COVID-19 County Level Severity Classification with Imbalanced Class: A NearMiss Under-sampling Approach

Timohty Oladunni, Sourou Tossou, Yayehyrad Haile and Adonias Kidane
Computer Science/Information Technology Department
University of the District of Columbia
Washington DC, USA
Timothy.oladunni@udc.edu, sourou.tossou@udc.edu,
yayehyrad.haile@udc.edu, adonias.kidane@udc.edu

***Abstract* — COVID-19 pandemic that broke out in the late 2019 has spread across the globe. The disease has infected millions of people. Thousands of lives have been lost. The momentum of the disease has been slowed by the introduction of vaccine. However, some countries are still recording high number of casualties. The focus of this work is to design, develop and evaluate a machine learning county level COVID-19 severity classifier. The proposed model will predict severity of the disease in a county into low, moderate, or high. Policy makers will find the work useful in the distribution of vaccines. Four learning algorithms (two ensembles and two non-ensembles) were trained and evaluated. Class imbalance was addressed using NearMiss under-sampling of the majority classes. The result of our experiment shows that the ensemble models outperformed the non-ensemble models by a considerable margin.**

Keywords—COVID-19, Classification, predictive model, KNN, Random Forest, Boosting, imbalance class

I. INTRODUCTION

Since the outbreak of the coronavirus pandemic, the Centers for Disease Control and Prevention (CDC) has recorded close to 30 million cases. Thousands of lives have been lost to COVID-19 [1]. While the United States and other developed countries has been able to bend the curve on the fatality rate, emerging evidence suggests that the disease is just taking root in some countries. As of May 19, 2021, Mexico tops the fatality rate with 9.3%. At a distant second is Peru with 3.5 %. Italy and Iran came third and fourth with 3% and 2.8% respectively. The origin of this pandemic is an ongoing research; however, most scientists believe that it originated from a bat in Wuhan, China.

The question now is: how do we categorize the severity of COVID-19 fatality in a county? We answered this question by building a machine learning classifier using the fatality rate dataset from the 3 006 counties in the US. Dataset was obtained from the John Hopkins University repository.

Machine learning algorithms have been shown to have the capability to learn pattern and discover knowledge from a dataset. It has been used in image recognition, fraud detection, voice recognition, malware detection etc. Since the outbreak of the coronavirus pandemic, several studies have been done using machine learning algorithms to understand the pandemic and provide strategies to reduce its spread.

Author [2] proposed a quantitative model to predict vulnerability to COVID-19 using genomes. Neural networks and Random Forests were used as learning algorithms. The result of the study confirmed previous work on phenotypic comorbidity patterns in susceptibility to COVID-19. In another study, Kexin studied nineteen risk factors associated with COVID-19 severity. The result suggested that severity relates to individual's characteristics, disease factors, and biomarkers [3]. Hina et al., proposed a model to predict patient COVID-19 severity in Pakistan. Seven learning algorithms were trained and evaluated. The result of the experiment showed that Random Forest had the best performance with 60% accuracy.

While there are several studies on COVID-19 severity, there seems to be a gap in machine learning literature on the imbalanced classification of COVID-19 severity at the county level. Therefore, the focus of this study is the algorithmic imbalance classification of COVID-19 of a county into low, moderate, or high. *We hypothesized that ensemble learning in conjunction with the under-sampled majority class of an imbalance COVID-19 dataset has a superior capability of predicting the severity of COVID-19 at the county level.*

We tested our hypothesis by experimenting with ensemble and non-ensemble learning algorithms.

Random Forest and Boosting Trees were trained and evaluated as our ensemble model, while Logistic Regression and K Nearest Neighbors as the non-ensemble models.

This paper is organized as follows: Section 2 describes the methodology for the study. Discussions, and conclusions are highlighted in sections 3 and 4 respectively. Finally, we acknowledged the source of our funding in section 5.

II. METHOGOLOGY

The experimental flowchart of the study is shown below. The diagram shows the information flow of the experiment. Four models were trained and evaluated. Performance comparison of the models was also done.

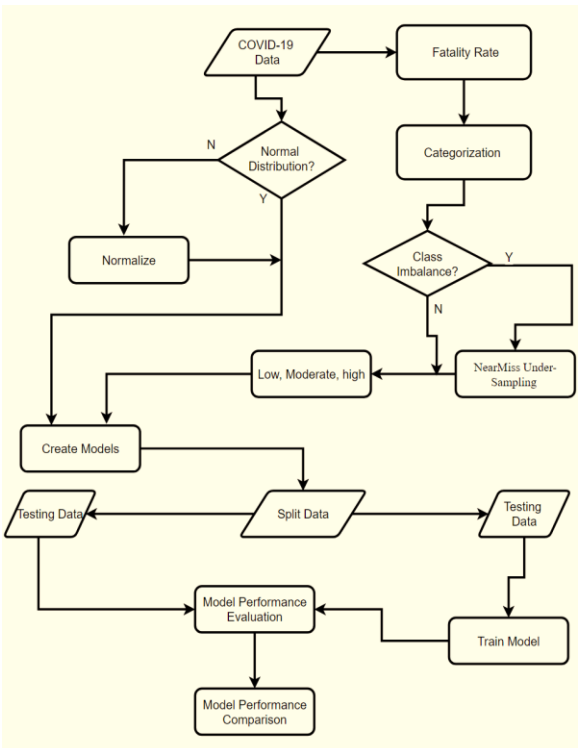

Figure 1. Experimental Flowchart

1. Dataset

Dataset was obtained from the John Hopkins University COVID-19 repository [4]. Dataset was cleansed at the processing stage. Non-numerical variables were converted into numerical variables. Data consisted of the 3 006 counties of the United States. Insignificant and redundant features were dropped during the cleaning phase. Normalization was also done.

2. Categorization

Severity of COVID-19 was measured using the fatality rate as the response variable. The fatality rate as recorded in the dataset was a continuous variable. Therefore, attributes were split into 3 groups based on the following criterion: counties with fatality rates less than 1 were categorized as low ($0 < x \leq 1$). Moderate class are the counties with fatality rate greater than 1 but less than or equal to 2 ($1 < x \leq 2$). Finally, the high class are counties that have greater than 2 but less than equal to 4 fatalities ($2 < x \leq 4$). Categorization or grouping is crucial for classification of continuous variables.

3. Imbalance Class

The above categorization resulted into skewed class distribution. This skewness of the class distribution is referred to as class imbalance. An imbalance dataset has one or more classes with low records (minority class) and one or more classes with many records (majority class). Class imbalance has been shown to have a considerable negative impact on the effectiveness of a learning algorithm.

4. Under-sampling of the majority class- The Near Miss Under-sampling (NMU) Approach

The question is, *how do we balance the dataset?* An imbalanced data can be balanced by oversampling of the minority class or under-sampling of the majority class. In oversampling approach, more data are created to increase the size of the minority class records to equal the majority class records. However, this approach has the risk of overfitting. On the other hand, in under-sampling, the size of the majority class is reduced to balance the class distribution. We believe this is a better approach. Therefore, in this study, we used the Near Miss Under-sampling (NMU) strategy.

NMU selection is based on distance of the majority records to the minority records. It is a k nearest neighbor approach. Distance is based on the Euclidean distance measure. NMU has three versions: version 1, version 2 and version 3. Version 1 is based on the smallest average distance between the majority class and three closest records of the minority class. Version 2 selects records from the majority class with farthest distance from three minority class. Lastly, in version 3, a given number of the majority class is selected for each closest example in the minority class. In this study, version 1 is used. The result of our experiment shows the effectiveness of our strategy. The NearMiss function from the imblearn.under_sampling library was used.

5. Experiment

We trained and evaluated 2 ensemble learning algorithms (Random Forest and Boosting). We also

trained and evaluated 2 non-ensembles (Logistic Regression and K Nearest Neighbors). Dataset was split into 90% and 10% for training and testing, respectively. Performance evaluation was based on precision, recall, accuracy and F1 score.

5.1 Performance Evaluation

To compare the results of our experiment, we used accuracy, the recall, and the f-1 score as our factors of comparison.

5.1.1 Accuracy

Accuracy is defined as the percentage of correct predictions for the test data. It can be calculated easily by dividing the number of correct predictions by the number of total predictions, and the formula is as follow:

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions} \quad (1)$$

5.1.2 Precision

Precision is a metric that quantifies the number of correct positive predictions made. And it is calculated using the following formula:

$$\text{Precision} = \frac{\text{TruePositives}}{(\text{TruePositives} + \text{FalsePositives})} \quad (2)$$

5.1.3 Recall

Recall is a metric that quantifies the number of correct positive predictions made from all positive predictions that could have been made. Its operation is as followed:

$$\text{Recall} = \frac{\text{TruePositives}}{(\text{TruePositives} + \text{FalseNegatives})} \quad (4)$$

5.1.4 F-Measure

F-Measure provides a way to combine both precision and recall into a single measure that captures both properties. Its formula is as given:

$$\text{F Measure} = \frac{(2 * \text{Precision} * \text{Recall})}{(\text{Precision} + \text{Recall})} \quad (5)$$

5.2 Learning Algorithms

We trained and evaluated the performances of 4 learning algorithm.

5.2.1 K-Nearest Neighboring (KNN)

In a dataset with y response variable y and **X** feature vectors, a KNN learning algorithm identifies K points in a training dataset that are closest to a new testing datapoint x_0.

$$\Pr(Y = j | X = x_0) = \frac{1}{K} \sum_{i \in N_0} I(y_i = j) \quad (6)$$

Where j is estimated response and y_i as the target (label). N_0 is the K points. In our experiment 5 was selected as the value of K. In addition, we used the MixedMeasures for the measure types. The Euclidean distance was used as the distance metric. [5]

$$d_{xy} = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \quad (7)$$

Where **d** represents the distance, **x** and **y** are 2 data points.

Performance of the KNN learning algorithm is shown in table 1. In all evaluation criterion, the result suggest that moderate class has the lowest prediction. Accuracy score was approximately 0.61.

Table 1. KNN performance

KNN TABLE

classification_report

	precision	recall	f1-score	support
High	0.61	0.77	0.68	235
low	0.68	0.71	0.69	222
moderate	0.47	0.27	0.35	172
accuracy			0.61	629
macro avg	0.59	0.59	0.57	629
weighted avg	0.60	0.61	0.59	629

```
confusion_matrix
 [[181  23  31]
 [ 42 158  22]
 [ 72  53  47]]
accuracy_score 0.6136724960254372
```

5.2.2 Logistic Regression

Logistic regression is a supervised learning algorithm for predicting the likelihood of a target variable. In a two-class problem, the target or dependent variable is dichotomous, which implies there would be just two potential classes [6]. The logistic function produces output between 0 and 1.

$$p(X) = \frac{e^{\beta_0 + \beta_1 X}}{1 + e^{\beta_0 + \beta_1 X}} \quad (8)$$

It can be shown that,

$$\frac{p(X)}{1 - p(X)} = e^{\beta_0 + \beta_1 X} \quad (9)$$

Taking logarithms,

$$log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X \qquad (10)$$

where b_0 is the bias or intercept term and b_1 is the coefficient for the single input value (x). L2 regularization was used as the overfitting control. Tolerance for stoppage criteria was 1e-4. Optimization was based on lbfgs. Table 2 shows the result of the Logistic Regression.

Table 2. Logistic Regression performance.

```
LOGISTIC REGRESSION TABLE
                        classification_report
              precision    recall  f1-score   support

        High       0.00      0.00      0.00       235
         low       0.35      1.00      0.52       222
    moderate       0.00      0.00      0.00       172

    accuracy                           0.35       629
   macro avg       0.12      0.33      0.17       629
weighted avg       0.12      0.35      0.18       629

confusion_matrix
 [[  0 235   0]
 [  0 222   0]
 [  0 172   0]]
accuracy_score 0.35294117647058826
```

As shown in table 2. The performance of Logistic Regression is worse than that of KNN.

5.2.3 Random Forest

Random forest is a supervised learning algorithm that is utilized for classifications as well as regression. A forest is comprising of trees and more trees suggest a stronger forest. Aggregating decision trees in ensemble learning, produces a better performance. Essentially, the Random forest calculation creates decision trees on bootstrapped training data samples. and afterward gets the forecast from every one of them and then lastly chooses the best solution through voting [7]. It is an ensemble method that is superior to a solitary decision tree since it decreases the over-fitting by averaging the outcome.

$$RFfi_i = \frac{\sum_j normfi_{ij}}{\sum_{j \in all\ features,\ k \in all\ trees} normfi_{jk}} \qquad (11)$$

where $RFfi_i$ is the importance of feature i calculated from all trees, $normfi_{ij}$ is the normalized feature importance for i in tree j.

Table 3. Random Forest performance.

```
RANDOM FOREST REGRESSION TABLE
                        classification_report
              precision    recall  f1-score   support

        High       0.52      0.91      0.66       235
         low       0.78      0.77      0.78       222
    moderate       0.00      0.00      0.00       172

    accuracy                           0.61       629
   macro avg       0.43      0.56      0.48       629
weighted avg       0.47      0.61      0.52       629

confusion_matrix
 [[213  22   0]
 [ 50 172   0]
 [145  27   0]]
accuracy_score 0.6120826709062003
```

Table 3 shows that the Random Forest model outperformed KNN and Logistic Regression models.

5.2.4 Boosting Tree

Boosting is an ensemble modeling technique that endeavors to fabricate a solid classifier from the number of weak classifiers. It is done by building a model by utilizing weak models in series like Random forest. First and foremost, a model is built from the training data. At that point the subsequent model is constructed which attempts to address the errors present in the first model. This method is proceeded, and models are added until either the total training data is predicted accurately, or the most extreme number of models are added [8].

Its implementation required us to 100 for the number of trees, a maximal depth of 5, a min rows of 10, a min split improvement of 1.0E-5, a number of bins equals to 20, a learning rate of 0.01, and a sample rate of 1.

Table 4. Boosting Tree Performance

```
BOOSTING TABLE
                        classification_report
              precision    recall  f1-score   support

        High       0.96      0.97      0.97       235
         low       0.96      0.98      0.97       222
    moderate       0.94      0.90      0.92       172

    accuracy                           0.95       629
   macro avg       0.95      0.95      0.95       629
weighted avg       0.95      0.95      0.95       629

confusion_matrix
 [[229   1   5]
 [  0 217   5]
 [  9   9 154]]
accuracy_score 0.9538950715421304
```

III. DISCUSSION

Accuracy of the models were compared. For each model, we also took the average performance of the precision, recall and F1 score. Table 5 shows the comparison table.

Table 5. Model Performance Comparison

	Logistic Reg.	**KNN**	**Random Forest**	**Boosting**
Accuracy	47.73%	61.72%	69.54%	93.41%
Avg.Precision	52%	61%	69%	93%
Avg. Recall	48%	62%	69%	93%
Avg.F1-Score	47%	62%	69%	93%

As shown in the experimental result, Random Forest and Boosting Tree models outperformed other models. These two models were built with large number of decision trees on bootstrapped training data. The results of shows that the Boosting model has the best performance with 93.41% of accuracy. Performances based on precision, recall and F1 showed an averaged value of 93%, 93%, and 93% respectively. The superior performance of the Boosting Model is not surprising because, a boosting tree is a large combination of decision trees grown sequentially. Random Forest and Boosting Tree are built on the ensemble of decision trees. However, the arrangement of fitting small trees with a few terminal nodes into the residual of the previous tress in a Boosting Tree sequentially improves the performance of the model.

IV. CONCLUSION

In this study we have designed, developed, and evaluated a COVID-19 severity classifier using imbalance class dataset. The proposed model has the capability of predicting the severity level of COVID-19 in a given county. Dataset was obtained from the JHU COVID-19 repository. COVID-19 Severity level was based on fatality rates in all the 3 006 counties of the US. For classification purpose, fatality rate was categorized into low, moderate and high. Imbalance class was addressed using the Near Miss Under-sampling (NMU) approach. Ensemble and non-ensemble learning algorithms were trained and evaluated. Ensemble models include Random Forest and Boosting Trees. KNN and Logistic Regression were used as the non-ensemble models.

The result of our experiment suggests that the ensemble models are the most effective in building a COVID-19 severity classifier at the county level using imbalanced dataset. Thus, we do not have sufficient evidence against our hypothesis. Therefore, we contend that *ensemble learning in conjunction with under-sampled majority class of an imbalance COVID-19 dataset has a superior capability of classifying the severity of COVID-19 at the county level.*

V. ACKNOWLEDGEMENT

This work is funded by the National Science Foundation grant number 2032345.

References

[1] C. f. D. C. a. P., ""CDC"," [Online]. Available: https://www.cdc.gov/.

[2] R. Y. Wang, T. Q. Guo, L. G. Li, J. Y. Jiao and L. Y. Wang, ""Predictions of COVID-19 Infection Severity Based on Co-associations between the SNPs of Co-morbid Diseases and COVID-19 through Machine Learning of Genetic Data," in *IEEE 8th International Conference on Computer Science and Network Technology (ICCSNT)*, 2020.

[3] K. Tang, "Risk factors and indicators for COVID-19 severity: Clinical severe cases and their implications to prevention and treatment," in *International Conference on Public Health and Data Science (ICPHDS)*, 2020.

[4] John Hopkis University COVID-19 Repository , [Online]. Available: https://coronavirus.jhu.edu/.

[5] I. G. M. a. N. G. I. R. Okfalisa, "Comparative analysis of k-nearest neighbor and modified knearest neighbor algorithm for data classification," in *2nd International conferences on Information Technology, Information Systems and Electrical Engineering (ICITISEE)*, Yogyakarta, Indonesia,, 2017.

[6] Y. H. Z. T. a. K. S. X. Zou, "Logistic Regression Model Optimization and Case Analysis," in *IEEE 7th International Conference on Computer Science and Network Technology (ICCSNT)*, Dalian, China, 2019.

[7] J. C. B. B. a. K. M. B. B. R. I. H. Ortiz, "Analysis model of the most important factors in Covid-19 through data mining, descriptive statistics and random forest," in *IEEE International Autumn Meeting on Power, Electronics and Computing (ROPEC)*, Ixtapa, Mexico, 2020.

[8] Y. W. K. a. D. D. J. Dutta, "Comparison of Gradient Boosting and Extreme Boosting Ensemble Methods for Webpage Classification," in *Fifth International Conference on Research in Computational*

Intelligence and Communication Networks (ICRCICN), Bangalore, India, 2020.

[9] Y. C. Y. Z. X. F. a. Y. L. F. Miao, "Predictive Modeling of Hospital Mortality for Patients With Heart Failure by Using an Improved Random Survival Forest," *IEEE Access*, vol. vol. 6, no. IEEE, pp. pp. 7244-7253, 2018.

[10] [. L. A. S. L. F. A. M. L. M. B. C. M. &. M. P. D. M. McCormack, "Gaps in knowledge about COVID-19 among US residents early in the outbreak," in *Public Health Reports*, United States, 2021..

[11] [. U. W. a. S. H. A. C. Braun, "Support vector machines, import vector machines and relevance vector machines for hyperspectral classification," in *in 3rd Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*, Lisbon, Portugal, 2011.

[12] Y. Z. H. Z. a. Q. W. C. Zhan, "Random-Forest-Bagging Broad Learning System with Applications for COVID-19 Pandemic,," *IEEE Internet of Things Journal*, 2021.