WQD 7009 Big Data Applications and Data Analytics

Assignment 3

K-means and Hierarchical Clustering Algorithms Comparison

Ng Kang Wei

WQD170068

Contents

Introduction

The dataset used in this clustering algorithm comparison is the Big Mart Sales dataset. The dataset is available at Kaggle: https://www.kaggle.com/brijbhushannanda1979/bigmart-sales-data. Each of the items or products has different sales in different outlets and each of the outlets has different tier. We would like to figure out a way to divide the products or sales data into different groups. Clustering algorithm would be a way to do that but there are several clustering algorithms. Each of the clustering algorithms has their own perks and advantages. The clustering algorithms would be applied in this case are K-means and hierarchical clustering algorithms. We would like to compare these 2 clustering algorithms in the performance of grouping the Big Mart Sales data.

Problem Statement

The Big Mart Sales dataset contains 12 features or attributes for 8523 records. This research aims to divide the datasets into several distinct categories that would provide insights to the owner of Big Mart to increase their sales data. The problems faced in this study are:

1. Out of the 12 features of the sales data, which of the features are most prominent and is best use in the clustering algorithms?
2. From the 2 clustering algorithms chosen, k-means and hierarchical clustering algorithms, which one of them would perform better?

Objectives

The objectives of this research are

1. Find the most prominent or important features to group the data in the dataset.
2. Compare the 2 clustering algorithms, k-means and hierarchical clustering algorithms.

Data Description

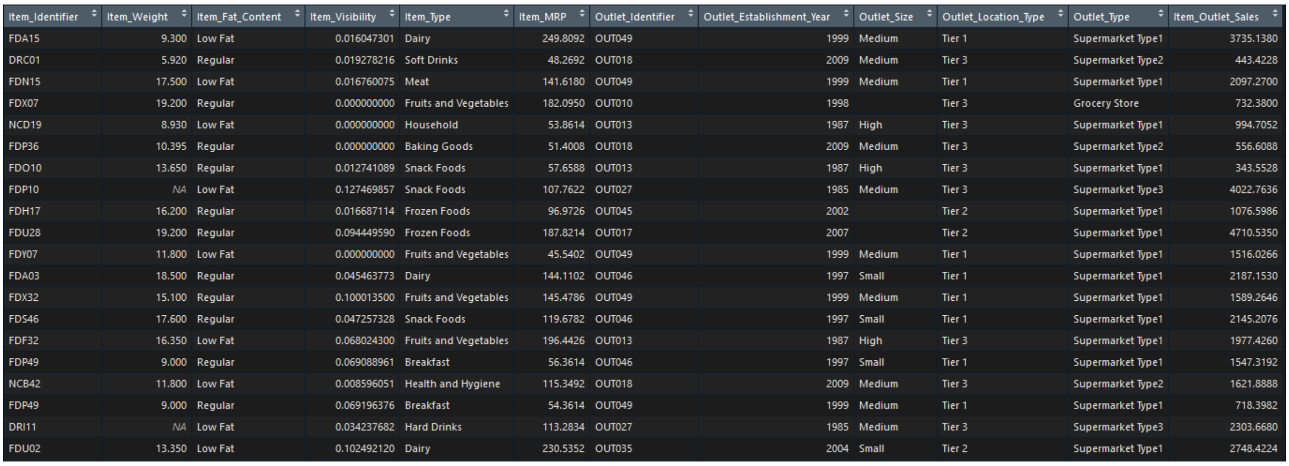
The dataset selected is the Big Mart Sales dataset. It is available for download from Kaggle.com. The Big Mart Sales data is the sales data of 1559 products across 10 stores in different cities in 2013. The dataset is divided into training set and testing set. The training set has the dimension of 8523 rows and 12 columns while the testing set has 5681 rows and 11 columns. The testing set has 1 less column than the training set because it lacks the target variable of the item outlet sales. This is the target variable for supervised learning algorithm to predict the sales of an item. The clustering algorithms which is an unsupervised learning algorithm would only focus on the training data.

Structure of the data

Column name	**Type**	**Remark**
Item Identifier	String	A unique string to identify each of the items
Item Weight	Numeric	The weight of the product
Item Fat Content	Ordinal	The fat content of the product is divided into several categories: low fat, regular
Item Visibility	Numeric	A numerical value to define how visible is the placement of the product
Item Type	Nominal	The categorical type of the product
Item MRP	Numeric	The value of the material requirement planning (MRP) needed for the product
Outlet Identifier	String	A unique string to identify each of the outlet
Outlet Establishment Year	Numeric	The year the outlet is established
Outlet Size	Ordinal	The size of the outlet in the rank of small, medium, and high
Outlet Location Type	Nominal	The location of the outlets in the categories of tier 1, tier 2 and tier 3

Outlet Type	Nominal	The type of the outlet in the categories of grocery store, supermarket type 1, supermarket type 2, and supermarket type 3
Item Outlet Sales	Numeric	The sales value of the item in an outlet

Sample Data



Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_Location_Type	Outlet_Type	Item_Outlet_Sales
FDA15	9.300	Low Fat	0.016047301	Dairy	249.8092	OUT049	1999	Medium	Tier 1	Supermarket Type1	3735.1380
DRC01	5.920	Regular	0.019278216	Soft Drinks	48.2692	OUT018	2009	Medium	Tier 3	Supermarket Type2	443.4228
FDN15	17.500	Low Fat	0.016760075	Meat	141.6180	OUT049	1999	Medium	Tier 1	Supermarket Type1	2097.2700
FDX07	19.200	Regular	0.000000000	Fruits and Vegetables	182.0950	OUT010	1998		Tier 3	Grocery Store	732.3800
NCD19	8.930	Low Fat	0.000000000	Household	53.8614	OUT013	1987	High	Tier 3	Supermarket Type1	994.7052
FDP36	10.395	Regular	0.000000000	Baking Goods	51.4008	OUT018	2009	Medium	Tier 3	Supermarket Type2	556.6088
FDO10	13.650	Regular	0.012741089	Snack Foods	57.6588	OUT013	1987	High	Tier 3	Supermarket Type1	343.5528
FDP10	NA	Low Fat	0.127469857	Snack Foods	107.7622	OUT027	1985	Medium	Tier 3	Supermarket Type3	4022.7636
FDH17	16.200	Regular	0.016687114	Frozen Foods	96.9726	OUT045	2002		Tier 2	Supermarket Type1	1076.5986
FDU28	19.200	Regular	0.094449590	Frozen Foods	187.8214	OUT017	2007		Tier 2	Supermarket Type1	4710.5350
FDY07	11.800	Low Fat	0.000000000	Fruits and Vegetables	45.5402	OUT049	1999	Medium	Tier 1	Supermarket Type1	1516.0266
FDA03	18.500	Regular	0.045463773	Dairy	144.1102	OUT046	1997	Small	Tier 1	Supermarket Type1	2187.1530
FDX32	15.100	Regular	0.100013500	Fruits and Vegetables	145.4786	OUT049	1999	Medium	Tier 1	Supermarket Type1	1589.2646
FDS46	17.600	Regular	0.047257328	Snack Foods	119.6782	OUT046	1997	Small	Tier 1	Supermarket Type1	2145.2076
FDF32	16.350	Low Fat	0.068024300	Fruits and Vegetables	196.4426	OUT013	1987	High	Tier 3	Supermarket Type1	1977.4260
FDP49	9.000	Regular	0.069088961	Breakfast	56.3614	OUT046	1997	Small	Tier 1	Supermarket Type1	1547.3192
NCB42	11.800	Low Fat	0.008596051	Health and Hygiene	115.3492	OUT018	2009	Medium	Tier 3	Supermarket Type2	1621.8888
FDP49	9.000	Regular	0.069196376	Breakfast	54.3614	OUT049	1999	Medium	Tier 1	Supermarket Type1	718.3982
DRI11	NA	Low Fat	0.034237682	Hard Drinks	113.2834	OUT027	1985	Medium	Tier 3	Supermarket Type3	2303.6680
FDU02	13.350	Low Fat	0.102492120	Dairy	230.5352	OUT035	2004	Small	Tier 2	Supermarket Type1	2748.4224

Data Preprocessing

Load the data

```
# Loading the data
trainingPath <- file.path('./bigMartTrain.csv')
martData <- read.csv(trainingPath)
dim(martData)
head(martData)
str(martData)
```

The Big Mart Sales dataset is divided into training set and testing set. Only the training set is loaded into the environment. Supervised learning methods need the dataset to be divided into training and testing, unsupervised learning does not. Clustering algorithm is an unsupervised learning method therefore does not need the testing set. The dimension of the training data is 8523 rows and 12 columns. The data of the dataset are of a mixed. Some of the columns have numerical values while some has categorical values. The details of the structure are listed in the section above.

Data Cleansing

After the data is loaded, the next step is to cleanse the data. The dataset would contain some missing values, empty values and some values that is not sensible. This data cleansing phase is to remove or impute a value for these missing values.

```
############### Data Preprocessing ###################s
library(dplyr)

# Checking the NA values
sum(is.na(martData))
View(martData[!complete.cases(martData),])
sum(is.na(martData$Item_Weight))

# Impute the NAs with the median
martData$Item_Weight[is.na(martData$Item_Weight)]  <-  median(martData$Item_Weight,
na.rm=T)

# No more NA after inferring the mean to the NAs
sum(is.na(martData))

# Some of the item visibility has the value of 0 which indicate some missing data
sum(martData$Item_Visibility == 0)

# Impute them with median
martData$Item_Visibility <- ifelse(martData$Item_Visibility == 0,
median(martData$Item_Visibility),
                          martData$Item_Visibility)

# Cleaning the levels of the factor
sum(martData$Item_Fat_Content=='LF')
martData$Item_Fat_Content[martData$Item_Fat_Content=='LF'] <- 'Low Fat'
sum(martData$Item_Fat_Content=='low fat')
martData$Item_Fat_Content[martData$Item_Fat_Content=='low fat'] <- 'Low Fat'
sum(martData$Item_Fat_Content=='reg')
martData$Item_Fat_Content[martData$Item_Fat_Content=='reg'] <- 'Regular'
martData$Item_Fat_Content <- factor(martData$Item_Fat_Content)
levels(martData$Item_Fat_Content)

# Cleaning the levels of the factor Outlet_Size
levels(martData$Outlet_Size) <- c('Unknown', 'High', 'Medium', 'Small')
```

In this cleansing phase, the dplyr library is needed. First, check the dataset for NAs or missing values. For the purpose of easier to pick out the columns with missing values, the incomplete records are viewed in a tabular form. From the table, it is found that most of the missing values are in the column “Item_Weight”. There are 1463 rows of missing values, all these missing values are contributed by the column “Item_Weight”. To handle the missing values, we can remove the rows that contain the missing values or impute a value to all the missing values. 1463 rows are a significant part of the dataset, to omit that much rows would cause the result to be inaccurate. Thus, we chose to impute a value to the missing values. The median of “Item_Weight” is calculated and all the missing values are replaced with the median. Now there are no missing values or NAs in the dataset.

However, there are some 0 values in the column “Item_Visibility”. A 0 value means the item is not visible to the customers at all, that is not possible, since the items have been sold. The solution is imputation as well. A median of the “Item_Visibility” is imputed to all the 0 values of “Item_Visibility”.

At this point, there are no more NAs values in the dataset. Even without NAs, there are still some further processing need to be done. The columns in string are in factors format and a factor have redundant levels. The processing is to remove the redundancy in the levels. The levels in “Item_Fat_Content” are “LF”, “low fat”, “Low Fat”, “Reg”, and “Regular”. There is a redundancy because “LF”, “low fat”, “Low Fat” are referring to the same thing and “Reg” and “Regular” are referring to the same fat content as well. Thus, we unify all these redundant values into 2 values “Low Fat” and “Regular”.

Beside “Item_Fat_Content”, there is another column that has insensible value in it, which is the “Outlet_Size” column. It has some empty values, but these empty values are not NAs. They are a level of the factor. An empty value is not ideal for further process or analysis on the data, so a new set of levels without empty value is assigned to the factor.

Now, the dataset is cleansed. The next part is selecting the data for clustering. Different columns or types of data selected could affect the outcome of the clustering algorithms.

Data Selection

Since the focus of this assignment is on clustering algorithm, clustering algorithm is an unsupervised learning method, it would not need the response variable, “Item_Outlet_Sales” and the identifiers, “Item_Identifier”, “Outlet_Identifier”. These columns should be removed from the dataset. The column “Outlet_Establisment_Year” is not a feature that can affect the clustering result thus it is removed as well.

Data Selection Without PCA

```
################# Data Selection #######################
# Clustering is unsupervised so remove the response variable and the identifiers (which are
not variables)
# Just keep the variables in numeric form
clusData <- martData %>% select(Item_Weight, Item_Visibility, Item_MRP)
```

For clustering algorithms without PCA, only the columns with numerical values are selected. Those columns are Item_Weight, Item_Visibility, and Item_MRP. The clustering algorithms calculate the similarity or dissimilarity between the objects with Euclidean distance. Categorical values do not have a natural origin, Euclidean distance does not work well with categorical values.

Data Selection Without PCA (Hierarchical clustering with Categorical values)

```
############ Hierarchical Clustering with Categorical Values ############
# Includes some categorical values in the data
hierData <- martData %>% select(Item_Fat_Content, Item_Type, Outlet_Type, Outlet_Size)
```

K-means algorithm is dependent on Euclidean distance to check the similarity between the objects. On the other hand, hierarchical clustering can use some other distance metric other than Euclidean distance metric. One the distance metric can be applied by hierarchical clustering is Gower’s distance metric. Gower’s distance metric is suitable for categorical values. Thus, only categorical values from the dataset are selected and Gower’s distance metric would be applied.

Data Selection With PCA

```
#################### Data Selection ############################
# Clustering is unsupervised so remove the response variable and the identifiers (which are
not variables)
clusData <- martData %>% select(Item_Weight, Item_Fat_Content, Item_Type,
Item_Visibility, Item_MRP, Outlet_Size, Outlet_Type, Outlet_Location_Type)
```

PCA would convert all the categorical values into numerical values therefore all the features are included in the data. PCA would also select the most important features of the data for clustering.

After data selection, the clustering algorithms can be applied.

Clustering Algorithms Without PCA

K-means

```
###################### K-means ####################
# Elbow method to detect the best number of clusters for K-means
set.seed(123)
vec <- vector()
for (i in 1:10) {
  vec[i] = sum(kmeans(clusData, i)$withinss)
}

plot(x = 1:10, y = vec, type = 'b', main = 'The Elbow Method', xlab = 'Number of Clusters',
ylab = 'WCSS')

# From the chart of the elbow method, the best number of cluster or k value is 4
# Fitting kmeans to the dataset
library(cluster)

set.seed(123)
kmeans <- kmeans(x = clusData, centers = 4)
ykmeans <- kmeans$cluster

# Cluster membership
table(ykmeans)

# Visualizing the clusters
clusplot(clusData, ykmeans, lines = 0, shade = T, color = T, plotchar = F, span = T,
        main = 'K-means clustering of Big Mart Sales data', xlab = 'X', ylab = 'Y')
```

The code above performed the K-means clustering algorithm on the Big Mart Sales data. Euclidean distance is used by K-means to calculate the similarity or dissimilarity of the objects. Euclidean distance is not suitable to be used on categorical values thus the data selected contains only the features with numerical values.

In K-means clustering algorithm, the number of clusters or the value of K, need to be pre-determined. The value of K can be based on heuristics or experience, but it may not be a good value. Another method would be the Elbow method. The Elbow method try the data with the number of clusters from 1 to 10 and check which value of K is the best suited for the dataset.

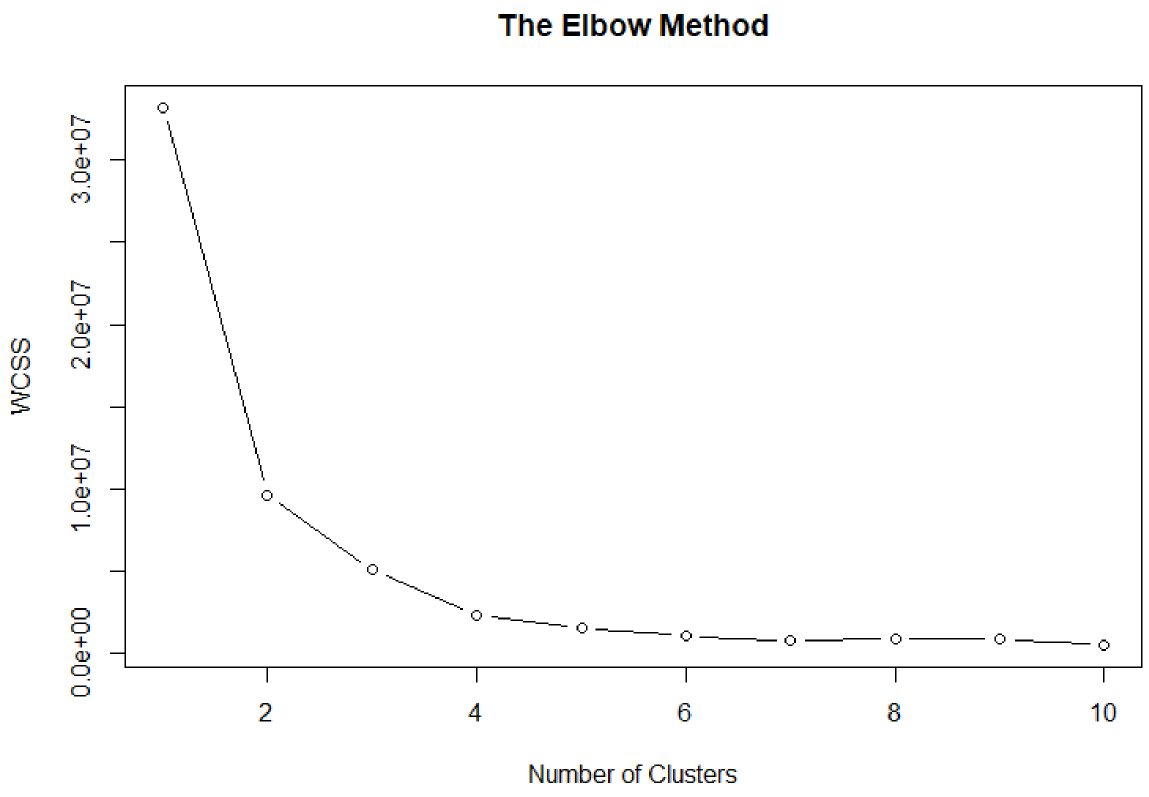


Figure 1 The Elbow method

From the graph shown above, plotted from the Elbow method, the best number of clusters is 4. Further increment of the clusters after 4, have small and insignificant improvements.

After determining the optimum number of clusters, K-means algorithm can be applied. The dataset based on the features selected are divided into 4 clusters. The membership of the clusters is as follow:

Cluster	1	2	3	4
Number of objects	1429	3001	1547	2546

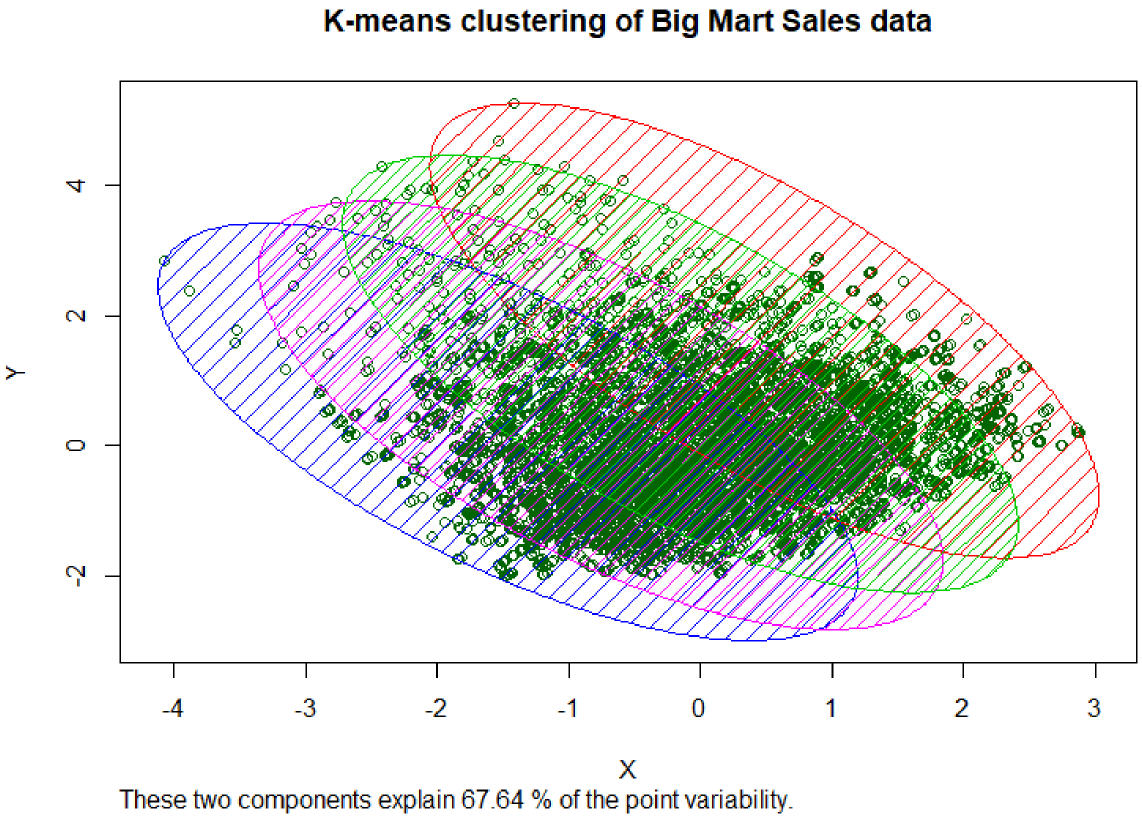


Figure 2 K-means clustering without PCA

Hierarchical Clustering with Euclidean distance

```
################## Hierarchical clustering ##################
library(cluster)

# Using Euclidean distance on the data with numerical values
hc <- hclust(d = dist(clusData, method = 'euclidean'), method = 'ward.D')
plot(hc, main = 'Dendrogram with Euclidean distance', xlab = 'Items', ylab = 'Euclidean
distances')

# Get the number of cluster based on the dendogram
rect.hclust(hc, k = 4, border = "red")
y_hc = cutree(hc, 4)

# Cluster membership
table(y_hc)

clusplot(clusData, y_hc, lines=0, shade=TRUE, color=TRUE, plotchar=FALSE, span=TRUE,
     main='Hierarchical clustering of Big Mart Sales data', xlab='X', ylab='Y')
```

Hierarchical clustering can handle both continuous variable and categorical variables. However, the distance metric used above is Euclidean distance, the selected features are the features with numerical values only. After the similarity or dissimilarity between the objects are calculated by Euclidean distance, a dendogram is plotted. Dendogram is a tree like chart, it shows how the dataset are divided by the variables and into separate clusters or groups. The dendogram generated is shown below.

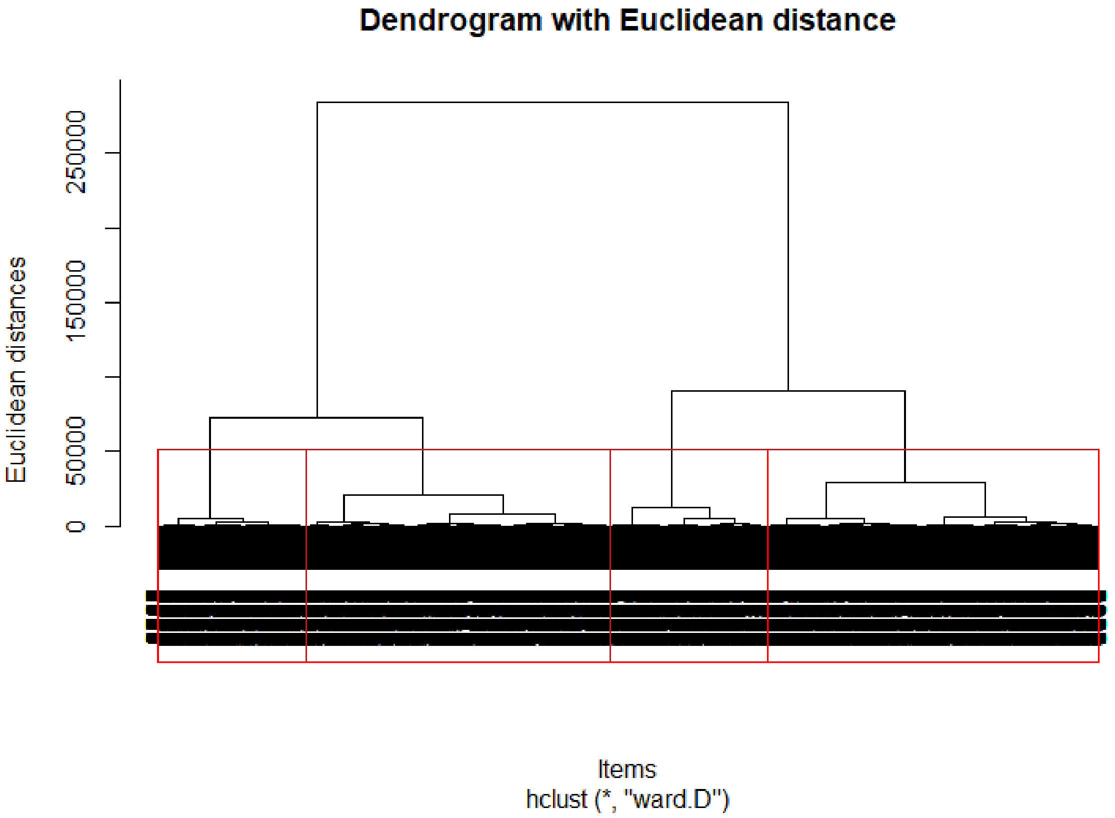


Figure 3 Dendogram without PCA, Euclidean distance

The number of clusters can be determined with the dendogram. The branches signify different clusters. From the dendogram it is decided that the dataset can be divided into 4 major clusters. The visualization of the datasets in 4 clusters is as follows.

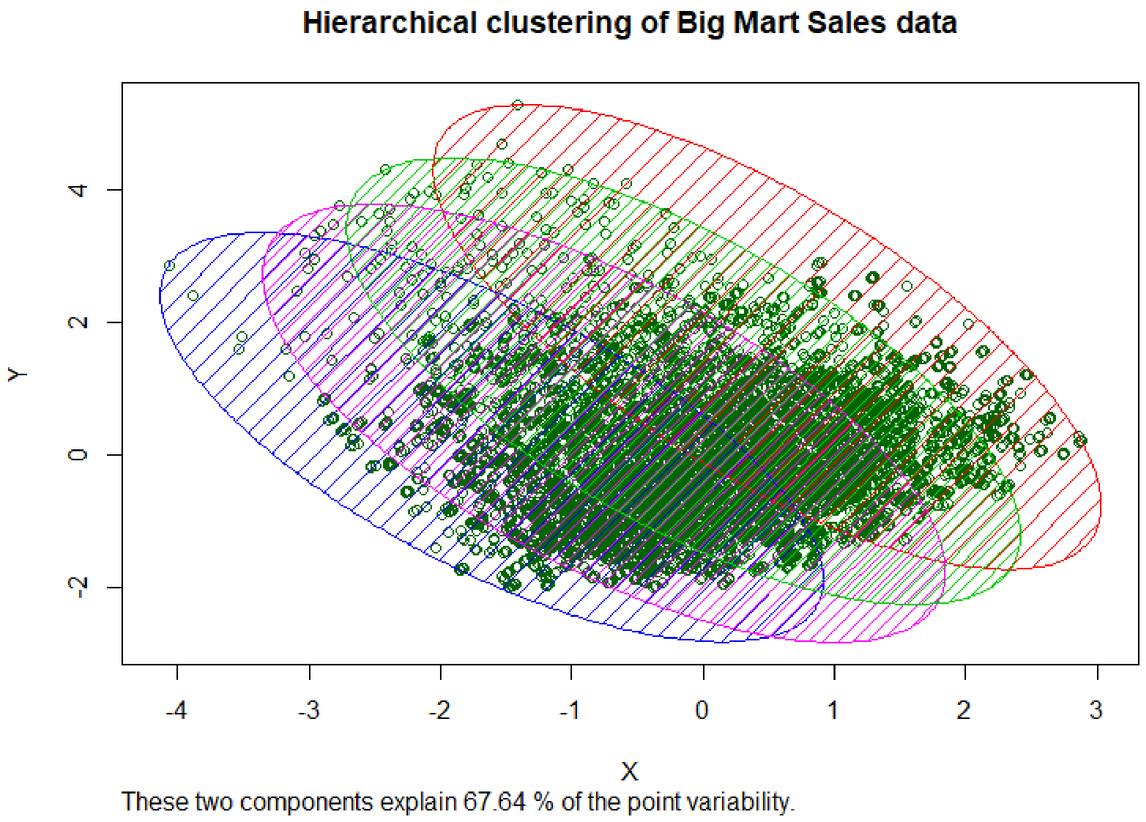


Figure 4 Hierarchical Clustering Without PCA, Euclidean distance (4 clusters)

The cluster membership with Hierarchical clustering without PCA, with Euclidean distance is as follows:

Cluster	1	2	3	4
Number of objects	1429	1341	3002	2751

Hierarchical Clustering with Gower Distance

```
########### Hierarchical Clustering with Categorical Values ###########
# Includes some categorical values in the data
hierData <- martData %>% select(Item_Fat_Content, Item_Type, Outlet_Type, Outlet_Size)

# Use gower distance instead of euclidean to calculate the similarity
gowerDist <- daisy(hierData, metric = 'gower')

# Aggloromerative clustering Dendogram
hc <- hclust(gowerDist, method = 'complete')
plot(hc, main = 'Agglomerative, complete linkage, Gower distance', xlab = 'Items', ylab =
'Gower distance')

# Get the number of clusters based on the dendogram
rect.hclust(hc, k = 6, border = "red")
y_hc = cutree(hc, 6)

# Cluster membership
table(y_hc)

# Visualising the clusters
clusplot(hierData, y_hc, lines = 0, shade = TRUE, color = TRUE, plotchar = FALSE, span =
TRUE,main = "Hierarchical clustering with Gower's distance metric", xlab = 'X', ylab = 'Y')
```

It is mentioned that hierarchical clustering method can handle both continuous values and categorical values. The previous hierarchical clustering employed Euclidean distance to calculate the similarity matrix thus only numerical values are used. In the hierarchical clustering algorithm above, Gower's distance metric is applied. Gower's distance metric is suitable for calculating the similarity matrix of categorical values. Only features with categorical values are selected for this section. There are 2 method for hierarchical clustering to construct the dendogram or group the data, divisive or agglomerative. The method used above is agglomerative and using complete linkage of the distance.

Similar to the hierarchical clustering algorithm before, a dendogram is constructed to determine the number of clusters. In comparison with the dendogram of using Euclidean distance, this dendogram has more branches which means more clusters.

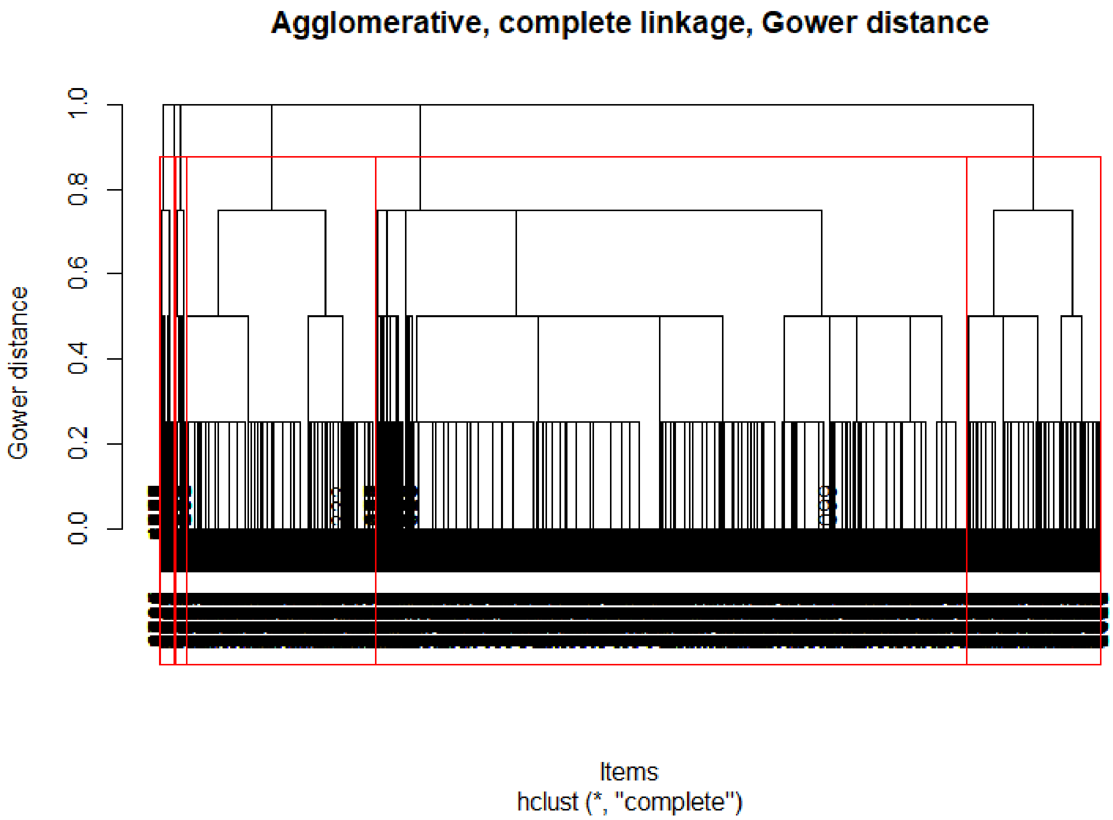


Figure 5 Dendogram without PCA, Gower's distance

From the dendogram shown above, the data can be grouped into 6 clusters. The membership of each of the clusters are:

Clusters	1	2	3	4	5	6
Number of objects	5349	1220	1717	99	13	125

The visualization of the 6 clusters is shown below.

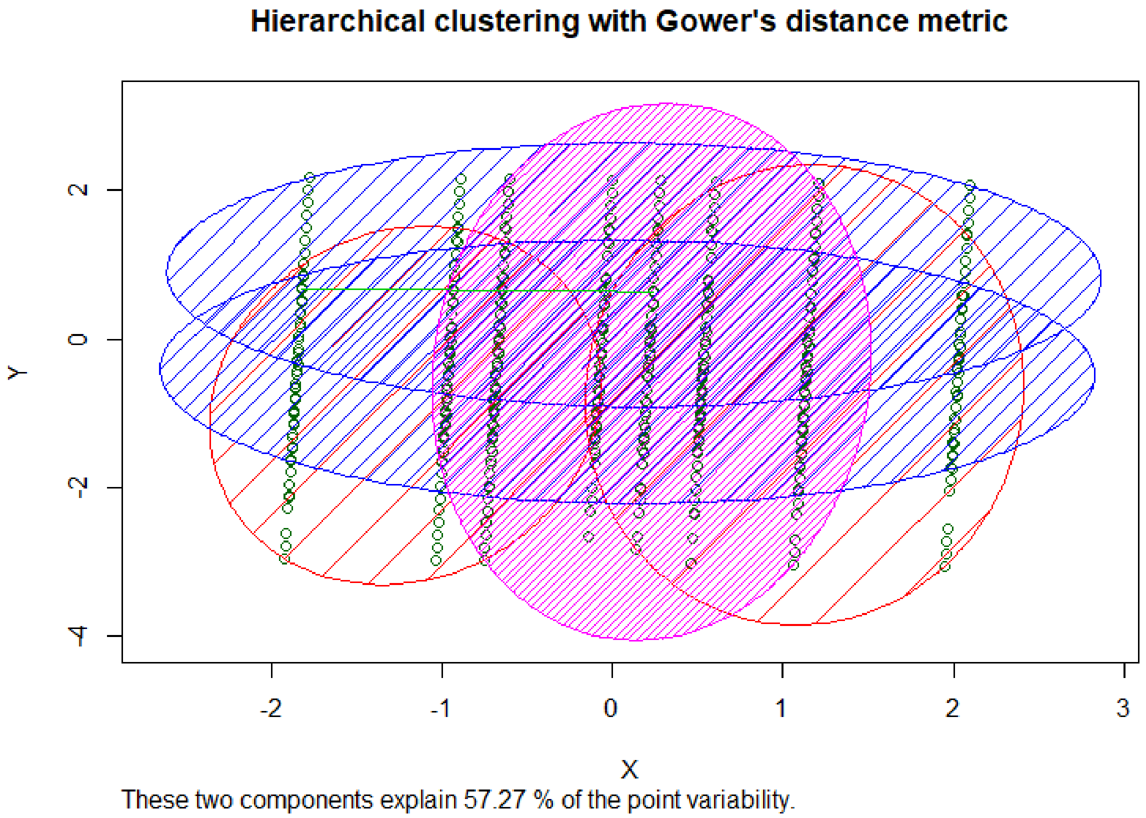


Figure 6 Hierarchical Clustering Without PCA, with Gower distance

Clustering Algorithms With PCA

Principal Component Analysis

There is a total of 10 features in the dataset. Some of the features would be more important others and can describe the data more than the others. We should only use those important features in our clustering algorithms, so the complexity of the data can be reduced. From the previous clustering algorithms, the features are chosen based on heuristics. There is a more logical method, that is principal component analysis (PCA). With PCA, the important features of the dataset can be found, and only these features are applied in the clustering algorithms.

PCA would need to be applied after the data is cleansed and before the application of the clustering algorithms.

```
############ Principal Component Analysis (PCA) #############
## Convert the categorical variables into continuous variables
library(dummies)
dummyDf <- dummy.data.frame(clusData, names = c('Item_Fat_Content', 'Item_Type',
'Outlet_Establishmen_Year', 'Outlet_Size', 'Outlet_Type', 'Outlet_Location_Type'))

# All the data is in numeric form
str(dummyDf)
impFeatures <- prcomp(dummyDf, scale. = T)
names(impFeatures)

# Center and Scale = means and std of the variables
impFeatures$center
impFeatures$scale

# Rotation = principal component loading, most important features
impFeatures$rotation

head(impFeatures$rotation)
dim(impFeatures$rotation)

dim(impFeatures$x)
biplot(impFeatures, scale=0)

# Compute the standard deviation for each of the component
std_dev <- impFeatures$sdev

variance <- std_dev^2

# Check the variance for the first 10 components
head(variance, 10)

# Proportion of variance explained
# The higher the percentage the more important the feature
propVariance <- variance/sum(variance)
propVariance[1:20]

# How many of these feature to select?
# Scree plot
plot(propVariance, xlab = 'Principal Component', ylab = 'Proportion of Variance Explained',
type = 'b', main = 'Scree Plot of proportion of variance')

# Confirmation check with a cumulative variance plot
plot(cumsum(propVariance), xlab = 'Principal Component', ylab = 'Cumulative Proportion of
Variance Explained', type = 'b', main = 'Cumulative Proportion of Variance')
train2 <- data.frame(impFeatures$x)
train2 <- train2[,1:25]
```

First, the library dummies is loaded and all the categorical values are converted into numerical values. This is because PCA only works on numerical values. After the conversion, PCA is performed on the data and the data is scaled or normalized so the standard deviation is equal to 1. After the categorical values conversion, the number of features increase to 32.

The *center* and *scale* output of the PCA function, are the means and standard deviation of the variables that are used to perform PCA. The *rotation* output is the output we are interested it. Each of the columns shows the principal component loading vectors.

The matrix x of the output stored all the principal component scores vector. It ranked each of the features or variables in importance. However, more work needs to be done before we can decide how many of these important features to select.

The proportion of variance is calculated to check how many of the features are needed to describe most of the data. From the proportion of variance calculated, a scree plot can be plotted to check how many of those important variables are needed.

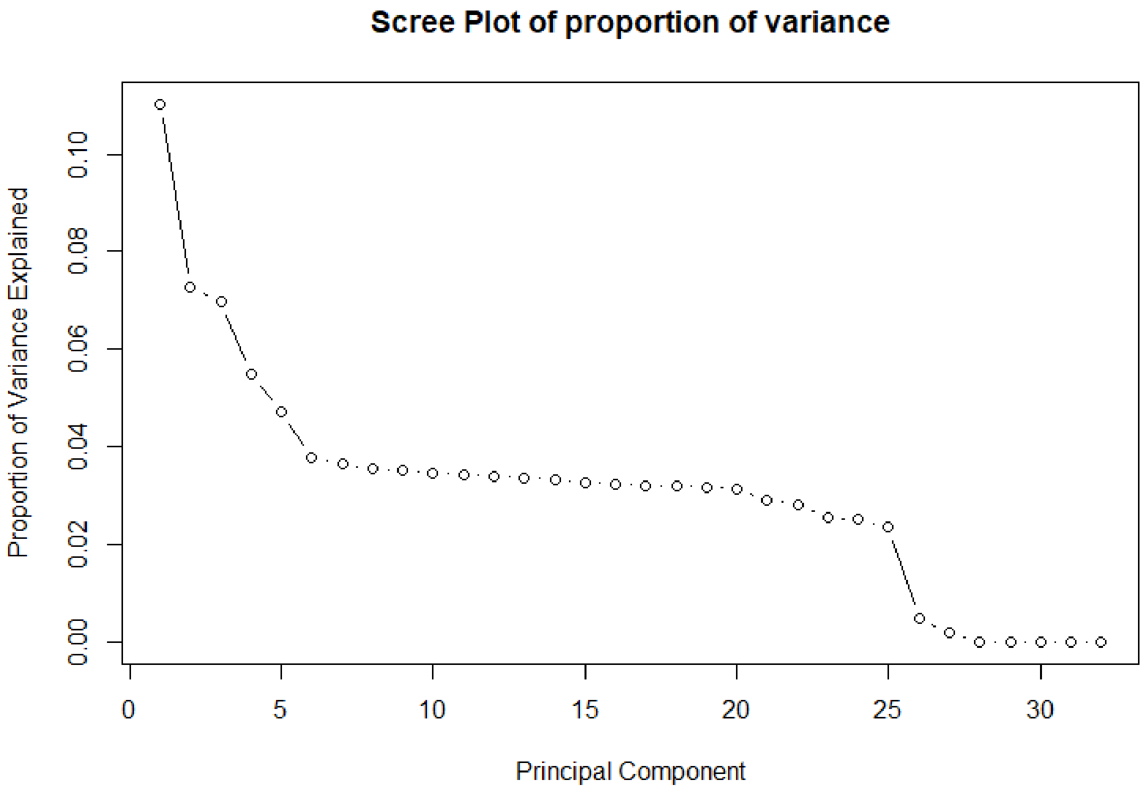


Figure 7 Scree plot of proportion of variance

From the scree plot, it is shown that around 25 components explain about 98% of the variance in the data. This shows that with PCA we have reduce the number of features from 32 to 25. To do a sanity check upon the scree plot, a cumulative scree plot is constructed.

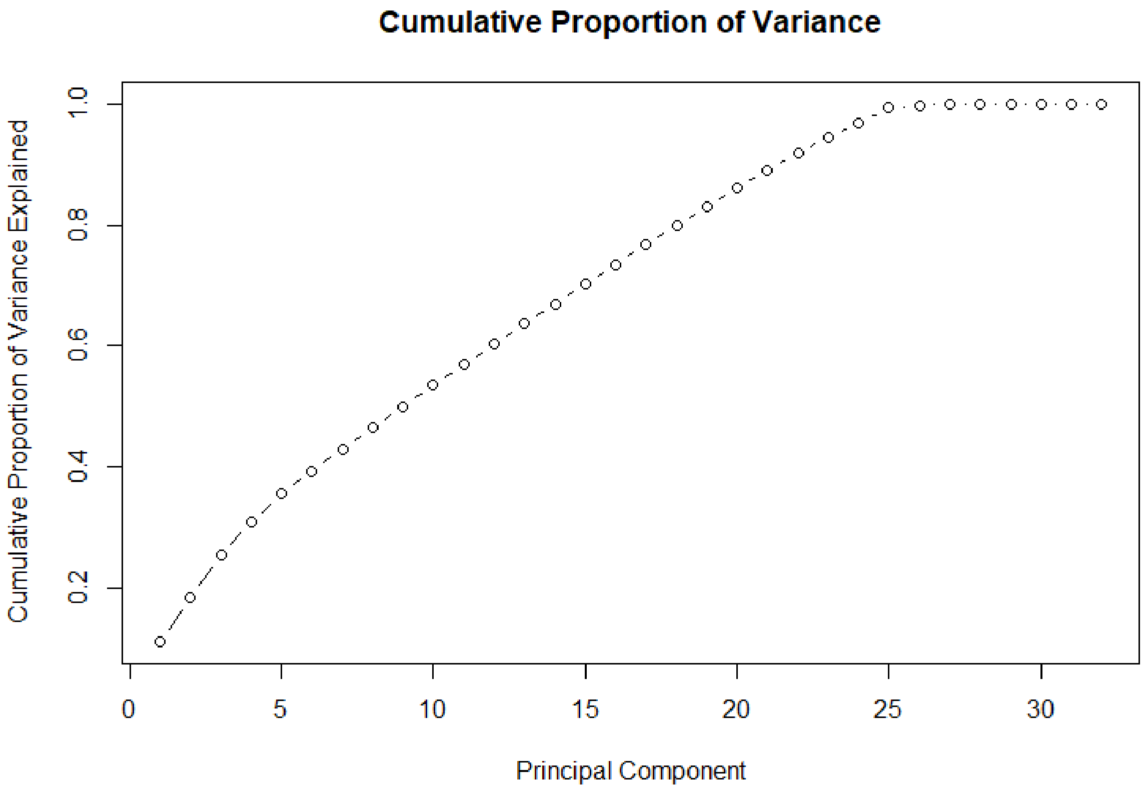


Figure 8 Cumulative Proportion of Variance

The cumulative proportion of variance scree plot affirmed the previous discovery. The principal components needed to describe most of the data is the first 25 features in the output of the PCA. Thus, only the first 25 features are selected and fed into the clustering algorithms.

K-means with PCA

```
###################### K-means ####################
# Elbow method to detect the best number of clusters for K-means
set.seed(123)
vec <- vector()
for (i in 1:10) {
  vec[i] = sum(kmeans(train2, i)$withinss)
}

plot(x = 1:10, y = vec, type = 'b', main = 'The Elbow Method', xlab = 'Number of Clusters',
ylab = 'WCSS')

# Fitting kmeans to the dataset
library(cluster)

set.seed(123)
kmeans <- kmeans(x = train2, centers = 8)
ykmeans <- kmeans$cluster

# Cluster membership
table(ykmeans)

# Visualizing the clusters
clusplot(train2, ykmeans, lines = 0, shade = T, color = T, plotchar = F, span = T,
        main = 'K-means clustering with Big Mart Sales data', xlab = 'X', ylab = 'Y')
```

PCA has narrowed down the number of features to be used in the clustering algorithm from 32 to 25. Now these 25 features are applied on K-means clustering algorithm. The K-means algorithm is similar as before. First, we use the Elbow method to determine the optimum number of clusters for clustering. The graph for the Elbow method is shown below.

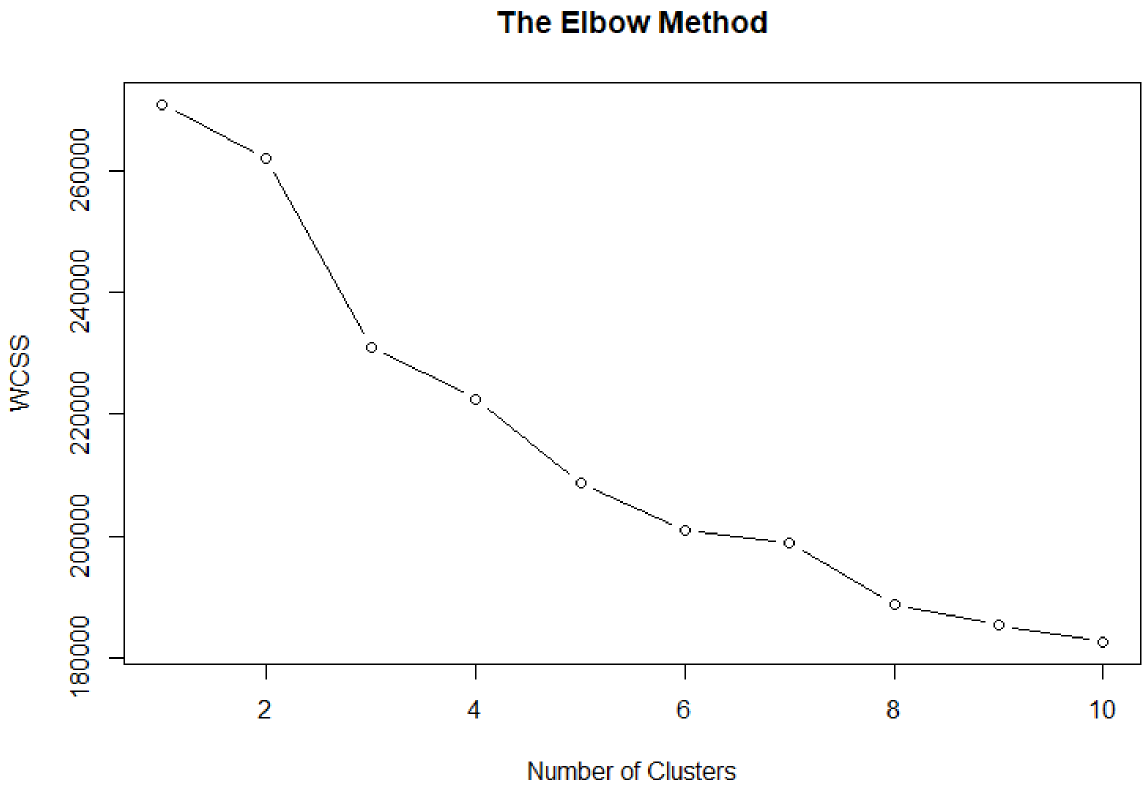


Figure 9 Elbow method with PCA

The result of the elbow method is not as definitive as before. It does not decrease and flatten out as the number of clusters increases. There is a spike as the number of cluster increases. The number of clusters chosen is 8 because the graph seems to flatten out after 8 clusters. With the number of clusters chosen, the data is visualized. The cluster membership is as follow.

Clusters	1	2	3	4	5	6	7	8
Number of objects	999	2013	528	1968	928	935	597	555

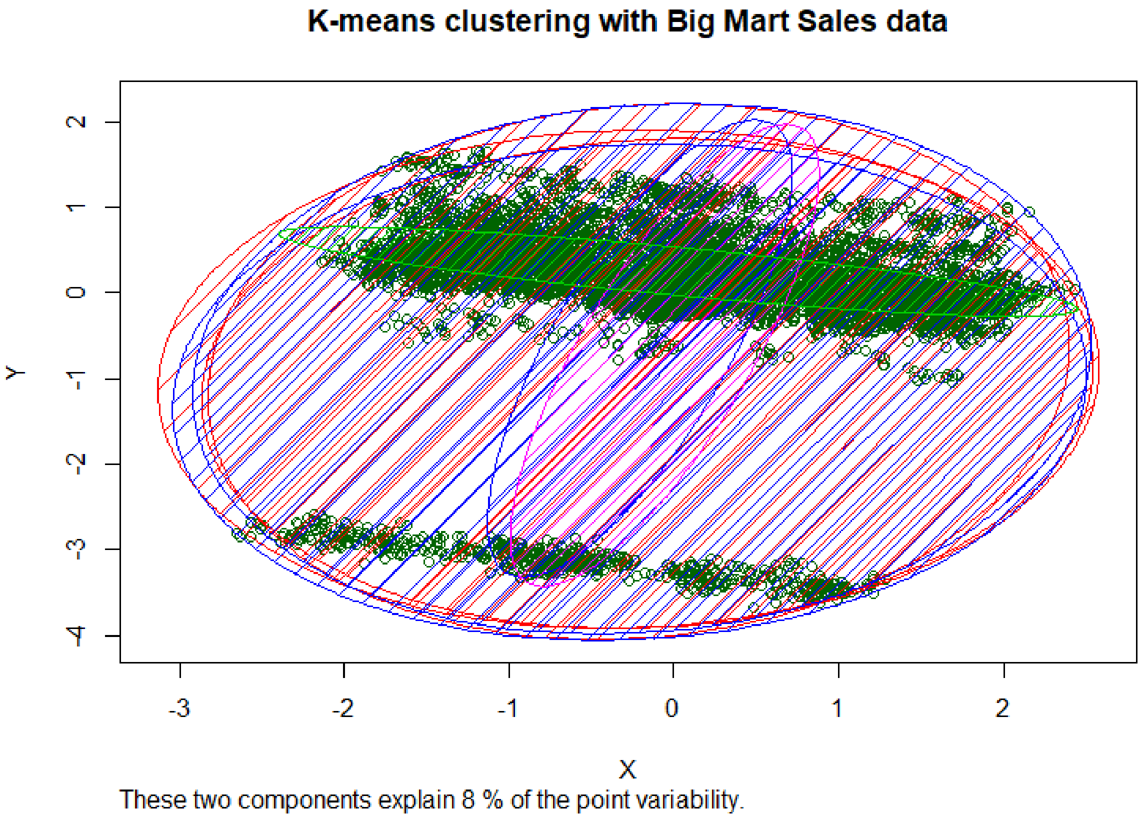


Figure 10 K-means clustering with PCA

The visualization of the clusters with K-means is shown above. There are 8 clusters visualized. It seems that the clusters are overlapping with one another. There are 25 features involved in the clustering, there would be complications to visualize such clusters in a 2D feature space.

Hierarchical Clustering with PCA

Based on the results of PCA, the first 25 features are the principal components or the most important features that describe the dataset. These 25 features are selected from the dataset and applied to hierarchical clustering.

```
################## Hierarchical clustering ##################
# Using the dendrogram to find the optimal number of clusters
hc = hclust(d = dist(train2, method = 'euclidean'), method = 'ward.D')
plot(hc, main = 'Dendrogram', xlab = 'Items', ylab = 'Euclidean distances')

# Fitting Hierarchical Clustering to the dataset
rect.hclust(hc, k = 3, border = "red")
y_hc = cutree(hc, 3)

table(y_hc)

# Visualising the clusters
library(cluster)
clusplot(train2, y_hc, lines = 0, shade = TRUE, color = TRUE, plotchar = FALSE, span = TRUE,
        main = 'Hierarchical clustering of the Big Mart Sales data', xlab = 'X', ylab = 'Y')
```

Similar to the hierarchical clustering performed before, a dendogram would be plotted to determine the number of clusters. The dendogram is shown below. After conducting PCA, it can be seen from the dendogram, the dataset can be grouped into more distinct clusters, the tree or dendogram has less branches than before. With this dendogram, we have to determine the number of clusters to group the data into. The chosen number of clusters is 3, which is the 3 main branches of the dendogram.

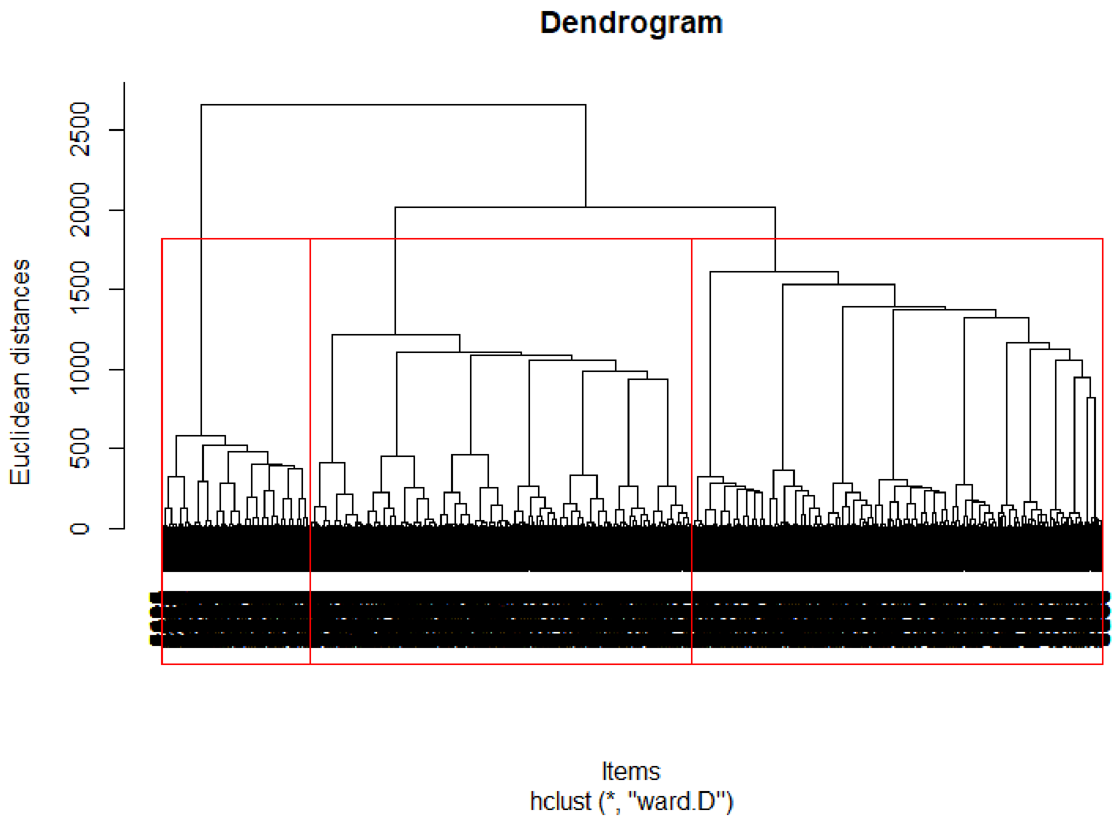


Figure 11 Dendogram with PCA

After the number of clusters is determined from the dendogram, we cut the tree and visualize the clusters. The cluster membership is shown below, and the visualization of the clusters is as follows.

Clusters	1	2	3
Number of objects	3454	3723	1346

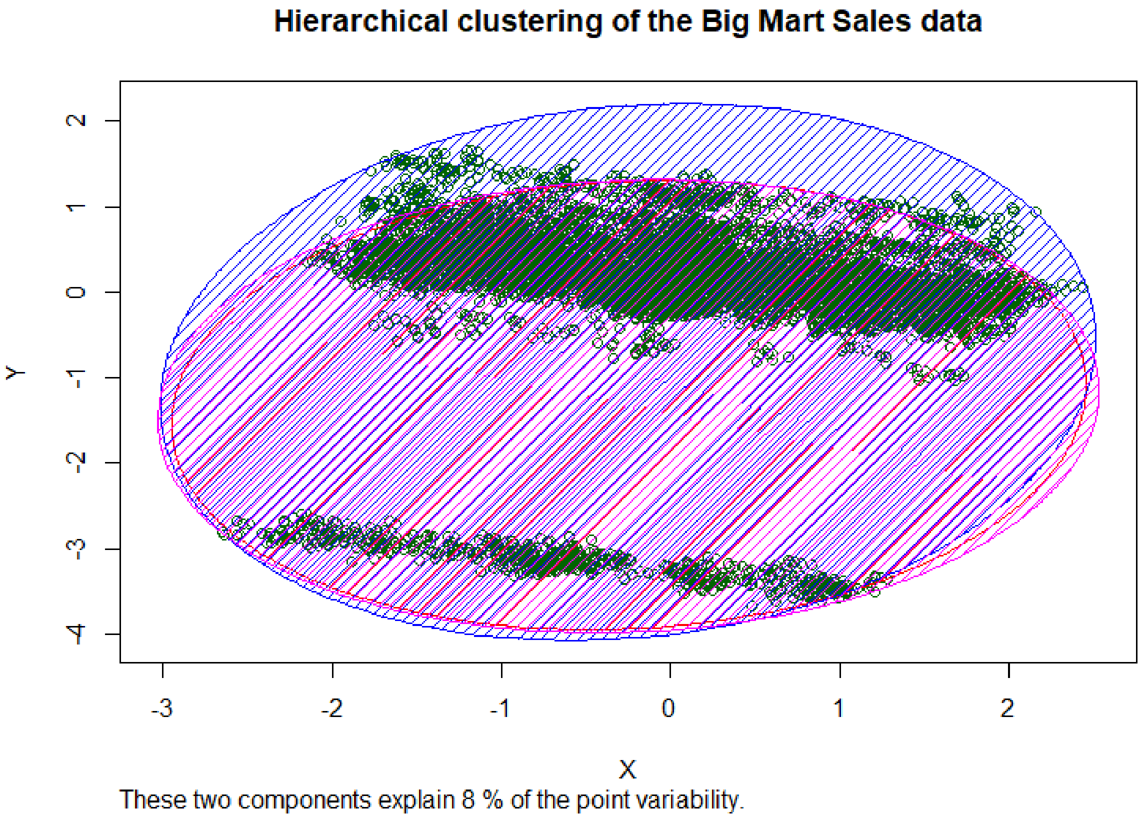


Figure 12 Hierarchical clustering with PCA

There are 3 clusters in the cluster visualization. The clusters seem to overlap one another. As mentioned, this might due to the number of features and the complexity to visualize the number of features in a 2D feature space.

Discussion

K value

For K-means clustering algorithms, the K value or the number of clusters have to be determined before clustering. The number of clusters can be determined with heuristics or the Elbow method shown above. The elbow method would try to group the data into 1 to 10 clusters and calculate the within cluster sum of error (WCSS) for each of the number of clusters. The number of WCSS would be high when the number of clusters is small, because it tries to group data with much dissimilarity into a cluster. As the number of clusters increase the WCSS would decrease, until it converges. When it converges, further increase in the number of clusters does not yield any significant decrease in WCSS. Then the number of clusters before the WCSS converges is chosen as the number of clusters for the dataset

In contrary, in hierarchical clustering there is no need to have the number of clusters determined in advance. The dendogram in hierarchical clustering method, group the data into different number of clusters based on the branched. The user can decide how many clusters should the data be divided into based on the dendogram.

Categorical data

K-means depends on Euclidean distance to calculate the similarity or dissimilarity between objects. Therefore, K-means does not work well with categorical data. The sample space of categorical data is discrete and does not have a natural origin. A Euclidean distance function on such a sample space (categorical) is not meaningful.

Hierarchical clustering can use different distance metric to calculate the dissimilarity and similarity of the objects. If Euclidean distance is used, it would not be suitable to have categorical values in the data. If there are categorical values in the data, another distance metric can be applied, the Gower's distance metric. Gower's distance metric could handle the categorical values.

Principal Component Analysis

There are 12 columns in the dataset, removing the response variable, the identifiers and columns that play no roles in clustering (‘Outlet_Establised_Year’), there are 9 features in the dataset. Which of these features are best described the data and would be best applied to the clustering algorithms? The PCA would provide an insight into this issue. Some of the features are in categorical values, PCA can only be applied on continuous values. Therefore, all the categorical values are converted into continuous values and PCA is applied to find out the features that best described the data. After the categorical values conversion, there are 32 features in the dataset. Then PCA is applied and the best features are narrowed down to 25 features. These 25 features are subsequently used in both K-means and hierarchical clustering.

Comparison of the performance

K-means is relatively more efficient and less costly in term of computing power. In the case without PCA, K-means clustering with just the features in numerical values, it can explain 67.64% of the variability in the data. In other words, the features show the 67.64% of the variance in the dataset. On the other hand, the same numerical features are applied to Hierarchical clustering without PCA and with Euclidean distance, it achieved the same result which is 67.64% of the variability of the data. This show that the algorithms may show similar result if the same features are used.

To drive home the point, we tried hierarchical clustering with categorical data using Gower distance to calculate the similarity matrix. With the categorical features, there are 8 clusters rather than 4 clusters and the features explained 57.27% of the variability of the data. The result is not an improvement on the result with just the numerical features but it do show that with different features, different levels of variability of the data explained can be achieved.

Theoretically, PCA would find the most important features in the dataset and using only the features in analysis would give a better if not comparable result. However, in this case, after applying PCA, the performance of the clustering algorithms has decreased. The 25 features found by PCA should be able to explain the variance in the dataset but based on the result of

K-means and hierarchical clustering, the 25 features can only explain 8% of the variability of the data. In comparison to 67.64% just by using the numerical features, the result from PCA is not great.

This dip in performance of the PCA could be due to the conversion of categorical values into continuous values and the number of features used in the clustering algorithms. The number of features is high and the complexity increases. There may a need to fine tune the PCA algorithm to suit the need of the Big Mart Sales dataset to get a better result.

Conclusion

After applying both K-means and hierarchical clustering algorithms to the Big Mart Sales dataset, it is found that the most important part of the task is at feature selection. With the same features, the result of both K-means and hierarchical clustering would be similar. The best features to be used in the Big Mart Sales dataset would be the columns with numerical values namely, Item_Weight, Item_Visibility, and Item_MRP. The result obtained here with the selected features is 67.64% which is better than the result from just the nominal features.

In this study, we have seen the advantages of each of the clustering algorithms at work. Hierarchical clustering does not need a predetermined K value to work while K-means is relatively more efficient. The output for both algorithms would be quite similar if the same features are used.

A further study would be to find a better set of features from the dataset that can explain more than 80% of the variance in the data and fine tune the PCA to obtain better result.