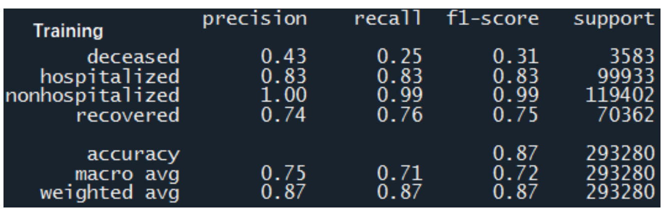
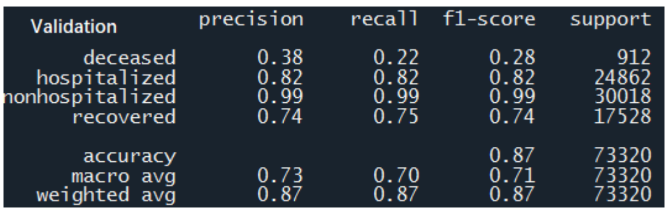
Milestone 1 - Report (amosl - 301315811 / Timothy 301307082 / Bianca B 3013043)
2.1 Splitting dataset
The train dataset has been split by train_test_split function into the given ratio with a seed of 1.

2.2 Build Model
The variant of boosting tree we chose is Catboost. Given that the source, additional information, and a few other derived column were categorical variables, we wanted to take advantage of the internal coding techniques, such as target encoding, as well as the text processing feature of Catboost as opposed to applying one-hot encoding if we chose to use xgboost or AdaBoost instead. Online articles also indicate that not applying one hot encoding relaxes memory consumption as the dimensionality would not increase. Furthermore, Catboost is known to be competitive to any of the gradient boosting algorithms and has a faster training and evaluation time. Another model we chose to construct is KNN because many important variables such as longitude, latitude, age, and date confirmation variables are numerical, maybe distance-based algorithms could capture some patterns within the differences. Additionally, KNN tends to have a low bias with simplified distance function and increase slowly with k, thus we want to check its performance. The computational cost was a heavy disadvantage because our dataset had a reasonable amount of dimensionality after one-hot-encoding all categorical data (Even after dropping a few variables, such as province and country, for which their data is encapsulated in other variables, longitude, and latitude). The last model we chose to use is Random Forest because it leverages the power of multiple decision trees. Unlike decision trees, it chooses features randomly during the training process meaning it will not create a highly biased model while lowering the variance. Other suggested classifiers such as SVM and Naive Bayes are not feasible because it is a multi-class problem and the conditional independence assumption of Naive Bayes is not likely to occur given the dataset.

2.3 Evaluation
The metrics that were generated are accuracy, precision, recall, and f1-score. We also generated a confusion matrix to confirm our observations. Since we know that we have an imbalanced dataset from the support column, we choose to not prioritize accuracy because it will be biased towards the majority class. The metrics that we will be analyzing are the independent and macro average of the remaining metrics. This is because the macro average will treat each label independently and produce an aggregated result. As you can see, all of our classifiers perform well for non hospitalized labels in both the training and validation set with over 99% in precision, recall, and f1-score. This indicates there exists a significant difference in features for the non hospitalized samples than samples from other labels and we will not have to worry as much about the impact of parameter tuning on this label. It also means that the false negative and false positive for this label is low. However, metrics for other labels indicate that there is still room for improvement. It is especially evident for the deceased label as metrics such as F1-score (Harmonic mean between Precision and Recall) can go as low as 26% and 17% on the training and validation set for the KNN model and performance for this label in other classifiers did not improve by much as well. This can be due to several reasons, such as the imbalanced class distribution or perhaps the deceased samples aren’t distinct enough compared to the samples from the hospitalized/recovered label. This suggests that regularization can be introduced to generalize the model more. Another method to remedy the problem is either through stratified sampling or providing class weights such that the classifier will focus on the minority class. One observation from the confusion matrix is that deceased are often labeled as hospitalized, which is understandable as patients are announced dead in the hospital. This means that further EDA has to be done on samples with these labels and perhaps induce new variables on any observed differences. A similar scenario is encountered for the recovered label being mistaken as hospitalized, however, the problem is not as severe as for the deceased label. We want to see whether parameter tuning could solve this problem before attempting any EDA for samples with these two labels as well. Overall, the macro average of the F1-score is the most important metric for the classification problem. This is because our group believes that there is no reason to prioritize precision or recall (reducing false positive or false negative) as it is equally catastrophic when we make either of these mistakes. Since we want to seek a balance between precision and recall, F1-Score has been selected as it provides a weighted average between precision and recall. Based on our most important metric, macro average F1 score, the performance of classifiers are ranked as it follows: Catboost, Random Forest, and KNN, but further parameter tuning is required to confirm these rankings.
Catboost: Evaluation

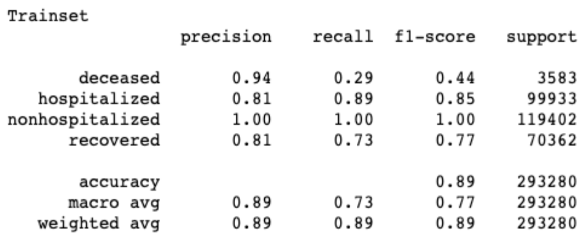


```
Training          precision    recall  f1-score   support
        deceased       0.43      0.25      0.31      3583
    hospitalized       0.83      0.83      0.83     99933
nonhospitalized        1.00      0.99      0.99    119402
       recovered       0.74      0.76      0.75     70362

        accuracy                           0.87    293280
       macro avg       0.75      0.71      0.72    293280
    weighted avg       0.87      0.87      0.87    293280
```

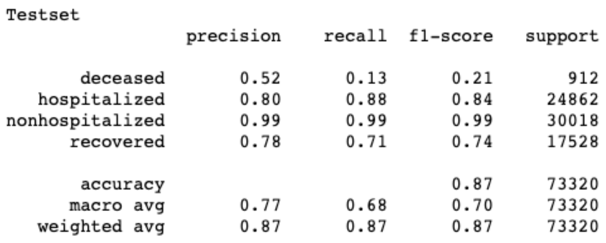


```
Validation        precision    recall  f1-score   support
        deceased       0.38      0.22      0.28       912
    hospitalized       0.82      0.82      0.82     24862
nonhospitalized        0.99      0.99      0.99     30018
       recovered       0.74      0.75      0.74     17528

        accuracy                           0.87     73320
       macro avg       0.73      0.70      0.71     73320
    weighted avg       0.87      0.87      0.87     73320
```

Random Forest Evaluation:

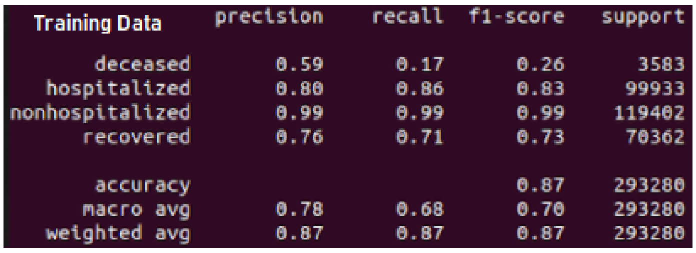


```
Trainset
                 precision    recall  f1-score   support

       deceased       0.94      0.29      0.44      3583
   hospitalized       0.81      0.89      0.85     99933
nonhospitalized       1.00      1.00      1.00    119402
      recovered       0.81      0.73      0.77     70362

       accuracy                           0.89    293280
      macro avg       0.89      0.73      0.77    293280
   weighted avg       0.89      0.89      0.89    293280
```

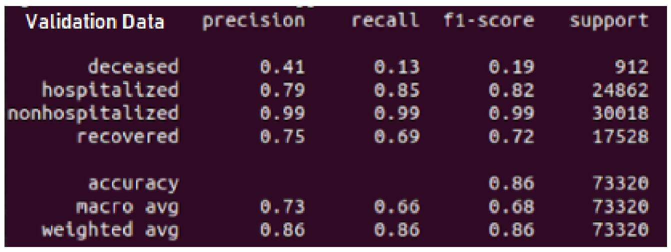


```
Testset
                 precision    recall  f1-score   support

       deceased       0.52      0.13      0.21       912
   hospitalized       0.80      0.88      0.84     24862
nonhospitalized       0.99      0.99      0.99     30018
      recovered       0.78      0.71      0.74     17528

       accuracy                           0.87     73320
      macro avg       0.77      0.68      0.70     73320
   weighted avg       0.87      0.87      0.87     73320
```

KNN Evaluation:

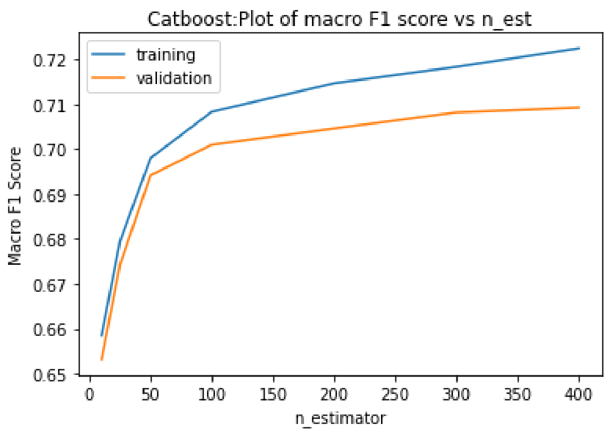


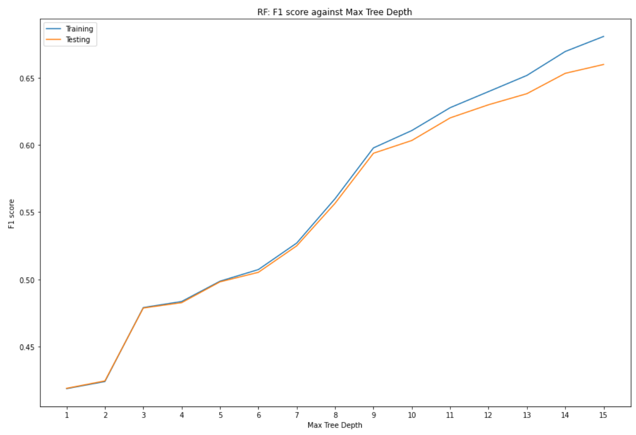
```
Training Data    precision    recall  f1-score   support

       deceased       0.59      0.17      0.26      3583
   hospitalized       0.80      0.86      0.83     99933
nonhospitalized       0.99      0.99      0.99    119402
      recovered       0.76      0.71      0.73     70362

       accuracy                           0.87    293280
      macro avg       0.78      0.68      0.70    293280
   weighted avg       0.87      0.87      0.87    293280
```



```
Validation Data  precision    recall  f1-score   support

       deceased       0.41      0.13      0.19       912
   hospitalized       0.79      0.85      0.82     24862
nonhospitalized       0.99      0.99      0.99     30018
      recovered       0.75      0.69      0.72     17528

       accuracy                           0.86     73320
      macro avg       0.73      0.66      0.68     73320
   weighted avg       0.86      0.86      0.86     73320
```

2.4 Overfitting





k	**Macro Avg - F1 Score: Train**	**Macro Avg - F1 Score: Validation**
3	**70%**	**68%**
5	**70%**	**68%**
10	**70%**	**67%**

As taught in the lecture, if there exist two models T and T’ such that T has better accuracy than T’ on training data but T’ has better accuracy than T on the test data, then it indicates overfitting has occurred. Catboost classifier and Random Forest classifier have been retrained at different values of parameters, specifically, from varying n_estimator in range of 50 - 400 for Catboost and changing maximum tree depth from 1 - 25 for Random Forest. The Macro average of the F1 score has been recorded for both training and validation data and plotted against our parameters. As you can see, the performance on the validation dataset follows the pattern in the training dataset. It increases when the score on training increases and decreases when the score on training decreases. Therefore, we can conclude that there is no overfitting for Random Forest and Catboost. Varying the parameter of neighbors and plotting the graphs are too computationally expensive due to the volume of data we have. However, the textbook mentions that it is expected to see overfitting if the value of k is too low, as it might be sensitive to noise in the training set and use it in calculating the label. Thus, we have built models at k=3,5,10 in attempts to capture any indication of overfitting by observing changes in the macro average of F1-score at training and validation dataset. Like our other classifiers, the score on the validation dataset follows the performance from the training dataset. Additionally, a similar macro average of f1-score showed for the 3 KNN models that were retrained, its value ranges from 68% to 70% on the training and validation data respectively. From the result, we can also say that there is no overfitting for the KNN classifier at the chosen n-neighbor parameter.