Name: Suresh Subramaniam

Subject : CMTH 642

Homework 3

Date: April 20, 2016

**Task**

**CMTH 642**

**Homework 3**

**PROBLEM DEFINITION**

Build a model to predict the quality of red wine using its attributes.

*This is an informally defined problem. First formally define this problem and then*

*build a model and report your findings.*

**DATA**

The dataset is related to white Portuguese "Vinho Verde" wine. For more info go:

https://archive.ics.uci.edu/ml/datasets/Wine+Quality

**TASK**

1. Import the data to R (no cleanup necessary).

2. Check data characteristics. What is the correlation?

3. Propose a model for the prediction. Give a few reasons for your selection

briefly. You may choose to model the problem as classification or

regression. Define the task, experience and performance criteria.

4. Report your results.

**OUTPUT**

A zip file containing the following

A report with:

o Model

o Results

o Brief discussion of the work done for each task

All the source code

**Report**

1. **What is the Problem?**
   1. **Informal definition of the problem**

Build a model to predict the quality of red wine using its attributes.

* 1. **Formal description of problem**

**Task(T)**: Classify the new red wine (that has not be tasted) to get the quality .i.e. Grade the wine quality between 0 (very bad) and 10 (very excellent)

**Experience(E)**: corpus of existing data for which experts have graded it.

**Performance(P)**: Classification accuracy, the number of wine sample predicted correctly out of all wine samples tested considered as percentage.

* 1. **Assumptions**

It is assumed that the tasting and gradation was done by a single wine taster and that there are no other attributes. We assume that there is relations between the input variables and the quality of the wine. We also assume collinearity among the input variables.

**2) Why does the problem need to be solved?**

**2.1) Motivation:** This has been given as part of the course assignment.

**2.2)** **Solution benefits**: Currently the wine samples are tasted by Expert wine tasters to grade them. The Wine tasters are very expensive and it is an subjective evaluation of the wine. The grading may vary from person to person. The model will classify and will eliminate the need of the wine taster.

**2**.3) **Solution use**: The model will be used to identify the quality of the wine.

**3) How would I solve the problem?**

3.1) Import the data into R.

3.2) Clean the data and understand it. ( cleaning process skipped as the data provided clean)

3.3) Build Support Vector machine classifier and check the accuracy

3.4) Build a Naïve Bayes Classifier and check the accuracy

3.5) Build a KNN classifier and check the accuracy …Choose the one with highest accuracy

1. **Data importing and understanding**

#

setwd("C:/Suresh Data science/CMTH642 Data Analytics Adv methods/Assignment3")

redwineQ <- read.table("winequality-red.csv", header= TRUE, sep = ";", strip.white = TRUE)

#

summary(redwineQ)

str(redwineQ)

> summary(redwineQ)

fixed.acidity volatile.acidity citric.acid residual.sugar chlorides

Min. : 4.60 Min. :0.1200 Min. :0.000 Min. : 0.900 Min. :0.01200

1st Qu.: 7.10 1st Qu.:0.3900 1st Qu.:0.090 1st Qu.: 1.900 1st Qu.:0.07000

Median : 7.90 Median :0.5200 Median :0.260 Median : 2.200 Median :0.07900

Mean : 8.32 Mean :0.5278 Mean :0.271 Mean : 2.539 Mean :0.08747

3rd Qu.: 9.20 3rd Qu.:0.6400 3rd Qu.:0.420 3rd Qu.: 2.600 3rd Qu.:0.09000

Max. :15.90 Max. :1.5800 Max. :1.000 Max. :15.500 Max. :0.61100

free.sulfur.dioxide total.sulfur.dioxide density pH

Min. : 1.00 Min. : 6.00 Min. :0.9901 Min. :2.740

1st Qu.: 7.00 1st Qu.: 22.00 1st Qu.:0.9956 1st Qu.:3.210

Median :14.00 Median : 38.00 Median :0.9968 Median :3.310

Mean :15.87 Mean : 46.47 Mean :0.9967 Mean :3.311

3rd Qu.:21.00 3rd Qu.: 62.00 3rd Qu.:0.9978 3rd Qu.:3.400

Max. :72.00 Max. :289.00 Max. :1.0037 Max. :4.010

sulphates alcohol quality

Min. :0.3300 Min. : 8.40 Min. :3.000

1st Qu.:0.5500 1st Qu.: 9.50 1st Qu.:5.000

Median :0.6200 Median :10.20 Median :6.000

Mean :0.6581 Mean :10.42 Mean :5.636

3rd Qu.:0.7300 3rd Qu.:11.10 3rd Qu.:6.000

Max. :2.0000 Max. :14.90 Max. :8.000

> str(redwineQ)

'data.frame': 1599 obs. of 12 variables:

$ fixed.acidity : num 7.4 7.8 7.8 11.2 7.4 7.4 7.9 7.3 7.8 7.5 ...

$ volatile.acidity : num 0.7 0.88 0.76 0.28 0.7 0.66 0.6 0.65 0.58 0.5 ...

$ citric.acid : num 0 0 0.04 0.56 0 0 0.06 0 0.02 0.36 ...

$ residual.sugar : num 1.9 2.6 2.3 1.9 1.9 1.8 1.6 1.2 2 6.1 ...

$ chlorides : num 0.076 0.098 0.092 0.075 0.076 0.075 0.069 0.065 0.073 0.071 ...

$ free.sulfur.dioxide : num 11 25 15 17 11 13 15 15 9 17 ...

$ total.sulfur.dioxide: num 34 67 54 60 34 40 59 21 18 102 ...

$ density : num 0.998 0.997 0.997 0.998 0.998 ...

$ pH : num 3.51 3.2 3.26 3.16 3.51 3.51 3.3 3.39 3.36 3.35 ...

$ sulphates : num 0.56 0.68 0.65 0.58 0.56 0.56 0.46 0.47 0.57 0.8 ...

$ alcohol : num 9.4 9.8 9.8 9.8 9.4 9.4 9.4 10 9.5 10.5 ...

$ quality : int 5 5 5 6 5 5 5 7 7 5 ...

1. **Understanding the data:**
2. Dependent variable ( quality)

#Checking the dependent variable

# changing the quality attribute to factor type.

#

redwineQ$quality <- as.factor(redwineQ$quality)

str(redwineQ$quality)

summary(redwineQ$quality)

#

> str(redwineQ$quality)

Factor w/ 6 levels "3","4","5","6",..: 3 3 3 4 3 3 3 5 5 3 ...

> summary(redwineQ$quality)

3 4 5 6 7 8

10 53 681 638 199 18

**Observation:** The train sample will have 6 levels of quality 3 to 8, The quality 1,2,9,10 are not available so the newdata can’t be classified in that category(1,2,9,10)

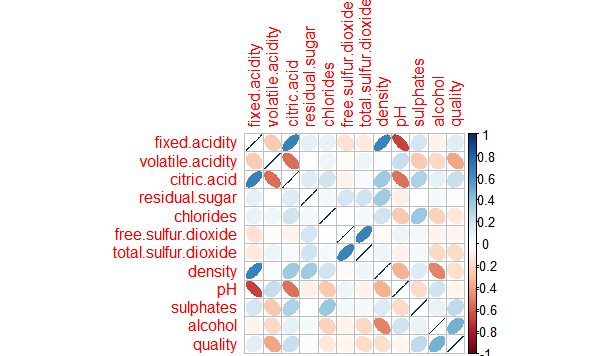
1. The independent variables and the correlation between them and the dependent variable

M <- cor(redwineQ)

corrplot(M, method="ellipse")

M <- cor(redwineQ)

> corrplot(M, method="ellipse")



**Observation:** Important observation is that the quality has a high positive correlation with alcohol and sulphate and have a strong negative correlation with Volatile acidity.

1. **Model selection: We will build SVM classifier first**

SUPPORT VECTOR MACHINE

# package e1071 is needed for SVM classification

install.packages("e1071", dependencies = TRUE)

library(e1071)

attach(redwineQ)

x <- subset(redwineQ, select=-quality)

y <- as.factor(quality)

#

# Apply SVM

svm\_model1 <- svm(x,y)

summary(svm\_model1)

> library(e1071)

> attach(redwineQ)

The following objects are masked from redwineQ (pos = 3):

alcohol, chlorides, citric.acid, density, fixed.acidity,

free.sulfur.dioxide, pH, quality, residual.sugar, sulphates,

total.sulfur.dioxide, volatile.acidity

> x <- subset(redwineQ, select=-quality)

> y <- as.factor(quality)

> #

> # Apply SVM

> svm\_model1 <- svm(x,y)

> summary(svm\_model1)

Call:

svm.default(x = x, y = y)

Parameters:

SVM-Type: C-classification

SVM-Kernel: radial

cost: 1

gamma: 0.09090909

Number of Support Vectors: 1341

( 495 574 191 53 18 10 )

Number of Classes: 6

Levels:

3 4 5 6 7 8

# run prediction and measure the execution time

pred <- predict(svm\_model1,x)

system.time(pred <- predict(svm\_model1,x))

table(pred,y)

pred <- predict(svm\_model1,x)

> system.time(pred <- predict(svm\_model1,x))

user system elapsed

0.56 0.00 0.57

> table(pred,y)

y

pred 3 4 5 6 7 8

3 0 0 0 0 0 0

4 0 1 0 0 0 0

5 7 38 549 169 10 0

6 3 13 129 450 115 12

7 0 1 3 19 74 6

8 0 0 0 0 0 0

**ACCURACY = 67.16%**

# tuning svm to get the best cost and Gamma

svm\_tune <- tune(svm, train.x=x, train.y=y,

kernel="radial", ranges=list(cost=10^(-1:2), gamma=c(.5,1,2)))

print(svm\_tune)

Parameter tuning of ‘svm’:

- sampling method: 10-fold cross validation

- best parameters:

cost gamma

1 1

- best performance: 0.3201808

# new model with Cost = 1 and gamma = 1

svm\_model2 <- svm(x,y, kernel="radial", cost=1, gamma=1)

summary(svm\_model2)

> svm\_model2 <- svm(x,y, kernel="radial", cost=1, gamma=1)

> summary(svm\_model2)

Call:

svm.default(x = x, y = y, kernel = "radial", gamma = 1, cost = 1)

Parameters:

SVM-Type: C-classification

SVM-Kernel: radial

cost: 1

gamma: 1

Number of Support Vectors: 1472

( 607 591 193 53 18 10 )

# run with new model

pred2 <- predict(svm\_model2,x)

system.time(pred2 <- predict(svm\_model2,x))

table(pred2,y)

> pred2 <- predict(svm\_model2,x)

> system.time(pred2 <- predict(svm\_model2,x))

user system elapsed

0.57 0.00 0.58

> table(pred2,y)

y

pred 3 4 5 6 7 8

3 6 0 0 0 0 0

4 0 30 0 0 0 0

5 3 13 656 27 1 0

6 1 10 24 609 15 2

7 0 0 1 2 183 4

8 0 0 0 0 0 12

**Accuracy of tuned SVM classifier = 93.55 %**

1. **Model selection: We will build Naïve Bayes classifier**

NAÏVE BAYES

#

# naive bayes

#

redwineQ$quality <- as.factor(redwineQ$quality)

model\_nb <- naiveBayes(quality ~ ., data = redwineQ)

pred <- predict(model\_nb, redwineQ[ ,-quality])

#

# form and display confusion matrix & overall accuracy

tab <- table(pred, redwineQ$quality)

tab

sum(tab[row(tab)==col(tab)])/sum(tab)

#

> # naive bayes

> #

> redwineQ$quality <- as.factor(redwineQ$quality)

> model\_nb <- naiveBayes(quality ~ ., data = redwineQ)

> pred <- predict(model\_nb, redwineQ[ ,-quality])

> #

> # form and display confusion matrix & overall accuracy

> tab <- table(pred, redwineQ$quality)

> tab

pred 3 4 5 6 7 8

3 2 1 4 0 0 0

4 1 7 6 10 0 0

5 6 32 531 258 14 0

6 1 13 129 307 101 9

7 0 0 10 60 82 7

8 0 0 1 3 2 2

> sum(tab[row(tab)==col(tab)])/sum(tab)

[1] 0.5822389

**Naïve bayes accuracy is 58.22**

**8)Model selection: We will build KNN classifier**

KNN Classifier

We will first Normalize the numeric data except the quality class and then build a KNN model

#

#USing KNN classifier

library(class)

#

# **Normalize**

normalize <- function(x) {

num <- x - min(x)

denom <- max(x) - min(x)

return (num/denom)

}

redwineQ\_a <- redwineQ[ ,-12]

redwineQ\_c <- redwineQ[ , 12]

redwineQ\_a\_scaled <- as.data.frame(lapply(redwineQ\_a, normalize))

redwine\_scaled <- cbind(redwineQ\_a\_scaled, quality = redwineQ\_c)

Index <- sample(nrow(redwine\_scaled), floor(nrow(redwine\_scaled)\*.8))

r\_train <- redwine\_scaled[Index,-12]

r\_test <- redwine\_scaled[-Index,-12]

trainLabel <- redwine\_scaled[Index,12]

testLabel <- redwine\_scaled[-Index,12]

#

# Build KNN Model

pred4 <- knn(train = r\_train, test = r\_test, cl = trainLabel, k=5)

pred4

tab <- table(pred4,testLabel)

sum(tab[row(tab)==col(tab)])/sum(tab)

#

> #USing KNN classifier

> library(class)

> #redwine\_n <- redwineQ[ , lapply(redwineQ, is.numeric) == TRUE]

> #

> # Normalize

> normalize <- function(x) {

+ num <- x - min(x)

+ denom <- max(x) - min(x)

+ return (num/denom)

+ }

> redwineQ\_a <- redwineQ[ ,-12]

> redwineQ\_c <- redwineQ[ , 12]

> redwineQ\_a\_scaled <- as.data.frame(lapply(redwineQ\_a, normalize))

> redwine\_scaled <- cbind(redwineQ\_a\_scaled, quality = redwineQ\_c)

> Index <- sample(nrow(redwine\_scaled), floor(nrow(redwine\_scaled)\*.8))

> r\_train <- redwine\_scaled[Index,-12]

> r\_test <- redwine\_scaled[-Index,-12]

> trainLabel <- redwine\_scaled[Index,12]

> testLabel <- redwine\_scaled[-Index,12]

> pred4 <- knn(train = r\_train, test = r\_test, cl = trainLabel, k=5)

> pred4

[1] 5 5 5 6 5 5 5 6 5 5 5 5 5 5 6 5 5 6 5 5 6 6 5 5 6 5 6 5 5 4 5 5 5 6 5 5 5 5 5 5 6

[42] 6 5 7 6 7 5 5 5 5 6 5 6 6 6 7 6 6 5 5 6 5 5 6 6 6 6 6 6 7 5 6 6 6 6 6 5 7 5 5 5 5

[83] 6 5 6 7 5 5 5 5 5 6 6 6 5 5 5 7 7 7 6 5 5 5 6 6 7 5 6 6 5 6 6 5 6 5 5 5 6 6 5 4 6

[124] 5 7 7 6 6 6 5 6 6 5 5 5 6 5 5 5 5 6 7 5 5 5 5 5 5 6 6 6 5 5 6 5 6 6 6 7 7 7 6 7 6

[165] 6 5 7 7 5 6 5 5 5 6 6 5 6 6 7 6 5 5 5 6 7 6 6 6 6 6 5 5 6 6 6 5 5 7 5 5 5 7 6 6 6

[206] 5 5 6 7 5 6 6 5 6 7 5 6 6 6 7 5 5 7 7 6 6 6 5 5 7 6 6 6 7 6 5 6 5 5 4 6 6 6 5 6 6

[247] 7 7 6 7 5 7 5 5 6 5 6 5 5 6 5 6 5 6 5 5 5 6 6 5 5 6 5 5 5 6 5 5 5 5 5 6 7 6 7 7 6

[288] 6 6 5 7 6 5 7 6 5 6 5 5 6 6 5 5 5 5 4 5 6 5 6 6 5 5 5 6 6 6 7 6 5

Levels: 3 4 5 6 7 8

> tab <- table(pred4,testLabel)

> sum(tab[row(tab)==col(tab)])/sum(tab)

[1] 0.575

**KNN classifier accuracy is 57.50 %**

**Conclusion**

Three different models were created and their accuracies are as below:

|  |  |
| --- | --- |
| Model | Accuracy |
| **Support Vector Machine** | 93.55 |
| **Naïve Bayes** | 58.22 |
| **KNN ( K=5)** | 57.50 % |

As the accuracy of the SVM model was very high, we will choose the same.