|  |  |
| --- | --- |
|  | |
| Data Advanced Data Analytics  CA TWO | |
| Module code : B8IT109 | |
| Ciaran Finnegan  Student No : 10524150  24/05/2020 |  |
|  |  |

Table of Contents

1 Question One 3

1.1 Question 1 – from PDF 3

1.2 Output From RStudio Cloud Console 3

2 Question Two 4

2.1 Question 2 – from PDF 4

2.2 Output from RStudio Cloud Console 4

3 Question Three 5

3.1 Question 2 – from PDF 5

3.2 Output from RStudio Cloud Console 5

4 Question Four 6

4.1 Question 2 – from PDF 6

4.2 Output from RStudio Cloud Console 7

# Question One

## Question 1 – from PDF

Use in-built dataset ‘airquality’,

a) explore the general feature of dataset using appropriate R functions.

(**5 Marks**)

b) perform data cleansing if required. (**5 Marks**)

c) consider ‘Temp’ attributes and compute the central and variational measures. (**10 Marks**)

d) apply boxplot technique to detect outlier of ‘wind’ attribute if any.

(**10 Marks**)

**(Total: 30 Marks)**

## Output From RStudio Cloud Console

To follow...

# Question Two

## Question 2 – from PDF

Use dataset available on http://users.stat.ufl.edu/~winner/data/nfl2008\_fga.csv , then:

(a) Train the model using 80% of this dataset and suggest an appropriate GLM to model **homekick** to **togo, ydline** and **kicker** variables.

**(5 Marks)**

(b) Specify the significant variables on **homekick** at the level of 𝛼=0.05, and estimate the parameters of your model.

**(5 Marks)**

(c) Predict the test dataset using the trained model. **(5 Marks)**

(d) Provide the confusion matrix and obtain the probability of correctness of predictions. **(10 Marks)**

(Total: 25 Marks)

## Output from RStudio Cloud Console

To follow..

# Question Three

## Question 3 – from PDF

Using Yahoo Finance API, select a specific stock market price, apply time series analysis, consider ***‘close price’*** as your time series variable:

1. Validate the assumptions using graphical visualization. (5 Marks)
2. Fit the optimized model for ‘***close price’*** and provide the coefficient estimates for the fitted model. **(5 Marks)**
3. What is the estimated order for AR and MA? (5 Marks)

(d) Forecast h=10 step ahead prediction of ***‘close price’*** on the plot of the original time series. **(10 Marks)**

**(Total: 25 Marks)**

## Output from RStudio Cloud Console

> ## CA Two Advanced Data Analytics : Module Code B8IT109

> ## Student Name : Ciaran Finnegan

>

> ## Student Number : 10524150

>

> ## May 2020

>

>

> ## Question Three

>

> ## References used to call Yahoo Finance API

> ## http://statmath.wu.ac.at/~hornik/QFS1/quantmod-vignette.pdf

> ## https://stackoverflow.com/questions/26666254/retrieve-monthly-adjusted-stock-quotes-using-the-quantmod-package-in-r

>

>

> #########################################################################

>

> ## Using Yahoo Finance API, select a specific stock market price, apply time series analysis,

> ## consider ‘close price' as your time series variable:

>

> #########################################################################

>

>

> # 'Quantmod' Package required to access Yahoo Finance API

> library(quantmod)

> library(forecast)

>

> # Create a sata structure that contains stock quote objects

> ETF\_Data <- new.env()

>

> # Assign dates to set range for daily stock closing prices.

> # This range of values over 13 months is set to be large enough for later

> # accurate forecasting but not too large to degrade the quality of the

> # graph visuals.

> sDate <- as.Date("2016-12-01")

> eDate <- as.Date("2018-01-01")

>

>

> # Use the stock price data over the given period of time (above) for chosen company

> # This code is written to allow the selection of different company stock prices

> # to compare trends and choose different time series.

> # The stock chosen is the AIG (American International Group) for the year 2017

> ticker\_symbol = "AIG" # AIG

>

>

> # Alternative company stocks - not used.

> #ticker\_symbol = "IBM" # International Business Machines

> #ticker\_symbol = "UA" # Under Armour, Inc

>

>

> # Invoke 'getSymbols' function to retrieve to stock price data over the time period for

> # the chosen company through the Yahoo Finance API

> getSymbols(ticker\_symbol, env=ETF\_Data, from=sDate, to=eDate, src = "yahoo", symbol.lookup = TRUE)

[1] "AIG"

>

>

> # Load a dataset with the time series of the chosen company stock prices

> CompanyStockPrices = get(ticker\_symbol, envir = ETF\_Data)

> # Review the layout of the stock price information in dataset

> head(CompanyStockPrices)

AIG.Open AIG.High AIG.Low AIG.Close AIG.Volume AIG.Adjusted

2016-12-01 63.66 64.15 63.60 64.11 5942700 58.60310

2016-12-02 64.09 64.20 63.69 63.75 5651000 58.27402

2016-12-05 64.14 64.22 63.83 64.22 4706000 58.70365

2016-12-06 64.21 64.36 63.76 64.20 4942600 58.97925

2016-12-07 64.29 64.99 64.21 64.92 5113100 59.64070

2016-12-08 64.98 66.10 64.80 65.82 5196500 60.46751

>

>

> # Consider ‘close\_price' as your time series variable in the solution to this question

> # Use the quantmod function 'Cl' to isolate the time series for closing prices

> close\_price = Cl(CompanyStockPrices)

> # Review initial record in 'close\_price' time series

> head(close\_price)

AIG.Close

2016-12-01 64.11

2016-12-02 63.75

2016-12-05 64.22

2016-12-06 64.20

2016-12-07 64.92

2016-12-08 65.82

>

>

> #########################################################################

> #########################################################################

>

>

>

>

>

> ## Q.3 (Part a)

>

> ## Using Yahoo Finance API, select a specific stock market price, apply time series analysis,

> ## consider ‘close price' as your time series variable:

>

> ##(a) Validate the assumptions using graphical visualization.

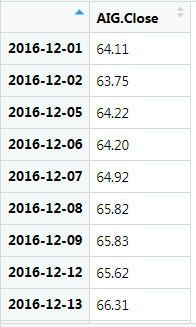
>

> ## Run functions to look at the structure of the closing price dataset for our chosen stock

> View(close\_price)

>

> ## <Screen shot of 'View' output...here>



>

> str(close\_price)

> ## <Output.. of str function on 'xts' object>

An ‘xts’ object on 2016-12-01/2017-12-29 containing:

Data: num [1:272, 1] 64.1 63.8 64.2 64.2 64.9 ...

- attr(\*, "dimnames")=List of 2

..$ : NULL

..$ : chr "AIG.Close"

Indexed by objects of class: [Date] TZ: UTC

xts Attributes:

List of 2

$ src : chr "yahoo"

$ updated: POSIXct[1:1], format: "2020-06-03 16:33:18"

>

>

>

> ## Invoke the 'ts' function on the 'close\_price' time series.

> T <- ts(close\_price, frequency = 1)

>

>

>

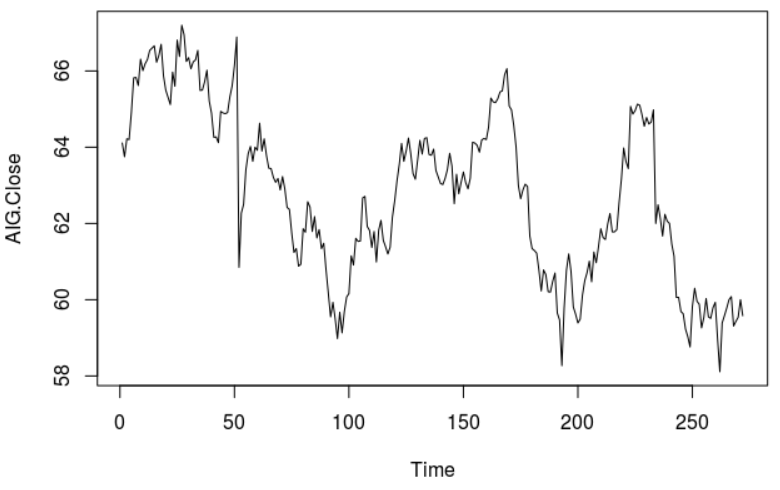
> ## Generate the plot of the time series variable- the range represents the closing prices extracted

> ## from the time range of data (frequency = 1 so every daily closing price is plotted).

> plot(T)

>

> ## <Put first plot graph here..>



>

> ## We can see that the time series is not particularly stationary in terms of mean or variance

>

>

>

## We apply 'diff' and 'log' functions to smooth out the graph plot

>

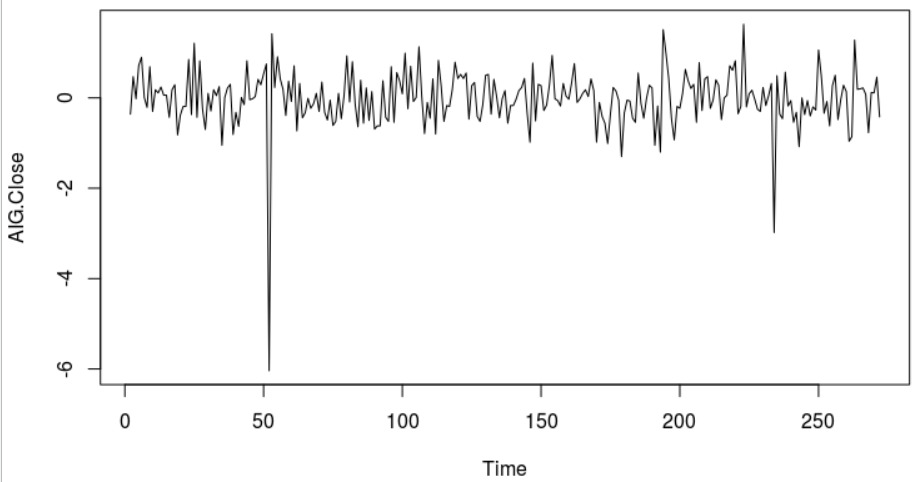
> ## Apply 'diff' function

> tssdiff=diff(T) # Stationary in mean

> plot(tssdiff)

>

> ## <Put diff plot graph here..>



>

> ## The plot of 'diff' is more stationary in mean, with an average approximately around zero.

>

>

> ## Apply log function

> tssLog = log(T)# Stationary in variance

> plot(tssLog)

>

>

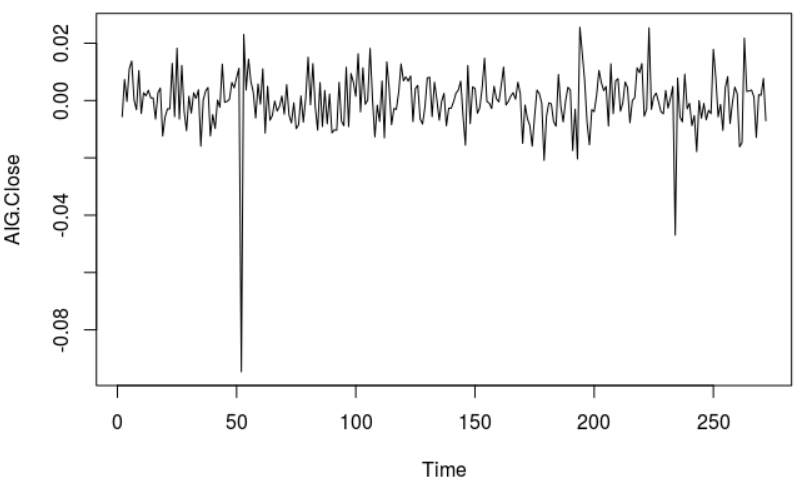
> ## Apply log function, then applying 'diff', to achieve a stationary visualisation in mean and variance

> tssdifflg = diff(log(T))

> plot(tssdifflg)

>

> ## <Put log plot graph here..>



>

> ## This graph shows mean as approximately stationary and the variance also stationary between -0.02 and 0.02, apart from a few outliers

>

>

> ## ---------------------------------------------------------------------------------------------------------

> ## ALSO - consider graphing the data to show normality in the data set maybe using ggplot

> ## ---------------------------------------------------------------------------------------------------------

>

>

>

>

>

>

>

>

>

> ## Q.3 (Part b)

>

> ## Fit the optimized model for ‘close price’ and provide the coefficient estimates for the fitted model.

>

> ## To compute optimised coefficient estimates for fitted model we have two approaches :-

> ## 1:- Apply 'acf' and 'pacf' to get estimation of order, and also estimate parameters.

> ## 2:- Apply ARIMA

>

> ## It is necessary to apply both methods and see which one has a lower AIC, then determine that method is optimised.

> ## Try and find as low a value of AIC as possible

>

>

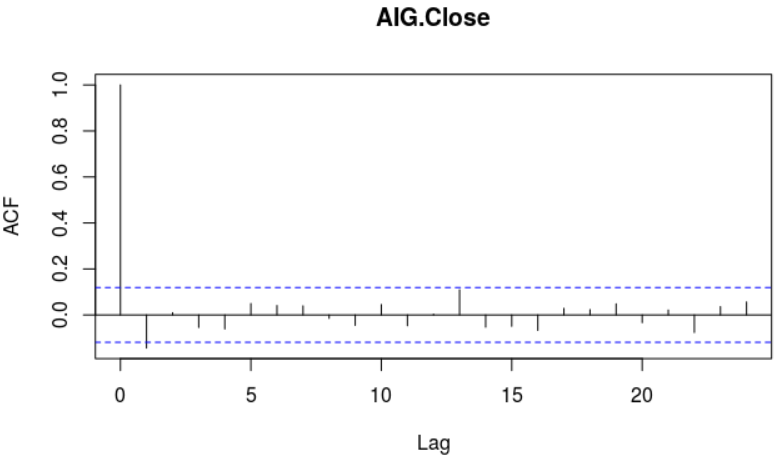
> ## 1 - Apply 'acf' and 'pacf' to get estimations of 'q' and 'p'

> ## acf = autocorrelation function. Gives us the estimation for 'q'

> acf(tssdifflg)

>

> ## <Put acf plot graph here..>



>

> ## There are two initial lags outside the boundary, therefore q = 2. (Above or below boundary line is not important).

>

>

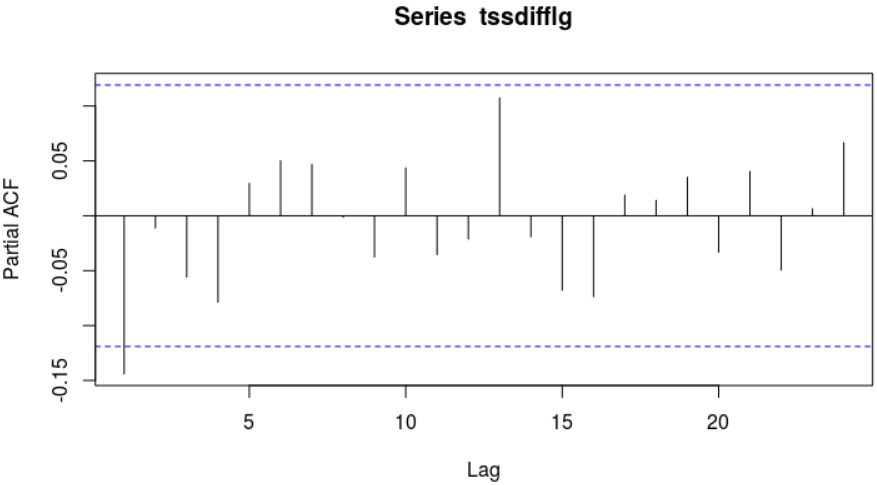
>

> ## pacf = partial autocorrelation function. Gives is the estimation for 'p'

> pacf(tssdifflg)

>

> ## <Put pacf plot graph here..>



>

> ## One initial lag is outside are outside the bounds, therefore p = 1

>

>

> ## Now use 'arima' function to fit a manual ARIMA; p = 1, (1 diff used), q = 2. The original time series with

> ## closing price is used.

> ## ARIMA (p,d,q) Model : Using original time series 'T'

> ## Parameter Estimation

> manual.fit <- arima(T, c(1,1,2)) # Fitted Model

> ## Display value of 'manual.fit'

> manual.fit

Call:

arima(x = T, order = c(1, 1, 2))

Coefficients:

ar1 ma1 ma2

-0.9986 0.8525 -0.1420

s.e. 0.0058 0.0621 0.0616

sigma^2 estimated as 0.4067: log likelihood = -262.82, aic = 533.63

>

> ## With p = 1, we see one value for the 'ar1' coefficient

> ## With q = 2, we see two values for the 'ma' (moving average) coefficients

> ## The values just under the 'ar1', 'ma1', and 'ma2' headings are the estimation

> ## of parameters

>

> ## We can see the aic (Akaike Information Criterion) value = 533.63

>

>

>

>

>

> ## Next we need to apply 'auto.arima' to generate a fitted model

> ## 'seasonal' = F - time series does not have a seasonality trend

> auto.fit <- auto.arima(T, seasonal = FALSE)

> auto.fit

Series: T

ARIMA(0,1,1)

Coefficients:

ma1

-0.1502

s.e. 0.0611

sigma^2 estimated as 0.4101: log likelihood=-263.27

AIC=530.54 AICc=530.58 BIC=537.74

>

> ## AIC = 530.54

>

>

> ## Automated coefficient are lower as 533.63 (Manual) > 530.54 (Auto, non seasonal). Therefore Auto ARIMA is better than manual fitting.

>

>

>

>

> ## Q.3 (Part c)

>

> ## What is the estimated order for AR and MA?

> auto.fit

Series: T

ARIMA(0,1,1)

Coefficients:

ma1

-0.1502

s.e. 0.0611

sigma^2 estimated as 0.4101: log likelihood=-263.27

AIC=530.54 AICc=530.58 BIC=537.74

> ## The best model shows Series: T ARIMA(0,1,1)

> ## Therefore the estimated order for AR is p = 0, and MA is q = 1

>

>

>

>

> ## Q.3 (Part d)

>

> ## Forecast h=10 step ahead prediction of wage on the plot of the

> ## original time series.

>

>

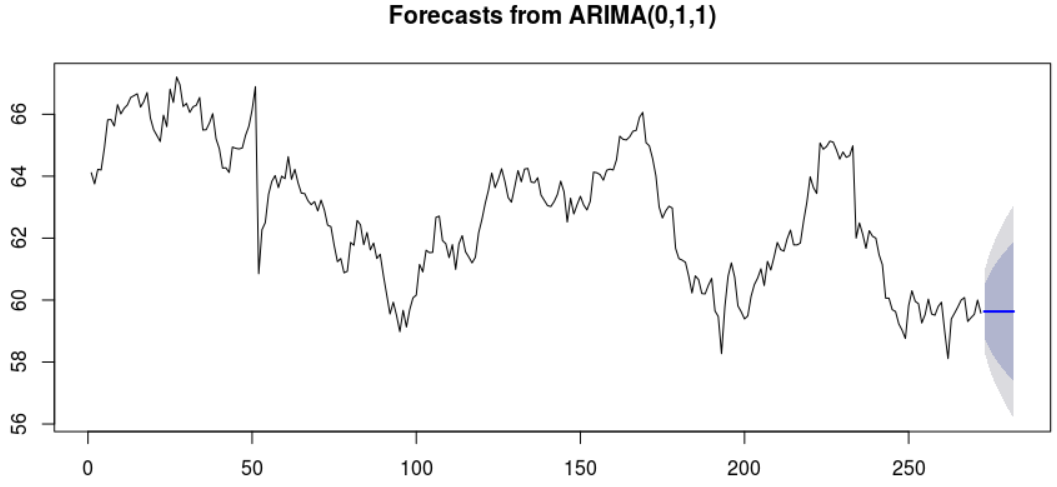
> # The best model to use is the auto fitting - as determined in the analysis in the previous

> ## steps in the question.

> auto.fcast <- forecast(auto.fit, h = 10) # Prediction for 10 step ahead

> ## Plot this forecast

> plot(auto.fcast)



>

>

> #########################################################################

> ## The forecast graph is still not particularly useful. Howeveer, if we

change the 'seasonal' = T paramer in the auto.arima function then the time series will incorpoate a seasonality trend

>

> ## It is necessary to 'force' the closing price time series into a

mult-seasonal time series in order to pick up the time pattern in the data.

>

> ## Set up mult-seasonal time series to match a working day period of 13

months (as defined when the stock data was first extracted via the Yahoo

API)

> stockSTS <- msts(T,seasonal.periods=c(5,270))

> ## Set 'D' parameter in auto.arima function so that the seasonal component in the

> ## data will be picked up

> auto.fitT <- auto.arima(stockSTS, D=1, seasonal = TRUE)

> auto.fcastT <- forecast(auto.fitT, h = 10)

> # Prediction for 10 steps ahead

>

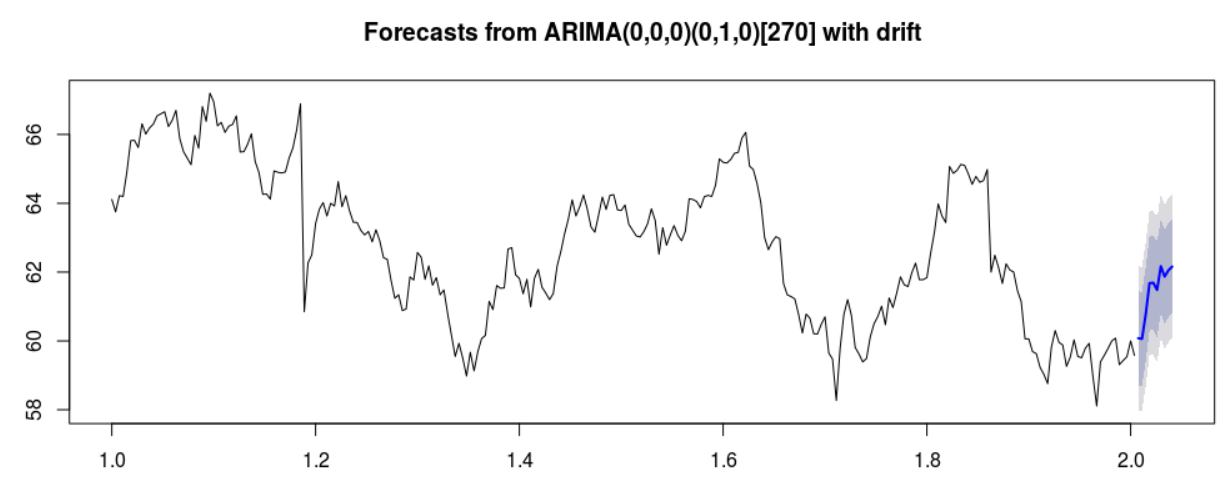
> ## Plot this forecast

> plot(auto.fcastT)

>

>

> ## <Put forecast plot graph here..>



# Question Four

## Question 4 – from PDF

Use dataset available on http://users.stat.ufl.edu/~winner/data/nfl2008\_fga.csv

1. Use LDA to classify the dataset into few classes so that at least 90% of information of dataset is explained through new classification. (**Hint**: model the variable “**qtr”** to variables “**togo**”, “**kicker**”, and “**ydline**”). How many LDs do you choose? Explain the reason.

**(5 Marks)**

*2.* Apply PCA, and identify the important principle components involving at least 90% of dataset variation. Explain your decision strategy? Plot principle components

versus their variance (**Hint**: to sketch the plot use the Scree plot). **(5 Marks)**

3. Split the dataset into two sets of variables so that **X**=( togo,kicker,ydline) and **Y**=( distance, homekick). Apply canonical correlation analysis to find the cross-correlation between **X** and **Y**. What is the correlation between ***ydline*** and ***distance***? **(5 Marks)**

4. Use K-means clustering analysis to identify the most important classes. How many classes do you select? Why?

**(6 Marks)**

**(Total: 20 Marks)**

## Output from RStudio Cloud Console

> ## CA Two Advanced Data Analytics : Module Code B8IT109

> ## Student Name : Ciaran Finnegan

>

> ## Student Number : 10524150

>

> ## May 2020

> >

> ## Question Four

>

> ## Q. 4(Part 1)

>

> ## Use LDA to classify the dataset into few classes so that at least 90% of information

> ## of dataset is explained through new classification.

> ## (Hint: model the variable “qtr” to variables “togo”, “kicker”, and “ydline”).

> ## How many LDs do you choose? Explain the reason.

> >

>

> ## Load MASS library to use LDA function

> library(MASS)

> library(CCA)

> >

> ## Read in the NFL dataset

> link='http://users.stat.ufl.edu/~winner/data/nfl2008\_fga.csv'

> datasetNFL=read.csv(link)

> head(datasetNFL)

GameDate AwayTeam HomeTeam qtr min sec kickteam def down togo kicker ydline name distance homekick kickdiff timerem offscore defscore season GOOD Missed

1 20081130 IND CLE 1 47 2 IND CLE 4 11 15 12 A.Vinatieri 30 0 -3 2822 0 3 2008 1 0

2 20081005 IND HOU 1 54 47 IND HOU 4 3 15 28 A.Vinatieri 46 0 0 3287 0 0 2008 1 0

3 20081228 TEN IND 1 45 20 IND TEN 4 3 15 10 A.Vinatieri 28 1 7 2720 7 0 2008 1 0

4 20081012 BAL IND 1 45 42 IND BAL 4 1 15 19 A.Vinatieri 37 1 14 2742 14 0 2008 1 0

5 20080907 CHI IND 1 50 56 IND CHI 4 21 15 21 A.Vinatieri 39 1 0 3056 0 0 2008 1 0

6 20081116 HOU IND 1 50 43 IND HOU 4 7 15 22 A.Vinatieri 40 1 -3 3043 0 3 2008 1 0

Blocked

1 0

2 0

3 0

4 0

5 0

6 0

>

> ## Minor Clean up of NFL dataset

> sum(is.na(datasetNFL))

[1] 4

> datasetNFL <- na.omit(datasetNFL)

> sum(is.na(datasetNFL))

[1] 0

>

> # Display the values for 'qtr'

> table(datasetNFL$qtr)

1 2 3 4 5

214 373 182 255 13

>

>

>

>

> ## Use LDA function to classify dataset. The output variable is 'qtr' and the input variables are

> ##'togo', 'kicker', and 'ydline'.

> datasetNFL.lda <- lda(qtr~togo+kicker+ydline, data=datasetNFL)

> datasetNFL.lda

Call:

lda(qtr ~ togo + kicker + ydline, data = datasetNFL)

Prior probabilities of groups:

1 2 3 4 5

0.20636451 0.35969142 0.17550627 0.24590164 0.01253616

Group means:

togo kicker ydline

1 6.481308 19.64486 17.22897

2 6.973190 18.77212 19.30027

3 6.543956 19.96703 19.03297

4 6.792157 20.20000 18.53725

5 5.923077 22.61538 19.53846

Coefficients of linear discriminants:

LD1 LD2 LD3

togo 0.06665269 0.12498308 0.20996464

kicker -0.04134867 -0.06009657 0.05013225

ydline 0.07726467 -0.07173243 -0.02257770

Proportion of trace:

LD1 LD2 LD3

0.615 0.322 0.063

>

> ## Two LDs are required - LD1 and LD2 - to explain at least 85% of formation of the NFL dataset is explained

> ## LD1 explains 61.5%. LD2 explains a further 32%. Hence LD1 and LD2 will explain 93.7 % together.

>

>

>

## Q. 4(Part 2)

>

> ## Apply PCA, and identify the important principle components involving at least 90% of dataset variation.

> ## Explain your decision strategy?

>

> ## We only use the input variables for the PCA question.This analysis is a type of 'unsepervised' learning.

> datasetNFL2 = cbind(datasetNFL$togo, datasetNFL$kicker, datasetNFL$ydline)

> fit <- princomp(datasetNFL2, cor = TRUE)

> summary(fit)

Importance of components:

Comp.1 Comp.2 Comp.3

Standard deviation 1.146726 0.9998433 0.8278479

Proportion of Variance 0.438327 0.3332289 0.2284441

Cumulative Proportion 0.438327 0.7715559 1.0000000

>

> ## Looking at the 'Cumulative Proportion' output line we can see that Comp1 captures 43.8% of dataset variation.

> ## Comp 1 and Comp2 togther capture 77.2% (approx) of dataset variation.

> ## However, ll three components (Comp1, Comp2, Comp3) are important to capture 90% of the dataset variation.

>

>

>

> ## Plot principle components versus their variance

> ## (Hint: to sketch the plot use the Scree plot).

> loadings(fit)

Loadings:

Comp.1 Comp.2 Comp.3

[1,] 0.707 0.707

[2,] 0.999

[3,] 0.706 -0.707

Comp.1 Comp.2 Comp.3

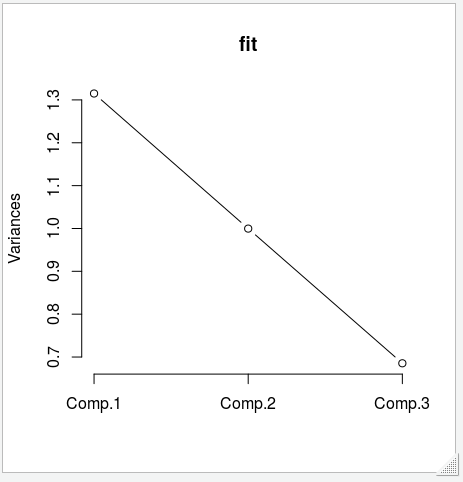
SS loadings 1.000 1.000 1.000

Proportion Var 0.333 0.333 0.333

Cumulative Var 0.333 0.667 1.000

> plot(fit, type = "lines")

>



> # The Plot confirms that all three components are important. There is no 'bend' in the line indicating that higher

> # components contribute less to the capture of dataset variation

>

>

>

>

## Q. 4(Part 3)

>

> ## Split the dataset into two sets of variables so that X=(togo, kicker, ydline) and Y=(distance, homekick).

> ## Apply canonical correlation analysis to find the cross-correlation between X and Y.

>

>

> ## Set up 'X' variable

> X <- cbind(datasetNFL$togo, datasetNFL$kicker, datasetNFL$ydline)

>

> ## Set up 'Y' variable

> Y <- cbind(datasetNFL$distance, datasetNFL$homekick)

> cor(X, Y)

[,1] [,2]

[1,] 0.315641454 -0.04838438

[2,] -0.001951722 -0.02363159

[3,] 0.998947222 0.04295427

>

>

> ## What is the correlation between 'ydline' and 'distance'?

>

> ## Read three down the X value and one across the Y value

> ## The correlation between 'ydline' and 'distance' is equal to '0.998947222'

> ## This value shows a high level of correlation between the 'ydline' and 'distance' values

>

>

>

>

>

>

>

>

## Q. 4(Part 4)

>

> ## Use K-means clustering analysis to identify the most important classes.

> ## How many classes do you select? Why?

>

> ## Again consider the input variables. We use the 'datasetNFL2' dataset because I want to

> ## just consider the 'togo', 'kicker', and 'ydline' input variables.

>

> ## K-Mean

> ##k.means.fit <- kmeans(datasetNFL2, 4)

> ##attributes(k.means.fit)

>

> # Centroids(arithmetic mean)

> ##k.means.fit$centers

>

> # Cluster size

> ##k.means.fit$size

>

>

> # Generate the plot K-Means clustering

> ## Write function for plot generation

> wssplot <- function(datasetNFL2, nc=10, seed=2343){

+

+ wss <- (nrow(datasetNFL2)-1) \* sum(apply(datasetNFL2, 2, var))

+

+ for (i in 2 : nc){

+

+ set.seed(seed)

+ wss[i] <- sum(kmeans(datasetNFL2, centers = i)$withinss)

+

+ }

+

+ plot(1:nc, wss, type = "b", xlab = "Numbers of Clusters", ylab = "Within Groups Sum of Squares")

+

+

+ }

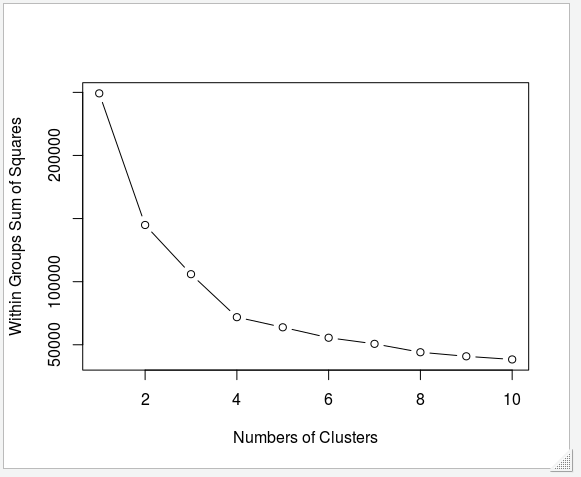
>

>

# Invoke plot function

> wssplot(datasetNFL2, nc = 10)

>



> ## In the Cluster graph we can see a definite 'elbox' at Number of Clusters = 4.

> ## After Cluster 4 the changes in variation are noticeably less

> ## Therefore the main cluster are clusters 1 through to cluster 4.

> ## We would select four classes as an answer to this question.