Milestone 2

Amritpal Singh (301336774)

Abhay Jolly (30138885)

Harnoor Singh (301355738)

CMPT 459 - SPECIAL TOPICS IN DATABASE SYSTEMS (3)

Data Mining

Dr. Martin Ester

Fall 2020

2.1 Introduction

Our dataset consists of 557,340 rows, which we got after joining the two given dataset after cleaning. We first joined based on the Province which was able to join about 440,000 rows, we joined the rest based on Country. This is how our dataset was prepared. After splitting the dataset into an 80:20 ratio, we got 445,872 rows for our training data and 111,468 for our testing data. We used the following command to split our data.

- *from sklearn.model_selection import train_test_split*
- *X_train, X_valid, y_train, y_valid = train_test_split(X, y, test_size=0.2)*

2.2 Training the Models

We trained and tested a total of 3 classifiers.

1. Gaussian Naive Bayes (GaussianNB)
2. Random Forest Classifier
3. XGBoost (XGBClassifier)

We used GaussianNB and Random Forest Classifier from the sci-kit-learn library, and the XGBClassifier is used from the XGBoost library. All of the three classifiers were trained on a random 80:20 split of the full data. After using the .fit method to train the data for all the classifiers the pickle library was used to save the models (in the same working directory).

```
target = "RandomForestModel.pkl"
pickle.dump(model,open(target, "wb"))
```

Fig 1: Saving the random forest

2.3 Evaluation

We Imported the saved models and used them to predict the test data, also we calculated the k-fold cross-validation score for each model. K-fold accuracy is a good metric because it's a way to resample the data on k distinct folds i.e k distinct train and test splits, this ensures that every observation from the original dataset has the chance of appearing in the training and test set and would reduce any bias in the sampling process.

The evaluation process involves the accuracy score of test and train data. The confusion matrix was implemented for each machine learning model. Through the matrix, we were able to find the true positive, true negative, false positive and false negative values for each class.

```
Train Accuracy      : 69.12%
Test Accuracy       : 68.54%
10-fold Accuracy    : 68.95%
```

```
Train Accuracy      : 81.26%
Test Accuracy       : 81.17%
10-fold Accuracy    : 80.45%
```

```
Train Accuracy      : 89.67%
Test Accuracy       : 89.02%
10-fold Accuracy    : 88.45%
```

Fig 2: (a) Gaussian Model Score (b)Random-Forest Model Score (c) XGboost Score

The most important things to note here were the training score, testing score, and k-fold accuracy. If the training score is very less than the testing score, this means that our model is overfitting on the training data and not performing well on the test data (i.e. not predicting well on testing data / unseen data). The K-fold accuracy was used to reduce the performance bias on one set of train-test splits.

This is a multi-class classification problem where we are trying to predict between the 4 classes.

So a Confusion Matrix seemed a right measure as it is used to evaluate the performance of a classification model, we mainly used misclassification rate i.e how often our model was making wrong predictions. Which is *(False-positive rate + False-negative rate) / total number of predictions,* we tried using different

hyperparameters and tried to minimize the misclassification rate. Below is the confusion matrix for the Random Forest classifier.

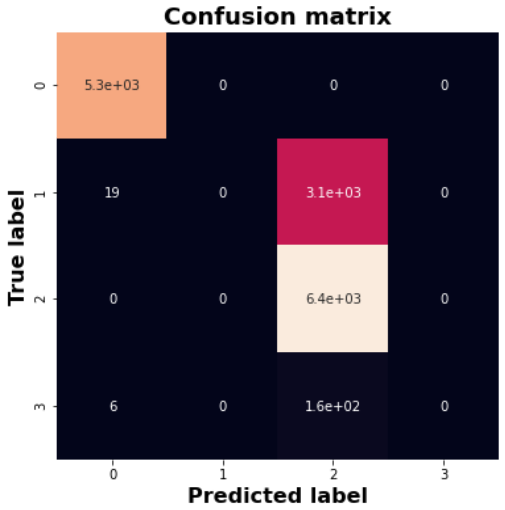

Fig 3: Confusion Matrix Random Forest

Note : This is the Encoding of label i.e column outcome is {'non hospitalized': 0, 'recovered': 1, 'hospitalized': 2, 'deceased': 3}

2.4 Overfitting

Initially, the models were trained on the default hyperparameters and it was noticed that they were not performing well as the train and test scores were very less. When some hyperparameters were introduced, the train and test scores started increasing. The K-fold accuracy was also increasing but for some hyperparameters, the k-fold accuracy was very less than the test and train score that means split bias was occurring and the model was overfitting for some splits. We have not applied parameters in the Gaussian model as training and testing score start decreases wherever we put any hyperparameters.

The table below includes the values of hyperparameters we used to fit the models. Mostly for all the hyperparameters, the train and test score was similar. The best models are highlighted in yellow, these models were chosen because the K-fold accuracy and misclassification rate was low.

max_depth	12				
n_estimators	100	200	300	400	500
Training Accuracy	81.26%	81.22%	81.20%	81.24%	81.20%
Testing Accuracy	81.17%	81.14%	81.13%	81.16%	81.13%

Table 1: Random Forest Scores On different hyperparameters.

max_depth	6	7	8	9	10
Training Accuracy	88.27%	88.68%	89.11%	89.44%	89.67%
Testing Accuracy	87.98%	88.30%	88.66%	88.86%	89.02%

Table 2: XGboost Scores On different hyperparameters.

Note: These values were calculated using the *train score*100*.