The University of Texas at Dallas

BA with R

Presented By

Bhupesh Kumar Srivastava
Net ID: BKS210000

Decision Tree & PCA

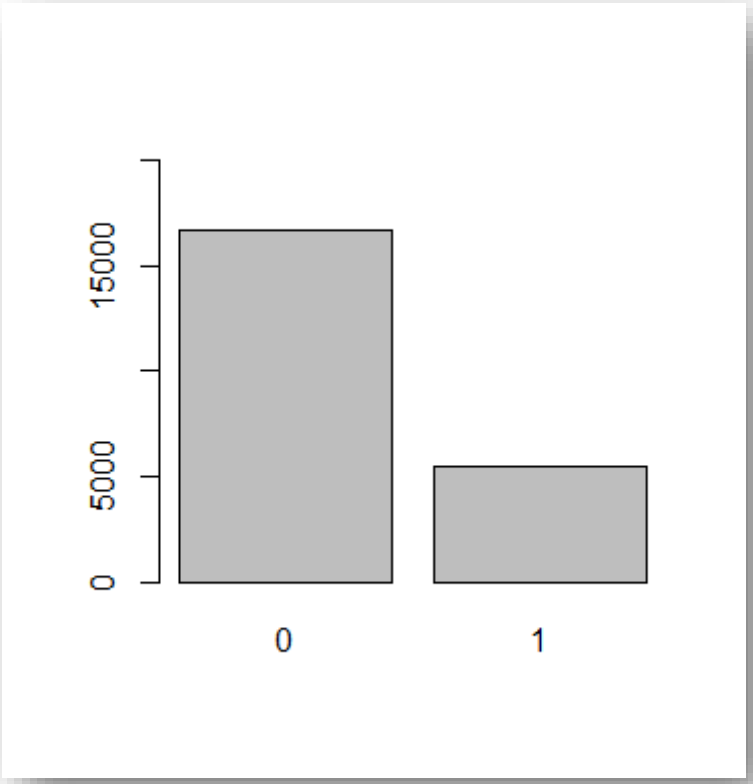
(a) Importing data to R code from the directory and saving it to organics

```
R>      library(readxl)
        organics <- read_excel("D:/MSBA/BUAN 6356/HW2/organics.xlsx")
        View(organics)
```

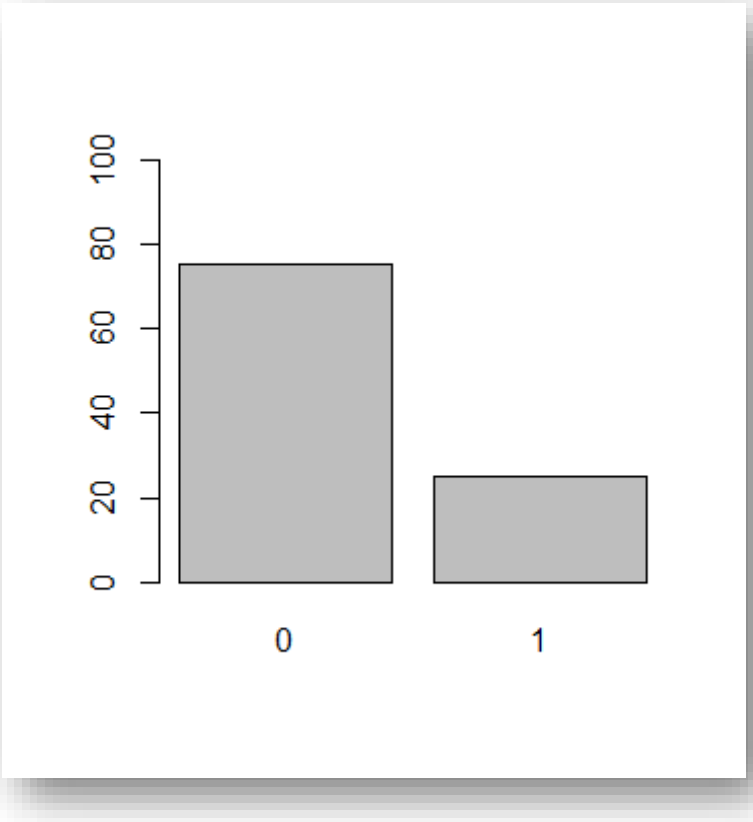
(b) Examine the distribution of the target variable:

(1) Plot a bar chart to show the number of observations in each category

```
R>      count <- table(organics$TargetBuy)
        count
        persentage <- count*100/22222
        persentage
        barplot(count)
```



Relative frequency Graph



(2) plot a bar chart to show the frequency of observations in each category

Individuals purchased organic products - 5505
Approximate proportion of individuals who purchased organic products- 24.77%

(c) The variable DemClusterGroup contains collapsed levels of the variable DemCluster. Presume that, based on previous experience, you believe that DemClusterGroup is sufficient for this type of modeling effort. Exclude the variable DemCluster for the analysis. Copy the R code used below.

```
R>          org <- organics[,-4]
            View(org)
            #removing ID and target amount
            org <- org[,-c(1,12)]
            View(org)
```

(d) As noted above, only TargetBuy will be used for this analysis and should have a role of target. Can TargetAmt be used as an input for a model used to predict TargetBuy? Why or why not?

Ans: No we cant use TargetAmount as it is also a Target variable.

(e) Partition the data: set records 1, 3, 5, … (the rows with odd numbers) as the training data, and set records 2, 4, 6, … (the rows with even numbers) as the validation data, which results in 50%/50% partition for training/validation. Copy the code used below.
Partitioning data:

Storing odd and even rows in odd_row variable

```
R>          odd_row <- seq_len(nrow(org))%%2
```

Making a data set of odd rows from org

```
R>          odd_data <- org[odd_row==1,]
            View(odd_data)
```

Making a data set of even rows from org

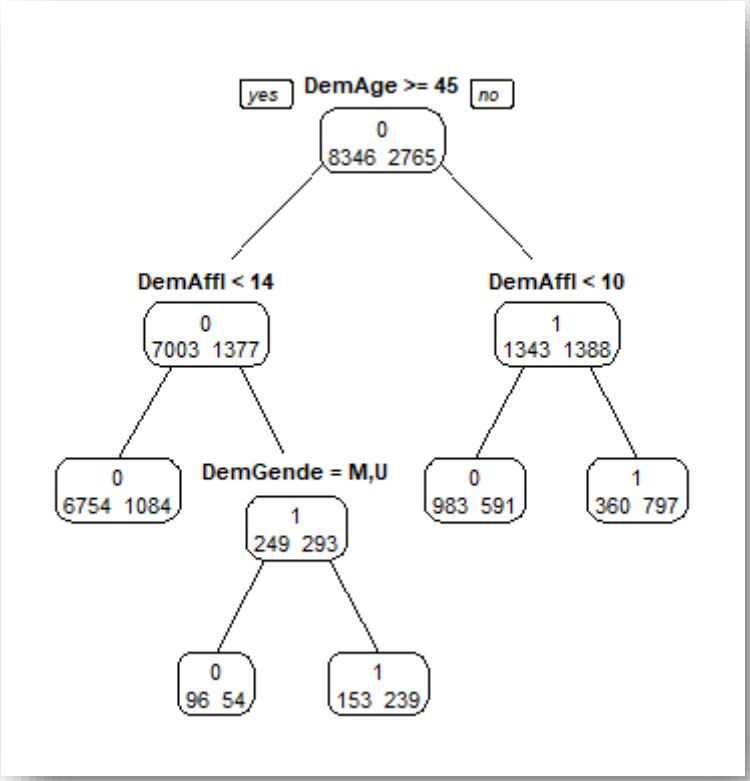
```
R>          even_data <- org[odd_row==0,]
            View(even_data)
```

(f) Implement a decision tree on the Training data to predict "TargetBuy" status. Plot the tree. Copy the code used and the result below. How many leaves are in the tree? Which variable was used for the first split? Create a confusion matrix which shows the accuracy rate of your classification. Copy the code used and the result below.

Removing TargetAmonut and Id column and creating tree on validation data
using rpart() to run a classification tree
using prp() in rpart.plot to plot tree

```
R>          library(rpart)
            library(rpart.plot)

            train.df <- odd_data[, -c(1,12)]
            valid.df <- even_data[, -c(1,12)]
```

```
R>          tree <- rpart(TargetBuy~., data = train.df, method = "class")
            prp(tree, type = 1, extra=1)
```




```
R>          default.t <- rpart(formula =TargetBuy ~ ., data=train.df, method ='class' )
            prp(default.t ,type =1 ,extra=1)
            default.train <- predict(default.t, train.df,type="class")
            confusionMatrix(default.train ,as.factor(train.df$TargetBuy))
```

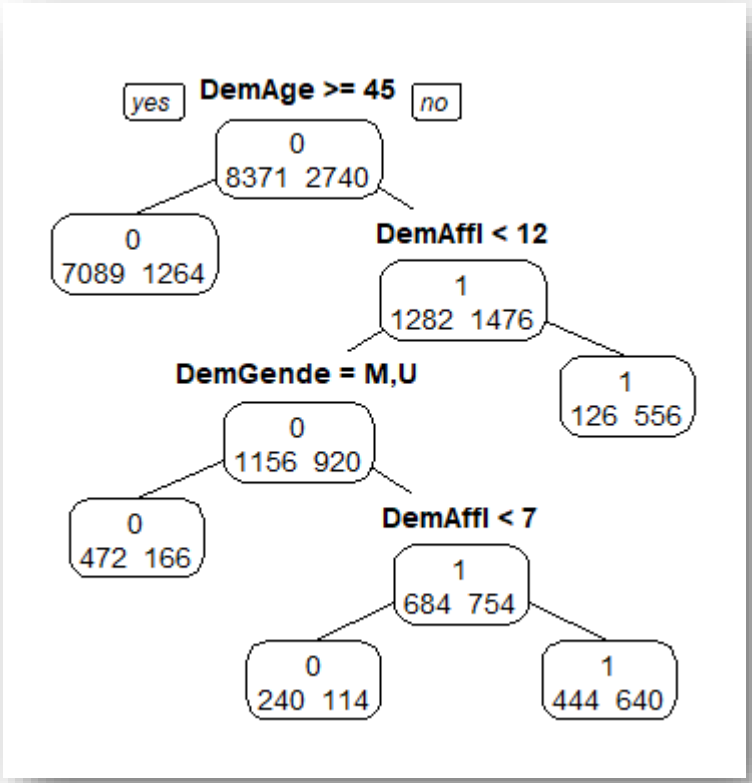
Confusion Matrix and Statistics

```
          Reference
Prediction    0    1
         0 7833 1729
         1  513 1036

 Accuracy : 0.798
```

(g) Apply your decision tree from the training data to the validation data, and compare the accuracy of classification of your validation and training data sets. Show the confusion matrix. Copy the code used and the results below. How is the accuracy using validation data different from that using training data? Is this what you expected? Why?

```
default.v <- rpart(formula =TargetBuy ~ ., data=valid.df, method ='class' )
```

```
prp(default.v ,type =1 ,extra=1)
default.valid <- predict(default.v, valid.df,type="class")
confusionMatrix(default.valid ,as.factor(valid.df$TargetBuy))
```



Confusion Matrix and Statistics

Reference
Prediction 0 1
0 7801 1544
1 570 1196

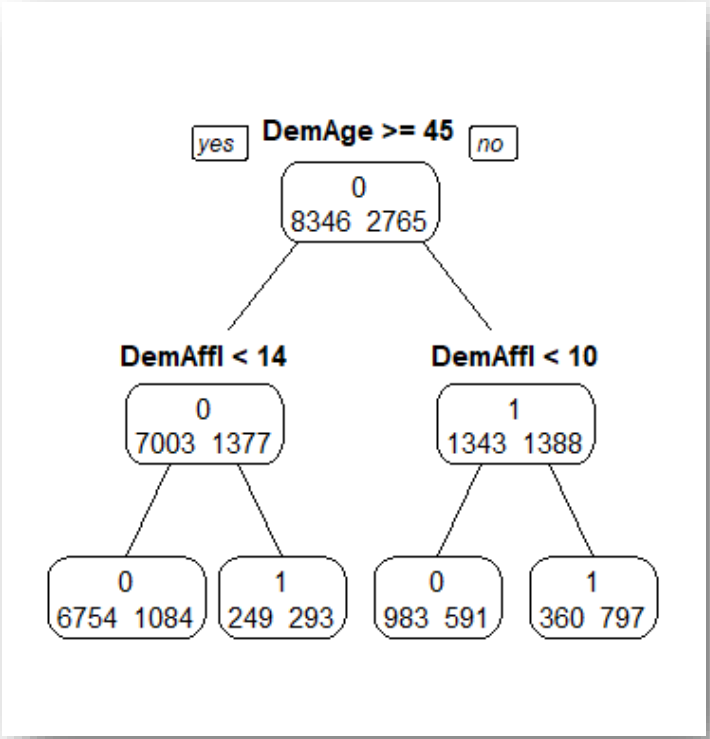
Accuracy : 0.81

Accuraccy is more for training data. I was expecting it near to the accuracy of training data because it the subset of same data.

(h) Imposing maxdepth = 2, create another decision tree on the training data to predict TargetBuy status. Plot the tree. Create a confusion matrix which shows the accuracy rate of your classification.Copy the code used and the result below. How many leaves are in the tree? Compared with the tree in (f), which one appears to be better? Is this what you expected? Why?

Creating new tree imposing maxdepth = 2

```
R>          newtree <- rpart(TargetBuy ~ ., data = train.df, control =
            rpart.control(maxdepth = 2), method = "class")
            prp(newtree, type =1, extra=1)
```




```
R>          default.v <- rpart(formula =TargetBuy ~ ., data=valid.df, method ='class' )
            prp(default.v ,type =1 ,extra=1)
            default.valid <- predict(default.v, valid.df,type="class")
            confusionMatrix(default.valid ,as.factor(valid.df$TargetBuy))
```

Confusion Matrix and Statistics

Reference
Prediction 0 1
0 7833 1729
1 513 1036

Accuracy : 0.79
Confusion Matrix

No of leaves for normal = 5
No. of leaves for max dept 2 = 4
The data with max depth appear better because it has same accuracy and simple.

(i) Next, consider using a logistic regression model. First, are there any missing values? If so, is any missing values imputation needed for logit model? Is imputation required before generating the decision tree models and why?

Ans: Yes imputation required before logit model as it cannot handle the null values.

```
R>
> #(i)Finding missing values
> sum(is.na(org$DemAge))
[1] 1508
> sum(is.na(org$DemAffl))
[1] 1085
> sum(is.na(org$DemClusterGroup))
[1] 674
> sum(is.na(org$DemGender))
[1] 2511
> sum(is.na(org$DemReg))
[1] 465
> sum(is.na(org$DemTVReg))
[1] 465
> sum(is.na(org$PromClass))
[1] 0
> sum(is.na(org$PromSpend))
[1] 0
> sum(is.na(org$PromTime))
[1] 281
> sum(is.na(org$TargetBuy))
[1] 0
```

(j) Impute: impute "U" for unknown class variable values and the overall mean for unknown interval variable values. Copy the code used below.

```
R>      org$DemAffl[is.na(org$DemAffl) == TRUE] <- "U"
        org$DemAge[is.na(org$DemAge) == TRUE] <- mean(organics$DemAge,
        na.rm = TRUE)
        org$DemClusterGroup[is.na(org$DemClusterGroup) == TRUE] <- "U"
        org$DemGender[is.na(org$DemGender) == TRUE] <- "U"
        org$DemReg[is.na(org$DemReg) == TRUE] <- "U"
        org$DemTVReg[is.na(org$DemTVReg) == TRUE] <- "U"
        org$PromClass[is.na(org$PromClass) == TRUE] <- "U"
        org$PromSpend[is.na(org$PromSpend) == TRUE] <-
        mean(organics$DemAge, na.rm = TRUE)
        org$PromTime[is.na(org$PromTime) == TRUE] <- "U"
        org$TargetBuy[is.na(org$TargetBuy) == TRUE] <-
        mean(organics$DemAge, na.rm = TRUE)
```

Post running this code all null values are zero-
CHK-

```
R>          sum(is.na(org$DemAge))
            [1] 0
            > sum(is.na(org$DemAffl))
            [1] 0
            > sum(is.na(org$DemClusterGroup))
            [1] 0
            > sum(is.na(org$DemGender))
            [1] 0
            > sum(is.na(org$DemReg))
            [1] 0
            > sum(is.na(org$DemTVReg))
            [1] 0
            > sum(is.na(org$PromClass))
            [1] 0
            > sum(is.na(org$PromSpend))
            [1] 0
            > sum(is.na(org$PromTime))
            [1] 0
            > sum(is.na(org$TargetBuy))
            [1] 0
```

(k) Use a logistic regression model to classify the data set using the same dependent variable, TargetBuy. Copy the code used and the result below.

```
R>          View(org)
            #Preparing data for logit midel with new data set without null values

            #storing odd and even rows in odd_row variable
            odd_row2 <- seq_len(nrow(org))%%2

            #Making a data set of odd rows from org
            odd_data2 <- org[odd_row==1,]
            View(odd_data2)

            train2.df <- odd_data2
            valid2.df <- even_data2

            #Making a data set of even rows from org
            even_data2 <- org[odd_row==0,]
            View(even_data2)

            #Logit Regression
            logit.reg <- glm(TargetBuy ~ ., data = train2.df, family = "binomial")
            summary(logit.reg)


Deviance Residuals:
   Min     1Q  Median     3Q    Max
-2.309  -0.695  -0.438  -0.001   3.258
```

Coefficients: (5 not defined because of singularities)

	Estimate	Std. Error	z value	Pr(>|z|)	
(Intercept)	-11.90298650	759.45636064	-0.02	0.987	
DemAffl1	0.09323974	827.73205196	0.00	1.000	
DemAffl10	14.16535092	759.45610780	0.02	0.985	
DemAffl11	14.40066696	759.45610844	0.02	0.985	
DemAffl12	14.54816441	759.45611004	0.02	0.985	
DemAffl13	14.74240451	759.45611240	0.02	0.985	
DemAffl14	15.30716776	759.45611477	0.02	0.984	
DemAffl15	15.53306135	759.45611961	0.02	0.984	
DemAffl16	15.87713690	759.45613648	0.02	0.983	
DemAffl17	16.15643740	759.45616073	0.02	0.983	
DemAffl18	16.55454384	759.45619407	0.02	0.983	
DemAffl19	16.96227773	759.45637552	0.02	0.982	
DemAffl2	12.62127950	759.45619738	0.02	0.987	
DemAffl20	18.28852422	759.45681079	0.02	0.981	
DemAffl21	30.56162831	811.30063706	0.04	0.970	
DemAffl22	30.41455861	928.70482521	0.03	0.974	
DemAffl23	29.73849993	881.55094422	0.03	0.973	
DemAffl24	30.26449180	903.75347568	0.03	0.973	
DemAffl25	29.64699608	957.88409463	0.03	0.975	
DemAffl26	31.99477712	1641.63198627	0.02	0.984	
DemAffl27	29.84602236	1273.38138900	0.02	0.981	
DemAffl29	29.09540483	1641.63198426	0.02	0.986	
DemAffl3	12.96843946	759.45613300	0.02	0.986	
DemAffl31	29.73287830	1641.63198450	0.02	0.986	
DemAffl4	12.79836996	759.45612257	0.02	0.987	
DemAffl5	12.95338140	759.45611403	0.02	0.986	
DemAffl6	13.31931189	759.45610954	0.02	0.986	
DemAffl7	13.43271561	759.45610890	0.02	0.986	
DemAffl8	13.55409302	759.45610815	0.02	0.986	
DemAffl9	13.92611403	759.45610769	0.02	0.985	
DemAfflU	14.13384785	759.45611148	0.02	0.985	
DemAge	-0.04893934	0.00229699	-21.31	<0.0000000000000002	***
DemClusterGroupB	-0.06611714	0.11119350	-0.59	0.552	
DemClusterGroupC	-0.05156257	0.10950242	-0.47	0.638	
DemClusterGroupD	-0.08212441	0.11044564	-0.74	0.457	
DemClusterGroupE	-0.01156731	0.11948956	-0.10	0.923	
DemClusterGroupF	-0.01140689	0.11229480	-0.10	0.919	
DemClusterGroupU	0.08010338	0.18783071	0.43	0.670	
DemGenderM	-0.91093042	0.06228728	-14.62	<0.0000000000000002	***
DemGenderU	-1.68306124	0.08827860	-19.07	<0.0000000000000002	***
DemRegNorth	-0.27543160	0.13664625	-2.02	0.044	*
DemRegScottish	0.04288485	0.27793558	0.15	0.877	
DemRegSouth East	-0.16577072	0.11777383	-1.41	0.159	
DemRegSouth West	0.15199996	0.16638671	0.91	0.361	
DemRegU	-0.19979119	0.19688802	-1.01	0.310	
DemTVRegC Scotland	-0.03214759	0.29406717	-0.11	0.913	

DemTVRegEast	-0.02638003	0.12873219	-0.20	0.838
DemTVRegLondon	0.10886080	0.09208337	1.18	0.237
DemTVRegMidlands	-0.04280109	0.11024546	-0.39	0.698
DemTVRegN East	0.23375386	0.16947406	1.38	0.168
DemTVRegN Scot	-0.11410224	0.34014216	-0.34	0.737
DemTVRegN West	0.14937277	0.13350036	1.12	0.263
DemTVRegS & S East	NA	NA	NA	NA
DemTVRegS West	NA	NA	NA	NA
DemTVRegU	NA	NA	NA	NA
DemTVRegUlster	-0.25059741	0.28996241	-0.86	0.387
DemTVRegWales & West	NA	NA	NA	NA
DemTVRegYorkshire	NA	NA	NA	NA
PromClassPlatinum	-0.27551371	0.21278965	-1.29	0.195
PromClassSilver	0.02264297	0.07992031	0.28	0.777
PromClassTin	0.06978131	0.09619431	0.73	0.468
PromSpend	-0.00000148	0.00000624	-0.24	0.813
PromTime1	-0.21212977	0.60402447	-0.35	0.725
PromTime10	-0.02719032	0.61669923	-0.04	0.965
PromTime11	-0.36941885	0.61679214	-0.60	0.549
PromTime12	-0.06734126	0.62842605	-0.11	0.915
PromTime13	0.13850039	0.65740138	0.21	0.833
PromTime14	0.11285809	0.69043261	0.16	0.870
PromTime15	-0.07573284	0.70309050	-0.11	0.914
PromTime16	-0.04665324	0.69091181	-0.07	0.946
PromTime17	-0.04899519	0.74762141	-0.07	0.948
PromTime18	0.03944536	0.74105475	0.05	0.958
PromTime19	0.21897125	0.75655859	0.29	0.772
PromTime2	-0.07871666	0.60535203	-0.13	0.897
PromTime20	0.12942350	0.82852386	0.16	0.876
PromTime21	-0.27815245	0.77843604	-0.36	0.721
PromTime22	0.11242256	1.00287231	0.11	0.911
PromTime23	0.87670284	0.86472920	1.01	0.311
PromTime24	0.00870642	0.98918857	0.01	0.993
PromTime25	-0.03883673	0.97722092	-0.04	0.968
PromTime26	-0.97669256	1.00428729	-0.97	0.331
PromTime27	-0.27394959	0.91117346	-0.30	0.764
PromTime28	0.85241930	0.90332823	0.94	0.345
PromTime29	0.81581377	0.78544132	1.04	0.299
PromTime3	-0.03044267	0.60761938	-0.05	0.960
PromTime30	-0.77954365	1.20514683	-0.65	0.518
PromTime31	-0.08878615	0.88283663	-0.10	0.920
PromTime32	-0.88136391	1.24381219	-0.71	0.479
PromTime33	-13.56825266	780.18697668	-0.02	0.986
PromTime35	0.54409456	1.37042372	0.40	0.691
PromTime36	-13.43214078	909.56892277	-0.01	0.988
PromTime4	-0.15399065	0.60193839	-0.26	0.798
PromTime5	-0.12913025	0.60037529	-0.22	0.830
PromTime6	0.01719713	0.60707365	0.03	0.977
PromTime7	-0.01558586	0.60523021	-0.03	0.979

```
PromTime8          -0.27512454   0.60129652  -0.46          0.647
PromTime9          -0.16307289   0.60400662  -0.27          0.787
PromTimeU           0.07800131   0.63414829   0.12          0.902
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 12468.1  on 11110  degrees of freedom
Residual deviance:  9788.7  on 11018  degrees of freedom
AIC: 9975

Number of Fisher Scoring iterations: 14
```

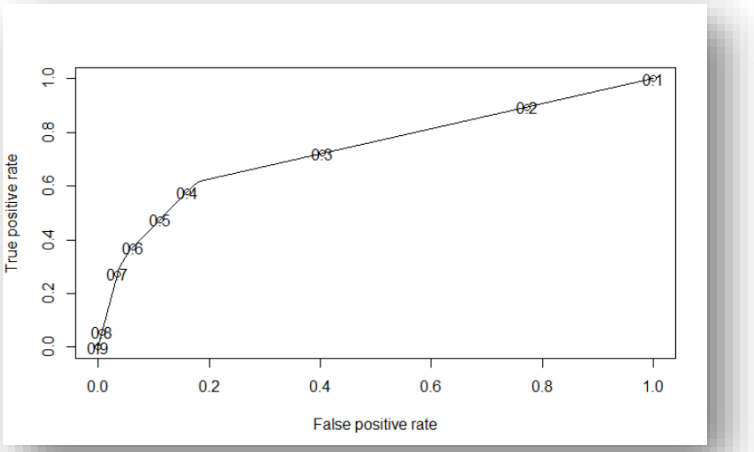
(l) Compare the performance of the logit model on the training and validation data sets by creating confusion matrixes which show the accuracy rates. Copy the code used and the result below. Which one appears to be better?

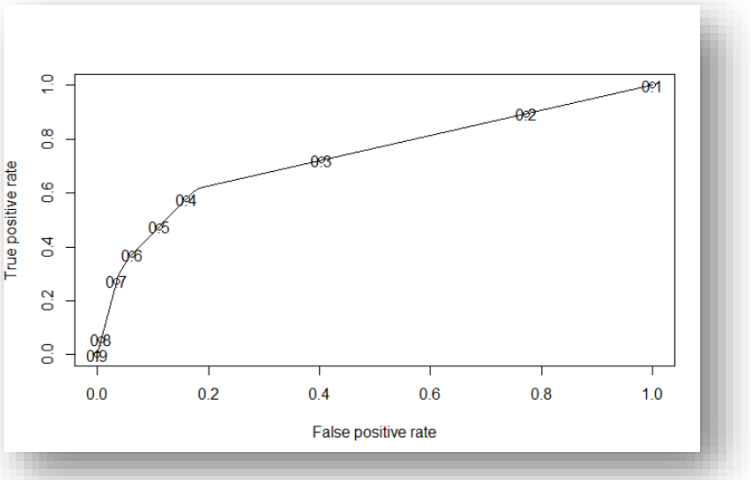
```
R>              logit.reg2 <- glm(TargetBuy ~ ., data = valid2.df, family = "binomial")
                summary(logit.reg2)

Call:
glm(formula = TargetBuy ~ ., family = "binomial", data = valid.df)

Deviance Residuals:
   Min     1Q  Median     3Q    Max
-2.438  -0.688  -0.406   0.438   3.024

Coefficients: (4 not defined because of singularities)
                  Estimate  Std. Error z value          Pr(>|z|)
(Intercept)       -0.12498158  0.24681940   -0.51           0.613
DemAffl           0.25859061  0.00967340   26.73 <0.0000000000000002 ***
DemAge            -0.05821341  0.00263254  -22.11 <0.0000000000000002 ***
DemClusterGroupB     0.08974614  0.13088945    0.69           0.493
DemClusterGroupC     0.19141234  0.12707853    1.51           0.132
DemClusterGroupD     0.18153211  0.12860522    1.41           0.158
DemClusterGroupE     0.18956744  0.14007854    1.35           0.176
DemClusterGroupF     0.19993147  0.13130429    1.52           0.128
DemClusterGroupU     0.53955030  0.53193755    1.01           0.310
DemGenderM         -1.05648563  0.07096328  -14.89 <0.0000000000000002 ***
DemGenderU         -2.13619480  0.17565848  -12.16 <0.0000000000000002 ***
DemRegNorth        -0.15265384  0.15622584   -0.98           0.329
DemRegScottish      -0.18661004  0.33218274   -0.56           0.574
DemRegSouth East    -0.01740141  0.14051795   -0.12           0.901
DemRegSouth West    -0.10579791  0.20591413   -0.51           0.607
DemTVRegC Scotland    0.11677158  0.34708258    0.34           0.737
DemTVRegEast        0.03927441  0.14794648    0.27           0.791
DemTVRegLondon      -0.02034694  0.10540248   -0.19           0.847
```

```
DemTVRegMidlands    -0.04508349  0.13214364   -0.34          0.733
DemTVRegN East       0.37566799  0.19097545    1.97          0.049 *
DemTVRegN Scot       0.07793493  0.39469856    0.20          0.843
DemTVRegN West      -0.11894427  0.14950224   -0.80          0.426
DemTVRegS & S East          NA          NA      NA            NA
DemTVRegS West              NA          NA      NA            NA
DemTVRegWales & West        NA          NA      NA            NA
DemTVRegYorkshire           NA          NA      NA            NA
PromClassPlatinum   -0.08849400  0.26560716   -0.33          0.739
PromClassSilver     -0.11710877  0.10045280   -1.17          0.244
PromClassTin        -0.06211581  0.11518335   -0.54          0.590
PromSpend            0.00000174  0.00000938    0.19          0.853
PromTime             0.00513252  0.00717680    0.72          0.475
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 9485.3  on 8223  degrees of freedom
Residual deviance: 7153.8  on 8197  degrees of freedom
  (2887 observations deleted due to missingness)
AIC: 7208

Number of Fisher Scoring iterations: 5
```

(m) Plot ROC curves for the decision tree in (f) and the logit model using validation data. Summarize each curve by its ROC index ("area under the curve (AUC)"). Copy the code used and the result below. In terms of ROC index, which model is better?

```
R>          prob=predict(logit.reg,train2.df,type=c("response"))
            install.packages("ROCR")
            library(ROCR)
            install.packages("prediction")
            library(prediction)
            train2.df$TargetBuy <- as.factor(train2.df$TargetBuy)

            pred <- prediction(prob, train2.df$TargetBuy)
            perf <- performance(pred, measure = "tpr", x.measure = "fpr")
            plot(perf, col=rainbow(7), main="ROC curve Admissions",
            xlab="Specificity",
              ylab="Sensitivity")
            abline(0, 1)
```





Logistic ROC is Better because false positive rate is less for Logistic ROC.

Market Basket Analysis

1. In Fall 2018, UTD opened a new buffet where there are many food selections for faculty and students. For simplicity, suppose five types of foods are offered daily: hamburger, pasta, salad, soup, and taco. Suppose you are the manager and you decide to use associate rules (manually) to figure out what foods customers tend to purchase together. You recorded selections by five customers as shown in the table below. You also decide to use the following cut-offs: minimum support 40% and minimum confidence 80%. What valid rules will you generate? Provide detailed steps with your relevant calculations. Also report support, confidence, and lift for the final rules you generate. (2.5 Points)

Ans:

Calculating support of individual item and accepting it for further consideration if support > 40% (Threshold)

Items	No. of Trans.	Support	Acc/Rej
{Hamburger}	3	60%	Accepted
{Pasta}	2	40%	Accepted
{Salad}	2	40%	Accepted
{Soup}	4	80%	Accepted
{Taco}	4	80%	Accepted

Calculating support of all possible two items and rejecting if it does not qualify the minimum support value.

Items	No. of Trans.	Support	Acc/Rej
{Hamburger} {Pasta}	1	20%	Rejected
{Hamburger} {Salad}	2	40%	Accepted
{Hamburger} {Soup}	2	40%	Accepted
{Hamburger} {Taco}	2	40%	Accepted
{Pasta} {Salad}	1	20%	Rejected
{Pasta} {Soup}	1	20%	Rejected
{Pasta} {Taco}	1	20%	Rejected
{Salad} {Soup}	1	20%	Rejected
{Salad} {Taco}	1	20%	Rejected
{Soup} {Taco}	4	80%	Accepted

Calculating support for three items together where support of two items was greater than 40%.

Items	No. of Trans	Support	Acc/Rej
{hamburger, soup, taco}	2	40%	Accepted

Calculating confidence for all those two and three items have support more than the threshold and rejecting if confidence < 80%

LHS		RHS	Confidence	Acc/Rej
{Hamburger}	=>	{Salad}	67%	Rejected
{Hamburger}	=>	{Soup}	67%	Rejected
{Hamburger}	=>	{Taco}	67%	Rejected
{Soup}	=>	{Taco}	100%	Accepted
{Salad}	=>	{Hamburger}	100%	Accepted
{Soup}	=>	{Hamburger}	50%	Rejected
{Taco}	=>	{Hamburger}	50%	Rejected
{Taco}	=>	{Soup}	100%	Accepted
{Hamburger, Soup}	=>	{Taco}	100%	Accepted
{Hamburger, Taco}	=>	{Soup}	100%	Accepted
{Taco, Soup}	=>	{Hamburger}	50%	Rejected

Final Rules:
1. {Salad} => {Hamburger}
2. {Soup} => {Taco}
3. {Taco} => {Soup}
4. {Hamburger, soup} => {taco}
5. {Hamburger, taco} => {Soup}

Support, Confidence and lift for the final rules:

	lhs		rhs	support	conf.	lift
[1]	{salad}	=>	{hamburger}	40%	100%	1.67
[2]	{soup}	=>	{taco}	80%	100%	1.25
[3]	{taco}	=>	{soup}	80%	100%	1.25
[4]	{hamburger, soup}	=>	{taco}	40%	100%	1.25
[5]	{hamburger, taco}	=>	{soup}	40%	100%	1.25

2.

(a) Suppose that the association rule "hamburgers => hot dogs" is mined. Given a minimum support threshold of 25% and a minimum confidence threshold of 50%, is this association rule valid? (0.5 Point)

Iitem	Support	Confidence	Lift
{Hamburger} => {Hot dog}	36%	60%	1.67

Support and confidence is more than minimum threshold. Moreover, lift is greater than 1, hence, this association rule is valid.

(b) Based on the given data, is the purchase of hamburgers independent of the purchase of hot dogs? If not, what kind of correlation relationship exists between the two (i.e., if a customer purchases hamburgers, will that increase or decrease her chance of purchasing hot dogs)? (1 Point)

P({Hamburger}/{Hotdog)	**72%**
P({Hamburger})	60%

Since both the probabilities are not equal hence purchase is dependent on each other.
Since Lift = 1.2
Means there is positive correlation exist between these two and if someone is buying a hot dog then chance that he will buy hamburger will increase.

3. Conducting an Association Analysis Using R: A store is interested in determining the associations between items purchased from the Health and Beauty Aids department and the Stationery Department. The store chose to conduct a market basket analysis of specific items purchased from these two departments. "transactions" contains information about over 400,000 transactions made over the past three months. The following 17 products are represented in the data set: bar soap, bows, candy bars, deodorant, greeting cards, magazines, markers, pain relievers, pencils, pens, perfume, photo processing, prescription medications, shampoo, toothbrushes, toothpaste, and wrapping paper. (4Points)

(a) Import the data to R. Copy the R code used below. (Tip: use read.transactions)

Ans:

```
# (a) Importing 'transactions' data set to R in the form of transitional data set
R>      library(arules)
        tran.data <- read.transactions("transactions.csv", format = "single", header =
        TRUE, sep =",", cols =c("Customer","Product"), rm.duplicates = TRUE)

# Inspecting 10 rows
R>      inspect(head(tran.data, n=10))
```

(b) Set Support to 0.01, Confidence to 0.10, and Min Length to 2. Run apriori to obtain the rules. Sort the rules according to "Lift" with descending order. Copy the R code used below.

Ans:

```
# (b) Running apriori(), include the minimum support 0.01, minimum confidence 0.10
R>      rules <- apriori(tran.data, parameter = list(supp = 0.01, conf = 0.10))

# sort rules by "lift"
R>      rules <- sort(rules, by = "lift", decreasing = TRUE)
```

(c) Show the top ten Association Rules. Copy the code used and the result below

Ans:

```
#(c) inspect the first 10 rules
R>      inspect(head(rules, n =10))
```

	lhs		rhs	support	conf.	coverage	lift	count
[1]	{Perfume}	=>	{Toothbrush}	0.022	0.243	0.090	3.601	4364
[2]	{Toothbrush}	=>	{Perfume}	0.022	0.324	0.067	3.601	4364
[3]	{Bow}	=>	{Toothbrush}	0.011	0.208	0.055	3.081	2268
[4]	{Toothbrush}	=>	{Bow}	0.011	0.168	0.067	3.081	2268
[5]	{Candy Bar, Magazine}	=>	{Greeting Cards}	0.017	0.411	0.041	2.799	3333
[6]	{Pencils, Toothpaste}	=>	{Candy Bar}	0.011	0.464	0.025	2.712	2278
[7]	{Greeting, Cards, Magazine}	=>	{Candy Bar}	0.017	0.459	0.036	2.682	3333
[8]	{Magazine, Toothpaste}	=>	{Greeting Cards}	0.012	0.377	0.032	2.568	2389
[9]	{Magazine, Toothpaste}	=>	{Candy Bar}	0.014	0.433	0.032	2.534	2744
[10]	{Greeting Cards, Toothpaste}	=>	{Candy Bar}	0.013	0.411	0.032	2.402	2635

(d) What is the highest lift value for the resulting rules? Which rule has this value? Show how this lift value was calculated.

Ans:

Highest lift is 3.60137

Rules with highest lift values are-

	lhs		rhs	lift
[1]	{Perfume}	=>	{Toothbrush}	3.60137
[2]	{Toothbrush}	=>	{Perfume}	3.60137

Lift calculation:
Lift of {Perfume} => {Toothbrush} = CONF({Perfume} => {Toothbrush}) / SUP({Toothbrush}
= 0.243/0.0675
= 3.6

Lift of {Toothbrush} => {Perfume} = CONF({Toothbrush} => {Perfume})/ SUP{Perfume}
= 0.324/0.08998
= 3.6

(e) Interpret the first five rules in the output in words.

[1]	{Perfume}	=>	{Toothbrush}
[2]	{Toothbrush}	=>	{Perfume}
[3]	{Bow}	=>	{Toothbrush}
[4]	{Toothbrush}	=>	{Bow}
[5]	{Candy Bar, Magazine}	=>	{Greeting Card}

1. Purchase of Perfume then Toothbrush is dependent and people who are buying Perfumes are 3.6 times more likely to buy Toothbrush than buying Toothpaste alone.

2. Purchase of Toothbrush then Perfume is dependent in each other and People who are buying Toothbrush are 3.6 times more likely to buy Perfumes than buying Perfumes alone.

3. Purchase of Bow and Toothbrush is dependent on each other and people who are buying Bow are 3.08 times more likely to buy Toothbrush than buying Toothbrush alone.

4. Purchase of Toothbrush and Bow is dependent on each other and people who are buying Toothbrush are 3.08 times more likely to buy Bow than buying Bow alone.

5. Purchase of greeting card with Candy Bar and Magazine is dependent on each other and people who are buying Candy Bar and Magazine are 2.8 times more likely to buy Greeting Card than buying Greeting cards alone.

(f) Reviewing the top 10 rules, based on their lift ratios, comment on their redundancy and how you would assess their utility as a decision maker.

	lhs		rhs	support	conf.	lift
[1]	{Perfume}	=>	{Toothbrush}	0.022	0.243	3.601
[2]	{Toothbrush}	=>	{Perfume}	0.022	0.324	3.601
[3]	{Bow}	=>	{Toothbrush}	0.011	0.208	3.081
[4]	{Toothbrush}	=>	{Bow}	0.011	0.168	3.081
[5]	{Candy Bar, Magazine}	=>	{Greeting Cards}	0.017	0.411	2.799

[6]	{Pencils, Toothpaste}	=>	{Candy Bar}	0.011	0.464	2.712
[7]	{Greeting, Cards, Magazine}	=>	{Candy Bar}	0.017	0.459	2.682
[8]	{Magazine, Toothpaste}	=>	{Greeting Cards}	0.012	0.377	2.568
[9]	{Magazine, Toothpaste}	=>	{Candy Bar}	0.014	0.433	2.534
[10]	{Greeting Cards, Toothpaste}	=>	{Candy Bar}	0.013	0.411	2.402

While making a decision for association rules, as the support and lift being the same, confidence is a deciding attribute to choose between two rules for similar set of products.

Rule no. 1 is a redundant rule when comparing to rule no.2 as the confidence of rule 2 is more than rule 1.

Rule no 4 is a redundant rule as compare to rule no 3 because confidence of rule 3 is more than the confidence of rule 4.

Rule no 5 is a redundant rule as compare to rule no rule no. 7 because the confidence of rule 7 is higher.

1. Clustering Stores: The DUNGAREE data set shows the number of pairs of four different types of dungarees sold at stores over a specific time period. Each row represents an individual store. There are six columns in the data set. One column is the store identification number, and the remaining columns contain the number of pairs of each type of jeans sold.
Use R to run k-mean clustering (based on the code shown in class):

(a) Import the data to R and remove the column(s) that you are not going to use. Copy the R code used below.

Ans:

```
#Importing data set to R
R>    dungaree <- read.csv("dungaree.csv")
      View(dungaree)

#removing STOREID and SALESTOT
R>    dungaree <- dungaree[,-c(1,6)]
      View(dungaree)
```

(b) Examine the input variables: Are there any unusual data values? Are there missing values that should be replaced?

Ans:

```
#checking for missing value
R>    sum(is.na(dungaree))
```

Since sum(is.na(dungaree)) is 0, there are no missing values in the data set hence we do not need to replace anything. However, some columns have bigger values (Leisure and Original) while some (Fashion and stretch) have relatively smaller values hence we need to normalize the data.

(c) Normalize the data. Copy the R code used below. What would happen if you did not standardize/normalize your inputs?

Ans:

```
#Normalizing the data
R>      dungaree.norm <- sapply(dungaree, scale)
```

If we will not normalize the data then the columns with bigger values will have more impact on the final results and columns having comparatively lower values will not have significant influence on the final results. In the above data set "FASHION" column has data in two to three digits (eg. 182, 107, 117, 79 & 496, 296, 276) while in "ORIGIONAL" column has data in 4 digits (1528, 2247,1652 & 2203, 1890, 2342) hence normalization will required to have equal weightage of all the columns.

(d) Run k-means clustering using a seed = 42, and choose k = 20. Copy the R code used below.

Ans:

```
#setting the seed = 42 and K = 20 and running the K-means clustering
R>      set.seed(42)
        km <- kmeans(dungaree.norm, 20)
        km$cluster
```

(e) Based on the results, does k=20 clusters seem appropriate? Why or why not?

Ans:

Based on k=20 cluster does not seem appropriate. Setting value of k = 20 will results in total 20 numbers of clusters. Clustering algorithm works on the method of least sum of square within a cluster and idea is to minimise it. However, after certain k value there is not significant change in the within sum of square values but the no of clusters keep on increasing hence it becomes difficult to analyse the results. Moreover, higher number of cluster forms artificial boundary between real data cluster. Furthermore, for 20 clusters it will be very hard to draw conclusions and study each cluster one by one.

(f) In the next run, specify a maximum of six clusters, and run the k-means clustering algorithm again. Copy the R code used below.

Ans:

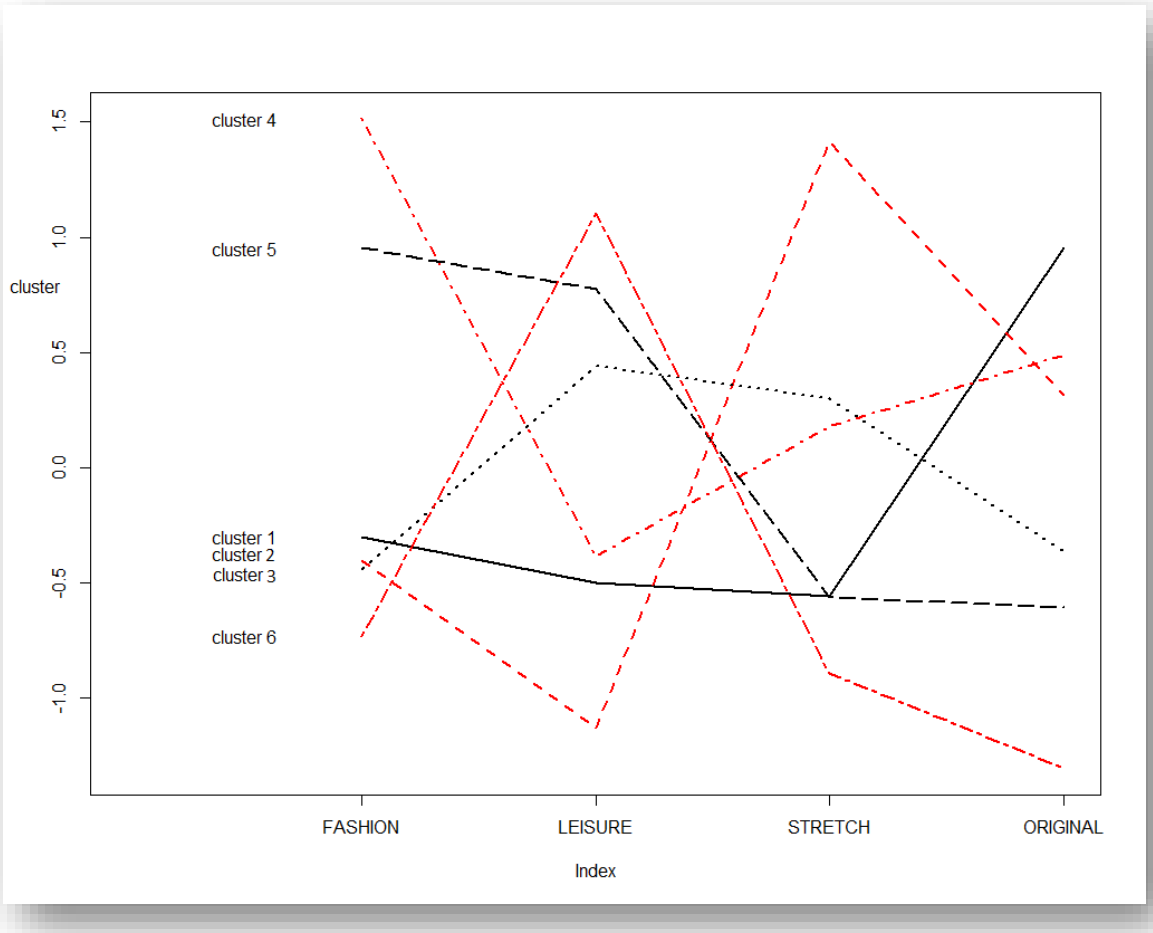
```
#Taking k = 6
R>      km <- kmeans(dungaree.norm, 6)
```

```
km$cluster
```

(g) Plot profile plot of centroids for the six clusters generated in (f). Copy the code used and the result below.

Ans:

```
R>  plot(c(0), xaxt = 'n', ylab = "", type = "l",
     ylim = c(min(km$centers), max(km$centers)), xlim = c(0, 4))

# label x-axes
R>  axis(1, at = c(1:4), labels = names(dungaree))

# plot centeriod
R>  for (i in c(1:6))
        lines(km$centers[i,], lty = i, lwd = 2, col = ifelse(i %in% c(1, 3, 5),
                "black", "red"))
#naming
R>  text(x = 0.5, y = km$centers[,1], labels = paste("cluster", c(1:6)))
```



(h) Using the profile plot of centroids, interpret the characteristics of each cluster as it relates to types of jeans sold at stores. Describe these clusters, and their similarities and differences in words.

Ans:

From the above profile plot we can conclude that cluster 1 has minimum variation of sale among all the clusters. Cluster 4 has highest sale for fashion but very low for leisure. Cluster 6 has highest sale for leisure but lowest for the stretch and original. Cluster 3 has low sale for leisure but very high for stretch.

Cluster 4 and 6 are farthest from each other and cluster 1 and 3 are closest to each other. Cluster 5 has high sale for fashion, leisure and original categories but low for the stretch. Moreover, spread of cluster for leisure is very high.

Clustering

2. Clustering Pharmaceutical Firms: An equities analyst is studying the pharmaceutical industry and would like your help in exploring and understanding the financial data collected by her firm. Her main objective is to understand the structure of the pha rmaceutical industry using some basic financial measures. Financial data gathered on 21 firms in the pharmaceutical industry are available in the file Pharmaceuticals.csv. For each firm, the following variables are recorded.

Use R to run hierarchical clustering (based on the code shown in class):

(a) Import the data to R, set row names to the "Symbol" column, and remove all the columns that you are not going to use for clustering. Copy the R code used below.

Ans:

```
#Importing data set to R
R>      pharma <- read.csv("Pharmaceuticals.csv")

#Setting the row names to "Symbol" columns
R>      row.names(pharma) <- pharma[,1]

#removing all the columns not useful for the clustering
R>      pharma.df <- pharma[,-c(1,2,12:14)]
        View(pharma.df)
```

(b) Normalize the data. Copy the R code used below.

Ans:

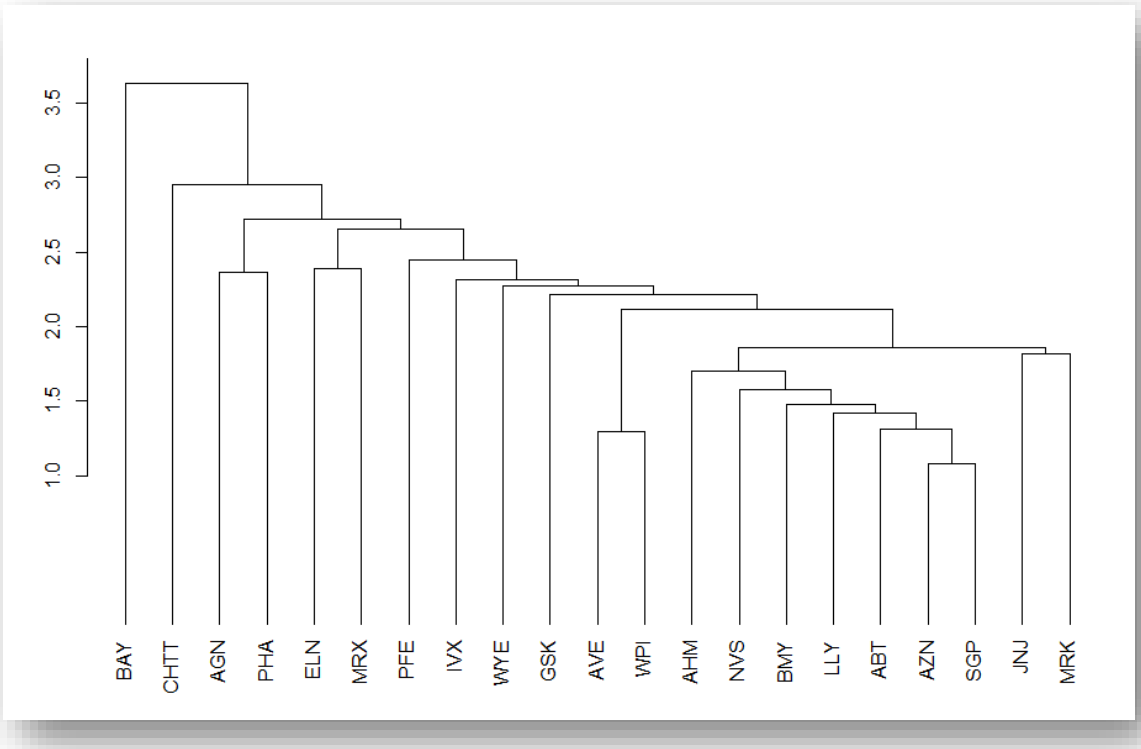
```
#Normalizing data
R>      pharma.norm <- sapply(pharma.df, scale)
```

```
#Setting row names to pharma column to the normalised data set
R>      row.names(pharma.norm) <- pharma[,1]
```

(c) Based on single linkage, run hierarchical clustering to generate Dendrogram. Copy the code used and the result below.

Ans:

```
#Calculating euclidean distance for hierarchiacal cludtering
R>      d.norm <- dist(pharma.norm, method = "euclidean");d.norm

#Single linkage dendogram
R>      hc <- hclust(d.norm, method = "single")
        plot(hc, hang = -1, ann = F)
```



(d) If we are interested in 6 clusters based on Dendrogram in (c), what are the members of each cluster? Copy the code used and the result below.

Ans:

```
#Finding 6 dendogram from the above single linkage
R>      cutree(hc, k=6)
```

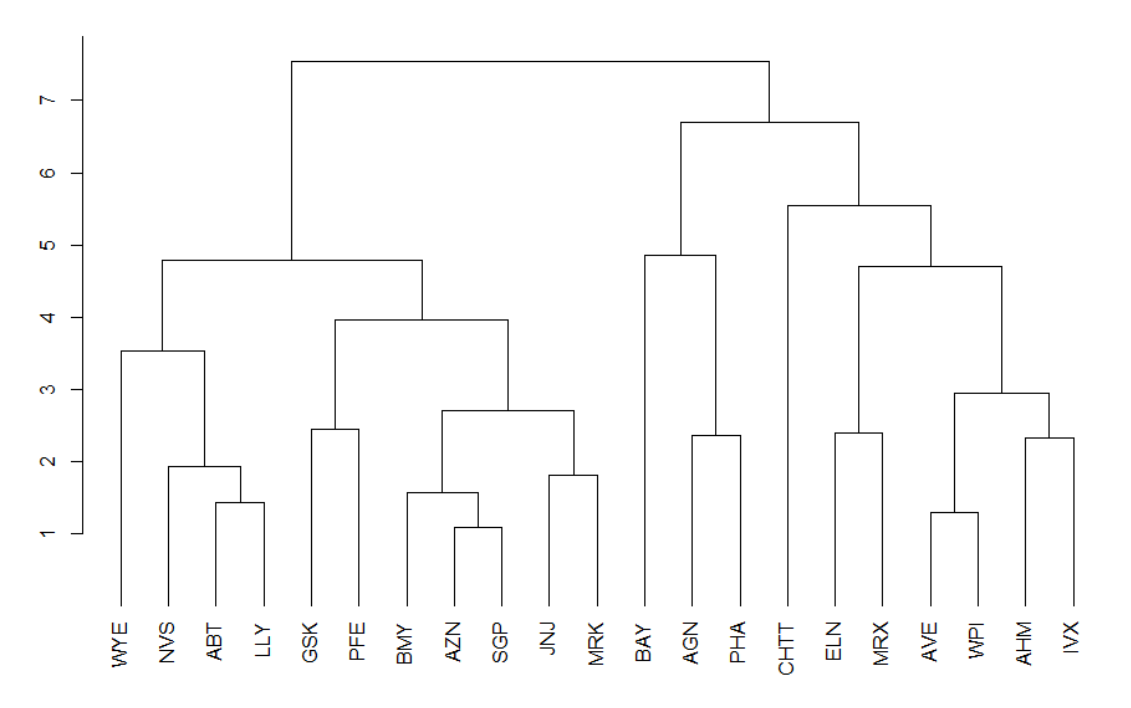
```
> #Single linkage dendogram
> hc <- hclust(d.norm, method = "single")
> plot(hc, hang = -1, ann = F)
> #Finding 6 dendogram from the above single linkage
> cutree(hc, k=6)
 ABT  AGN  AHM  AZN  AVE  BAY  BMY CHTT  ELN  LLY  GSK  IVX  JNJ  MRX  MRK  NVS  PFE  PHA  SGP  WPI  WYE
   1    2    1    1    1    3    1    4    5    1    1    1    1    5    1    1    6    2    1    1    1
>
```

(e) Based on complete linkage, run hierarchical clustering to generate Dendrogram. Copy the code used and the result below.

Ans:

#Complete linkage dendogram

```
R>      hc2 <- hclust(d.norm, method = "complete")
        plot(hc2, hang = -1, ann = F)
```



(f) If we are interested in 6 clusters based on Dendrogram in (e), what are the members of each cluster? Copy the code used and the result below.

Ans:

#Finding 6 dendogram from the double likage

```
R>      clust <- cutree(hc2,k = 6);clust
```

```
> #Complete linkage dendogram
> hc2 <- hclust(d.norm, method = "complete")
> plot(hc2, hang = -1, ann = F)
> #Finding 6 dendogram from the double likage
> clust <- cutree(hc2,k = 6);clust
 ABT  AGN  AHM  AZN  AVE  BAY  BMY CHTT  ELN  LLY  GSK  IVX  JNJ  MRX  MRK  NVS  PFE  PHA  SGP  WPI  WYE
   1    2    3    4    3    5    4    6    3    1    4    3    4    3    4    1    4    2    4    3    1
>
```

(g) Do (d) and (f) lead to the same six clusters? Explain why.

Ans:

No, (d) and (f) do not lead to the same cluster. (d) is derived by cutting single linkage dendogram where clustering is done as per the minimum distance between two cluster and cluster distance is defined as the minimum distance between two elements (one from each cluster). However, (f) is obtained by cutting the complete linkage dendogram at k=6 where clustering is done as per the minimum distance between two cluster where cluster distance is defined as the farthest distance between two elements (one from each cluster). The clustering arrangement is different for single and complete linkage hence cutting it at k=6 will give different clusters for single and complete linkage.