ST3189

Machine Learning

Coursework Project

Student number:

220682130

2024

Contents

1 Unsupervised learning

DataFrame for Tasks 1 and 2.

The dataset, which was used for the first task contains information about socio-economic and health factors of all of the countries.

1.1 Dataset description

- country - name of country.
- child_mort - death of children under 5 years of age per 1000 live births.
- exports - exports of goods and services per capita, given as %age of the GDP per capita.
- health - total health spending per capita, given as %age of GDP per capita.
- imports - imports of goods and services per capita, given as %age of the GDP per capita.
- income - net income per person.
- inflation - the measurement of the annual growth rate of the total GDP.
- life_expec - the average number of years a new born child would live if the current mortality patterns are to remain the same.
- total_fer - the number of children that would be born to each woman if the current age-fertility rates remain the same.
- gdpp - the GDP per capita.

1.2 Objective

The main objective of this task is to obtain a clustering unsupervised machine learning algorythm, which would successfully identify homogeneous country groups. It is also essential to define the exact reasonable number of clusters and to describe each cluster after the algorythm implementation.

1.3 EDA

Initially, before implementing the algorithms, it is important to conduct the explanatory data analysis to identify outliers, suspicious values or variables in our dataframe.

	count	mean	std	min	25%	50%	75%	max
child_mort	167.0	38.270060	40.328931	2.6000	8.250	19.30	62.10	208.00
exports	167.0	41.108976	27.412010	0.1090	23.800	35.00	51.35	200.00
health	167.0	6.815689	2.746837	1.8100	4.920	6.32	8.60	17.90
imports	167.0	46.890215	24.209589	0.0659	30.200	43.30	58.75	174.00
income	167.0	17144.688623	19278.067698	609.0000	3355.000	9960.00	22800.00	125000.00
inflation	167.0	7.781832	10.570704	-4.2100	1.810	5.39	10.75	104.00
life_expec	167.0	70.555689	8.893172	32.1000	65.300	73.10	76.80	82.80
total_fer	167.0	2.947964	1.513848	1.1500	1.795	2.41	3.88	7.49
gdpp	167.0	12964.155689	18328.704809	231.0000	1330.000	4660.00	14050.00	105000.00

	Unique	%Unique	Null	%Null	Zero	%Zero
country	167	100.000000	0	0.0	0	0.0
child_mort	139	83.233533	0	0.0	0	0.0
exports	147	88.023952	0	0.0	0	0.0
health	147	88.023952	0	0.0	0	0.0
imports	151	90.419162	0	0.0	0	0.0
income	156	93.413174	0	0.0	0	0.0
inflation	156	93.413174	0	0.0	0	0.0
life_expec	127	76.047904	0	0.0	0	0.0
total_fer	138	82.634731	0	0.0	0	0.0
gdpp	157	94.011976	0	0.0	0	0.0

Figure 1.1: **On the left:** dataframe description. **On the right:** the amount of unique, null and zero values among all variables.

At first sight, there is nothing to be suspicious about, however further analysis is required, thus I have decided to take a look at the correlation matrix.

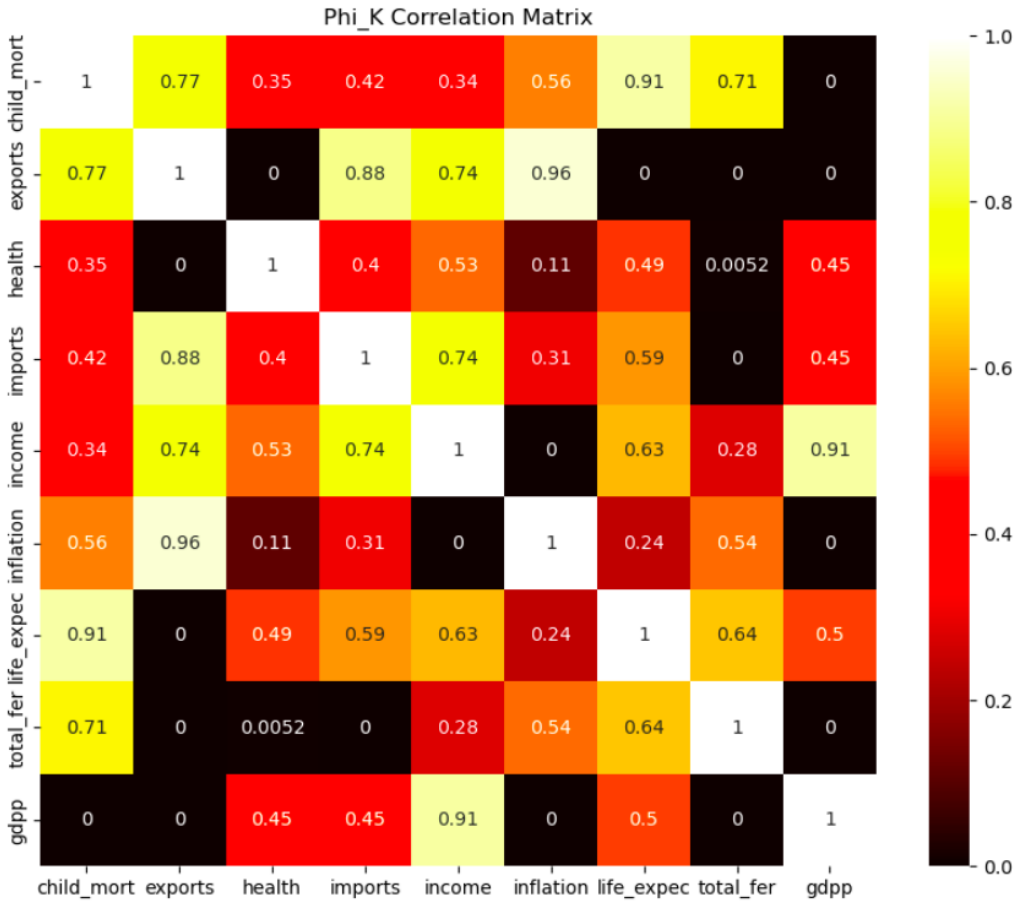
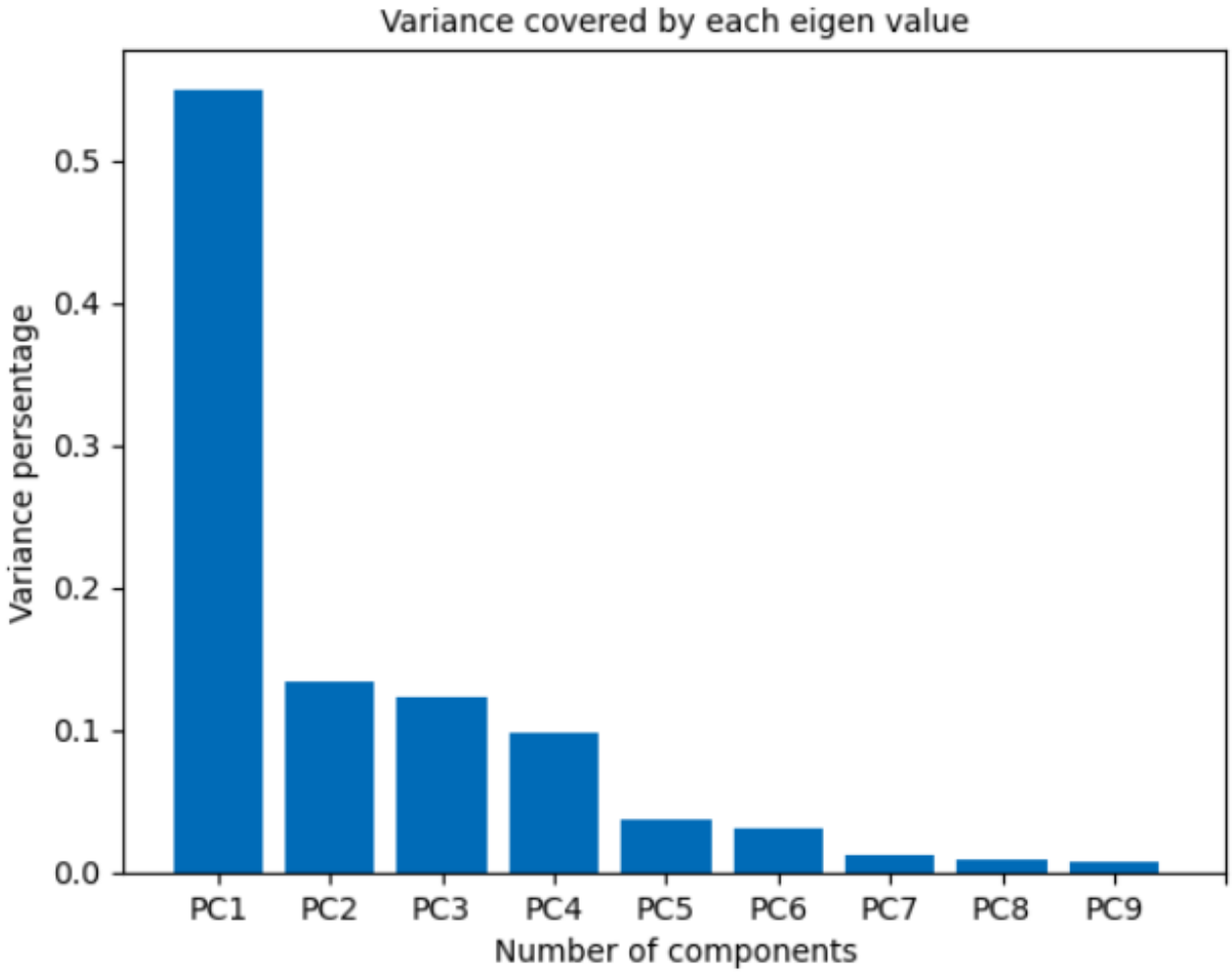


Figure 1.2: Correlation matrix for the dataframe variables.

As it can clearly be seen there is quite high correlation between some of the variables, that is the **child_mort and total_fer, life_expec**, which can be explained by the nature of the origin of those variables; **export and inflation, imports**, **income and gdpp**, which seems reasonable due to export, import and income being directly included in the GDP, as well as inflation being calculated with relation to GDP.

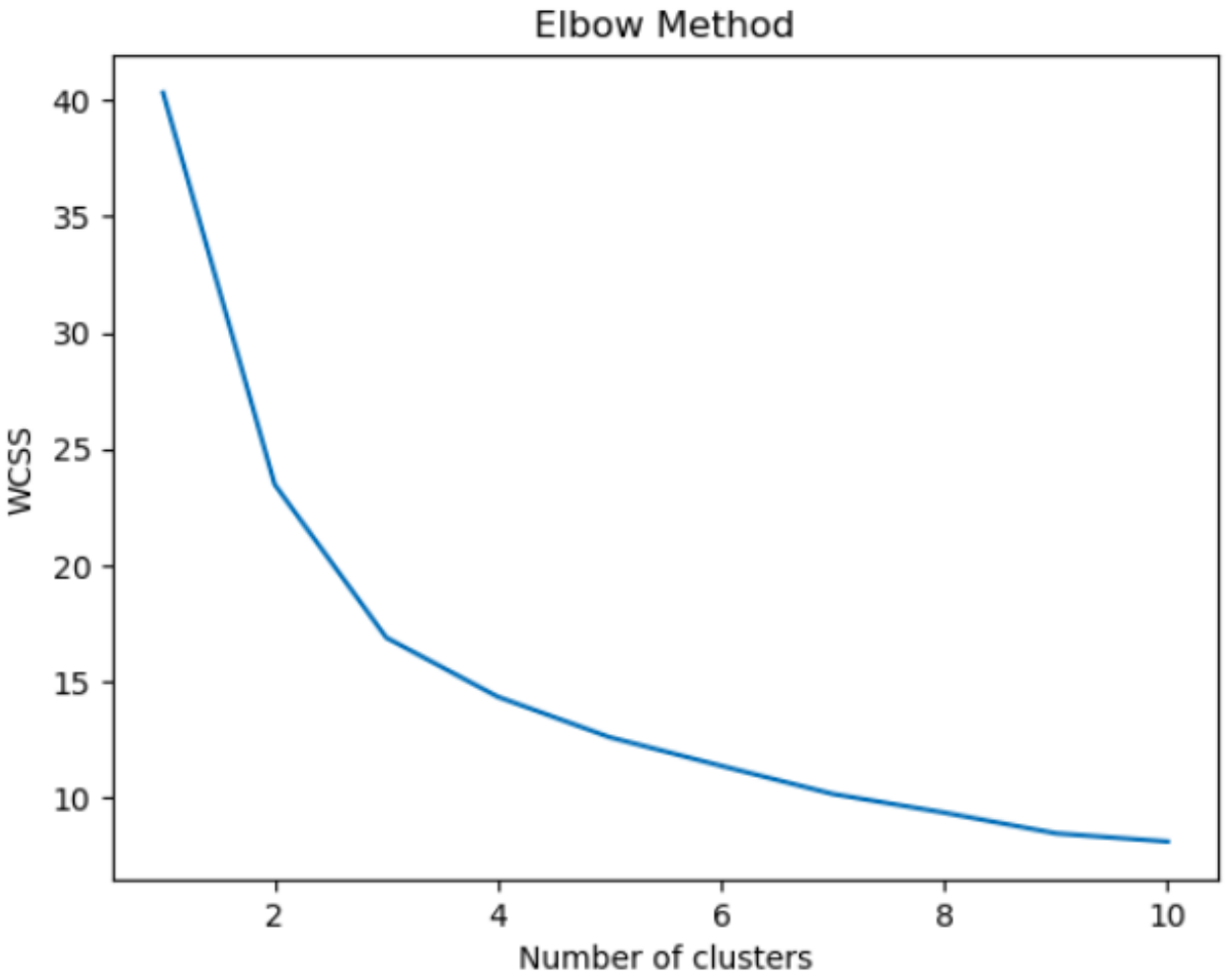
1.4 Main Part

The data was scaled and the principal component analysis was applied to it, due to the relatively high dimensionality and the big spread among the values. The results are below.



Around 95% of variance is the usual percentage for eigen value, which according to the conducted elbow method is corresponding to the five components, thus 5 components are chosen.

Then, the elbow method is conducted to determine the appropriate number of clusters for this particular task.



After the elbow method implementation it is reasonable to choose 3 clusters. Choosing 2 clusters might also seem to be optimal, but there are risks of low detailing.

Finally, the clustering is done with the usage of two algorithms, which have provided almost the same results. I have chosen the KMeans algorithm. The country map with the clustering map is provided below.

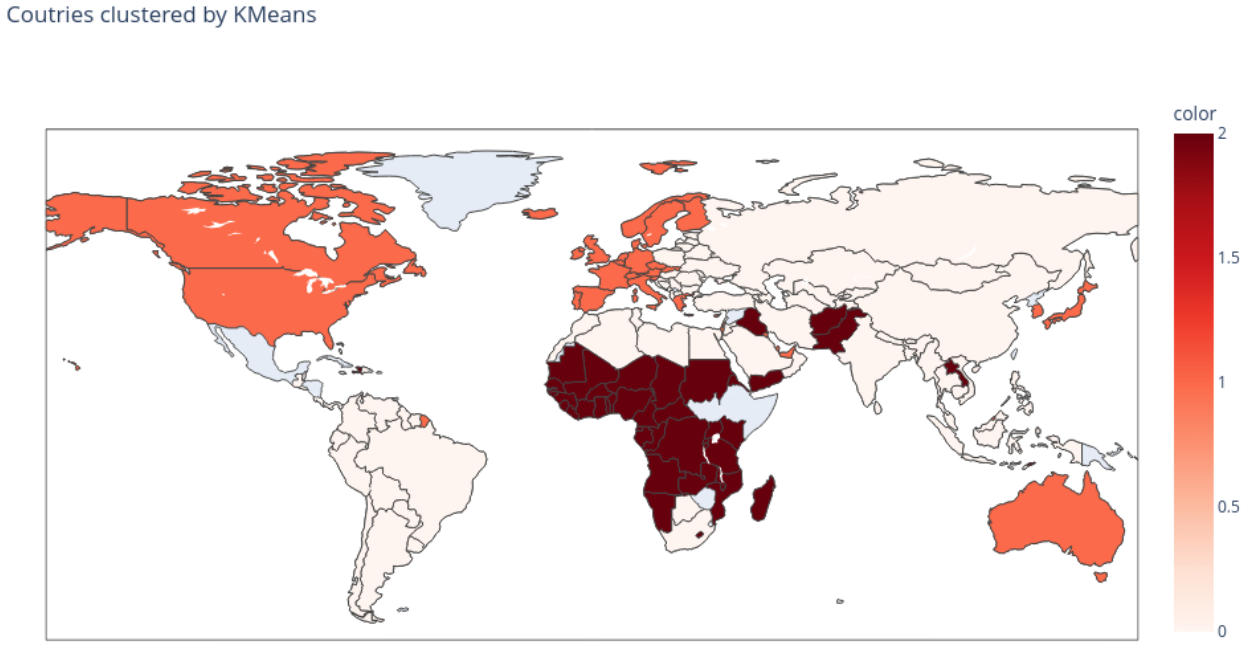


Figure 1.3: On the map: the **dark red colour** stands for the countries in dire need of aid, based on the socio-economic and health factors, provided in the dataset. The **red colour** stands for the countries with leading socio-economic and health factors, which could be advanced. The countries of the **white colour** are developing countries, which does not require special aid.

2 Regression

2.1 Dataset description

For this task the same dataset was used, except for the health variable, which was used as a target variable for regression and popped from the dataframe.

2.2 Objective

The task was to try to predict the total health spending per capita, given as %age of GDP per capita, based on the other socio-economic and health-related factors.

2.3 EDA

It is essential that we check for the multi-collinearity among the variables and the target variable, thus I have once again returned to the correlation matrix, luckily there were no such highly correlated variables, therefore we can enhance with our study without changes to the data.

2.4 Preprocessing

The data was separated into the train and test parts, the scaling was applied to it.

2.5 Main Part

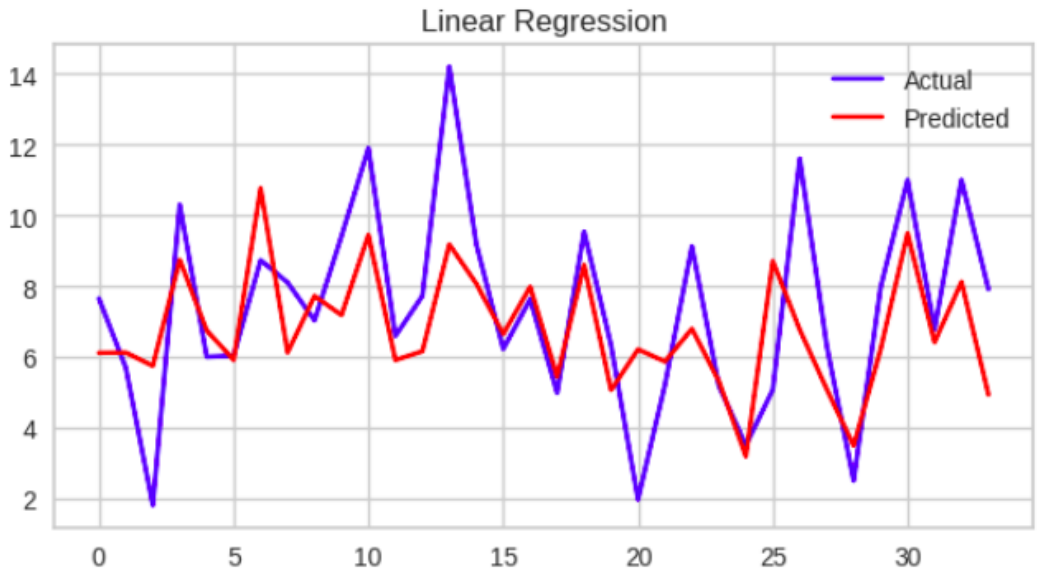
After the three models, that is the linear, the lasso and the ridge regressions were implemented the following root mean squared errors were obtained.

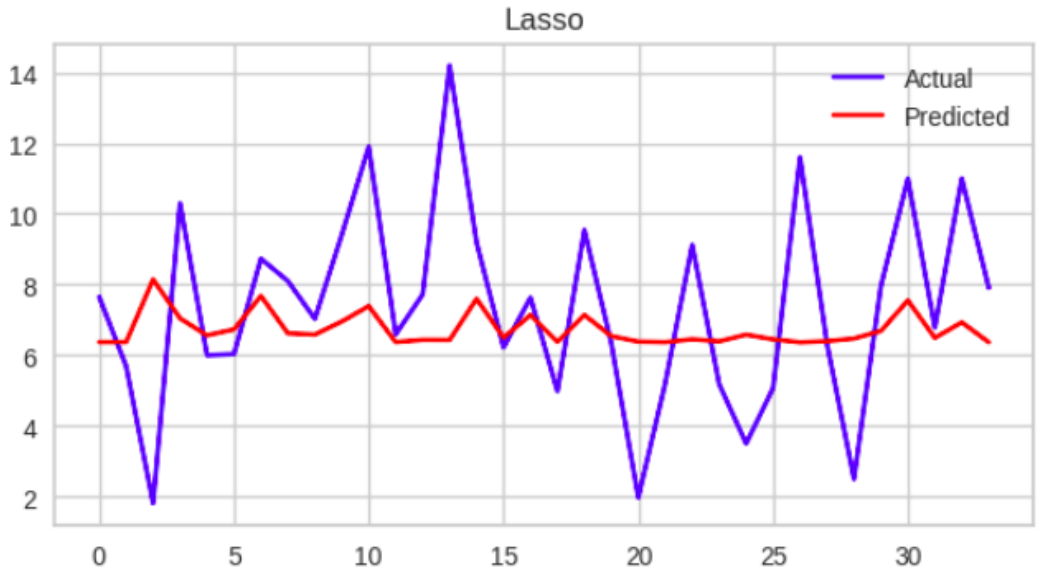
```
print(np.sqrt(mean_squared_error(vY, lr_pred)))
print(np.sqrt(mean_squared_error(vY, lasso_pred)))
print(np.sqrt(mean_squared_error(vY, ridge_pred)))
```

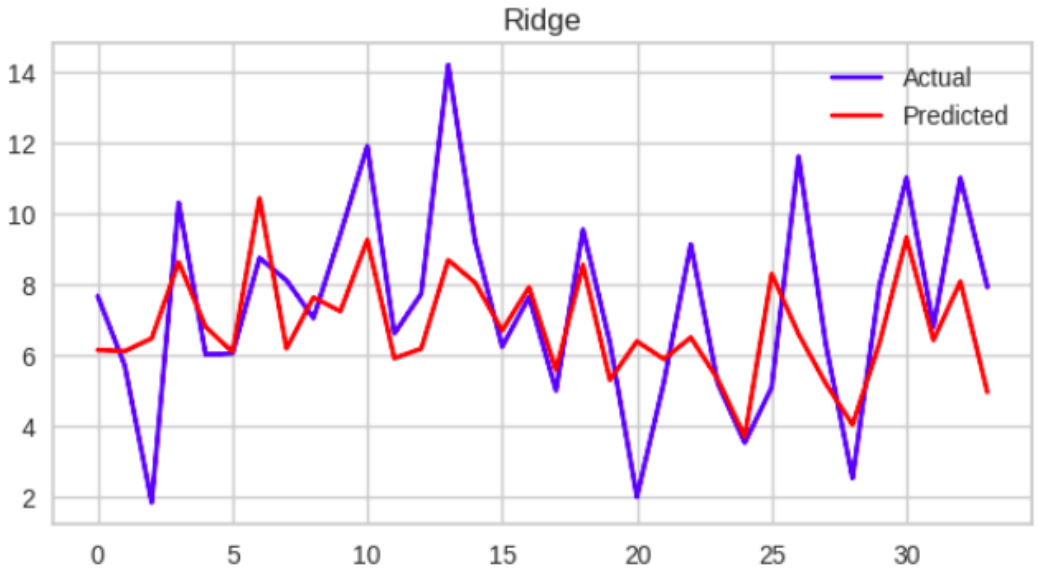
```
2.1602116020067146
2.8323396761791075
2.254739466903066
```

Figure 2.1: The linear regression, the lasso regression and the ridge regression root mean squared errors respectively.

For greater clarity, there is the visual representation of the obtained results.







Based on the visual results and on the results of the root mean squared error the basic linear regression is chosen as best and most satisfactory.

3 Classification

DataFrame for Task 3.

The data is about Asteroids - NeoWs. NeoWs (Near Earth Object Web Service) is a RESTful web service for near earth Asteroid information. With NeoWs a user can: search for Asteroids based on their closest approach date to Earth, lookup a specific Asteroid with its NASA JPL small body id, as well as browse the overall data-set.

3.1 Data Description

Unfortunately, the full data description would take too much space, as the dataset contains of the 40 uqique columns. Here is a full list of the columns, as well as some insights into the dataframe.

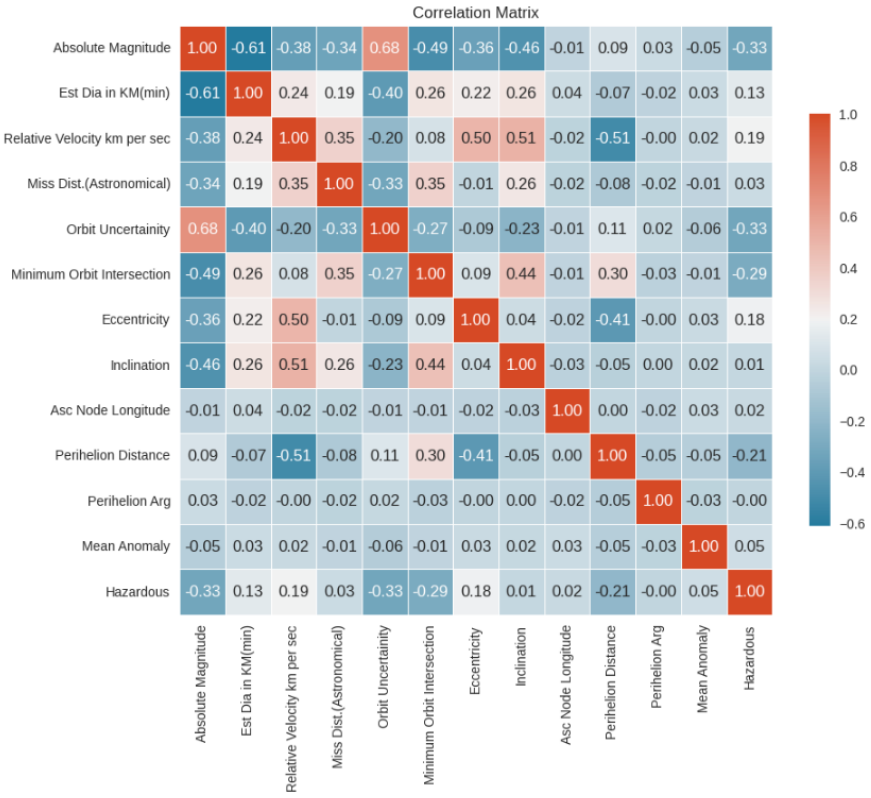
```
Data columns (total 40 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   Neo Reference ID              4687 non-null   int64
 1   Name                          4687 non-null   int64
 2   Absolute Magnitude            4687 non-null   float64
 3   Est Dia in KM(min)            4687 non-null   float64
 4   Est Dia in KM(max)            4687 non-null   float64
 5   Est Dia in M(min)             4687 non-null   float64
 6   Est Dia in M(max)             4687 non-null   float64
 7   Est Dia in Miles(min)         4687 non-null   float64
 8   Est Dia in Miles(max)         4687 non-null   float64
 9   Est Dia in Feet(min)          4687 non-null   float64
 10  Est Dia in Feet(max)          4687 non-null   float64
 11  Close Approach Date           4687 non-null   object
 12  Epoch Date Close Approach     4687 non-null   int64
 13  Relative Velocity km per sec  4687 non-null   float64
 14  Relative Velocity km per hr   4687 non-null   float64
 15  Miles per hour                4687 non-null   float64
 16  Miss Dist.(Astronomical)      4687 non-null   float64
 17  Miss Dist.(lunar)             4687 non-null   float64
 18  Miss Dist.(kilometers)        4687 non-null   float64
 19  Miss Dist.(miles)             4687 non-null   float64
 20  Orbiting Body                 4687 non-null   object
 21  Orbit ID                      4687 non-null   int64
 22  Orbit Determination Date      4687 non-null   object
 23  Orbit Uncertainity            4687 non-null   int64
 24  Minimum Orbit Intersection    4687 non-null   float64
 25  Jupiter Tisserand Invariant   4687 non-null   float64
 26  Epoch Osculation              4687 non-null   float64
 27  Eccentricity                  4687 non-null   float64
 28  Semi Major Axis               4687 non-null   float64
 29  Inclination                   4687 non-null   float64
 30  Asc Node Longitude            4687 non-null   float64
 31  Orbital Period                4687 non-null   float64
 32  Perihelion Distance           4687 non-null   float64
 33  Perihelion Arg                4687 non-null   float64
 34  Aphelion Dist                 4687 non-null   float64
 35  Perihelion Time               4687 non-null   float64
 36  Mean Anomaly                  4687 non-null   float64
 37  Mean Motion                   4687 non-null   float64
 38  Equinox                       4687 non-null   object
 39  Hazardous                     4687 non-null   bool
```

However, it is essential to say that the target for this classification task is the "Hazardous" variable, which consists of either true or false values.

3.2 EDA

The very first step was the data description, which has provided some insights on the usefulness of some of the id variables in this dataset, which were then removed, as they does not posess any valuable information for the classification. Then the correlation matrix was obtained, but due to the rather large it cannot be adequately represented as a figure fully. The matrix has shown that there are several variables, which possess the same information, but in different number systems. Thus they were removed, as well as the variables with high correlation, which is

considered to be above 0.8 or less than -0.8.



After all transformation, the remaining variables were scaled, due to the large range of values and several models for classification were used, among which the XGBClassifier has shown the best results, thus it was chosen as the final model with the accuracy of 99.3%.