Appendix

Loading Packages

```
library(tidyverse)
library(ggplot2)
library(GGally)
library(plotly)
library(MASS)
library(car)
library(pls)
```

PCA__QDA__Function

```
# Importing Data setets
Training<-read_csv("Training Data.csv") # This is for training set
Testing<-read_csv("Testing Data.csv") # This is for testing set

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

# Function for Data analysis
Data_Analysis<-function(Training, Testing){

  ## Training Data prep
  Training <- rename(Training, Concentration = `Palm olein concentration(C)`,
                     Replicate = `Replicate No`, W=`Wave Number (cm-1)(W)`,
                     A=`Absorption (A)`)

  # Filtering Wavelenghts
  filtertraindata2<-Training %>% filter(W>=3000 & W<=3010)
  filtertraindata3<-Training %>% filter(W>=1650 & W<=1660)
  filtertraindata4<-Training %>% filter(W>=1105 & W<=1120)

  # Combining filtered datasets
  filtertraindata5<-rbind(filtertraindata2, filtertraindata3, filtertraindata4)

  # Selecting VCO and adulterated
  VCOtraindata<-filtertraindata5 %>% filter(Series=="Pure VCO")
  Adulttraindata<-filtertraindata5 %>% filter(Series=="Adulterated")

  # Combining datasets
  PCAtraindata<-rbind(VCOtraindata, Adulttraindata)
```

```
# Putting PCA data in wider format
PCAtraindata<-pivot_wider(PCAtraindata, names_from = W, values_from = A)
PCAtraindata<-PCAtraindata %>% mutate(Index=1:n()) %>%
  relocate(Index, .before = Series)

pcatraindata <-  PCAtraindata
pcatraindata_v2 <- pcatraindata %>%
  subset(select = -c(Index, Series, Concentration, Replicate))

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

# Testing data prep
Testing <- rename(Testing, Concentration = `Palm olein concentration(C)`,
  Replicate = `Replicate No`, W=`Wave Number (cm-1)(W)`, A=`Absorption (A)`)

# Filtering Wavelenghts
filtertestdata2<-Testing %>% filter(W>=3000 & W<=3010)
filtertestdata3<-Testing %>% filter(W>=1650 & W<=1660)
filtertestdata4<-Testing %>% filter(W>=1105 & W<=1120)

# Combining filtered datasets
filtertestdata5<-rbind(filtertestdata2, filtertestdata3, filtertestdata4)

# Selecting VCO and adulterated
VCOtestdata<-filtertestdata5 %>% filter(Series=="Pure VCO")
Adulttestdata<-filtertestdata5 %>% filter(Series=="Adulterated")

# Combining datasets
PCAtestdata<-rbind(VCOtestdata, Adulttestdata)

# Putting PCA data in wider format
PCAtestdata<-pivot_wider(PCAtestdata, names_from = W, values_from = A)
PCAtestdata<-PCAtestdata %>% mutate(Index=1:n()) %>%
  relocate(Index, .before = Series)

pcatestdata <-  PCAtestdata
pcatestdata_v2 <- pcatestdata %>%
  subset(select = -c(Index, Series, Concentration, Replicate))

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

## PCA Analysis

# PC calculation
pc <- prcomp(pcatraindata_v2,
             center = TRUE,
             scale. = TRUE)

# Scree Plot
#calculate total variance explained by each principal component
var_explained <- pc$sdev^2 / sum(pc$sdev^2)

#create scree plot
```

```
p<-qplot(c(1:38), var_explained) +
  geom_line() +
  xlab("Principal Component") +
  ylab("Variance Explained") +
  ggtitle("Scree Plot") +
  ylim(0, 1)

# Summary of PCA
Summary_PCA<-summary(pc)

# extracting PCA scores
Y1 <- pc$x[ , 1]
Y2 <- pc$x[ , 2]
Y3 <- pc$x[ , 3]

# PC data
PC_Scores <- cbind(pcatraindata[,1:4], Y1, Y2, Y3) %>% as.data.frame()


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

## DA Analysis

# load the data
DA_data <- PC_Scores

# Group into Pure VCO and Adulterated
Pure_VCO <- DA_data %>% filter(Series == "Pure VCO")
Adulterated <- DA_data %>% filter(Series == "Adulterated")

# Checking the Assumption of Equal Covariance
# Levene's test
levene_data <- rbind(Pure_VCO, Adulterated)
levene_result_Y1 = leveneTest(Y1 ~ Series, levene_data)
levene_result_Y2 = leveneTest(Y2 ~ Series, levene_data)
levene_result_Y3 = leveneTest(Y3 ~ Series, levene_data)

# QDA
QDA_test_data = subset(DA_data, select = -c(Index, Concentration, Replicate))

qda_results <- qda(Series~., QDA_test_data)

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

## Prediction from testing set

# Predicting PCs
test_pcs <- predict(pc, newdata = pcatestdata_v2)
test_pcs3 <- data.frame(Y1 = test_pcs[,1],
                        Y2 = test_pcs[,2],
                        Y3 = test_pcs[,3])
test_pcs3 <- as.data.frame(cbind(Series=pcatestdata$Series, test_pcs3))
```

```
  # Confusion Matrix
  pred <- predict(qda_results, test_pcs3)$class

  confusion_matrix <- table(Predicted = pred, Actual = test_pcs3$Series)


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

  ## Outputs
  list(`Scree Plot`=p, `PCA Summary`=Summary_PCA,
       `Levene test for PCA1` =levene_result_Y1,
       `Levene test for PCA2` =levene_result_Y2,
       `Levene test for PCA3` =levene_result_Y3, `QDA Results`= qda_results,
       `Confusion Matrix`= confusion_matrix)
}

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

Data_Analysis(Training, Testing)
```

PLS_Function1

```
# Importing Data sets
Training<-read_csv("Training Data.csv") # This is for training set
Testing<-read_csv("Testing Data.csv") # This is for testing set


#=============================================================================

# Function
PLS_Function1<-function(Training, Testing){
  ## Training Data
  Training <- rename(Training, Concentration = `Palm olein concentration(C)`,
    Replicate = `Replicate No`, W=`Wave Number (cm-1)(W)`, A=`Absorption (A)`)

  # Filtering Wavelenghts
  filtertraindata2<-Training %>% filter(W>=3000 & W<=3010)
  filtertraindata3<-Training %>% filter(W>=1650 & W<=1660)
  filtertraindata4<-Training %>% filter(W>=1105 & W<=1120)

  # Combining filtered datasets
  filtertraindata5<-rbind(filtertraindata2, filtertraindata3, filtertraindata4)

  # Selecting adulterated
  Adulttraindata <-filtertraindata5 %>% filter(Series=="Adulterated")


  # Putting PCA data in wider format
 PLStraindata<-pivot_wider(Adulttraindata, names_from = W, values_from = A)
 PLStraindata <-PLStraindata %>% mutate(Index=1:n()) %>%
   relocate(Index, .before = Series)

 # Converting concentration to numeric vector in training data
 PLStraindata <- PLStraindata %>%
   separate(Concentration, into = c("Concentration", "percent"))
 PLStraindata$Concentration <- as.numeric(PLStraindata$Concentration)/100

 PLStraindata <- PLStraindata %>%
   subset(select = -c(Index, Series, percent, Replicate))


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

 ## Testing Data prep
 Testing <- rename(Testing, Concentration = `Palm olein concentration(C)`,
    Replicate = `Replicate No`, W=`Wave Number (cm-1)(W)`, A=`Absorption (A)`)

 # Filtering Wavelenghts
 filtertestdata2<- Testing %>% filter(W>=3000 & W<=3010)
 filtertestdata3<- Testing %>% filter(W>=1650 & W<=1660)
 filtertestdata4<- Testing %>% filter(W>=1105 & W<=1120)

 # Combining filtered datasets
 filtertestdata5<-rbind(filtertestdata2, filtertestdata3, filtertestdata4)

 # Selecting adulterated
 Adulttestdata <-filtertestdata5 %>% filter(Series=="Adulterated")


 # Putting PCA data in wider format
 PLStestdata<-pivot_wider(Adulttestdata, names_from = W, values_from = A)
 PLStestdata <-PLStestdata %>% mutate(Index=1:n()) %>%
   relocate(Index, .before = Series)

 # Coverting concentration to numeric vector
 PLStestdata <- PLStestdata %>%
   separate(Concentration, into = c("Concentration", "percent"))
 PLStestdata$Concentration <- as.numeric(PLStestdata$Concentration)/100

 PLStestdata <- PLStestdata %>%
   subset(select = -c(Index, Series, percent, Replicate))

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


 # Fitting model for training data set

 # Split the column names in X and Y
 X_colnames <- colnames(PLStraindata)[2:39]
 Y_colnames <- colnames(PLStraindata)[1]

 # Split train data into matrices
 X_train_matrix <- as.matrix(PLStraindata[X_colnames])
```

```
 Y_train_matrix <- as.matrix(PLStraindata[Y_colnames])

 # PLS Regression
 pls <- plsr(Y_train_matrix ~ X_train_matrix, scale=TRUE, validation="CV")

 summary <- summary(pls)

 # Create a plot to define the number of components
 plot(RMSEP(pls))

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

 ## Prediction for testing dataset

 # prediction
 pcr_pred <- predict(pls, PLStestdata[,2:39], ncomp=8)
 predicted_PCR <- pcr_pred*100

 #calculate RMSE
 RMSE <- sqrt(mean((pcr_pred - PLStestdata$Concentration)^2))

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

 ## Outputs

 list(`PLS Summary`= summary, `Predicted values for testing set` = predicted_PCR,
      RMSE = RMSE)


}
#==============================================================================

# Inputs for function

PLS_Function1(Training, Testing)
```

PLS_Function2

```
# Importing Data setets
Training<-read_csv("Training Data.csv") # This is for training set
Validation <- read_csv("Validation.csv") # This is for validation set



#===============================================================================

# Function
PLS_Function2 <-function(Training, Validation){
  ## Training Data
  Training <- rename(Training, Concentration = `Palm olein concentration(C)`,
   Replicate = `Replicate No`, W=`Wave Number (cm-1)(W)`, A=`Absorption (A)`)

  # Filtering Wavelenghts
  filtertraindata2<-Training %>% filter(W>=3000 & W<=3010)
  filtertraindata3<-Training %>% filter(W>=1650 & W<=1660)
  filtertraindata4<-Training %>% filter(W>=1105 & W<=1120)

  # Combining filtered datasets
  filtertraindata5<-rbind(filtertraindata2, filtertraindata3, filtertraindata4)

  # Selecting adulterated
  Adulttraindata <-filtertraindata5 %>% filter(Series=="Adulterated")


  # Putting PCA data in wider format
  PLStraindata<-pivot_wider(Adulttraindata, names_from = W, values_from = A)
  PLStraindata <-PLStraindata %>% mutate(Index=1:n()) %>%
    relocate(Index, .before = Series)

  # Converting concentration to numeric vector in training data
  PLStraindata <- PLStraindata %>%
    separate(Concentration, into = c("Concentration", "percent"))
  PLStraindata$Concentration <- as.numeric(PLStraindata$Concentration)/100

  PLStraindata <- PLStraindata %>%
    subset(select = -c(Index, Series, percent, Replicate))


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

  # Fitting model for training data set

  # Split the column names in X and Y
  X_colnames <- colnames(PLStraindata)[2:39]
  Y_colnames <- colnames(PLStraindata)[1]

  # Split train data into matrices
  X_train_matrix <- as.matrix(PLStraindata[X_colnames])
  Y_train_matrix <- as.matrix(PLStraindata[Y_colnames])

  # PLS Regression
  pls <- plsr(Y_train_matrix ~ X_train_matrix, scale=TRUE, validation="CV")

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

  # Prediction for validation set

  ## Validation Data
  Validation <- rename(Validation, Concentration = `Palm olein concentration(C)`,
  Replicate = `Replicate No`, W=`Wave Number (cm-1)(W)`, A=`Absorption (A)`)

  # Filtering Wavelenghts
  filterValidationdata2<-Validation %>% filter(W>=3000 & W<=3010)
```

```
  filterValidationdata3<-Validation %>% filter(W>=1650 & W<=1660)
  filterValidationdata4<-Validation %>% filter(W>=1105 & W<=1120)

  # Combining filtered datasets
  filterValidationdata5<-rbind(filterValidationdata2, filterValidationdata3,
                               filterValidationdata4)

  # Selecting adulterated
  AdultValidationdata <-filterValidationdata5 %>% filter(Series=="Adulterated")


  # Putting PCA data in wider format
  PLSValidationdata<-pivot_wider(AdultValidationdata, names_from = W, values_from = A)
  PLSValidationdata <-PLSValidationdata %>% mutate(Index=1:n()) %>%
    relocate(Index, .before = Series)

  # Converting concentration to numeric vector in training data
  PLSValidationdata <- PLSValidationdata %>%
    separate(Concentration, into = c("Concentration", "percent"))
  PLSValidationdata$Concentration <- as.numeric(PLSValidationdata$Concentration)/100

  PLSValidationdata <- PLSValidationdata %>%
    subset(select = -c(Index, Series, percent, Replicate))

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

  ## Prediction for validation dataset

  # prediction
  pcr_validpred <- predict(pls, PLSValidationdata[,2:39], ncomp=8)
  validpredicted_PCR <- pcr_validpred*100

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

  ## Outputs

  list(`Predicted values for validation set` = validpredicted_PCR)
}


#=========================================================================

# Inputs for function

PLS_Function2(Training, Validation)
```