Predictive Modelling for Data Science

DS432

Kotalwar Sangamesh

U101115FCS210

Course in-charge: Prof. Suman Sanyal

Contents

1 Acknowledgement

I'm highly indebted to Prof. Suman Sanyal for his guidance and constant supervision as well as for providing necessary information regarding the assignments.

I acknowledge that any work that I submit for assessment at NIIT University:

1. Must be all my own work.
2. Must not have been prepared with the assistance of any other person, except those permitted within University guidelines or the specific assessment guidelines for the piece of work.
3. Has not previously been submitted for assessment at this University or elsewhere.

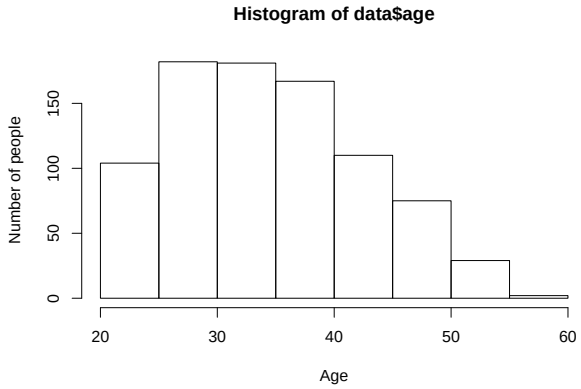
2 Assignment A

2.1 Q.1

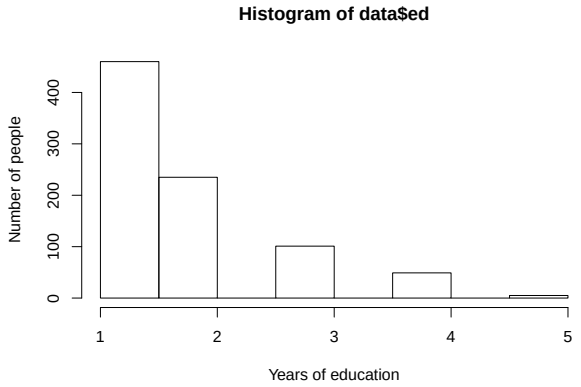
Step 1) Load the data in

```
library(data.table)
#data <-  as.data.frame(fread("BankBayesLoan.txt"))
data <- read.csv("BankBayesLoan.txt", stringsAsFactors = FALSE,header=TRUE, sep="\t")
```

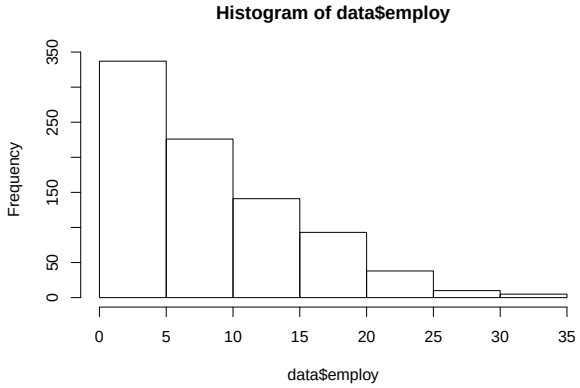
```
hist(data$age,xlab="Age",ylab="Number of people")
```



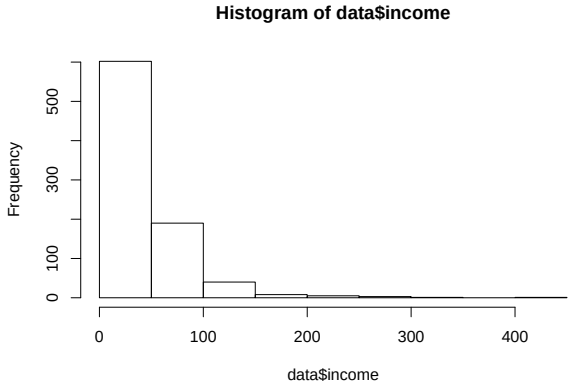

```
hist(data$ed,xlab="Years of education",ylab="Number of people")
```



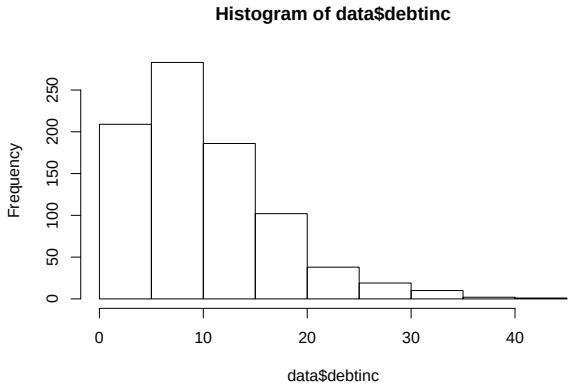

```
hist(data$employ)
```



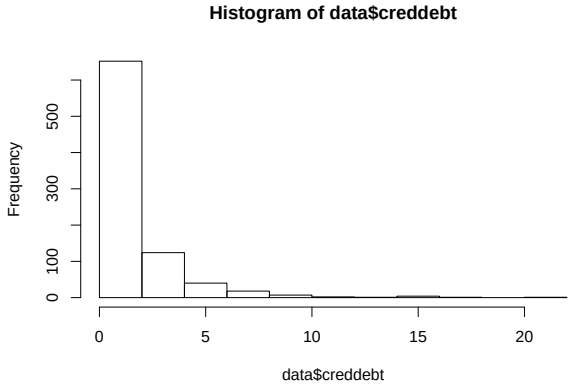

```
hist(data$income)
```



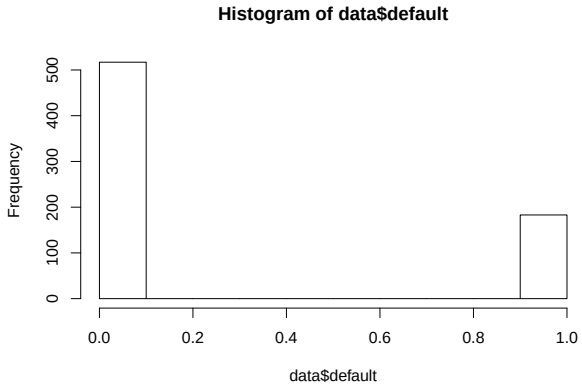

```
hist(data$debtinc)
```




```
hist(data$creddebt)
```




```
#hist(data$otherdebt)
hist(data$default)
```




```
testdata <- subset(data,default==0 | default==1)
#show(testdata)
```

Step 3: Make a correlation matrix and plot it

```
M<-cor(testdata,use="complete.obs")
show(M)
```

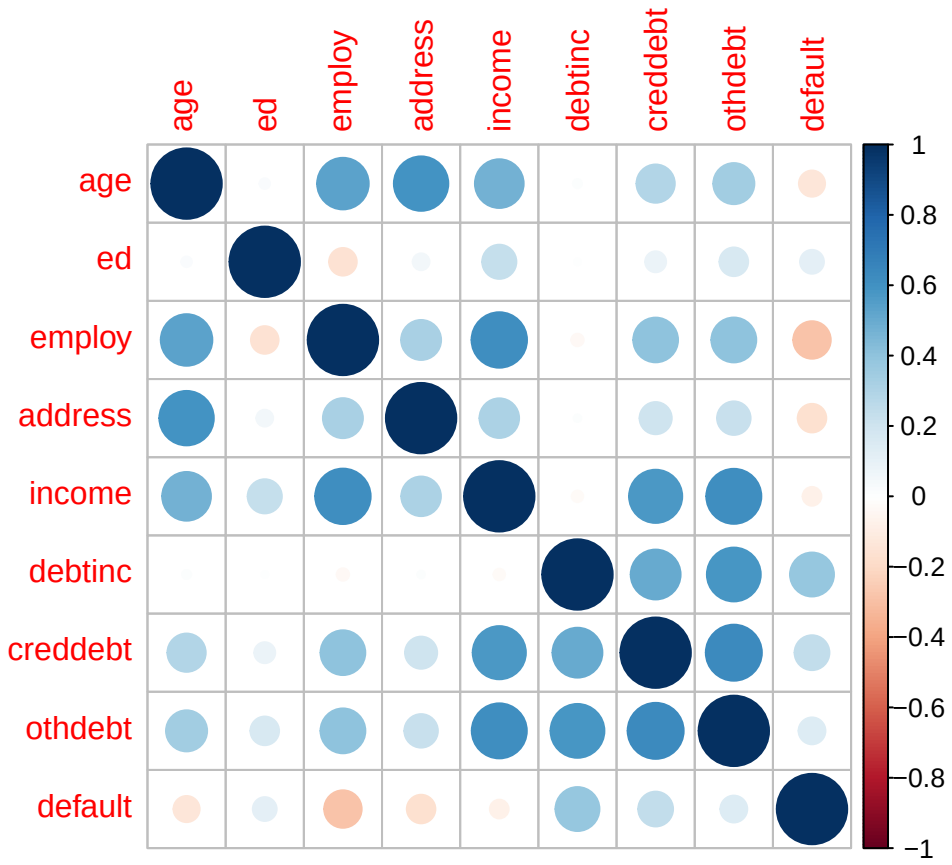
```
##                 age          ed      employ     address      income
## age      1.00000000  0.02232500  0.53649678  0.59759074  0.47870987
## ed       0.02232500  1.00000000 -0.15362077  0.05691913  0.23519050
## employ   0.53649678 -0.15362077  1.00000000  0.32233430  0.61968132
## address  0.59759074  0.05691913  0.32233430  1.00000000  0.31624514
## income   0.47870987  0.23519050  0.61968132  0.31624514  1.00000000
## debtinc  0.01639808  0.00883843 -0.03118221  0.01132298 -0.02677729
## creddebt 0.29520667  0.08827406  0.40369370  0.20843505  0.57019866
## othdebt  0.34021695  0.16545872  0.40609121  0.22651449  0.61065941
## default -0.13765710  0.11467555 -0.28297839 -0.16445116 -0.07096966
##              debtinc   creddebt   othdebt     default
## age       0.01639808 0.29520667 0.3402169 -0.13765710
## ed        0.00883843 0.08827406 0.1654587  0.11467555
## employ   -0.03118221 0.40369370 0.4060912 -0.28297839
## address   0.01132298 0.20843505 0.2265145 -0.16445116
## income   -0.02677729 0.57019866 0.6106594 -0.07096966
```

```
## debtinc   1.000000000 0.50176719 0.5848696  0.38957476
## creddebt  0.501767186 1.00000000 0.6331036  0.24473973
## othdebt   0.584869569 0.63310361 1.0000000  0.14571257
## default   0.389574760 0.24473973 0.1457126  1.00000000
```

```
library('corrplot')
```

```
## corrplot 0.84 loaded
```

```
corrplot(M, method = "circle")
```



Step 4: PCA on data

```
library(ggplot2)
pca <- prcomp(data[c(1:8)] , center = TRUE , scale.=TRUE )
summary(pca)
```

```
## Importance of components:
##                           PC1    PC2    PC3    PC4     PC5     PC6     PC7
## Standard deviation     1.8112 1.2772 1.0430 0.9241 0.62729 0.58127 0.54097
## Proportion of Variance 0.4101 0.2039 0.1360 0.1067 0.04919 0.04223 0.03658
## Cumulative Proportion  0.4101 0.6140 0.7499 0.8567 0.90588 0.94811 0.98469
##                           PC8
## Standard deviation     0.34992
## Proportion of Variance 0.01531
## Cumulative Proportion  1.00000
```

The eigen values are nothing but normalized sd squared

```
eigen_data <- pca$sd^2
```

Step 5: Find the scree plot to know how many variables to choose

```
#lines(1:8,eigen_data)
```

Step 6 : Infer from loadings

```
otherpca <- princomp(data[c(1:8)],cor = TRUE)
summary(otherpca)
```

```
## Importance of components:
##                            Comp.1    Comp.2    Comp.3    Comp.4     Comp.5
## Standard deviation     1.8112019 1.2772065 1.0429798 0.9241319 0.62729100
## Proportion of Variance 0.4100566 0.2039071 0.1359759 0.1067525 0.04918675
## Cumulative Proportion  0.4100566 0.6139636 0.7499395 0.8566919 0.90587869
##                            Comp.6     Comp.7     Comp.8
## Standard deviation     0.58127359 0.54096678 0.34992352
## Proportion of Variance 0.04223487 0.03658063 0.01530581
## Cumulative Proportion  0.94811356 0.98469419 1.00000000
```

```
otherpca$loadings
```

```
##
## Loadings:
##          Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8
## age      -0.384 -0.361        -0.315  0.469  0.514 -0.368
## ed                0.123  0.914         0.164  0.129  0.301
## employ   -0.402 -0.280 -0.242  0.351  0.211         0.729
## address  -0.287 -0.389        -0.644 -0.438 -0.367  0.156
## income   -0.450 -0.115  0.224  0.429 -0.157 -0.211 -0.339  0.605
## debtinc  -0.189  0.620 -0.224 -0.401  0.203         0.222  0.528
## creddebt -0.414  0.339               -0.603  0.513        -0.277
## othdebt  -0.441  0.336                0.296 -0.525 -0.231 -0.524
##
##                Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8
## SS loadings     1.000  1.000  1.000  1.000  1.000  1.000  1.000  1.000
## Proportion Var  0.125  0.125  0.125  0.125  0.125  0.125  0.125  0.125
## Cumulative Var  0.125  0.250  0.375  0.500  0.625  0.750  0.875  1.000
```

```
library(factoextra)
```

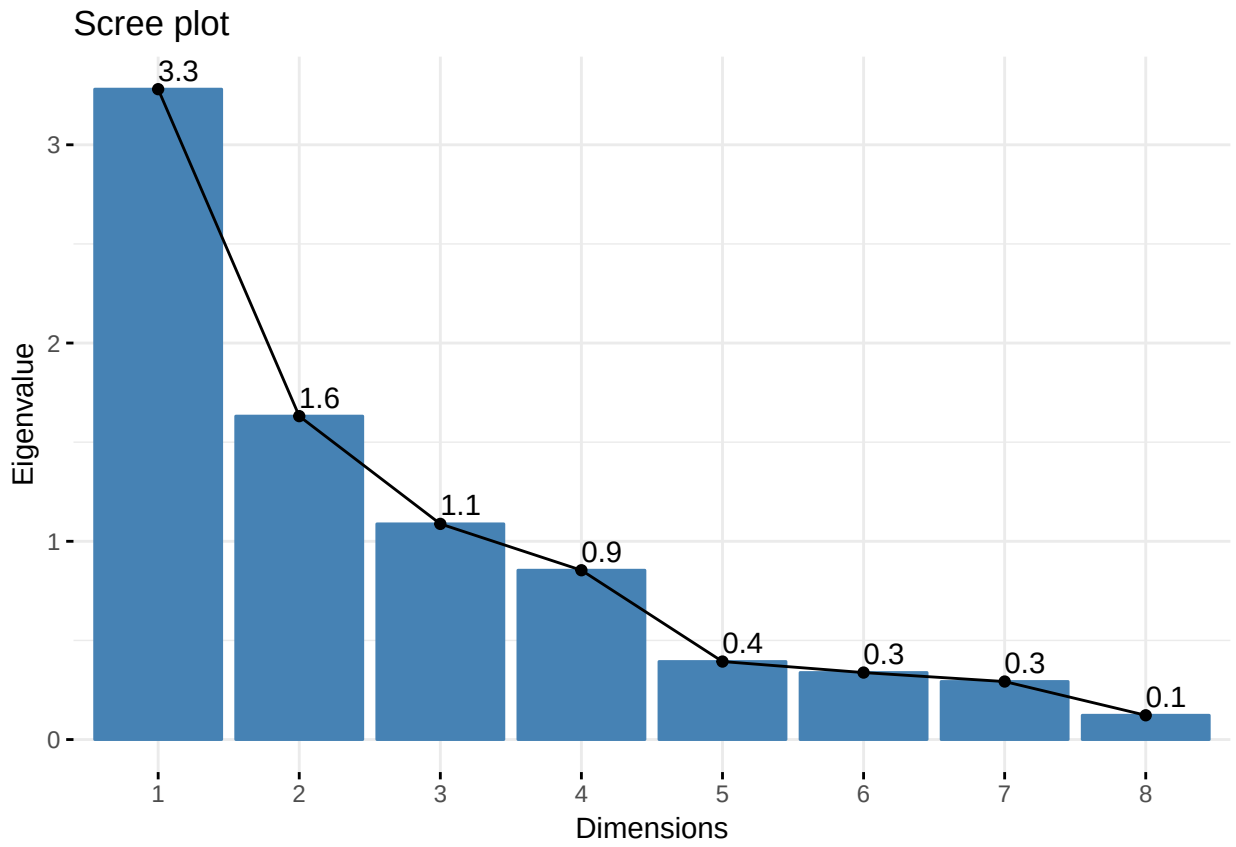
```
## Welcome! Related Books: `Practical Guide To Cluster Analysis in R` at https://goo.gl/13EFCZ
```

```
eigen_val <- get_eigenvalue(otherpca)
show(eigen_val)
```

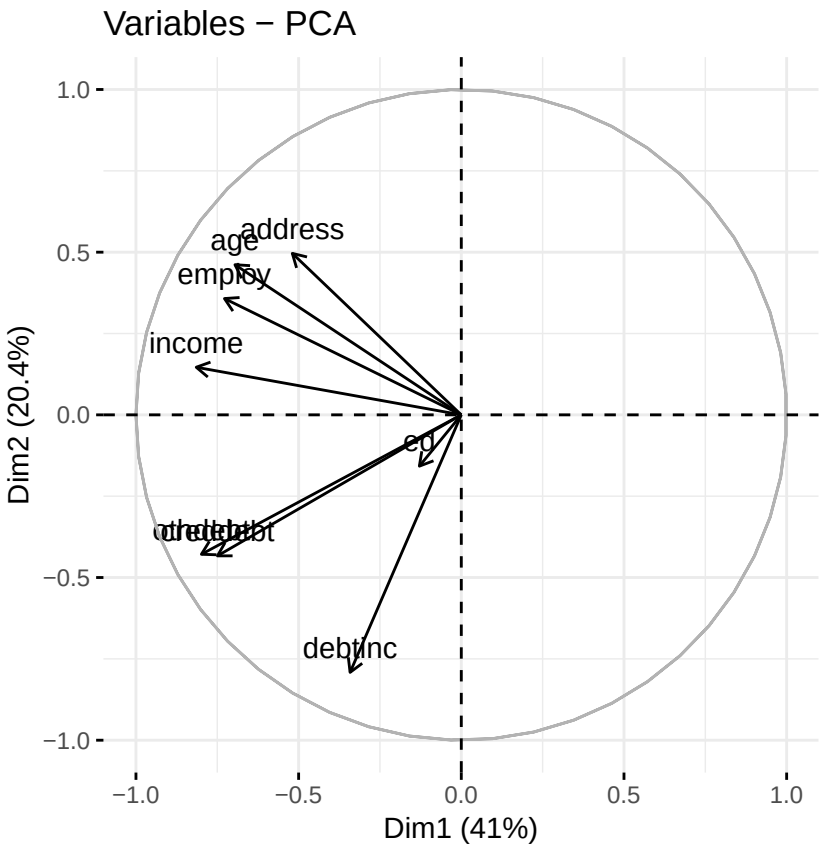
```
##       eigenvalue variance.percent cumulative.variance.percent
## Dim.1  3.2804524        41.005655                    41.00566
## Dim.2  1.6312564        20.390705                    61.39636
## Dim.3  1.0878070        13.597587                    74.99395
## Dim.4  0.8540197        10.675246                    85.66919
## Dim.5  0.3934940         4.918675                    90.58787
## Dim.6  0.3378790         4.223487                    94.81136
## Dim.7  0.2926451         3.658063                    98.46942
## Dim.8  0.1224465         1.530581                   100.00000
```

SCREE PLOT

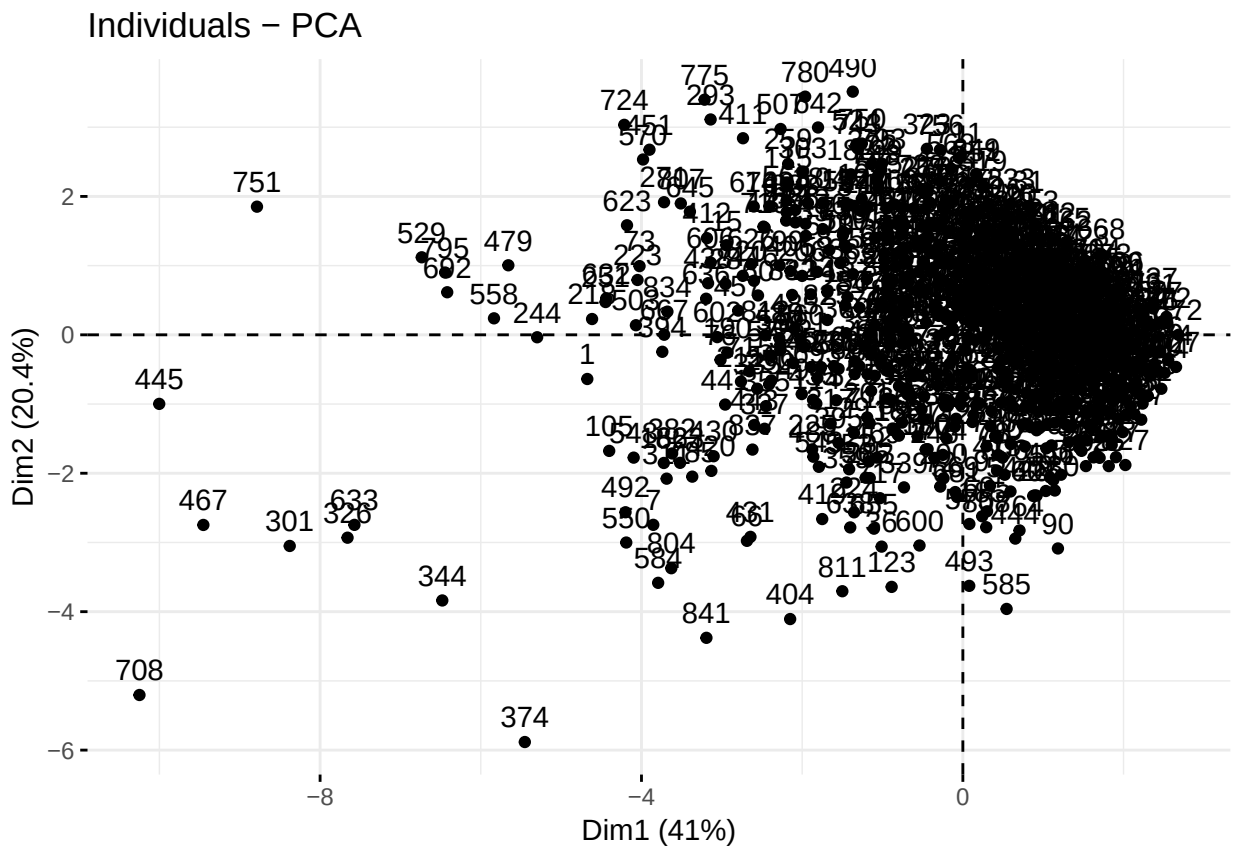
```
fviz_eig(pca, choice = "eigenvalue", addlabels=TRUE)
```



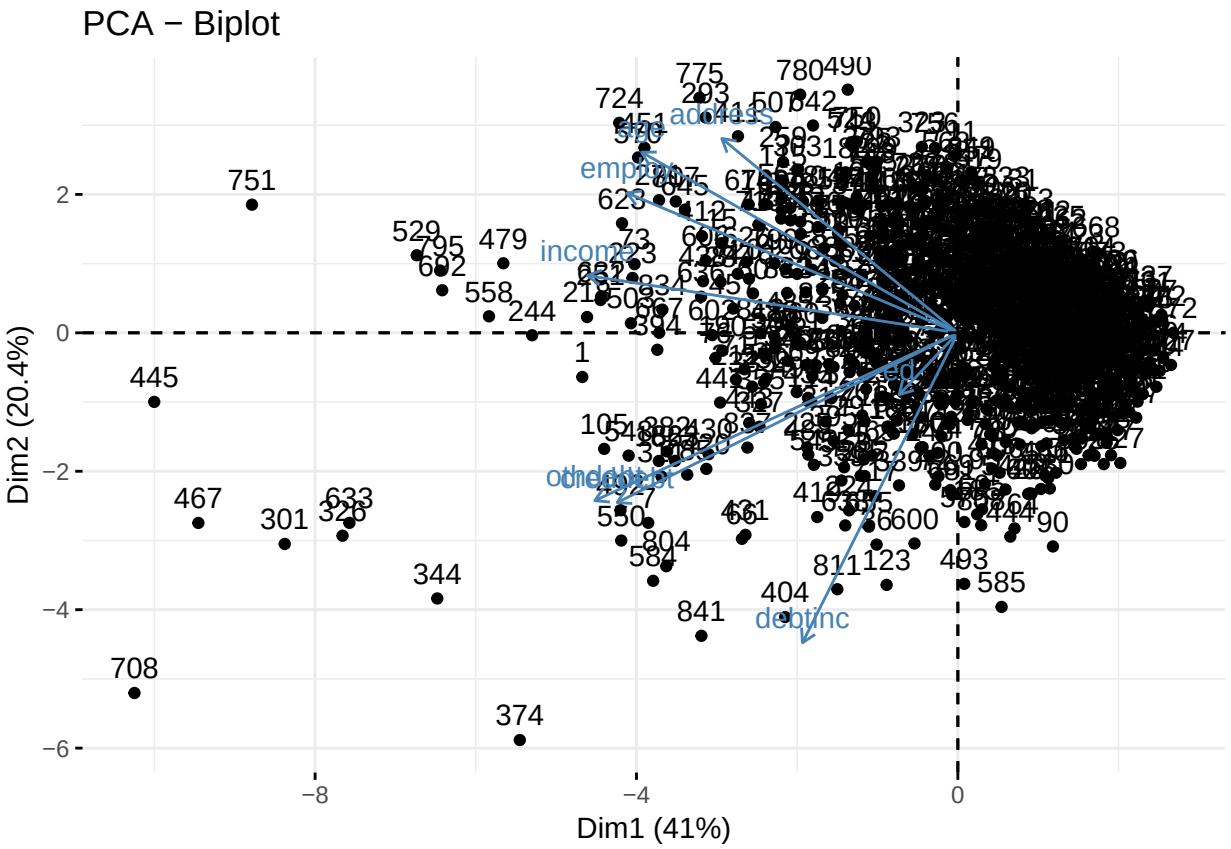

```
library(factoextra)
fviz_pca_var(pca)
```




```
fviz_pca_ind(pca)
```




```
fviz_pca_biplot(pca)
```



2.2 Q. 2

CRISP - DM for sales in the company

2.2.1 Reading the Dataset

```
##         date      men    women    jewel mail page phone    print service
## 1 1989-01-01 11357.92 16578.93 10776.38 7978   73    34 22294.48      20
## 2 1989-02-01 10605.95 18236.13 10821.97 8290   88    29 27426.47      20
## 3 1989-03-01 16998.57 43393.55 22845.79 8029   65    24 27978.66      26
## 4 1989-04-01  6563.75 30908.49 11102.62 7752   85    20 28949.65      22
## 5 1989-05-01  6607.69 28701.58 16066.57 8685   74    17 22642.27      21
## 6 1989-06-01  9839.00 29647.57 11061.28 7847   87    30 27210.61      23
##   YEAR_ MONTH_    DATE_ Seasonal_Err_Men Seasonal_AdjSer_Men
## 1  1989      1 JAN 1989         0.919551           11932.955
## 2  1989      2 FEB 1989         0.846981           12550.903
## 3  1989      3 MAR 1989         1.496159           19971.321
## 4  1989      4 APR 1989         0.678450            7727.042
## 5  1989      5 MAY 1989         0.753748            7732.708
## 6  1989      6 JUN 1989         1.134113           11363.275
##   Seasonal_Factors_Men Seasonal_TrendCycle_Men
## 1             0.951811                12976.93
## 2             0.845035                14818.39
## 3             0.851149                13348.39
```

```
## 4             0.849452               11389.26
## 5             0.854512               10259.01
## 6             0.865860               10019.53
```

2.2.2 Data Pre-Processing

Step 1: Checking for any Missing Data and Statistical Summary

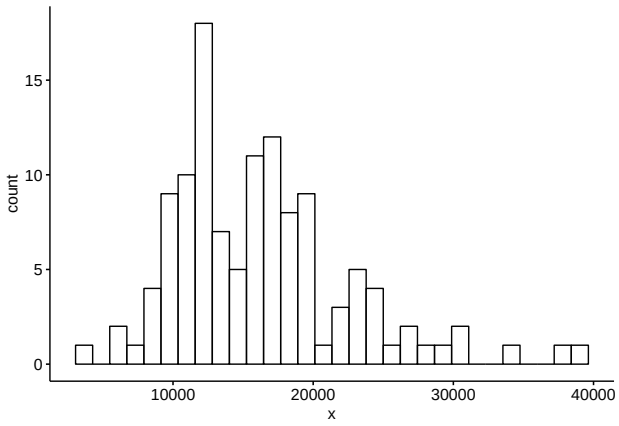
```
## data
##
##  16  Variables      120  Observations
## ---------------------------------------------------------------------------
## date
##        n  missing distinct
##      120        0      120
##
## lowest : 1989-01-01 1989-02-01 1989-03-01 1989-04-01 1989-05-01
## highest: 1998-08-01 1998-09-01 1998-10-01 1998-11-01 1998-12-01
## ---------------------------------------------------------------------------
## men
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      120        0      120        1    16243     6819     8395    10039
##      .25      .50      .75      .90      .95
##    11750    15452    19094    24305    27875
##
## lowest :  3245.18  6563.75  6607.69  6766.94  8214.14
## highest: 30208.17 31090.05 34028.01 38236.59 38609.66
## ---------------------------------------------------------------------------
## women
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      120        0      120        1    40584    13526    21714    24420
##      .25      .50      .75      .90      .95
##    33010    39779    46877    54789    61311
##
## lowest : 16578.93 18103.06 18236.13 20078.40 20789.67
## highest: 65196.81 69112.81 73918.34 75818.17 80245.97
## ---------------------------------------------------------------------------
## jewel
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      120        0      120        1    16741     7036     9925    10988
##      .25      .50      .75      .90      .95
##    12077    14241    20311    26049    32893
##
## lowest :  5983.55  8641.83  8713.27  9595.20  9597.03
## highest: 34065.23 34645.89 34929.23 35893.86 38231.57
## ---------------------------------------------------------------------------
## mail
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      120        0      117        1    10132     1784     8026     8432
##      .25      .50      .75      .90      .95
##     9099    10074    11182    11731    12769
##
## lowest :  1147  7752  7811  7847  7881, highest: 13194 13666 14370 14508 15263
## ---------------------------------------------------------------------------
```

```
## page
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      120        0       48    0.999    80.58    14.65    57.95    64.90
##      .25      .50      .75      .90      .95
##    72.00    80.50    88.25    95.10   104.05
##
## lowest :  51  52  54  57  58, highest: 105 106 110 111 114
## ------------------------------------------------------------------------
## phone
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      120        0       36    0.998    34.83    9.628    22.95    24.90
##      .25      .50      .75      .90      .95
##    28.00    34.00    40.25    47.00    49.05
##
## lowest : 17 20 21 22 23, highest: 50 52 53 54 59
## ------------------------------------------------------------------------
## print
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      120        0      120        1    28519     3827    22722    24443
##      .25      .50      .75      .90      .95
##    26750    28513    30532    32486    34387
##
## lowest : 18061.20 20911.52 21447.37 22294.48 22642.27
## highest: 34843.65 35957.27 36666.60 38738.65 40027.78
## ------------------------------------------------------------------------
## service
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      120        0       43    0.999    35.97    12.48     20.0     22.0
##      .25      .50      .75      .90      .95
##     28.0     36.0     43.0     50.1     54.0
##
## lowest : 15 16 18 19 20, highest: 55 56 58 63 68
## ------------------------------------------------------------------------
## YEAR_
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      120        0       10     0.99     1994    3.328     1989     1990
##      .25      .50      .75      .90      .95
##     1991     1994     1996     1997     1998
##
## Value      1989 1990 1991 1992 1993 1994 1995 1996 1997 1998
## Frequency    12   12   12   12   12   12   12   12   12   12
## Proportion  0.1  0.1  0.1  0.1  0.1  0.1  0.1  0.1  0.1  0.1
## ------------------------------------------------------------------------
## MONTH_
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      120        0       12    0.993      6.5    4.006     1.00     2.00
##      .25      .50      .75      .90      .95
##     3.75     6.50     9.25    11.00    12.00
##
## Value          1     2     3     4     5     6     7     8     9    10
## Frequency     10    10    10    10    10    10    10    10    10    10
## Proportion 0.083 0.083 0.083 0.083 0.083 0.083 0.083 0.083 0.083 0.083
##
## Value         11    12
```

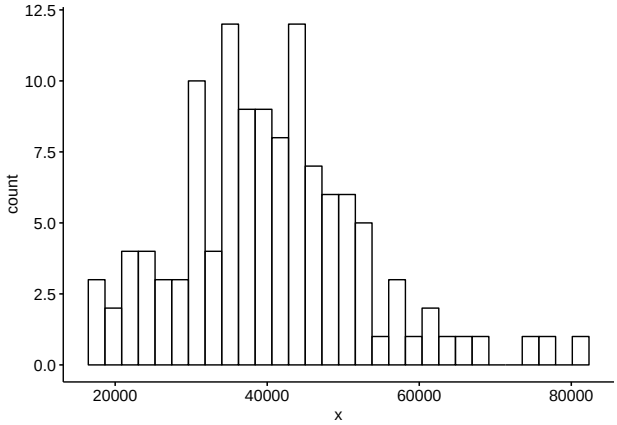
```
## Frequency     10    10
## Proportion 0.083 0.083
## ---------------------------------------------------------------------------
## DATE_
##        n  missing distinct
##      120        0      120
##
## lowest : APR 1989 APR 1990 APR 1991 APR 1992 APR 1993
## highest: SEP 1994 SEP 1995 SEP 1996 SEP 1997 SEP 1998
## ---------------------------------------------------------------------------
## Seasonal_Err_Men
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      120        0      120        1   0.9949   0.1651   0.8049   0.8296
##      .25      .50      .75      .90      .95
##   0.9188   0.9892   1.0698   1.1388   1.2256
##
## lowest : 0.235599 0.621646 0.678450 0.703119 0.738985
## highest: 1.365981 1.396972 1.417609 1.441676 1.496159
## ---------------------------------------------------------------------------
## Seasonal_AdjSer_Men
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      120        0      120        1    16235     5390     9784    10912
##      .25      .50      .75      .90      .95
##    12741    15291    20005    21956    23384
##
## lowest :  3486.580  7727.042  7732.708  7815.286  9008.905
## highest: 25181.443 26288.041 26561.129 28934.535 31765.709
## ---------------------------------------------------------------------------
## Seasonal_Factors_Men
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      120        0       12    0.993        1     0.22   0.8450   0.8495
##      .25      .50      .75      .90      .95
##   0.8537   0.8983   0.9911   1.1409   1.7946
##
## Value      0.845035 0.849452 0.851149 0.854512 0.855873 0.865860 0.930763
## Frequency        10       10       10       10       10       10       10
## Proportion    0.083    0.083    0.083    0.083    0.083    0.083    0.083
##
## Value      0.950968 0.951811 1.109078 1.140925 1.794575
## Frequency        10       10       10       10       10
## Proportion    0.083    0.083    0.083    0.083    0.083
## ---------------------------------------------------------------------------
## Seasonal_TrendCycle_Men
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      120        0      120        1    16233     4235    11397    11945
##      .25      .50      .75      .90      .95
##    13101    15235    19417    21134    21619
##
## lowest : 10019.53 10259.01 10866.22 10895.33 11378.31
## highest: 22714.96 23524.17 24702.62 25270.89 26004.76
## ---------------------------------------------------------------------------
```

Results: 1: No missing values were found. 2: The highest and lowest values of each parameter in the dataset indicates that those five are almost closer to each other i.e. no outliers.

Histogram Plot for the sales of men's clothing



Histogram Plot for the sales of women's clothing



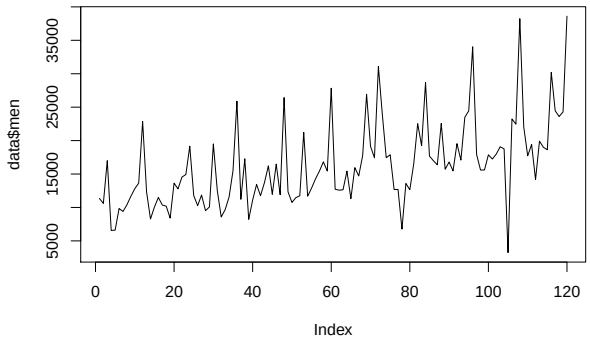
=> Testing for normality using Shapiro-Wilk Test

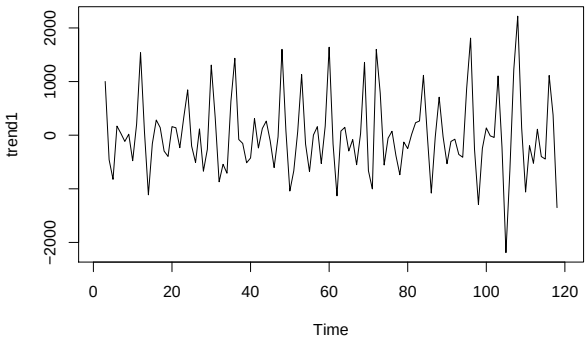
```
##
##  Shapiro-Wilk normality test
##
## data:  data$men
## W = 0.92988, p-value = 9.464e-06
```

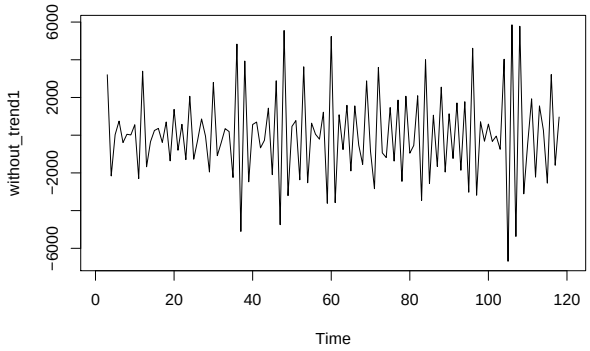
```
##
##  Shapiro-Wilk normality test
##
## data:  data$women
## W = 0.97396, p-value = 0.01978
```

Result: 1: The p-value for both the sales data is not normally distributed. 2: Data Normalization is required.

ii -> Calculating the trend and removing it for men's sales







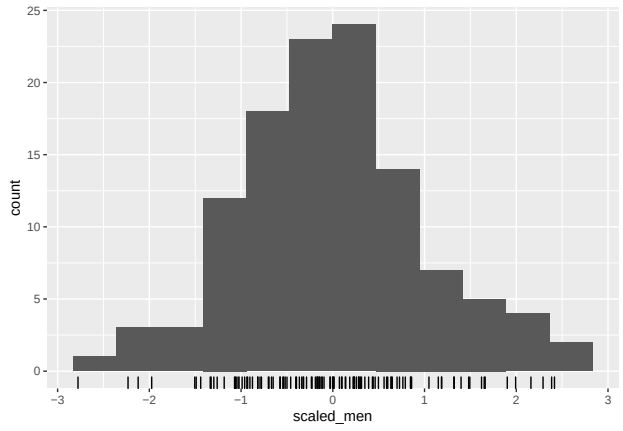
ii -> Normalizing the data using skewness reduction techniques

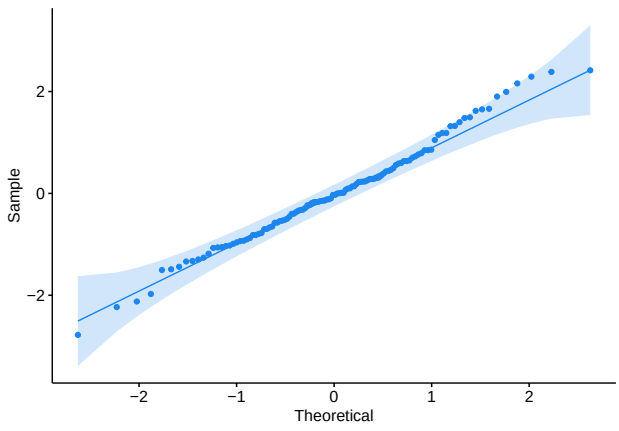
```
## Skewness of the original data of the women's sales is: 0.5876654
```

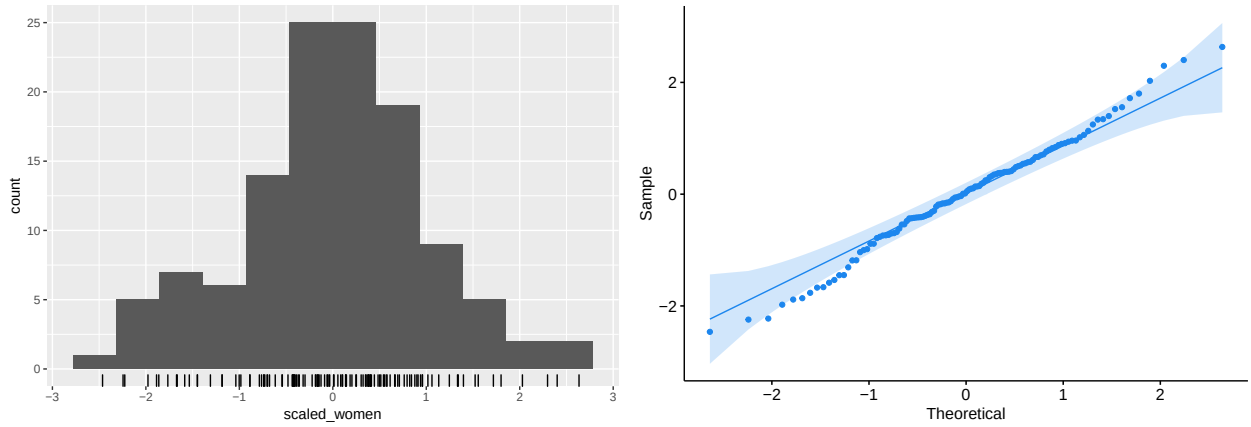
```
## SKewness of the normalized data for the women's sales is: -0.05918168
```

The skewness of the women's sales data shows a positive skewness of 0.58 and thus is right skewed. Right skewed data can be reduced by cube root method and as a result of which the skewness was reduced by 10 times.

iii -> Z-Score Standardisation

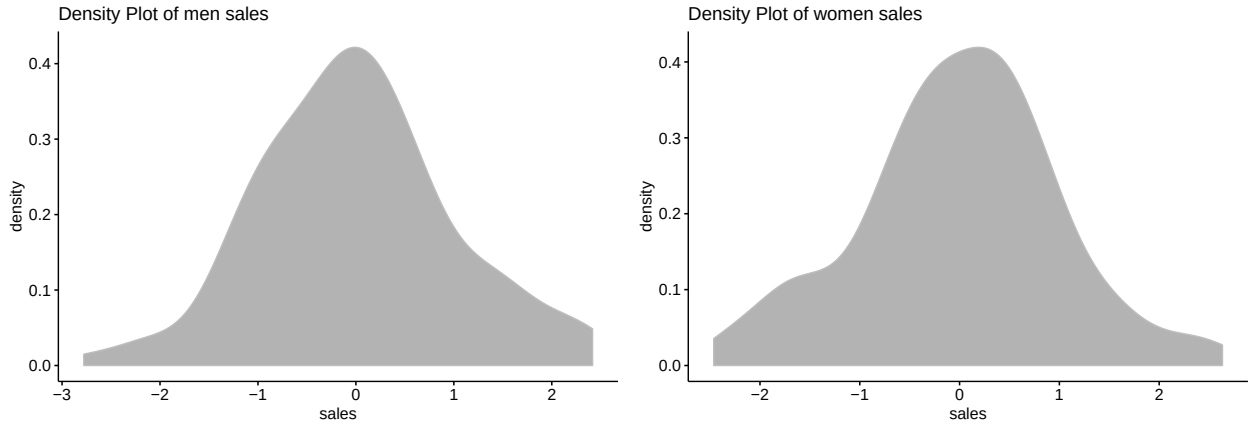






Result: The above plot shows that the points now lie closer to the line, after removing the seasonality.

2.2.3 EDA



=> Testing for normality

```
##
##  Shapiro-Wilk normality test
##
## data:  scaled_men
## W = 0.98955, p-value = 0.5204
```

```
##
##  Shapiro-Wilk normality test
##
## data:  scaled_women
## W = 0.99092, p-value = 0.619
```

Result: The data has a considerably good p-value and thus cleares the test for normality

```
## Correlation coeffficient between the men's and women's sales is 0.4793316
```

```
## [1] "Thus the sales of both the genders is closely related."
```

2.3 Q.3

2.3.1 PHASE 1: Reading the Dataset

2.3.2 PHASE 2 => Data Pre-Processing

Step 1: Checking for any Missing Data and Statistical Summary

```
## data
##
##  42  Variables      1000  Observations
## --------------------------------------------------------------------------
## region
##        n  missing distinct     Info     Mean      Gmd
##     1000        0        3    0.889    2.022   0.8888
##
## Value          1     2     3
## Frequency    322   334   344
## Proportion 0.322 0.334 0.344
## --------------------------------------------------------------------------
## tenure
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0       72        1    35.53    24.65        4        7
##      .25      .50      .75      .90      .95
##       17       34       54       66       70
##
## lowest :  1  2  3  4  5, highest: 68 69 70 71 72
## --------------------------------------------------------------------------
## age
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0       60    0.999    41.68    14.33       23       26
##      .25      .50      .75      .90      .95
##       32       40       51       59       64
##
## lowest : 18 19 20 21 22, highest: 73 74 75 76 77
## --------------------------------------------------------------------------
## marital
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2     0.75      495    0.495   0.5005
##
## --------------------------------------------------------------------------
## address
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0       50    0.998    11.55    10.92      0.0      1.0
##      .25      .50      .75      .90      .95
##      3.0      9.0     18.0     26.1     31.0
##
## lowest :  0  1  2  3  4, highest: 45 46 48 49 55
## --------------------------------------------------------------------------
## income
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0      218        1    77.53     75.3     18.0     21.0
##      .25      .50      .75      .90      .95
##     29.0     47.0     83.0    155.4    232.2
```

```
##
## lowest :    9   10   11   12   13, highest:  732  928  944 1131 1668
## ---------------------------------------------------------------------------
## ed
##        n  missing distinct     Info     Mean      Gmd
##     1000        0        5    0.946    2.671    1.369
##
## Value          1     2     3     4     5
## Frequency    204   287   209   234    66
## Proportion 0.204 0.287 0.209 0.234 0.066
## ---------------------------------------------------------------------------
## employ
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0       46    0.997    10.99    10.93        0        0
##      .25      .50      .75      .90      .95
##        3        8       17       25       31
##
## lowest :  0  1  2  3  4, highest: 41 43 44 45 47
## ---------------------------------------------------------------------------
## retire
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2    0.134       47    0.047  0.08967
##
## ---------------------------------------------------------------------------
## gender
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2    0.749      517    0.517   0.4999
##
## ---------------------------------------------------------------------------
## reside
##        n  missing distinct     Info     Mean      Gmd
##     1000        0        8    0.923    2.331     1.52
##
## Value          1     2     3     4     5     6     7     8
## Frequency    375   272   138   120    60    29     4     2
## Proportion 0.375 0.272 0.138 0.120 0.060 0.029 0.004 0.002
## ---------------------------------------------------------------------------
## tollfree
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2    0.748      474    0.474   0.4991
##
## ---------------------------------------------------------------------------
## equip
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2    0.711      386    0.386   0.4745
##
## ---------------------------------------------------------------------------
## callcard
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2    0.655      678    0.678   0.4371
##
## ---------------------------------------------------------------------------
## wireless
##        n  missing distinct     Info      Sum     Mean      Gmd
```

```
##     1000        0        2    0.625      296    0.296   0.4172
##
## ------------------------------------------------------------------------
## longmon
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0      425        1    11.72    9.559    2.998    3.645
##      .25      .50      .75      .90      .95
##    5.200    8.525   14.413   23.960   31.615
##
## lowest :  0.90  1.05  1.10  1.35  1.45, highest: 62.30 75.45 81.55 89.40 99.95
## ------------------------------------------------------------------------
## tollmon
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0      162    0.855    13.27    16.92     0.00     0.00
##      .25      .50      .75      .90      .95
##     0.00     0.00    24.25    34.75    41.77
##
## lowest :   0.00   5.75   9.00   9.25   9.50, highest:  77.75  78.75  84.00  87.00 173.00
## ------------------------------------------------------------------------
## equipmon
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0      310    0.769    14.22     19.2     0.00     0.00
##      .25      .50      .75      .90      .95
##     0.00     0.00    31.47    42.66    48.65
##
## lowest :  0.00 15.40 19.55 19.75 19.80, highest: 64.20 69.40 70.05 73.80 77.70
## ------------------------------------------------------------------------
## cardmon
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0      176    0.967    13.78    14.67     0.00     0.00
##      .25      .50      .75      .90      .95
##     0.00    12.00    20.50    31.50    37.79
##
## lowest :   0.00   2.75   3.75   4.50   4.75, highest:  74.00  82.25  84.75  87.50 109.25
## ------------------------------------------------------------------------
## wiremon
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0      263    0.651    11.58    17.76     0.00     0.00
##      .25      .50      .75      .90      .95
##     0.00     0.00    24.71    42.11    51.35
##
## lowest :   0.00  14.90  15.10  16.85  17.45, highest:  83.70  85.85  96.25 109.70 111.95
## ------------------------------------------------------------------------
## longten
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0      960        1    574.1      699    12.30    29.18
##      .25      .50      .75      .90      .95
##    90.14   285.48   755.02  1537.09  2144.79
##
## lowest :    0.90    1.05    1.10    1.35    1.45
## highest: 4333.00 5464.60 5988.50 6353.90 7257.60
## ------------------------------------------------------------------------
## tollten
##        n  missing distinct     Info     Mean      Gmd      .05      .10
```

```
##     1000        0      473    0.854    551.3    823.1      0.0      0.0
##      .25      .50      .75      .90      .95
##      0.0      0.0    846.9   1838.4   2424.3
##
## lowest :    0.00    5.75   20.75   23.05   26.50
## highest: 4748.45 4905.85 4938.60 5850.25 5916.00
## ---------------------------------------------------------------------------
## equipten
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0      386    0.769    465.6    736.6      0.0      0.0
##      .25      .50      .75      .90      .95
##      0.0      0.0    579.5   1854.9   2435.6
##
## lowest :    0.00   15.40   17.25   22.60   22.80
## highest: 3925.50 4167.70 4758.05 4980.80 5028.65
## ---------------------------------------------------------------------------
## cardten
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0      338    0.967    605.8    775.2      0.0      0.0
##      .25      .50      .75      .90      .95
##      0.0    332.5    910.0   1505.5   2181.0
##
## lowest :    0.00    2.75    5.00   10.00   15.00
## highest: 4975.00 5115.00 5455.00 5520.00 7515.00
## ---------------------------------------------------------------------------
## wireten
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0      297    0.651    442.7    743.3      0.0      0.0
##      .25      .50      .75      .90      .95
##      0.0      0.0    316.5   1833.7   2727.6
##
## lowest :    0.00   10.45   16.85   20.95   24.20
## highest: 4399.60 4828.20 6132.20 6444.95 7856.85
## ---------------------------------------------------------------------------
## multline
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2    0.748      475    0.475   0.4992
##
## ---------------------------------------------------------------------------
## voice
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2    0.635      304    0.304   0.4236
##
## ---------------------------------------------------------------------------
## pager
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2    0.579      261    0.261   0.3861
##
## ---------------------------------------------------------------------------
## internet
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2    0.698      368    0.368   0.4656
##
## ---------------------------------------------------------------------------
```

```
## callid
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2    0.749      481    0.481   0.4998
##
## ------------------------------------------------------------------------
## callwait
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2    0.749      485    0.485   0.5001
##
## ------------------------------------------------------------------------
## forward
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2     0.75      493    0.493   0.5004
##
## ------------------------------------------------------------------------
## confer
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2     0.75      502    0.502   0.5005
##
## ------------------------------------------------------------------------
## ebill
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2      0.7      371    0.371   0.4672
##
## ------------------------------------------------------------------------
## loglong
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0      425        1    2.182    0.829    1.098    1.293
##      .25      .50      .75      .90      .95
##    1.649    2.143    2.668    3.176    3.454
##
## lowest : -0.105361  0.048790  0.095310  0.300105  0.371564
## highest:  4.131961  4.323470  4.401216  4.493121  4.604670
## ------------------------------------------------------------------------
## logtoll
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      475      525      161        1     3.24   0.4583    2.621    2.757
##      .25      .50      .75      .90      .95
##    2.970    3.209    3.489    3.751    3.928
##
## lowest : 1.749200 2.197225 2.224624 2.251292 2.277267
## highest: 4.353499 4.366278 4.430817 4.465908 5.153292
## ------------------------------------------------------------------------
## logequi
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      386      614      309        1    3.568   0.3162    3.121    3.201
##      .25      .50      .75      .90      .95
##    3.368    3.572    3.757    3.933    3.994
##
## lowest : 2.734368 2.972975 2.983153 2.985682 2.990720
## highest: 4.162003 4.239887 4.249209 4.301359 4.352855
## ------------------------------------------------------------------------
## logcard
##        n  missing distinct     Info     Mean      Gmd      .05      .10
```

```
##      678      322      175        1    2.854   0.6274    1.946    2.169
##      .25      .50      .75      .90      .95
##    2.464    2.848    3.209    3.577    3.732
##
## lowest : 1.011601 1.321756 1.504077 1.558145 1.609438
## highest: 4.304065 4.409763 4.439706 4.471639 4.693639
## ---------------------------------------------------------------------------
## logwire
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      296      704      262        1    3.598    0.417    3.033    3.140
##      .25      .50      .75      .90      .95
##    3.334    3.595    3.862    4.083    4.192
##
## lowest : 2.701361 2.714695 2.824351 2.859340 2.873565
## highest: 4.427239 4.452602 4.566949 4.697749 4.718052
## ---------------------------------------------------------------------------
## lninc
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0      218        1    3.957   0.8912    2.890    3.045
##      .25      .50      .75      .90      .95
##    3.367    3.850    4.419    5.046    5.448
##
## lowest : 2.197225 2.302585 2.397895 2.484907 2.564949
## highest: 6.595781 6.833032 6.850126 7.030857 7.419381
## ---------------------------------------------------------------------------
## custcat
##        n  missing distinct     Info     Mean      Gmd
##     1000        0        4    0.936    2.487    1.252
##
## Value          1     2     3     4
## Frequency    266   217   281   236
## Proportion 0.266 0.217 0.281 0.236
## ---------------------------------------------------------------------------
## churn
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2    0.597      274    0.274   0.3982
##
## ---------------------------------------------------------------------------
```

So, we find missing data in 4 columns namely logtoll, logequi, logcard, logwire.

Since all the data that have missing values are of [log of the user], we will replace the missing values with mean of the data.

```
## data
##
##  42  Variables      1000  Observations
## ---------------------------------------------------------------------------
## region
##        n  missing distinct     Info     Mean      Gmd
##     1000        0        3    0.889    2.022   0.8888
##
## Value          1     2     3
## Frequency    322   334   344
## Proportion 0.322 0.334 0.344
```

```
## ---------------------------------------------------------------------------
## tenure
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0       72        1    35.53    24.65        4        7
##      .25      .50      .75      .90      .95
##       17       34       54       66       70
##
## lowest :  1  2  3  4  5, highest: 68 69 70 71 72
## ---------------------------------------------------------------------------
## age
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0       60    0.999    41.68    14.33       23       26
##      .25      .50      .75      .90      .95
##       32       40       51       59       64
##
## lowest : 18 19 20 21 22, highest: 73 74 75 76 77
## ---------------------------------------------------------------------------
## marital
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2     0.75      495    0.495   0.5005
##
## ---------------------------------------------------------------------------
## address
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0       50    0.998    11.55    10.92      0.0      1.0
##      .25      .50      .75      .90      .95
##      3.0      9.0     18.0     26.1     31.0
##
## lowest :  0  1  2  3  4, highest: 45 46 48 49 55
## ---------------------------------------------------------------------------
## income
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0      218        1    77.53     75.3     18.0     21.0
##      .25      .50      .75      .90      .95
##     29.0     47.0     83.0    155.4    232.2
##
## lowest :    9   10   11   12   13, highest:  732  928  944 1131 1668
## ---------------------------------------------------------------------------
## ed
##        n  missing distinct     Info     Mean      Gmd
##     1000        0        5    0.946    2.671    1.369
##
## Value          1     2     3     4     5
## Frequency    204   287   209   234    66
## Proportion 0.204 0.287 0.209 0.234 0.066
## ---------------------------------------------------------------------------
## employ
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0       46    0.997    10.99    10.93        0        0
##      .25      .50      .75      .90      .95
##        3        8       17       25       31
##
## lowest :  0  1  2  3  4, highest: 41 43 44 45 47
## ---------------------------------------------------------------------------
```

```
## retire
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2    0.134       47    0.047  0.08967
##
## ------------------------------------------------------------------------
## gender
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2    0.749      517    0.517   0.4999
##
## ------------------------------------------------------------------------
## reside
##        n  missing distinct     Info     Mean      Gmd
##     1000        0        8    0.923    2.331     1.52
##
## Value          1     2     3     4     5     6     7     8
## Frequency    375   272   138   120    60    29     4     2
## Proportion 0.375 0.272 0.138 0.120 0.060 0.029 0.004 0.002
## ------------------------------------------------------------------------
## tollfree
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2    0.748      474    0.474   0.4991
##
## ------------------------------------------------------------------------
## equip
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2    0.711      386    0.386   0.4745
##
## ------------------------------------------------------------------------
## callcard
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2    0.655      678    0.678   0.4371
##
## ------------------------------------------------------------------------
## wireless
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2    0.625      296    0.296   0.4172
##
## ------------------------------------------------------------------------
## longmon
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0      425        1    11.72    9.559    2.998    3.645
##      .25      .50      .75      .90      .95
##    5.200    8.525   14.413   23.960   31.615
##
## lowest :  0.90  1.05  1.10  1.35  1.45, highest: 62.30 75.45 81.55 89.40 99.95
## ------------------------------------------------------------------------
## tollmon
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0      162    0.855    13.27    16.92     0.00     0.00
##      .25      .50      .75      .90      .95
##     0.00     0.00    24.25    34.75    41.77
##
## lowest :   0.00   5.75   9.00   9.25   9.50, highest:  77.75  78.75  84.00  87.00 173.00
## ------------------------------------------------------------------------
```

```
## equipmon
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0      310    0.769    14.22     19.2     0.00     0.00
##      .25      .50      .75      .90      .95
##     0.00     0.00    31.47    42.66    48.65
##
## lowest :  0.00 15.40 19.55 19.75 19.80, highest: 64.20 69.40 70.05 73.80 77.70
## ------------------------------------------------------------------------------
## cardmon
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0      176    0.967    13.78    14.67     0.00     0.00
##      .25      .50      .75      .90      .95
##     0.00    12.00    20.50    31.50    37.79
##
## lowest :   0.00   2.75   3.75   4.50   4.75, highest:  74.00  82.25  84.75  87.50 109.25
## ------------------------------------------------------------------------------
## wiremon
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0      263    0.651    11.58    17.76     0.00     0.00
##      .25      .50      .75      .90      .95
##     0.00     0.00    24.71    42.11    51.35
##
## lowest :   0.00  14.90  15.10  16.85  17.45, highest:  83.70  85.85  96.25 109.70 111.95
## ------------------------------------------------------------------------------
## longten
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0      960        1    574.1      699    12.30    29.18
##      .25      .50      .75      .90      .95
##    90.14   285.48   755.02  1537.09  2144.79
##
## lowest :    0.90    1.05    1.10    1.35    1.45
## highest: 4333.00 5464.60 5988.50 6353.90 7257.60
## ------------------------------------------------------------------------------
## tollten
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0      473    0.854    551.3    823.1      0.0      0.0
##      .25      .50      .75      .90      .95
##      0.0      0.0    846.9   1838.4   2424.3
##
## lowest :    0.00    5.75   20.75   23.05   26.50
## highest: 4748.45 4905.85 4938.60 5850.25 5916.00
## ------------------------------------------------------------------------------
## equipten
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0      386    0.769    465.6    736.6      0.0      0.0
##      .25      .50      .75      .90      .95
##      0.0      0.0    579.5   1854.9   2435.6
##
## lowest :    0.00   15.40   17.25   22.60   22.80
## highest: 3925.50 4167.70 4758.05 4980.80 5028.65
## ------------------------------------------------------------------------------
## cardten
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0      338    0.967    605.8    775.2      0.0      0.0
```

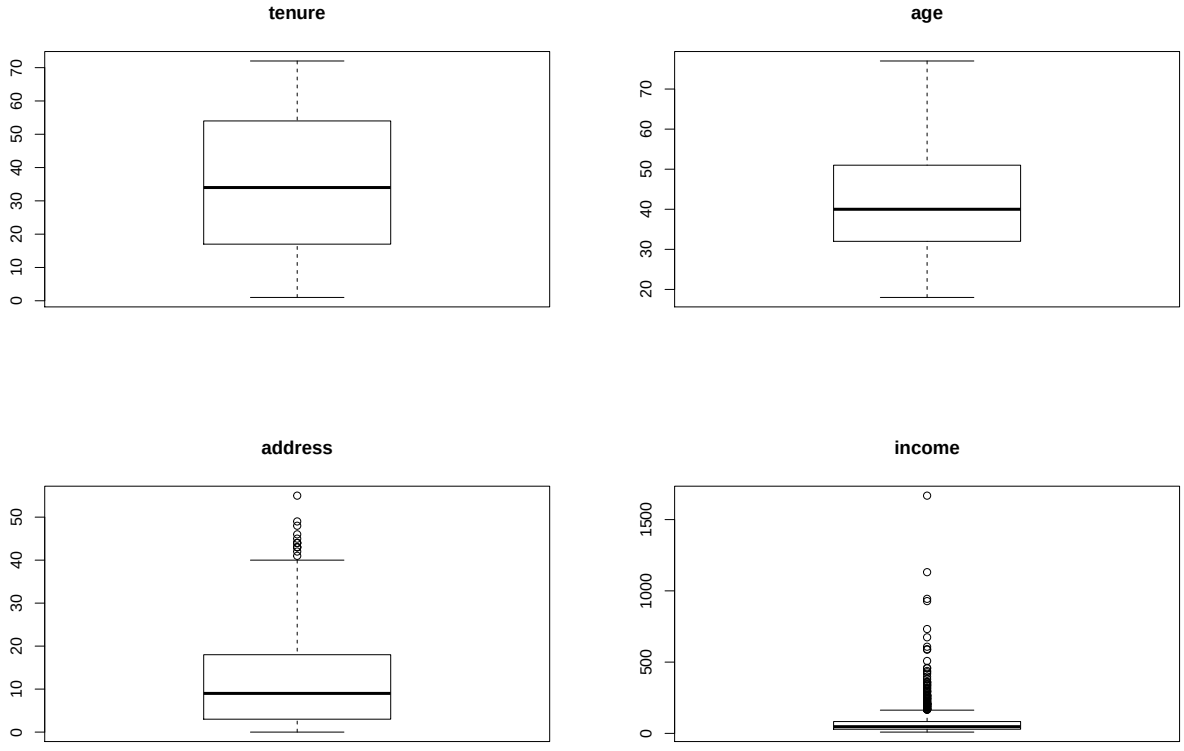
```
##      .25      .50      .75      .90      .95
##      0.0    332.5    910.0   1505.5   2181.0
##
## lowest :    0.00    2.75    5.00   10.00   15.00
## highest: 4975.00 5115.00 5455.00 5520.00 7515.00
## ---------------------------------------------------------------------------
## wireten
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0      297    0.651    442.7    743.3      0.0      0.0
##      .25      .50      .75      .90      .95
##      0.0      0.0    316.5   1833.7   2727.6
##
## lowest :    0.00   10.45   16.85   20.95   24.20
## highest: 4399.60 4828.20 6132.20 6444.95 7856.85
## ---------------------------------------------------------------------------
## multline
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2    0.748      475    0.475   0.4992
##
## ---------------------------------------------------------------------------
## voice
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2    0.635      304    0.304   0.4236
##
## ---------------------------------------------------------------------------
## pager
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2    0.579      261    0.261   0.3861
##
## ---------------------------------------------------------------------------
## internet
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2    0.698      368    0.368   0.4656
##
## ---------------------------------------------------------------------------
## callid
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2    0.749      481    0.481   0.4998
##
## ---------------------------------------------------------------------------
## callwait
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2    0.749      485    0.485   0.5001
##
## ---------------------------------------------------------------------------
## forward
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2     0.75      493    0.493   0.5004
##
## ---------------------------------------------------------------------------
## confer
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2     0.75      502    0.502   0.5005
##
```

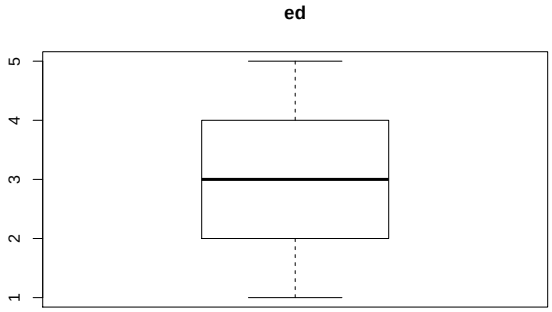
```
## ------------------------------------------------------------------------
## ebill
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2      0.7      371    0.371   0.4672
##
## ------------------------------------------------------------------------
## loglong
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0      425        1    2.182    0.829    1.098    1.293
##      .25      .50      .75      .90      .95
##    1.649    2.143    2.668    3.176    3.454
##
## lowest : -0.105361  0.048790  0.095310  0.300105  0.371564
## highest:  4.131961  4.323470  4.401216  4.493121  4.604670
## ------------------------------------------------------------------------
## logtoll
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0      162    0.855     3.24   0.2626    2.757    2.918
##      .25      .50      .75      .90      .95
##    3.240    3.240    3.240    3.548    3.732
##
## lowest : 1.749200 2.197225 2.224624 2.251292 2.277267
## highest: 4.353499 4.366278 4.430817 4.465908 5.153292
## ------------------------------------------------------------------------
## logequi
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0      310    0.769    3.568   0.1544    3.240    3.372
##      .25      .50      .75      .90      .95
##    3.568    3.568    3.568    3.753    3.885
##
## lowest : 2.734368 2.972975 2.983153 2.985682 2.990720
## highest: 4.162003 4.239887 4.249209 4.301359 4.352855
## ------------------------------------------------------------------------
## logcard
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0      176    0.967    2.854   0.4806    2.079    2.277
##      .25      .50      .75      .90      .95
##    2.674    2.854    3.020    3.450    3.632
##
## lowest : 1.011601 1.321756 1.504077 1.558145 1.609438
## highest: 4.304065 4.409763 4.439706 4.471639 4.693639
## ------------------------------------------------------------------------
## logwire
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0      263    0.651    3.598     0.16    3.231    3.425
##      .25      .50      .75      .90      .95
##    3.598    3.598    3.598    3.740    3.939
##
## lowest : 2.701361 2.714695 2.824351 2.859340 2.873565
## highest: 4.427239 4.452602 4.566949 4.697749 4.718052
## ------------------------------------------------------------------------
## lninc
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     1000        0      218        1    3.957   0.8912    2.890    3.045
```

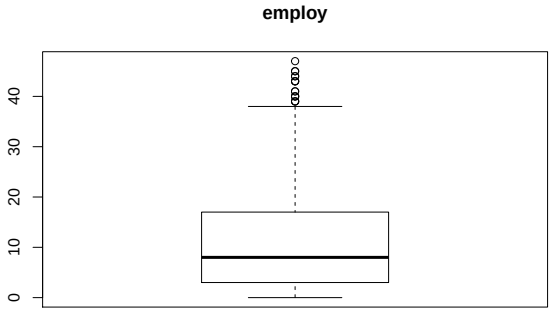
```
##      .25      .50      .75      .90      .95
##    3.367    3.850    4.419    5.046    5.448
##
## lowest : 2.197225 2.302585 2.397895 2.484907 2.564949
## highest: 6.595781 6.833032 6.850126 7.030857 7.419381
## ---------------------------------------------------------------------------
## custcat
##        n  missing distinct     Info     Mean      Gmd
##     1000        0        4    0.936    2.487    1.252
##
## Value          1     2     3     4
## Frequency    266   217   281   236
## Proportion 0.266 0.217 0.281 0.236
## ---------------------------------------------------------------------------
## churn
##        n  missing distinct     Info      Sum     Mean      Gmd
##     1000        0        2    0.597      274    0.274   0.3982
##
## ---------------------------------------------------------------------------
```

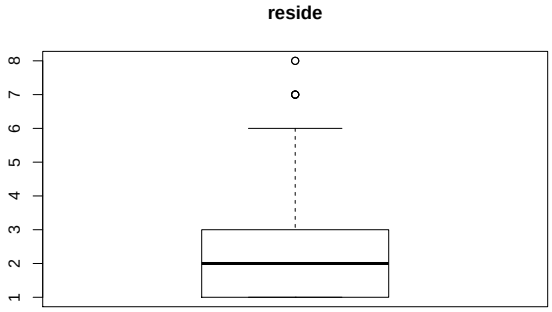
Result: So no missing values present now.

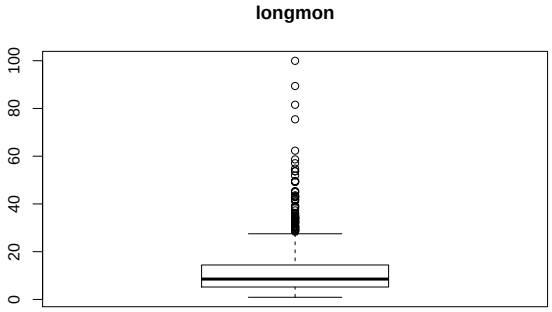
Step 2: Checking for any outliers by histogram method.

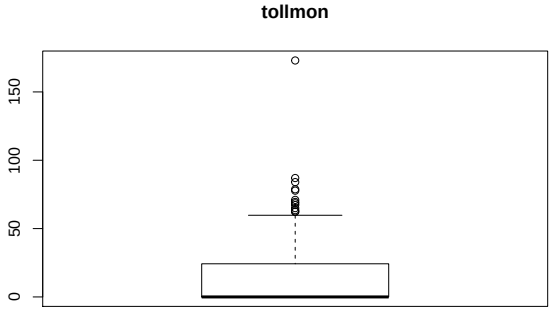


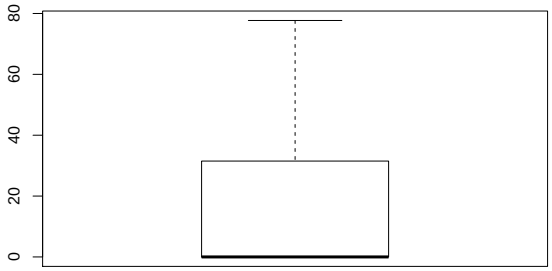

ed
1
2
3
4
5

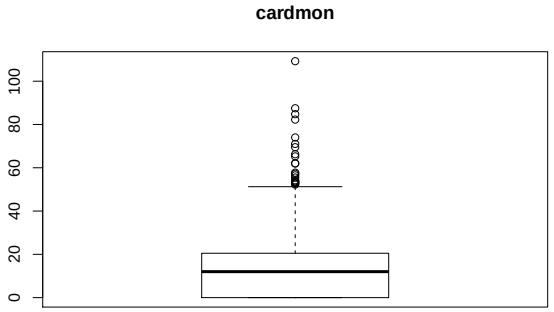

employ
0
10
20
30
40

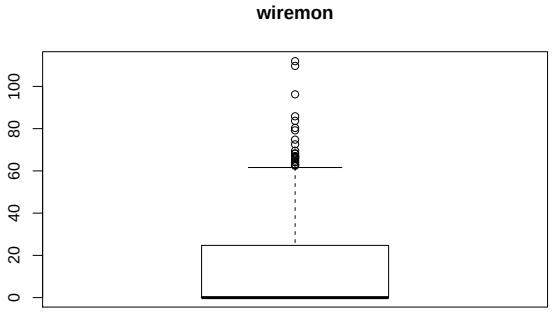

reside
1
2
3
4
5
6
7
8

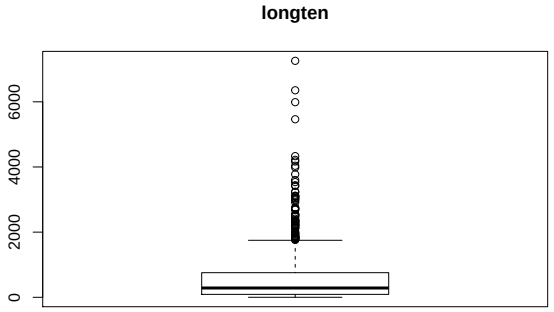

longmon
0
20
40
60
80
100

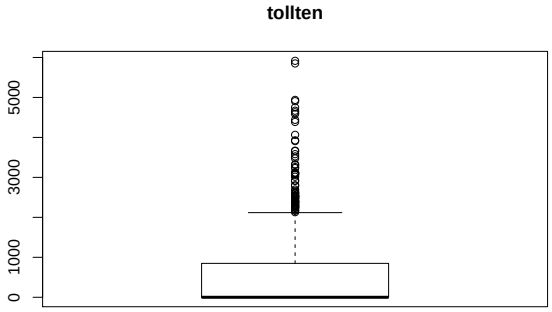

tollmon
0
50
100
150

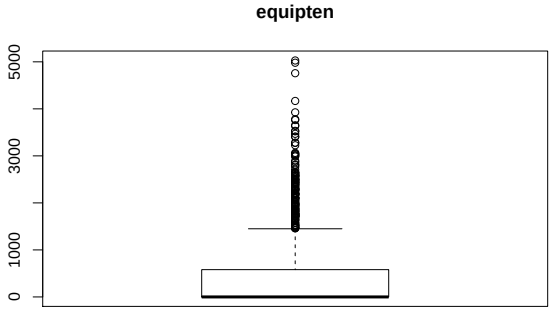

equipmon
0
20
40
60
80

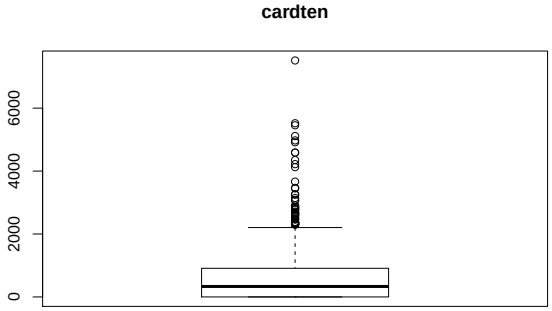

cardmon
0
20
40
60
80
100

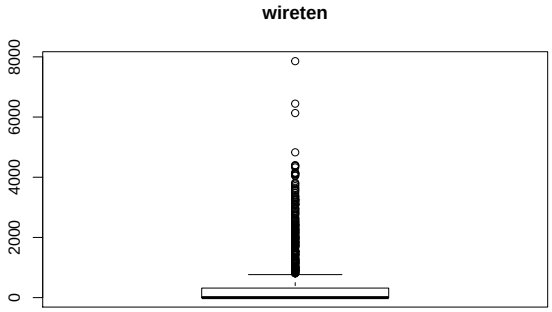

wiremon
0
20
40
60
80
100

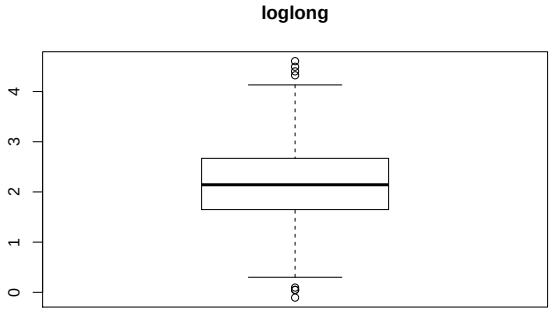

longten
0
2000
4000
6000

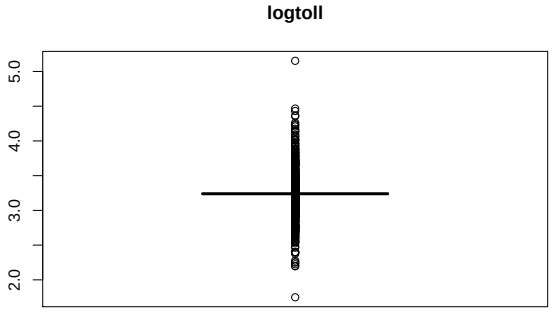

tollten
0
1000
3000
5000

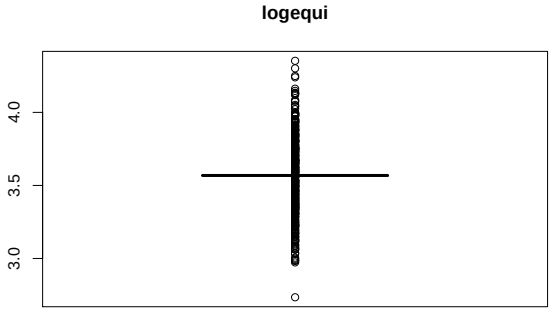

equipten
0
1000
3000
5000

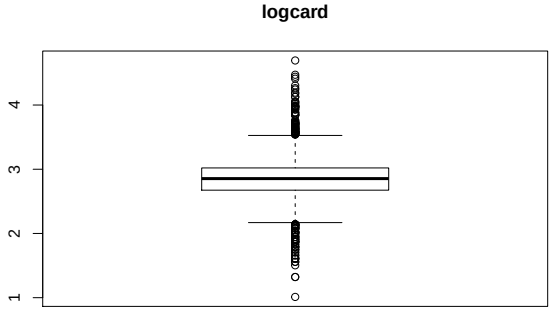

cardten
0
2000
4000
6000

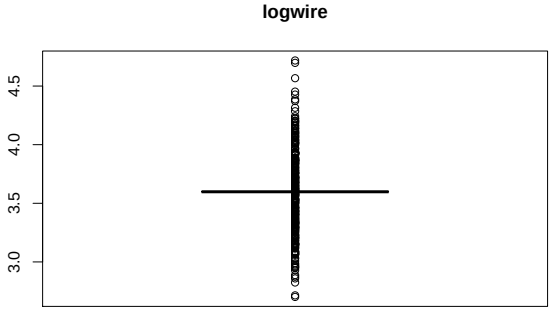

wireten
0
2000
4000
6000
8000

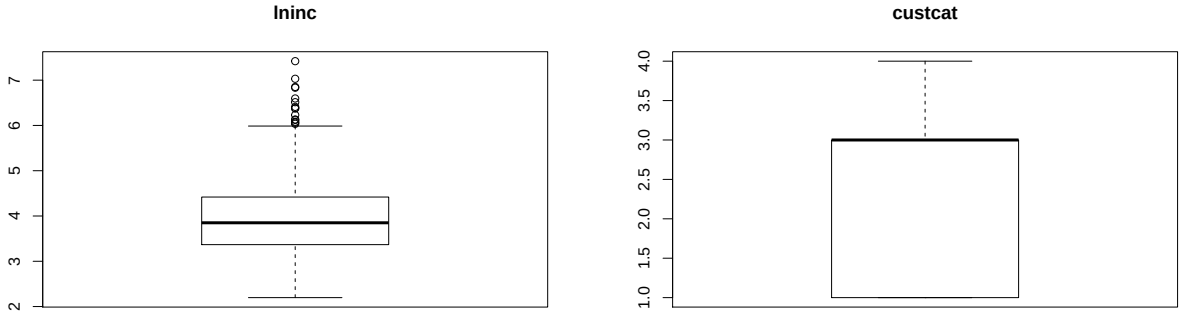

loglong
0
1
2
3
4


logtoll
2.0
3.0
4.0
5.0


logequi
3.0
3.5
4.0

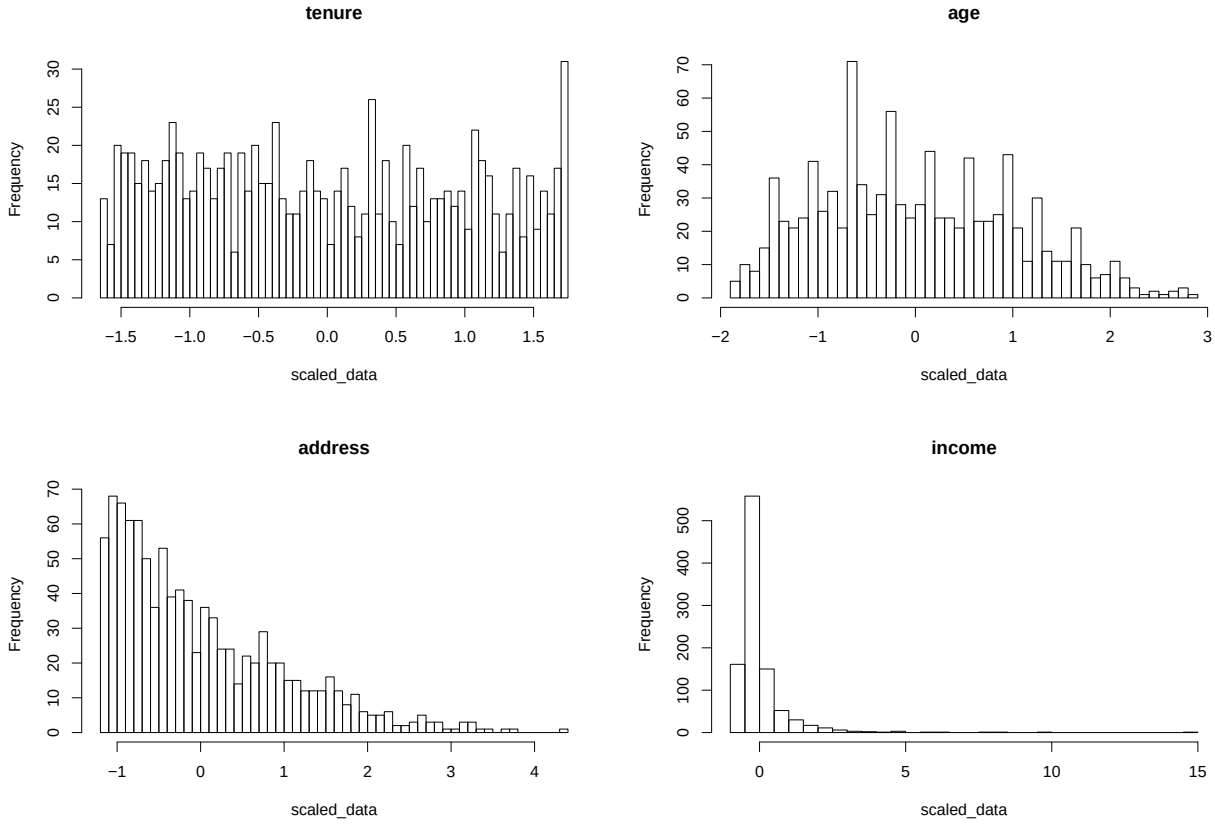

logcard
1
2
3
4

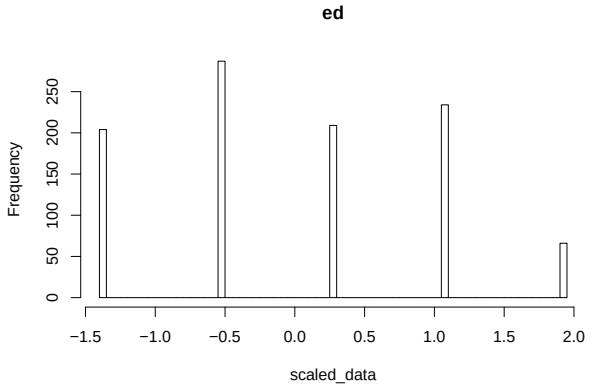

logwire
3.0
3.5
4.0
4.5

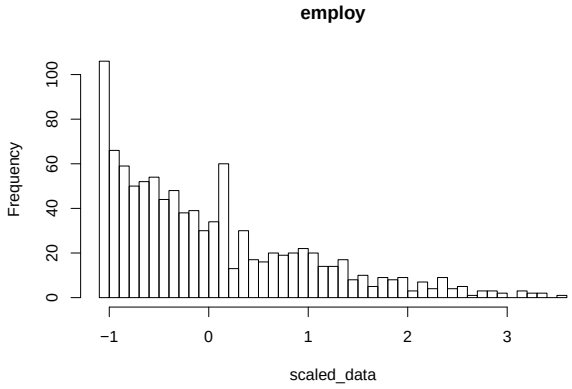


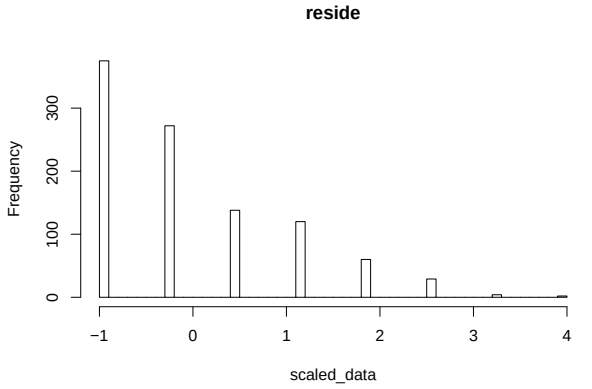
Result : There are outliers in many columns but it is not necessary that all the parameters of the same row are having extreme data, so we are not removing any data from it.

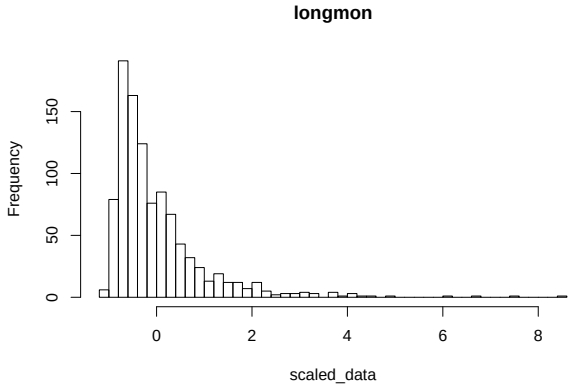
Step 3: Normality test

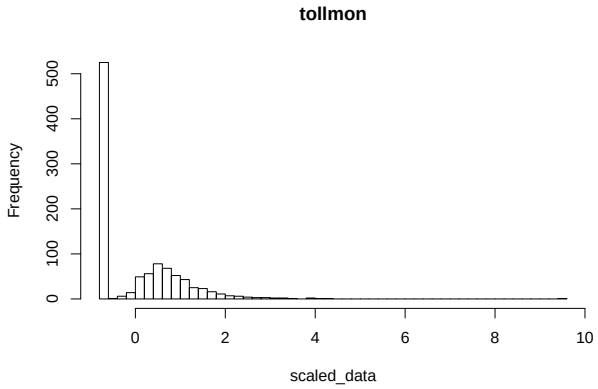


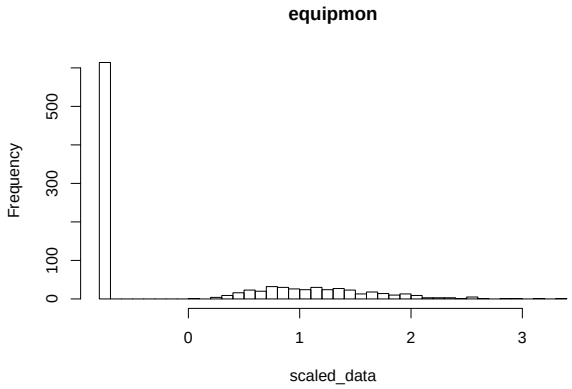

ed
Frequency
0
50
100
150
200
250
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
scaled_data

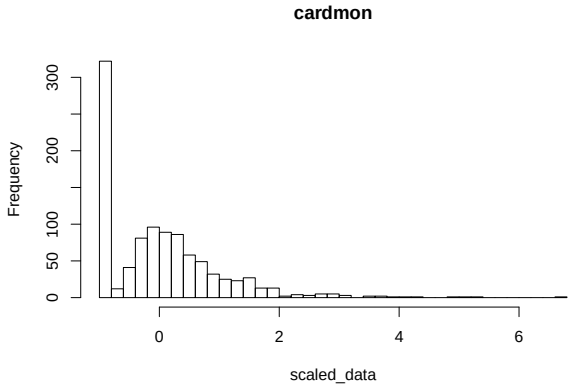

employ
Frequency
0
20
40
60
80
100
−1
0
1
2
3
scaled_data

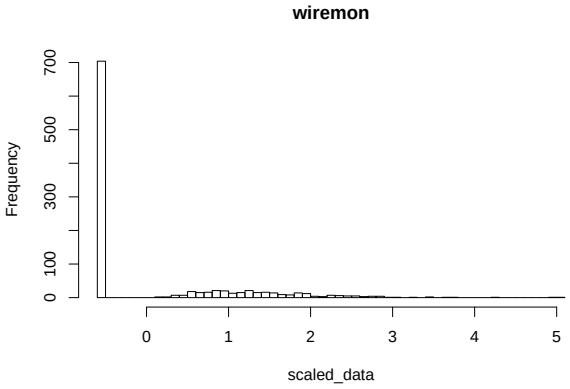

reside
Frequency
0
100
200
300
−1
0
1
2
3
4
scaled_data

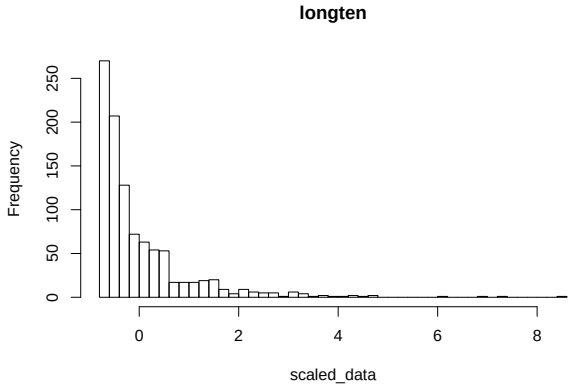

longmon
Frequency
0
50
100
150
0
2
4
6
8
scaled_data

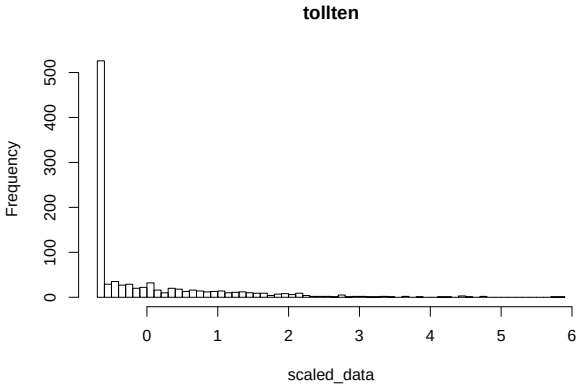

tollmon
Frequency
0
100
200
300
400
500
0
2
4
6
8
10
scaled_data

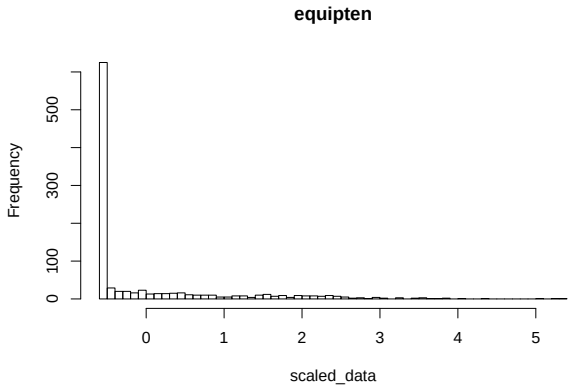

equipmon
Frequency
0
100
300
500
0
1
2
3
scaled_data

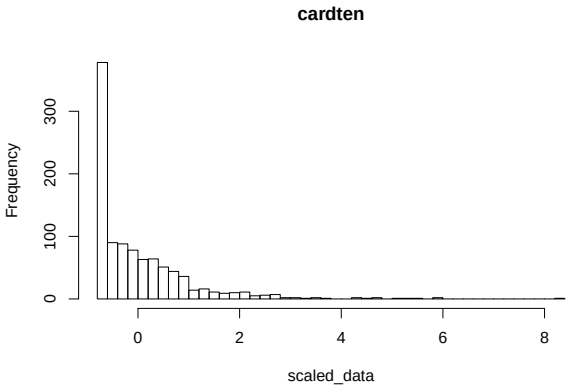

cardmon
Frequency
0
50
100
200
300
0
2
4
6
scaled_data

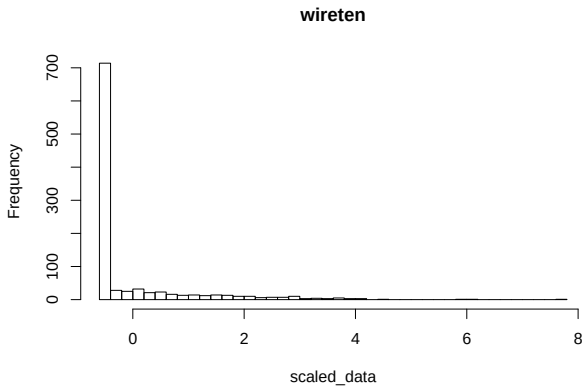

wiremon
Frequency
0
100
300
500
700
0
1
2
3
4
5
scaled_data

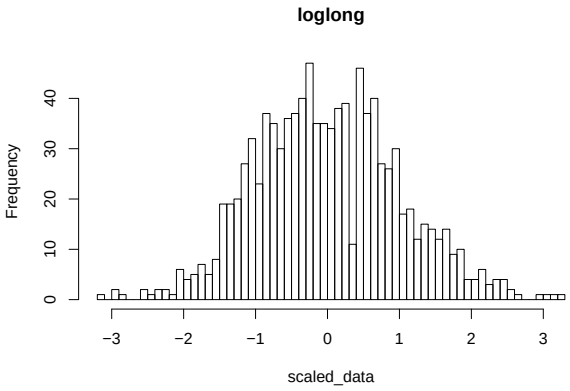

longten
Frequency
0
50
100
150
200
250
0
2
4
6
8
scaled_data

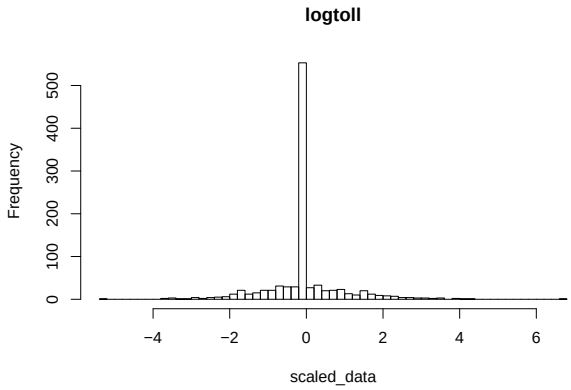

tollten
Frequency
0
100
200
300
400
500
0
1
2
3
4
5
6
scaled_data

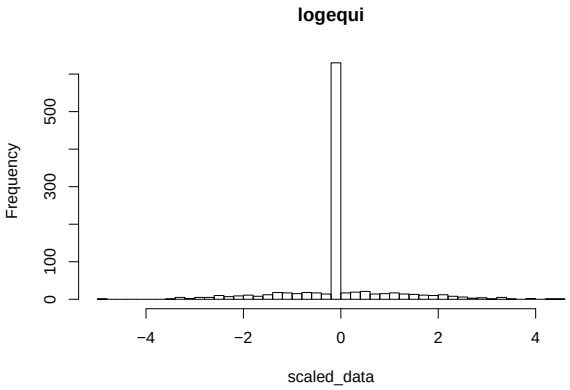

equipten
Frequency
0
100
300
500
0
1
2
3
4
5
scaled_data

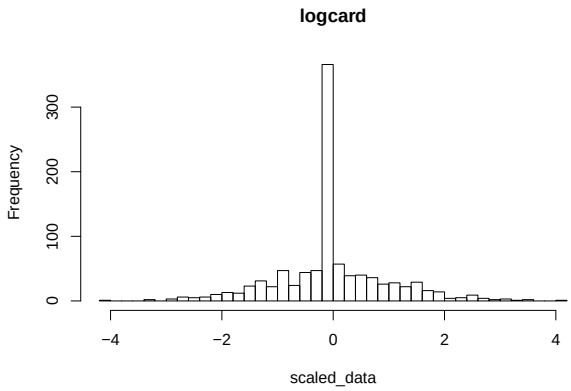

cardten
Frequency
0
100
200
300
0
2
4
6
8
scaled_data

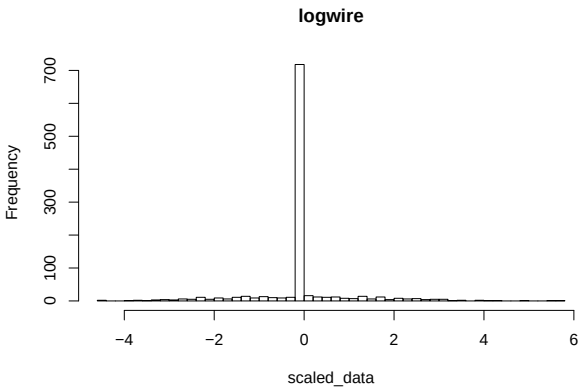

wireten
Frequency
0
100
300
500
700
0
2
4
6
8
scaled_data

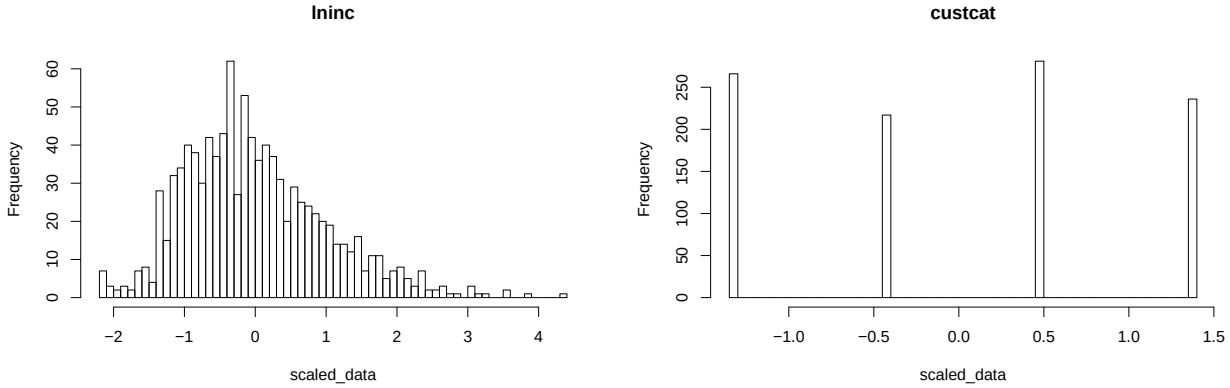

loglong
Frequency
0
10
20
30
40
−3
−2
−1
0
1
2
3
scaled_data


logtoll
Frequency
0
100
200
300
400
500
−4
−2
0
2
4
6
scaled_data


logequi
Frequency
0
100
300
500
−4
−2
0
2
4
scaled_data


logcard
Frequency
0
100
200
300
−4
−2
0
2
4
scaled_data


logwire
Frequency
0
100
300
500
700
−4
−2
0
2
4
6
scaled_data



2.3.3 PCA on Scaled Data

```
##
## Attaching package: 'psych'

## The following object is masked from 'package:Hmisc':
##
##     describe

## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha

## Principal Components Analysis
## Call: principal(r = data, nfactors = 4, residuals = FALSE, rotate = "none",
##     scores = TRUE)
## Standardized loadings (pattern matrix) based upon correlation matrix
##             PC1   PC2   PC3   PC4     h2   u2 com
## region    0.01 -0.02 -0.02 -0.10 0.0099 0.99 1.2
## tenure    0.55 -0.60  0.34  0.07 0.7865 0.21 2.6
## age       0.37 -0.54  0.14 -0.38 0.5875 0.41 2.8
## marital   0.12 -0.04  0.06  0.51 0.2820 0.72 1.2
## address   0.35 -0.53  0.18 -0.22 0.4798 0.52 2.4
## income    0.33 -0.19  0.06 -0.62 0.5308 0.47 1.8
## ed        0.15  0.50  0.32 -0.14 0.3951 0.60 2.1
## employ    0.42 -0.56  0.07 -0.44 0.6903 0.31 2.8
## retire    0.13 -0.31  0.02  0.00 0.1116 0.89 1.4
## gender    0.01 -0.02  0.04 -0.04 0.0035 1.00 2.4
## reside    0.01  0.13 -0.05  0.54 0.3152 0.68 1.1
## tollfree  0.60  0.14 -0.54 -0.06 0.6779 0.32 2.1
## equip     0.17  0.65  0.50 -0.05 0.7123 0.29 2.0
## callcard  0.58 -0.22 -0.11  0.09 0.4102 0.59 1.4
## wireless  0.63  0.55 -0.02 -0.04 0.6939 0.31 2.0
## longmon   0.50 -0.59  0.35  0.11 0.7318 0.27 2.7
## tollmon   0.70 -0.01 -0.45  0.03 0.7012 0.30 1.7
## equipmon  0.36  0.68  0.48 -0.02 0.8238 0.18 2.4
## cardmon   0.59 -0.30  0.03  0.34 0.5519 0.45 2.2
## wiremon   0.73  0.51  0.06  0.00 0.7927 0.21 1.8
## longten   0.52 -0.60  0.35  0.10 0.7610 0.24 2.7
## tollten   0.74 -0.21 -0.23  0.01 0.6435 0.36 1.4
```

```
## equipten  0.51  0.36  0.56  0.04 0.7076 0.29 2.7
## cardten   0.57 -0.49  0.25  0.30 0.7198 0.28 2.9
## wireten   0.74  0.27  0.18  0.01 0.6510 0.35 1.4
## multline  0.40  0.05  0.49  0.06 0.4009 0.60 2.0
## voice     0.55  0.48 -0.01 -0.02 0.5344 0.47 2.0
## pager     0.57  0.51 -0.03 -0.02 0.5814 0.42 2.0
## internet  0.17  0.60  0.40 -0.06 0.5547 0.45 2.0
## callid    0.60  0.17 -0.51 -0.04 0.6438 0.36 2.1
## callwait  0.60  0.13 -0.52 -0.02 0.6445 0.36 2.1
## forward   0.60  0.15 -0.48  0.02 0.6144 0.39 2.1
## confer    0.61  0.11 -0.46 -0.03 0.5916 0.41 1.9
## ebill     0.10  0.57  0.40 -0.11 0.5050 0.50 2.0
## loglong   0.54 -0.59  0.34  0.12 0.7619 0.24 2.7
## logtoll   0.43 -0.23  0.00  0.13 0.2536 0.75 1.7
## logequi   0.59  0.18  0.02  0.06 0.3875 0.61 1.2
## logcard   0.28 -0.19  0.11  0.37 0.2628 0.74 2.6
## logwire   0.39  0.02  0.18  0.08 0.1960 0.80 1.5
## lninc     0.41 -0.20  0.11 -0.60 0.5917 0.41 2.1
## custcat   0.76  0.32 -0.27  0.03 0.7560 0.24 1.6
## churn    -0.15  0.46  0.04 -0.02 0.2341 0.77 1.2
##
##                        PC1  PC2  PC3  PC4
## SS loadings           9.82 6.51 3.78 2.18
## Proportion Var        0.23 0.15 0.09 0.05
## Cumulative Var        0.23 0.39 0.48 0.53
## Proportion Explained  0.44 0.29 0.17 0.10
## Cumulative Proportion 0.44 0.73 0.90 1.00
##
## Mean item complexity =  2
## Test of the hypothesis that 4 components are sufficient.
##
## The root mean square of the residuals (RMSR) is  0.06
##  with the empirical chi square  6880.17  with prob <  0
##
## Fit based upon off diagonal values = 0.95
```

Result: As a result of PCA, Components having highest variability and which will show the same variation as complete data according to PCA are custcat, equipmon, equipten, marital

3 Assignment B

3.1 Q.1]

3.1.1 A] Reading data and making data into training and test data.

```
## [1] "head of training data"
```

```
##           Y1         X1        X2
## 1 -1.0565192  -6.236444 0.9615355
## 2 -0.5754127  -3.873848 0.5050130
## 3  5.0910630   5.640287 0.7175317
## 4  2.9475637   1.191125 0.3074231
## 5  2.9519538 -10.849769 0.5960600
## 6  3.1685278   2.603705 0.3109550
```

```
## [1] "head of testing data"
```

```
##            Y1         X1         X2
## 10 -0.7316911 -2.2906586 0.43611757
## 20  1.1998000  7.7123714 0.47222562
## 30  0.6124209 -1.4169026 0.01161898
## 40 -0.1139879  0.6901132 0.48192669
## 50  3.5655124  1.3302962 0.49526489
## 60  4.3900710  3.1920603 0.98896327
```

Fitting data for linear regression model

```
##
## Call:
## lm(formula = Y1 ~ X1 + X2, data = trainData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.1226 -1.3189 -0.0519  1.1825  3.3815
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.77301    0.33680   8.233 1.66e-12 ***
## X1           0.19480    0.03465   5.623 2.24e-07 ***
## X2          -0.15753    0.61320  -0.257    0.798
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.711 on 87 degrees of freedom
## Multiple R-squared:  0.2666, Adjusted R-squared:  0.2498
## F-statistic: 15.81 on 2 and 87 DF,  p-value: 1.387e-06
```

Several metrics useful for regression diagnostics : model.diag.metrics

```
## # A tibble: 6 x 11
##   .rownames     Y1     X1    X2 .fitted .se.fit   .resid   .hat .sigma
##   <chr>      <dbl>  <dbl> <dbl>   <dbl>   <dbl>    <dbl>  <dbl>  <dbl>
## 1 1         -1.06   -6.24 0.962   1.41    0.460 -2.46    0.0724   1.70
## 2 2         -0.575  -3.87 0.505   1.94    0.265 -2.51    0.0240   1.70
## 3 3          5.09    5.64 0.718   3.76    0.272  1.33    0.0252   1.71
## 4 4          2.95    1.19 0.307   2.96    0.203 -0.00905 0.0141   1.72
```

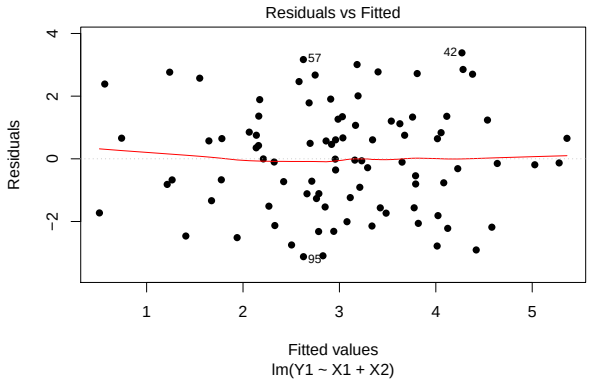
```
## 5 5          2.95  -10.8  0.596   0.566   0.480  2.39    0.0788   1.70
## 6 6          3.17    2.60 0.311   3.23    0.206 -0.0627  0.0145   1.72
## # ... with 2 more variables: .cooksd <dbl>, .std.resid <dbl>
```

Meta Data for model.diag.metrics Among the table columns, there are:

Y1: original values X1, X2: the observed values .fitted: the fitted values .resid: the residual errors

Let's see correlation between the features:

We can see the plot in residual and fitted plot here now:



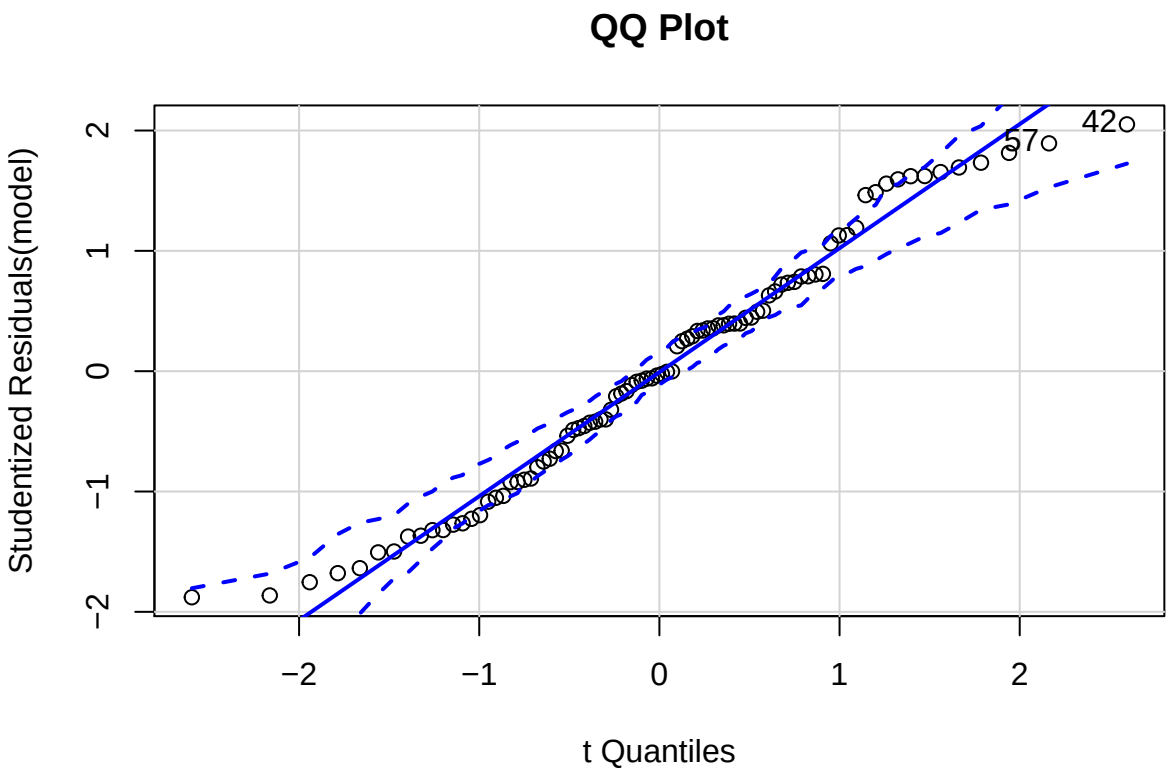
Note how the residuals plot of this last model shows some important points still lying far away from the middle area of the graph. Since the behaviour is random in nature we were successful in this test.

```
## Loading required package: carData
```

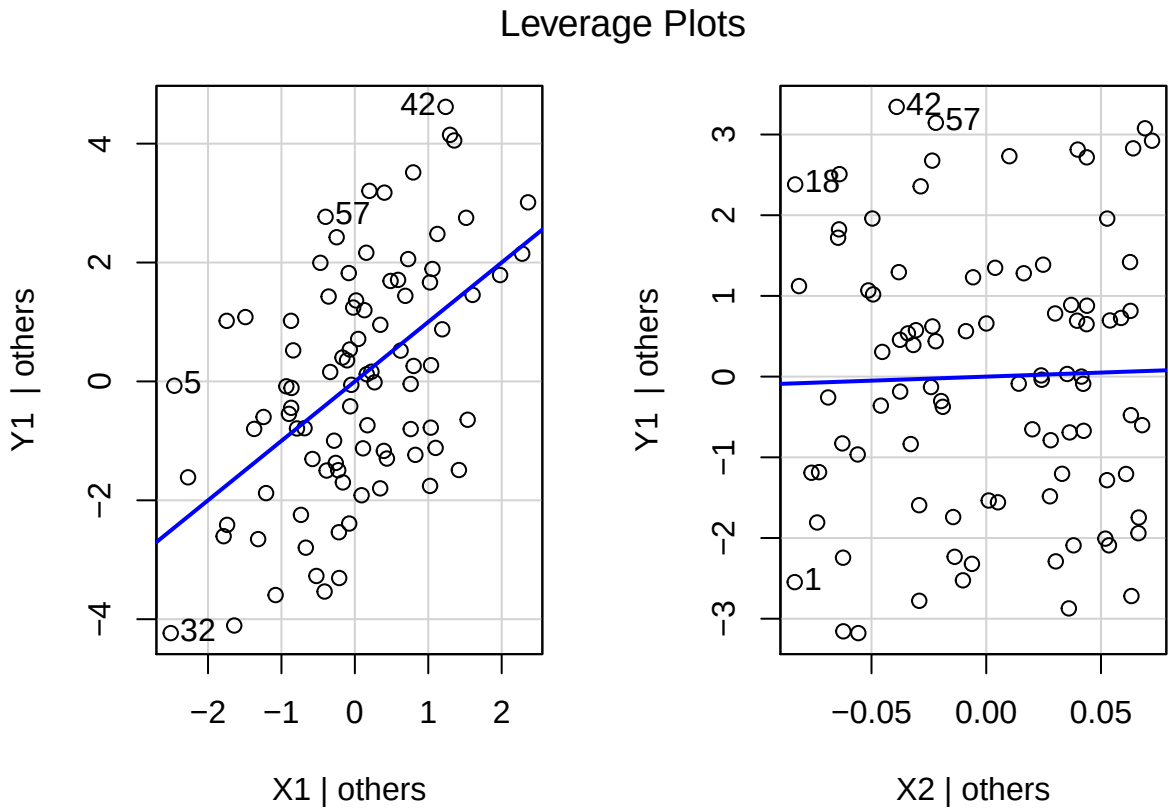
```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:psych':
##
##     logit
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```


QQ Plot
Studentized Residuals(model)
t Quantiles
42
57
−2
−1
0
1
2

```
## 42 57
## 38 52
```



Check outliers:

Check if errors are auto corelated

```
## Loading required package: zoo
```

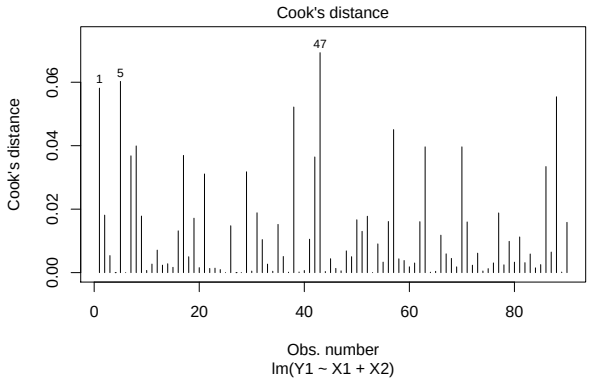
```
##
## Attaching package: 'zoo'
```

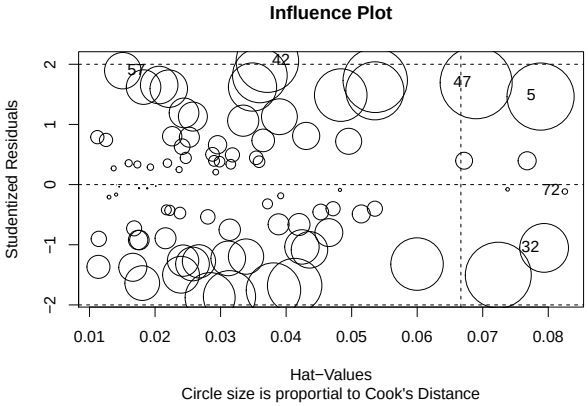
```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
##
##  Durbin-Watson test
##
## data:  model
## DW = 2.1766, p-value = 0.7891
## alternative hypothesis: true autocorrelation is greater than 0
```

We can see that there are outliers in this dataset mainly row 5,42,57,32 from X1, and 18,42,57,1 from X2. So, overall number 42 and 57 are outliers.

Influential Variables : Fiding influential data points is required. So let us look at variable plots:






```
##       StudRes        Hat        CookD
## 5   1.463036 0.07882556 0.0602640116
## 32 -1.052152 0.07936606 0.0317723820
## 42  2.051376 0.03714193 0.0521849091
## 47  1.692974 0.06899694 0.0693171600
## 57  1.893423 0.01507526 0.0177631223
## 72 -0.115048 0.08253501 0.0004014576
```

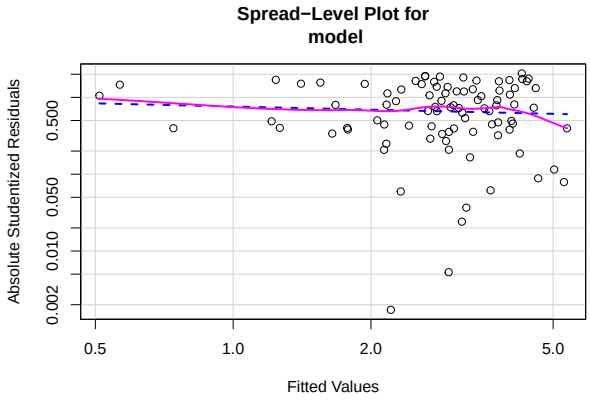
Checking for Multi-collinearity:

```
##       X1       X2
## 1.004315 1.004315
```

```
##    X1    X2
## FALSE FALSE
```

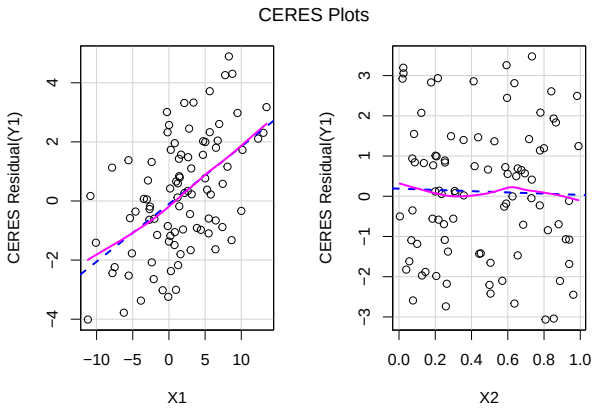
Non-constant Error Variance:

```
## Non-constant Variance Score Test
## Variance formula: ~ fitted.values
## Chisquare = 0.0003181395, Df = 1, p = 0.98577
```




```
##
## Suggested power transformation:  1.136489
```

Nonlinearity test:



So all the factors are linear which is required.

Non-independence of Errors:

```
##  lag Autocorrelation D-W Statistic p-value
##    1      -0.1061408       2.17664   0.444
##  Alternative hypothesis: rho != 0
```

3.1.2 q.1] B] Reading data and making data into training and test data.

```
## [1] "head of training data"
```

```
##         Y1          X1        X2 X3
## 1 4.327805  1.33515849 0.4730140  6
## 2 8.139683 17.33750408 0.3195393  7
## 3 3.193994  6.79640276 0.3452665  3
## 4 1.869722  0.08149526 0.1059610  3
## 5 6.148741  2.87826189 0.2465312  8
## 6 7.611793  6.02848032 0.2045437  9
```

```
## [1] "head of testing data"
```

```
##             Y1         X1         X2 X3
## 10 5.36699195   4.395668 0.84454209  7
## 20 2.15690801  -5.920465 0.11914961  5
## 30 3.07860126 -12.195155 0.05738101  8
## 40 0.98051223  -1.257590 0.21801173  2
## 50 0.09861866  -2.467828 0.26064943  1
## 60 4.70909399   7.368437 0.20267976  5
```

Fitting data for linear regression model

```
##
## Call:
## lm(formula = Y1 ~ X1 + X2 + X3, data = trainData)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.21901 -0.06800 -0.01703  0.07815  0.28281
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.037081   0.027754  -1.336    0.185
## X1           0.201507   0.002523  79.857   <2e-16 ***
```

```
## X2          -0.454286   0.038373 -11.839   <2e-16 ***
## X3           0.705114   0.003951 178.466   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1068 on 86 degrees of freedom
## Multiple R-squared:  0.9977, Adjusted R-squared:  0.9976
## F-statistic: 1.219e+04 on 3 and 86 DF,  p-value: < 2.2e-16
```

Several metrics useful for regression diagnostics : model.diag.metrics

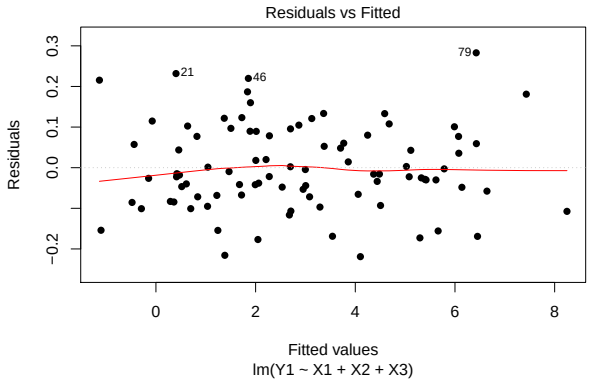
```
## # A tibble: 6 x 12
##   .rownames    Y1      X1    X2    X3 .fitted .se.fit  .resid   .hat .sigma
##   <chr>     <dbl>   <dbl> <dbl> <int>   <dbl>   <dbl>   <dbl>  <dbl>  <dbl>
## 1 1          4.33  1.34   0.473     6    4.25  0.0140  0.0800 0.0172  0.107
## 2 2          8.14 17.3    0.320     7    8.25  0.0433 -0.107  0.164   0.107
## 3 3          3.19  6.80   0.345     3    3.29  0.0173 -0.0969 0.0263  0.107
## 4 4          1.87  0.0815 0.106     3    2.05  0.0194 -0.177  0.0329  0.106
## 5 5          6.15  2.88   0.247     8    6.07  0.0215  0.0769 0.0404  0.107
## 6 6          7.61  6.03   0.205     9    7.43  0.0274  0.181  0.0657  0.106
## # ... with 2 more variables: .cooksd <dbl>, .std.resid <dbl>
```

Meta Data for model.diag.metrics Among the table columns, there are:

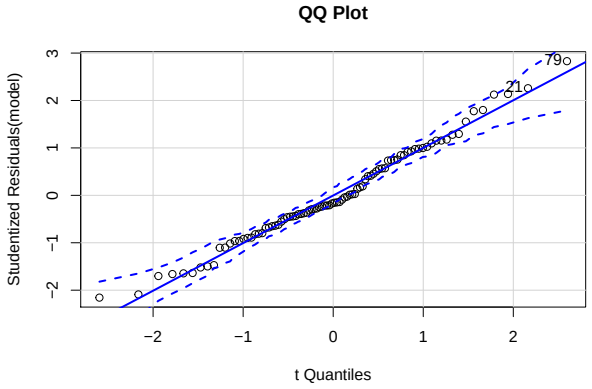
Y1: original values X1, X2: the observed values .fitted: the fitted values .resid: the residual errors

Let's see correlation between the features:

We can see the plot in residual and fitted plot here now:

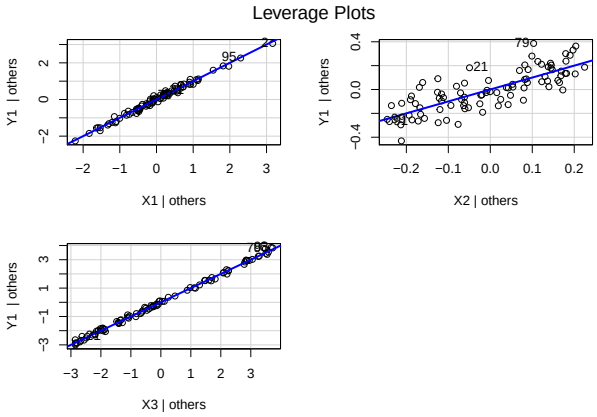


Note how the residuals plot of this last model shows some important points still lying far away from the middle area of the graph. Since the behaviour is random in nature we were successful in this test.




```
## 21 79
## 19 72
```

Check outliers:

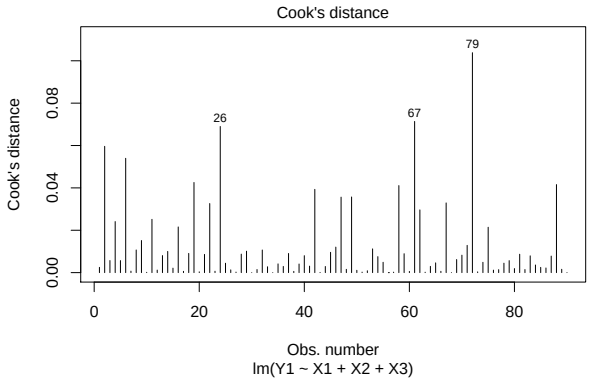


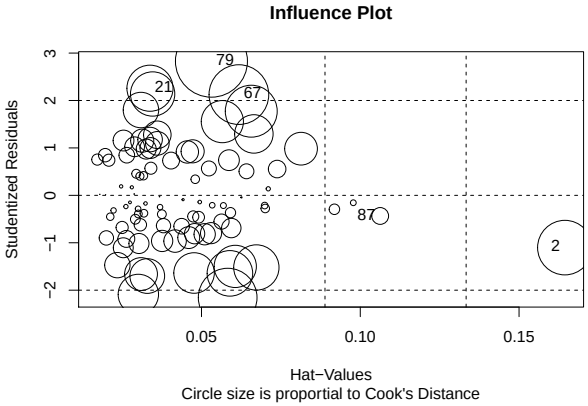
Check if errors are auto corelated

```
##
##  Durbin-Watson test
##
## data:  model
## DW = 2.2712, p-value = 0.9044
## alternative hypothesis: true autocorrelation is greater than 0
```

We can see that there are outliers in this dataset mainly row 5,42,57,32 from X1, and 18,42,57,1 from X2. So, overall number 42 and 57 are outliers.

Influential Variables : Fiding influential data points is required. So let us look at variable plots:






```
##       StudRes        Hat       CookD
## 2  -1.1022153 0.16443269 0.059620533
## 21  2.2588599 0.03377374 0.042557952
## 67  2.1243499 0.06177616 0.071370638
## 79  2.8291379 0.05313276 0.103828905
## 87 -0.4347168 0.10626600 0.005670926
```

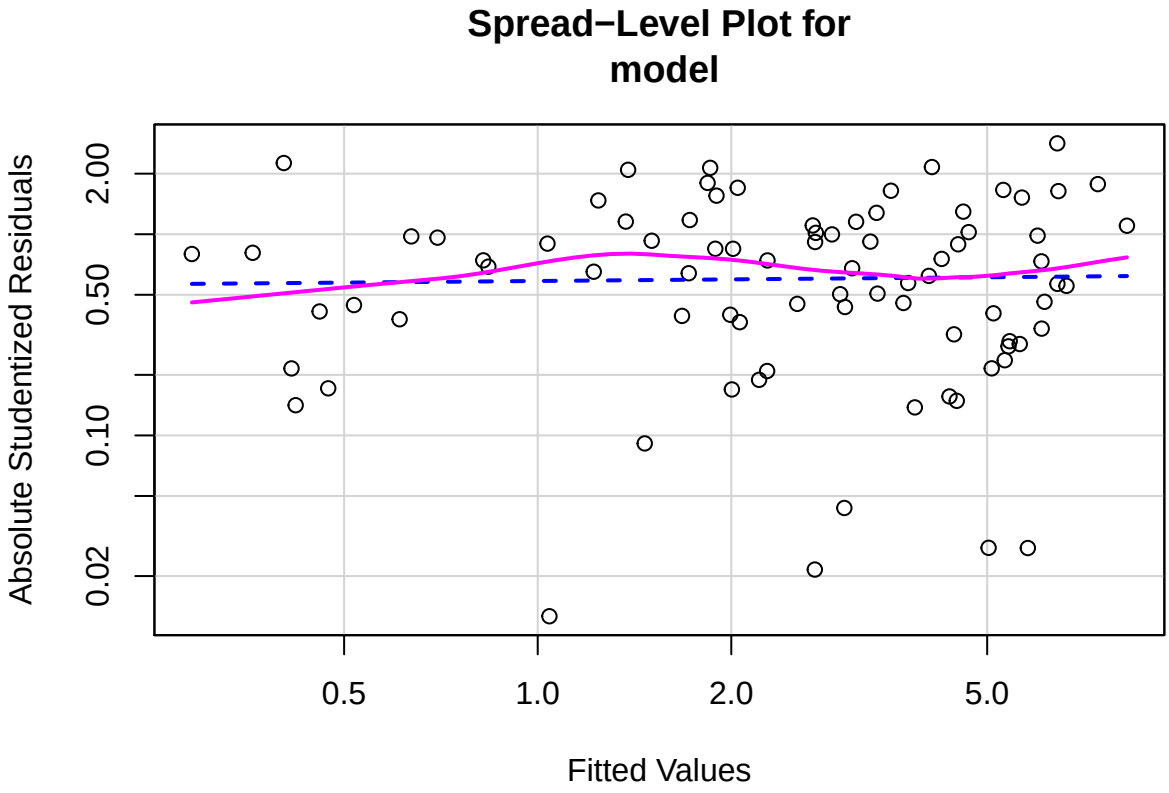
Checking for Multi-collinearity:

```
##       X1       X2       X3
## 1.014390 1.008497 1.010662
```

```
##    X1    X2    X3
## FALSE FALSE FALSE
```

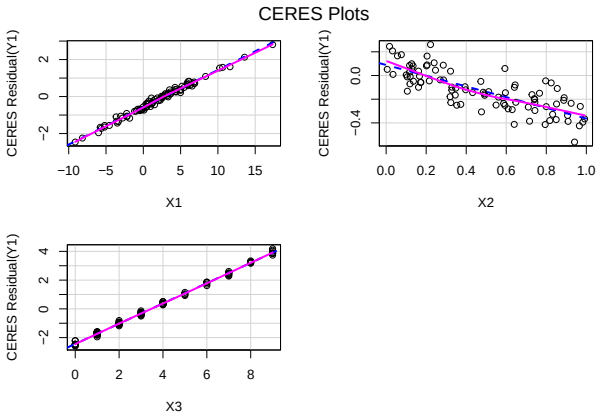
Non-constant Error Variance:

```
## Non-constant Variance Score Test
## Variance formula: ~ fitted.values
## Chisquare = 0.007201054, Df = 1, p = 0.93237
```




```
##
## Suggested power transformation:  0.9733684
```

Nonlinearity test:



So all the factors are linear which is required.

Non-independence of Errors:

```
##  lag Autocorrelation D-W Statistic p-value
##    1       -0.138871      2.271194   0.186
##  Alternative hypothesis: rho != 0
```

3.1.3 q.1] C]

Reading data and making data into training and test data.

```
## [1] "head of training data"
```

```
##         Y1        X1        X2       X4
## 1 3.394395  3.442101 0.3286708 2.531468
## 2 2.783206 -1.913935 0.6070572 3.242823
## 3 5.332787 13.909262 0.5780297 2.631212
## 4 4.446709  3.046078 0.7084997 3.883400
## 5 5.543534  4.404784 0.2912077 4.316483
## 6 3.916571 -4.656940 0.1686171 4.267447
```

```
## [1] "head of testing data"
```

```
##            Y1         X1         X2        X4
## 10  2.9731911  4.4548862 0.03022539 1.8120901
## 20  5.9505248  9.6564968 0.36800421 3.7472017
## 30  1.3414354  1.2784499 0.48953224 1.3958129
## 40 -0.4241599  0.3036483 0.31690684 0.1267627
## 50  1.5125398 -0.6742719 0.94945253 2.1797810
## 60  0.9853793  3.2237377 0.85544483 0.9421779
```

Fitting data for linear regression model

```
##
## Call:
## lm(formula = Y1 ~ X1 + X2 + X4, data = trainData)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.210075 -0.064858 -0.001555  0.069632  0.256274
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.066625   0.027747   2.401   0.0185 *
## X1           0.199642   0.002292  87.102   <2e-16 ***
## X2          -1.051784   0.037636 -27.946   <2e-16 ***
## X4           1.160520   0.005730 202.526   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.09636 on 86 degrees of freedom
## Multiple R-squared:  0.9982, Adjusted R-squared:  0.9982
## F-statistic: 1.618e+04 on 3 and 86 DF,  p-value: < 2.2e-16
```

Several metrics useful for regression diagnostics : model.diag.metrics

```
## # A tibble: 6 x 12
##   .rownames    Y1     X1    X2    X4 .fitted .se.fit   .resid   .hat .sigma
##   <chr>     <dbl>  <dbl> <dbl> <dbl>   <dbl>   <dbl>    <dbl>  <dbl>  <dbl>
## 1 1          3.39   3.44 0.329  2.53    3.35  0.0122  0.0485  0.0160 0.0968
## 2 2          2.78  -1.91 0.607  3.24    2.81  0.0155 -0.0262  0.0258 0.0969
## 3 3          5.33  13.9  0.578  2.63    5.29  0.0275  0.0437  0.0817 0.0968
## 4 4          4.45   3.05 0.708  3.88    4.44  0.0139  0.0104  0.0208 0.0969
## 5 5          5.54   4.40 0.291  4.32    5.65  0.0155 -0.106   0.0260 0.0962
## 6 6          3.92  -4.66 0.169  4.27    3.91  0.0251  0.00456 0.0680 0.0969
```

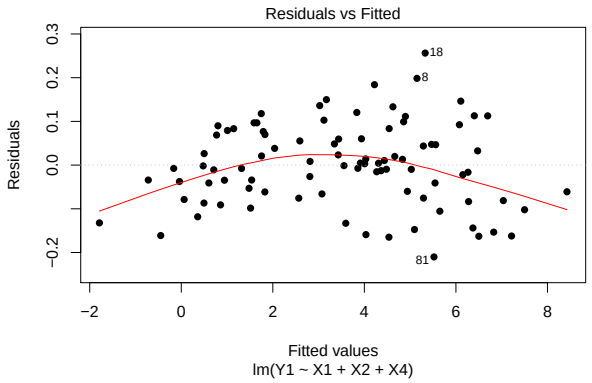
```
## # ... with 2 more variables: .cooksd <dbl>, .std.resid <dbl>
```

Meta Data for model.diag.metrics Among the table columns, there are:

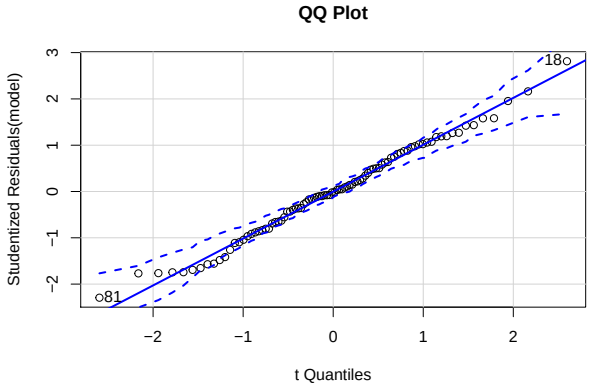
Y1: original values X1, X2: the observed values .fitted: the fitted values .resid: the residual errors

Let's see correlation between the features:

We can see the plot in residual and fitted plot here now:

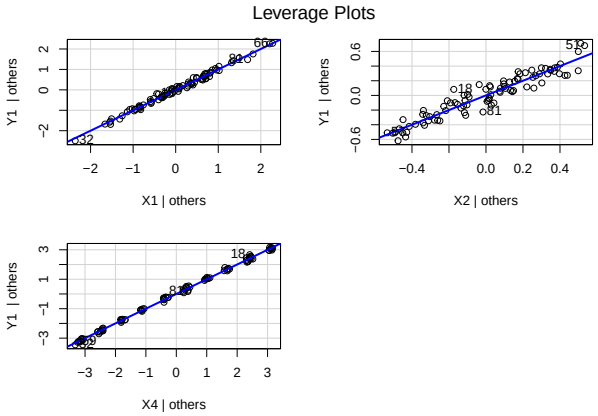


Note how the residuals plot of this last model shows some important points still lying far away from the middle area of the graph. Since the behaviour is random in nature we were successful in this test.




```
## 18 81
## 17 73
```

Check outliers:

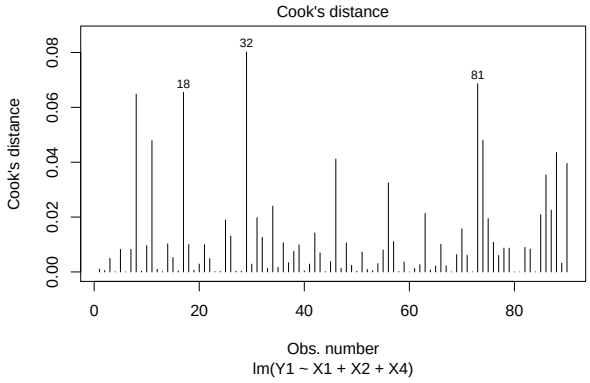


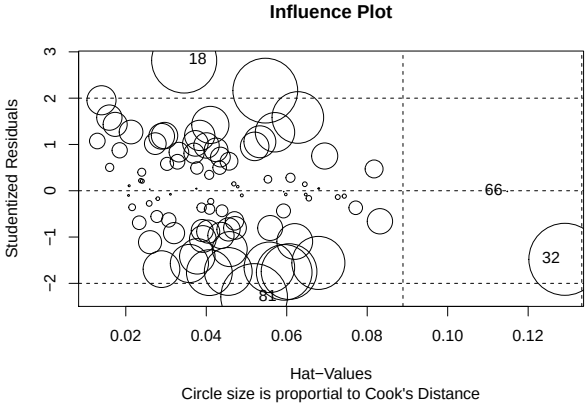
Check if errors are auto corelated

```
##
##  Durbin-Watson test
##
## data:  model
## DW = 1.5493, p-value = 0.01472
## alternative hypothesis: true autocorrelation is greater than 0
```

We can see that there are outliers in this dataset mainly row 5,42,57,32 from X1, and 18,42,57,1 from X2. So, overall number 42 and 57 are outliers.

Influential Variables : Fiding influential data points is required. So let us look at variable plots:






```
##         StudRes        Hat        CookD
## 18  2.81338698 0.03450780 6.546052e-02
## 32 -1.48083549 0.12912835 8.017492e-02
## 66 -0.01405232 0.11483649 6.479936e-06
## 81 -2.29378874 0.05189565 6.859899e-02
```

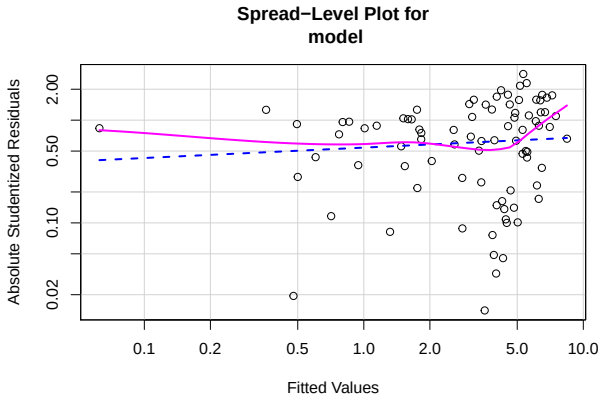
Checking for Multi-collinearity:

```
##       X1       X2       X4
## 1.001778 1.002729 1.001661
```

```
##    X1    X2    X4
## FALSE FALSE FALSE
```

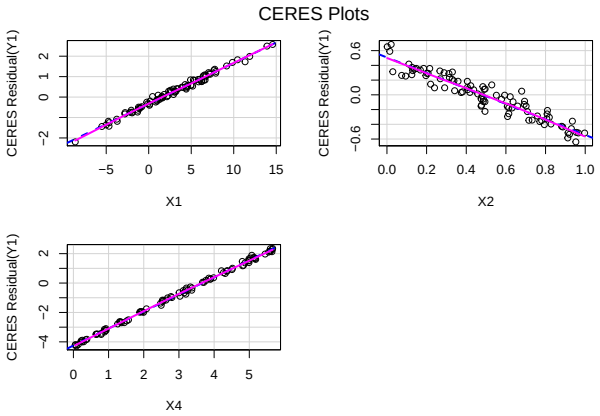
Non-constant Error Variance:

```
## Non-constant Variance Score Test
## Variance formula: ~ fitted.values
## Chisquare = 3.815566, Df = 1, p = 0.050778
```




```
##
## Suggested power transformation:  0.898781
```

Nonlinearity test:



So all the factors are linear which is required.

Non-independence of Errors:

```
##  lag Autocorrelation D-W Statistic p-value
##    1        0.209135      1.549331    0.02
##  Alternative hypothesis: rho != 0
```

3.2 Q.2] Question: Use the data set , to fit a model. Perform regression diagnostics on this model. Display any plots that are relevant.

(a) Check and comment on the constant variance assumption for the errors.
(b) Check and comment on the normality assumption.
(c) Check and comment on the large leverage points.
(d) Check and comment on the outliers.
(e) Check and comment on the influential points.
(f) Check and comment on the structure of the relationship between the predictors and the response.
(g) Compute and comment on the condition numbers.
(h) Compute and comment on the correlations between the predictors.

(i) Compute and comment on the VIF.

3.2.0.1 Linear Model

```
##
## Call:
## lm(formula = Y ~ ., data = train4)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -1.126e-13 -6.087e-15  2.561e-15  7.474e-15  7.213e-14
##
## Coefficients:
##               Estimate Std. Error    t value Pr(>|t|)
## (Intercept)  1.892e-13  1.412e-13  1.340e+00   0.1882
## X1          -1.435e-14  8.831e-15 -1.624e+00   0.1126
## X2          -2.959e-15  2.561e-15 -1.156e+00   0.2551
## X3          -3.057e-15  1.894e-15 -1.614e+00   0.1148
## X4           1.524e-15  3.925e-16  3.882e+00   0.0004 ***
## X5           1.000e+00  4.940e-16  2.024e+15   <2e-16 ***
## X6           1.000e+00  3.962e-16  2.524e+15   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.501e-14 on 38 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 6.277e+31 on 6 and 38 DF,  p-value: < 2.2e-16
```

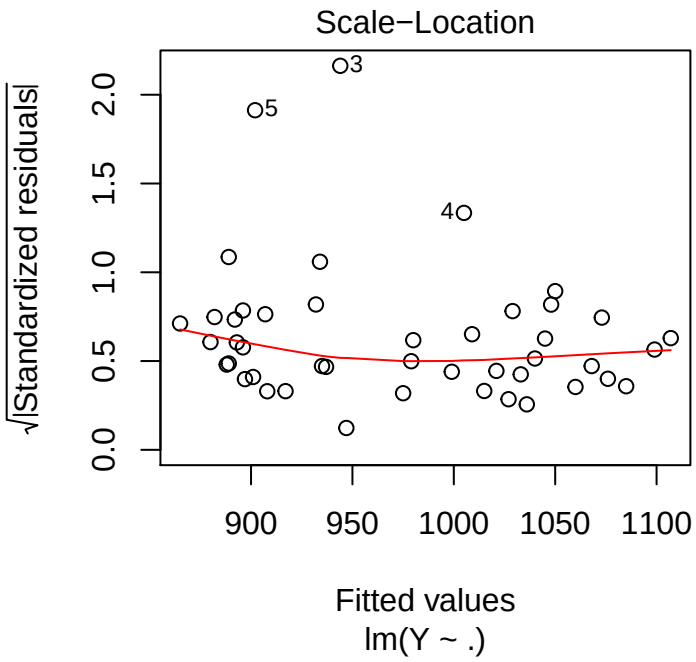
3.2.0.2 DIAGNOSIS

3.2.0.3 Regression assumptions

Linear regression makes several assumptions about the data, such as : i) Linearity of the data. The relationship between the predictor (x) and the outcome (y) is assumed to be linear. ii) Normality of residuals. The residual errors are assumed to be normally distributed. iii) Homogeneity of residuals variance. The residuals are assumed to have a constant variance (homoscedasticity) iv) Independence of residuals error terms.

All these assumptions and potential problems can be checked by producing some diagnostic plots visualizing the residual errors. **Diagnostic plots**

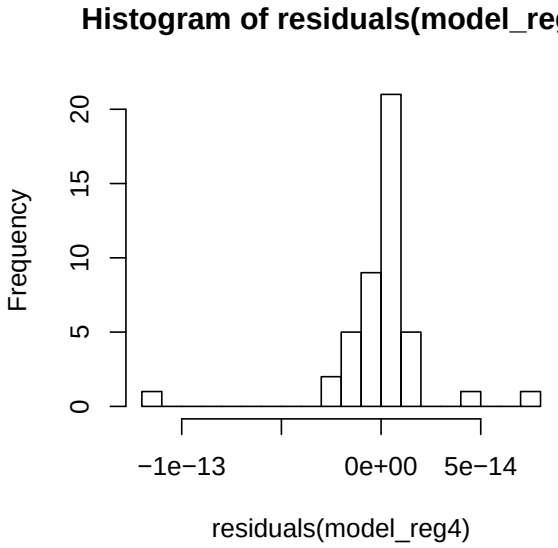
a) Check and comment on the constant variance assumption for the errors.
Scale-Location (or Spread-Location) is used to check the homogeneity of variance of the residuals i.e. they have a constant variance (homoscedasticity).
Horizontal line with equally spread points is a good indication of homoscedasticity.

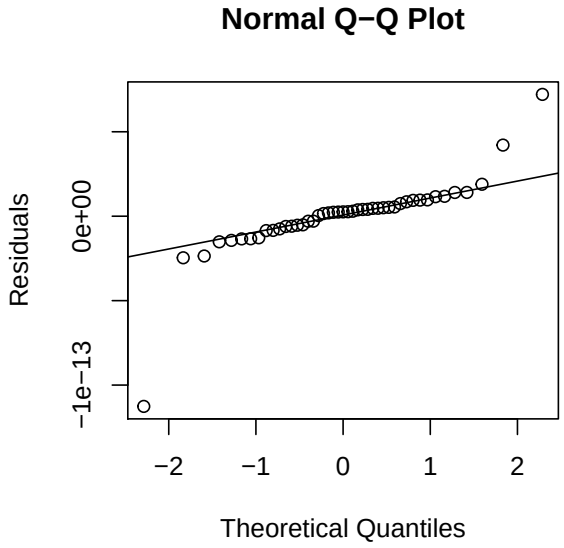


Inference: The variance is always constant. Hence our assumption holds true.

(b) Check and comment on the normality assumption.

Normal Q-Q is used to examine whether the residuals are normally distributed. It's good if residuals points follow the straight dashed line.





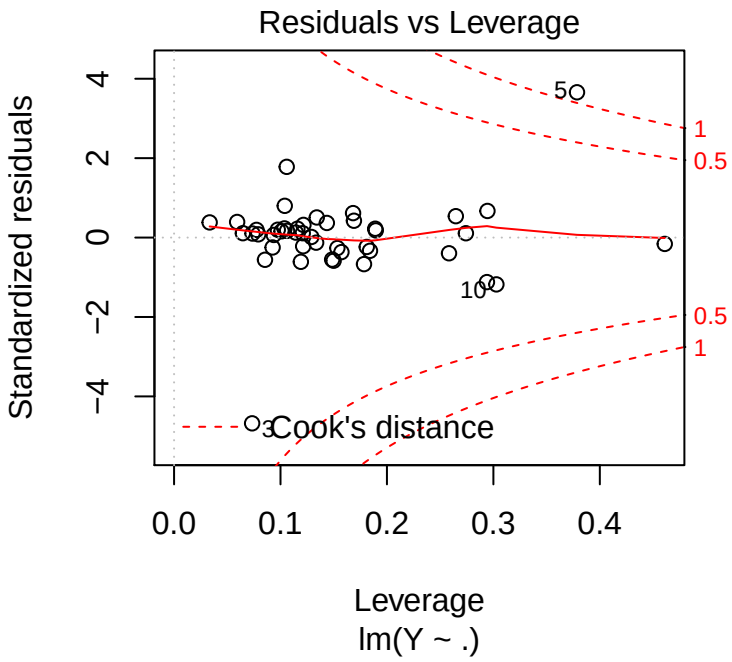
Inference: The residual errors are normally distributed and hence our second assumption also holds true.But, there are three extreme outliers.

.

(c) Check and comment on the large leverage points.

A data point has high leverage, if it has extreme predictor x values. This can be detected by examining the leverage statistic or the hat-value. A value of this statistic above 2(p + 1)/n indicates an observation with high leverage where, p is the number of predictors and n is the number of observations.

The Residuals vs Leverage plot can help us to find influential observations if any.

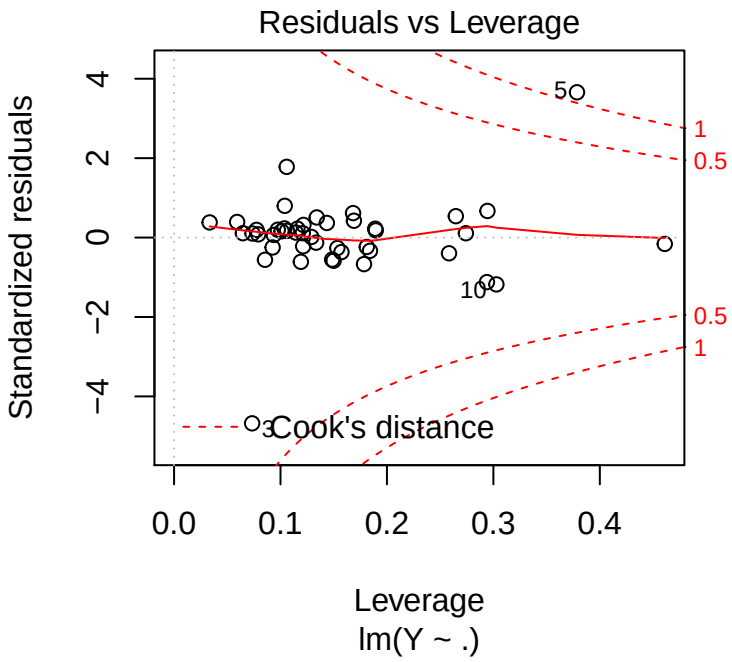


Inference: Corresponding to the leverage statistic i.e. [[2*(p+1)]/n]] 0.31, there are only 2 points having high leverage.

(d) Check and comment on the outliers.

Outliers can be identified by examining the standardized residual (or studentized residual), which is the residual divided by its estimated standard error. Standardized residuals can be interpreted as the number of standard errors away from the regression line.
Observations whose standardized residuals are greater than 3 in absolute value are possible outliers

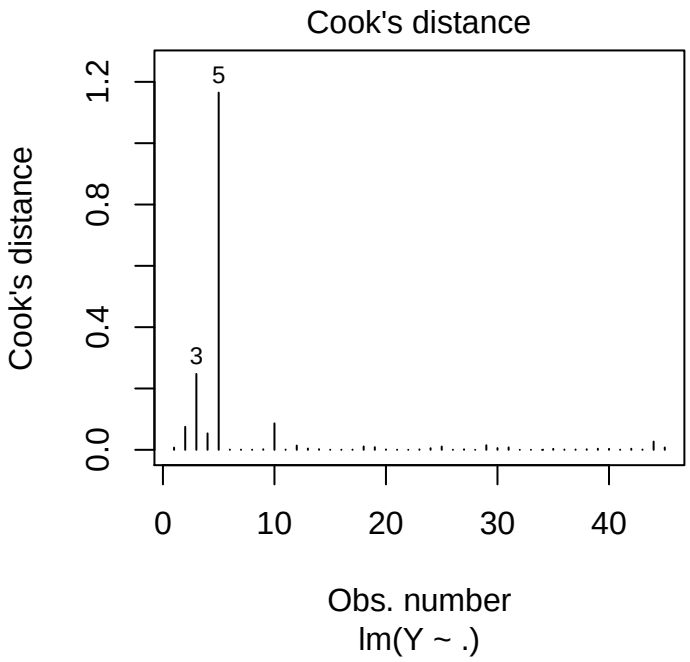
On this plot, outlying values are generally located at the upper right corner or at the lower right corner. Those spots are the places where data points can be influential against a regression line.



Inference: There are approx 2-3 outliers present.

(e) Check and comment on the influential points.
An influential value is a value, which inclusion or exclusion can alter the results of the regression analysis. Such a value is associated with a large residual.

Statisticians have developed a metric called Cook's distance to determine the influence of a value. This metric defines influence as a combination of leverage and residual size.
A rule of thumb is that an observation has high influence if Cook's distance exceeds 4/(n - p - 1), where n is the number of observations and p the number of predictor variables.



Inference:From above plot we can see there are 2 points(#3,#5) exceeding Cook's distance(in our case, 0.10).

(g) Compute and comment on the correlations between the predictors.

```
##        X1     X2     X3     X4     X5     X6
## X1  1.000 -0.340  0.865  0.625 -0.452 -0.399
## X2 -0.340  1.000  0.039 -0.200  0.029  0.070
## X3  0.865  0.039  1.000  0.630 -0.517 -0.445
## X4  0.625 -0.200  0.630  1.000 -0.889 -0.867
## X5 -0.452  0.029 -0.517 -0.889  1.000  0.967
## X6 -0.399  0.070 -0.445 -0.867  0.967  1.000
```

Inference: There are several large pairwise correlations both between predictors and between predictors and the response.

(h) Compute and comment on the condition numbers.
Condition numbers, indicate whether more than just one independent linear combination is to blame.

```
##
## Eigen Values:
## [1] 2.159771e+07 3.769651e+04 2.098266e+03 7.857397e+02 2.602308e+02
## [6] 7.544769e+00 3.135491e-02
##
## Condition Numbers:
## [1]     1.0000    23.9361   101.4550   165.7924   288.0876  1691.9242
```

```
## [7] 26245.2745
```

Inference: There is a wide range in the eigenvalues and several condition numbers are large. This means that problems are being caused by more than just one linear combination.

(i) Compute and comment on the VIF.
VIF Test for removal of multicollinearity.
VIFi>10 indicates serious multicollinearity for the predictor.

```
##
## Call:
## lm(formula = Y ~ ., data = train4)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -1.126e-13 -6.087e-15  2.561e-15  7.474e-15  7.213e-14
##
## Coefficients:
##               Estimate Std. Error    t value Pr(>|t|)
## (Intercept)  1.892e-13  1.412e-13  1.340e+00   0.1882
## X1          -1.435e-14  8.831e-15 -1.624e+00   0.1126
## X2          -2.959e-15  2.561e-15 -1.156e+00   0.2551
## X3          -3.057e-15  1.894e-15 -1.614e+00   0.1148
## X4           1.524e-15  3.925e-16  3.882e+00   0.0004 ***
## X5           1.000e+00  4.940e-16  2.024e+15   <2e-16 ***
## X6           1.000e+00  3.962e-16  2.524e+15   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.501e-14 on 38 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 6.277e+31 on 6 and 38 DF,  p-value: < 2.2e-16
```

```
##        X1        X2        X3        X4        X5        X6
##  8.997990  2.486768  8.768358  7.596735 20.243542 17.181520
```

Re-computing VIF's after removing variable with highest VIF value

```
##       X1       X2       X6
## 1.345153 1.137033 1.195742
```

```
##
## Call:
## lm(formula = Y ~ . - X5 - X4 - X3, data = train4)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -27.1004  -3.7548   0.3675   6.4044  13.2715
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 80.28370   25.19558   3.186  0.00275 **
## X1          -2.83751    1.14607  -2.476  0.01751 *
## X2          -1.06177    0.58114  -1.827  0.07498 .
```

```
## X6           1.80927    0.03508  51.569  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.393 on 41 degrees of freedom
## Multiple R-squared:  0.9877, Adjusted R-squared:  0.9868
## F-statistic:  1101 on 3 and 41 DF,  p-value: < 2.2e-16
```

Inference: we see that the accuracy is mostly unaffected even after removing the correlated variables and reducing the dimension by 3

3.3 Q.3] Question: Use the data to fit a model using the following methods.

(a) Least squares.
(b) Least absolute deviations.
(c) Huber method.
(d) Least trimmed squares. Compare the results. Use diagnostic methods to detect any outliers or influential points. Remove these points and then use least squares. Compare the results.

(a) Least squares.
Least squares is a statistical method used to determine a line of best fit by minimizing the sum of squares created by a mathematical function. A "square" is determined by squaring the distance between a data point and the regression line. The least squares approach limits the distance between a function and the data points that a function is trying to explain. It is used in regression analysis, often in nonlinear regression modeling in which a curve is fit into a set of data.

```
##
## Call:
## lm(formula = Y1 ~ ., data = train5)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.0091 -1.6878 -0.3379  2.6231  5.7106
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -43.1625    13.5862  -3.177 0.006252 **
## X1            0.6981     0.1476   4.731 0.000268 ***
## X2            1.3489     0.4048   3.332 0.004551 **
## X3           -0.1173     0.1741  -0.674 0.510759
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.408 on 15 degrees of freedom
## Multiple R-squared:  0.915,  Adjusted R-squared:  0.898
## F-statistic: 53.81 on 3 and 15 DF,  p-value: 2.919e-08
```

(b) Least absolute deviations.
The method of least absolute deviations fits a line to a set of (x,y) data by choosing slope and intercept parameters to minimize the SAE, or the sum of absolute errors. As with the SSE associated with least squares, the errors in question are the differences between the actual y data values and the corresponding y values defined by the line.

```
##
## Call: rq(formula = Y1 ~ ., data = train5)
##
## tau: [1] 0.5
##
## Coefficients:
##             coefficients lower bd  upper bd
## (Intercept) -39.98645    -72.97779 -30.76573
## X1            0.83469      0.51985   1.17008
## X2            0.56369      0.29209   1.92083
## X3           -0.05691     -0.26029  -0.03097
```

(c) Huber method.

```
##
## Call: rlm(formula = Y1 ~ ., data = train5)
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.5665 -1.9082 -0.4962  1.8642  6.3810
##
## Coefficients:
##             Value    Std. Error t value
## (Intercept) -43.2967  12.3948    -3.4931
## X1            0.8051   0.1346     5.9807
## X2            1.0156   0.3693     2.7496
## X3           -0.1077   0.1588    -0.6785
##
## Residual standard error: 2.833 on 15 degrees of freedom
```

(c) Least trimmed squares. Least trimmed squares (LTS), or least trimmed sum of squares, is a robust statistical method that fits a function to a set of data whilst not being unduly affected by the presence of outliers. It is one of a number of methods for robust regression. Instead of the standard least squares method, which minimises the sum of squared residuals over n points, the LTS method attempts to minimise the sum of squared residuals over a subset k of those points. The unused n-k points do not influence the fit.

```
##
## Call:
## ltsReg.formula(formula = Y1 ~ ., data = train5)
##
## Residuals (from reweighted LS):
##     Min      1Q  Median      3Q     Max
## -2.1813 -0.5008  0.0000  0.5969  1.3970
##
## Coefficients:
##            Estimate Std. Error t value Pr(>|t|)
## Intercept -39.80574    4.85748  -8.195 5.19e-06 ***
## X1          0.78574    0.06893  11.399 1.97e-07 ***
## X2          0.60214    0.16945   3.553  0.00453 **
## X3         -0.04176    0.06277  -0.665  0.51955
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.203 on 11 degrees of freedom
## Multiple R-Squared: 0.9805,  Adjusted R-squared: 0.9752
## F-statistic: 184.3 on 3 and 11 DF,  p-value: 1.103e-09
```

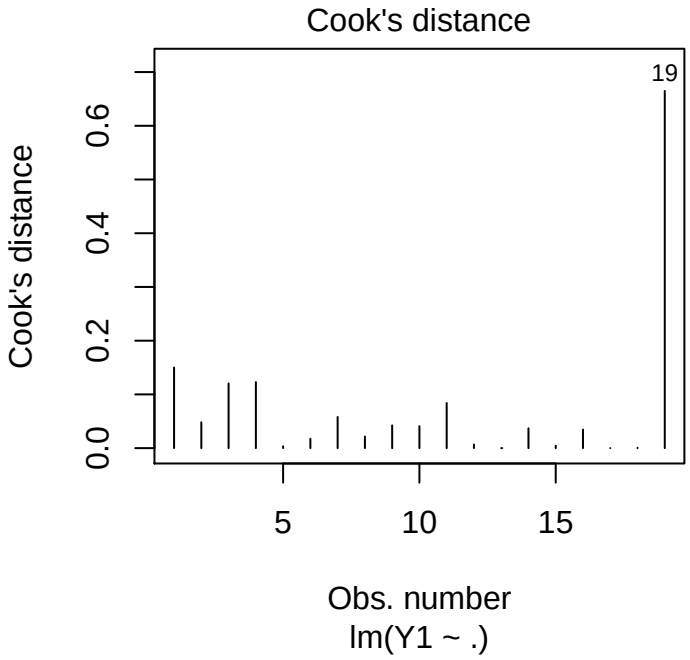
Inference: In all the applied regression methods the variable X3 is consistently insignificant therefore it can be removed. There doesn't seem to be a significant change in the coefficient of other variables when compared across the techniques. So, its better to use least squares method.

DIAGNOSIS TO FIND INFLUENTIAL POINTS AND OUTLIERS

Influential Points

An influential value is a value, which inclusion or exclusion can alter the results of the regression analysis. Such a value is associated with a large residual.

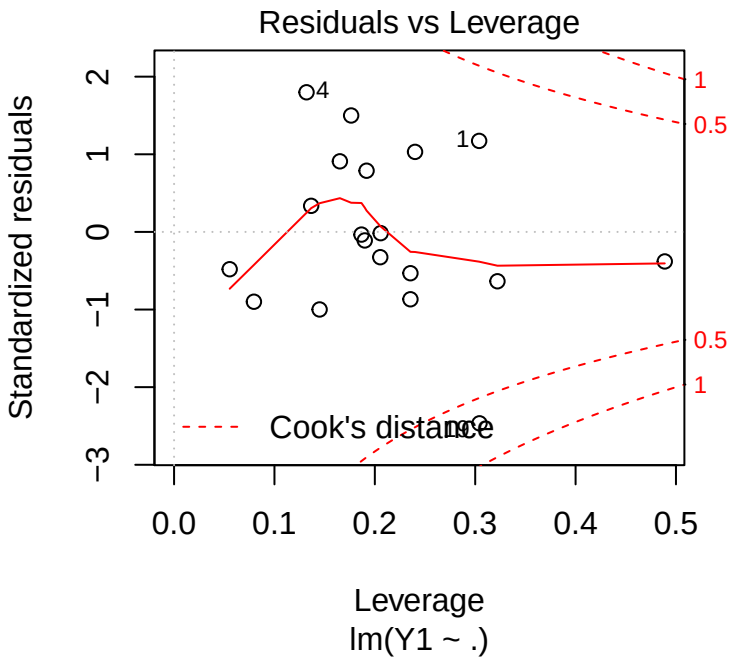
Statisticians have developed a metric called Cook's distance to determine the influence of a value. This metric defines influence as a combination of leverage and residual size.
A rule of thumb is that an observation has high influence if Cook's distance exceeds 4/(n - p - 1), where n is the number of observations and p the number of predictor variables.



Inference: From above plot we can see there is one point(#19) exceeding Cook's distance(in our case, 0.26).

Outliers

Outliers can be identified by examining the standardized residual (or studentized residual), which is the residual divided by its estimated standard error. Standardized residuals can be interpreted as the number of standard errors away from the regression line.
Observations whose standardized residuals are greater than 3 in absolute value are possible outliers.



Inference: Observation #19 and #4 are outliers.

Computing least squares after removal of the outlier

```
##
## Call:
## lm(formula = Y1 ~ ., data = train5[-c(19, 4), ])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.0196 -1.4739  0.1148  1.0726  3.6254
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -43.42368    8.34583  -5.203  0.00017 ***
## X1            0.96210    0.10441   9.214 4.61e-07 ***
## X2            0.54141    0.29313   1.847  0.08763 .
## X3           -0.09883    0.10677  -0.926  0.37151
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.087 on 13 degrees of freedom
## Multiple R-squared:  0.9707, Adjusted R-squared:  0.964
## F-statistic: 143.7 on 3 and 13 DF,  p-value: 3.235e-10
```

Inference: After removal of outliers, the value of R-squared increased from 0.915 to 0.9707and Adjusted R-squared increased from 0.898 to 0.964.
Also, the significance of predictor X2 reduced.

3.4 Q.4] Question: Use the data to fit a model with Y as the response and only X3, X4, and X5 as predictors. Use the Box-Cox method to determine the best transformation on the response.

```
##
## Call:
## lm(formula = Y ~ X3 + X4 + X5, data = train7)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.8044  -2.9246  -0.1525   2.7149  13.2581
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.076e+01  1.646e+00  -6.537 2.79e-10 ***
## X3           8.755e-02  1.388e-02   6.309 1.03e-09 ***
## X4           3.263e-01  2.141e-02  15.243  < 2e-16 ***
## X5          -1.026e-03  1.669e-04  -6.148 2.56e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.436 on 293 degrees of freedom
## Multiple R-squared:  0.6983, Adjusted R-squared:  0.6952
## F-statistic:   226 on 3 and 293 DF,  p-value: < 2.2e-16
```

Normality test

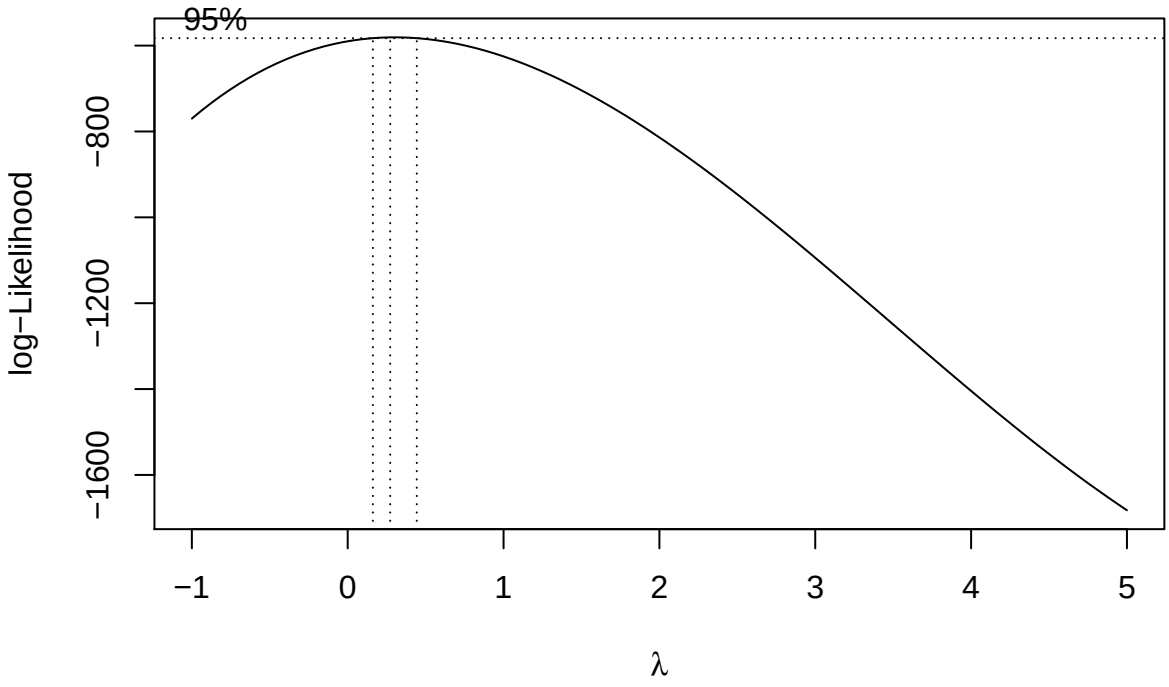
Shapiro-Wilk's method is widely recommended for normality test and it provides better power than K-S. It is based on the correlation between the data and the corresponding normal scores.

```
##
##  Shapiro-Wilk normality test
##
## data:  train7$Y
## W = 0.91153, p-value = 3.24e-12
```

Inference: Since the p-value is less than 0.05, the variable is not normally distibuted and we need to power transformation on it.

Box-Cox Method: Determining the best transformation on the response

The Box-Cox method is a popular way to determine a transformation on the response. It is designed for strictly positive responses and chooses the transformation to find the best fit to the data.
Some general considerations concerning the BoxCox method are:
1. The Box-Cox method gets upset by outliers if you find lambda=5, then this is probably the reason there can be little justification for actually making such an extreme transformation.
2. If some yi <0, we can add a constant to all the y. This can work provided the constant is small, but this is an inelegant solution.
3. If max i y i /min i y i is small, then the Box-Cox will not have much real effect because power transforms are well approximated by linear transformations over short intervals far from the origin.




```
## [1] 0.2727273
```

Inference: Since value of lambda is reaching 5, we have to perform extreme transformation with lambda=0.27.

```
##
##  Shapiro-Wilk normality test
##
## data:  powerTransform(train7$Y, lambda)
## W = 0.96987, p-value = 7.081e-06
```

Inference: After doing power transform, the shapiro test significantly increases the p-value, and the optimal valyue of lambda for this p-value is 0.27

3.5 Q.5] Question: Use the data to fit a linear model. Implement the following variable selection

methods to determine the "best" model.
(a) Backward Elimination.
(b) AIC, AICC, BIC.
(c) R2, R2a
(d) Mallows Cp.

(a) Backward Elimination.
It starts with all predictors in the model (full model), iteratively removes the least contributive predictors,

and stops when you have a model where all predictors are statistically significant.

```
## Start:  AIC=-45.86
## Y ~ X1 + X2 + X3 + X4 + X5 + X6 + X7 + X8
##
##        Df Sum of Sq    RSS     AIC
## - X7    1    0.0526 42.643 -47.753
## - X8    1    0.2481 42.839 -47.350
## - X6    1    0.3429 42.934 -47.156
## <none>               42.591 -45.861
## - X4    1    1.1332 43.724 -45.551
## - X3    1    1.4618 44.053 -44.892
## - X2    1    3.4329 46.024 -41.040
## - X5    1    3.6873 46.278 -40.555
## - X1    1   21.2914 63.882 -12.186
##
## Step:  AIC=-47.75
## Y ~ X1 + X2 + X3 + X4 + X5 + X6 + X8
##
##        Df Sum of Sq    RSS     AIC
## - X6    1    0.3311 42.975 -49.072
## - X8    1    0.6587 43.302 -48.404
## <none>               42.643 -47.753
## - X4    1    1.1321 43.776 -47.447
## - X3    1    1.4124 44.056 -46.885
## - X2    1    3.3828 46.026 -43.035
## - X5    1    3.6347 46.278 -42.555
## - X1    1   22.4912 65.135 -12.477
##
## Step:  AIC=-49.07
## Y ~ X1 + X2 + X3 + X4 + X5 + X8
##
##        Df Sum of Sq    RSS     AIC
## - X8    1    0.3634 43.338 -50.331
## <none>               42.975 -49.072
## - X4    1    1.0741 44.049 -48.900
## - X3    1    1.2275 44.202 -48.594
## - X2    1    3.3314 46.306 -44.502
## - X5    1    3.3944 46.369 -44.382
## - X1    1   24.6196 67.594 -11.216
##
## Step:  AIC=-50.33
## Y ~ X1 + X2 + X3 + X4 + X5
##
##        Df Sum of Sq    RSS     AIC
## <none>               43.338 -50.331
## - X3    1    1.0003 44.338 -50.323
## - X4    1    1.2194 44.557 -49.889
## - X2    1    3.0812 46.419 -46.287
## - X5    1    4.4404 47.778 -43.747
## - X1    1   27.2323 70.570  -9.424
```

Best Model suggested by Backward elimination : lm(Y~X1+X2+X3+X4+X5)

Predicting the test data

```
##        RMSE  Rsquare
## 1 0.5063763 0.756056
```

(b.1) AIC

AIC stands for (Akaike's Information Criteria). The basic idea of AIC is to penalize the inclusion of additional variables to a model. It adds a penalty that increases the error when including additional terms. The lower the AIC, the better the model.

```
## Start:  AIC=30.01
## Y ~ 1
##
##        Df Sum of Sq     RSS     AIC
## + X1    1    66.875  54.103 -38.807
## + X5    1    39.003  81.976  -2.240
## + X6    1    37.791  83.188  -0.949
## + X8    1    18.947 102.032  17.019
## + X7    1    14.974 106.005  20.381
## + X2    1    14.819 106.160  20.509
## + X4    1     3.702 117.276  29.273
## + X3    1     2.995 117.984  29.803
## <none>              120.979  30.008
##
## Step:  AIC=-38.81
## Y ~ X1
##
##        Df Sum of Sq     RSS     AIC
## + X2    1     5.057  49.047 -45.442
## + X5    1     4.006  50.097 -43.578
## + X4    1     2.011  52.093 -40.140
## <none>               54.103 -38.807
## + X6    1     0.826  53.277 -38.161
## + X8    1     0.788  53.315 -38.098
## + X7    1     0.094  54.009 -36.960
## + X3    1     0.056  54.047 -36.899
## - X1    1    66.875 120.979  30.008
##
## Step:  AIC=-45.44
## Y ~ X1 + X2
##
##        Df Sum of Sq     RSS     AIC
## + X5    1     3.942  45.104 -50.815
## <none>               49.047 -45.442
## + X8    1     1.000  48.046 -45.255
## + X6    1     0.891  48.155 -45.055
## + X3    1     0.693  48.354 -44.694
## + X7    1     0.353  48.694 -44.077
## + X4    1     0.264  48.783 -43.916
```

```
## - X2    1     5.057  54.103 -38.807
## - X1    1    57.113 106.160  20.509
##
## Step:  AIC=-50.82
## Y ~ X1 + X2 + X5
##
##        Df Sum of Sq    RSS     AIC
## <none>              45.104 -50.815
## + X4    1    0.7662 44.338 -50.323
## + X3    1    0.5471 44.557 -49.889
## + X8    1    0.2552 44.849 -49.314
## + X7    1    0.1669 44.938 -49.141
## + X6    1    0.0056 45.099 -48.826
## - X5    1    3.9421 49.047 -45.442
## - X2    1    4.9922 50.097 -43.578
## - X1    1   27.0771 72.182 -11.437
```

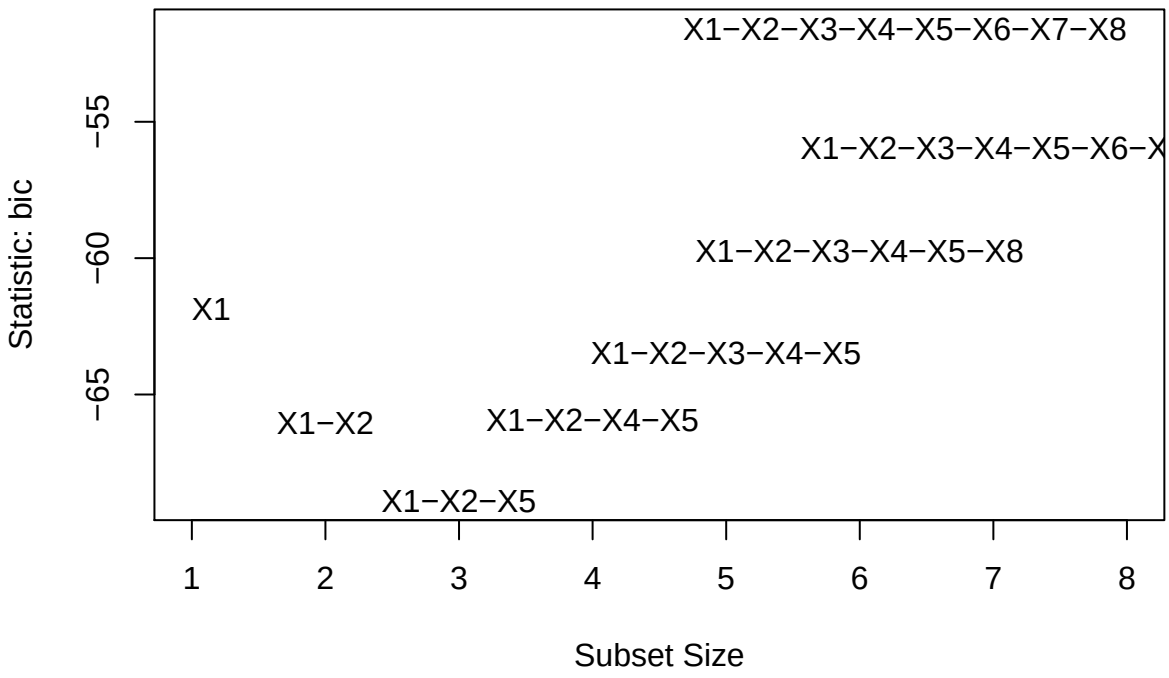
Best Model suggested by AIC : lm(Y~X1+X2+X5)

Predicting the test data

```
##        RMSE   Rsquare
## 1 0.5521792 0.6483517
```

(b.2)BIC

Bayesian Information Criterion is a variant of AIC with a stronger penalty for including additional variables to the model.
The lower the BIC, the better the model.




```
##    Abbreviation
## X1           X1
## X2           X2
## X3           X3
## X4           X4
## X5           X5
## X6           X6
## X7           X7
## X8           X8
```

Best Model suggested by BIC : lm(Y~X1+X2+X5)

Predicting the test data

```
##        RMSE   Rsquare
## 1 0.5521792 0.6483517
```

(b.3)AICc

```
# library(glmulti)
# aicc_model=glmulti(Y~.,data=train8,crit="aicc",level=1,confsetsize=50,fitfunction=lm,plotty = F, repo
# summary(aicc_model@objects[[1]])
## gmulti does not work since it required rJava. and rJava requires to have the system in 32 bit
```

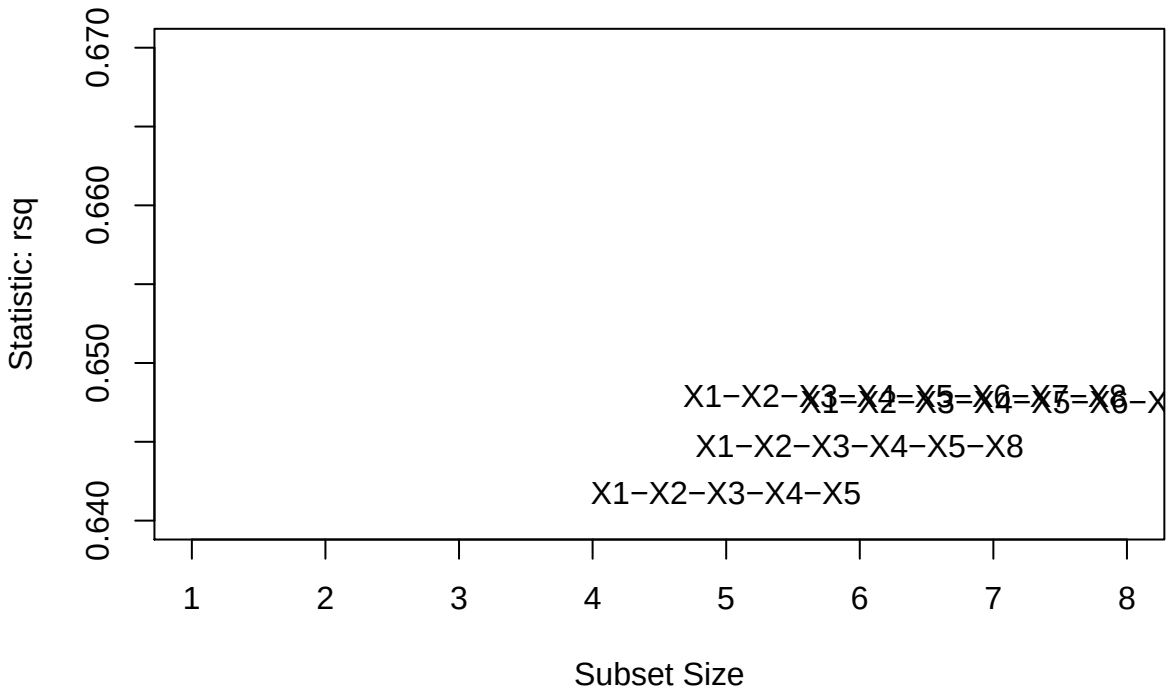
Best Model suggested by AICc : lm(Y~X1+X2+X5)

Predicting the test data

```
##        RMSE   Rsquare
## 1 0.5521792 0.6483517
```

(d.1)R2

It is the square of the sample correlation coefficient between the observed outcomes and the observed predictor values. Ranges from 0 to 1. Higher the R2, better the model.



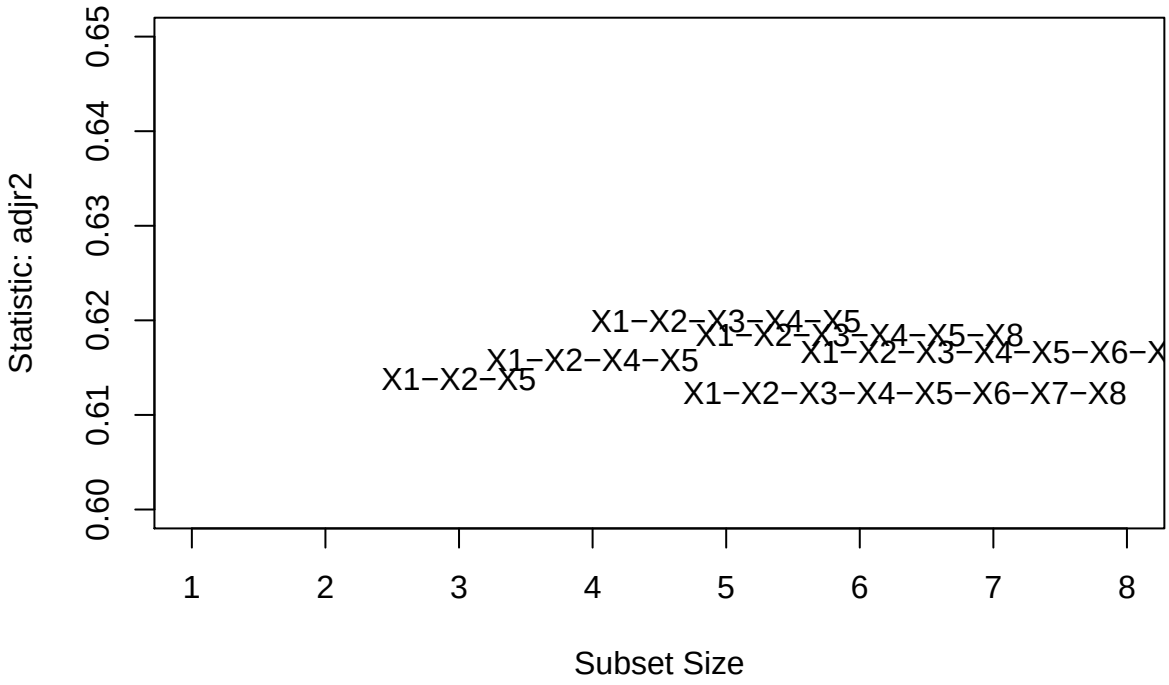
Best Model suggested by R2 : lm(Y~X1+X2+X3+X4+X5+X6+X8)

Predicting the test data

```
##        RMSE   Rsquare
## 1 0.4380371 0.8502812
```

(d.2)R2a

The adjusted R-squared compares the descriptive power of regression models that include diverse numbers of predictors. Every predictor added to a model increases R-squared and never decreases it. Thus, a model with more terms may seem to have a better fit just for the fact that it has more terms, while the adjusted R-squared compensates for the addition of variables and only increases if the new term enhances the model above what would be obtained by probability and decreases when a predictor enhances the model less than what is predicted by chance.




```
##    Abbreviation
## X1           X1
## X2           X2
## X3           X3
## X4           X4
## X5           X5
## X6           X6
## X7           X7
## X8           X8
```

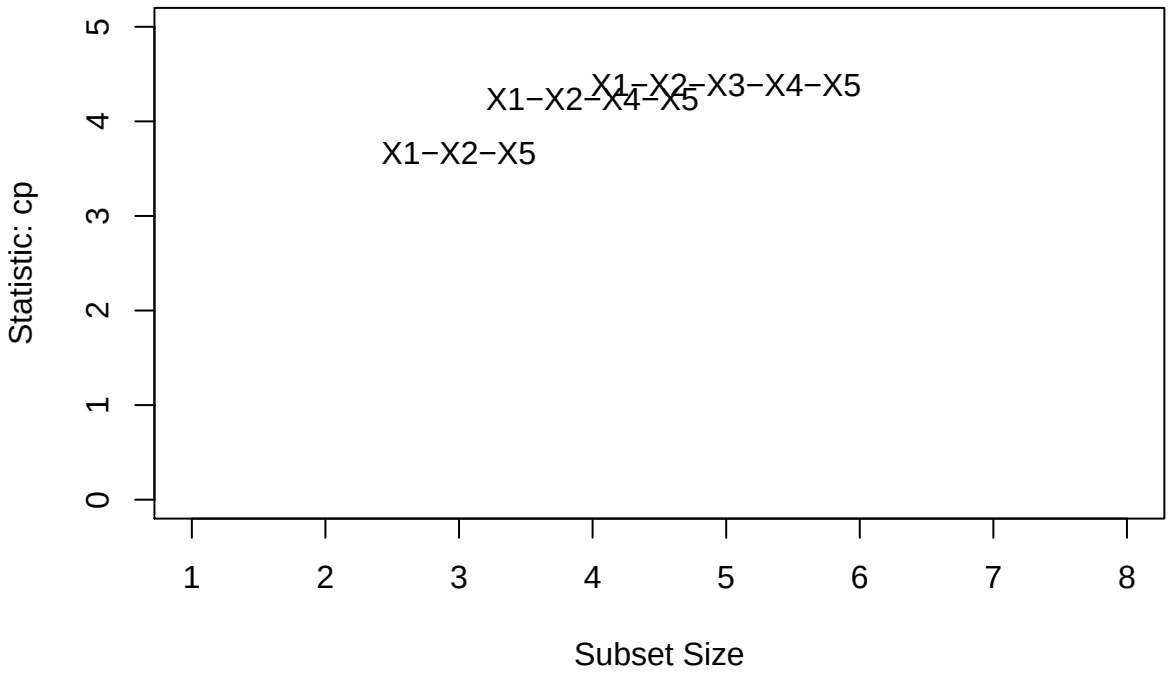
Best Model suggested by R2 : lm(Y~X1+X2+X3+X4+X5)

Predicting the test data

```
##        RMSE  Rsquare
## 1 0.5063763 0.756056
```

(d)Mallow's Cp

A small Mallows' Cp value indicates that the model is relatively precise (has small variance) in estimating the true regression coefficients and predicting future responses.




```
##    Abbreviation
## X1           X1
## X2           X2
## X3           X3
## X4           X4
## X5           X5
## X6           X6
## X7           X7
## X8           X8
```

Best Model suggested by Mallow's Cp : lm(Y~X1+X2+X5)

Predicting the test data

```
##        RMSE   Rsquare
## 1 0.5521792 0.6483517
```

Final Inferences
a) Highest accuracy is achieved by the variables selected in R2 model though it contains the maximum number of variables
b) The variables selected by R2a and backward elimination rank second in terms of accuracy but have a reduced number of variables.
c) AIC, BIC, AICc and Mallow's Cp yielded the same subset variables and have the least accuracy and number of variables..

3.6 Q.6] Question: Consider the data RegD9.txt remove every tenth observation from the data for use as a test sample. Use the remaining data as a training sample building the following models.

(a) Linear regression with all predictors.
(b) Linear regression with variables selected using AIC.
(c) Principle component regression.
(d) Partial least squares.
(e) Ridge regression. Use the models you find to predict the response in the test sample. Make a report on the performance of the models.

(a) Linear regression with all predictors

```
##
## Call:
## lm(formula = Y ~ ., data = train9)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.4129 -0.3986 -0.1291  0.2184 14.9851
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.457e+02  1.180e+01  37.769   <2e-16 ***
## X1          -4.079e+02  9.124e+00 -44.704   <2e-16 ***
## X2           1.464e-02  1.062e-02   1.378    0.170
## X3           9.085e-03  1.733e-02   0.524    0.601
## X4          -2.627e-03  3.047e-02  -0.086    0.931
## X5          -2.656e-02  7.647e-02  -0.347    0.729
## X6           2.149e-02  3.203e-02   0.671    0.503
## X7           2.692e-02  3.622e-02   0.743    0.458
## X8           2.484e-02  4.763e-02   0.521    0.603
## X9          -1.489e-02  4.677e-02  -0.318    0.750
## X10         -3.238e-03  7.892e-02  -0.041    0.967
## X11         -8.871e-02  6.991e-02  -1.269    0.206
## X12         -5.824e-02  5.574e-02  -1.045    0.297
## X13          4.173e-02  6.384e-02   0.654    0.514
## X14         -2.830e-02  1.776e-01  -0.159    0.874
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.339 on 212 degrees of freedom
## Multiple R-squared:  0.9769, Adjusted R-squared:  0.9754
## F-statistic: 640.1 on 14 and 212 DF,  p-value: < 2.2e-16
```

Inference: Value of for the linear model using predictor variables from Linear regression with all predictors: R-squared=0.9769, Adjusted R-squared= 0.9754

(b) Linear regression with variables selected using AIC.

```
## Start:  AIC=974.3
## Y ~ 1
##
##        Df Sum of Sq     RSS    AIC
## + X1    1   16026.4   425.4 146.60
## + X7    1   10938.9  5512.9 728.11
## + X6    1    7943.7  8508.1 826.61
## + X8    1    6696.4  9755.4 857.66
## + X3    1    6329.4 10122.4 866.05
## + X9    1    5393.3 11058.5 886.12
## + X10   1    4491.4 11960.4 903.92
## + X5    1    4079.0 12372.8 911.62
## + X12   1    4047.8 12404.0 912.19
## + X13   1    2076.8 14375.0 945.66
## + X14   1    1953.6 14498.2 947.60
## + X2    1    1374.3 15077.5 956.49
## + X11   1    1190.6 15261.2 959.24
## <none>              16451.8 974.30
## + X4    1      87.0 16364.8 975.09
##
## Step:  AIC=146.6
## Y ~ X1
##
##        Df Sum of Sq     RSS    AIC
## + X7    1      32.0   393.4 130.83
## + X6    1      27.8   397.7 133.27
## + X3    1      20.3   405.1 137.48
## + X8    1      18.0   407.4 138.76
## + X5    1      14.1   411.4 140.97
## + X14   1      12.9   412.5 141.59
## + X10   1      10.9   414.5 142.69
## + X2    1       6.0   419.5 145.40
## + X9    1       5.5   419.9 145.64
## + X13   1       4.3   421.1 146.27
## + X12   1       3.9   421.5 146.49
## <none>              425.4 146.60
## + X4    1       1.3   424.2 147.92
## + X11   1       0.4   425.0 148.36
## - X1    1   16026.4 16451.8 974.30
##
## Step:  AIC=130.83
## Y ~ X1 + X7
##
##        Df Sum of Sq    RSS    AIC
## + X2    1       5.4  388.0 129.69
## + X11   1       3.6  389.8 130.72
## <none>              393.4 130.83
## + X9    1       3.0  390.4 131.10
## + X12   1       2.2  391.2 131.57
```

```
## + X6    1       0.7  392.7 132.40
## + X3    1       0.6  392.8 132.48
## + X8    1       0.5  392.9 132.56
## + X10   1       0.3  393.1 132.66
## + X5    1       0.2  393.2 132.71
## + X4    1       0.2  393.2 132.73
## + X13   1       0.0  393.4 132.82
## + X14   1       0.0  393.4 132.83
## - X7    1      32.0  425.4 146.60
## - X1    1    5119.5 5512.9 728.11
##
## Step:  AIC=129.69
## Y ~ X1 + X7 + X2
##
##        Df Sum of Sq    RSS    AIC
## <none>               388.0 129.69
## + X11   1       2.2  385.8 130.40
## - X2    1       5.4  393.4 130.83
## + X12   1       0.9  387.1 131.16
## + X6    1       0.9  387.2 131.19
## + X14   1       0.3  387.7 131.52
## + X8    1       0.3  387.7 131.54
## + X9    1       0.2  387.8 131.56
## + X5    1       0.2  387.8 131.59
## + X13   1       0.1  387.9 131.61
## + X3    1       0.1  387.9 131.64
## + X10   1       0.0  388.0 131.68
## + X4    1       0.0  388.0 131.69
## - X7    1      31.5  419.5 145.40
## - X1    1    4949.2 5337.2 722.75
##
## Call:
## lm(formula = Y ~ X1 + X7 + X2, data = train9)
##
## Coefficients:
## (Intercept)           X1           X7           X2
##   441.69179   -405.80610      0.05691      0.01273



##
## Call:
## lm(formula = Y ~ X1 + X2 + X7, data = train9)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.3202 -0.3790 -0.0913  0.2408 15.4098
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.417e+02  9.085e+00  48.616  < 2e-16 ***
## X1          -4.058e+02  7.609e+00 -53.333  < 2e-16 ***
## X2           1.273e-02  7.226e-03   1.762   0.0795 .
```

```
## X7           5.691e-02  1.338e-02   4.253  3.1e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.319 on 223 degrees of freedom
## Multiple R-squared:  0.9764, Adjusted R-squared:  0.9761
## F-statistic:  3077 on 3 and 223 DF,  p-value: < 2.2e-16
```

Inference: Value of for the linear model using predictor variables from AIC: R-squared=0.9764, Adjusted R-squared= 0.9761

(c) Principle component regression.

```
## Principal Components Analysis
## Call: principal(r = data_regd9, nfactors = 3, residuals = FALSE, rotate = "none",
##     scores = TRUE)
## Standardized loadings (pattern matrix) based upon correlation matrix
##       PC1   PC2   PC3   h2    u2 com
## X1  -0.70  0.60  0.16 0.88 0.121 2.1
## Y    0.72 -0.60 -0.14 0.89 0.109 2.0
## X2   0.08 -0.63  0.72 0.93 0.073 2.0
## X3   0.97  0.11 -0.03 0.95 0.047 1.0
## X4   0.23  0.61  0.34 0.55 0.452 1.9
## X5   0.85  0.10  0.20 0.77 0.227 1.1
## X6   0.91 -0.15  0.05 0.86 0.139 1.1
## X7   0.92 -0.28 -0.01 0.93 0.071 1.2
## X8   0.93  0.04 -0.18 0.89 0.110 1.1
## X9   0.87  0.13 -0.33 0.89 0.113 1.3
## X10  0.86  0.16  0.00 0.77 0.235 1.1
## X11  0.62  0.35  0.04 0.51 0.490 1.6
## X12  0.84  0.15 -0.05 0.72 0.276 1.1
## X13  0.69  0.25  0.04 0.54 0.464 1.3
## X14  0.76  0.21  0.45 0.82 0.177 1.8
##
##                        PC1  PC2  PC3
## SS loadings           8.92 1.90 1.07
## Proportion Var        0.59 0.13 0.07
## Cumulative Var        0.59 0.72 0.79
## Proportion Explained  0.75 0.16 0.09
## Cumulative Proportion 0.75 0.91 1.00
##
## Mean item complexity =  1.4
## Test of the hypothesis that 3 components are sufficient.
##
## The root mean square of the residuals (RMSR) is  0.05
##  with the empirical chi square  119.76  with prob <  2.1e-05
##
## Fit based upon off diagonal values = 0.99
```

Inference: After looking at different eigen values we come to conclusion that we require 3 PC's. After looking at the correlation matrix, we conclude the PCs are X3(0.97), X1(0.61), X2(0.72)

```
##
## Call:
## lm(formula = Y ~ X1 + X2 + X3, data = train9)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.4579 -0.3992 -0.0748  0.2487 15.5531
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.556e+02  6.938e+00  65.671  < 2e-16 ***
## X1          -4.167e+02  6.043e+00 -68.963  < 2e-16 ***
## X2           1.930e-02  7.434e-03   2.596 0.010047 *
## X3           1.454e-02  3.766e-03   3.859 0.000149 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.328 on 223 degrees of freedom
## Multiple R-squared:  0.9761, Adjusted R-squared:  0.9758
## F-statistic:  3036 on 3 and 223 DF,  p-value: < 2.2e-16
```

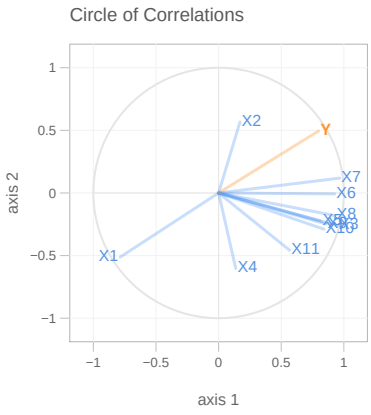
*Inference: Value of for the linear model using predictor variables from PCA: R-squared=0.9761, Adjusted R-squared= 0.9758 *

(c) Partial Least Squares

R2 value for each of our components

```
##         t1         t2         t3
## 0.63692869 0.24501181 0.08635417
```

Now let's look at what is highly correlated with Y with a plot



From the plot we see that X1 with negative impact, X6 and x7 with positive impact make better correlation with Y so we will make our model accoring to these predictors

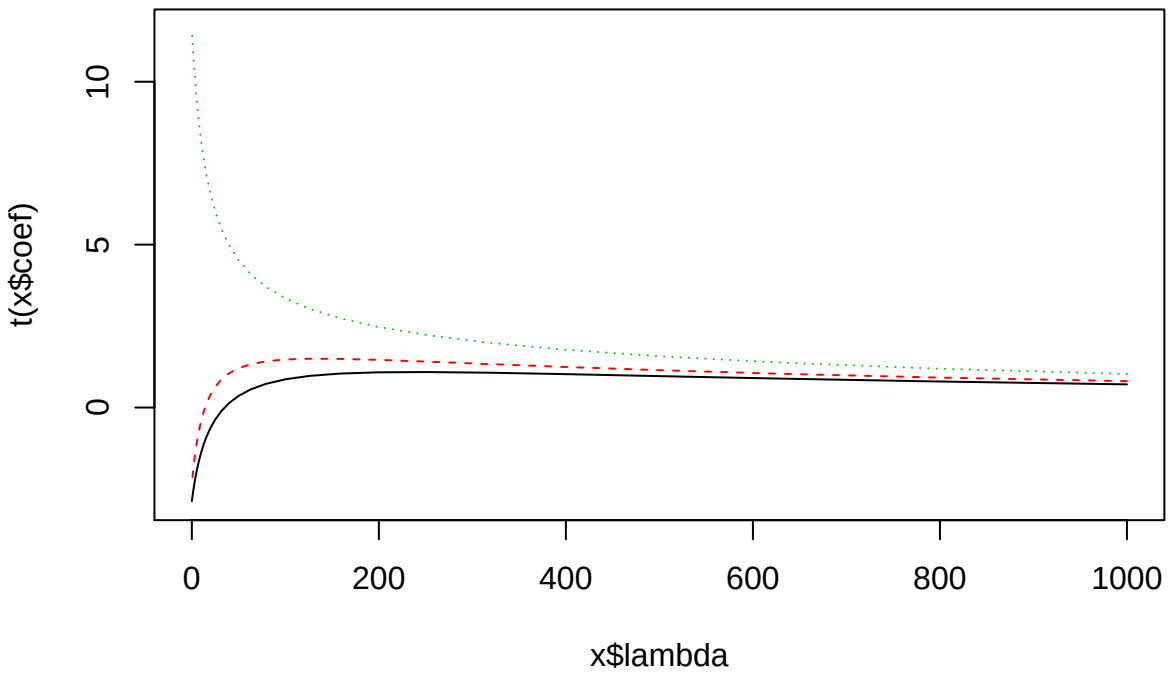
```
##
## Call:
## lm(formula = Y ~ X1 + X6 + X7, data = train9)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.3460 -0.3720 -0.0892  0.2233 15.5363
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  445.27474    9.11326  48.860   <2e-16 ***
## X1          -409.10651    7.80418 -52.421   <2e-16 ***
## X6             0.01723    0.02654   0.649   0.5167
## X7             0.04322    0.02564   1.685   0.0933 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.327 on 223 degrees of freedom
## Multiple R-squared:  0.9761, Adjusted R-squared:  0.9758
## F-statistic:  3040 on 3 and 223 DF,  p-value: < 2.2e-16
```

*Inference: Value of for the linear model using predictor variables from PCA: R-squared=0.9761, Adjusted R-squared= 0.9758 *

(c) Ridge regression Ridge attempts to minimize residual sum of squares of predictors in a given model. However, ridge regression includes an additional 'shrinkage' term - the square of the coefficient estimate - which shrinks the estimate of the coefficients towards zero. The impact of this term is controlled by another term, lambda (determined seperately).

```
##        Length Class  Mode
## coef   153    -none- numeric
## scales   3    -none- numeric
## Inter    1    -none- numeric
## lambda  51    -none- numeric
## ym       1    -none- numeric
## xm       3    -none- numeric
## GCV     51    -none- numeric
## kHKB     1    -none- numeric
## kLW      1    -none- numeric
```


t(x$coef)
10
5
0
0
200
400
600
800
1000
x$lambda

4 Assignment C

Loading of data Data files with n1 = 10^6, 10^7 and 5x10^7 are generated and their files are also created. But for n1 = 10^8, the memory reached limit and the PC stopped responding.

```
# library(data.table)
# library(biglm)
# library(biganalytics)
# n1 <- 1000000
# x1 <- 1:n1
# x2 <-runif(n1,5,95)
# x3 <- rbinom(n1,1,.4)
# x4 <- rnorm(n1, mean=-30, sd=200)
# x5 <- runif(n1,-5000,5000)
# b0 <- 17; b1 <- -0.466; b2 <- 0.037; b3 <- -5.2; b4 <- 2; b5 <- 0.00876
# sigma <- 1.4
# epsilon <- rnorm(x1,0,sigma)
# y <- b0 + b1*x1  + b2*x2  + b3*x3 + b4*x4 + b5*x5 + x1*x2 + epsilon
# fwrite(data.frame(cbind(y,x1,x2,x3,x4,x5)), file = "data14a.txt", sep = " ", quote=T)
```

```
# n1 <- 10000000
# x1 <- 1:n1
# x2 <-runif(n1,5,95)
# x3 <- rbinom(n1,1,.4)
# x4 <- rnorm(n1, mean=-30, sd=200)
# x5 <- runif(n1,-5000,5000)
# b0 <- 17; b1 <- -0.466; b2 <- 0.037; b3 <- -5.2; b4 <- 2; b5 <- 0.00876
# sigma <- 1.4
# epsilon <- rnorm(x1,0,sigma)
# y <- b0 + b1*x1  + b2*x2  + b3*x3 + b4*x4 + b5*x5 + x1*x2 + epsilon
# fwrite(data.frame(cbind(y,x1,x2,x3,x4,x5)), file = "data14b.txt", sep = " ", quote=T)
```

```
# n1 <- 50000000
# x1 <- 1:n1
# x2 <-runif(n1,5,95)
# x3 <- rbinom(n1,1,.4)
# x4 <- rnorm(n1, mean=-30, sd=200)
# x5 <- runif(n1,-5000,5000)
# b0 <- 17; b1 <- -0.466; b2 <- 0.037; b3 <- -5.2; b4 <- 2; b5 <- 0.00876
# sigma <- 1.4
# epsilon <- rnorm(x1,0,sigma)
# y <- b0 + b1*x1  + b2*x2  + b3*x3 + b4*x4 + b5*x5 + x1*x2 + epsilon
# fwrite(data.frame(cbind(y,x1,x2,x3,x4,x5)), file = "data14c.txt", sep = " ", quote=T)
```

```
# library(data.table)
# data1 = fread("data14a.txt")
# data2 = fread("data14b.txt")
# data3 = fread("data14c.txt")
```

lm command

```
# gc(reset = TRUE)
# start.time<-proc.time()
# lm1 = lm(y~., data1)
# summary(lm1)
# end.time<-proc.time()
```

```
# save.time<-end.time-start.time
# cat("\nNumber of minutes running: ", save.time[3]/60, "\n\n")
# gc()
```

```
# gc(reset = TRUE)
# start.time<-proc.time()
# summary(lm(y~., data2))
# end.time<-proc.time()
# save.time<-end.time-start.time
# cat("\nNumber of minutes running: ", save.time[3]/60, "\n\n")
# gc()
```

- lm command worked for n1 = 10^6 and 10^7 data files, but the memory allocation reached limit for further data files.

biglm command

```
# library(biglm)
# gc(reset = TRUE)
# start.time<-proc.time()
# summary(biglm(y~x1+x2+x3+x4+x5, data1))
# end.time<-proc.time()
# save.time<-end.time-start.time
# cat("\nNumber of minutes running: ", save.time[3]/60, "\n\n")
# gc()
```

```
# gc(reset = TRUE)
# start.time<-proc.time()
# summary(biglm(y~x1+x2+x3+x4+x5, data2))
# end.time<-proc.time()
# save.time<-end.time-start.time
# cat("\nNumber of minutes running: ", save.time[3]/60, "\n\n")
# gc()
```

- biglm command worked for n1 = 10^6 and 10^7 data files, but the memory allocation reached limit for further data files. But in comparison with lm command, it is way faster in processing, as being shown by the RAM allocation and "Number of minutes running".

BigMemory & BigAnalytics

```
# library(bigmemory)
# library(biganalytics)
# gc(reset = TRUE)
# start.time<-proc.time()
#
# n1 <- 1000000
# x1 <- 1:n1
# x2 <-runif(n1,5,95)
# x3 <- rbinom(n1,1,.4)
# x4 <- rnorm(n1, mean=-30, sd=200)
# x5 <- runif(n1,-5000,5000)
# b0 <- 17; b1 <- -0.466; b2 <- 0.037; b3 <- -5.2; b4 <- 2; b5 <- 0.00876
# sigma <- 1.4
# epsilon <- rnorm(x1,0,sigma)
# y <- b0 + b1*x1  + b2*x2  + b3*x3 + b4*x4 + b5*x5 + x1*x2 + epsilon
#
```

```
#
# write.big.matrix(as.big.matrix(cbind(y,x1,x2,x3,x4,x5)), file = "data14Tempa.txt", sep = " ")
#
# end.time<-proc.time()
# save.time<-end.time-start.time
# cat("\nNumber of minutes running to write: ", save.time[3]/60, "\n\n")
#
# start.time<-proc.time()
#
# w<-read.big.matrix("data14Tempa.txt", header=F, sep=" ", col.names = c("y","x1","x2","x3","x4","x5"))
#
# end.time<-proc.time()
# save.time<-end.time-start.time
# cat("\nNumber of minutes running to read: ", save.time[3]/60, "\n\n")
#
# start.time<-proc.time()
#
# summary(biglm.big.matrix(y~x1+x2+x3+x4+x5, w))
#
# end.time<-proc.time()
# save.time<-end.time-start.time
# cat("\nNumber of minutes running to find biglm: ", save.time[3]/60, "\n\n")
#
# gc()
```

```
# gc(reset = TRUE)
# start.time<-proc.time()
#
# w<-read.big.matrix("data14b.txt", header=F, sep=" ", col.names = c("y","x1","x2","x3","x4","x5"))
#
# summary(bigglm.big.matrix(y~x1+x2+x3+x4+x5, w))
#
# end.time<-proc.time()
# save.time<-end.time-start.time
# cat("\nNumber of minutes running: ", save.time[3]/60, "\n\n")
# gc()
```

```
# gc(reset = TRUE)
# start.time<-proc.time()
#
# w<-read.big.matrix("data14c.txt", header=F, sep=" ", col.names = c("y","x1","x2","x3","x4","x5"))
#
# summary(biglm.big.matrix(y~x1+x2+x3+x4+x5, w))
#
# end.time<-proc.time()
# save.time<-end.time-start.time
# cat("\nNumber of minutes running: ", save.time[3]/60, "\n\n")
# gc()
```

```
# library(biglm)
# #library(ncdf4)
#
# n1 <- 1000000
# x1 <- 1:n1
```

```
# x2 <-runif(n1,5,95)
# x3 <- rbinom(n1,1,.4)
# x4 <- rnorm(n1, mean=-30, sd=200)
# x5 <- runif(n1,-5000,5000)
# b0 <- 17; b1 <- -0.466; b2 <- 0.037; b3 <- -5.2; b4 <- 2; b5 <- 0.00876
# sigma <- 1.4
# epsilon <- rnorm(x1,0,sigma)
# y <- b0 + b1*x1  + b2*x2  + b3*x3 + b4*x4 + b5*x5 + x1*x2 + epsilon
#
# ncpath <- ""
# ncname <- "ncdTemp1"
# ncfile <- paste(ncpath, ncname,".nc4",sep="")
# fwrite(data.frame(cbind(y,x1,x2,x3,x4,x5)),ncfile, row.names=FALSE, sep=" ")
```

```
# library(biglm)
# #library(ncdf4)
#
# n1 <- 10000000
# x1 <- 1:n1
# x2 <-runif(n1,5,95)
# x3 <- rbinom(n1,1,.4)
# x4 <- rnorm(n1, mean=-30, sd=200)
# x5 <- runif(n1,-5000,5000)
# b0 <- 17; b1 <- -0.466; b2 <- 0.037; b3 <- -5.2; b4 <- 2; b5 <- 0.00876
# sigma <- 1.4
# epsilon <- rnorm(x1,0,sigma)
# y <- b0 + b1*x1  + b2*x2  + b3*x3 + b4*x4 + b5*x5 + x1*x2 + epsilon
#
# ncpath <- ""
# ncname <- "ncdTemp2"
# ncfile <- paste(ncpath, ncname,".nc",sep="")
# fwrite(data.frame(cbind(y,x1,x2,x3,x4,x5)),ncfile, row.names=FALSE, sep=" ")
```

```
# library(biglm)
# library(ncdf4)
#
# n1 <- 50000000
# x1 <- 1:n1
# x2 <-runif(n1,5,95)
# x3 <- rbinom(n1,1,.4)
# x4 <- rnorm(n1, mean=-30, sd=200)
# x5 <- runif(n1,-5000,5000)
# b0 <- 17; b1 <- -0.466; b2 <- 0.037; b3 <- -5.2; b4 <- 2; b5 <- 0.00876
# sigma <- 1.4
# epsilon <- rnorm(x1,0,sigma)
# y <- b0 + b1*x1  + b2*x2  + b3*x3 + b4*x4 + b5*x5 + x1*x2 + epsilon
#
# ncpath <- ""
# ncname <- "ncdTemp3"
# ncfile <- paste(ncpath, ncname,".nc",sep="")
# fwrite(data.frame(cbind(y,x1,x2,x3,x4,x5)),ncfile, row.names=FALSE, sep=" ")
```

5 Assignment D

5.1 Q.1:

Identify the ARIMA(p, d, q) model and the white noise variance estimate for the given data sets. (Use ts.plot, acf, pacf, eacf, arima, etc. Avoid using auto.arima except for verifying your answer.)

```
# Read all data sets

TSD1 <- read.table("TSD1.txt", stringsAsFactors=FALSE)
TSD2 <- read.table("TSD2.txt", stringsAsFactors=FALSE)
TSD3 <- read.table("TSD3.txt", stringsAsFactors=FALSE)
TSD4 <- read.table("TSD4.txt", stringsAsFactors=FALSE)
TSD5 <- read.table("TSD5.txt", stringsAsFactors=FALSE)
TSD6 <- read.table("TSD6.txt", stringsAsFactors=FALSE)
TSD7 <- read.table("TSD7.txt", stringsAsFactors=FALSE)
TSD8 <- read.table("TSD8.txt", stringsAsFactors=FALSE)
TSD9 <- read.table("TSD9.txt", stringsAsFactors=FALSE)
TSD10 <- read.table("TSD10.txt", stringsAsFactors=FALSE)

summary(TSD1)
```

```
##       V1                 V2
##  Length:101         Length:101
##  Class :character   Class :character
##  Mode  :character   Mode  :character
```

```
summary(TSD2)
```

```
##        x
##  Min.   :-4.0165
##  1st Qu.:-0.9441
##  Median : 0.4825
##  Mean   : 0.4906
##  3rd Qu.: 2.3186
##  Max.   : 5.3325
```

```
summary(TSD3)
```

```
##        x
##  Min.   :-3.798148
##  1st Qu.:-1.031366
##  Median :-0.005729
##  Mean   : 0.045748
##  3rd Qu.: 1.141568
##  Max.   : 3.339151
```

```
summary(TSD4)
```

```
##        x
##  Min.   :-4.37936
##  1st Qu.:-1.58074
##  Median :-0.08189
##  Mean   :-0.03051
##  3rd Qu.: 1.24958
##  Max.   : 7.62453
```

```
summary(TSD5)
```

```
##        x
##  Min.   :-43.099
##  1st Qu.:-25.925
##  Median :-14.696
##  Mean   :-17.397
##  3rd Qu.: -8.464
##  Max.   :  0.000
```

```
summary(TSD6)
```

```
##        x
##  Min.   :-8.8881
##  1st Qu.:-4.2119
##  Median : 0.2404
##  Mean   : 1.1125
##  3rd Qu.: 4.9456
##  Max.   :18.5225
```

```
summary(TSD7)
```

```
##        x
##  Min.   :-1.129
##  1st Qu.: 1.773
##  Median : 4.775
##  Mean   : 6.467
##  3rd Qu.:10.743
##  Max.   :19.927
```

```
summary(TSD8)
```

```
##        x
##  Min.   :-4.13392
##  1st Qu.:-1.94503
##  Median : 0.03018
##  Mean   :-0.12059
##  3rd Qu.: 1.37310
##  Max.   : 4.87460
```

```
summary(TSD9)
```

```
##        x
##  Min.   :-6.8082
##  1st Qu.:-1.8236
##  Median : 0.1982
##  Mean   :-0.2035
##  3rd Qu.: 1.4928
##  Max.   : 5.7262
```

```
summary(TSD10)
```

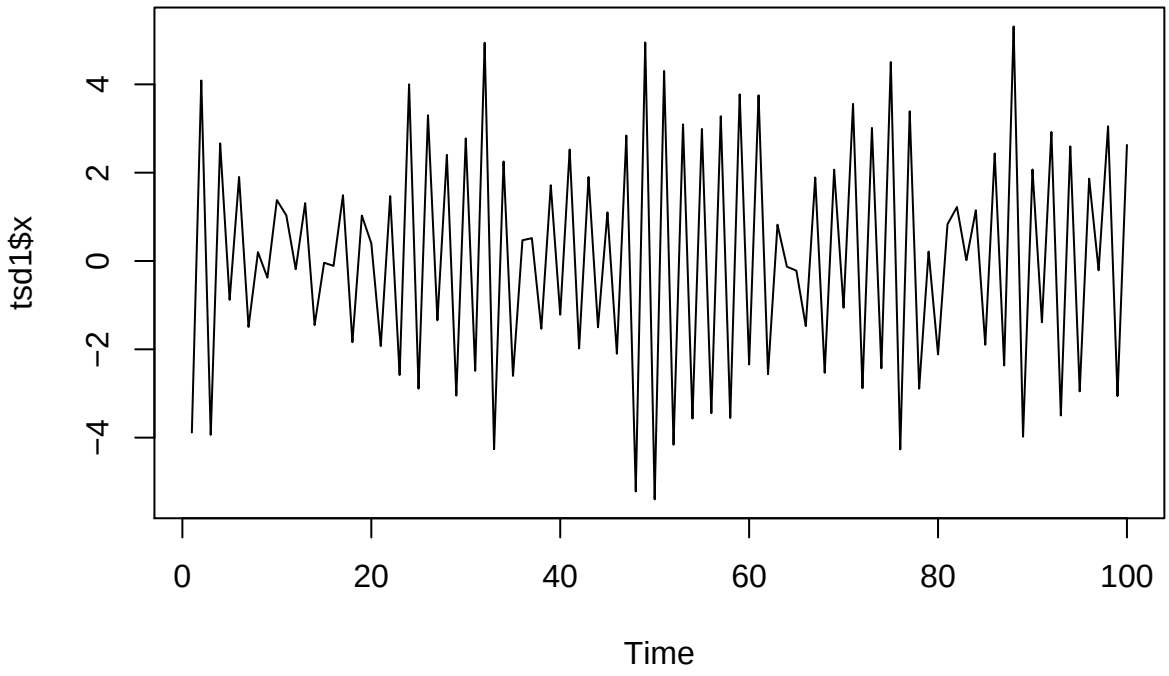
```
##        x
##  Min.   :-6031.949
##  1st Qu.:-4754.002
##  Median :-2725.071
##  Mean   :-2808.512
```

```
##  3rd Qu.: -916.601
##  Max.   :    7.283
```

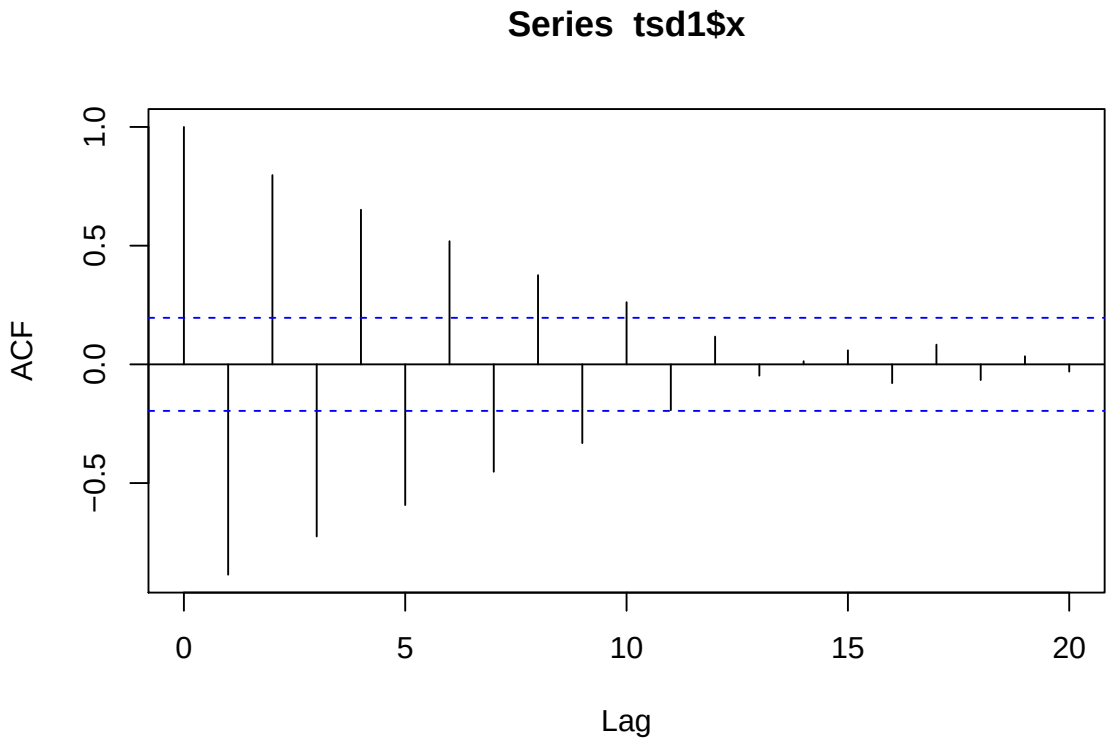
```
# as.numeric(TSD1$x)
# as.numeric(TSD2$x)
# as.numeric(TSD3$x)
# as.numeric(TSD4$x)
# as.numeric(TSD5$x)
# as.numeric(TSD6$x)
# as.numeric(TSD7$x)
# as.numeric(TSD8$x)
# as.numeric(TSD9$x)
# as.numeric(TSD10$x)
```

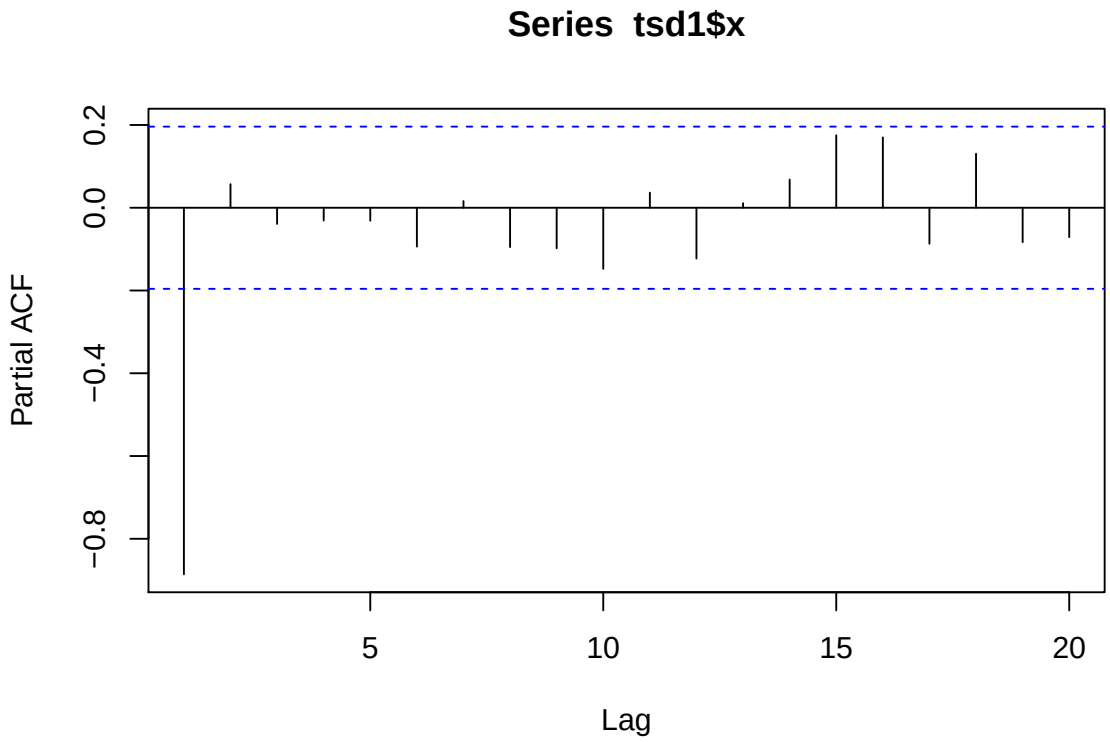
Identify the ARIMA(p, d, q) model and the white noise variance estimate for the given data sets. (Use ts.plot, acf, pacf, eacf, arima, etc. Avoid using auto.arima except for verifying your answer.)

Dataset1 : TSD1.txt



- Here we can see there is no trend and no seasonal component. Let's now plot acf and pacf


Series tsd1$x
ACF
1.0
0.5
0.0
−0.5
0
5
10
15
20
Lag


Series tsd1$x
Partial ACF
0.2
0.0
−0.4
−0.8
5
10
15
20
Lag

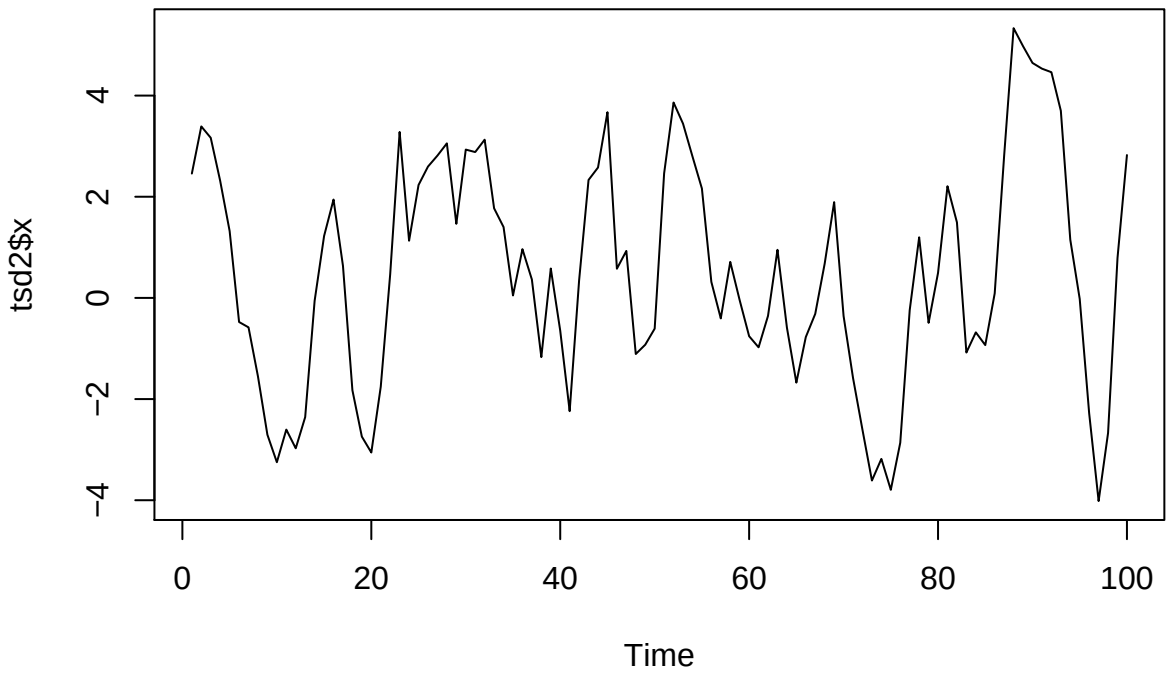
Here we can see that the ACF plot is tailing off and the PACF plot is cut off after p=1, So the best suitable model is AR(1)

Let us verify with auto.arima

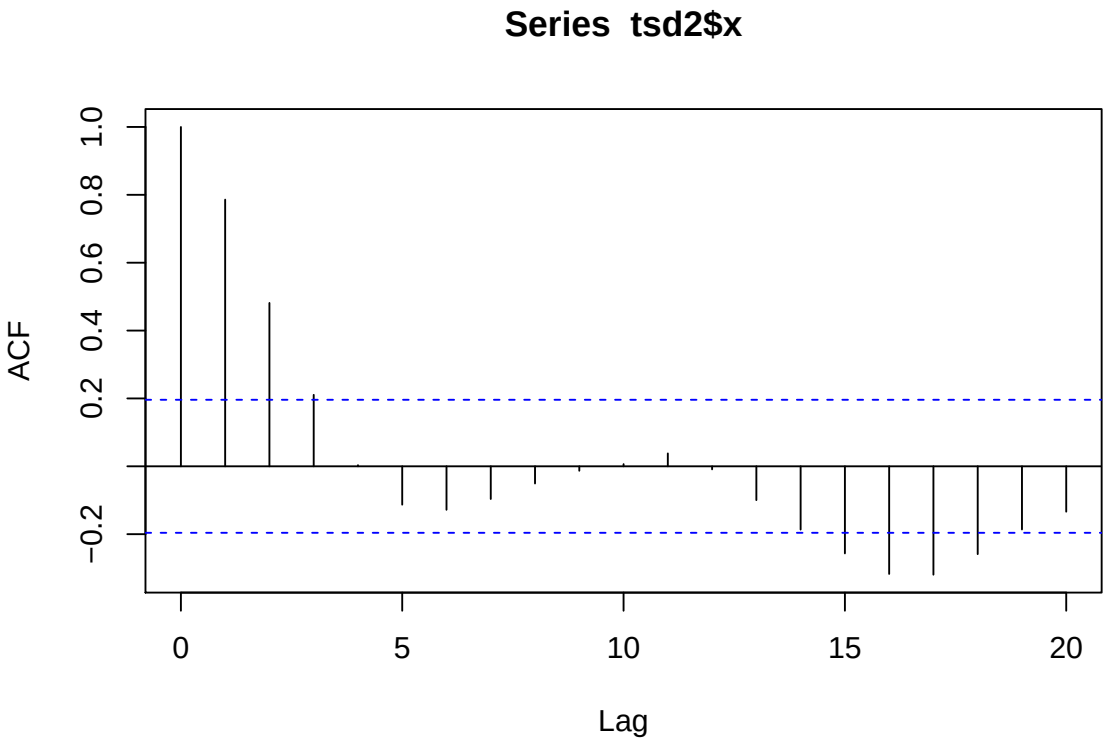
```
## Series: tsd1$x
## ARIMA(1,0,0) with zero mean
##
## Coefficients:
##           ar1
##       -0.9030
## s.e.   0.0424
##
## sigma^2 estimated as 1.4:  log likelihood=-159.07
## AIC=322.13   AICc=322.26   BIC=327.35
```

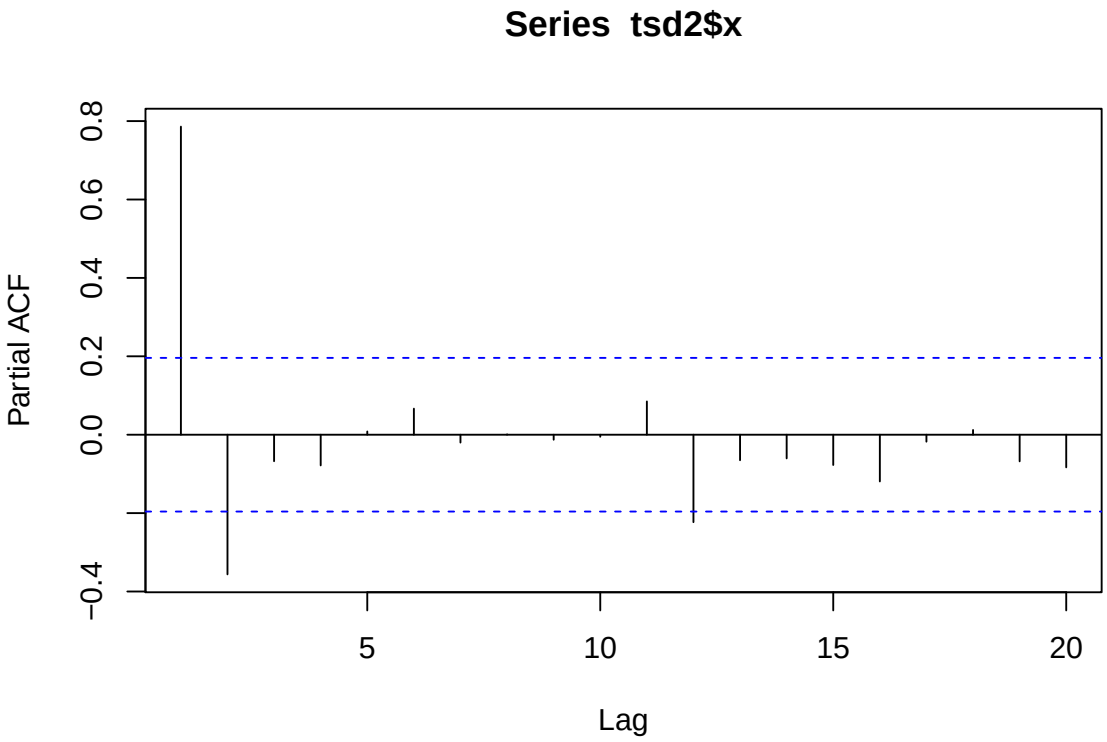
So we verify that the model is AR(1)

Dataset1 : TSD2.txt



- Here we can see there is no trend and no seasonal component. Let's now plot acf and pacf


Series tsd2$x
ACF
1.0
0.8
0.6
0.4
0.2
−0.2
0
5
10
15
20
Lag


Series tsd2$x
Partial ACF
0.8
0.6
0.4
0.2
0.0
−0.4
5
10
15
20
Lag

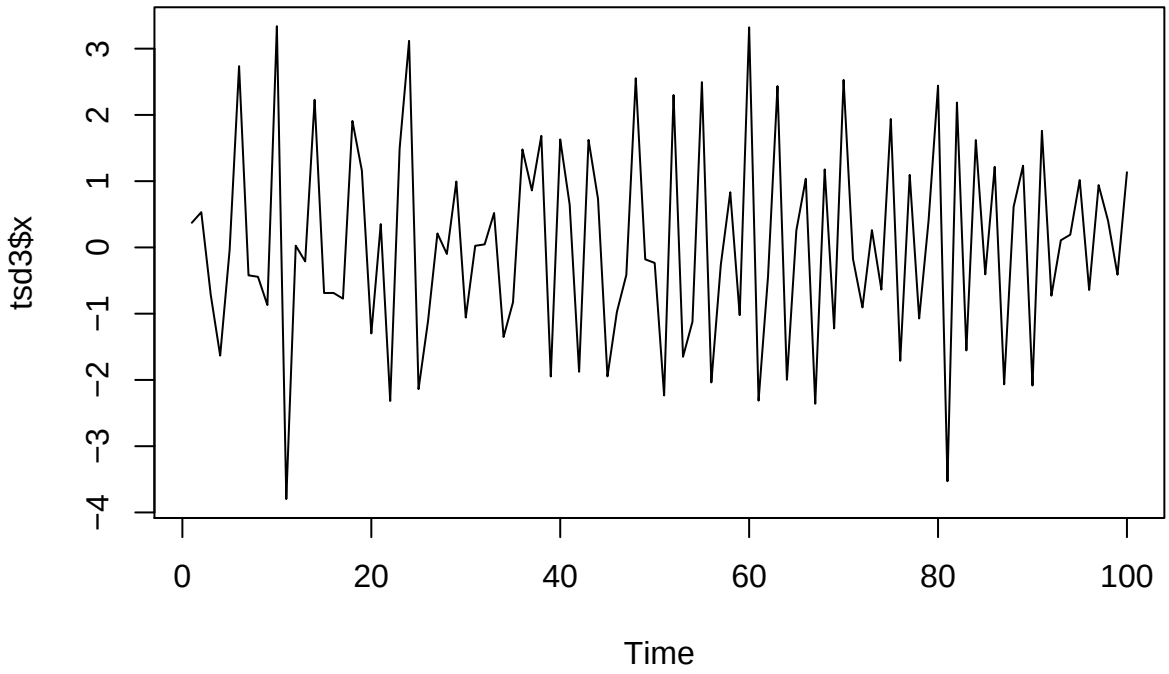
Here we can see that the ACF plot is tailing off and the PACF plot is cut off after p=2, So the best suitable model is AR(2)

Let us verify with auto.arima

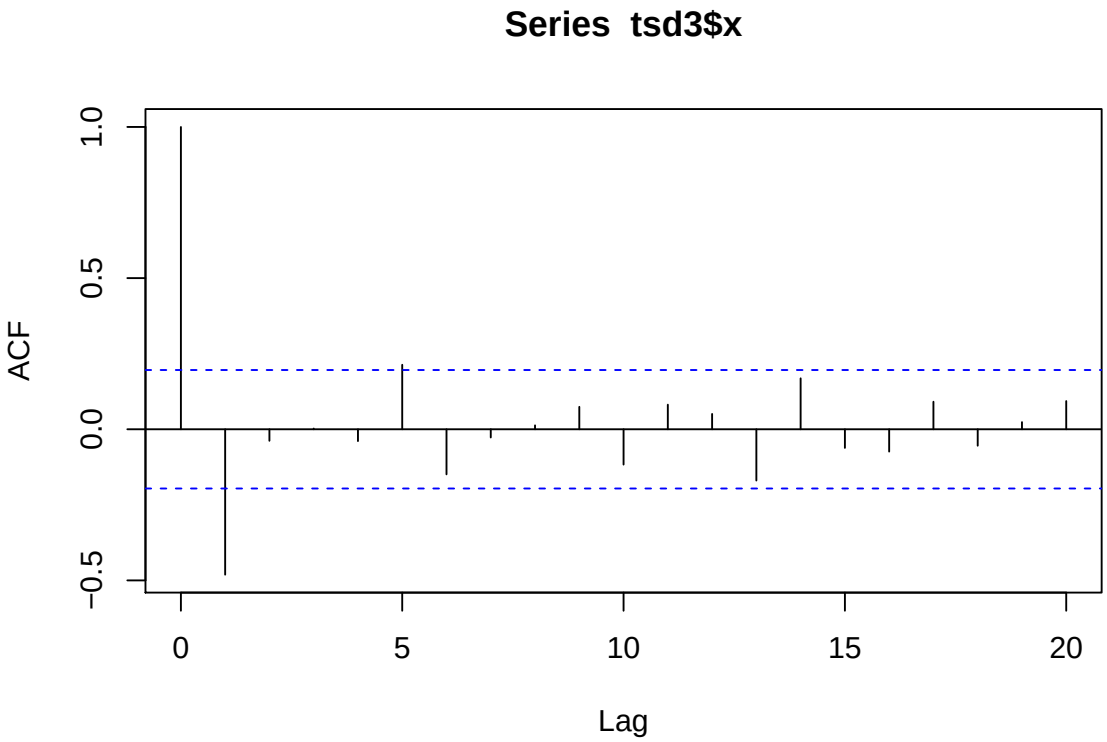
```
## Series: tsd2$x
## ARIMA(2,0,0) with zero mean
##
## Coefficients:
##          ar1      ar2
##       1.1051  -0.3747
## s.e.  0.0932   0.0939
##
## sigma^2 estimated as 1.619:  log likelihood=-165.64
## AIC=337.28   AICc=337.53   BIC=345.09
```

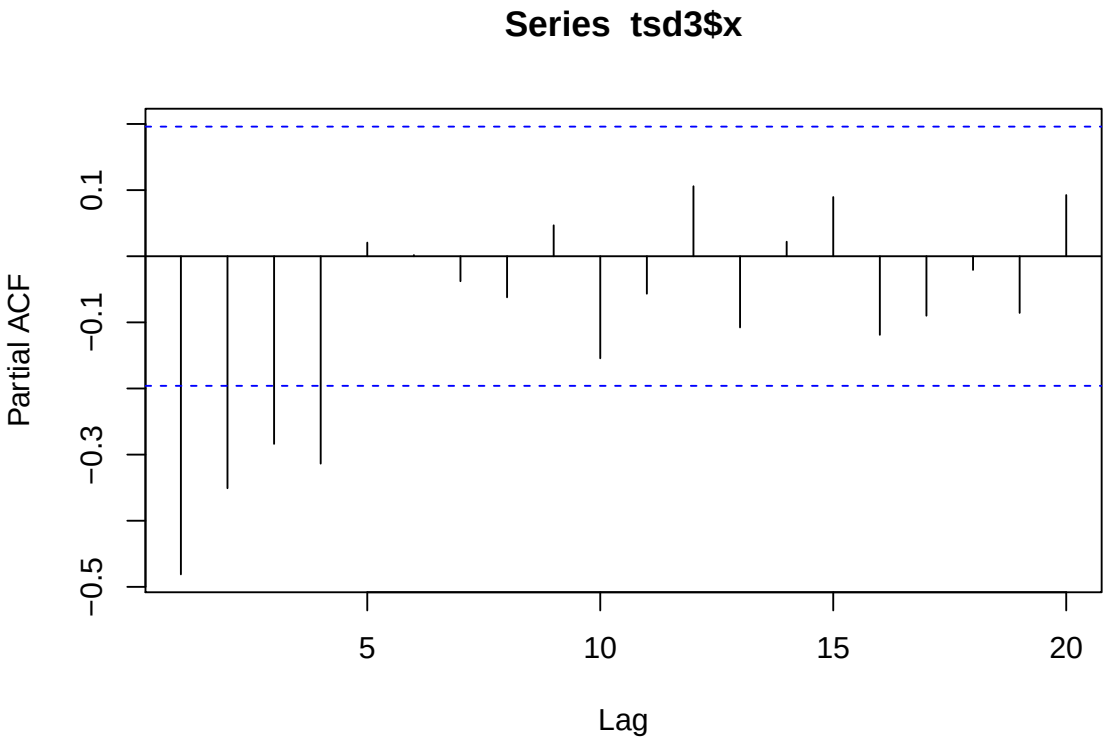
So we verify that the model is AR(2)

Dataset1 : TSD3.txt



- Here we can see there is no trend and no seasonal component. Let's now plot acf and pacf


Series tsd3$x
ACF
1.0
0.5
0.0
−0.5
0
5
10
15
20
Lag


Series tsd3$x
Partial ACF
0.1
−0.1
−0.3
−0.5
5
10
15
20
Lag

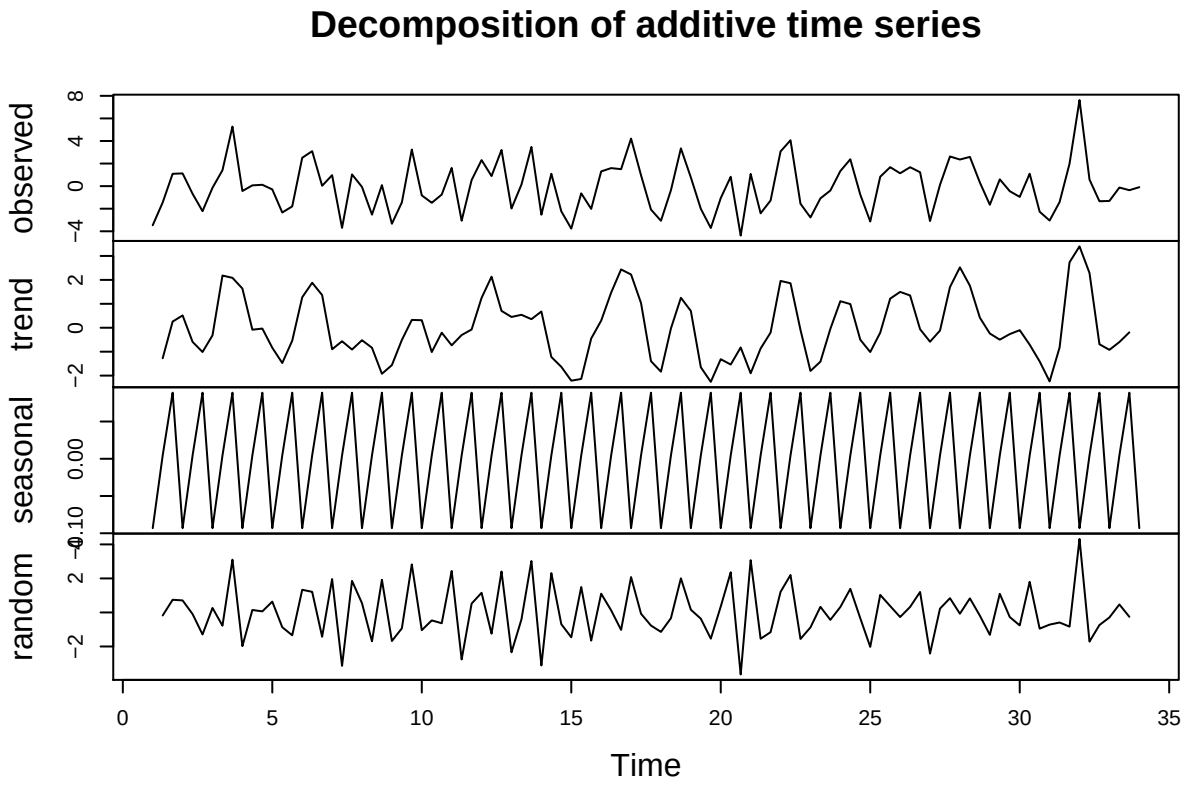
Here we can see that the ACF plot cuts off at q=1, So the best suitable model is MA(1)

Let us verify with auto.arima

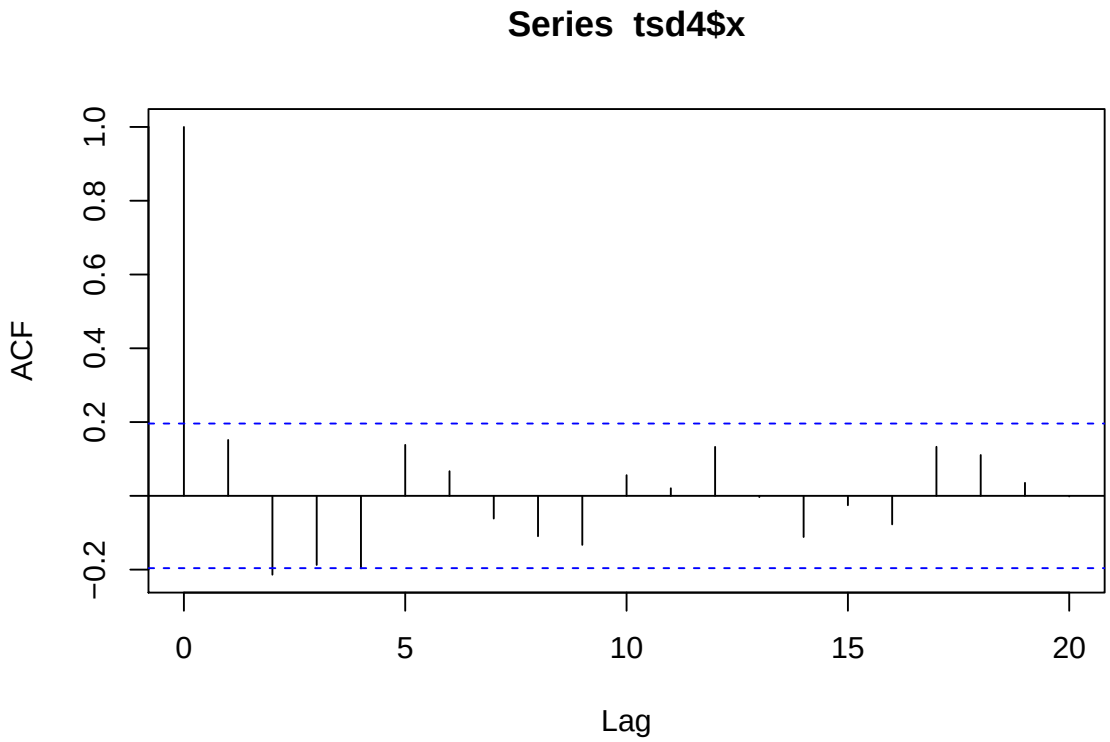
```
## Series: tsd3$x
## ARIMA(0,0,1) with non-zero mean
##
## Coefficients:
##           ma1    mean
##       -0.8675  0.0312
## s.e.   0.0666  0.0168
##
## sigma^2 estimated as 1.373:  log likelihood=-157.44
## AIC=320.89   AICc=321.14   BIC=328.7
```

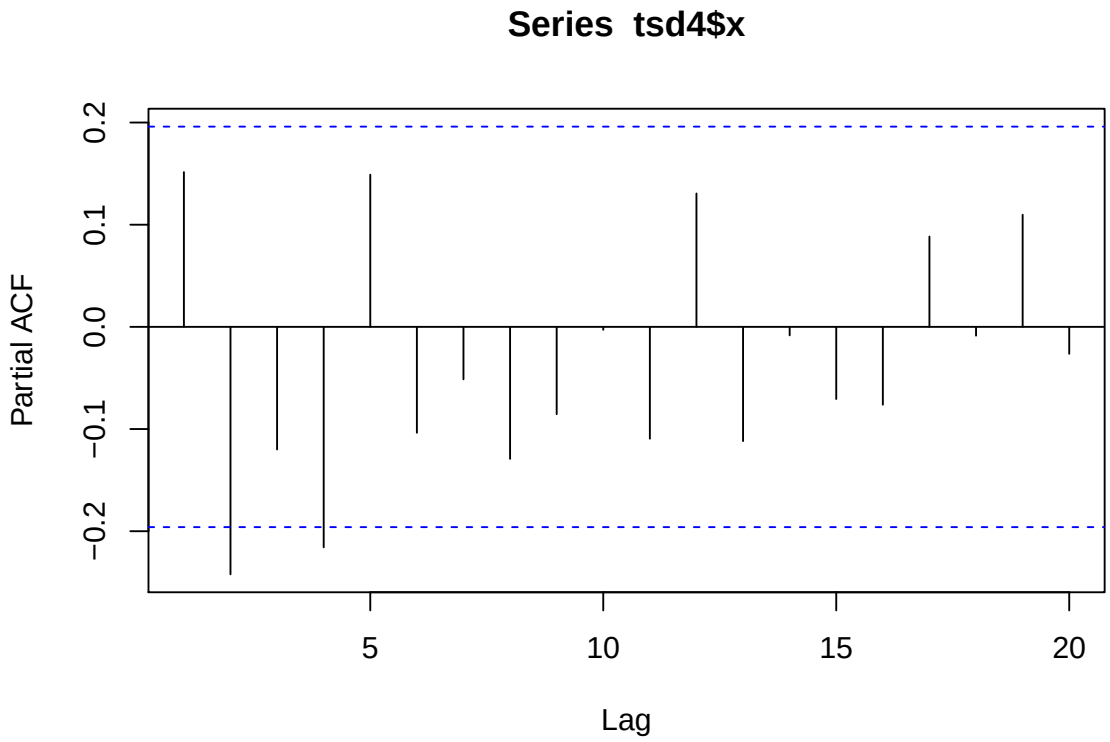
So we verify that the model is MA(1)

Dataset1 : TSD4.txt



- Here we can see there is no trend and no seasonal component. Let's now plot acf and pacf






```
## AR/MA
##   0 1 2 3 4 5 6 7 8 9 10 11 12 13
```

```
## 0 o x o o o o o o o o o  o  o  o
## 1 x x o o x o o o o o o  o  o  o
## 2 x x o x o o o o o o o  o  o  o
## 3 x x x x o o o o o o o  o  o  o
## 4 x o x x o o o o o o o  o  o  o
## 5 x o o o x o o o o o o  o  o  o
## 6 x o o o x o o o o o o  o  o  o
## 7 x x o o o o x o o o o  o  o  o
```

```
##
##  Box-Pierce test
##
## data:  tsd4$x
## X-squared = 2.2957, df = 1, p-value = 0.1297
```

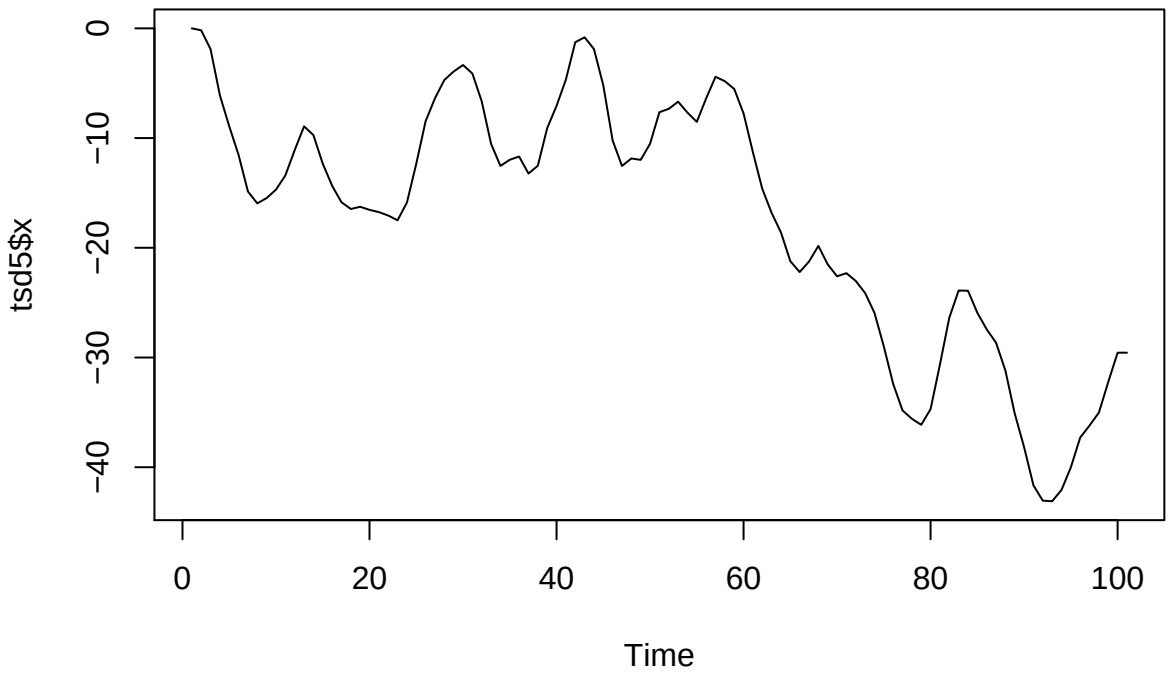
Here we can see that the ACF and pacf plot doesn't give us much thought so we go for a eacf plot and the eacf gives us the model as ARMA(0,0) with box-pierce test giving positive sign, we may also conclude that it is a white noise.

Let us verify with auto.arima

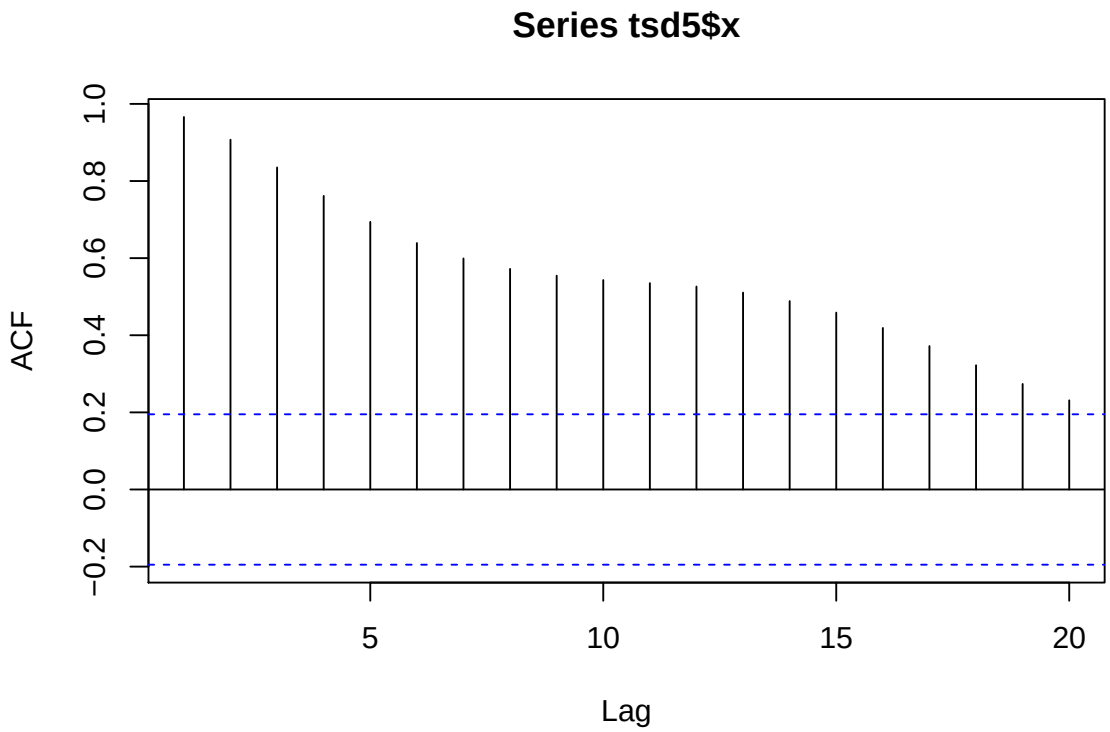
```
## Series: tsd4$x
## ARIMA(2,0,1) with zero mean
##
## Coefficients:
##          ar1      ar2      ma1
##       0.8938  -0.3164  -0.7896
## s.e.  0.1879   0.1025   0.1913
##
## sigma^2 estimated as 4.352:  log likelihood=-214.09
## AIC=436.18   AICc=436.6   BIC=446.6
```

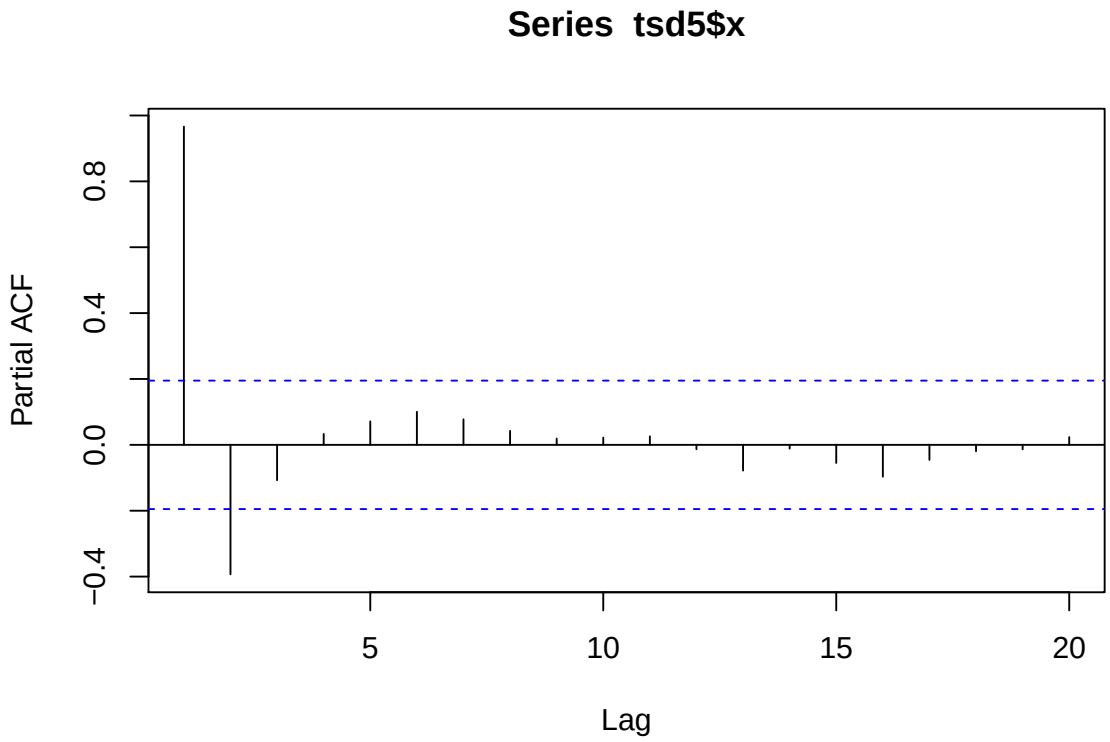
So we could not verify that the model is ARMA(0,0) as we get it as ARIMA(2,0,1)

Dataset1 : TSD5.txt



- Here we can see there is no trend and no seasonal component. Let's now plot acf and pacf


Series tsd5$x
ACF
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
5
10
15
20
Lag


Series tsd5$x
Partial ACF
0.8
0.4
0.0
−0.4
5
10
15
20
Lag

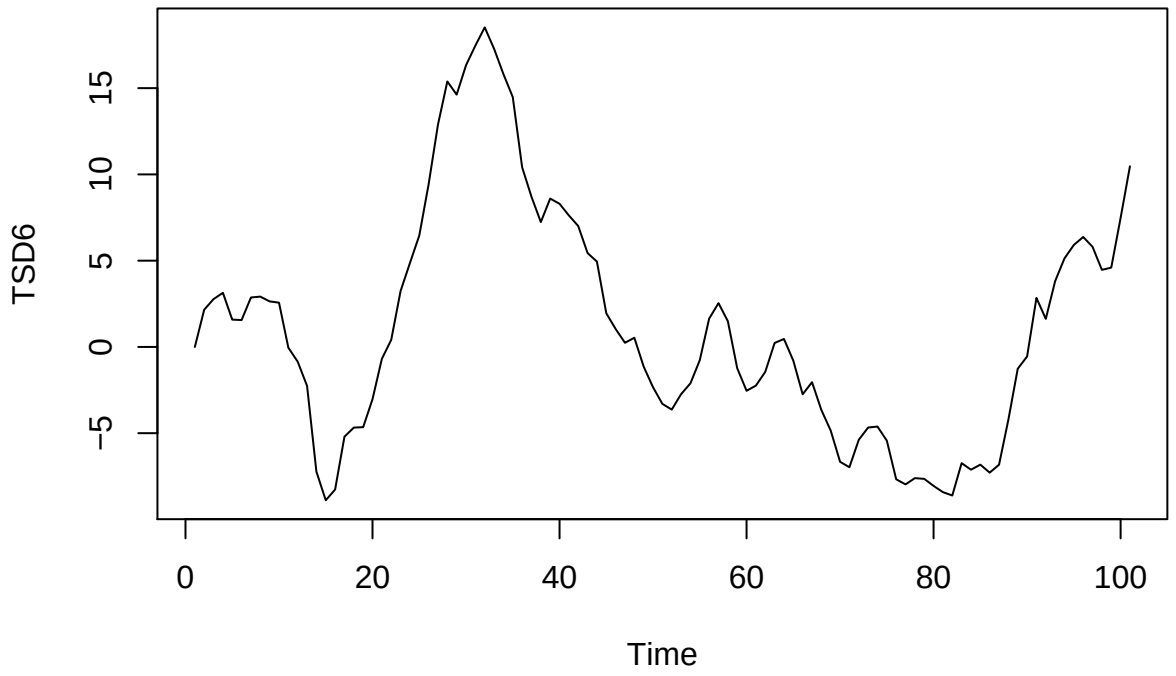
Here we can see that the ACF and pacf plot doesn't give us much thought so we go for a eacf plot and the eacf gives us the model as ARMA(0,0) with box-pierce test giving positive sign, we may also conclude that it is a white noise.

Let us verify with auto.arima

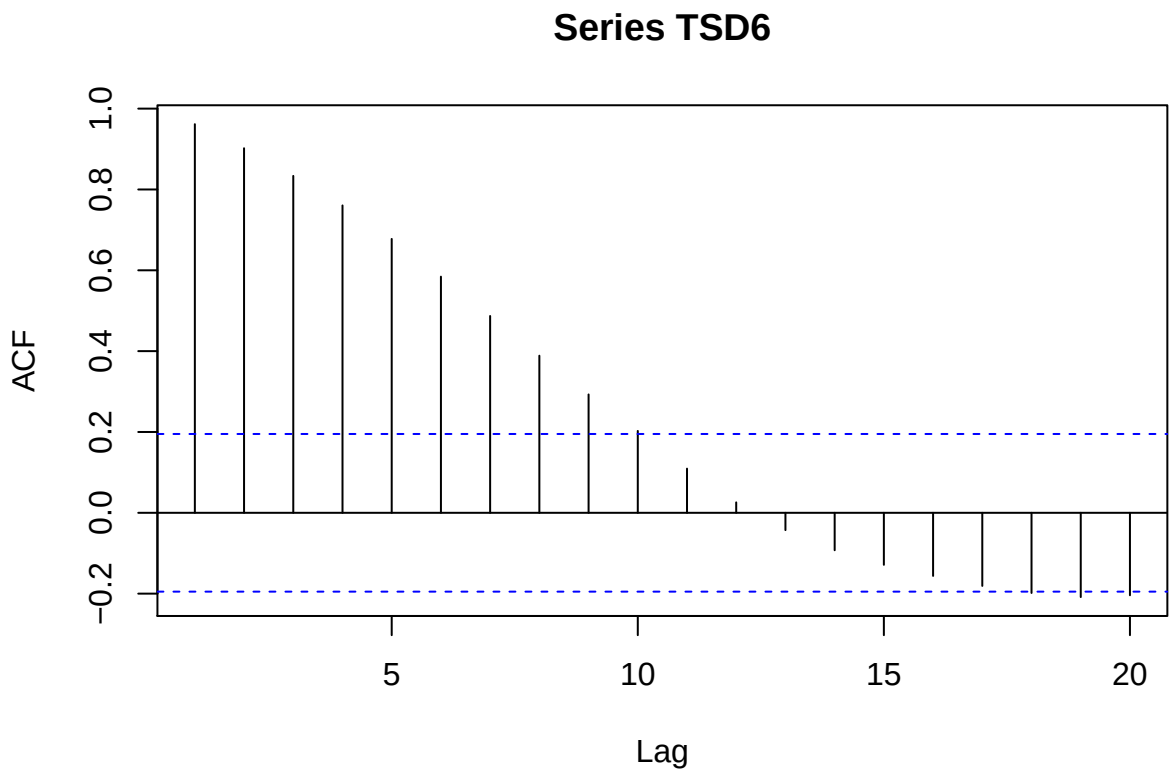
```
## Series: tsd5$x
## ARIMA(2,1,2) with drift
##
## Coefficients:
##          ar1      ar2      ma1      ma2    drift
##       1.4339  -0.5827  -0.2787  -0.5223  -0.3061
## s.e.  0.1060   0.0941   0.1174   0.1132   0.1692
##
## sigma^2 estimated as 1.52:  log likelihood=-161.13
## AIC=334.26   AICc=335.16   BIC=349.89
```

So we could not verify that the model is ARMA(0,0) as we get it as ARIMA(2,0,1)

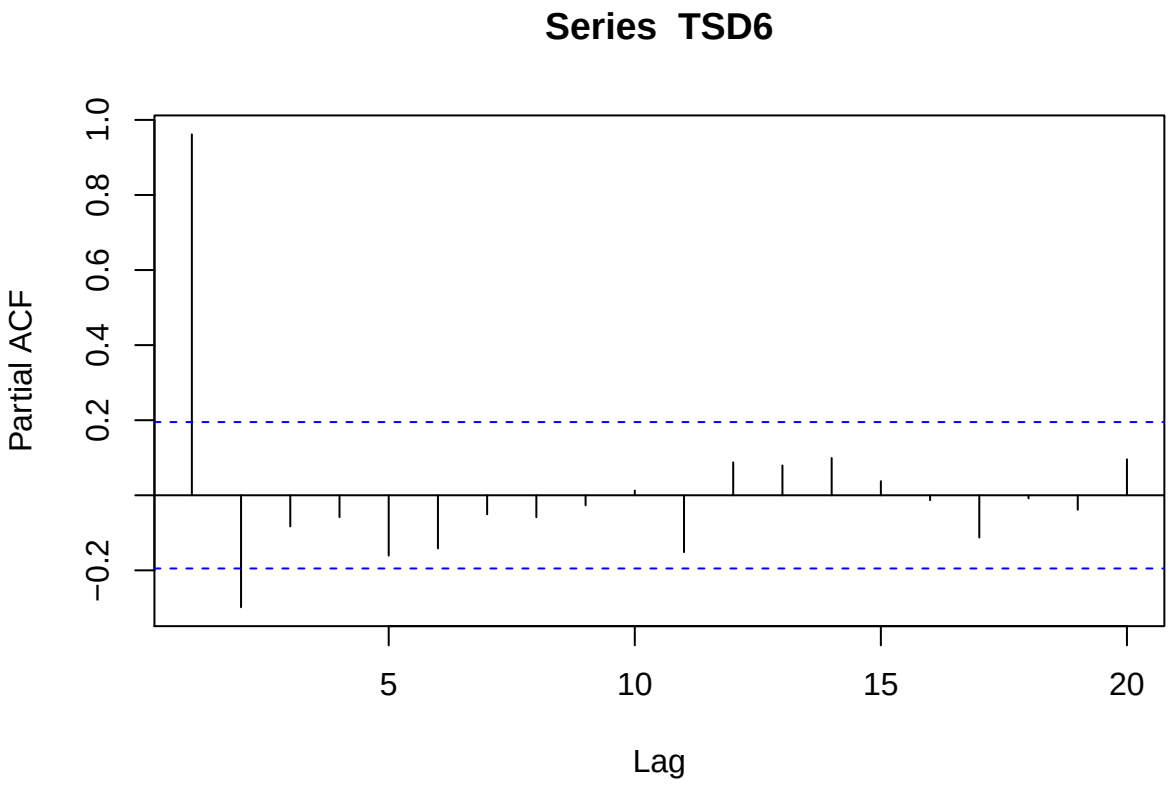
```
#TSD6
mean6<-mean(TSD6$x)
sd6<-sd(TSD6$x)
ts.plot(TSD6)
```



```
acf(TSD6)
```



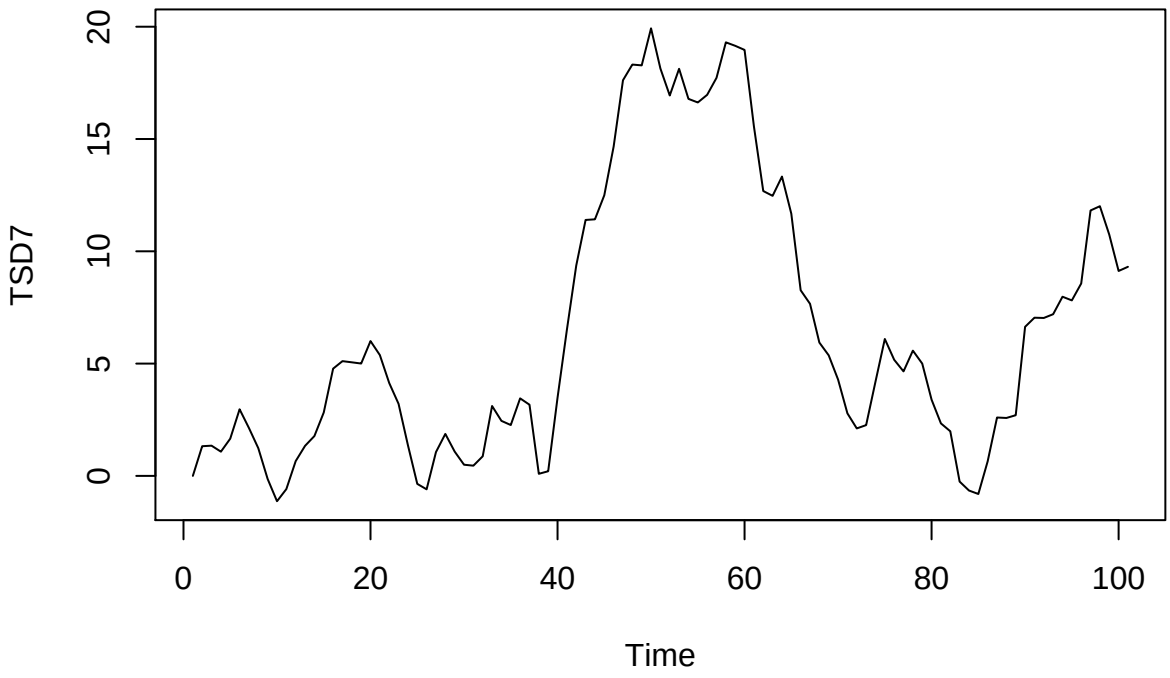

```
pacf(TSD6)
```



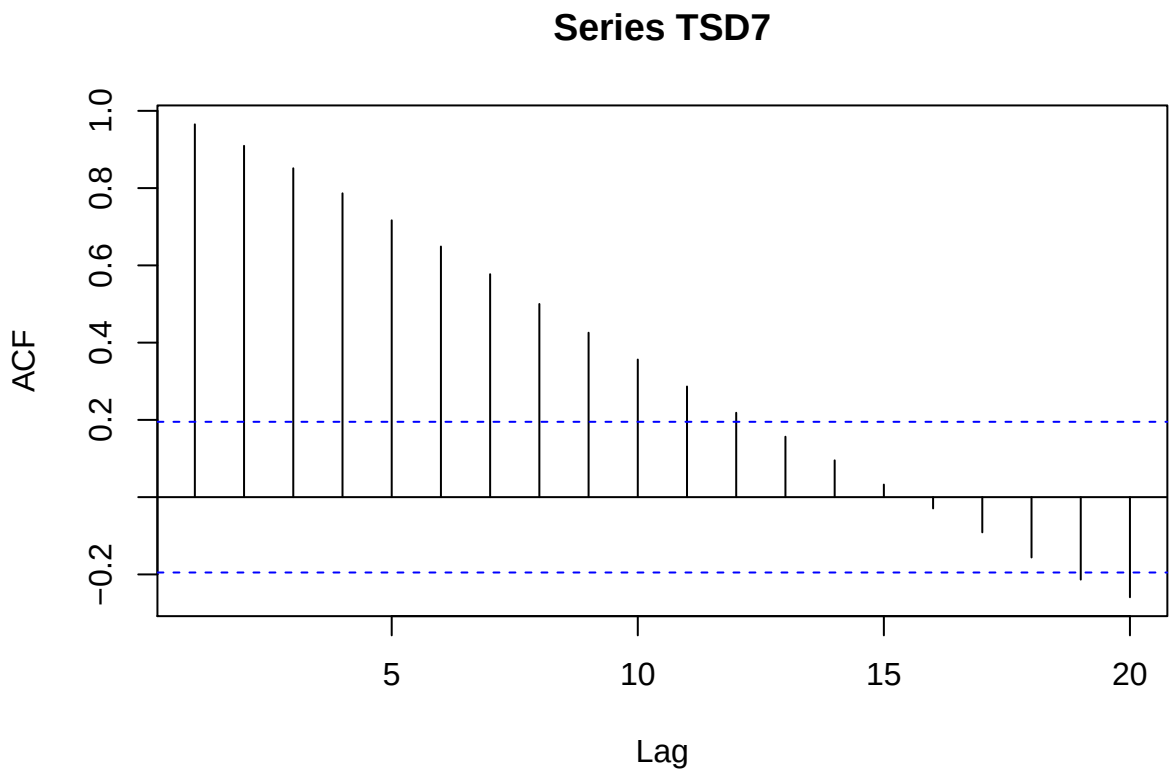

```
arima(TSD6)
```

```
##
## Call:
## arima(x = TSD6)
##
## Coefficients:
##       intercept
##          1.1125
## s.e.     0.6723
##
## sigma^2 estimated as 45.65:  log likelihood = -336.27,  aic = 674.54
```

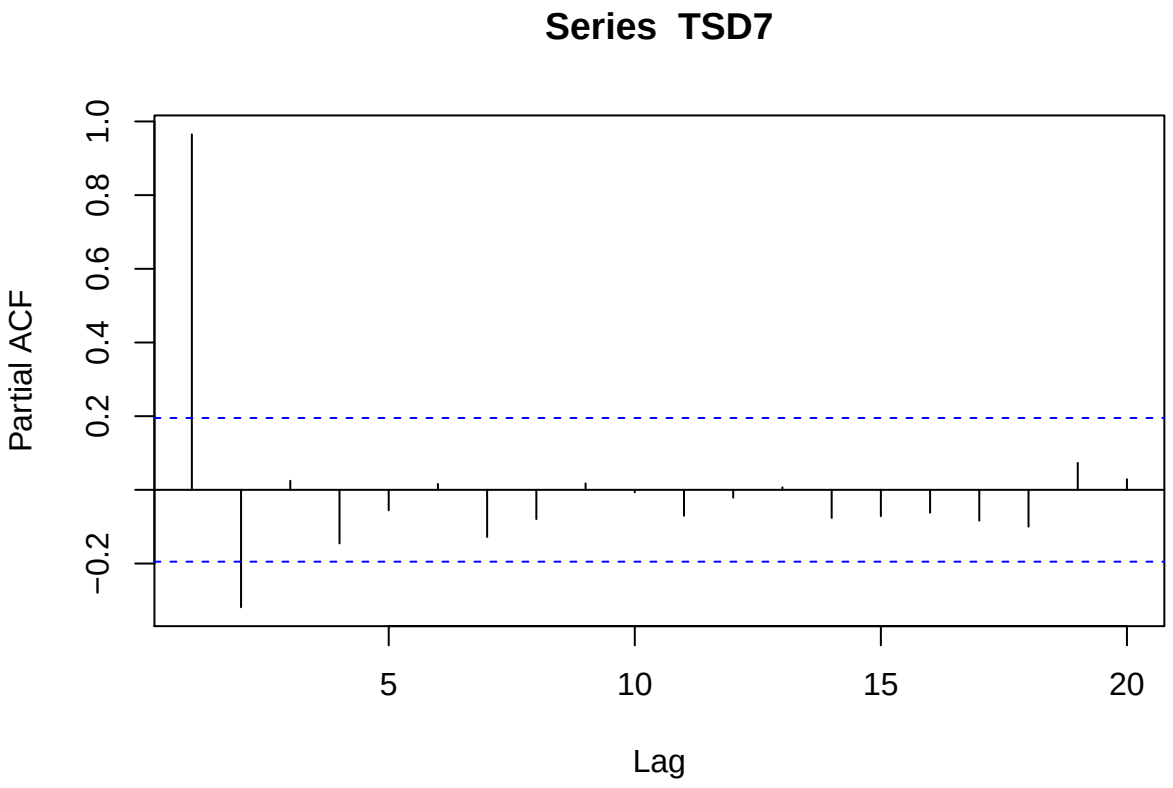
```
#TSD7
mean7<-mean(TSD7$x)
sd7<-sd(TSD7$x)
ts.plot(TSD7)
```




```
acf(TSD7)
```



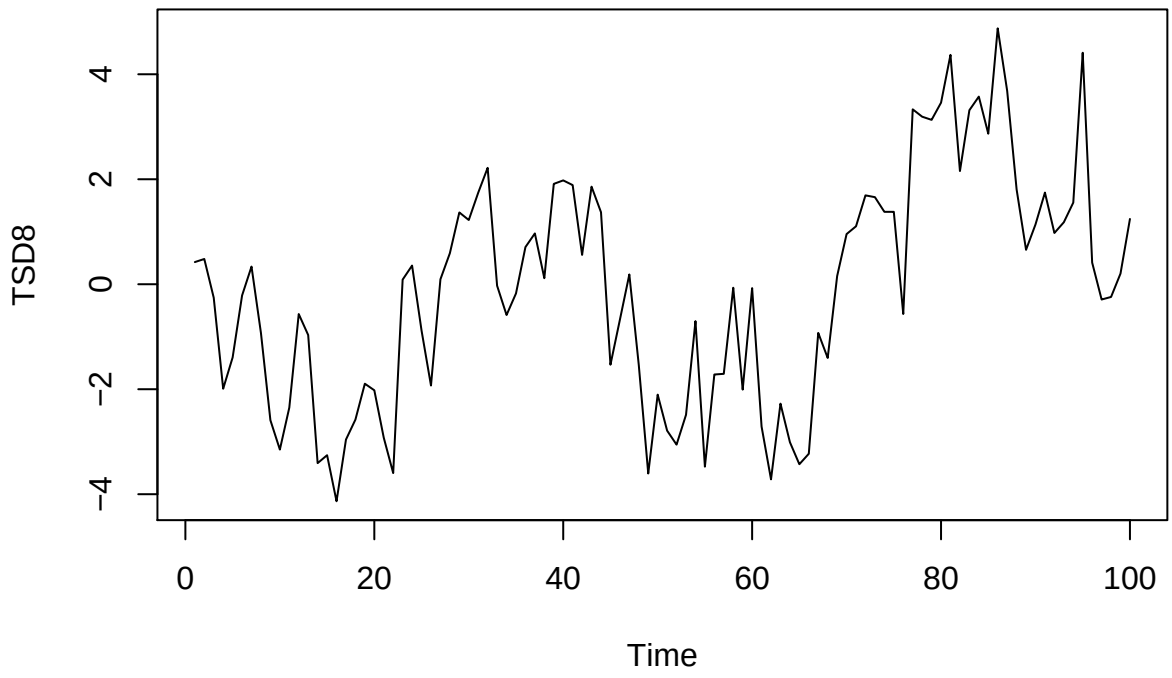

```
pacf(TSD7)
```



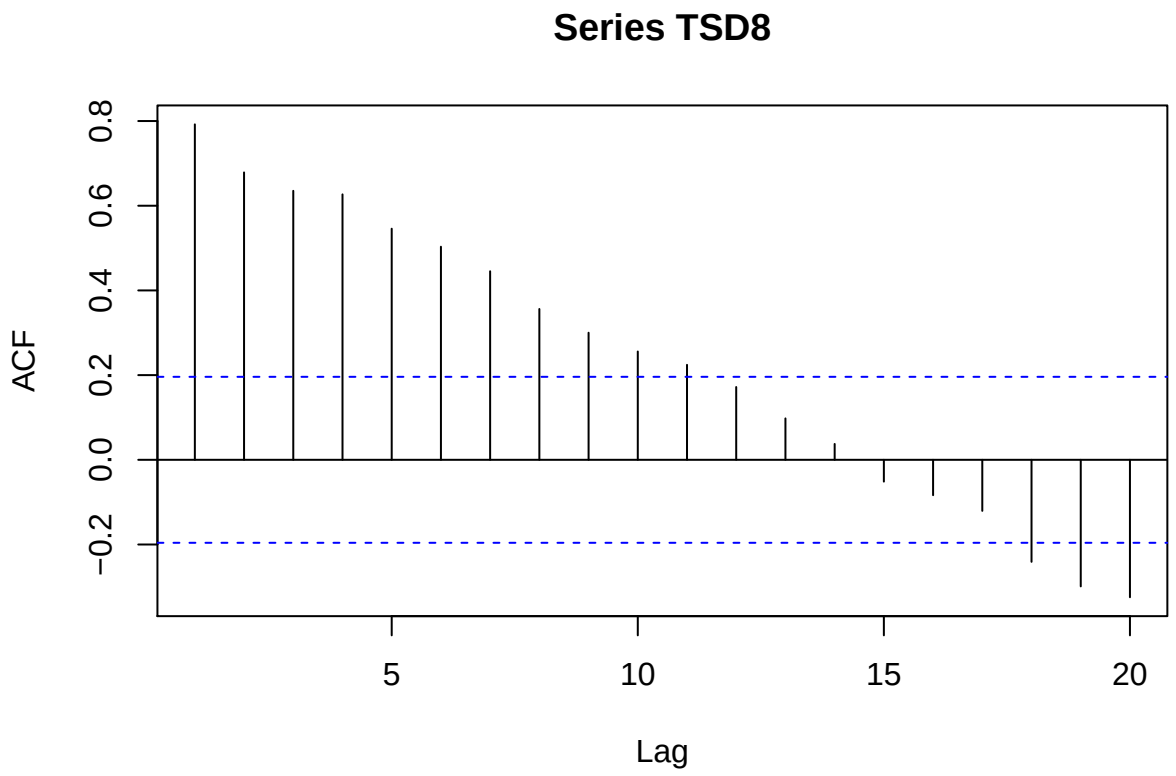

```
arima(TSD7)
```

```
##
## Call:
## arima(x = TSD7)
##
## Coefficients:
##       intercept
##          6.4674
## s.e.     0.6002
##
## sigma^2 estimated as 36.38:  log likelihood = -324.82,  aic = 651.63
```

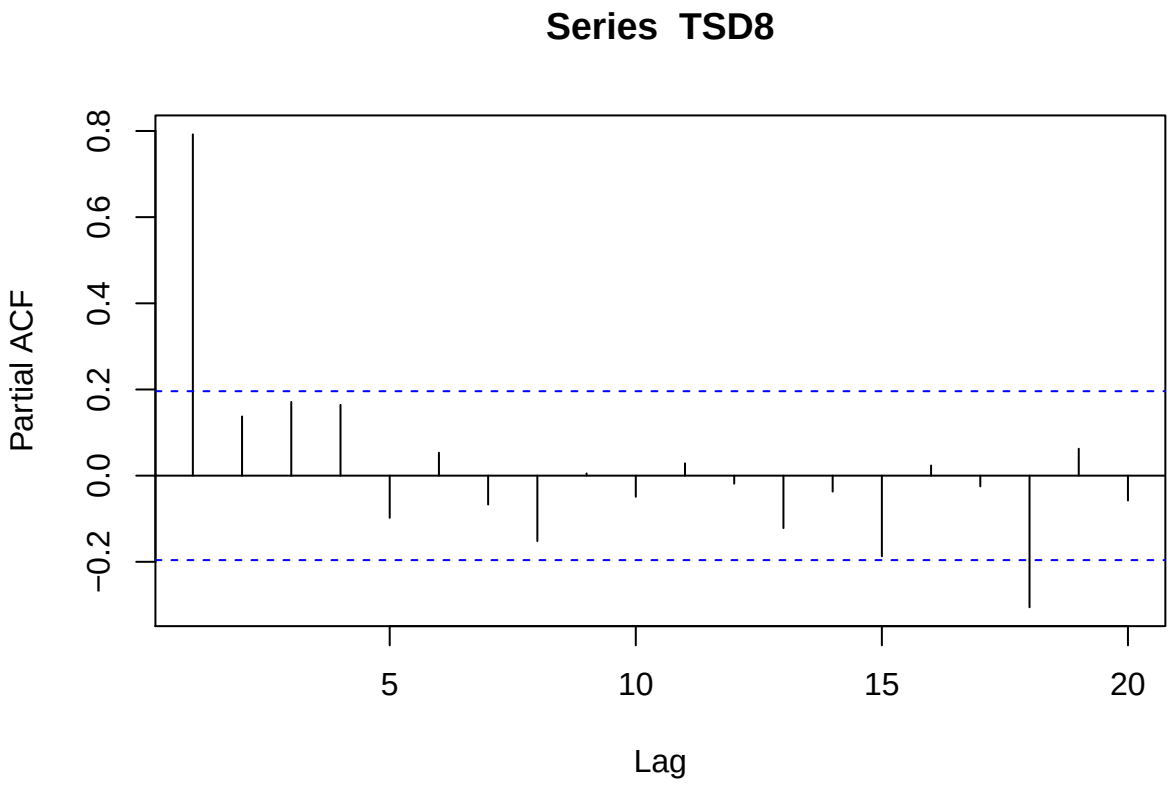
```
#TSD8
mean8<-mean(TSD8$x)
sd8<-sd(TSD8$x)
ts.plot(TSD8)
```




```
acf(TSD8)
```



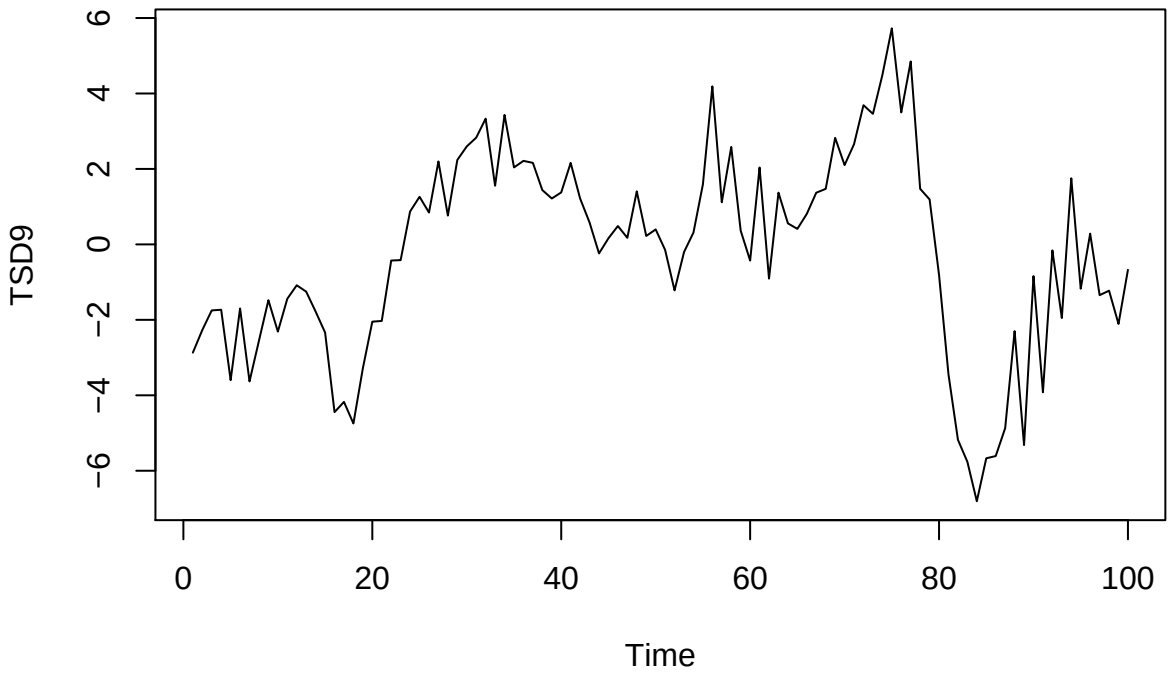

```
pacf(TSD8)
```



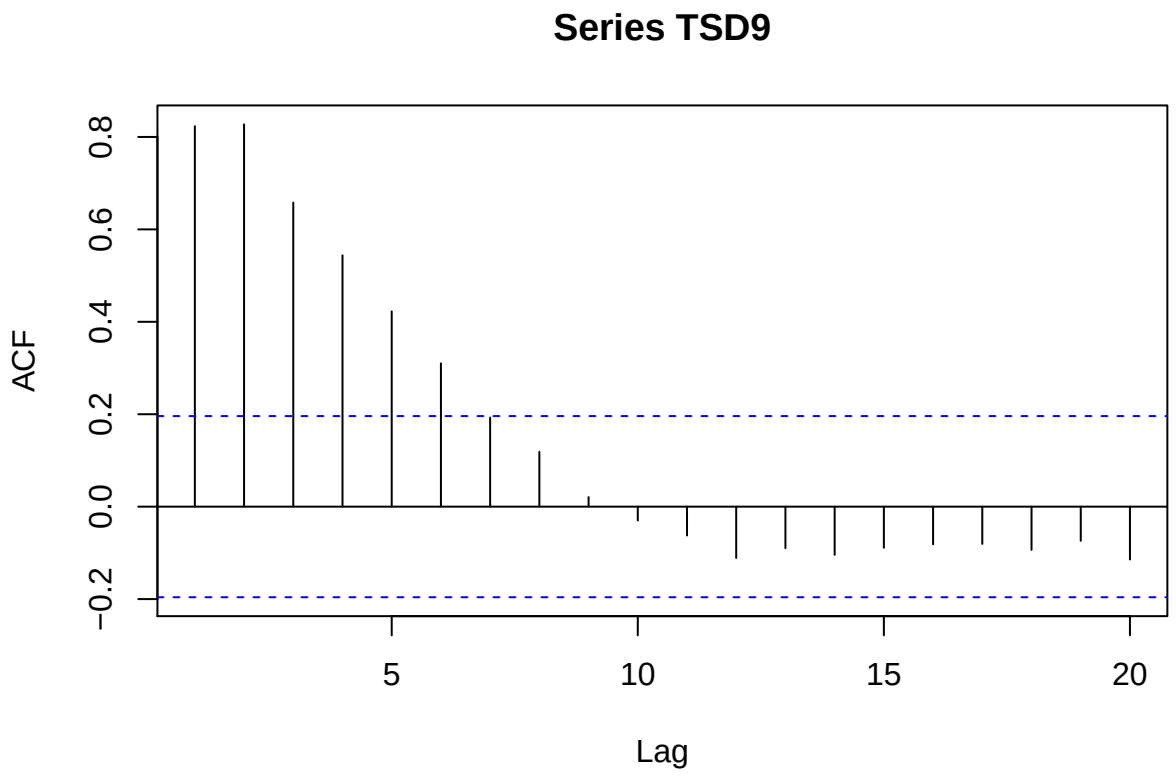

```
arima(TSD8)
```

```
##
## Call:
## arima(x = TSD8)
##
## Coefficients:
##       intercept
##         -0.1206
## s.e.     0.2143
##
## sigma^2 estimated as 4.592:  log likelihood = -218.11,  aic = 438.22
```

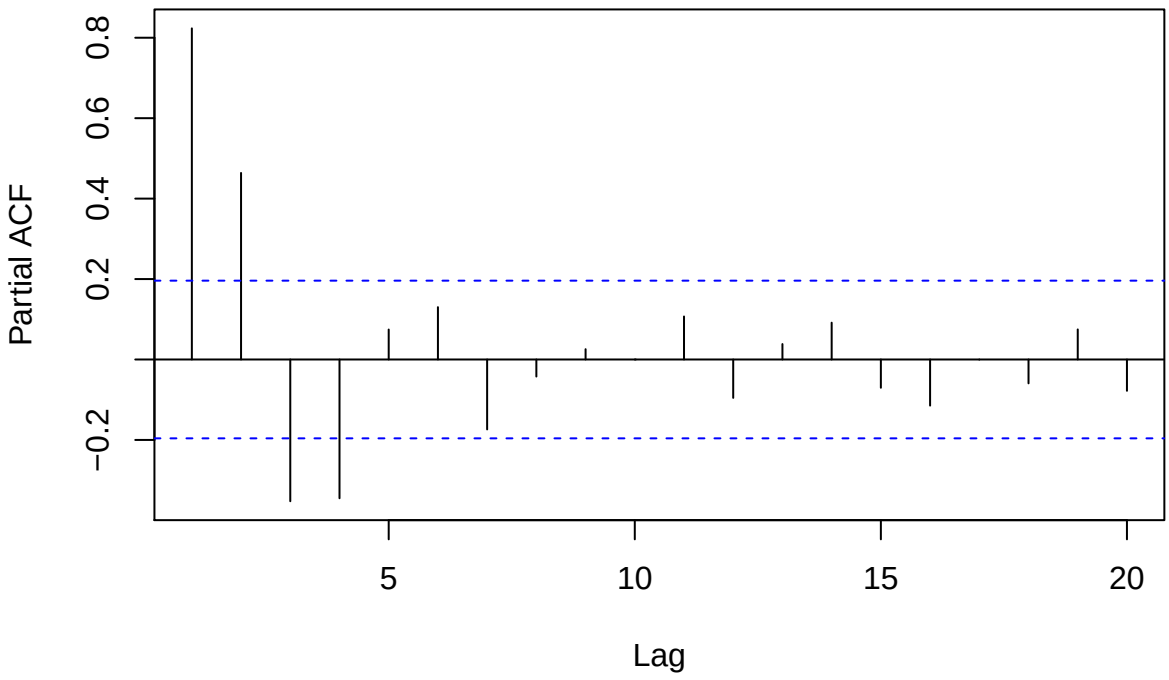
```
#TSD9
mean9<-mean(TSD9$x)
sd9<-sd(TSD9$x)
ts.plot(TSD9)
```




```
acf(TSD9)
```



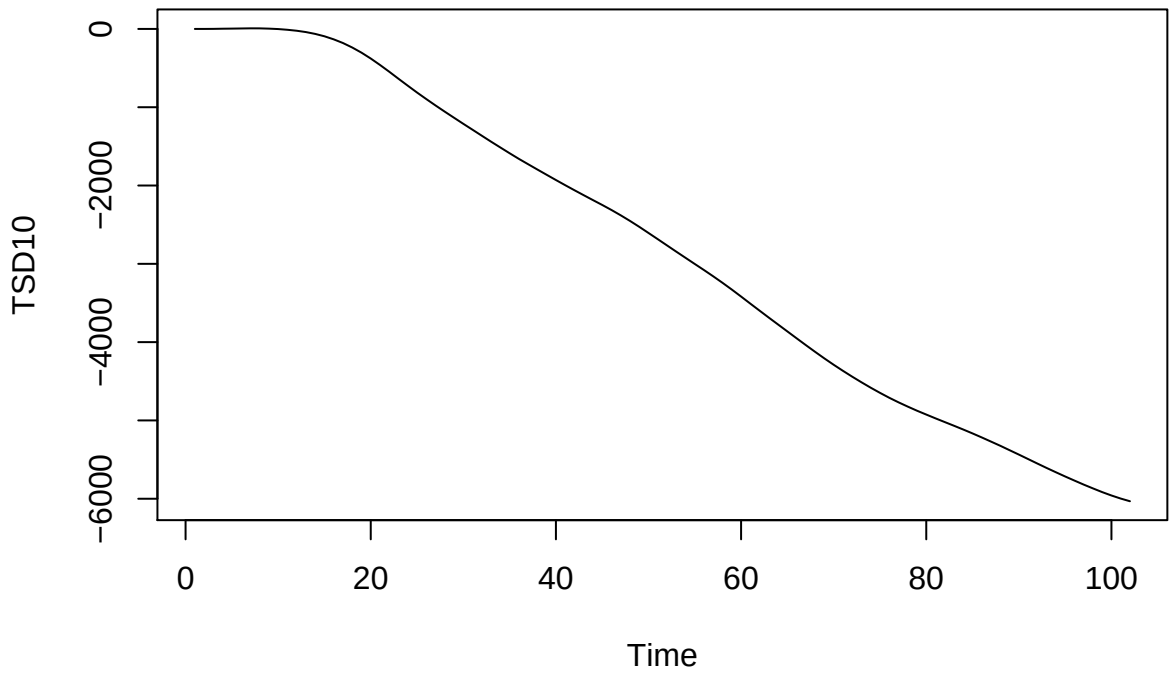

```
pacf(TSD9)
```



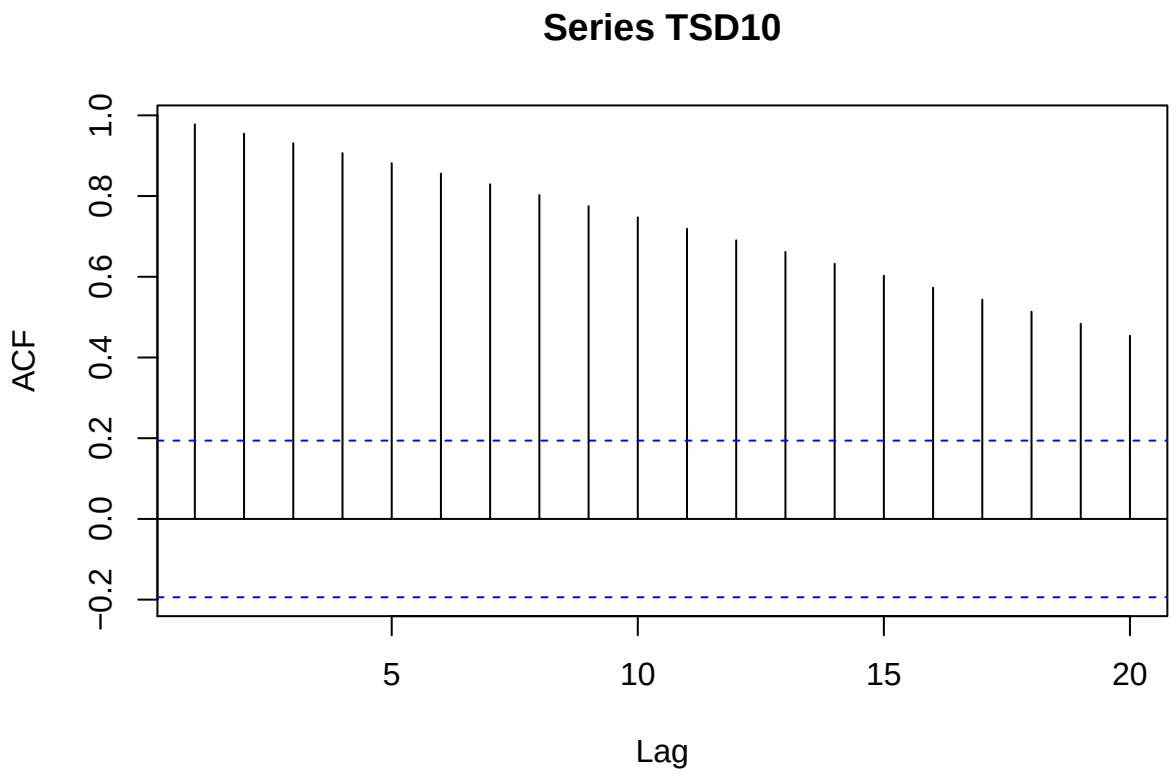

```
arima(TSD9)
```

```
##
## Call:
## arima(x = TSD9)
##
## Coefficients:
##       intercept
##         -0.2035
## s.e.     0.2616
##
## sigma^2 estimated as 6.846:  log likelihood = -238.08,  aic = 478.15
```

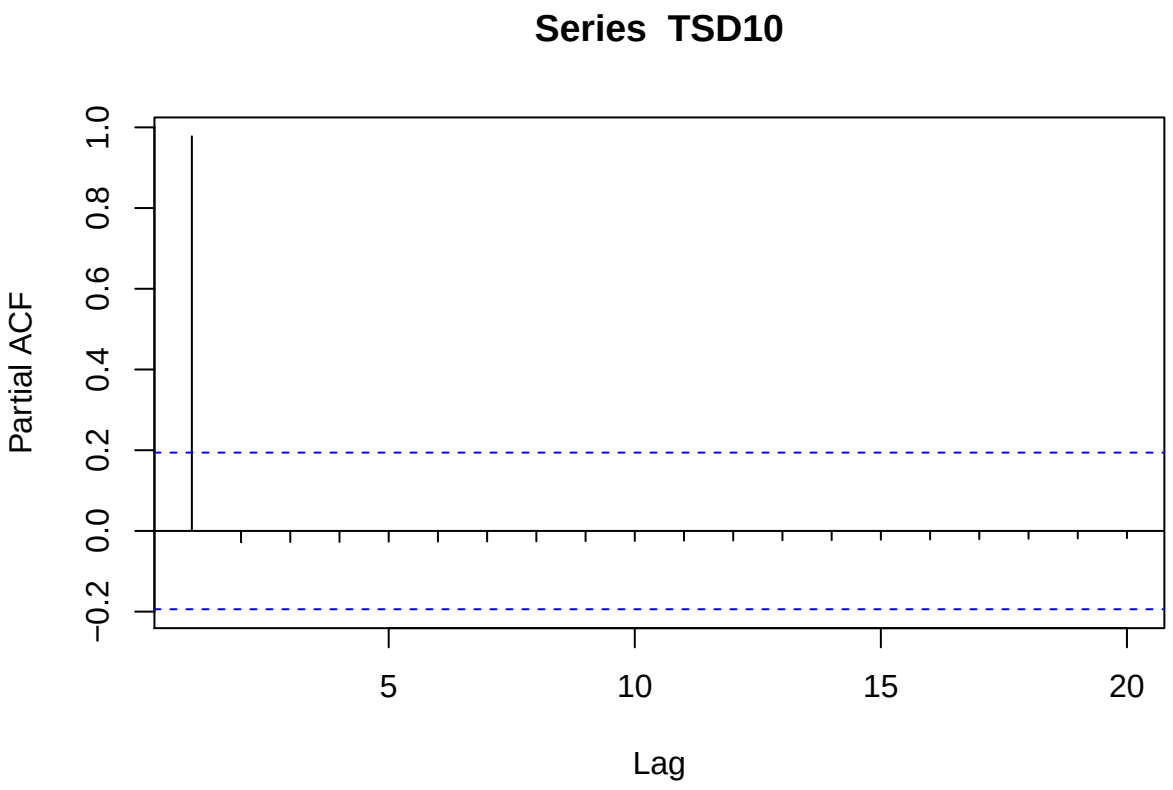
```
#TSD10
mean10<-mean(TSD10$x)
sd10<-sd(TSD10$x)
ts.plot(TSD10)
```




```
acf(TSD10)
```




```
pacf(TSD10)
```




```
arima(TSD10)
```

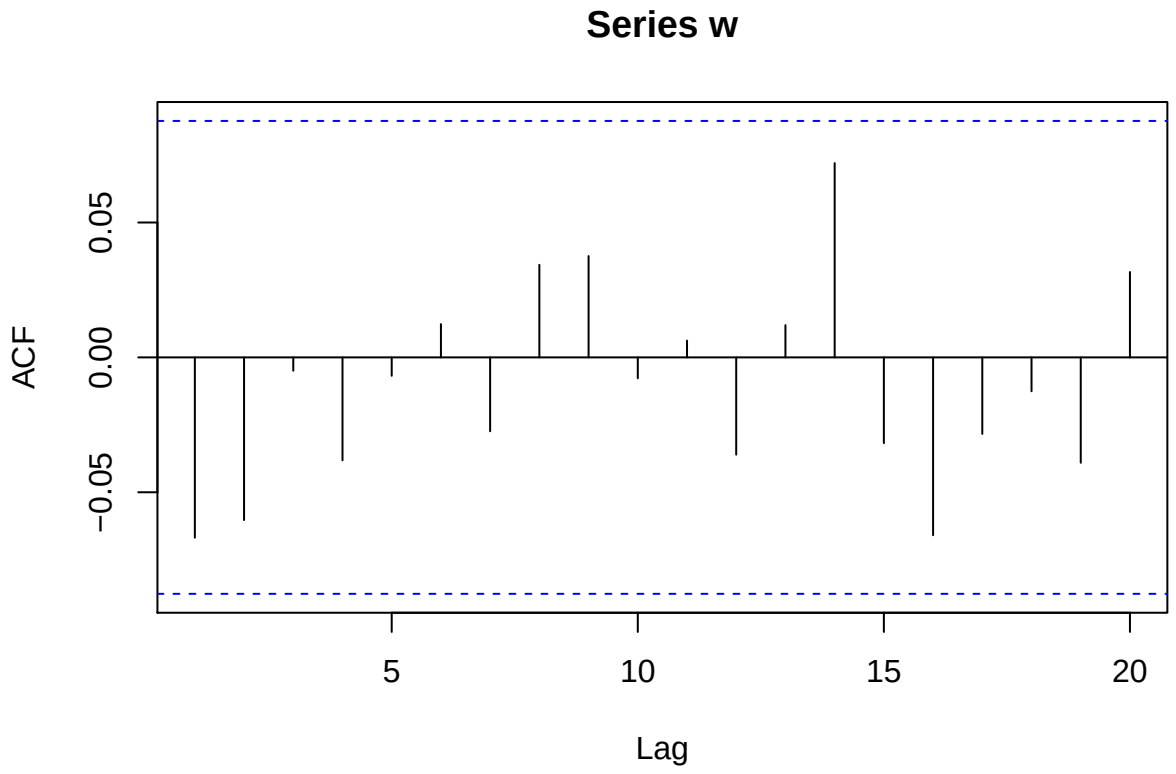
```
##
## Call:
## arima(x = TSD10)
##
## Coefficients:
##         intercept
##       -2808.5121
## s.e.    200.7082
##
## sigma^2 estimated as 4108940:  log likelihood = -921.39,  aic = 1844.79
```

5.2 Question 2:

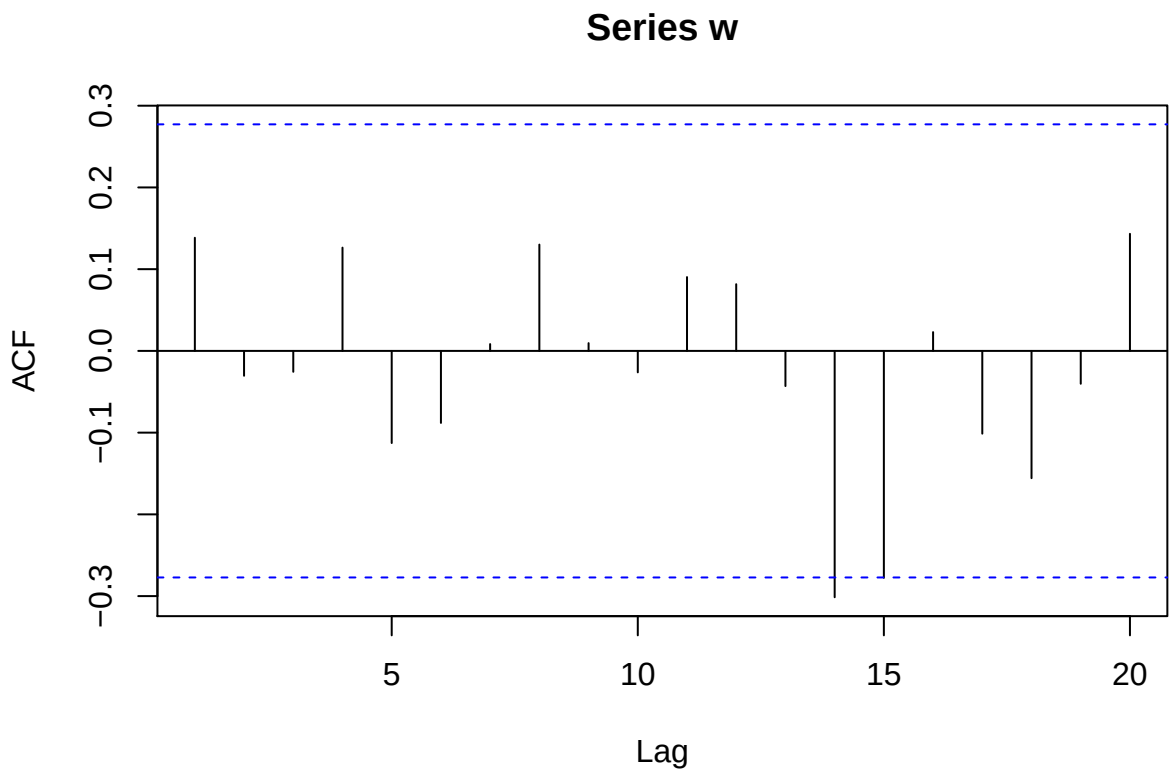
Simulate a series of n = 500 Gaussian white noise observations as in and compute the sample ACF to lag 20. Compare the sample ACF you obtain to the actual ACF. Now repeat the same by using only n = 50. How does changing n affect the results?

```
library(ggplot2)

#Population ACF

LAG = c(0:20) # lag up to 20
acf.p = c(1, rep(0,20)) # population acf (as above)
ACF.df.p = data.frame(lag = LAG, acf = acf.p) # turn the above into data frame
```

```r
# n=500 gaussian white noise ACF:

w = rnorm(500,0,1) # n=500 gaussian white noise
ACF.500 = acf(w, lag.max = 20)
```




```r
ACF.500 = acf(w, lag.max = 20, plot = FALSE)
ACF.df.500 = with(ACF.500, data.frame(lag, acf))

# n=50 gaussian white noise ACF:

w = rnorm(50,0,1) # n=50 gaussian white noise
ACF.50 = acf(w, lag.max = 20)
```




```
ACF.50 = acf(w, lag.max = 20, plot = FALSE)
ACF.df.50 = with(ACF.50, data.frame(lag, acf))

# #combine the acfs (population, n=500, n=50)
# dat = data.frame(rbind(ACF.df.500, ACF.df.50, ACF.df.p),
# n = c(rep("n=500", 21), rep("n=50", 21), rep("population", 21)))
#
# # comparsion:
#
# ggplot(data = dat, mapping = aes(x = lag, y = acf)) +
# geom_hline(aes(yintercept = 0)) +
# facet_grid(n ~ . )+
# geom_segment(mapping = aes(xend = lag, yend = 0))
```

As we can see from above comnparison that as the sample size n gets larger the sample autocorrelation function gets close to its popoulation counterpart.

5.3 Question 3:

Consider the so2 data set, which is part of astsa package. Fit an ARIMA(p, d, q) model to the data, performing all of the necessary diagnostics. After deciding on an appropriate model, forecast the data into the future four time periods ahead (about one month) and calculate 95% prediction intervals for each of the four forecasts.

```
library(astsa, quietly=TRUE, warn.conflicts=FALSE)
require(knitr)
```

```r
library(ggplot2)
#Read in the data
so2
```

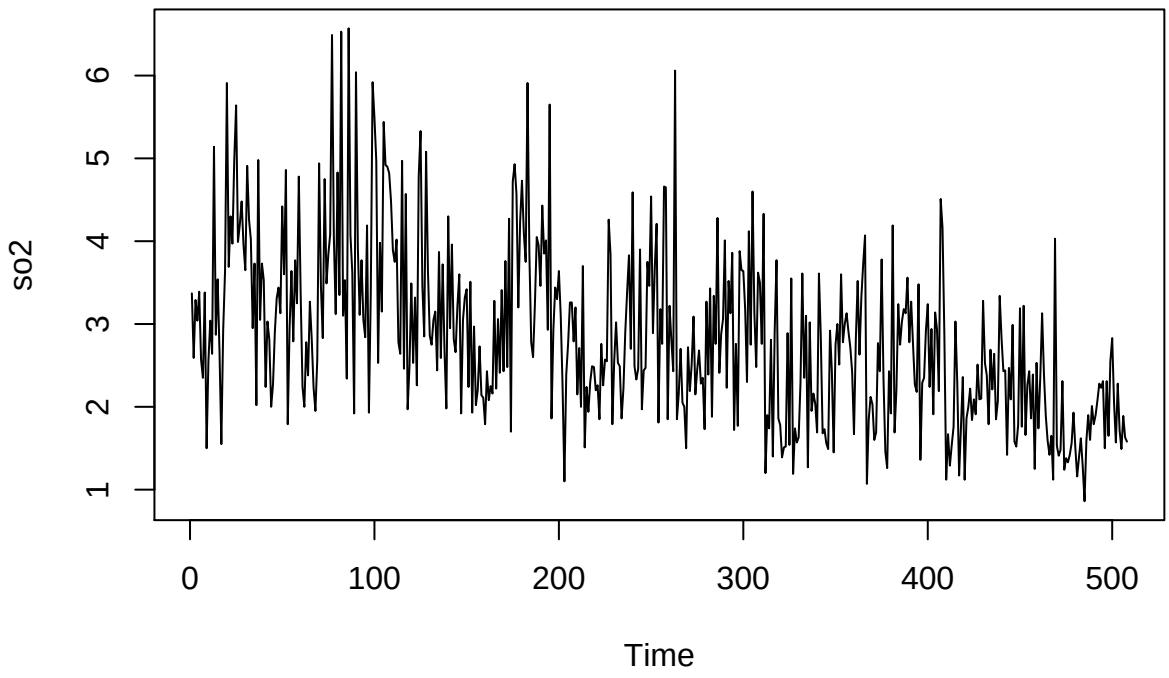
```
## Time Series:
## Start = c(1970, 1)
## End = c(1979, 40)
## Frequency = 52
##   [1] 3.37 2.59 3.29 3.04 3.39 2.57 2.35 3.38 1.50 2.56 3.04 2.64 5.14 2.87
##  [15] 3.54 2.67 1.55 2.93 3.64 5.91 3.69 4.30 3.97 4.98 5.64 3.99 4.20 4.48
##  [29] 3.96 3.65 4.91 4.26 4.02 2.95 3.73 2.02 4.98 3.05 3.73 3.53 2.24 3.03
##  [43] 2.81 2.00 2.27 2.90 3.31 3.44 3.13 4.42 3.60 4.86 1.79 2.95 3.64 2.79
##  [57] 3.77 3.25 4.78 3.37 2.24 2.00 2.78 2.38 3.27 2.82 2.23 1.95 2.57 4.94
##  [71] 3.45 2.83 4.75 3.49 3.84 4.08 6.49 4.05 3.12 4.83 3.35 6.53 3.10 3.53
##  [85] 2.34 6.57 4.07 3.62 1.92 6.04 3.92 3.11 3.77 3.05 2.84 4.19 1.93 3.36
##  [99] 5.92 5.49 4.97 2.53 3.98 3.15 5.44 4.92 4.90 4.82 4.48 3.89 3.75 4.02
## [113] 2.78 2.64 4.97 2.46 4.57 1.97 2.49 3.49 2.53 3.32 2.26 4.78 5.33 3.46
## [127] 2.85 5.08 3.65 2.86 2.75 3.05 3.15 2.44 3.87 2.59 3.72 2.67 1.98 4.30
## [141] 2.95 3.96 2.82 2.66 3.22 3.60 1.92 3.05 3.32 3.42 2.24 3.51 1.93 2.97
## [155] 2.02 2.20 2.73 2.14 2.11 1.79 2.43 2.08 2.25 2.16 3.28 2.22 3.06 2.41
## [169] 3.41 2.43 3.76 2.48 4.27 1.70 4.72 4.93 4.57 3.20 4.24 4.73 4.10 3.75
## [183] 5.91 3.85 2.78 2.60 3.23 4.05 3.95 3.46 4.43 3.85 4.01 2.93 5.65 1.86
## [197] 2.92 3.44 3.30 3.64 3.10 2.25 1.10 2.39 2.76 3.26 3.26 2.79 3.20 2.15
## [211] 2.71 2.00 3.70 1.51 2.24 1.94 2.30 2.49 2.48 2.20 2.26 1.85 2.76 2.26
## [225] 2.57 2.55 4.26 3.84 1.79 2.55 3.02 2.53 2.49 1.86 2.22 2.90 3.39 3.83
## [239] 2.70 4.59 2.49 2.33 2.46 3.90 1.97 2.44 2.47 3.75 3.46 4.54 2.89 3.68
## [253] 4.21 1.81 3.18 2.76 4.66 4.65 1.85 3.22 2.80 2.43 6.06 1.85 2.16 2.70
## [267] 2.05 2.00 1.50 2.72 2.19 2.51 3.09 2.15 2.45 2.68 2.28 2.35 1.73 3.27
## [281] 2.39 3.43 1.88 3.34 2.76 4.28 2.41 2.88 3.06 4.01 2.23 3.52 3.13 3.86
## [295] 1.72 2.76 1.77 3.88 3.65 3.64 3.18 2.30 4.12 2.75 4.60 3.13 2.48 3.62
## [309] 3.50 2.76 4.33 1.20 1.90 1.74 2.81 1.40 2.91 3.77 1.86 1.78 1.39 1.51
## [323] 1.52 2.89 1.54 3.55 1.19 1.74 1.57 1.63 2.29 3.61 2.35 3.10 1.27 3.02
## [337] 1.95 2.16 2.04 1.69 3.61 2.85 1.68 1.73 1.56 1.49 2.92 2.43 1.45 2.75
## [351] 3.00 2.51 3.60 2.78 3.00 3.13 2.91 2.72 2.42 1.67 2.81 3.52 2.63 3.33
## [365] 3.73 4.07 1.07 1.85 2.12 2.03 1.60 1.69 2.77 2.43 3.78 2.33 1.46 1.26
## [379] 2.43 1.92 4.19 1.69 2.21 3.24 2.75 3.03 3.18 3.13 3.56 2.78 3.27 2.76
## [393] 2.27 2.18 3.48 1.36 2.29 2.35 2.92 3.24 2.24 2.94 1.91 3.14 2.94 2.19
## [407] 4.51 4.16 2.87 1.12 1.67 1.29 1.54 1.76 3.03 2.29 1.17 1.75 2.36 1.12
## [421] 1.86 2.00 2.22 1.84 2.09 1.91 2.51 2.09 2.10 3.28 2.53 2.38 1.79 2.69
## [435] 2.21 2.64 1.85 2.07 3.34 2.83 2.43 2.44 1.42 2.47 2.09 2.99 1.58 1.52
## [449] 1.78 3.19 1.76 3.22 1.66 2.26 2.43 1.86 2.39 1.25 2.53 1.74 2.36 3.13
## [463] 2.39 1.89 1.59 1.42 1.65 1.12 4.03 1.53 1.41 1.48 2.31 1.24 1.38 1.33
## [477] 1.42 1.56 1.93 1.51 1.16 1.39 1.62 1.29 0.86 1.59 1.90 1.60 2.01 1.79
## [491] 1.90 2.08 2.28 2.23 2.31 1.50 2.31 1.65 2.55 2.83 2.05 1.57 2.28 1.72
## [505] 1.49 1.89 1.63 1.58
```

```r
#Convert the data into a time series
so2 <- ts(so2)
so2
```

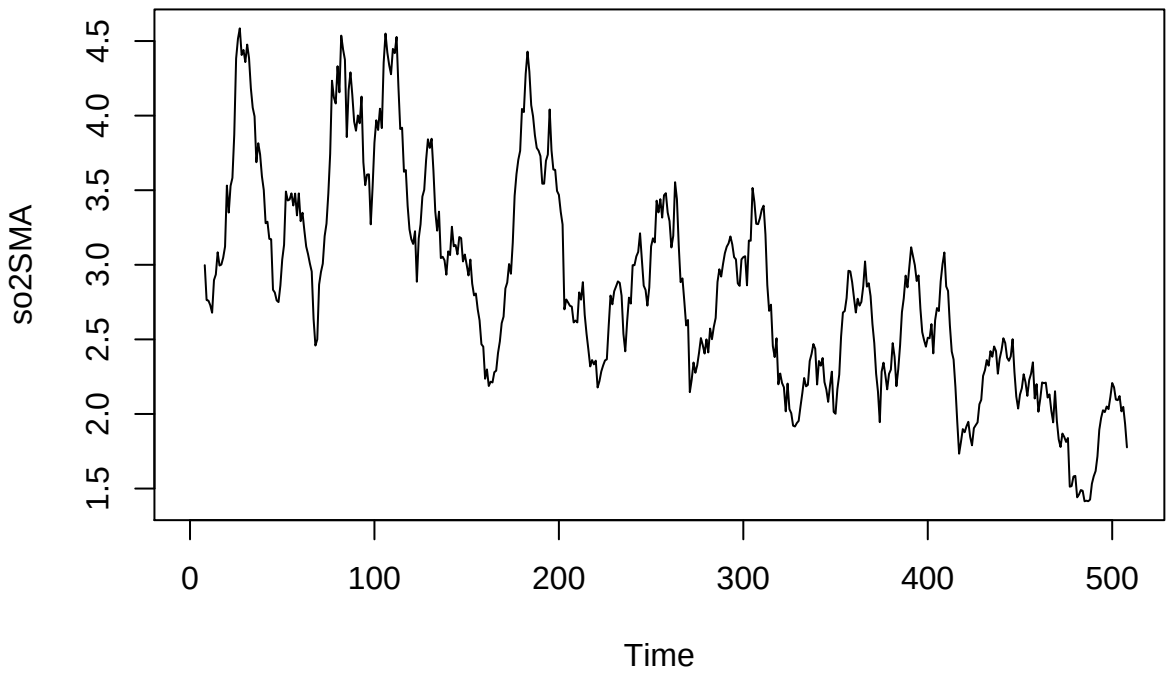
```
## Time Series:
## Start = 1
## End = 508
## Frequency = 1
##   [1] 3.37 2.59 3.29 3.04 3.39 2.57 2.35 3.38 1.50 2.56 3.04 2.64 5.14 2.87
```

```
##  [15] 3.54 2.67 1.55 2.93 3.64 5.91 3.69 4.30 3.97 4.98 5.64 3.99 4.20 4.48
##  [29] 3.96 3.65 4.91 4.26 4.02 2.95 3.73 2.02 4.98 3.05 3.73 3.53 2.24 3.03
##  [43] 2.81 2.00 2.27 2.90 3.31 3.44 3.13 4.42 3.60 4.86 1.79 2.95 3.64 2.79
##  [57] 3.77 3.25 4.78 3.37 2.24 2.00 2.78 2.38 3.27 2.82 2.23 1.95 2.57 4.94
##  [71] 3.45 2.83 4.75 3.49 3.84 4.08 6.49 4.05 3.12 4.83 3.35 6.53 3.10 3.53
##  [85] 2.34 6.57 4.07 3.62 1.92 6.04 3.92 3.11 3.77 3.05 2.84 4.19 1.93 3.36
##  [99] 5.92 5.49 4.97 2.53 3.98 3.15 5.44 4.92 4.90 4.82 4.48 3.89 3.75 4.02
## [113] 2.78 2.64 4.97 2.46 4.57 1.97 2.49 3.49 2.53 3.32 2.26 4.78 5.33 3.46
## [127] 2.85 5.08 3.65 2.86 2.75 3.05 3.15 2.44 3.87 2.59 3.72 2.67 1.98 4.30
## [141] 2.95 3.96 2.82 2.66 3.22 3.60 1.92 3.05 3.32 3.42 2.24 3.51 1.93 2.97
## [155] 2.02 2.20 2.73 2.14 2.11 1.79 2.43 2.08 2.25 2.16 3.28 2.22 3.06 2.41
## [169] 3.41 2.43 3.76 2.48 4.27 1.70 4.72 4.93 4.57 3.20 4.24 4.73 4.10 3.75
## [183] 5.91 3.85 2.78 2.60 3.23 4.05 3.95 3.46 4.43 3.85 4.01 2.93 5.65 1.86
## [197] 2.92 3.44 3.30 3.64 3.10 2.25 1.10 2.39 2.76 3.26 3.26 2.79 3.20 2.15
## [211] 2.71 2.00 3.70 1.51 2.24 1.94 2.30 2.49 2.48 2.20 2.26 1.85 2.76 2.26
## [225] 2.57 2.55 4.26 3.84 1.79 2.55 3.02 2.53 2.49 1.86 2.22 2.90 3.39 3.83
## [239] 2.70 4.59 2.49 2.33 2.46 3.90 1.97 2.44 2.47 3.75 3.46 4.54 2.89 3.68
## [253] 4.21 1.81 3.18 2.76 4.66 4.65 1.85 3.22 2.80 2.43 6.06 1.85 2.16 2.70
## [267] 2.05 2.00 1.50 2.72 2.19 2.51 3.09 2.15 2.45 2.68 2.28 2.35 1.73 3.27
## [281] 2.39 3.43 1.88 3.34 2.76 4.28 2.41 2.88 3.06 4.01 2.23 3.52 3.13 3.86
## [295] 1.72 2.76 1.77 3.88 3.65 3.64 3.18 2.30 4.12 2.75 4.60 3.13 2.48 3.62
## [309] 3.50 2.76 4.33 1.20 1.90 1.74 2.81 1.40 2.91 3.77 1.86 1.78 1.39 1.51
## [323] 1.52 2.89 1.54 3.55 1.19 1.74 1.57 1.63 2.29 3.61 2.35 3.10 1.27 3.02
## [337] 1.95 2.16 2.04 1.69 3.61 2.85 1.68 1.73 1.56 1.49 2.92 2.43 1.45 2.75
## [351] 3.00 2.51 3.60 2.78 3.00 3.13 2.91 2.72 2.42 1.67 2.81 3.52 2.63 3.33
## [365] 3.73 4.07 1.07 1.85 2.12 2.03 1.60 1.69 2.77 2.43 3.78 2.33 1.46 1.26
## [379] 2.43 1.92 4.19 1.69 2.21 3.24 2.75 3.03 3.18 3.13 3.56 2.78 3.27 2.76
## [393] 2.27 2.18 3.48 1.36 2.29 2.35 2.92 3.24 2.24 2.94 1.91 3.14 2.94 2.19
## [407] 4.51 4.16 2.87 1.12 1.67 1.29 1.54 1.76 3.03 2.29 1.17 1.75 2.36 1.12
## [421] 1.86 2.00 2.22 1.84 2.09 1.91 2.51 2.09 2.10 3.28 2.53 2.38 1.79 2.69
## [435] 2.21 2.64 1.85 2.07 3.34 2.83 2.43 2.44 1.42 2.47 2.09 2.99 1.58 1.52
## [449] 1.78 3.19 1.76 3.22 1.66 2.26 2.43 1.86 2.39 1.25 2.53 1.74 2.36 3.13
## [463] 2.39 1.89 1.59 1.42 1.65 1.12 4.03 1.53 1.41 1.48 2.31 1.24 1.38 1.33
## [477] 1.42 1.56 1.93 1.51 1.16 1.39 1.62 1.29 0.86 1.59 1.90 1.60 2.01 1.79
## [491] 1.90 2.08 2.28 2.23 2.31 1.50 2.31 1.65 2.55 2.83 2.05 1.57 2.28 1.72
## [505] 1.49 1.89 1.63 1.58
```

```r
#plot the data
plot.ts(so2)
```




```
#decomposing non-seasonal data(smoothing)
library(TTR)
so2SMA <- SMA(so2, n=8) #moving average of n=8
plot.ts(so2SMA)
```




```
#Forecast:
so2F <-  HoltWinters(so2, gamma = F)
so2F
```

```
## Holt-Winters exponential smoothing with trend and without seasonal component.
##
## Call:
## HoltWinters(x = so2, gamma = F)
##
## Smoothing parameters:
##  alpha: 0.2844364
##  beta : 0.1568771
##  gamma: FALSE
##
## Coefficients:
##          [,1]
## a  1.69689149
## b -0.04985262
```

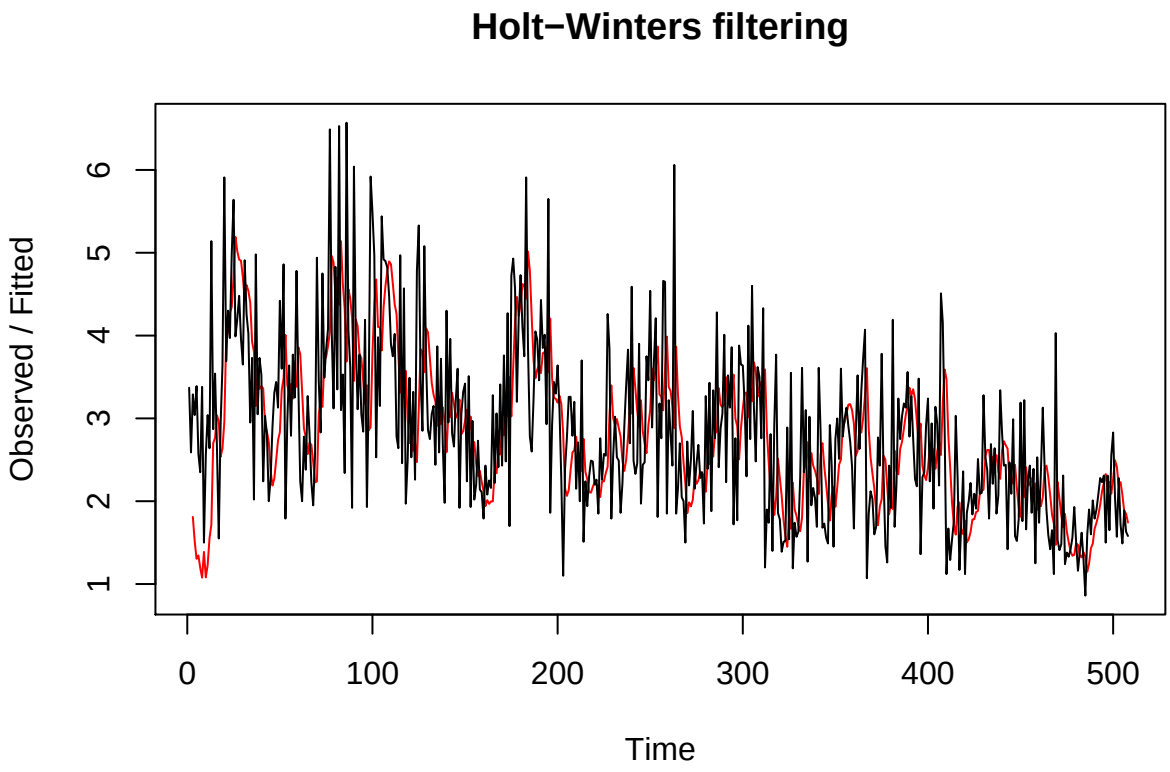
```
so2F; so2F$SSE
```

```
## Holt-Winters exponential smoothing with trend and without seasonal component.
##
## Call:
## HoltWinters(x = so2, gamma = F)
##
## Smoothing parameters:
```

```
##  alpha: 0.2844364
##  beta : 0.1568771
##  gamma: FALSE
##
## Coefficients:
##          [,1]
## a  1.69689149
## b -0.04985262
```

```
## [1] 469.5215
```

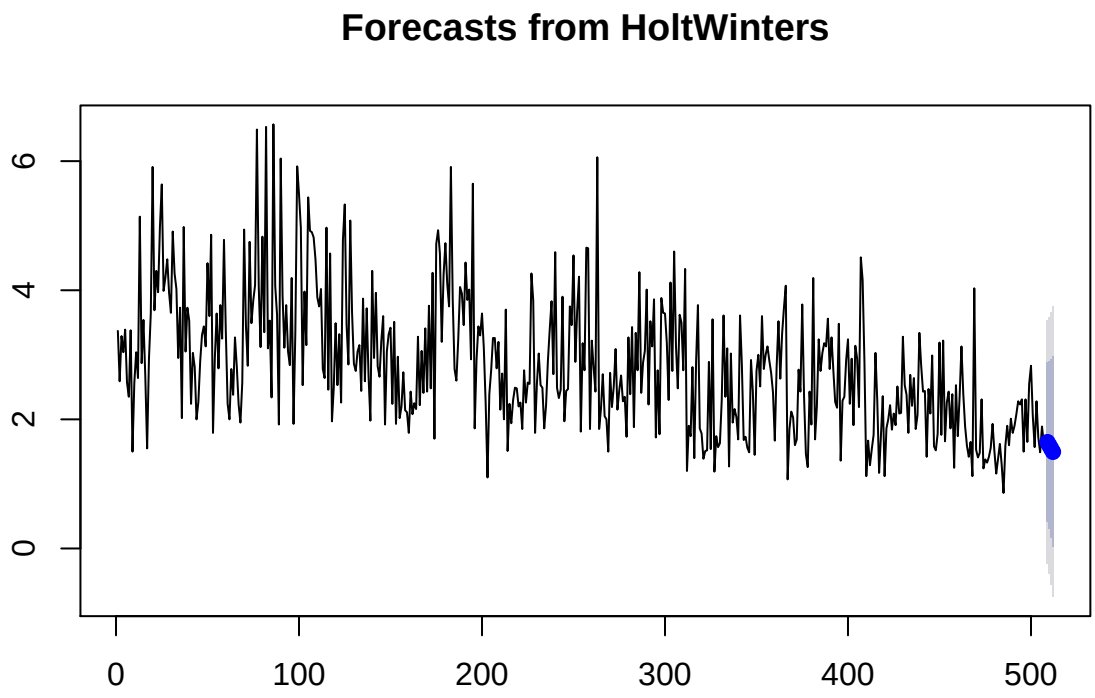
```
plot(so2F) #observed v/s fitted
```



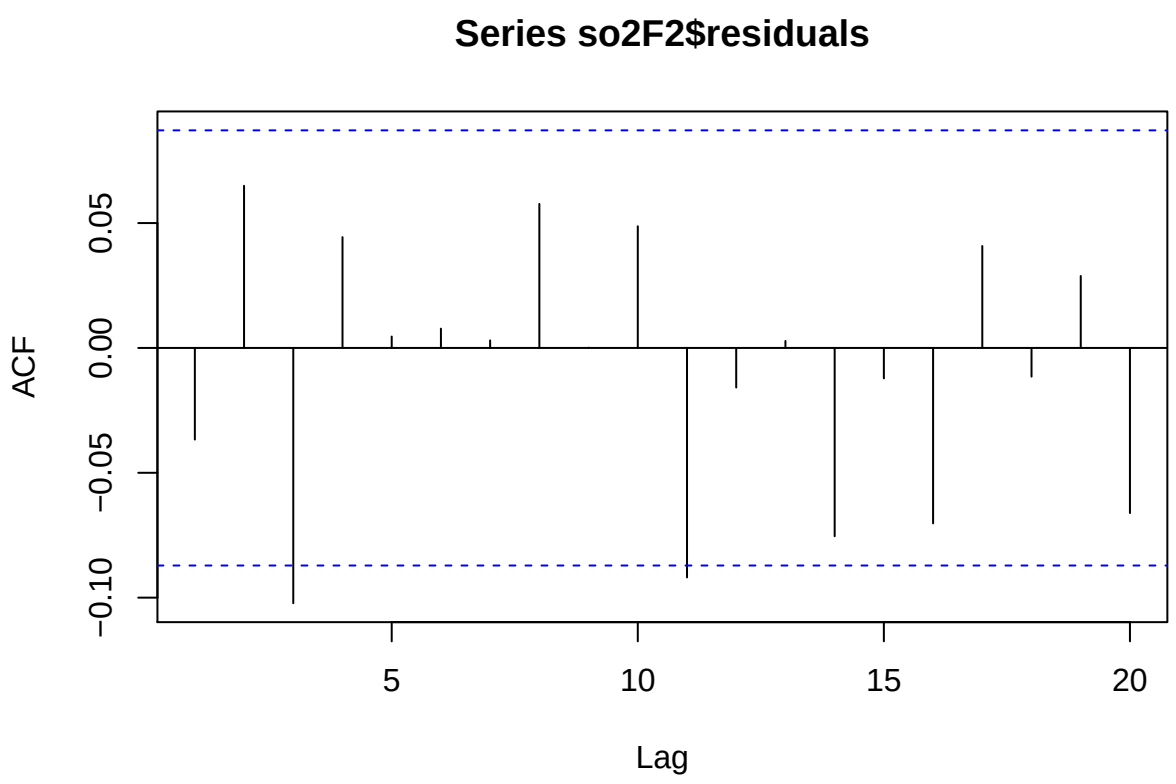

```
#Forecast for 4 time periods
library("forecast")
so2F2 <- forecast:::forecast.HoltWinters(so2F, h=4)
so2F2
```

```
##     Point Forecast      Lo 80    Hi 80      Lo 95    Hi 95
## 509       1.647039 0.41202091 2.882057 -0.2417581 3.535836
## 510       1.597186 0.29702309 2.897349 -0.3912417 3.585614
## 511       1.547334 0.16769332 2.926974 -0.5626441 3.657311
## 512       1.497481 0.02428969 2.970672 -0.7555707 3.750533
```

```
plot(forecast(so2F2))
```



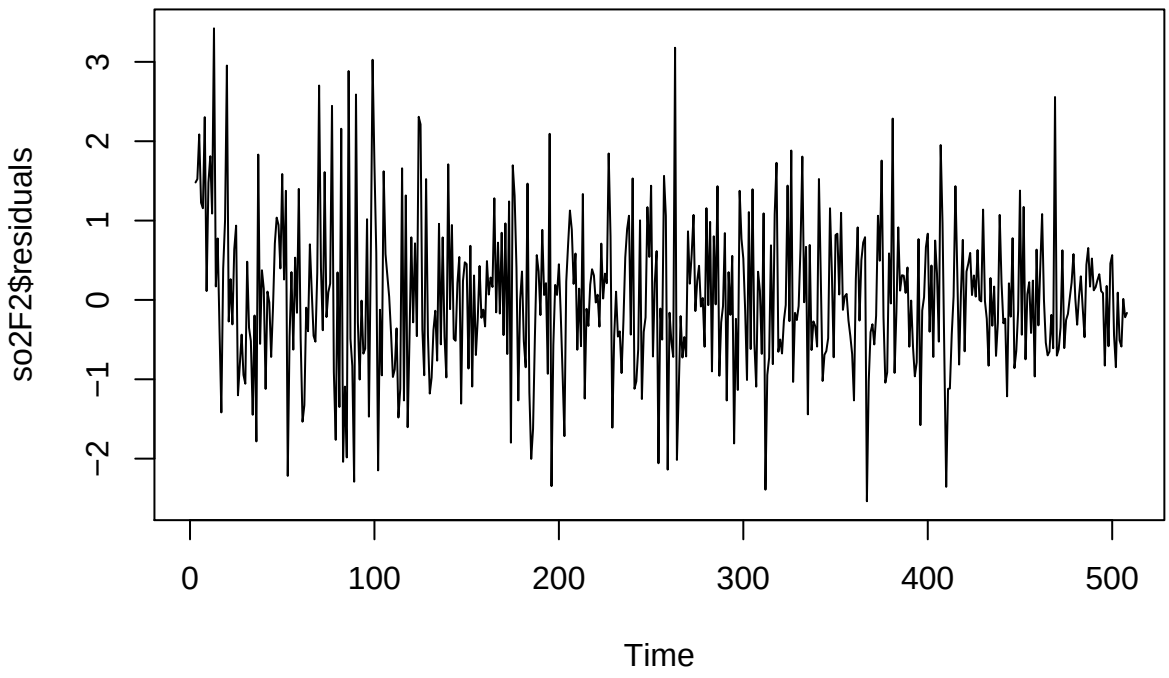

```
acf(so2F2$residuals, lag.max=20, na.action = na.omit)
```




```
Box.test(so2F2$residuals, lag=20, type='Ljung-Box')
```

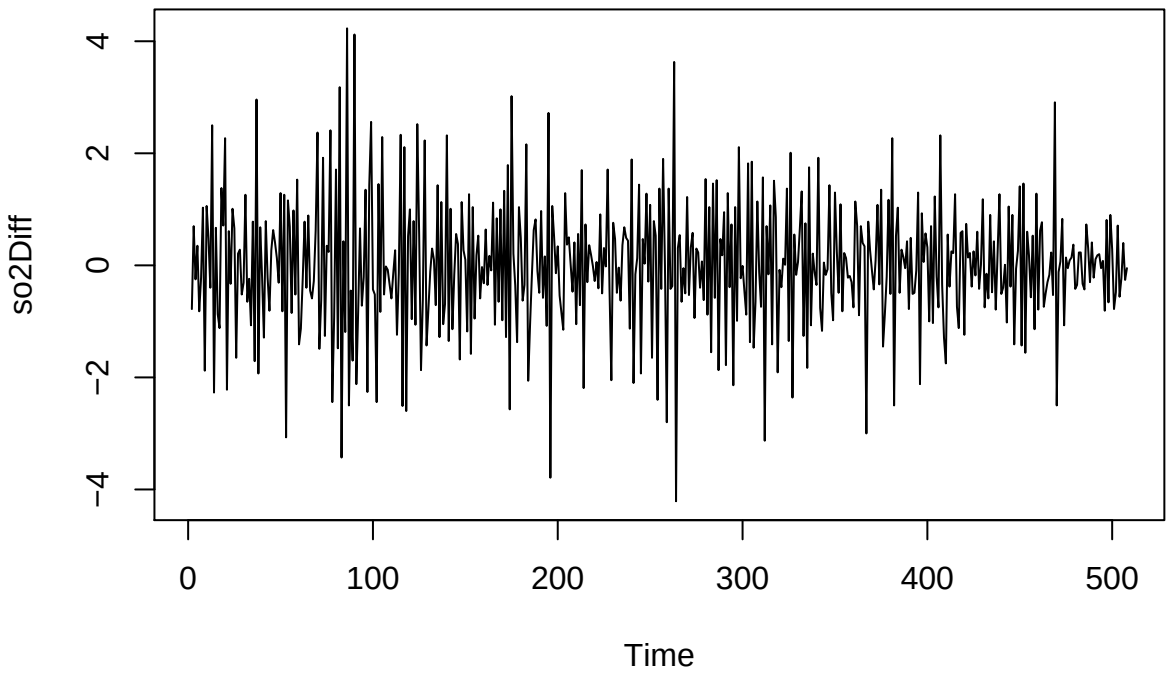
```
##
##  Box-Ljung test
##
## data:  so2F2$residuals
## X-squared = 26.038, df = 20, p-value = 0.1645
```

```
plot.ts(so2F2$residuals)
```



ARIMA model :

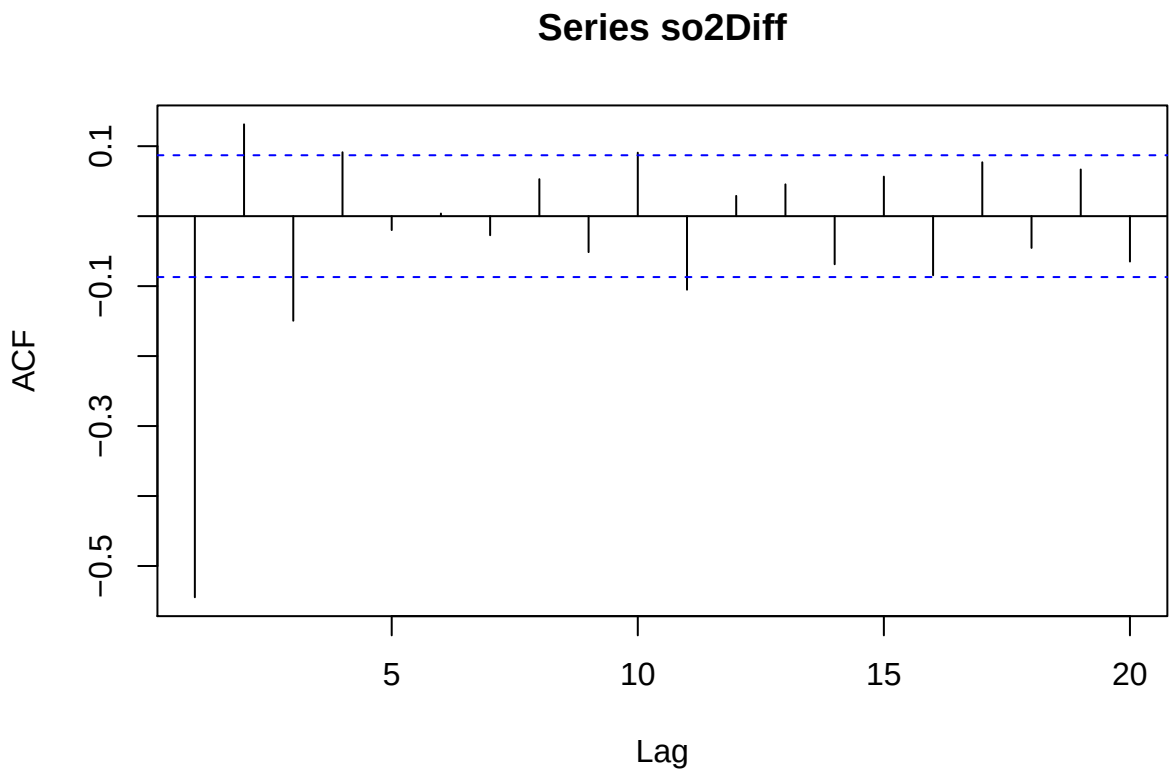
```
#Differencing a time series
#First differencing
so2Diff <- diff(so2, differences = 1)
plot.ts(so2Diff)
```




```
#The time series of first differences (above) does appear to be stationary
#in mean and variance, as the level of the series stays roughly constant
#over time, and the variance of the series appears roughly constant over time.

#selecting a candidate ARIMA Model
# Install the tools
source(url("http://lib.stat.cmu.edu/general/tsa2/Rcode/itall.R"))
```

```
##   itall has been installed
```

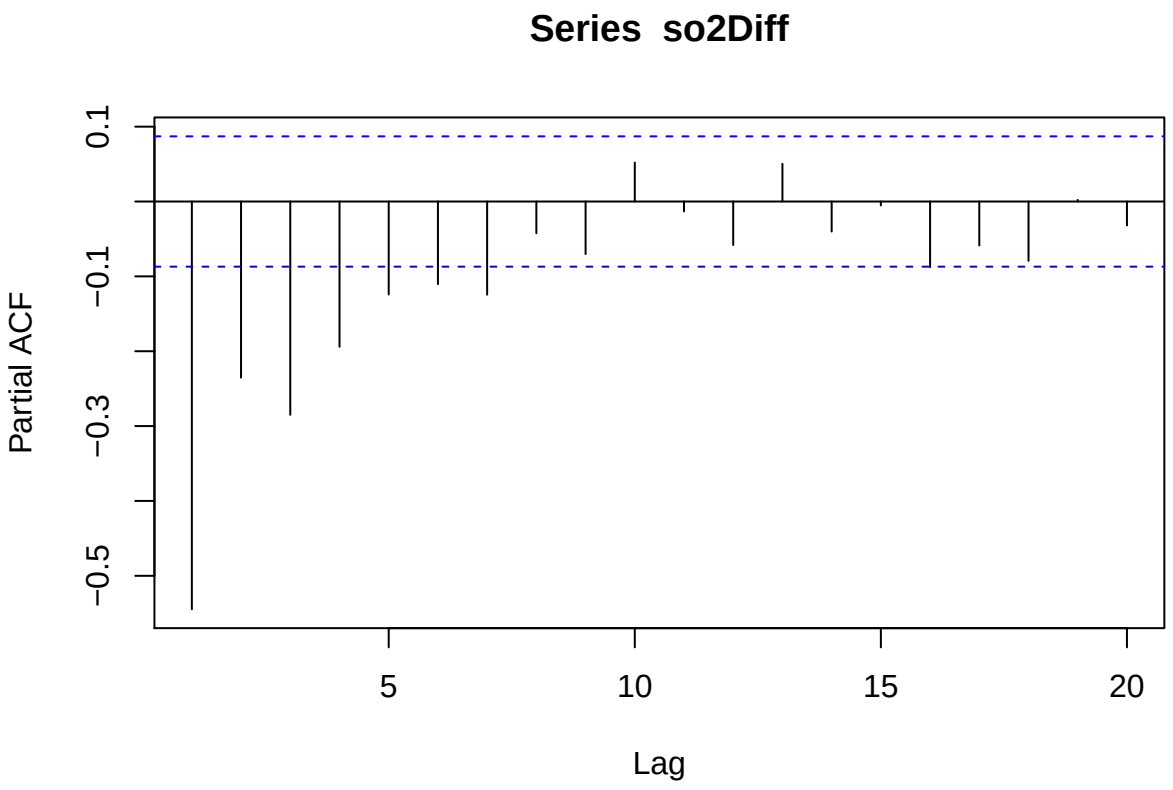
```
# Plot ACF and PACF
acf(so2Diff, lag.max=20)             # plot a correlogram
```




```
acf(so2Diff, lag.max=20, plot=FALSE) # get the autocorrelation values
```

```
##
## Autocorrelations of series 'so2Diff', by lag
##
##      1      2      3      4      5      6      7      8      9     10
## -0.545  0.131 -0.149  0.091 -0.020  0.004 -0.027  0.053 -0.051  0.091
##     11     12     13     14     15     16     17     18     19     20
## -0.105  0.029  0.045 -0.069  0.056 -0.084  0.077 -0.045  0.067 -0.065
```

```
pacf(so2Diff, lag.max=20)             # plot a partial correlogram
```




```
pacf(so2Diff, lag.max=20, plot=FALSE) # get the partial autocorrelation values
```

```
##
## Partial autocorrelations of series 'so2Diff', by lag
##
##      1      2      3      4      5      6      7      8      9     10
## -0.545 -0.235 -0.285 -0.194 -0.124 -0.110 -0.125 -0.042 -0.070  0.052
##     11     12     13     14     15     16     17     18     19     20
## -0.013 -0.058  0.050 -0.040 -0.005 -0.088 -0.059 -0.079  0.002 -0.032
```

```
library(forecast)
auto.arima(so2)
```

```
## Series: so2
## ARIMA(1,1,2)
##
## Coefficients:
##           ar1      ma1      ma2
##       -0.8146  -0.0705  -0.6178
## s.e.   0.1122   0.1250   0.1108
##
## sigma^2 estimated as 0.7809:  log likelihood=-655.81
## AIC=1319.63   AICc=1319.71   BIC=1336.54
```

Forecasting Using an ARIMA Model:

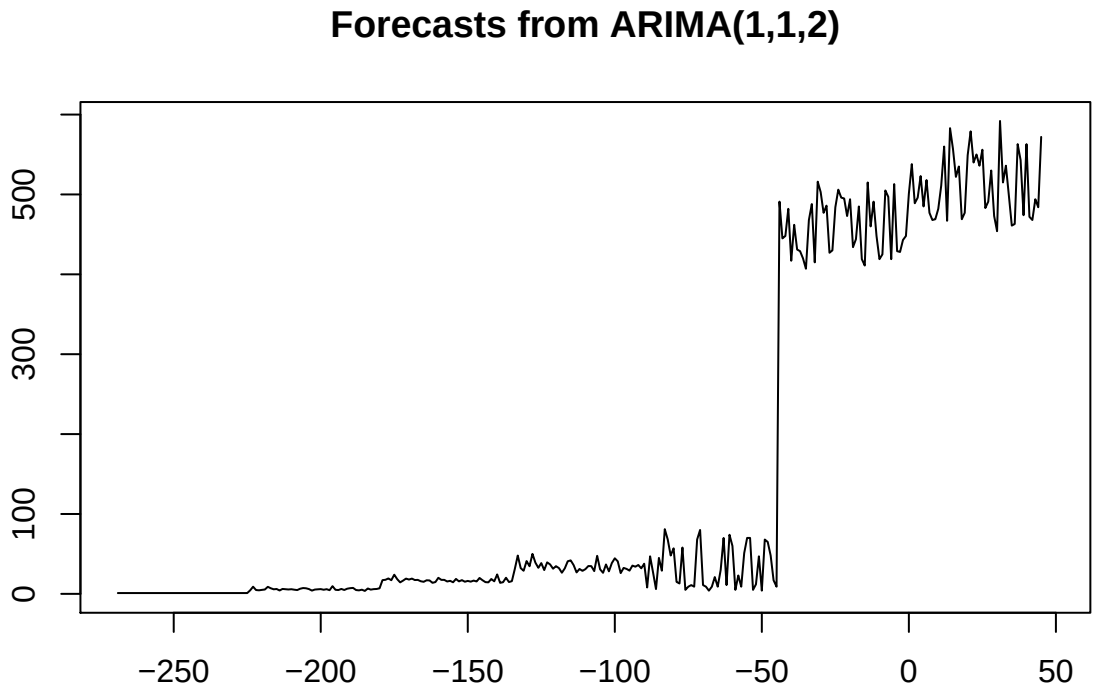
```
so2arima <- arima(so2, order=c(1,1,2)) # fit an ARIMA(1,1,2) model
so2arima
```

```
##
## Call:
## arima(x = so2, order = c(1, 1, 2))
##
## Coefficients:
##           ar1      ma1      ma2
##       -0.8146  -0.0705  -0.6178
## s.e.   0.1122   0.1250   0.1108
##
## sigma^2 estimated as 0.7763:  log likelihood = -655.81,  aic = 1317.63
```

```
library("forecast") # load the "forecast" R library
so2forecasts <- forecast:::forecast.Arima(so2arima, h=4, level=c(95))
so2forecasts
```

```
##     Point Forecast      Lo 95    Hi 95
## 509       1.827613 0.10075314 3.554474
## 510       1.811609 0.07337157 3.549846
## 511       1.824646 0.04608489 3.603206
## 512       1.814026 0.02045457 3.607598
```

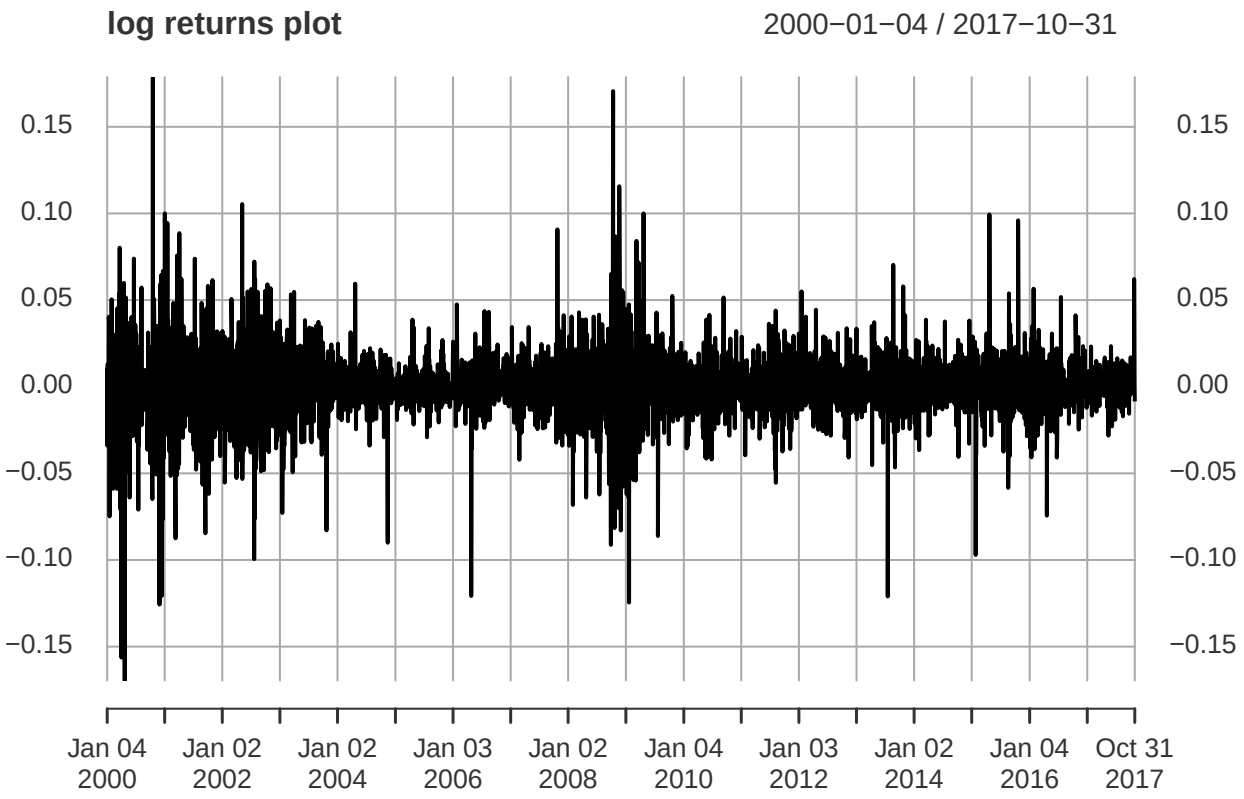
```
plot(forecast(so2forecasts))
```



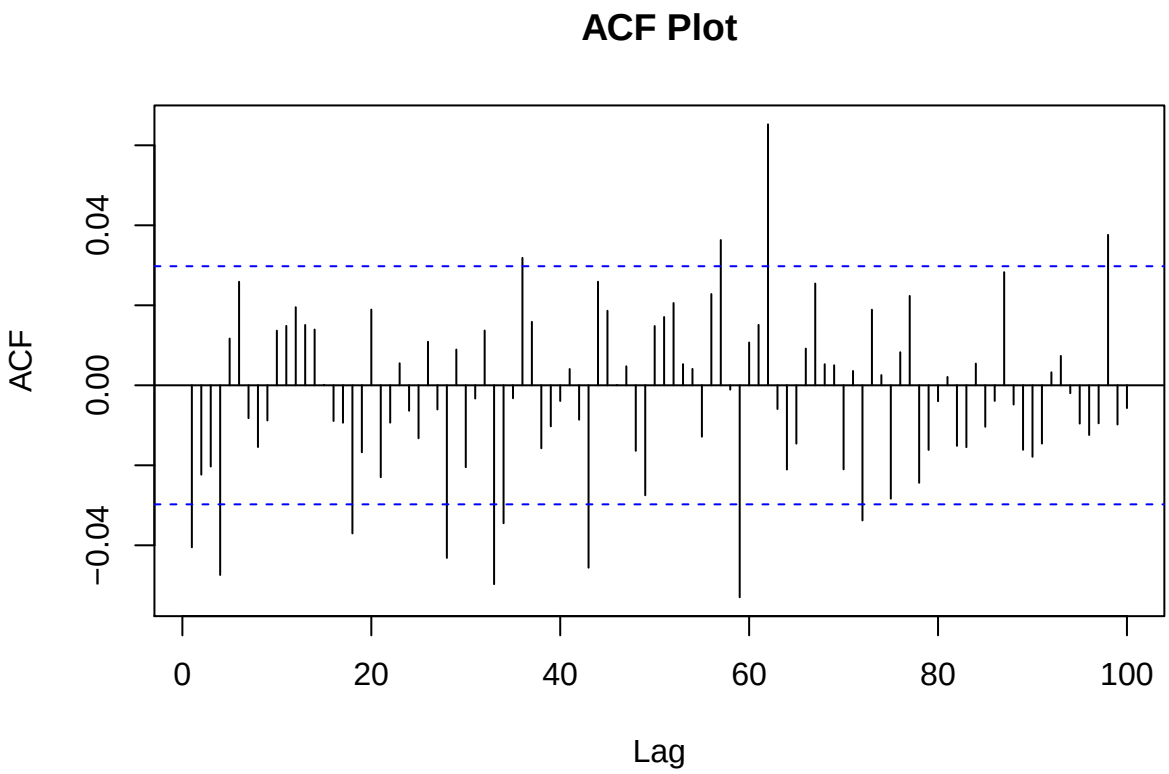
5.4 Question 4:

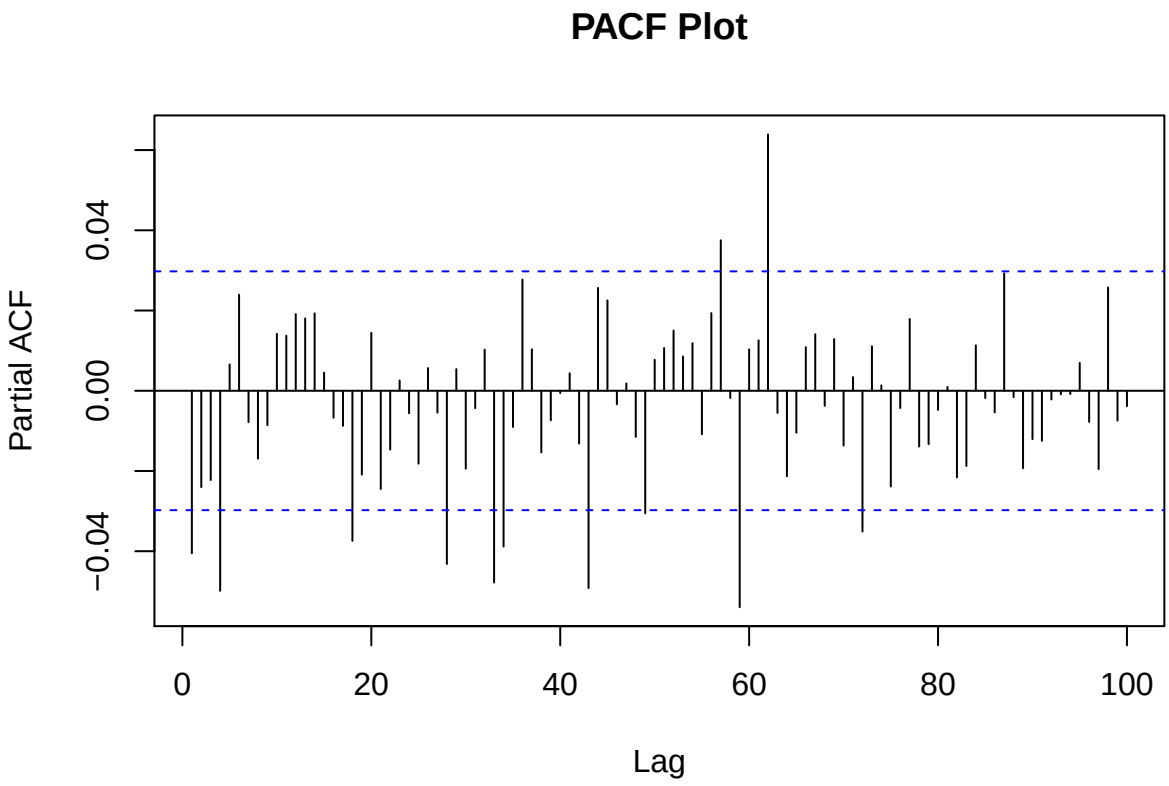
Perform a time series model specification, estimation, model diagnostics, and forecasting for a financial stock of your choice.

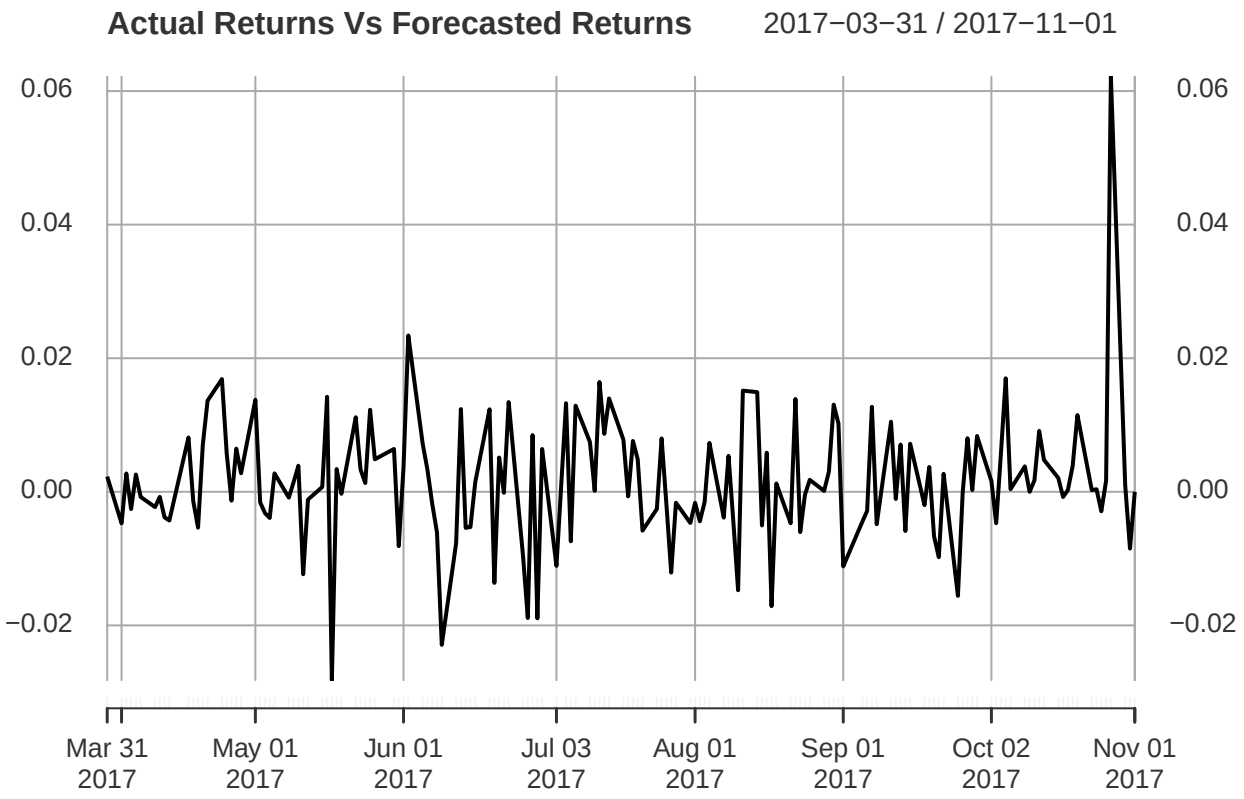
```
## [1] "MSFT"
```

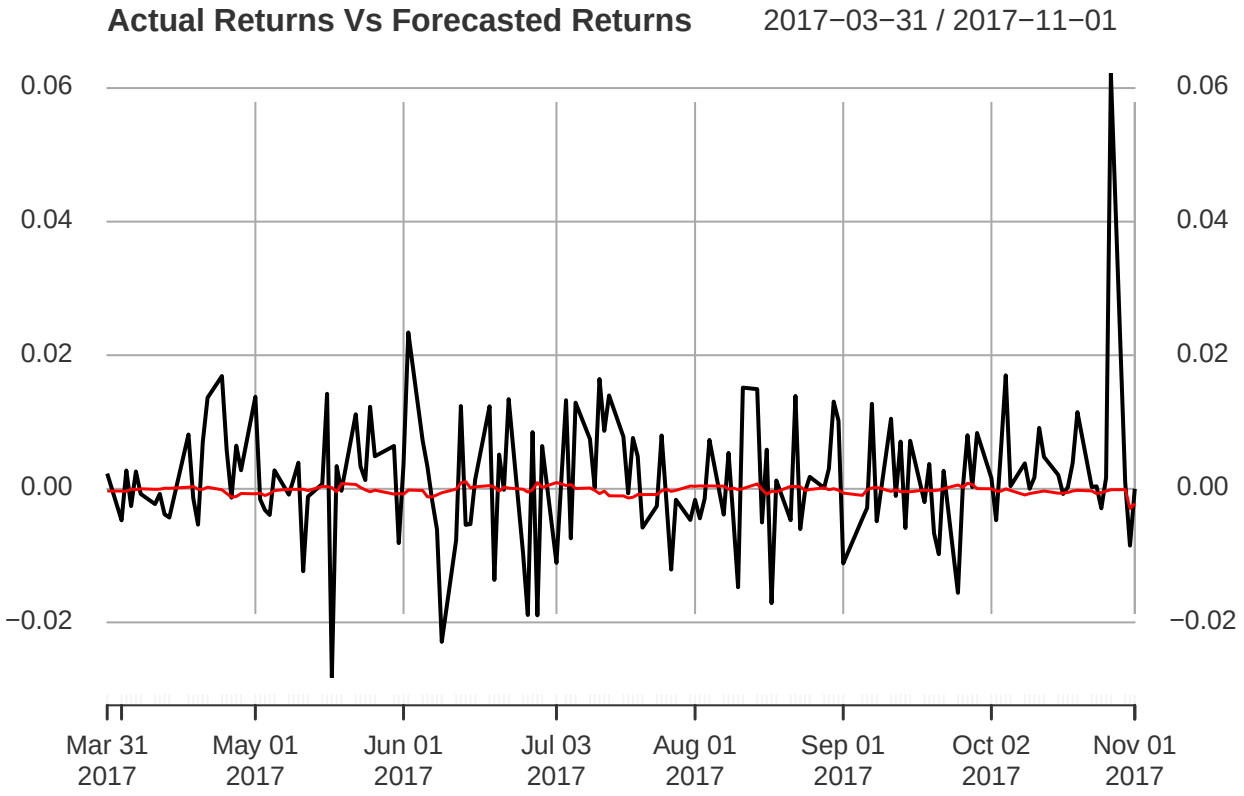



```
##
##  Augmented Dickey-Fuller Test
##
## data:  stock
## Dickey-Fuller = -16.022, Lag order = 16, p-value = 0.01
## alternative hypothesis: stationary
```


ACF Plot
ACF
0.04
0.00
−0.04
0
20
40
60
80
100
Lag


PACF Plot
Partial ACF
0.04
0.00
−0.04
0
20
40
60
80
100
Lag


Actual Returns Vs Forecasted Returns
2017−03−31 / 2017−11−01
0.06
0.04
0.02
0.00
−0.02
Mar 31 2017
May 01 2017
Jun 01 2017
Jul 03 2017
Aug 01 2017
Sep 01 2017
Oct 02 2017
Nov 01 2017




```
##            Actual_series    Forecasted Accuracy
## 2017-03-31  0.0022801864 -3.289748e-04        0
## 2017-04-03 -0.0047180362 -3.365804e-04        1
## 2017-04-04  0.0027422319 -3.794319e-04        0
## 2017-04-05 -0.0025897645 -2.483940e-05        1
## 2017-04-06  0.0025897645 -1.314576e-04        0
## 2017-04-07 -0.0007610228  3.494051e-06        0
## 2017-04-10 -0.0022864274 -7.803956e-05        1
## 2017-04-11 -0.0007632395 -5.979131e-05        1
## 2017-04-12 -0.0038252665  9.327934e-05        0
## 2017-04-13 -0.0043018349  7.877919e-05        0
## 2017-04-17  0.0081271014  2.335788e-04        1
## 2017-04-18 -0.0013754720  3.477183e-04        0
## 2017-04-19 -0.0053668457 -9.033307e-05        1
## 2017-04-20  0.0070476620 -6.595851e-05        0
## 2017-04-21  0.0136469440  2.431953e-04        1
## 2017-04-24  0.0168748422 -1.524425e-04        0
## 2017-04-25  0.0057585838 -7.089536e-04        0
## 2017-04-26 -0.0013259081 -1.256023e-03        1
## 2017-04-27  0.0064657830 -1.142395e-03        0
## 2017-04-28  0.0027792310 -7.010131e-04        0
## 2017-05-01  0.0137813885 -7.490743e-04        0
## 2017-05-02 -0.0015860575 -6.634409e-04        1
## 2017-05-03 -0.0031796673 -1.011952e-03        1
## 2017-05-04 -0.0039162281 -7.093379e-04        1
## 2017-05-05  0.0027574504 -2.596178e-04        0
```

```
## 2017-05-08 -0.0008699145 -6.317336e-05        1
## 2017-05-09  0.0014494712 -9.329333e-05        0
## 2017-05-10  0.0039031058 -9.635365e-05        0
## 2017-05-11 -0.0123395486 -6.379707e-05        1
## 2017-05-12 -0.0011692782 -2.671010e-04        1
## 2017-05-15  0.0007309846  3.653054e-04        1
## 2017-05-16  0.0142196820  3.475128e-04        1
## 2017-05-17 -0.0281997048  1.692196e-04        0
## 2017-05-18  0.0034025626 -4.604935e-04        0
## 2017-05-19 -0.0002953767  7.844770e-04        0
## 2017-05-22  0.0111650198  6.522227e-04        1
## 2017-05-23  0.0033545281  1.901514e-04        1
## 2017-05-24  0.0013095237 -1.583272e-04        0
## 2017-05-25  0.0122843658 -4.459050e-04        0
## 2017-05-26  0.0048717104 -2.073734e-04        0
## 2017-05-30  0.0064117195 -7.938493e-04        0
## 2017-05-31 -0.0081285011 -7.032946e-04        1
## 2017-06-01  0.0037159113 -8.064869e-04        0
## 2017-06-02  0.0234044801 -1.923354e-04        0
## 2017-06-05  0.0072202063 -2.553099e-04        0
## 2017-06-06  0.0033148926 -1.213674e-03        0
## 2017-06-07 -0.0017941900 -1.206418e-03        1
## 2017-06-08 -0.0060967629 -8.949360e-04        1
## 2017-06-09 -0.0229151383 -5.851241e-04        1
## 2017-06-12 -0.0077088319 -4.990966e-05        1
## 2017-06-13  0.0123907160  8.928514e-04        1
## 2017-06-14 -0.0053932149  1.074195e-03        0
## 2017-06-15 -0.0052792447  1.325543e-04        0
## 2017-06-16  0.0014295642  3.008658e-04        1
## 2017-06-19  0.0123520131  4.705426e-04        1
## 2017-06-20 -0.0136384974  2.521060e-04        0
## 2017-06-21  0.0051361648 -3.535153e-04        0
## 2017-06-22 -0.0001422472  2.673678e-04        0
## 2017-06-23  0.0134305686  9.963631e-05        1
## 2017-06-26 -0.0095951069 -6.478164e-05        1
## 2017-06-27 -0.0188927906 -4.936590e-04        1
## 2017-06-28  0.0084887063 -9.887688e-05        0
## 2017-06-29 -0.0189463327  9.249868e-04        0
## 2017-06-30  0.0064037769  2.116203e-04        1
## 2017-07-03 -0.0110869409  9.366107e-04        0
## 2017-07-05  0.0132607262  4.819567e-04        1
## 2017-07-06 -0.0074101609  6.401906e-04        0
## 2017-07-07  0.0128959116  4.037088e-05        1
## 2017-07-10  0.0074584969  1.285082e-04        1
## 2017-07-11  0.0001428163 -2.381587e-04        0
## 2017-07-12  0.0164380065 -7.182740e-04        0
## 2017-07-13  0.0086761670 -3.370932e-04        0
## 2017-07-14  0.0139746588 -1.050360e-03        0
## 2017-07-17  0.0078012989 -1.061281e-03        0
## 2017-07-18 -0.0006818275 -1.369681e-03        1
## 2017-07-19  0.0076107731 -1.261569e-03        0
## 2017-07-20  0.0048622461 -8.473224e-04        0
## 2017-07-21 -0.0058104345 -8.657521e-04        1
## 2017-07-24 -0.0025782361 -8.457616e-04        1
```

```
## 2017-07-25  0.0079843986 -2.982100e-04        0
## 2017-07-26 -0.0018888159 -7.417800e-05        1
## 2017-07-27 -0.0120917030 -3.852272e-04        1
## 2017-07-28 -0.0016416282 -2.264537e-04        1
## 2017-07-31 -0.0046659067  4.116521e-04        0
## 2017-08-01 -0.0016519139  3.748488e-04        0
## 2017-08-02 -0.0044186759  4.333916e-04        0
## 2017-08-03 -0.0015234405  3.996185e-04        0
## 2017-08-04  0.0073189306  4.316802e-04        1
## 2017-08-07 -0.0038599165  3.978658e-04        0
## 2017-08-08  0.0053722699 -7.074285e-05        0
## 2017-08-09 -0.0044058999  9.485244e-05        0
## 2017-08-10 -0.0147347253 -1.238103e-04        1
## 2017-08-11  0.0151485903  3.917017e-05        1
## 2017-08-14  0.0149225307  7.289001e-04        1
## 2017-08-15 -0.0050404712 -1.309913e-04        1
## 2017-08-16  0.0058555487 -8.295540e-04        0
## 2017-08-17 -0.0171178430 -3.416328e-04        1
## 2017-08-18  0.0012422667 -4.460039e-04        0
## 2017-08-21 -0.0047012808  3.673327e-04        0
## 2017-08-22  0.0139015653  3.454396e-04        1
## 2017-08-23 -0.0060324146  2.991812e-04        0
## 2017-08-24 -0.0004126126 -2.575168e-04        1
## 2017-08-25  0.0017867917 -1.047613e-04        0
## 2017-08-28  0.0001373429  1.367387e-04        1
## 2017-08-29  0.0030161936 -1.366151e-04        0
## 2017-08-30  0.0130560670  1.613192e-05        1
## 2017-08-31  0.0102164487 -2.110333e-04        0
## 2017-09-01 -0.0111627143 -6.393791e-04        1
## 2017-09-05 -0.0044730809 -9.855498e-04        1
## 2017-09-06 -0.0028569368 -1.438709e-04        1
## 2017-09-07  0.0127251482  1.265525e-04        1
## 2017-09-08 -0.0048542841  2.059495e-04        0
## 2017-09-11  0.0104881824 -3.866841e-04        0
## 2017-09-12 -0.0010706907 -1.415325e-04        1
## 2017-09-13  0.0070718689 -4.414029e-04        0
## 2017-09-14 -0.0058674927 -3.931126e-04        1
## 2017-09-15  0.0071962067 -4.422340e-04        0
## 2017-09-18 -0.0019936738 -1.854932e-04        1
## 2017-09-19  0.0037184371 -2.845161e-04        0
## 2017-09-20 -0.0066498448 -2.653258e-04        1
## 2017-09-21 -0.0097889215 -1.952583e-04        1
## 2017-09-22  0.0026914967  1.845415e-05        1
## 2017-09-25 -0.0155756105  5.784026e-04        0
## 2017-09-26  0.0000000000  2.072645e-04        0
## 2017-09-27  0.0080211972  8.418910e-04        1
## 2017-09-28  0.0002708503  6.216847e-04        1
## 2017-09-29  0.0083580292  2.602111e-05        1
## 2017-10-02  0.0016096985 -1.468398e-06        0
## 2017-10-03 -0.0047020842 -3.532794e-04        1
## 2017-10-04  0.0057737655 -3.570344e-04        0
## 2017-10-05  0.0169922975 -5.498649e-06        0
## 2017-10-06  0.0003948016 -2.498991e-04        0
## 2017-10-09  0.0038085409 -9.262056e-04        0
```

```
## 2017-10-10  0.0000000000 -7.078270e-04        0
## 2017-10-11  0.0017025346 -5.805418e-04        0
## 2017-10-12  0.0091182734 -4.556621e-04        0
## 2017-10-13  0.0047861808 -3.228548e-04        0
## 2017-10-16  0.0020627054 -6.590485e-04        0
## 2017-10-17 -0.0007730740 -6.517744e-04        1
## 2017-10-18  0.0002577964 -5.444536e-04        0
## 2017-10-19  0.0038580679 -3.259211e-04        0
## 2017-10-20  0.0114855013 -2.286508e-04        0
## 2017-10-23  0.0002537935 -3.185658e-04        0
## 2017-10-24  0.0003804807 -7.222723e-04        0
## 2017-10-25 -0.0029208733 -5.426922e-04        1
## 2017-10-26  0.0016520113 -3.398360e-04        0
## 2017-10-27  0.0621470291 -1.367763e-04        0
## 2017-10-30  0.0009540967 -1.170762e-04        0
## 2017-10-31 -0.0084994706 -2.946505e-03        1
## 2017-11-01  0.0000000000 -2.210642e-03        0
```

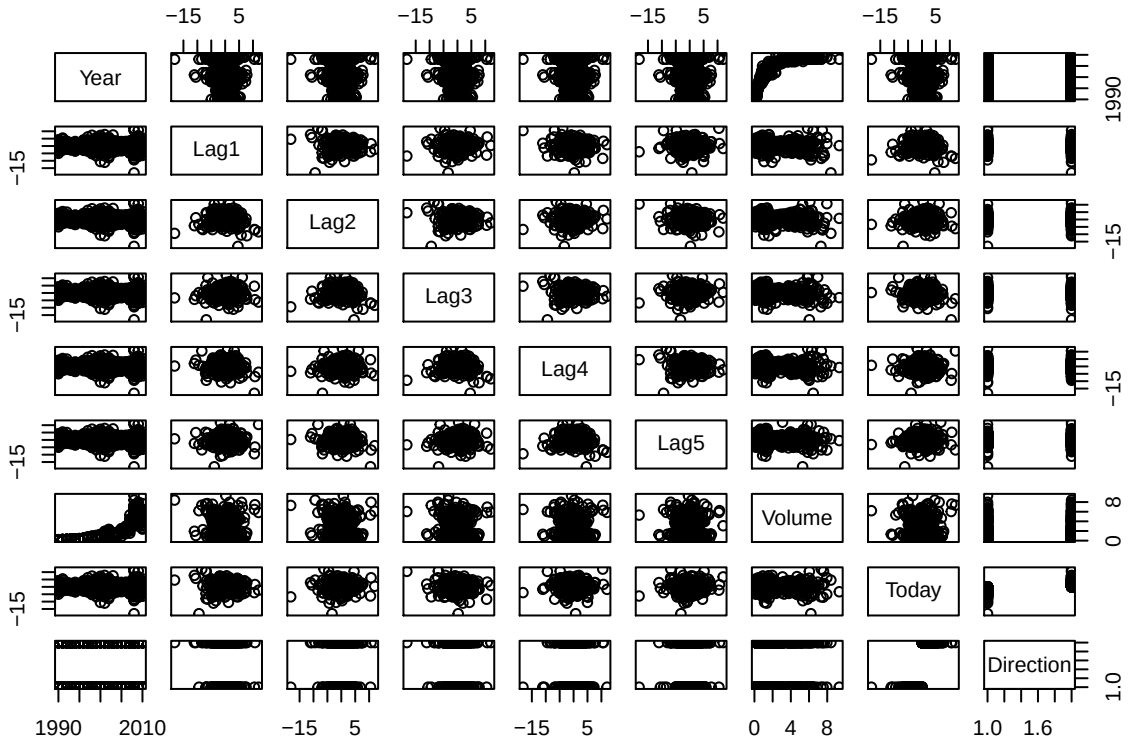
```
## [1] 44
```

6 Assignment E

6.1 Q.1] Consider the Weekly data set, which is part of ISLR package. It contains the weekly stock market returns for 21 years.

6.1.1 a] Produce some numerical and graphical summaries of the Weekly data. Do there appear to be any pattern?

```
##       Year           Lag1               Lag2               Lag3         
##  Min.   :1990   Min.   :-18.1950   Min.   :-18.1950   Min.   :-18.1950  
##  1st Qu.:1995   1st Qu.: -1.1540   1st Qu.: -1.1540   1st Qu.: -1.1580  
##  Median :2000   Median :  0.2410   Median :  0.2410   Median :  0.2410  
##  Mean   :2000   Mean   :  0.1506   Mean   :  0.1511   Mean   :  0.1472  
##  3rd Qu.:2005   3rd Qu.:  1.4050   3rd Qu.:  1.4090   3rd Qu.:  1.4090  
##  Max.   :2010   Max.   : 12.0260   Max.   : 12.0260   Max.   : 12.0260  
##       Lag4               Lag5              Volume       
##  Min.   :-18.1950   Min.   :-18.1950   Min.   :0.08747  
##  1st Qu.: -1.1580   1st Qu.: -1.1660   1st Qu.:0.33202  
##  Median :  0.2380   Median :  0.2340   Median :1.00268  
##  Mean   :  0.1458   Mean   :  0.1399   Mean   :1.57462  
##  3rd Qu.:  1.4090   3rd Qu.:  1.4050   3rd Qu.:2.05373  
##  Max.   : 12.0260   Max.   : 12.0260   Max.   :9.32821  
##      Today          Direction 
##  Min.   :-18.1950   Down:484  
##  1st Qu.: -1.1540   Up  :605  
##  Median :  0.2410             
##  Mean   :  0.1499             
##  3rd Qu.:  1.4050             
##  Max.   : 12.0260             
```



We can observe that the Weekly data from ISLR has Volume and Year taken together has logarithmic distribution.

6.1.2 b] Use the full data set to perform a logistic regression with Direction as the response and the five lag variables plus Volume as predictors. Use the summary function to print the results. Do any of the predictors appears to be statistically significant? If so, which ones?

```
##
## Call:
## glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +
##     Volume, family = "binomial", data = Weekly)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6949  -1.2565   0.9913   1.0849   1.4579
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.26686    0.08593   3.106   0.0019 **
## Lag1        -0.04127    0.02641  -1.563   0.1181
## Lag2         0.05844    0.02686   2.175   0.0296 *
## Lag3        -0.01606    0.02666  -0.602   0.5469
## Lag4        -0.02779    0.02646  -1.050   0.2937
```

```
## Lag5        -0.01447    0.02638  -0.549   0.5833
## Volume      -0.02274    0.03690  -0.616   0.5377
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1496.2  on 1088  degrees of freedom
## Residual deviance: 1486.4  on 1082  degrees of freedom
## AIC: 1500.4
##
## Number of Fisher Scoring iterations: 4
```

Statistically significant predictor among the given is Lag2 only since the p-value is greater than the significant code attached to it.

6.1.3 c] Compute the confusion matrix and overall fraction of correct predictions. Explain what the confusion matrix is telling you about the types of mistakes made by logistic regression.

```
##       Weeklyglm.preds
##        Down  Up
##   Down   54 430
##   Up     48 557
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Down  Up
##       Down   54 430
##       Up     48 557
##
##                Accuracy : 0.5611
##                  95% CI : (0.531, 0.5908)
##     No Information Rate : 0.9063
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.035
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.52941
##             Specificity : 0.56434
##          Pos Pred Value : 0.11157
##          Neg Pred Value : 0.92066
##              Prevalence : 0.09366
##          Detection Rate : 0.04959
##    Detection Prevalence : 0.44444
##       Balanced Accuracy : 0.54687
##
##        'Positive' Class : Down
##
```

There are a predominance of Up prediction. The model predicts well the Up direction, but it predict poorly

the Down direction.

6.1.4 d] Now fit the logistic regression model using a training data period from 1990 to 2008, with Lag2 as the only predictor. Compute the confusion matrix and the overall fraction of correct predictions for the held out data (that is, the data from 2009 and 2010.)

```
##       glm.preds.d
##        Down Up
##   Down    9 34
##   Up      5 56
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Down Up
##       Down    9 34
##       Up      5 56
##
##                Accuracy : 0.625
##                  95% CI : (0.5247, 0.718)
##     No Information Rate : 0.8654
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1414
##  Mcnemar's Test P-Value : 7.34e-06
##
##             Sensitivity : 0.64286
##             Specificity : 0.62222
##          Pos Pred Value : 0.20930
##          Neg Pred Value : 0.91803
##              Prevalence : 0.13462
##          Detection Rate : 0.08654
##    Detection Prevalence : 0.41346
##       Balanced Accuracy : 0.63254
##
##        'Positive' Class : Down
##
```

Overall fraction of correct predictions for the held out data is accuracy is 0.625

6.1.5 e] Repeat (d) using linear discriminant analysis (LDA).

```
##
##        Down Up
##   Down    9 34
##   Up      5 56
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Down Up
##       Down    9 34
##       Up      5 56
```

```
##
##                Accuracy : 0.625
##                  95% CI : (0.5247, 0.718)
##     No Information Rate : 0.8654
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1414
##  Mcnemar's Test P-Value : 7.34e-06
##
##             Sensitivity : 0.64286
##             Specificity : 0.62222
##          Pos Pred Value : 0.20930
##          Neg Pred Value : 0.91803
##              Prevalence : 0.13462
##          Detection Rate : 0.08654
##    Detection Prevalence : 0.41346
##       Balanced Accuracy : 0.63254
##
##        'Positive' Class : Down
##
```

Overall fraction of correct predictions for the held out data is accuracy is 0.625

6.1.6 f] Repeat (d) using quadratic discriminant analysis (QDA).

```
##
##        Down Up
##   Down    0 43
##   Up      0 61
```

```
## [1] 0.5865385
```

Overall fraction of correct predictions for the held out data is accuracy is 0.5865

6.1.7 g] Repeat (d) using KNN with =1.

```
##       knn.pred
##        Down Up
##   Down   21 22
##   Up     30 31
```

```
## [1] 0.5
```

Overall fraction of correct predictions for the held out data is accuracy is 0.5865

6.1.8 h] Which of these methods appears to provide the best results on this data?

The models from letter d and e, respectively Logistic Regression and LDA.

6.2 Q.2] This problem involves predicting Salary from the Hitters data set which is part of the

ISLR package.

6.2.1 a] Remove the observations for whom the salary information is unknown, and then log-transform the salaries.

```
##      AtBat            Hits         HmRun            Runs
##  Min.   : 16.0   Min.   :  1   Min.   : 0.00   Min.   :  0.00
##  1st Qu.:255.2   1st Qu.: 64   1st Qu.: 4.00   1st Qu.: 30.25
##  Median :379.5   Median : 96   Median : 8.00   Median : 48.00
##  Mean   :380.9   Mean   :101   Mean   :10.77   Mean   : 50.91
##  3rd Qu.:512.0   3rd Qu.:137   3rd Qu.:16.00   3rd Qu.: 69.00
##  Max.   :687.0   Max.   :238   Max.   :40.00   Max.   :130.00
##
##       RBI             Walks            Years            CAtBat
##  Min.   :  0.00   Min.   :  0.00   Min.   : 1.000   Min.   :   19.0
##  1st Qu.: 28.00   1st Qu.: 22.00   1st Qu.: 4.000   1st Qu.:  816.8
##  Median : 44.00   Median : 35.00   Median : 6.000   Median : 1928.0
##  Mean   : 48.03   Mean   : 38.74   Mean   : 7.444   Mean   : 2648.7
##  3rd Qu.: 64.75   3rd Qu.: 53.00   3rd Qu.:11.000   3rd Qu.: 3924.2
##  Max.   :121.00   Max.   :105.00   Max.   :24.000   Max.   :14053.0
##
##      CHits            CHmRun           CRuns             CRBI
##  Min.   :   4.0   Min.   :  0.00   Min.   :   1.0   Min.   :   0.00
##  1st Qu.: 209.0   1st Qu.: 14.00   1st Qu.: 100.2   1st Qu.:  88.75
##  Median : 508.0   Median : 37.50   Median : 247.0   Median : 220.50
##  Mean   : 717.6   Mean   : 69.49   Mean   : 358.8   Mean   : 330.12
##  3rd Qu.:1059.2   3rd Qu.: 90.00   3rd Qu.: 526.2   3rd Qu.: 426.25
##  Max.   :4256.0   Max.   :548.00   Max.   :2165.0   Max.   :1659.00
##
##      CWalks        League  Division    PutOuts          Assists
##  Min.   :   0.00   A:175   E:157    Min.   :   0.0   Min.   :  0.0
##  1st Qu.:  67.25   N:147   W:165    1st Qu.: 109.2   1st Qu.:  7.0
##  Median : 170.50                    Median : 212.0   Median : 39.5
##  Mean   : 260.24                    Mean   : 288.9   Mean   :106.9
##  3rd Qu.: 339.25                    3rd Qu.: 325.0   3rd Qu.:166.0
##  Max.   :1566.00                    Max.   :1378.0   Max.   :492.0
##
##      Errors          Salary       NewLeague
##  Min.   : 0.00   Min.   :  67.5   A:176
##  1st Qu.: 3.00   1st Qu.: 190.0   N:146
##  Median : 6.00   Median : 425.0
##  Mean   : 8.04   Mean   : 535.9
##  3rd Qu.:11.00   3rd Qu.: 750.0
##  Max.   :32.00   Max.   :2460.0
##                  NA's   :59
```

6.2.2 b] Create a training set consisting of the first 200 observations, and a test set consisting of the remaining observations.

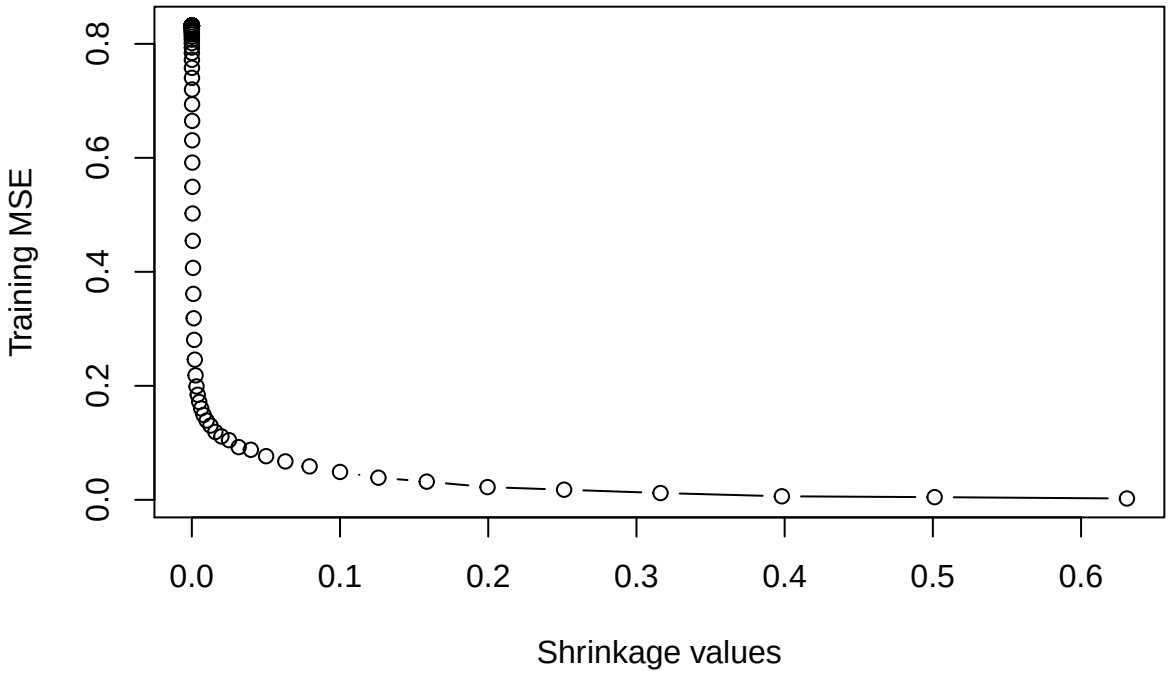
```
## [1] "Training data head: "
```

	AtBat	Hits	HmRun	Runs	RBI	Walks	Years	CAtBat	CHits	CHmRun	CRuns	CRBI	CWalks	League	Division	PutOuts	Assists	Errors	Salary	NewLeague
-Alan Ashby	315	81	7	24	38	39	14	3449	835	69	321	414	375	N	W	632	43	10	6.163315	N
-Alvin Davis	479	130	18	66	72	76	3	1624	457	63	224	266	263	A	W	880	82	14	6.173786	A
-Andre Dawson	496	141	20	65	78	37	11	5628	1575	225	828	838	354	N	E	200	11	3	6.214608	N
-Andres Galarraga	321	87	10	39	42	30	2	396	101	12	48	46	33	N	E	805	40	4	4.516339	N
-Alfredo Griffin	594	169	4	74	51	35	11	4408	1133	19	501	336	194	A	W	282	421	25	6.620073	A
-Al Newman	185	37	1	23	8	21	2	214	42	1	30	9	24	N	E	76	127	7	4.248495	A

```
## [1] "Test data head: "
```

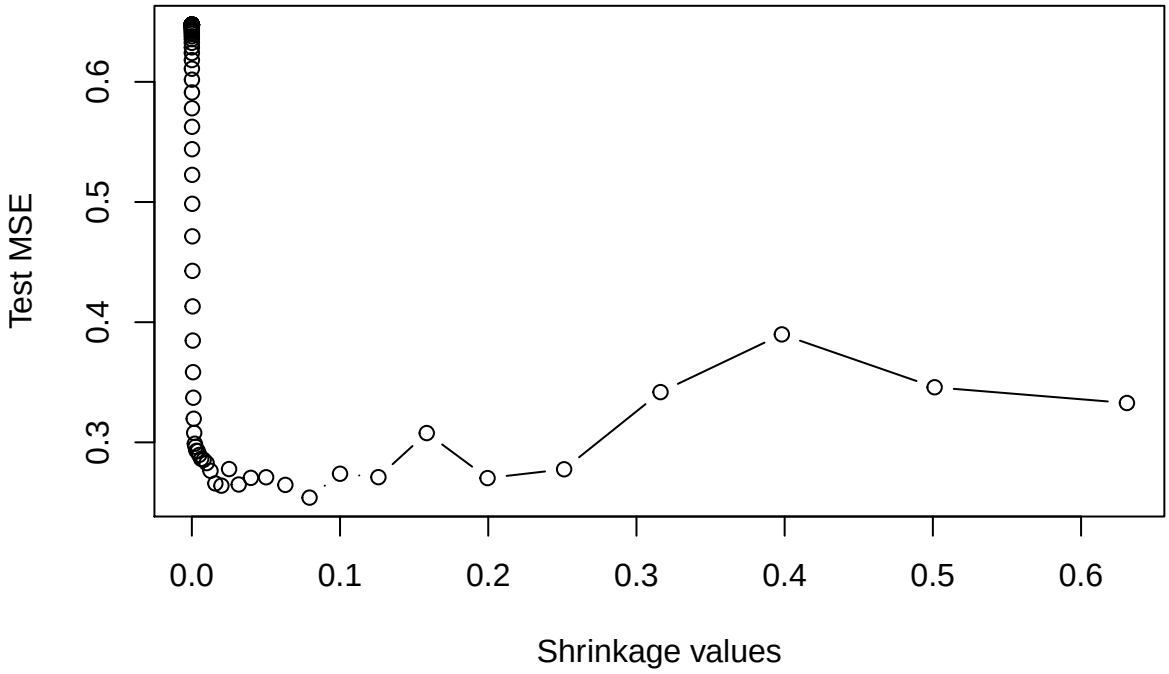
	AtBat	Hits	HmRun	Runs	RBI	Walks	Years	CAtBat	CHits	CHmRun	CRuns	CRBI	CWalks	League	Division	PutOuts	Assists	Errors	Salary	NewLeague
-Reggie Jackson	419	101	18	65	58	92	20	9528	2510	548	1509	1659	1342	A	W	0	0	0	6.189290	A
-Ron Kittle	376	82	21	42	60	35	5	1770	408	115	238	299	157	A	W	0	0	0	6.052089	A
-Ray Knight	486	145	11	51	76	40	11	3967	1102	67	410	497	284	N	E	88	204	16	6.214608	A
-Rick Leach	246	76	5	35	39	13	6	912	234	12	102	96	80	A	E	44	0	1	5.521461	A
-Rick Manning	205	52	8	31	27	17	12	5134	1323	56	643	445	459	A	E	155	3	2	5.991465	A
-Rance Mulliniks	348	90	11	50	45	43	10	2288	614	43	295	273	269	A	E	60	176	6	6.109248	A

6.2.3 c] Perform boosting on the training set with 1000 trees for a range of values of the shrinkage parameter lambda. Produce a plot with different shrinkage values on the x-axis and the corresponding training set MSE on the y-axis.



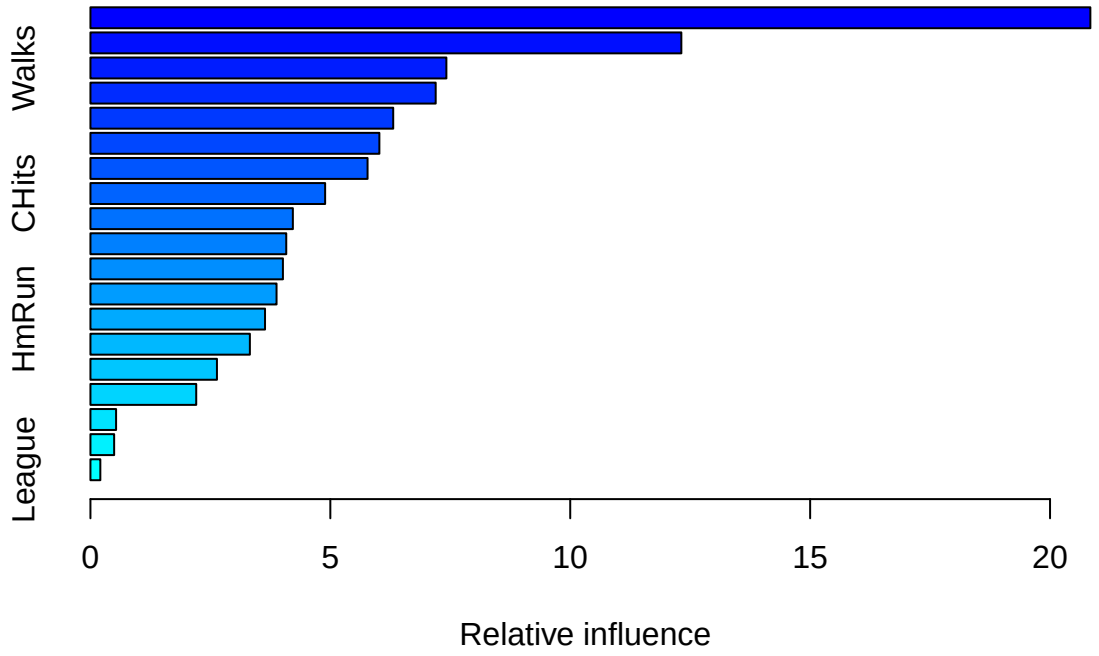
We observe, as shrinkage value increases the training MSE value exponentially decreases.

6.2.4 d] Produce a plot with different shrinkage values on the x-axis and the corresponding test set MSE on the y-axis.




```
## [1] "The minimum test MSE is  0.254026510444201  , and is obtained for lambda = 0.0794328234724282"
```

6.2.5 e] Which variable appear to be the most important predictors in the boosted model?




```
##                 var    rel.inf
## CAtBat       CAtBat 20.8404970
## CRBI           CRBI 12.3158959
## Walks         Walks  7.4186037
## PutOuts     PutOuts  7.1958539
## Years         Years  6.3104535
## CWalks       CWalks  6.0221656
## CHmRun       CHmRun  5.7759763
## CHits         CHits  4.8914360
## AtBat         AtBat  4.2187460
## RBI             RBI  4.0812410
## Hits           Hits  4.0117255
## Assists     Assists  3.8786634
## HmRun         HmRun  3.6386178
## CRuns         CRuns  3.3230296
## Errors       Errors  2.6369128
## Runs           Runs  2.2048386
## Division   Division  0.5347342
## NewLeague NewLeague  0.4943540
## League       League  0.2062551
```

We see that CAtBat is most important variable in all the variables list, relatively. Also, relative influence of Walks is found to be highest.

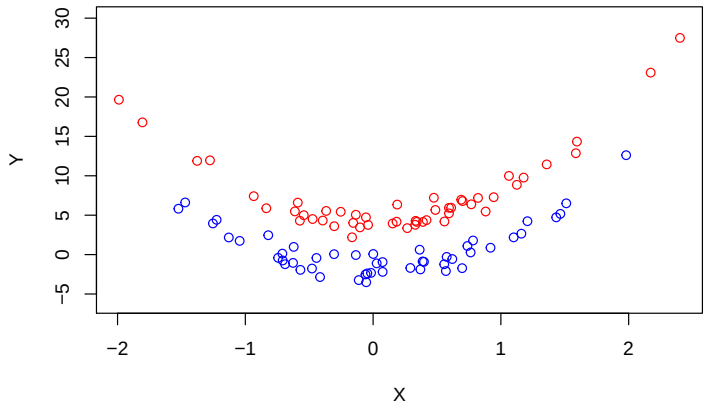
6.2.6 f] Apply bagging to the training set. What is the test set MSE for this approach.

```
## [1] "The test MSE for bagging is 0.22993242086693, which is slightly lower than the test MSE for boos
```

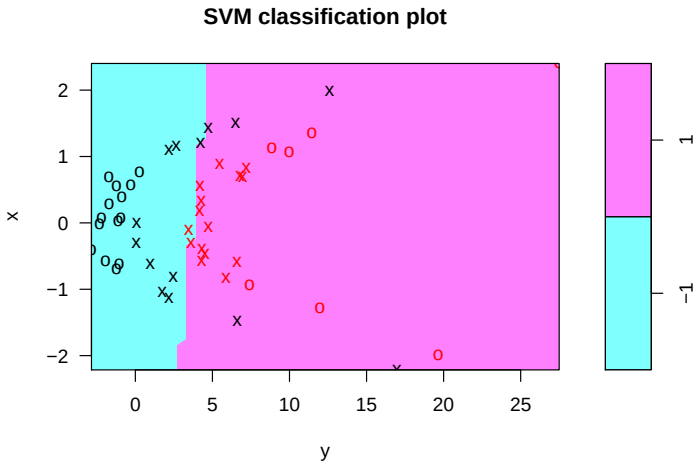
6.2.7 g] Apply random forests to the training set. What is the test set MSE for this approach.

```
## [1] "The test MSE for Random Forest is 0.214033998567829, which is slightly lower than the test MSE f
```

6.3 Q.3] Generate a simulated two-class data set with 100 observations and two features in which there is a visible but non-linear seperation between the classes. Show that in this setting, a support vector machine with a polynomial kernal (with degree greater than 1) or a radial kernal will outperform a support vector classifier on the training data. Which technique performs best on the test data? Make plots and report training and test error rates in order to back up your assertions.

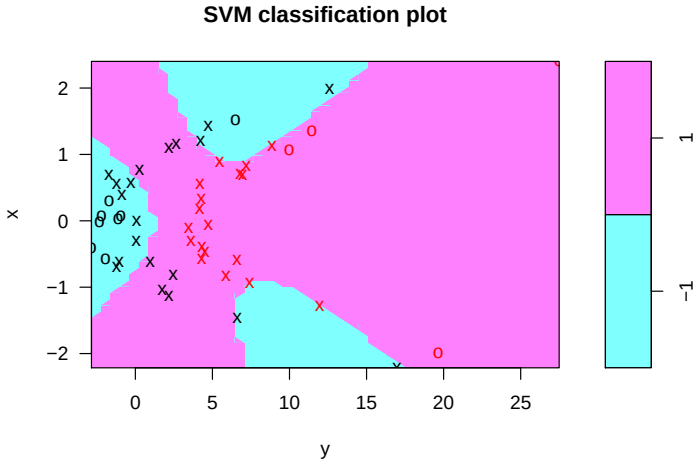


We can clearly see the seperation between two classes - Non linear
Now, we fit a support vector classifier on the training data




```
##         truth
## predict -1  1
##      -1 22  0
##      1   6 22
```

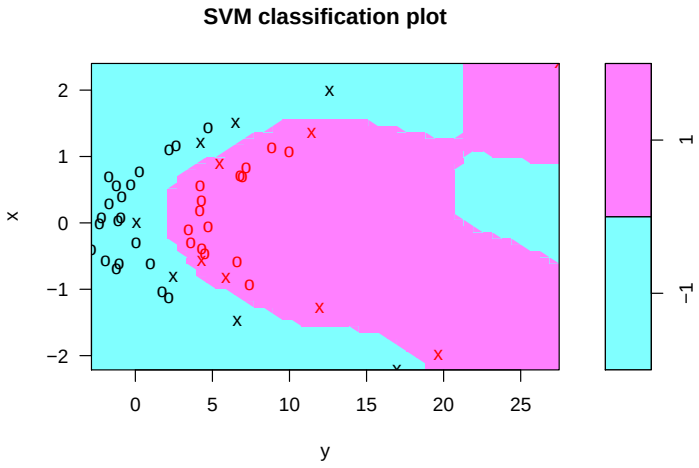
The support vector classifier makes 6 errors on the training data. Next, we fit a support vector machine with a polynomial kernel.




```
##         truth
## predict -1  1
##      -1 19  0
##      1   9 22
```

The support vector machine with a polynomial kernel of degree 3 makes 9 errors on the training data.

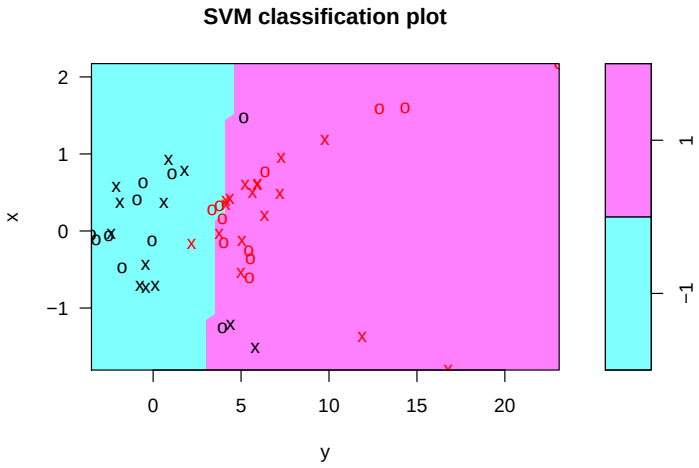
Finally, we fit a support vector machine with a radial kernel and a gamma of 1.



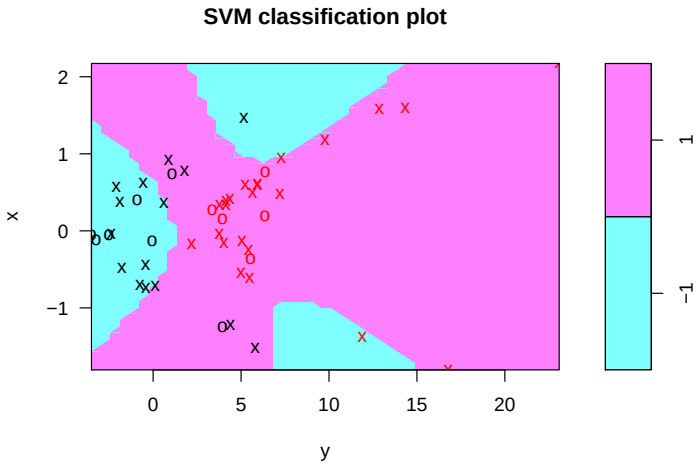

```
##         truth
## predict -1  1
##      -1 28  0
##      1   0 22
```

The support vector machine with a radial kernel makes 0 error on the training data.

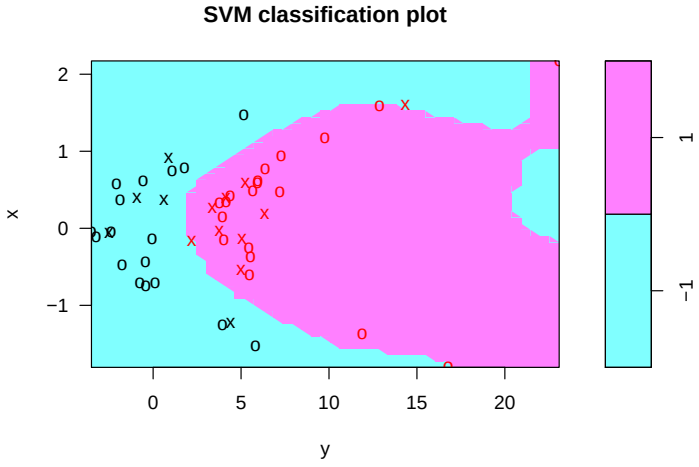
Now, we check how these models fare when applied to the test data.




```
##         truth
## predict -1  1
##      -1 18  2
##      1   4 26
```




```
##         truth
## predict -1  1
##      -1 14  1
##       1  8 27
```




```
##         truth
## predict -1  1
##      -1 22  1
##       1  0 27
```

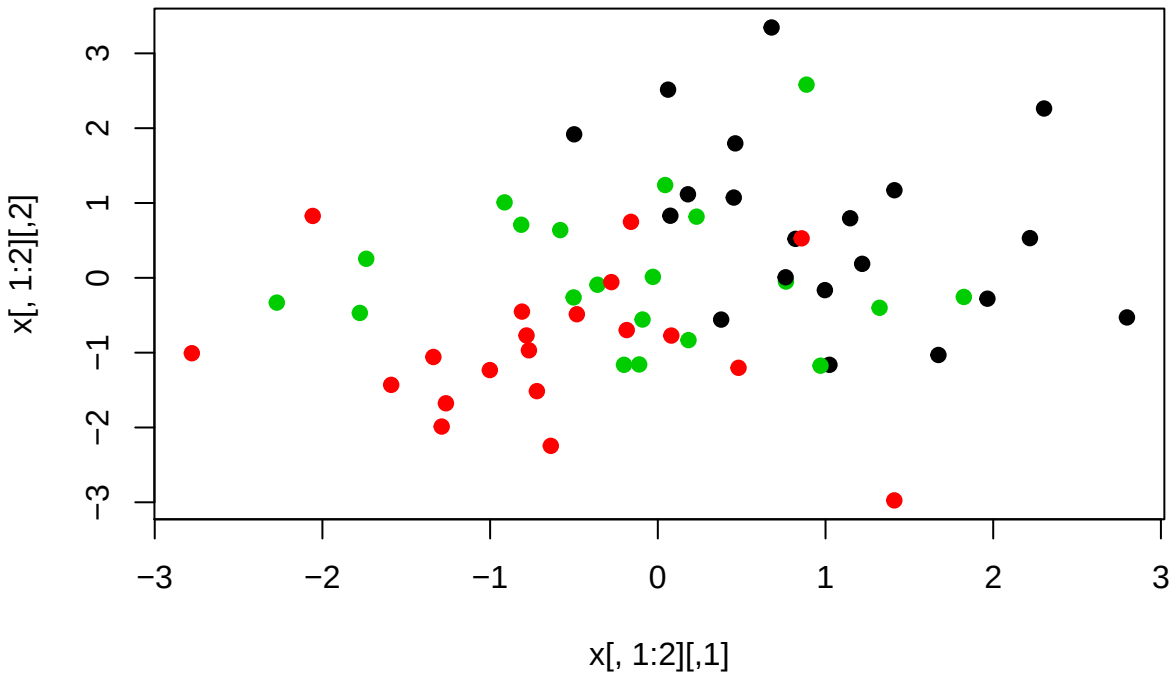
We may see that the linear, polynomial and radial support vector machines classify respectively 9, 6 and 1 observations incorrectly. So, radial kernel is the best model in this setting.

```
# Published assignment can be found at https://rpubs.com/SangamKotalwar/Assignment3_E
```

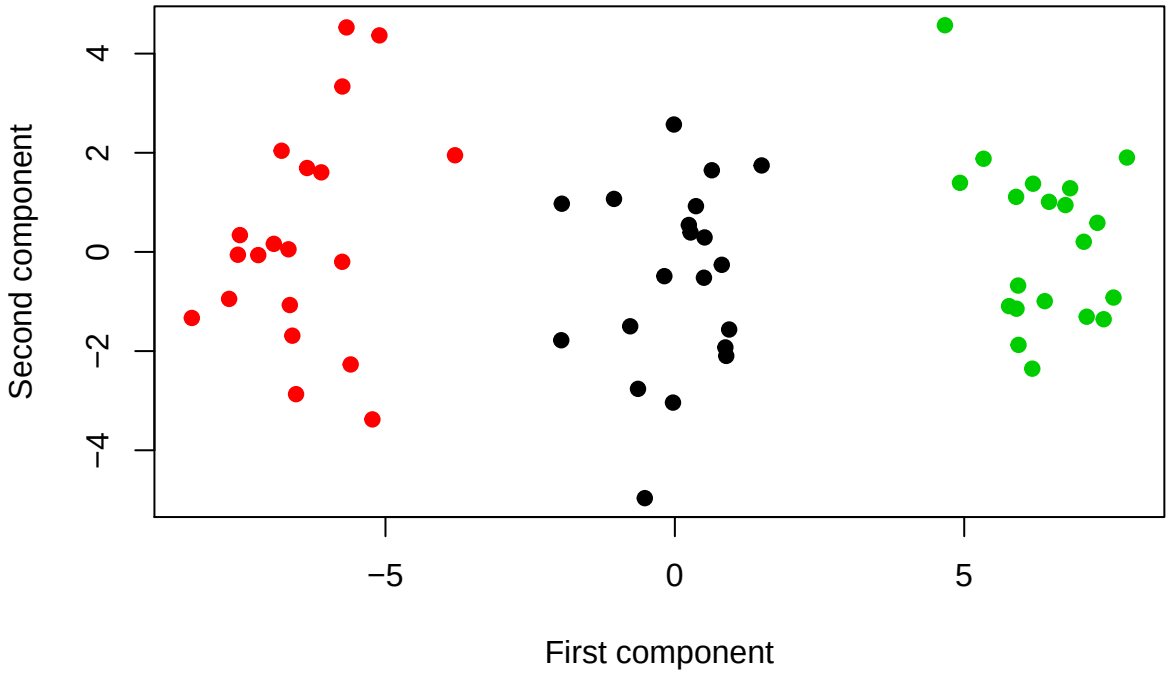
7 Assignment F

7.1 Q.1]

7.1.1 (a) Generate a simulated data set with 20 observations in each of three classes (i.e. 60 observations total), and 50 variables. (Be sure to add a mean shift to the observations in each class so that there are three distinct classes.)



7.1.2 (b) Perform PCA on the 60 observations and plot the first two principal component score vectors. Use a different color to indicate the observations in each of the three classes. If the three classes appear separated in this plot, then continue on to part (c). If not, then return to part (a) and modify the simulation so that there is greater separation between the three classes. Do not continue to part (c) until the three classes show at least some separation in the first two principal component score vectors.



7.1.3 (c) Perform K-means clustering of the observations with K = 3. How well do the clusters that you obtained in K-means cluster- ing compare to the true class labels?

```
##         class
## cluster  1  2  3
##       1  0 20  0
##       2  0  0 20
##       3 20  0  0
```

all the classes have been assigned to individual clusters.

7.1.4 (d) Perform K-means clustering with K = 2. Describe your results.

```
##         class
## cluster  1  2  3
```

```
##       1  2 20  0
##       2 18  0 20
```

class 1 has been assigned to cluster 1 and 2, class 2 to cluster 1 and class 3 to cluster 2.

7.1.5 (e) Now perform K-means clustering with K = 4, and describe your results.

```
##        class
## cluster  1  2  3
##       1  0  0 20
##       2  0  9  0
##       3 20  0  0
##       4  0 11  0
```

class 1 has been assigned to cluster 3, class 2 has been assigned to cluster 2 and 4, class 3 has been assigned to cluster 1.

7.1.6 (f) Now perform K-means clustering with K = 3 on the first two principal component score vectors, rather than on the raw data. That is, perform K-means clustering on the 60 * 2 matrix of which the first column is the first principal component score vector, and the second column is the second principal component score vector. Comment on the results.

```
##        class
## cluster  1  2  3
##       1  0  0 20
##       2 20  0  0
##       3  0 20  0
```

every class has been assigned to each cluster individually, similar accuracy as in part (c).

7.1.7 (g) Using the scale() function, perform K-means clustering with K = 3 on the data after scaling each variable to have standard deviation one. How do these results compare to those obtained in (b)? Explain.

```
##        class
## cluster  1  2  3
##       1  0  0 20
##       2 20  0  0
##       3  0 20  0
```

We have worse results than with unscaled data, as scaling affects the distance between the observations.