

FINAL EXAM

1. a) For the dataset `BSE_Sensex_Index.csv`, create an extra column of successive differences for each column of numeric values in this data file. Extract two simple random samples with replacement of 1000 and 3000 observations (rows). Show your R commands for doing this.

Do the same thing by using Excel. Show your Excel commands.

Note: Successive difference for date d1= (date d1 value-immediate available previous date of d1 value)/immediate available previous date of d1. For the last row fill up values with mean of its immediate three previous row values.

```

C:\Users\LAKEHM\MSH\MSH\Desktop\Repositores\Data Science 2019\105105 Data Mining\Final exam/ >
122 11/29/2010 1189.08 1190.34 1177.64 1187.76 3673450000 1187.76
123 11/22/2010 1194.16 1194.16 1186.91 1189.40 1613820000 1189.40
124 11/24/2010 1181.70 1198.62 1183.70 1198.35 3384250000 1198.35
125 11/23/2010 1192.51 1192.51 1176.90 1180.73 4133070000 1180.73
126 11/22/2010 1198.07 1198.04 1184.58 1197.84 3689500000 1197.84
127 11/29/2010 1196.12 1199.97 1189.44 1199.73 3673390000 1199.73
128 11/20/2010 1181.75 1200.29 1181.75 1196.69 4887260000 1196.69
129 11/17/2010 1178.33 1183.56 1175.82 1178.59 3904780000 1178.59
130 11/16/2010 1194.79 1194.79 1177.00 1178.34 5116380000 1178.34
131 11/15/2010 1200.44 1207.43 1187.11 1187.75 35013070000 1187.75
132 11/12/2010 1209.07 1210.50 1194.08 1199.21 4211620000 1199.21
133 11/11/2010 1211.04 1215.45 1206.49 1213.54 3931210000 1213.54
134 11/10/2010 1211.14 1218.75 1206.31 1218.71 4961300000 1218.71
135 11/9/2010 1223.59 1226.64 1208.94 1213.40 4848040000 1213.40
136 11/8/2010 1221.24 1224.57 1217.51 1223.25 3937230000 1223.25
137 11/5/2010 1221.20 1227.08 1220.29 1225.85 5637460000 1225.85
138 11/4/2010 1196.34 1221.25 1196.34 1221.06 5695470000 1221.06
139 11/3/2010 1191.79 1198.30 1181.56 1197.96 4661480000 1197.96
140 11/2/2010 1187.86 1195.88 1187.86 1193.57 3866200000 1193.57
141 11/1/2010 1185.71 1195.83 1177.65 1184.38 4122948000 1184.38
142 10/29/2010 1183.87 1185.46 1179.70 1183.26 3573880000 1183.26
[ reached 'max' / getoption("max.print") -- omitted 15305 rows ]

> summary(data)
      Date      open      high      low      close      volume
length:1547    min.: 16.66  min.: 16.66  min.: 16.66  min.: 16.66  min.: 16.800e+05
Class:character 1st Qu.: 79.98  1st Qu.: 80.72  1st Qu.: 79.39  1st Qu.: 79.98  1st Qu.: 5.830e+06
Mode:character  median: 115.97 median: 117.01 median: 114.85 median: 116.00 median: 14.326e+07
                mean: 391.98  mean: 396.59  mean: 394.05  mean: 394.05  mean: 15.864e+08
                3rd Qu.: 619.74 3rd Qu.: 621.40 3rd Qu.: 616.46 3rd Qu.: 620.07 3rd Qu.: 13.832e+08
                Max.: 1364.98  Max.: 1376.09  Max.: 1355.46  Max.: 1365.13  Max.: 1.146e+10

Adj.Close
min.: 16.66
1st Qu.: 79.98
median: 116.00
mean: 394.05
3rd Qu.: 620.07
Max.: 1365.13

> dataDate = as.Date(dataDate, format = "%m/%d/%Y")

> randomnew = function(df, n){
+   return(df[sample(nrow(df), n, replace = TRUE),])
+ }

>
+ successive_diff <- function(x){
+   n = length(x)
+   for (i in 1:(length(x)-1)) {
+     x[i] <- (x[i] - x[i+1]) / x[i+1]
+   }
+   x[length(x)] = (x[length(x)] - x[length(x) - 2] + x[length(x) - 3]) / 3
+   return(x)
+ }

>
> dataDate <- NULL
>
> dataOpenNew <- successive_diff(dataOpen)
> dataCloseNew <- successive_diff(dataClose)

```

```

C:\Users\MLAKSHMI\MAHURU\Desktop\Repositories\data_Science\201901105\data Mining final exam //
> data$open_new <- successive_diff(data$open)
> data$high_new <- successive_diff(data$high)
> data$low_new <- successive_diff(data$low)
> data$close_new <- successive_diff(data$close)
> data$volume_new <- successive_diff(data$volume)
> data$adj_close_new <- successive_diff(data$adj_close)

> set.seed(123)

d_1000m <- randomRows(data, 1000)

summary(d_1000m)

      open      high      low      close      volume      adj_close
Min.   : 17.08  Min.   : 17.08  Min.   : 17.08  Min.   : 17.08  Min.   :17.800e+05  Min.   : 17.08
1st Qu.: 83.17  1st Qu.: 83.17  1st Qu.: 82.50  1st Qu.: 83.17  1st Qu.:9.030e+06  1st Qu.: 83.47
Median :116.45  Median :117.59  Median :115.03  Median :116.34  Median :4.390e+07  Median :116.34
Mean   :136.28  Mean   :140.03  Mean   :139.52  Mean   :138.50  Mean   :5.964e+08  Mean   :138.50
3rd Qu.: 650.67  3rd Qu.: 654.12  3rd Qu.: 644.93  3rd Qu.: 648.62  3rd Qu.:4.035e+08  3rd Qu.: 648.62
Max.   :1552.19  Max.   :1526.45  Max.   :1550.74  Max.   :1506.34  Max.   :1.829e+09  Max.   :1506.34

      open_new      high_new      low_new      close_new      volume_new      adj_close_new
Min.   :-0.0582780  Min.   :-0.0432817  Min.   :-0.0474458  Min.   :-0.0402908  Min.   :-0.178888
1st Qu.: -0.0034812  1st Qu.: -0.0034812  1st Qu.: -0.0034812  1st Qu.: -0.0042513  1st Qu.: -0.101633
Median : -0.0005554  Median : -0.0003488  Median : -0.0008122  Median : -0.0003301  Median : -0.002597
Mean : -0.0009595  Mean : -0.0004185  Mean : -0.0005022  Mean : -0.0003370  Mean : -0.007352
3rd Qu.: -0.0009595  3rd Qu.: -0.0043302  3rd Qu.: -0.0047861  3rd Qu.: -0.0048606  3rd Qu.: -0.103772
Max.   : -0.1067121  Max.   : -0.0343908  Max.   : -0.0910833  Max.   : -0.0573273  Max.   : -0.167715

adj_close_new
Min.   :-0.0402908
1st Qu.: -0.0042513
Median : -0.0003301
Mean : -0.0003370
3rd Qu.: -0.0048606
Max.   : -0.0573273

d_3000m <- randomRows(data, 3000)

summary(d_3000m)

      open      high      low      close      volume      adj_close
Min.   :16.72  Min.   :16.72  Min.   :16.72  Min.   :16.72  Min.   :17.400e+05  Min.   :16.72
1st Qu.: 79.61  1st Qu.: 80.10  1st Qu.: 78.94  1st Qu.: 79.42  1st Qu.:5.972e+06  1st Qu.: 79.42
Median :113.11  Median :114.21  Median :111.98  Median :112.88  Median :4.036e+07  Median :112.88
Mean   :135.96  Mean   :138.37  Mean   :138.19  Mean   :138.19  Mean :5.448e+08  Mean   :138.19
3rd Qu.: 495.77  3rd Qu.: 497.82  3rd Qu.: 494.57  3rd Qu.: 497.14  3rd Qu.:3.181e+08  3rd Qu.: 497.14
Max.   :1556.51  Max.   :1563.03  Max.   :1554.09  Max.   :1561.80  Max.   :1.148e+09  Max.   :1561.80

      open_new      high_new      low_new      close_new      volume_new      adj_close_new
Min.   :-0.0871188  Min.   :-0.0883302  Min.   :-0.0821116  Min.   :-0.0880141  Min.   :-0.754927
1st Qu.: -0.0038618  1st Qu.: -0.0039849  1st Qu.: -0.0040374  1st Qu.: -0.0040403  1st Qu.: -0.097642
Median : -0.0005062  Median : -0.0004148  Median : -0.0005066  Median : -0.0004453  Median : -0.004051
Mean : -0.0001592  Mean : -0.0003885  Mean : -0.0004167  Mean : -0.0004045  Mean : -0.011772
3rd Qu.: -0.0048881  3rd Qu.: -0.0046277  3rd Qu.: -0.0047436  3rd Qu.: -0.0050338  3rd Qu.: -0.109569
Max.   : -0.0594595  Max.   : -0.0340658  Max.   : -0.1067194  Max.   : -0.0789000  Max.   : 2.998667

adj_close_new
Min.   :-0.0880141
1st Qu.: -0.0040403
Median : -0.0004453
Mean : -0.0004045
3rd Qu.: -0.0050338
Max.   : -0.1078900

```

Global Environment

Data

d_1000

1000 obs. of 12 variables

d_3000

3000 obs. of 12 variables

d

15447 obs. of 12 variables

Functions

randomRows

function (df, n)

successive_diff

function (x)

Files Plots Packages Help Viewer

New Folder

Delete

Rename

More

Home

Library

58

Oct 21, 2020, 9:25

cache

Custom Office Templates

desktop.ini

My Music

My Pictures

My Videos

Python Scripts

R

singlepageexploration

Visual Studio 2019

Zoom

b) For your samples, use the functions `mean()`, `max()`, `var()` and `quartile(.,25)` to compute the mean, maximum, variance and 1st quartile respectively for each column which has successive differences. Show your R code and the resulting values.

Do the same thing by using Excel. Show your Excel commands.

```

C:\Users\M. LAKSHMI MADHURI\Desktop\Repositories\Data_Science_2019501105\Data Mining\Final exam/ > #b
> #d_1000
> mean(d_1000$open_new)
[1] 0.0005955025
> mean(d_1000$high_new)
[1] 0.0004184797
> mean(d_1000$low_new)
[1] 0.0005022487
> mean(d_1000$close_new)
[1] 0.0003369592
> mean(d_1000$volume_new)
[1] 0.007551912
> mean(d_1000$adj_close_new)
[1] 0.0003369592
> var(d_1000$open_new)
[1] 8.714339e-05
> var(d_1000$high_new)
[1] 6.119132e-05
> var(d_1000$low_new)
[1] 8.313995e-05
> var(d_1000$close_new)
[1] 7.637739e-05
> var(d_1000$volume_new)
[1] 0.0327711
> var(d_1000$adj_close_new)
[1] 7.637739e-05
> max(d_1000$open_new)
[1] 0.1067121
> max(d_1000$high_new)
[1] 0.03439077
> max(d_1000$low_new)
[1] 0.0910832
> max(d_1000$close_new)
[1] 0.05732732
> max(d_1000$volume_new)
[1] 1.677175
> max(d_1000$adj_close_new)
[1] 0.05732732
> quantile(d_1000$open_new,0.25)
25%
-0.003961827
> quantile(d_1000$high_new,0.25)
25%
-0.003443228
> quantile(d_1000$low_new,0.25)
25%
-0.003897253
> quantile(d_1000$close_new,0.25)
25%
-0.004251294
> quantile(d_1000$volume_new,0.25)
25%
-0.1056329
> quantile(d_1000$adj_close_new,0.25)
25%
-0.004251294
> #d_3000

```

Data	
d_1000	1000 obs. of 12 variables
d_3000	3000 obs. of 12 variables
data	15447 obs. of 12 variables

Functions	
randomRows	function (df, n)
successive_diff	function (x)

Files Plots Packages Help Viewer

Zoom Export

```

C:\Users\M. LAKSHMI MADHURI\Desktop\Repositories\Data_Science_2019501105\Data Mining\Final exam/ > #d_3000
> mean(d_3000$open_new)
[1] 0.0003591911
> mean(d_3000$high_new)
[1] 0.0003884621
> mean(d_3000$low_new)
[1] 0.0004167
> mean(d_3000$close_new)
[1] 0.0004044752
> mean(d_3000$volume_new)
[1] 0.0171718
> mean(d_3000$adj_close_new)
[1] 0.0004044752
> var(d_3000$open_new)
[1] 8.509529e-05
> var(d_3000$high_new)
[1] 6.81047e-05
> var(d_3000$low_new)
[1] 8.768766e-05
> var(d_3000$close_new)
[1] 8.588174e-05
> var(d_3000$volume_new)
[1] 0.03939109
> var(d_3000$adj_close_new)
[1] 8.588174e-05
> max(d_3000$open_new)
[1] 0.05945946
> max(d_3000$high_new)
[1] 0.05406578
> max(d_3000$low_new)
[1] 0.1067194
> max(d_3000$close_new)
[1] 0.10789
> max(d_3000$volume_new)
[1] 2.996867
> max(d_3000$adj_close_new)
[1] 0.10789
> quantile(d_3000$open_new,0.25)
25%
-0.003965834
> quantile(d_3000$high_new,0.25)
25%
-0.003945885
> quantile(d_3000$low_new,0.25)
25%
-0.004170403
> quantile(d_3000$close_new,0.25)
25%
-0.00440009
> quantile(d_3000$volume_new,0.25)
25%
-0.09264194
> quantile(d_3000$adj_close_new,0.25)
25%
-0.00440009
> |
>

```

Data	
d_1000	1000 obs. of 12 variables
d_3000	3000 obs. of 12 variables
data	15447 obs. of 12 variables

Functions	
randomRows	function (df, n)
successive_diff	function (x)

Files Plots Packages Help Viewer

Zoom Export

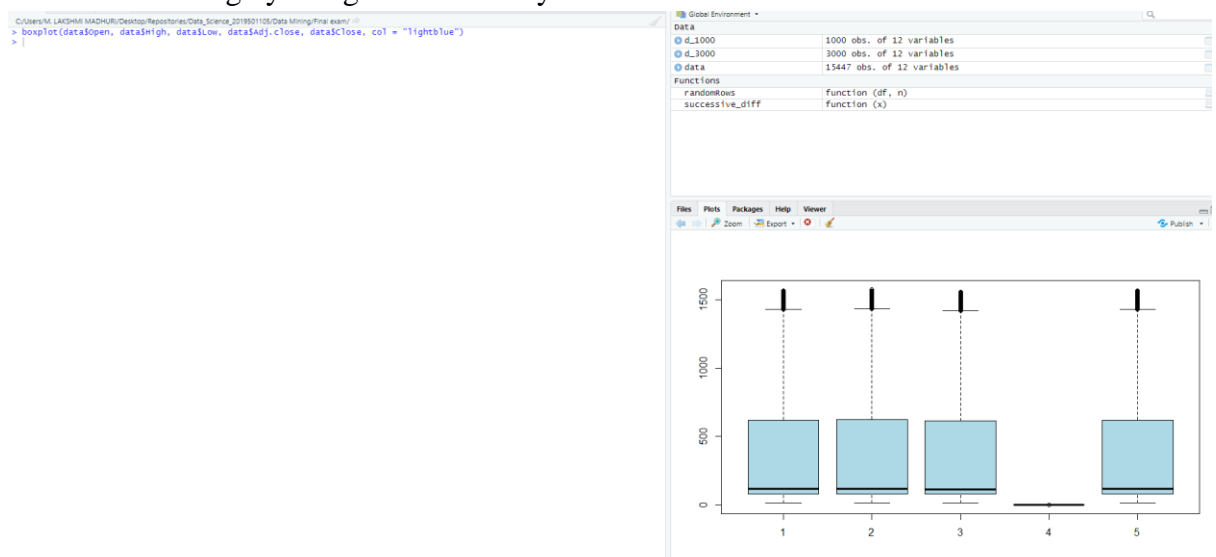
c) Compute the same quantities in part b on the entire data set and show your answers. How much do they differ from your answers in part b? Do you find any significant difference between two sample values like mean in comparison with entire data? If so what explanation you can give for that? Do the same thing by using Excel. Show your Excel commands.

```

C:\Users\M. LAKSHMI MADHURI\Desktop\Repositories\Data_Science_2019501105\Data Mining\Final exam/ > #1c
> mean(data$open_new)
[1] 0.000329528
> mean(data$high_new)
[1] 0.0003188991
> mean(data$low_new)
[1] 0.0003266191
> mean(data$close_new)
[1] 0.0003303709
> mean(data$volume_new)
[1] 0.02062874
> mean(data$adj.close_new)
[1] 0.0003303709
> var(data$open_new)
[1] 9.027493e-05
> var(data$high_new)
[1] 6.939914e-05
> var(data$low_new)
[1] 8.646474e-05
> var(data$close_new)
[1] 9.350347e-05
> var(data$volume_new)
[1] 0.09080738
> var(data$adj.close_new)
[1] 9.350347e-05
> max(data$open_new)
[1] 0.1067121
> max(data$high_new)
[1] 0.08037943
> max(data$low_new)
[1] 0.1067194
> max(data$close_new)
[1] 0.1158004
> max(data$volume_new)
[1] 26.51968
> max(data$adj.close_new)
[1] 0.1158004
> quantile(data$open_new,0.25)
25%
-0.004110794
> quantile(data$high_new,0.25)
25%
-0.003772912
> quantile(data$low_new,0.25)
25%
-0.003996406
> quantile(data$close_new,0.25)
25%
-0.004121264
> quantile(data$volume_new,0.25)
25%
-0.09553922
> quantile(data$adj.close_new,0.25)
25%
-0.004121264
>

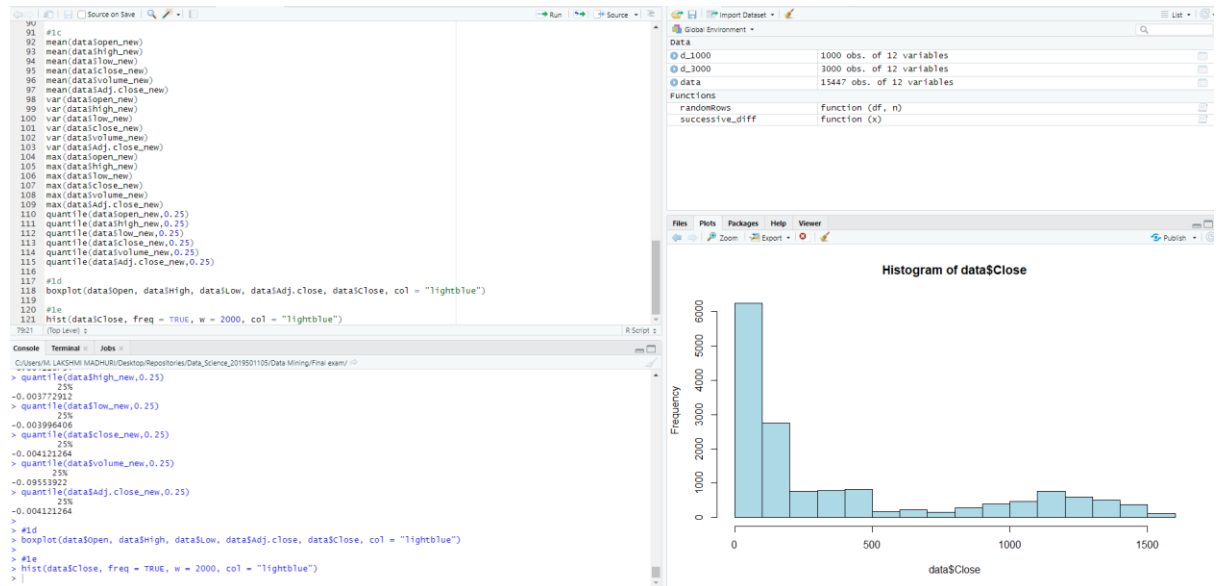
```

d) Use R to produce a single graph displaying a boxplot for open, close, high and low. Include the R commands and the plot. Do the same thing by using Excel. Show your Excel commands



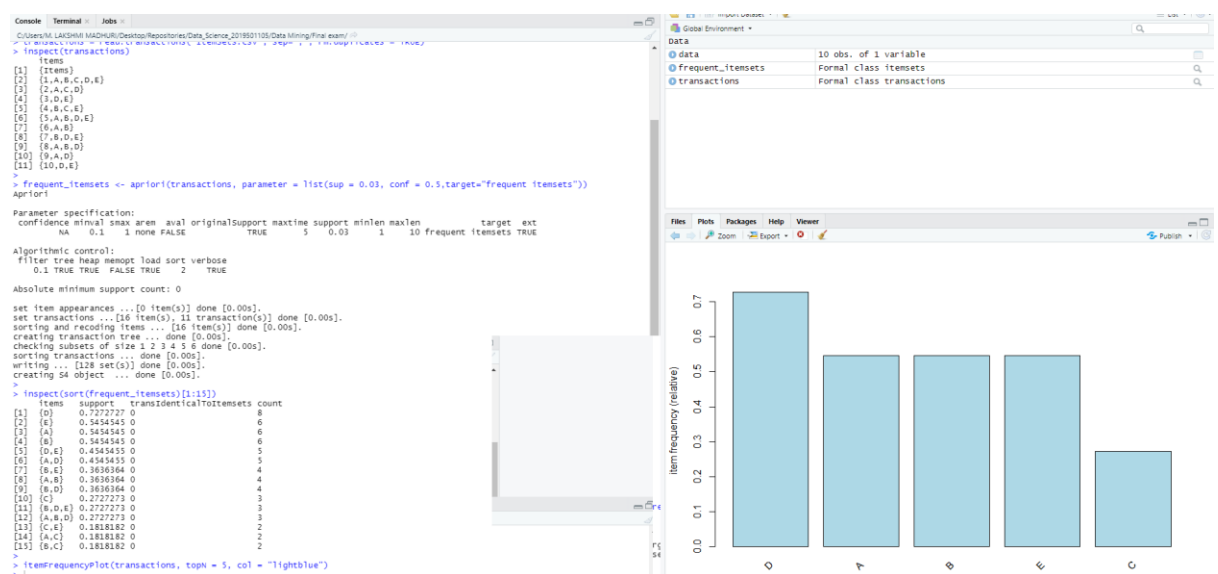
e) Use R to produce a frequency histogram for Close values. Use intervals of width 2000 beginning at 0. Include the R commands and the plot.

Do the same thing by using Excel. Show your Excel commands. (10+10=20M)

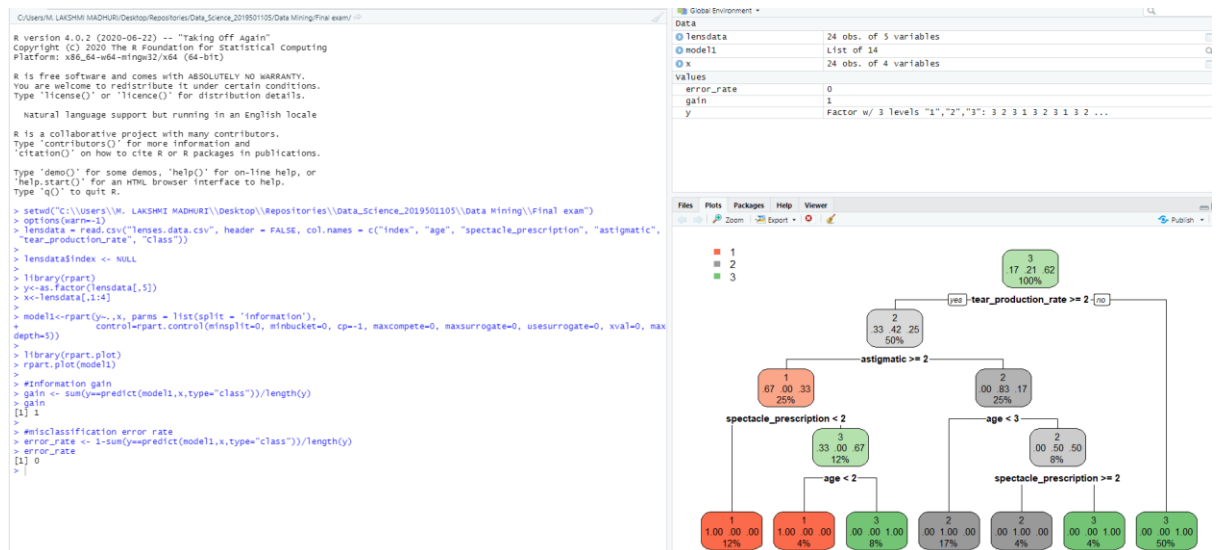


- Implement Apriori Algorithm or use built in packages to find out the frequent itemsets and generate rules for frequent itemsets. Trace and submit the program output for the following given dataset of transactions with a minimum support of 3. (10M)

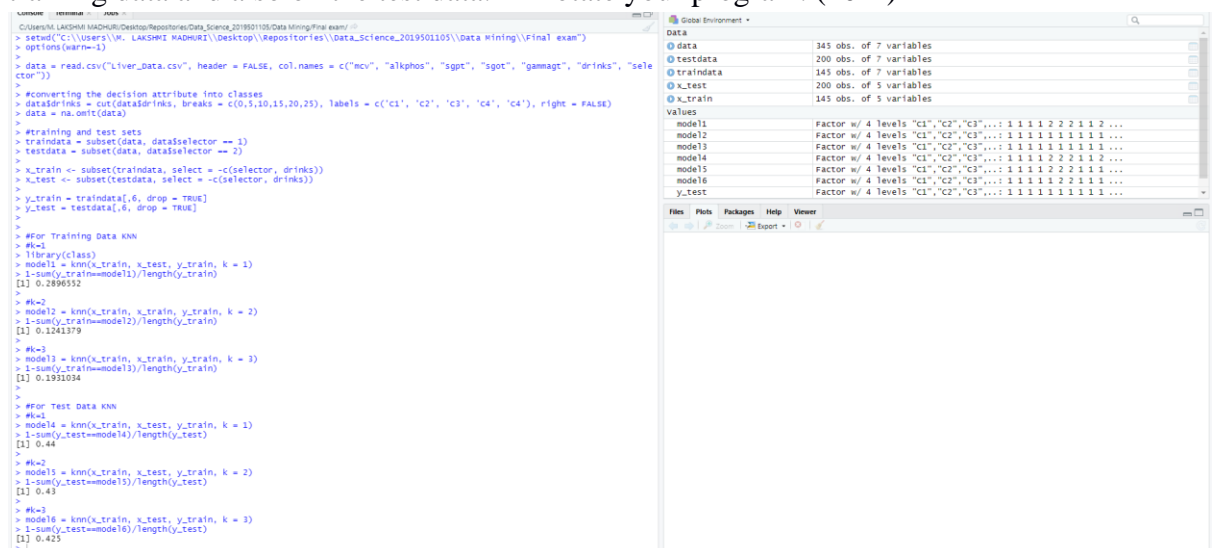
TID, Items
101, A, B, C, D, E
102, A, C, D
103, D, E
104, B, C, E
105, A, B, D, E
106, A, B
107, B, D, E
108, A, B, D
109, A, D
110, D, E



3. Build Decision Trees by using i) information gain and ii) misclassification error rate for Lenses Data Set provided at <http://archive.ics.uci.edu/ml/datasets/Lenses>. In terms of tree size what do you conclude comparing these two? (10M)



4. Fit 1, 2 and 3-nearest-neighbor classifiers to the Liver Disorders Data Set at <http://archive.ics.uci.edu/ml/datasets/Liver+Disorders> for measures Euclidean and cosine. Last but one column is a decision attribute. Replace decision values in to 4 classes ($0 \leq c_1 < 5$, $5 \leq c_2 < 10$, $10 \leq c_3 < 15$, $15 \leq c_4 \leq 20$). Last column is a data split column in to training and test sets. 1 means the object is used for training. 2 means the object is used for testing. Explain the input parameters you provided for the classifier. Compute the misclassification error on the training data and also on the test data. Annotate your program. (10M)



5. Use Support Vector machine for above problem. And compare the performance of both. Explain the input parameters you provided for the classifier. (10M)

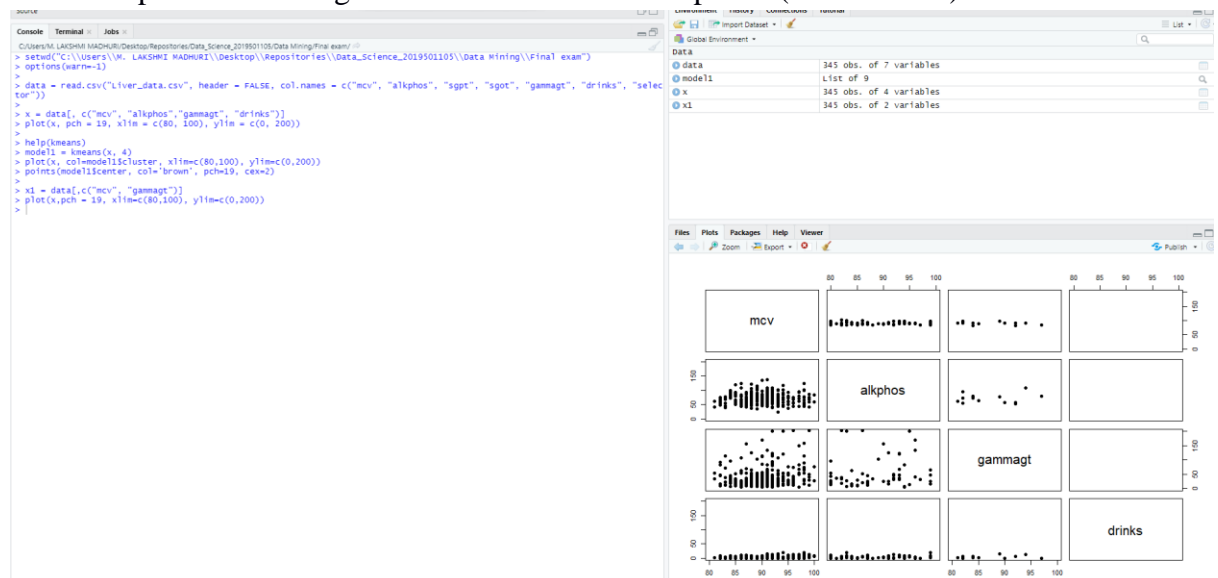
The screenshot shows an R script in RStudio. The script reads 'Liver_data.csv', preprocesses the data by splitting it into training and testing sets based on the 'drinks' variable, and then trains an SVM model using the 'e1071' package. The environment pane on the right shows the objects created: 'model' (a list of 29 SVM models), 'testdata' (200 obs. of 7 variables), 'traindata' (145 obs. of 7 variables), 'x_test' (200 obs. of 5 variables), 'x_train' (145 obs. of 5 variables), and 'y_train' (145 obs. of 5 variables). The console shows the execution of the script, including the training and testing steps.

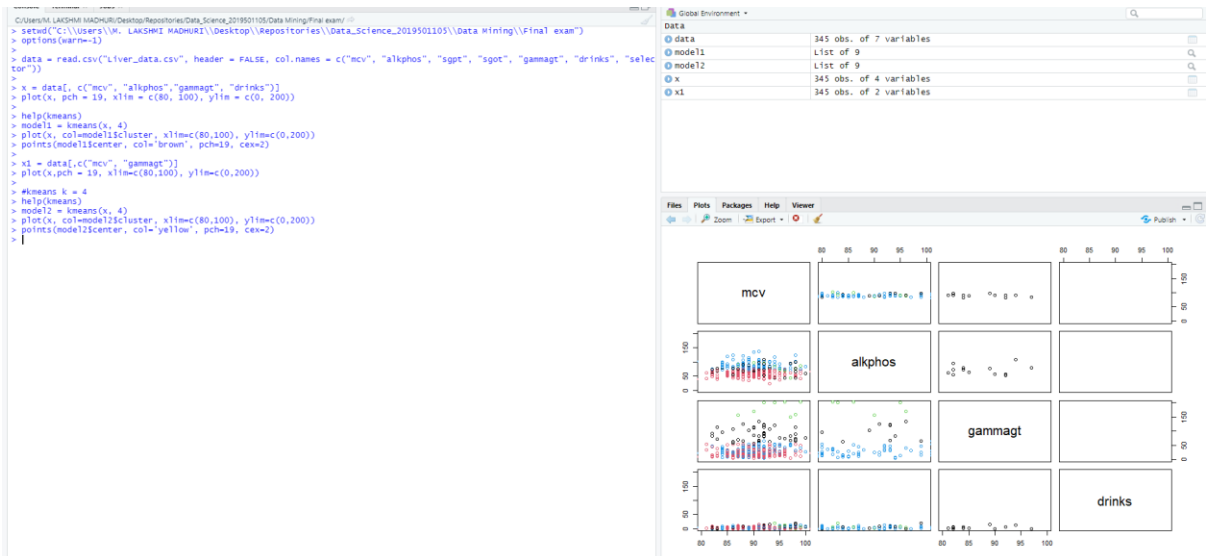
```

1 options(warn=-1)
2 data = read.csv("Liver_data.csv", header = FALSE, col.names = c("mcv", "alkphos", "sgpt", "sgot", "gammagt", "drinks", "select"))
3
4 data$drinks = cut(data$drinks, breaks = c(0,5,10,15,20,25), labels = c("C1", "C2", "C3", "C4", "C4"), right = FALSE)
5 data = na.omit(data)
6
7 #training and test sets
8 traindata = subset(data, data$selector == 1)
9 testdata = subset(data, data$selector == 2)
10
11 x_train <- subset(traindata, select = -c(selector, drinks))
12 x_test <- subset(testdata, select = -c(selector, drinks))
13
14 y_train = traindata[,6, drop = TRUE]
15 y_test = testdata[,6, drop = TRUE]
16
17 library(e1071)
18
19 #training data
20 model = svm(x_train, y_train)
21 1-sum(y_train==predict(model,x_train))/length(y_train)
22
23 #test data
24 1-sum(y_test==predict(model,x_test))/length(y_test)

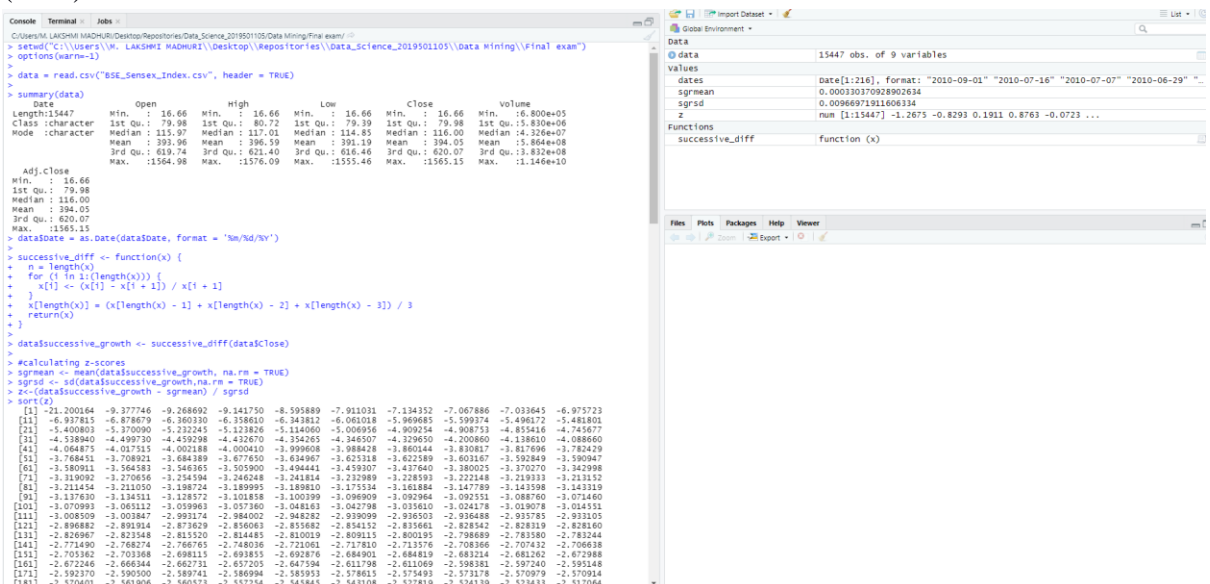
```

6. Create k-means clusters for k=4 for the Liver Disorders Data Set at <http://archive.ics.uci.edu/ml/datasets/Liver+Disorders> . Explain the input parameters you provided for the clustering algorithm. Plot the fitted cluster centers using a different color. Finally assign the cluster membership for the points to the nearest cluster center. Color the points according to their cluster membership. (10+10=20M)





7. Compute the misclassification error that would result if you used your clustering rule to classify the data by assigning the majority class of the cluster. (10M)
8. Consider the dataset BSE_Sensex_Index.csv. Create an extra column of successive growth rate for column close where the successive growth rate is defined as $(\text{value of day } x - \text{value of day } x-1) / \text{value of day } x-1$. Use a z score cut off of 3 to identify any outliers. List the respective dates from the csv file on which day these outliers fall. (10M)




```

C:/Users/M. LAKSHMI MADHURI/Desktop/Repositories/Data_Science_2019501105/Data Mining/Final exam/
[801] -1.480835 -1.480189 -1.480096 -1.479982 -1.479827 -1.479415 -1.478708 -1.478207 -1.477655 -1.477494
[811] -1.477464 -1.475556 -1.474689 -1.473455 -1.472859 -1.472644 -1.472447 -1.472386 -1.471786 -1.471159
[831] -1.470927 -1.470546 -1.470337 -1.470054 -1.468870 -1.468867 -1.467686 -1.466569 -1.466097 -1.465232
[841] -1.464190 -1.462344 -1.461426 -1.461369 -1.461190 -1.460763 -1.460113 -1.457398 -1.457349 -1.456155
[851] -1.455739 -1.455686 -1.455235 -1.455198 -1.455016 -1.454341 -1.453796 -1.453590 -1.453399 -1.453068
[861] -1.452906 -1.451952 -1.450818 -1.449287 -1.447390 -1.446905 -1.446385 -1.445375 -1.445363 -1.445034
[871] -1.445027 -1.444987 -1.441605 -1.440731 -1.440410 -1.440146 -1.439633 -1.436170 -1.436094 -1.435973
[881] -1.433590 -1.433567 -1.433210 -1.433186 -1.432202 -1.431674 -1.430946 -1.428230 -1.427024 -1.426090
[891] -1.425999 -1.424783 -1.423981 -1.423193 -1.422761 -1.422607 -1.422158 -1.422094 -1.421540 -1.420730
[901] -1.420679 -1.419969 -1.419312 -1.417832 -1.416308 -1.416218 -1.415207 -1.414047 -1.413762 -1.413268
[911] -1.411751 -1.410838 -1.410105 -1.408879 -1.408103 -1.407752 -1.406047 -1.404931 -1.403099 -1.402938
[921] -1.400542 -1.400333 -1.399500 -1.399271 -1.398907 -1.398231 -1.396046 -1.394765 -1.394478 -1.393779
[931] -1.393586 -1.393435 -1.393284 -1.393194 -1.392305 -1.389947 -1.389915 -1.389368 -1.388574 -1.388493
[941] -1.388149 -1.388138 -1.386722 -1.386128 -1.386004 -1.385513 -1.385181 -1.385039 -1.383151 -1.382864
[951] -1.382830 -1.382830 -1.382748 -1.379377 -1.379336 -1.379104 -1.378806 -1.378641 -1.376414 -1.375134
[961] -1.374976 -1.374949 -1.374707 -1.374220 -1.374176 -1.373894 -1.372775 -1.371636 -1.371541 -1.371007
[971] -1.369255 -1.369064 -1.367802 -1.366266 -1.365630 -1.364852 -1.362457 -1.361316 -1.361288 -1.360593
[981] -1.359933 -1.358861 -1.358227 -1.358002 -1.357692 -1.356240 -1.355144 -1.353995 -1.352105 -1.351562
[991] -1.349644 -1.349380 -1.349251 -1.348937 -1.348035 -1.347070 -1.346192 -1.346087 -1.346066 -1.345068
[ reached getoption("max.print") -- omitted 14447 entries ]
> data$zscores <- z
>
> #Dates of the outliers
> dates <- subset(data[,1],data[, "zscores"] >= 3.0 | data[, "zscores"] <= -3.0)
> print(dates)
[1] "2010-09-01" "2010-07-16" "2010-07-07" "2010-06-29" "2010-06-10" "2010-06-04" "2010-05-27" "2010-05-20"
[9] "2010-05-10" "2010-05-06" "2010-02-04" "2009-07-15" "2009-07-02" "2009-06-22" "2009-05-18" "2009-05-04"
[17] "2009-04-20" "2009-04-09" "2009-03-30" "2009-03-23" "2009-03-17" "2009-03-12" "2009-03-10" "2009-03-05"
[25] "2009-03-02" "2009-02-24" "2009-02-23" "2009-02-17" "2009-02-10" "2009-01-29" "2009-01-28" "2009-01-21"
[33] "2009-01-20" "2009-01-14" "2009-01-07" "2009-01-02" "2008-12-16" "2008-12-08" "2008-12-05" "2008-12-04"
[41] "2008-12-02" "2008-12-01" "2008-11-26" "2008-11-24" "2008-11-21" "2008-11-20" "2008-11-19" "2008-11-14"
[49] "2008-11-13" "2008-11-12" "2008-11-06" "2008-11-05" "2008-11-04" "2008-10-28" "2008-10-27" "2008-10-24"
[57] "2008-10-22" "2008-10-21" "2008-10-20" "2008-10-16" "2008-10-15" "2008-10-13" "2008-10-09" "2008-10-07"
[65] "2008-10-06" "2008-10-02" "2008-09-30" "2008-09-29" "2008-09-22" "2008-09-19" "2008-09-18" "2008-09-17"
[73] "2008-09-15" "2008-09-09" "2008-09-04" "2008-06-26" "2008-06-06" "2008-04-01" "2008-03-18" "2008-03-11"
[81] "2008-02-05" "2008-01-17" "2007-11-07" "2007-08-09" "2007-02-27" "2003-03-24" "2003-03-17" "2003-03-13"
[89] "2003-01-24" "2003-01-02" "2002-10-15" "2002-10-11" "2002-10-10" "2002-10-01" "2002-09-27" "2002-09-19"
[97] "2002-09-03" "2002-08-14" "2002-08-08" "2002-08-06" "2002-08-05" "2002-08-01" "2002-07-29" "2002-07-24"
[105] "2002-07-22" "2002-07-19" "2002-07-10" "2002-07-05" "2002-05-08" "2001-09-24" "2001-09-20" "2001-09-17"
[113] "2001-04-18" "2001-04-05" "2001-04-03" "2001-03-12" "2001-01-03" "2000-12-20" "2000-12-05" "2000-10-19"
[121] "2000-10-13" "2000-05-30" "2000-04-25" "2000-04-17" "2000-04-14" "2000-03-16" "2000-02-18" "2000-01-04"
[129] "1999-10-28" "1998-10-15" "1998-10-01" "1998-09-30" "1998-09-23" "1998-09-11" "1998-09-08" "1998-09-01"
[137] "1998-08-31" "1998-08-27" "1998-08-04" "1998-01-09" "1997-10-28" "1997-10-27" "1997-09-02" "1996-03-08"
[145] "1991-11-15" "1991-08-21" "1991-01-17" "1990-08-27" "1990-08-23" "1990-08-06" "1989-10-13" "1988-05-31"
[153] "1988-04-14" "1988-01-08" "1988-01-04" "1987-12-03" "1987-11-30" "1987-11-09" "1987-10-29" "1987-10-26"
[161] "1987-10-22" "1987-10-21" "1987-10-20" "1987-10-19" "1987-10-16" "1987-10-14" "1986-09-11" "1986-07-07"
[169] "1982-11-30" "1982-11-03" "1982-10-25" "1982-10-06" "1982-08-20" "1982-08-17" "1981-08-24" "1980-04-22"
[177] "1980-03-24" "1980-03-17" "1979-10-09" "1978-11-01" "1975-01-27" "1974-11-18" "1974-10-29" "1974-10-23"
[185] "1974-10-09" "1974-10-07" "1974-09-19" "1974-09-05" "1974-08-30" "1974-07-12" "1974-07-08" "1973-12-26"
[193] "1973-11-26" "1973-11-19" "1973-05-24" "1971-08-16" "1970-05-27" "1963-11-26" "1962-10-24" "1962-06-28"
[201] "1962-06-04" "1962-05-29" "1962-05-28" "1961-04-18" "1961-04-17" "1957-10-23" "1957-10-21" "1955-10-10"
[209] "1955-09-26" "1955-07-06" "1955-06-06" "1953-02-09" "1950-12-04" "1950-11-28" "1950-06-29" "1950-06-26"
>
> #Storing
> write.csv(dates, "DateOutliersData.csv", quote = FALSE, row.names = TRUE)
>

```