Final Exam

Ryan Greenup 1780515

24/10/2019

Contents

Preamble

In order to load the packages etc use the following code:

```
knitr::opts_chunk$set(echo = TRUE)
# Load Packages
if(require('pacman')){
  library('pacman')
}else{
  install.packages('pacman')
  library('pacman')
}
```

```
## Loading required package: pacman
```

```
pacman::p_load(caret, scales, ggplot2, rmarkdown, shiny,
    ISLR, class, BiocManager,
              corrplot, plotly, tidyverse, latex2exp,
                  stringr, reshape2, cowplot, ggpubr,
```

```
                rstudioapi, wesanderson, RColorBrewer,
                    colorspace, gridExtra, grid, car,
                boot, colourpicker, tree, ggtree, mise,
                    rpart, rpart.plot, knitr, MASS,
                magrittr,
                    EnvStats,tidyverse,tidyr,devtools,
                    bookdown, leaps, car, clipr,
                tikzDevice, e1071, ggbiplot, base)
#install.packages("ggbiplot")

mise()
```

```
set.seed(23)


knitTable <- function(table){
if(isTRUE(getOption('knitr.in.progress'))){
  kable(table)
} else {
  table
  }
}
```

Import the DataSet

In order to import the dataset:

```
glassDF <- read.csv(file = "./dataset/glass(1).csv")
wineDF <- read.csv(file =
    "./dataset/Wine_Quality(3).csv")
winedf <- read.csv(file =
    "./dataset/Wine_Quality(3).csv")

knitTable(head(winedf))
```

WineQuality	FixedAcidity	VolatileAcidity	CitricAcid	ResidualSugar	Chlorides	FreeSulfurDioxide	TotalS
Low	7.9	0.18	0.37	1.2	0.040	16	
Low	6.5	0.31	0.14	7.5	0.044	34	
High	6.2	0.66	0.48	1.2	0.029	29	
High	6.4	0.31	0.38	2.9	0.038	19	
High	7.0	0.28	0.39	8.7	0.051	32	
High	7.0	0.32	0.34	1.3	0.042	20	

```
knitTable(head(glassDF))
```

RI	Na	Mg	Al	Si	K	Ca	Ba	Fe	Type
1.52101	13.64	4.49	1.10	71.78	0.06	8.75	0	0.00	1
1.51761	13.89	3.60	1.36	72.73	0.48	7.83	0	0.00	1
1.51618	13.53	3.55	1.54	72.99	0.39	7.78	0	0.00	1
1.51766	13.21	3.69	1.29	72.61	0.57	8.22	0	0.00	1
1.51742	13.27	3.62	1.24	73.08	0.55	8.07	0	0.00	1
1.51596	12.79	3.61	1.62	72.97	0.64	8.07	0	0.26	1

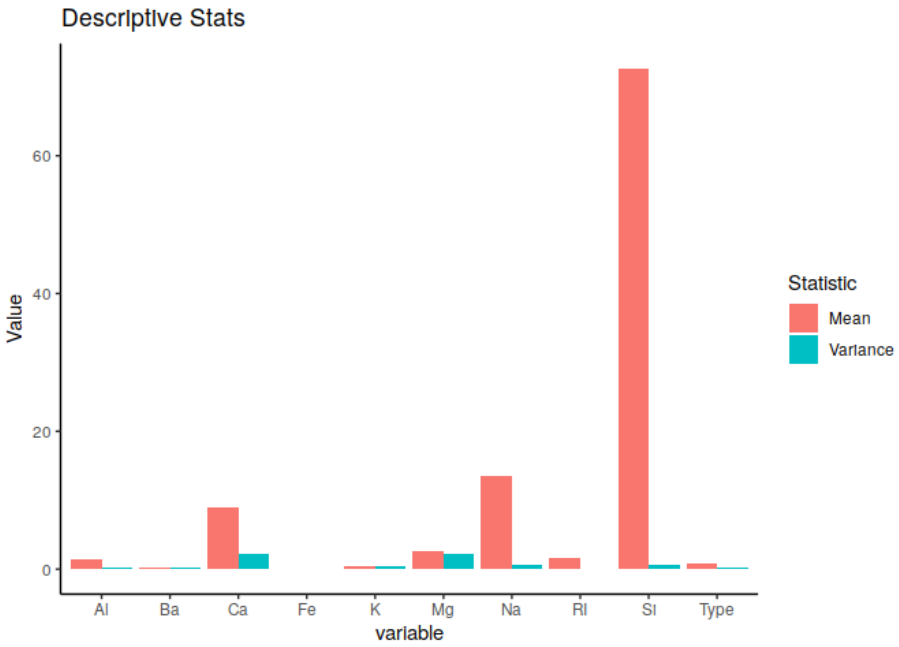
Question 1

This Question uses the data set "glass.csv". The data represents the weight percent in corresponding oxides and refractive index together with the type of glass (window glasses and non-window glasses).

Sub Question A

Calculate the mean and the variance for each appropriate variable and discuss if scaling is necessary and justify your findings. The mean and variance can be calculated by using `apply`:

```
desc.stats <- data.frame(
Mean = apply(glassDF, 2, mean), # 1 is rows, 2 is cols
    p. 401 ISL TB
Variance = apply(glassDF, 2, var) # 1 is rows, 2 is cols
    p. 401 ISL TB
)
desc.stats$variable <- row.names(desc.stats)

descStatsTidy <- pivot_longer(desc.stats, cols = c(Mean,
    Variance), names_to = "Statistic", values_to =
    "Value")

ggplot(descStatsTidy, aes(x = variable, y = Value, fill
    = Statistic)) +
  geom_col(position = "dodge") +
  theme_classic() +
  labs(title = "Descriptive Stats")
```



This clearly indicates that the silicon mean value is extremely high for this reason sensible/useful results will only come from first scaling the data.

Sub Question B

Apply scaling and derive the principal components. R code and output should be clearly stated. The type of glass may be more appropriately interpreted as a response variable, moreover it is non-continuous and for this reason it will be removed from the data frame before applying PCA:

```
glassFeat <- subset(glassDF, select = -c(Type))

pcaMod <- prcomp(glassFeat, scale = TRUE)
pcaMod
```

```
## Standard deviations (1, .., p=9):
## [1] 1.58993664 1.42810241 1.19298870 1.06306123
    0.95576080 0.72682711
## [7] 0.61052378 0.25147486 0.03960487
##
## Rotation (n x k) = (9 x 9):
```

```
##           PC1         PC2         PC3         PC4
    PC5         PC6
## RI -0.5394788 0.29394585 -0.08638006 -0.1549780
    0.08044391 -0.10673760
## Na  0.2819860 0.25518026 0.35361026 -0.5105281
    -0.13071871 0.56510345
## Mg -0.1106014 -0.59664310 -0.03091599 -0.3843663
    -0.09975186 -0.30607443
## Al  0.4247159 0.29235315 -0.32956973 0.1536455
    -0.02629504 0.01496627
## Si  0.2223203 -0.14866779 0.49772743 0.6286031
    -0.03402182 -0.09597319
## K   0.2078198 -0.15369150 -0.66087820 0.1106703
    0.29407686 0.24601727
## Ca -0.4897128 0.35497736 0.02593283 0.2692515
    0.17392671 0.14891254
## Ba  0.2538890 0.48048046 -0.08484292 -0.1451014
    -0.25520134 -0.65128708
## Fe -0.1998413 -0.06657274 -0.25605242 0.2075040
    -0.88475004 0.24346351
##            PC7         PC8         PC9
## RI -0.08366256 -0.75203415 -0.02568729
## Na -0.14843605 -0.12617171 0.30871663
## Mg  0.20213519 -0.08050737 0.57629768
## Al  0.70048221 -0.27351280 0.18847508
## Si -0.20442143 -0.38169850 0.29622098
## K  -0.50570422 -0.11221598 0.26238721
## Ca  0.10368207 0.39760944 0.58286778
## Ba -0.35601156 0.14324828 0.19964786
## Fe -0.06643995 -0.01830694 0.01429302
```

Sub Question C

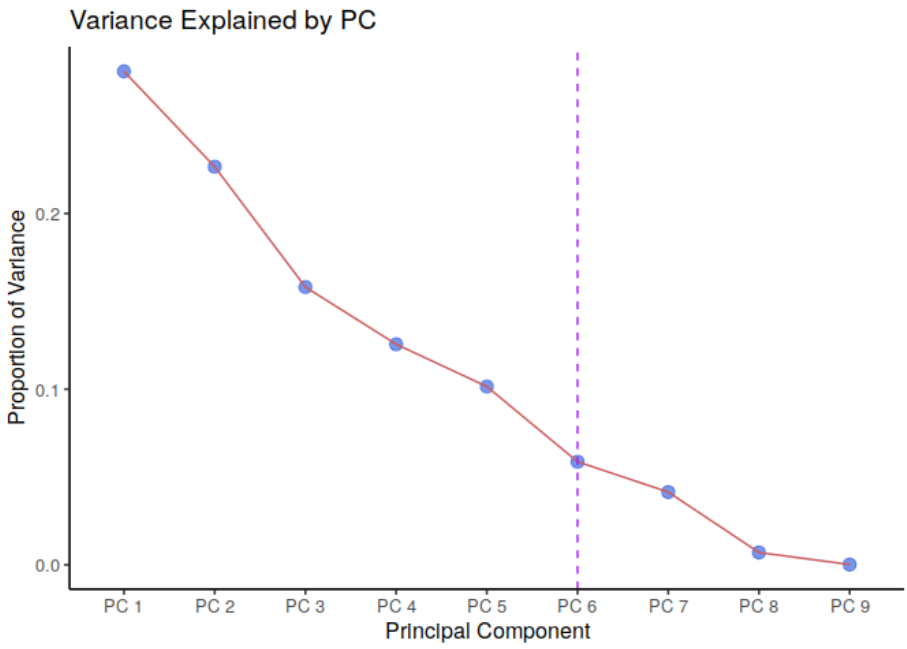
c. Explain the percentage variation captured by each principal component. Use and give relevant R output and support your findings.

Plot of Proportion of Variance

The amount of variance explained by each component can be explained by using a scree plot:

```
pcaVar <- pcaMod$sdev^2
pcaVarpr <- pcaVar/sum(pcaVar)
pcaVarpr <- enframe(pcaVarpr)
# pcaVarpr <- dplyr::rename(pcaVarpr,
```

```
#                           "Principal.Component" = name,
#                           "Proportion.Variance" = value
#                           )

names(pcaVarpr) <- c("Principal.Component",
    "Proportion.Variance") # This gives a warning

for (i in 1:nrow(pcaVarpr)) {
  pcaVarpr[["Principal.Component"]][i] <- paste("PC", i)
}



ggplot(data = pcaVarpr, aes( x = Principal.Component, y
    = Proportion.Variance, group = 1)) +
  geom_point(size = 3, alpha = 0.7, col = "RoyalBlue") +
  geom_line(col = "IndianRed") +
  labs(x = "Principal Component", y = "Proportion of
      Variance", title = "Variance Explained by PC") +
  theme_classic2() +
  geom_vline(xintercept = 6, col = "purple", lty = 2)
```




```
print(pcaVarpr, digits = 1)
```

```
##   Principal.Component Proportion.Variance
## 1                PC 1              3e-01
## 2                PC 2              2e-01
## 3                PC 3              2e-01
## 4                PC 4              1e-01
## 5                PC 5              1e-01
## 6                PC 6              6e-02
## 7                PC 7              4e-02
## 8                PC 8              7e-03
## 9                PC 9              2e-04
```

Table Of Proportion of Variance

A table of the proportion of variance values corresponding to each principal component can be returned by printing the created dataframe:

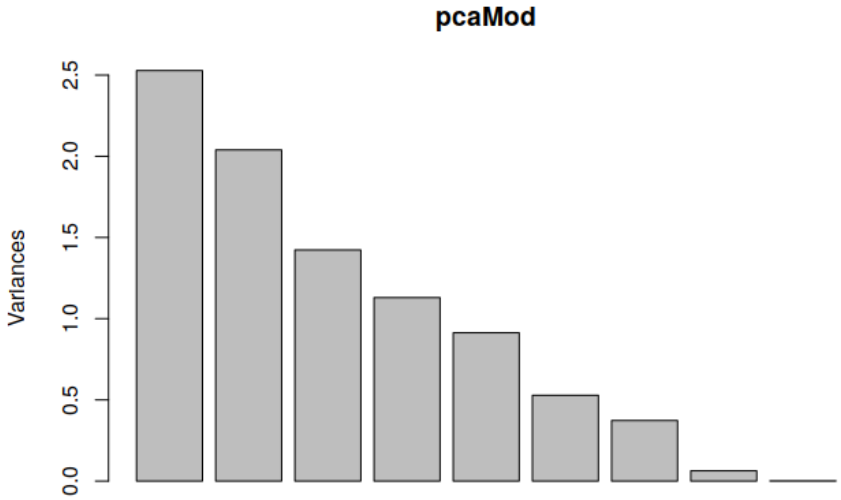
```
pcaVarpr %>% knitTable()
```

Principal.Component	Proportion.Variance
PC 1	0.2808776
PC 2	0.2266085
PC 3	0.1581358
PC 4	0.1255666
PC 5	0.1014976
PC 6	0.0586975
PC 7	0.0414155
PC 8	0.0070266
PC 9	0.0001743

Base Scree Plot

The scree plot could also be produced by using:

```
screeplot(pcaMod)
```



Summary of PCA Model

```
summary(pcaMod)
```

```
## Importance of components:
##                           PC1    PC2    PC3    PC4    PC5
    PC6     PC7
## Standard deviation   1.5899 1.4281 1.1930 1.0631
    0.9558 0.7268 0.61052
## Proportion of Variance 0.2809 0.2266 0.1581 0.1256
    0.1015 0.0587 0.04142
## Cumulative Proportion 0.2809 0.5075 0.6656 0.7912
    0.8927 0.9514 0.99280
##                            PC8     PC9
## Standard deviation   0.25147 0.03960
## Proportion of Variance 0.00703 0.00017
## Cumulative Proportion 0.99983 1.00000
```

Sub Question D

Write the first two principal components in terms of the original variables in the given dataset. The first two principal components may be expressed as a linear combination.

Consider the rotation matrix

```
print(pcaMod$rotation, digits = 2)
```

```
##      PC1    PC2    PC3   PC4    PC5    PC6    PC7    PC8
    PC9
## RI -0.54 0.294 -0.086 -0.15 0.080 -0.107 -0.084
    -0.752 -0.026
## Na  0.28  0.255  0.354 -0.51 -0.131 0.565 -0.148 -0.126
    0.309
## Mg -0.11 -0.597 -0.031 -0.38 -0.100 -0.306 0.202
    -0.081 0.576
## Al  0.42  0.292 -0.330 0.15 -0.026 0.015 0.700 -0.274
    0.188
## Si  0.22 -0.149 0.498 0.63 -0.034 -0.096 -0.204 -0.382
    0.296
## K   0.21 -0.154 -0.661 0.11 0.294 0.246 -0.506 -0.112
    0.262
## Ca -0.49 0.355 0.026  0.27  0.174  0.149  0.104  0.398
    0.583
## Ba  0.25  0.480 -0.085 -0.15 -0.255 -0.651 -0.356 0.143
    0.200
## Fe -0.20 -0.067 -0.256 0.21 -0.885 0.243 -0.066
    -0.018 0.014
```

```
names(glassDF)
```

```
##  [1] "RI"   "Na"   "Mg"   "Al"   "Si"   "K"    "Ca"   "Ba"
    "Fe"   "Type"
```

```
print(pcaMod$rotation[,1], digits = 2)
```

```
##    RI    Na    Mg    Al    Si     K    Ca    Ba    Fe
```

```
## -0.54 0.28 -0.11 0.42 0.22  0.21 -0.49 0.25 -0.20
```

```
print(glassDF[1,], digits = 2)
```

```
##    RI Na  Mg  Al Si   K  Ca Ba Fe Type
## 1 1.5 14 4.5 1.1 72 0.06 8.8 0 0  1
```

```
print(pcaMod$rotation[,2], digits = 2)
```

```
##    RI    Na    Mg    Al    Si     K    Ca    Ba
    Fe
## 0.294 0.255 -0.597 0.292 -0.149 -0.154 0.355 0.480
    -0.067
```

Express the Linear Combination

In terms of the dot product the first principal components may be expressed as Z_1 and Z_2 respectively, defined such that:

$$

PC2:

$$\mathsf{Z}_2 = \begin{bmatrix} \mathsf{Ri}_i & \mathsf{Na} & \mathsf{Mg}_i & \mathsf{Al}_i & \mathsf{Si}_i & \mathsf{K}_i & \mathsf{Ca}_i & \mathsf{Ba}_i & \mathsf{Fe}_i \end{bmatrix} \cdot \begin{bmatrix} 0.294 \\ 0.255 \\ -0.597 \\ 0.292 \\ -0.149 \\ -0.154 \\ 0.355 \\ 0.480 \\ -0.067 \end{bmatrix}$$

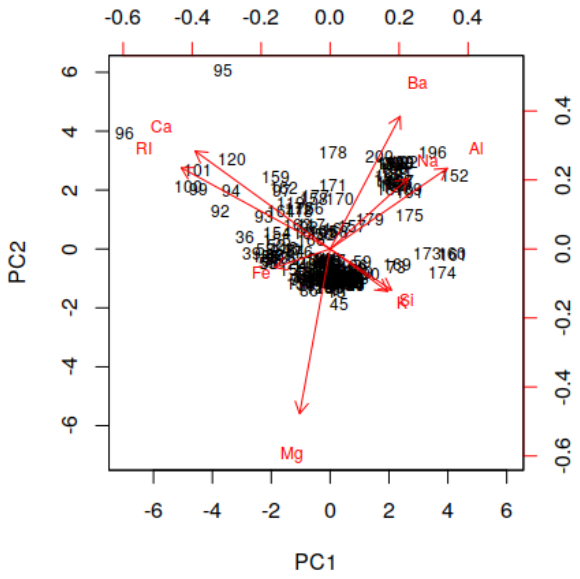
$$

Sub Question E

Using Biplot, explain the association between original variables in the dataset.

The Biplot may be produced thusly:

```
biplot(pcaMod, scale = 0, cex = 0.75)
```



From the bi plot it appears that Ba proides a high positive amout of the contribution to PC2, Mg contributes a very strong negative contribution to PC2. Ri and Ca provide equally to PC1 and PC2 in a negative and positive sense respectively. Ai and Na provide an equally positive contribution to both PC's. K and Si provide an equal amount of contribution to both PC1 and PC2, positively and negatively respectively.

It appears from the biplot that observation 96 and 95 have a high level of Ca and Ri while 196 and 152 have high Na, Ba and Al.

Question 2

This Question uses the data set "glass.csv" used in Question 1

Sub Question A

Use K Means Clustering method and identify two clusters with K=2.

Now in order to perform k-means clustering with $K = 2$ use the `kmeans` function; Be mindful to remove the response variable and the primary key:

```
km.out <- kmeans(glassFeat, 2, nstart = 20)
```

The assignments of the 50 observatoins are contained in `$cluster` and a summary of the clustering model is given by:

```
km.out
```

```
## K-means clustering with 2 clusters of sizes 50, 152
##
## Cluster means:
##        RI       Na       Mg       Al       Si        K
          Ca
## 1 1.519226 13.71240 0.210400 1.837800 72.87120
    0.4910000 10.160400
## 2 1.518078 13.32796 3.429803 1.340197 72.57329
    0.4980263 8.573947
##          Ba        Fe
## 1 0.56580000 0.04800000
## 2 0.05901316 0.06171053
##
## Clustering vector:
##   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
    16  17  18
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
      2   2   2
##  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33
    34  35  36
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
      2   2   2
##  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51
    52  53  54
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
      2   2   2
##  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69
    70  71  72
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
      2   2   2
```

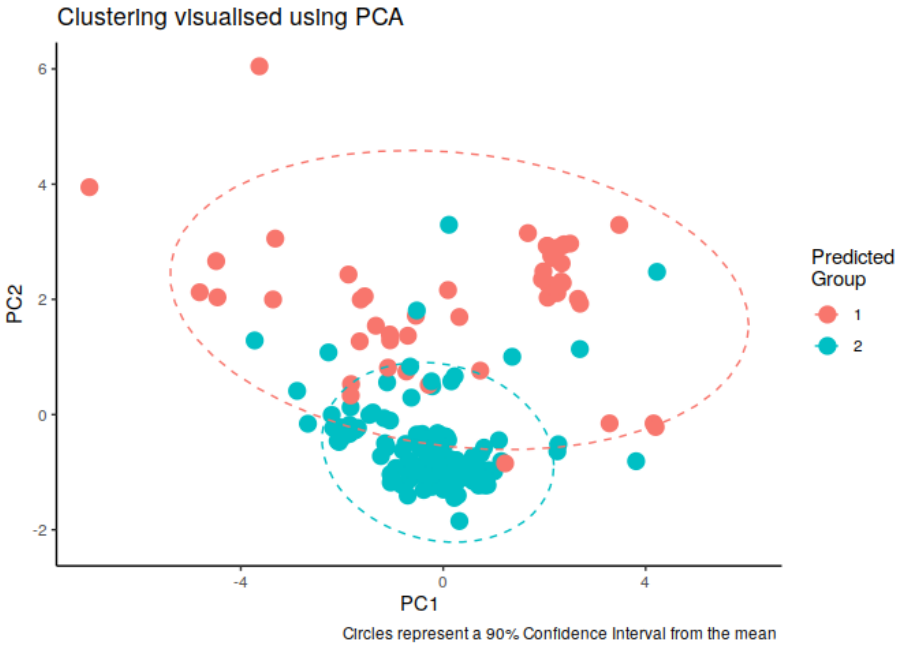
```
##  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87
    88  89  90
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
     2   2   2
##  91  92  93  94  95  96  97  98  99 100 101 102 103 104
    105 106 107 108
##   2   2   2   1   1   1   1   1   1   1   1   2   2   2   2
     2   2   2
## 109 110 111 112 113 114 115 116 117 118 119 120 121
    122 123 124 125 126
##   2   2   2   2   2   2   2   2   2   1   1   1   2   2   2
     2   2   2
## 127 128 129 130 131 132 133 134 135 136 137 138 139
    140 141 142 143 144
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
     2   2   2
## 145 146 147 148 149 150 151 152 153 154 155 156 157
    158 159 160 161 162
##   2   2   2   2   2   2   2   2   1   1   1   1   1   1   1
     1   1   1
## 163 164 165 166 167 168 169 170 171 172 173 174 175
    176 177 178 179 180
##   1   1   2   2   2   2   2   1   1   1   1   2   2   2   2
     2   2   1
## 181 182 183 184 185 186 187 188 189 190 191 192 193
    194 195 196 197 198
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
     1   1   1
## 199 200 201 202
##   1   1   1   1
##
## Within cluster sum of squares by cluster:
## [1] 531.2172 275.3888
##  (between_SS / total_SS = 38.9 %)
##
## Available components:
##
## [1] "cluster"    "centers"    "totss"      "withinss"
## [5] "tot.withinss" "betweenss" "size"       "iter"
## [9] "ifault"
```

```
#km.out$cluster
km2in <- km.out$tot.withinss
km2bet <- km.out$betweenss
```

Sub Question B

b. In order to visually display the two clusters obtained in part a, plot the first two principal components obtained in question 1 and colour according to the k-means classes.

```
## create a DF of PCA Data
pcaDF <- data.frame(pcaMod$x) %>% as_tibble()
pcaDF$group <- factor(km.out$cluster)

# Plot the PCA Reduction
ggplot(pcaDF, aes(x = PC1, y=PC2, col = group)) +
  geom_point(size = 4) +
  labs(col = "Predicted\nGroup",
       title = "Clustering visualised using PCA",
       caption = "Circles represent a 90% Confidence
           Interval from the mean ") +
  theme_classic() +
    stat_ellipse(type = 'norm', level = 0.9, lty = 2)
```




```
# PC11C <- sum(km.out$centers[1,]*
    pcaEnvMod$rotation[,1])
```

```
  # PC21C <- sum(km.out$centers[2,]*
      pcaEnvMod$rotation[,1])
  # PC12C <- sum(km.out$centers[1,]*
      pcaEnvMod$rotation[,2])
  # PC22C <- sum(km.out$centers[2,]*
      pcaEnvMod$rotation[,2])
```

Sub Question C

c. Construct the misclassification table and calculate the misclassification rate. Discuss the accuracy of classifying window and non-window glasses when using k-means clustering.

```
# Now create the confusion Matrix
  # This package prevents making mistakes
#  conf.mat <-  caret::confusionMatrix(data =
    factor(km.out$cluster), reference =
    factor(envDF$asthma))
   # We don't know which cluster is truth though...

  # This could otherwise be created by using, always go
      prediction, reference as a standard
  k2ConfMat <- table("ClusterPred" = (km.out$cluster-1),
      "Obsered" = glassDF$Type)
 k2ConfMat
```

```
##           Obsered
## ClusterPred  0   1
##           0  39  11
##           1  12 140
```

```
(k2ConfMat[1,1] +k2ConfMat[2,2])/(sum(k2ConfMat))
```

```
## [1] 0.8861386
```

```
 1-(k2ConfMat[1,1] +k2ConfMat[2,2])/(sum(k2ConfMat))
```

```
## [1] 0.1138614
```

```
conf.mat <-  caret::confusionMatrix(data =
    factor(as.numeric(!as.logical(km.out$cluster-1))),
    reference =
    factor(as.numeric(as.logical(glassDF$Type))))
conf.mat$table
```

```
##          Reference
## Prediction  0   1
##          0  12 140
##          1  39  11
```

The first matrix mismatches classification, using logicals this can be fixed and is shown in the second confusion matrix above.

Presuming that the clustering model performs better (as opposed to worse!) than chance, the prediction 1 matches with the class 1 and hence the misclassification rate may be determined by performing:

$$\mathsf{M}_c = \frac{12 + 11}{12 + 140 + 39 + 11} = 0.11$$

This shows that the misclassification rate is 0.11 which is fairly accurate. The clusters show good results in correctly clustering type 1 glass. the performance is slightly worse for type 0 glass however this could be because there are less observations.

Sub Question D

---

d.Alternatively perform Hierarchical Clustering on glass dataset. [Consider Euclidean distance as the dissimilarity measure and the closest distance between two clusters as the maximum distance between them]

Creating the Model

The maximum distance refers to the complete linkage function.

now begin by clustering the observations using complete linkage using the `hclust()` function and specifying the metod as `method = 'complete'`.

By Wine Quality `dist(x)` uses *Euclidean* distance and hence that will be used as the parameter for `hclust`.

```
hc.comp <- hclust(dist(glassFeat), method = 'complete')
summary(hc.comp)
```

```
##             Length Class  Mode
## merge       402    -none- numeric
## height      201    -none- numeric
## order       202    -none- numeric
## labels      202    -none- character
## method        1    -none- character
## call          3    -none- call
## dist.method 1    -none- character
```

```
hc.comp
```

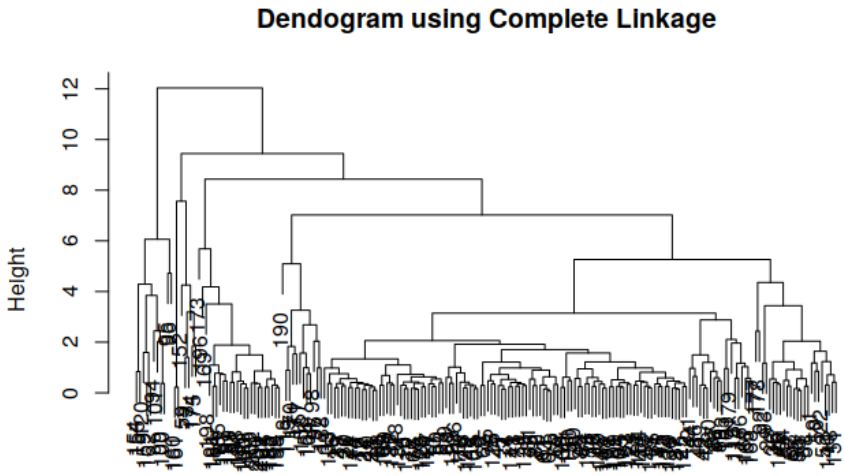
```
##
## Call:
## hclust(d = dist(glassFeat), method = "complete")
##
## Cluster method : complete
## Distance       : euclidean
## Number of objects: 202
```

Sub Question E

e. Display the dendrogram and cut it at a height that results in two distinct clusters.

Plot the Dendogram

```
plot(hc.comp, main = "Dendogram using Complete Linkage",
    sub="", xlab = "")
```



Cut the Tree

In order to return a vector with the assignment of observations with two categories the `cutree` function can be used:

```
cutree(hc.comp, k = 2)
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
   16 17 18
##  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
     1  1  1
## 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33
   34 35 36
##  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
     1  1  1
## 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51
   52 53 54
##  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
     1  1  1
## 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69
   70 71 72
##  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
     1  1  1
```

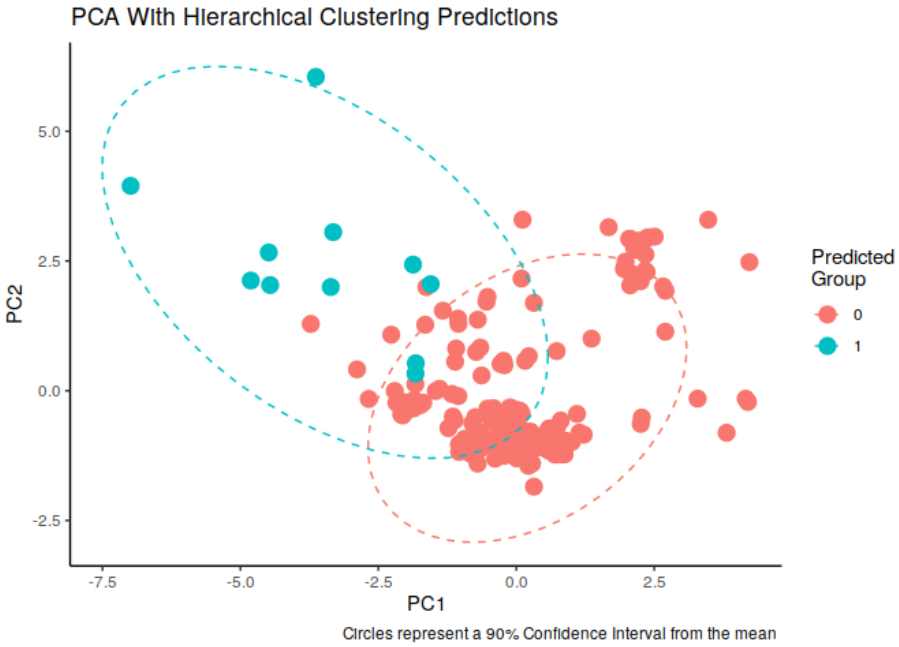
```
##  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87
    88  89  90
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
      1   1   1
##  91  92  93  94  95  96  97  98  99 100 101 102 103 104
    105 106 107 108
##   1   1   1   2   2   2   1   1   2   2   2   1   1   1   1
      1   1   1
## 109 110 111 112 113 114 115 116 117 118 119 120 121
    122 123 124 125 126
##   1   1   1   1   1   1   1   1   1   1   1   2   1   1   1
      1   1   1
## 127 128 129 130 131 132 133 134 135 136 137 138 139
    140 141 142 143 144
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
      1   1   1
## 145 146 147 148 149 150 151 152 153 154 155 156 157
    158 159 160 161 162
##   1   1   1   1   1   1   1   1   1   2   2   1   1   1   2
      1   1   2
## 163 164 165 166 167 168 169 170 171 172 173 174 175
    176 177 178 179 180
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
      1   1   1
## 181 182 183 184 185 186 187 188 189 190 191 192 193
    194 195 196 197 198
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
      1   1   1
## 199 200 201 202
##   1   1   1   1
```

Sub Question F

f. In order to visually display the two clusters obtained in part e, plot the first two principal components obtained in question 1 and colour according to the cluster profiles in part e.

```
# Create the predicted classes by using cutree
EucDist2Pred <- cutree(hc.comp, k = 2)


# Create the Data Frame
hcDF <- data.frame(
  glassFeat,
  "PredGroup" = EucDist2Pred
```

```
) %>% as_tibble()



## create a DF of PCA Data
pcaDF <- data.frame(pcaMod$x) %>% as_tibble()
pcaDF$group <- factor(cutree(hc.comp, 2)-1)

# Plot the PCA Reduction
ggplot(pcaDF, aes(x = PC1, y=PC2, col = group)) +
  geom_point(size = 4) +
  labs(col = "Predicted\nGroup",
       title = "PCA With Hierarchical Clustering
           Predictions",
       caption = "Circles represent a 90% Confidence
           Interval from the mean ") +
  theme_classic() +
    stat_ellipse(type = 'norm', level = 0.9, lty = 2)
```



Sub Question G

g. Construct the misclassification table and calculate the misclassification rate. Discuss the accuracy of classifying window and non-window glasses when using hierarchical clustering

```
# Now create the confusion Matrix
  # This package prevents making mistakes
#  conf.mat <-  caret::confusionMatrix(data =
    factor(km.out$cluster), reference =
    factor(envDF$asthma))
   # We don't know which cluster is truth though...

  # This could otherwise be created by using, always go
      prediction, reference as a standard
  k2ConfMat <- table("ClusterPred" = (EucDist2Pred-1),
      "Obsered" = glassDF$Type)
 k2ConfMat
```

```
##           Obsered
## ClusterPred  0   1
##           0  47 144
##           1   4   7
```

```
(k2ConfMat[1,1] +k2ConfMat[2,2])/(sum(k2ConfMat))
```

```
## [1] 0.2673267
```

```
 1-(k2ConfMat[1,1] +k2ConfMat[2,2])/(sum(k2ConfMat))
```

```
## [1] 0.7326733
```

```
  conf.mat <-  caret::confusionMatrix(data =
      factor(as.numeric(!as.logical(EucDist2Pred-1))),
      reference =
      factor(as.numeric(as.logical(glassDF$Type))))
  conf.mat$table
```

```
##          Reference
## Prediction  0   1
##          0   4   7
##          1  47 144
```

The first matrix mismatches classification, using logicals this can be fixed and is shown in the second confusion matrix above.

Presuming that the clustering model performs better (as opposed to worse!) than chance, the prediction 1 matches with the class 1 and hence the misclassification rate may be determined by performing:

$$\mathsf{M}_c = \frac{7 + 47}{7 + 47 + 144 + 4} = 0.27$$

This shows that the misclassification rate is 0.11 which is fairly accurate. The clusters show good results in correctly clustering type 1 glass. the performance is slightly worse for type 0 glass however this could be because there are less observations.

Sub Question H

h. Compare results obtained in parts "b and c" with parts "f and g" and justify most appropriate clustering method to classify the dataset as window glasses and non-window glasses.

```
km.out$betweenss
```

```
## [1] 512.5338
```

```
km.out$tot.withinss
```

```
## [1] 806.606
```

```
summary(hc.comp)
```

```
##             Length Class  Mode
## merge       402    -none- numeric
## height      201    -none- numeric
## order       202    -none- numeric
## labels      202    -none- character
## method        1    -none- character
## call          3    -none- call
## dist.method   1    -none- character
```

Here we have the luxury of being able to outright consider the misclassification rate, in this context the misclassification rate is lowest for the clustering and so that would be the appropriate technique to employ.

the between squared error is for the the clustering is 512 and the total within squared error is 806. comparing this to the heirarchical clustering would be interesting if it was easy to have an analogous value returned.

Question 3

Wine should have 261 observations

Sub Question A

a. Identify and state the Quantitative and Qualitative variables in the given dataset.

```
knitTable(head(wineDF))
```

WineQuality	FixedAcidity	VolatileAcidity	CitricAcid	ResidualSugar	Chlorides	FreeSulfurDioxide	TotalS
Low	7.9	0.18	0.37	1.2	0.040	16	
Low	6.5	0.31	0.14	7.5	0.044	34	
High	6.2	0.66	0.48	1.2	0.029	29	
High	6.4	0.31	0.38	2.9	0.038	19	
High	7.0	0.28	0.39	8.7	0.051	32	
High	7.0	0.32	0.34	1.3	0.042	20	

```
str(wineDF)
```

```
## 'data.frame':  261 obs. of 12 variables:
## $ WineQuality     : Factor w/ 2 levels "High","Low":
    2 2 1 1 1 1 1 2 1 2 ...
## $ FixedAcidity    : num 7.9 6.5 6.2 6.4 7 7 7.2 7.4
    7.1 7.4 ...
## $ VolatileAcidity : num 0.18 0.31 0.66 0.31 0.28
    0.32 0.39 0.25 0.23 0.39 ...
## $ CitricAcid      : num 0.37 0.14 0.48 0.38 0.39
    0.34 0.63 0.37 0.35 0.23 ...
## $ ResidualSugar   : num 1.2 7.5 1.2 2.9 8.7 1.3 11
    13.5 16.5 7 ...
## $ Chlorides       : num 0.04 0.044 0.029 0.038 0.051
    0.042 0.044 0.06 0.04 0.033 ...
## $ FreeSulfurDioxide : num 16 34 29 19 32 20 55 52 60
    29 ...
## $ TotalSulfurDioxide: int 75 133 75 102 141 69 156
    192 171 126 ...
## $ Density         : num 0.992 0.996 0.989 0.991
    0.996 ...
## $ PH              : num 3.18 3.22 3.33 3.17 3.38
    3.31 3.09 3 3.16 3.14 ...
## $ Sulphates       : num 0.63 0.5 0.39 0.35 0.53 0.65
    0.44 0.44 0.59 0.42 ...
## $ Alcohol         : num 10.8 9.5 12.8 11 10.5 12 8.7
    9.1 9.1 10.5 ...
```

```
summary(wineDF)
```

```
##  WineQuality FixedAcidity VolatileAcidity  CitricAcid
##  High:178   Min.   :4.200  Min.   :0.1200  Min.   :0.0000
##  Low : 83   1st Qu.:6.400  1st Qu.:0.2100  1st
    Qu.:0.2600
##             Median :6.800  Median :0.2600  Median
    :0.3100
##             Mean   :6.947  Mean   :0.2751  Mean   :0.3209
##             3rd Qu.:7.500  3rd Qu.:0.3200  3rd
    Qu.:0.3700
##             Max.   :9.700  Max.   :0.6600  Max.   :0.7400
##  ResidualSugar    Chlorides     FreeSulfurDioxide
    TotalSulfurDioxide
##  Min.   : 0.800  Min.   :0.0180  Min.   :  4.00   Min.
    : 24.0
##  1st Qu.: 1.700  1st Qu.:0.0360  1st Qu.: 24.00   1st
    Qu.:110.0
##  Median : 5.400  Median :0.0420  Median : 34.00   Median
    :138.0
```

```
##  Mean   : 6.329  Mean   :0.0439  Mean   : 36.06   Mean
    :138.9
##  3rd Qu.: 9.500  3rd Qu.:0.0490  3rd Qu.: 46.00   3rd
    Qu.:166.0
##  Max.   :19.800  Max.   :0.2400  Max.   :108.00   Max.
    :240.0
##    Density           PH         Sulphates
    Alcohol
##  Min.   :0.9891  Min.   :2.740  Min.   :0.2700  Min.   :
    8.00
##  1st Qu.:0.9919  1st Qu.:3.100  1st Qu.:0.4200  1st Qu.:
    9.50
##  Median :0.9940  Median :3.170  Median :0.4800  Median
    :10.40
##  Mean   :0.9941  Mean   :3.193  Mean   :0.5056  Mean
    :10.46
##  3rd Qu.:0.9960  3rd Qu.:3.290  3rd Qu.:0.5600  3rd
    Qu.:11.20
##  Max.   :1.0018  Max.   :3.680  Max.   :0.9500  Max.
    :14.05
```

```
dim(wineDF)
```

```
## [1] 261 12
```

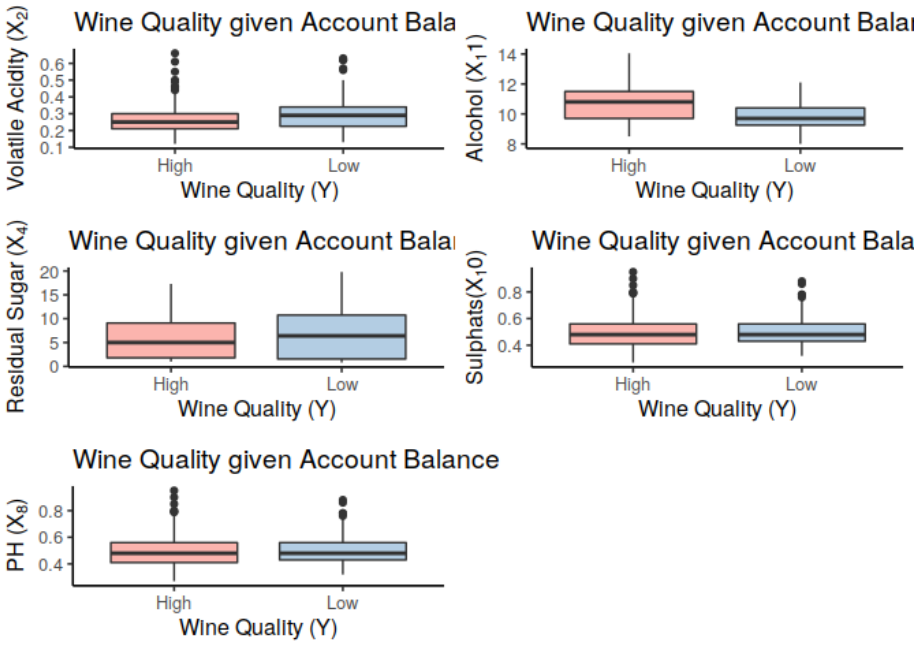
The Wine data is 261 observations of 11 continous predictive variables and one categorical response variable.

Sub Question B

b. Construct the box plot for Volatile Acidity, Alcohol, Residual Sugar, Sulphates and pH on the target variable "WineQuality" and interpret your findings.

```
volplot <- ggplot(data = winedf, aes(x = WineQuality, y
    = VolatileAcidity, fill = WineQuality)) +
  geom_boxplot() +
  labs(x = TeX("Wine Quality $(Y)$"), y = TeX("Volatile
      Acidity $(X_2)$"), title = "Wine Quality given
      Account Balance") +
  theme_classic2() +
  guides(fill = FALSE) +
```

```
  scale_fill_brewer(palette = "Pastel1")


alcplot <- ggplot(data = winedf, aes(x = WineQuality, y
    = Alcohol, fill = WineQuality)) +
  geom_boxplot() +
  labs(x = TeX("Wine Quality $(Y)$"), y = TeX("Alcohol
      $(X_11)$"), title = "Wine Quality given Account
      Balance") +
  theme_classic2() +
  guides(fill = FALSE) +
  scale_fill_brewer(palette = "Pastel1")


rsugplot <- ggplot(data = winedf, aes(x = WineQuality, y
    = ResidualSugar, fill = WineQuality)) +
  geom_boxplot() +
  labs(x = TeX("Wine Quality $(Y)$"), y = TeX("Residual
      Sugar $(X_4)$"), title = "Wine Quality given
      Account Balance") +
  theme_classic2() +
  guides(fill = FALSE) +
  scale_fill_brewer(palette = "Pastel1")


sulphplot <- ggplot(data = winedf, aes(x = WineQuality,
    y = Sulphates, fill = WineQuality)) +
  geom_boxplot() +
  labs(x = TeX("Wine Quality $(Y)$"), y =
      TeX("Sulphats$(X_10)$"), title = "Wine Quality
      given Account Balance") +
  theme_classic2() +
  guides(fill = FALSE) +
  scale_fill_brewer(palette = "Pastel1")

phplot <- ggplot(data = winedf, aes(x = WineQuality, y =
    Sulphates, fill = WineQuality)) +
  geom_boxplot() +
  labs(x = TeX("Wine Quality $(Y)$"), y = TeX("PH
      $(X_8)$"), title = "Wine Quality given Account
      Balance") +
  theme_classic2() +
  guides(fill = FALSE) +
  scale_fill_brewer(palette = "Pastel1")

grid.arrange(volplot, alcplot, rsugplot, sulphplot,
    phplot, nrow = 3)
```



This clearly shows that having lower acidity, sulphates, residual sugar is indicative of a better wine, while having a higher alcohol content is indicative of a good wine. It isn't clear if ph is indicative of lower or higher.

None of these differences are outright statiscially signicant however, they are mere trends.

Sub Question C

c. Build a logistic regression model to classify the "WineQuality" in terms of Volatile acidity, Alcohol and Residual Sugar. (No need to prove the significance of the model)

In order to create a logistic model use the `glm` function which is a *generalised linear model* function, because this is just an exponentiated linear function.

- Choose `family = binomial`
  - categorical variables are not normally distributed because they are not continuous, a binomial distribution is the corresponding distribution for discrete data
- Choose `link = logit`
  - Use `probit` for when the seperation of data is distinct, it performs slightly better.
  - the term *logit* is another word for *log odds*:

* $\log\left(\frac{\mathrm{P}(X)}{1-\mathrm{P}(X)}\right)$

```
winedf$WineQuality <- factor(winedf$WineQuality)

ModDef <- glm(formula = WineQuality ~ VolatileAcidity +
    Alcohol + ResidualSugar , family = binomial(link =
    "logit"), data = winedf)
contrasts(winedf$WineQuality)
```

```
##      Low
## High   0
## Low    1
```

The `contrasts` shows that the dummy variables for wine quality have been encoded such that low is 1 and high is 0, this is a better way to do it because it means we do not need to manipulate the data in order to model it, however before creating the model we have factorised the data just to be sure it behaves well.

Summarise and Inspect the Model

```
ModDef
```

```
##
## Call: glm(formula = WineQuality ~ VolatileAcidity +
    Alcohol + ResidualSugar,
##     family = binomial(link = "logit"), data = winedf)
##
## Coefficients:
##     (Intercept) VolatileAcidity       Alcohol
    ResidualSugar
##          9.3297          6.5013         -1.0952
    -0.1019
##
## Degrees of Freedom: 260 Total (i.e. Null); 257
    Residual
## Null Deviance:     326.4
## Residual Deviance: 268.9  AIC: 276.9
```

```
ModDef %>% summary()
```

```
##
## Call:
```

```
## glm(formula = WineQuality ~ VolatileAcidity + Alcohol
    + ResidualSugar,
##     family = binomial(link = "logit"), data = winedf)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8399  -0.8234  -0.4743   0.9836   2.1256
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)     9.32966    1.95533   4.771 1.83e-06 ***
## VolatileAcidity 6.50130    1.69998   3.824 0.000131 ***
## Alcohol        -1.09521    0.18608  -5.886 3.96e-09 ***
## ResidualSugar  -0.10186    0.03649  -2.792 0.005243 **
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.'
    0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be
    1)
##
##     Null deviance: 326.44 on 260 degrees of freedom
## Residual deviance: 268.91 on 257 degrees of freedom
## AIC: 276.91
##
## Number of Fisher Scoring iterations: 5
```

this model indicates that the appropriate model is of the form:

$$p\left(X\right) = \frac{e^{\beta_0+\beta_1 X_1+\cdots\beta_p Xp}}{1+e^{\beta_0+\beta_1 X_1+\cdots\beta_p Xp}}$$

by putting in the predicted values we have:

$$p\left(X\right) = \frac{e^{9.33+6.5\cdot \text{V.Acid}-1.09\cdot \text{Alc}-0.10\cdot \text{RSug}}}{1+e^{9.33+6.5\cdot \text{V.Acid}-1.09\cdot \text{Alc}-0.10\cdot \text{RSug}}}$$

Sub Question D

---

d. Build a logistic regression model to classify the "WineQuality" in terms of Volatile acidity, Alcohol, Residual Sugar, Sulphates and pH.

```
ModDef2 <- glm(formula = WineQuality ~ VolatileAcidity +
    Alcohol + ResidualSugar + Sulphates + PH, family =
    binomial(link = "logit"), data = winedf)
```

```
contrasts(winedf$WineQuality)
```

```
##      Low
## High   0
## Low    1
```

The `contrasts` again shows that the dummy variables for wine quality have been encoded such that low is 1 and high is 0, this is a better way to do it because it means we do not need to manipulate the data in order to model it.

Summarise and Inspect the Model

```
ModDef2
```

```
##
## Call: glm(formula = WineQuality ~ VolatileAcidity +
    Alcohol + ResidualSugar +
##     Sulphates + PH, family = binomial(link = "logit"),
    data = winedf)
##
## Coefficients:
##     (Intercept) VolatileAcidity       Alcohol
    ResidualSugar
##         10.2582          6.8388         -1.1056
    -0.1048
##      Sulphates             PH
##          1.7115         -0.5560
##
## Degrees of Freedom: 260 Total (i.e. Null); 255
    Residual
## Null Deviance:     326.4
## Residual Deviance: 267 AIC: 279
```

```
ModDef2 %>% summary()
```

```
##
## Call:
## glm(formula = WineQuality ~ VolatileAcidity + Alcohol
    + ResidualSugar +
##     Sulphates + PH, family = binomial(link = "logit"),
    data = winedf)
##
## Deviance Residuals:
```

```
##    Min      1Q  Median      3Q     Max
## -1.8292 -0.8153 -0.4718  0.9591  2.1661
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)    10.25825   3.63028  2.826 0.00472 **
## VolatileAcidity 6.83879   1.72932  3.955 7.67e-05 ***
## Alcohol        -1.10560   0.18657 -5.926 3.11e-09 ***
## ResidualSugar -0.10477   0.03678 -2.848 0.00439 **
## Sulphates       1.71147   1.26122  1.357 0.17478
## PH             -0.55602   1.01321 -0.549 0.58316
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.'
    0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be
    1)
##
##    Null deviance: 326.44 on 260 degrees of freedom
## Residual deviance: 266.98 on 255 degrees of freedom
## AIC: 278.98
##
## Number of Fisher Scoring iterations: 5
```

Sub Question E

e. Considering the significance of the parameter estimates, select and state the best model out of the models obtained in part c and d to classify the quality of wine. Clearly, explain your answer with proper justification.

Significance

From the summary above it can be seen that Sulphates and PH are non-significant predictor values, with high p-values of 0.174 and 0.58 respectively

Residual Deviance

The deviance of the residuals does not improve between the models despite the added precictors, further evidence to reject the model for fear of over fitting (bias/variance) trade off.

AIC

The second model uses more parameters without improving the AIC value, further evidence to reject the model.

ANOVA

```
anova(ModDef, ModDef2)
```

```
## Analysis of Deviance Table
##
## Model 1: WineQuality ~ VolatileAcidity + Alcohol +
    ResidualSugar
## Model 2: WineQuality ~ VolatileAcidity + Alcohol +
    ResidualSugar + Sulphates +
##    PH
##   Resid. Df Resid. Dev Df Deviance
## 1       257     268.90
## 2       255     266.98 2   1.9234
```

This indicates that the second model is the superior model

Sub Question F

f. Write down the equation to calculate the probability of getting low wine quality for a given set of predictors using part e.

The probability of getting low wine quality can be predicted, using the model from e by using the following formula:

$$p(X) = \frac{e^{10.26+6.83 \cdot \text{V.Acid}-1.1 \cdot \text{Alc}-0.105 \cdot \text{RSug}+1.7 \cdot \text{Sulph}-0.55 \cdot \text{PH}}}{1+e^{10.26+6.83 \cdot \text{V.Acid}-1.1 \cdot \text{Alc}-0.105 \cdot \text{RSug}+1.7 \cdot \text{Sulph}-0.55 \cdot \text{PH}}}$$

Sub Question G **g. Classify the observations with probability > 0.6 as "low" and "high" otherwise. Hence, calculate the misclassification matrix and misclassification rate and comment on your results.**

Classify Values

In order to move from the probability to the decision the threshold would have to be specified:

```
#attach(winedf)
winedf2 <- subset(winedf, select = c(WineQuality ,
    VolatileAcidity, Alcohol, ResidualSugar, Sulphates,
    PH))


threshold <- 0.6
```

```
ModelPreds <- cbind(winedf, "Probability" =
    ModDef$fitted.values)
ModelPreds <- cbind(winedf2, "Probability" =
    ModDef$fitted.values)
ModelPreds$prediction <- rep_len(x ="High", length.out =
    nrow(ModelPreds))
ModelPreds$prediction[ModelPreds$Probability >
    threshold] <- "Low"
ModelPreds$prediction <- as.factor(ModelPreds$prediction)
#ModelPreds <- ModelPreds[,c(2,6,7,3,4,5)]
ModelPreds <- subset(ModelPreds, select = c(WineQuality,
    prediction , VolatileAcidity, Alcohol,
    ResidualSugar, Sulphates, PH))
head(ModelPreds)
```

```
##   WineQuality prediction VolatileAcidity Alcohol
    ResidualSugar Sulphates
## 1         Low       High           0.18   10.8
    1.2      0.63
## 2         Low       High           0.31    9.5
    7.5      0.50
## 3        High       High           0.66   12.8
    1.2      0.39
## 4        High       High           0.31   11.0
    2.9      0.35
## 5        High       High           0.28   10.5
    8.7      0.53
## 6        High       High           0.32   12.0
    1.3      0.65
##    PH
## 1 3.18
## 2 3.22
## 3 3.33
## 4 3.17
## 5 3.38
## 6 3.31
```

```
knitTable(ModelPreds[1:6,])
```

WineQuality	prediction	VolatileAcidity	Alcohol	ResidualSugar	Sulphates	PH
Low	High	0.18	10.8	1.2	0.63	3.18
Low	High	0.31	9.5	7.5	0.50	3.22
High	High	0.66	12.8	1.2	0.39	3.33
High	High	0.31	11.0	2.9	0.35	3.17
High	High	0.28	10.5	8.7	0.53	3.38

WineQuality	prediction	VolatileAcidity	Alcohol	ResidualSugar	Sulphates	PH
High	High	0.32	12.0	1.3	0.65	3.31

```
#Alternative approach
#threshold <- 0.5
#Predictions <- ifelse(ModelPreds$Probability <
    threshold, 0, 1)
#ModelPreds <- cbind(ModelPreds, "Prediction" =
    Predictions)
```

This shows that for the first few points that the prediction accuracy isn't that great (or that possibly I have it backwards in my code). It could be possible that a ROC curve could be used to improve the performance of the model by tuning the threshold value.

Create the Misclassification matrix

```
#ifelse(ModelPreds$prediction == "no",
    ModelPreds$prediction <- "Low",
    ModelPreds$prediction <- "High" )
#ModelPreds$prediction <- factor(ModelPreds$prediction)

conf.mat <- table(ModelPreds$prediction,
    ModelPreds$WineQuality, dnn = c("Predicted",
    "Observed"))
print(conf.mat)
```

```
##         Observed
## Predicted High Low
##      High  171  62
##      Low     7  21
```

```
# It can be a little difficult to determine which one is
    the prediction using table
  #if you don't get the order of prediction then
      observation right, so
    # you can also use `confusionMatrix` from the `caret`
        package to triple check.


conf.mat2 <- confusionMatrix(ModelPreds$prediction,
    ModelPreds$WineQuality)
print(conf.mat2)
```

```
## Confusion Matrix and Statistics
##
##          Reference
## Prediction High Low
##      High  171  62
##      Low     7  21
##
##               Accuracy : 0.7356
##                 95% CI : (0.6777, 0.7881)
##    No Information Rate : 0.682
##    P-Value [Acc > NIR] : 0.0348
##
##                  Kappa : 0.2596
##
##  Mcnemar's Test P-Value : 7.987e-11
##
##             Sensitivity : 0.9607
##             Specificity : 0.2530
##          Pos Pred Value : 0.7339
##          Neg Pred Value : 0.7500
##              Prevalence : 0.6820
##          Detection Rate : 0.6552
##    Detection Prevalence : 0.8927
##       Balanced Accuracy : 0.6068
##
##        'Positive' Class : High
##
```

this Misclassification Matrix may be caluculated by Using:

$$\mathsf{M}_c = \frac{62 + 7}{171 + 62 + 7 + 21} = 0.2644$$

The misclassification rate is 0.2644 which is reasonably low which indicates this is a good model, it is possible that this model is however overfit.

The p-value is highly significant.

Sub Question H

h. Calculate four other measures such as True Positive Rate, False Positive Rate, True Negative Rate and False Negative Rate that can be used to assess the accuracy of the model.

The true positive rate is the number of positives that are true divided by the number of positives observed

$$\textbf{TruePosRate} \text{ is } \textit{Sensitivity} = \frac{\text{of True Positives}}{\text{of True Positives} + \text{ of False Negatives}}$$

$$\textbf{FalsePosRate} \text{ is } (1 - \textit{Sensitivity}) = \frac{\text{of False Positives}}{\text{of False Positive} + \text{ of True True Negative}}$$

Sensitivity is **TruePosRate**
Specificity is **TrueNegRate**

A ROC Curve is the TruePosRate (i.e. the Sensitivity) on the y-axis against the FalsePosRate (i.e. 1-Sensitivity) on the x-axis, this could be used to improve the model.

False Negative Rate

The false negative rate is the number of negative observations incorrectly classified

```
conf.mat
```

```
##         Observed
## Predicted High Low
##      High  171  62
##      Low     7  21
```

```
fnr <- 40/(228+105) #fnr = fn/pos
fnr
```

```
## [1] 0.1201201
```

True Negative Rate

```
conf.mat
```

```
##         Observed
## Predicted High Low
##      High  171  62
##      Low     7  21
```

```
tnr <- 40/(9627+40) #fnr = tn/neg
tnr
```

```
## [1] 0.004137788
```

True Positive Rate

```
conf.mat
```

```
##         Observed
## Predicted High Low
##      High  171  62
##      Low     7  21
```

```
tpr <- 105/(228+105) #tpr= tp/p
tnr
```

```
## [1] 0.004137788
```

This is defined:

$$\text{TruePos} = \frac{105}{105 + 40} = 0.72$$

False Positive Rate

```
conf.mat
```

```
##         Observed
## Predicted High Low
##      High  171  62
##      Low     7  21
```

```
fpr <- 228/(9627+40)  #fpr=fp/n
fpr
```

```
## [1] 0.02358539
```

Sensitivity

```
sens <- 228/(228+9627)
sens
```

```
## [1] 0.02313546
```

Sub Question I

Mention another supervised learning method that can be used to solve the given problem.
Linear Discriminant Analysis is another technique to perform supervised classification, it is the preferred method when there are more than two output classes .

Use Pandoc to Create PDF

```
pandoc --toc --listings -s FinalExam.md -o FinalExam.tex
    && pdflatex -interaction=nonstopmode FinalExam.tex
xdg-open FinalExam.pdf & disown
```

$H_0: \ \beta = 0$ The Slope Parameter is not significantly different from 0
$H_a: \ \beta \neq 0$ The Slope Parameter **is** significantly different from 0