Multivariate Analysis Assignment

Suraj Bodhanandan Nhattuvetty - 23200338

Question 1

```
#Load the dataset
set.seed(23200338)
data_temp <- read.csv("Temperature_data.csv")

#Load dplyr package
library(dplyr)
```

```
Warning: package 'dplyr' was built under R version 4.3.3
```

```

Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```
#Sample 1000 observations
data <- sample_n(data_temp, size = 1000)
```

Question 2

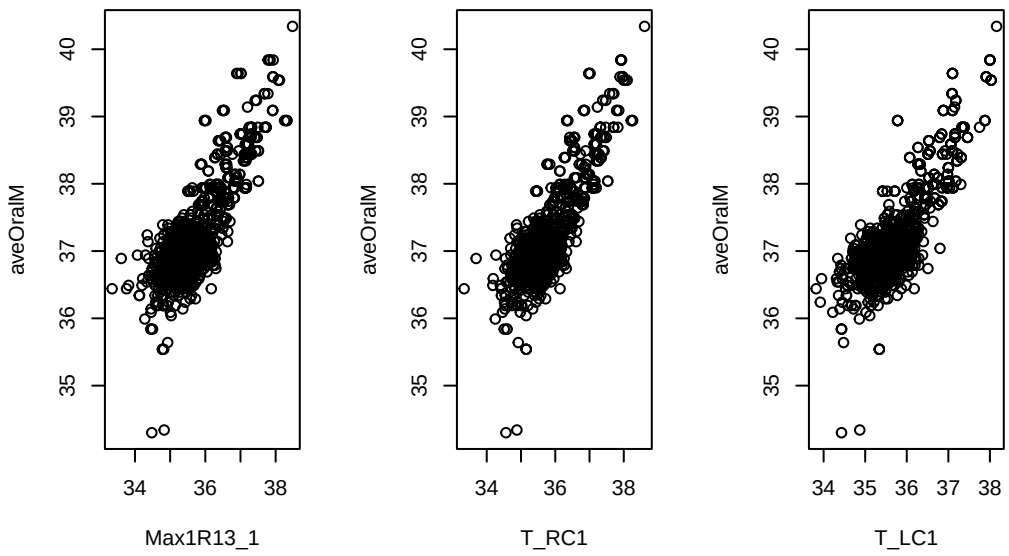
We check if there are missing or NA values in the Oral_Temp variable

```
#Check for missing values
sum(is.na(data$aveOralM))
```

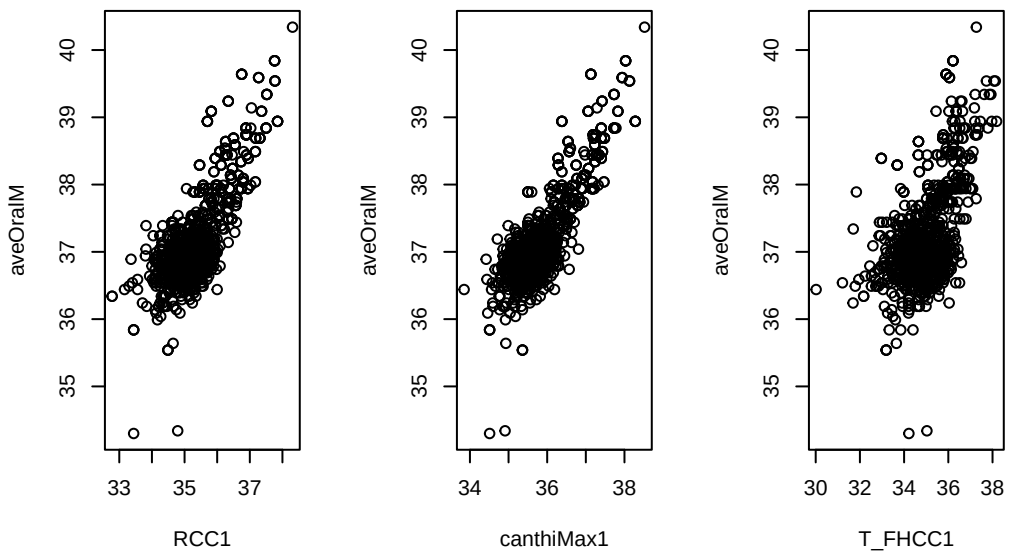
```
[1] 0
```

There are no missing values so no rows are removed.

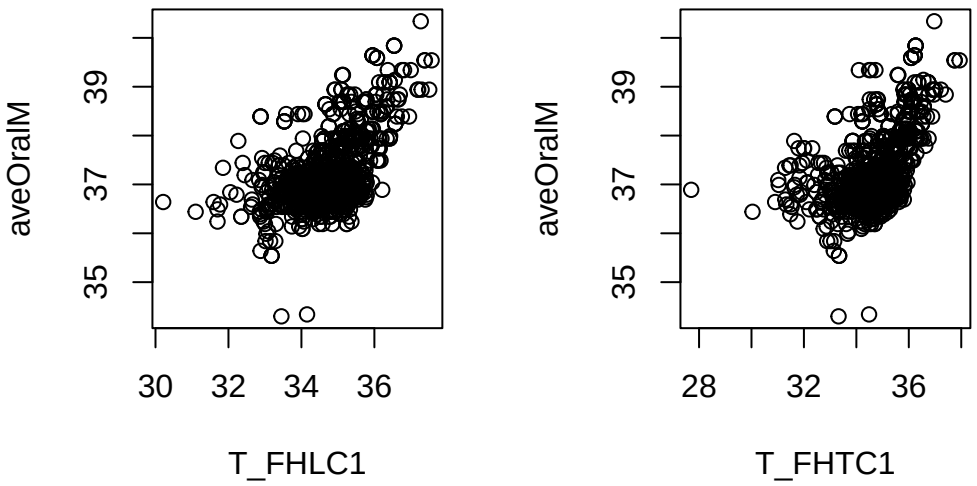
```
#Plot the facial and oral temperatures
par(mfrow = c(1,3))
plot(data$Max1R13_1, data$aveOralM, xlab = "Max1R13_1", ylab = "aveOralM")
plot(data$T_RC1, data$aveOralM, xlab = "T_RC1", ylab = "aveOralM")
plot(data$T_LC1, data$aveOralM, xlab = "T_LC1", ylab = "aveOralM")
```




```
par(mfrow = c(1,3))
plot(data$RCC1, data$aveOralM, xlab = "RCC1", ylab = "aveOralM")
plot(data$canthiMax1, data$aveOralM, xlab = "canthiMax1", ylab = "aveOralM")
plot(data$T_FHCC1, data$aveOralM, xlab = "T_FHCC1", ylab = "aveOralM")
```




```
par(mfrow = c(1,2))
plot(data$T_FHLC1, data$aveOralM, xlab = "T_FHLC1", ylab = "aveOralM")
plot(data$T_FHTC1, data$aveOralM, xlab = "T_FHTC1", ylab = "aveOralM")
```



It can be seen from all the plots that the relation between the variables are positive. The average oral temperature is around 36 to 37 and the average facial temperature seems to be around 34 to 35 as most number of points are plotted in this region. It can also be seen that there are some outliers in the dataset.

```
#Removing outlier
threshold <- mean(data$aveOralM) - 4 * sd(data$aveOralM)
data <- filter(data, aveOralM >= threshold)
nrow(data)
```

```
[1] 999
```

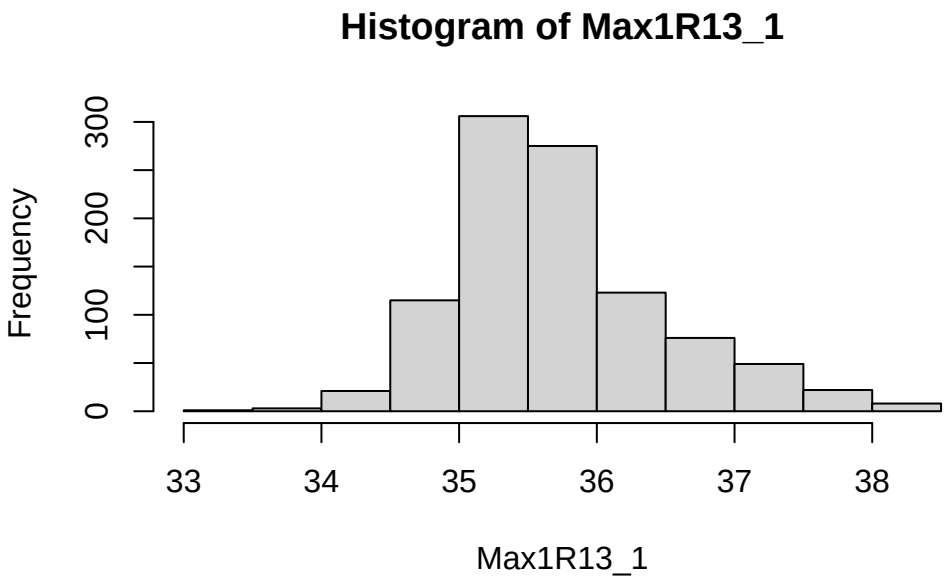
One outlier has been removed now.

Question 3

Hierarchical clustering

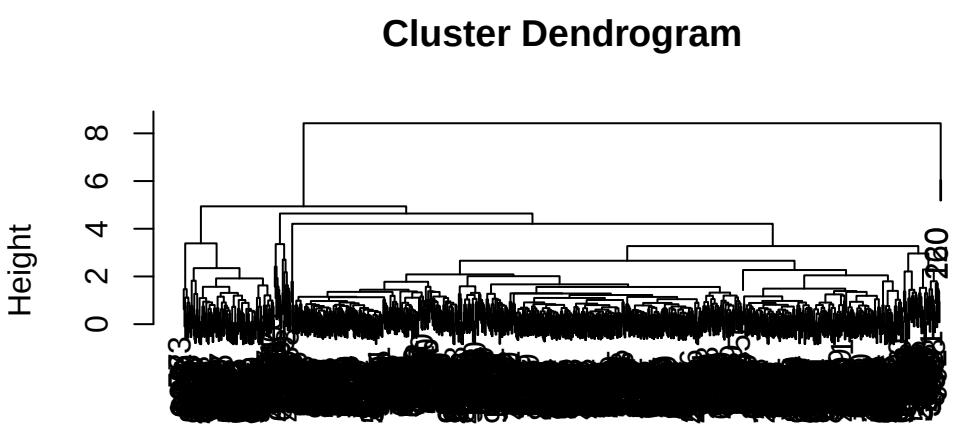
We extract the required variables and plot a histogram of one variable to check if it is normally distributed. We only check for one variable as the variables are very similar to each other.

```r
#Extract facial temperatures
data_facial <- data[,1:8]
#Plot histogram of one variable
hist(data_facial$Max1R13_1, xlab = "Max1R13_1", main = "Histogram of Max1R13_1" )
```



From the histogram it can be seen that the data is normally distributed. Therefore we decide to use average linkage method for hierarchical clustering and Euclidean distance is used for dissimilarity matrix as the data is numerical.

```r
dist_eucl <- dist(data_facial, method = "euclidean")
hcluster <- hclust(dist_eucl, method = "average")
plot(hcluster, xlab="Average linkage", sub="")
```

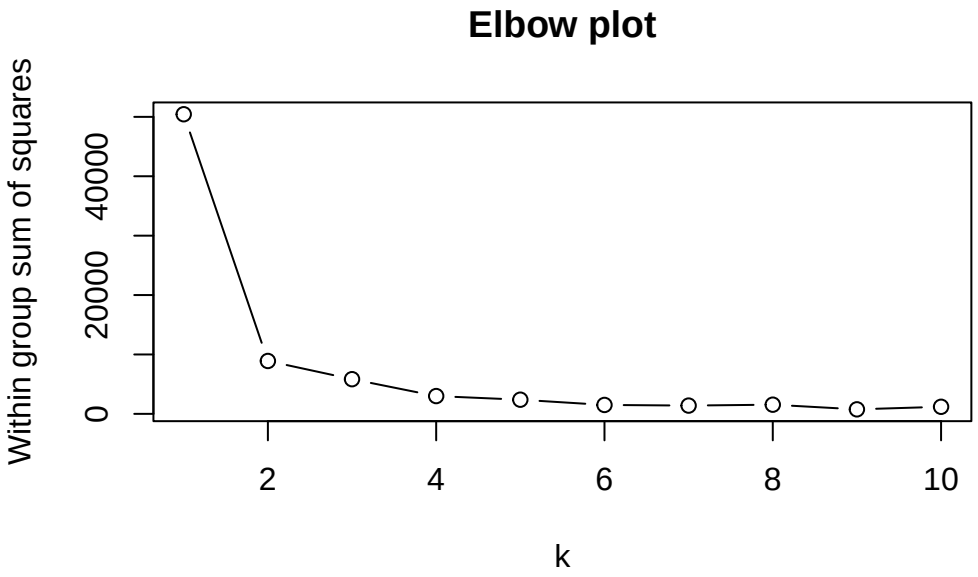


From the dendrogram it can be seen that there may be two clusters in the data but they are very close as the height is not very different for the two topmost clusters.

K-means

For K-means, we need to decide on a K value. We find the within group sum of squares for K over the range 1 to 10 and plot it against the K values to find the optimal value.

```
#Calculate sum of squares for elbow plot
WGSS = rep(0,10)
n = nrow(data_facial)

for(k in 1:10)
{
  WGSS[k] = sum(kmeans(faithful, centers = k)$withinss)
}

plot(1:10, WGSS, type="b", xlab="k", ylab="Within group sum of squares",
     main = "Elbow plot")
```



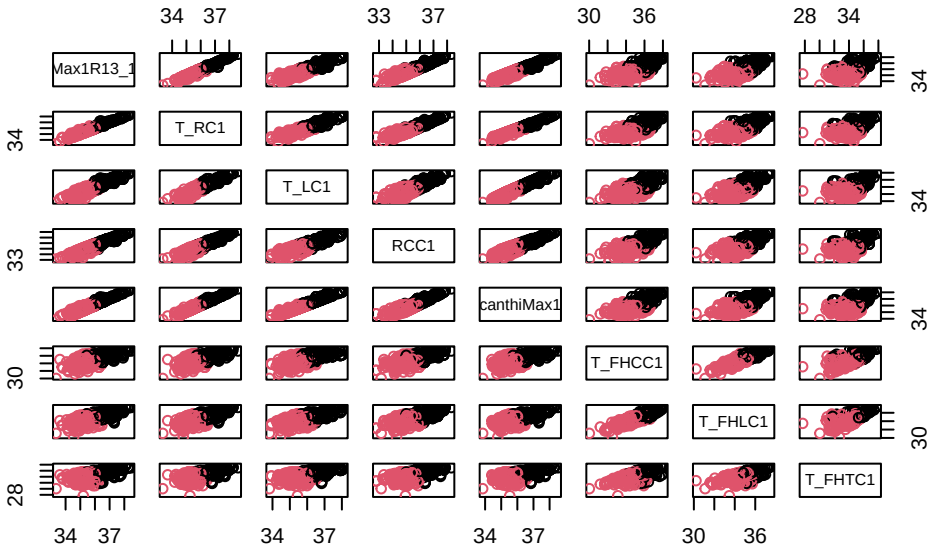
From the elbow plot it can be clearly seen that K should be 2.

```
library(tidyr)
#Dropping the missing values
data_facial <- drop_na(data_facial)
```

```
#Performing k means clustering
k = 2
kcluster = kmeans(data_facial, center=k, nstart=10)
table(kcluster$cluster)
```

```
  1   2
284 711
```

```
#Plotting the clusters
plot(data_facial, col = kcluster$cluster)
points(kcluster$centers, col=1:k, pch=8, cex=2)
```



The clusters formed are meaningful. There is one cluster which consists of higher temperatures recorded and another cluster of lower temperatures. There is not a big distance between the clusters which was seen in the dendrogram of the hierarchical clustering. The initial assumption of two clusters in hierarchical clustering also turned out to be correct.

Question 4

LDA

We extract the necessary data and perform PCA on it as we need to plot the decision boundaries of the LDA.

```
library(MASS)
```

```
Attaching package: 'MASS'

The following object is masked from 'package:dplyr':

    select
```

```r
#Extract the data
data_facial_2 <- cbind(data[,1:8], Gender = data$Gender)
data_facial_2 <- drop_na(data_facial_2)
#Standardise the data
data_facial_lda <- scale(data_facial_2[,1:8])
#Perform PCA
pca_lda <- prcomp(data_facial_lda)
summary(pca_lda)
```

```
Importance of components:
                          PC1    PC2     PC3     PC4    PC5     PC6     PC7
Standard deviation     2.539 0.9660 0.54076 0.36252 0.3323 0.23785 0.13630
Proportion of Variance 0.806 0.1166 0.03655 0.01643 0.0138 0.00707 0.00232
Cumulative Proportion  0.806 0.9226 0.95916 0.97559 0.9894 0.99646 0.99878
                           PC8
Standard deviation     0.09871
Proportion of Variance 0.00122
Cumulative Proportion  1.00000
```

We select 2 principal components as total variance explained is close to 92%.

```r
#Calculate PCA scores and selecting only first 2 prinicipal components
new_data <- predict(pca_lda)
data_facial_3 <- cbind(as.data.frame(new_data[,1:2]),
                       Gender = data_facial_2$Gender)
```

Splitting the data into train and test for the LDA. We decide to do 70-30 split as we have close to 1000 observations.

```r
#Splitting the data into train and test
sample <- sample(c(TRUE, FALSE), nrow(data_facial_3),
                 replace=TRUE, prob=c(0.7,0.3))
train <- data_facial_3[sample, ]
test <- data_facial_3[!sample, ]
```

```r
#Fitting the LDA model
lda1 <- lda(Gender ~ ., data = train)
lda1
```

```
Call:
```

```
lda(Gender ~ ., data = train)

Prior probabilities of groups:
   Female      Male
0.5583685 0.4416315

Group means:
              PC1         PC2
Female -0.6086612  0.11844399
Male    0.8147318 -0.09371578

Coefficients of linear discriminants:
           LD1
PC1  0.3746137
PC2 -0.4023883
```

```r
#Testing LDA
predict_lda <- predict(lda1, test)$class

#Confusion matrix and accuracy
tab1 <- table(Predicted = predict_lda, Actual = test$Gender)
print(tab1)
```

```
         Actual
Predicted Female Male
   Female    123   76
   Male       27   58
```

```r
print(sum(diag(tab1))/sum(tab1))
```

```
[1] 0.6373239
```

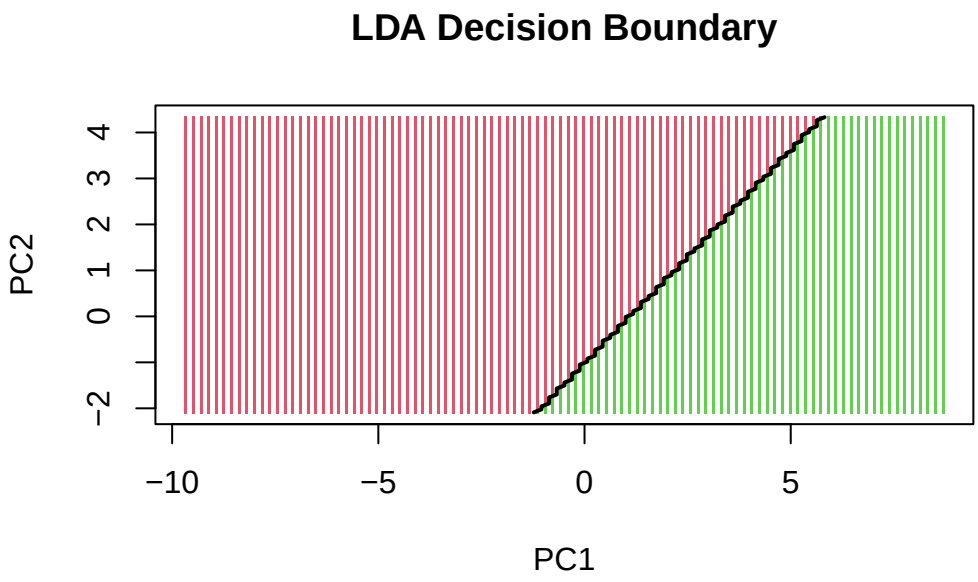
The accuracy for LDA is around 63% which is not that great. This may be because of performing PCA before LDA.

We plot the decision boundary for the LDA

```r
#Plot decision boundary
boundary <- function(model, data, class = NULL, predict_type = "class",
  resolution = 100, showgrid = TRUE, ...) {
```

```
  if(!is.null(class)) cl <- data[,class] else cl <- 1
  data <- data[,1:2]
  k <- length(unique(cl))

  plot(data, col = as.integer(cl)+1L, pch = as.integer(cl)+1L, ...)

  r <- sapply(data, range, na.rm = TRUE)
  xs <- seq(r[1,1], r[2,1], length.out = resolution)
  ys <- seq(r[1,2], r[2,2], length.out = resolution)
  g <- cbind(rep(xs, each=resolution), rep(ys, time = resolution))
  colnames(g) <- colnames(r)
  g <- as.data.frame(g)

  p <- predict(model, g, type = predict_type)
  if(is.list(p)) p <- p$class
  p <- as.factor(p)

  if(showgrid) points(g, col = as.integer(p)+1L, pch = ".")

  z <- matrix(as.integer(p), nrow = resolution, byrow = TRUE)
  contour(xs, ys, z, add = TRUE, drawlabels = FALSE,
    lwd = 2, levels = (1:(k-1))+.5)

  invisible(z)
}

boundary(lda1, train, class = "Gender", main = "LDA Decision Boundary")
```

```
Warning in plot.xy(xy, type, ...): NAs introduced by coercion

Warning in plot.xy(xy, type, ...): NAs introduced by coercion
```



QDA

Now we perform QDA and check how it compares to LDA.

```
#Splitting the data into train and test
sample <- sample(c(TRUE, FALSE), nrow(data_facial_2),
                 replace=TRUE, prob=c(0.7,0.3))
train_q <- data_facial_2[sample, ]
test_q <- data_facial_2[!sample, ]
```

```
#Fitting QDA
qda1 <- qda(Gender ~ ., data = train_q)
qda1
```

```
Call:
qda(Gender ~ ., data = train_q)

Prior probabilities of groups:
  Female     Male
0.534384 0.465616
```

```
Group means:
       Max1R13_1    T_RC1    T_LC1     RCC1 canthiMax1  T_FHCC1  T_FHLC1
Female  35.57347 35.65478 35.63190 35.21021   35.78201 34.52954 34.56804
Male    35.91288 35.96637 35.91535 35.57123   36.08283 35.51021 34.94940
        T_FHTC1
Female 34.64668
Male   34.84312
```

```
#Testing QDA
predict_qda <- predict(qda1, test_q)$class

#Confusion matrix and accuracy
tab2 <- table(Predicted = predict_qda, Actual = test_q$Gender)
print(tab2)
```

```
         Actual
Predicted Female Male
   Female    162   32
   Male       12   91
```

```
print(sum(diag(tab2))/sum(tab2))
```

```
[1] 0.8518519
```

The accuracy for QDA is around 85% which is much higher than that of LDA meaning QDA performs better than LDA. A QDA is usually better for large datasets like in this case and LDA is suitable for smaller datasets.

Question 5

We perform PCA on the facial temperature data. Before that we standardise the dataset as PCA is sensitive to the scale of the data.

```
#Standardise the data
data_facial <- scale(data_facial)
#Perform PCA
pca_fit <- prcomp(data_facial)
```

```r
summary(pca_fit)
```

```
Importance of components:
                          PC1    PC2     PC3     PC4    PC5     PC6     PC7
Standard deviation     2.539 0.9660 0.54076 0.36252 0.3323 0.23785 0.13630
Proportion of Variance 0.806 0.1166 0.03655 0.01643 0.0138 0.00707 0.00232
Cumulative Proportion  0.806 0.9226 0.95916 0.97559 0.9894 0.99646 0.99878
                           PC8
Standard deviation     0.09871
Proportion of Variance 0.00122
Cumulative Proportion  1.00000
```

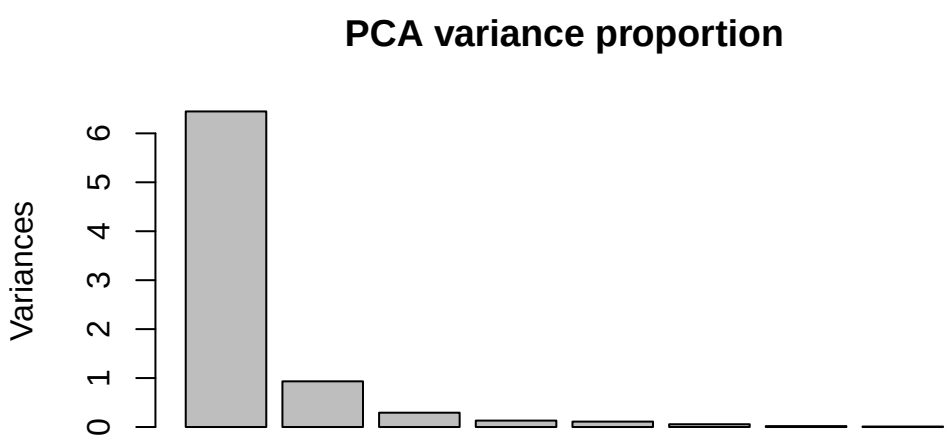
It can be seen that just 2 components explain about 92% of the variance and therefore we only select 2 components.

```r
round(pca_fit$rotation, 2)
```

```
             PC1   PC2   PC3   PC4   PC5   PC6   PC7   PC8
Max1R13_1  0.38  0.24  0.06  0.19 -0.14 -0.46  0.70 -0.19
T_RC1      0.38  0.25  0.04  0.05  0.01 -0.46 -0.43  0.62
T_LC1      0.37  0.24  0.01 -0.47  0.41  0.48  0.32  0.29
RCC1       0.37  0.22  0.01  0.41 -0.54  0.57 -0.13  0.00
canthiMax1 0.38  0.26  0.03 -0.20  0.23 -0.09 -0.45 -0.70
T_FHCC1    0.33 -0.43 -0.47  0.50  0.48  0.06  0.00 -0.01
T_FHLC1    0.33 -0.47 -0.40 -0.52 -0.48 -0.10 -0.01 -0.01
T_FHTC1    0.29 -0.54  0.79  0.03  0.04  0.03 -0.01  0.00
```

Looking at the loadings, first component seems to be looking at the overall temperature of the face as every loading is positive. The second component seems to be looking at the temperature of the face excluding the forehead area the loadings for the forehead temperatures (T_FHCC1, T_FHLC1, T_FHTC1) are all negative.

```r
plot(pca_fit, main = "PCA variance proportion")
```



From the plot too it can be understood that only 2 components will be needed.

Question 6

We calculate the PCA scores for each subject using the data without using the predict function.

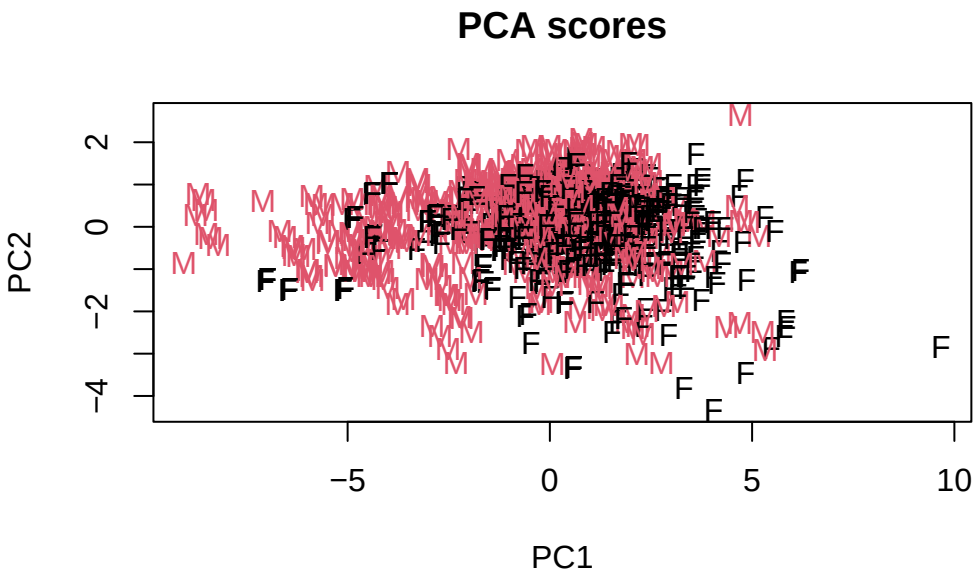
```
#Calculate the pca score from the data
#Construct covariance matrix
cov_data_facial <- cov(data_facial)
#Find eigenvalues and eigenvectors
e_list <- eigen(cov_data_facial)
e_list$vectors
```

```
           [,1]       [,2]         [,3]        [,4]        [,5]        [,6]
[1,] -0.3767358 -0.2436132 -0.056175095 -0.19467285 -0.14444767  0.45685833
[2,] -0.3781544 -0.2489547 -0.038624460 -0.05387214  0.01368148  0.46064255
[3,] -0.3704190 -0.2352991 -0.008566334  0.47331168  0.41067057 -0.47556531
[4,] -0.3692963 -0.2233179 -0.009619778 -0.41297017 -0.54431583 -0.57423958
[5,] -0.3768125 -0.2624093 -0.032927591  0.20037616  0.22772495  0.09135048
[6,] -0.3297980  0.4348997  0.466106473 -0.50357697  0.47750534 -0.05633592
```

```
[7,] -0.3259518  0.4708156  0.398057575  0.51975446 -0.48212417  0.10442900
[8,] -0.2906552  0.5420091 -0.786382403 -0.02732959  0.04127877 -0.02741154
             [,7]          [,8]
[1,]  0.700612340 -1.929436e-01
[2,] -0.433470531  6.247810e-01
[3,]  0.324154886  2.888559e-01
[4,] -0.130634196 -4.043117e-03
[5,] -0.445986389 -6.991505e-01
[6,]  0.004699760 -6.586249e-03
[7,] -0.005646422 -1.061033e-02
[8,] -0.012088650 -1.038958e-06
```

```r
#Calculate the pca score
pca_score <- data_facial %*% e_list$vectors
```

```r
#Plotting the pca scores
test_factor <- as.factor(data_facial_2$Gender)
plot(pca_score[,1], pca_score[,2], type="n", xlab="PC1", ylab="PC2",
     main = "PCA scores")
text(pca_score[,1], pca_score[,2], labels = substr(data_facial_2[,9],1,1),
     col = as.integer(test_factor))
```



From the score plot, most points are in the centre which is probably the average temperature of the subjects. A few outliers can be seen on the left and right sides of the plot. There is no big distinction of clusters but maybe there are 2 clusters which can been seen by the points on the left and right side of the plot. This could be an indication for clusters for higher and lower facial temperatures.

Question 7

Principal Components Regression (PCR) is a regression technique which is like a combination of Principal Component Analysis (PCA) and Multiple Linear Regression. It is mainly used for data that has a large number of variables and suffers from multicollinearity which results in larger standard errors and affects the accuracy of the regression coefficients.

Working

PCA is first performed on the data to reduce the dimensionality of the data. We will then have to select an appropriate number of principal components. This can be based on the total variance explained by the principal components.

Once the principal components are selected, they are used as predictors in a multiple linear regression model. This will estimate the coefficients and the model now can then be used for predictions on other data.

Since PCA is being done in PCR, the main decision to be made is the number of principal components to be used for prediction.

Advantages

- PCR reduces the dimensionality of the dataset making it suitable for wide data (variables $>$ observations).
- PCR also helps in reducing collinearity which would have been a problem for linear regression.

Disadvantages

- PCR works well only with linear data. In case of non-linear relationships, PCR will not perform as well.
- Another drawback seen in PCR is that it does not take into account of the response variable while selecting the principal components. The selection is done by looking at the variance explained by principal components.

Question 8

We split the data into train and test for PCR. We decide to do 70-30 split as we have close to 1000 observations and this split should be enough for both training and testing.

```
library(pls)
```

```
Warning: package 'pls' was built under R version 4.3.3
```

```
Attaching package: 'pls'

The following object is masked from 'package:stats':

    loadings
```

```
#Splitting the data into train and test
data_facial_pcr <- cbind(data[,1:8], AveOralM = data[,11])
sample <- sample(c(TRUE, FALSE), nrow(data_facial_pcr),
                 replace=TRUE, prob=c(0.7,0.3))
train_pcr <- data_facial_pcr[sample, ]
test_pcr <- data_facial_pcr[!sample, ]
y_test <- test_pcr[,9]
```

We fit the PCR model.

```
#Fitting the model
pcr_model <- pcr(AveOralM ~ ., data = train_pcr)
summary(pcr_model)
```

```
Data:   X dimension: 672 8
    Y dimension: 672 1
Fit method: svdpc
Number of components considered: 8
TRAINING: % variance explained
          1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps
X           77.09    90.58    96.15    97.98    99.17    99.73    99.91
AveOralM    62.13    70.82    71.42    71.58    72.55    72.58    73.02
          8 comps
X          100.00
AveOralM    73.08
```

From the summary it can be seen that the first two components explain around 91% of the variance. Therefore we decide to use only 2 components for prediction on the new data.

```
pcr_pred <- predict(pcr_model, test_pcr[,1:8], ncomp=2)
head(pcr_pred)
```

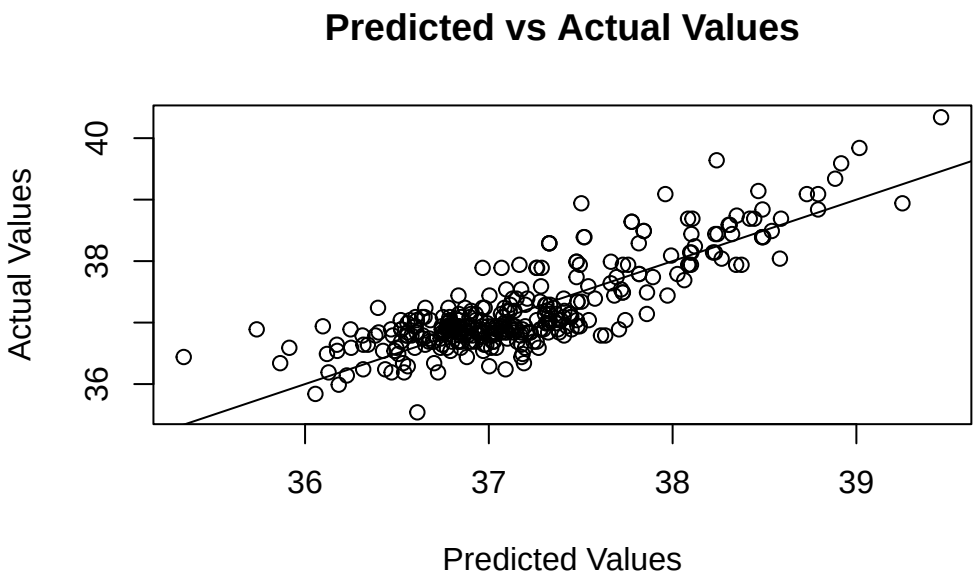
```
, , 2 comps

   AveOralM
4  37.32928
5  36.55043
6  36.57930
8  37.09506
9  36.47090
15 36.47712
```

```
sqrt(mean((pcr_pred - y_test)^2, na.rm = TRUE))
```

```
[1] 0.4067399
```

We get an RMSE close to 0.37. This means that the PCR model is relatively for good for accurately predicting on new data.

```
plot(x = pcr_pred, y = y_test,
     xlab = "Predicted Values",
     ylab = "Actual Values",
     main = "Predicted vs Actual Values")
abline(a=0, b=1)
```



When we plot the predicted values and actual values, it can be seen that the points are pretty regression line so the model is a pretty good fit for this data.