CSE454

DATA MINING

MIDTERM PROJECT

CAN BEYAZNAR

161044038

Dataset

The name of the dataset used is World Happiness Report. The World Happiness Report is a landmark survey of the state of global happiness. The first report was published in 2012, the second in 2013, the third in 2015, and the fourth in the 2016 Update. The World Happiness 2017, which ranks 155 countries by their happiness levels, was released at the United Nations at an event celebrating International Day of Happiness on March 20th. The report continues to gain global recognition as governments, organizations and civil society increasingly use happiness indicators to inform their policy-making decisions. Leading experts across fields – economics, psychology, survey analysis, national statistics, health, public policy and more – describe how measurements of well-being can be used effectively to assess the progress of nations. The reports review the state of happiness in the world today and show how the new science of happiness explains personal and national variations in happiness.

First of all, we have a total of 5 datasets representing each year (between 2015-2019). There are parameters that affect happiness in this dataset. However, the number and names of these parameters vary over the years. Therefore, when making transactions in these datasets, transactions were made taking into account the values generally belonging to one year. Apart from that, the names of these different parameters are tried to be extracted in the preprocess section.

The following columns: GDP per Capita, Family, Life Expectancy, Freedom, Generosity, Trust Government Corruption describe the extent to which these factors contribute in evaluating the happiness in each country.

The Dystopia Residual metric actually is the Dystopia Happiness Score(1.85) + the Residual value or the unexplained value for each country as stated in the previous answer.

For example, 2015 data looks like this:

```
#Table of 2015 happiness data:
data_2015.head()
```

	Country	Region	Happiness Rank	Happiness Score	Standard Error	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Freedom	Trust (Government Corruption)	Generosity	Dystopia Residual
0	Switzerland	Western Europe	1	7.587	0.03411	1.39651	1.34951	0.94143	0.66557	0.41978	0.29678	2.51738
1	Iceland	Western Europe	2	7.561	0.04884	1.30232	1.40223	0.94784	0.62877	0.14145	0.43630	2.70201
2	Denmark	Western Europe	3	7.527	0.03328	1.32548	1.36058	0.87464	0.64938	0.48357	0.34139	2.49204
3	Norway	Western Europe	4	7.522	0.03880	1.45900	1.33095	0.88521	0.66973	0.36503	0.34699	2.46531
4	Canada	North America	5	7.427	0.03553	1.32629	1.32261	0.90563	0.63297	0.32957	0.45811	2.45176

2019 data looks like this:

```
data_2019.head()
```

	Overall rank	Country or region	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption
0	1	Finland	7.769	1.340	1.587	0.986	0.596	0.153	0.393
1	2	Denmark	7.600	1.383	1.573	0.996	0.592	0.252	0.410
2	3	Norway	7.554	1.488	1.582	1.028	0.603	0.271	0.341
3	4	Iceland	7.494	1.380	1.624	1.026	0.591	0.354	0.118
4	5	Netherlands	7.488	1.396	1.522	0.999	0.557	0.322	0.298

What operations have been applied with this dataset?

First, csv files were read by applying preprocess processes. As mentioned earlier, we have data between 2015-2019. And these are in separate csv files. The parameter names in these datasets are different. For this reason, the names of the parameters in the csv files read in order to eliminate these defects are named in a way that they are common with all datasets. For example :

```
# Editing the name of the parameters in the csv file
# Preprocessing data
data_2015.columns=[each.split()[0] if(len(each.split())>2) else each.replace(" ","_") for each in data_2015.columns]
data_2016.columns=[each.split()[0] if(len(each.split())>2) else each.replace(" ","_") for each in data_2016.columns]
data_2017.columns=[each.replace("."," ") for each in data_2017.columns]
data_2017.columns=[each.split()[0] if(len(each.split())>2) else each.replace(" ","_") for each in data_2017.columns]
data_2018.columns=[each.split()[0] if(len(each.split())>2) else each.replace(" ","_") for each in data_2018.columns]

```

```
data_2019 = data_2019.rename(columns = {'Score': 'Happiness_Score', 'Freedom to make life choices': 'Freedom',
                                        'Healthy life expectancy':'Health'}, inplace=False)
data_2019.columns=[each.split()[0] if(len(each.split())>2) else each.replace(" ","_") for each in data_2019.columns]
```

Next, the datasets were checked for missing values and debugged.

After these operations, visualization was made using mostly data found in 2015. Here, by using different visualizations as possible, it was aimed to examine the effects of factors on people's happiness.

After these operations, visualization was made using mostly data found in 2015. Here, by using different visualizations as possible, it was aimed to examine the effects of factors on people's happiness. Tools were used in these visualizations.

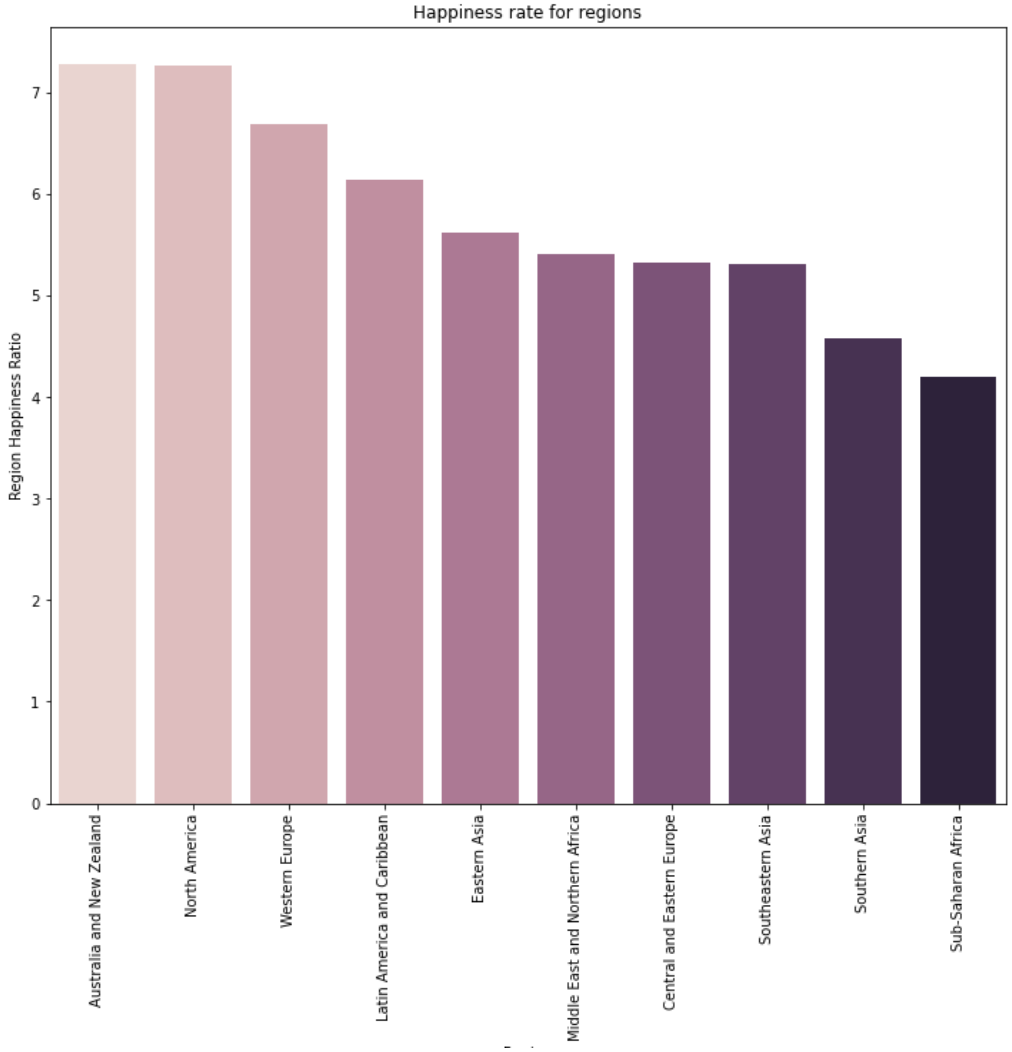
And finally, clustering algorithms have been implemented in accordance with this data set. These algorithms are not readily available.

Visualization

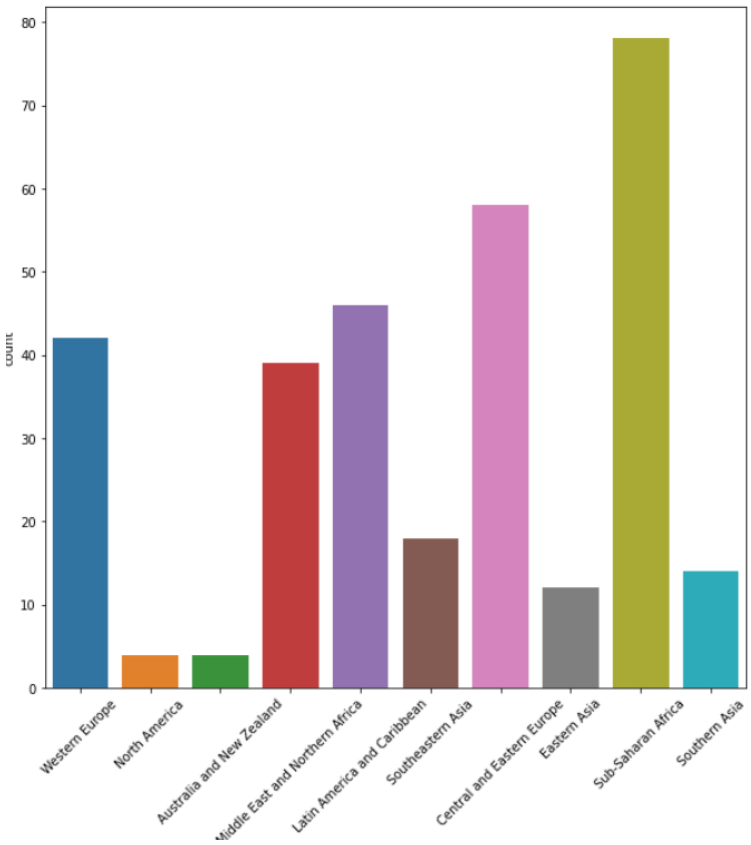
The happiness score of people generally varies according to the living standards of the countries or regions they live in. So what are these regions?

	region
0	Western Europe
1	North America
2	Australia and New Zealand
3	Middle East and Northern Africa
4	Latin America and Caribbean
5	Southeastern Asia
6	Central and Eastern Europe
7	Eastern Asia
8	Sub-Saharan Africa
9	Southern Asia

As it seems, there are 10 regions in total. Let's examine how the happiness score of the countries in these regions ranks.

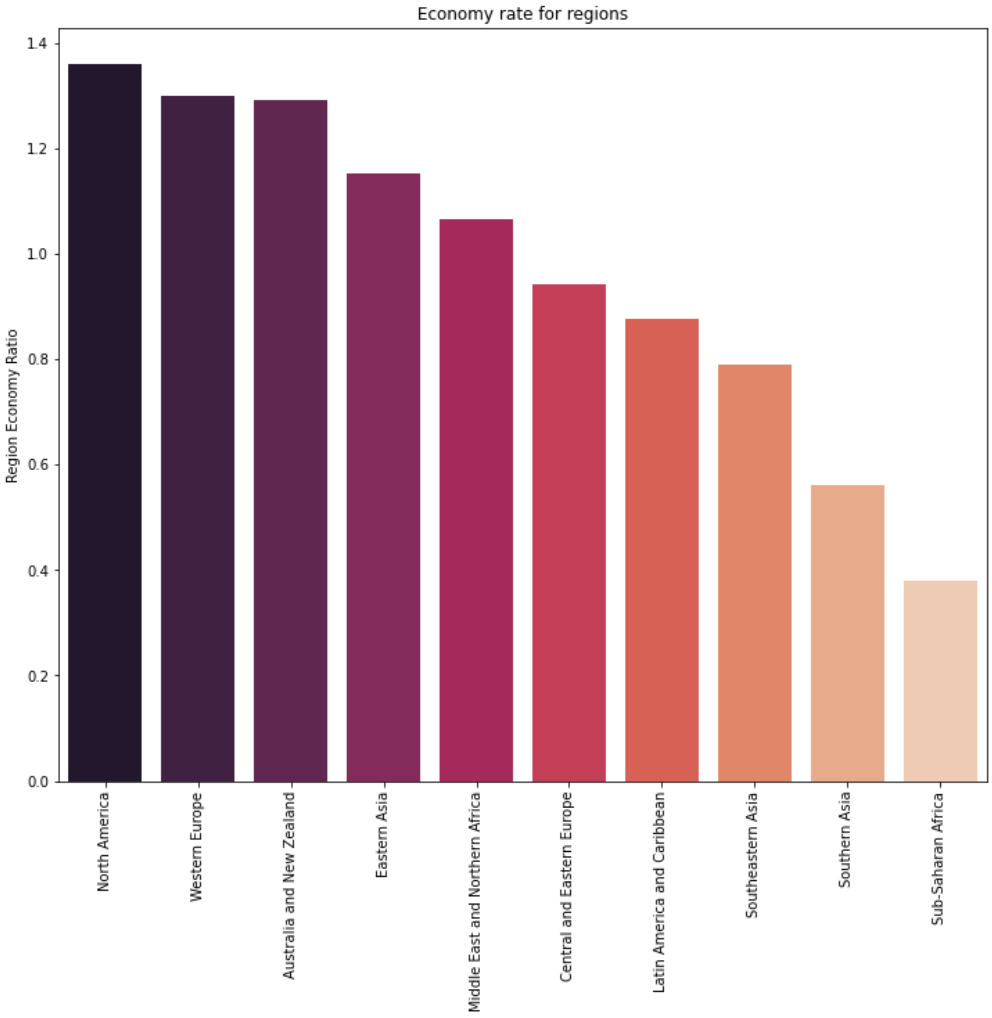


Here you can see the number of countries in the regions.

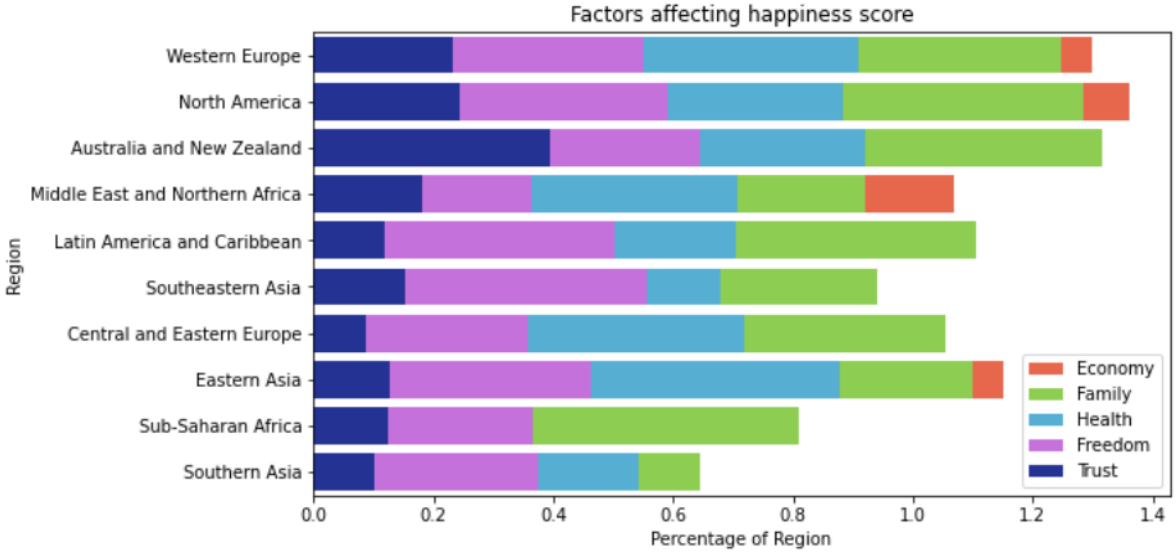


If we create a table ranked from the highest score to the lowest score, we see that Australia and New Zealand are in the first place, North America in the second place and Sub-Saharan Africa in the last place. These scores can vary depending on people's economic status, health, and various factors. So how much do these factors affect?

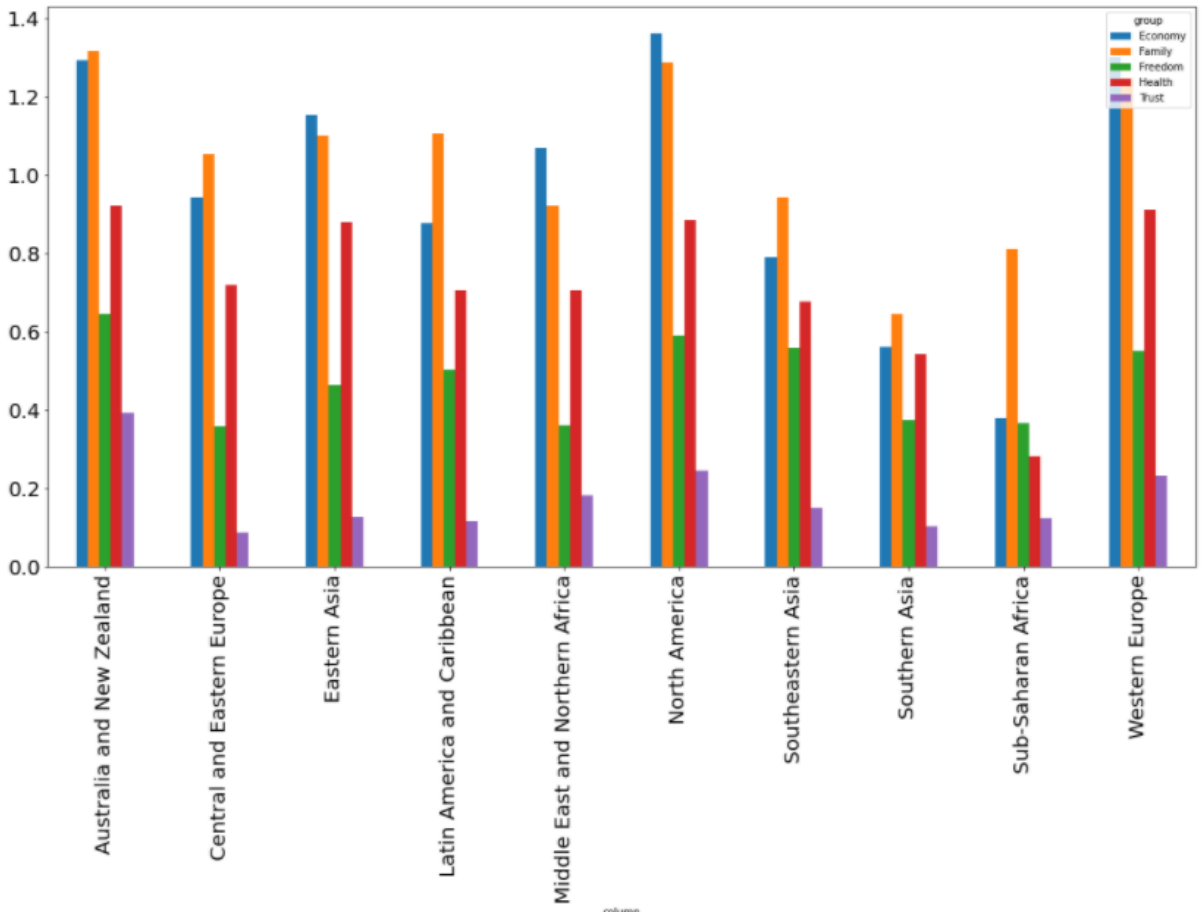
First of all, let's create a small table to show the effect of even one parameter on the happiness score in countries. Let's examine the order of regions according to the economy score.



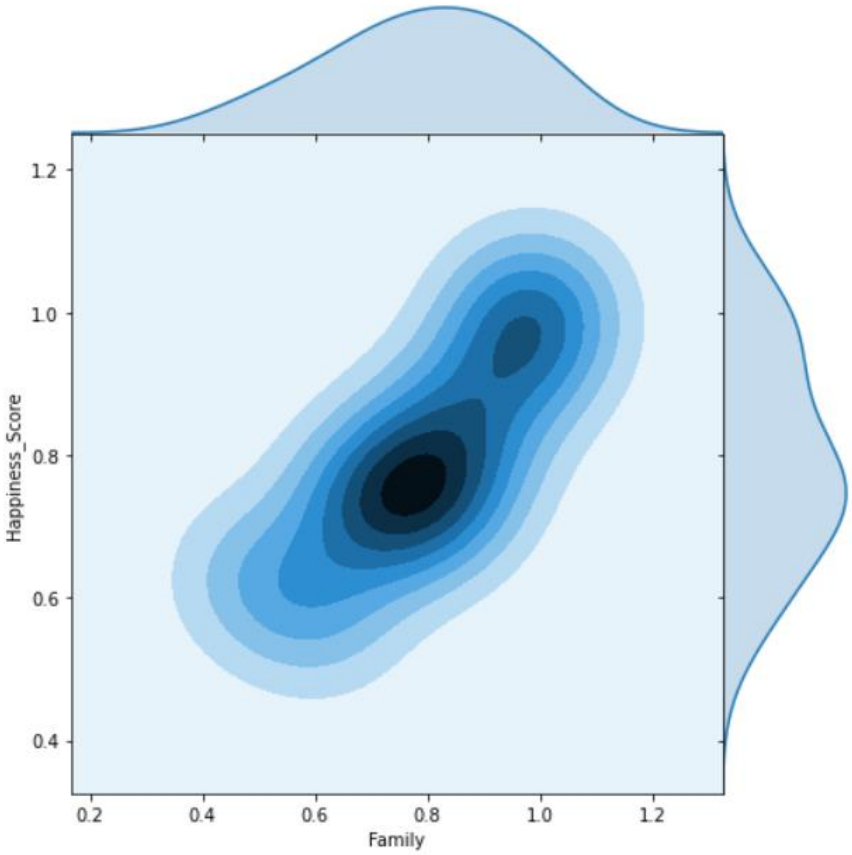
Here we have seen how much even one factor affects the happiness rate. Now let's examine the picture that emerges when we combine more than one factor.



In this table, a total of 5 parameters were examined. These are the most important and influential factors in our lives. Economy, family, health, freedom and trust. Factors in some regions do not appear in the table. Garph bars representing these values are only an overview as they overlap each other. To examine the effect of these factors one by one:

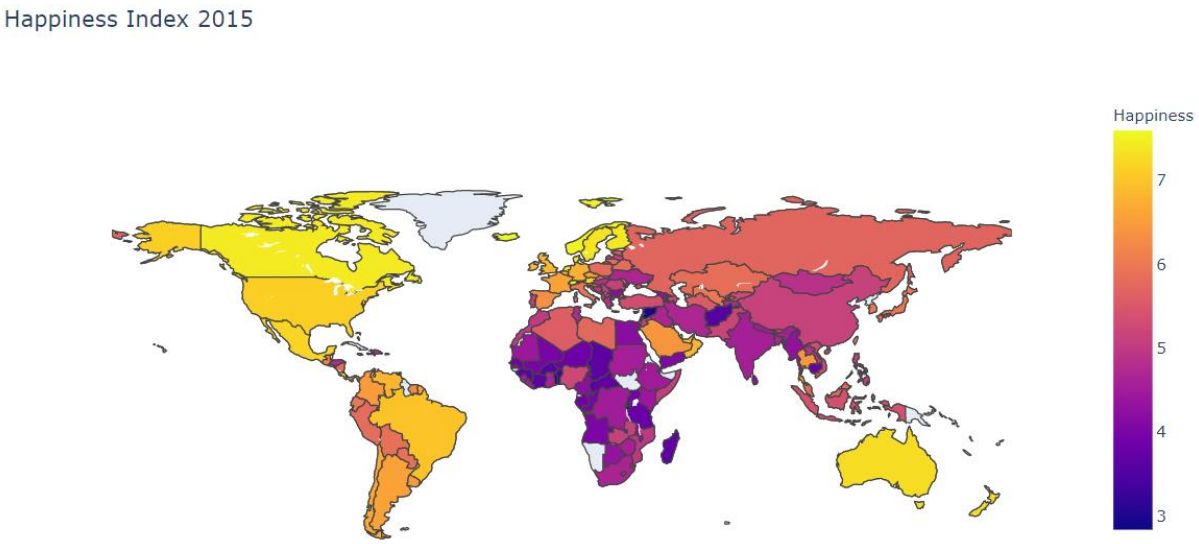


When we examine the graph, we see that the highest values for each region are economy and family. We can see that trust is the lowest. So how do the values of these parameters vary by country? How close are these values between countries? Let's answer these questions based on a factor by creating a chart. For example, let's examine the Family factor.

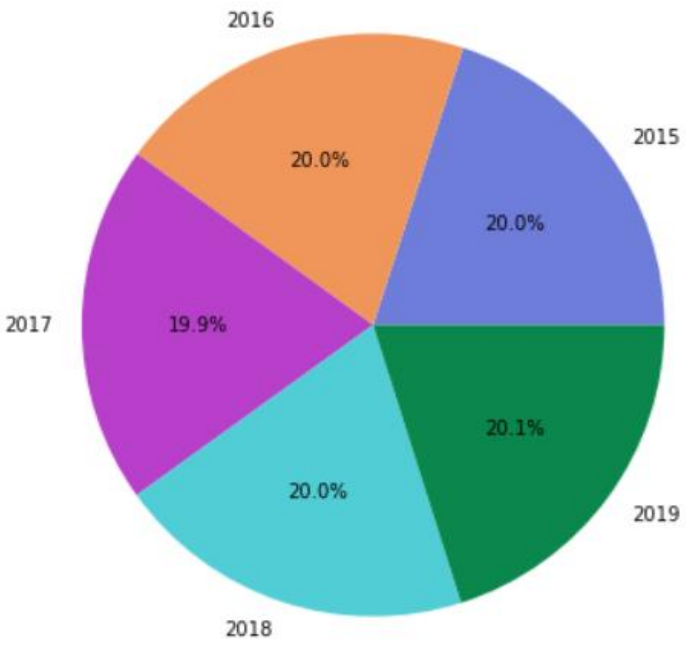


When we look at this graph, we find that there is a positive correlation between family and happiness score.

Let's look at the distribution of the happiness score from the world map.



Using the 2015 dataset, we examined the effects of the factors we have on the happiness score. So, has there been a change from 2015 to 2019? Let's compare the happiness rates between 2015-2019.



Finally, when the pizza chart is examined, we see that the overall rate of happiness has not changed over the years.

Now we have finished the visualization part. Next, we will test the clustering algorithms we have implemented and examine the results.

Clustering Algorithms

While analyzing these algorithms, data from 2015 were used. The dataset related to the happiness score and the Health score was examined.

A) DBSCAN Clustering

Density-based spatial clustering of applications with noise (DBSCAN) is a well-known data clustering algorithm that is commonly used in data mining and machine learning.

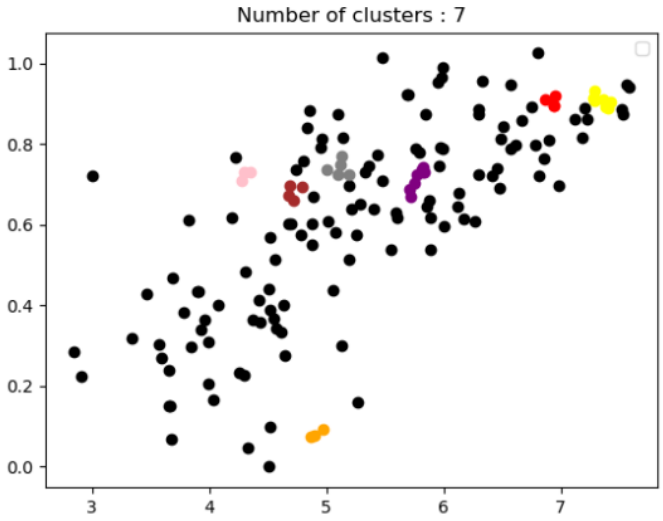
Based on a set of points (let’s think in a bidimensional space as exemplified in the figure), DBSCAN groups together points that are close to each other based on a distance measurement (usually Euclidean distance) and a minimum number of points. It also marks as outliers the points that are in low-density regions.

eps: specifies how close points should be to each other to be considered a part of a cluster. It means that if the distance between two points is lower or equal to this value (eps), these points are considered neighbors.

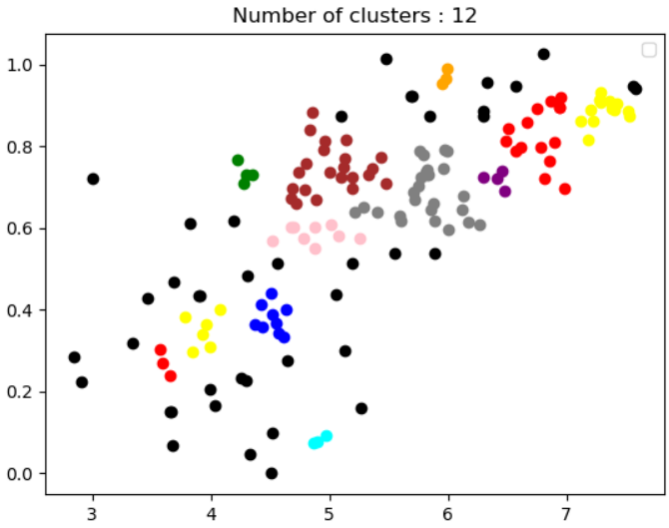
minPoints: the minimum number of points to form a dense region. For example, if we set the minPoints parameter as 5, then we need at least 5 points to form a dense region.

In the dbscan algorithm that I have implemented, the min points and epsilon parameters are input from the user. Results vary according to these two parameters. Let's examine the effect on the dataset we use by changing these parameters.

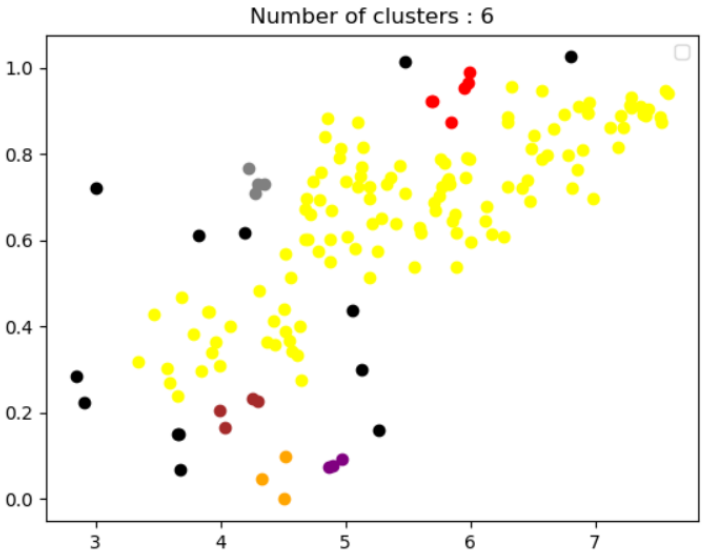
eps=0.1, minpts=3



Eps=0.2, minpts=3



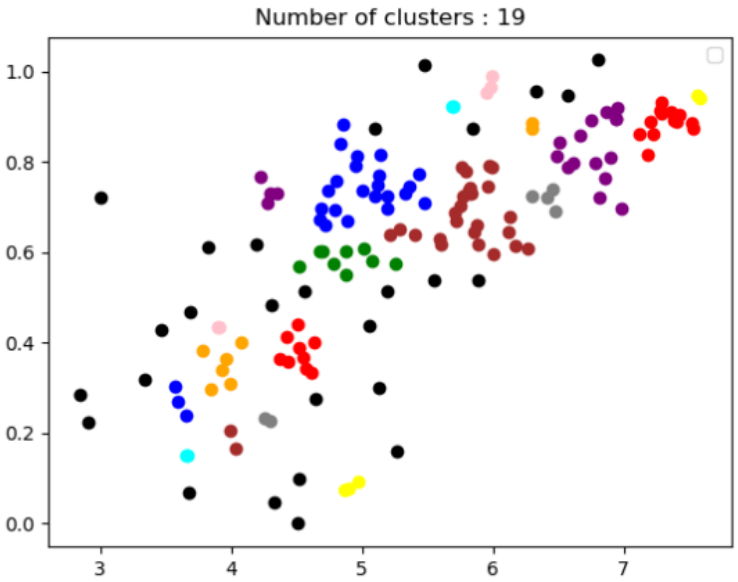
Eps=0.3, minpts=3



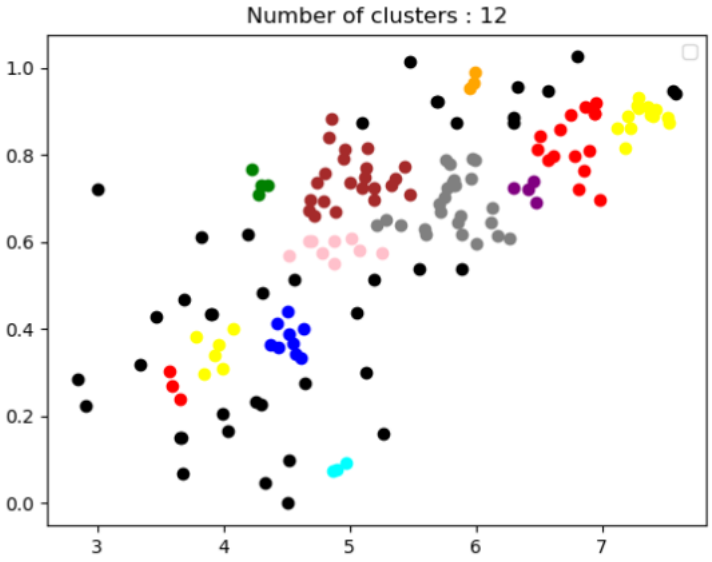
When we change the Epsilon parameter, we see that the number of clusters first increases and then decreases. In the first picture, we see that there is a lot of noisy data. This is because the number of objects in close proximity in the epsilon is very low. Then, when we make the epsilon value 0.2, we see that the number of clusters increases to 12 and the number of data in the clusters increases. We can think of this because the distance between the data is closer to the epsilon. Finally, when we make the epsilon value 0.3, we see that the number of clusters is as in the first graph. However, the number of noise is less. As the epsilon increases, this causes the data that is too far apart to merge. We can think that the ideal epsilon is 0.2 here.

Now let's change the minpts parameter

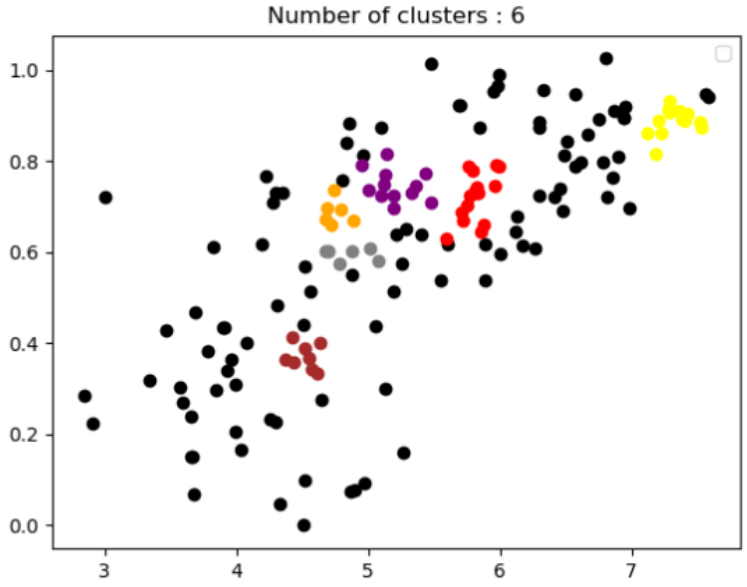
Eps=0.2, minpts=2



Eps=0.2, minpts=3



Eps=0.2, minpts=6



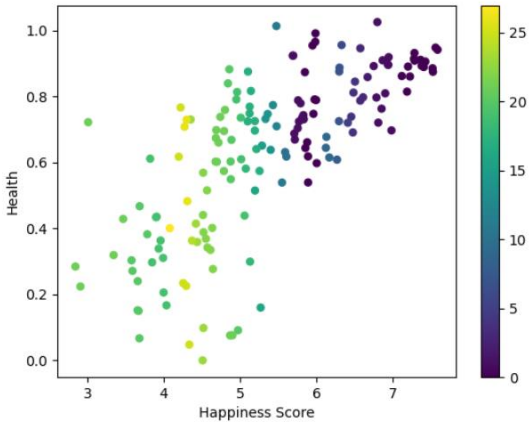
When we change the minpts parameter, we see that the number of clusters decreases as the value increases. It has been observed that the values around are less than the eps = 0.2 value. The ideal minpts value here is 3.

B) Mean Shift Clustering

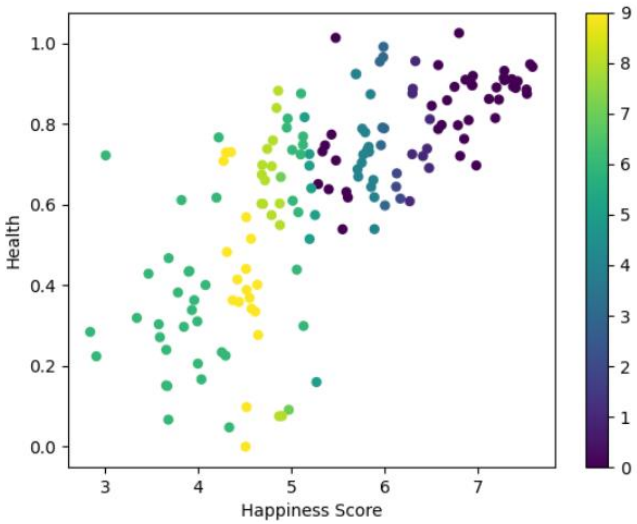
Meanshift is falling under the category of a clustering algorithm in contrast of Unsupervised learning that assigns the data points to the clusters iteratively by shifting points towards the mode (mode is the highest density of data points in the region, in the context of the Meanshift). As such, it is also known as the Mode-seeking algorithm. Mean-shift algorithm has applications in the field of image processing and computer vision.

We have only 1 parameter for mean shift clustering. The remaining parameters are fixed due to the properties of the data we have. When a change was made to these values, healthy results could not be obtained. Therefore, only changes in the kernel bandwidth parameter will be evaluated.

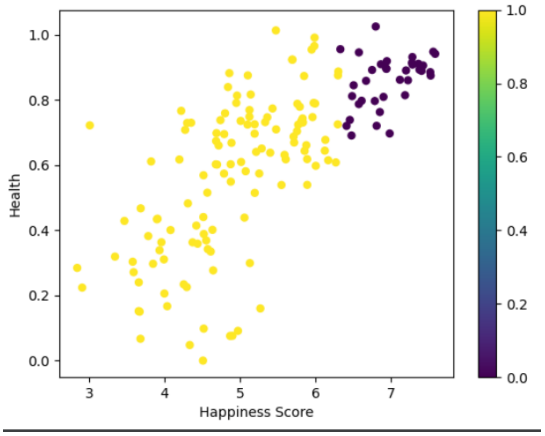
Kernel_bandwitdh=1



Kernel_bandwitdh=3



Kernel_bandwitdh=9



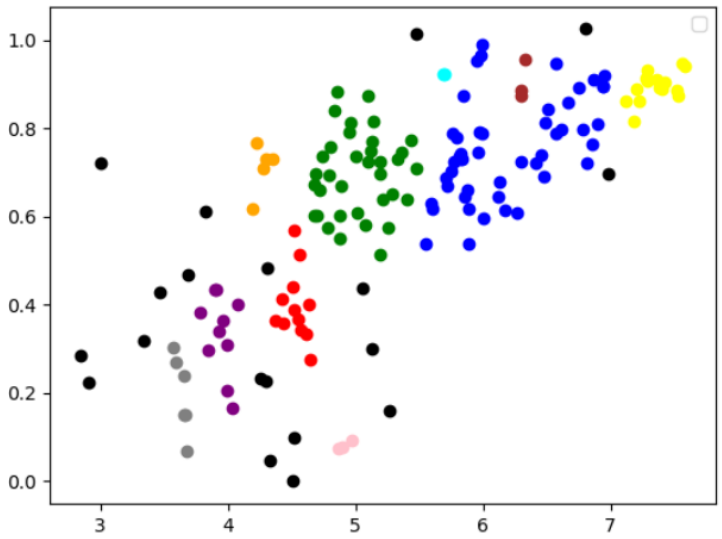
We observe that when we increase the kernel_bandwidth value, the number of clusters decreases. And we can see that the distant points are gathered into a common cluster. The ideal value for the kernel_bandwidth parameter is 3.

C) Agglomerative Hierarchical Clustering

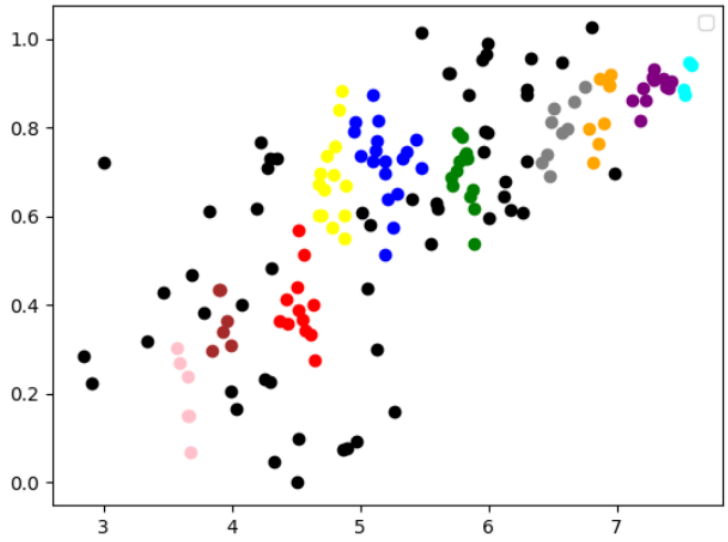
The **agglomerative clustering** is the most common type of hierarchical clustering used to group objects in clusters based on their similarity. It's also known as AGNES (Agglomerative Nesting). The algorithm starts by treating each object as a singleton cluster. Next, pairs of clusters are successively merged until all clusters have been merged into one big cluster containing all objects. The result is a tree-based representation of the objects, named dendrogram.

In this algorithm, outputs will be evaluated through a parameter. Normally it is evaluated over two parameters. These parameters are named clusterCenterNumber and Merge_Ratio. However, our aim here is to be able to cluster the data according to the regions in the world. So we will take the clusterCenterNumber parameter as 10, which is the number of regions in the world.

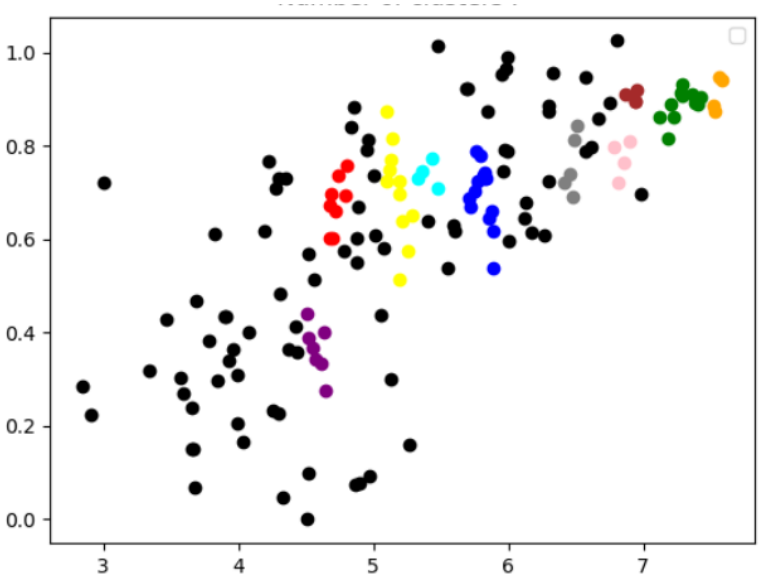
Merge_Ratio=0.165



Merge_Ratio=0.3



Merge_Ratio=0.4



When we look at these results, we observe that as the Merge_ratio value increases, the noise increases. And the number of data within clusters is decreasing. The lower value Merge_ratio parameter has more data in clusters. I think the ideal parameter value here is 0.165. The high number of noise negatively affects the regional clustering of countries. That's why I think a data graph with less noise is healthier.

In this project, I applied the visualization and clustering processes using the dataset containing the happiness report in the world. In the visualization part, we examined what factors affect happiness in countries. In the clustering part, we tried to cluster data across countries by addressing a happiness factor. And I tried classifying between them.

References

1- https://towardsdatascience.com/how-dbscan-works-and-why-should-i-use-it-443b4a191c80
2- https://www.geeksforgeeks.org/ml-mean-shift-clustering/
3- https://www.datanovia.com/en/lessons/agglomerative-hierarchical-clustering/