Università degli studi di Milano Bicocca

Progetto del corso di Machine Learning

Modelli di classificazione della qualità di compressioni JPEG di tipo lossy

11 settembre 2021

Authors
Gabriele Di Lieto 874143 g.dilieto@campus.unimib.it
Marco Poveromo 830626 m.poveromo@campus.unimib.it

Indice

Introduzione

Nel seguente elaborato analizzeremo alcuni modelli di classificazione per predire la qualità di compressioni di immagini ottenute tramite un algoritmo JPEG di tipo lossy.
Nel **capitolo 1** analizzeremo il dominio di riferimento, e gli obiettivi dell'elaborato.

Nel **capitolo 2** andremo a descrivere come è stato costruito il dataset di riferimento per i modelli.

Nel **capitolo 3** effettueremo un'analisi esplorativa ed eseguiremo una principal component analysis.

Nei **capitoli 4,5 e 6** descriveremo, motiveremo e analizzeremo i modelli utilizzati per la classificazione.

Infine nel **capitolo 7** andremo a commentare i risultati ottenuti dai modelli sulla cui base trarremo delle conclusioni.

Capitolo 1

Descrizione del dominio di riferimento e obiettivi dell'elaborato

1.1 Compressione JPEG lossy

La **compressione JPEG** di tipo **lossy** che andremo a considerare è un tipo di compressione di immagini che si basa sulla trasformazione tramite **DCT** (Discrete cosine transform) dell'immagine originale, successiva compressione della trasformazione ottenuta (attraverso il taglio delle frequenze meno significative) e ricostruzione dell'immagine originale dalla trasformazione compressa tramite **IDCT** (Inverse discrete cosine transform).

L'immagine originale è una matrice

$$\mathbf{I} = \begin{array}{c} \\ \mathrm{r}_1 \\ \mathrm{r}_2 \\ \vdots \\ \mathrm{r}_N \end{array} \begin{array}{c} \begin{array}{cccc} \mathrm{c}_1 & \mathrm{c}_2 & \cdots & \mathrm{c}_M \end{array} \\ \begin{pmatrix} i_{0,0} & i_{0,1} & \cdots & i_{0,M-1} \\ i_{1,0} & i_{1,1} & \cdots & i_{1,M-1} \\ \vdots & \vdots & \ddots & \vdots \\ i_{N-1,0} & i_{N-1,1} & \cdots & i_{N-1,M-1} \end{pmatrix} \end{array}$$

contenente NxM pixel, che sono rappresentabili come degli interi (nel nostro caso interi a 8 bit poiché consideriamo immagini in scala di grigio).
Nell'algoritmo JPEG lossy (di cui consideriamo la versione senza matrice di quantizzazione, che sbilancia euristicamente le frequenze ottenute nella trasformazione), suddividiamo la matrice I in **chunk**, cioè sottomatrici di dimensione 8x8. Successivamente su ogni chunk viene effettuata una trasformazione tramite DCT (ulteriori

dettagli sulla libreria Python utilizzata per effettuare la DCT a questo link).
Una volta trasformati i chunk, l'algoritmo JPEG lossy elimina alcune frequenze (partendo da quelle in basso a destra del chunk trasformato), azzerandone il valore e memorizzando solo le frequenze rimanenti.
Uno degli approcci che si può avere alla compressione è quello di eliminare le frequenze a partire dalla diagnonale d del chunk trasformato (Figure 1.1 e 1.2).

5	7	1	122	12	4	0	200
150	20	133	2	255	2	1	33
11	0	7	4	17	111	23	0
41	10	117	44	117	4	1	22
231	77	150	11	55	0	8	9
20	127	15	111	60	3	58	119
5	177	22	154	78	240	1	3
3	66	18	67	1	0	31	9

Figura 1.1: Chunk trasformato

5	7	1	122	0	0	0	0
150	20	133	0	0	0	0	0
11	0	0	0	0	0	0	0
41	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Figura 1.2: Chunk trasformato con frequenze tagliate dalla diagonale 5 (in rosso)

1.2 Obiettivi

L'obiettivo di questo elaborato è quello di descrivere dei modelli che possano classificare se un determinato taglio delle frequenze di un chunk possa portare ad una buona compressione o meno di quest'ultimo.
Il chunk analizzato dal modello viene letto prima della sua trasformazione, in modo che i modelli possano essere utilizzati prima del processo di compressione, e quindi per ottimizzare il taglio delle frequenze sul potenziale chunk trasformato.
L'idea di fondo è di utilizzare questo modello per effettuare una ricerca binaria e trovare il taglio che massimizza la compressione possibile mantenendo un'alta qualità, cioè quello che appena diminuito genera un taglio di bassa qualità.

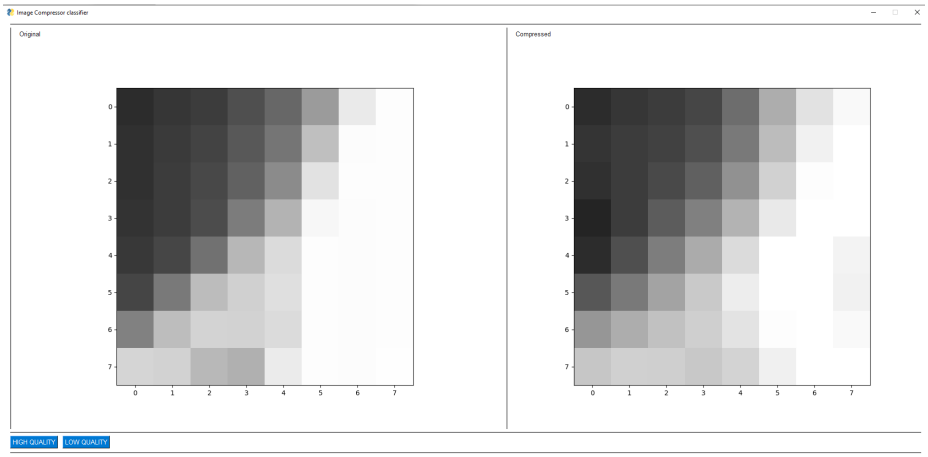
Dal chunk originale vengono estratti vari indici che rappresenteranno l'istanza su cui i modelli lavoreranno.
Nel successivo capitolo analizzeremo gli indici che abbiamo voluto tenere in considerazione per ogni chunk e come abbiamo generato di conseguenza il dataset.

Capitolo 2

Scelte di design per la creazione del data set, eventuali ipotesi o assunzioni

2.1 Creazione del dataset

Per la costruzione del dataset è stato sviluppato un piccolo software in Python. Il software estrae un gruppo di chunk da una cartella di immagini bitmap in grayscale. Successivamente, per ogni chunk, sceglie un taglio random delle frequenze ed effettua la compressione.



Nell'interfaccia vengono mostrati i chunk compressi affiancati al chunk originale. L'utente può classificare le coppie come low_quality e high_quality, e mano mano la classificazione viene aggiunta al dataset nel file *image_compression.csv*.

Il file presenta tre colonne:

1 *Bmp image*, che contiene la matrice dei pixel del chunk originale;

2 *Threshold*, che contiene il taglio generato con cui è stato compresso il chunk;

3 *Quality*, che contiene la classificazione scelta dall'utente per la specifica compressione.

La dimensione del dataset dopo essere stato caricato in R è **1177 x 3**, e presenta le covariate mostrate in figura.

	Bmp.image	Threshold	Quality
1	46 44 45 45 45 47 47 42 45 47 46 46 46 44 47 49 4...	1	low quality
2	55 64 77 80 108 143 143 147 60 57 58 65 73 88 97 ...	5	high quality
3	255 255 255 255 255 255 255 255 255 255 255 255...	4	high quality
4	180 171 161 173 183 191 194 193 159 151 155 174...	9	high quality
5	43 64 60 67 58 43 48 57 52 62 64 63 70 51 59 64 5...	12	high quality

Successivamente si eliminano i duplicati per le righe, e viene prodotto un dataset di dimensione inferiore: **1154 x 3**.

Il nuovo dataset viene poi preprocessato per ottenere nuove covariate sulla base dei chunk dell'immagine: media, mediana, varianza, contrasto (**Rms constrast** e **M contrast**, ed entropia.

	Bmp.image	Threshold	Quality	Mean	Median	Variance	Rms.constrast	M.contrast	Entropy
1	46 44 45 45 45 47 47 42 45 47 46 46 46 44 47 49 4...	1	low quality	45.59375	45.0	11.260913	3.355728	0.16483516	1.998718
2	55 64 77 80 108 143 143 147 60 57 58 65 73 88 97 ...	5	high quality	76.68750	75.0	379.392857	19.478010	0.47738693	1.986304
3	255 255 255 255 255 255 255 255 255 255 255 255...	4	high quality	254.57812	255.0	3.231895	1.797747	0.02204409	1.999988
4	180 171 161 173 183 191 194 193 159 151 155 174...	9	high quality	171.34375	172.5	282.451389	16.806290	0.15451895	1.997715
5	43 64 60 67 58 43 48 57 52 62 64 63 70 51 59 64 5...	12	high quality	57.60938	56.0	127.448165	11.289294	0.39344262	1.991097
6	75 71 71 80 77 83 86 80 75 71 77 79 78 81 84 82 7...	1	low quality	73.96875	72.5	70.411706	8.391168	0.24528302	1.997012
7	14 15 14 13 14 21 10 20 7 10 19 14 5 5 5 19 10 18 ...	12	high quality	12.57812	13.5	36.311260	6.025883	1.00000000	1.931777
8	102 111 132 149 153 151 141 180 108 109 116 144...	4	high quality	116.56250	108.5	712.662698	26.695743	0.36567164	1.988225
9	109 104 94 81 72 75 86 100 94 90 83 71 69 73 76 8...	13	high quality	74.84375	73.0	118.991071	10.908303	0.29761905	1.995161
10	76 77 74 70 64 54 51 57 87 86 83 77 66 59 51 52 9...	3	high quality	76.75000	79.0	98.952381	9.947481	0.30136986	1.995797
11	33 33 41 46 36 39 39 32 35 34 40 40 36 37 34 45 3...	4	high quality	37.56250	37.0	12.789683	3.576267	0.21052632	1.997859
12	38 40 44 50 48 45 45 46 39 38 45 49 48 47 48 48 4...	3	high quality	47.32812	48.0	26.890625	5.185617	0.27659574	1.997153
13	21 33 43 37 42 78 62 47 20 23 32 33 30 42 56 43 3...	5	high quality	31.43750	30.0	91.805556	9.581522	0.59183673	1.981589
14	128 126 118 111 112 108 111 122 127 121 119 113...	5	high quality	109.07812	110.0	118.073165	10.866148	0.24271845	1.997587
15	190 167 137 130 126 127 126 126 195 183 165 150...	2	low quality	158.67188	159.5	579.430308	24.071359	0.23028391	1.994510

Abbiamo poi osservato che l'indice *Variance* e uno degli indici di contrasto (*Rms constrast*) sono uno funzione dell'altro, e di conseguenza abbiamo scelto di rimuovere

il primo.
Il dataset finale presenta quindi una dimensionalità di **1154 x 7**, dove le covariate ottenute nel preprocessing sono di tipo "numeric", la threshold è di tipo "integer", mentre la Quality è di tipo "factor".

```
> sapply(image_compression,class)
   Threshold         Mean       Median Rms.contrast   M.contrast      Entropy      Quality
   "integer"    "numeric"    "numeric"    "numeric"    "numeric"    "numeric"     "factor"
```

Capitolo 3

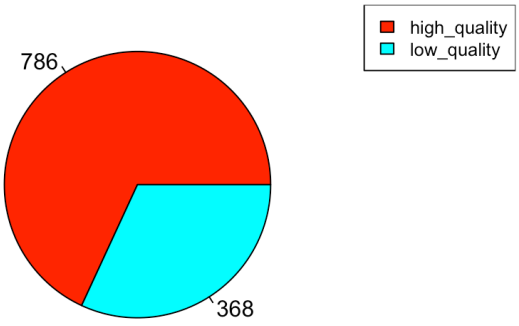
Analisi esplorativa

3.1 Analisi esplorativa

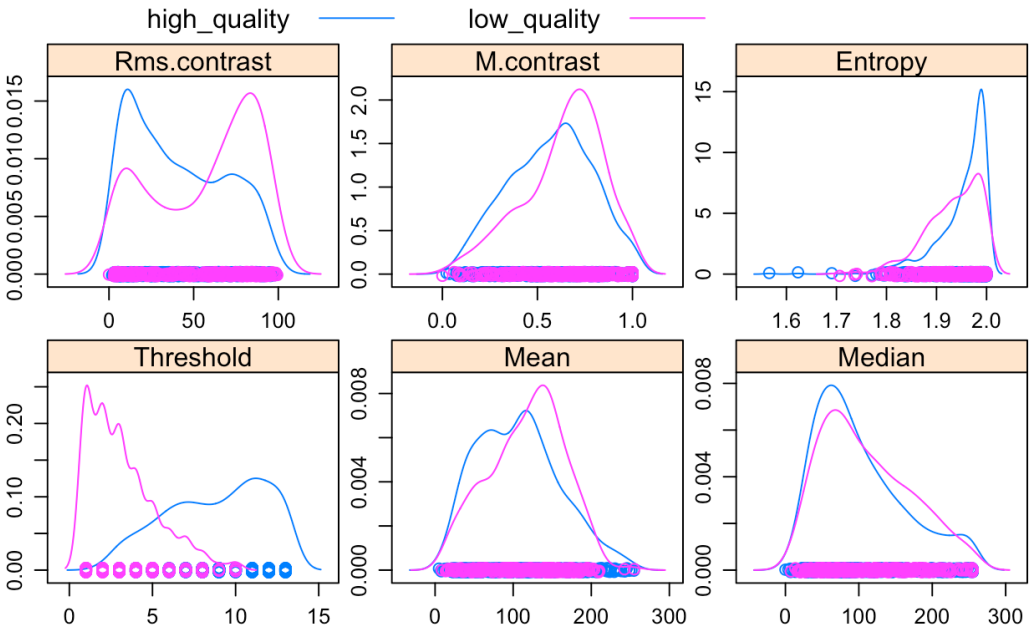
Per prima cosa la generazione di alcune **statistiche di sintesi** sulle covariate permette di osservare, per esempio, che la covariata *Threshold* assume valori nell'intervallo [1,13] e che l'entropia massima di un chunk è 2.

```
> summary(data.frame(image_compression.features, image_compression.target))
   Threshold          Mean             Median        Rms.contrast     M.contrast        Entropy
 Min.   : 1.000   Min.   :  4.984   Min.   :  0.00   Min.   : 0.00   Min.   :0.0000   Min.   :1.566
 1st Qu.: 4.000   1st Qu.: 69.672   1st Qu.: 57.50   1st Qu.:16.79   1st Qu.:0.4263   1st Qu.:1.919
 Median : 7.000   Median :111.570   Median : 89.25   Median :43.91   Median :0.6193   Median :1.963
 Mean   : 7.042   Mean   :110.282   Mean   :104.66   Mean   :45.81   Mean   :0.5906   Mean   :1.946
 3rd Qu.:11.000   3rd Qu.:145.293   3rd Qu.:143.00   3rd Qu.:73.83   3rd Qu.:0.7543   3rd Qu.:1.988
 Max.   :13.000   Max.   :255.000   Max.   :255.00   Max.   :99.35   Max.   :1.0000   Max.   :2.000
 image_compression.target
 high_quality:786
 low_quality :368
```

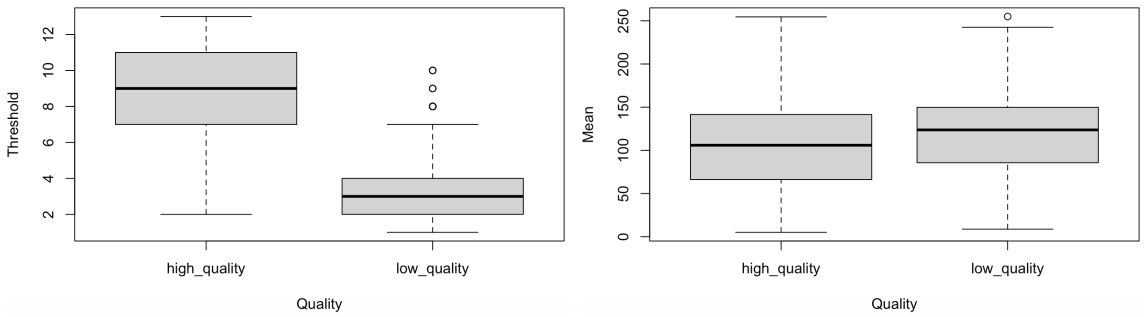
La covariata *Quality* è il target del modello e può assumere valore "high quality" oppure "low quality". Il grafico a torta mette in evidenza il fatto che il dataset è leggermente sbilanciato verso chunk di alta qualità.

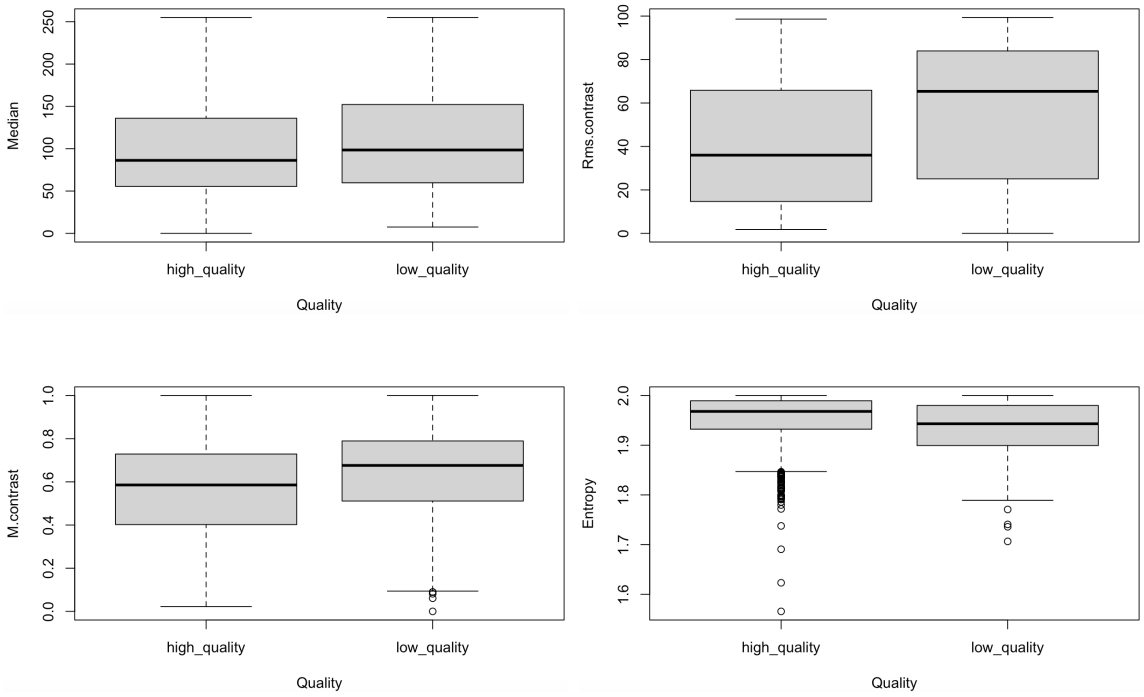


Visualizzando le distribuzioni delle covariate rispetto al target si nota che sono in generale molto sovrapposte, a eccezione della covariata *Threshold* che presenta un buon potere di discriminazione della qualità. Le covariate con peggiore potere discriminativo sono *Mean* e *Median*



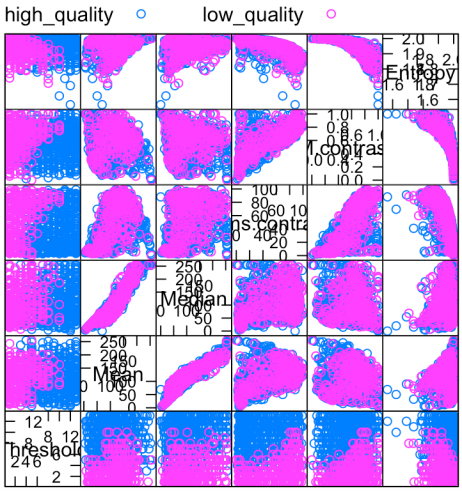
La generazione dei boxplot delle covariate rispetto al target, permette di osservare, anche in questo caso, come *Threshold* sia l'attributo che discrimina meglio la qualità di un chunk.





Una analisi multivariata permette di analizzare le relazioni tra le covariate, in particolare è possibile vedere come la covariata *Threshold* ha un buon potere discriminativo anche in relazione con le altre covariate, in particolare con quelle relative agli indici di contrasto.
In particolare ci aspettiamo che il dataset sia parzialmente separabile sfruttando queste particolari coppie di covariate.

Inoltre si può osservare che le covariate *Mean* e *Median*, come atteso, sono correlate, e che anche *Entropy* e gli indici di contrasto sono correlati.



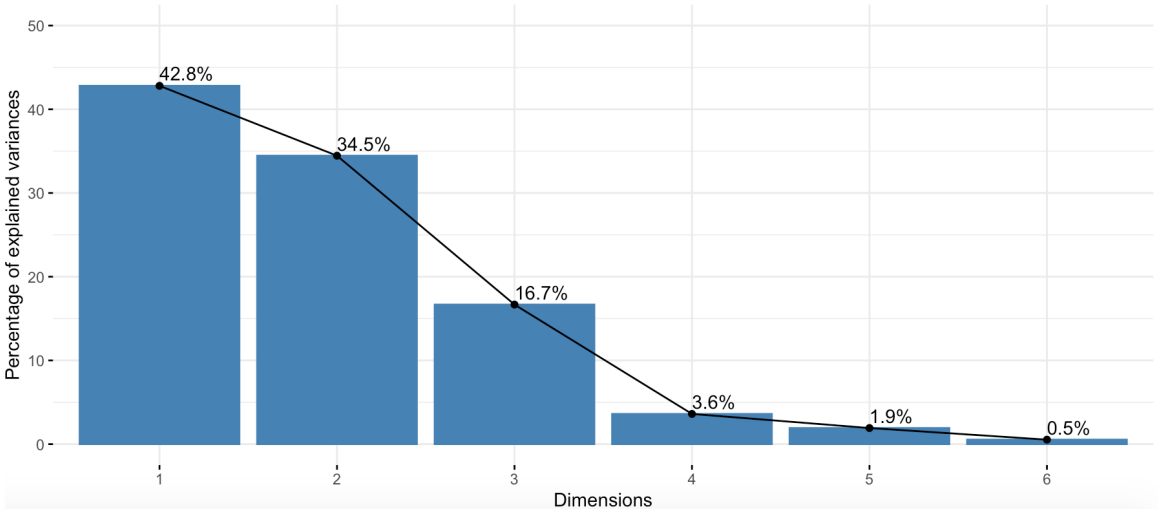
3.2 Principal component analysis - PCA

Una valutazione preliminare della PCA permette di vedere gli autovalori che corrispondono alla varianza spiegata per una specifica dimensione.
In particolare, si è ritenuto ragionevole scegliere di conservare le prime tre dimensioni, che permettono di spiegare il 93.93169 % di varianza cumulata.

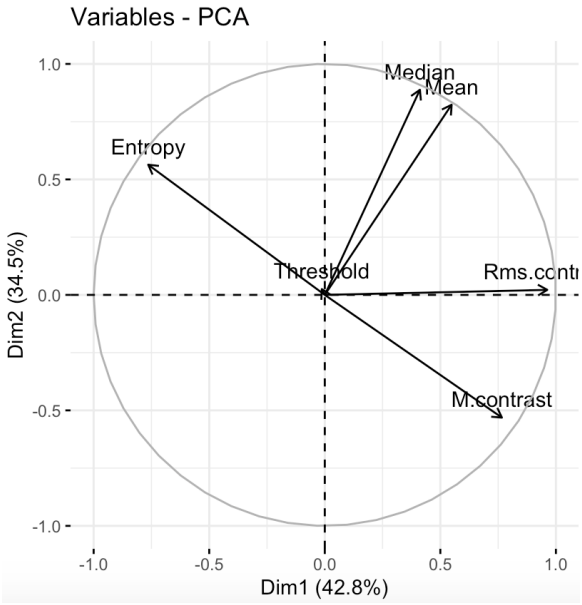
Si è scelto di conservare le prime 3 dimensioni secondo la regola di **Kaiser (1961)**, secondo il quale le componenti principali più significative sono indicate da un autovalore maggiore di 1. Si nota inoltre che i dati sono stati standardizzati.

```
> get_eigenvalue(image_compression.pca)
      eigenvalue variance.percent cumulative.variance.percent
Dim.1 2.56831719       42.8052865                    42.80529
Dim.2 2.06729497       34.4549162                    77.26020
Dim.3 1.00028943       16.6714906                    93.93169
Dim.4 0.21647047        3.6078411                    97.53953
Dim.5 0.11546517        1.9244195                    99.46395
Dim.6 0.03216277        0.5360462                   100.00000
```

E' possibile avere anche una rappresentazione grafica della varianza spiegata come riportato nel seguente grafico. Si nota che il grafico si appiattisce molto a partire dalla quarta dimensione.



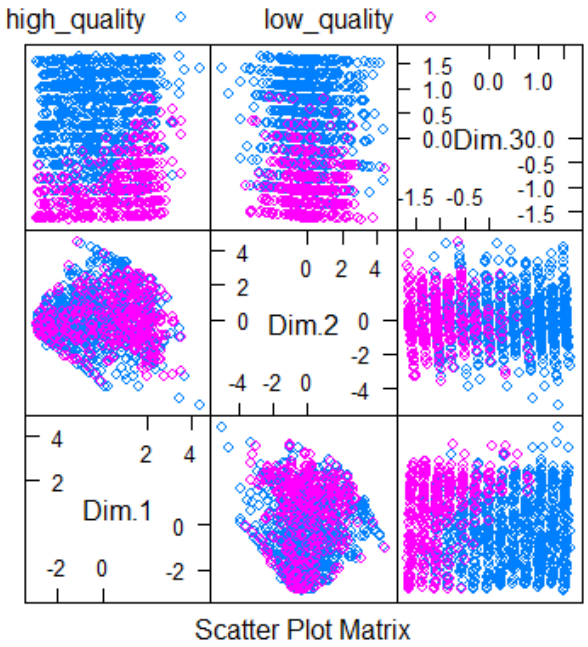
E' possibile inoltre notare come ci siano delle correlazioni all'interno del dataset, come è stato osservato anche nel grafico multivariato. In particolare *Median* e *Mean* sono positivamente correlate, *Entropy* e *M.contrast* sono negativamente correlate, mentre *Rms.contrast* e *Threshold* non presentano correlazioni con altre variabili. Questa è una ulteriore indicazione che la scelta più ragionevole sia conservare un numero di componenti principali pari a 3.



Per quanto riguarda il contributo delle covariate nel nuovo spazio tridimensionale delle componenti principali, è possibile osservare che *Threshold* contribuisce circa del 99.87 % nella rappresentazione della terza dimensione. Ci aspettiamo questo risultato in quanto questa covariata è stata generata randomicamente ed è indipendente dalle altre. La prima dimensione ha contributo maggioritario degli indici di contrasto e di *Entropy*, mentre la seconda di *Mean* e *Median*.

```
> image_compression.pca$var$contrib
                   Dim.1       Dim.2        Dim.3
Threshold     0.006658045  0.03034823 99.876591757
Mean         11.721168832 32.78197261  0.013586755
Median        6.594814414 38.14108137  0.004151011
Rms.contrast 36.083883378  0.02304048  0.017528743
M.contrast   22.864404397 13.65582107  0.075935671
Entropy      22.729070934 15.36773623  0.012206062
```

Infine, come è visibile nella figura di seguito, la Dimensione 1 e la Dimensione 3 ci suggeriscono una potenziale, seppur parziale,separabilità del dataset già nello spazio di rappresentazione originario. Di seguito terremo conto di questo fattori per la scelta dei modelli di addestramento più idonei. Un suggerimento di ciò era emerso anche durante l'analisi multivariata tra *Threshold* e gli indici di contrasto.



3.3 Metodo di valutazione

Le valutazioni sulle performance del modello sono state generate addestrando i modelli sul **dataset completo**, effettuando una **10 times 10-fold cross validation**.

Abbiamo ritenuto valido questo processo in quanto la dimensione ridotta del set di individui all'interno del dataset è tale limitarci nella scelta di una tecnica come quella di **holdout**, a causa del rischio di avere tagli con scarse capacità rappresentative dei campioni, e nella scelta di una tecnica come **leave-one-out** che avrebbe invece richiesto un'eccessiva spesa computazionale.

La scelta del processo di cross validation garantisce di **sfruttare tutti i nostri dati** per il training del modello, allo stesso tempo riducendo la potenziale varianza grazie alle 10 ripetizioni, e impiegando un **tempo accettabile** anche nel caso peggiore, cioè quello del training di neural network.
Abbiamo deciso di omologare il processo (10 ripetizioni per 10 fold) per tutti i modelli addestrati, in modo da effettuare una comparazione sullo stesso livello, sia dal punto di vista della **spesa computazionale dei modelli**, sia da quello dell'**affidabilità** dei risultati ottenuti.

Capitolo 4

Modello 1 - Decision tree

Il primo modello che si è analizzato a seguito dello svolgimento della PCA è il **decision tree** o **albero di decisione**. Esso ci ha permesso di ottenere una discreta classificazione della qualità dei chunk, e nel complesso rappresenta un compromesso **efficace** ed **efficiente** dal punto di vista della complessità computazionale.

Nel seguito è presentata una descrizione del modello e le motivazioni che ci hanno portato a considerarlo come valida soluzione in relazione agli obiettivi del progetto. Inoltre sono state generate ed analizzate le misure di performance del modello proposto, valutate tramite l'utilizzo della 10 times 10-cross fold validation.

4.1 Motivazione

Gli obiettivi del progetto hanno portato alla scelta degli alberi di decisione come possibile modello ad apprendimento supervisionato per la classificazione della qualità dei chunk. Le motivazioni che si sono considerate al fine di utilizzarlo sono molteplici e riguardano in primo luogo il contesto di classificazione. Esso è uno dei modelli più **semplici e veloci**, durante la comparazione dei modelli questo risulta sicuramente un parametro da tenere in considerazione. Infatti la scelta del modello comporta in generale una valutazione del **tradeoff** che esiste tra complessità del modello e accuratezza.

La **comunicazione** è uno dei punti chiave del modello, infatti consente una rappresentazione visiva del processo decisionale nel quale i rami dell'albero mostrano esplicitamente tutti quei fattori all'interno dell'analisi che sono considerati rilevanti per la decisione.

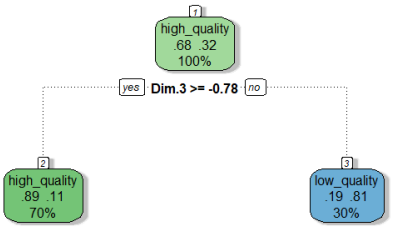
Infine, l'analisi del dataset ci ha fatto notare che **alcune covariate sono particolarmente discriminative** (in particolare *Threshold* e *Rms.contrast*), e nonostante pensiamo di ottenere performance migliori dai successivi due modelli (**Support vec-**

tor machine e **Neural Network**) a causa della parziale separabilità del dataset, riteniamo comunque interessante compararli ad un modello più semplice, in modo da valutare se in questo contesto è realmente necessario l'utilizzo di modelli più complessi.

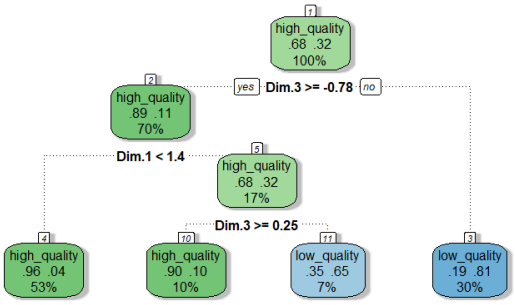
4.2 Descrizione

Per il training del modello sono stati valutati dalla libreria tre possibili tagli di generazione dell'albero, in funzione di tre possibili complexity parameter: 0.01494565 (modello rpart1), 0.03125 (modello rpart2) e 0.58152174 (modello rpart3).

(a) Modello rpart3



(b) Modello rpart2

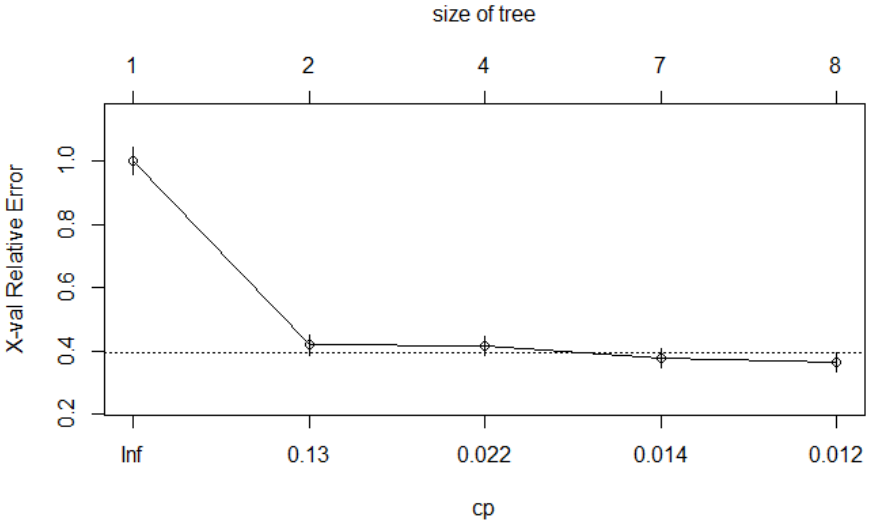


(c) Modello rpart1

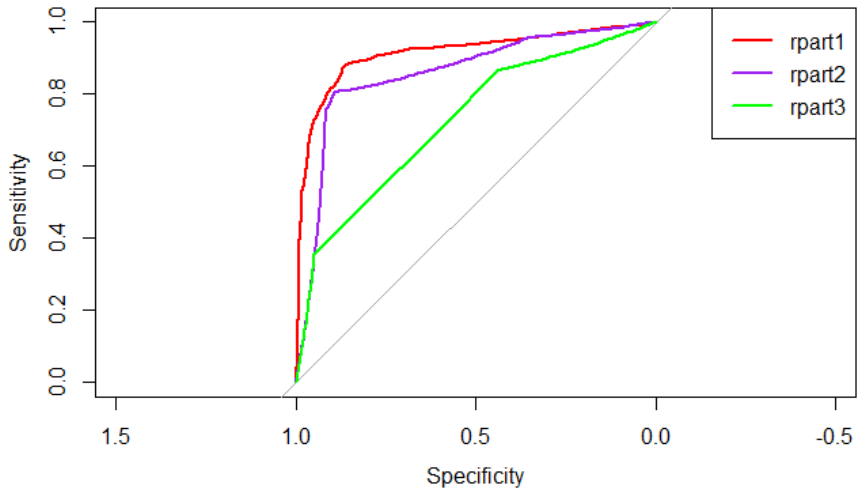
Il **modello rpart3** è rappresentato dalla sola radice e classifica le istanze come high_quality o low_quality con probabilità basata sulla suddivisione del dataset (68:32), Tale modello è stato scartato nella selezione ma è stato interessante valutarlo per capire come si paragona un modello unilaterale rispetto agli altri due, anche per capire se il dataset fosse eccessivamente sbilanciato.

Il **modello rpart2** presenta solo la condizione sulla Dimensione 3 (caratterizzata da un contributo maggioritario del 99,88% della covariata *Threshold*), che ottiene delle buone performance generali, ma che intrinsecamente rigetta ogni possibilità delle altre covariate di influenzare la classificazione. E' interessante per comprendere fino a quanto è possibile ottenere una buona classificazione dei chunk valutando **solo la misura di taglio** (che è indipendente dal chunk stesso). Nel caso in cui non si riuscissero ad ottenere miglioramenti netti rispetto a questo modello le proprietà stesse dell'immagine sarebbero scarsamente influenti sulla classificazione e basterebbe scegliere solo la soglia di taglio (che è indipendente dall'immagine) per decidere la qualità della compressione.

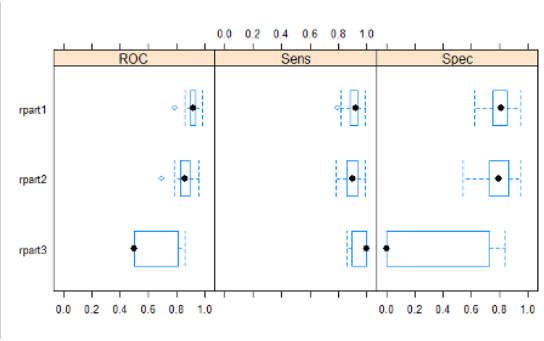
Il modello migliore scelto tra i decision tree è stato il **modello rpart1** che presenta performance migliori, ed è anche quello scelto dalla libreria rpart, nonostante ci sia una leggera tendenza all'overfitting in quanto, come possiamo notare dalla figura di seguito, la diminuzione dell'errore relativo è discreta per tale complexity parameter. Come atteso e visibile dai modelli, le dimensioni che vengono selezionate dagli alberi per discernere sono principalmente la Dimensione 1 (caratterizzata principalmente dalla *Threshold*) e la Dimensione 2 (caratterizzata principalmente dalle misure di contrasto). Infatti la Dimensione 3, che è principalmente caratterizzata da *Mean* e *Median* con scarsa capacità discriminatoria sul dataset, non viene tenuta in conto da nessuno dei decision tree.

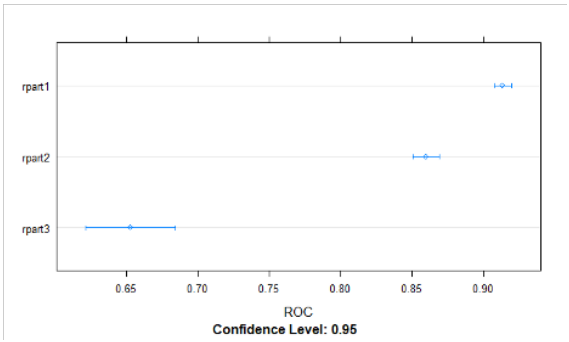


Vengono quindi riportate sul grafico le curve ROC relative ai tre modelli degli alberi di decisione. Anche in questo caso è possibile osservare come il modello rpart1 sia quello più performante.



Inoltre, gli intervalli di confidenza confrontati per i tre modelli rivelano come il modello rpart1 sia quello con intervallo più stretto e vicino al valore ottimo, mentre i grafici boxplot mostrano come anche le distribuzioni dei valori di sensibility e specificity per rpart1 siano meglio rappresentate delle altre.





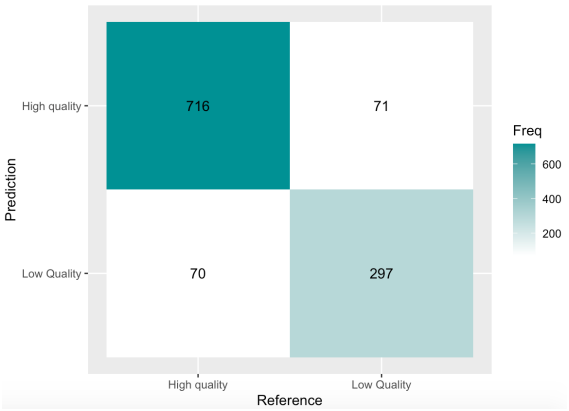
In seguito sarà quindi preso in considerazione solamente il modello rpart1.

4.3 10-fold cross validation e misure di performance

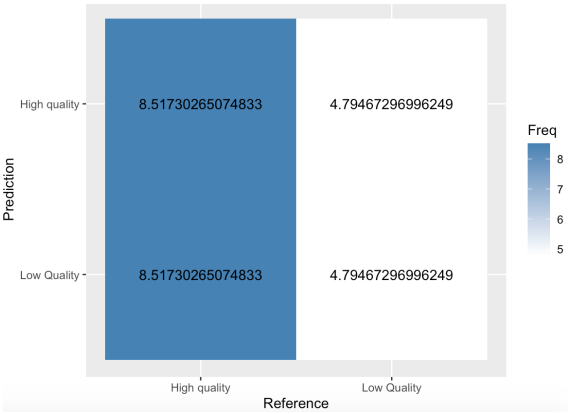
Nella seguente sezione verranno valutate le misure di performance specifiche del modello scelto per il confronto finale (**rpart1**).

4.3.1 Matrice di confusione complessiva

In seguito all'esecuzione della 10 times 10-fold cross validation, è stata valutata la matrice di confusione complessiva sommando le 10 matrici di confusione associate alle 10 fold di test prodotte in ogni ripetizione, e trovandone la media aritmetica per le 10 ripetizioni, arrotondata poi all'intero.



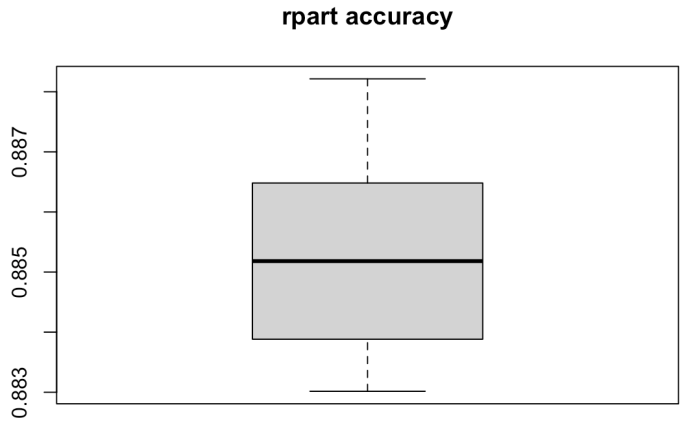
(a) matrice di confusione



(b) cm - standard deviation

Il summary delle accuratezze mostra i valori rappresentati sul boxplot.

Min	1° Qu.	Median	Mean	3° Qu.	Max
0.8830	0.8839	0.8852	0.8853	0.8865	0.8882



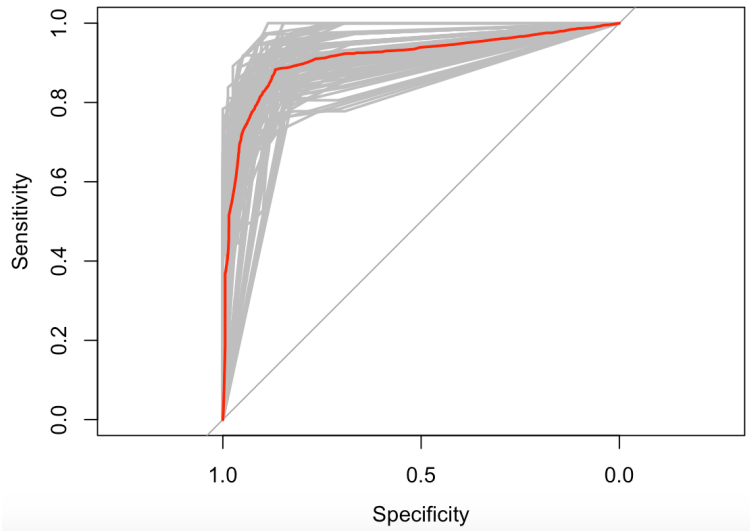
4.3.2 Precision, recall, f-measure, ROC e AUC

La matrice di confusione complessiva ha permesso di calcolare le misure di precision e recall, per il quale sono anche state calcolate le ulteriori macro e micro average. I valori $F_{0.5}, F_1$ ed F_2 sono invece riferiti alle F-measure.

Decision tree	Precision	Recall
High quality	0.9117647	0.9071247
Low quality	0.8037634	0.8125
Macro avg	0.8577641	0.8598123
Micro avg	0.8773241	0.8769497

$F_{0.5}$	F_1	F_2
0.9108329	0.9094388	0.9080489

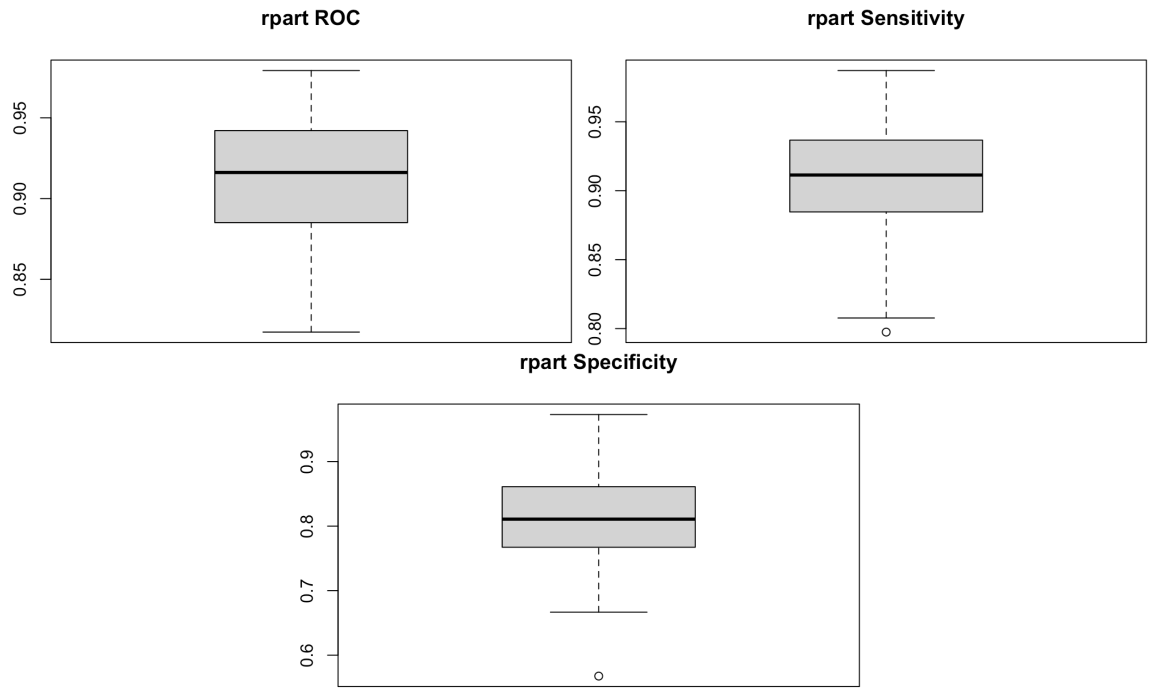
Il seguente grafico mostra le **curve roc** eseguite sulle 10 times 10-cross fold. Si osserva che il grafico in rosso è la ROC media, mentre in grigio è possibile osservare quanto variano le curve ROC, e di conseguenza, la stabilità del modello.



AUC
0.9127

Il **summary** sulle misure ROC, sensibility e specificity mostrano come sono distribuiti i dati, anche in questo caso è stata prodotto una rappresentazione grafica tramite **boxplot**.

```
> summary(rpart.model$resample)[,1:3]
      ROC              Sens             Spec
 Min.   :0.8173   Min.   :0.7975   Min.   :0.5676
 1st Qu.:0.8854   1st Qu.:0.8846   1st Qu.:0.7725
 Median :0.9162   Median :0.9114   Median :0.8108
 Mean   :0.9127   Mean   :0.9070   Mean   :0.8120
 3rd Qu.:0.9420   3rd Qu.:0.9367   3rd Qu.:0.8611
 Max.   :0.9793   Max.   :0.9872   Max.   :0.9730
```



In conclusione vengono riasunti nella seguente tabella i **tempi**, espressi in secondi, per l'addestramento dei modelli e per le predizioni.

Timing [s]	rpart
Everything	2.83
Final	0.00
Prediction	NA

Capitolo 5

Modello 2 - Support vector machine

Il secondo modello che si è analizzato a seguito dello svolgimento della PCA è la **support vector machine** o **svm**. Questo modello introduce una maggior, seppure accettabile, complessità computazionale rispetto al decision tree, anche se come vedremo presenta dei vantaggi in termini di performance.

Nel seguito sono presentate le motivazioni che ci hanno portato a considerarlo come valida soluzione in relazione agli obiettivi del progetto. Come nel capitolo precedente sono state generate ed analizzate le misure di performance del modello, valutate tramite predizione con l'utilizzo della 10-fold cross validation.

5.1 Motivazione

Gli obiettivi del progetto hanno portato alla scelta della svm come possibile modello per la classificazione della qualità dei chunk. Come per l'albero di decisione, si è cercato un modello che fosse utile per l'apprendimento supervisionato, e che riuscisse a classificare la qualità dei chunk.

La scelta è ricaduta sulla svm, in primo luogo perchè l'analisi esplorativa del dataset ha suggerito la possibilità di **separare i dati tramite un iperpiano**, questo è stato ulteriormente confermato nella pratica come presentato in seguito.
In particolare, dall'analisi multivariata delle dimensioni abbiamo notato che Dimensione 1 e la Dimensione 3 garantiscono una parziale separabilità lineare del dataset, per cui abbiamo deciso di utilizzare un **kernel lineare**, che ci permette, inoltre, di visualizzare efficacemente il risultato.

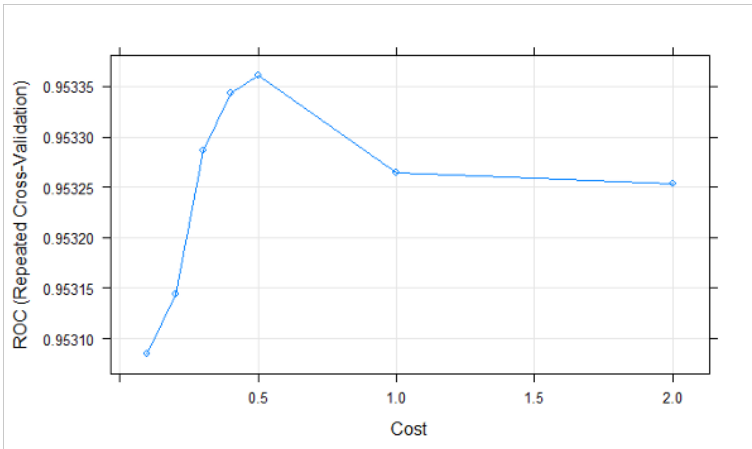
In secondo luogo la svm ha dei costi computazionali che, seppur superiori a quelli del decision tree, sono comunque accettabili nel contesto del problema e quindi po-

trebbe essere un modello valido anche nel caso di estensione del dominio, sia in termini di numero di covariate o dimensioni, sia in termini di istanze del dataset, per permettere una migliore validazione del modello o per una possibile estensione del modello ad immagini a colori o con chunk di dimensione variabile, che può comportare una crescita spazio di rappresentazione degli indici con potenziale necessità di addensare le distribuzioni.

5.2 Descrizione

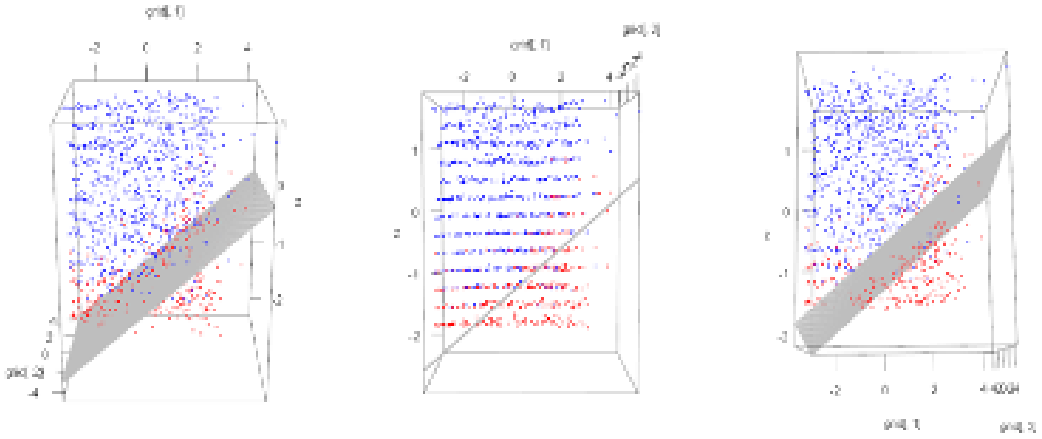
Per la selezione del modello svm finale sono stati valutati dalla libreria tre possibili modelli di svm, ognuno con un costo diverso. In particolare i costi analizzati sono stati: 0.002, 0.5 e 1. I rispettivi modelli sono **svm0002**, **svm05** e **svm1**.

Il seguente grafico rappresenta sulle ordinate il costo mentre sulle ascisse il valore AUC in funzione del costo con cui si è trainato il modello. E' possibile osservare che il valore massimo della funzione si raggiunge con un costo pari a 0.5. Per cui potrebbe esserci una tendenza all'overfitting per costi più alti.

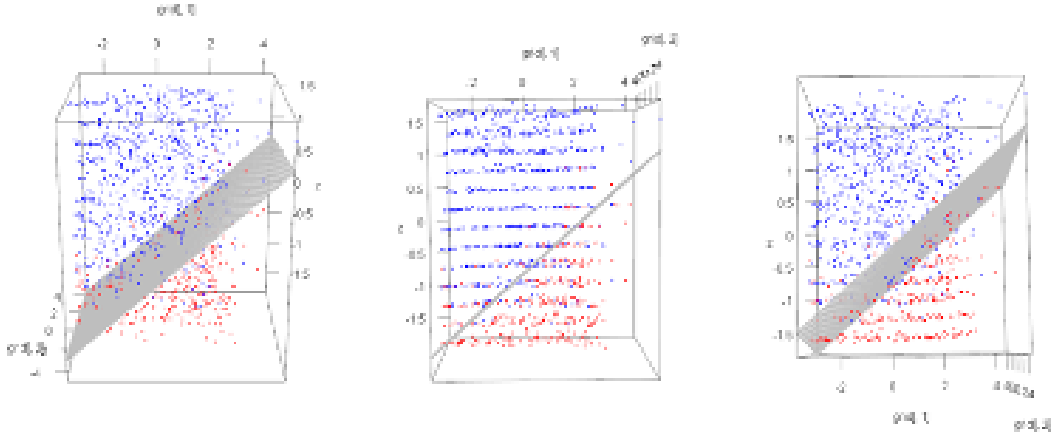


La dimensionalità del dataset ridotto permette la rappresentazione in tre dimensioni dei piani utilizzati dai tre modelli di svm.

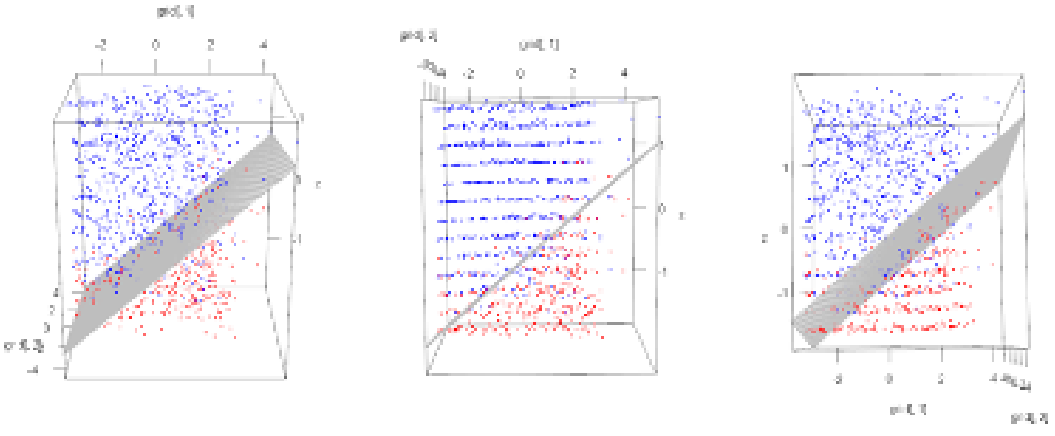
Graficamente i tre modelli sembrano molto simili, coerentemente con la similitudine delle loro misure AUC. Si nota che non vengono rappresentati i margini delle tre svm. Dai grafici si può però notare che svm0002 ha un iperpiano che tende verso il basso, classificando meglio le istanze *low_quality* (in blu), e peggio quelle *high_quality* (in rosso).



(a) svm c = 0.002

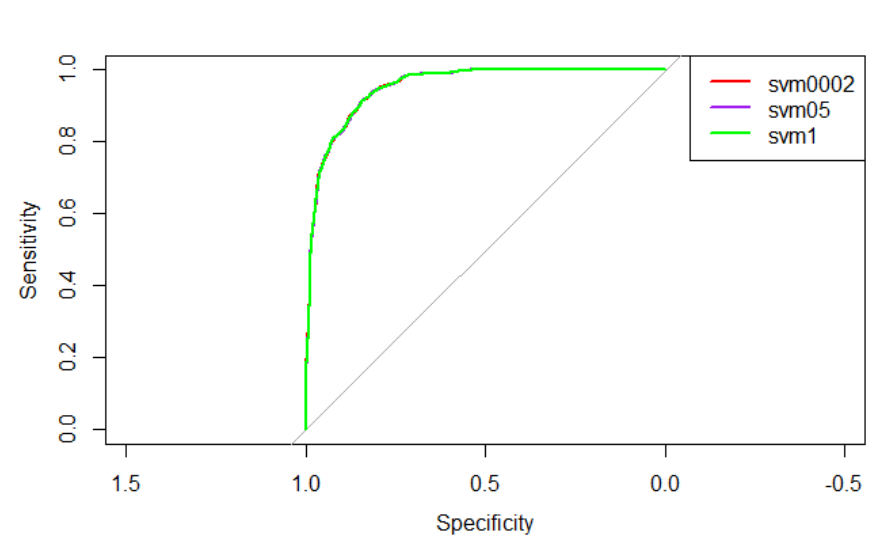


(b) svm c = 0.5

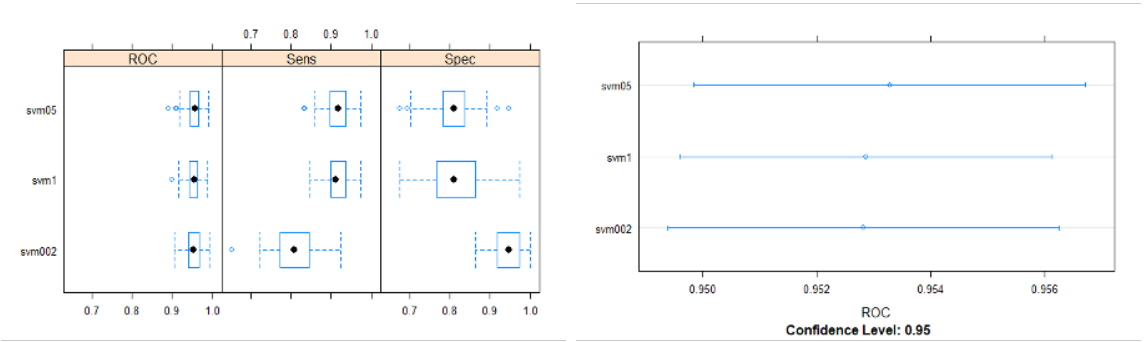


(c) svm c = 1

Vengono quindi riportate sul grafico le curve ROC relative ai tre modelli di svm proposti. In questo caso le tre curve ROC sono molto simili come ci si poteva attendere anche dalle AUC molto simili.



Inoltre, gli intervalli di confidenza confrontati per i tre modelli rivelano come il modello preferibile sia svm05, tuttavia i modelli in questo caso sono molto simili.



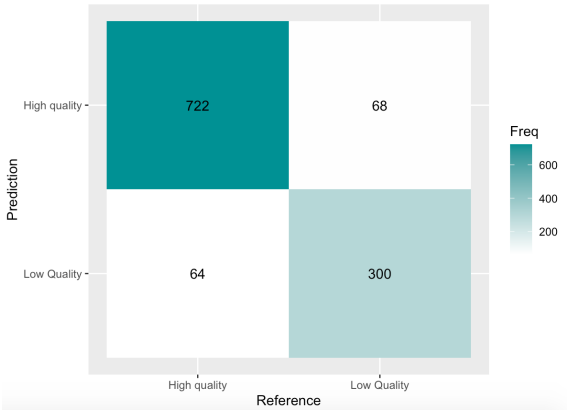
Come visibile nel grafico a sinistra, il modello **svm0002** presenta poco equilibrio tra Sensitivity e Specificity, a causa della sua preferenza per la classificazione delle istanze *low_quality*, e per questo motivo è stato scartato. Per quanto riguarda *svm1* e *svm05*, abbiamo preferito il modello a costo minore in quanto preferibile per ridurre un potenziale overfitting.

5.3 10 times 10-fold cross validation e stima delle seguenti misure di performance

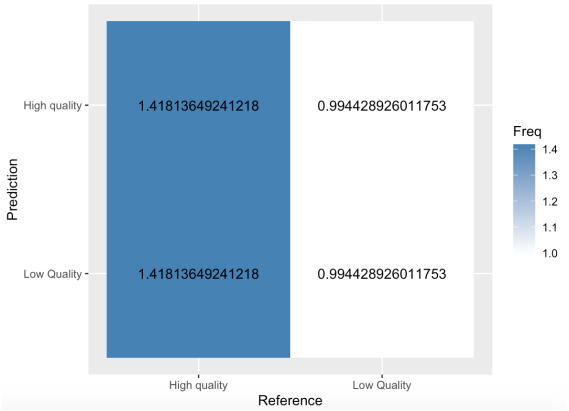
Nella seguente sezione verranno valutate le misure di performance specifiche del modello scelto per il confronto finale (**svm05**).

5.3.1 Matrice di confusione complessiva

In seguito all'esecuzione della 10 times 10-fold cross validation, è stata valutata la matrice di confusione complessiva sommando le 10 matrici di confusione associate alle 10 fold di test prodotte in ogni ripetizione, e trovandone la media aritmetica per le 10 ripetizioni, arrotondata poi all'intero.



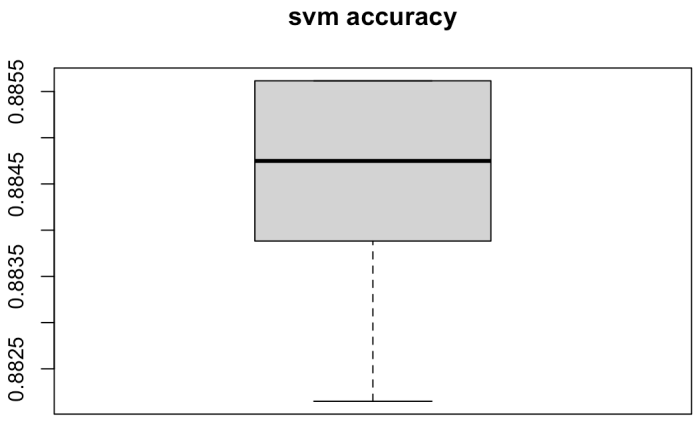
(a) matrice di confusione



(b) cm - standard deviation

Il summary delle accuratezze mostra i valori rappresentati sul boxplot.

Min.	1° Qu.	Median	Mean	3° Qu	Max
0.8821	0.8839	0.8847	0.8846	0.8856	0.8856



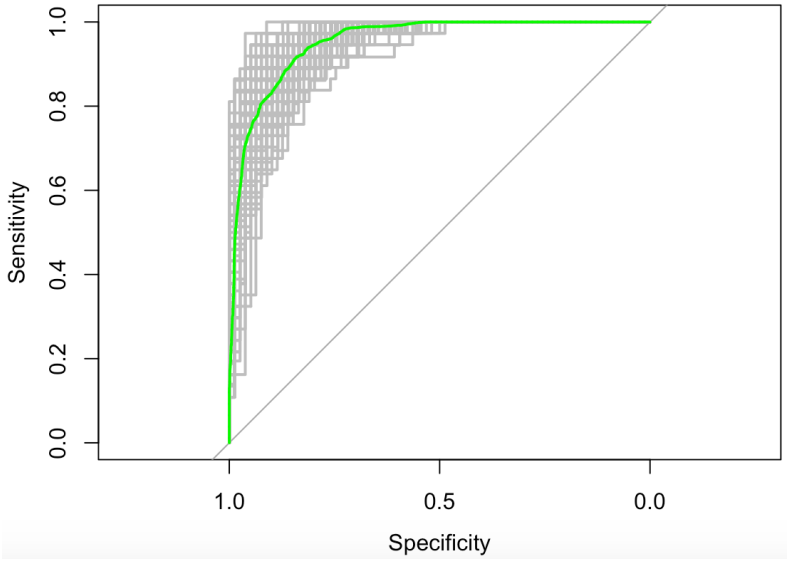
5.3.2 Precision, recall, f-measure, ROC e AUC

La matrice di confusione complessiva ha permesso di calcolare le misure di precision e recall, per il quale sono anche state calcolate le ulteriori macro e micro average. I valori $F_{0.5}, F_1$ ed F_2 sono invece riferiti alle F-measure.

SVM	Precision	Recall
High quality	0.9137056	0.9160305
Low quality	0.8196721	0.8152174
Macro avg	0.8666889	0.865624
Micro avg	0.8837192	0.8838821

$F_{0.5}$	F_1	F_2
0.9141696	0.9148666	0.9155646

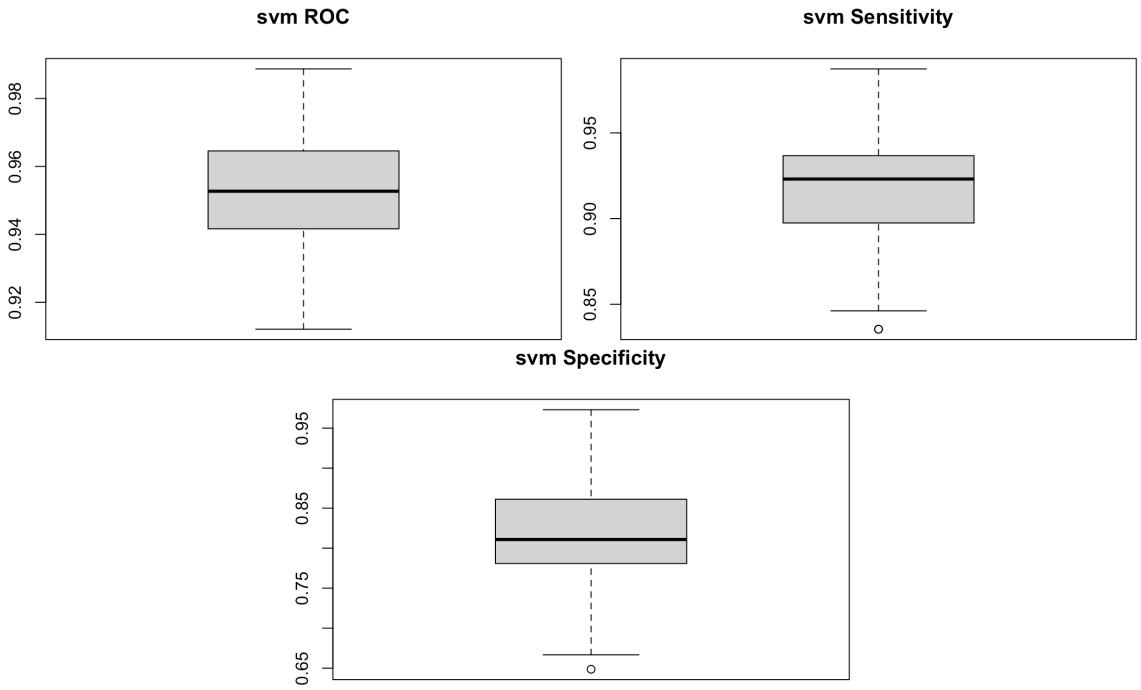
Il seguente grafico mostra le **curve roc** eseguite sulle 10 times 10-cross fold. Si osserva che il grafico in rosso è la ROC media, mentre in grigio è possibile osservare quanto variano le curve ROC, e di conseguenza, la stabilità del modello.



AUC
0.9530

Il **summary** sulle misure ROC, sensibility e specificity mostrano come sono distribuiti i dati, anche in questo caso è stata prodotto una rappresentazione grafica tramite **boxplot**.

```
> summary(svm.model$resample)[,1:3]
      ROC              Sens              Spec
 Min.   :0.9121   Min.   :0.8354   Min.   :0.6486
 1st Qu.:0.9419   1st Qu.:0.8974   1st Qu.:0.7823
 Median :0.9527   Median :0.9231   Median :0.8108
 Mean   :0.9530   Mean   :0.9166   Mean   :0.8144
 3rd Qu.:0.9643   3rd Qu.:0.9367   3rd Qu.:0.8611
 Max.   :0.9887   Max.   :0.9873   Max.   :0.9730
```



In conclusione vengono riassunti nella seguente tabella i **tempi**, espressi in secondi, per l'addestramento dei modelli e per le predizioni.

Timing [s]	svm
Everything	6.95
Final	0.05
Prediction	NA

Capitolo 6

Modello 3 - Neural Network

Il terzo modello che è stato analizzato è la **neural network** o **rete neurale**. Il modello della rete neurale, come mostrato in seguito, presenta una complessità computazione, in fase di addestramento, maggiore degli altri due modelli.
Come per gli altri modelli, sono state presentate le motivazioni che hanno contribuito a considerare la rete neurale come valida soluzione in relazione agli obiettivi del progetto.

6.1 Motivazione

Come terzo modello è stato scelto quello della rete neurale per riuscire a classificare la qualità dei chunk mediande addestramento supervisionato.

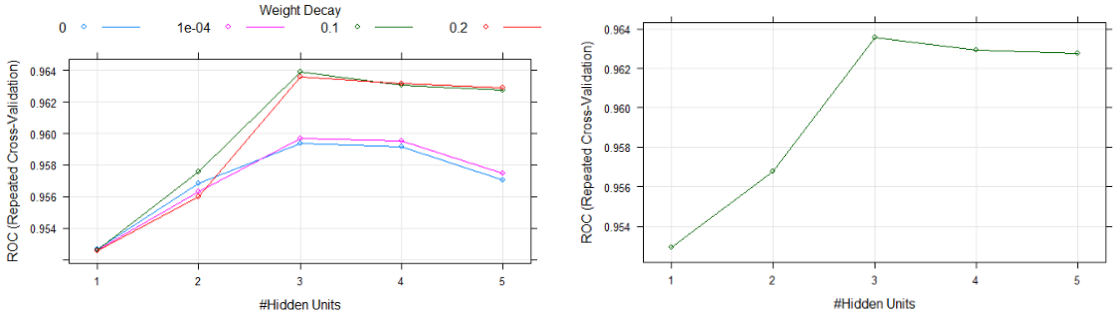
L'analisi iniziale delle dimensioni, nonchè la visualizzazione dei dati nello spazio tridimensionale effettuata nel training della svm, hanno suggerito una possibilità di utilizzare un **classificatore non lineare**. Nonostante quindi la svm fornisca un miglioramento prestazionale rispetto agli alberi di decisione, si è considerata una ulteriore possibilità nella scelta dei modelli.
La rete neurale si è rivelata, in questo specifico caso, la migliore scelta, almeno dal **punto di vista prestazionale**, data la sua capacità di generare un modello di separazione non lineare nel dominio di riferimento. Infatti ci aspettiamo che questo modello abbia prestazioni anche superiori alla svm con kernel lineare poichè il dataset è solo parzialmente linearmente separabile.

Inoltre, poiché il numero di istanze e covariate è ridotto, ci permette di ottenere i migliori risultati ad un costo accettabile. Tuttavia non siamo sicuri che, **ampliando il dominio di riferimento**, essa possa mantenere la stessa flessibilità. In tal caso per raggiungere un buon tuning dei parametri potrebbero essere necessarie maggiori risorse.

6.2 Descrizione

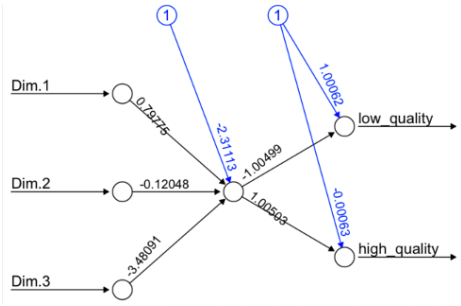
Il modello della rete neurale è stato addestrato, come gli altri modelli, con il package caret. Per la selezione del modello di rete neurale finale sono stati valutati dalla libreria tre possibili modelli di rete.

Come è possibile osservare dal grafico, il valore di unità nascoste che massimizza la misura auc è 3, mentre per il valore di decay il massimo si ottiene per 0.1.

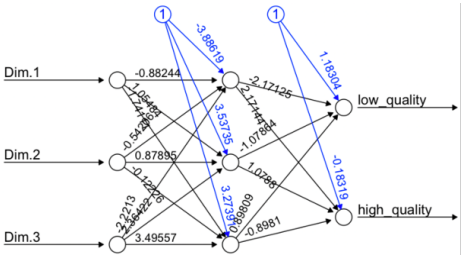


Le tre reti neurali addestrate si presentano come in figura.

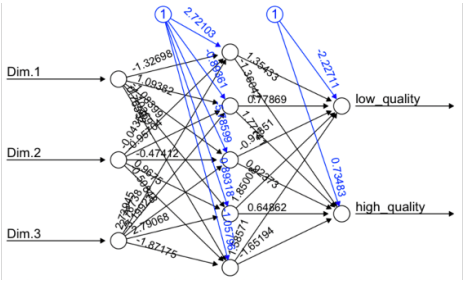
(a) 1 hidden unit



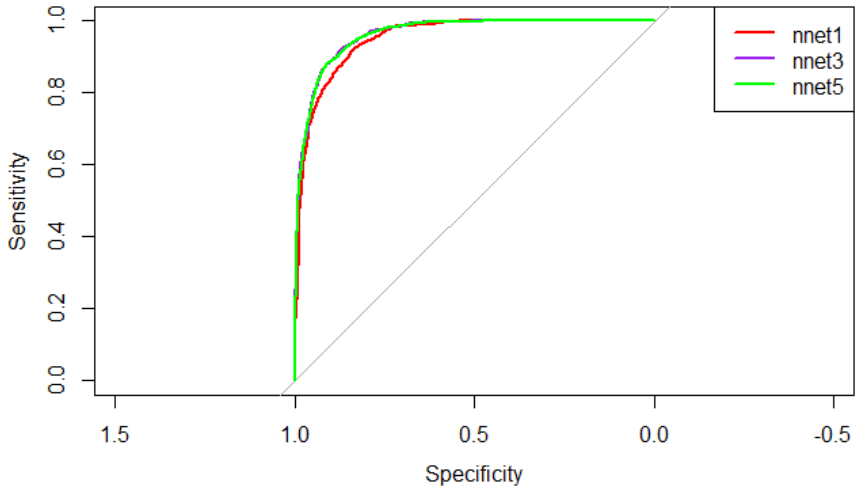
(b) 3 hidden units



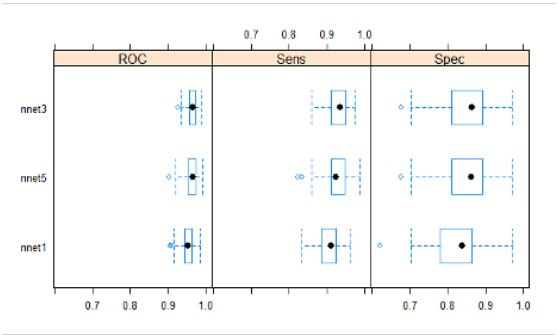
(c) 5 hidden units

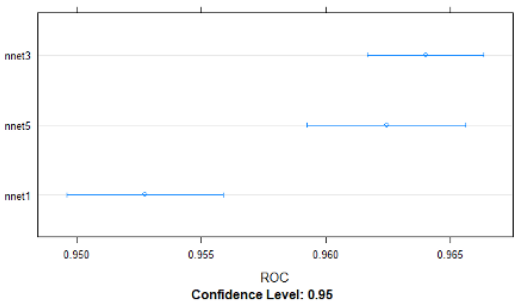


Tutte le reti sono state addestrate con weight decay pari a 0.1, mentre presentano 1,3 e 5 nodi nello strato nascosto.
I relativi modelli sono chiamati **nnet1**, **nnet3** e **nnet5**. Visualizzando il grafico delle curve roc per tutti i tre modelli, si nota che il grafico meglio rappresentato è quello di **nnet3**, nonostante anche gli altri due modelli offrano ottime performance.



Inoltre, gli intervalli di confidenza confrontati per i tre modelli rivelano come il modello preferibile sia nnet a 3 hidden units, che ha sia **una mediana più alta che una varianza minore**.





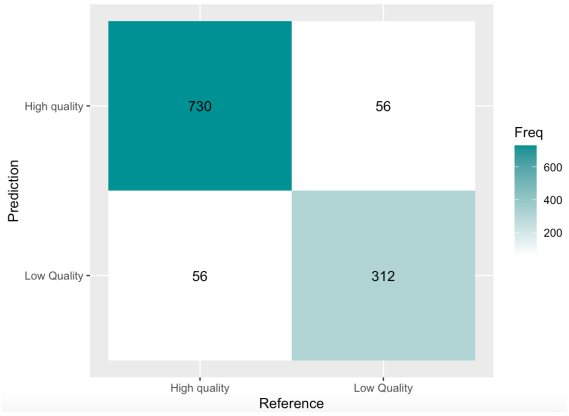
Riguardo l'addestramento della rete, si è notata una tendenza ad andare in overfitting al crescere degli strati nascosti. In generale secondo alcune prove, e secondo quanto mostrato nei grafici sul valore AUC in funzione di hidden units e weight decay, il miglioramento delle prestazioni tende sempre ad appiattirsi oltre le 3 hidden units.

6.3 10 times 10-fold cross validation e stima delle seguenti misure di performance

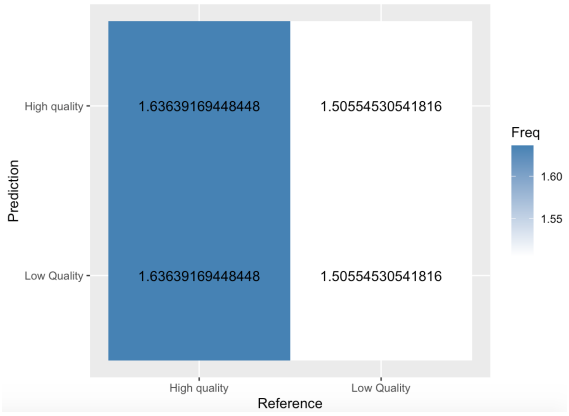
Nella seguente sezione verranno valutate le misure di performance specifiche del modello scelto per il confronto finale (**nnet3**).

6.3.1 Matrice di confusione complessiva

In seguito all'esecuzione della 10 times 10-fold cross validation, è stata valutata la matrice di confusione complessiva sommando le 10 matrici di confusione associate alle 10 fold di test prodotte in ogni ripetizione, e trovandone la media aritmetica per le 10 ripetizioni, arrotondata poi all'intero.



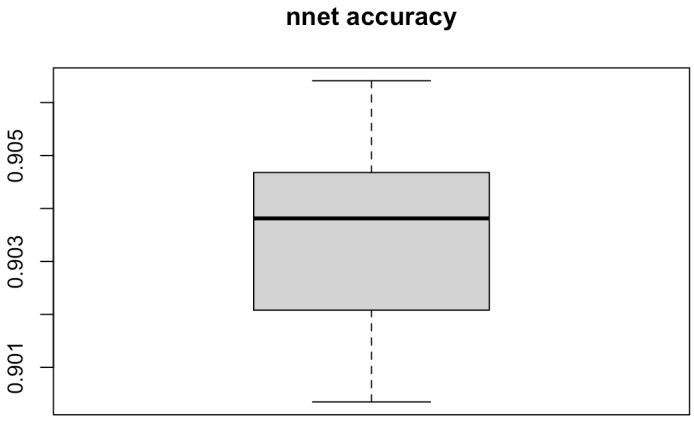
(a) matrice di confusione



(b) cm - standard deviation

Il summary delle accuratezze mostra i valori rappresentati sul boxplot.

Min.	1° Qu.	Median	Mean	3° Qu	Max
0.9003	0.9025	0.9038	0.9036	0.9047	0.9064



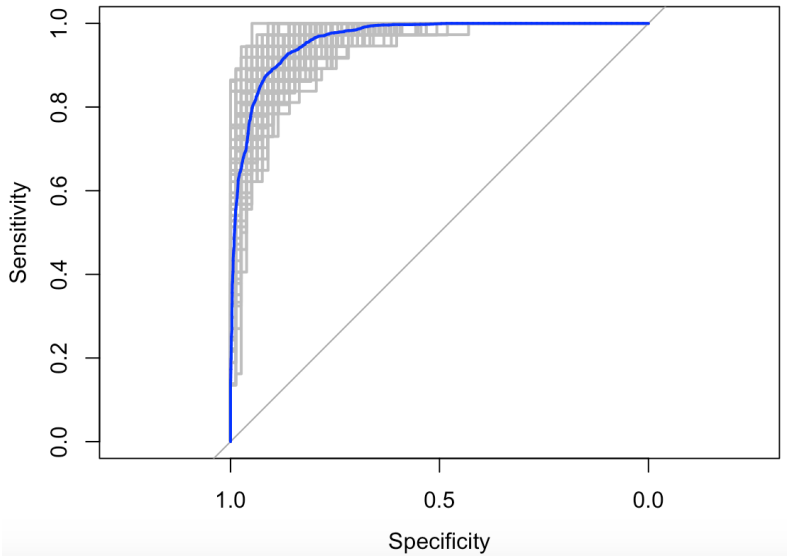
6.3.2 Precision, recall, f-measure, ROC e AUC

La matrice di confusione complessiva ha permesso di calcolare le misure di precision e recall, per il quale sono anche state calcolate le ulteriori macro e micro average. I valori $F_{0.5}, F_1$ ed F_2 sono invece riferiti alle F-measure.

NNET	Precision	Recall
High quality	0.9287532	0.9287532
Low quality	0.8478261	0.8478261
Macro avg	0.8882896	0.8882896
Micro avg	0.9029463	0.9029463

$F_{0.5}$	F_1	F_2
0.9287532	0.9287532	0.9287532

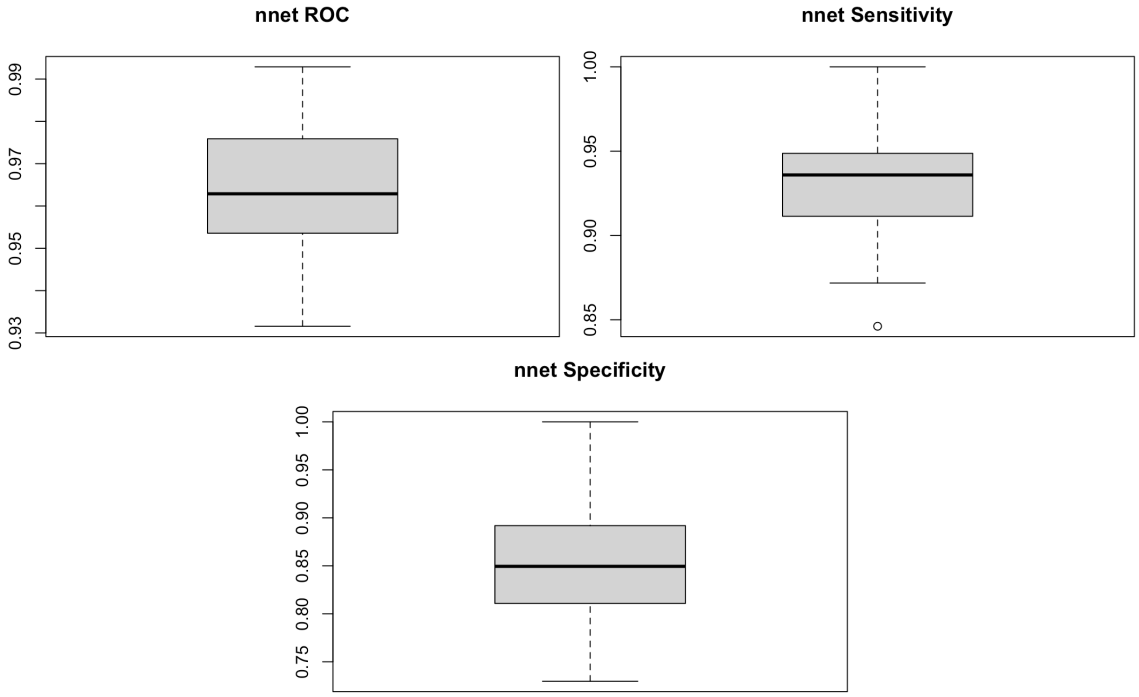
Il seguente grafico mostra le **curve roc** eseguite sulle 10 times 10-cross fold. Si osserva che il grafico in rosso è la ROC media, mentre in grigio è possibile osservare quanto variano le curve ROC, e di conseguenza, la stabilità del modello.



AUC
0.9638

Il **summary** sulle misure ROC, sensibility e specificity mostrano come sono distribuiti i dati, anche in questo caso è stata prodotto una rappresentazione grafica tramite **boxplot**.

```
> summary(nnet.model$resample)[,1:3]
      ROC              Sens             Spec
 Min.   :0.9316   Min.   :0.8462   Min.   :0.7297
 1st Qu.:0.9537   1st Qu.:0.9114   1st Qu.:0.8108
 Median :0.9629   Median :0.9359   Median :0.8495
 Mean   :0.9638   Mean   :0.9291   Mean   :0.8489
 3rd Qu.:0.9758   3rd Qu.:0.9487   3rd Qu.:0.8919
 Max.   :0.9929   Max.   :1.0000   Max.   :1.0000
```



In conclusione vengono riassunti nella seguente tabella i **tempi**, espressi in secondi, per l'addestramento dei modelli e per le predizioni.

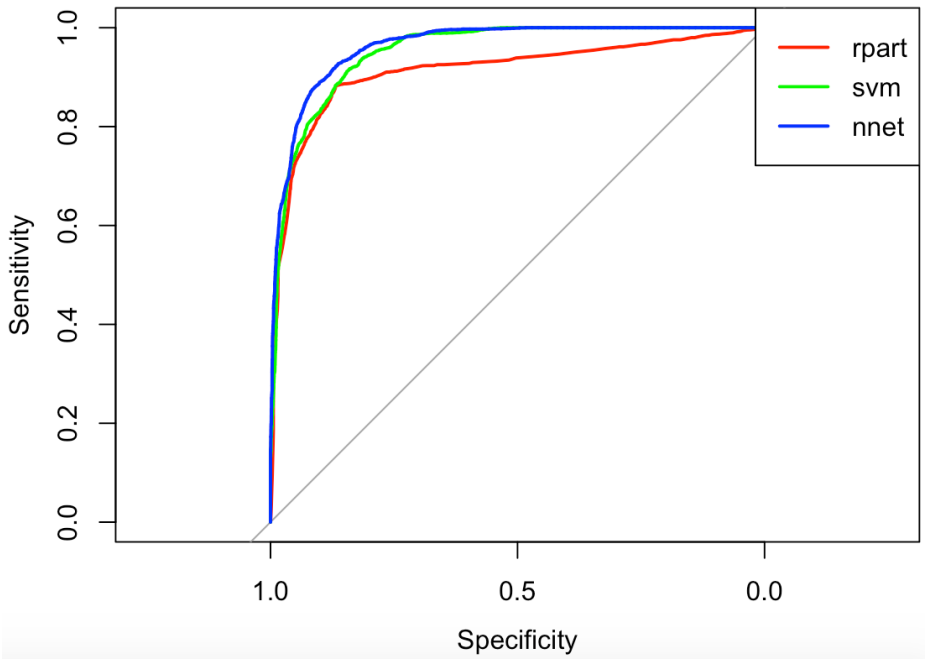
Timing [s]	nnet
Everything	13.80
Final	0.14
Prediction	NA

Capitolo 7

Analisi dei risultati ottenuti e conclusioni

L'ultimo capitolo è dedicato alla decisione del modello migliore in seguito alla comparazione dei modelli analizzati.

Si rappresentano le curve ROC dei tre modelli, e si osserva come la curva della rete neurale sia la migliore. La curva che rappresenta il modello svm è leggermente peggiore della rete neurale, mentre la curva dell'albero di decisione è la peggiore.

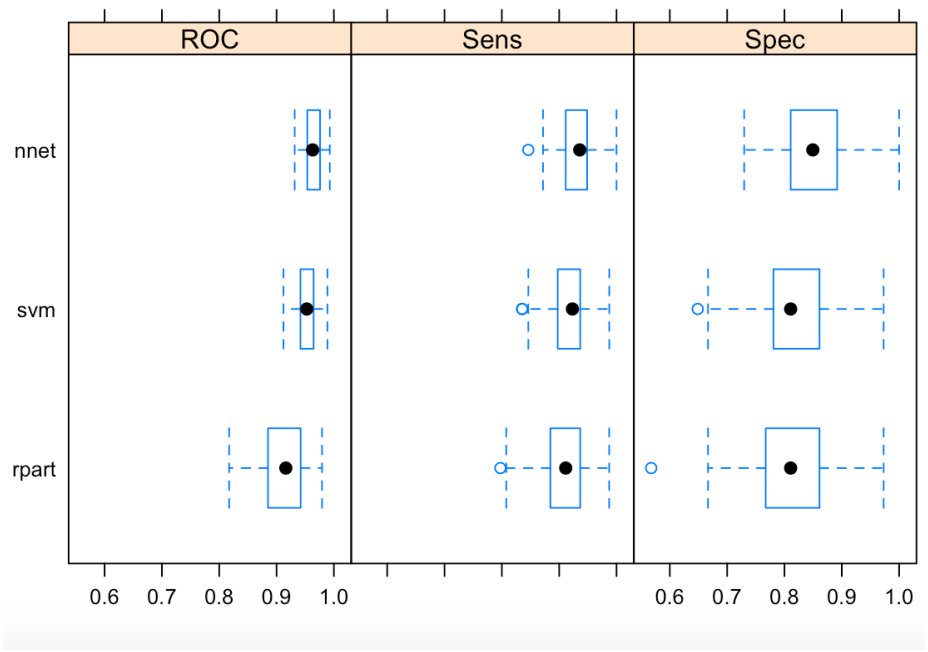


Il **summary** delle misure di performance permette di visualizzare, per i tre modelli,i valori per le misure ROC, Sensibility e Specificity, essi vengono successivamente rappresentati graficamente.

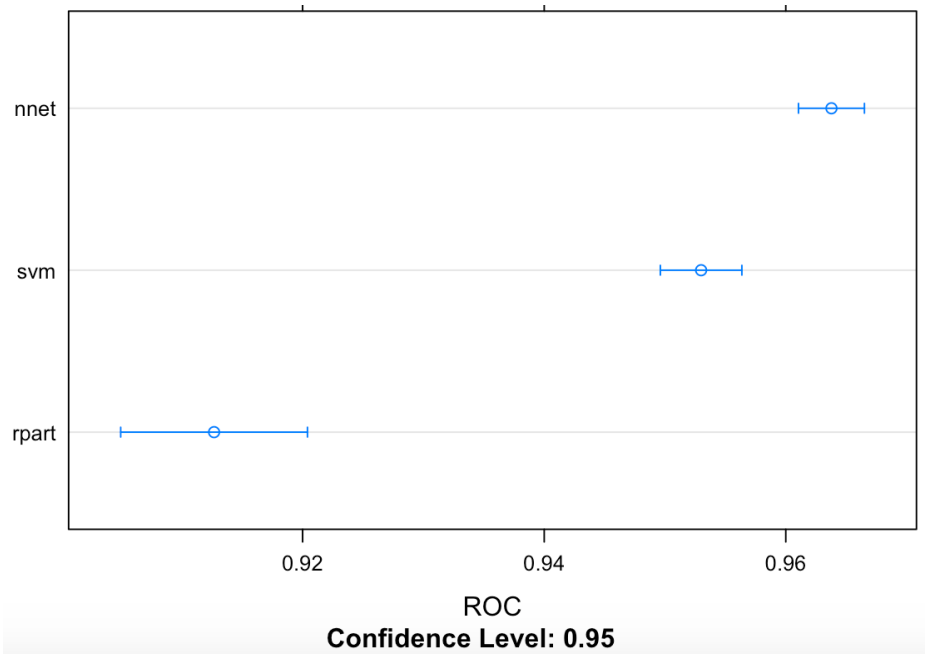
```
Models: svm, rpart, nnet
Number of resamples: 100

ROC
           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
svm   0.9120766 0.9419108 0.9527027 0.9529893 0.9642996 0.9887102    0
rpart 0.8173077 0.8854109 0.9161557 0.9126712 0.9419689 0.9793021    0
nnet  0.9315771 0.9537422 0.9628806 0.9637785 0.9757954 0.9928775    0

Sens
           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
svm   0.8354430 0.8974359 0.9230769 0.9165514 0.9367089 0.9873418    0
rpart 0.7974684 0.8846154 0.9113924 0.9069880 0.9367089 0.9871795    0
nnet  0.8461538 0.9113924 0.9358974 0.9291253 0.9487179 1.0000000    0

Spec
           Min.   1st Qu.    Median      Mean   3rd Qu.     Max. NA's
svm   0.6486486 0.7822823 0.8108108 0.8143844 0.8611111 0.972973    0
rpart 0.5675676 0.7725225 0.8108108 0.8120420 0.8611111 0.972973    0
nnet  0.7297297 0.8108108 0.8494745 0.8489414 0.8918919 1.000000    0
```

Il grafico seguente permette di comparare le distrubuzioni, in particolare è possibile notare come la ROC della rete neurale sia quella che varia meno rispetto alle altre, oltre che avvicinarsi di più verso il valore massimo.

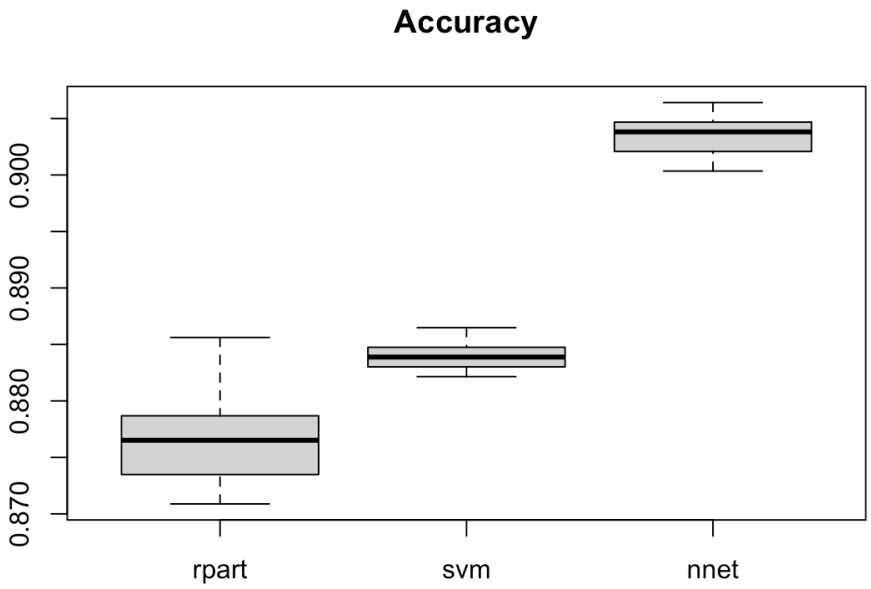
Per i valori di Sensibilità e Specificità, al contrario, le distribuzioni sono molto sovrapposte, ma anche in questo caso la rete neurale è quella meglio rappresentata.



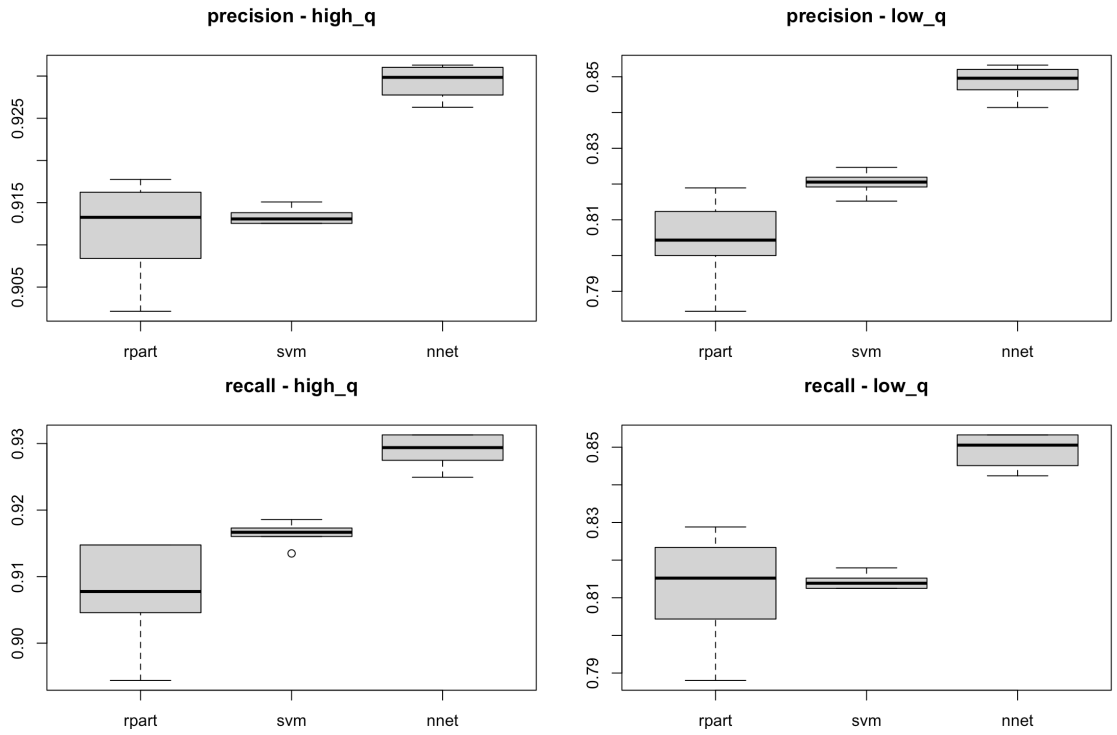
La visualizzazione degli intervalli di confidenza permette di osservare come essi non si sovrappongano, questa situazione è ideale in quanto evidenzia come la rete neurale abbia l'intervallo di confidenza più piccolo e più vicino al valore 1.



Per quanto riguarda l'**accuracy**, possiamo notare che nnet presenta in assoluto un miglior profilo rispetto a svm e rpart. Tra rpart e svm abbiamo performance comparabili, ma svm presenta una minore variabilità.



Anche per precision e recall abbiamo risultati simili a quelli discussi per l'accuracy. In generale per l'utilizzo che intendiamo fare di questo modello diamo una maggiore importanza alla metrica di precision.



Vengono poi confrontati i tempi di addestramento e di predizione dei modelli. E' possibile notare come il modello degli alberi di decisione sia il più veloce, mentre quello della rete neurale sia molto più lento. In fase di predizione, invece, tutti i modelli rispondono in tempi trascurabili.

Model	Everything	Final Model	Prediction
nnet3	13.80	0.14	NA
svm05	6.95	0.05	NA
rpart1	2.83	0.00	NA

7.1 Conclusioni

In conclusione, il modello che presenta in assoluto **le migliori performance** è **nnet**. Il modello **svm** risulta **più equilibrato** tra performance e costi computazionali, mentre **il costo computazionale basso** dell'**albero di decisione** non giustifica a pieno la sua forte varianza nei risultati.

Nell'utilizzo **in questo particolare contesto** optiamo per il modello **nnet**, in quanto ha costi computazionali accettabili e migliori prestazioni.
Nell'ipotesi, invece, di un'**estensione del dominio** (ad esempio se esteso il numero di indici, se considerate immagini a colori o nel caso di aumento delle istanze) si potrebbe rivalutare questa scelta in favore di **svm**, che comunque presenta delle ottime performance, ma con costi decisamente ridotti.