1. Clustering and Visualizations

The World Happiness Report is an annual survey published by the United Nations. The dataset contains the happiness score of 150 countries across the globe based on ten factors including geography, Religion, level of religiosity, economy, family, life expectancy, freedom, trust in Government, generosity and Dystopia residual.

Dystopia - the opposite of utopia is an imaginary country with the lowest level of happiness, thus having the least happiness score. The purpose of setting Dystopia is to have a reference for other countries to compare for each of the six key variables (no country scores worse than Dystopia) so that each sub-bar can have a positive width.

The objective of the given task is to identify the main groups of the countries based on their happiness score by performing clustering techniques such as K-means clustering and Agglomerative hierarchical clustering after performing dimensionality reduction (PCA).

Pre-Processing task – Understanding and cleaning our data

From the below output shape of our data set is (149, 20) – it has 149 observations and 20 attributes (This includes all observations that might be removed as part of the data cleaning process).

Country name and Reginal indicator are categorical variables and all others are numerical variables(float datatype).

```
shape of the dataset: (149, 20)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 149 entries, 0 to 148
Data columns (total 20 columns):
Country name                              149 non-null object
Regional indicator                        149 non-null object
Ladder score                              149 non-null float64
Standard error of ladder score            149 non-null float64
upperwhisker                              149 non-null float64
lowerwhisker                              149 non-null float64
Logged GDP per capita                     149 non-null float64
Social support                            149 non-null float64
Healthy life expectancy                   149 non-null float64
Freedom to make life choices              149 non-null float64
Generosity                                149 non-null float64
Perceptions of corruption                 149 non-null float64
Ladder score in Dystopia                  149 non-null float64
Explained by: Log GDP per capita          149 non-null float64
Explained by: Social support              149 non-null float64
Explained by: Healthy life expectancy     149 non-null float64
Explained by: Freedom to make life choices 149 non-null float64
Explained by: Generosity                  149 non-null float64
Explained by: Perceptions of corruption   149 non-null float64
Dystopia + residual                       149 non-null float64
dtypes: float64(18), object(2)
```

Figure 1: Shape and Variables information of the dataset

Data Cleaning:

Identification and treatment of missing values: It appears to be no missing values and duplicates in our dataset.

```
#Identifying any missing values - n.a, NA, na, n/a,? etc
missing_value_formats = ["n.a.","?","NA","n/a", "na", "--"]
df = pd.read_csv('/Users/himan/OneDrive/Desktop/Data Mining/Course_work/Happiness-Data.csv", na_values = missing_value_formats)
print(df.isnull().sum())
```

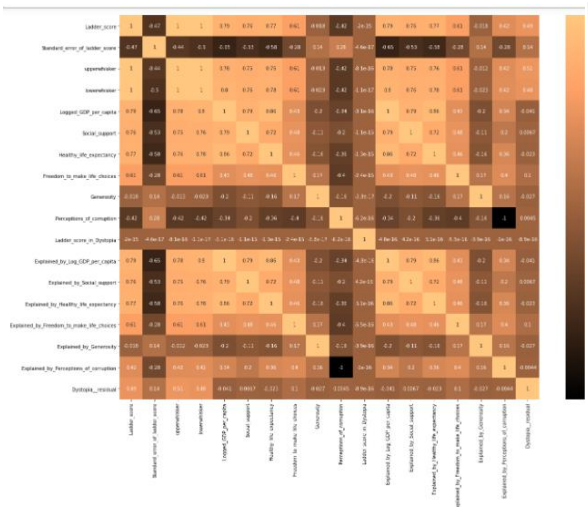
```
Country name                              0
Regional indicator                        0
Ladder score                              0
Standard error of ladder score            0
upperwhisker                              0
lowerwhisker                              0
Logged GDP per capita                     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
```

Figure 2: Number of missing values

Checking correlation between all the variables and dropping the highly correlated variables:

All the correlated variables would be reduced to set of linearly uncorrelated variables (principal components) after preforming PCA(Dimensionality Reduction). However, we can see from the above output that most of the columns are repeated, let us check for highly correlated variable and drop them to make our dataset simple and more readable

Below are the columns that have correlation greater than 90%:



```
#Find features having correlation greater than 90%
Highly_correlated_columns = corr(features, 0.90)
Highly_correlated_columns
```

```
{'Explained_by_Freedom_to_make_life_choices',
 'Explained_by_Generosity',
 'Explained_by_Healthy_life_expectancy',
 'Explained_by_Log_GDP_per_capita',
 'Explained_by_Perceptions_of_corruption',
 'Explained_by_Social_support',
 'lowerwhisker',
 'upperwhisker'}
```

Figure 3 : Heatmap of our dataset before dropping highly correlated columns
Figure 4: output of the function – highly correlated columns

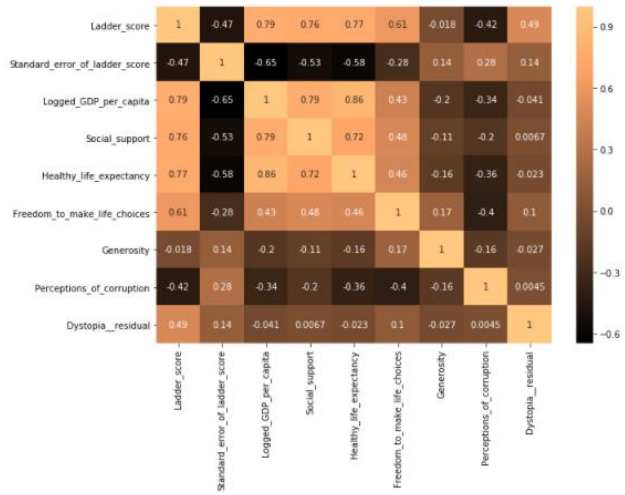


Figure 5: Heatmap of our dataset after dropping the highly correlated variables

After dropping the highly correlated variables, the shape of our dataframe has changed from (149, 20) to (149, 12) - Current dataframe has 149 observations and 12 variables.

Identification of outliers using boxplot:

K-means algorithm is very sensitive to outliers as it tries to optimize the sum of squared distance and mean of the clusters, thus any deviation like outliers might increase the variation with in the clusters. So it is always advisable to remove the outliers before preforming K-means.

From the below output it is evident that the variables – Ladder score, Standard error of ladder score, Social support, Freedom to make life choices, Generosity and Dystopia + residual have a few outliers.

We'll drop outliers from the below variables, except 'Perceptions of corruption' – as it's outliers are spread across the axis, it will have zero to negligible affect on our output.

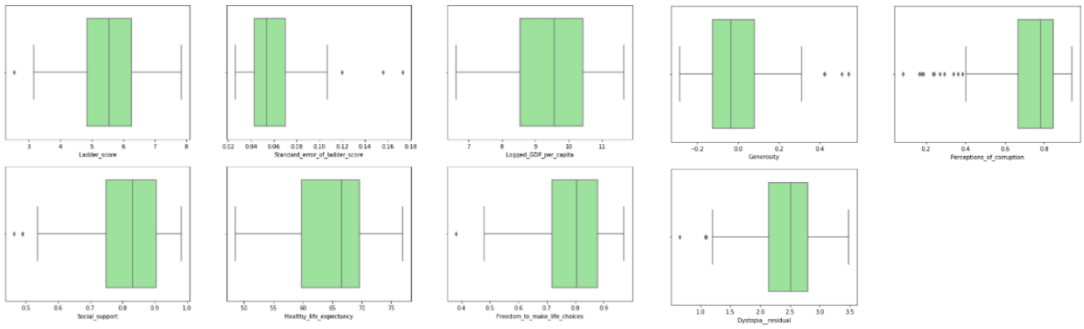


Figure 6: Output of boxplot with outliers

Data Normalisation:

It necessary to scale or normalize our data before performing PCA. As PCA reduces the dimensionality of our dataset by calculating a projection of the data based on the standard deviation of variables.

If we don't normalize our dataset then the variable with a high standard deviation will have a higher weight than the variables with lower standard deviation for the calculation of principle component axis, thus the result of PCA will be biased.

After normalizing our data, all the variable will have the same weightage and will get relevant PCs axis.

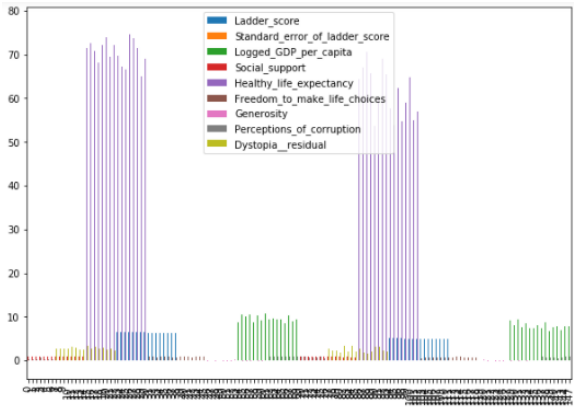


Figure 7: plot of our dataset before Normalization

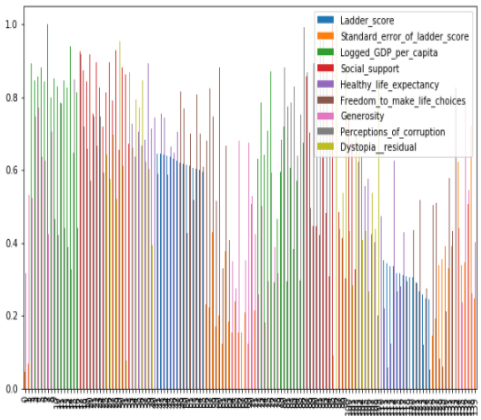


Figure 8: After Normalization

It is evident from the above graph that our dataset us not normally distributed. We'll use MinMaxScaler() to normalize the data, it scales down the values of all the features between 0 and 1.

Dimensionality Reduction Using PCA:

Principle Component Analysis (PCA) is one of the simplest linear dimensionality reduction techniques. Using PCA we reduce the dimensionality of large data sets by projecting it into lower sub-dimensional by preserving only the essential parts that have more variation of the data and removing the non-essential parts with a fewer variation.

Principle Components have directions and magnitudes and are called Eigen Values and Eigen Vectors respectively and the percentage of variation in the given data is explained by each Eigen Vectors.

Selecting no. of Principle Components:

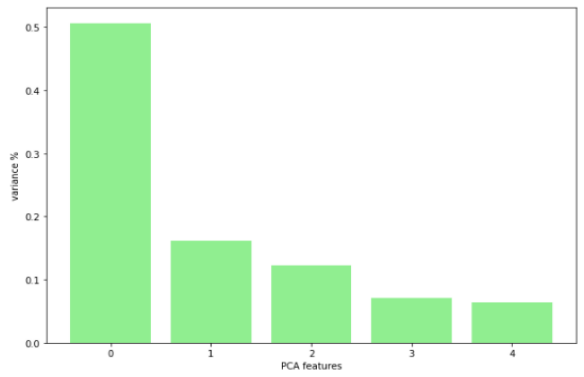
We would select no. PCs such that the total variation is greater than 90%, to preserve as much information as possible.

For No. PCs =4: The total variance ratio is 86% (<90%)

```
pca_4PCs.explained_variance_ratio_
array([0.50529757, 0.16169726, 0.12306431, 0.07084812])
pca_4PCs.explained_variance_ratio_.sum()
0.8609072684974579
```

Figure 9: Sum of total variance of Principle component 4

For PCs 5: The total variance ratio is 92%
Since the total variance is >90%, we would no. of principle components = 5




```
pca.explained_variance_ratio_
array([0.50529757, 0.16169726, 0.12306431, 0.07084812, 0.06326119])
pca.explained_variance_ratio_.sum()
0.9241684598930537
```

Figure 10: Sum of total variance of Principle component 5

Variance levels of different Principle components.

Eigen Values and Eigen Vectors of each PCs:

```
print('Eigen Values ', pca.explained_variance_ratio_)
```

```
Eigen Values  [0.50529757 0.16169726 0.12306431 0.07084812 0.06326119]
```

```
print('Eigen vectors ', pca.components_)
```

```
Eigen vectors  [[-0.43316668  0.22480083 -0.45535577 -0.4028529  -0.46103511 -0.34692818
  -0.00176857  0.24229198 -0.02389177]
 [-0.22113677 -0.16759793  0.23611111  0.16930799  0.20756597 -0.40825728
  -0.50795424  0.23675768 -0.56104051]
 [-0.24861998 -0.12837138 -0.04065796 -0.01453733 -0.02407346  0.17322658
   0.59151356 -0.28610862 -0.67639565]
 [-0.01356115  0.08573899 -0.04734443  0.34474837 -0.03804128  0.36480631
   0.22609703  0.82492241 -0.07227969]
 [ 0.15812593 -0.13649273  0.15295237  0.15943272  0.09912304 -0.71936889
   0.56499898  0.14433956  0.20044445]]
```

Figure 11: Eigen values and Eigen vectors

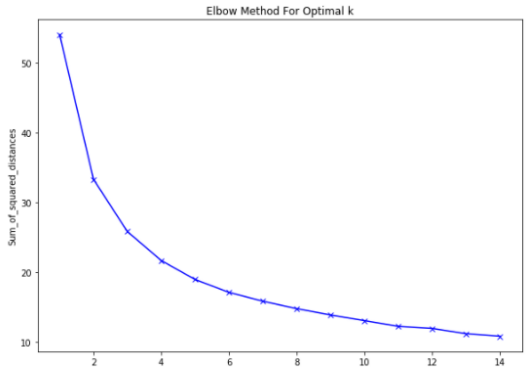
Finding the ideal number of clusters

I have used the Elbow method and silhouette score to identify the ideal number of clusters

Elbow method:

The Elbow method is a method that looks at the percentage of variance within the clusters and plots it against the number of clusters. The idea behind this method is to choose the number of clusters such that adding another cluster doesn't give an improved modelling of the data.

The first clusters would add much information but at some point, the marginal gain will drop dramatically and gives an angle in the graph. The ideal number of clusters(k) is chosen at this point, hence the elbow method.



It is evident from the above graph that there is no particular elbow in our dataset.

Choosing number of clusters K based on silhouette score:

Silhouette score is used to evaluate the quality of clusters created using clustering algorithms such as K-Means in terms of how well samples are clustered with other samples that are similar to each other.

Higher the Silhouette score, better and well separated the clusters.

```
from sklearn.metrics import silhouette_score


n_clusters = [2,3,4,5,6]


for K in n_clusters:
    cluster = KMeans (n_clusters= K, random_state= 10)
    predict = cluster.fit_predict(scaled_df)
    score = silhouette_score(scaled_df, predict, random_state= 10)
    print ("For n_clusters = {}, silhouette score is {})".format(K, score))
```

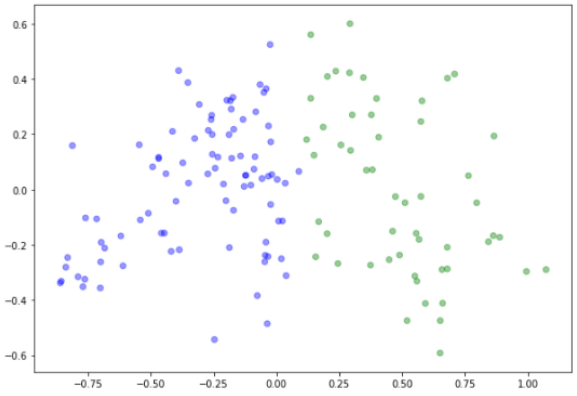
```
For n_clusters = 2, silhouette score is 0.3117478830113238)
For n_clusters = 3, silhouette score is 0.27175512128336354)
For n_clusters = 4, silhouette score is 0.25679562085177643)
For n_clusters = 5, silhouette score is 0.2484822680661873)
For n_clusters = 6, silhouette score is 0.24436068208215522)
```

From the above output silhouette score for no. of clusters k=2 is the highest. So, we would consider 2 clusters.

Performing K-means on the reduced dataset:

K-means clustering is a centroid based clustering technique. It divides the given unsupervised dataset into K clusters or groups such that similar objects/observations are placed in the same cluster/group and each cluster are represented by their centroids.

After performing PCA variables in our dataset has been reduced from 8 variable to 5PCs.



scatterplot of K-means algorithm for k=2

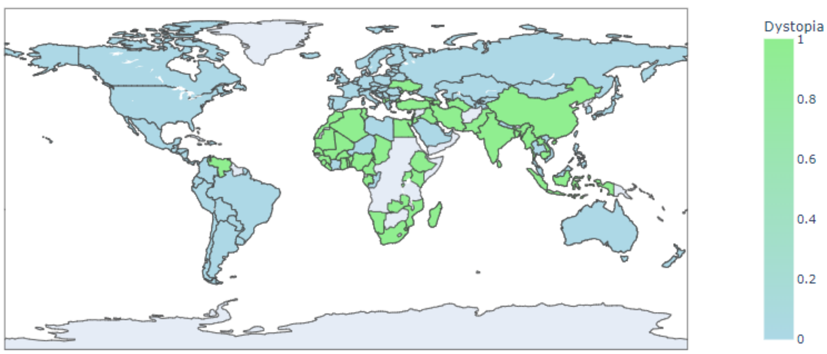
```
clusterer = KMeans(n_clusters=2, random_state=10)
cluster_labels = clusterer.fit_predict(PCA_components)
print(cluster_labels)
```

```
[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 0 1 0 1 0 1 1 1 0 0 0 0 0 0 1 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 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
```

```
scaled_df['k_means_cluster'] = label
scaled_df['k_means_cluster'].value_counts()
#scaled_df.head(5)
```

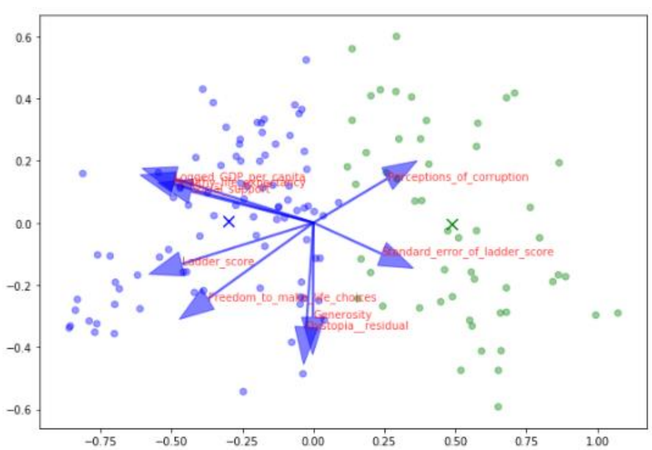
```
0    87
1    53
Name: k_means_cluster, dtype: int64
```

No. of countries in cluster 0 (Blue): 87
No. of countries in cluster 1(Green): 53



World map based on the K-means clustering

Projection of each original feature on the principle component axis:



Projection of each feature on the principle component axis

The arrows represent each feature onto the principle component axis and the magnitude of each arrow represents the level of importance of each feature in the multidimensional scaling.

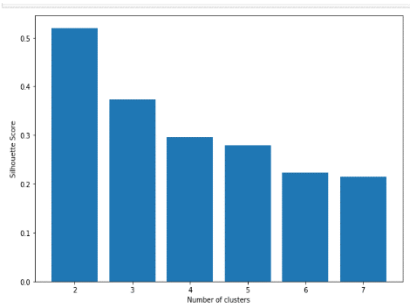
From the above output, we can conclude that the 'Logged GDP per capita’ has the most impact whereas Generosity has the least impact on the happiness score.

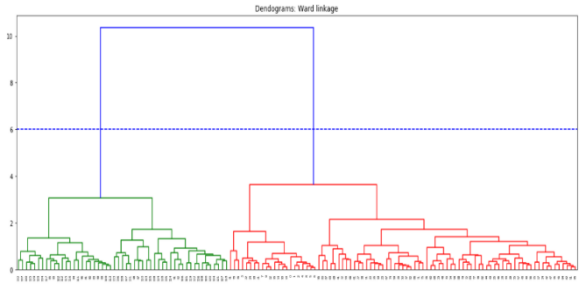
Agglomerative hierarchical clustering:
Agglomerative clustering is a bottom-up approach. Initially, each data point is considered as a cluster and then the most similar two clusters are merged to form a bigger cluster. This step is repeated until all datapoints are the members of one final cluster.

The height of the link (on the vertical axis), indicates dissimilarity between two clusters. The higher the height of the link, less similar are the clusters. It is called the cophenetic distance between two clusters/objects.

Identifying the ideal number of clusters:
I have used two methods to identify the ideal number of clusters:

i) Silhouette scores
ii) Cophenetic distance of the dendrogram





From Silhouette scores: Since the silhouette score of K=2 is the highest, so we will consider 2 clusters. Higher the Silhouette score, well separated the clusters are.

Cophenetic distance of the dendrogram: Higher the Cophenetic distance more dissimilar the clusters. From the above graph, we get two clusters at the highest Cophenetic distance.

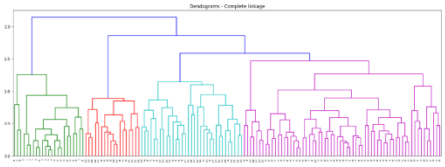
The above two methods suggest that we should take k-2 i.e. two clusters

Justification of selected linkage method:

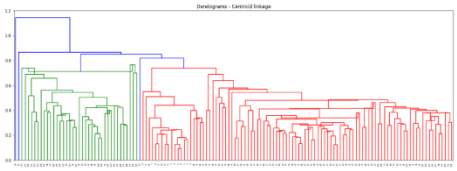
In the agglomerative approach, there are four important methods of merging clusters, which are called linkage methods.

i) Single linkage (nearest neighbour): It merges the clusters based on the distance between the closest members of the two clusters.

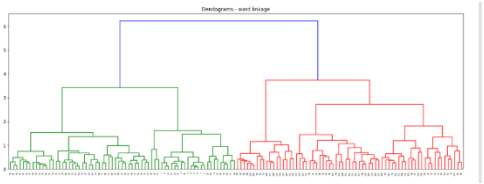
ii) Complete linkage (farthest neighbour): It merges the clusters based on the distance between the farthest members of the two clusters



iii) Centroid linkage: It takes the geometric centroids of clusters to merge them



iv) Wards method: It takes the sum of squared distance of two clusters.

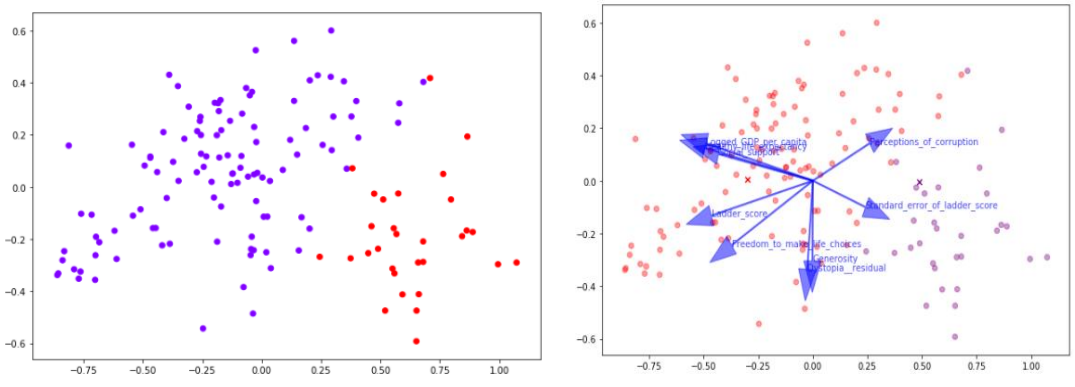


As we have used the sum of squared distance between the datapoints group them into clusters (identify their similarity/dissimilarity) in the above K-means method and since Wards method is the closest to the k-means clustering by it's concept and properties. We will consider the Wards method as our linkage method.

And also in the above methods, single, complete and centroid – out clusters are not well separated.

Projection of each original feature on the principle component axis:

scatterplot of hierarchical clustering and projection of each original feature on the PC axis



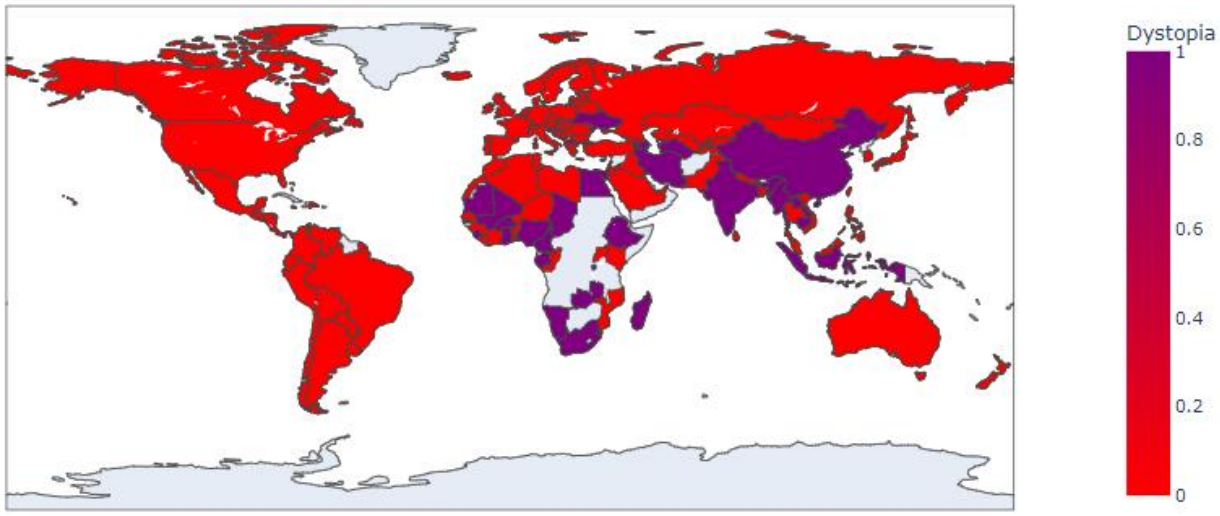
It is evident from the above figure fig that the Logged GDP per capita, Social support and Perceptions of corruption have the most variance and Generosity has the least variance in determining the happiness score.

df1.sort_values('Ladder_score', ascending=False)

After sorting our dataframe by Ladder_score' , we could see utopian countries are in cluster 0.

```
df1.sort_values('Ladder_score', ascending=False)
```

iDP_per_capita	Social_support	Healthy_life_expectancy	Freedom_to_make_life_choices	Generosity	Perceptions_of_corruption	Dystopia__residual	Hierical_cluster
10.775	0.954	72.000	0.949	-0.098	0.186	3.253	0
10.933	0.954	72.700	0.946	0.030	0.179	2.868	0
11.117	0.942	74.400	0.919	0.025	0.292	2.839	0
10.878	0.983	73.000	0.955	0.160	0.673	2.957	0
10.932	0.942	72.400	0.913	0.175	0.338	2.798	0
11.053	0.954	73.300	0.960	0.093	0.270	2.580	0
10.867	0.934	72.700	0.945	0.086	0.237	2.683	0
11.647	0.908	72.600	0.907	-0.034	0.386	2.653	0
10.643	0.948	73.400	0.929	0.134	0.242	2.612	0
10.905	0.934	73.300	0.908	0.042	0.481	2.784	0
10.796	0.940	73.900	0.914	0.159	0.442	2.598	0

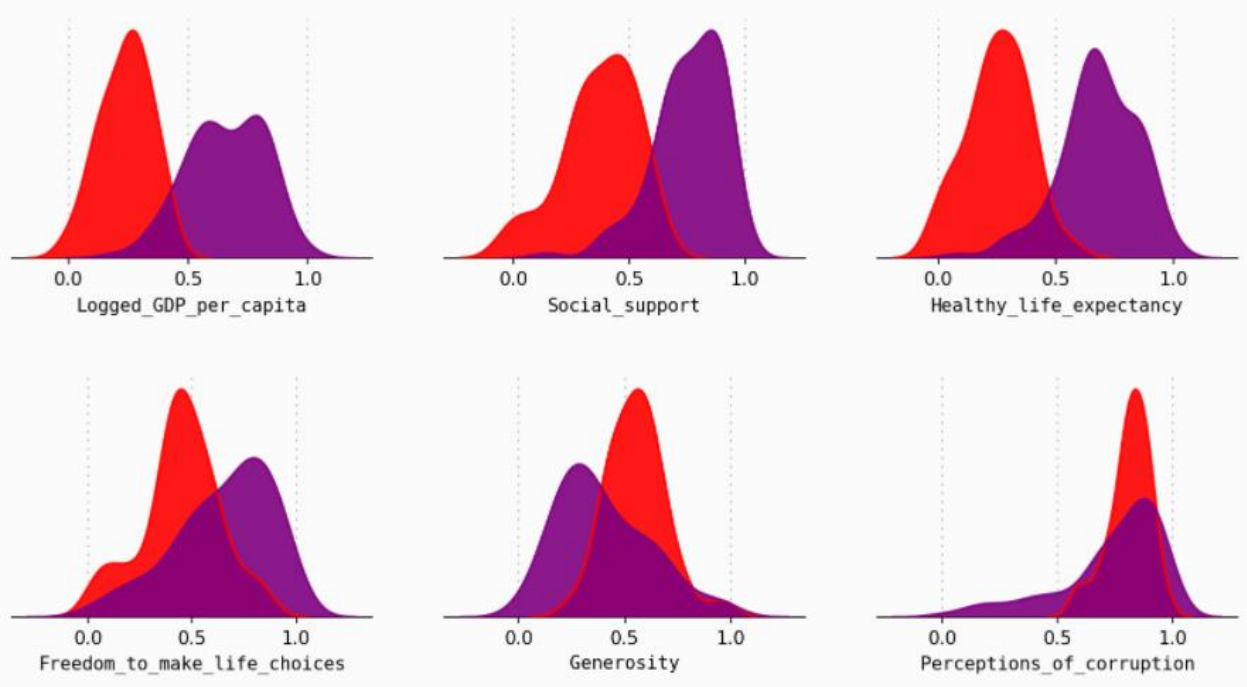


World map based on the hierarchical clustering

Comparison between Dystopian and Utopian countries based : Agglomerative hierarchical clustering

```
scaled_df_AH['Hierical_cluster'].value_counts()
```

```
0    110
1     30
Name: Hierical_cluster, dtype: int64
```



Red – Utopian country

Purple – Dystopian country

From the above graph, it is evident that GDP, Generosity, Freedom to make life choices and Perceptions of corruption are the main factors that contribute to the happiness level of a country.

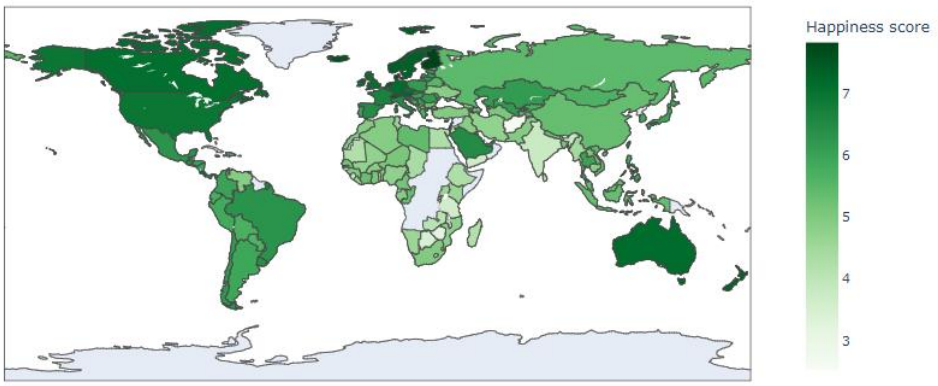
GDP and Social Support are positively correlated with the happiness score of a country, i.e., higher the GDP higher the happiness level and Perceptions of corruption is negatively correlated with the happiness score.
Dystopian countries seem to be more corrupted compared to utopian countries.

It is also interesting to see that both the dystopian and utopian countries are equally generous.

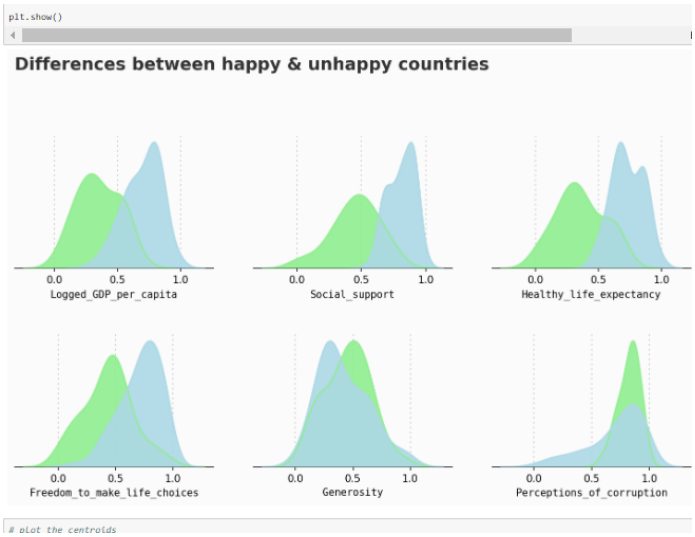
Conclusion:

Our dataset is divided into two clusters, which means we have two groups of countries – Utopian countries (Happy country) and Dystopian countries (Unhappy country).

Below is the world representation of the countries based on the happiness/ladder score.



Comparison between Dystopian and Utopian countries based on all the variables – K means



From the above graph, we can conclude that GDP, Social Support and Perceptions of corruption are the main factors that contribute to the happiness level of a country. However, GDP and Social Support are positively correlated with the happiness score of a country, i.e., higher the GDP higher the happiness level and Perceptions of corruption is negatively correlated with the happiness score.
Dystopian countries seem to be more corrupted compared to utopian countries.

It is also interesting to see that both the dystopian and utopian countries are equally generous.