Progetto Machine Learning

Romanato Luca 807691
Romanato Matteo 816852

Capitolo 1

Introduzione

Negli ultimi anni nel campo dell'astronomia, in particolare nell'astrofisica, si sono susseguiti risultati eclatanti, come la prima fotografia di un buco nero e la scoperta delle onde gravitazionali, e risultati con meno impatto mediatico ma che stanno aprendo nuove frontiere e rivoluzionando le piccole branchie di questa scienza come lo studio dei raggi Gamma. Il progetto si focalizza proprio sull'analizzare i dati derivati dalle osservazioni di quest'ultimi, elaborate da MAGIC (Major Atmospheric Gamma Imaging Cherenkov telescope) e resi disponibili online. Al fine di apprendere meglio il contesto scientifico occorre scendere più nel dettaglio di cosa sia MAGIC ed i raggi gamma.

1.1 Raggi Gamma

I raggi gamma sono fotoni, come quelli che compongono la luce visibile, ma molto più carichi di energia. Sono prodotti da eventi straordinariamente energetici che avvengono in situazioni particolari nel nostro universo come collassi gravitazionali e onde d'urto che si generano in prossimità di buchi neri durante il loro accrescimento, resti di supernova o Gamma Ray Bursts (Grb), cioè violente emissioni di raggi gamma di altissima energia che durano pochi attimi e di cui non si è ancora compresa a fondo la natura.

1.2 MAGIC

Ci sono due possibilità per poter osservare i raggi gamma: si possono collocare i rivelatori su satelliti in orbita nello spazio così da rivelarli prima che interagiscano con l'atmosfera terrestre; Oppure, si costruiscono rivelatori a terra che osservano, come MAGIC, la cosiddetta "luce Cherenkov", cioè il bagliore emesso dalle particelle prodotte nell'interazione dei fotoni coi nuclei dell'atmosfera terrestre. Il problema, nonché obbiettivo del progetto, sta nel capire se tale bagliore captato è dato dall'interazione di fotoni provenienti dai raggi gamma o da altri raggi cosmici.

Capitolo 2

Dataset

Il dataset, presente sul Kaggle, è stato generato attraverso un algoritmo con metodologia Monte Carlo descritto in "D. Heck et al., CORSIKA, A Monte Carlo code to simulate extensive air showers, Forschungszentrum Karlsruhe FZKA 6019 (1998)" dove i parametri in input permettevano di osservare eventi con energia al di sotto di 50 GeV. Inoltre, per ragioni tecniche tra le quali delle classi non troppo sbilanciate, il numero degli eventi di background è stato sottostimato. Nella realtà rappresenta la classe di maggioranza.

2.1 Analisi del dataset

Il dataset è composto da 12 attributi totali, come rappresentato nella tabella sottostante, per un totale di 19020 record.

Attributo	Dato	Descrizione
index	integer	indice
fLength	continuous	asse maggiore dell'ellisse [mm]
fWidth	continuous	asse minore dell'ellisse[mm]
fSize	continuous	somma in base 10-log del contenuto di tutti i pixels dell'immagine [in #phot]
fConc	continuous	ratio della somma dei due pixel con valore più alto presenti in fSize [ratio]
fConc1	continuous	ratio del pixel con valore più alto presente in fSize [ratio]
fAsym	continuous	distanza del pixel con valore più alto dal centro, proiettato sull'asse maggiore[mm]
fM3Long	continuous	radice cubica del momento (probabilità) sull'asse maggiore [mm]
fM3Trans	continuous	radice cubica del momento (probabilità) sull'asse minore [mm]
fAlpha	continuous	angolo tra asse maggiore e vettore rispetto all'origine [deg]
fDist	continuous	distanza tra origine e centro dell'ellisse[mm]
class	g, h	gamma (signal), hadron (background)

All'interno del dataset non sono presenti valori mancanti, il che non rende necessaria alcuna tipologia di preprocessing al fine di eliminare o rimpiazzare i valori inesistenti. Inoltre, come si evidenzia anche dal seguente grafico abbiamo un'unico attributo discreto ed undici attributi numerici.

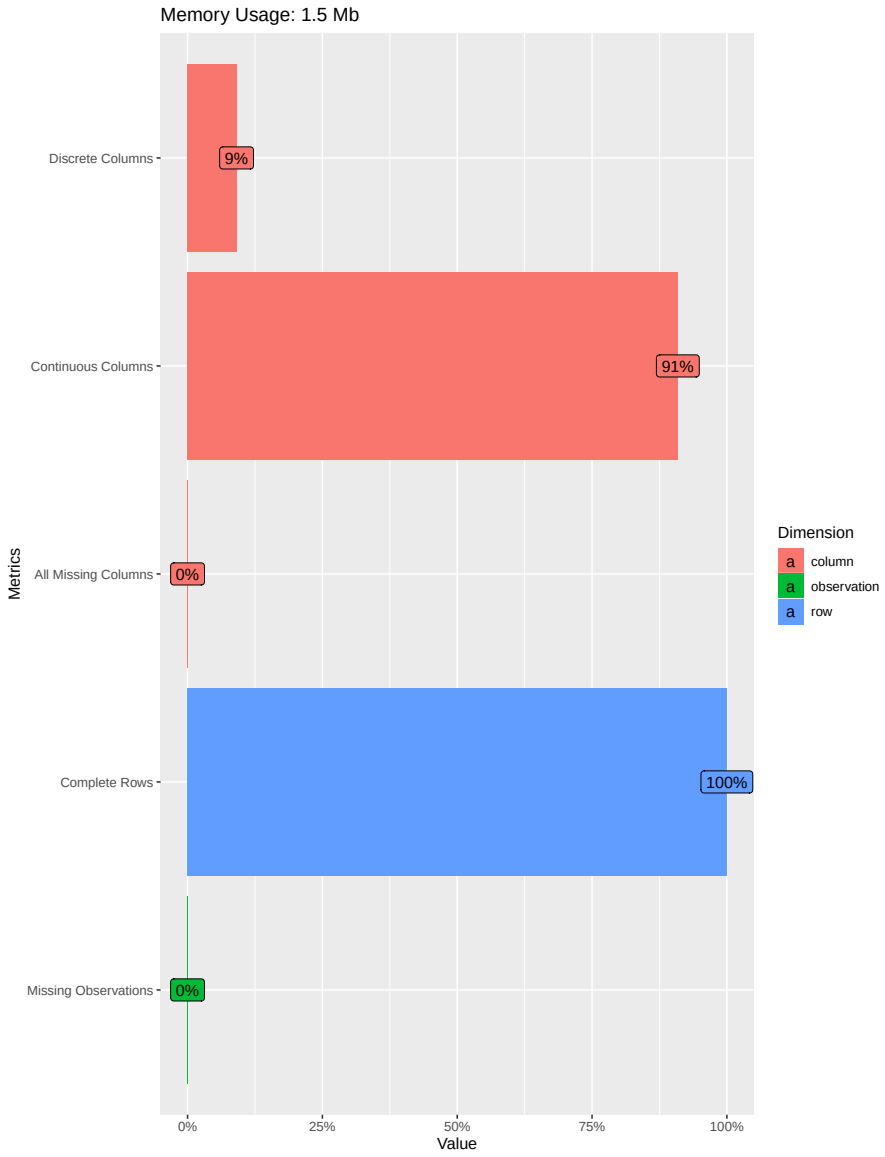

Figura 2.1: analisi valori e colonne

Osservando la tabella iniziale possiamo dedurre che la nostra classificazione sarà di tipo binario avendo per la classe target "class" unicamente 2 valori, g se sono raggi gamma e h se invece sono raggi cosmici. Nel grafico possiamo vedere come le due classi non sono perfettamente bilanciate ma nemmeno tanto sbilanciate da dover richiedere un'intervento. Dunque, si presume una classificazione senza overfitting da parte della classe di maggioranza.

Figura 2.2: distribuzione valori della classe target

I restanti attributi hanno invece le seguenti distribuzioni.

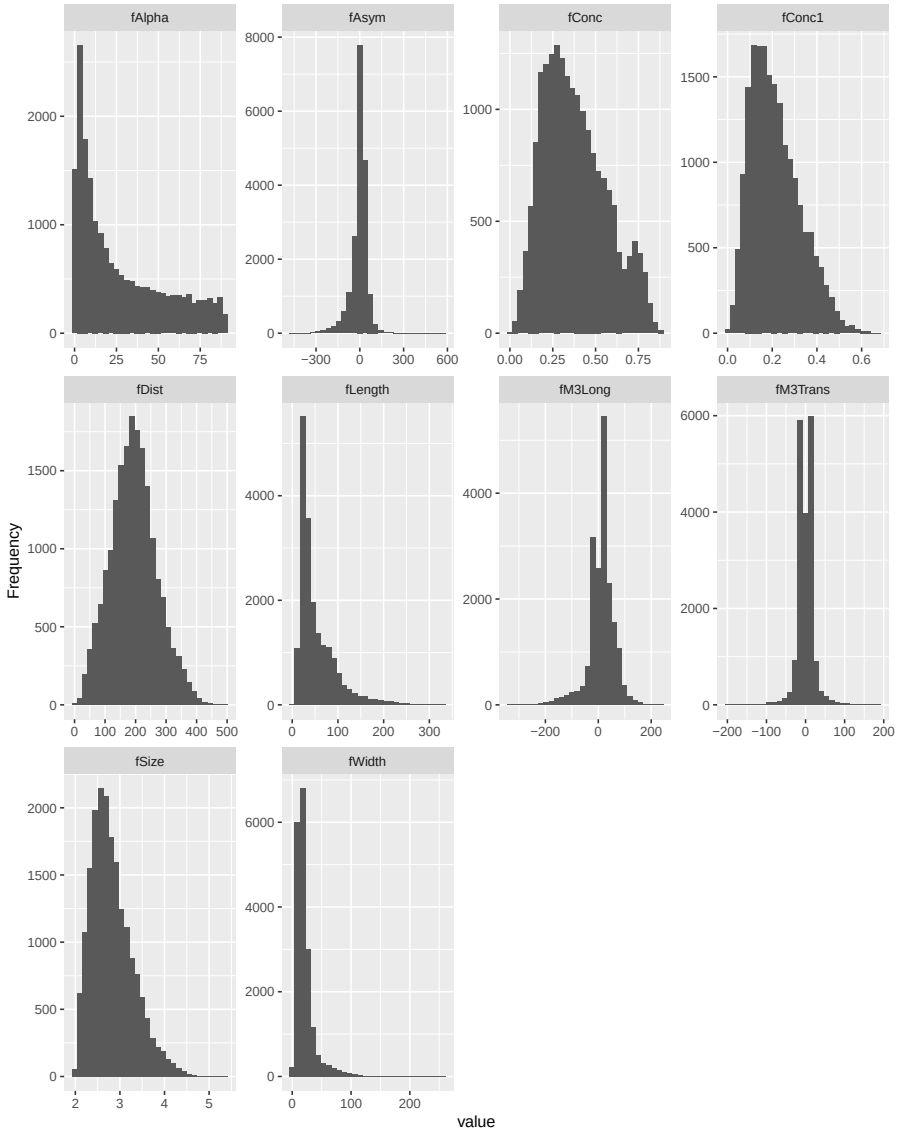

Figura 2.3: distribuzione valori dei restanti attributi

Capitolo 3

Training Set

La creazione del training set si è sviluppata su due fasi, la prima con la visualizzazione della matrice di confusione e poi degli autovalori proveniente dalle PCA (Principal Component Analysis) per vedere quali attributi considerare.

3.1 Matrice di confusione

Come prima operazione si è andati a visualizzare la matrice di correlazione per vedere quanto gli attributi fossero correlati tra di loro e dunque, se erano presenti variabili molto simili tra di loro. Osservando la matrice di correlazione sotto riportata possiamo notare come le variabili tra di loro siano veramente poco correlate eccetto qualche caso come fWidth,fSize con fLenght e fConc con fConc1. Possiamo dunque dire che la dimensione del training set sarà molto vicina a quella del dataset iniziale.

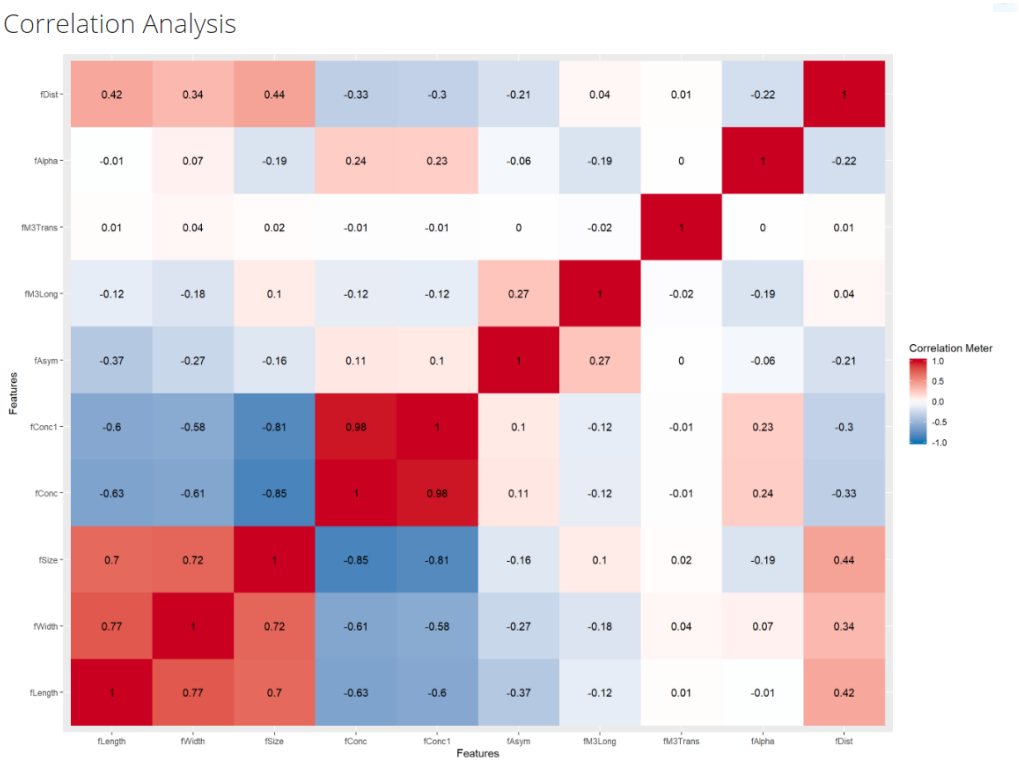

Figura 3.1: Matrice di correlazione

3.2 Autovalori

Sono stati poi plottati gli autovalori in ordine crescente come varianza spiegata, come rappresenta il grafico qui sotto riportato. Possiamo dunque dire che le prime cinque componenti principali spiegano già il 78% della varianza del nostro problema.

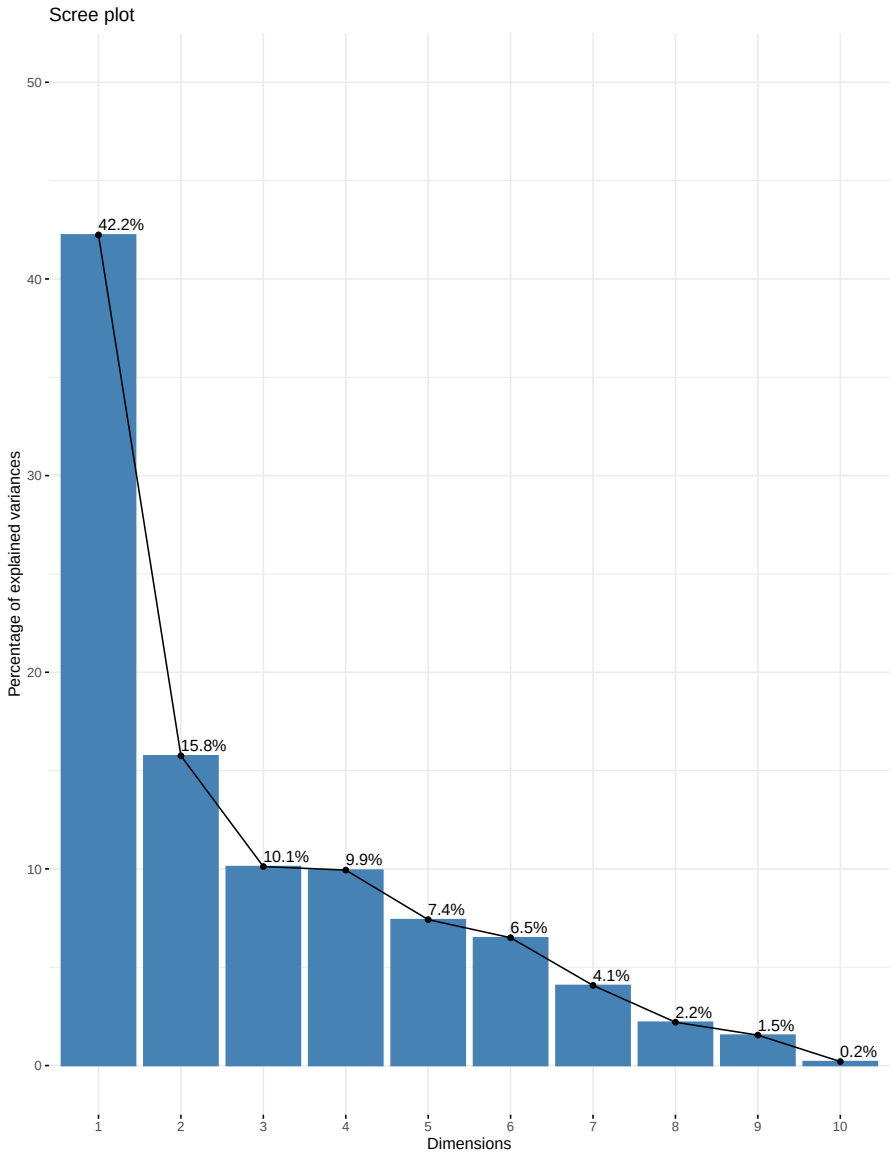

Figura 3.2: autovalori

3.3 PCA

Infine, applicate le PCA tramite autovalori a tutte le colonne del dataset eccetto la classe target e, ovviamente, l'indice, si è confrontato la matrice di correlazione con il contributo delle variabili e la varianza spiegata tramite il grafico degli autovalori. Si è constatato che fConc e fConc1 sono quasi una la copia dell'altro, mentre fM3Trans ed fM3Long molto spesso danno pochissimo contributo rispetto agli altri attributi nelle dimensioni che spiegano più varianza.

	Dim.1	Dim.2	Dim.3	Dim.4	Dim.5
fLength	16.668992522	5.09399215	2.506402e-05	0.06419378	1.42741412
fWidth	15.607007161	6.01630550	2.332268e+00	0.89531796	0.33521178
fSize	20.153089816	0.87068522	4.984569e-01	0.33057046	0.02535317
fConc	19.502156343	3.54854402	1.662027e+00	1.17978254	3.26415123
fConc1	18.424507917	3.83971153	1.937602e+00	1.35288343	4.22571709
fAsym	2.093461257	23.70744161	1.423353e+01	4.57602860	0.05169843
fM3Long	0.002702298	35.95088349	3.401957e-01	0.28637642	35.83987834
fM3Trans	0.015256181	0.09908158	3.634521e+01	63.15251280	0.00658962
fAlpha	0.944297480	20.85780921	2.208237e+01	14.92047468	27.64586156
fDist	6.588529025	0.01554569	2.056832e+01	13.24185933	27.17812466

Figura 3.3: contributo delle variabili per le prime 5 componenti

Si è deciso dunque di creare il training set mantenendo tutte le variabili eccetto fConc1, fM3Trans ed fM3Long. Il nostro training set sarà dunque composto da 8 variabili totali.

Variabile	Esito	Motivazione
index	scartato	inutilità
fLength	mantenuto	buon contributo nelle PCA delle prime cinque dimensioni
fWidth	mantenuto	buon contributo nelle PCA delle prime cinque dimensioni
fSize	mantenuto	buon contributo nelle PCA delle prime cinque dimensioni
fConc	mantenuto	buon contributo nelle PCA delle prime cinque dimensioni
fConc1	scartato	molto simile a fConc
fAsym	mantenuto	buon contributo nelle PCA delle prime cinque dimensioni
fM3Long	scartato	contributo non sufficiente
fM3Trans	scartato	contributo non sufficiente
fAlpha	mantenuto	buon contributo nelle PCA delle prime cinque dimensioni
fDist	mantenuto	buon contributo nelle PCA delle prime cinque dimensioni
class	mantenuto	classe target

Capitolo 4

Modelli utilizzati

I modelli utilizzati sono stati due, entrambi per apprendimento supervisionato in quanto per tutti i dati di test è presente la classe di appartenenza. Sono stati scelti Random Forest e una rete neurale in quanto sono largamente utilizzati nel campo dell'astrofisica e, a mio parere, sono stati i più interessanti visti all'interno del corso.

4.1 Rete Neurale

La rete neurale è stata implementata tramite il package Keras. Tale package è stato preferito a neuralnet ed h2o, per la semplicità con cui si costruisce e si addestra la rete e per la velocità di esecuzione. La rete per questo progetto consiste in un'architettura di 4 layer, con funzione di attivazione relu, così costruita:

layer	neuroni	# parametri di input
input	8	
nascosto	64	512
nascosto	20	1300
output	2	42

In input si ha un neurone per ogni variabile del training set mentre per l'output si hanno due neuroni, uno se la classe predetta è gamma e uno se hadron. Conseguenza diretta di questa architettura è la funzione di loss, categorical crossentropy, e la funzione di attivazione del layer di output, softmax. La rete è stata allenata con l'ottimizzatore Adam, che permette una convergenza verso il minimo globale migliore dell'algoritmo del gradiente, ed un numero di epoche pari a 50, che permette di vedere se la rete è in overfitting.

4.1.1 Esperimenti

Gli esperimenti si sono basati sulla stima delle misure di performance quali: matrice di confusione, precision, recall, f-measure, ROC e AUC, ottenute attraverso una 10 fold cross validation. Al fine di eseguire questi esperimenti la rete è stata prima trainata e testata su un validation set

corrispondente al 10% del training set, precedentemente ricavato dalla suddivisione 80% - 20% del dataset in train - test. Per ogni iterazione della cross validation non si è mai riscontrato overfitting dato che la curva della loss del validation set e del training set convergevano allo stesso punto, come rappresentato dal grafico sottostante.

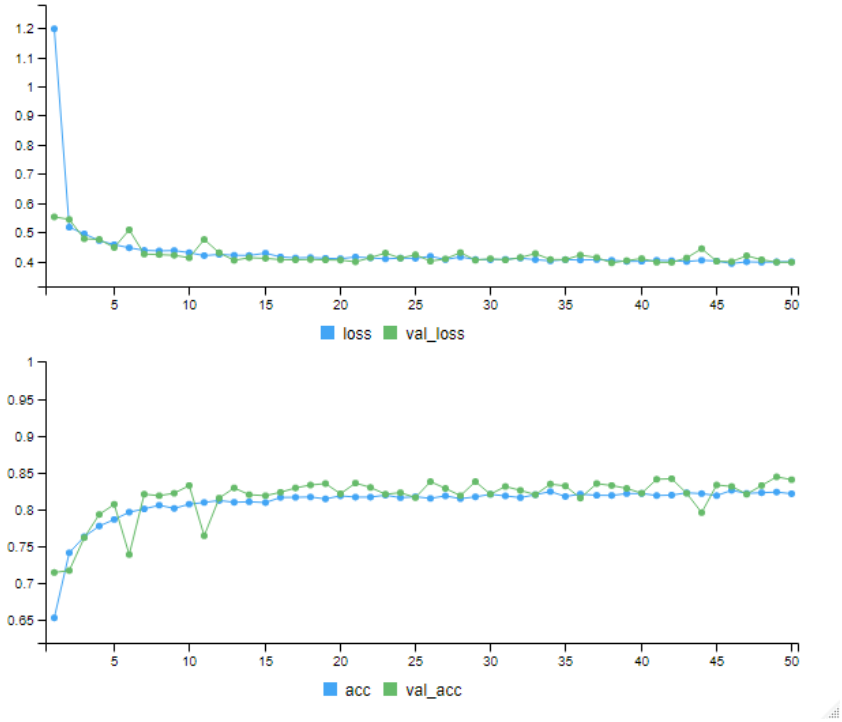

Figura 4.1: esecuzione di un fold attraverso rete neurale

Sono state poi calcolate le misure di performance richieste e mediate alla fine della cross validation, ottenendo i seguenti risultati.

Accuratezza	Precision	Recall	f-measure	AUC
0.814	0.818	0.918	0.863	0.880

con una matrice di confusione pari a

	g	h
g	2148	502
h	199	849

la curva ROC risultante è stata:

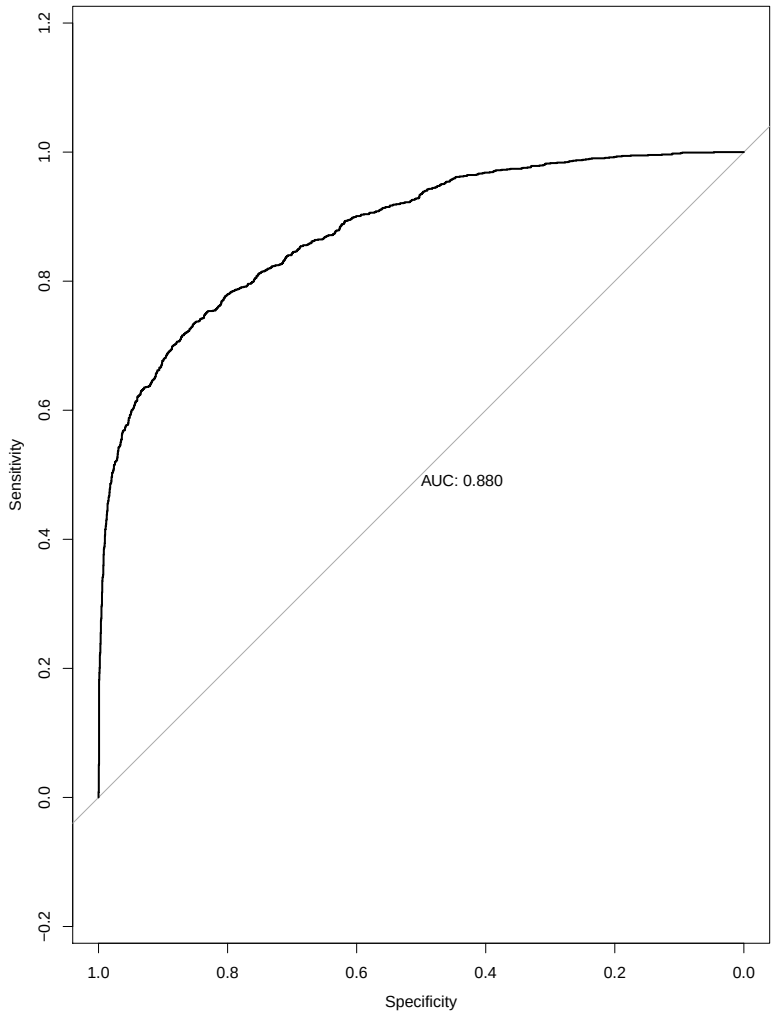

Figura 4.2: Curva ROC rete neurale

4.2 Random Forest

Random Forest è stata implementata tramite il package caret, che mette a disposizione una grande varietà di parametri per trovare la combinazione migliore oltre ad una buona documentazione.

4.2.1 Esperimenti

Per quanto riguarda Random Forest è stato scelto di creare 64 alberi di decisione. Si è deciso un numero pari a 64 in quanto facendo prove usando 10, 32 e 128 alberi le prestazioni peggioravano con 10 e 32 mentre con 128 non miglioravano di molto e quindi era inutile aumentare ulteriormente il numero di alberi. Il numero di variabili campionate casualmente come target ad ogni separazione all'interno di un albero è stata sempre mantenuta a 2. Anche per random forest è avvenuta una suddivisione in 80% train e 20% test del dataset e non è presente overfitting. Dunque, dalla 10

fold cross validation si sono ottenuti i seguenti risultati

Accuratezza	Precision	Recall	f-measure	AUC
0.875	0.878	0.936	0.906	0.929

con una matrice di confusione pari a

	g	h
g	2301	330
h	140	927

la curva ROC risultante è stata:

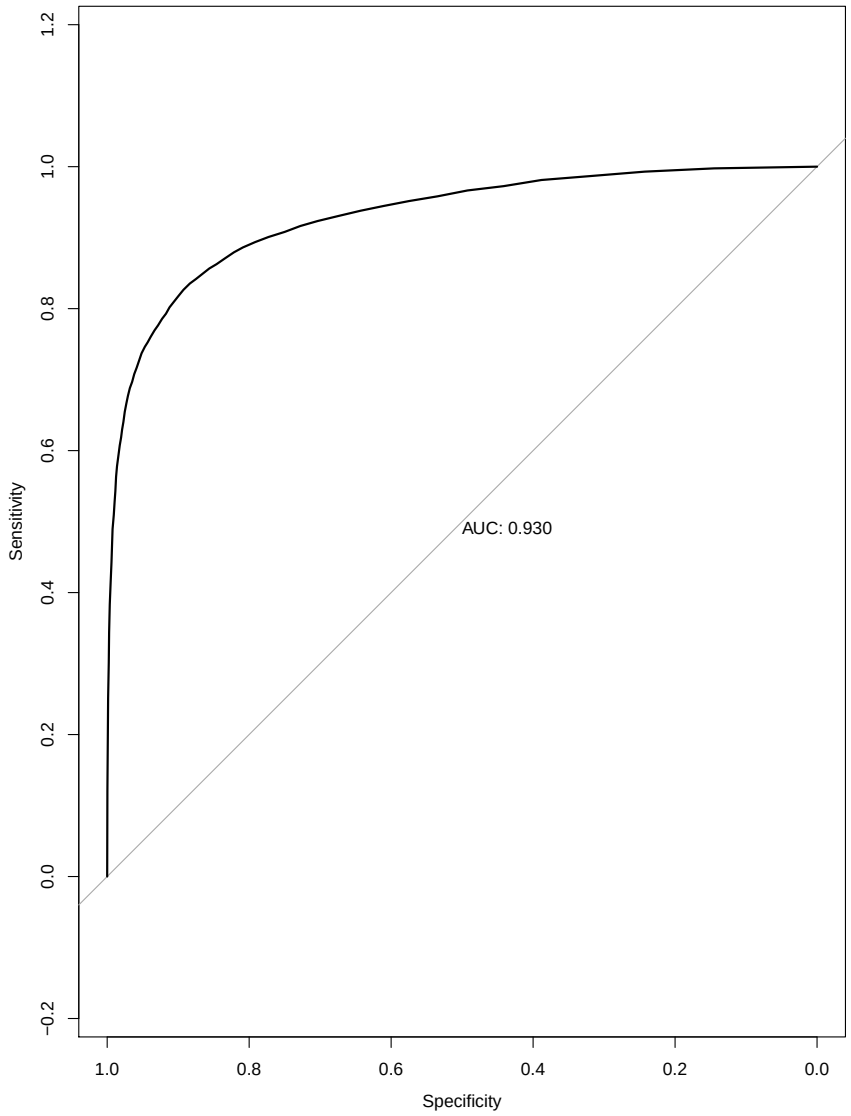

Figura 4.3: Curva ROC Random Forest

Capitolo 5

Analisi dei Risultati

Confrontando le matrici di confusione dei due modelli appare subito chiaro il migliore, ovvero Random Forest. La rete neurale non riesce a predire la classe di minoranza tanto bene quanto random forest, 502 vs 330, tutta via entrambi i modelli hanno un'accuratezza soddisfacente; 87,5% per Random Forest e 81,4% per la rete neurale. Anche la curva ROC evidenzia la superiorità degli alberi decisionali, infatti la nostra AUC (Area Under Curve) è migliore di 0.05, passando infatti da 0.88 per la rete a un 0.93 della Random Forest. Per quanto rigurda invece f-measure, precisione e recall, anche su queste misure Random Forest è migliore della rete neurale.

Capitolo 6

Conclusioni

In conclusione, su questo dataset con questa tipologia di dati, il modello migliore si è dimostrato essere Random Forest superando in qualsiasi misura di performance la rete neurale. Molto evidente è come gli alberi decisionali riescano a classificare meglio la classe di minoranza e di conseguenza avere performance migliori in tutte le misure analizzate. Random Forest inoltre vince anche su fattore tempo, in quanto la cross validation impiega pochi secondi mentre sulla rete neurale dura minuti.
Sarebbe interessante riuscire a provare questo modello su dei dati reali per vedere effettivamente se, con dati fortemente sbilanciati, la rete neurale riesca ad adattarsi meglio rispetto agli alberi decisionali.