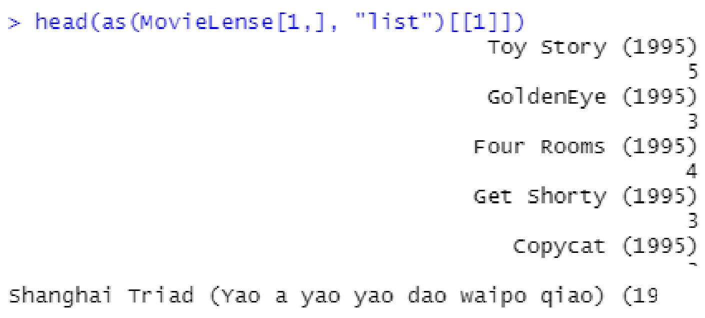
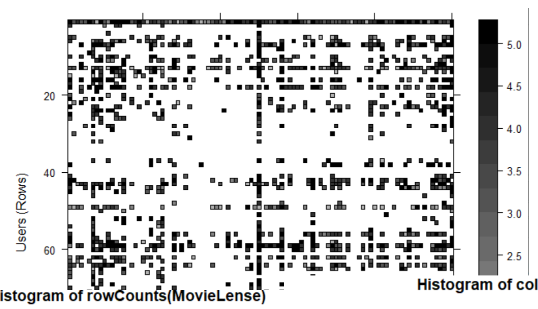
##################################### Task 1#######################################

Visualizing the Movie Lens Dataset ratings for 1st user

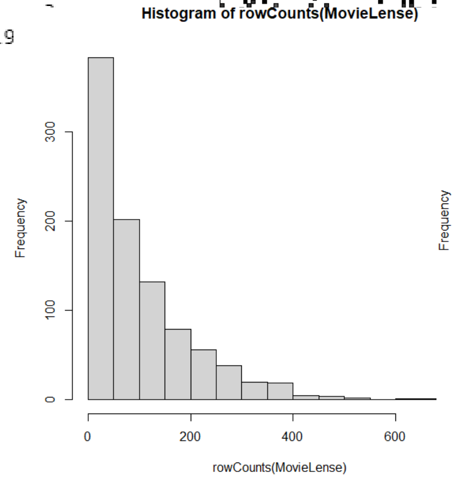


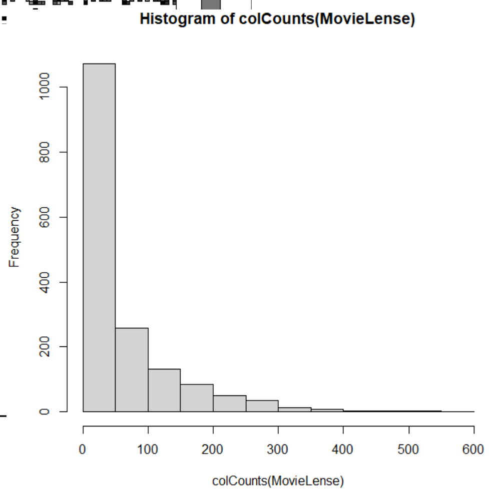
```
> head(as(MovieLense[1,], "list")[[1]])
                              Toy Story (1995)
                                             5
                              GoldenEye (1995)
                                             3
                             Four Rooms (1995)
                                             4
                             Get Shorty (1995)
                                             3
                                Copycat (1995)
                                             
Shanghai Triad (Yao a yao yao dao waipo qiao) (19
```

Visualizing part of the matrix



A

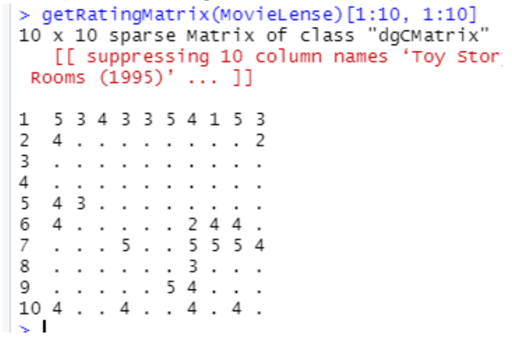




histogram plot of no. of ratings per user

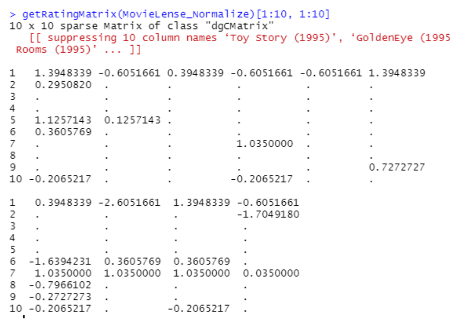
A histogram plot of no. of ratings per movie

A sparse Rating Matrix Display for first 10 users



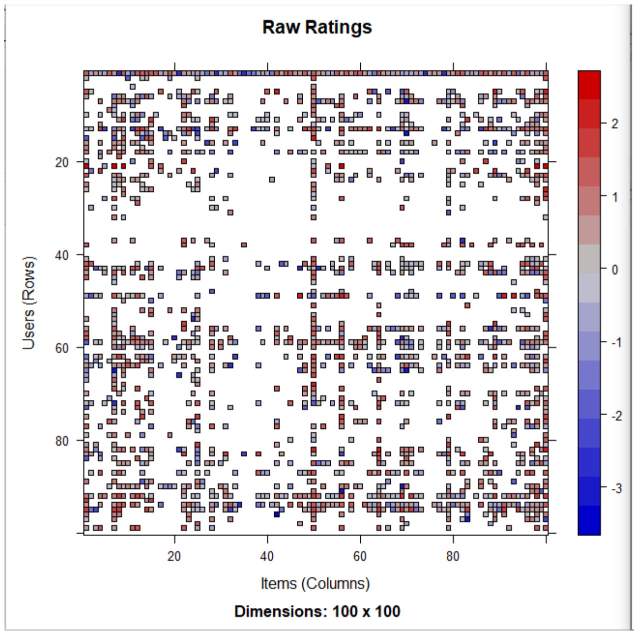
```
> getRatingMatrix(MovieLense)[1:10, 1:10]
10 x 10 sparse Matrix of class "dgCMatrix"
   [[ suppressing 10 column names ‘Toy Stor
 Rooms (1995)' ... ]]

1   5 3 4 3 3 5 4 1 5 3
2   4 . . . . . . . . 2
3   . . . . . . . . . .
4   . . . . . . . . . .
5   4 3 . . . . . . . .
6   4 . . . . . . 2 4 4 .
7   . . . 5 . . 5 5 5 4
8   . . . . . . . 3 . . .
9   . . . . . . 5 4 . . .
10  4 . . 4 . . 4 . 4 .
> 
```

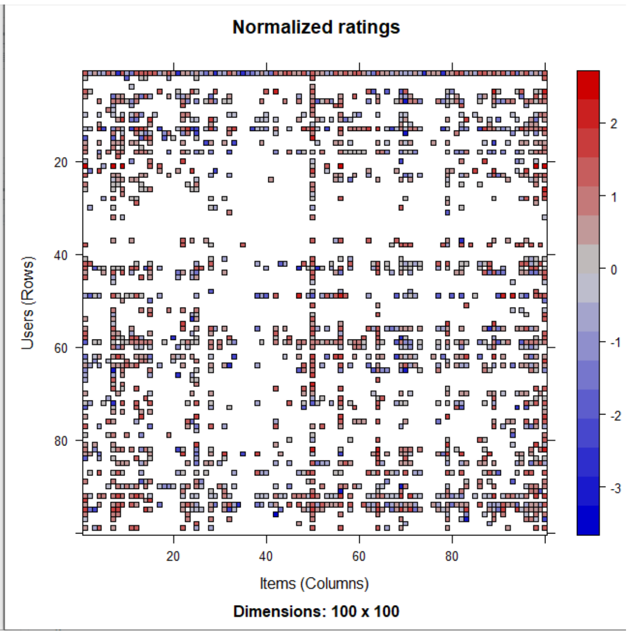
A normalized Rating Matrix for first 10 users

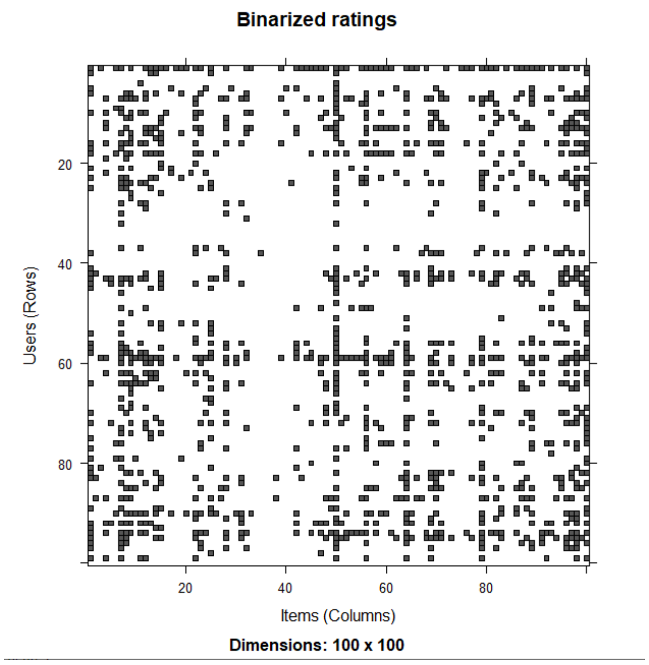


```
> getRatingMatrix(MovieLense_Normalize)[1:10, 1:10]
10 x 10 sparse Matrix of class "dgCMatrix"
   [[ suppressing 10 column names ‘Toy Story (1995)’, ‘GoldenEye (1995
 Rooms (1995)’ ... ]]

1   1.3948339 -0.6051661 0.3948339 -0.6051661 -0.6051661 1.3948339
2   0.2950820  .          .          .          .          .
3   .          .          .          .          .          .
4   .          .          .          .          .          .
5   1.1257143  0.1257143 .          .          .          .
6   0.3605769  .          .          .          .          .
7   .          .          .          1.0350000 .          .
8   .          .          .          .          .          .
9   .          .          .          .          .          0.7272727
10 -0.2065217  .          .         -0.2065217 .          .

1   0.3948339 -2.6051661  1.3948339 -0.6051661
2   .          .          .         -1.7049180
3   .          .          .          .
4   .          .          .          .
5   .          .          .          .
6  -1.6394231  0.3605769  0.3605769  .
7   1.0350000  1.0350000  1.0350000  0.0350000
8  -0.7966102  .          .          .
9  -0.2727273  .          .          .
10 -0.2065217  .         -0.2065217  .
```

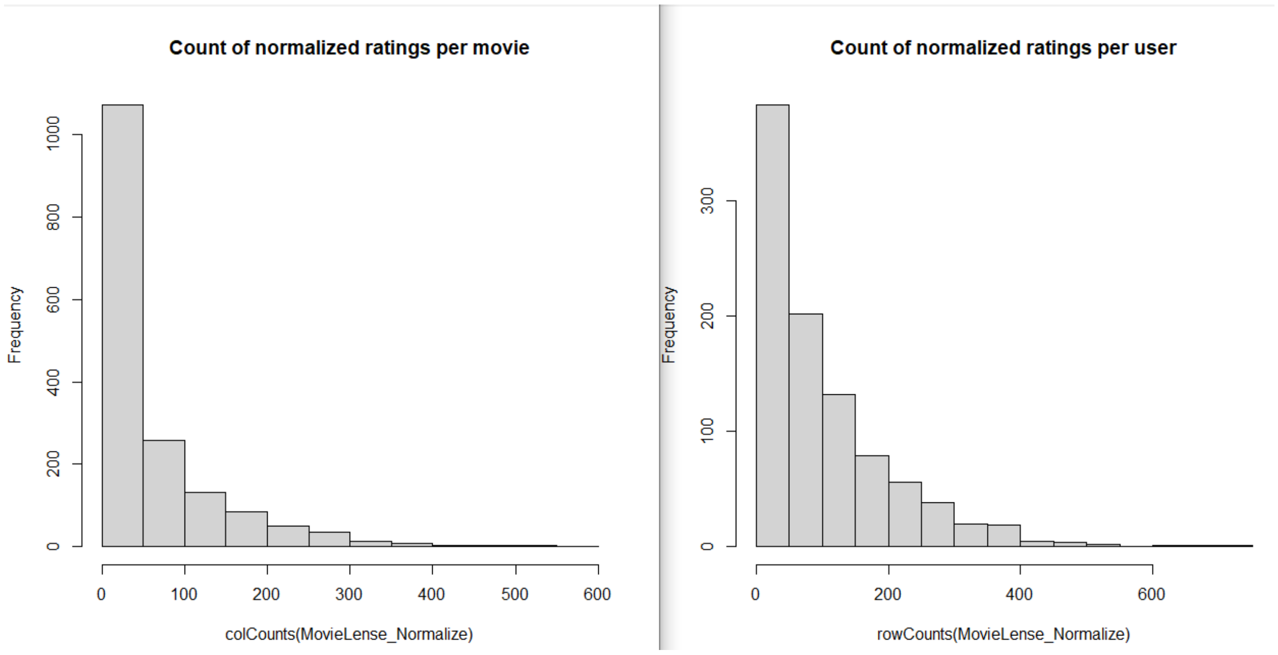
Visualization of sparse matrix ratings and normalized ratings and binarized ratings





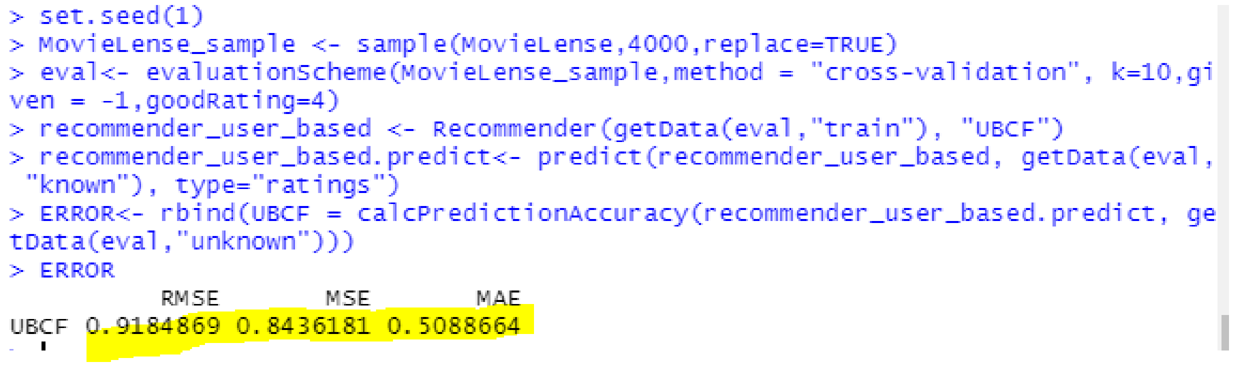


A histogram plot of normalized ratings per user and per movie



The above-normalized ratings are in quite a sync with prior knowledge of user behavior because most users don't give feedback in writing.

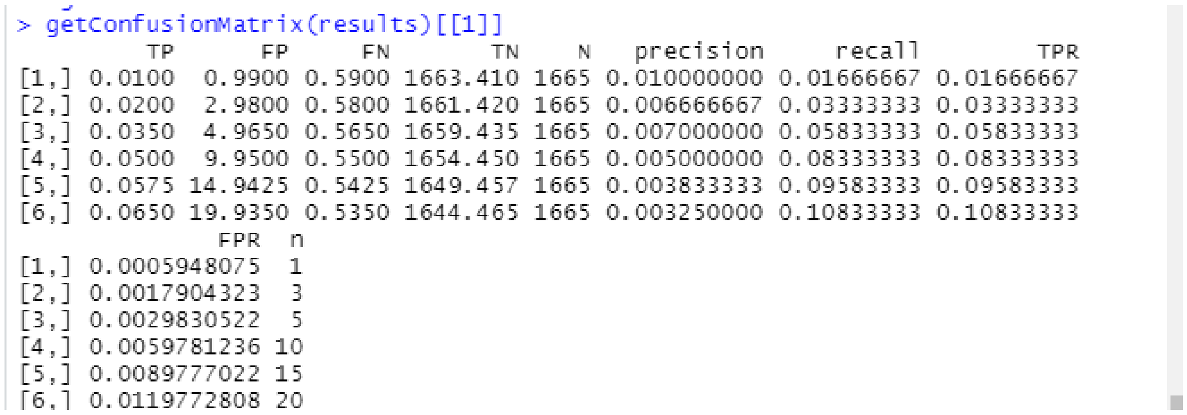
Designed a recommender using user-based collaborative filtering from the recommender lab package. Also evaluated the recommender based on Accuracy performance matrix:



```
> set.seed(1)
> MovieLense_sample <- sample(MovieLense,4000,replace=TRUE)
> eval<- evaluationScheme(MovieLense_sample,method = "cross-validation", k=10,gi
ven = -1,goodRating=4)
> recommender_user_based <- Recommender(getData(eval,"train"), "UBCF")
> recommender_user_based.predict<- predict(recommender_user_based, getData(eval,
 "known"), type="ratings")
> ERROR<- rbind(UBCF = calcPredictionAccuracy(recommender_user_based.predict, ge
tData(eval,"unknown")))
> ERROR
          RMSE       MSE       MAE
UBCF 0.9184869 0.8436181 0.5088664
```

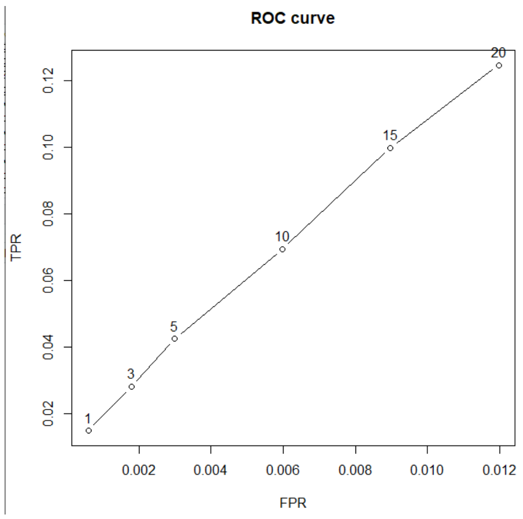
On sampling with different lengths, the recommender system is performing badly which is the lowest though, also if we look into the splitting, we have used the method as cross-validation which usually performs better than random split, need to dig further

<u>Evaluating the user-based recommender for top 1,3,5,10,15and 20 recommendations and checking the confusion matrix for the same</u>



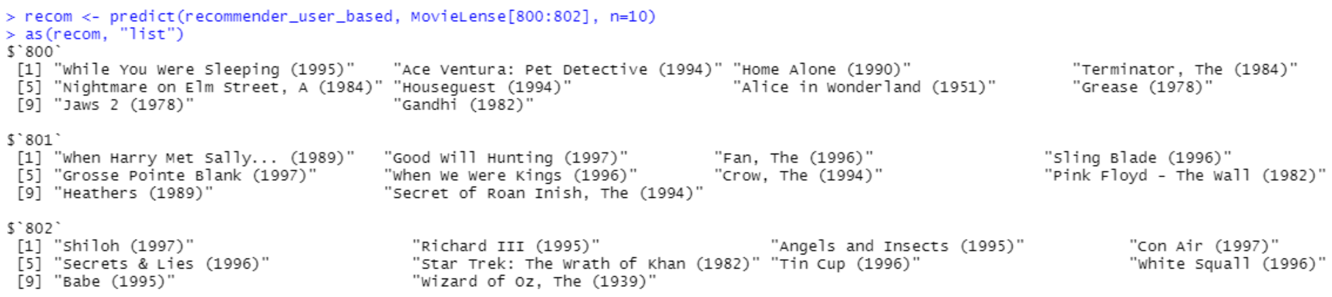
```
> getConfusionMatrix(results)[[1]]
         TP      FP     FN       TN    N   precision     recall        TPR
[1,] 0.0100  0.9900 0.5900 1663.410 1665 0.010000000 0.01666667 0.01666667
[2,] 0.0200  2.9800 0.5800 1661.420 1665 0.006666667 0.03333333 0.03333333
[3,] 0.0350  4.9650 0.5650 1659.435 1665 0.007000000 0.05833333 0.05833333
[4,] 0.0500  9.9500 0.5500 1654.450 1665 0.005000000 0.08333333 0.08333333
[5,] 0.0575 14.9425 0.5425 1649.457 1665 0.003833333 0.09583333 0.09583333
[6,] 0.0650 19.9350 0.5350 1644.465 1665 0.003250000 0.10833333 0.10833333
              FPR  n
[1,] 0.0005948075  1
[2,] 0.0017904323  3
[3,] 0.0029830522  5
[4,] 0.0059781236 10
[5,] 0.0089777022 15
[6,] 0.0119772808 20
```

<u>Plotting the ROC curve for the above user-based recommender system:</u>



<u>From the ROC curve, it's obvious that the recommender system is making a random guess. This cannot be suggested as a good recommender system.</u>

<u>Predicted the top ten movies using the above recommender system for three users</u>



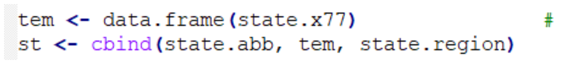
```
> recom <- predict(recommender_user_based, MovieLense[800:802], n=10)
> as(recom, "list")
$`800`
 [1] "While You Were Sleeping (1995)"    "Ace Ventura: Pet Detective (1994)" "Home Alone (1990)"              "Terminator, The (1984)"
 [5] "Nightmare on Elm Street, A (1984)" "Houseguest (1994)"                 "Alice in Wonderland (1951)"      "Grease (1978)"
 [9] "Jaws 2 (1978)"                     "Gandhi (1982)"

$`801`
 [1] "When Harry Met Sally... (1989)"  "Good Will Hunting (1997)"        "Fan, The (1996)"                 "Sling Blade (1996)"
 [5] "Grosse Pointe Blank (1997)"      "When We Were Kings (1996)"       "Crow, The (1994)"                "Pink Floyd - The Wall (1982)"
 [9] "Heathers (1989)"                 "Secret of Roan Inish, The (1994)"

$`802`
 [1] "Shiloh (1997)"                      "Richard III (1995)"                 "Angels and Insects (1995)"          "Con Air (1997)"
 [5] "Secrets & Lies (1996)"              "Star Trek: The Wrath of Khan (1982)" "Tin Cup (1996)"                    "White Squall (1996)"
 [9] "Babe (1995)"                        "Wizard of Oz, The (1939)"
```

############################# Task2###

Note- Could not find data with data(state)

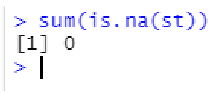
Used this reference: [The R data set state.x77, collected by the US Bureau of the Census in… | Unifolks]

And this command

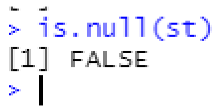


```
tem <- data.frame(state.x77)          #
st <- cbind(state.abb, tem, state.region)
```

Basic Visualization:

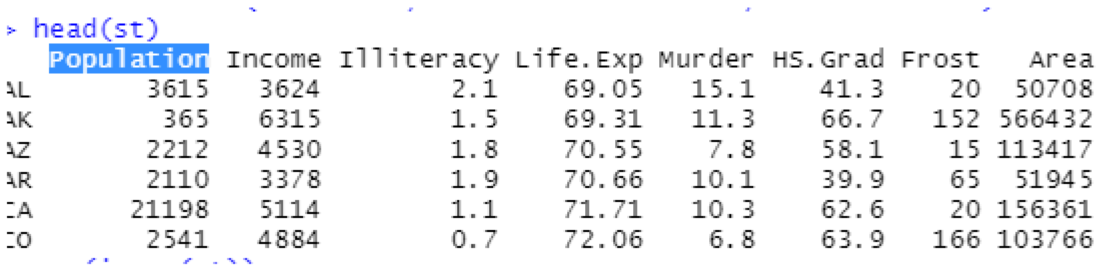


```
> sum(is.na(st))
[1] 0
> |
```



```
> is.null(st)
[1] FALSE
> |
```

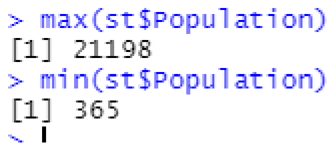
No missing values or null values for the above dataset.



```
> head(st)
   Population Income Illiteracy Life.Exp Murder HS.Grad Frost   Area
AL       3615   3624        2.1    69.05   15.1    41.3    20  50708
AK        365   6315        1.5    69.31   11.3    66.7   152 566432
AZ       2212   4530        1.8    70.55    7.8    58.1    15 113417
AR       2110   3378        1.9    70.66   10.1    39.9    65  51945
CA      21198   5114        1.1    71.71   10.3    62.6    20 156361
CO       2541   4884        0.7    72.06    6.8    63.9   166 103766
```

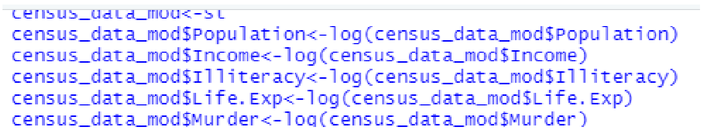
All the above features are at different scales we need to bring all these features variables within the same range of value considering the clustering algorithm which is based on Euclidean distance and complete/single/average linkage for hierarchical clustering otherwise this Euclidean distance would be influenced by features have higher weights and thereby the clusters formed won't be correct.

Below is the snapshot of the code that we would be scaling



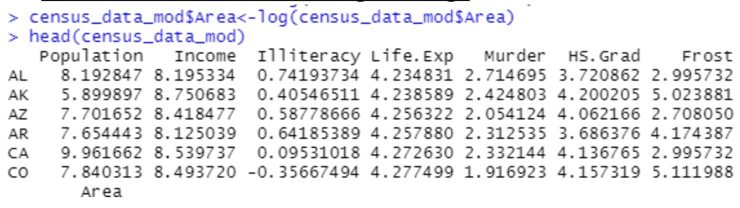
```
> max(st$Population)
[1] 21198
> min(st$Population)
[1] 365
> |
```

The population has two extreme values which cannot be plotted over a single axis hence using a log to scale the feature variables.



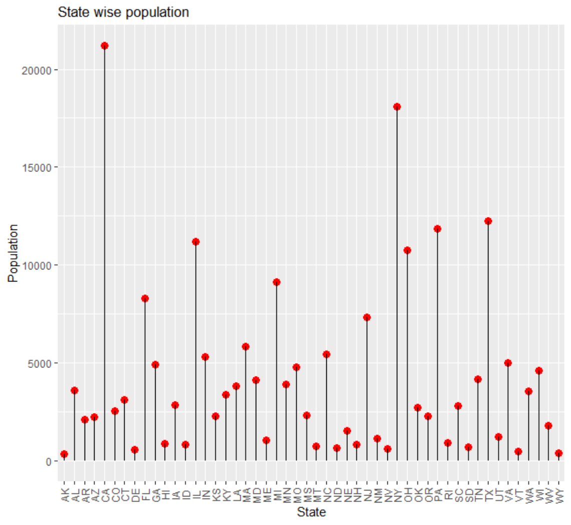
```
census_data_mod<-st
census_data_mod$Population<-log(census_data_mod$Population)
census_data_mod$Income<-log(census_data_mod$Income)
census_data_mod$Illiteracy<-log(census_data_mod$Illiteracy)
census_data_mod$Life.Exp<-log(census_data_mod$Life.Exp)
census_data_mod$Murder<-log(census_data_mod$Murder)
```

Transformed data after log scaling:

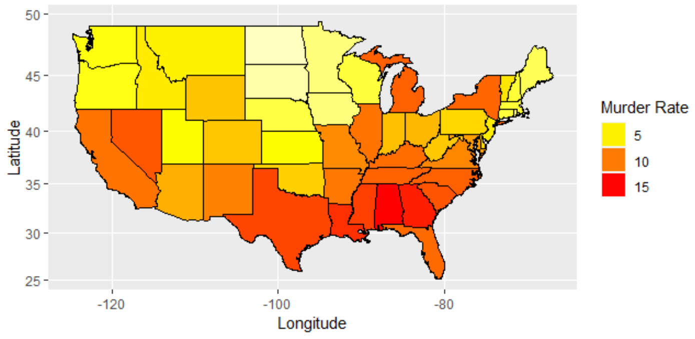


```
> census_data_mod$Area<-log(census_data_mod$Area)
> head(census_data_mod)
   Population   Income  Illiteracy Life.Exp   Murder  HS.Grad    Frost
AL   8.192847 8.195334  0.74193734 4.234831 2.714695 3.720862 2.995732
AK   5.899897 8.750683  0.40546511 4.238589 2.424803 4.200205 5.023881
AZ   7.701652 8.418477  0.58778666 4.256322 2.054124 4.062166 2.708050
AR   7.654443 8.125039  0.64185389 4.257880 2.312535 3.686376 4.174387
CA   9.961662 8.539737  0.09531018 4.272630 2.332144 4.136765 2.995732
CO   7.840313 8.493720 -0.35667494 4.277499 1.916923 4.157319 5.111988
       Area
```

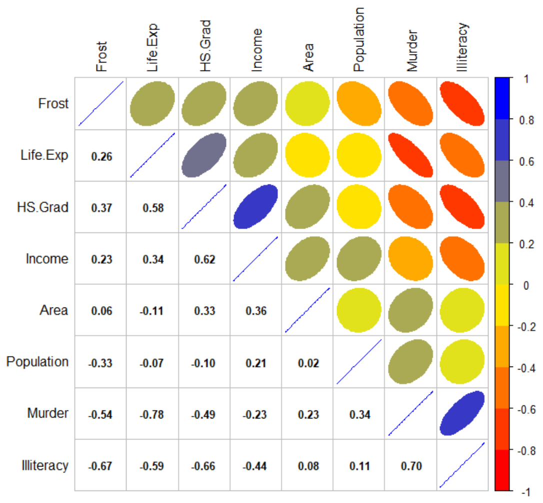
Visualizing the data

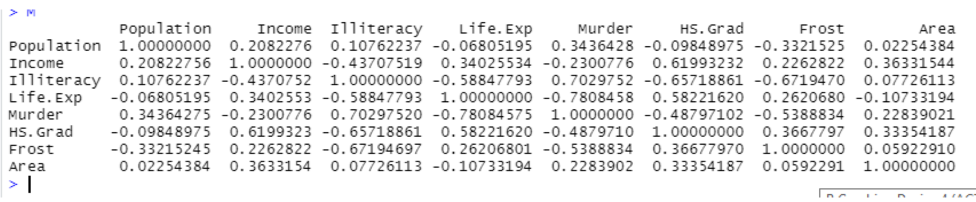


From the plot, we can see that California and New York are the top two states in population. South Dakota has the least population.



The murder rate is higher in south and east states but less in north-central, northwest, and northeast states.





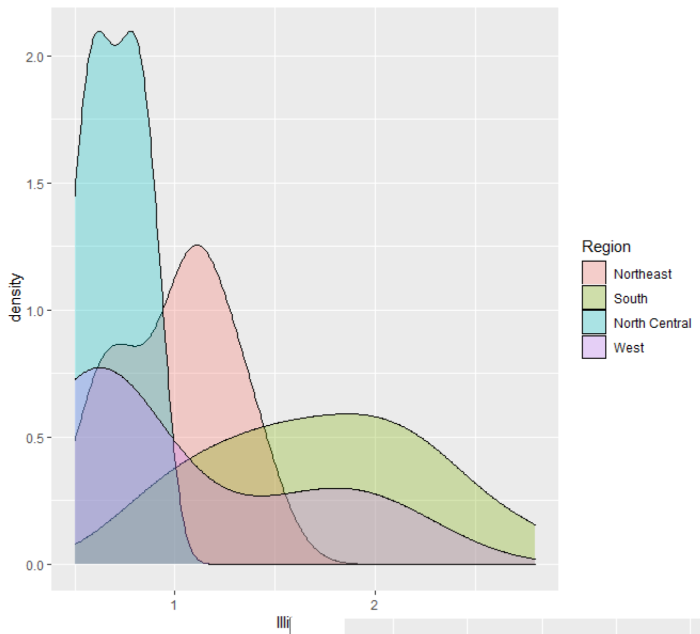
```
> M
            Population     Income  Illiteracy    Life.Exp     Murder     HS.Grad      Frost        Area
Population  1.00000000  0.2082276  0.10762237 -0.06805195  0.3436428 -0.09848975 -0.3321525  0.02254384
Income      0.20822756  1.0000000 -0.43707519  0.34025534 -0.2300776  0.61993232  0.2262822  0.36331544
Illiteracy  0.10762237 -0.4370752  1.00000000 -0.58847793  0.7029752 -0.65718861 -0.6719470  0.07726113
Life.Exp   -0.06805195  0.3402553 -0.58847793  1.00000000 -0.7808458  0.58221620  0.2620680 -0.10733194
Murder      0.34364275 -0.2300776  0.70297520 -0.78084575  1.0000000 -0.48797102 -0.5388834  0.22839021
HS.Grad    -0.09848975  0.6199323 -0.65718861  0.58221620 -0.4879710  1.00000000  0.3667797  0.33354187
Frost      -0.33215245  0.2262822 -0.67194697  0.26206801 -0.5388834  0.36677970  1.0000000  0.05922910
Area        0.02254384  0.3633154  0.07726113 -0.10733194  0.2283902  0.33354187  0.0592291  1.00000000
> |
```

<u>From the above correlation plot, we can make the following inferences</u>

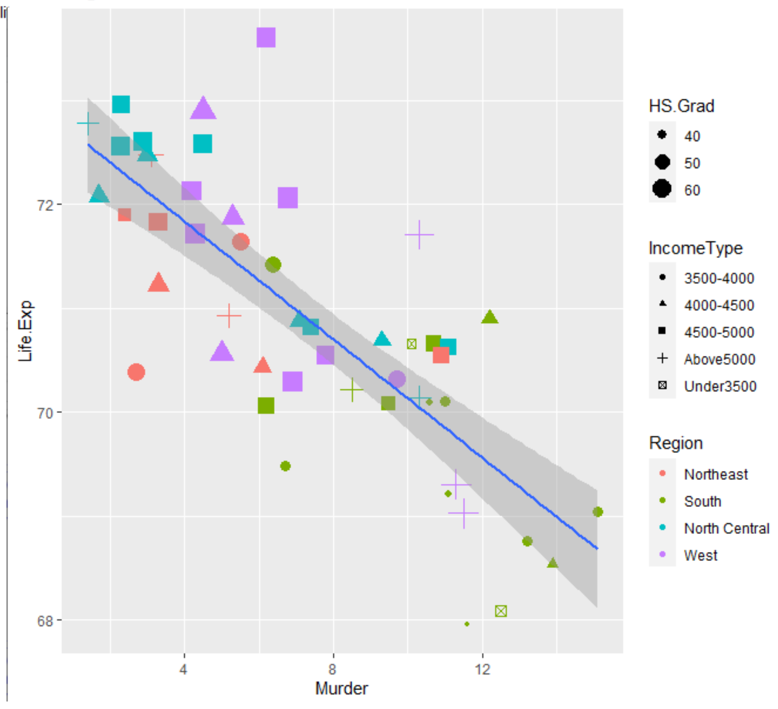
1) ***Murder and Life. Exp shows a high negative correlation***
2) ***Murder and Illiteracy shows a high positive correlation***

The Pearson and Spearman correlation matrix on the bottom-left gives us the R-values between each pair of the variables, which confirm the correlation shape on the top-right.

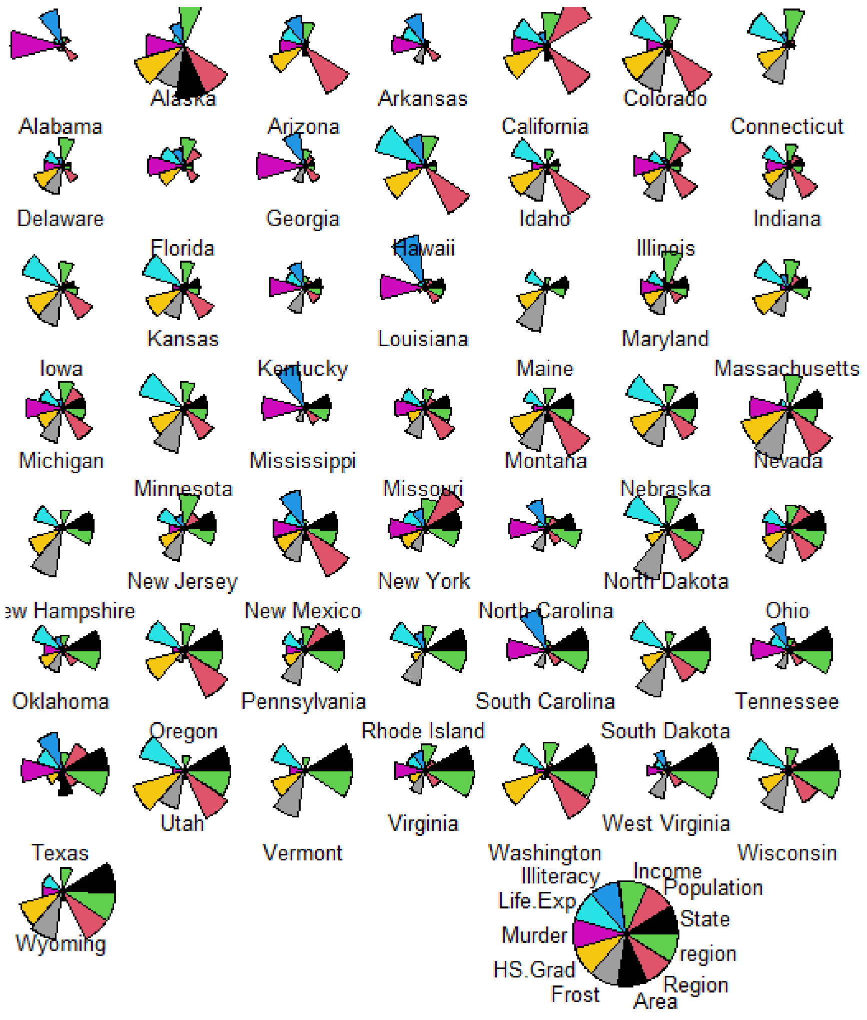
Positive correlation between Murder and Illiteracy with an R-value of 0.70 means that the lower education level the state has, the higher chance of murder rate the state will have; Negative correlations between Murder and Life. Exp, Frost, with R-values of -0.78, and -0.54 illustrate that the more occurrence of murder, the shorter life the state will expect; And the colder the weather, the lower chance the murder will occur.



We can see that the north-central region has narrow density distribution with most Illiteracy less than 1 percent of the population. While the south region has an open and bell-shaped distribution with illiteracy covered from 0.5 to 3. Though region west has a spread-out distribution too, it's left-skewed; There are lots of west states with illiteracy of less than 1% of its population. Most northeast region states have illiteracy less than 1.5% of their population.



Murder is negatively correlated with Life.Exp. Some states with higher murder rates over 12 have relatively small symbols, which means their high school graduation rates are close to 40%; And these small symbols with murder rates bigger than 12 are all colored as green, which means they all belong to the south region. It looks like the income type does not affect the murder rate a lot because all different symbols scatter around in different murder rates, especially between murder rates 8 and 10.Most southern states have lower HS.Grad, lower Life.Exp but higher murder frequency, while states in the other three regions have relatively higher high school graduation rate and income but lower murder rate.

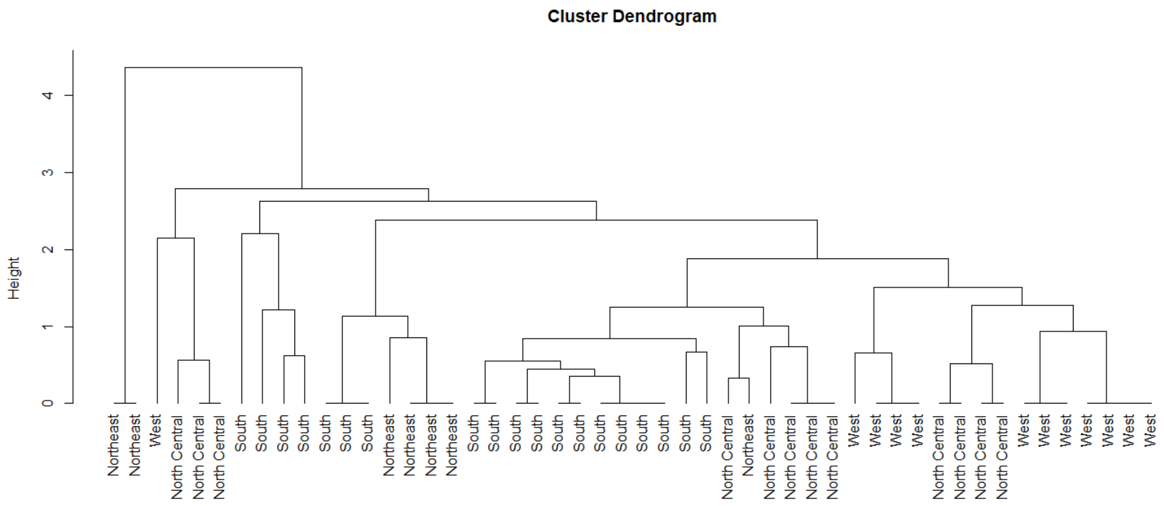


The segment Diagram shows us different aspects of each state. For example, South Dakota has big Frost(yellow), big Life Expectancy(blue), a relatively high percentage of high school graduation rate(pink), and good income(red), but has a small area, and very tiny population, illiteracy, and murder rate compared with the other states.

PartA

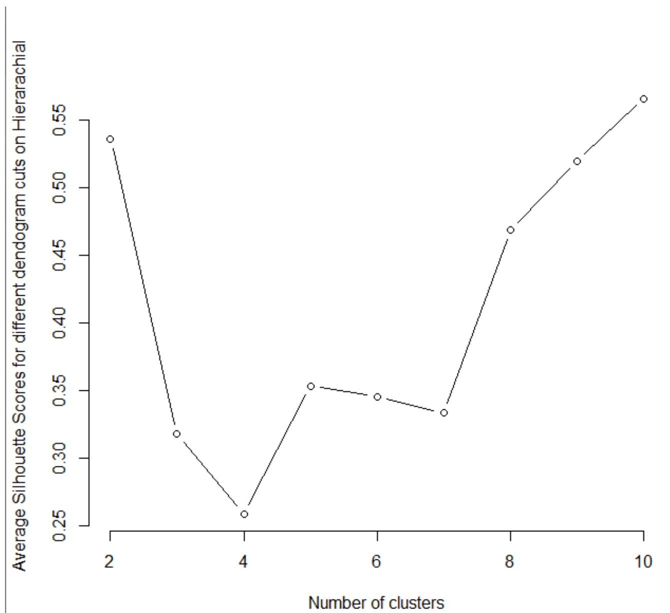
Hierarchical Clustering Visualization

Cluster Dendogram with Region Labels

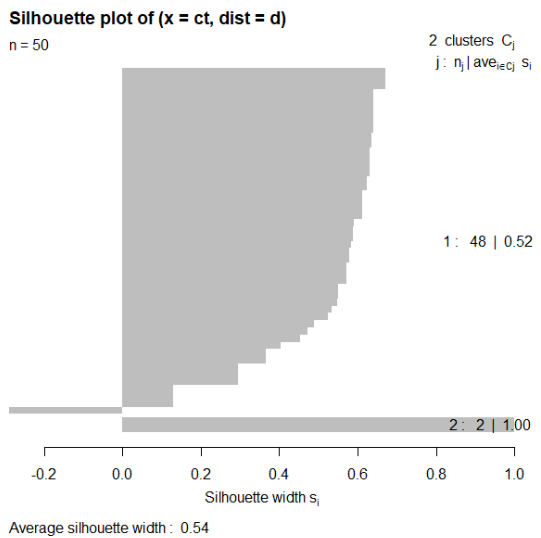


Cluster Dendogram with State Labels

Though the above dendrograms obtained are not very healthy looking but one thing is clear as has been obtained from previous visualizations that many of the southern region states fall close to each other because they have HS. Grad, lower Life. Exp but higher murder frequency, while states in the other three regions have relatively higher high school graduation rate and income but lower murder rate. Also, some northern and western states have higher Life. Exp, Income, and HS. Grad and lower Area, Population, Murder, and Illiteracy, like Nebraska and South Dakota fall together because they have got similarities in this respect which we have visualized earlier. Also, some of the North Central and western states fall close together because these states have high life expectancy rates as has been explained earlier from the density distribution.



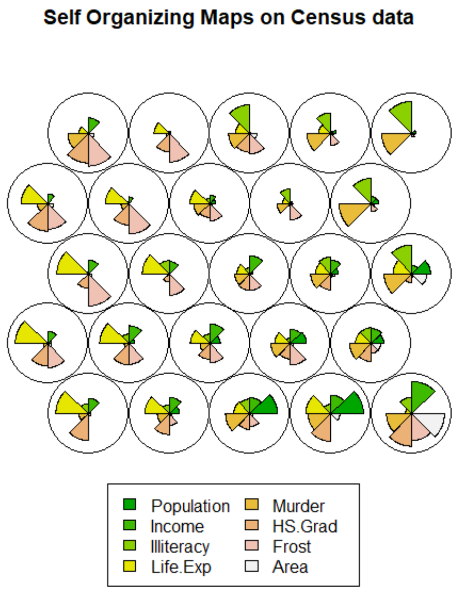
<u>Also, it seems we can get an average</u>



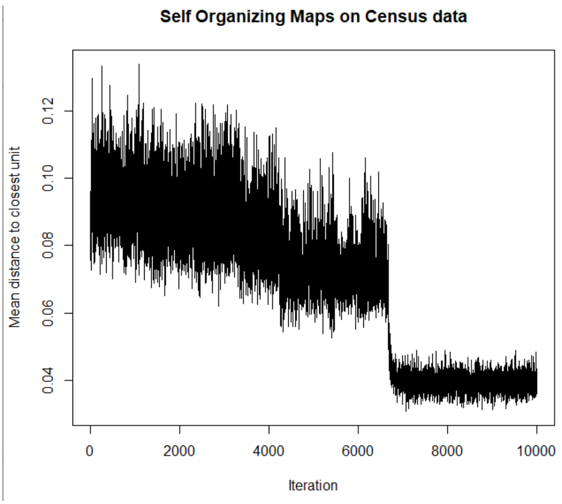
silhouette score of 0.55 with just two clusters however the silhouette plot that we obtained is a very poor plot because we see one region is having a high density and other region is having a very low density, there is negative silhouette score in between. We understand that region-wise, as well as state-wise some of them, have similarities in some respects, but the actual distinguishing feature is murder rates and life expectancy, on the basis of this too we think the states and regions have been clustered into two groups.

Part B)

Modeling on Census data with Self Organizing Maps

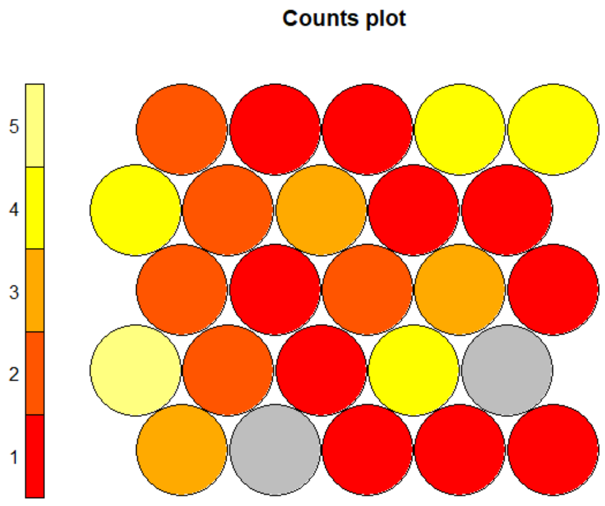


-

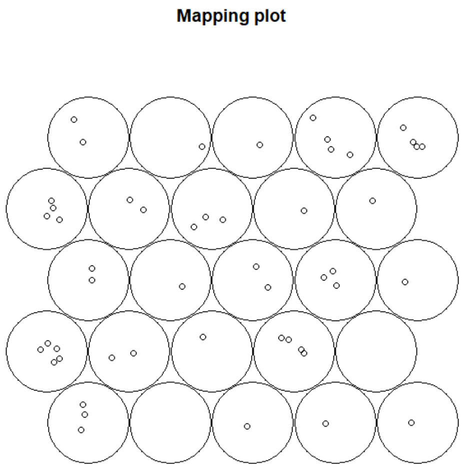


The left figure here represents the 5*5 hexagonal prototypes where each of these prototypes represents all of the census data variables, what distinguishes these prototypes is the magnitude of these variables. So, one thing is very clear mostly the prototypes that have a high value of murder are also having a high value of illiteracy. Further, the life expectancy is also low for such prototypes. We can also see in some of the prototypes in the top left where illiteracy is low, life expectancy is high and murder is also having a good magnitude, maybe there are other factors in these regions that are contributing to murder.

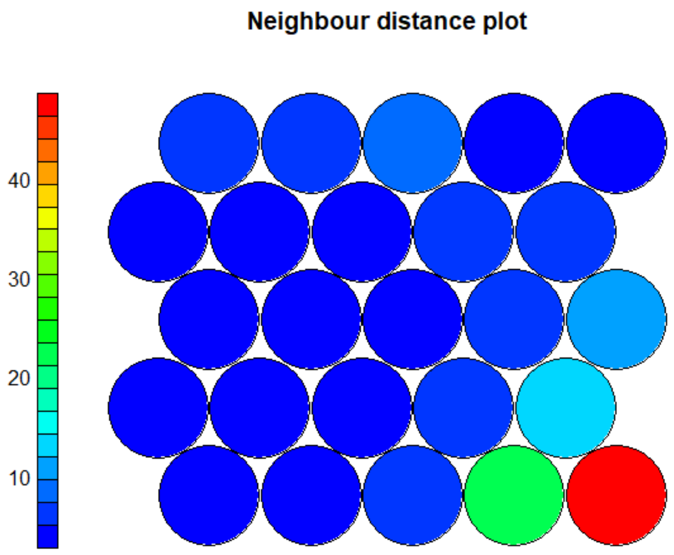
The entire dataset has been exposed to the prototypes 10000 times, what I expected to see is that the distance of each of these random prototypes to the actual dataset has finally converged to a certain minimum distance and that's what I see from my right figure.



The plots indicates the mapping of data points within the

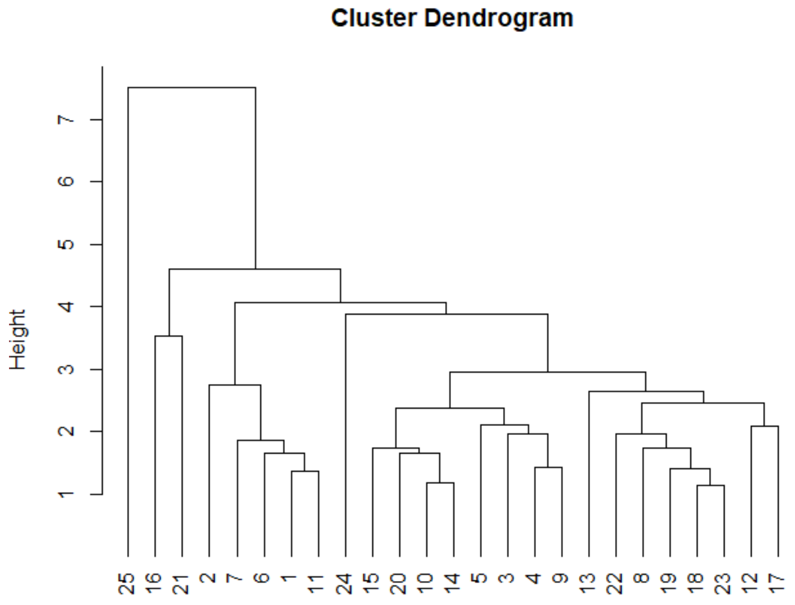


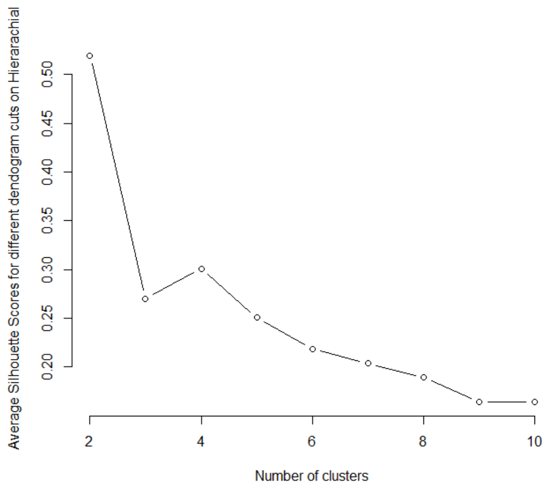
prototypes. There are a total of five mappings here out of which some are even empty (gray/white color)indicating that none of the data points have been mapped to these prototypes. This gives an impression of the distribution of data points in terms of where they map over the grid.

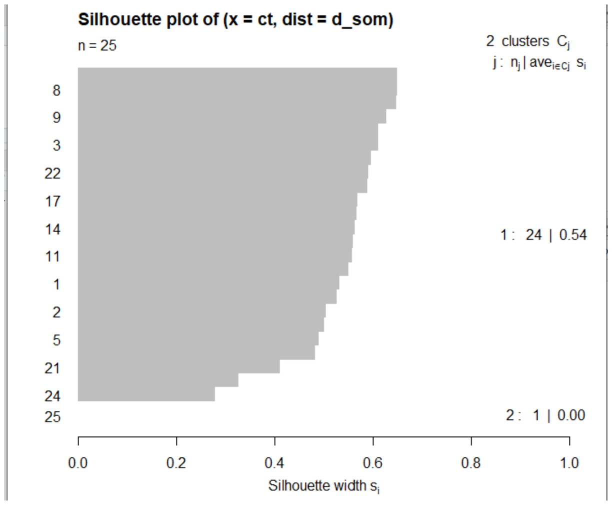


From this neighbor distance plot, we see that the pink prototype one is different from other prototypes and when compared to the code vector plot, we see that for this prototype all the feature variables are having considerable values, life hs grad and population are the highest among all other variables however we cannot ignore the other variables such as murder, life expectancy these are also having a decent magnitude. Also, for this prototype illiteracy is quite low but murder is considerably high, which is quite amazing when compared with the population size, this is something we need to dig into further. Also, we see the black ones in the middle and lower region are really similar maybe because if I compare this distance plot to the code vector plot, I see that these prototypes are having high magnitudes of life expectancy and hs grad and low murder and low illiteracy.

Hierarchical Clustering Dendogram plot with SOMs

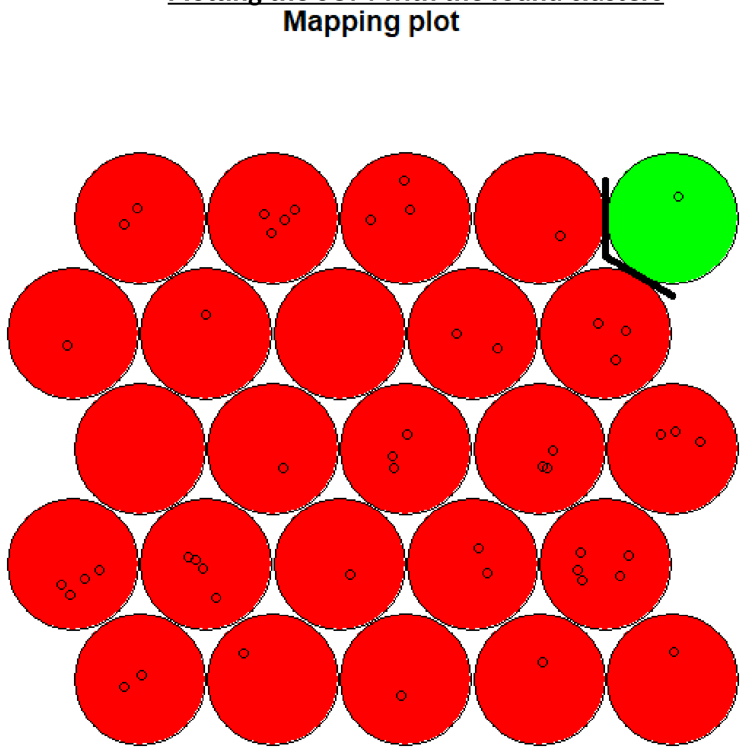






The average silhouette score of Hierarchical clustering on SOMs is high for k=2 which is also quite in line with what we found with Hierarchical clusterings on actual data however the silhouette plot for k=2 is quite different if we compare it with the one obtained from Hierarchial clustering on actual data because on performing Hierarchical clustering on the actual data we were finding two groups one group being highly dense however for Hierarchical clustering on SOMs the silhouette plot for k=2 suggest that this prototype 25 is entirely different from other prototypes and cannot be grouped with any other prototype.

Plotting the SOM with the found clusters



This mapping plot of SOMs with the clusters for k=2 again conforms with our earlier findings and visualizations on the dataset which is indicative of the fact that there are mainly 2 distinctive clusters in our dataset. We cannot directly say whether the clustering obtained is correct or not, but we can infer from this analysis, as well as our prior knowledge that mostly the southern regions are having high murder rates and low literacy, and the other regions, are having high literacy as we as high life expectancy. These are the distinctive features of these clusters.

Part C)

There are two important differences between clustering with SOMs and clustering directly on data with the context of this problem.

1)***SOMs give a better interpretation of data in terms of the feature variables, we can look at the visualization and try to figure out what is the distinctive feature between these clusters in terms of the feature variables however with hierarchical clustering directly on data I won't be able to say why or in what sense my clusters differ from each other.***

2)***If the data set is simple so that we can make inferences about the data through our existing statistical tools and visualization problems for example in this problem I could figure out which regions are having high murder rate and what could be the possible reasons behind that without using SOMS, from the existing graphical summaries and plotting's, in this scenario we can opt for direct clustering on the data because with SOMs we need to tune several parameters.***

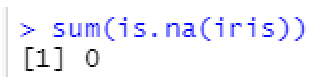
3)In general SOM is superior in dealing with processes that have multiple optima.

4)***Also, SOM offers the opportunity for an early exploration of the search space, and as the process continues it gradually narrows the search.***

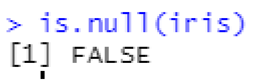
######################Task3###

__Visualizing the using graphical and statistical summaries__

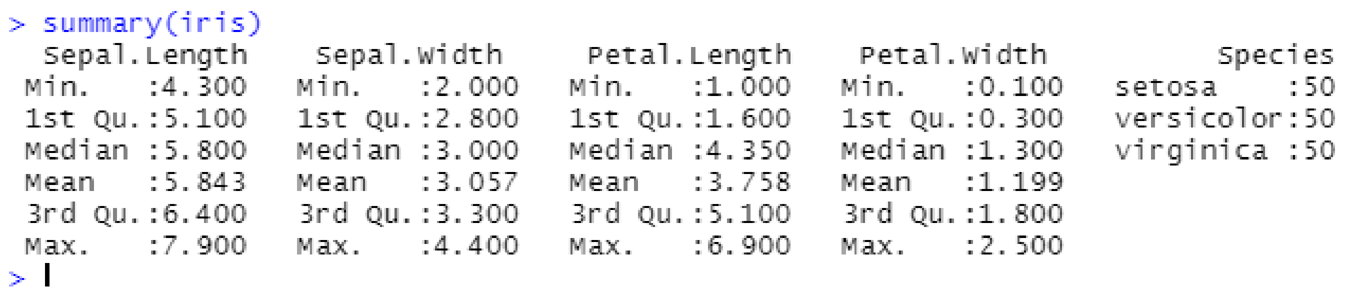
__Checking for missing and null values__



```
> sum(is.na(iris))
[1] 0
```

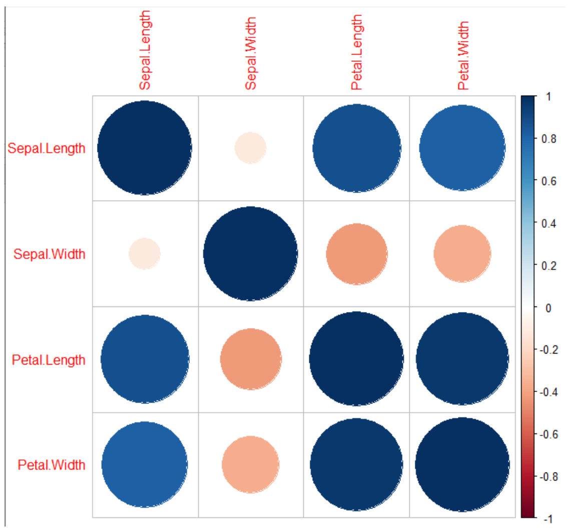


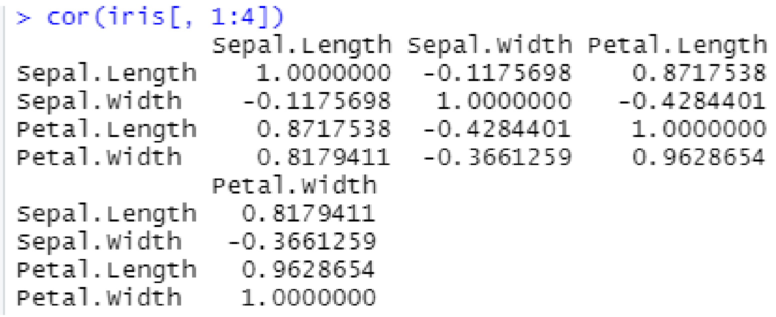
```
> is.null(iris)
[1] FALSE
```



```
> summary(iris)
  Sepal.Length    Sepal.Width     Petal.Length    Petal.Width          Species  
 Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100   setosa    :50  
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300   versicolor:50  
 Median :5.800   Median :3.000   Median :4.350   Median :1.300   virginica :50  
 Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199                  
 3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800                  
 Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500                  
> 
```

<u>Correlation plot of all the variables of iris data</u>

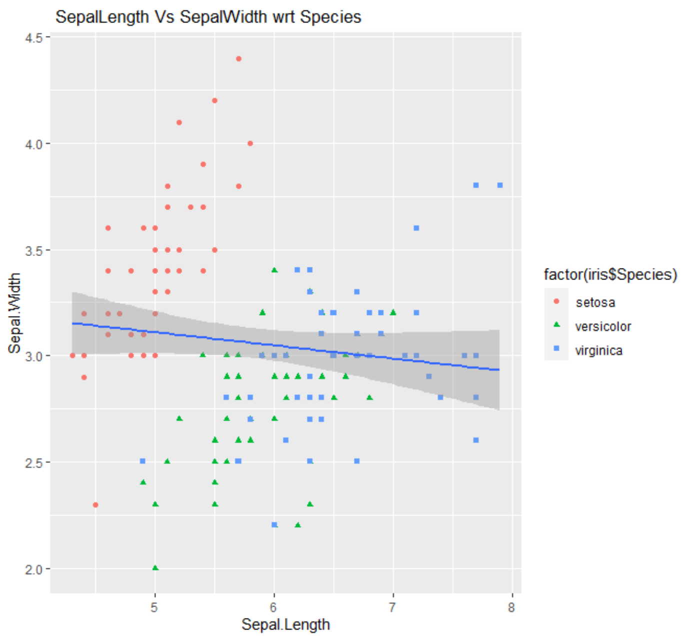


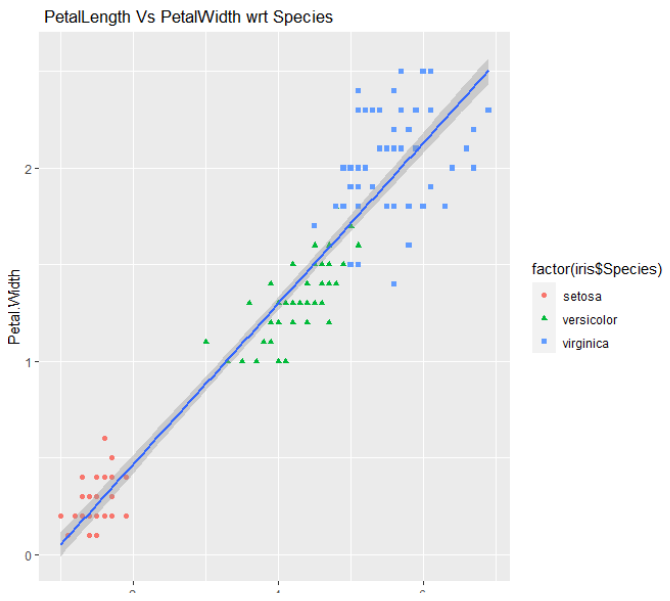


```
> cor(iris[, 1:4])
             Sepal.Length Sepal.Width Petal.Length
Sepal.Length    1.0000000  -0.1175698    0.8717538
Sepal.Width    -0.1175698   1.0000000   -0.4284401
Petal.Length    0.8717538  -0.4284401    1.0000000
Petal.Width     0.8179411  -0.3661259    0.9628654
             Petal.Width
Sepal.Length   0.8179411
Sepal.Width   -0.3661259
Petal.Length   0.9628654
Petal.Width    1.0000000
```

We see that petal length and petal width of the flowers are highly positively correlated also petal length and petal width are highly correlated, but these correlations don't give us much distinctive information about the different iris species hence we need some morevisualizations and statistical analysis.

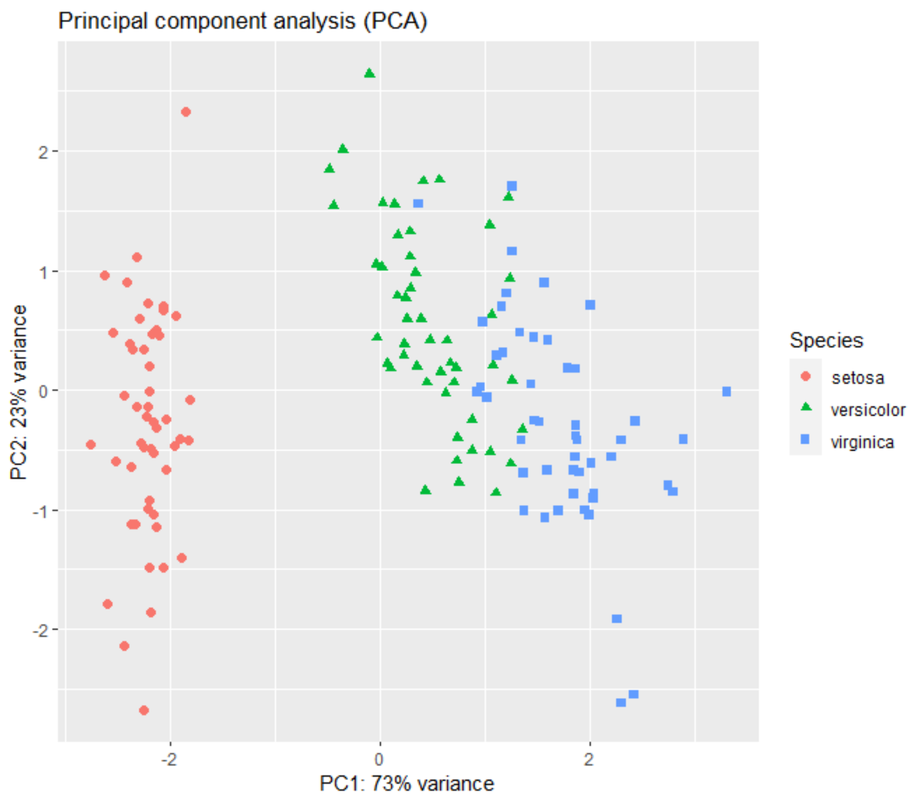
<u>Identifying the features which would help the Kmeans model to perform best on this iris dataset</u>





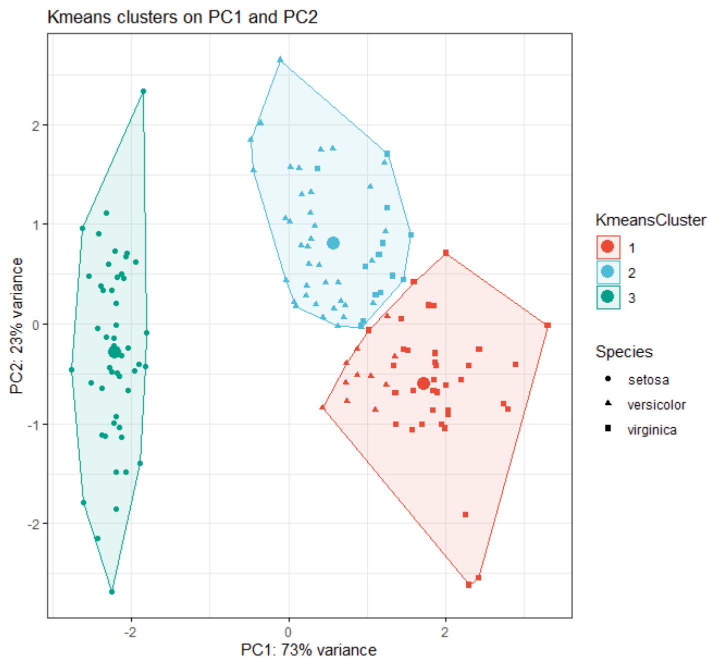
So here we are trying to plot the iris species concerning different feature variables on the top of which we are fitting a linear regression line to check which variables can create the maximum separation between the different species. The left figure shows iris species plotted concerning feature variables Sepal Length and Sepal Width and the right figure shows the iris species plotted concerning feature variables Petal Length and Petal Width. From the first figure, most of the Versicolor and Virginica is captured on one side of the line and most of the setosa on the other side of the line with few misclassifications. However, in the second figure, we see approximately half of every species lies on each side of the line. Though one thing is very clear that Virginica and Versicolor are very similar from both the plots, there does not seem to be lots of variation maybe because it's nonlinear data and we are trying to separate these different classes with a linear line segment. However, in terms of feature selections, we can use Sepal length and Sepal Width for KMeans.

Part a) ***Creating a plot using the first two principal components and coloring the iris species by class.***



We can see that the first principal component alone is useful in distinguishing the three species.

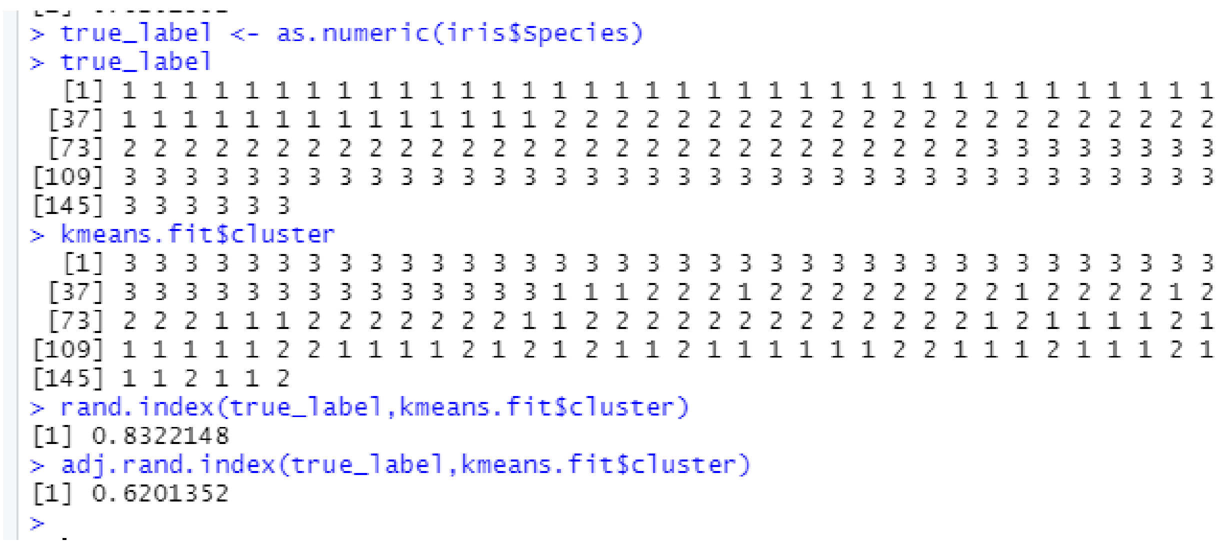
Part b)***Creating a plot of the clusters of different colors and specifying different symbols to depict the species labels***.



<u>Thus, from the above gscatter plot, we see that with k=3 species “Setosa” is well separated from the other two varieties of Iris Flower as it has been captured in a different cluster however the other combination of the other two falls into two different clusters. Thus, Kmeans along with PCA performs a very good job in capturing this information which indicates that the two varieties of Iris Species are quite similar however Setosa is quite different from these two varieties of IRIS flower.</u>

Part C)

<u>Rand Index and Adjusted Rand Index</u>



```
> true_label <- as.numeric(iris$Species)
> true_label
  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [37] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 [73] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3
[109] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
[145] 3 3 3 3 3 3
> kmeans.fit$cluster
  [1] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
 [37] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 1 1 2 2 2 1 2 2 2 2 2 2 2 2 1 2 2 2 2 1 2
 [73] 2 2 2 1 1 1 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 1 1 1 1 2 1
[109] 1 1 1 1 1 2 2 1 1 1 1 2 1 2 1 2 1 1 2 1 1 1 1 1 1 2 2 1 1 1 2 1 1 1 2 1
[145] 1 1 2 1 1 2
> rand.index(true_label,kmeans.fit$cluster)
[1] 0.8322148
> adj.rand.index(true_label,kmeans.fit$cluster)
[1] 0.6201352
>
```

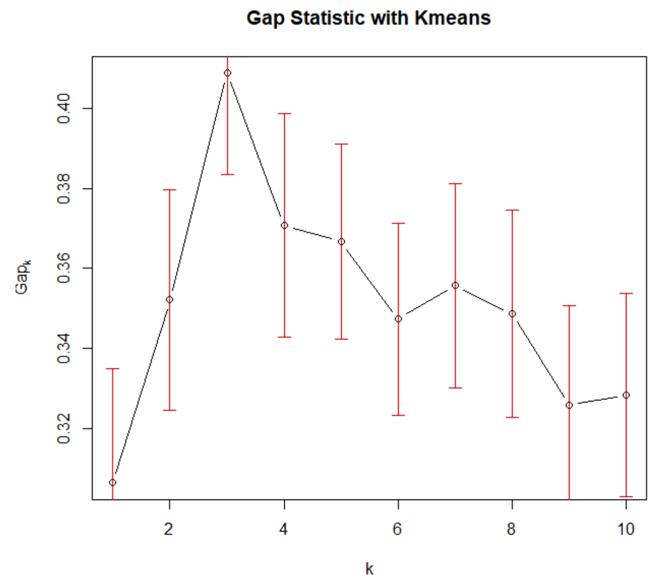
$$AdjustedIndex = \frac{Index - ExpectedIndex}{MaxIndex - ExpectedIndex}$$

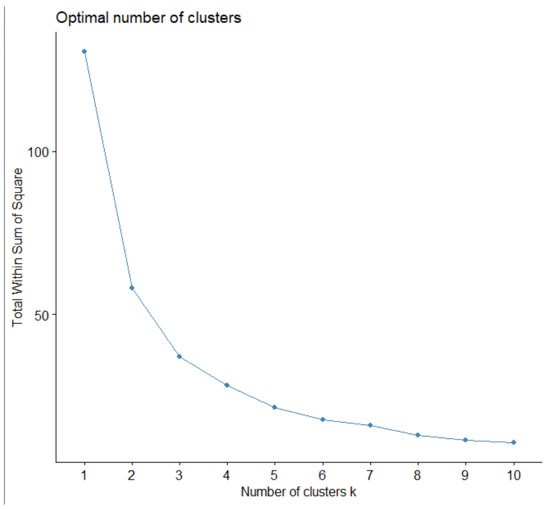
<u>The Rand index suggests that the k-means clustering of the iris data using sepal length and sepal width measurements is similar to the real “clustering” of the data. However, since the Adjusted Rand index is a correction of the Rand index that measures the similarity between two classifications which in this case is basically the true labels and Kmeans clusters of the same objects by the proportions of agreements between the true labels and Kmeans clustering therefore on the basis of Adjusted Rand Index we can say that the Kmeans doesn’t seem to be in line with the actual labels but what we cannot confirm is that whether Kmeans clustering is truly good on iris dataset because Adjusted Rand Index has also some limitations such as it is not appropriate when the two clusterings are dependent and it ignores randomness of the sampling.</u>

Part D)

<u>Gap statistic and silhouette plots to determine the number of clusters</u>

$$\hat{k} = \text{smallest } k \text{ such that } \mathrm{Gap}(k) \geqslant \mathrm{Gap}(k+1) - s_{k+1}.$$

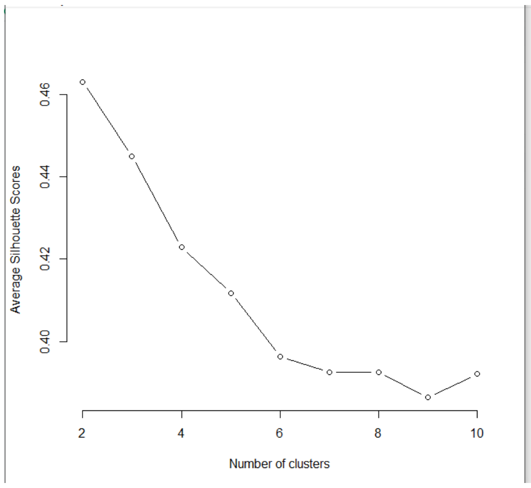


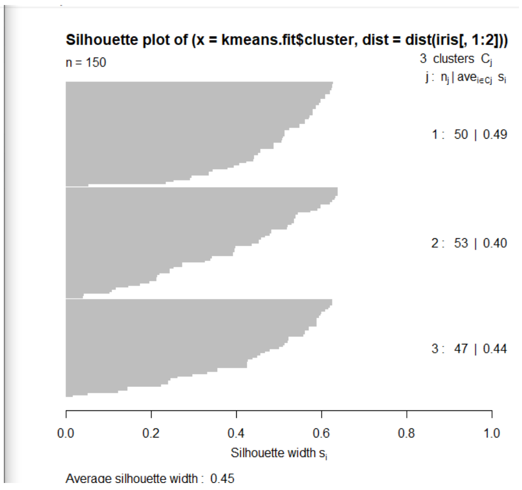


We have already shown how sepal length and sepal width are stronger predictors compared to Petal Length and Petal width hence for Gap Statistics as well as Silhouette plots, we have considered these two features. Below is the snapshot of the code for both plots.

The gap statistic compares the total within intracluster variation for different values of k with their expected values under null reference distribution of the data, i.e., a distribution with no obvious clustering. To obtain an ideal clustering, we should select k such that we maximize the gap statistic.

However, in many real-world datasets, the clusters are not as well-defined, and we want to be able to balance maximizing the gap statistic with the parsimony of the model. Thus, according to this statement and the formula for gap statistics we should look at the ideal value for k so that there is a difference of one standard error between k and k+1 clusters which is 3 in this case as reflected in the left figure. Also, these gap statistics results match the elbow method shown under the right-hand side plot where for k=3 we find the least within-cluster sum of squares.

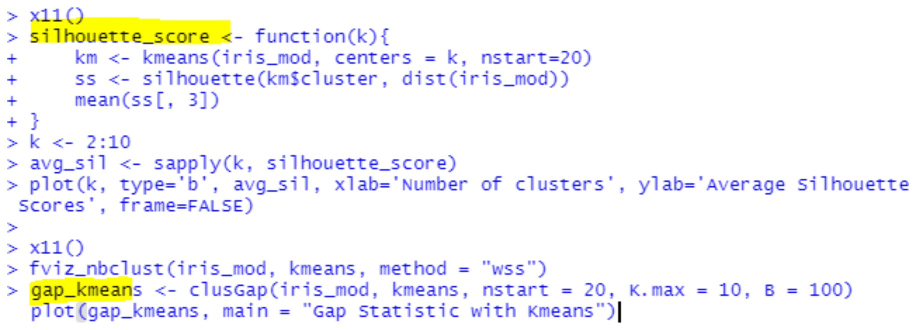




<u>The left figure depicts the average silhouette score for k within a range of 2 to 10 and the right figure represents the average Silhouette width for k=3 which is 0.45. From the left figure, we can infer if we take k=3 then the average silhouette width decreases from 0.46 for k=2 to 0.45 for k=3 which is not a significant decrease. We desire high Silhouette scores, the more the better so we can take k=3 for the iris dataset.</u>

<u>Thus, gap statistics, elbow method, and silhouette plot, the results from these three plots are in line with each other suggesting that k=3 is considered as the optimal no. of clusters when applying means on the iris dataset.</u>

<u>Code for Gap Statistics and Silhouette:</u>


```
> x11()
> silhouette_score <- function(k){
+     km <- kmeans(iris_mod, centers = k, nstart=20)
+     ss <- silhouette(km$cluster, dist(iris_mod))
+     mean(ss[, 3])
+ }
> k <- 2:10
> avg_sil <- sapply(k, silhouette_score)
> plot(k, type='b', avg_sil, xlab='Number of clusters', ylab='Average Silhouette
 Scores', frame=FALSE)
>
> x11()
> fviz_nbclust(iris_mod, kmeans, method = "wss")
> gap_kmeans <- clusGap(iris_mod, kmeans, nstart = 20, K.max = 10, B = 100)
  plot(gap_kmeans, main = "Gap Statistic with Kmeans")
```