Foundation of Financial Data Science (FE 582)

(Homework 3)

Prof. Dragos Bozdog

Student Name: **Paras Garg**

Course Section: **FE 582 A**

Problem 1 –

Follow the example in Lecture 5 R code to analyze the pair trading strategy for several pairs taken from the following list: PEP, KO, DPS for a period of at least 10 years. You may download stock data from WRDS (you can register and create an account as Stevens student at https://wrds-web.wharton.upenn.edu/wrds/) or other free data sources such that Yahoo. Include transaction cost of 0.1% (or 10 basis points) for each transaction. Discuss the results.

Analysis –

```
# Environment Setup
rm(list = ls())
setwd("E:/FE Assignment/")
```

```
# Read data (downloaded from yahoo)
# URLs:
#   PEP:
https://finance.yahoo.com/quote/PEP/history?period1=1167627600&period2=1509854400&interval=1d&filter=history&frequency=1d
#   KO:
https://finance.yahoo.com/quote/KO/history?period1=1167627600&period2=1509854400&interval=1d&filter=history&frequency=1d
#   DPS:
https://finance.yahoo.com/quote/DPS/history?period1=1167627600&period2=1509854400&interval=1d&filter=history&frequency=1d
pepData <- read.table(file="PEP.CSV", header = TRUE, sep = ",")
koData <- read.table(file="KO.CSV", header = TRUE, sep = ",")
dpsData <- read.table(file="DPS.CSV", header = TRUE, sep = ",")

# Read data Analysis
head(pepData)
head(koData)
head(dpsData)

# Loaded data summary
summary(pepData)
pepData$Date = as.Date(pepData$Date)
summary(pepData)

summary(koData)
koData$Date = as.Date(koData$Date)
summary(koData)

summary(dpsData)
dpsData$Date = as.Date(dpsData$Date)
summary(dpsData)
```

Functions

```r
# Combine stocks
formStocksPair = function(df1, df2, stockNames = c(deparse(substitute(df1)),
                                                   deparse(substitute(df2)))) {
  inter = intersect(df1$Date, df2$Date)
  df1.sub = df1[which(df1$Date %in% inter), ]
  df2.sub = df2[which(df2$Date %in% inter), ]
  structure(data.frame(Date = as.Date(df1.sub$Date),
                      df1.sub$Adj.Close,
                      df2.sub$Adj.Close),
            names = c("Date", stockNames))
}

# Next Trade
nextTrade = function(pair, k = 1, start = 1) {
  mpairs = mean(pair)
  sdpairs = sd(pair)
  if (start > 1) {
    pair <- pair[-(1:start - 1)]
  }
  upperband = mpairs + k * sdpairs
  lowerband = mpairs - k * sdpairs
  meanupband = pair > mpairs
  meandownband = pair < mpairs
  conditionsSatisfied = pair > upperband | pair < lowerband
  if (!any(conditionsSatisfied)) {
    return(NA)
  }
  tstart = which(conditionsSatisfied)[1]
  temppair = pair > mpairs | pair < mpairs
  if (pair[tstart] > mpairs) {
    tend = which(!meanupband[tstart:length(meanupband)])[1] +
      tstart
  } else {
    tend = which(!meandownband[tstart:length(meandownband)])[1] +
      tstart
  }
  if (is.na(tend)) {
    tend = length(pair)
  }
  return(c(tstart, tend) + (start - 1))
}

# Trade Points
tradePoints = function(pair, k = 1) {
  options(warn = -1)
```

```r
  start = 1
  tpoint = data.frame(Start = integer(), End = integer())
  while (start < length(pair)) {
    temp = nextTrade(pair, k, start)
    if (is.na(temp)) {
      break
    }
    tpoint = rbind(tpoint, temp)
    start = temp[2]
  }
  names(tpoint) = c("Start", "End")
  options(warn = 0)
  return(tpoint)
}

# Plot
plotPos = function(stk1, stk2, k = 1, title = c(deparse(substitute(stk1)),
deparse(substitute(stk2)))) {
  title = paste(paste(title, collapse = " "), "pair", sep = " ")
  stk1stk2 = formStocksPair(stk1, stk2)
  pair = stk1stk2$stk1/stk1stk2$stk2
  mpairs = mean(pair)
  sdpairs = sd(pair)
  plot(stk1stk2$Date, pair, type = "l", col = "grey", ylab = "Price Ration",
       xlab = "Date", main = title)
  abline(h = c(mpairs, mpairs + k * sdpairs, mpairs - k * sdpairs),
         col = c("darkgreen", rep("red", 2 * length(k))), lty = 2)
  tpoints = tradePoints(pair, k)
  invisible(lapply(tpoints$Start, function(x) showPosition(stk1stk2$Date[x],
                                                           pair[x])))
  invisible(lapply(tpoints$End, function(x) showPosition(stk1stk2$Date[x],
                                                         pair[x], fg = "red")))
}
showPosition = function(x, y, fg = "green") {
  symbols(x, y, fg = fg, circles = 30, add = TRUE, inches = FALSE,
          bg = fg)
}
pofPostion = function(position, pair, stk1stk2, c = 0.01) {
  if (is.data.frame(position)) {
    position = as.vector(as.matrix(position))
  }
  mpair = mean(pair)
  trade = if (pair[position[1]] > mpair) {
    1
  } else {
    0
```

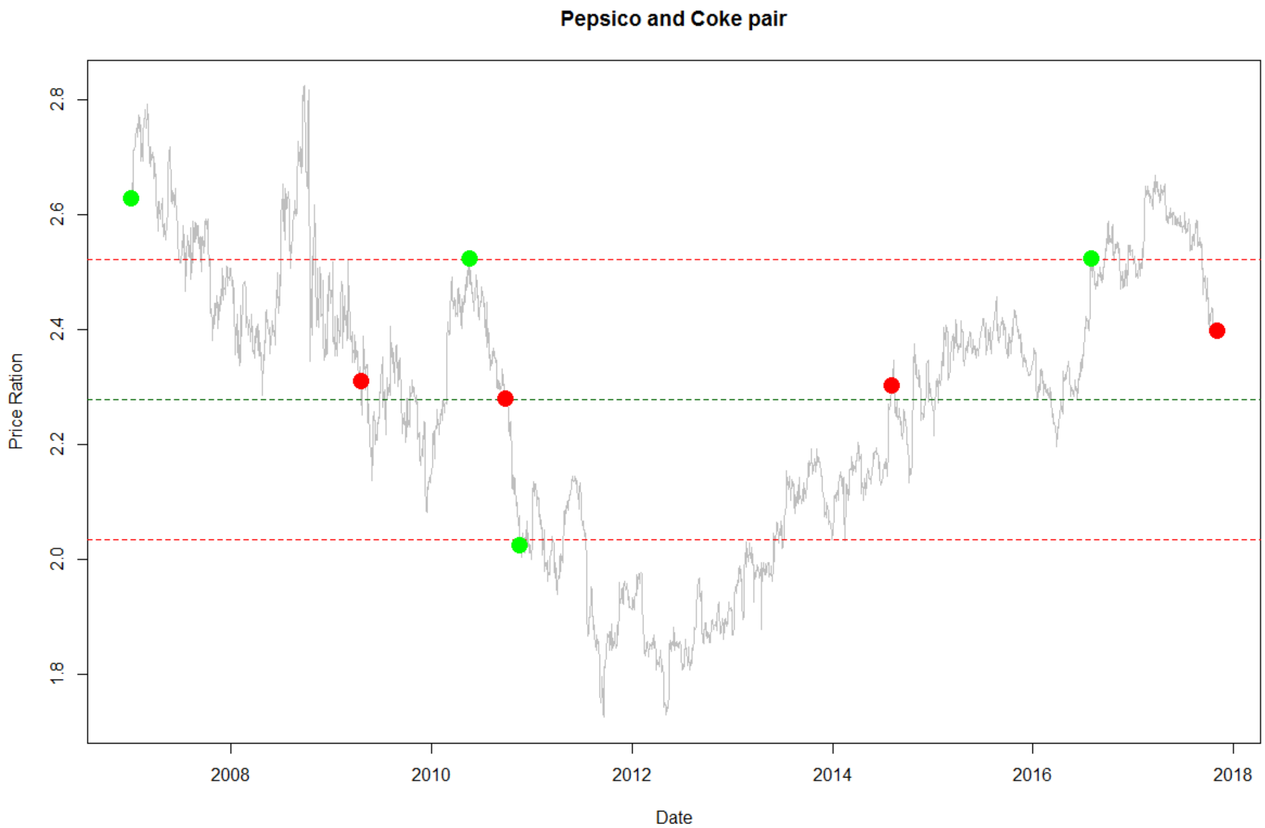
```r
  }
  unitstk1 <- 1/stk1stk2$stk1[position[1]]
  unitstk2 <- 1/stk1stk2$stk2[position[1]]
  amt <- sum(unitstk1 * stk1stk2$stk1[position[2]], unitstk2 *
             stk1stk2$stk2[position[2]])
  if (trade == 1) {
    prof <- sum(c(1 - unitstk1 * stk1stk2$stk1[position[2]],
                unitstk2 * stk1stk2$stk2[position[2]] - 1, -1 * c *
                  amt))
  } else {
    prof <- sum(c(unitstk1 * stk1stk2$stk1[position[2]] -
                  1, 1 - unitstk2 * stk1stk2$stk2[position[2]], -1 *
                  c * amt))
  }
  return(c(prof))
}

# Calculate Total Profit
calTotalProfit <- function(stk1, stk2, k = 1, c = 0.01) {
  stk1stk2 <- formStocksPair(stk1, stk2)
  pair <- stk1stk2$stk1/stk1stk2$stk2
  mpairs <- mean(pair)
  sdpairs <- sd(pair)
  tpoints <- tradePoints(pair, k)
  tprofits <- vector()
  for (i in 1:nrow(tpoints)) {
    tprofits <- c(tprofits, pofPostion(tpoints[i, ], pair, stk1stk2, c = c))
  }
  return(tprofits)
}

# Adjusting for stock split of 1:2 in coke before August 10,
k <- 1
commision <- 0.010
```

```
# For Pepsico and Coke pair

# Plot
plotPos(pepData, koData, k, title = "Pepsico and Coke")
```



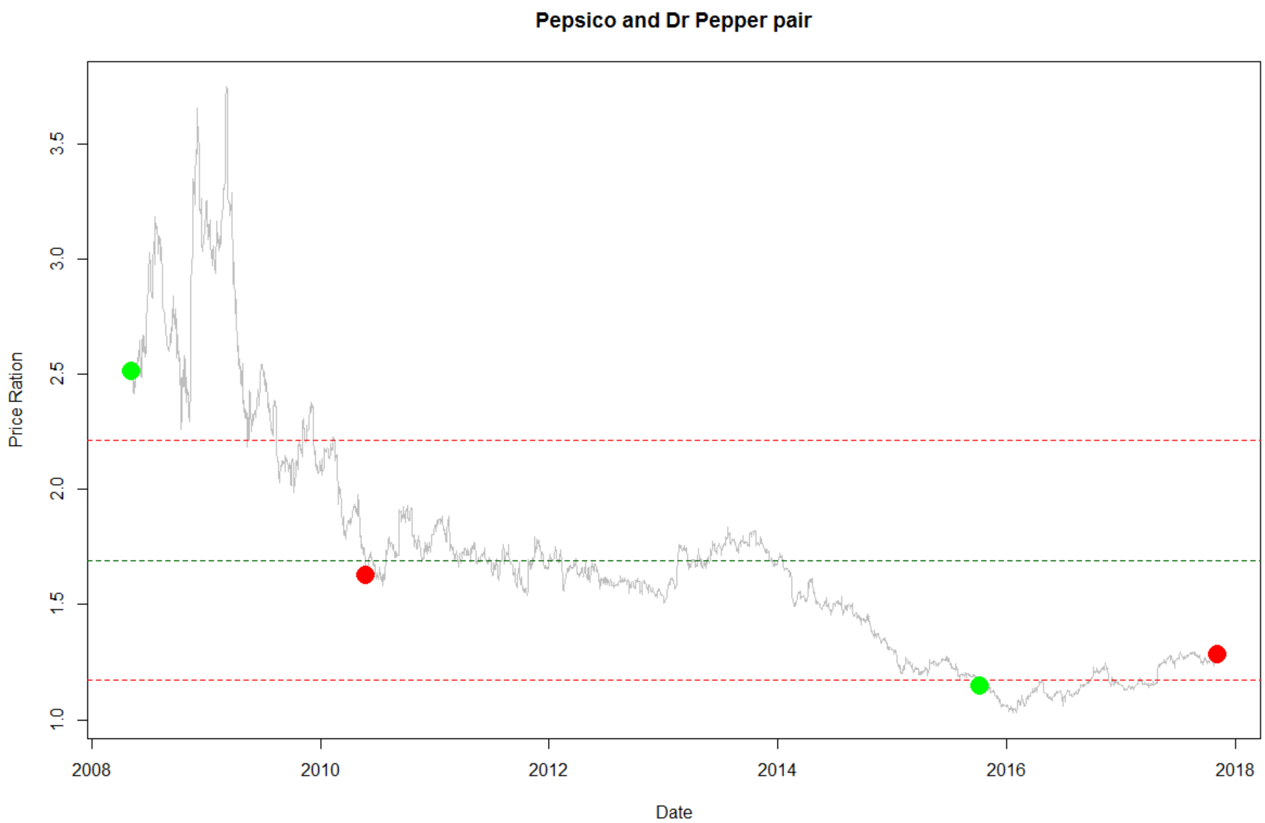

```
# Profit
calTotalProfit(pepData, koData, k = k, c = commision) #Profit

[1] 0.09671643 0.08714430 0.15745969 0.03331101
```

For Pepsico and Dr Pepper pair

Plot

```
plotPos(pepData, dpsData, k, title = "Pepsico and Dr Pepper")
```



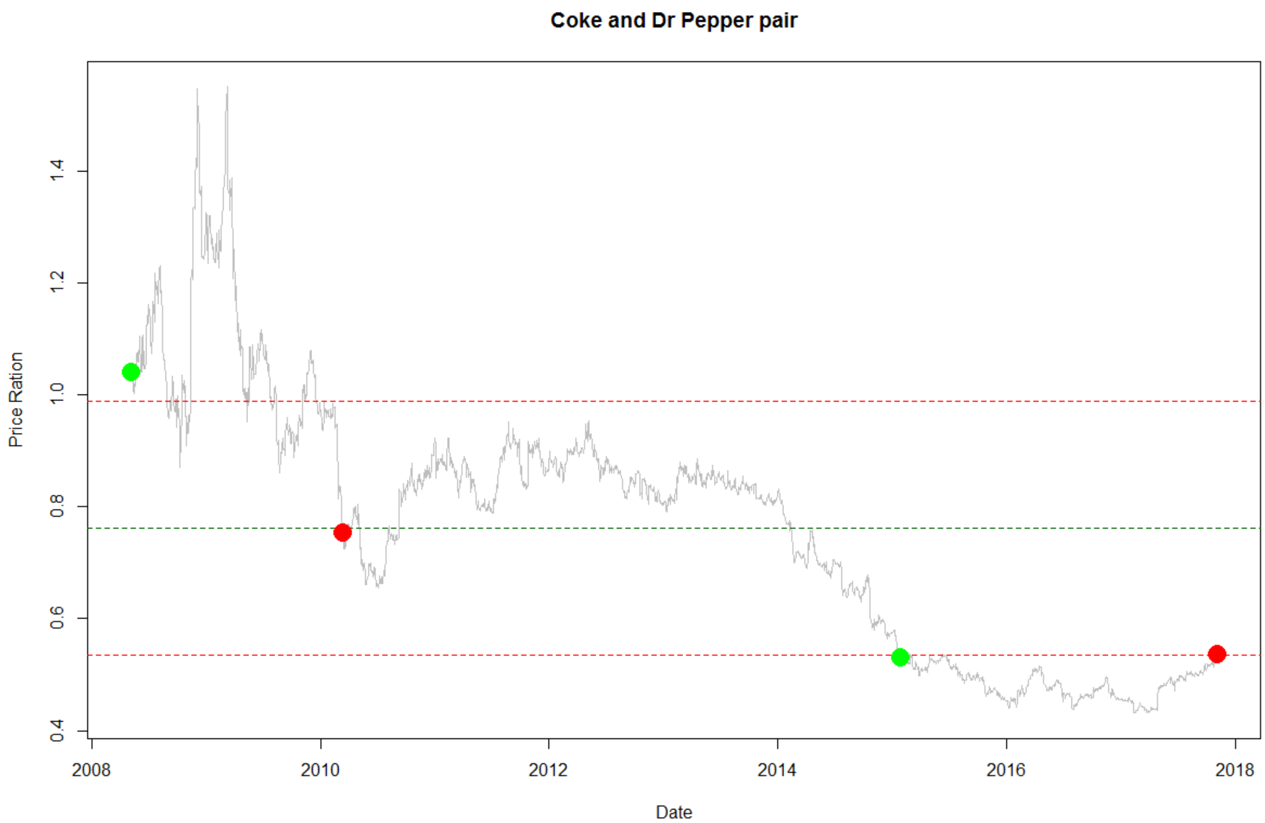
Profit

```
calTotalProfit(pepData, dpsData, k = k, c = commision) #Profit
```

```
[1] 0.4965158 0.1071199
```

```
# For Coke and Dr Pepper pair

# Plot
plotPos(koData, dpsData, k, title = "Coke and Dr Pepper")
```




```
# Profit
calTotalProfit(koData, dpsData, k = k, c = commision) #Profit
```

```
[1]  0.3572142 -0.0105421
```

Problem 2 –

Use the dataset "Default.csv" which has 7,000 observations on the following 4 variables:

- Default: A factor with levels No and Yes indicating whether the customer defaulted in their debt
- Student: A factor with levels No and Yes indicating whether the customer is a student
- Balance: The average balance that the customer has remaining on their credit card after making their monthly payment.
- Income: Income of customer

Apply logistic regression, linear discriminant analysis, quadratic discriminant analysis and K-nearest neighbor classification methods to predict customers that are likely to default in DefaultPredict.csv dataset. Please use several values of K in the KNN classification method such that you can minimize the errors. Compare the errors for all the methods and draw conclusions.

Analysis –

```
# Environment Setup#
rm(list = ls())
setwd("E:/FE Assignment/")

# Import data from the csv file
defaultData <- read.csv("Default.csv")
defaultPredictData <- read.csv("DefaultPredict.csv")
# Removing the index column
defaultData <- defaultData[,!names(defaultData)%in%"index"]
defaultPredictData <- defaultPredictData[,!names(defaultPredictData)%in%"index"]
```

 # Logistic Regression

```
# Logistic Regression
logRegFit <- glm(default ~ student + balance + income, data = defaultData,family =
binomial)
logRegPredict <- predict(logRegFit, defaultPredictData, type = "response") > 0.5
logRegPredictedDefault <- rep("No", length(logRegPredict))
logRegPredictedDefault[logRegPredict] <- "Yes"
logRegPredict <- cbind(logRegPredictedDefault, defaultPredictData)
logRegPredict
```

```
   logRegPredictedDefault student      balance   income
1                      No     Yes 1823.6626760 24905.04
2                      No      No  343.9838524 46100.00
3                      No     Yes 1543.4079290 20567.95
4                      No      No 1509.2208760 61052.94
5                      No      No 1237.5958490 89252.07
6                      No      No 1020.1992510 74755.64
7                      No      No 1772.5240950 79278.65
8                      No      No 1662.3205560 59916.36
9                      No     Yes  801.9435528 22976.07
10                     No      No 2184.5639060 68203.29
11                     No      No    0.3595209 62587.41
12                     No     Yes 1993.8585060 30085.52
```

```
13                      No      No  752.7173224 38130.18
14                      No      No 1308.0362050 82266.06
15                      No      No 2218.2017980 68288.28
16                      No      No 1874.5182410 85044.05
17                      No     Yes  368.7584176 22493.03
18                      No      No 1421.8435040 79505.71
19                      No      No 1494.3217040 29474.88
20                      No      No 1705.0553350 87942.50
21                      No      No 3135.8374280 55002.50
22                      No     Yes  374.9970976 25328.89
23                      No     Yes 2981.7718800 26849.13
24                     Yes     Yes 4409.0833880 21464.55
25                      No     Yes 3555.4169680 30544.68
26                     Yes      No 3754.2430820 73466.31
27                      No     Yes 3805.1029400 30862.90
28                      No     Yes 3143.6118200 22366.74
29                     Yes      No 3937.9976140 58626.83
30                      No      No 3052.2599140 45013.13
31                      No      No 3283.1186320 70227.38
32                     Yes      No 4009.0920100 63244.56
33                      No      No 3100.8985280 84387.15
34                      No      No 2655.6774620 52079.36
35                      No      No 3400.5958060 45687.20
36                      No     Yes 2236.2194820 32826.19
37                      No      No 2265.0498860 55863.57
38                     Yes      No 3961.4604460 42202.05
39                      No      No 3417.4048280 57525.88
40                      No      No 2914.0248860 88047.15
41                      No      No 3534.3202820 69314.53
42                      No     Yes 3518.2155900 23970.18
```

 # Linear discriminant Analysis

```
# Linear discriminant Analysis
library("MASS")
ldaFit <- lda(default ~ student + balance + income, data = defaultData)
ldaPredictedDefault <- predict(ldaFit, defaultPredictData)$class
ldaPredicted <- cbind(ldaPredictedDefault, defaultPredictData)
ldaPredicted
```

```
  ldaPredictedDefault student      balance   income
1                  No     Yes 1823.6626760 24905.04
2                  No      No  343.9838524 46100.00
3                  No     Yes 1543.4079290 20567.95
4                  No      No 1509.2208760 61052.94
5                  No      No 1237.5958490 89252.07
```

```
6                     No      No 1020.1992510 74755.64
7                     No      No 1772.5240950 79278.65
8                     No      No 1662.3205560 59916.36
9                     No     Yes  801.9435528 22976.07
10                    No      No 2184.5639060 68203.29
11                    No      No    0.3595209 62587.41
12                    No     Yes 1993.8585060 30085.52
13                    No      No  752.7173224 38130.18
14                    No      No 1308.0362050 82266.06
15                    No      No 2218.2017980 68288.28
16                    No      No 1874.5182410 85044.05
17                    No     Yes  368.7584176 22493.03
18                    No      No 1421.8435040 79505.71
19                    No      No 1494.3217040 29474.88
20                    No      No 1705.0553350 87942.50
21                    No      No 3135.8374280 55002.50
22                    No     Yes  374.9970976 25328.89
23                    No     Yes 2981.7718800 26849.13
24                   Yes     Yes 4409.0833880 21464.55
25                    No     Yes 3555.4169680 30544.68
26                    No      No 3754.2430820 73466.31
27                    No     Yes 3805.1029400 30862.90
28                    No     Yes 3143.6118200 22366.74
29                   Yes      No 3937.9976140 58626.83
30                    No      No 3052.2599140 45013.13
31                    No      No 3283.1186320 70227.38
32                   Yes      No 4009.0920100 63244.56
33                    No      No 3100.8985280 84387.15
34                    No      No 2655.6774620 52079.36
35                    No      No 3400.5958060 45687.20
36                    No     Yes 2236.2194820 32826.19
37                    No      No 2265.0498860 55863.57
38                    No      No 3961.4604460 42202.05
39                    No      No 3417.4048280 57525.88
40                    No      No 2914.0248860 88047.15
41                    No      No 3534.3202820 69314.53
42                    No     Yes 3518.2155900 23970.18
```

 # Quadratic discriminant Analysis

```
# Quadratic discriminant Analysis
qdaFit <- qda(default ~ student + balance + income, data = defaultData)
qdaPredictedDefault <- predict(qdaFit, defaultPredictData)$class
qdaPredicted <- cbind(qdaPredictedDefault, defaultPredictData)
qdaPredicted
```

	qdaPredictedDefault	student	balance	income
1	No	Yes	1823.6626760	24905.04
2	No	No	343.9838524	46100.00
3	No	Yes	1543.4079290	20567.95
4	No	No	1509.2208760	61052.94
5	No	No	1237.5958490	89252.07
6	No	No	1020.1992510	74755.64
7	No	No	1772.5240950	79278.65
8	No	No	1662.3205560	59916.36
9	No	Yes	801.9435528	22976.07
10	No	No	2184.5639060	68203.29
11	No	No	0.3595209	62587.41
12	No	Yes	1993.8585060	30085.52
13	No	No	752.7173224	38130.18
14	No	No	1308.0362050	82266.06
15	No	No	2218.2017980	68288.28
16	No	No	1874.5182410	85044.05
17	No	Yes	368.7584176	22493.03
18	No	No	1421.8435040	79505.71
19	No	No	1494.3217040	29474.88
20	No	No	1705.0553350	87942.50
21	No	No	3135.8374280	55002.50
22	No	Yes	374.9970976	25328.89
23	No	Yes	2981.7718800	26849.13
24	Yes	Yes	4409.0833880	21464.55
25	No	Yes	3555.4169680	30544.68
26	No	No	3754.2430820	73466.31
27	No	Yes	3805.1029400	30862.90
28	No	Yes	3143.6118200	22366.74
29	Yes	No	3937.9976140	58626.83
30	No	No	3052.2599140	45013.13
31	No	No	3283.1186320	70227.38
32	Yes	No	4009.0920100	63244.56
33	No	No	3100.8985280	84387.15
34	No	No	2655.6774620	52079.36
35	No	No	3400.5958060	45687.20
36	No	Yes	2236.2194820	32826.19
37	No	No	2265.0498860	55863.57
38	No	No	3961.4604460	42202.05
39	No	No	3417.4048280	57525.88
40	No	No	2914.0248860	88047.15
41	No	No	3534.3202820	69314.53
42	No	Yes	3518.2155900	23970.18

K Nearest Neighbor

```
# K nearest neighbor
library(class)
defaultData$student <- as.character(defaultData$student)
defaultData$student[defaultData$student == "Yes"] <- 1
defaultData$student[defaultData$student == "No"] <- 0
defaultData$student <- as.numeric(defaultData$student)
alpha <- defaultData[2:4]
defaultData_class <- defaultData[, 1]
student <- defaultPredictData$student
defaultPredictData$student <- as.character(defaultPredictData$student)
defaultPredictData$student[defaultPredictData$student == "Yes"] <- 1
defaultPredictData$student[defaultPredictData$student == "No"] <- 0
defaultPredictData$student <- as.numeric(defaultPredictData$student)

# For 3 nearnest neighbour, k = 3
knnPredictedDefault <- knn(alpha, defaultPredictData, defaultData_class,k = 3)
knnPredictedDefault <- cbind(knnPredictedDefault, student,defaultPredictData[2:3])
knnPredictedDefault
```

```
   knnPredictedDefault student      balance   income
1                   No     Yes 1823.6626760 24905.04
2                   No      No  343.9838524 46100.00
3                   No     Yes 1543.4079290 20567.95
4                   No      No 1509.2208760 61052.94
5                   No      No 1237.5958490 89252.07
6                   No      No 1020.1992510 74755.64
7                   No      No 1772.5240950 79278.65
8                   No      No 1662.3205560 59916.36
9                   No     Yes  801.9435528 22976.07
10                  No      No 2184.5639060 68203.29
11                  No      No    0.3595209 62587.41
12                  No     Yes 1993.8585060 30085.52
13                  No      No  752.7173224 38130.18
14                  No      No 1308.0362050 82266.06
15                  No      No 2218.2017980 68288.28
16                  No      No 1874.5182410 85044.05
17                  No     Yes  368.7584176 22493.03
18                  No      No 1421.8435040 79505.71
19                  No      No 1494.3217040 29474.88
20                  No      No 1705.0553350 87942.50
21                  No      No 3135.8374280 55002.50
22                  No     Yes  374.9970976 25328.89
23                  No     Yes 2981.7718800 26849.13
24                  No     Yes 4409.0833880 21464.55
25                  No     Yes 3555.4169680 30544.68
26                  No      No 3754.2430820 73466.31
```

27	No	Yes	3805.1029400	30862.90
28	No	Yes	3143.6118200	22366.74
29	No	No	3937.9976140	58626.83
30	No	No	3052.2599140	45013.13
31	No	No	3283.1186320	70227.38
32	Yes	No	4009.0920100	63244.56
33	No	No	3100.8985280	84387.15
34	No	No	2655.6774620	52079.36
35	No	No	3400.5958060	45687.20
36	No	Yes	2236.2194820	32826.19
37	No	No	2265.0498860	55863.57
38	Yes	No	3961.4604460	42202.05
39	Yes	No	3417.4048280	57525.88
40	No	No	2914.0248860	88047.15
41	No	No	3534.3202820	69314.53
42	No	Yes	3518.2155900	23970.18