İstanbul
Bilgi University
LAUREATE INTERNATIONAL UNIVERSITIES

CMPE 407 - Machine Learning
Term Project Report

by
Batuhan Şeremet
118200054

Istanbul Bilgi University
Spring 2021

Chapter 1

Introduction

Since December 2019, the term coronavirus is a huge part of the world. A disease first seen in Wuhan, China; spread worldwide and effected everybody's life. World Health Organization announced the pandemic in March 2020 and since it's beginning, more than 150 millions of cases have been confirmed with more than 3 million deaths. The pandemic is still going on and it has been one of the most effective pandemics of the human history. It leads billions of people to wear face masks, keep their social distance and be in lockdown.In the second half of 2020, several vaccines for COVID-19 have been developed and since then, millions of people are being vaccinated around the world.

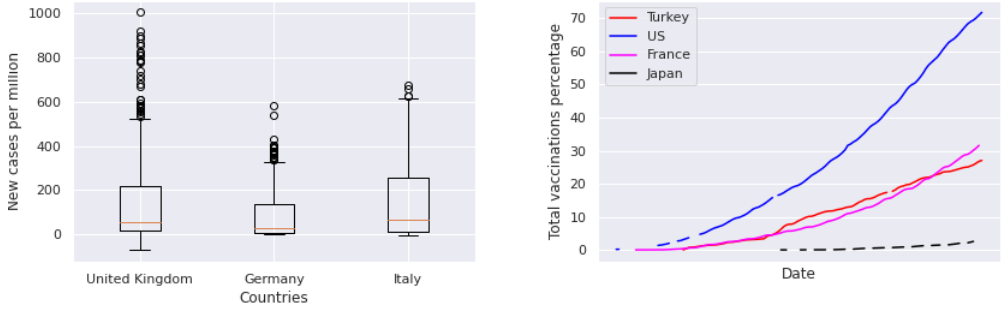
This project aims to find the fully vaccinated people(who took 2 doses) per hundred for each country, for a given time. To achieve this goal, verified COVID-19 data[1] is used. Machine Learning method which is suitable for this goal is Regression. Different Regression models are used to get better results.

[1]https://github.com/owid/covid-19-data/tree/master/public/data

Chapter 2

Dataset

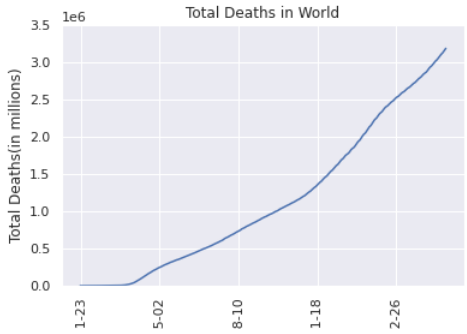
The COVID-19 dataset collected by *Our World in Data*, includes coronavirus based information provided by WHO, the governments and Ministries/Secretaries of Health of the governments.It has 85171 examples from various countries and dates, with 59 features such as; total cases, new cases, total deaths, new deaths, life expectancy, human development index, population, male and female smokers, people fully vaccinated etc.[1]

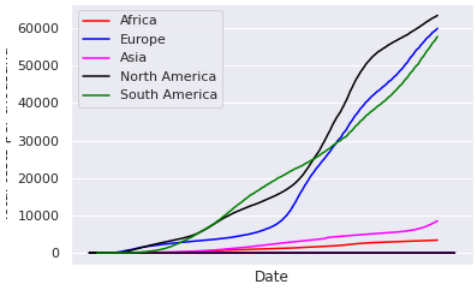


The boxplot on the left shows new cases per million for United Kingdom, Germany and Italy. Their mean values are close to each other but Germany seems more consistent than the others. United Kingdom has so many outliers, from the days which the pandemic was going bad in there. The graph on the right shows the total vaccination percentage of Turkey, United States, France, Japan. United States' vaccination percentage increased way more

[1]The data is updated daily, the dataset which the project will work on was downloaded at May 2, 2021.

faster than the other 3 countries. Turkey and France seems to have vaccinated near amounts of their populations. Japan seems like they are slow at vaccinations.





The graph on the left shows the death count of the world due to coronavirus. Unfortunately, more than 3 million people have died. The amount of deaths increased exponentially. The graph on the right shows the total cases per million for 5 continents. Africa and Asia seems to have way less cases than Europe and American continents. The reason may be the emptiness of the information or people's request for a test. Also, Asia has more than half of the worlds population, which may lead to better results than the other continents, in the case of good protection.

Chapter 3

Preprocessing

Dataset initially had 85171 examples. The project aims to predict the people_fully_vaccinated_per_hundred feature which is not null in only 6431 examples.

Firstly, 14 features are dropped because of the reasons they are null in most of the data or they are related to other features in the dataset. The

examples which does not provide continent or population are dropped. After this operations, 80960 examples with 45 features are remaining. First fully vaccinated people data is at 27 December, 2020 but total vaccinations and people vaccinated features are provided since 15 December. Since fully vaccinated means people who took 2 doses of vaccine, empty fully vaccinated values of examples with not-null total vaccinations and people vaccinated can be filled with this features using the following formula:

$$people\_fully\_vaccinated = total\_vaccinations - people\_vaccinated$$

The examples before 15 December, 2020 are can't be filled so they are dropped. Also 9 features are dropped because the information they provide are reachable with the per million/hundred versions of this features and population. After dropping these, there are 27412 examples with 36 features.

By using the formula above, people fully vaccinated column of examples are filled. Using that column and population, people fully vaccinated per hundred column is filled. Since it still has lots of null values, empty features with country x and date $t-1$ can be filled with same country's features at date t using backfill method. After this operations, 'total_vaccinations','people_vaccinated', 'people_fully_vaccinated' features are dropped. Since enough of empty examples are filled, examples with null 'people_fully_vaccinated_per_hundred' can be dropped.

After this operations, there are 8966 examples remaining. Also 4 features are dropped because they have so much empty data. The remaining null values are filled with means of the features. Indexes of the dataset has reset. After looking at the correlation map, 5 features are dropped due to their high correlation with other features. Finally, the y (target) column is extracted from the data. Finally, the data has 8966 examples with 23 features.

Since the continent, location and date columns are objects, they need to be converted to the numbers before feeding them to the model. LabelEncoder method of sklearn is used to do this operations.

The dataset is a time-series so, in case of wanting to make a prediction at some day t, the data should be trained with the days before t, not with the data after it. To apply this, the last $\frac{x}{5}$ examples of countries with x examples, where x is higher than or equal to 5, are moved to the test set. 1715 of the examples have moved to the test set, while 7251 of them stayed, as training set. After this, indexes of X and y has reset because their iloc and loc should be same to split again for cross-validation. PredefinedSplit is used to

do cross-validation based on time. The same approach, the last $\frac{x}{5}$ examples of countries with more than or equal to 5, set to be used in cross-validation folds. The total number of folds selected as 5.

As the last step of the preprocessing, standardization of dataset is done by sklearn's StandardScaler.

Chapter 4

Experiments

To predict the selected target, the technique suitable is regression. Linear Regression, Neural Networks, Decision Trees, Random Forests and Extreme Gradient Boosting(XGBoost) are used. Linear Regression is special for regression and the others are can be used for different techniques as well as regression.

Linear Regression is the basic algorithm which tries to minimize $\sum_{i=1}^{m} (\hat{y}^{(i)} - y^{(i)})^2$ with coefficients w, where m is number of examples in the training set.

Linear Regression has the structure input layer $\rightarrow$ output layer. Neural Networks can be simplified as input layer$\rightarrow$ hidden layers$\rightarrow$ output layer.

Decision Trees are tree-like models which all branches of the tree are possible outcomes.Random Forests use multiple decision trees with different subset of features and average their predictions. XGBoost is a computationally efficient gradient boosting algorithm. It starts with training a decision tree on the data. Then, it trains another decision tree to fix the mistakes that the previous tree has made, and it goes on.

Linear Regression is selected as the starting point, first algorithm to see how accurate can a basic model can be on the dataset. Neural Networks, Decision Trees, Random Forests and Support Vector Machines are considered as models to be tuned but Support Vector Machines are deleted from that list because training even one Support Vector Machine takes so much time.

In the case of not getting good results with these algorithms, XGBoost is selected as the most advanced algorithm.

In order to get better results, different hyper-parameters are tuned for different algorithms and using GridSearchCV, the highest scored hyper-parameters of each algorithm is stored in a dataframe. Since all of the models have R^2 score has their default evaluation metric, no changes are made for this parameter.

Linear Regression does not have any hyper-parameters to tune, so there is only model from this algorithm. Because of that, instead of GridSearchCV, cross_val_score method is used, with predefined cross-validation splits. The average score of this model is 0.047404.

For Neural Network, hidden layer sizes and initial learning rate hyper-parameters are tuned. The best model from this algorithm has hidden layer sizes=(30,) and initial learning rate=0.01, the average score of this model, on cross-validation is 0.829394.

For Decision Trees, max depth and min samples split hyper-parameters are tuned. The best model from this algorithm has max depth=10 and min samples split=2, the average score of this model, on cross-validation is 0.693566.

For Random Forests, number of estimators, max depth and min samples split hyper-parameters are tuned. The best model from this algorithm has number of estimators=500, max depth=10 and min samples split=2, the average score of this model, on cross-validation is 0.827313.

For XGBoost, number of estimators, max depth and learning rate hyper-parameters are tuned. The best model from this algorithm has number of estimators=700, max depth=5 and learning rate=0.1, the average score of this model, on cross-validation is 0.894918.

Chapter 5

Results

Table 5.1: Best models with their parameters for each algorithm.

ALGORITHM	**HYPER-PARAMETERS**	**R^2 score**
Linear Regression	None	0.047404
Neural Network	hidden layer size=(30,) initial learning rate=0.01	0.829394
Decision Tree	max depth=10 min sample split=2	0.693566
Random Forest	number of estimators=500 max depth=10 min sample split=2	0.827313
XGBoost	number of estimators=700 max depth=5 learning rate=0.1	0.894918

Linear Regression is clearly the worst algorithm out of this 5, for this problem. It is miles away from the others. Since the problem is complex, Linear Regression could not find good coefficients to get good results.

Since Random Forest is like an improved Decision Tree and XGBoost is like an improved Random Forest; also because of the problem is not so basic,

XGBoost became the most successful one.

The best model of the best algorithm XGBoost is trained with the dataset and predicted on test set. It got the R^2 score of 0.923 and mean squared error of it is 15.23.

It predicts very good but for some examples it makes huge error. The example with the maximum error is from the country Seychelles. It does not provide death values continuously and some of it's critical features are filled with the mean of the features.

Chapter 6

Conclusion

The aim is selected as predicting the fully COVID-19 vaccinated people percentage of countries by date. The dataset is visually analyzed and many pre-processing expressions have been executed. 5 different regression algorithms are tuned with different hyper-parameters and best one is selected as XGBoost. XGBoost model with selected hyper-parameters is trained with the training set and predicted on test set. It was very successful at predicting the fully vaccinated people percentage of countries.

In order to increase the performance, more data can be collected. Dataset had 85 thousands of examples but for this aim, most of this data was not useful since the vaccinations have started way after the COVID-19 started. Also the examples, which lead the model to learn and predict at wrong way can be deleted with outlier analysis.

Chapter 7

References

1. Hasell, J., Mathieu, E., Beltekian, D. et al. A cross-country database of COVID-19 testing. Sci Data 7, 345 (2020).
2. https://scikit-learn.org/stable/modules/classes.html
3. https://machinelearningmastery.com/xgboost-for-regression/
4. https://xgboost.readthedocs.io/en/latest/parameter.html