General Sir John Kotelawala Defence University,

Faculty of Management, Social Sciences and Humanities,

Department of Languages.

BSc in Applied Data Science Communication

Y. K. N. Rathnasiri-D/ADC/21/0034

K.M.P.D. Bandara – D/ADC/21/0026

W.B.R. Munasinghe - D/ADC/21/0011

[Fundamentals of Data Mining / LB2114]

[Year 2: Semester 1]

[Assignment number1 / Group number 10]

[05/23/2022]

Content: - Page no: -

Implementing classification in data mining to prevent diabetes impacting on pregnancies.

Task-1 classification.

Introduction.

Diabetes is a chronic health condition that affects how our body turns the food we consume into the energy we use in our day-to-day life. Most of the food we ate was broken down into glucose and released into our bloodstream. When our blood sugar goes up, our body automatically releases insulin to decrease the blood sugar level. But sometimes our body cannot produce the insulin hormone and due to this problem, our body's blood sugar level goes up without any stop. Because of this problem diabetes was born.

But when it comes to pregnant women diabetes would affect the mother and the baby. So having diabetes during pregnancy was a significant risk. Therefore, here we plan to use an Indian data set about women who are older than twenty-one and check their diagnostic measurements to understand if they are having diabetes or not. By using this data set we also plan to understand how the diagnostic measurements affect diabetes.

Datasets.

Pima Indians Diabetes Dataset was from the National Institute of Diabetes and Digestive and Kidney Diseases. In this dataset it predicts the diabetes diagnosis and that was predict by the diagnostic measurements whether a patient has diabetes or not. In this dataset all the patients are females at least twenty-one years old. Therefore, by using this we could Esley obtain the result of diabetes diagnosis's patients.

So, we plan to use this dataset to understand how the diagnosis measurements such as blood sugar, BMI value, Skin-thickness would affect the diabetes level of a Pregnant women and how it will be affecting their health. In the globe there were so many pregnancies happened with having diabetes and quite a lot of lives were lost due to this health problem. That’s why we chose this data set to analyze and help with this problem.

Not only to measure the diabetes level but also, we plan to use this data set to understand how these different types of diagnosis measurements would affect their long-term health. So, we also understand that by using this dataset and predicting data mining analysis on this we could help the medical community with understanding these patterns.

Explanation and preparation of datasets.

Explanation: -

This dataset was created in 2016 by Pima Indians Diabetes Database. Its objective was to understand and predict whether a patient has diabetes by using diagnostic measurements. In this dataset, there were 8 main diagnosis measurements and one column showing whether the patient has diabetes or not. Therefore, we used a diabetes data set with 9 main variables to do the classification in data mining. Here are the main 9 variables of the dataset.

- Pregnancies: Number of times pregnant
- Glucose: Plasma glucose concentration 2 hours in an oral glucose tolerance test
- Blood Pressure: Diastolic blood pressure (mm Hg)
- Skin Thickness: Triceps skin fold thickness (mm)
- Insulin: 2-Hour serum insulin (mu U/ml)
- BMI: Body mass index (weight in kg/ (height in m) ^2)
- Diabetes Pedigree Function: Diabetes pedigree function
- Age: Age (years)
- Outcome: Class variable (0 or 1)

	A	B	C	D	E	F	G	H	I	J
1	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome	
2	6	148	72	35	0	33.6	0.627	50	1	
3	1	85	66	29	0	26.6	0.351	31	0	
4	8	183	64	0	0	23.3	0.672	32	1	
5	1	89	66	23	94	28.1	0.167	21	0	
6	0	137	40	35	168	43.1	2.288	33	1	
7	5	116	74	0	0	25.6	0.201	30	0	
8	3	78	50	32	88	31	0.248	26	1	
9	10	115	0	0	0	35.3	0.134	29	0	
10	2	197	70	45	543	30.5	0.158	53	1	
11	8	125	96	0	0	0	0.232	54	1	
12	4	110	92	0	0	37.6	0.191	30	0	
13	10	168	74	0	0	38	0.537	34	1	

So, in this dataset, there are 8 independent values and 1 dependent value. The independent values are Pregnancy, Glucose, Blood Pressure, Skin Thickness, Insulin, BMI, Diabetes Pedigree Function, and lastly Age. The only dependent variable is the outcome, and that variable depends on all the other variables, or else we can call them diagnosis measurements.

Preparation of datasets: -

Now we are going to talk about the preparation of this data set. Before we used this dataset, we investigated it and chose if this dataset was accurate to do our project. Then we had to investigate the data and understand the data of this dataset and if this dataset would be accurately operated.

After that, we had to normalize this dataset according to our use. But we didn't have to do much to prepare this dataset. Because most of this dataset was conversing to do our work.

Implementation in R.

We used r software to do this dataset's preparation. Here we are going to discuss the codes and functions we used in r software. Therefore, to prepare this data set, we chose a few functions. So now we are going to talk about the data packages we install in r language to do the classification in data mining.

'install.packages ('caTools')' we used this package to train, test data, and split data in the R language. Then we used the 'install.packages('dplyr')' to manipulate the data that we are going to use doing classification data mining. 'install.packages('ggplot2')' this R package was used to do data visualization. Then we used 'install.packages('caret')' command to get the R package to do the Confusion Matrix in data mining. Lastly, we used the 'install.packages('corrplot')' command to be able to do the Correlation Plot part.

First, we used the 'library(readr)' function to... here is the example for that.

```
library(readr)

## Warning: package 'readr' was built under R version 4.1.3

testss <- read_csv("diabetes.csv")

## Rows: 768 Columns: 9
## -- Column specification --------------------------------------------------------
------
## Delimiter: ","
## dbl (9): Pregnancies, Glucose, BloodPressure, SkinThickness, Insulin, BMI,
D...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this
message.
```

Then we used the 'names(testss)' function to understand the main categories of this data set.

```
names(testss)

## [1] "Pregnancies"              "Glucose"
## [3] "BloodPressure"            "SkinThickness"
## [5] "Insulin"                  "BMI"
## [7] "DiabetesPedigreeFunction" "Age"
## [9] "Outcome"
```

We used the ‘summary(testss)’ function to get the summary of this data set.

```
summary(data)

##   Pregnancies         Glucose        BloodPressure     SkinThickness
##  Min.   :-1.1411   Min.   :-3.7812   Min.   :-3.5703   Min.   :-1.2874
##  1st Qu.:-0.8443   1st Qu.:-0.6848   1st Qu.:-0.3671   1st Qu.:-1.2874
##  Median :-0.2508   Median :-0.1218   Median : 0.1495   Median : 0.1544
##  Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.6395   3rd Qu.: 0.6054   3rd Qu.: 0.5629   3rd Qu.: 0.7186
##  Max.   : 3.9040   Max.   : 2.4429   Max.   : 2.7327   Max.   : 4.9187
##     Insulin             BMI             DiabetesPedigreeFunction
##  Min.   :-0.6924   Min.   :-4.057829   Min.   :-1.1888
##  1st Qu.:-0.6924   1st Qu.:-0.595191   1st Qu.:-0.6885
##  Median :-0.4278   Median : 0.000941   Median :-0.2999
##  Mean   : 0.0000   Mean   : 0.000000   Mean   : 0.0000
##  3rd Qu.: 0.4117   3rd Qu.: 0.584390   3rd Qu.: 0.4659
##  Max.   : 6.6485   Max.   : 4.452906   Max.   : 5.8797
```

Then we used the ‘head(testss)’ to get the first few rows of the dataset and by using this function we could get an overall idea about the data set.

```
head(testss)

## # A tibble: 6 x 9
##   Pregnancies Glucose BloodPressure SkinThickness Insulin   BMI
DiabetesPedigre~
##         <dbl>   <dbl>         <dbl>         <dbl>   <dbl> <dbl>
<dbl>
## 1           6     148            72            35       0  33.6
0.627
## 2           1      85            66            29       0  26.6
0.351
## 3           8     183            64             0       0  23.3
0.672
## 4           1      89            66            23      94  28.1
0.167
## 5           0     137            40            35     168  43.1
2.29
## 6           5     116            74             0       0  25.6
```

By using the str function we could get the names of the columns, class of each column, followed by some of the initial observations of each of the columns.

```
str(testss)
```

```
## spec_tbl_df [768 x 9] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Pregnancies             : num [1:768] 6 1 8 1 0 5 3 10 2 8 ...
##  $ Glucose                 : num [1:768] 148 85 183 89 137 116 78 115 197
125 ...
##  $ BloodPressure           : num [1:768] 72 66 64 66 40 74 50 0 70 96 ...
##  $ SkinThickness           : num [1:768] 35 29 0 23 35 0 32 0 45 0 ...
##  $ Insulin                 : num [1:768] 0 0 0 94 168 0 88 0 543 0 ...
##  $ BMI                     : num [1:768] 33.6 26.6 23.3 28.1 43.1 25.6 31
35.3 30.5 0 ...
##  $ DiabetesPedigreeFunction: num [1:768] 0.627 0.351 0.672 0.167 2.288 ...
##  $ Age                     : num [1:768] 50 31 32 21 33 30 26 29 53 54 ...
##  $ Outcome                 : num [1:768] 1 0 1 0 1 0 1 0 1 1 ...
```

Then we used the dim function to get the dimensions of the specified matrix on the data.

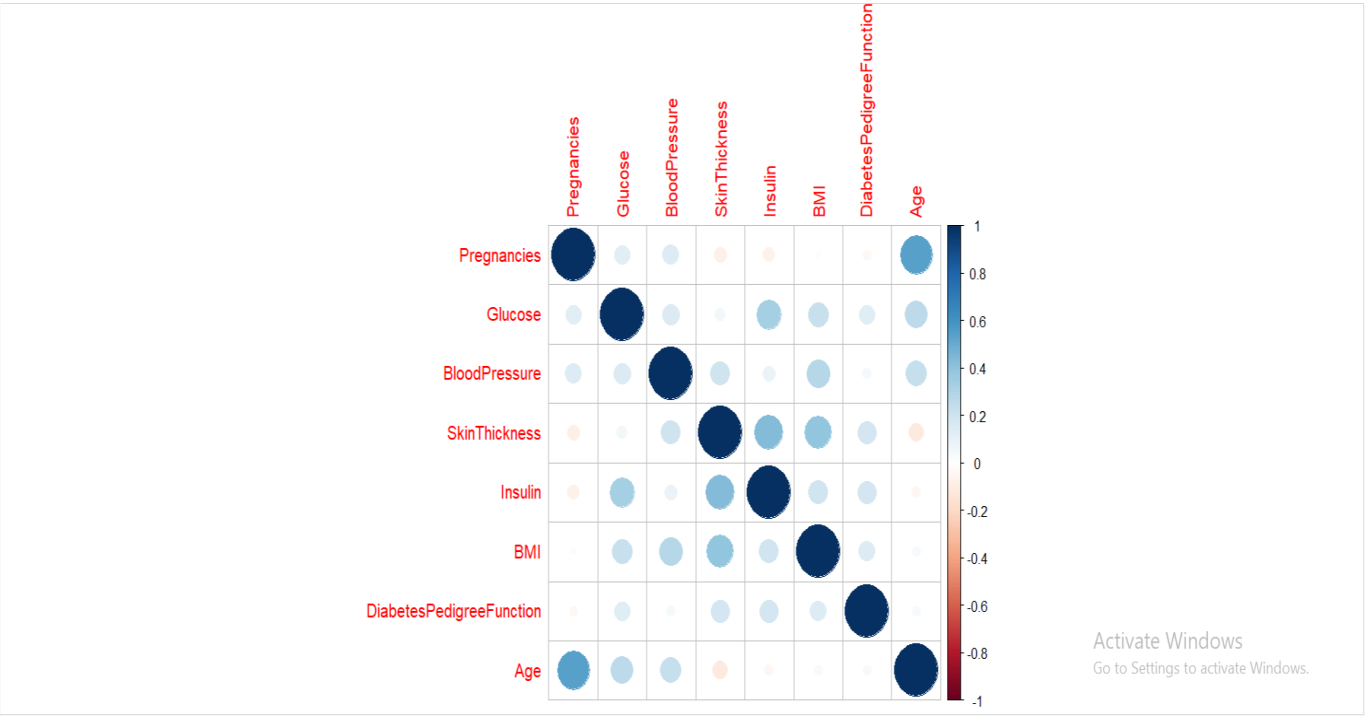
```
dim(data)
```

```
## [1] 768   9
```

In here we used corrplot to understand the distribution of this dataset by this plot. And it makes it easier to understand the data set more easily.

And corrplot is a visual exploratory graph we could easily obtain the correlation matrix that supports automatic variable reordering to help detect hidden patterns among variables. corrplot is very easy to use and provides a rich array of plotting options in visualization method too. Therefore, we chose to use this graph method to visualize our data.

```
corrplot(cor(data[,-9]))
```



Data mining.

Classification in data mining is a function that is a way of assigning items in a collection to target categories or classes. We use the classification method to accurately predict the target class for each case in the data. There are two main types of classification parts. They are binary classification and multiclass classification. Binary classification is used to attribute two possible values and multiclass classification targets more than two values.

Now we are going to talk about the main steps we took to do this classification in data mining.

We used the dim function to find how many columns and how many rows are there in this dataset. As you can see there are 768 rows and 9 columns in this dataset.

```
dim(data)

## [1] 768   9
```

We determined the class label for the query data point using this command. And this code shows us we reserved 89.59% accuracy for our model data.

```
predicted.type = knn(train[1:9], test[1:9], train$Outcome, k=1)

error = mean(predicted.type!=test$Outcome)

confusionMatrix(predicted.type, as.factor(test$Outcome))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 139  13
##          1  11  67
##
##                Accuracy : 0.8957
##                  95% CI : (0.8487, 0.932)
##     No Information Rate : 0.6522
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.7687
##
##  Mcnemar's Test P-Value : 0.8383
##
##             Sensitivity : 0.9267
##             Specificity : 0.8375
##          Pos Pred Value : 0.9145
##          Neg Pred Value : 0.8590
##              Prevalence : 0.6522
##          Detection Rate : 0.6043
##    Detection Prevalence : 0.6609
##       Balanced Accuracy : 0.8821
```

We used this command to calculate the Euclidean distances in our dataset.

```
predicted.type = NULL
error.rate = NULL

for (i in 1:10){
  predicted.type = knn(train[1:9], test[1:9], train$Outcome, k=i)
  error.rate[i] = mean(predicted.type!=test$Outcome)
}
```

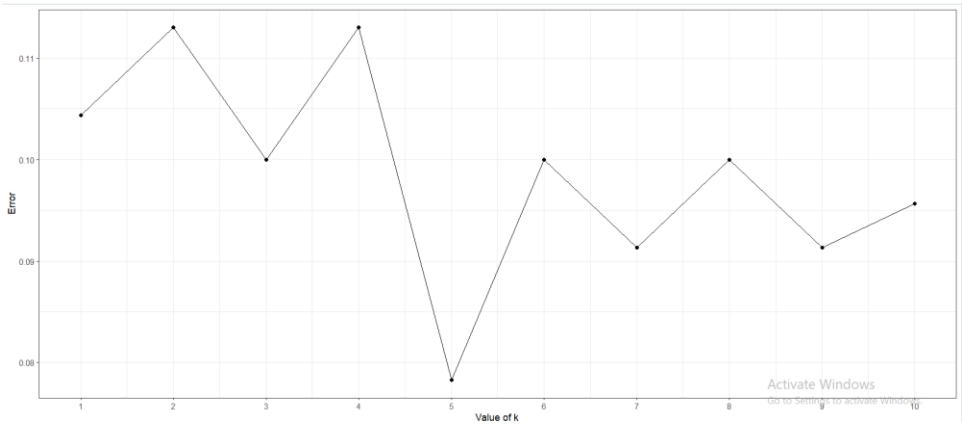
We used this code to create a graph to find the lowest possible error and we obtain the result of lowest error, or we can call it as the k value is at 5.

```
knn.error = as.data.frame(cbind(k=1:10, error.type=error.rate))
knn.error
```

```
##     k error.type
## 1   1 0.10434783
## 2   2 0.12608696
## 3   3 0.10000000
## 4   4 0.10434783
## 5   5 0.07826087
## 6   6 0.10434783
## 7   7 0.09130435
## 8   8 0.10869565
## 9   9 0.09130435
## 10 10 0.10434783
```

We used this code to create a graph to find the lowest possible error and we obtain the result of lowest error, or we can call it as the k value is at 5.

```
ggplot(knn.error, aes(k,error.type))+
  geom_point()+
  geom_line()+
  scale_x_continuous(breaks = 1:10)+
  theme_bw()+
  xlab("Value of k")+
  ylab("Error")
```




```
predicted.type = knn(train[1:9], test[1:9], train$Outcome, k=5)
```

After we found the lowest error, we implement our knowledge to predict the error.

```
error = mean(predicted.type!=test$Outcome)
error
```

```
## [1] 0.07826087
```

In the end, we used this command to get the Confusion Matrix and the final accuracy level. so, in the end, we obtain an accuracy level of 92.17%. Therefore, we can predict that our Accuracy is good.

```
confusionMatrix(predicted.type, as.factor(test$Outcome))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 143  11
##          1   7  69
##
##                Accuracy : 0.9217
##                  95% CI : (0.8791, 0.953)
##     No Information Rate : 0.6522
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8255
##
##  Mcnemar's Test P-Value : 0.4795
##
##             Sensitivity : 0.9533
##             Specificity : 0.8625
##          Pos Pred Value : 0.9286
##          Neg Pred Value : 0.9079
##              Prevalence : 0.6522
##          Detection Rate : 0.6217
##    Detection Prevalence : 0.6696
##       Balanced Accuracy : 0.9079
##
##        'Positive' Class : 0
```

Results analysis and discussion.

From the above result, we have confirmed that our classification of data mining results was accurate. Therefore, I think we can use this result to understand more about the women pregnancies and how diabetes would affect them using the diagnostic measurements.

So, the main point of applying this dataset to the classification rule of data mining is to understand the pattern of this dataset and use this dataset to have a better idea about what to expect from women who have diabetes while pregnant and after their pregnancy. So according to this data set we have calculated the accuracy of this data set is around 92.17%. Then we also obtain the result of positive class is zero.

Conclusions.

We used a dataset from Pima Indians Diabetes Dataset from the National Institute of Diabetes and Digestive and Kidney Diseases. This dataset was used to calculate if a patient has diabetes or not by using the diagnostic measurements of whether a patient has diabetes or not. The subject to create this dataset was female patients who were at-least twenty-one years old. Therefore, by using this we could easily obtain the result of diabetes diagnosis's patients. Also, we could use this result to understand how a diabetes pregnant woman can could survive childbirth.

References.

Pima Indians Diabetes Database (2016).data world

https://data.world/data-society/pima-indians-diabetes-database

Implementing clustering in data mining to have the distribution of death count in England and Wales.

Task2 - Clustering.

Introduction.

When it comes to death, we cannot stop It or push it for later reference. Most of the time death rate of a country recode by the local authority and because of this, it was hard to get accurate data about a whole area or reigns' death frequency. and sometimes when it comes to studying diseases or viruses understanding the area that cursed death was very sufficient method to stop those types of curses.

In 2020 we had to face a global virus called covid-19 and even with modern technology, it was hard to number down the number of patients and the areas where the virus was spreading. There was also the problem of understanding how fast the virus was spreading and how the virus was affecting to the normal death rate of a country.

So now we are going to use Clustering datamining method to get the death rate of England and divide the data into groups of categories to understand the types of death cursed through the country and help to have a better understanding of deathrate and types of death happened in region wise in Londen and Wales.

And Clustering is a way of dividing data into several groups so that the data in the same groups are more like other data points in the same group than those in other groups. In simple words, the aim is to separate groups with similar traits and assign them into clusters.

Datasets.

To do this project of clustering in data mining we plan to use a data set about the death count of England and Wales. The data set was created by collecting data from local authority, health board and place of death in the latest weeks for which data are available. This dataset was started to be created in 2020 and its latest edition was in 2022 May 17.

So, by using this dataset we plan to get to know more about the death rate of England and Wales and, we could understand how much the death of this country affecting to its culture and economics also some other aspects. The biggest point was to use this dataset in applying clustering in data mining to categorize the data to understand the types of death cursed in England. Also, we plan to get to know about the frequency of death happened in region wise in England and Wales to create categories to obtain the result of how to organize the health care system according to the death that happened in England.

Explanation and preparation of datasets.

Explanation: -

This dataset is named as Provisional count of the number of deaths registered in England and Wales. This dataset was created around 2020-2021. So, as we can see it was the primary time that the Corna virus was spreading. Therefore, in this dataset, the death of covid-19 patients was added to this dataset too. This dataset was created by using the data from local authorities of England and Wales, health boards, and places of death in the latest weeks for which data are available in that country. This dataset was last updated on 22 June 2021.

In this dataset, there were 7 columns in this dataset and this dataset has more than 7000 rows in it. Also, there are few dependent and independent variables in this data set. Here are the seven columns in this dataset.

- Area code.
- Geography type.
- Area name.
- Cause of death.
- Week number.
- Place of death.
- Number of deaths.

1	Area code	Geography type	Area name	Cause of death	Week number	Place of death	Number of deaths
2	E06000001	Local Authority	Hartlepool	All causes	1	Care home	9
3	E06000001	Local Authority	Hartlepool	All causes	1	Elsewhere	0
4	E06000001	Local Authority	Hartlepool	All causes	1	Home	7
5	E06000001	Local Authority	Hartlepool	All causes	1	Hospice	1
6	E06000001	Local Authority	Hartlepool	All causes	1	Hospital	6

Preparation of datasets: -

Before we used this provisional count of the number of deaths in the England and Wales dataset in the clustering part of machine learning we had to prepare this dataset to be able to apply it in clustering. Therefore, we check if there are any null values in this dataset. But this dataset was clear from any null values, so we didn't have to do too much about it. Also, to normalize this

dataset there was no problem with dataset's rows and columns therefore there wasn't much to do in this part too.

Implementation in R.

To make this dataset work in the R language we used some r packages in r software. So here we are going to do a small description of those r packages we used to do the clustering in data mining.

'install.packages ('caTools')' we used this package to train, test data, and split data in the R language. Then we used the 'install.packages('dplyr')' to manipulate the data that we are going to use doing classification data mining. 'install.packages('ggplot2')' this R package was used to do data visualization. Then we used 'install.packages('caret')' command to get the R package to do the Confusion Matrix in data mining. Lastly, we used the 'install.packages('corrplot')' command to be able to do the Correlation Plot part.

Now we are going to talk about the codes we used in the R language to get to know about this dataset more efficiently.

From this code, we obtain the result of the main columns in the dataset so we can get a quick overview of the data we are going to use.

```
names(Deaths_on_Eng_Data)

## [1] "Area.code"        "Geography.type"   "Area.name"
"Cause.of.death"
## [5] "Week.number"      "Place.of.death"   "Number.of.deaths"
```

This 'head' function was used to get the first few rows in this dataset therefore we could understand what type of data there is in this dataset.

```
head(Deaths_on_Eng_Data)

##   Area.code  Geography.type  Area.name Cause.of.death Week.number
## 1 E06000001 Local Authority Hartlepool     All causes           1
## 2 E06000001 Local Authority Hartlepool     All causes           1
## 3 E06000001 Local Authority Hartlepool     All causes           1
## 4 E06000001 Local Authority Hartlepool     All causes           1
## 5 E06000001 Local Authority Hartlepool     All causes           1
## 6 E06000001 Local Authority Hartlepool     All causes           1
##                 Place.of.death Number.of.deaths
## 1                    Care home                9
## 2                    Elsewhere                0
## 3                         Home                7
## 4                      Hospice                1
## 5                     Hospital                6
## 6 Other communal establishment                0
```

This code is the same as the ‘head’ function but does the opposite reaction so, this ‘tail’ code gives us the last few rows in this dataset and helps us to understand the distribution of this dataset as we compared it to the head function.

```
tail(Deaths_on_Eng_Data)

##       Area.code Geography.type                          Area.name
## 73003 W11000031   Health Board Swansea Bay University Health Board
## 73004 W11000031   Health Board Swansea Bay University Health Board
## 73005 W11000031   Health Board Swansea Bay University Health Board
## 73006 W11000031   Health Board Swansea Bay University Health Board
## 73007 W11000031   Health Board Swansea Bay University Health Board
## 73008 W11000031   Health Board Swansea Bay University Health Board
##       Cause.of.death Week.number              Place.of.death
Number.of.deaths
## 73003       COVID 19          18                   Care home
0
## 73004       COVID 19          18                   Elsewhere
0
## 73005       COVID 19          18                        Home
1
```

This ‘summary’ code is used to get the summary of this dataset so using this code we could understand the length of each column, class and the character of each column.

```
summary(Deaths_on_Eng_Data)

##   Area.code         Geography.type      Area.name         Cause.of.death
##  Length:73008       Length:73008       Length:73008       Length:73008
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##   Week.number   Place.of.death     Number.of.deaths
##  Min.   : 1.0   Length:73008       Min.   :  0.000
##  1st Qu.: 5.0   Class :character   1st Qu.:  0.000
##  Median : 9.5   Mode  :character   Median :  0.000
##  Mean   : 9.5                      Mean   :  3.221
##  3rd Qu.:14.0                      3rd Qu.:  3.000
##  Max.   :18.0                      Max.   :137.000
```

We used ‘str’ code to get the result to get the compactly displaying the internal structure of deaths in England.

```
str(Deaths_on_Eng_Data)

## 'data.frame':    73008 obs. of  7 variables:
##  $ Area.code       : chr  "E06000001" "E06000001" "E06000001" "E06000001"
...
##  $ Geography.type  : chr  "Local Authority" "Local Authority" "Local
Authority" "Local Authority" ...
##  $ Area.name       : chr  "Hartlepool" "Hartlepool" "Hartlepool"
"Hartlepool" ...
##  $ Cause.of.death  : chr  "All causes" "All causes" "All causes" "All
causes" ...
##  $ Week.number     : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ Place.of.death  : chr  "Care home" "Elsewhere" "Home" "Hospice" ...
##  $ Number.of.deaths: int  9 0 7 1 6 0 0 0 1 0 ...
```

Then we used this ‘nrow’, ‘ncol’, and ‘dim’ function to obtain the result of a number of rows and columns.

```
nrow(Deaths_on_Eng_Data)

## [1] 73008

ncol(Deaths_on_Eng_Data)

## [1] 7

dim(Deaths_on_Eng_Data)

## [1] 73008     7
```

So, these are some codes we used in R software to understand this dataset and get some general idea about it.

Data mining.

Cluster analysis is widely used in many areas. Traditionally, clustering is considered unsupervised learning because it does not have a class label or quantitative response variable, which is present in supervised learning such as classification and regression. Clustering is a division of data into groups of similar objects. Representing the data by fewer clusters necessarily loses certain fine details but achieves simplification. Data is formed through its groups. Data modeling puts clustering in a historical perspective rooted in mathematics, statistics, and numerical analysis.

In this project, we will be using "cluster" packages that use the K-means clustering Algorithm. K-means (MacQueen, 1967) is one of the simplest unattended learning algorithms that solve the well-known cluster problem. And we are going to use the ‘factoextra’ package to visualize the matrix.

We Selected 3 columns for the cluster analysis by using this code.

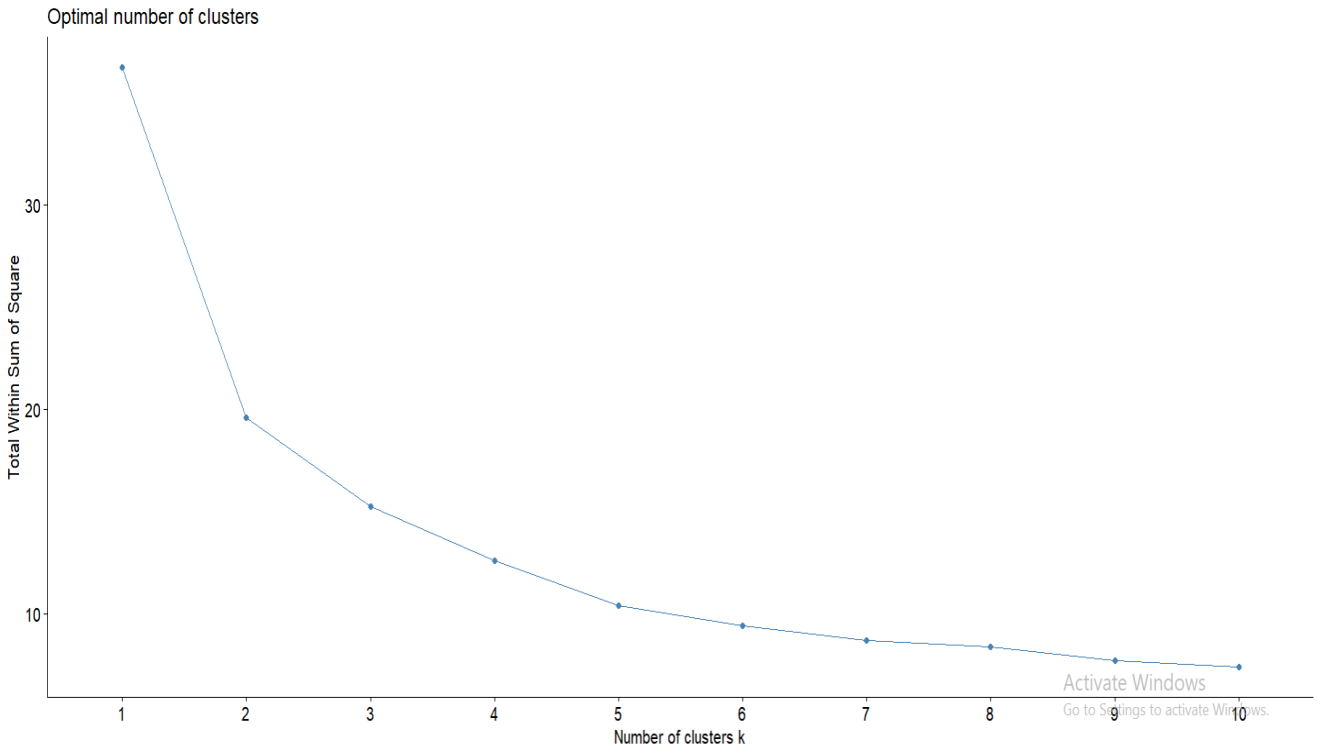
```
Deaths_on_Eng_Data_A = Deaths_on_Eng_Data[,c(3,6,7)]
names(Deaths_on_Eng_Data_A)
```

Then we normalized the dataset using 'normalise' function.

```
normalise <- function(df)
{
  return(((df- min(df)) /(max(df)-min(df))*(1-0))+0)
}

Area.name=rownames(Deaths_on_Eng_Data_pivot)
Deaths_on_Eng_Data_pivot_n=as.data.frame(lapply(Deaths_on_Eng_Data_pivot,norm
alise))
rownames(Deaths_on_Eng_Data_pivot_n)=Area.name
```

We created a graph to choose the right number of expected clusters, or we can say the k values. To create this graph, we used the 'fviz_nbclust' code.

```
fviz_nbclust(Deaths_on_Eng_Data_pivot_n, kmeans, method = "wss")
```



Then we performed k-means clustering on our dataset of death count of England and Wales dataset with k=3.

```
set.seed(130)
km.fit <- kmeans(Deaths_on_Eng_Data_pivot_n, 3, nstart = 30)
km.fit$cluster
```

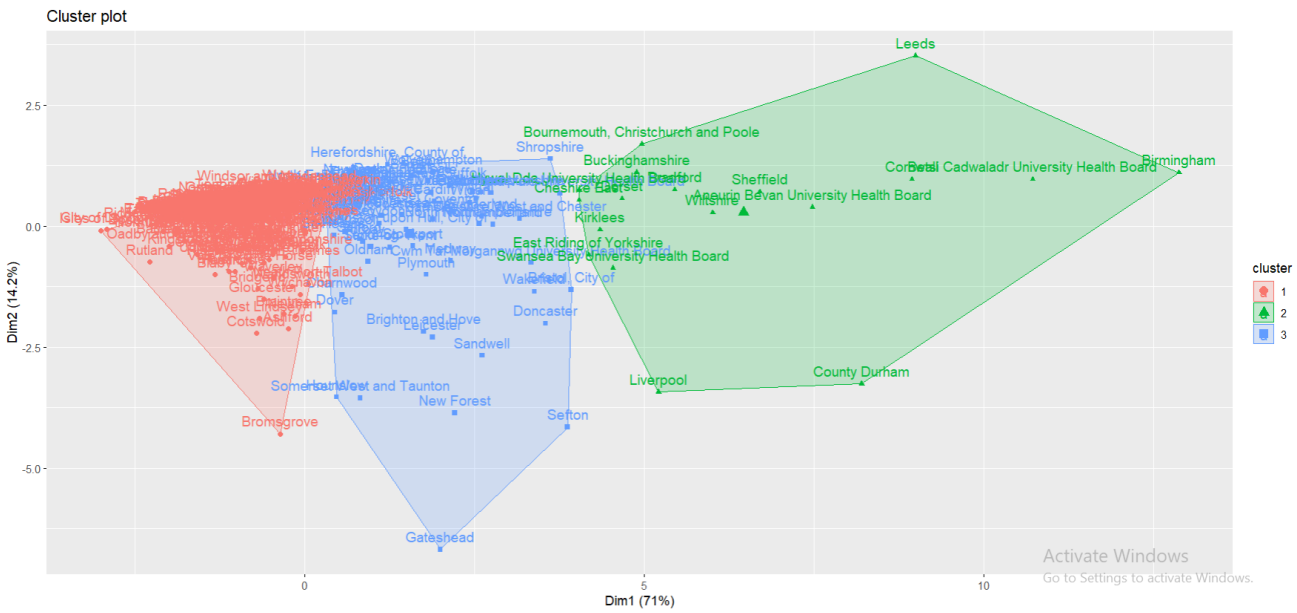
```
##                                       Adur
##                                          1
##                                  Allerdale
##                                          1
##                                Amber Valley
##                                          1
##     Aneurin Bevan University Health Board
##                                          2
##                                       Arun
##                                          3
##                                  Ashfield
##                                          1
##                                   Ashford
##                                          1
##                                   Babergh
##                                          1
##                       Barking and Dagenham
##                                          1
##                                    Barnet
##                                          3
##                                  Barnsley
##                                          3
##                            Barrow-in-Furness
```

After that we Visualize clusters using the 'fviz_cluster()' function in factoextra package to have an idea about this dataset.

```
km.fit$size
```

```
## [1] 245  18  75
```

```
fviz_cluster(km.fit,Deaths_on_Eng_Data_pivot_n)
```



Since "death count of England and Wales" has a large number of data, therefore we perform k-means clustering on 'Deaths_on_Eng_Data_pivot_' with k=3.

```
Deaths_on_Eng_Data_pivot_n2 = subset(Deaths_on_Eng_Data_pivot_n,
rownames(Deaths_on_Eng_Data_pivot_n)!="Aneurin Bevan University Health
Board")

set.seed(130)
km.fit2 <- kmeans(Deaths_on_Eng_Data_pivot_n2, 3, nstart = 30)
km.fit2$cluster

##                                    Adur
##                                       1
##                                Allerdale
##                                       1
##                             Amber Valley
##                                       1
```

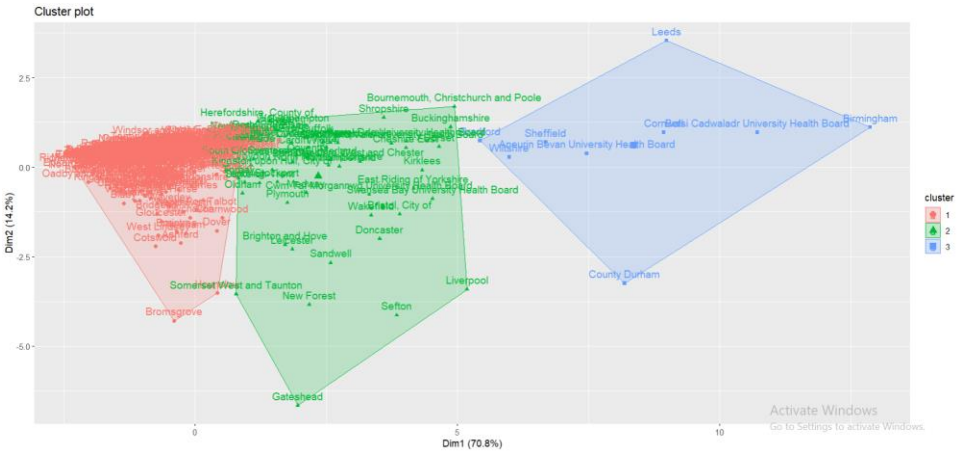
We performed k-means clustering on a 'Deaths_on_Eng_Data_pivot_' dataset with k=3, but without the Isles of Scilly, Eden, and Rutland.

```
Deaths_on_Eng_Data_pivot_n3=subset(Deaths_on_Eng_Data_pivot_n,
!(rownames(Deaths_on_Eng_Data_pivot_n) %in% c("City of London",

"Isles of Scilly",

"Eden",
```

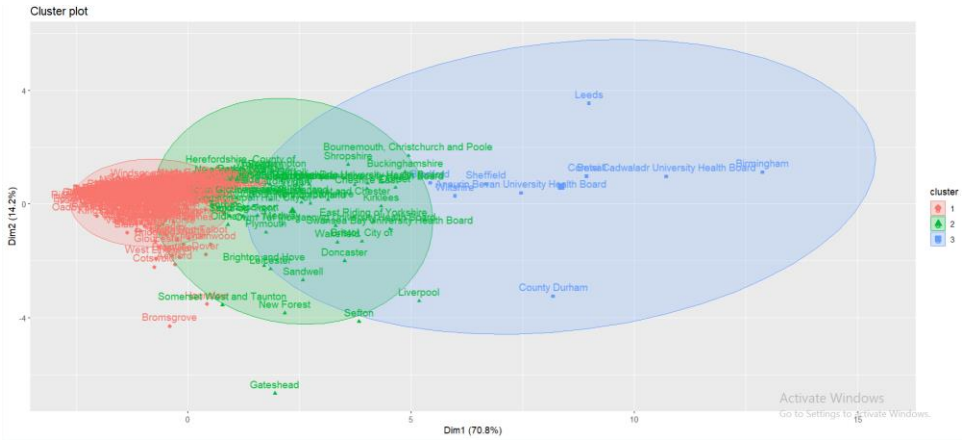
```
"Rutland")))
set.seed(130)
km.fit3 <- kmeans(Deaths_on_Eng_Data_pivot_n3, 3, nstart = 30)
km.fit3$cluster
```

Lastly, we used the fviz_cluster code to create a plot about the death count in England and Wales.

```
km.fit3$size

## [1] 265  60   9

fviz_cluster(km.fit3,Deaths_on_Eng_Data_pivot_n3)
```




```
fviz_cluster(km.fit3,Deaths_on_Eng_Data_pivot_n3,ellipse.type = "norm")
```



Results analysis and discussion.

In this dataset, we have obtained some results of clustering data in data mining. After we applied our dataset to clustering data mining, we could get some results of clustered data in data mining. If simply said, we had been able to divide our dataset about the death count of England and Wales. It was divided into mainly three categories and using that data we could accomplish our expectations using this dataset.

Our goal was to use this dataset to get a better knowledge about the data distribution of England and Wales. We also had an idea about how to apply this England and Wales death deference according to the regions. I think we obtain accurate results from this data set too. If we could develop this idea, we could make a big impact on the England and Wales healthcare system by understanding the frequent types of death cusses around region-wise and developing the healthcare system accordingly.

Conclusions.

Cluster analysis is widely used in many areas. As a rule, clustering is considered unsupervised learning because it has no class label or a quantitative response variable that is present in supervised learning such as classification and regression. Clustering is dividing data into similar object groups. Clustering plays an outstanding role in data mining applications such as scientific data exploration, information retrieval, text mining, and medical diagnostics.

References.

Death registrations and occurrences by local authority and health board (22 June 2021). Office for national statistics.

https://www.ons.gov.uk/peoplepopulationandcommunity/healthandsocialcare/causesofdeath/datasets/deathregistrationsandoccurrencesbylocalauthorityandhealthboard

Appling association rules in data mining to market basket analysis.

Task 3 - Association rule mining.

Introduction.

Association rule mining is an unsupervised non-linear algorithm in R language to understand the depth of some item's association. By using this algorithm, we can understand how some goods depend on each other's sales, or we also could predict what a customer would buy after they bought one item. There would also be times when a company's sales department would easily understand the minds of their customers by using this association rule of data mining.

Therefore, we plan to use the Market Basket Analysis Data structure from Kaggle.com and use this dataset to understand how the demand changes with the choice of their foods. Also, we plan to use this data to understand the depth of association rule mining in data mining. From now on we are going to understand how we used this dataset and what kind of approach we took with this dataset.

Datasets.

Here we are going to discuss the dataset we are going to use. First, this dataset is a dataset that contains market goods that we buy from the everyday market. And therefore, it has some market goods like corn, eggs, apples, ice cream, etc. so I think from this data we could not only understand the customers' usual foods and brewages they normally use in day-to-day life but also, I think we could gain a little bit of knowledge about their income and expenditure and their lifestyle. So, I think this data set would be much more useful than predicting the data set's value for the first time.

Explanation and preparation of datasets.

Explanation: -

Maret basket analysis data set is a dataset that contains the data of common goods we normally buy from the store. As a customer, when we go to a store and buy some goods from it there are some goods we always buy and there are some goods we buy from time to time. As an example, when we go to a store to buy necessities like rice, dal, and potato. But sometimes we brought things other than necessary things like ice cream and chocolate.

So, by using this dataset we plan to understand what the necessary goods are and what are the goods that customers buy rather than necessary. Also, we plan to increase the sales of this store by understanding customers' needs and wants.

In this data set, it contains 1,0 data or if we investigate the data they contain only yes and no answers in this data set. As you can see this is an example table extracted from the dataset.

TRUE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE	TRUE	FALSE	TRUE	FALSE	FALSE
FALSE	FALSE	TRUE	TRUE	FALSE	TRUE	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
TRUE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
TRUE	TRUE	TRUE	TRUE	FALSE	TRUE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE
FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
TRUE	FALSE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	TRUE	FALSE
TRUE	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	FALSE	TRUE	TRUE	TRUE
TRUE	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	FALSE	TRUE	FALSE	TRUE
TRUE	TRUE	FALSE	TRUE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE
TRUE	TRUE	FALSE	TRUE	FALSE	TRUE	TRUE	TRUE	TRUE	FALSE	TRUE	FALSE

Almost all the data in this dataset are dependent values. Because all those yes and no answers came from the customer's choice, therefore all the values in this dataset are dependent values.

And now we are going to talk about the columns in this dataset. There are sixteen columns in this dataset and all of them are the goods the customers bought from the store.

Apple, Bread, Butter, Cheese, Corn, Drill, Eggs, Ice cream, Milk, Nutmeg, Onion, Sugar, Unicorn, Yogurt, Chocolate.

	Apple	Bread	Butter	Cheese	Corn	Dill	Eggs	Ice cream	Kidney Bea	Milk	Nutmeg	Onion	Sugar	Unicorn	Yogurt	chocolate
2	FALSE	TRUE	FALSE	FALSE	TRUE	TRUE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	TRUE	TRUE
3	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
4	TRUE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE	TRUE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE
5	FALSE	FALSE	TRUE	TRUE	FALSE	TRUE	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE	FALSE
6	TRUE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
7	TRUE	TRUE	TRUE	TRUE	FALSE	TRUE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE	FALSE	TRUE	TRUE	TRUE
8	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE	TRUE	FALSE
9	TRUE	FALSE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	TRUE	FALSE	TRUE	FALSE	TRUE	FALSE

Preparation of datasets: -

In this dataset we didn't have too much work to do to prepare this dataset. Because it has close to zero null values and there isn't much to do with normalizing this data set either. Therefore, we did not have to do anything other than understand the data in this data set and apply the association rule of data mining into the dataset using R software.

Implementation in R.

R is a language and an environment that is used to create statistical computing and graphics. We use this environment to apply the association rule of data mining. Before we started the part of data mining, we downloaded some R packages to make our R environment possible to do the association rule of data mining.

This is the r code to get the head of each column. So, when we use this code, we can get a small idea about the overall dataset. It would also help us to understand what kind of data we have in this dataset.

```
marketbasket <-read.csv("basket_analysis.csv",header=T, colClasses="factor")

names(marketbasket)

##  [1] "Apple"        "Bread"        "Butter"       "Cheese"       "Corn"
##  [6] "Dill"         "Eggs"         "Ice.cream"    "Kidney.Beans" "Milk"
## [11] "Nutmeg"       "Onion"        "Sugar"        "Unicorn"      "Yogurt"
## [16] "chocolate"
```

This is the 'head' code we use in r and by using this code we obtain the result of the first few columns in the dataset in this case we got the first six columns in the dataset. This r code is a great help to understand the datatypes on the table and to understand the distribution of this dataset.

```
head(marketbasket)

##   Apple Bread Butter Cheese  Corn  Dill  Eggs Ice.cream Kidney.Beans  Milk
## 1 FALSE  TRUE  FALSE  FALSE  TRUE  TRUE FALSE      TRUE        FALSE FALSE
## 2 FALSE FALSE  FALSE  FALSE FALSE FALSE FALSE     FALSE        FALSE  TRUE
## 3  TRUE FALSE   TRUE  FALSE FALSE  TRUE FALSE      TRUE        FALSE  TRUE
## 4 FALSE FALSE   TRUE   TRUE FALSE  TRUE FALSE     FALSE        FALSE  TRUE
## 5  TRUE  TRUE  FALSE  FALSE FALSE FALSE FALSE     FALSE        FALSE FALSE
## 6  TRUE  TRUE   TRUE   TRUE FALSE  TRUE FALSE      TRUE        FALSE FALSE
##   Nutmeg Onion Sugar Unicorn Yogurt chocolate
## 1  FALSE FALSE  TRUE   FALSE   TRUE      TRUE
## 2  FALSE FALSE FALSE   FALSE  FALSE     FALSE
## 3  FALSE FALSE FALSE   FALSE   TRUE      TRUE
## 4   TRUE  TRUE FALSE   FALSE  FALSE     FALSE
## 5  FALSE FALSE FALSE   FALSE  FALSE     FALSE
## 6   TRUE FALSE FALSE    TRUE   TRUE      TRUE
```

This is the opposite code for the 'head' code, or it gives us the opposite result for the head code. in that case this we received the last few rows in the dataset.

```
tail(marketbasket)

##     Apple Bread Butter Cheese  Corn  Dill  Eggs Ice.cream Kidney.Beans
Milk
## 994 FALSE FALSE   TRUE  FALSE FALSE  TRUE FALSE     FALSE        FALSE
FALSE
## 995 FALSE  TRUE  FALSE  FALSE FALSE FALSE  TRUE     FALSE        FALSE
FALSE
## 996  TRUE FALSE  FALSE  FALSE  TRUE FALSE FALSE     FALSE         TRUE
TRUE
## 997  TRUE FALSE  FALSE  FALSE  TRUE  TRUE FALSE     FALSE        FALSE
FALSE
## 998 FALSE FALSE   TRUE   TRUE  TRUE FALSE  TRUE      TRUE         TRUE
FALSE
## 999 FALSE FALSE  FALSE  FALSE FALSE FALSE FALSE     FALSE        FALSE
TRUE
```

We used this r function to display the structure of the dataset we are going use in this project.

```
str(marketbasket)

## 'data.frame':    999 obs. of  16 variables:
##  $ Apple       : Factor w/ 2 levels "FALSE","TRUE": 1 1 2 1 2 2 1 2 2 2
...
##  $ Bread       : Factor w/ 2 levels "FALSE","TRUE": 2 1 1 1 2 2 1 1 1 1
...
##  $ Butter      : Factor w/ 2 levels "FALSE","TRUE": 1 1 2 2 1 2 2 1 1 1
...
##  $ Cheese      : Factor w/ 2 levels "FALSE","TRUE": 1 1 1 2 1 2 1 2 1 1
...
##  $ Corn        : Factor w/ 2 levels "FALSE","TRUE": 2 1 1 1 1 1 1 1 2 1
...
##  $ Dill        : Factor w/ 2 levels "FALSE","TRUE": 2 1 2 2 1 2 1 1 2 2
...
##  $ Eggs        : Factor w/ 2 levels "FALSE","TRUE": 1 1 1 1 1 1 2 2 2 2
...
##  $ Ice.cream   : Factor w/ 2 levels "FALSE","TRUE": 2 1 2 1 1 2 2 1 2 2
...
##  $ Kidney.Beans: Factor w/ 2 levels "FALSE","TRUE": 1 1 1 1 1 1 2 1 1 1
...
##  $ Milk        : Factor w/ 2 levels "FALSE","TRUE": 1 2 2 2 1 1 2 1 2 2
...
##  $ Nutmeg      : Factor w/ 2 levels "FALSE","TRUE": 1 1 1 2 1 2 2 2 2 1
...
##  $ Onion       : Factor w/ 2 levels "FALSE","TRUE": 1 1 1 2 1 1 2 1 2 2
...
##  $ Sugar       : Factor w/ 2 levels "FALSE","TRUE": 2 1 1 1 1 1 1 2 2 2
...
##  $ Unicorn     : Factor w/ 2 levels "FALSE","TRUE": 1 1 1 1 1 2 1 1 2 2
...
##  $ Yogurt      : Factor w/ 2 levels "FALSE","TRUE": 2 1 2 1 1 2 2 2 2 1
```

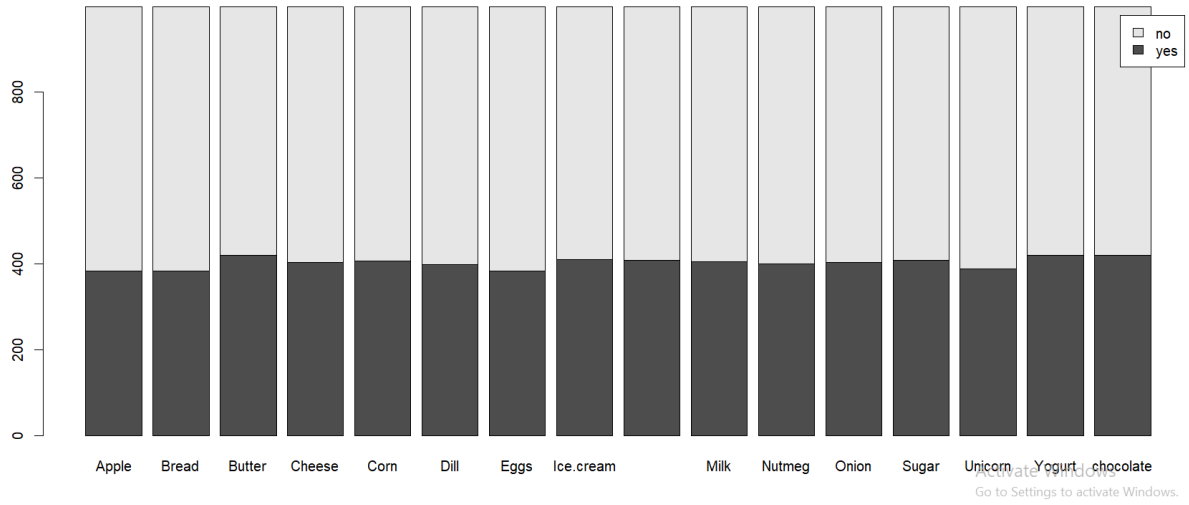
We used this ‘yes= colSums’ r code to get the number of yes variables in each column. And next we used ‘no = colSums’ code to get the number of ‘No’ variables in each column.

```
yes = colSums(marketbasket == "TRUE")
yes

##        Apple        Bread       Butter       Cheese         Corn
Dill
##          383          384          420          404          407
398
##         Eggs    Ice.cream Kidney.Beans         Milk       Nutmeg
Onion
##          384          410          408          405          401
403
##        Sugar      Unicorn       Yogurt    chocolate
##          409          389          420          421
```

```
no = colSums(marketbasket=="FALSE")
no

##        Apple        Bread       Butter       Cheese         Corn
Dill
##          616          615          579          595          592
601
##         Eggs    Ice.cream Kidney.Beans         Milk       Nutmeg
Onion
##          615          589          591          594          598
596
##        Sugar      Unicorn       Yogurt    chocolate
##          590          610          579          578
```

In hear we created a bar plot to have an overall idea of this dataset. Hear is the code for the bar plot. And after that, there is the bar plot that represents the yes and no data of the dataset.

```
barplot(purchased,legend=rownames(purchased)) #Plot 1
```



Data mining.

The Association rule of data mining was Proposed by Agrawal in 1993. The Association rule of data mining is an important data mining model studied extensively by the database and data mining community. In this concept of data mining, it was to assume all data in a dataset could be categorical. In this association rule of data mining, there wasn't any good algorithm for numeric data. Most of the time this was used for Market Basket Analysis to find how items purchased by customers are related. When we used this rule the result of this rule was to give transactions with multiple items, it tries to find the rules that govern how or why such items are often bought together.

In this project, we used some packages to make it possible to get the final result. So, at first, we used the “arules” package which used Apriori Algorithm. The Apriori Algorithm is an influential algorithm for mining frequent item sets for Boolean association rules. Therefore, we had to install this package using the ‘install.packages(“arules”)’ code.

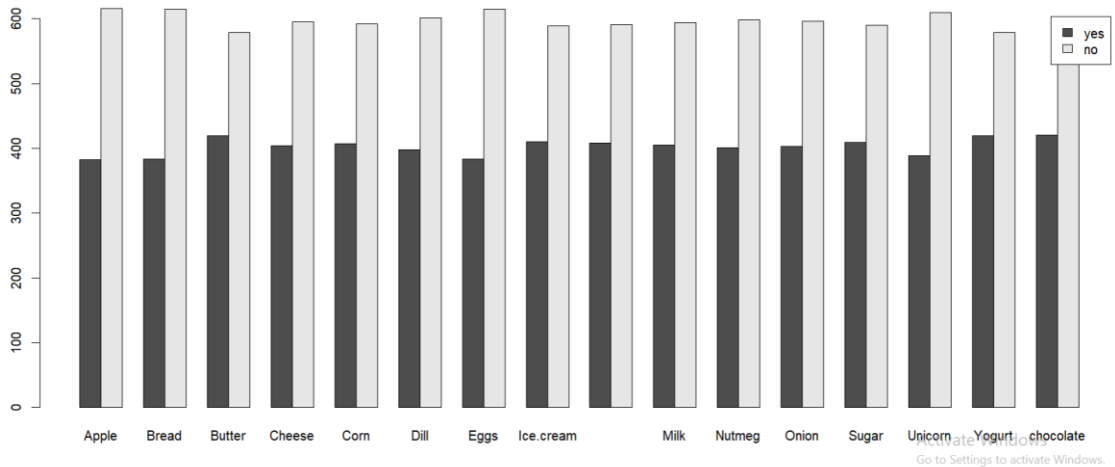
We also had to install the arulesViz package which provides various visualization techniques for association rules and item sets. So, we used ‘install.packages(“arulesViz”)’ code to install this package.

We used the ‘apriori ()’ function from the “arule” package to implement the Apriori algorithm to create frequent item sets. This feature performs all iterations at the same time.

```
rules <- apriori(marketbasket,   parameter = list(minlen=2, maxlen=3,conf =
0.4), appearance= list(rhs=c("chocolate=TRUE"),default="lhs"))

## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.4    0.1    1 none FALSE            TRUE       5     0.1      2
##  maxlen target  ext
##       3  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
```

Then, the rules are summarized and inspected and Next to find the most popular products bar plot is plotted.

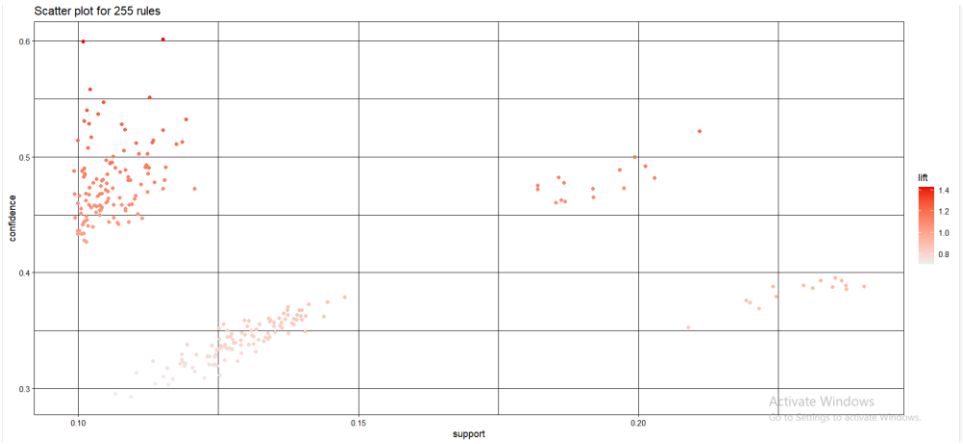
```
barplot(purchased, beside=T,legend=rownames(purchased))
```



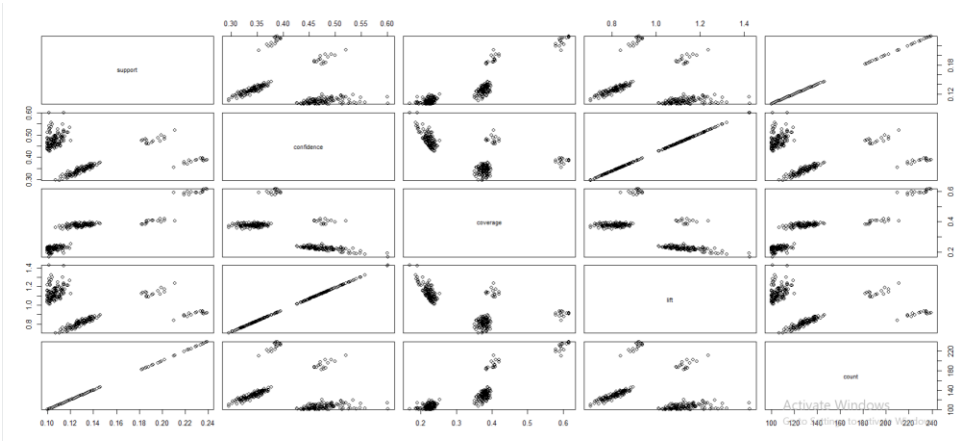
According to the plot we can see that chocolate is the most popular item. The following code is used to see the rules by which customers purchase more items.

```
rules <- apriori(marketbasket,   parameter = list(minlen=2, maxlen=3,conf = 0.4), appearance= list(rhs=c("chocolate=TRUE"),default="lhs"))
```

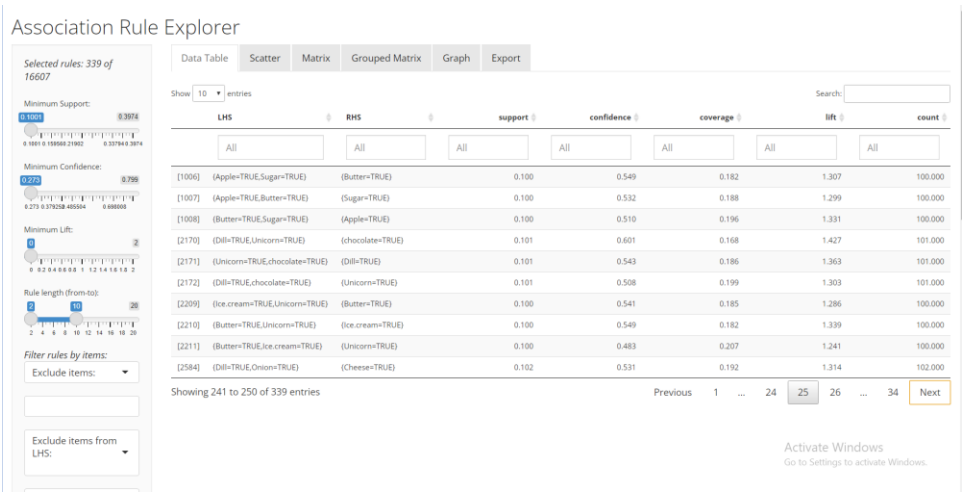
Then we visualized the rules using the "arulesViz" package. This package provides a variety of visualization techniques for association rules and sets of items.



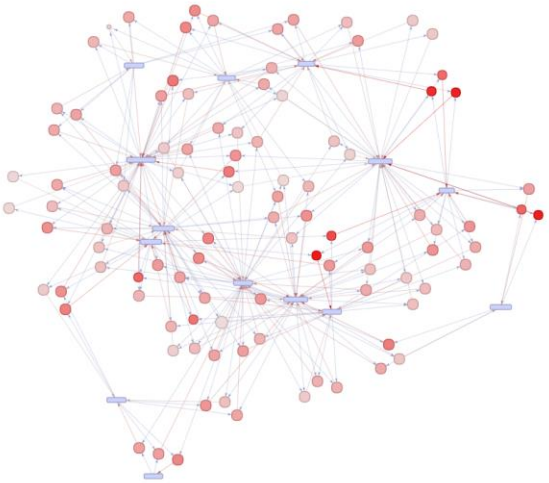
The scatter plot matrix is then used to compare support with the goods.



Then we used the rule Explorer function to explore association rules using interactive manipulations and viewing using shiny.



Items which have false values are selected from the right-hand side and left-hand side to exclude because in most cases any rules which contain purchased items “No” cannot be seen.



Now we can get a general idea about what we have done using R language and other data mining concepts we used.

Results analysis and discussion.

So, far after we used the association rule of data mining in R software with visualization tools and other tools and obtain our results as we created them. In hear we are going to look at the result we obtain from using our dataset in R software by using the association rule in data mining. First, we are going to get to know about the result we obtained from using this dataset.

	B	C	D	E	F	G
	rules	support	confidenc	coverage	lift	count
1	{Apple=TRUE} => {Bread=TRUE}	0.154154	0.402089	0.383383	1.046059	154
2	{Bread=TRUE} => {Apple=TRUE}	0.154154	0.401042	0.384384	1.046059	154
3	{Apple=TRUE} => {Eggs=TRUE}	0.156156	0.407311	0.383383	1.059644	156
4	{Eggs=TRUE} => {Apple=TRUE}	0.156156	0.40625	0.384384	1.059644	156
5	{Apple=TRUE} => {Unicorn=TRUE}	0.166166	0.43342	0.383383	1.113077	166
6	{Unicorn=TRUE} => {Apple=TRUE}	0.166166	0.426735	0.389389	1.113077	166
7	{Apple=TRUE} => {Dill=TRUE}	0.179179	0.467363	0.383383	1.173104	179
8	{Dill=TRUE} => {Apple=TRUE}	0.179179	0.449749	0.398398	1.173104	179
9	{Apple=TRUE} => {Nutmeg=TRUE}	0.172172	0.449086	0.383383	1.118796	172
10	{Nutmeg=TRUE} => {Apple=TRUE}	0.172172	0.428928	0.401401	1.118796	172
11	{Apple=TRUE} => {Onion=TRUE}	0.167167	0.436031	0.383383	1.080882	167
12	{Onion=TRUE} => {Apple=TRUE}	0.167167	0.414392	0.403403	1.080882	167
13	{Apple=TRUE} => {Cheese=TRUE}	0.162162	0.422977	0.383383	1.045925	162
14	{Cheese=TRUE} => {Apple=TRUE}	0.162162	0.40099	0.404404	1.045925	162
15	{Apple=TRUE} => {Milk=TRUE}	0.184184	0.480418	0.383383	1.18503	184
16	{Milk=TRUE} => {Apple=TRUE}	0.184184	0.454321	0.405405	1.18503	184
17	{Apple=TRUE} => {Corn=TRUE}	0.186186	0.48564	0.383383	1.192025	186
18	{Corn=TRUE} => {Apple=TRUE}	0.186186	0.457002	0.407407	1.192025	186
19	{Apple=TRUE} => {Kidney.Beans=TRUE}	0.176176	0.45953	0.383383	1.125173	176
20	{Kidney.Beans=TRUE} => {Apple=TRUE}	0.176176	0.431373	0.408408	1.125173	176
21	{Apple=TRUE} => {Sugar=TRUE}	0.182182	0.475196	0.383383	1.160686	182
22	{Sugar=TRUE} => {Apple=TRUE}	0.182182	0.444988	0.409409	1.160686	182
23	{Apple=TRUE} => {Ice.cream=TRUE}	0.172172	0.449086	0.383383	1.094237	172
24	{Ice.cream=TRUE} => {Apple=TRUE}	0.172172	0.419512	0.41041	1.094237	172
25	{Apple=TRUE} => {Butter=TRUE}	0.188188	0.490862	0.383383	1.167549	188

Above picture if customer buy an apple is 40% therefore, we can confidently say in this script customer will buy bread 40% or we can confidently say that the customer will buy eggs, 42% confidence can say customer will buy cheese, 44% confidently can say customer will buy ice cream and 49.08% confidently can say customer will buy butter.

In the implementation of R different rules were minded finding and check how items are purchased by customers. According to the dataset, chocolate is the most popular item in the basket.

1006	{Apple=TRUE,Sugar=TRUE} => {Butter=TRUE}	0.1001	0.549451	0.182182	1.306907	100
1007	{Apple=TRUE,Butter=TRUE} => {Sugar=TRUE}	0.1001	0.531915	0.188188	1.299225	100
1008	{Butter=TRUE,Sugar=TRUE} => {Apple=TRUE}	0.1001	0.510204	0.196196	1.330793	100
2170	{Dill=TRUE,Unicorn=TRUE} => {chocolate=TRUE}	0.101101	0.60119	0.168168	1.426578	101
2171	{Unicorn=TRUE,chocolate=TRUE} => {Dill=TRUE}	0.101101	0.543011	0.186186	1.362984	101
2172	{Dill=TRUE,chocolate=TRUE} => {Unicorn=TRUE}	0.101101	0.507538	0.199199	1.303419	101
2209	{Ice.cream=TRUE,Unicorn=TRUE} => {Butter=TRUE}	0.1001	0.540541	0.185185	1.285714	100
2210	{Butter=TRUE,Unicorn=TRUE} => {Ice.cream=TRUE}	0.1001	0.549451	0.182182	1.338783	100
2211	{Butter=TRUE,Ice.cream=TRUE} => {Unicorn=TRUE}	0.1001	0.483092	0.207207	1.240639	100
2584	{Dill=TRUE,Onion=TRUE} => {Cheese=TRUE}	0.102102	0.53125	0.192192	1.31366	102
2585	{Cheese=TRUE,Dill=TRUE} => {Onion=TRUE}	0.102102	0.576271	0.177177	1.428523	102
2586	{Cheese=TRUE,Onion=TRUE} => {Dill=TRUE}	0.102102	0.551351	0.185185	1.38392	102
2587	{Dill=TRUE,Onion=TRUE} => {chocolate=TRUE}	0.103103	0.536458	0.192192	1.272974	103
2588	{Dill=TRUE,chocolate=TRUE} => {Onion=TRUE}	0.103103	0.517588	0.199199	1.283053	103
2589	{Onion=TRUE,chocolate=TRUE} => {Dill=TRUE}	0.103103	0.52551	0.196196	1.319057	103
2611	{Dill=TRUE,Milk=TRUE} => {chocolate=TRUE}	0.114114	0.6	0.19019	1.423753	114
2612	{Dill=TRUE,chocolate=TRUE} => {Milk=TRUE}	0.114114	0.572864	0.199199	1.413065	114
2613	{Milk=TRUE,chocolate=TRUE} => {Dill=TRUE}	0.114114	0.540284	0.211211	1.356141	114
2656	{Dill=TRUE,Ice.cream=TRUE} => {chocolate=TRUE}	0.103103	0.556757	0.185185	1.32114	103
2657	{Dill=TRUE,chocolate=TRUE} => {Ice.cream=TRUE}	0.103103	0.517588	0.199199	1.261147	103
2658	{Ice.cream=TRUE,chocolate=TRUE} => {Dill=TRUE}	0.103103	0.509901	0.202202	1.279877	103
3037	{Nutmeg=TRUE,Onion=TRUE} => {Cheese=TRUE}	0.1001	0.512821	0.195195	1.268088	100
3038	{Cheese=TRUE,Nutmeg=TRUE} => {Onion=TRUE}	0.1001	0.520833	0.192192	1.291098	100
3039	{Cheese=TRUE,Onion=TRUE} => {Nutmeg=TRUE}	0.1001	0.540541	0.185185	1.346633	100

Above this picture if customer buy apple and sugar together script say 54.94% confidently, he will buy butter in row 1006. And also, if the customer buys milk and chocolate together script say 54.02% confidently, they he will buy dill in row 2613.

In implementing R, different rules were to find and check how items are purchased by customers. Chocolate is the most popular item in the basket, according to the data set. Finally, by using the rule explorer function all the false values were executed because when we purchase an item false or not cannot be used. Thus, the graph is the result of all the true values. So, if we apply this method to understand the sales of the store and use this method to find the relationship between the goods, we think it would be helpful for increasing huge part of efficiency in a store.

Conclusion.

Association rule in data mining is mostly used to understand the relationship between two or values. Therefore, we applied this rule to a Convenience store's customers' choice of buying goods and other brewages. By using this method, we plan to understand the relationship between each of the goods customers brought from the store and use that knowledge to understand when a customer brought goods, what kind of goods he was going to buy or else we can simply say what are the goods that have positive relationship between each other and what are the goods that have negative relationship between each other.

We also had the idea of using this method to increase the sales of this store. For example, if we take bread and butter, if we give a discount for butter the customers would buy more bread to use butter. Because of this positive relationship between bread and butter, the sales at this store will increase. e. This is why we plan to use this association rule for data mining to increase the sales of the store. If we investigate the bigger picture, we can use this method for more than a Convenience we could use this on supermarket stores like keels or Kargels they could increase their sales rate and increase efficiency more easily.

References.

Market Basket Analysis Data (2021, May 28). Kaggle.

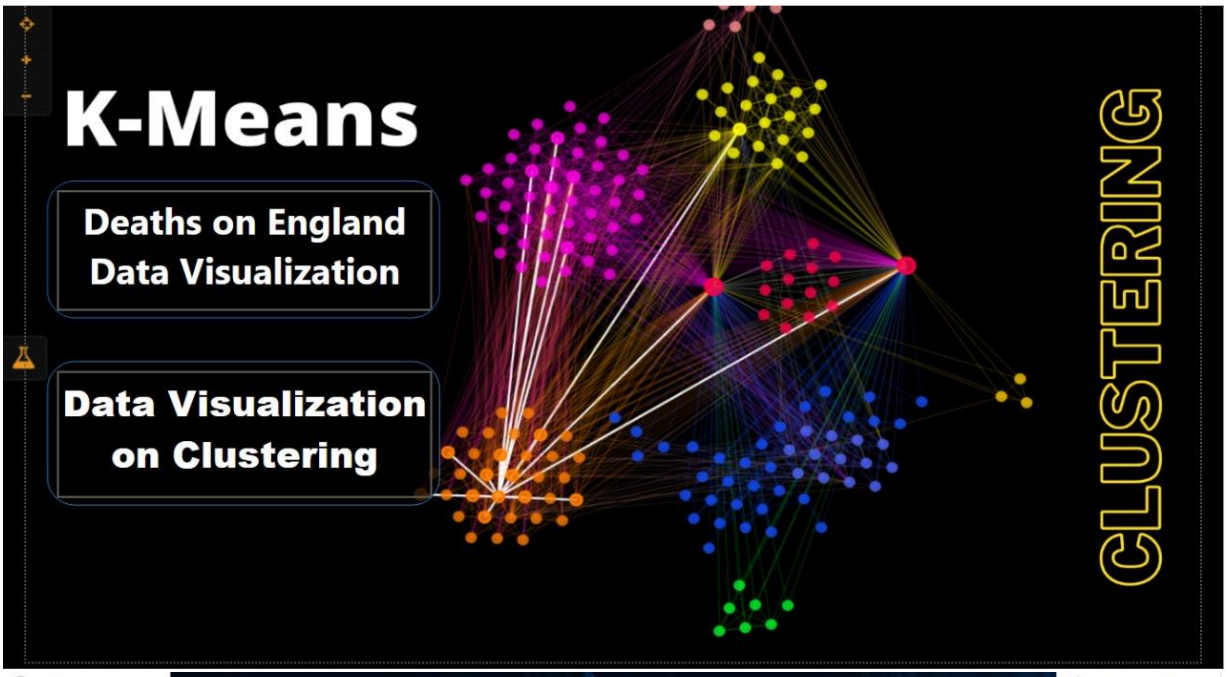
https://www.kaggle.com/datasets/ahmtcnbs/datasets-for-appiori?select=basket_analysis.csv

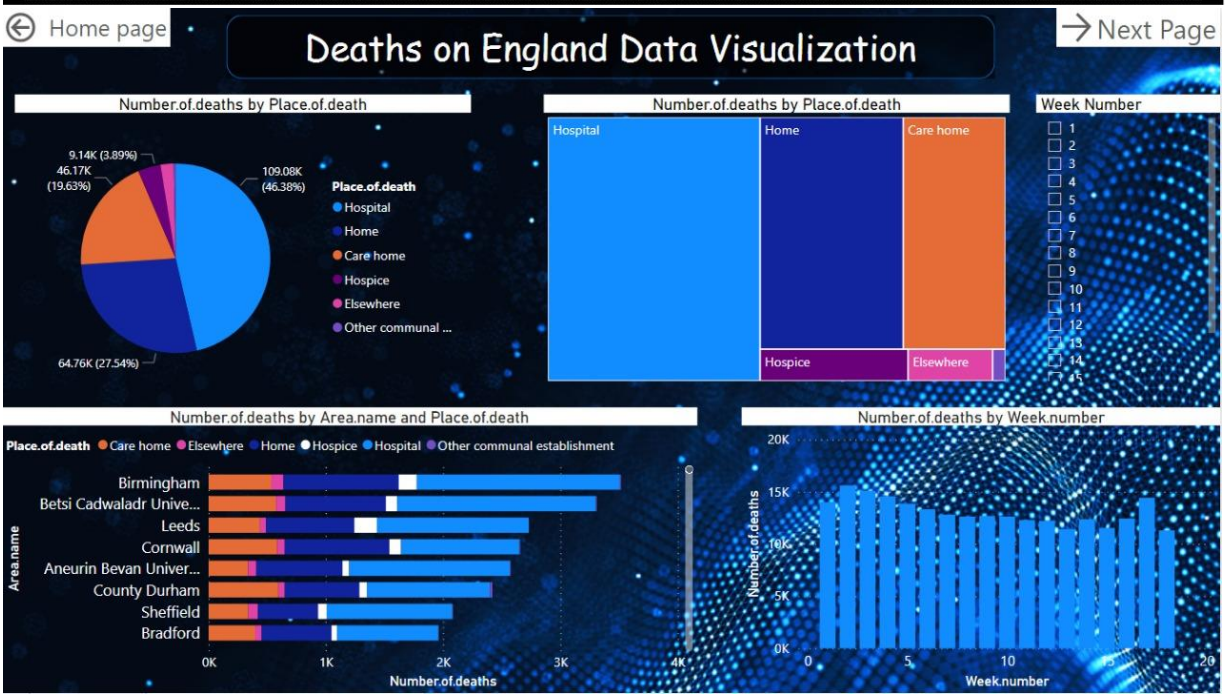
Appling classification in data mining to Power BI visualization tool.

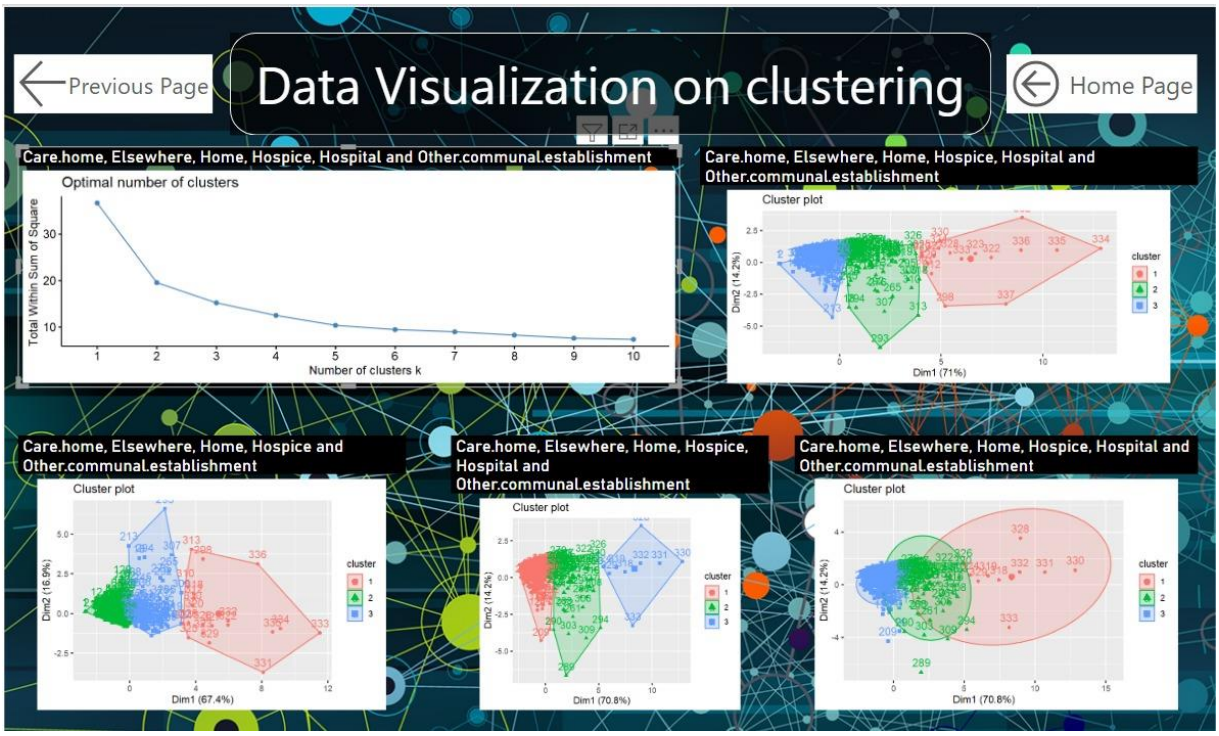
Task4 – Power BI.

Power BI Work.

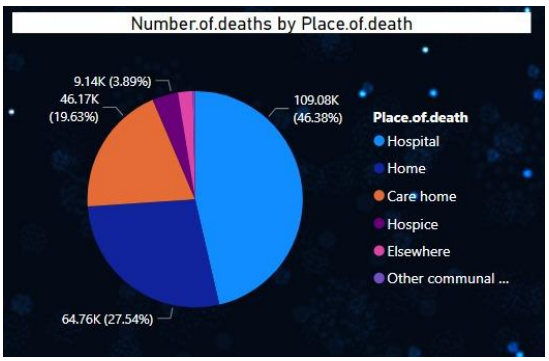
To precent the clustering in data mining on Death registrations and occurrences by local authority and health board in England and Wales. So, we are going to use a power bi tool to understand the patterns in deaths and then it would be easy to apply our work in it.

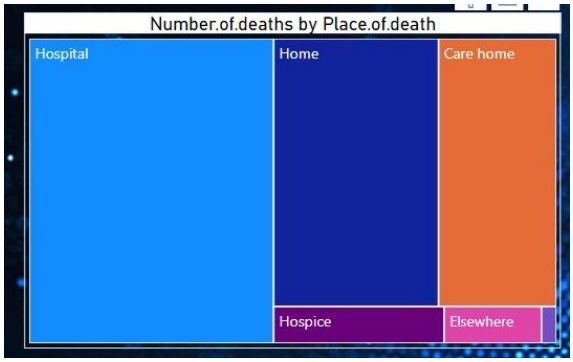

K-Means
Deaths on England Data Visualization
Data Visualization on Clustering
CLUSTERING


Home page
Deaths on England Data Visualization
Next Page
Number.of.deaths by Place.of.death
9.14K (3.89%)
46.17K (19.63%)
109.08K (46.38%)
64.76K (27.54%)
Place.of.death
Hospital
Home
Care home
Hospice
Elsewhere
Other communal ...
Number.of.deaths by Place.of.death
Hospital
Home
Care home
Hospice
Elsewhere
Week Number
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Number.of.deaths by Area.name and Place.of.death
Place.of.death Care home Elsewhere Home Hospice Hospital Other communal establishment
Birmingham
Betsi Cadwaladr Unive...
Leeds
Cornwall
Aneurin Bevan Univer...
County Durham
Sheffield
Bradford
Area.name
0K
1K
2K
3K
4K
Number.of.deaths
Number.of.deaths by Week.number
20K
15K
10K
5K
0K
Number.of.deaths
0
5
10
15
20
Week.number



This pie chart was showing us the count of death in England and Wales according to the places the death happened. So, as we can see, the majority of deaths happened in hospitals.

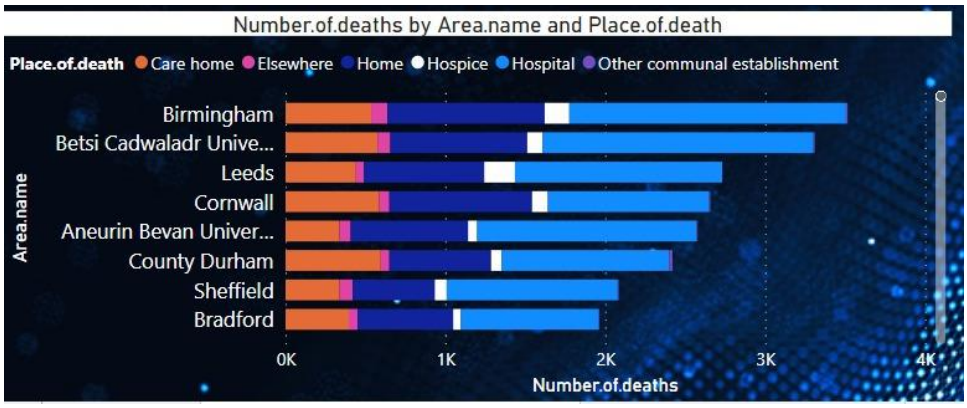




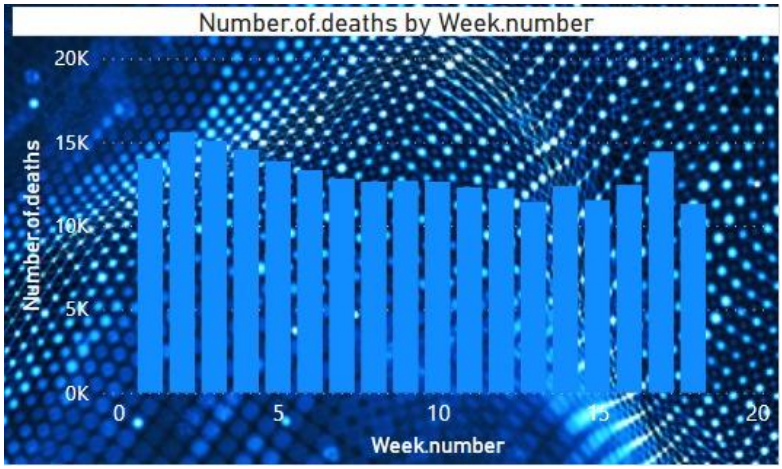
This chart shows the same result as the above chart but if we click on hospital or care home like that, we could have the data that refer to those deaths happened.

We used this visualization to obtain the result of each week's death count across the England and Wales with the place of death and the cities and added to their town names that the death happened was also there.

In this horizontal bar chart, it shows us the number of deaths that happened in each city like Birmingham and Leeds and then if we look into the horizontal axis, it shows us number death happened those cities. Lastly the color code is showing us the place of the death happened.



In this bar chart, it shows us simply the number of deaths that happened according to the weeks.



So, using this dataset we can understand the data visualization in England and Wales.

Now we are going to talk about data visualization on clustering. To do this part we used the Power BI data visualization part to create these plots.