UNIVERSITY OF
WESTMINSTER

MSC BUSINESS INTELLIGENCE & ANALYTICS

STUDENT NAME : RAJ PRAVIN RAJENDRAN

STUDENT ID : W1795435

SUBMISSION DATE : 24-11-2021

MODULE CODE : 7BUIS008W

MODULE TITLE : DATA MINING & MACHINE LEARNING

MODULE LEADER : DR. P.I. CHOUNTAS

ASSIGNMENT : COURSEWORK1

TABLE OF CONTENTS:

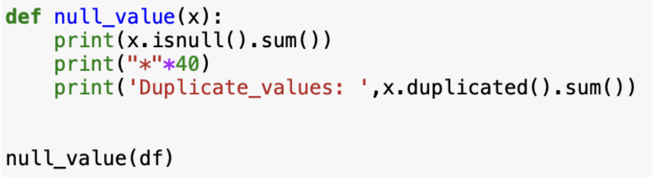
Task 1: Identify the main groups of countries and their global happiness characteristics using Clustering techniques

1. **Importing the libraries and DataFrame:**

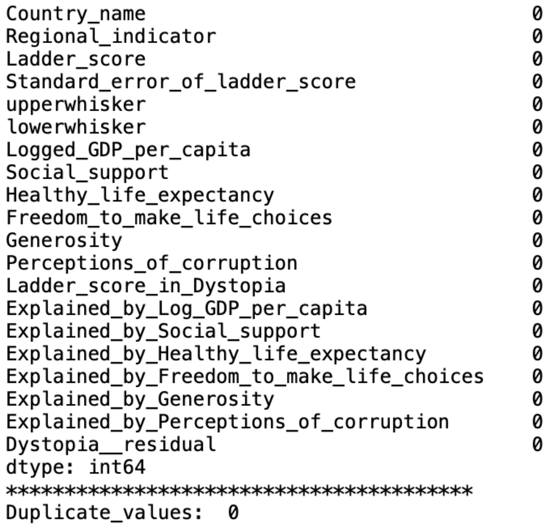
I have imported all the libraries which we need to apply for the below algorithms. I used pandas library to import the dataset into the jupyter notebook. Now the shape of the data frame is (149,20) means 149 rows and 20 columns.

2. **Identification and treatment of any Missing Values:**

I found that the data frame does not have missing and duplicate values.



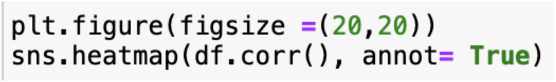
```
def null_value(x):
    print(x.isnull().sum())
    print("*"*40)
    print('Duplicate_values: ',x.duplicated().sum())


null_value(df)
```



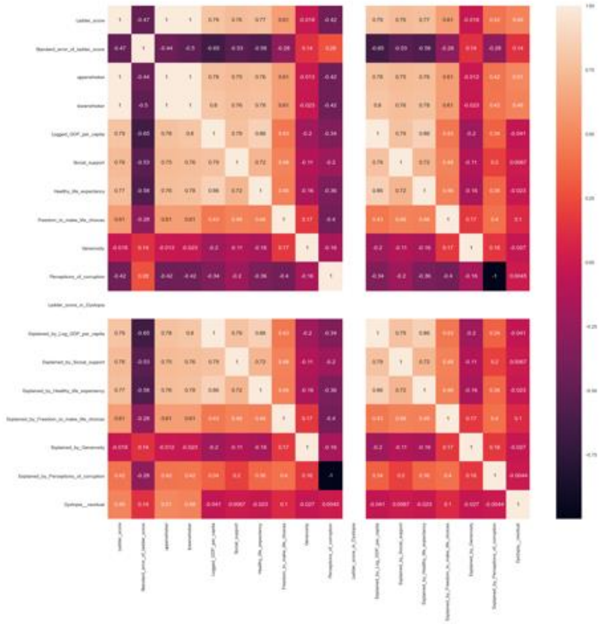
```
Country_name                                  0
Regional_indicator                            0
Ladder_score                                  0
Standard_error_of_ladder_score                0
upperwhisker                                  0
lowerwhisker                                  0
Logged_GDP_per_capita                         0
Social_support                                0
Healthy_life_expectancy                       0
Freedom_to_make_life_choices                  0
Generosity                                    0
Perceptions_of_corruption                     0
Ladder_score_in_Dystopia                      0
Explained_by_Log_GDP_per_capita               0
Explained_by_Social_support                   0
Explained_by_Healthy_life_expectancy          0
Explained_by_Freedom_to_make_life_choices     0
Explained_by_Generosity                       0
Explained_by_Perceptions_of_corruption        0
Dystopia__residual                            0
dtype: int64
****************************************
Duplicate_values:  0
```

3. Finding the corelation between all variables:

Heatmaps represent data in a two-dimensional format, with colours representing values and a two-dimensional plot providing an immediate visual representation of correlations between variables. To find the correlation between all the variables, we can use scatter plot as well as heat plot. I used a heatmap.

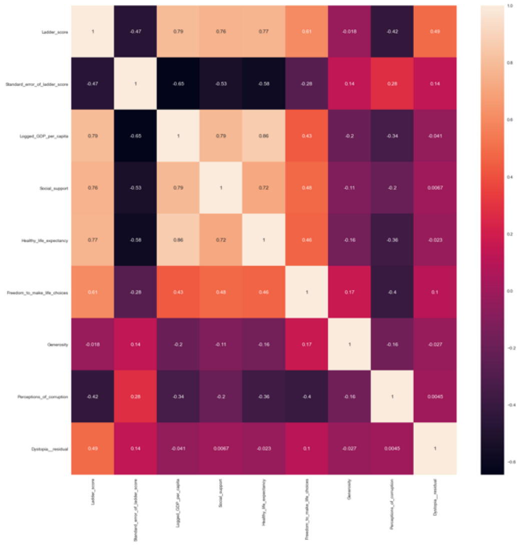


```
plt.figure(figsize =(20,20))
sns.heatmap(df.corr(), annot= True)
```



The graph clearly shows the correlation between each variable. The value near 1 and -1 have a strong correlation. Hence, we can eliminate the columns which have a strong correlation.

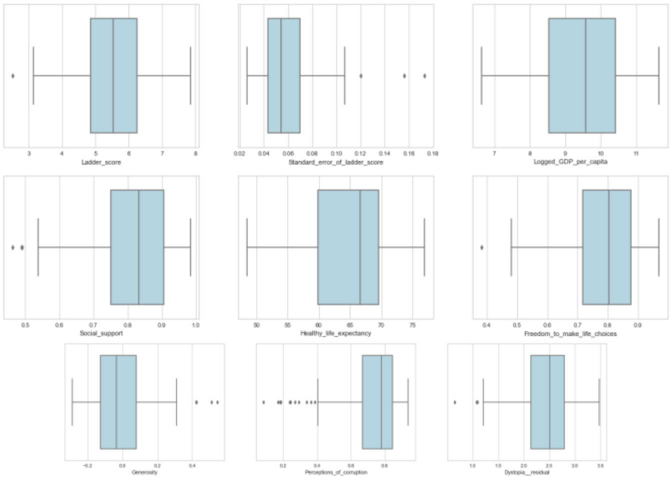
Furthermore, eliminating the columns, ['upperwhisker','lowerwhisker','Explained_by_Log_GDP_per_capita','Explained_by_Social_support','Explained_by_Healthy_life_expectancy','Explained_by_Freedom_to_make_life_choices','Explained_by_Generosity'].



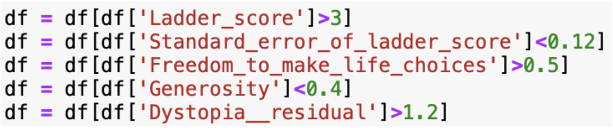
Post eliminating, we can see the variables healthy_life_expectancy and Logged_GDP_per_capita correlate about 86%. This can be eliminated if it is still greater than 90%.

4. Identification and treatment of any Outliers

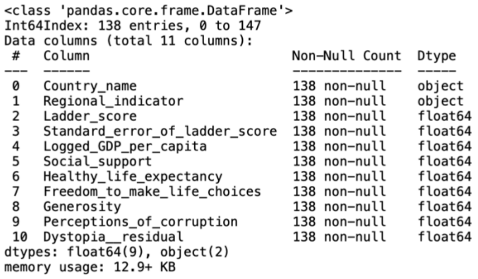
Statistical outliers are observations that differ significantly from other data points in a dataset. This can be caused by measurement errors and misinterpretation of filling in data points. A scatterplot and boxplot are two types of graphical methods you can use to detect outliers. I prefer a boxplot.



I found most of the columns have outliers, and I have eliminated them except the variable **Perceptions_of_corruption** since the two columns have more extreme values. If we remove the outliers, the data would be biased.



```
df = df[df['Ladder_score']>3]
df = df[df['Standard_error_of_ladder_score']<0.12]
df = df[df['Freedom_to_make_life_choices']>0.5]
df = df[df['Generosity']<0.4]
df = df[df['Dystopia__residual']>1.2]
```

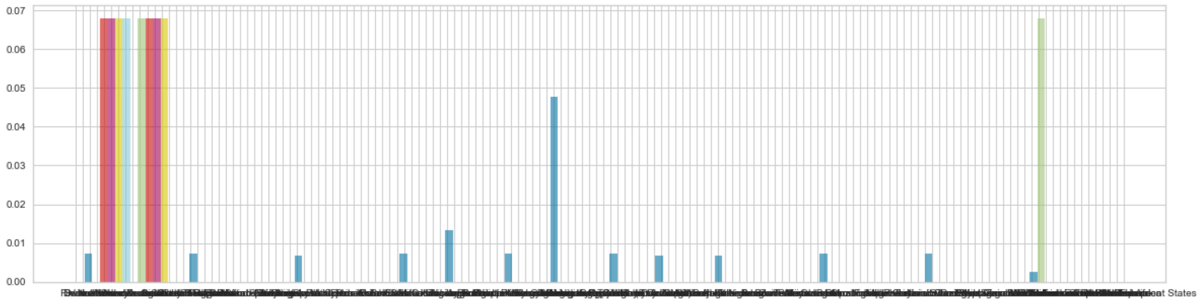


```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 138 entries, 0 to 147
Data columns (total 11 columns):
 #   Column                          Non-Null Count  Dtype  
---  ------                          --------------  -----  
 0   Country_name                    138 non-null    object 
 1   Regional_indicator              138 non-null    object 
 2   Ladder_score                    138 non-null    float64
 3   Standard_error_of_ladder_score  138 non-null    float64
 4   Logged_GDP_per_capita           138 non-null    float64
 5   Social_support                  138 non-null    float64
 6   Healthy_life_expectancy         138 non-null    float64
 7   Freedom_to_make_life_choices    138 non-null    float64
 8   Generosity                      138 non-null    float64
 9   Perceptions_of_corruption       138 non-null    float64
 10  Dystopia__residual              138 non-null    float64
dtypes: float64(9), object(2)
memory usage: 12.9+ KB
```

Subsequently, eliminating the outliers and the shape of the data has become **138 rows and columns are 10 from 149 rows and 10 columns.**

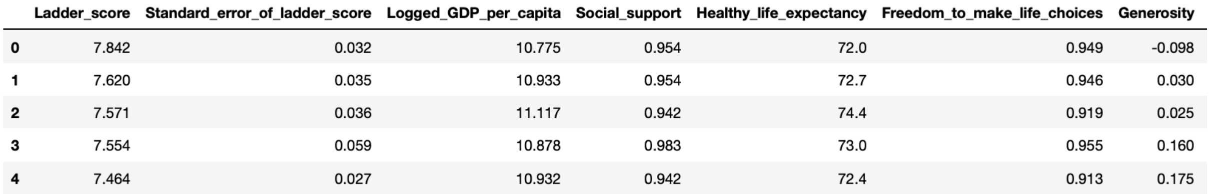
5. Data Normalisation

Normalization is a procedure frequently applied as a feature of information groundwork for Machine learning. The objective of normalization is to change the upsides of numeric segments in the dataset to utilize a typical scale, without contorting contrasts in the scopes of qualities or losing data.



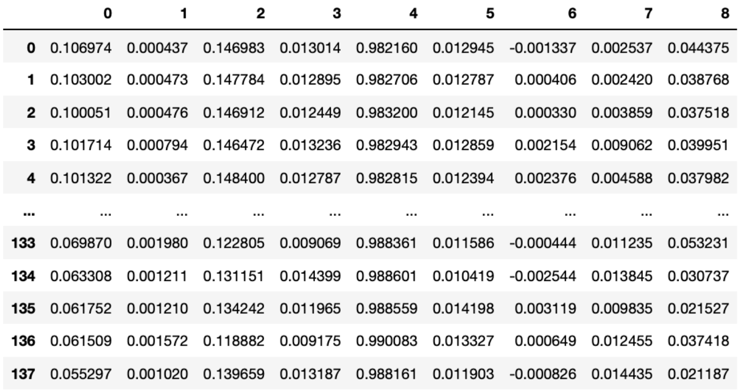
According to the standard, I should use normalization if I realize that the conveyance of the information doesn't follow a Gaussian distribution. I used the library matplotlib to plot the information. In this way, we can find whether the entire informational index is distributed normally. By seeing the chart, we get to realize that we can normalise the data since the data is not normally distributed.

To give equivalent significance to all elements, we really want to normalise the persistent elements. We utilized **sklearn preprocessing** to normalize the data. Below is the information which is introduced before the Normalization.



	Ladder_score	Standard_error_of_ladder_score	Logged_GDP_per_capita	Social_support	Healthy_life_expectancy	Freedom_to_make_life_choices	Generosity
0	7.842	0.032	10.775	0.954	72.0	0.949	-0.098
1	7.620	0.035	10.933	0.954	72.7	0.946	0.030
2	7.571	0.036	11.117	0.942	74.4	0.919	0.025
3	7.554	0.059	10.878	0.983	73.0	0.955	0.160
4	7.464	0.027	10.932	0.942	72.4	0.913	0.175

As seen in the below fig. the normalization values have changed.



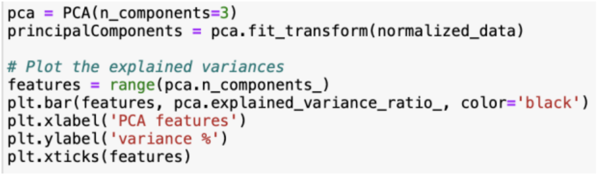
	0	1	2	3	4	5	6	7	8
0	0.106974	0.000437	0.146983	0.013014	0.982160	0.012945	-0.001337	0.002537	0.044375
1	0.103002	0.000473	0.147784	0.012895	0.982706	0.012787	0.000406	0.002420	0.038768
2	0.100051	0.000476	0.146912	0.012449	0.983200	0.012145	0.000330	0.003859	0.037518
3	0.101714	0.000794	0.146472	0.013236	0.982943	0.012859	0.002154	0.009062	0.039951
4	0.101322	0.000367	0.148400	0.012787	0.982815	0.012394	0.002376	0.004588	0.037982
...	...	...	...	...	...	...	...	...	...
133	0.069870	0.001980	0.122805	0.009069	0.988361	0.011586	-0.000444	0.011235	0.053231
134	0.063308	0.001211	0.131151	0.014399	0.988601	0.010419	-0.002544	0.013845	0.030737
135	0.061752	0.001210	0.134242	0.011965	0.988559	0.014198	0.003119	0.009835	0.021527
136	0.061509	0.001572	0.118882	0.009175	0.990083	0.013327	0.000649	0.012455	0.037418
137	0.055297	0.001020	0.139659	0.013187	0.988161	0.011903	-0.000826	0.014435	0.021187

6. Dimensionality Reduction Using PCA

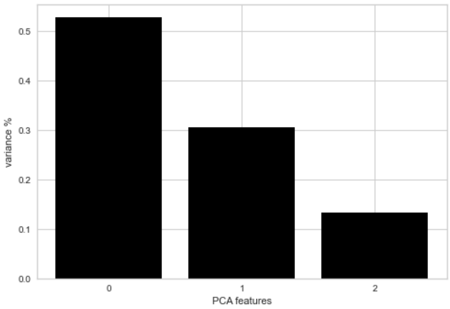
Dimensionality reduction involves **reducing the number of input variables or columns in modeling data**. PCA is a technique from linear algebra that can be used to automatically perform dimensionality reduction.

The initial data had 19 variables. We dropped the correlated variables and the total variable was 10. In this part, the code projects the first information which is 10 dimensional into 3 dimensions. I should take note that after dimensionality reduction, there generally isn't a specific significance allocated to every principal component. The new components are just the 3 main dimensions of variation.

Presently 3 dimensions appear to be entirely sensible. With 3 variables we can clarify more than 95% of the variety in the first information which is displayed in the underneath plot!



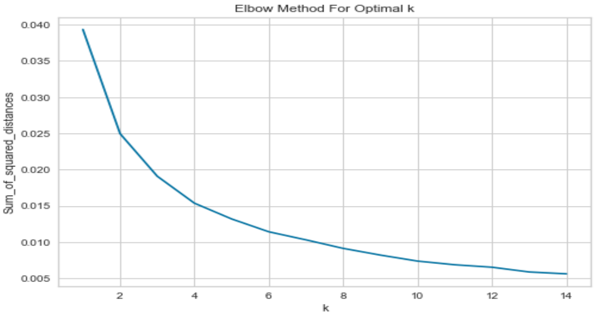
```
pca = PCA(n_components=3)
principalComponents = pca.fit_transform(normalized_data)

# Plot the explained variances
features = range(pca.n_components_)
plt.bar(features, pca.explained_variance_ratio_, color='black')
plt.xlabel('PCA features')
plt.ylabel('variance %')
plt.xticks(features)
```



The Eigen Value is denoted by **pca.explained_variance_ratio_** which is, array([0.52702324, 0.30538928, 0.13355203])
The percentage of variation explained by each eigen Vector is denoted by pca.components_

7. Finding the ideal number of clusters using Elbow method :

In cluster analysis, the elbow method is a heuristic utilized in deciding the number of clusters in an informational collection. The technique comprises of plotting the clarified variety as an element of the number of groups and picking the elbow of the curve as the number of clusters to utilize.



For every K value, I initialized k-means and utilize the inertia attribute to distinguish the sum of squared distances of samples to the closest group community. As K increases, the sum of squared distance tends to zero. Imagine we set K to its greatest worth n (where n is a number of samples) each example will shape its own bunch importance amount of squared distances rises to nothing. Above is a plot of the sum of squared distances for K in the range indicated previously. Assuming the plot resembles an arm, the elbow on the arm is ideal K.

The above chart says that the cluster can be 2 since the curve starts bends at 2. In any case, the silhouette_score is utilized to find the ideal clusters. Higher the value of silhoutte score, better the cluster. To find the silhouette_score we utilized the library silhouette_score from sklearn.metrics

8.Finding the ideal number of clusters using Silhouette method :

Silhouette score refers to a method of interpretation and validation of consistency within clusters of data. The technique provides a succinct graphical representation of how well each object has been classified.

The silhouette value is a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation). The silhouette ranges from −1 to +1, where a high value indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters. If most objects have a high value, then the clustering configuration is appropriate. If many points have a low or negative value, then the clustering configuration may have too many or too few clusters.

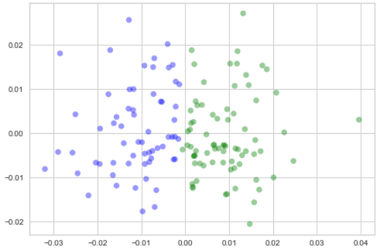
Furthermore, we used the code,(silhouette_score(data, foresee, random_state= 10))
to track down the score.

The score we discover after we drew nearer silhouette_score is,

For n_clusters = 2, silhouette score is 0.621182661392437)
For n_clusters = 3, silhouette score is 0.5526260806237652)
For n_clusters = 4, silhouette score is 0.5289564205836438)
For n_clusters = 5, silhouette score is 0.4827379082632182)
For n_clusters = 6, silhouette score is 0.4316693441954367)

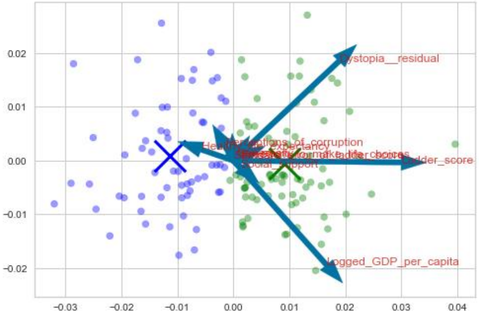
The above score for cluster 2 is 0.62 which is the most elevated one. Thus, the ideal value of cluster is 2. Moreover, we are writing the algorithm with 2 clusters. Post applying the algorithm with 2 clusters we found that th e 2 clusters created are 0,1.

9. Creating a scatter plot with the K-means clusters formed:



From the above scatter plot, it very well may be observed that 2 distinct clusters have dissipated in a distinct pattern. Additionally, we can see for the small outliers presented in it. Furthermore, we have added all the clusters in the original Data Frame.

10. Projecting the feature for K-means Clusters,

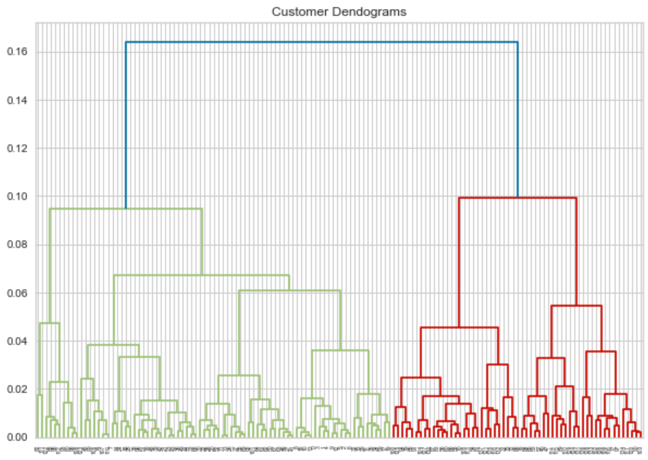


To project each principal component axis of their significance we are utilizing arrow and scatter plot which is available in the matplotlib.pyplot. The function arrow() adds the arrow to the diagram dependent on the directions passed to it. x, y: The x and y directions of arrow base. dx, dy: The length of the arrow along x and y heading.

The graph above gives a detailed information on the location of each of the variables and the locations of centroids. In the K-means algorithm after the centroids are determined each of the data points are allocated into the closest centroid. The illustration above gives a clear division between more and less developed countries. The variables such as GDP per capita, ladder score showing a larger impact on determining the clusters than the rest. Dystopia + residual shows a large component score as well, however it is just the score of dystopia (1.95) + the distance of each data point from the actual model, it is a good way to
determine how well each data point fits the "Dystopia" model, but isn't as useful in our research.

The arrows are the projection of each element on the principal component axis. These arrows address the degree of significance of each element in the multidimensional scaling. For instance, Ladder_Score and Logged_GDP_capita offer more than different highlights.

11. Plotting the dendrogram for Agglomerative clustering,



Agglomerative clustering makes **decisions by considering the local patterns or neighbour points** without initially taking into account the global distribution of data.

If we draw a horizontal line that passes through the longest distance without crossing a horizontal line, we get 3 clusters shown in the above figure.

Presently we know the number of clusters for our dataset, the subsequent stage is to group the information focuses into these 2 clusters. To do as such we will again utilize the AgglomerativeClustering class of the sklearn.cluster library.

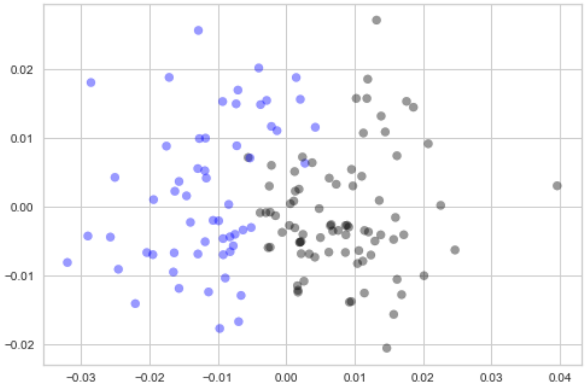
12. Justification of the selected linkage method,

The linkage methods work by calculating the distances or likenesses between all articles. Then, at that point, the nearest pair of groups are joined into a solitary bunch, diminishing the quantity of groups remaining. The cycle is then rehashed until there is just a solitary bunch left.

There are normally a few linkage strategies that could be utilized in hierarchical clustering. Among the fundamental ones are: complete, single, normal and wards techniques. Deciding the right technique is significant since it would produce an off-base number of groups or dole out information into less ideal groups in any case. The strategy is picked relying upon the sort of information utilized in the bunching investigation. The default method in clustering is the wards method which is not at all like other referenced techniques don't investigate the distance straightforwardly and dissect the difference all things considered. Accordingly, at the lower part of the dendrogram, the amount of squared distances begins from nothing and increments as we combine bunches. In this specific examination, the ward's technique was the ideal decision since it is the most suitable strategy for quantitative datasets. It is additionally founded on multidimensional changes same as head parts dependent on which we will play out the grouping investigation. The still up in the air were two clear groups that could be seen underneath the dabbed red line. It is noticeable that one of the groups is genuinely bigger than the other one.

I used "**ward**" technique for linkage function since Wards method looks to pick the progressive grouping steps in order to limit the expansion in Error Sum of Squares at each step. Additionally, the ward method is used because we performed the PCA in this dataset. It is based on multidimensional variance.

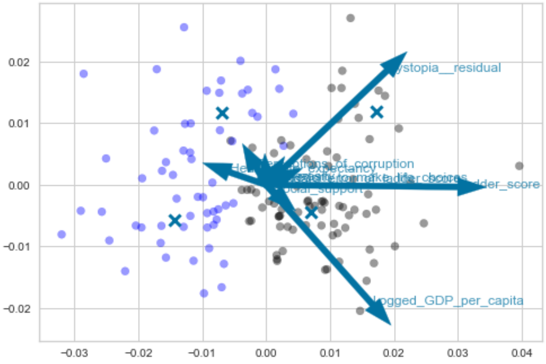
Moreover, I used "**Euclidean distance"** to track down the distance between two points. The Euclidean distance is the most generally utilized distance measure when the variables are continuous (either interval or ratio scale). The Euclidean distance between two points computes the length of a portion associating the two points. It is the most obvious method of addressing the distance between two points.



From the above scatter plot, it very well may be observed that 2 distinct clusters have dissipated in a distinct pattern. Furthermore, we have added all the clusters in the original Data Frame.

13. Projecting the feature for Agglomerative Clusters,

To project each principal component axis of their significance we are utilizing arrow and scatter plot which is available in the matplotlib.pyplot. The function arrow() adds the arrow to the diagram dependent on the directions passed to it. ... x, y: The x and y directions of arrow base. dx, dy: The length of the arrow along x and y heading.



The final results within hierarchical clustering analysis produced similar results to k-means clustering. It can be seen some of the data points in one cluster are colliding with the other, thus more overlapping is indicated in this case in comparison to k-means. Similar to before, the attributes that explain most of the principal components are Logged GDP per capita and Ladder score, with the rest still contributing to only a minor portion of the overall variances in the dataset.

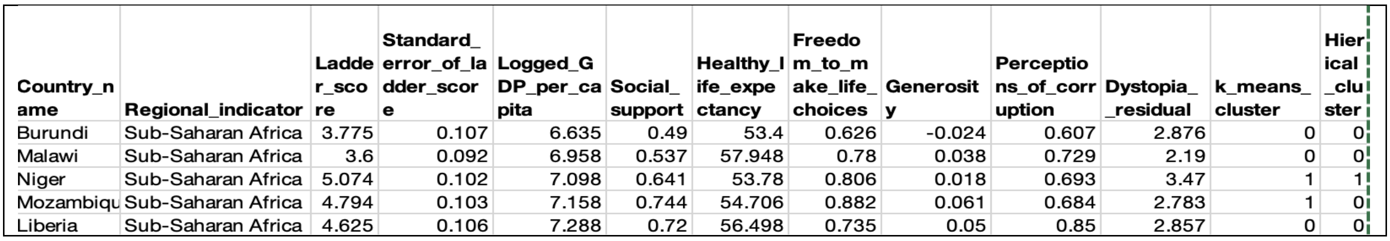
14.Meaning of the obtained clusters with reference to "Dystopia," in contrast to "Utopia";

In the question, that is given that the main 6 features of the data determine the meaning of Dystopia which is the world's lowest incomes, lowest life expectancy, lowest generosity, most corruption, least freedom and least social support in contrast to "Utopia". In this way, I found that every variable is independent except the "Perceptions_of_a_corruption" because the variable corruption can be higher for both dystopian and utopian countries. Additionally, in my arrow plot, it shows that the top variable contributing to the PCA is Ladder_score, Dystopia__residual and Logged_GDP_per_capita. However, I cannot justify by sorting the Dystopia__residual and ladder_score as Dystopian country since those two variables don't count as the key feature.

Hence, sorting out all the values in ascending order of the 6 features gives me the countries of "dystopia" and Vice versa.

Dystopia=df.sort_values(by=['Logged_GDP_per_capita','Generosity','Healthy_life_expectancy','Social_support','Freedom_to_make_life_choices'], ascending=True)

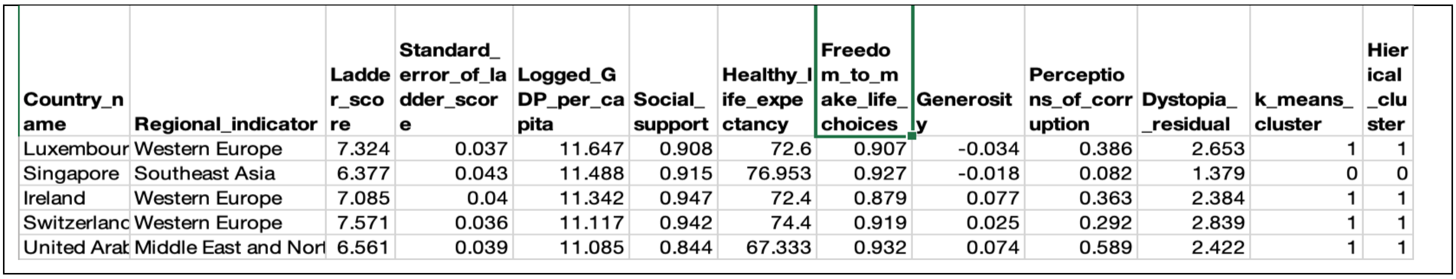
The above code gives the rundown of Dystopian countries since we have sorted the data with Logged_GDP_per_capita. Furthermore, I added every one of the clusters into the data frame.



Country_name	Regional_indicator	Ladder_score	Standard_error_of_ladder_score	Logged_GDP_per_capita	Social_support	Healthy_life_expectancy	Freedom_to_make_life_choices	Generosity	Perceptions_of_corruption	Dystopia_residual	k_means_cluster	Hierical_cluster
Burundi	Sub-Saharan Africa	3.775	0.107	6.635	0.49	53.4	0.626	-0.024	0.607	2.876	0	0
Malawi	Sub-Saharan Africa	3.6	0.092	6.958	0.537	57.948	0.78	0.038	0.729	2.19	0	0
Niger	Sub-Saharan Africa	5.074	0.102	7.098	0.641	53.78	0.806	0.018	0.693	3.47	1	1
Mozambiqu	Sub-Saharan Africa	4.794	0.103	7.158	0.744	54.706	0.882	0.061	0.684	2.783	1	0
Liberia	Sub-Saharan Africa	4.625	0.106	7.288	0.72	56.498	0.735	0.05	0.85	2.857	0	0

Post sorting, I found the top 5 countries Burundi, Malawi, Niger, Mozambique, Liberia, determine Dystopia because these countries have lower GDP, lower generosity, low freedom, lower life. Moreover, most affected countries from dystopia belong to the Region Sub-Saharan Africa. In addition, the clusters we got for the countries Burundi, Malawai, and Liberia is 0 in both K-means and Hierarchical clusters. Whereas the country Niger has cluster 1 in both K-means and Hierarchical. Finally, the country Mozambique obtains 1 in K-means and 0 in Hierarchical.

On the other hand, we can find the Utopian countries by sorting the data in higher values of GDP, Generosity, Freedom, Higher life expectancy. The data shown belongs to Utopia.



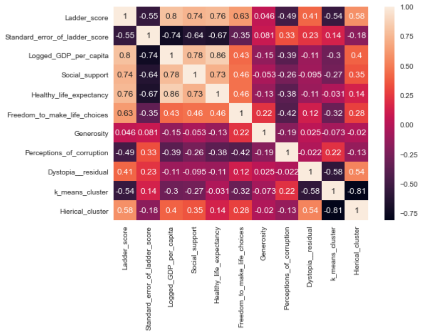
Country_name	Regional_indicator	Ladder_score	Standard_error_of_ladder_score	Logged_GDP_per_capita	Social_support	Healthy_life_expectancy	Freedom_to_make_life_choices	Generosity	Perceptions_of_corruption	Dystopia_residual	k_means_cluster	Hierical_cluster
Luxembour	Western Europe	7.324	0.037	11.647	0.908	72.6	0.907	-0.034	0.386	2.653	1	1
Singapore	Southeast Asia	6.377	0.043	11.488	0.915	76.953	0.927	-0.018	0.082	1.379	0	0
Ireland	Western Europe	7.085	0.04	11.342	0.947	72.4	0.879	0.077	0.363	2.384	1	1
Switzerlanc	Western Europe	7.571	0.036	11.117	0.942	74.4	0.919	0.025	0.292	2.839	1	1
United Arak	Middle East and Nort	6.561	0.039	11.085	0.844	67.333	0.932	0.074	0.589	2.422	1	1

The top 5 countries that belong to Utopia is Luxembourg, Singapore, Ireland, UAE since the countries have larger GDP, larger social support, and larger health expectancy where all the countries belong to cluster 1 in K-means and Hierarchical except Singapore.

15. Justify your findings using the results of tasks 2 and 3

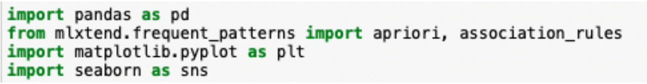
K-mean clustering additionally expands its time of execution. Notwithstanding, when contrasted with the hierarchical clustering, its presentation is better. Hierarchical clustering shows greater quality when contrasted with the k-mean calculation. As an overall end, the k-mean calculation is useful for enormous datasets and hierarchical is useful for little datasets.

To sum up the winning methods, I say both the models K-means and Hierarchical worked well for me. In Hierarchical, with the help of the below heatmap graph, I can see that ladder_score is correlated with it at 0.58, since the ladder_score captures the overall "happiness" of the data it could be concluded that the hierarchical algorithm did a better job at clustering than k-means (which correlated -0.54). On the other hand, the k-means algorithm showed no in-between cluster overlapping, with clusters separated. Overall, both of the clustering algorithms did a good job at determining between Dystopic and Utopic countries, there was a 0.80 correlation between the two indicating some division on how the data points were allocated.



2. Market Basket Analysis.

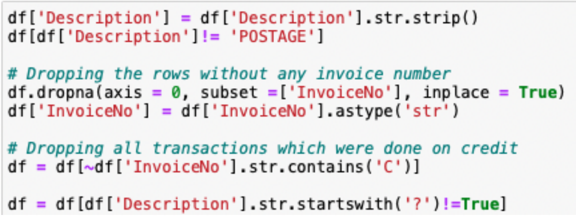
Importing the libraries and Data Set:



```
import pandas as pd
from mlxtend.frequent_patterns import apriori, association_rules
import matplotlib.pyplot as plt
import seaborn as sns
```

I have imported the libraries pandas, apriori and association_rules to deal with Market Basket Algorithm. I used pandas library to import the dataset into the jupyter notebook. Now the shape of the data frame is (541909,8) means 541909 rows and 8 columns.

Data Pre-processing:



```
df['Description'] = df['Description'].str.strip()
df[df['Description']!= 'POSTAGE']

# Dropping the rows without any invoice number
df.dropna(axis = 0, subset =['InvoiceNo'], inplace = True)
df['InvoiceNo'] = df['InvoiceNo'].astype('str')

# Dropping all transactions which were done on credit
df = df[~df['InvoiceNo'].str.contains('C')]

df = df[df['Description'].str.startswith('?')!=True]
```

Since the column description is in str() variable, there may be a possibility that the variable has additional spaces. Subsequently, I utilized the "str. strip" technique to eliminate the additional spaces. Besides, I erased the unnecessary value in the portrayal called "POSTAGE" which doesn't require the rundown to examine. Moreover, I changed the data type of the column Invoice to str() and dropped the columns that don't have receipt numbers and took out the credit transactions (those with invoice numbers containing C). Moreover, eliminated the rows which start with “?”. Now the shape of the data has become (532554,8).

Missing values:

There are **135080** missing values available for the column **CustomerID**. Be that as it may, Market Basket Analysis depends on the variable description alone. Henceforth, I deleted the whole variable of CustomerID. Additionally, there are 1454 missing values in the column description, which is just 0.268% of the whole information. Subsequently, I deleted the null values for the column description.

Duplicate values:

I have seen that there are 5268 duplicates available in the whole dataset. Consequently, I removed the duplicates. The size of the data frame after eliminating the missing and duplicate value is (532554, 7)

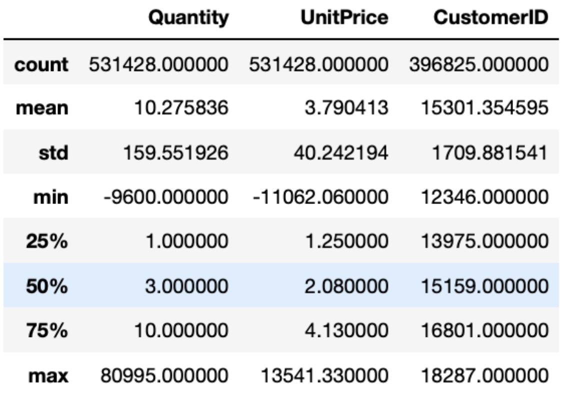
Filtering the values which is equal to or more than 0:

data = df[df['Quantity']>=0]

In this dataset, the Quantity column shows us the number of things that are purchased in every exchange (InvoiceNo). Once in a while, the exchange gets dropped, because this is online retail. When there is an undoing on a specific exchange, it will be certificated in the Quantity section as a negative worth. Since we're doing market basket analysis, we fundamentally might want to investigate what's inside the basket that our client purchased. This negative worth isn't one of them. That is the reason we're not going to utilize them. Once we have filtered the dataset of quantity >= 0, the shape of the data has become (530693,7).

Statistical Exploration:

I have only 3 columns with continuous variable. I used df.describe() function to describe about the statistical analysis for the 3 continuous variable.

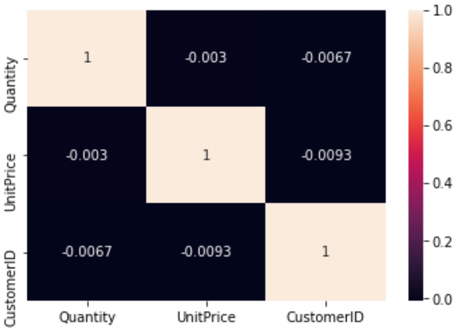


	Quantity	UnitPrice	CustomerID
count	531428.000000	531428.000000	396825.000000
mean	10.275836	3.790413	15301.354595
std	159.551926	40.242194	1709.881541
min	-9600.000000	-11062.060000	12346.000000
25%	1.000000	1.250000	13975.000000
50%	3.000000	2.080000	15159.000000
75%	10.000000	4.130000	16801.000000
max	80995.000000	13541.330000	18287.000000

The column Quantity has a mean value around 10.2 and Unitprice mean value is around 4.6. Hence, we don't have to concentrate on this since for MBA we need to use column description alone.

Correlation analysis:

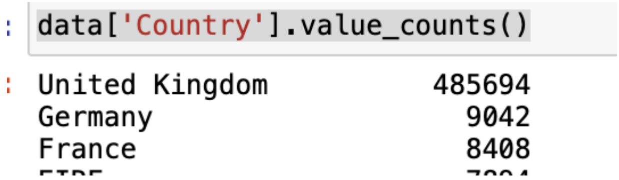
I used a heatmap to check there was any correlation between the variables.



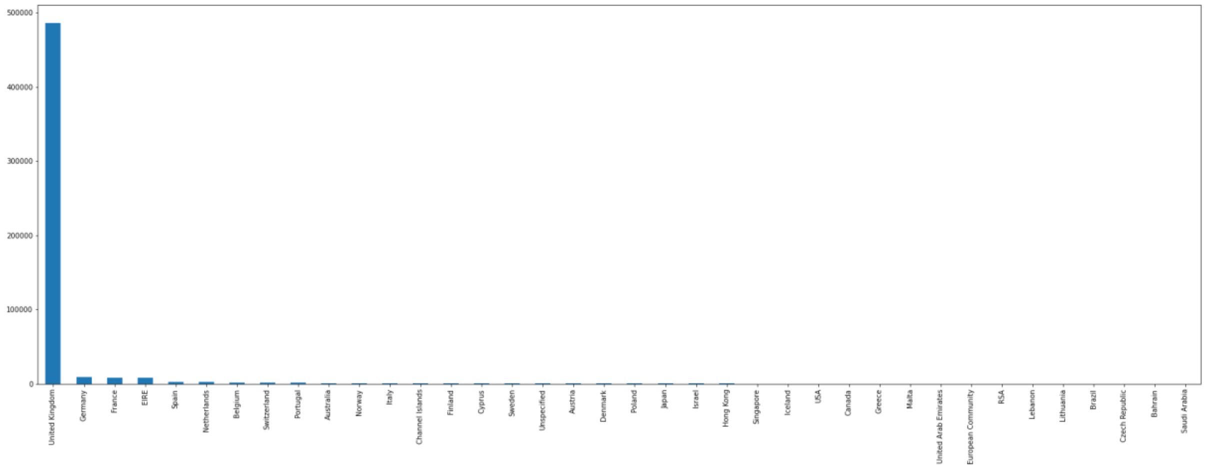
By picturing the plot, we can obviously say none of the values is close by (1 0r - 1). Henceforth, the persistent qualities are not connected.

Filtering the top Countries with distribution analysis:

I used the code data['Country'].value_counts() to check the number of features is available for every classification. Finally, I tracked down that UK, Germany and France have higher features than other countries.



```
: data['Country'].value_counts()

: United Kingdom          485694
  Germany                   9042
  France                    8408
```



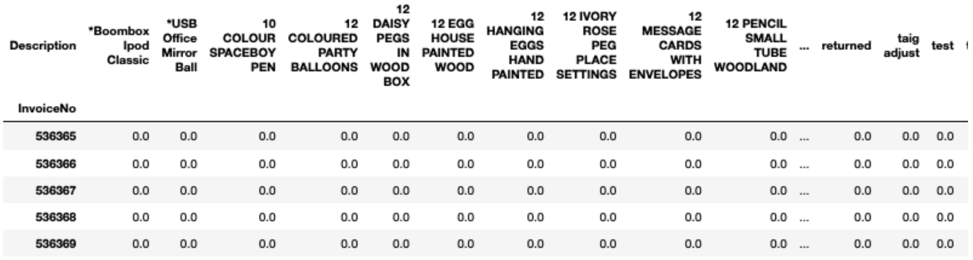
Consequently, I filtered the top 3 basket which is UK, Germany and France alone to squeeze the model into the "apriori" calculation.

Grouping the Data:

The next step I did is to group the data to get the binary value 0 and 1 as same as one-hot encoding.

UK = (data[data['Country']== 'United Kingdom']).groupby(['InvoiceNo','Description'])['Quantity'].sum().unstack().reset_index().fillna(0).set_index('InvoiceNo').

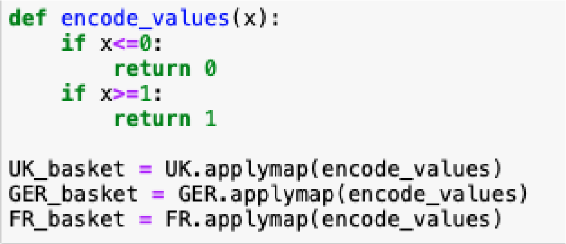
The above code filters the country of the UK and groups the factors 'InvoiceNo' and 'Description' with the variable Quantity. The SUM() function gives how frequently the items are ordered. Unstack() unstacks the information once grouping is done. reset_index() resets the index from 0. The below picture tells once the grouping function is
done



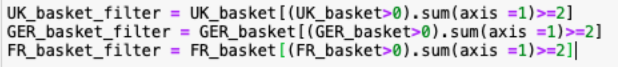
Description	*Boombox Ipod Classic	*USB Office Mirror Ball	10 COLOUR SPACEBOY PEN	12 COLOURED PARTY BALLOONS	12 DAISY PEGS IN WOOD BOX	12 EGG HOUSE PAINTED WOOD	12 HANGING EGGS HAND PAINTED	12 IVORY ROSE PEG PLACE SETTINGS	12 MESSAGE CARDS WITH ENVELOPES	12 PENCIL SMALL TUBE WOODLAND	...	returned	taig adjust	test
InvoiceNo														
536365	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0
536366	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0
536367	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0
536368	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0
536369	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0

Encoding Data:

Once the grouping is done, I supplanted every one of the features which are lesser than or equivalent to 0 with 0 and supplanted every one of the features with 1 which is more noteworthy than or equivalent to 1.



```
def encode_values(x):
    if x<=0:
        return 0
    if x>=1:
        return 1

UK_basket = UK.applymap(encode_values)
GER_basket = GER.applymap(encode_values)
FR_basket = FR.applymap(encode_values)
```

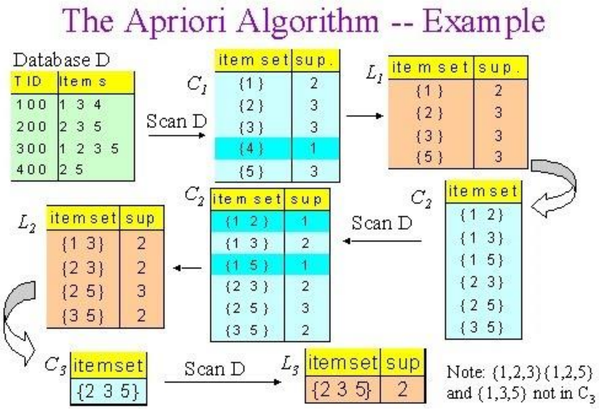


```
UK_basket_filter = UK_basket[(UK_basket>0).sum(axis =1)>=2]
GER_basket_filter = GER_basket[(GER_basket>0).sum(axis =1)>=2]
FR_basket_filter = FR_basket[(FR_basket>0).sum(axis =1)>=2]
```

In market basket analysis, we will reveal the association between at least 2 things that are bought according to historical data. Thus, it is less helpful assuming an exchange just purchased a solitary thing. That is to say, how is it that we could reveal the relationship between at least 2 things in case there is just 1 thing purchased? Subsequently, the following stage is to sift through the exchanges that are purchased more than 1 thing.

Generate Frequent Item sets:

Apriori algorithm is a classical algorithm in data mining. It is used for mining frequent itemsets and relevant association rules. It is devised to operate on a database containing a lot of transactions, for instance, items brought by customers in a store.



The Apriori Algorithm -- Example

Database D

TID	Items
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

C_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{4}	1
{5}	3

L_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{5}	3

C_2

itemset
{1 2}
{1 3}
{1 5}
{2 3}
{2 5}
{3 5}

C_2

itemset	sup
{1 2}	1
{1 3}	2
{1 5}	1
{2 3}	2
{2 5}	3
{3 5}	2

L_2

itemset	sup
{1 3}	2
{2 3}	2
{2 5}	3
{3 5}	2

C_3

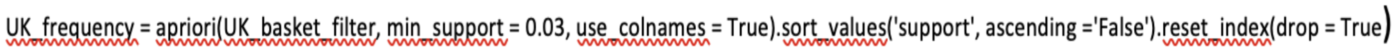
itemset
{2 3 5}

L_3

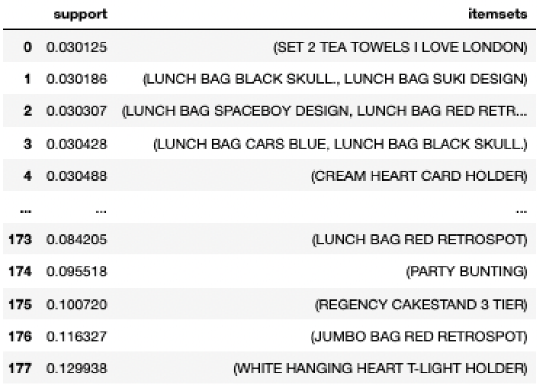
itemset	sup
{2 3 5}	2

Note: {1,2,3}{1,2,5} and {1,3,5} not in C_3

Now that the data is structured properly, we can generate frequent item sets by using “apriori” algorithm that have a support of at least 3% (this number was chosen so that I could get enough useful examples).



```
UK_frequency = apriori(UK_basket_filter, min_support = 0.03, use_colnames = True).sort_values('support', ascending ='False').reset_index(drop = True)
```

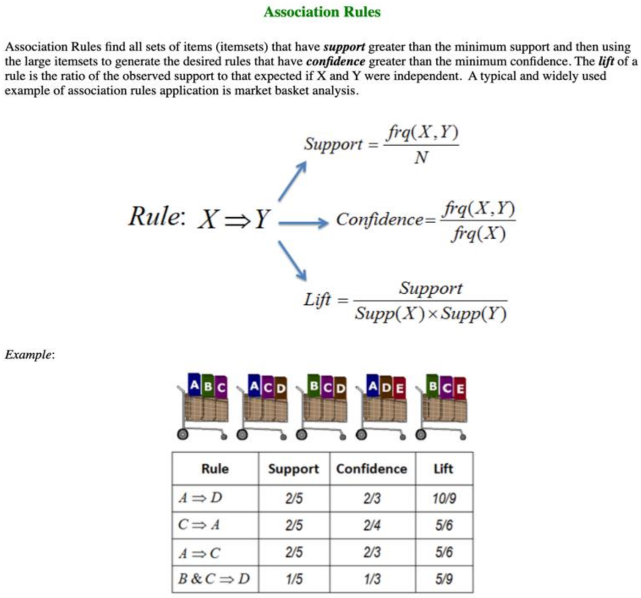


	support	itemsets
0	0.030125	(SET 2 TEA TOWELS I LOVE LONDON)
1	0.030186	(LUNCH BAG BLACK SKULL., LUNCH BAG SUKI DESIGN)
2	0.030307	(LUNCH BAG SPACEBOY DESIGN, LUNCH BAG RED RETR...
3	0.030428	(LUNCH BAG CARS BLUE, LUNCH BAG BLACK SKULL.)
4	0.030488	(CREAM HEART CARD HOLDER)
...	...	...
173	0.084205	(LUNCH BAG RED RETROSPOT)
174	0.095518	(PARTY BUNTING)
175	0.100720	(REGENCY CAKESTAND 3 TIER)
176	0.116327	(JUMBO BAG RED RETROSPOT)
177	0.129938	(WHITE HANGING HEART T-LIGHT HOLDER)

In the above image we can see that the itemset which contains 3% support.

Building Association Rules Using Frequent Itemset:

The last advance is to produce the rules with their corresponding support, confidence and lift. Association rules are made via scanning information for successive if-then patterns and utilizing the criteria support and confidence to distinguish between the main connections. Support means that how frequently now and again the things show up in the data. Confidence shows the occasions if assertions are tracked down obvious. A third measurement, called lift, is used to compare confidence with expected confidence, or how often an if-then statement is expected to be found True.



Association Rules

Association Rules find all sets of items (itemsets) that have ***support*** greater than the minimum support and then using the large itemsets to generate the desired rules that have ***confidence*** greater than the minimum confidence. The ***lift*** of a rule is the ratio of the observed support to that expected if X and Y were independent. A typical and widely used example of association rules application is market basket analysis.

$$Rule:\ X \Rightarrow Y$$

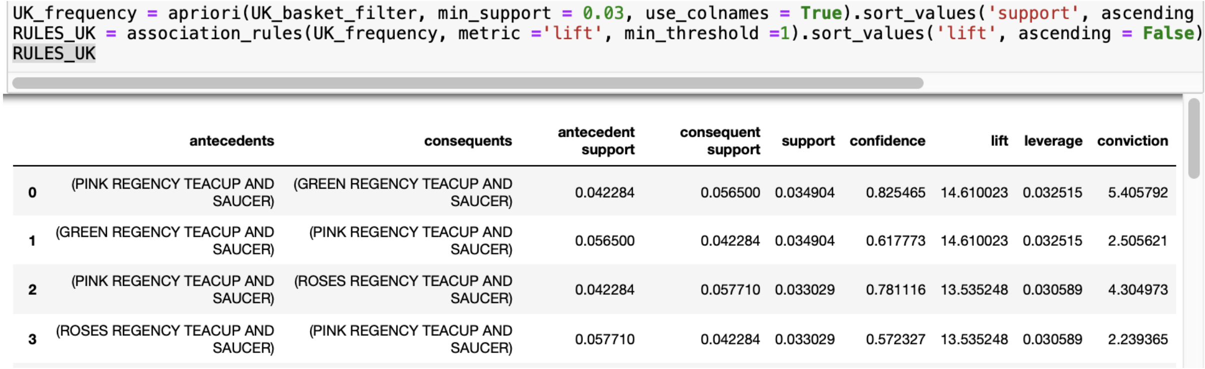
$$Support = \frac{frq(X,Y)}{N}$$

$$Confidence = \frac{frq(X,Y)}{frq(X)}$$

$$Lift = \frac{Support}{Supp(X) \times Supp(Y)}$$

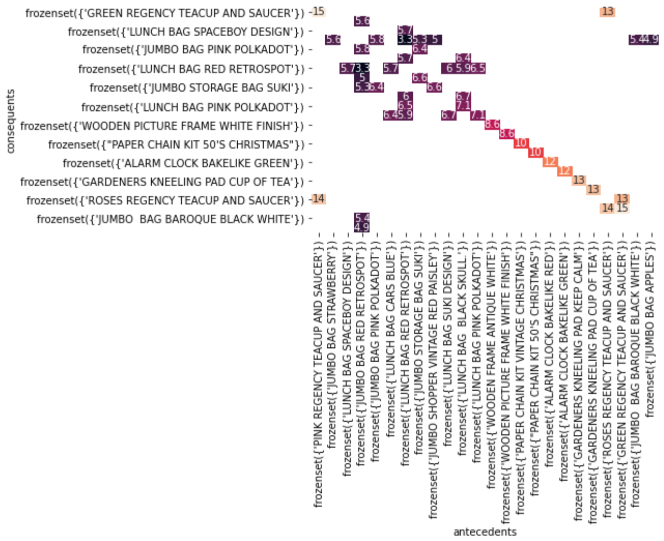
Example:

Rule	Support	Confidence	Lift
$A \Rightarrow D$	2/5	2/3	10/9
$C \Rightarrow A$	2/5	2/4	5/6
$A \Rightarrow C$	2/5	2/3	5/6
$B \& C \Rightarrow D$	1/5	1/3	5/9



```
UK_frequency = apriori(UK_basket_filter, min_support = 0.03, use_colnames = True).sort_values('support', ascending
RULES_UK = association_rules(UK_frequency, metric ='lift', min_threshold =1).sort_values('lift', ascending = False)
RULES_UK
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(PINK REGENCY TEACUP AND SAUCER)	(GREEN REGENCY TEACUP AND SAUCER)	0.042284	0.056500	0.034904	0.825465	14.610023	0.032515	5.405792
1	(GREEN REGENCY TEACUP AND SAUCER)	(PINK REGENCY TEACUP AND SAUCER)	0.056500	0.042284	0.034904	0.617773	14.610023	0.032515	2.505621
2	(PINK REGENCY TEACUP AND SAUCER)	(ROSES REGENCY TEACUP AND SAUCER)	0.042284	0.057710	0.033029	0.781116	13.535248	0.030589	4.304973
3	(ROSES REGENCY TEACUP AND SAUCER)	(PINK REGENCY TEACUP AND SAUCER)	0.057710	0.042284	0.033029	0.572327	13.535248	0.030589	2.239365



From the association_rules results above, we could see that “**PINK REGENCY TEACUP AND SAUCER**” and “**GREEN REGENCY TEACUP AND SAUCER**” are the things that have the most elevated association with each other since these two things have the most elevated "lift" esteem. The higher the lift esteem, the higher the relationship between the things will. If the lift value is more than 1, it is sufficient for us to say that those two things are related to one another. For this situation, the most elevated worth is 14.6 which is extremely high. It implies these 2 things are generally excellent to be sold together.

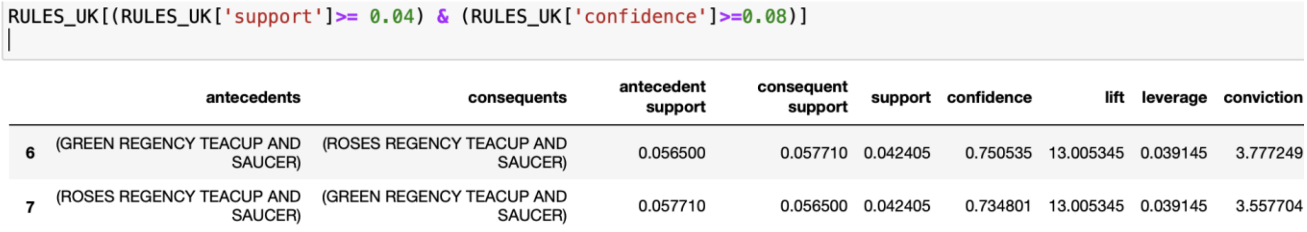
Next to that, we could likewise see the support value of “**PINK REGENCY TEACUP AND SAUCER**” and “**GREEN REGENCY TEACUP AND SAUCER**” are 0.034% which implies there is 3.04% out of all-out a transaction that these 2 things were sold together.

From the confidence, we could even extract more data. Recall that the confidence value is affected by the antecedent and consequent. Assuming the antecedent is higher than the consequent, the standard to be applied is rule number 1 (not number 2). the other way around. For this situation, the antecedent value is higher than the consequent value. It implies we apply rule number 1 that is "**PINK REGENCY TEACUP AND SAUCER**" and "**GREEN REGENCY TEACUP AND SAUCER**". In a more detailed clarification, it implies that a client will tend to be purchased GREEN REGENCY TEACUP AND SAUCER after they purchase “**PINK REGENCY TEACUP AND SAUCER**”. Not in the opposite way around. This could be entirely important data since we are currently mindful of which items should we put the limits on. We could give a discount on "**GREEN REGENCY TEACUP AND SAUCER**" if a client purchases "**PINK REGENCY TEACUP AND SAUCER**".

Rules for the basket:

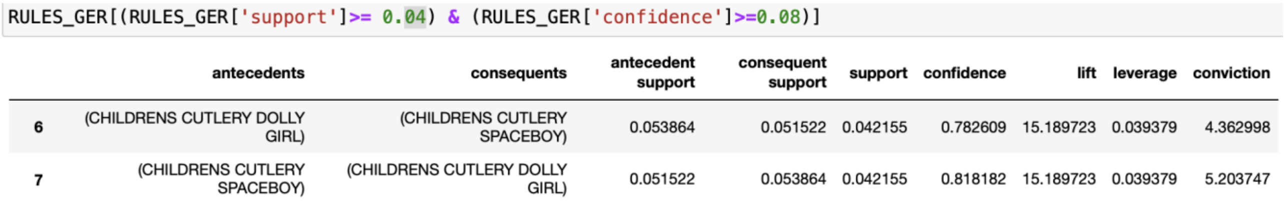
Market Basket Analysis creates If-Then scenario rules, for example, if item A is purchased then item B is likely to be purchased. The rules are probabilistic or, in other words, they are derived from the frequencies of co-occurrence in the observations.

I have applied two rules to get the products which can be sold together, the first rule is finding the basket which says that we can sell two products together,



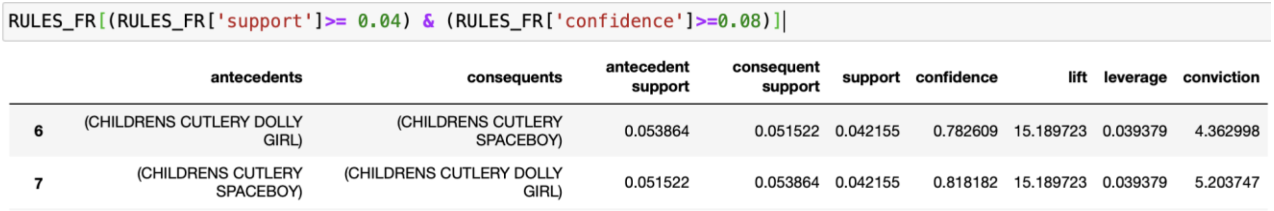
```
RULES_UK[(RULES_UK['support']>= 0.04) & (RULES_UK['confidence']>=0.08)]
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
6	(GREEN REGENCY TEACUP AND SAUCER)	(ROSES REGENCY TEACUP AND SAUCER)	0.056500	0.057710	0.042405	0.750535	13.005345	0.039145	3.777249
7	(ROSES REGENCY TEACUP AND SAUCER)	(GREEN REGENCY TEACUP AND SAUCER)	0.057710	0.056500	0.042405	0.734801	13.005345	0.039145	3.557704



```
RULES_GER[(RULES_GER['support']>= 0.04) & (RULES_GER['confidence']>=0.08)]
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
6	(CHILDRENS CUTLERY DOLLY GIRL)	(CHILDRENS CUTLERY SPACEBOY)	0.053864	0.051522	0.042155	0.782609	15.189723	0.039379	4.362998
7	(CHILDRENS CUTLERY SPACEBOY)	(CHILDRENS CUTLERY DOLLY GIRL)	0.051522	0.053864	0.042155	0.818182	15.189723	0.039379	5.203747



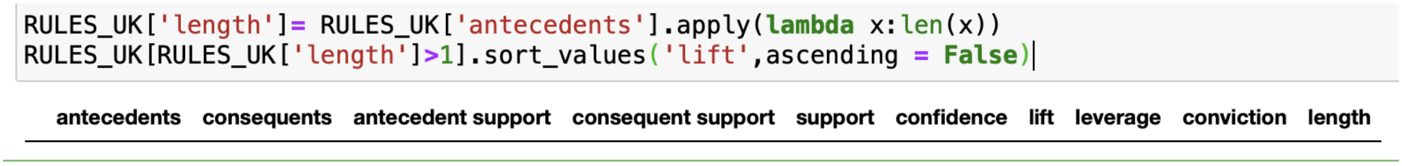
```
RULES_FR[(RULES_FR['support']>= 0.04) & (RULES_FR['confidence']>=0.08)]
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
6	(CHILDRENS CUTLERY DOLLY GIRL)	(CHILDRENS CUTLERY SPACEBOY)	0.053864	0.051522	0.042155	0.782609	15.189723	0.039379	4.362998
7	(CHILDRENS CUTLERY SPACEBOY)	(CHILDRENS CUTLERY DOLLY GIRL)	0.051522	0.053864	0.042155	0.818182	15.189723	0.039379	5.203747

When the support is greater than or equal to 0.04 and confidence is greater than or equal to 0.08, I can say that the products can be sold together for the basket UK is GREEN REGENCY TEACUP AND SAUCER and ROSES REGENCY TEACUP AND SAUCER.

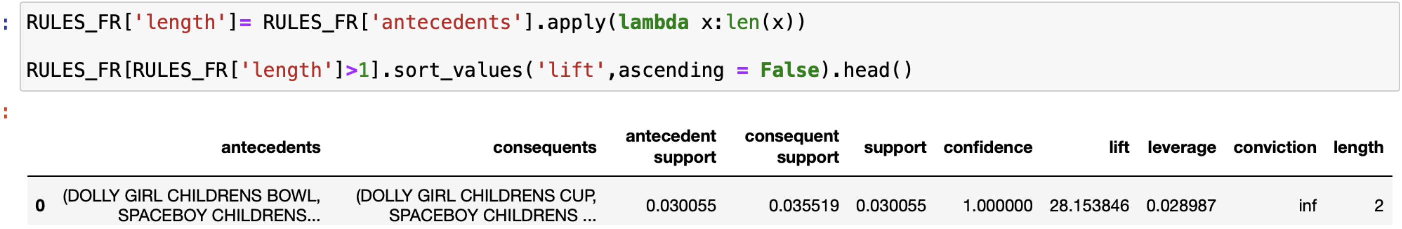
Similarly, for the market region Germany, I found that the product that can be sold together is CHILDRENS CUTLERY DOLLY GIRL, CHILDRENS CUTLERY SPACEBOY. Additionally, I found that the products that can be sold together in France is, CHILDRENS CUTLERY DOLLY GIRL and CHILDRENS CUTLERY SPACEBOY.

The second rule I tried is to check 3 products which can be sold together,



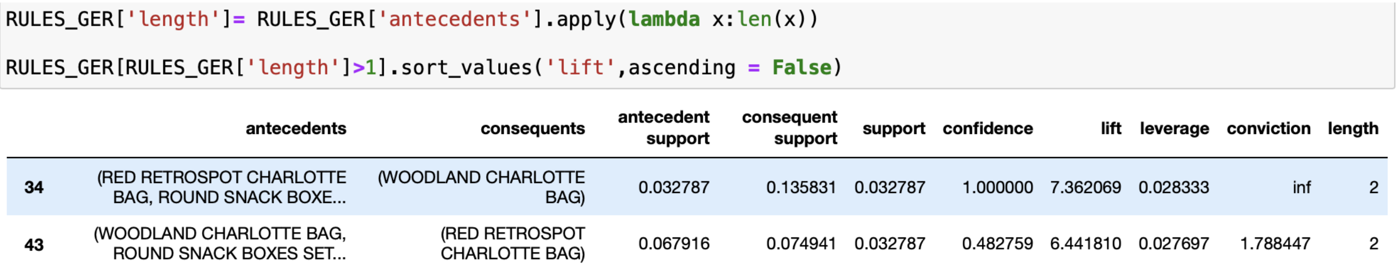
```
RULES_UK['length']= RULES_UK['antecedents'].apply(lambda x:len(x))
RULES_UK[RULES_UK['length']>1].sort_values('lift',ascending = False)
```

antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	length



```
RULES_FR['length']= RULES_FR['antecedents'].apply(lambda x:len(x))

RULES_FR[RULES_FR['length']>1].sort_values('lift',ascending = False).head()
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	length
0	(DOLLY GIRL CHILDRENS BOWL, SPACEBOY CHILDRENS...	(DOLLY GIRL CHILDRENS CUP, SPACEBOY CHILDRENS ...	0.030055	0.035519	0.030055	1.000000	28.153846	0.028987	inf	2



```
RULES_GER['length']= RULES_GER['antecedents'].apply(lambda x:len(x))

RULES_GER[RULES_GER['length']>1].sort_values('lift',ascending = False)
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	length
34	(RED RETROSPOT CHARLOTTE BAG, ROUND SNACK BOXE...	(WOODLAND CHARLOTTE BAG)	0.032787	0.135831	0.032787	1.000000	7.362069	0.028333	inf	2
43	(WOODLAND CHARLOTTE BAG, ROUND SNACK BOXES SET...	(RED RETROSPOT CHARLOTTE BAG)	0.067916	0.074941	0.032787	0.482759	6.441810	0.027697	1.788447	2

For the market region of the UK, I could not find any basket with more than 3 products that can be sold together since I have used the support value of 0.03. I cannot give the reduced support value since the data in the basket is enormous and we get a lot of items if we reduce the support rate. However, the marketplace France best-selling more than 2 products are DOLLY GIRL CHILDRENS BOWL, SPACEBOY CHILDRENS EGG CUP, SPACEBOY CHILDRENS EGG CUP and DOLLY GIRL CHILDRENS CUP, SPACEBOY CHILDRENS EGG CUP. Additionally, for the basket Germany I found the products which is more than 2 can be sold together is ROUND SNACK BOXES SET OF4 WOODLAND, RED RETROSPOT SHOPPER BAG and WOODLAND CHARLOTTE BAG.

Conclusion:

The aftereffect of the above investigation is a piece of important data and it very well may be utilized for information-driven campaigning, building advertising methodologies and independent direction.

The methodologies for non-store online retail might incorporate: We can give discounts on the product B if someone's buy product A. Additionally, we can sell the item in a pack of 2 or 3 with the antecedents and consequents product. Furthermore, with the help of recommendation system algorithm, we can recommend the product to customers when they are searching online.

Reference:

1. https://www.saedsayad.com/association_rules.htm

2.https://www.google.com/search?q=apriori&rlz=1C5CHFA_enIN965IN965&sxsrf=AOaemvK3kIDGmTYSdYkc_fhERpPzG3is1w:1637700703764&source=lnms&tbm=isch&sa=X&ved=2ahUKEwi4ncjFrq_0AhWkoVwKHdY7BxkQ_AUoAnoECAEQBA&biw=1440&bih=789&dpr=2

3. https://www.kaggle.com