RELAZIONE MACHINE LEARNING:

Descrizione dominio di riferimento:

Il dataset di riferimento (*https://archive.ics.uci.edu/ml/datasets/Wine+Quality*) rappresenta una raccolta di misurazioni chimico fisiche inerenti alla composizione di diversi campioni di vino, e i rispettivi giudizi espressi soggettivamente con un voto da 3 a 9.
Ci poniamo come obiettivo principale quello di identificare i vini qualitativamente buoni per essere commercializzati, stabilendo eventuali correlazioni tra giudizi e misurazioni.

Descrizione di tutti gli attributi: (lista con spiegazione)

Attributes	Description
Fixed Acidity	Fixed acids, numeric from 3.8 to 15.9
Volatile Acidity	Volatile acids, numeric from 0.1 to 1.6
Citric Acid	Citric acids, numeric from 0.0 to 1.7
Residual Sugar	residual sugar, numeric from 0.6 to 65.8
Chlorides	Chloride, numeric from 0.01 to 0.61
Free Sulfur Dioxide	Free sulfur dioxide, numeric: from 1 to 289
Total Sulfur Dioxide	Total sulfur dioxide, numeric: from 6 to 440
Density	Density, numeric: from 0.987 to 1.039
PH	pH, numeric: from 2.7 to 4.0
Sulfates	Sulfates, numeric: from 0.2 to 2.0
Alcohol	Alcohol, numeric: from 8.0 to 14.9
Quality	Quality, numeric: from 0 to 10, the output target

Scelte di design:

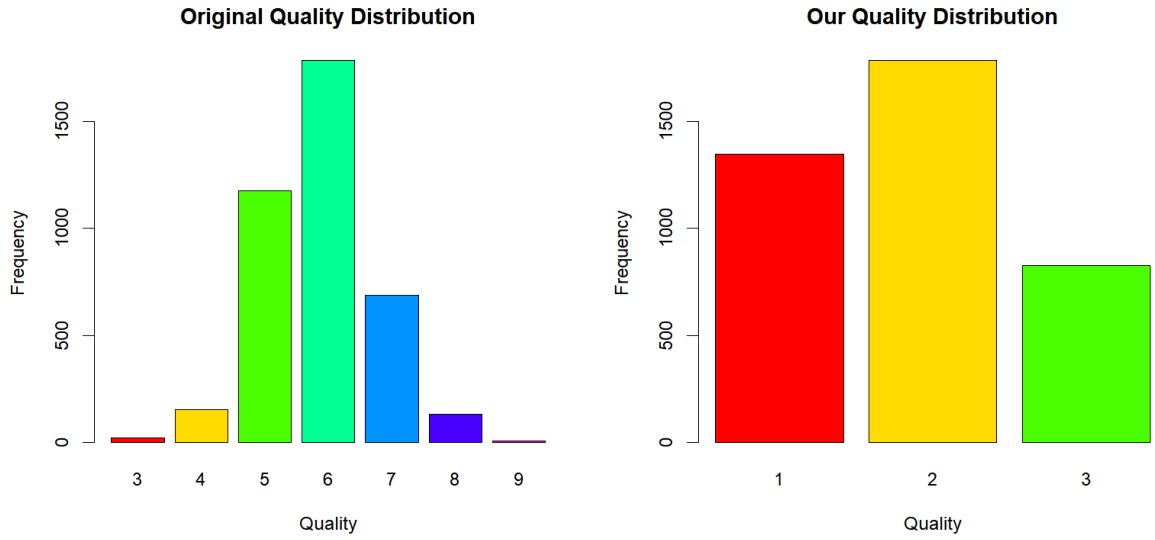
Dopo aver effettuato un'attenta analisi del dataset e formulato le dovute considerazioni, abbiamo applicato una serie di modifiche atte alla preparazione del dataset.
La tabella seguente riporta i valori ottenuti dall'istruzione *'summary(winequality.white)'*

		FixedAcidity	**VolatileAcidity**	**CitricAcid**	**ResidualSugar**
Min.	**:**	3.800	0.0800	0.0000	0.600
1st Qu.	**:**	6.300	0.2100	0.2700	1.700
Median	**:**	6.800	0.2600	0.3200	5.200
Mean	**:**	6.855	0.2782	0.3342	6.391
3rd Qu.	**:**	7.300	0.3200	0.3900	9.900
Max.	**:**	14.200	1.1000	1.6600	65.800
		Chlorides	**FreeSulfurDioxide**	**TotalSulfurDioxide**	**Density**
Min.	**:**	0.00900	2.00	9.0	0.9871
1st Qu.	**:**	0.03600	23.00	108.0	0.9917
Median	**:**	0.04300	34.00	134.0	0.9937
Mean	**:**	0.04577	35.31	138.4	0.9940
3rd Qu.	**:**	0.05000	46.00	167.0	0.9961
Max.	**:**	0.34600	289.00	440.0	1.0390
		PH	**Sulphates**	**Alcohol**	**Quality**
Min.	**:**	2.720	0.2200	8.00	3.000
1st Qu.	**:**	3.090	0.4100	9.50	5.000
Median	**:**	3.180	0.4700	10.40	6.000
Mean	**:**	3.188	0.4898	10.51	5.878
3rd Qu.	**:**	3.280	0.5500	11.40	6.000
Max.	**:**	3.820	1.0800	14.20	9.000

Innanzitutto, abbiamo riscontrato che la classe Quality fosse sbilanciata (come mostrato nell'immagine seguente). Il problema dello sbilanciamento tra le classi comporta che l'algoritmo tenderà a favorire le classi con più elementi, ovvero quelle più preponderanti, a discapito delle classi con pochi elementi rispetto alle quali non riuscirebbe ad apprendere alcuna informazione.

Di conseguenza abbiamo optato per un approccio a tre classi, mappando rispettivamente le classi originali in nuove classi come segue:

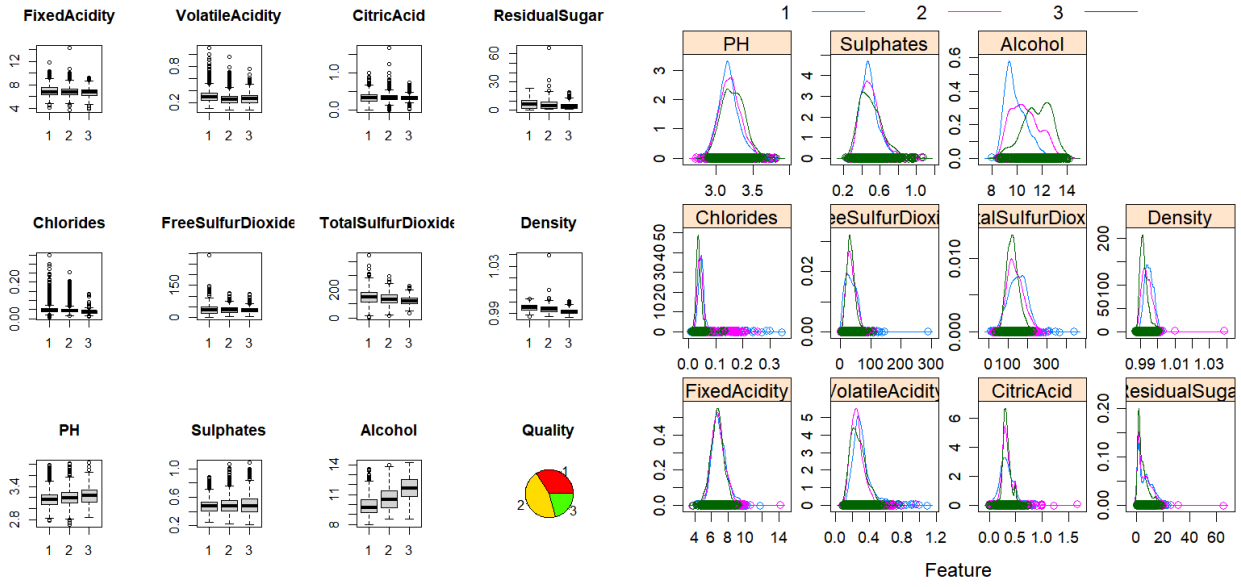
Nuove classi:		Classi originali:
Q1 (non commerciabile)	←	3 - 4 - 5
Q2 (fascia B)	←	6
Q3 (fascia A)	←	7 - 8 – 9



A seguito di questa operazione di mapping possiamo notare come il problema del bilanciamento non venga risolto, ma in confronto alla suddivisione iniziale, siamo certamente in una situazione migliore.
(Nota: In letteratura esistono diversi metodi che permettono di ovviare al problema dello sbilanciamento delle classi, ma che non sono stati presi in considerazione in quanto non presentati in questo corso.)

La classe Quality, che nel dataset di partenza risultava di tipo numerico, è stata convertita in categorico in modo da evitare qualsiasi eventuale errore dovuto al tipo di dato. In merito alla gestione di elementi duplicati abbiamo proceduto alla rimozione degli stessi; non sono invece presenti istanze con attributi mancanti. Oltre alle correzioni appena descritte non sono state ritenute necessarie ulteriori modifiche al dataset.

Descrizione del training set: analisi esplorativa del training set (analisi delle covariate e/o PCA)



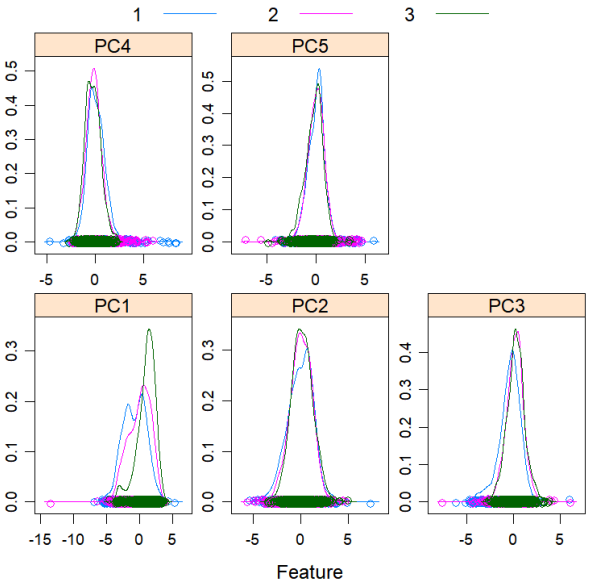
Dalle immagini sopra riportate si evince come gli attributi non siano distribuiti secondo una normale. Infatti, nel caso della figura a sinistra, si può notare come i boxplot non siano simmetrici e siano presenti dei valori esterni al range atteso. Analogamente, dal plot della densità di probabilità (immagine a destra), si evince ulteriormente come la maggior parte degli attributi non rispecchino la tipica distribuzione normale.
Siccome il dataset presenta un elevato numero di features (tutte numeriche reali), abbiamo deciso di applicare la tecnica della PCA al fine di ridurne la dimensionalità preservando la massima quantità di informazioni.
Per prima cosa sono state standardizzate le features, e in seguito è stata applicata la PCA utilizzando la funzione *prcomp()* [built-in R stats package]. Mediante la funzione *get_eigenvalue(prin_comp)* vengono estratti gli autovalori che misurano la varianza di ciascuna componente principale. Le prime componenti principali corrispondono alle direzioni con la massima quantità di variazione nel dataset.

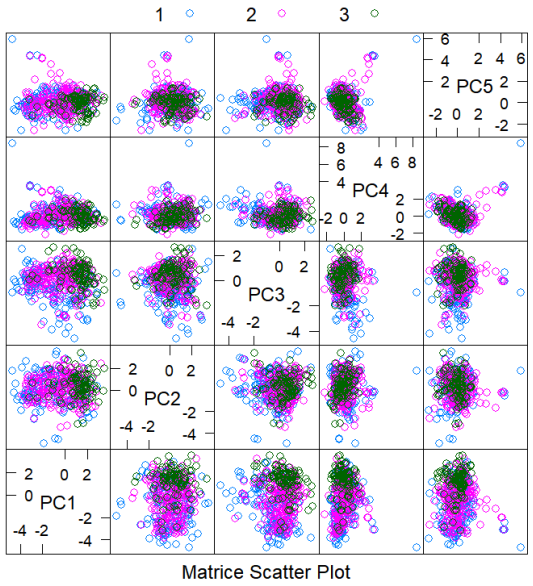
Utilizzando gli autovalori per stabilire il numero delle componenti principali definiamo che:

- Vengono mantenute le 4 PC con gli autovalori > 1, indice che le PC rappresentano una varianza maggiore di quella calcolata, ed essendo i dati precedentemente standardizzati viene utilizzata come soglia discriminante;
- Considerato il fatto che, con i soli autovalori > 1, non si raggiunge il limite di 70% di varianza totale cumulativa è stato deciso di mantenere anche le componenti principali necessari a raggiungerla.

	Eigenvalue	Variance.percent	Cumulative.variance.percent
Dim 1	3.18353626	28.9412387	28.94124
Dim 2	1.59646600	14.5133273	43.45457
Dim 3	1.21304637	11.0276943	54.48226
Dim 4	1.03904742	9.4458856	63.92815
Dim 5	0.98113079	8.9193709	72.84752
Dim 6	0.92929956	8.4481779	81.29569
Dim 7	0.72204707	6.5640643	87.85976
Dim 8	0.60204013	5.4730921	93.33285
Dim 9	0.42883417	3.8984925	97.23134
Dim 10	0.28090551	2.5536864	99.78503
Dim 11	0.02364671	0.2149701	100.00000

In particolare, come è possibile constatare dalla tabella precedente, vengono mantenute le prime 5 componenti principali che permettono di raggiungere una varianza cumulativa del 72%.





Il grafico relativo alla matrice scatter plot è stato calcolato su un insieme di 500 elementi solo per esigenze grafiche e rendere il risultato comprensibile.
Dopo aver applicato la PCA non si notano attributi preponderanti al fine della classificazione.

Motivazione modelli:

In merito alla scelta dei modelli da adottare, abbiamo tenuto conto dei seguenti aspetti:

- Alberi di decisione: abbiamo valutato che questo modello fosse maggiormente efficiente con attributi categorici anziché continui, e quindi poco efficace rispetto al dataset in questione;
- Naive bayes: non è possibile supporre l'indipendenza condizionale tra gli attributi (anche in questo caso funziona meglio con attributi di tipo categorico);

Nel presente progetto, abbiamo scelto infine di utilizzare i seguenti modelli:

- ✓ SVM: con kernel radiale,
- ✓ Rete neurale: feed forward;

in quanto abbiamo valutato essere i più indicati rispetto al dataset in questione.
Infatti, avere attributi continui favorisce la scelta dell'SVM dato che nasce come una specifica del percettrone e lavora sullo spazio R-dimensionale. Allo stesso tempo la rete neurale viene utilizzata per la sua efficienza e versatilità, vista anche l'elevata quantità di dati che vengono trattati.

Esperimenti:

Il dataset di partenza è stato suddiviso in '*train set*' e '*test set*' secondo la divisione classica presente in letteratura, ovvero in 70% train set e 30% test set.

Avendo un approccio multi-classe abbiamo eseguito il training con una cross-validation a 10 fold, ripetuta 3 volte e utilizzando la raccolta dati della classificazione multipla non binaria.

```
trainControl(   method = "repeatedcv",
                number = 10,
                repeats = 3,
                classProbs = TRUE,
                summaryFunction = multiClassSummary)
```

La cross validation ci permette di settare gli iperparametri sui modelli migliori appena calcolati:

- Rete neurale:
  - layer_nascosti = 1;
  - neuroni_layer_nascosti = 5.
- SVM:
  - C = 0.5;
  - $\sigma \approx 0.22$.

rete = train(Quality ~., data = trainset, method="nnet", metric="AUC", trace = FALSE, nmax=10000, trControl = control)

svm2 = train(Quality ~ ., data = trainset, method = "svmRadial", metric="AUC", trControl = control)

resamples(list(net = rete, svm = svm2))

Models: net, svm

Number of resamples: 30

AUC	**Min.**	**1st Qu.**	**Median**	**Mean**	**3rd Qu.**	**Max.**
net	0.6873849	0.7204329	0.7363465	0.7354494	0.7521273	0.7793609
svm	0.6862249	0.7249118	0.7395492	0.7378477	0.7510064	0.7780287

Accuracy	**Min.**	**1st Qu.**	**Median**	**Mean**	**3rd Qu.**	**Max.**
net	0.4964286	0.5380472	0.5688748	0.5638433	0.5908698	0.6250000
svm	0.5178571	0.5575369	0.5678571	0.5696726	0.5842294	0.5842294

A questo punto calcoliamo la matrice di confusione che restituisce una rappresentazione dell'accuratezza di classificazione statistica, con i valori predetti rappresentati sulle righe e quelli reali sulle colonne. In particolare, possiamo notare che lungo la diagonale principale vengano riportati i valori delle istanze classificate correttamente, mentre nelle altre celle le istanze che vengono classificate in modo errato.

Rete	Q1	Q2	Q3
Q1	226	127	16
Q2	147	339	167
Q3	4	59	79

Overall Statistics:

Accuracy : 0.5533

95% CI : (0.5242, 0.5821)

Statistics by Class:

Rete	Q1	Q2	Q3
Precision	0.6125	0.5191	0.55634
Recall	0.5995	0.6457	0.30153
F1-measure	0.6059	0.5756	0.39109

SVM	Q1	Q2	Q3
Q1	226	107	16
Q2	148	386	173
Q3	3	32	73

Overall Statistics:

Accuracy : 0.5885

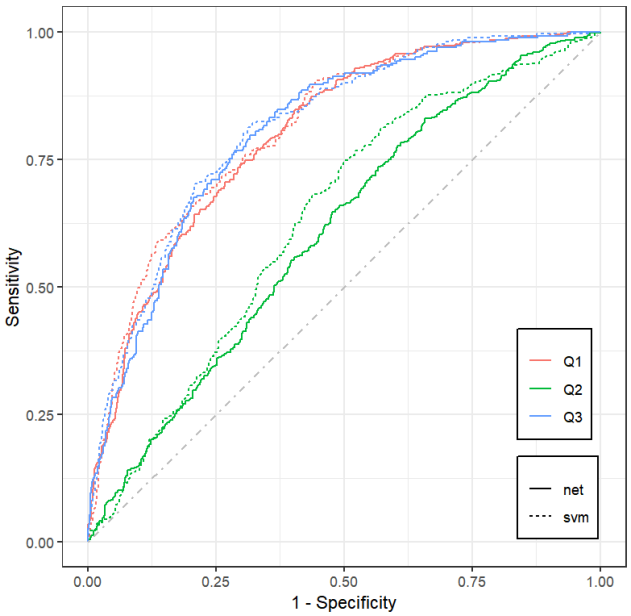
95% CI : (0.5596, 0.6169)

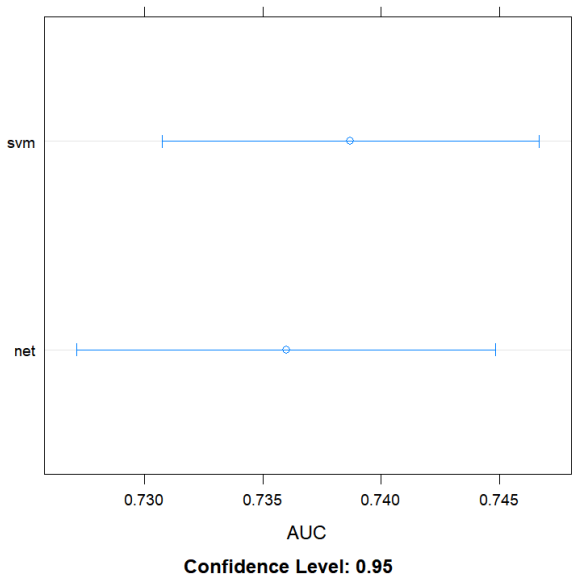
Statistics by Class:

SVM	Q1	Q2	Q3
Precision	0.6476	0.5460	0.67593
Recall	0.5995	0.7352	0.27863
F1-measure	0.6226	0.6266	0.39459

Grafico curve ROC e AUC:

Il grafico sottostante rappresenta le curve ROC (Receiver Operating Characteristic), ovvero un grafico che illustra le prestazioni di un sistema di classificazione e traccia il tasso vero positivo contro il tasso di falsi positivi per diversi punti di taglio. Per effettuare il confronto tra le prestazioni dei due modelli calcoliamo la misura di AUC (Area Under Curve) e quindi proseguiamo con il calcolo dell'intervallo di confidenza.





AUC	SVM	Rete neurale
Q1	0.8087658	0.8078693
Q2	0.6330546	0.611366
Q3	0.8118304	0.8048569

Analisi dei risultati:

Dai grafici sopra riportati si può asserire che la SVM sia generalmente un approccio migliore rispetto alla rete neurale, in quanto restituisce un modello più preciso e con intervallo di confidenza migliore. A conferma di ciò, possiamo vedere come l'AUC dell'SVM sia migliore rispetto a quello relativo alla rete neurale. Inoltre, osservando l'intervallo di confidenza è possibile notare come il modello SVM abbia una performance media più elevata ma che allo stesso tempo sia più robusto e con meno variabilità.

Misure di performance:

L'accuratezza è una misura di performance globale, e che quindi tende ad essere distorta rispetto al numero di elementi che appartengono alla classe predominante.

- Accuracy = $\frac{TP + TN}{TP + TN + FP + FN}$

Sfruttiamo quindi la matrice di confusione per calcolare delle ulteriori misure di performance che possono essere regolate rispetto alla dimensionalità di classi particolarmente rumorose:

- Precision = $\frac{TP}{TP + FP}$
- Recall = $\frac{TP}{TP + FN}$
- F-measure = $\frac{2 * Precision * Recall}{Precision + Recall}$

La precisione è una misura che ci indica quanto siamo abili nel riconoscere gli elementi di una fissata classe. La recall invece, data la totalità delle istanze di una determinata classe, indica il numero di elementi che si riesce a calcolare correttamente. Un buon modello di solito tende a massimizzare entrambe le metriche, ed è possibile calcolare l'F1-measure che è una misura aggregata e definita come media armonica tra la precision e la recall.

In generale, dunque, alti valori di F1-measure sono un buon indice da considerare per tutte le classi, poiché rispecchiano buoni valori di precision e recall. I due modelli hanno misure simili, con quelle della SVM leggermente superiori.

In ottica di marketing, nel caso del nostro progetto, lo scopo principale è non commercializzare vini scadenti. Quindi se fissassimo Q1 come classe positiva (vini non commercializzabili), i valori da tenere in considerazione sono quelli di recall: infatti vogliamo evitare soprattutto di mettere in commercio vini scadenti (falsi negativi). Per la suddetta classe, abbiamo in entrambi i modelli anche simili valori di precision, cioè di non rischiare di non vendere un vino in realtà commerciabile (falsi positivi); per questo motivo, i valori di F1-measure si discostano poco dalle altre due misure.

Al contrario, per la classe Q3, vogliamo principalmente evitare di commercializzare come vini di fascia A quelli che non lo siano: in questo caso, i valori da tenere come riferimento sono in primo luogo quelli di precision. Ulteriore motivo per cui scegliere la SVM, dunque, è la maggiore precisione sulla classe Q3 rispetto alla rete.

Conclusioni:

I risultati ottenuti non sono del tutto soddisfacenti, anche a causa del problema dello sbilanciamento. Da non trascurare la natura soggettiva del giudizio sulla qualità, che difficilmente potrà essere completamente spiegato dalle sole misurazioni oggettive.

Degno di nota, il fatto che la maggior parte delle istanze di categoria 1 e 3 classificate erroneamente sono in realtà individuate di categoria 2: in entrambi i modelli, infatti, i vini di classe 1 scambiati durante la fase di test per vini di classe 3 (e viceversa) sono in numero piuttosto esiguo. Perciò, si può dedurre che gli algoritmi abbiano trovato una soluzione robusta in grado di distinguere i vini di qualità alta da quelli di qualità bassa.