Vinho Verde Wines Quality Data Mining

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 certification step. We are using a dataset related to red vinho verde wine samples, from the north of Portugal. The goal is to model wine quality based on physicochemical tests. The classes are ordered and not balanced (e.g. there are much more normal wines than excellent or poor ones). The method used to solve the problem are Random Forests, Clustering, Neural Networks and upport Vector Machine Clasification (SVM). Resuts of those methoda applicaion are compared and analyzed.

Introduction

Data mining (DM) techniques aim at extracting knowledge from raw data. Several DM algorithms have been developed, each one with its own advantages and disadvantages (Witten and Frank, 2005). DM 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., 2009; Yin and Han, 2003). 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 (Ebeler, 1999). 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 paper wine taste preferences are modelled by DM 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 Linear Regression methods. The following methodology 'check list' standard for a Linear Regression tasks will be applied to the problem at hand:

- Put all relevant variables in the model
- Leave the irrelevant variables out
- Check linearity
- Check regression assumptions:
  - Residuals have a mean of zero
  - Normality of errors
  - Residuals are not auto correlated

- Linearity of variables
- More data than independent variables is used in model building
- No excessive collinearity

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).

```
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 (score between 0 and 10) - (QLT)
```

Volatile acidity refers to the steam distillable acids present in wine, primarily acetic acid but also lactic, formic, butyric, and propionic acids. Commonly, these acids are measured by Cash Still, though now they can be measured by gas chromatography, HPLC or enzymatic methods.

Sulphur dioxide (SO2) is the most widely used and controversial additive in winemaking. Its main functions are 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 ros? as the above regulations show. Generally speaking, the drier the wine, the lesser the amount of SO2 it contains. Ref.

In wine tasting, the term "acidity" 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. Three primary acids are found in wine grapes: tartaric, malic and citric acids

Sulfates 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.

Red wines from different countries have been assessed in order to determine the influence of terroir and grape variety in their concentration of chloride. Ref

Data Preparation

To perform the analysis, certain R libraries were used. The code below was used to load and initialize the libraries. The first line invoking seed function was applied to enforce the repeatability of the calculation results.

Reading red wines dataset

The dataset (UCI Wine Data Set) of red wine quality has 12 attributes and 1599 instances. For more information, read (Cortez et al., 1998). The following is the concept structure of the dataset:

```
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
```

Reading white wines dataset

```
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
```

Combining the datasets

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

```
#>        FA               VA               CA               RS
#>  Min.   : 3.800   Min.   :0.0800   Min.   :0.0000   Min.   : 0.600
#>  1st Qu.: 6.400   1st Qu.:0.2300   1st Qu.:0.2500   1st Qu.: 1.800
#>  Median : 7.000   Median :0.2900   Median :0.3100   Median : 3.000
#>  Mean   : 7.215   Mean   :0.3397   Mean   :0.3186   Mean   : 5.443
#>  3rd Qu.: 7.700   3rd Qu.:0.4000   3rd Qu.:0.3900   3rd Qu.: 8.100
#>  Max.   :15.900   Max.   :1.5800   Max.   :1.6600   Max.   :65.800
#>        CH               FSD              TSD              DEN
#>  Min.   :0.00900   Min.   :  1.00   Min.   :  6.0   Min.   :0.9871
#>  1st Qu.:0.03800   1st Qu.: 17.00   1st Qu.: 77.0   1st Qu.:0.9923
#>  Median :0.04700   Median : 29.00   Median :118.0   Median :0.9949
#>  Mean   :0.05603   Mean   : 30.53   Mean   :115.7   Mean   :0.9947
#>  3rd Qu.:0.06500   3rd Qu.: 41.00   3rd Qu.:156.0   3rd Qu.:0.9970
#>  Max.   :0.61100   Max.   :289.00   Max.   :440.0   Max.   :1.0390
#>        pH             SUL              ALC             QLT
#>  Min.   :2.720   Min.   :0.2200   Min.   : 8.00   Min.   :3.000
#>  1st Qu.:3.110   1st Qu.:0.4300   1st Qu.: 9.50   1st Qu.:5.000
#>  Median :3.210   Median :0.5100   Median :10.30   Median :6.000
#>  Mean   :3.219   Mean   :0.5313   Mean   :10.49   Mean   :5.818
#>  3rd Qu.:3.320   3rd Qu.:0.6000   3rd Qu.:11.30   3rd Qu.:6.000
#>  Max.   :4.010   Max.   :2.0000   Max.   :14.90   Max.   :9.000
#>       TYPE
#>  Min.   :0.0000
#>  1st Qu.:1.0000
#>  Median :1.0000
#>  Mean   :0.7539
#>  3rd Qu.:1.0000
#>  Max.   :1.0000
```

Preview of the data

To pretty-print the first rows of the dataset xtable (Dahl, 2016) library was used to generate 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

Table 1: Red Wines Quality Dataset - first rows

Data attributes summary

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.

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 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.

3	0.00
4	0.03
5	0.33
6	0.44
7	0.17
8	0.03
9	0.00

Table 3: `Distribution of the target QLT attribute in the original dataset`

```
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)))
```

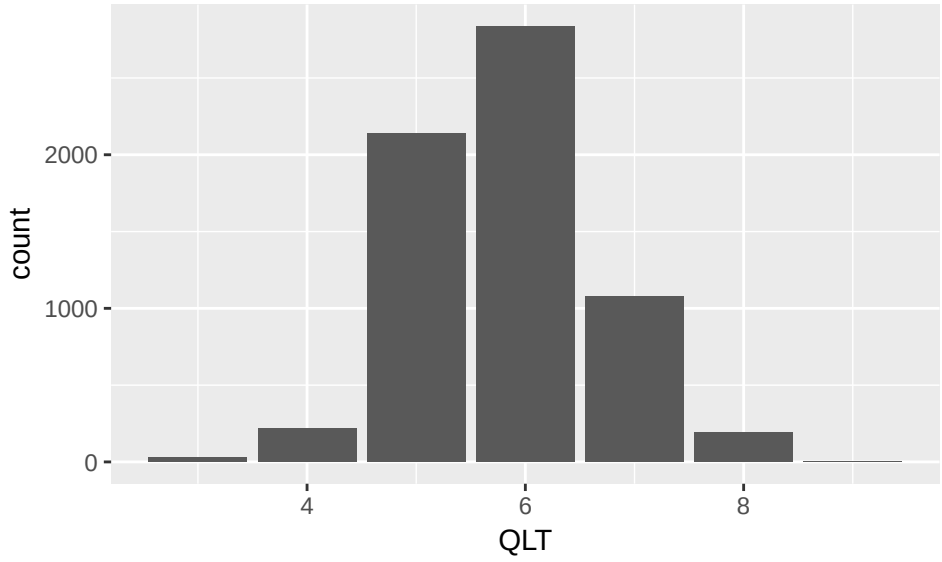

Figure 1: Distribution of Wine Quality Attribute

```
#>
#>            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)))

#>
#>           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

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

varImpPlot(fitRF1, main="")


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))
```

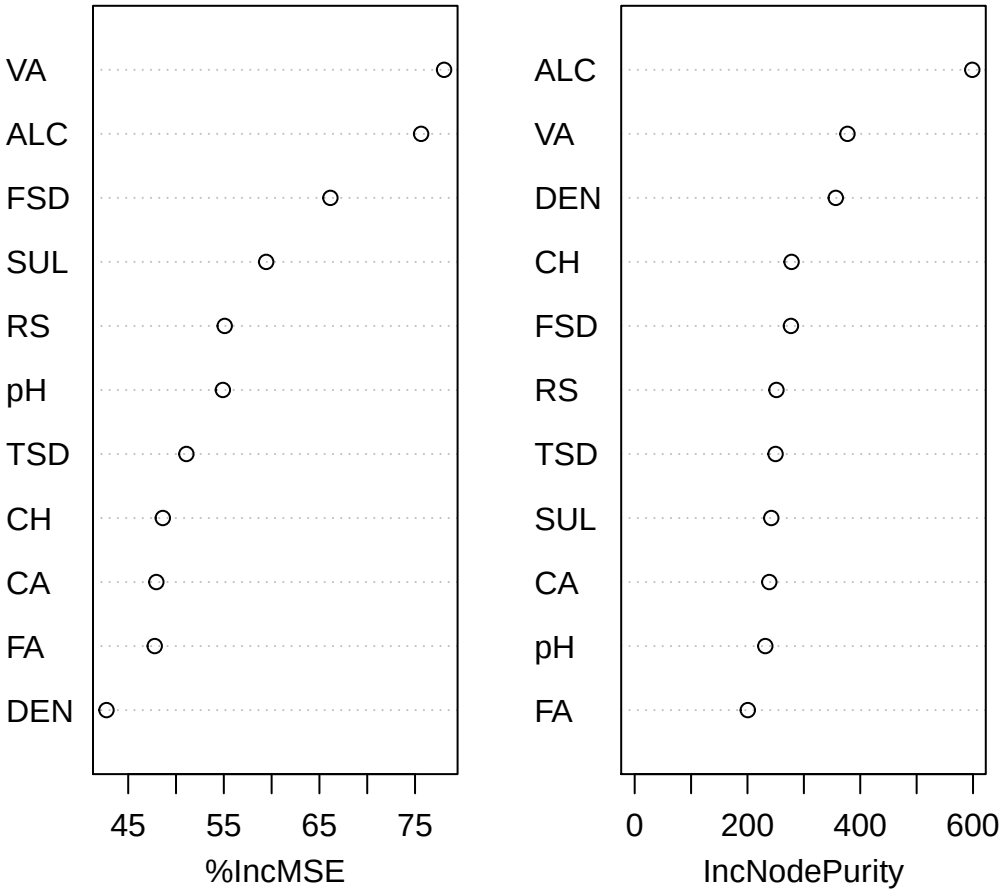

Figure 2: Importance of the dataset attributes for the prediction of the QLT attribute

	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 4: Random Forest Pledictor Confusion Matrix

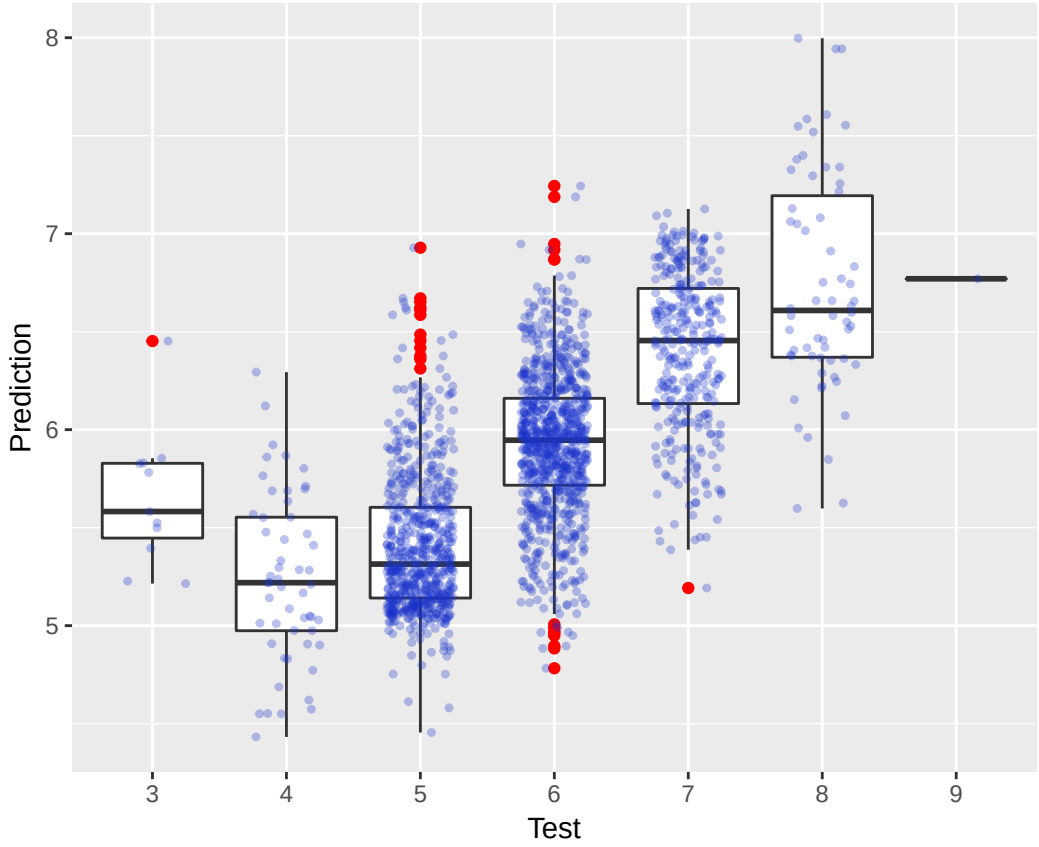

Figure 3: Random Forest Prediction

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
```

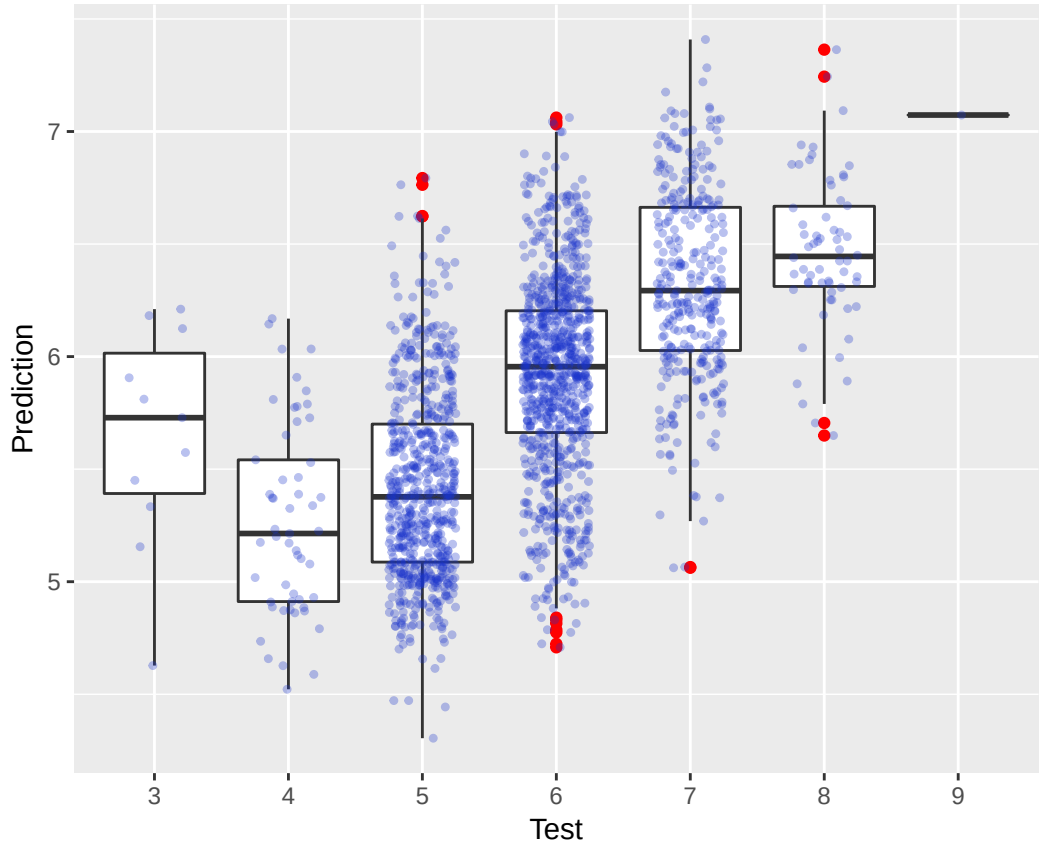

Figure 4: SVM Prediction

Run SVM Prediction

```
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 5: 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)
nn <- neuralnet(f,data=train_,linear.output=F)
```

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

```
plot(nn)
```

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)

```
pr.nn <- compute(nn,test_[,1:11])
pr.nn_ <- pr.nn$net.result*(max(data$QLT)-min(data$QLT))+min(data$QLT)
test.r <- (test_$QLT)*(max(data$QLT)-min(data$QLT))+min(data$QLT)
#MSE.nn <- sum((test.r - pr.nn_)^2)/nrow(test_)
cor(test.r,pr.nn_)

#>              [,1]
#> [1,] 0.5286179844
```

	3	4	5	6	7	8	9
5	4	23	297	170	18	0	0
6	5	20	222	503	171	27	0
7	0	0	7	69	73	13	2

Table 6: NN Pledictor Confusion Matrix

```
library(ggplot2)
df2 = data.frame(as.factor(test.r), pr.nn_)
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))
```

Preparation for cluster analysis

Wines dataset normalizing

Normalizing wine dataset in preparation for clustering

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

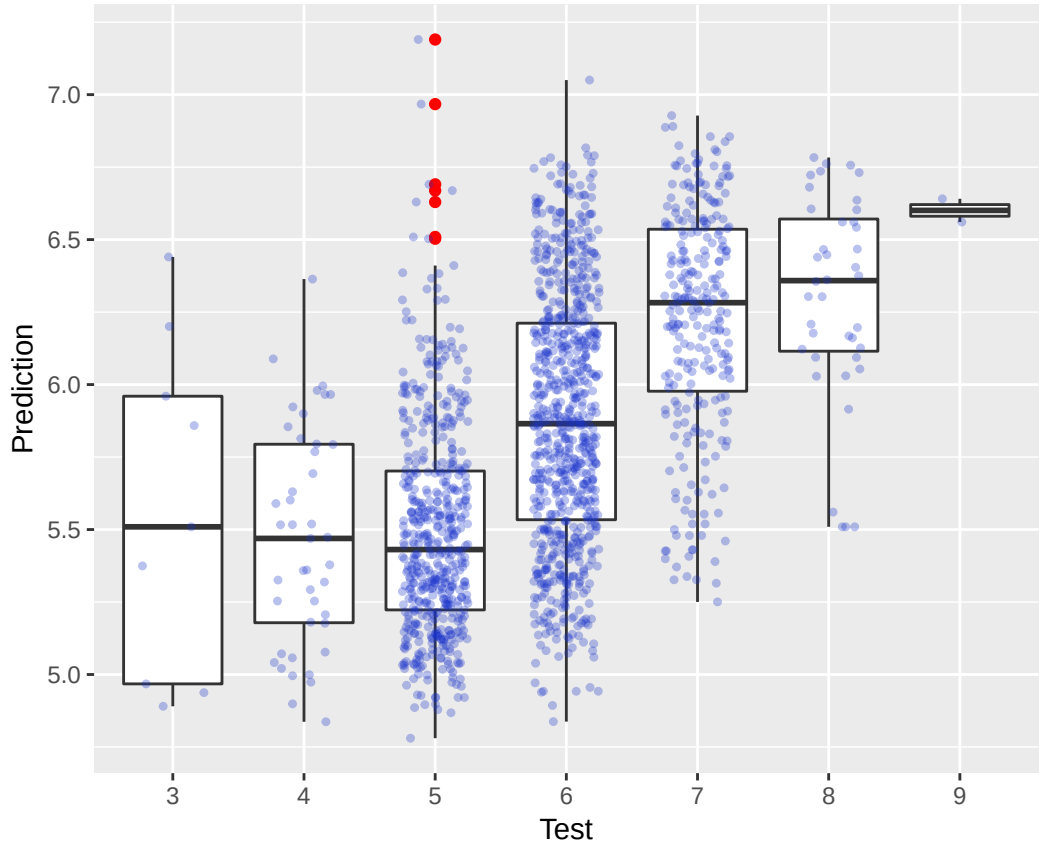

Figure 5: SVM Prediction

```
#>                FA            VA           CA            RS           CH
#> [1,] 0.1424623002  2.1886644640 -2.192663755 -0.7447207852 0.5699139522
#> [2,] 0.4510010108  3.2819823390 -2.192663755 -0.5975940776 1.1978825063
#> [3,] 0.4510010108  2.5531037557 -1.917405100 -0.6606483809 1.0266183552
#> [4,] 3.0735800508 -0.3624105778  1.660957411 -0.7447207852 0.5413699270
#> [5,] 0.1424623002  2.1886644640 -2.192663755 -0.7447207852 0.5699139522
#> [6,] 0.1424623002  1.9457049362 -2.192663755 -0.7657388863 0.5413699270
#>                FSD           TSD           DEN            pH           SUL
#> [1,] -1.1000551922 -1.4462472102 1.0349131650  1.8129499714 0.1930819102
#> [2,] -0.3112961255 -0.8624022483 0.7014323224 -0.1150641737 0.9995016912
#> [3,] -0.8746954589 -1.0924017788 0.7681284909  0.2580998544 0.7978967459
#> [4,] -0.7620155922 -0.9862481493 1.1016093335 -0.3638401924 0.3274852071
#> [5,] -1.1000551922 -1.4462472102 1.0349131650  1.8129499714 0.1930819102
#> [6,] -0.9873753255 -1.3400935808 1.0349131650  1.8129499714 0.1930819102
#>               ALC
#> [1,] -0.9153937087
#> [2,] -0.5800234900
#> [3,] -0.5800234900
#> [4,] -0.5800234900
#> [5,] -0.9153937087
#> [6,] -0.9153937087
```

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 an angle in the graph. The number of clusters is chosen at this point. This method is called the 'elbow

Figure 6: Elbow Criterion Diagram

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 7.

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

#> [1] 1895 3009 1593
```

	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 7: 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)
```

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.

```
table(wines_data[,12],k.means.fit$cluster)
```

```
#>
#>        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
```

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 wines.

```
table(wines_data[,13],k.means.fit$cluster)
```

```
#>
#>        1    2    3
#>   0    4   57 1538
#>   1 1891 2952   55
```

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:

```
v12 <- k.means.fit$centers[1,] - k.means.fit$centers[2,]
v12 <- v12[order(abs(v12), decreasing = T)]
print(v12)
```

```
#>             RS            DEN            ALC            FSD            TSD
#>  1.63613496843  1.60991184338 -1.36447615456  0.93786425106  0.92277030364
#>             CH             pH             CA             FA             VA
#>  0.35467498938 -0.34813273149  0.28454502075  0.16536506894  0.04956478354
#>            SUL
#>  0.02527705163
```

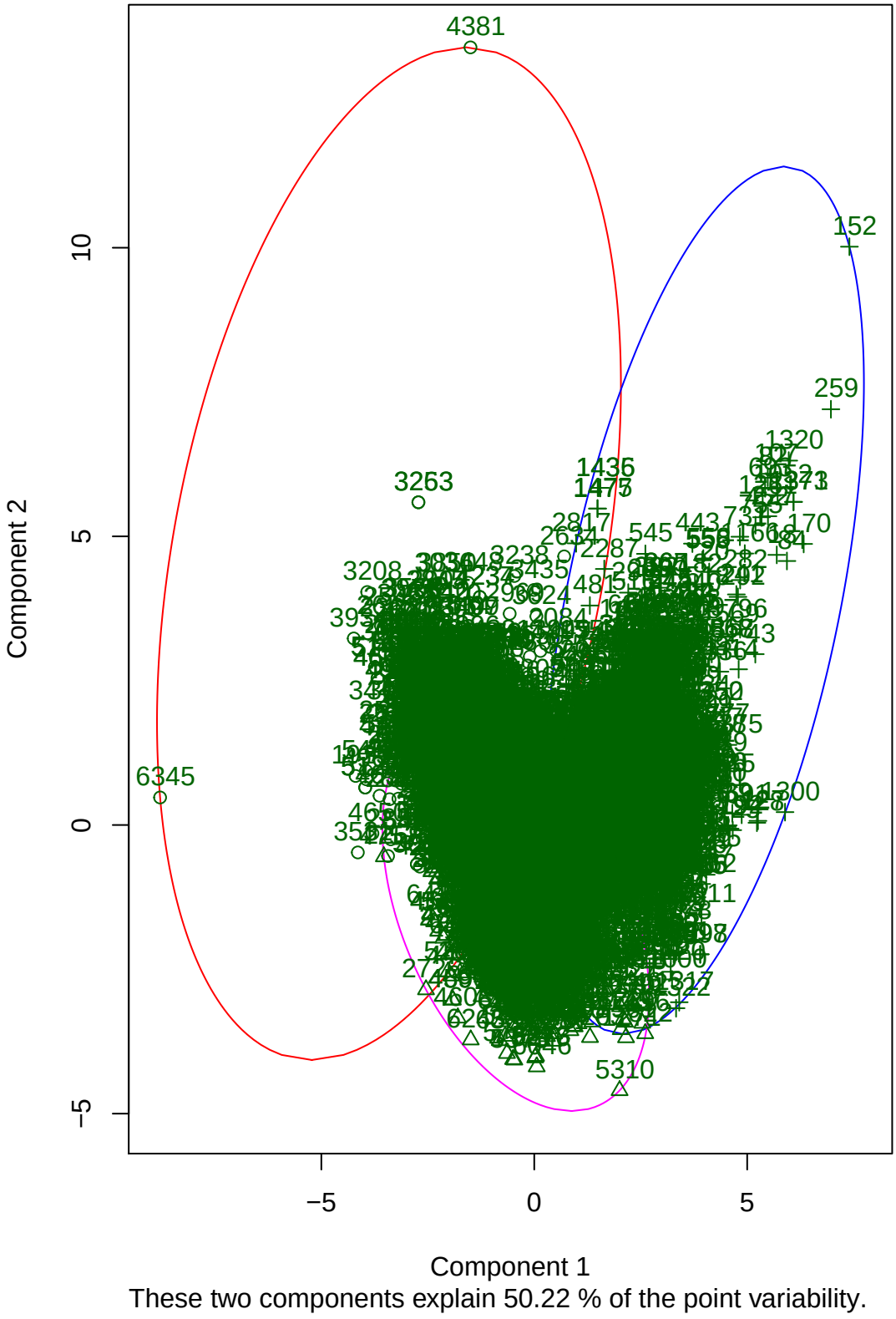

Figure 7: 2D representation of the Cluster solution

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
#> 1 -0.17159719083 -0.3490199151  0.31565629188  1.4578300323 -0.1424724482
#> 2 -0.55122949726 -0.2604945870 -0.03411137878 -0.4675512229 -0.5870817669
#> 3  2.01336810518  0.5015977119  0.96093141556 -0.5583690056  1.2655358413
#> 4  0.08647483985  1.6832856773 -1.25231715863 -0.6290660347  0.6799823518
#> 5 -0.59649827525 -0.5099758476 -0.15932912927 -0.2684651377 -0.2719430364
#> 6  0.12785003101 -0.4759223689  0.26262960140 -0.3075337120 -0.2266868647
#>             FSD            TSD           DEN              pH
#> 1  0.9312447744  0.99603290217  0.9151437793 -0.503766204250
#> 2 -0.1064106849 -0.16151381023 -1.3386493855  0.006282307649
#> 3 -0.8973891490 -1.25430119700  0.9883243872 -0.067282154509
#> 4 -0.7982045305 -1.16402574371  0.4941802840  0.955234543422
#> 5  0.3963864250  0.54343719667 -0.2905899742  0.762396539192
#> 6 -0.2758534432  0.05434808849 -0.4784972541 -0.772767076449
#>              SUL            ALC
#> 1 -0.27882703200 -0.87587844656
#> 2 -0.28339250921  1.43435397525
#> 3  1.42340509078  0.04216737053
#> 4  0.39920461335 -0.24182035591
#> 5  0.03378433676 -0.17543086275
#> 6 -0.48738406863 -0.01514628393
```

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

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

Conclusion

Through exploring the Red Wine Quality dataset and using a different methods of Linear Regression we developed an algorithm to predict the wine quality using its chemical characteristics.

First we applied the Linear Regression OLS method and through several steps of correcting the model and adjusting for skewness and then comparing it to automated Stepwise Regression method we achieved the accuracy of RT at 0.5480833 is 50% higher than OLS.

Next we applied tree-based regression methods. First we used a Regression Tree which gave us accuracy of 0.5480833. The final method was the Random Forest Regressor and achieved 80% better accuracy at 0.6734 comparing to OLS.

The project was a success, however, none of the Linear Regression methods used would give us reliable precision. All the plots show the presence of outliers and inaccuracy in the areas of low and high scores. We conclude that the current problem could not be solved by the Linear Regression methods only, it looks that additional methods like Clustering is required to split the dataset into smaller sets to satisfy Linear Regression limitations.

Bibliography

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. [p3]

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. [p4]

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. [p3]

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