COVID-19 outcome prediction analysis

Project Overview:

The rapid spread of the novel coronavirus disease 2019 (COVID-19) has become a health challenge worldwide. At this time, spread forecasting using AI and ML methods is an important task to track the growth of the pandemic and it's effect on people's health. The main focus of this project is to track the population's recuperate of the virus country-wise as well as globally based on some pre-defined standard symptoms, and to perform Logistic regression, Support vector machine, K-Nearest Neighbors, Naïve Bayes, Decision Tree and Prophet on the COVID-19 daily level informational data to forecast the number of death cases caused by the pandemic . Depending on the study of the impact of some parameters such as Country and Gender, etc…., in prediction of weather the case of interest shall survive or not.

Problem statement

The provided dataset is available from 22 Jan, 2020. And contains the reported daily level confirmed information on the number of affected cases, deaths and recovery from 2019 novel coronavirus. The data is mainly used to make a prediction of whether a person is going to recover from corona virus symptoms or not with the consideration of the pre-defined standard symptoms acquired by the given guidelines by the World Health Organization (WHO).

Data description

The dataset contains 14 major variables (13 features that shows symptoms and one label Column) that will be having an impact on whether someone has recovered or not.

The description of each variable are as follows:

1. Country: where the person resides
2. Location: which part in the Country
3. Age: Classification of the age group for each person, based on WHO Age Group Standard
4. Gender: Male or Female
5. Visited_Wuhan: whether the person has visited Wuhan, China or not
6. From_Wuhan: whether the person is from Wuhan, China or not
7. Symptoms: there are six families of symptoms that are coded in six fields.

13. Time_before_symptoms_appear:
14. Result: death (1) or recovered (0)

Data preprocessing

Data preprocessing was not our most challenging task in this project as the data was already clean, processed and ready to be used directly in the upcoming tasks. The only preprocessing task was Re-scaling the data (Explained in details below).

An appropriate data frame has been used so the data becomes as follows:

	location	country	gender	age	vis_wuhan	from_wuhan	symptom1	symptom2	symptom3	symptom4	symptom5	symptom6	diff_sym_hos	result
0	104	8	1	66.0	1	0	14	31	19	12	3	1	8	1
1	101	8	0	56.0	0	1	14	31	19	12	3	1	0	0
2	137	8	1	46.0	0	1	14	31	19	12	3	1	13	0
3	116	8	0	60.0	1	0	14	31	19	12	3	1	0	0
4	116	8	1	58.0	0	0	14	31	19	12	3	1	0	0

Dimensions of the data

The number of rows and columns in data is :

```
# shape of dataset
df.shape
```

```
(863, 14)
```

Overall description of the data:

	location	country	gender	age	vis_wuhan	from_wuhan	symptom1	symptom2	symptom3	symptom4	symptom5	symptom6	diff_sym_hos	result
count	863.000000	863.000000	863.000000	863.000000	863.000000	863.000000	863.00000	863.000000	863.000000	863.000000	863.000000	863.000000	863.000000	863.000000
mean	76.645423	16.995365	0.849363	49.400000	0.181924	0.107764	12.13905	28.002317	18.298957	11.840093	2.993048	0.998841	0.995365	0.125145
std	39.200264	7.809951	0.726062	15.079203	0.386005	0.310261	3.99787	7.473231	2.864064	1.183771	0.127251	0.034040	2.358767	0.331075
min	0.000000	0.000000	0.000000	2.000000	0.000000	0.000000	0.00000	0.000000	0.000000	0.000000	0.000000	0.000000	-5.000000	0.000000
25%	45.000000	11.000000	0.000000	40.000000	0.000000	0.000000	14.00000	31.000000	19.000000	12.000000	3.000000	1.000000	0.000000	0.000000
50%	87.000000	18.000000	1.000000	49.400000	0.000000	0.000000	14.00000	31.000000	19.000000	12.000000	3.000000	1.000000	0.000000	0.000000
75%	110.000000	24.000000	1.000000	57.000000	0.000000	0.000000	14.00000	31.000000	19.000000	12.000000	3.000000	1.000000	1.000000	0.000000
max	138.000000	33.000000	2.000000	96.000000	1.000000	1.000000	24.00000	31.000000	19.000000	12.000000	3.000000	1.000000	15.000000	1.000000

Scaling the data→ preprocessing

This process has been performed to enhance the training criteria and make it goes faster as it prevents the optimizer from getting stuck in any local optima points. and gives a better error surface shape for a better conclusion.

```
scaler = MinMaxScaler()
x_train_scaled = scaler.fit_transform(x_train)
x_test_scaled = scaler.transform(x_test)
```

The process has been performed using → "MinMaxScaler"

```
array([[0.82608696, 0.87878788, 1.        , ..., 1.        , 1.        ,
        0.25      ],
       [0.84782609, 0.54545455, 0.5       , ..., 1.        , 1.        ,
        0.25      ],
       [0.15942029, 0.54545455, 0.5       , ..., 1.        , 1.        ,
        0.4       ],
       ...,
       [0.85507246, 0.21212121, 0.        , ..., 1.        , 1.        ,
        0.25      ],
       [0.82608696, 0.87878788, 1.        , ..., 1.        , 1.        ,
        0.25      ],
       [0.08695652, 0.36363636, 0.5       , ..., 1.        , 1.        ,
        0.25      ]])
```

After exploring the data we went directly to the classification phase as we trained the data using the following five different classifiers, holding a comparison in order to indicate the classifier achieves the best accuracy:

- K-Nearest Neighbors
- Logistic Regression
- Naïve Bayes
- Decision Trees
- Support Vector Machines

GridSearch:

We started to use the gridsearch technique with cv equals to 10 in order to find the optimal hyperparameters which results in the most 'accurate' predictions could be obtained from the implemented algorithms.

Note:

The GridSearch technique couldn't be applied to the Naive Bayes Classifier as there were no hyperparameters to perform the tuning on . However, we tried it ourselves to ensure this and also tried the model using validation data and get the best recall score was obtained using the last method.

Accuracy:

In order to compare the performance of all classifiers we used different metrics such as the

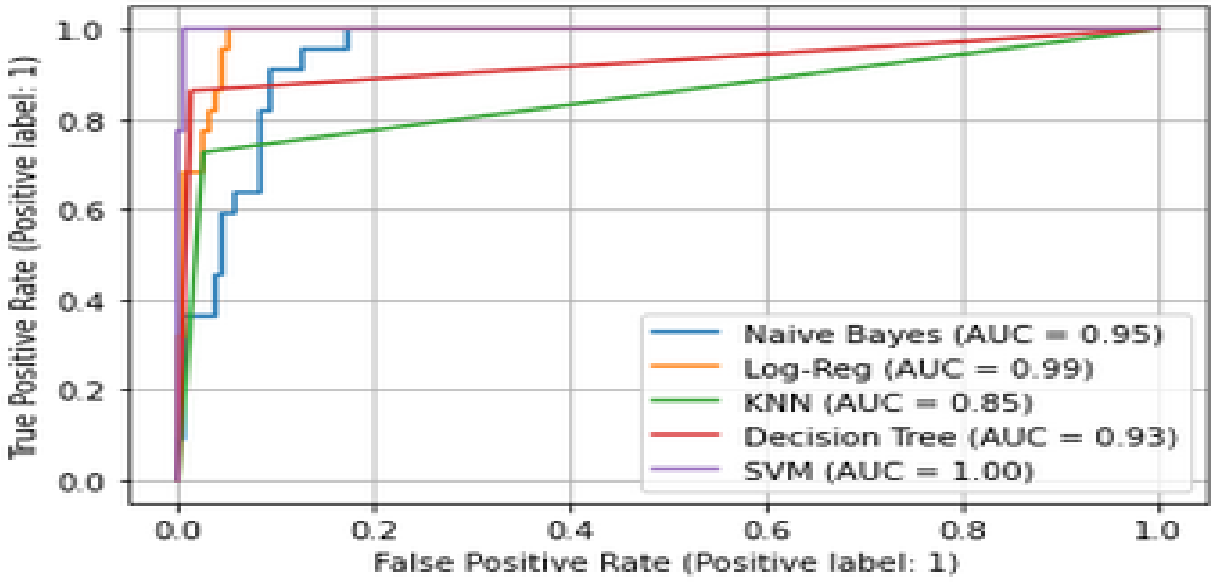
precision, recall, F1-score, and AUC curve.

The following table clarifies the accuracy we achieved after tuning the parameters in each classifier in order to maintain the best accuracy.

Model	Accuracy training	Accuracy testing	Recall	precision	F1 score	AUC
K-Nearest Neighbors	1.0	0.94	0.72	0.8	0.76	0.85
Logistic Regression	0.94	0.94	0.77	0.80	0.79	0.99
Naïve Bayes with grid	0.886	0.884	0.4	0.562	0.474	0.95
Decision Trees	1.0	0.97	0.86	0.90	0.88	0.93
Naive Bayes with Validation data	0.185	0.196	1.0	0.1366	0.2404	0.94
SVM	0.99	0.99	1.0	0.95	0.97	1.00

In order to establish the selection of the model with the highest score, We started plotting thee confusion matrix and ROC curve and AUC for each one of them.

The following figure contains a combined plotting for all of them with the AUC result stated below.



Conclusion

As a conclusion based on the analysis of the observations, it seems that by data modeling and prediction based on univariate timeseries, using Logistic regression, Support vector machine, K-Nearest Neighbors, Naïve Bayes, Decision Tree. Concluded that Support vector machine performed the best accuracy = 0.99, **recall** =1.0 and **AUC**=1.0.