"Vinho Verde" Wines Quality Modeling

by Viviane Adohouannon, Kate Alexander, Diana Azbel, Igor Baranov

Abstract Wine classification is a difficult task since taste is the least understood of the human senses. In this research we propose a data mining approach to predict human wine taste preferences that is based on easily available analytical tests at the wine certification step. We are using a dataset related to red and white Vinho Verde wine samples, from the north of Portugal. The goal is to model wine quality based on physicochemical tests. The classes are not balanced, there are much more normal wines than excellent or poor ones. The methods used to solve the problem are Random Forests, Clustering, Neural Networks and Support Vector Machine. Resuts of those methods applicaion are compared and analyzed.

Introduction

Data mining techniques aim at extracting knowledge from raw data. Several DM algorithms have been developed, each one with its own advantages and disadvantages (Witten et al., 2011). Those approaches have been applied to a large variety of problems, either for classification or regression. An interesting problem that has captured the attention of several researches is the prediction of wine quality (Cortez et al., 1998). Wine industry is investing in new technologies for wine making and selling processes. A key issue in this context is wine certification which prevents the illegal adulteration and assures the wine quality. Wine certification is often assessed by physicochemical and sensory tests. The development of an accurate, computationally efficient and understandable prediction model can be of great utility for the wine industry. On the one hand, a good wine quality prediction can be very useful in the certification phase, since currently the sensory analysis is performed by human tasters, being clearly a subjective approach. An automatic predictive system can be integrated into a decision support system, helping the speed and quality of the oenologist performance. If it is concluded that several input variables are highly relevant to predict the wine quality, since in the production process some variables can be controlled, this information can be used to improve the wine quality. In this research wine taste preferences are modelled by algorithms.

Background

Portugal is a top ten wine exporting country with 3.17% of the market share in 2005 (FAOSTAT). Exports of its Vinho Verde wine (from the northwest region) have increased by yearly. To support its growth, the wine industry is investing in new technologies for both wine making and selling processes. Wine certification and quality assessment are key elements within this context. Certification prevents the illegal adulteration of wines (to safeguard human health) and assures quality for the wine market. Quality evaluation is often part of the certification process and can be used to improve wine making (by identifying the most influential factors) and to stratify wines such as premium brands (useful for setting prices). Wine certification is generally assessed by physicochemical and sensory tests (Teranishi et al., 1999). Physicochemical laboratory tests routinely used to characterize wine include determination of density, alcohol or pH values, while sensory tests rely mainly on human experts. It should be stressed that taste is the least understood of the human senses, thus wine classification is a difficult task. Moreover, the relationships between the physicochemical and sensory analysis are complex and still not fully understood (Legin et al., 2003).

Objective

The objective of this project is is to provide a reliable and feasible recommendation algorithm to predict wine quality based on physicochemical tests. The target value is a numeric value of wine 'quality', hence the task could be solved by several regression methods, like Random Forest, Support Vector Machines and Neural Networks. In addition, it was decided to apply Clustering Analysis to investigate if we could predict the quality better or if we could get any new knowledge from the dataset.

Data understanding

The two datasets presented in (UCI Wine Data Set) are related to red and white variants of the Portuguese "Vinho Verde" wine. For more details, consult (Cortez et al., 1998). Due to privacy and logistic issues, only physicochemical (inputs) and sensory (the output) variables are available (e.g. there is no data about grape types, wine brand, wine selling price, etc.). The classes are ordered and not balanced (e.g. there are much more normal wines than excellent or poor ones).

```
The dataset contains 6497 observations:

Input variables (based on physicochemical tests):
  1 - fixed acidity        (FA)
  2 - volatile acidity     (VA)
  3 - citric acid          (CA)
  4 - residual sugar       (RS)
  5 - chlorides            (CH)
  6 - free sulfur dioxide  (FSD)
  7 - total sulfur dioxide (TSD)
  8 - density              (DEN)
  9 - pH                   (pH)
  10 - sulphates           (SUL)
  11 - alcohol             (ALC)

Output variable (based on sensory data):
  12 - quality, based on sensory data (score between 0 and 10) - (QLT)
  13 - wine type (0 - red whine, 1 - white wine)
```

Let's try to make sence of those attributes. There are many so called "impact compounds" in wine defining its taste and smell. Some of them are obvious, leke ALC, DEN and RS. Some of them are really bizarre (noa, 2017). The chemical components mentioned in the list have the following influence on the wine taste.

First is "Type" - red wine is different from white wine (Busch, 2011). They look different and they certainly taste different as well. The culprit in both cases: the skins, and a little something they bring to the party called tannins. Tannin provides the backbone of red wine, which is why you might describe a red wine as "firm" or "leathery" or just plain "bitter." White wine has tannin, but not enough to make it the star of the show. Instead, white wines are backboned by acidity. That's why you might say a wine is "crisp" or "tart."

Then there is volatile acidity (VA) that intensifies the taste of the other acids and tannins (Corison et al., 1979). As the name suggests it is referencing volatility in wine, which causes it to go bad. Acetic acid builds up in wine when there's too much exposure to oxygen during winemaking and is usually caused by acetobacter (the vinegar-making bacteria!). VA is considered a fault at higher levels (1.4 g/L in red and 1.2 g/L in white) and can smell sharp like nail polish remover. But at lower levels, it can add fruity-smelling raspberry, passion fruit, or cherry-like flavors

Sulphur dioxide (SO2) is used to inhibit or kill unwanted yeasts and bacteria, and to protect wine from oxidation. Important concentrations of SO2 can affect the smell of the wine. It is also most-often noted on the finish, with some wines displaying a strong flavor of Sulphur after you've tasted (or swallowed) on the back of the mouth. Red wine contains less Sulphur Dioxide than white and rose as the above regulations show. Generally speaking, the drier the wine, the lesser the amount of SO2 it contains. (noa).

In wine tasting, the general term "acidity" defined by pH, FA and CA, refers to the fresh, tart and sour attributes of the wine which are evaluated in relation to how well the acidity balances out the sweetness and bitter components of the wine such as tannins.

Sulfates (SUL) aren't involved in wine production, but some beer makers use calcium sulfate—also known as brewers' gypsum—to correct mineral deficiencies in water during the brewing process. Sulfites are naturally occurring compounds found in all wines; they act as a preservative by inhibiting microbial growth.

The amount of CH in wine is influenced by both the terroir and type of grape (Coli et al., 2015), and the importance of quantification lies in the fact that wine flavor is strongly impacted by this particular ion, which, in high concentration, gives the wine an undesirable salty taste.

Data Preparation

To perform the analysis, certain R libraries were used. The code below was used to load and initialize the librarie, then loads the data. To pretty-print the tables in this report we used xtable (Dahl, 2016) library.

```
set.seed(42)
library(ggplot2)
library(reshape2)
library(plyr)
library(readr)
library(fpc)
library(data.table)
library(ggplot2)

wines_red_data <-
  read.csv(
    "http://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-red.csv",
    sep=";",
    header = TRUE,
    col.names = c("FA","VA","CA","RS","CH","FSD","TSD","DEN","pH","SUL","ALC","QLT"))

wines_red_data$TYPE <- 0

wines_white_data <-
  read.csv(
    "http://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-white.csv",
    sep=";",
    header = TRUE,
    col.names = c("FA","VA","CA","RS","CH","FSD","TSD","DEN","pH","SUL","ALC","QLT"))

wines_white_data$TYPE <- 1

wines_data <- rbind(wines_red_data, wines_white_data)
```

Preview of the data

Quick view of the data attributes statistics presented in the Table 2. For each attribute in the dataset this table shows min, max, mean and normal distribution 1st and 3rd quartiles values. The first rows of the dataset are presented in Table 1.

	FA	VA	CA	RS	CH	FSD	TSD	DEN	pH	SUL	ALC	QLT	TYPE
1	7.40	0.70	0.00	1.90	0.08	11.00	34.00	1.00	3.51	0.56	9.40	5	0.00
2	7.80	0.88	0.00	2.60	0.10	25.00	67.00	1.00	3.20	0.68	9.80	5	0.00
3	7.80	0.76	0.04	2.30	0.09	15.00	54.00	1.00	3.26	0.65	9.80	5	0.00
4	11.20	0.28	0.56	1.90	0.07	17.00	60.00	1.00	3.16	0.58	9.80	6	0.00
5	7.40	0.70	0.00	1.90	0.08	11.00	34.00	1.00	3.51	0.56	9.40	5	0.00
6	7.40	0.66	0.00	1.80	0.07	13.00	40.00	1.00	3.51	0.56	9.40	5	0.00
7	7.90	0.60	0.06	1.60	0.07	15.00	59.00	1.00	3.30	0.46	9.40	5	0.00
8	7.30	0.65	0.00	1.20	0.06	15.00	21.00	0.99	3.39	0.47	10.00	7	0.00
9	7.80	0.58	0.02	2.00	0.07	9.00	18.00	1.00	3.36	0.57	9.50	7	0.00
10	7.50	0.50	0.36	6.10	0.07	17.00	102.00	1.00	3.35	0.80	10.50	5	0.00
11	6.70	0.58	0.08	1.80	0.10	15.00	65.00	1.00	3.28	0.54	9.20	5	0.00
12	7.50	0.50	0.36	6.10	0.07	17.00	102.00	1.00	3.35	0.80	10.50	5	0.00
13	5.60	0.61	0.00	1.60	0.09	16.00	59.00	0.99	3.58	0.52	9.90	5	0.00
14	7.80	0.61	0.29	1.60	0.11	9.00	29.00	1.00	3.26	1.56	9.10	5	0.00
15	8.90	0.62	0.18	3.80	0.18	52.00	145.00	1.00	3.16	0.88	9.20	5	0.00
16	8.90	0.62	0.19	3.90	0.17	51.00	148.00	1.00	3.17	0.93	9.20	5	0.00
17	8.50	0.28	0.56	1.80	0.09	35.00	103.00	1.00	3.30	0.75	10.50	7	0.00
18	8.10	0.56	0.28	1.70	0.37	16.00	56.00	1.00	3.11	1.28	9.30	5	0.00
19	7.40	0.59	0.08	4.40	0.09	6.00	29.00	1.00	3.38	0.50	9.00	4	0.00
20	7.90	0.32	0.51	1.80	0.34	17.00	56.00	1.00	3.04	1.08	9.20	6	0.00

Table 1: `Red Wines Quality Dataset - first rows`

FA	VA	CA	RS	CH	FSD
Min. : 3.800	Min. :0.0800	Min. :0.0000	Min. : 0.600	Min. :0.00900	Min. : 1.00
1st Qu.: 6.400	1st Qu.:0.2300	1st Qu.:0.2500	1st Qu.: 1.800	1st Qu.:0.03800	1st Qu.: 17.00
Median : 7.000	Median :0.2900	Median :0.3100	Median : 3.000	Median :0.04700	Median : 29.00
Mean : 7.215	Mean :0.3397	Mean :0.3186	Mean : 5.443	Mean :0.05603	Mean : 30.53
3rd Qu.: 7.700	3rd Qu.:0.4000	3rd Qu.:0.3900	3rd Qu.: 8.100	3rd Qu.:0.06500	3rd Qu.: 41.00
Max. :15.900	Max. :1.5800	Max. :1.6600	Max. :65.800	Max. :0.61100	Max. :289.00

TSD	DEN	pH	SUL	ALC	QLT
Min. : 6.0	Min. :0.9871	Min. :2.720	Min. :0.2200	Min. : 8.00	Min. :3.000
1st Qu.: 77.0	1st Qu.:0.9923	1st Qu.:3.110	1st Qu.:0.4300	1st Qu.: 9.50	1st Qu.:5.000
Median :118.0	Median :0.9949	Median :3.210	Median :0.5100	Median :10.30	Median :6.000
Mean :115.7	Mean :0.9947	Mean :3.219	Mean :0.5313	Mean :10.49	Mean :5.818
3rd Qu.:156.0	3rd Qu.:0.9970	3rd Qu.:3.320	3rd Qu.:0.6000	3rd Qu.:11.30	3rd Qu.:6.000
Max. :440.0	Max. :1.0390	Max. :4.010	Max. :2.0000	Max. :14.90	Max. :9.000

Table 2: `Wines Dataset Attributes Summary`

Check for missing values

The dataset has no missing values. Code below calculate number of rows with missing values and checks if there is at list one.

```
any(is.na(wines_data))
```

```
#> [1] FALSE
```

Distribution of target value in the dataset

As we mentioned before, the target value QLT of the wine quality is not equally distributed. The Figure 1 demonstrates the distribution. As we can see, dataset covers mostly medium-quality wines with QLT between 5 and 7 well, low and high quality wines represented poorly.

```
prop.table(table(wines_data$QLT))
```

```
#>
#>           3           4           5           6           7           8
#> 0.004617516 0.033246114 0.329074958 0.436509158 0.166076651 0.029706018
#>           9
#> 0.000769586
```

```
ggplot(data = wines_data, mapping = aes(x = QLT)) + geom_bar()
```

Splitting the wine into train and test sets

The wines dataset has been split in such a way that train and test sets would have the same distribution of the QLT attribute. We used 70:30 split ratio.

```
library(caret)
cluster1 <- wines_data[,1:12]
train1.rows<- createDataPartition(y= cluster1$QLT, p=0.7, list = FALSE)
train1.data<- cluster1[train1.rows,]
prop.table((table(train1.data$QLT)))
```

```
#>
#>            3            4            5            6            7
#> 0.0041767421 0.0358320510 0.3268850297 0.4365794680 0.1668498571
#>            8            9
#> 0.0287975379 0.0008793141
```

```
test1.data<- cluster1[-train1.rows,]
prop.table((table(test1.data$QLT)))
```

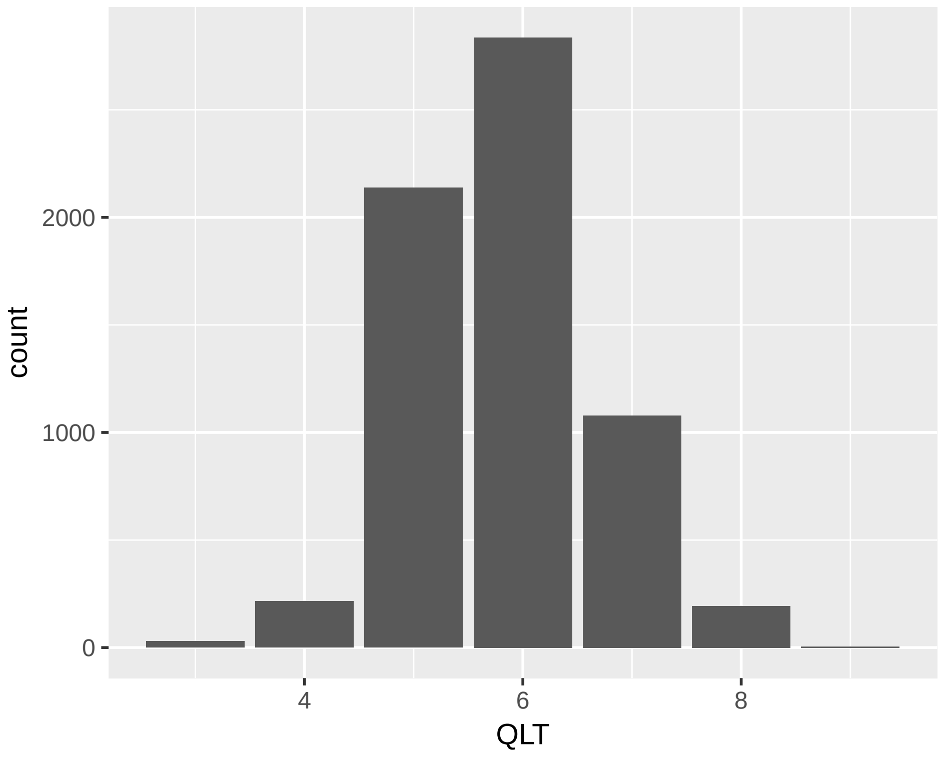


Figure 1: Distribution of Wine Quality Attribute

```
#>
#>           3           4           5           6           7           8
#> 0.005646817 0.027207392 0.334188912 0.436344969 0.164271047 0.031827515
#>           9
#> 0.000513347
```

Modelling of the Wine Quality

Random Forests Prediction

First we use advanced but computationally demanding Random Forest method (noa, 2018). Decision trees are a popular method for various machine learning tasks. Tree learning "come[s] closest to meeting the requirements for serving as an off-the-shelf procedure for data mining because it is invariant under scaling and various other transformations of feature values, is robust to inclusion of irrelevant features, and produces inspectable models. However, they are seldom accurate". In particular, trees that are grown very deep tend to learn highly irregular patterns: they overfit their training sets, i.e. have low bias, but very high variance. Random forests are a way of averaging multiple deep decision trees, trained on different parts of the same training set, with the goal of reducing the variance. This comes at the expense of a small increase in the bias and some loss of interpretability, but generally greatly boosts the performance in the final model.

```
library(randomForest)
fitRF1 <- randomForest(
  QLT ~ ., method="anova",
  data=train1.data, importance=TRUE, ntree=500)
```

Random forests can be used to rank the importance of variables in a regression or classification problem in a natural way. To measure the importance of the j-th feature after training, the values of the j-th feature are permuted among the training data and the out-of-bag error is again computed on this perturbed data set. The importance score for thej-th feature is computed by averaging the difference in out-of-bag error before and after the permutation over all trees. The score is normalized by the standard deviation of these differences. Features which produce large values for this score are ranked as more important than features which produce small values. The Figure 2 presents this analysi.

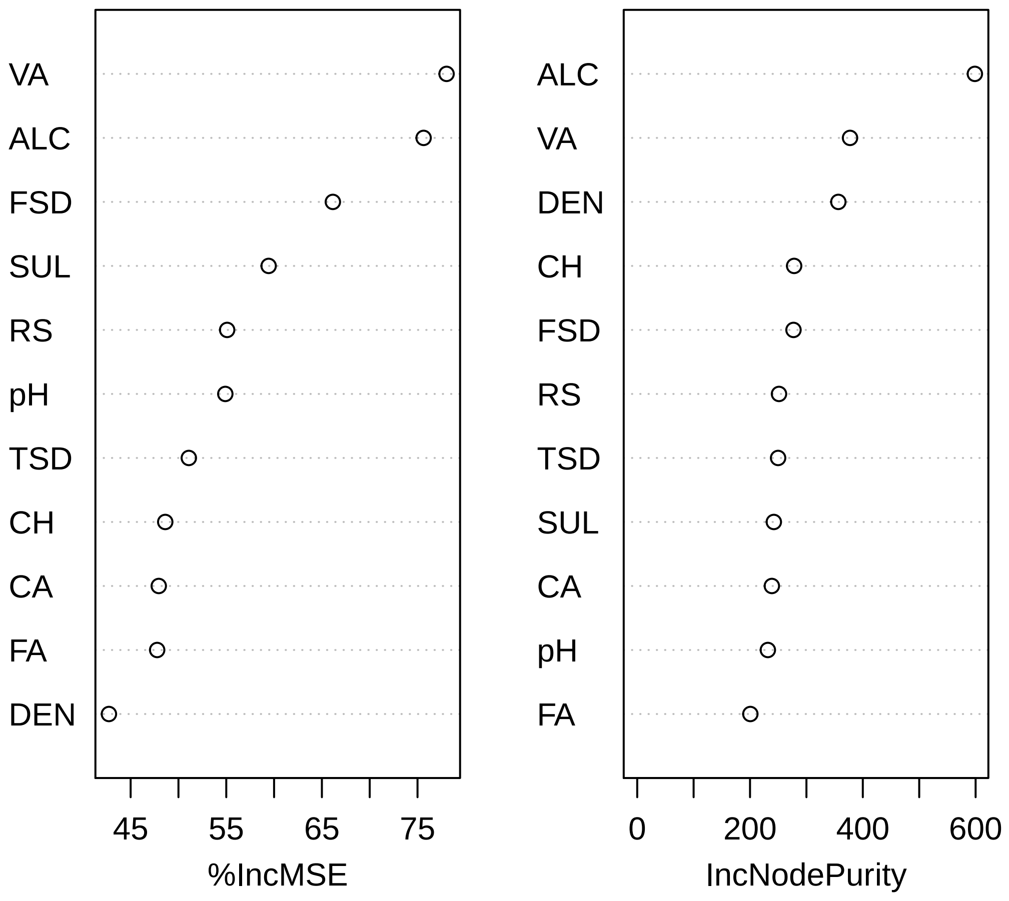


Figure 2: Importance of the dataset attributes for the prediction of the QLT attribute

```
varImpPlot(fitRF1, main="")
```

The accuracy of the RF prediction is calsulated below. Table 3 presents re results of the RF analysis in the form of a confusion matrix. The viasual presentation of the calsulations is presented in Figure 3.

```
PredictionRF1 <- predict(fitRF1, test1.data)
cor(PredictionRF1,test1.data$QLT)
```

```
#> [1] 0.6981861
```

```
library(ggplot2)
df2 = data.frame(as.factor(test1.data$QLT), PredictionRF1)
colnames(df2) <- c("Test","Prediction")
ggplot(df2, aes(x = Test, y = Prediction)) +
        geom_boxplot(outlier.colour = "red") +
        geom_jitter(width = 0.25, pch=20, col=rgb(0.1, 0.2, 0.8, 0.3))
```

	3	4	5	6	7	8	9
4	0	1	1	0	0	0	0
5	4	37	434	107	7	0	0
6	7	15	210	694	166	24	0
7	0	0	6	49	147	30	1
8	0	0	0	0	0	8	0

Table 3: `Random Forest Pledictor Confusion Matrix`

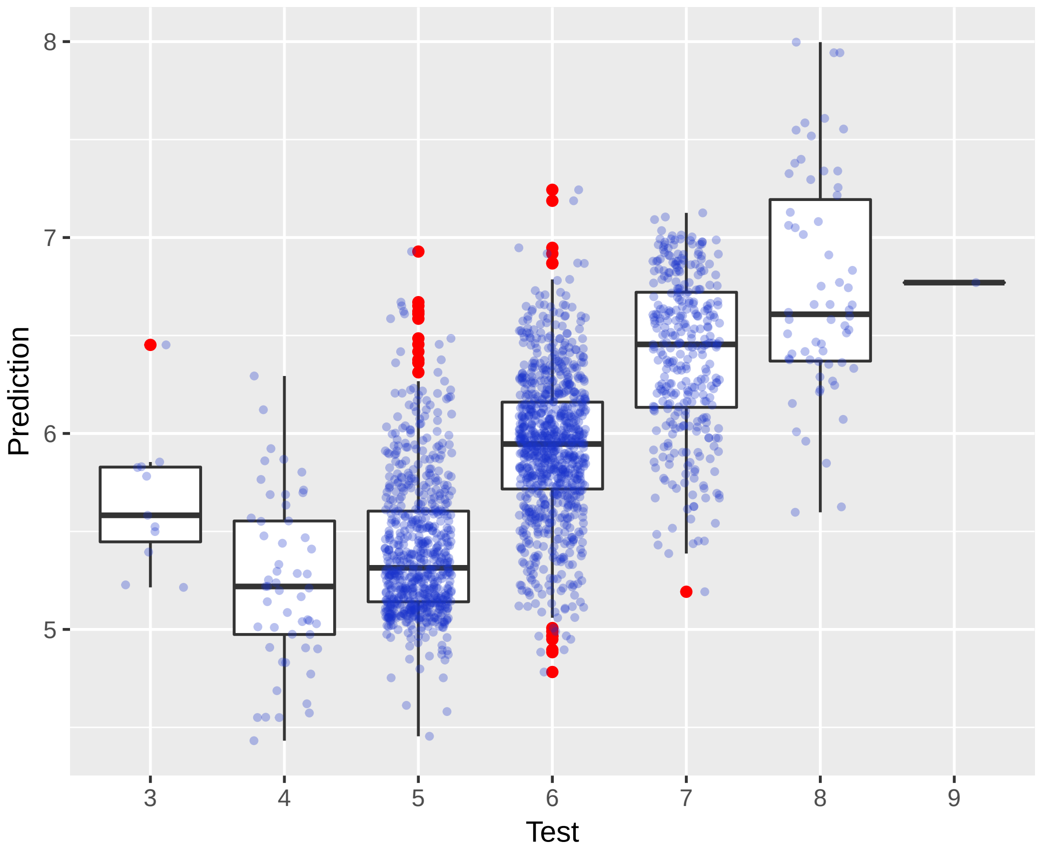


Figure 3: Random Forest Prediction

Support Vector Machine Clasification

Create SVM Model and show summary

```
library("e1071")
svm_model <- svm(QLT ~ ., data=train1.data)
summary(svm_model)

#>
#> Call:
#> svm(formula = QLT ~ ., data = train1.data)
#>
#>
#> Parameters:
#>    SVM-Type:  eps-regression
#>  SVM-Kernel:  radial
#>        cost:  1
#>       gamma:  0.09090909
#>     epsilon:  0.1
#>
#>
#> Number of Support Vectors:  3901
```

Run SVM Prediction

The Support Vector Regression (SVR) uses the same principles as the SVM for classification, with only a few minor differences. First of all, because output is a real number it becomes very difficult to predict the information at hand, which has infinite possibilities. In the case of regression, a margin of tolerance (epsilon) is set in approximation to the SVM which would have already requested from the problem. But besides this fact, there is also a more complicated reason, the algorithm is more complicated therefore to be taken in consideration. However, the main idea is always the same: to minimize error, individualizing the hyperplane which maximizes the margin, keeping in mind that part of the error is tolerated.

```
predSVM <- predict(svm_model, test1.data)
cor(predSVM,test1.data$QLT)
```

```
#> [1] 0.6064237
```

	3	4	5	6	7	8	9
4	0	0	4	0	0	0	0
5	4	38	393	150	8	0	0
6	7	15	247	623	203	34	0
7	0	0	7	77	109	28	1

Table 4: `SVM Pledictor Confusion Matrix`

```
library(ggplot2)
df2 = data.frame(as.factor(test1.data$QLT), predSVM)
colnames(df2) <- c("Test","Prediction")
ggplot(df2, aes(x = Test, y = Prediction)) +
        geom_boxplot(outlier.colour = "red") +
        geom_jitter(width = 0.25, pch=20, col=rgb(0.1, 0.2, 0.8, 0.3))
```

Neural Networks Prediction

source

Preparing scaled data

```
set.seed(4231)
data <- wines_data[,1:12]
index <- sample(1:nrow(data),round(0.75*nrow(data)))
#index <- createDataPartition(y= data$QLT, p=0.5, list = FALSE)
maxs <- apply(data, 2, max)
mins <- apply(data, 2, min)
scaled <- as.data.frame(scale(data, center = mins, scale = maxs - mins))
train_ <- scaled[index,]
test_ <- scaled[-index,]
```

```
library(neuralnet)
n <- names(train_)
f <- as.formula(paste("QLT ~", paste(n[!n %in% "QLT"], collapse = " + ")))
f
```

```
#> QLT ~ FA + VA + CA + RS + CH + FSD + TSD + DEN + pH + SUL + ALC
```

```
#nn <- neuralnet(f,data=train_,hidden=c(6,3),linear.output=F)
```

The Figure 5 demostrates the NN model with the weights on each connection.

```
#plot(nn)
knitr::include_graphics("images/nn_lg.png")
```

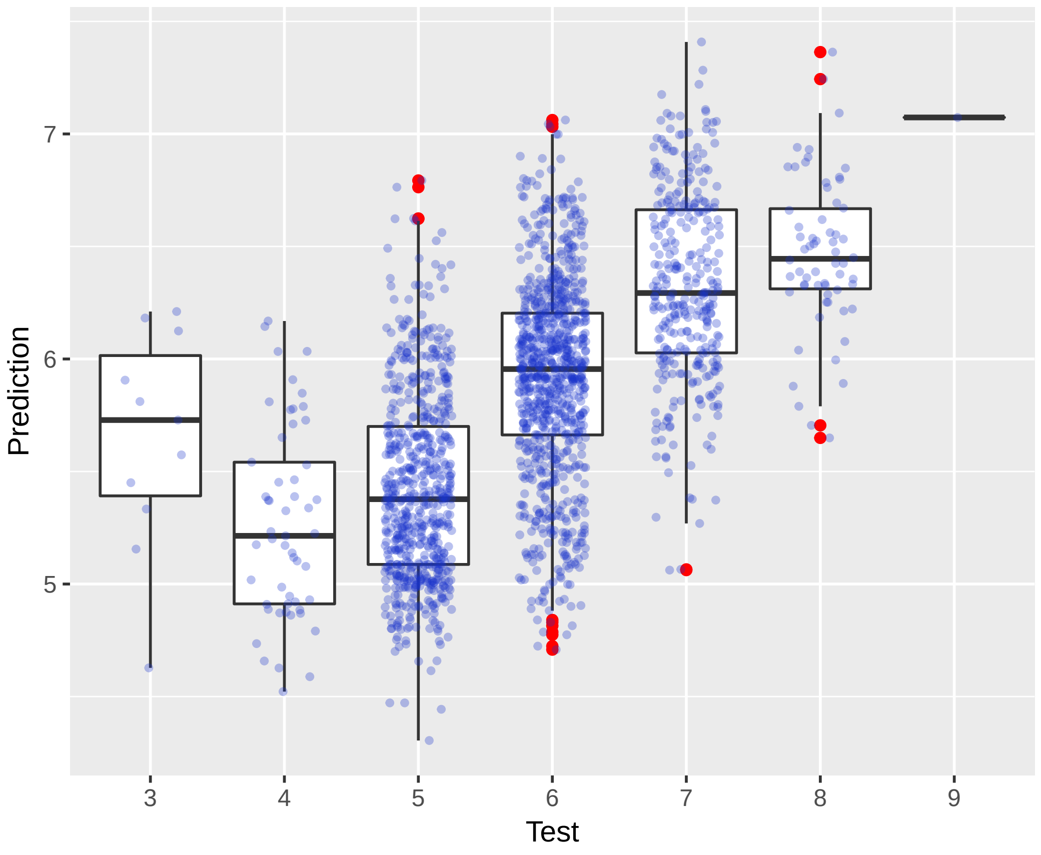


Figure 4: SVM Prediction

	3	4	5	6	7	8	9
4	0	2	5	0	0	0	0
5	8	28	336	189	10	0	0
6	1	13	179	456	137	19	0
7	0	0	6	97	115	21	2

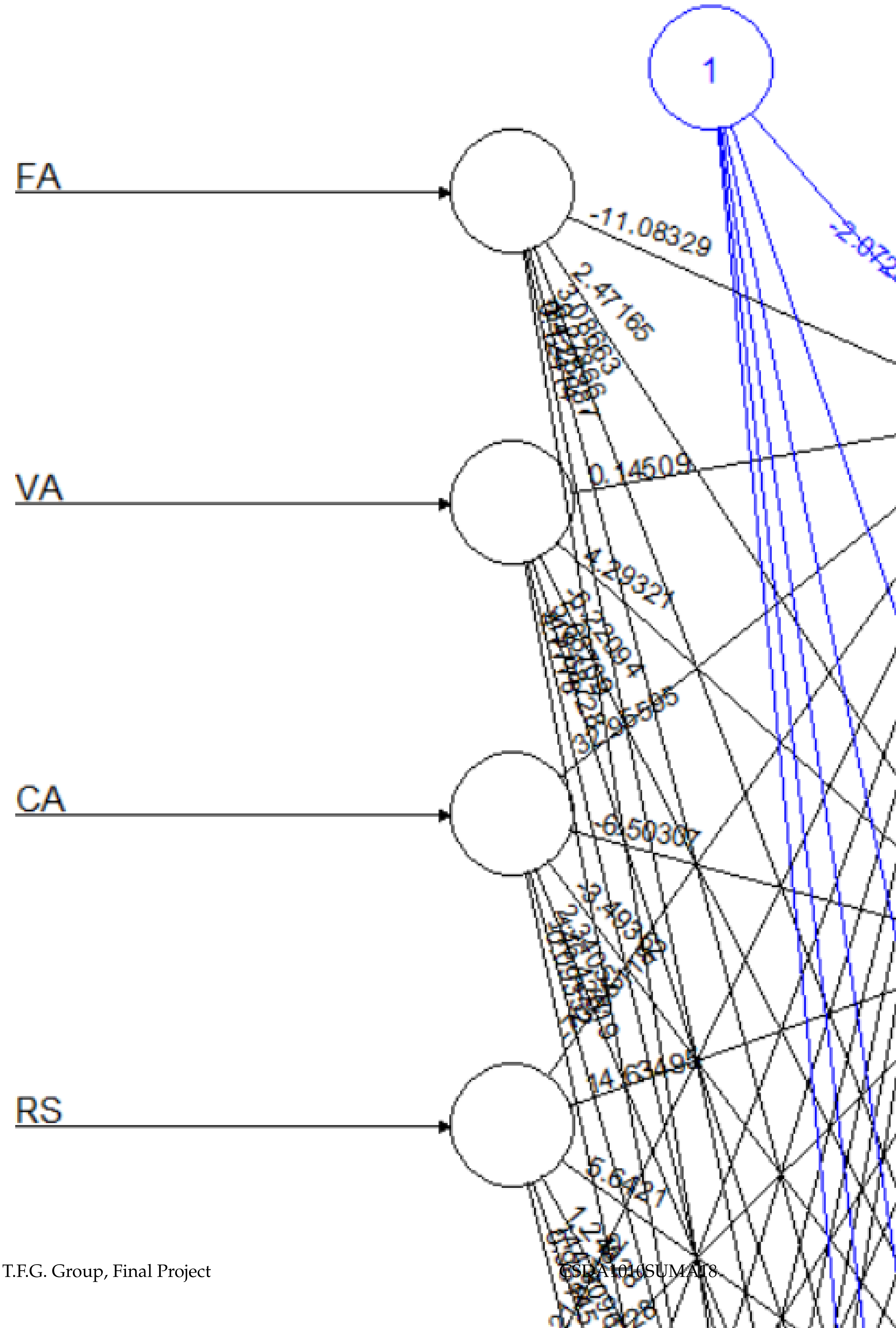
Table 5: `NN Pledictor Confusion Matrix`

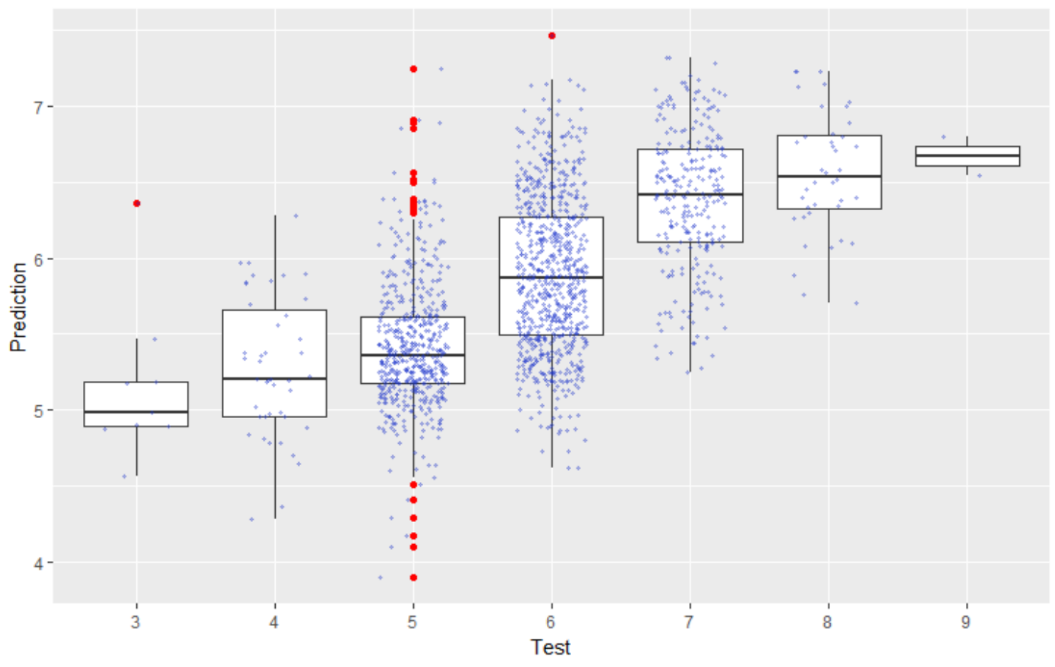
Predicting whine quality using neural networks

NN outputs a normalized prediction, so we need to scale it back in order to make a meaningful comparison (or just a simple prediction)

```
## [1] 0.6043741164
```

```
knitr::include_graphics("images/nn_scatter.png")
```


1
FA
VA
CA
RS
-11.08329
2.47165
0.14509
4.29327
-2.72094
-3.29595
-6.50307
3.49313
14.63495
6.6421



Can we Guess Wine Type by its Biochemical Content?

```
knitr::include_graphics("images/vinho-verde-wine-region-20.jpg")
```

Preparation for cluster analysis

Wines dataset normalizing

Normalizing wine dataset in preparation for clustering

```
wines_data.std <- scale(wines_data[1:11])
```

Determine optimal number of clusters

First we need to determine number of clusters. Looking at the percentage of variance explained as: a function of the number of clusters, we should choose a number of clusters in order to ensure that too much modeling of the data is not given. More precisely, if one plots the percentage of variance explained by the clusters against the number of clusters, the first clusters will add much more information (explains a lot of variance); but at some point, the marginal gain will drop, giving

FA	VA	CA	RS	CH	FSD
Min. :0.1425	Min. :-0.3624	Min. :-2.193	Min. :-0.7657	Min. :0.5414	Min. :-1.1001
1st Qu.:0.1425	1st Qu.: 2.0064	1st Qu.:-2.193	1st Qu.:-0.7447	1st Qu.:0.5485	1st Qu.:-1.0719
Median :0.2967	Median : 2.1887	Median :-2.193	Median :-0.7447	Median :0.5699	Median :-0.9310
Mean :0.7338	Mean : 1.9660	Mean :-1.505	Mean :-0.7097	Mean :0.7412	Mean :-0.8559
3rd Qu.:0.4510	3rd Qu.: 2.4620	3rd Qu.:-1.986	3rd Qu.:-0.6817	3rd Qu.:0.9124	3rd Qu.:-0.7902
Max. :3.0736	Max. : 3.2820	Max. : 1.661	Max. :-0.5976	Max. :1.1979	Max. :-0.3113

TSD	DEN	pH	SUL	ALC
Min. :-1.4462	Min. :0.7014	Min. :-0.36384	Min. :0.1931	Min. :-0.9154
1st Qu.:-1.4197	1st Qu.:0.8348	1st Qu.:-0.02177	1st Qu.:0.1931	1st Qu.:-0.9154
Median :-1.2162	Median :1.0349	Median : 1.03552	Median :0.2603	Median :-0.7477
Mean :-1.1956	Mean :0.9460	Mean : 0.86967	Mean :0.4507	Mean :-0.7477
3rd Qu.:-1.0128	3rd Qu.:1.0349	3rd Qu.: 1.81295	3rd Qu.:0.6803	3rd Qu.:-0.5800
Max. :-0.8624	Max. :1.1016	Max. : 1.81295	Max. :0.9995	Max. :-0.5800

an angle in the graph. The number of clusters is chosen at this point. This method is called the 'elbow criterion'. The diagram presented in Figure 6 demonstrates the 'elbow' curve. From this diagram we decided to use five (5) clusters in our analysis.

```
wssplot <- function(data, nc=15, seed=1234){
  wss <- (nrow(data)-1)*sum(apply(data,2,var))
  for (i in 2:nc){
    set.seed(seed)
    wss[i] <- sum(kmeans(data, centers=i)$withinss)}
  plot(1:nc, wss, type="b", xlab="Number of Clusters",
       ylab="Within groups sum of squares")}

wssplot(wines_data.std, nc=8)
```

Clustering using K-means method

k-means clustering (?) is a method of vector quantization, which is popular for cluster analysis in data mining. k-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster. This results in a partitioning of the data space into Voronoi cells (?).

The problem is computationally difficult (NP-hard), k-means clustering tends to find clusters of comparable spatial extent, while the expectation-maximization mechanism allows clusters to have different shapes. The algorithm has a loose relationship to the k-nearest neighbor classifier. One can apply the 1-nearest neighbor classifier on the cluster centers obtained by k-means to classify new data into the existing clusters. This is known as nearest centroid classifier or Rocchio algorithm.

Resulting cluster centers are presented in Table 6.

```
set.seed(420)
clusters_num = 3
k.means.fit <- kmeans(wines_data.std, clusters_num,iter.max = 1000)
```

	FA	VA	CA	RS	CH	FSD	TSD	DEN	pH	SUL	ALC
1	-0.18	-0.35	0.28	1.20	-0.09	0.85	0.96	0.76	-0.39	-0.26	-0.80
2	-0.35	-0.40	-0.01	-0.44	-0.44	-0.09	0.03	-0.85	-0.04	-0.28	0.57
3	0.88	1.18	-0.32	-0.60	0.94	-0.84	-1.20	0.71	0.54	0.84	-0.13

Table 6: `K-means Resulting Cluster Centers`

```
library(cluster)
clusplot(wines_data.std, k.means.fit$cluster, main='',
         color=TRUE, shade=FALSE,
         labels=clusters_num, lines=0)
```

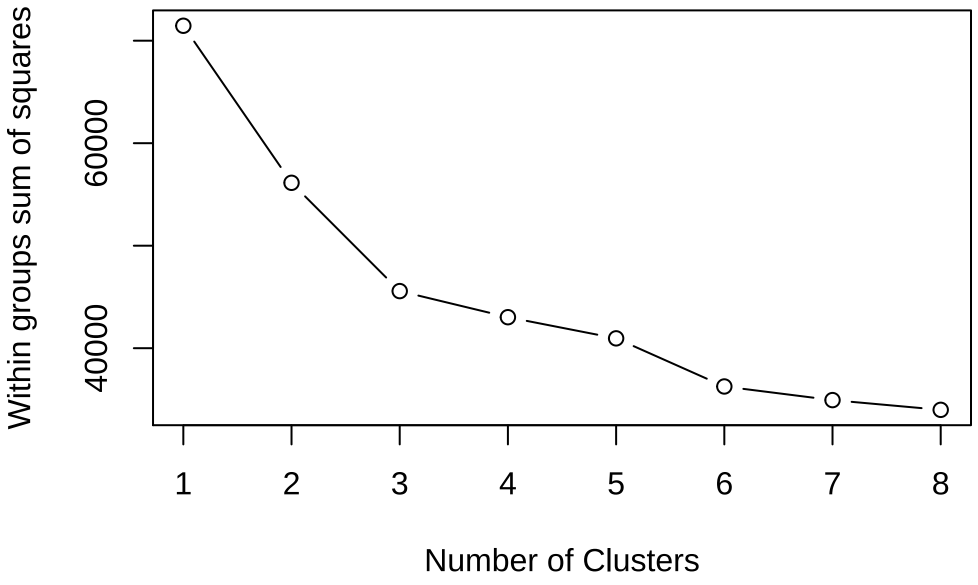


Figure 6: Elbow Criterion Diagram

Explain clusters

Explain by wine quality

Let's try to explain clusters by the wine quality. Code below builds a matrix whe columns are cluster numbers and rows are wine types. As we can see, quality does not explain clusters, it's evenly distributed among them. QLT does not explain clusters, see Table 7.

\begin{Sinput} xtable(table(wines_data[,12],k.means.fit$cluster), caption = "\tt Explaining Clusters by QLT", label = "table:clust_qlt") \end{Sinput}

	1	2	3
3	12	8	10
4	48	100	68
5	804	638	696
6	843	1371	622
7	157	740	182
8	30	148	15
9	1	4	0

Table 7: `Explaining Clusters by QLT`

Explain by wine type

We have the following types:

- 0 is red wine
- 1 is red wine

Let's try to explain clusters by the wine type. Code below builds a matrix whe columns are cluster numbers and rows are wine types. As we can see, cluster 3 contains red wines, cluster 1 and 2 contain white wine, see Table 10.

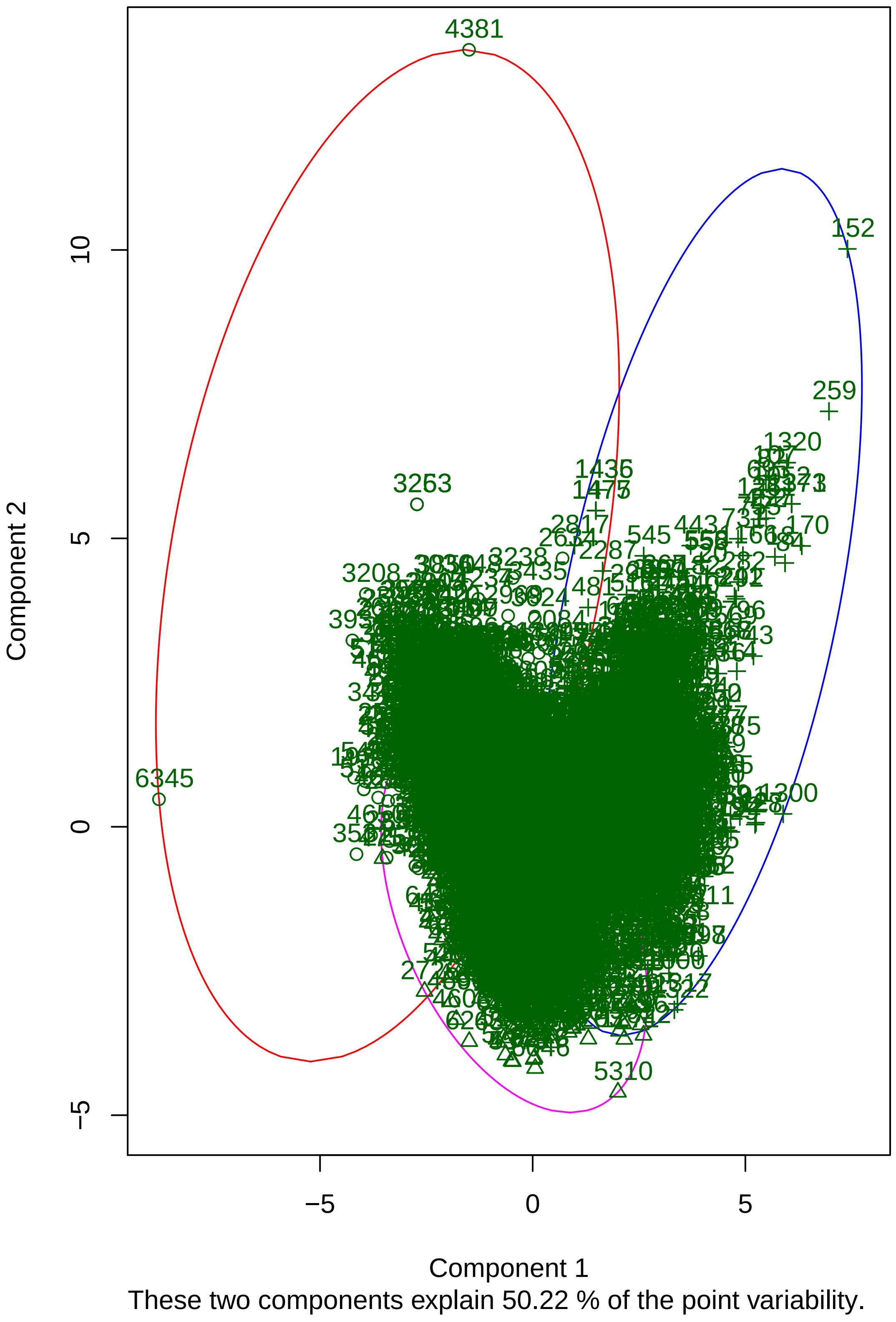


Figure 7: 2D representation of the Cluster solution

\begin{Sinput} centers <- table(wines_data[,13],k.means.fit$cluster) rownames(centers) <- c("Red Wine", "White Wine") xtable(centers, caption = "\tt Explaining Clusters by Wine Type", label = "table:clust_type1") \end{Sinput}

	1	2	3
Red Wine	4	57	1538
White Wine	1891	2952	55

Table 8: `Explaining Clusters by Wine Type`

Let's try to figure out how white wine clusters 1 and 2 differ from each other. Code below calculates a difference vector of cluster 1 and 2 and the sort the attributes in the order of the most influence:

```
Difference <- k.means.fit$centers[1,] - k.means.fit$centers[2,]
Difference <- Difference[order(abs(Difference), decreasing = T)]
```

Let's find what are the most significant factors that separate group 1 from group 2. Code below calculates the different results presented in Table 9.

	Difference
RS	1.64
DEN	1.61
ALC	-1.36
FSD	0.94
TSD	0.92
CH	0.35
pH	-0.35
CA	0.28
FA	0.17
VA	0.05
SUL	0.03

Table 9: `Difference between Clusters 1 and 1`

The table below shows that clusters 1 and 2 are mostly differ in sweetness (RS), viscosity (DEN) and alcohol content (ALC). This make cluster 1 sweet wite wine group and cluster 2 contains dry white wines.

More wine groups

Trying second "elbow" at 6.

```
set.seed(420)
clusters_num = 6
k.means.fit <- kmeans(wines_data.std, clusters_num,iter.max = 1000)
k.means.fit$centers
```

```
#>            FA         VA          CA         RS         CH        FSD
#> 1 -0.17159719 -0.3490199  0.31565629  1.4578300 -0.1424724  0.9312448
#> 2 -0.55122950 -0.2604946 -0.03411138 -0.4675512 -0.5870818 -0.1064107
#> 3  2.01336811  0.5015977  0.96093142 -0.5583690  1.2655358 -0.8973891
#> 4  0.08647484  1.6832857 -1.25231716 -0.6290660  0.6799824 -0.7982045
#> 5 -0.59649828 -0.5099758 -0.15932913 -0.2684651 -0.2719430  0.3963864
#> 6  0.12785003 -0.4759224  0.26262960 -0.3075337 -0.2266869 -0.2758534
#>           TSD        DEN           pH         SUL         ALC
#> 1  0.99603290  0.9151438 -0.503766204 -0.27882703 -0.87587845
#> 2 -0.16151381 -1.3386494  0.006282308 -0.28339251  1.43435398
#> 3 -1.25430120  0.9883244 -0.067282155  1.42340509  0.04216737
#> 4 -1.16402574  0.4941803  0.955234543  0.39920461 -0.24182036
#> 5  0.54343720 -0.2905900  0.762396539  0.03378434 -0.17543086
#> 6  0.05434809 -0.4784973 -0.772767076 -0.48738407 -0.01514628
```

```
k.means.fit$size
```

```
#> [1] 1481 1186  643  952 1042 1193
```

Cluster Analysis K-Means - more wine types

\begin{Sinput} centers <- table(wines_data[,13],k.means.fit$cluster) rownames(centers) <- c("Red Wine", "White Wine") xtable(centers, caption = "\tt Explaining Clusters by Wine Type", label = "table:clust_type1") \end{Sinput}

	1	2	3	4	5	6
Red Wine	2	39	624	901	22	11
White Wine	1479	1147	19	51	1020	1182

Table 10: `Explaining Clusters by Wine Type`

Conclusion

Through exploring the Wine Quality dataset we developed an algorithm to predict the wine quality using its chemical characteristics and extracted some interesting information about the wines presented.

First we applied the Random Forest Regresson methos and achieved accuracy of predicting wine quality of 72%. The method has shown that the most important attriutes that influence the quality estimation of wines by human experts are alcohol (ALC), volatile acidity (VA) and free sulfur dioxide (FSD). The RF method showed low precision in the area of poor and high quality wines.

Next we applied a Support Vector Machine Regression method (SVM) and achieved accuracy of 62, lower than RF, but the precision in the area of poor and high quality wines increased at least 20%.

Next we applied we applied a Neural Networks (NN) regression cofigured with 11:6:3:1 layers and achieved overall accuracy of 60%, lower than both RF and SVM. Interestingly, the precision of NN in the area of poor and high quality wines was almost 10% higher than SVM, making NN the best method in quality prediction of the most interesting and least presented sector of wines in the dataset.

Finally we applied Cluster Analysis (CA) to investigate if we could predict the quality better or if we could get any new knowledge from the dataset. We discovered that there is no correlation between wine quality and biochemical data from the CA point of view. On the other hand, there is a strong correlation between clusters and wine types. Even though the information about influence of 11 chemical characteristics on the taste of wine is very limited, we discovered clusters that contain such types of wine as white sweet, white dry, rose and old dry red. We can add that more subtypes of wine were discovered, but it is diffucult to give them specific names using only the dataset at hand.

The project was a success. Next steps would be collecting more information about relation of the base chemical component on the wine taste and finding datasets with additional wine attributes, related more to the human interpretation of wines, and connets those datasets with the one used in the project.

Bibliography

Sulfites (So2) in wine: top 7 facts - social vignerons. URL http://socialvignerons.com/2017/03/02/sulphites-so2-in-wine-top-7-facts/. [p2]

Weird wine flavors and the science behind them, Feb. 2017. URL https://winefolly.com/tutorial/weird-wine-flavors-and-the-science-behind-them/. [p2]

Random forest, Aug. 2018. URL https://en.wikipedia.org/w/index.php?title=Random_forest&oldid=854167685. Page Version ID: 854167685. [p5]

J. Busch. Learn about wine: an easy explanation of wine types, Sept. 2011. URL https://www.primermagazine.com/2011/learn/an-easy-explanation-of-wine-types. [p2]

M. S. Coli, A. G. P. Rangel, E. S. Souza, M. F. Oliveira, A. C. N. Chiaradia, M. S. Coli, A. G. P. Rangel, E. S. Souza, M. F. Oliveira, and A. C. N. Chiaradia. Chloride concentration in red wines: influence of terroir and grape type. *Food Science and Technology*, 35(1):95–99, Mar. 2015. ISSN 0101-2061. doi: 10.1590/1678-457X.6493. URL http://www.scielo.br/scielo.php?script=sci_abstract&pid=S0101-20612015000100095&lng=en&nrm=iso&tlng=en. [p2]

C. A. Corison, C. S. Ough, H. W. Berg, and K. E. Nelson. Must acetic acid and ethyl acetate as mold and rot indicators in grapes. *American Journal of Enology and Viticulture*, 30(2):130–134, Jan. 1979. ISSN 0002-9254. URL http://www.ajevonline.org/content/30/2/130. [p2]

P. Cortez, A. Cerdeira, F. Almeida, T. Matos, and J. Reis. Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems*, 47(4):547–553, 1998. [p1, 2]

D. B. Dahl. *xtable: Export Tables to LaTeX or HTML*, 2016. URL https://CRAN.R-project.org/package=xtable. R package version 1.8-2. [p3]

FAOSTAT. Faostat. URL http://faostat.fao.org/site/535/DesktopDefault.aspx?PageID=535. [p1]

A. Legin, A. Rudnitskaya, L. Lvova, Y. Vlasov, C. Di Natale, and A. D'Amico. Evaluation of Italian wine by the electronic tongue: recognition, quantitative analysis and correlation with human sensory perception. *Analytica Chimica Acta*, 484(1):33–44, May 2003. ISSN 00032670. doi: 10.1016/S0003-2670(03)00301-5. URL http://linkinghub.elsevier.com/retrieve/pii/S0003267003003015. [p1]

R. Teranishi, E. L. Wick, and I. Hornstein, editors. *Flavor chemistry: thirty years of progress*. Kluwer Academic/Plenum Publishers, New York, 1999. ISBN 9780306461996. [p1]

UCI Wine Data Set. Uci machine learning repository: wine quality data set. URL `http://archive.ics.uci.edu/ml/datasets/Wine+Quality`. [p2]

I. H. Witten, E. Frank, and M. A. Hall. *Data mining: practical machine learning tools and techniques*. Morgan Kaufmann series in data management systems. Morgan Kaufmann, Burlington, MA, 3rd ed edition, 2011. ISBN 9780123748560. OCLC: ocn262433473. [p1]

Note from the Authors

This file was generated using *The R Journal* style article template, additional information on how to prepare articles for submission is here - Instructions for Authors. The article itself is an executable R Markdown file that could be downloaded from Github with all the necessary artifacts.

Viviane Adohouannon
York University School of Continuing Studies

https://learn.continue.yorku.ca/user/view.php?id=21444

Kate Alexander
York University School of Continuing Studies

https://learn.continue.yorku.ca/user/view.php?id=21524

Diana Azbel
York University School of Continuing Studies

https://learn.continue.yorku.ca/user/view.php?id=20687

Igor Baranov
York University School of Continuing Studies

https://learn.continue.yorku.ca/user/profile.php?id=21219