R Programming: K-Means Clustering

```r
# Question: Perform clustering analysis on the following dataset using the K-Means clustering algorithm
#
# Load and view the dataset
# ---
# Importing the dataset
# ---
#require("datasets")
# Loading the Iris dataset
# ---
#
data("iris")
# Viewing the structure of the dataset
# ---
#
str(iris)
```

```
## 'data.frame':    150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
```

Finding the summary of the dataset

```r
# Viewing the statistical summary of the dataset
# ---
#
summary(iris)
```

```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width   
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100  
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300  
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300  
##  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  
##        Species  
##  setosa    :50  
##  versicolor:50  
##  virginica :50  
##                 
##                 
## 
```

Checking the head of the dataset

```r
# Previewing the dataset
# ---
#
head(iris)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

```r
# Pre processing the dataset
# Since clustering is a type of Unsupervised Learning,
# we would not require Class Label(output) during execution of our algorithm.
# We will, therefore, remove Class Attribute "Species" and store it in another variable.
# We would then normalize the attributes between 0 and 1 using our own function.
# ---
#
iris.new<- iris[, c(1, 2, 3, 4)]
iris.class<- iris[, "Species"]
head(iris.new)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width
## 1          5.1         3.5          1.4         0.2
## 2          4.9         3.0          1.4         0.2
## 3          4.7         3.2          1.3         0.2
## 4          4.6         3.1          1.5         0.2
## 5          5.0         3.6          1.4         0.2
## 6          5.4         3.9          1.7         0.4
```

```r
# Previewing the class column
# ---
#
head(iris.class)
```

```
## [1] setosa setosa setosa setosa setosa setosa
## Levels: setosa versicolor virginica
```

Normalizing the our dataset

```r
# Normalizing the dataset so that no particular attribute
# has more impact on clustering algorithm than others.
# ---
#
normalize <- function(x){
 return ((x-min(x)) / (max(x)-min(x)))
}
iris.new$Sepal.Length<- normalize(iris.new$Sepal.Length)
iris.new$Sepal.Width<- normalize(iris.new$Sepal.Width)
iris.new$Petal.Length<- normalize(iris.new$Petal.Length)
```

```r
iris.new$Petal.Width<- normalize(iris.new$Petal.Width)
head(iris.new)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width
## 1   0.22222222   0.6250000   0.06779661  0.04166667
## 2   0.16666667   0.4166667   0.06779661  0.04166667
## 3   0.11111111   0.5000000   0.05084746  0.04166667
## 4   0.08333333   0.4583333   0.08474576  0.04166667
## 5   0.19444444   0.6666667   0.06779661  0.04166667
## 6   0.30555556   0.7916667   0.11864407  0.12500000
```

```r
# Applying the K-means clustering algorithm with no. of centroids(k)=3
# ---
#
result<- kmeans(iris.new,3)
# Previewing the no. of records in each cluster
#
result$size
```

```
## [1] 50 39 61
```

```r
# Getting the value of cluster center datapoint value(3 centers for k=3)
# ---
#
result$centers
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width
## 1    0.1961111   0.5950000   0.07830508  0.06083333
## 2    0.7072650   0.4508547   0.79704476  0.82478632
## 3    0.4412568   0.3073770   0.57571548  0.54918033
```

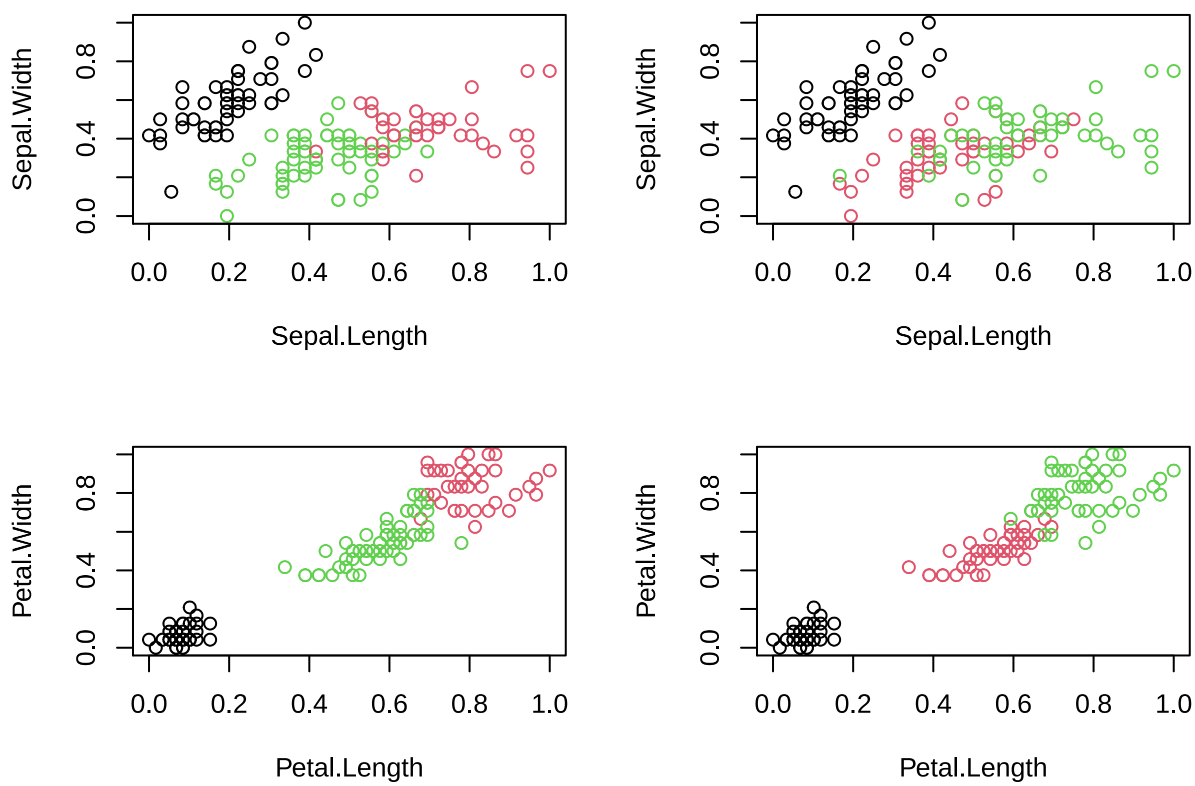
```r
# Getting the cluster vector that shows the cluster where each record falls
# ---
#
result$cluster
```

```
##   [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 1
##  [38] 1 1 1 1 1 1 1 1 1 1 1 1 1 2 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [75] 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 3 2 2 2 2 3 2 2 2 2
## [112] 2 2 3 2 2 2 2 2 2 3 2 3 2 3 2 2 3 3 2 2 2 2 2 3 3 2 2 2 3 2 2 2 3 2 2 2 3 2
## [149] 2 3
```

```r
# The graph shows that we have got 3 clearly distinguishable clusters for Ozone and Solar.R data points
# Let's see how clustering has performed on Wind and Temp attributes.
```

```r
# Verifying the results of clustering
# ---
#
par(mfrow = c(2,2), mar = c(5,4,2,2))
# Plotting to see how Sepal.Length and Sepal.Width data points have been distributed in clusters
```

```
plot(iris.new[c(1,2)], col = result$cluster)
# Plotting to see how Sepal.Length and Sepal.Width data points have been distributed
# originally as per "class" attribute in dataset
# ---
#
plot(iris.new[c(1,2)], col = iris.class)
# Plotting to see how Petal.Length and Petal.Width data points have been distributed in clusters
# ---
#
plot(iris.new[c(3,4)], col = result$cluster)
plot(iris.new[c(3,4)], col = iris.class)
```




```
# Result of table shows that Cluster 1 corresponds to Virginica,
# Cluster 2 corresponds to Versicolor and Cluster 3 to Setosa.
# ---
#
table(result$cluster, iris.class)
```

```
##    iris.class
##     setosa versicolor virginica
##   1     50          0         0
##   2      0          3        36
##   3      0         47        14
```

In order to improve this accuracy further, we may try different values of "k". In some cases, it is also beneficial to change the algorithm in case k-means is unable to yield good results.

Challenge 1

```
# Question: Apply unsupervised learning to the given airquality dataset below.
#
# Load and view the dataset
# ---
# Importing the dataset
# ---
#
data("airquality")
str(airquality)
```

```
## 'data.frame':    153 obs. of  6 variables:
##  $ Ozone  : int  41 36 12 18 NA 28 23 19 8 NA ...
##  $ Solar.R: int  190 118 149 313 NA NA 299 99 19 194 ...
##  $ Wind   : num  7.4 8 12.6 11.5 14.3 14.9 8.6 13.8 20.1 8.6 ...
##  $ Temp   : int  67 72 74 62 56 66 65 59 61 69 ...
##  $ Month  : int  5 5 5 5 5 5 5 5 5 5 ...
##  $ Day    : int  1 2 3 4 5 6 7 8 9 10 ...
```

Filling the ozone columnj

```
# The output shows that only Ozone and Solar.R attributes have NA i.e. some missing value.
# Impute monthly mean in Ozone by running the code shown below
#
airquality$Ozone[is.na(airquality$Ozone)] <- mean(airquality$Ozone, na.rm = TRUE)
# print the dt table below
#head(airquality)
```

filling the solar column

```
# The output shows that only Ozone and Solar.R attributes have NA i.e. some missing value.
# Impute monthly mean in Ozone by running the code shown below
#
airquality$Solar.R[is.na(airquality$Solar.R)] <- mean(airquality$Solar.R, na.rm = TRUE)
# print the dt table below
head(airquality)
```

```
##      Ozone  Solar.R Wind Temp Month Day
## 1 41.00000 190.0000  7.4   67     5   1
## 2 36.00000 118.0000  8.0   72     5   2
## 3 12.00000 149.0000 12.6   74     5   3
## 4 18.00000 313.0000 11.5   62     5   4
## 5 42.12931 185.9315 14.3   56     5   5
## 6 28.00000 185.9315 14.9   66     5   6
```

Finding the summary of the dataset

```
# Viewing the statistical summary of the dataset
# ---
#
str(airquality)
```

```
## 'data.frame':    153 obs. of  6 variables:
##  $ Ozone  : num  41 36 12 18 42.1 ...
##  $ Solar.R: num  190 118 149 313 186 ...
##  $ Wind   : num  7.4 8 12.6 11.5 14.3 14.9 8.6 13.8 20.1 8.6 ...
##  $ Temp   : int  67 72 74 62 56 66 65 59 61 69 ...
##  $ Month  : int  5 5 5 5 5 5 5 5 5 5 ...
##  $ Day    : int  1 2 3 4 5 6 7 8 9 10 ...
```

```
summary(airquality)
```

```
##      Ozone           Solar.R           Wind             Temp
##  Min.   :  1.00   Min.   :  7.0   Min.   : 1.700   Min.   :56.00
##  1st Qu.: 21.00   1st Qu.:120.0   1st Qu.: 7.400   1st Qu.:72.00
##  Median : 42.13   Median :194.0   Median : 9.700   Median :79.00
##  Mean   : 42.13   Mean   :185.9   Mean   : 9.958   Mean   :77.88
##  3rd Qu.: 46.00   3rd Qu.:256.0   3rd Qu.:11.500   3rd Qu.:85.00
##  Max.   :168.00   Max.   :334.0   Max.   :20.700   Max.   :97.00
##      Month            Day
##  Min.   :5.000   Min.   : 1.0
##  1st Qu.:6.000   1st Qu.: 8.0
##  Median :7.000   Median :16.0
##  Mean   :6.993   Mean   :15.8
##  3rd Qu.:8.000   3rd Qu.:23.0
##  Max.   :9.000   Max.   :31.0
```

```
# Pre processing the dataset
# Since clustering is a type of Unsupervised Learning,
# we would not require Class Label(output) during execution of our algorithm.
# We would then normalize the attributes between 0 and 1 using our own function.
# ---
#
airquality.new<- airquality[, c(1, 2, 3, 4)]
airquality.class<- airquality[, "Month"]
head(airquality.new)
```

```
##      Ozone  Solar.R Wind Temp
## 1 41.00000 190.0000  7.4   67
## 2 36.00000 118.0000  8.0   72
## 3 12.00000 149.0000 12.6   74
## 4 18.00000 313.0000 11.5   62
## 5 42.12931 185.9315 14.3   56
## 6 28.00000 185.9315 14.9   66
```

```
# Previewing the class column
# ---
#
head(airquality.class)
```

```
## [1] 5 5 5 5 5 5
```

Normalizing the our dataset

```r
# Normalizing the dataset so that no particular attribute
# has more impact on clustering algorithm than others.
# ---
#
normalize <- function(x){
 return ((x-min(x)) / (max(x)-min(x)))
}
airquality.new$Ozone<- normalize(airquality.new$Ozone)
airquality.new$Solar.R<- normalize(airquality.new$Solar.R)
airquality.new$Wind<- normalize(airquality.new$Wind)
airquality.new$Temp<- normalize(airquality.new$Temp)
head(airquality.new)
```

```
##        Ozone   Solar.R      Wind      Temp
## 1 0.23952096 0.5596330 0.3000000 0.2682927
## 2 0.20958084 0.3394495 0.3315789 0.3902439
## 3 0.06586826 0.4342508 0.5736842 0.4390244
## 4 0.10179641 0.9357798 0.5157895 0.1463415
## 5 0.24628330 0.5471912 0.6631579 0.0000000
## 6 0.16167665 0.5471912 0.6947368 0.2439024
```

```r
# Applying the K-means clustering algorithm with no. of centroids (k)=3
# ---
#
result<- kmeans(airquality.new,5)
# Previewing the no. of records in each cluster
#
result$size
```

```
## [1] 38 42 25 32 16
```

```r
# Getting the value of cluster center datapoint value(3 centers for k=5)
# ---
#
result$centers
```

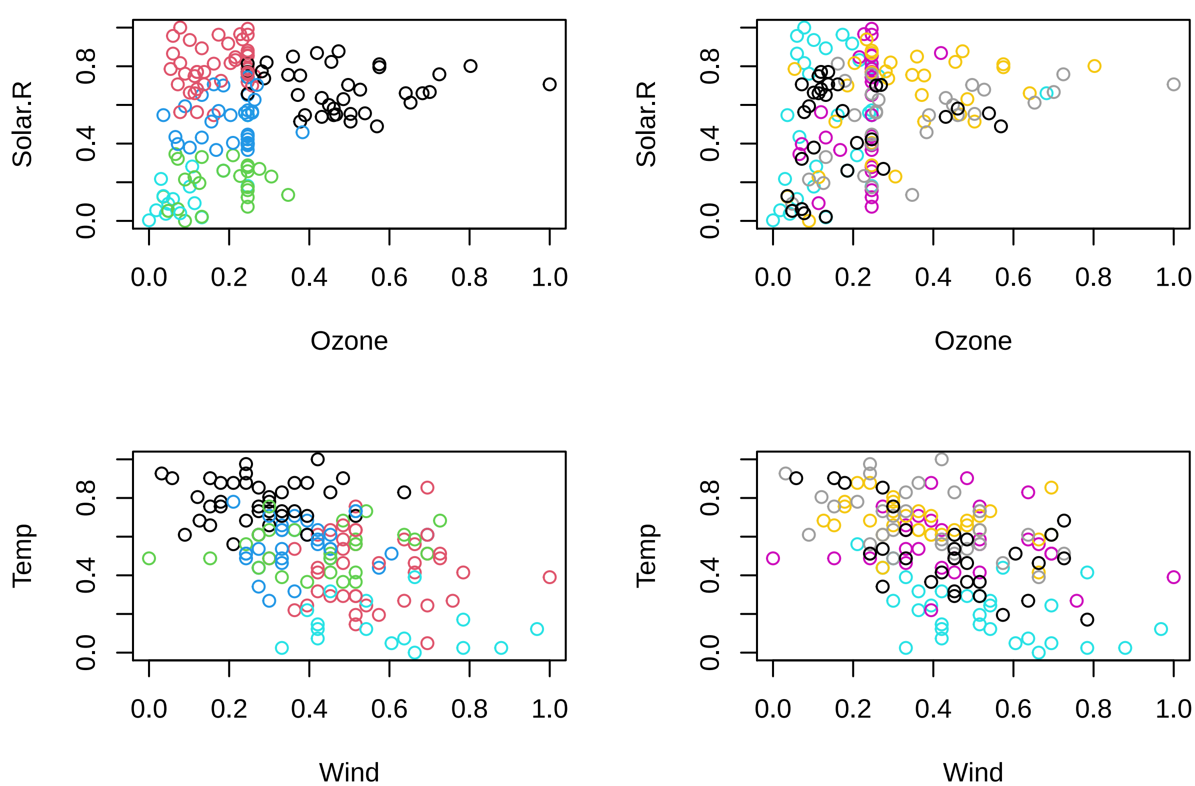
```
##       Ozone   Solar.R      Wind      Temp
## 1 0.4761294 0.6842800 0.2768698 0.7849807
## 2 0.1687229 0.8018008 0.5636591 0.4378630
## 3 0.1719533 0.1904587 0.4233684 0.5346341
## 4 0.2035089 0.5154686 0.3814145 0.5602134
## 5 0.1056841 0.1740946 0.5944079 0.1341463
```

```r
# Getting the cluster vector that shows the cluster where each record falls
# ---
#
result$cluster
```

```
##   [1] 4 3 4 2 5 2 2 5 5 4 4 2 2 2 5 2 2 5 2 5 5 2 5 5 5 2 5 5 2 1 2 2 2 2 4 4 2
##  [38] 4 1 1 2 1 1 4 2 2 2 2 5 3 4 4 3 3 4 4 4 3 3 3 4 1 1 4 3 1 2 1 1 1 1 4 2 4
##  [75] 2 3 1 2 1 1 1 3 2 2 1 1 3 3 1 1 1 1 3 3 3 1 4 1 1 1 1 1 4 4 2 4 3 3 3 3 2
## [112] 4 2 5 2 4 1 1 4 1 1 1 1 1 1 1 1 3 3 2 4 2 2 2 2 2 4 3 3 4 2 3 2 4 2 3 4 5 5
## [149] 4 4 2 4 2
```

```
# The graph shows that we have got 5 clearly distinguishable clusters for Ozone and Solar.R data points
# Let's see how clustering has performed on Wind and Temp attributes.
```

```
# Verifying the results of clustering
# ---
#
par(mfrow = c(2,2), mar = c(5,4,2,2))
# Plotting to see how Sepal.Length and Sepal.Width data points have been distributed in clusters
plot(airquality.new[c(1,2)], col = result$cluster)
# Plotting to see how Sepal.Length and Sepal.Width data points have been distributed
# originally as per "class" attribute in dataset
# ---
#
plot(airquality.new[c(1,2)], col = airquality.class)
# Plotting to see how Petal.Length and Petal.Width data points have been distributed in clusters
# ---
#
plot(airquality.new[c(3,4)], col = result$cluster)
plot(airquality.new[c(3,4)], col = airquality.class)
```




```
# Result of table shows that Cluster 1 corresponds to Virginica,
# Cluster 2 corresponds to Versicolor and Cluster 3 to Setosa.
# ---
#
table(result$cluster, airquality.class)
```

```
##    airquality.class
##      5  6  7  8  9
##   1  1  4 17 12  4
##   2 13  9  6  4 10
##   3  1  6  5  7  6
##   4  4 10  3  7  8
##   5 12  1  0  1  2
```

Challenge 2

```r
# Create a model that clusters the following dataset.
# ---
# Dataset = http://bit.ly/SalaryDatasetClustering
# ---
salary_df <- read.csv(file.choose(), header = T)
# previewing the data
head(salary_df)
```

```
##   Id      EmployeeName                                       JobTitle   BasePay
## 1  1    NATHANIEL FORD GENERAL MANAGER-METROPOLITAN TRANSIT AUTHORITY 167411.18
## 2  2      GARY JIMENEZ                CAPTAIN III (POLICE DEPARTMENT) 155966.02
## 3  3    ALBERT PARDINI                CAPTAIN III (POLICE DEPARTMENT) 212739.13
## 4  4 CHRISTOPHER CHONG           WIRE ROPE CABLE MAINTENANCE MECHANIC   77916.0
## 5  5   PATRICK GARDNER   DEPUTY CHIEF OF DEPARTMENT,(FIRE DEPARTMENT)  134401.6
## 6  6    DAVID SULLIVAN                      ASSISTANT DEPUTY CHIEF II  118602.0
##   OvertimePay  OtherPay Benefits TotalPay TotalPayBenefits Year Notes
## 1         0.0 400184.25          567595.4         567595.4 2011    NA
## 2   245131.88 137811.38          538909.3         538909.3 2011    NA
## 3   106088.18   16452.6          335279.9         335279.9 2011    NA
## 4    56120.71  198306.9          332343.6         332343.6 2011    NA
## 5      9737.0 182234.59          326373.2         326373.2 2011    NA
## 6      8601.0 189082.74          316285.7         316285.7 2011    NA
##          Agency Status
## 1 San Francisco
## 2 San Francisco
## 3 San Francisco
## 4 San Francisco
## 5 San Francisco
## 6 San Francisco
```

Finding the summary and the data types

```r
summary(salary_df)
```

```
##        Id         EmployeeName         JobTitle            BasePay
##  Min.   :     1   Length:148654      Length:148654      Length:148654
##  1st Qu.: 37164   Class :character   Class :character   Class :character
##  Median : 74328   Mode  :character   Mode  :character   Mode  :character
##  Mean   : 74328
##  3rd Qu.:111491
##  Max.   :148654
##  OvertimePay          OtherPay           Benefits            TotalPay
```

```
##  Length:148654      Length:148654      Length:148654      Min.   :  -618.1
##  Class :character   Class :character   Class :character   1st Qu.: 36169.0
##  Mode  :character   Mode  :character   Mode  :character   Median : 71426.6
##                                                           Mean   : 74768.3
##                                                           3rd Qu.:105839.1
##                                                           Max.   :567595.4
##  TotalPayBenefits        Year       Notes            Agency
##  Min.   :  -618.1   Min.   :2011   Mode:logical   Length:148654
##  1st Qu.: 44065.7   1st Qu.:2012   NA's:148654    Class :character
##  Median : 92404.1   Median :2013                  Mode  :character
##  Mean   : 93692.6   Mean   :2013
##  3rd Qu.:132876.5   3rd Qu.:2014
##  Max.   :567595.4   Max.   :2014
##     Status
##  Length:148654
##  Class :character
##  Mode  :character
##
##
##
```

```
str(salary_df)
```

```
## 'data.frame':    148654 obs. of  13 variables:
##  $ Id              : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ EmployeeName    : chr  "NATHANIEL FORD" "GARY JIMENEZ" "ALBERT PARDINI" "CHRISTOPHER CHONG" ...
##  $ JobTitle        : chr  "GENERAL MANAGER-METROPOLITAN TRANSIT AUTHORITY" "CAPTAIN III (POLICE DEPA
##  $ BasePay         : chr  "167411.18" "155966.02" "212739.13" "77916.0" ...
##  $ OvertimePay     : chr  "0.0" "245131.88" "106088.18" "56120.71" ...
##  $ OtherPay        : chr  "400184.25" "137811.38" "16452.6" "198306.9" ...
##  $ Benefits        : chr  "" "" "" "" ...
##  $ TotalPay        : num  567595 538909 335280 332344 326373 ...
##  $ TotalPayBenefits: num  567595 538909 335280 332344 326373 ...
##  $ Year            : int  2011 2011 2011 2011 2011 2011 2011 2011 2011 2011 ...
##  $ Notes           : logi  NA NA NA NA NA NA ...
##  $ Agency          : chr  "San Francisco" "San Francisco" "San Francisco" "San Francisco" ...
##  $ Status          : chr  "" "" "" "" ...
```

Challenge 3

```
# Cluster customers from the given wholesale customer database.
# ---
# Dataset source = https://archive.ics.uci.edu/ml/datasets/Wholesale+customers
# --
customer_db <- read.csv(file.choose(), header = T)
#
head(customer_db)
```

```
##   Channel Region Fresh Milk Grocery Frozen Detergents_Paper Delicassen
## 1       2      3 12669 9656    7561    214             2674       1338
## 2       2      3  7057 9810    9568   1762             3293       1776
```

```
## 3       2      3  6353 8808    7684   2405             3516       7844
## 4       1      3 13265 1196    4221   6404              507       1788
## 5       2      3 22615 5410    7198   3915             1777       5185
## 6       2      3  9413 8259    5126    666             1795       1451
```

Finding the summary and data type

```r
# Viewing the statistical summary of the dataset
# ---
#
str(customer_db)
```

```
## 'data.frame':    440 obs. of  8 variables:
##  $ Channel         : int  2 2 2 1 2 2 2 2 1 2 ...
##  $ Region          : int  3 3 3 3 3 3 3 3 3 3 ...
##  $ Fresh           : int  12669 7057 6353 13265 22615 9413 12126 7579 5963 6006 ...
##  $ Milk            : int  9656 9810 8808 1196 5410 8259 3199 4956 3648 11093 ...
##  $ Grocery         : int  7561 9568 7684 4221 7198 5126 6975 9426 6192 18881 ...
##  $ Frozen          : int  214 1762 2405 6404 3915 666 480 1669 425 1159 ...
##  $ Detergents_Paper: int  2674 3293 3516 507 1777 1795 3140 3321 1716 7425 ...
##  $ Delicassen      : int  1338 1776 7844 1788 5185 1451 545 2566 750 2098 ...
```

```r
summary(customer_db)
```

```
##     Channel          Region          Fresh             Milk
##  Min.   :1.000   Min.   :1.000   Min.   :     3   Min.   :   55
##  1st Qu.:1.000   1st Qu.:2.000   1st Qu.:  3128   1st Qu.: 1533
##  Median :1.000   Median :3.000   Median :  8504   Median : 3627
##  Mean   :1.323   Mean   :2.543   Mean   : 12000   Mean   : 5796
##  3rd Qu.:2.000   3rd Qu.:3.000   3rd Qu.: 16934   3rd Qu.: 7190
##  Max.   :2.000   Max.   :3.000   Max.   :112151   Max.   :73498
##     Grocery          Frozen          Detergents_Paper    Delicassen
##  Min.   :    3   Min.   :   25.0   Min.   :    3.0   Min.   :    3.0
##  1st Qu.: 2153   1st Qu.:  742.2   1st Qu.:  256.8   1st Qu.:  408.2
##  Median : 4756   Median : 1526.0   Median :  816.5   Median :  965.5
##  Mean   : 7951   Mean   : 3071.9   Mean   : 2881.5   Mean   : 1524.9
##  3rd Qu.:10656   3rd Qu.: 3554.2   3rd Qu.: 3922.0   3rd Qu.: 1820.2
##  Max.   :92780   Max.   :60869.0   Max.   :40827.0   Max.   :47943.0
```

Making classes

```r
# Pre processing the dataset
# Since clustering is a type of Unsupervised Learning,
# we would not require Class Label(output) during execution of our algorithm.
# We would then normalize the attributes between 0 and 1 using our own function.
# ---
#
customer.new<- customer_db[, c(1, 3, 4, 5, 6, 7, 8)]
customer.class<- customer_db[, "Region"]
head(customer.new)
```

```
##   Channel Fresh Milk Grocery Frozen Detergents_Paper Delicassen
```

```
## 1       2 12669 9656    7561    214             2674        1338
## 2       2  7057 9810    9568   1762             3293        1776
## 3       2  6353 8808    7684   2405             3516        7844
## 4       1 13265 1196    4221   6404              507        1788
## 5       2 22615 5410    7198   3915             1777        5185
## 6       2  9413 8259    5126    666             1795        1451
```

```
# Previewing the class column
# ---
#
head(customer.class)
```

```
## [1] 3 3 3 3 3 3
```

Normalizing the our dataset

```
# Normalizing the dataset so that no particular attribute
# has more impact on clustering algorithm than others.
# ---
#
normalize <- function(x){
 return ((x-min(x)) / (max(x)-min(x)))
}
customer.new$Channel<- normalize(customer.new$Channel)
customer.new$Fresh<- normalize(customer.new$Fresh)
customer.new$Milk<- normalize(customer.new$Milk)
customer.new$Grocery<- normalize(customer.new$Grocery)
customer.new$Frozen<- normalize(customer.new$Frozen)
customer.new$Detergents_Paper<- normalize(customer.new$Detergents_Paper)
customer.new$Delicassen<- normalize(customer.new$Delicassen)
head(customer.new)
```

```
##   Channel      Fresh       Milk    Grocery      Frozen Detergents_Paper
## 1       1 0.11294004 0.13072723 0.08146416 0.003106305       0.06542720
## 2       1 0.06289903 0.13282409 0.10309667 0.028548419       0.08058985
## 3       1 0.05662161 0.11918086 0.08278992 0.039116429       0.08605232
## 4       0 0.11825445 0.01553586 0.04546385 0.104841891       0.01234568
## 5       1 0.20162642 0.07291369 0.07755155 0.063933995       0.04345483
## 6       1 0.08390698 0.11170568 0.05521843 0.010535139       0.04389575
##   Delicassen
## 1 0.02784731
## 2 0.03698373
## 3 0.16355861
## 4 0.03723404
## 5 0.10809345
## 6 0.03020442
```

```
# Applying the K-means clustering algorithm with no. of centroids (k)=3
# ---
#
result<- kmeans(customer.new, 3)
# Previewing the no. of records in each cluster
#
result$size
```

```
## [1] 142  45 253
```

```r
# Getting the value of cluster center datapoint value(3 centers for k=5)
# ---
#
result$centers
```

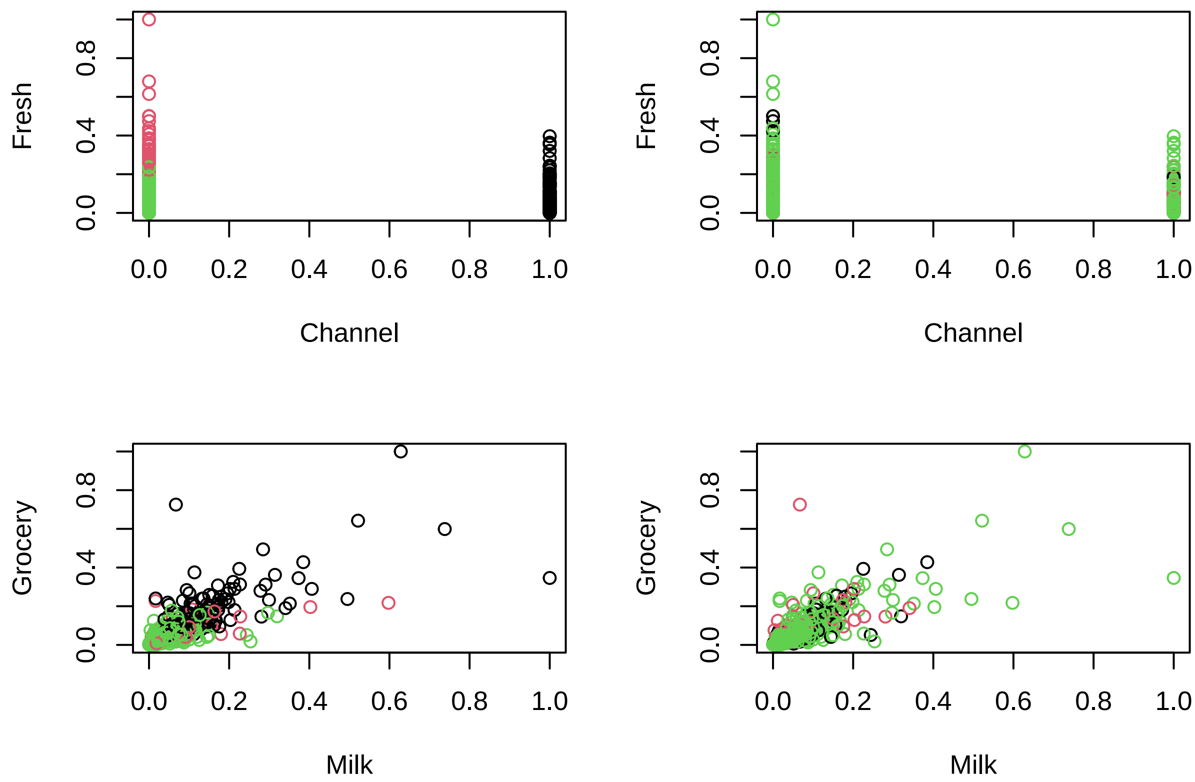
```
##   Channel      Fresh       Milk    Grocery     Frozen Detergents_Paper
## 1       1 0.07937122 0.14516700 0.17590407 0.02675059       0.17799596
## 2       0 0.34053414 0.08313507 0.06944202 0.15170052       0.01956148
## 3       0 0.08093001 0.03968919 0.03791253 0.04509530       0.01924360
##   Delicassen
## 1 0.03651307
## 2 0.07754230
## 3 0.02092363
```

```r
# Getting the cluster vector that shows the cluster where each record falls
# ---
#
result$cluster
```

```
##   [1] 1 1 1 3 1 1 1 1 3 1 1 1 1 1 1 3 1 3 1 3 1 3 2 1 1 1 3 3 1 2 3 3 3 2 3 1 2
##  [38] 1 1 2 2 3 1 1 1 1 1 1 1 1 3 3 1 1 3 3 1 1 3 3 1 1 1 1 3 1 3 1 3 3 3 2 3 1
##  [75] 1 3 3 1 3 3 3 1 1 3 1 1 1 2 3 3 3 3 1 2 1 3 1 3 3 3 1 1 1 2 3 3 1 1 1 1 3
## [112] 1 3 3 3 3 3 3 3 3 3 3 3 1 2 2 3 1 3 2 3 3 3 3 3 3 3 3 3 3 3 2 2 3 3 1 3 3
## [149] 3 2 3 3 3 3 3 1 1 3 1 1 1 3 3 1 1 1 1 3 3 3 1 1 3 1 3 1 2 3 3 3 3 2 3 2 3
## [186] 3 3 3 1 1 3 3 3 1 3 3 2 1 3 3 1 1 2 3 3 1 3 1 3 1 3 1 3 3 1 3 1 3 1 3 3 3
## [223] 3 1 3 3 1 3 3 3 1 3 3 3 3 3 3 3 3 2 2 3 3 3 3 1 3 3 3 3 3 1 3 2 3 2 3 3 2
## [260] 2 3 3 3 3 1 3 1 3 1 3 3 3 3 2 3 3 2 3 3 1 3 1 2 2 2 2 3 3 3 2 3 3 3 1 3 1
## [297] 3 1 1 3 1 1 1 1 1 1 1 1 3 3 1 3 2 1 3 3 1 3 3 3 1 3 3 3 3 3 2 3 3 3 3 3 1 3
## [334] 1 1 1 3 3 3 3 1 1 3 1 3 3 1 1 3 1 3 1 3 1 3 3 3 1 3 3 3 3 3 3 3 1 3 3 3 3
## [371] 1 3 3 1 3 3 1 2 3 1 2 3 2 3 3 3 3 3 3 3 3 3 3 2 3 3 1 3 3 3 3 2 3 3 3 3 2
## [408] 1 1 3 3 3 3 3 3 1 1 3 1 3 3 1 3 1 1 3 3 2 3 3 3 3 3 3 3 2 2 1 3 3
```

```r
# The graph shows that we have got 5 clearly distinguishable clusters for Ozone and Solar.R data points
# Let's see how clustering has performed on Wind and Temp attributes.
```

```r
# Verifying the results of clustering
# ---
#
par(mfrow = c(2,2), mar = c(5,4,2,2))
# Plotting to see how Sepal.Length and Sepal.Width data points have been distributed in clusters
plot(customer.new[c(1,2)], col = result$cluster)
# Plotting to see how Sepal.Length and Sepal.Width data points have been distributed
# originally as per "class" attribute in dataset
# ---
#
plot(customer.new[c(1,2)], col = customer.class)
# Plotting to see how Petal.Length and Petal.Width data points have been distributed in clusters
# ---
#
plot(customer.new[c(3,4)], col = result$cluster)
plot(customer.new[c(3,4)], col = customer.class)
```




```r
# Result of table shows that Cluster 1 corresponds to Virginica,
# Cluster 2 corresponds to Versicolor and Cluster 3 to Setosa.
# ---
#
table(result$cluster, customer.class)
```

```
##    customer.class
##       1   2   3
##   1  18  19 105
##   2   8   2  35
##   3  51  26 176
```