Data Mining- Lab Exam

Time: 24 hours

Marks:100

Open a document and update document with your answers for each question and submit it.

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.

```
Console   Terminal ×   Jobs ×
F:/Data Science/DataScience_2019501111/Data Mining/Final exam/
      Open             High             Low              Close
 Min.   :  17.08   Min.   :  17.08   Min.   :  17.08   Min.   :  17.08
 1st Qu.:  83.43   1st Qu.:  84.07   1st Qu.:  82.50   1st Qu.:  83.17
 Median : 116.45   Median : 117.59   Median : 115.03   Median : 116.34
 Mean   : 398.28   Mean   : 401.03   Mean   : 395.52   Mean   : 398.50
 3rd Qu.: 650.67   3rd Qu.: 654.12   3rd Qu.: 644.93   3rd Qu.: 648.62
 Max.   :1522.19   Max.   :1526.45   Max.   :1500.74   Max.   :1506.34
     Volume            Adj.Close          open_new             high_new
 Min.   :7.800e+05   Min.   :  17.08   Min.   :-0.0582780   Min.   :-0.0432817
 1st Qu.:9.030e+06   1st Qu.:  83.17   1st Qu.:-0.0039618   1st Qu.:-0.0034432
 Median :4.390e+07   Median : 116.34   Median : 0.0005554   Median : 0.0003948
 Mean   :5.964e+08   Mean   : 398.50   Mean   : 0.0005955   Mean   : 0.0004185
 3rd Qu.:4.035e+08   3rd Qu.: 648.62   3rd Qu.: 0.0050955   3rd Qu.: 0.0045302
 Max.   :8.926e+09   Max.   :1506.34   Max.   : 0.1067121   Max.   : 0.0343908
    low_new             close_new            volume_new          Adj.close_new
 Min.   :-0.0474458   Min.   :-0.0402908   Min.   :-0.718888   Min.   :-0.0402908
 1st Qu.:-0.0038973   1st Qu.:-0.0042513   1st Qu.:-0.105633   1st Qu.:-0.0042513
 Median : 0.0008122   Median : 0.0003301   Median :-0.002597   Median : 0.0003301
 Mean   : 0.0005022   Mean   : 0.0003370   Mean   : 0.007552   Mean   : 0.0003370
 3rd Qu.: 0.0047861   3rd Qu.: 0.0048696   3rd Qu.: 0.103772   3rd Qu.: 0.0048696
 Max.   : 0.0910833   Max.   : 0.0573273   Max.   : 1.677175   Max.   : 0.0573273
> data_3000 = randomRows(data, 3000)
>
> summary(data_3000)
      Open             High             Low              Close
 Min.   :  16.72   Min.   :  16.72   Min.   :  16.72   Min.   :  16.72
 1st Qu.:  79.61   1st Qu.:  80.10   1st Qu.:  78.94   1st Qu.:  79.42
 Median : 113.11   Median : 114.21   Median : 111.98   Median : 112.88
 Mean   : 379.96   Mean   : 382.57   Mean   : 377.36   Mean   : 380.19
 3rd Qu.: 495.77   3rd Qu.: 497.82   3rd Qu.: 494.57   3rd Qu.: 497.14
 Max.   :1556.51   Max.   :1563.03   Max.   :1554.09   Max.   :1561.80
     Volume            Adj.Close          open_new             high_new
 Min.   :7.400e+05   Min.   :  16.72   Min.   :-0.0871188   Min.   :-0.0685302
 1st Qu.:5.972e+06   1st Qu.:  79.42   1st Qu.:-0.0039658   1st Qu.:-0.0039459
 Median :4.013e+07   Median : 112.88   Median : 0.0005062   Median : 0.0004148
 Mean   :5.449e+08   Mean   : 380.19   Mean   : 0.0003592   Mean   : 0.0003885
 3rd Qu.:3.181e+08   3rd Qu.: 497.14   3rd Qu.: 0.0049885   3rd Qu.: 0.0046277
 Max.   :1.146e+10   Max.   :1561.80   Max.   : 0.0594595   Max.   : 0.0540658
    low_new             close_new            volume_new          Adj.close_new
 Min.   :-0.0821116   Min.   :-0.0680141   Min.   :-0.754927   Min.   :-0.0680141
```

```
Max.   :1.146e+10   Max.   :1561.80   Max.   : 0.0594595   Max.   : 0.0540658
   low_new             close_new          volume_new          Adj.close_new
Min.   :-0.0821116   Min.   :-0.0680141   Min.   :-0.754927   Min.   :-0.0680141
1st Qu.:-0.0041704   1st Qu.:-0.0044001   1st Qu.:-0.092642   1st Qu.:-0.0044001
Median : 0.0005606   Median : 0.0004455   Median : 0.004051   Median : 0.0004455
Mean   : 0.0004167   Mean   : 0.0004045   Mean   : 0.017172   Mean   : 0.0004045
3rd Qu.: 0.0047436   3rd Qu.: 0.0050338   3rd Qu.: 0.109569   3rd Qu.: 0.0050338
Max.   : 0.1067194   Max.   : 0.1078900   Max.   : 2.996867   Max.   : 0.1078900
>
```

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.

Console Terminal Jobs

F:/Data Science/DataScience_2019501111/Data Mining/Final exam/

```
> mean(data_1000$open_new)
[1] 0.0005955025
> mean(data_1000$high_new)
[1] 0.0004184797
> mean(data_1000$low_new)
[1] 0.0005022487
> mean(data_1000$close_new)
[1] 0.0003369592
> mean(data_1000$volume_new)
[1] 0.007551912
> mean(data_1000$Adj.close_new)
[1] 0.0003369592
> var(data_1000$open_new)
[1] 8.714339e-05
> var(data_1000$high_new)
[1] 6.119132e-05
> var(data_1000$low_new)
[1] 8.313995e-05
> var(data_1000$close_new)
[1] 7.637739e-05
> var(data_1000$volume_new)
[1] 0.0327711
> var(data_1000$Adj.close_new)
[1] 7.637739e-05
> max(data_1000$open_new)
[1] 0.1067121
> max(data_1000$high_new)
[1] 0.03439077
> max(data_1000$low_new)
[1] 0.09108332
> max(data_1000$close_new)
[1] 0.05732732
> max(data_1000$volume_new)
[1] 1.677175
> max(data_1000$Adj.close_new)
[1] 0.05732732
> quantile(data_1000$open_new,0.25)
         25%
-0.003961827
> quantile(data_1000$high_new,0.25)
```

Console Terminal Jobs

F:/Data Science/DataScience_2019501111/Data Mining/Final exam/

```
-0.003961827
> quantile(data_1000$high_new,0.25)
         25% 
-0.003443228 
> quantile(data_1000$low_new,0.25)
         25% 
-0.003897353 
> quantile(data_1000$close_new,0.25)
         25% 
-0.004251294 
> quantile(data_1000$volume_new,0.25)
       25% 
-0.1056329 
> quantile(data_1000$Adj.close_new,0.25)
         25% 
-0.004251294 
> 
> 
> mean(data_3000$open_new)
[1] 0.0003591911
> mean(data_3000$high_new)
[1] 0.0003884621
> mean(data_3000$low_new)
[1] 0.0004167
> mean(data_3000$close_new)
[1] 0.0004044752
> mean(data_3000$volume_new)
[1] 0.0171718
> mean(data_3000$Adj.close_new)
[1] 0.0004044752
> var(data_3000$open_new)
[1] 8.509529e-05
> var(data_3000$high_new)
[1] 6.81047e-05
> var(data_3000$low_new)
[1] 8.768766e-05
> var(data_3000$close_new)
[1] 8.588174e-05
> var(data_3000$volume_new)
[1] 0.03939109
```

Console Terminal Jobs

F:/Data Science/DataScience_2019501111/Data Mining/Final exam/

```
[1] 6.81047e-05
> var(data_3000$low_new)
[1] 8.768766e-05
> var(data_3000$close_new)
[1] 8.588174e-05
> var(data_3000$volume_new)
[1] 0.03939109
> var(data_3000$Adj.close_new)
[1] 8.588174e-05
> max(data_3000$open_new)
[1] 0.05945946
> max(data_3000$high_new)
[1] 0.05406578
> max(data_3000$low_new)
[1] 0.1067194
> max(data_3000$close_new)
[1] 0.10789
> max(data_3000$volume_new)
[1] 2.996867
> max(data_3000$Adj.close_new)
[1] 0.10789
> quantile(data_3000$open_new,0.25)
         25%
-0.003965834
> quantile(data_3000$high_new,0.25)
         25%
-0.003945885
> quantile(data_3000$low_new,0.25)
         25%
-0.004170403
> quantile(data_3000$close_new,0.25)
        25%
-0.00440009
> quantile(data_3000$volume_new,0.25)
        25%
-0.09264194
> quantile(data_3000$Adj.close_new,0.25)
        25%
-0.00440009
>
```

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.

Console | Terminal | Jobs

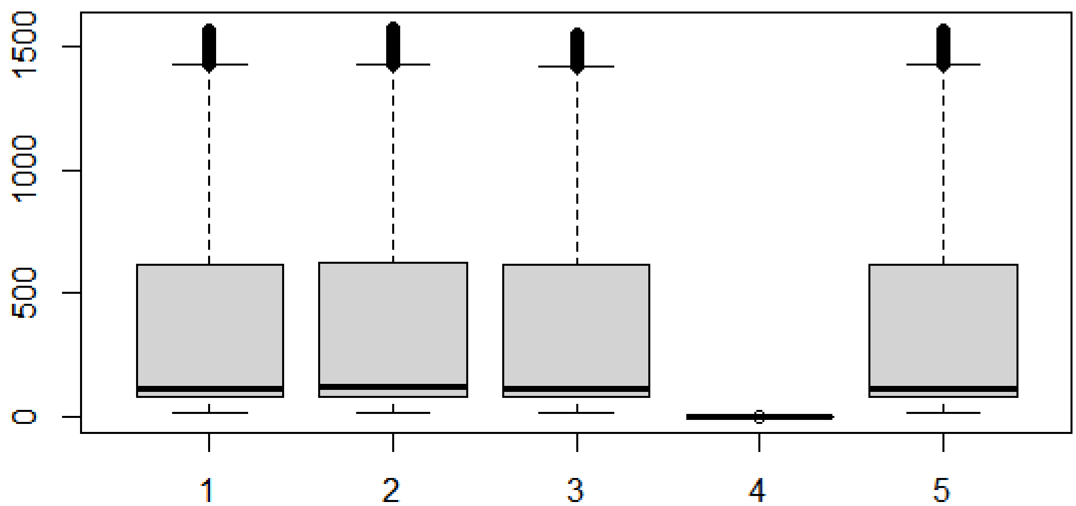
F:/Data Science/DataScience_2019501111/Data Mining/Final exam/

```
> 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)
```

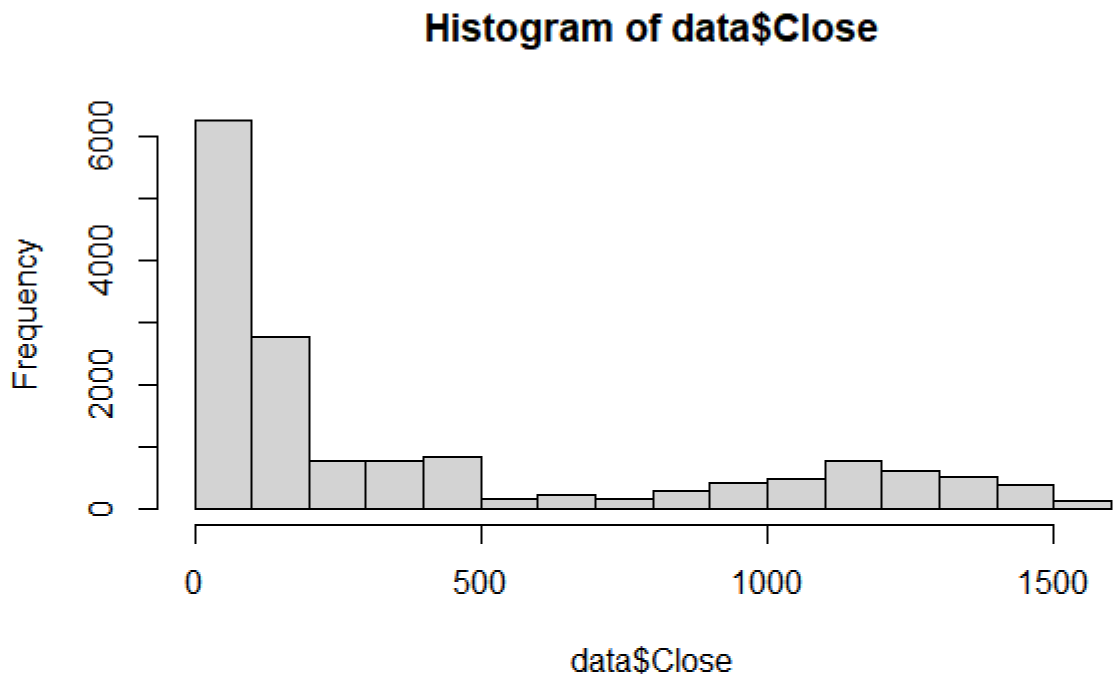

> quantile(data$open_new,0.25)
25%
-0.004110794
> quantile(data$high_new,0.25)


```
         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)



2. 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

Console Terminal Jobs

F:/Data Science/DataScience_2019501111/Data Mining/Final exam/

```
> write.csv(data, "ItemList.csv", quote = FALSE, row.names = TRUE)
> transactions = read.transactions("ItemList.csv", sep=',', rm.duplicates = TRUE)
> inspect(transactions)
     items
[1]  {Items}
[2]  {1,A,B,C,D,E}
[3]  {2,A,C,D}
[4]  {3,D,E}
[5]  {4,B,C,E}
[6]  {5,A,B,D,E}
[7]  {6,A,B}
[8]  {7,B,D,E}
[9]  {8,A,B,D}
[10] {9,A,D}
[11] {10,D,E}
>
> freqItemsets <- apriori(transactions, parameter = list(sup = 0.03, conf = 0.5,target
="frequent itemsets"))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen
         NA    0.1    1 none FALSE            TRUE       5    0.03      1     10
            target  ext
 frequent itemsets TRUE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 0

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[16 item(s), 11 transaction(s)] done [0.00s].
sorting and recoding items ... [16 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 5 6 done [0.00s].
sorting transactions ... done [0.00s].
writing ... [128 set(s)] done [0.00s].
creating S4 object  ... done [0.06s].
```

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)

Console Terminal Jobs

F:/Data Science/DataScience_2019501111/Data Mining/Final exam/

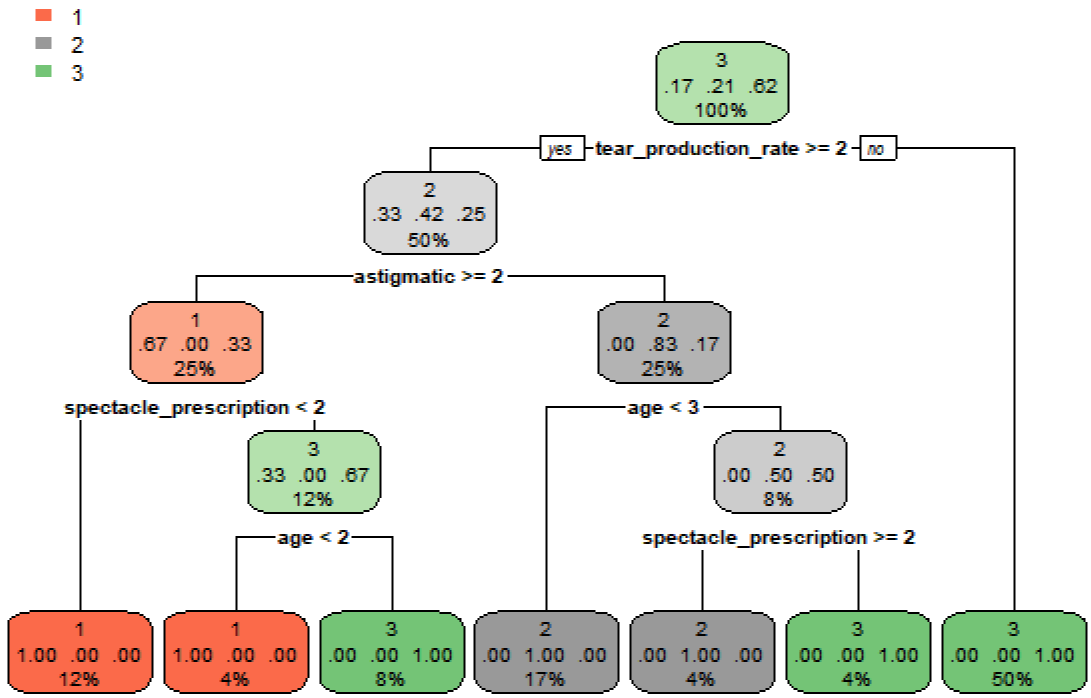
```
The downloaded binary packages are in
        C:\Users\hp\AppData\Local\Temp\RtmpOqkJyh\downloaded_packages
> install.packages("rpart.plot")
WARNING: Rtools is required to build R packages but is not currently installed. Please
 download and install the appropriate version of Rtools before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into 'C:/Users/hp/Documents/R/win-library/4.0'
(as 'lib' is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.0/rpart.plot_3.0.9.zip'
Content type 'application/zip' length 1033909 bytes (1009 KB)
downloaded 1009 KB

package 'rpart.plot' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\hp\AppData\Local\Temp\RtmpOqkJyh\downloaded_packages
> setwd("F:\\Data Science\\DataScience_2019501111\\Data Mining\\Final exam")
> lensdata = read.csv("lenses.data.csv", header = FALSE, col.names = c("index", "age",
 "spectacle_prescription", "astigmatic", "tear_production_rate", "Class"))
>
> lensdata$index <- NULL
>
> library(rpart)
Warning message:
package 'rpart' was built under R version 4.0.3
> y<-as.factor(lensdata[,5])
> x<-lensdata[,1:4]
>
> model1<-rpart(y~.,x, parms = list(split = 'information'),
+              control=rpart.control(minsplit=0,minbucket=0,cp=-1, maxcompete=0, maxsu
rrogate=0, usesurrogate=0, xval=0,maxdepth=5))
>
>
> library(rpart.plot)
Warning message:
package 'rpart.plot' was built under R version 4.0.3
> rpart.plot(model1)
> |
```

```
>
>
> library(rpart.plot)
> rpart.plot(model1)
>
> #Information gain
> gain <- sum(y==predict(model1,x,type="class"))/length(y)
> gain
[1] 1
>
> #misclassification
> error_rate <- 1-sum(y==predict(model1,x,type="class"))/length(y)
> error_rate
[1] 0
> |
```



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<=c1<5, 5<=c2<10, 10<=c3<15, 15<=c4<=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)

Source

Console Terminal Jobs

F:/Data Science/DataScience_2019501111/Data Mining/Final exam/

```
> setwd("F:\\Data Science\\DataScience_2019501111\\Data Mining\\Final exam")
> data = read.csv("Liver_Data.csv", header = FALSE, col.names = c("mcv", "alkphos", "sg
pt", "sgot", "gammagt", "drinks", "selector"))
>
> #converting the decision attribute into classes
> data$drinks = cut(data$drinks, breaks = c(0,5,10,15,20,25), labels = c('C1', 'C2', 'C
3', 'C4', 'C4'), right = FALSE)
> data = na.omit(data)
>
> #traing and test sets
> traindata = subset(data, data$selector == 1)
> testdata = subset(data, data$selector == 2)
>
> x_train <- subset(traindata, select = -c(selector, drinks))
> x_test <- subset(testdata, select = -c(selector, drinks))
>
> y_train = traindata[,6, drop = TRUE]
> y_test = testdata[,6, drop = TRUE]
>
>
> #For Training Data
> #knn for k=1
> library(class)
> model1 = knn(x_train, x_test, y_train, k = 1)
> 1-sum(y_train==model1)/length(y_train) # 0
[1] 0.2827586
Warning messages:
1: In `==.default`(y_train, model1) :
  longer object length is not a multiple of shorter object length
2: In is.na(e1) | is.na(e2) :
  longer object length is not a multiple of shorter object length
>
> #knn for k=2
> model2 = knn(x_train, x_train, y_train, k = 2)
> 1-sum(y_train==model2)/length(y_train) #  0.1586207
[1] 0.1655172
>
> #knn for k=3
> model3 = knn(x_train, x_train, y_train, k = 3)
> 1-sum(y_train==model3)/length(y_train) # 0.2137931
```

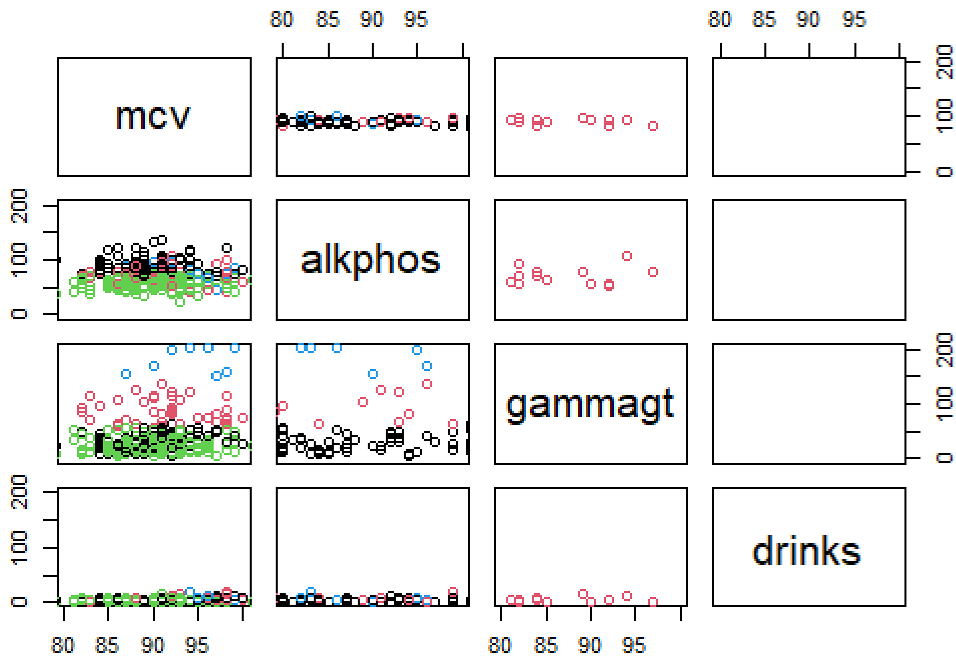
```
>
> #knn for k=2
> model2 = knn(x_train, x_train, y_train, k = 2)
> 1-sum(y_train==model2)/length(y_train) #  0.1586207
[1] 0.1655172
>
> #knn for k=3
> model3 = knn(x_train, x_train, y_train, k = 3)
> 1-sum(y_train==model3)/length(y_train) # 0.2137931
[1] 0.2068966
>
>
> #For Test Data
> #knn for k=1
> model4 = knn(x_train, x_test, y_train, k = 1)
> 1-sum(y_test==model4)/length(y_test) # 0.44
[1] 0.445
>
> #knn for k=2
> model5 = knn(x_train, x_test, y_train, k = 2)
> 1-sum(y_test==model5)/length(y_test) # 0.42
[1] 0.445
>
> #knn for k=3
> model6 = knn(x_train, x_test, y_train, k = 3)
> 1-sum(y_test==model6)/length(y_test) # 0.405
[1] 0.4
>
```

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

```
package 'e1071' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\hp\AppData\Local\Temp\RtmpOqkJyh\downloaded_packages
> setwd("F:\\Data Science\\DataScience_2019501111\\Data Mining\\Final exam")
> data = read.csv("Liver_data.csv", header = FALSE, col.names = c("mcv", "alkphos", "sg
pt", "sgot", "gammagt", "drinks", "selector"))
>
> #converting the decision attribute into classes
> data$drinks = cut(data$drinks, breaks = c(0,5,10,15,20,25), labels = c('C1', 'C2', 'C
3', 'C4', 'C4'), right = FALSE)
> data = na.omit(data)
>
> #traing and test sets
> traindata = subset(data, data$selector == 1)
> testdata = subset(data, data$selector == 2)
>
> x_train <- subset(traindata, select = -c(selector, drinks))
> x_test <- subset(testdata, select = -c(selector, drinks))
>
> y_train = traindata[,6, drop = TRUE]
> y_test = testdata[,6, drop = TRUE]
>
> library(e1071)
Warning message:
package 'e1071' was built under R version 4.0.3
>
> #For training
> model = svm(x_train, y_train)
> 1-sum(y_train==predict(model,x_train))/length(y_train) # 0.2137931
[1] 0.2137931
>
> #For test data
> 1-sum(y_test==predict(model,x_test))/length(y_test) #  0.285
[1] 0.285
>
> #The misclassification error is high for KNN so, we can prefer SVM over KNN
> |
```

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
(value of day x- value of day x-1)/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)

Source

Console Terminal Jobs

F:/Data Science/DataScience_2019501111/Data Mining/Final exam/

```
> data = read.csv("BSE_Sensex_Index.csv", header = TRUE)
>
> View(data)
> summary(data)
     Date                Open              High              Low         
 Length:15447       Min.   :  16.66   Min.   :  16.66   Min.   :  16.66  
 Class :character   1st Qu.:  79.98   1st Qu.:  80.72   1st Qu.:  79.39  
 Mode  :character   Median : 115.97   Median : 117.01   Median : 114.85  
                    Mean   : 393.96   Mean   : 396.59   Mean   : 391.19  
                    3rd Qu.: 619.74   3rd Qu.: 621.40   3rd Qu.: 616.46  
                    Max.   :1564.98   Max.   :1576.09   Max.   :1555.46  
     Close             Volume            Adj.Close      
 Min.   :  16.66   Min.   :6.800e+05   Min.   :  16.66  
 1st Qu.:  79.98   1st Qu.:5.830e+06   1st Qu.:  79.98  
 Median : 116.00   Median :4.326e+07   Median : 116.00  
 Mean   : 394.05   Mean   :5.864e+08   Mean   : 394.05  
 3rd Qu.: 620.07   3rd Qu.:3.832e+08   3rd Qu.: 620.07  
 Max.   :1565.15   Max.   :1.146e+10   Max.   :1565.15  
> data$Date = as.Date(data$Date, format='%m/%d/%Y')
>
> successive_difference <- function(x) {
+   n = length(x)
+   for (i in 1:(length(x))) {
+     x[i] <- (x[i] - x[i + 1]) / x[i + 1]
+   }
+   x[length(x)] = (x[length(x) - 1] + x[length(x) - 2] + x[length(x) - 3]) / 3
+   return(x)
+ }
>
> data$successive_growth <- successive_difference(data$Close)
>
>
> #calculating z-scores
> sgrmean <- mean(data$successive_growth, na.rm=TRUE)
> sgrsd <- sd(data$successive_growth,na.rm=TRUE)
> z<-(data$successive_growth - sgrmean) / sgrsd
> sort(z)
   [1] -21.200164  -9.377746  -9.268692  -9.141750  -8.595889  -7.911031  -7.134352
   [8]  -7.067886  -7.033645  -6.975723  -6.937815  -6.878679  -6.360330  -6.358610
  [15]  -6.343812  -6.061018  -5.969685  -5.599374  -5.496172  -5.481801  -5.400803
```

Source

Console Terminal Jobs

F:/Data Science/DataScience_2019501111/Data Mining/Final exam/

```
 [785]  -1.504077  -1.503930  -1.503767  -1.502479  -1.502254  -1.501164  -1.499653
 [792]  -1.498687  -1.496642  -1.496166  -1.495191  -1.493551  -1.493444  -1.493292
 [799]  -1.491062  -1.490834  -1.490794  -1.489958  -1.485868  -1.484670  -1.483748
 [806]  -1.483516  -1.482876  -1.482448  -1.482356  -1.481304  -1.480835  -1.480189
 [813]  -1.480096  -1.479982  -1.479827  -1.479415  -1.478708  -1.478207  -1.477655
 [820]  -1.477494  -1.477464  -1.475556  -1.474689  -1.473455  -1.472859  -1.472644
 [827]  -1.472447  -1.472386  -1.471786  -1.471159  -1.470927  -1.470546  -1.470337
 [834]  -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
 [848]  -1.457398  -1.457349  -1.456155  -1.455739  -1.455686  -1.455235  -1.455198
 [855]  -1.455016  -1.454341  -1.453796  -1.453590  -1.453399  -1.453068  -1.452906
 [862]  -1.451952  -1.450818  -1.449287  -1.447390  -1.446905  -1.446385  -1.445375
 [869]  -1.445363  -1.445034  -1.445027  -1.444987  -1.441605  -1.440731  -1.440410
 [876]  -1.440146  -1.439633  -1.436170  -1.436094  -1.435973  -1.433590  -1.433567
 [883]  -1.433210  -1.433186  -1.432202  -1.431674  -1.430946  -1.428230  -1.427024
 [890]  -1.426090  -1.425999  -1.424783  -1.423981  -1.423193  -1.422761  -1.422607
 [897]  -1.422158  -1.422094  -1.421540  -1.420730  -1.420679  -1.419969  -1.419312
 [904]  -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
 [918]  -1.404931  -1.403099  -1.402938  -1.400542  -1.400333  -1.399500  -1.399271
 [925]  -1.398907  -1.398231  -1.396046  -1.394765  -1.394478  -1.393779  -1.393586
 [932]  -1.393435  -1.393284  -1.393194  -1.392305  -1.389947  -1.389915  -1.389368
 [939]  -1.388574  -1.388493  -1.388149  -1.388138  -1.386722  -1.386128  -1.386004
 [946]  -1.385513  -1.385181  -1.385039  -1.383151  -1.382864  -1.382830  -1.382830
 [953]  -1.382748  -1.379377  -1.379336  -1.379104  -1.378806  -1.378641  -1.376414
 [960]  -1.375134  -1.374976  -1.374949  -1.374707  -1.374220  -1.374176  -1.373894
 [967]  -1.372775  -1.371636  -1.371541  -1.371007  -1.369255  -1.369064  -1.367802
 [974]  -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
 [988]  -1.353995  -1.352105  -1.351562  -1.349644  -1.349380  -1.349251  -1.348937
 [995]  -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)
> View(dates)
>
> write.csv(dates, "OutliersDatesData_.csv", quote = FALSE, row.names = TRUE)
>
```