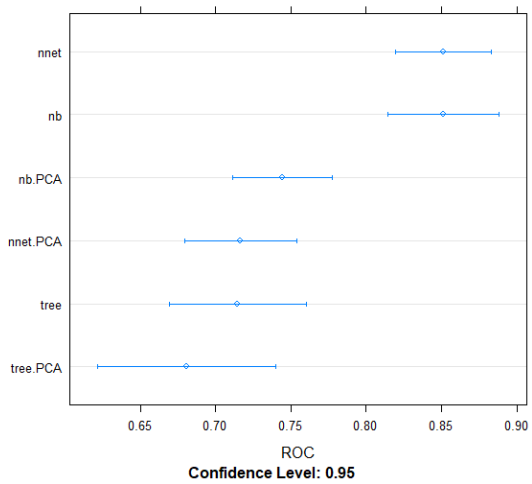
È possibile stimare la positività di un paziente al COVID-19, sulla base dei suoi parametri sanguinei?

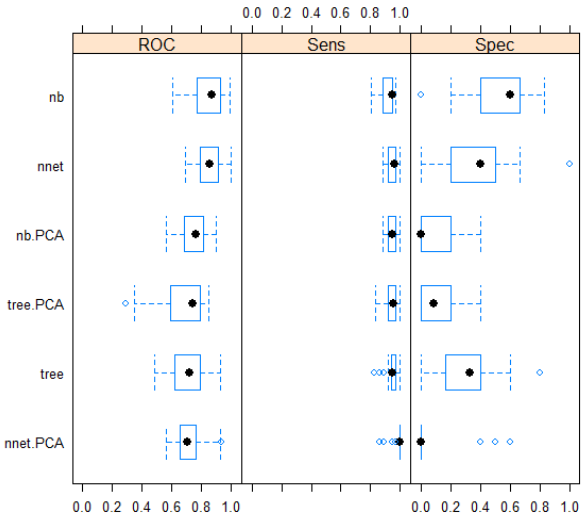
In base all'evidenza sperimentale ottenuta la risposta sembra essere no, ma con riserva, infatti, come illustrato nei vari paragrafi, il dataset risulta essere molto sbilanciato e non si hanno dati a sufficienza per eseguire un training dei modelli abbastanza consistente da poter essere usato in un ambito medico. Ciò non toglie che, avendo a disposizione molti più dati, sia possibile affinare i modelli per riuscire ad ottenere un modello predittivo efficace e usabile in un contesto reale e così delicato come quello sanitario.

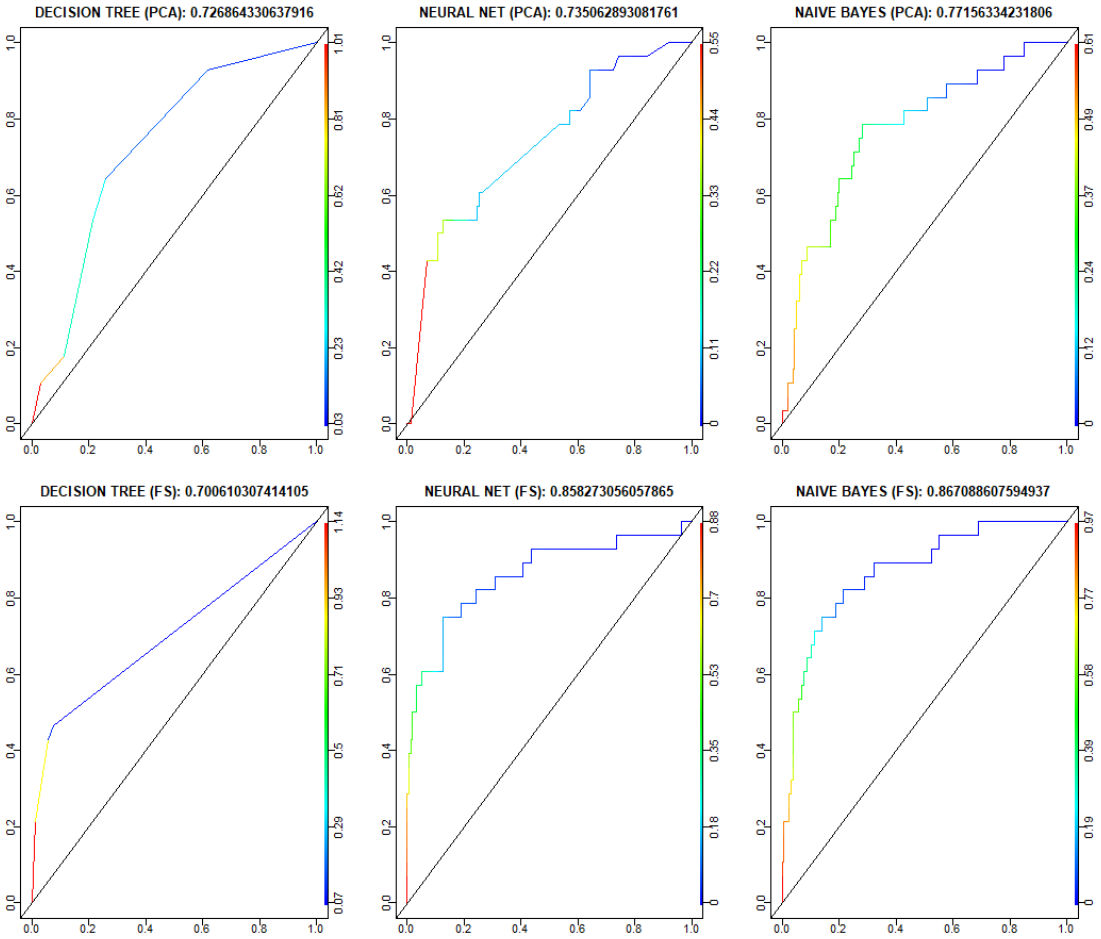
La matrice di confusione complessiva ottenuta con il dataset a disposizione è la seguente:

	Accuracy	Kappa	AccuracyLower	AccuracyUpper	AccuracyPValue
TREE (PCA)	0.8395722	0.10725652	0.7789683	0.8890663	0.70189384
NEURAL NETWORK (PCA)	0.8395722	-0.02037104	0.7789683	0.8890663	0.70189384
NAIVE BAYES (PCA)	0.8449198	0.21847528	0.7849318	0.8936138	0.62890045
TREE (FS)	0.8709677	0.28553137	0.8141178	0.9155466	0.24012212
NEURAL NETWORK (FS)	0.8978495	0.48767759	0.8450802	0.9373700	0.03582648
NAIVE BAYES (FS)	0.8709677	0.48020494	0.8141178	0.9155466	0.24012212

Le misure di performance ottenute sono le seguenti:







È possibile notare come il training dei modelli sul dataset a cui è stata applicata la feature selection, avendo così poche osservazioni e un dominio con un ridotto numero di feature, fornisca risultati più consistenti, rispetto al training degli stessi modelli sul dataset a cui è stata applicata la PCA.

I modelli risultanti più accurati dal confronto sono Neural Network e Naive Bayes. Tuttavia, dal punto di vista computazionale il modello Naive Bayes risulta, tra i due, il migliore, come visto nei paragrafi precedenti.

In futuro, avendo a disposizione dataset più ricchi e meno sbilanciati, si potranno eseguire training più consistenti e ottenere risultati più soddisfacenti.