Project 2

Laukik Upadhye - 01833608

A. Dataset Used

Data set used for this project is observations for “Red wine quality”. This data is taken from Kaggle, it is about red variants of the Portuguese "Vinho Verde" wine. As it is testing measures, classes are not balanced. It has 11 independent variables and 1 dependent variable. It is multiclass classification problem, where classes are quality of wine denoted by 1 to 10.

Variables:

1. Fixed acidity
2. Volatile acidity
3. Citric acid
4. Residual sugar
5. Chlorides
6. Free sulfur dioxide
7. Total sulfur dioxide
8. Density
9. pH
10. Sulphates
11. Alcohol

As this data is not balanced Multinomial regression was failing to generate predictions for classes which are very low in quantity. Hence, I am amplifying data with SmoteClassif function. This function creates data with equal quantity with KNN algorithm.

I am using training and testing data set as 50% - 50% percent

```
#install.packages("UBL")

library(UBL)

data <- read.csv("winequality-red.csv",header = T,sep = ",")
data$quality <- as.factor(data$quality)
data<-SmoteClassif(quality~.,data,C.perc="balance",k=5,repl=FALSE, dist="Euclidean", p=2)
str(data)

## 'data.frame':    1596 obs. of  12 variables:
##  $ fixed.acidity       : num  7.6 6.4 11.7 8 5.6 9.4 6.9 9.4 10.7 7.2 ...
##  $ volatile.acidity    : num  0.55 0.57 0.49 1.18 0.54 0.4 0.63 0.34 0.43 0.725 ...
##  $ citric.acid         : num  0.21 0.14 0.49 0.21 0.04 0.47 0.02 0.37 0.39 0.05 ...
##  $ residual.sugar      : num  2.2 3.9 2.2 1.9 1.7 2.5 1.9 2.2 2.2 4.65 ...
##  $ chlorides           : num  0.071 0.07 0.083 0.083 0.049 0.087 0.078 0.075 0.106 0.086
...
```

```
##  $ free.sulfur.dioxide : num  7 27 5 14 5 6 18 5 8 4 ...
##  $ total.sulfur.dioxide: num  28 73 15 41 13 20 30 13 32 11 ...
##  $ density             : num  0.996 0.997 1 0.995 0.994 ...
##  $ pH                  : num  3.28 3.32 3.19 3.34 3.72 3.15 3.4 3.22 2.89 3.41 ...
##  $ sulphates           : num  0.55 0.48 0.43 0.47 0.58 0.5 0.75 0.62 0.5 0.39 ...
##  $ alcohol             : num  9.7 9.2 9.2 10.5 11.4 10.5 9.8 9.2 9.6 10.9 ...
##  $ quality             : Factor w/ 6 levels "3","4","5","6",..: 3 3 3 3 3 3 3 3 3 3 ...
```

```
summary(data)
```

```
##  fixed.acidity    volatile.acidity  citric.acid     residual.sugar
##  Min.   : 4.600   Min.   :0.1200   Min.   :0.0000   Min.   : 0.900
##  1st Qu.: 7.200   1st Qu.:0.3900   1st Qu.:0.0700   1st Qu.: 1.941
##  Median : 8.057   Median :0.5481   Median :0.2728   Median : 2.200
##  Mean   : 8.390   Mean   :0.5873   Mean   :0.2753   Mean   : 2.611
##  3rd Qu.: 9.488   3rd Qu.:0.7300   3rd Qu.:0.4432   3rd Qu.: 2.796
##  Max.   :15.900   Max.   :1.5800   Max.   :1.0000   Max.   :13.800
##    chlorides       free.sulfur.dioxide total.sulfur.dioxide    density
##  Min.   :0.01200   Min.   : 1.00       Min.   :  6.00       Min.   :0.9901
##  1st Qu.:0.06900   1st Qu.: 6.00       1st Qu.: 15.54       1st Qu.:0.9954
##  Median :0.07900   Median :10.34       Median : 27.00       Median :0.9966
##  Mean   :0.08876   Mean   :13.40       Mean   : 36.49       Mean   :0.9966
##  3rd Qu.:0.09200   3rd Qu.:18.00       3rd Qu.: 47.96       3rd Qu.:0.9977
##  Max.   :0.61000   Max.   :68.00       Max.   :289.00       Max.   :1.0032
##        pH          sulphates         alcohol       quality
##  Min.   :2.740   Min.   :0.3300   Min.   : 8.400   3:266
##  1st Qu.:3.219   1st Qu.:0.5492   1st Qu.: 9.714   4:266
##  Median :3.318   Median :0.6280   Median :10.598   5:266
##  Mean   :3.324   Mean   :0.6561   Mean   :10.713   6:266
##  3rd Qu.:3.420   3rd Qu.:0.7492   3rd Qu.:11.500   7:266
##  Max.   :3.900   Max.   :2.0000   Max.   :14.900   8:266
```

```
n=nrow(data)
n
```

```
## [1] 1596
```

```
p=ncol(data)
p
```

```
## [1] 12
```

As we can see from above, the 8 column names with their datatypes are shown above.Niether it has any missing values which free the work of data preprocessing. Above, We also have the statistical information for all the 8 attributes. Our aim is to apply “Multinomial linear Regression” to predict the home team outcome (Win (1), Loss(2) & Draw (3)) in all International football matches. The first 7 columns are the features and the last attribute “home_team_result” is the response variable. We will divide our dataset in two partitions: training (50%) & testing (50%). Before Partioning the data, we need to divide predictor variables in one matrix and the response variable in another matrix.

```
#data$home_team_result <- as.numeric(data$home_team_result)
mat <- matrix(0,n,p-1)
table(data$quality)
```

```
##   3   4   5   6   7   8
## 266 266 266 266 266 266
```

```
for (i in 1:(p-1)) {
  mat[,i]  <- data[,i]

}
label <- matrix(0,n,1)
label[,1] <- data[,p]

ind <- sample(1:n,floor(n/2),replace = F)

trn <- mat[ind,]
tst <- mat[-ind,]

trn_class <- label[ind]
tst_class <- label[-ind]
```

B. Model

Multinomial regression is used to perform regression where prediction classes are more than two. This technique is used to predict dependent variable based on number of independent variables. The dependent variable describes the outcome of stochastic event with density function.

To measure the performance of the medel we can use confusion matrix.

C. Performance

For measuring the performance of model, we can view by confusion matrix, where true positive and true negative values are given.

```
library(glmnet)

## Loading required package: Matrix

## Loaded glmnet 3.0-2

crossvalidation <- cv.glmnet(trn, trn_class, type.measure="mse", alpha=0,
                             family="multinomial",trace.it = 1)

crossvalidation

##
## Call:  cv.glmnet(x = trn, y = trn_class, type.measure = "mse", trace.it = 1,      alpha
= 0, family = "multinomial")
##
## Measure: Mean-Squared Error
##
##      Lambda Measure       SE Nonzero
## min 0.01958  0.5724 0.010000      11
## 1se 0.02841  0.5808 0.009622      11

lambda_min <- crossvalidation$lambda.min
model <- glmnet(trn, trn_class, family = "multinomial", alpha = 0,
                            lambda = lambda_min)
model
```

```
##
## Call:  glmnet(x = trn, y = trn_class, family = "multinomial", alpha = 0,      lambda =
lambda_min)
##
##   Df   %Dev  Lambda
## 1 11 0.4095 0.01958
```

```
pred <- predict(model, tst, type = "class")
```

library(e1071)

confusionMatrix(table(pred,tst_class))

```
```{r}
library(e1071)
confusionMatrix(table(pred,tst_class))
```
```

```
Confusion Matrix and Statistics

    tst_class
pred   1   2   3   4   5   6
   1 126  33  16   6   2   0
   2   2  52  13   9   8   0
   3   0  31  72  31   6   0
   4   3  19  22  26  18   1
   5   0   2   7  32  47  19
   6   0   0   1  22  59 113

Overall Statistics

               Accuracy : 0.5464
                 95% CI : (0.5111, 0.5813)
    No Information Rate : 0.1754
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.4558
```

With this I am getting accuracy of ~55%. This accuracy is impacted by unbalanced data points and many classes.

Similar model has been created by other user, he received by 85% of accuracy as they had developed model on random forest (it creates multiple decision trees, which results in more accuracy).