STAT 6340 Statistical Machine Learning

Mini Project 2

Author:
Suarez Rodes, Randy

Supervisor:
Choudhary, Pankaj Ph.D.

March 2, 2021

1 Answers

Question 1.a We explored the correlations of our variables (predictors and response) on Figure 1. Visualizing the correlation matrix we can see that predictors Oakiness and Clarity have a very low correlation with Quality. We infer that these variables are not much relevant to predict Quality and are the most likely to be removed from future models. On the other hand predictors Aroma, Flavor and Region has a high correlation with Quality, making them good candidates to build our models.
The variable Body presents some correlation with Quality, although, it is not as high as other predictors, we require further analysis to determine its relevance for the response variable. Finally, predictors Aroma, Body, Flavor and Region are highly correlated. This can cause overfitting issues, since the data would be over explained, so with further analysis one or some of these predictors might be dropped since they can be explained by other predictors.

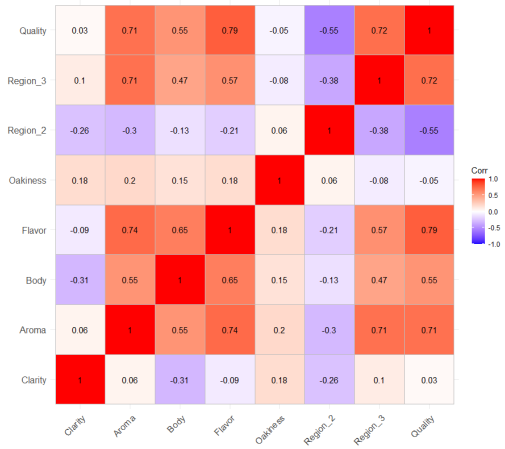

Figure 1: Correlation Matrix

We present the scatterplot of our variables on Figure 2. As expected from the correlation matrix we can notice that the predictors Aroma, Body and Flavor exhibit a positive linear relation with Quality. On Figure 3 we present the Quality boxplots by Region. We see a clear distribution of the Quality determine by the Region (high on 3, medium on 1, low on 2). This fact evidences the importance of this predictor for determining the Quality of a wine.

Figure 2: Scatter Plots

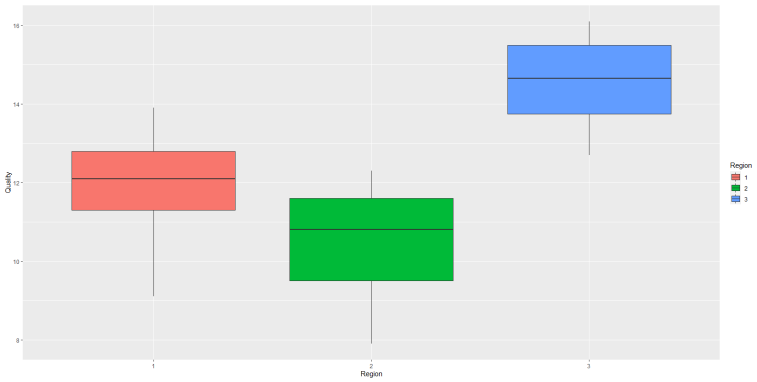

Figure 3: Quality by Region

Question 1.b We explored Quality using boxplots to identify possible outliers (Figure 4). We did not encounter any evident outlier. We also explore the distribution of Quality using histograms (Figure 5). We can appreciate that the data distribution looks approximately normal and is not highly skewed. At this moment we consider that a transformation is not required and we will check model assumptions once we build an appropriate model.

Question 1.c We fitted a simple linear regression model for every predictor (Tables 1 to 6). Taking a look to the p-values of every model we can see that for these simple models there is a statistically significant association between the predictor and the response variable for all predictors except for Clarity (p-value = 0.865) and Oakiness (p-value = 0.779). This is consistent with our previous exploration. We can see on Figure 6 how there is a clear linear trend for predictors Flavor, Aroma and Body. Also we discussed on question 1.a how Region is relevant to predict Quality (Figure 3).

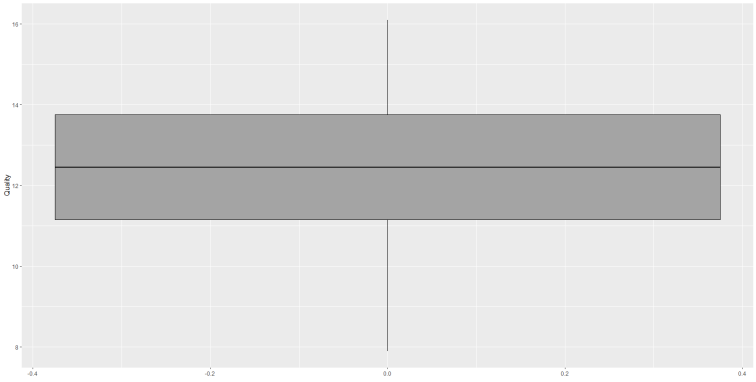

Figure 4: Quality Boxplot

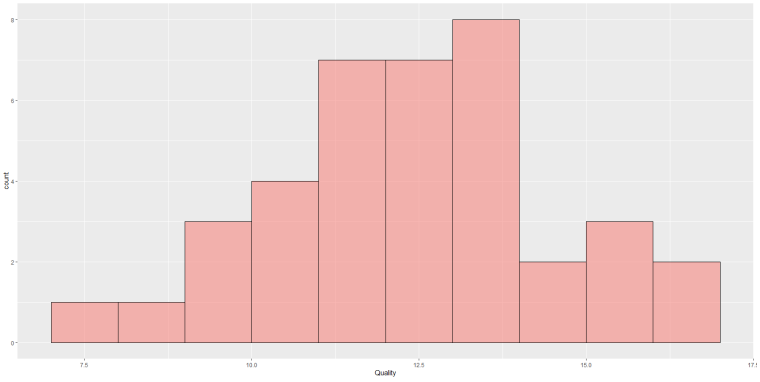

Figure 5: Quality Histogram

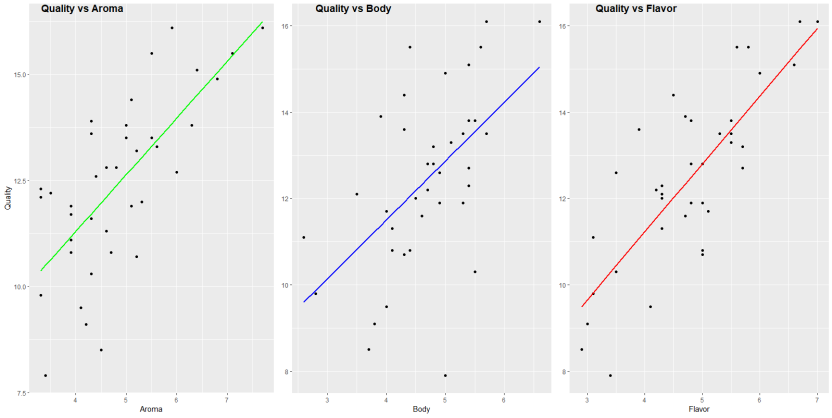

Figure 6: Quality vs Continuous Predictors

Table 1:

	Estimate	Pr(>|t|)
(Intercept)	12.0034	3.89e-5
Clarity	0.4692	0.865

Table 2:

	Estimate	Pr(>|t|)
(Intercept)	5.9583	4.51e-6
Aroma	1.3365	6.87e-7

Table 3:

	Estimate	Pr(>|t|)
(Intercept)	6.0580	0.0007
Body	1.3618	0.0004

Table 4:

	Estimate	Pr(>|t|)
(Intercept)	4.9414	1.57e-5
Flavor	1.5719	3.68e-9

Table 5:

	Estimate	Pr(>|t|)
(Intercept)	12.9916	1.4e-7
Oakiness	-0.1304	0.779

Table 6:

	Estimate	Pr(>|t|)
(Intercept)	11.9765	2e-16
Region2	-1.5320	0.00757
Region3	2.6069	7.01e-6

Question 1.d From the summary of the full model we can reject the null hypothesis for the Flavor and Region predictors, meaning that this predictors are relevant in order to predict Quality.

Table 7: Full Model

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	7.8144	1.9694	3.97	0.0004
Clarity	0.0171	1.4563	0.01	0.9907
Aroma	0.0890	0.2525	0.35	0.7269
Body	0.0797	0.2677	0.30	0.7681
Flavor	1.1172	0.2403	4.65	6.25e-5
Oakiness	-0.3464	0.2330	-1.49	0.1475
Region2	-1.5129	0.3923	-3.86	0.0006
Region3	0.9726	0.5102	1.91	0.0662

Table 8: Resulting Model

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	7.0943	0.7912	8.97	1.76e-10
Flavor	1.1155	0.1738	6.42	2.49e-07
Region2	-1.5335	0.3688	-4.16	0.0002
Region3	1.2234	0.4003	3.06	0.0043

Question 1.e To start building our model, we will proceed to drop one variable at a time from the full model starting from the highest p-value predictor and rechecking the test hypothesis results for every new model until

all predictors remaining are relevant for the response. After this process we ended up with the model Quality ~Flavor + Region (Table 8). The partial F-test (Table 9) between the full model and the current model confirms our findings. With a p-value of 0.6528, we fail to reject the null hypothesis, this is all extra predictors are 0.

Table 9: Anova Table vs Full Model

	Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
1	34	27.21				
2	30	25.14	4	2.07	0.62	0.6528

Table 10: Anova Table vs Interactions

	Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
1	34	27.21				
2	32	25.43	2	1.78	1.12	0.3378

Now we will proceed to explore interactions between these two predictors. We will compare the resulting model with the interactions model Quality ~Flavor + Region + Flavor:Region. By carrying out a partial F-test test we obtained a p-value of 0.3378 (Table 10), meaning that we fail to reject the null hypothesis, therefore the interactions between Flavor and Region are not meaningful for our model.

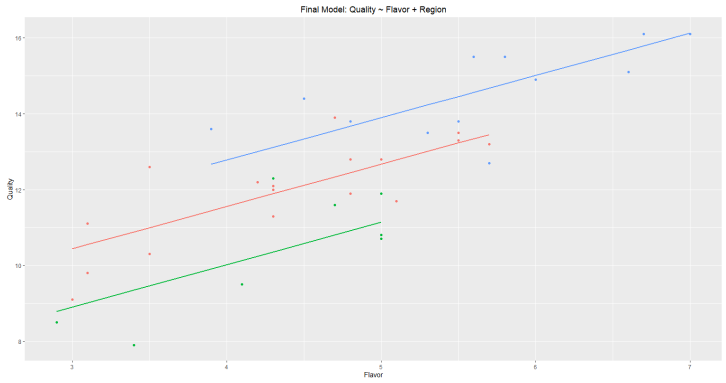

Figure 7: Final Model

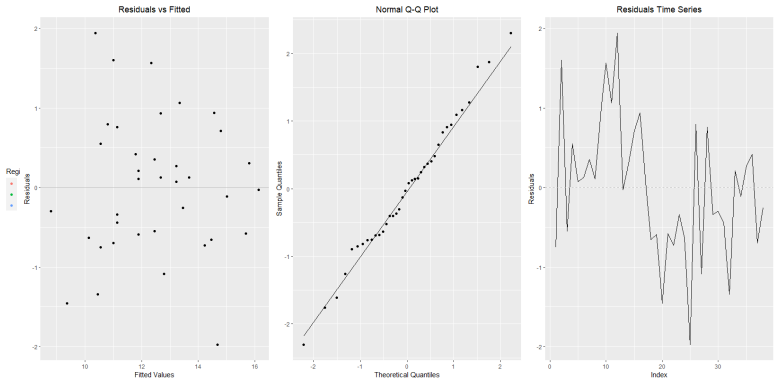

Figure 8: Model Assumptions

We can appreciate the three resulting regression lines of our model for each region on Figure 7. Clearly our model captures the linear trend on each region. Also we present several diagnostics plots on Figure 8 in order to check for model assumptions. We can see in the Residuals vs Fitted plot how the points are scattered around zero and there is no a clear pattern, verifying that our errors have an approximate zero mean and constant variance. For the Q-Q Plot, observe that there is not a significant violation of the normality assumption, being the data not seriously separated from a straight line. Finally checking on the Residuals Time Series, there is not a clear dependence of the residuals, verifying the independence assumption.

Question 1.f Final model: Quality = 7.0943 + 1.1155(Flavor) -1.53351$\mathbb{1}_{(Region2)}$ + 1.22341$\mathbb{1}_{(Region3)}$
Where $\mathbb{1}$ is an indicator function. We can interpret our model as follows: Approximately every unit change in Flavor will directly change the Quality by one unit. Region 1 wines has approximately a base quality of 7, Region 3 offers an additional unit (quality base around 8) and Region 2 a decrease of 1.5 (quality base approximately of 5.5).

Question 1.g Having an average Flavor wine from our sample and assuming Region 1, our prediction of Quality is: 12.41371. We obtained a 95% confidence interval [11.95152, 12.8759], this is that given an average Flavor wine and assuming it is from Region 1, 95% of intervals of this form will contain the true value of the wine Quality. Also we obtained a 95% prediction interval [10.53775, 14.28967], meaning that we can be 95% confident that the Quality of an average Flavor wine from Region 1 will lie in this range.

Question 2.a On Figure 9 we present several plots to explore how GMAT and GPA can help to predict the group of an applicant. We can observe in the graph of GMAT vs GPA, how the GPA predictor can clearly separate the three groups. Moreover in the GPA boxplot we have a clear distribution of the GPA by group, evidence of how much this predictor contributes to identify the correct group. For the GMAT boxplot there is some confusion for applicants of group 2 and 3, but a clear distinction for applicants of group 1, meaning that GMAT also helps us to discern the possible group of an applicant.

Question 2.b We fitted an LDA model for the training set and superimposed the decision boundary on these data (Figure 10). There is misclassification between the boundary of group 1 and 3 and the boundary of 2 and 3. With the amount of data used, we consider the decision boundary is sensible for new data points. We present the confusion matrix for training and testing sets on Tables 11 and 12. The misclassification error rate for training set is around 8.5% while it jumps to 20% for the testing set confirming the sensibility of our decision boundary.

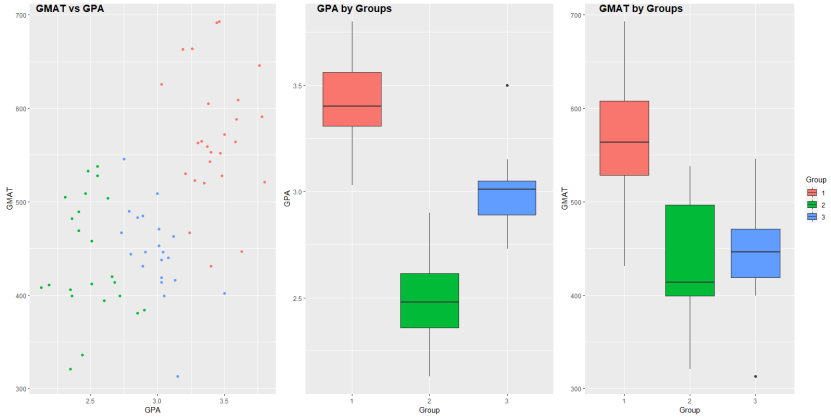

Figure 9:

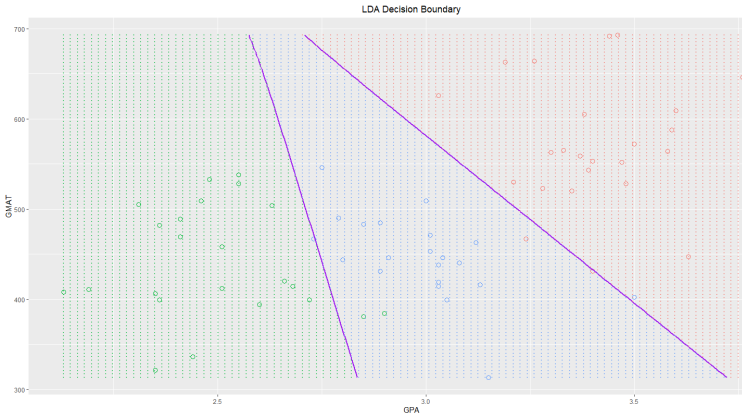

Figure 10: LDA Decision Boundary

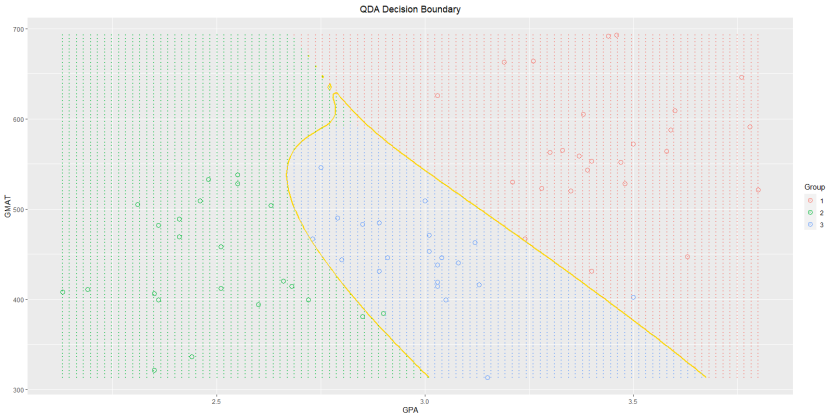

Figure 11: QDA Decision Boundary

Question 2.c We fitted a QDA model for the training set and superimposed the decision boundary on these data (Figure 11). We can see how QDA manage to correctly classify the challenging observations that LDA missed. The QDA decision boundary is less sensible than LDA's. We present the confusion matrix for training and testing sets on Tables 13 and 14. The misclassification error rate is around 2.8% and 6.7% for training and testing set respectively. We concluded that QDA decision boundary is not as much sensible to new data points.

Table 11: LDA Training

		Actual		
		1	2	3
	1	24	0	1
	2	0	21	1
Predicted	3	2	2	19

Class Error Rate: 0.08571

Table 12: LDA Testing

		Actual		
		1	2	3
	1	2	0	0
	2	0	5	0
Predicted	3	3	0	5

Class Error Rate: 0.2

Table 13: QDA Training

		Actual		
		1	2	3
	1	26	0	1
	2	0	22	0
Predicted	3	0	1	20

Class Error Rate: 0.02857

Table 14: QDA Testing

		Actual		
		1	2	3
	1	4	0	0
	2	0	5	0
Predicted	3	1	0	5

Class Error Rate: 0.06667

Question 2.d We recommend QDA over LDA as the classifier for our data. QDA performed very well on both training and testing data and offered a less sensible decision boundary than LDA.

Question 3.a The data is unbalanced having 1316 observations labelled as 0 (non diabetic) and 684 as 1(diabetic). This fact can cause that classification algorithms may have a decent accuracy at the cost of a low sensitivity (setting 1 as the positive response). We explored several boxplots to investigate how our predictors affect the classes (Figure 12, Figure 13). Predictors Glucose and Age separates the two classes in some degree (Figure 14).

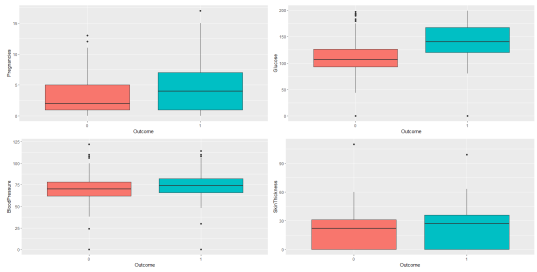

Figure 12:

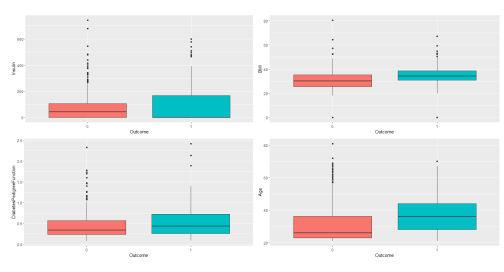

Figure 13:

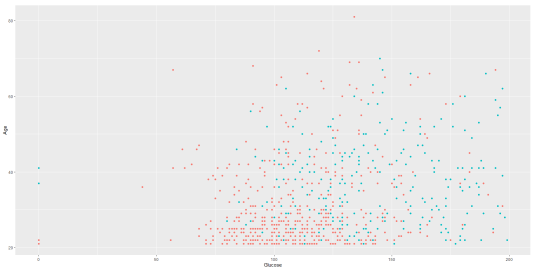

Figure 14: Age vs Glucose

Table 15: LDA Classifier

	Actual		
		0	1
Predicted	0	1174	298
	1	142	386

Table 16: QDA Classifier

	Actual		
		0	1
Predicted	0	1135	290
	1	181	394

Table 17:

	Error.Rate	Sensitivity	Specificity
LDA	0.22	0.5643275	0.8920973
QDA	0.2355	0.5760234	0.8624620

Question 3.b and Question 3.c The confusion matrix for LDA and QDA are presented on Tables 15 and 16 respectively. On Table 17 we present the sensitivity, specificity, and overall misclassification rate of both methods. We can observed that even if both methods had an error close to 20%, both exhibit a low sensitivity, this fact is caused by the unbalanced of both classes as we pointed out before. The ROC graphs for both methods are very similar. Both methods perform significantly better than a random guess for all possible cutoffs.

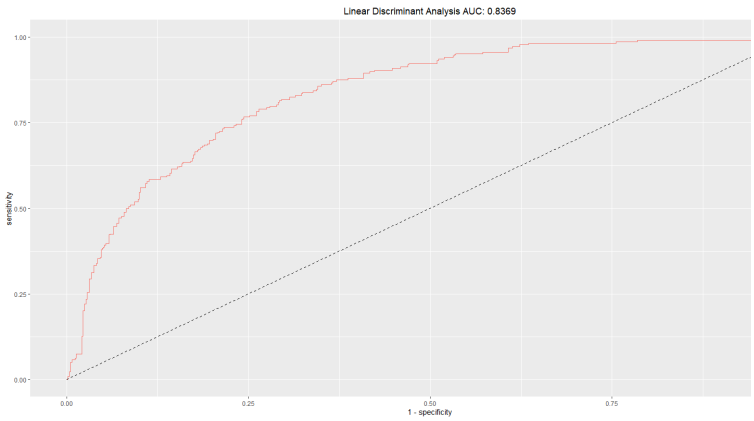

Figure 15: ROC LDA

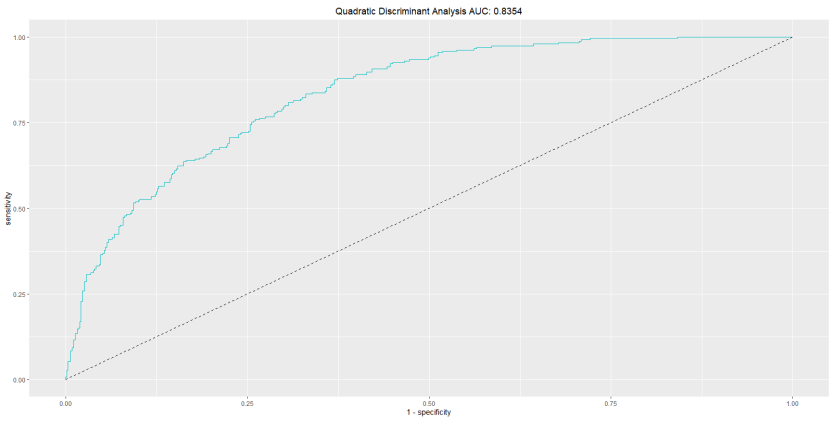

Figure 16: ROC QDA

Question 3.d We recommend the use of the LDA classifier, since it has a slightly greater AUC of .8369 vs .8354 of QDA and also an error rate of 22% inferior to the 23.5% of QDA. We would like to improve the sensitivity without sacrifying much specificity and overall accuracy. So we will find a cutoff that gives the best trade-off between sensitivity and specificity using Youden's J statistic. This cutoff maximizes the value of J = sensitivity + specificity - 1. Its geometrical interpretation is the maximum distance from the random guess which is a desirable result. The pROC package offers a way to automatically find this cutoff using the function coords(x,"best"). The cutoff obtained for this method is 0.2906709 and the associated confusion matrix is presented on Table 18 along with the error, sensitivity and specificity Table 19. We can see how our error slightly grew but we improved our sensitivity without giving away much specificity.

Table 18:

	Actual		
		0	1
Predicted	0	968	144
	1	348	540

Table 19:

	Error.Rate	Sensitivity	Specificity
New Cutoff	0.246	0.7894737	0.7355623

2 Code

```
#Packages
library(ggplot2) #Used for graphics and visual representations
library(GGally) #Used for visualization of pairs
library(ggpubr) #Used for graphics handling
library(ggcorrplot) #Used for visualization of correlations


library(fastDummies) #Used for One hot encoding categorical data
library(MASS) #Used for LD and QD analysis
library(pROC) #Used to obtain ROC curve and find cutoff
library(caret) #Used to obtain Confusion matrix and classification metrics


rdseed=8467 #Seed to replicate results in case of a tie on LDA or QDA

#Experiment 1
print("Experiment 1")

#Reading the data
raw_data=read.delim("wine.txt")

wine=raw_data

#Converting Region to factor
wine$Region=as.factor(wine$Region)

####Question 1.a####

#Exploring correlations

#We want now to visualize correlations.
#Since regions is a factor variable, to be able to find the correlations with it
#we need to one hot encode the data.
#We proceed to one hot encode Region leaving one class out.
#We manipulate a copy of wine for this purpose.

wine_dummy=wine
#One hot encode Region leaving one class out
wine_dummy=dummy_cols(wine_dummy, select_columns = "Region",
                      remove_first_dummy = TRUE)
wine_dummy$Region=NULL #dropping the Region factors

#Placing Quality as last variable
temp=wine_dummy$Quality
wine_dummy$Quality=NULL
wine_dummy$Quality=temp

#Finding correlation matrix
corr1 = round(cor(wine_dummy), 2)

print(ggcorrplot(corr1,lab=TRUE,type = "full"))

#Exploring scatterplots
print(ggpairs(wine[,1:6], upper=list(continuous="cor"),axisLabels="internal"))
#As expected from the correlation matrix we can notice that the predictors
#Aroma, Body and Flavor exhibit a positive linear relation with Quality.

#Exploring distribution by region
g_region=ggplot(wine,aes(Region,Quality,fill=Region))+geom_boxplot()
print(g_region)
```

```
####Question 1.b####

#Exploring boxplot of Quality to identify possible outliers
print(ggplot(wine ,aes(y=Quality))+
        geom_boxplot(fill="gray64", color="black",
                     outlier.colour = "red", outlier.shape = 1))
#No visible outliers detected

#Exploring Histogram
print(ggplot(wine, aes(x=Quality)) +
        geom_histogram(col="black",fill="salmon",breaks=7:17,
          position="identity", alpha=0.5))
#The quality distribution looks approximately normal.

#At this moment we consider that transformation is not required and
#we will check model assumptions once we find an appropriate model


####Question 1.c####

#Fitting one simple regression model for each predictor
m_Clarity=lm(Quality~Clarity,data=wine)

m_Aroma=lm(Quality~Aroma,data=wine)

m_Body=lm(Quality~Body,data=wine)

m_Flavor=lm(Quality~Flavor,data=wine)

m_Oakiness=lm(Quality~Oakiness,data=wine)

m_Region=lm(Quality~Region,data=wine)

summary(m_Clarity)
summary(m_Aroma)
summary(m_Body)
summary(m_Flavor)
summary(m_Oakiness)
summary(m_Region)

#Visualizing some models
g_Aroma=ggplot(m_Aroma,aes(x=Aroma,y=Quality))+geom_point()+
  geom_line(aes(y = .fitted),size=1,color="green")

g_Body=ggplot(m_Body,aes(x=Body,y=Quality))+geom_point()+
  theme(axis.title.y = element_blank())+
  geom_line(aes(y = .fitted),size=1,color="blue")

g_Flavor=ggplot(m_Flavor,aes(x=Flavor,y=Quality))+geom_point()+
  theme(axis.title.y = element_blank())+
  geom_line(aes(y = .fitted),size=1,color="red")

#Visualizing significant continuous predictors
print(ggarrange(g_Aroma, g_Body, g_Flavor,
      labels = c("Quality vs Aroma", "Quality vs Body", "Quality vs Flavor"),
      ncol = 3, nrow = 1))


####Question 1.d####

#Fitting the full model
m_Full=lm(Quality~.,data=wine)
```

```r
summary(m_Full)


####Question 1.e####

#Dropping one predictor at a time
m1=lm(Quality~.-Clarity,data=wine) #dropping Clarity
summary(m1)

m2=lm(Quality~.-Clarity-Body,data=wine) #dropping Clarity and Body
summary(m2)

m3=lm(Quality~.-Clarity-Body-Aroma,data=wine) #dropping Clarity, Body and Aroma
summary(m3)

m4=lm(Quality~.-Clarity-Body-Aroma-Oakiness,data=wine) #dropping Clarity,
                                                       #Body, Aroma and Oakiness
summary(m4)

#Finally on m4 all predictors are statistically significant
#for the response variable

print(anova(m4,m_Full))


#Exploring interactions
m_final=lm(Quality~Flavor + Region,data=wine)
summary(m_final)
m_final2=lm(Quality~Flavor + Region + Flavor:Region,data=wine)
summary(m_final2)

print(anova(m_final,m_final2))
#With a p-value of 0.3378, we fail to reject the null hypothesis,
#this is all additional predictors are 0, meaning that
#the interactions between Flavor and Region are not meaningful for our model


#We can appreciate the lines of our final model for each region.
print(ggplot(m_final, aes(Flavor, Quality,color=Region)) +
  geom_point()+geom_line(aes(y = .fitted),size=1, show.legend =F)+
  theme(plot.title=element_text(hjust=0.5))+
  ggtitle("Final Model: Quality ~ Flavor + Region"))


#Verifying model assumptions:

#Residual plot
gR=ggplot(m_final,aes(m_final$fitted.values, m_final$residuals)) +
  geom_point() +
  geom_hline(yintercept=0,color="gray")+
  theme(plot.title=element_text(hjust=0.5))+
  ggtitle("Residuals vs Fitted")+
  labs(y="Residuals", x = "Fitted Values")

#QQ plot
gQ=ggplot(m_final, aes(sample = .stdresid))+ stat_qq() + stat_qq_line()+
  theme(plot.title=element_text(hjust=0.5))+
  ggtitle("Normal Q-Q Plot")+
  labs(y="Sample Quantiles", x = "Theoretical Quantiles")


#Time series plot of residuals
#Data Frame to plot Residuals Time Series
```

```
df.resid=data.frame(Index=1:nrow(wine),Residuals=m_final$residuals)

gT=ggplot(df.resid,aes(Index,Residuals))+geom_line()+
  geom_hline(yintercept=0,color="gray",linetype="dashed")+
  theme(plot.title=element_text(hjust=0.5))+
  ggtitle("Residuals Time Series")

print(ggarrange(gR, gQ, gT,
                ncol = 3, nrow = 1))

####Question 1.f####

print(m_final)



####Question 1.g####

#Predicted for mean Flavor and Region 1
predicted=predict(m_final,data.frame(Flavor=mean(wine$Flavor),
                                     Region=as.factor(1)))

#95% prediction intervals
print(predict (m_final ,data.frame(Flavor=mean(wine$Flavor),
                                   Region=as.factor(1)),
               interval ="prediction"))

#95% confidence intervals
print(predict (m_final ,data.frame(Flavor=mean(wine$Flavor)
                                   ,Region=as.factor(1)),
               interval ="confidence"))


##########################

#Experiment 2
print("Experiment 2")

#Reading the data
admission = read.csv("admission.csv")
admission$Group=as.factor(admission$Group)

#Building training and testing sets
k1=which(admission$Group==1)
k2=which(admission$Group==2)
k3=which(admission$Group==3)

adm.test=rbind(admission[k1,][1:5,],
               admission[k2,][1:5,],
               admission[k3,][1:5,])

adm.train=rbind(admission[k1,][-(1:5),],
                admission[k2,][-(1:5),],
                admission[k3,][-(1:5),])


####Question 2.a####

#Exploring importance of predictors for response

#Scatterplot
g=ggplot(adm.train,aes(x=GPA,y=GMAT,color=Group))+geom_point()
```

```r
#Boxplot of GPA by groups
g_GPA=ggplot(adm.train,aes(Group,GPA,fill=Group))+geom_boxplot()+
  theme(legend.position = "none")

#Boxplot of GMAT by groups
g_GMAT=ggplot(adm.train,aes(Group,GMAT,fill=Group))+geom_boxplot()

print(ggarrange(g+theme(legend.position = "none"),g_GPA, g_GMAT,
          labels = c("GMAT vs GPA","GPA by Groups", "GMAT by Groups"),
          ncol = 3, nrow = 1))



####Question 2.b####

#Fitting LDA
mlda = lda(Group ~ GPA + GMAT, data = adm.train)

#Representing the decision boundary over the Training Set
#Creating grid
x1=seq(min(adm.train[,1]),max(adm.train[,1]),length.out=100)
x2=seq(min(adm.train[,2]),max(adm.train[,2]),length.out=100)
grid = expand.grid(x=x1, y=x2)

#Classifying the grid
names(grid)=names(adm.train)[1:2]
grid.pred.lda = predict(mlda, grid)

prob = grid.pred.lda$posterior
prob12=pmax(prob[,1],prob[,2]) #using maximum probabilities over Group 1 and 2

#Data Frame to generate the surface for the contour
df_contour.lda=data.frame(x=grid[,1],y=grid[,2],z=prob12)

#Adding the classes to the predicted classes to the grid
grid$Group=grid.pred.lda$class

#Plotting the training set and decision boundary
glda=ggplot(adm.train,aes(x=GPA,y=GMAT,color=Group))+
  geom_point(aes(x=GPA, y=GMAT, color=Group), size=3, shape=1)+
  geom_contour(aes(x=x,y=y,z=z), data=df_contour.lda,size=1,
               color="purple",breaks = 0.5)+
  geom_point(aes(x=GPA, y=GMAT, color=Group),data=grid,alpha=0.5,size=0.5)+
  theme(plot.title=element_text(hjust=0.5))+
  ggtitle("LDA Decision Boundary")
print(glda)

#Predicted classes for Training Set
pred_train_lda=predict(mlda, adm.train)$class

#LDA Confusion Matrix for Training Set
M1=table(pred_train_lda,adm.train$Group,dnn=c("predicted","actual"))
print(M1)
acc_train_lda=sum(diag(M1))/sum(M1)
print(paste("Misclassifcation Rate Training LDA:",1-acc_train_lda))

#Predicted classes for Testing Set
pred_test_lda=predict(mlda, adm.test)$class

#LDA Confusion Matrix for Testing Set
M2=table(pred_test_lda,adm.test$Group,dnn=c("predicted","actual"))
print(M2)
acc_test_lda=sum(diag(M2))/sum(M2)
```

```
print(paste("Misclassifcation Rate Test LDA:",1-acc_test_lda))


####Question 2.c####

#Fitting QDA
mqda = qda(Group ~ GPA + GMAT, data = adm.train)

#Representing the decision boundary over the Training Set
#Creating grid
grid.pred.qda = predict(mqda, grid)

prob = grid.pred.qda$posterior
prob12=pmax(prob[,1],prob[,2]) #using maximum probabilities over Group 1 and 2

#Data Frame to generate the surface for the contour
df_contour.qda=data.frame(x=grid[,1],y=grid[,2],z=prob12)

#Adding the classes to the predicted classes to the grid
grid$Group=grid.pred.qda$class

#Plotting the training set and decision boundary
gqda=ggplot(adm.train,aes(x=GPA,y=GMAT,color=Group))+
  geom_point(aes(x=GPA, y=GMAT, color=Group), size=3, shape=1)+
  geom_contour(aes(x=x,y=y,z=z), data=df_contour.qda,size=1,
               color="gold1",breaks = 0.5)+
  geom_point(aes(x=GPA, y=GMAT, color=Group),data=grid,alpha=0.5,size=0.5)+
  theme(plot.title=element_text(hjust=0.5))+
  ggtitle("QDA Decision Boundary")
print(gqda)

#Predicted classes for Training Set
pred_train_qda=predict(mqda, adm.train)$class

#QDA Confusion Matrix for Training Set
M3=table(pred_train_qda,adm.train$Group,dnn=c("predicted","actual"))
print(M3)
acc_train_qda=sum(diag(M3))/sum(M3)
print(paste("Misclassifcation Rate Training QDA:",1-acc_train_qda))

#Predicted classes for Testing Set
pred_test_qda=predict(mqda, adm.test)$class

#QDA Confusion Matrix for Testing Set
M4=table(pred_test_qda,adm.test$Group,dnn=c("predicted","actual"))
print(M4)
acc_test_qda=sum(diag(M4))/sum(M4)
print(paste("Misclassifcation Rate Testing LDA:",1-acc_test_qda))


####Question 2.d####
#Included in report


#########################


#Experiment 3

print("Experiment 3")

#Reading the data
diabetes = read.csv("diabetes.csv")
```

```r
#Renaming predictors
names(diabetes)=c("Pregnancies", "Glucose", "BloodPressure", "SkinThickness",
                  "Insulin",   "BMI",
                  "DiabetesPedigreeFunction", "Age","Outcome" )

#Converting Outcome to factor
diabetes$Outcome=as.factor(diabetes$Outcome)


####Question 3.a####

#Exploring data
print(table(diabetes$Outcome) )
#We can notice how the data is unbalanced for our classes

#Visualizing Boxplots of several predictors
g1=ggplot(diabetes,aes(Outcome,Pregnancies,fill=Outcome))+geom_boxplot()+
  theme(legend.position = "none")
g2=ggplot(diabetes,aes(Outcome,Glucose,fill=Outcome))+geom_boxplot()+
  theme(legend.position = "none")
g3=ggplot(diabetes,aes(Outcome,BloodPressure,fill=Outcome))+geom_boxplot()+
  theme(legend.position = "none")
g4=ggplot(diabetes,aes(Outcome,SkinThickness,fill=Outcome))+geom_boxplot()+
  theme(legend.position = "none")
print(ggarrange(g1, g2, g3, g4, ncol = 2, nrow = 2))

g5=ggplot(diabetes,aes(Outcome,Insulin,fill=Outcome))+geom_boxplot()+
  theme(legend.position = "none")
g6=ggplot(diabetes,aes(Outcome,BMI,fill=Outcome))+geom_boxplot()+
  theme(legend.position = "none")
g7=ggplot(diabetes,aes(Outcome,DiabetesPedigreeFunction,fill=Outcome))+
  geom_boxplot()+  theme(legend.position = "none")
g8=ggplot(diabetes,aes(Outcome,Age,fill=Outcome))+geom_boxplot()+
  theme(legend.position = "none")
print(ggarrange(g5, g6, g7, g8, ncol = 2, nrow = 2))

#From the previous boxplots we consider Glucose and Age are relevant predictors
print(ggplot(diabetes,aes(x=Glucose,y=Age,color=Outcome))+geom_point()+
        theme(legend.position = "none"))


#Storing true classes
actual=diabetes$Outcome

####Question 3.b####

#Fitting LDA
m_diabetes_lda=lda(Outcome~., data=diabetes)

#Predicted Classes
predicted_lda=predict(m_diabetes_lda,diabetes)

#Confusion Matrix for LDA
CM1=confusionMatrix(predicted_lda$class, actual, positive="1")
print(CM1$table)


#ROC Curve for LDA
roc.lda = roc(diabetes$Outcome, predicted_lda$posterior[, "1"],
              levels = c("0", "1"),direction="<")
```

```r
print(ggroc(roc.lda,color = "salmon"  ,legacy.axes = TRUE)+
  geom_segment(aes(x = 0, xend = 1, y = 0, yend = 1), color="black",
               linetype="dashed")+ theme(plot.title=element_text(hjust=0.5))+
  ggtitle(paste("Linear Discriminant Analysis AUC:",round(roc.lda$auc,4))))

####Question 3.c####

#Fitting QDA
m_diabetes_qda=qda(Outcome~., data=diabetes)

#Predicted Classes
predicted_qda=predict(m_diabetes_qda,diabetes)

#Confusion Matrix for QDA
CM2=confusionMatrix(predicted_qda$class, actual, positive="1")
print(CM2$table)


#ROC Curve for QDA
roc.qda = roc(diabetes$Outcome, predicted_qda$posterior[, "1"],
              levels = c("0", "1"),direction="<")


print(ggroc(roc.qda, color="turquoise3",legacy.axes = TRUE)+
  geom_segment(aes(x = 0, xend = 1, y = 0, yend = 1), color="black",
               linetype="dashed")+ theme(plot.title=element_text(hjust=0.5))+
    ggtitle(paste("Quadratic Discriminant Analysis AUC:",round(roc.qda$auc,4))))


#Visualizing Error, Sensitivity and Specificity of both models.
df.tab=data.frame("Error Rate" =c(1-CM1$overall[["Accuracy"]],
                                  1-CM2$overall[["Accuracy"]]),
                  "Sensitivity"=c(CM1$byClass[["Sensitivity"]],
                                  CM2$byClass[["Sensitivity"]]),
                  "Specificity"=c(CM1$byClass[["Specificity"]],
                                  CM2$byClass[["Specificity"]]))
row.names(df.tab)=c("LDA","QDA")

print(df.tab)

####Question 3.d####

#Finding new cutoff
cut.lda=coords(roc.lda,"best") #best cutoff determine by Youden's Index

#Classifying using the new Cutoff
pred_newcutoff=ifelse(predicted_lda$posterior[,2]>=cut.lda[1,1],"1","0")
pred_newcutoff=as.factor(pred_newcutoff)

#New Confusion Matrix
CM3=confusionMatrix(pred_newcutoff, actual, positive="1")
print(CM3$table)

df.tab.new=data.frame("Error Rate" =1-CM3$overall[["Accuracy"]],
                      "Sensitivity"=CM3$byClass[["Sensitivity"]],
                      "Specificity"=CM3$byClass[["Specificity"]])
row.names(df.tab.new)="New Cutoff"

print(df.tab.new)
```