Furqaan Ali, Greg Baranov, Shawn John

CS 412

Professor McCarty

5/2/2020

Machine Learning Techniques on Covid-19 Data

The Coronavirus Epidemic is modern day catastrophe that forever changed the world at the start of the year 2020. While this epidemic continues to stress and change societal norms, it is important to analyze such data and see if there is way to accurately predict a country's level of preparedness for an event of this magnitude. The authors of this paper thought it would be interesting and mind-opening to use machine learning on such data, especially because at the time this paper is written the authors are currently facing the challenges that the Coronavirus has introduced.

Every day if not every minute each country updates their own statistics on the coronavirus. Due to this continuous update to the dataset the models we create today will most likely be less accurate than the models of tomorrow but that does not mean that these models will not show some insight about this epidemic. The coronavirus data comes from ourworldindata.org which updates the statistics daily. The dataset is a credible one because it is a joint effort between researchers at the University of Oxford as well as the non-profit organization, Global Change Data Lab. Their data has been used by The New York Times, The Wall Street Journal and Stanford. Since such reputable organization uses Ourwoldindata the credibility of the data is very high. The statistics on GDP and population was provided by Worldometer another credible

website used by respected newspapers such as the Oxford University Press and the Atlantic. It has also provided information to Fortune 500 companies such as Amazon and Google.

The coronavirus dataset consists of the number of cases and the number of deaths per country per day. Originally the data had multiple rows for each country to account for past daily measurements. However, the number of rows per country was variant. The data for each country was aggregated so that each data point will be the most up to date information about an individual country.

The dimensionality of the coronavirus dataset is lacking compared to most machine learning dataset, such as digits data which can have over 200 dimensions. In order to combat the problem of not having enough attributes the coronavirus data was combined with other datasets that describe other characteristics of a country. The features of a country that was thought was relevant to the Covid-19 epidemic was GDP of a country, population of a country, as well as population density for that country. By adding these attributes to the original datasets, the coronavirus dataset's dimensionality increased giving the machine learning models more features to train on.

The coronavirus data was quite comprehensive by having statistics on some of the smallest countries. After manually merging these datasets the obvious issue of missing data arose. The most immediate solution was to take the average of the column the missing data is from and putting that in for the missing data. Putting zero into the missing data entry would not standardize the data well and could potentially skew the data. However, the averages for population and GDP are also skewed because there are countries with huge GDP, for example the US with a GDP of 19 trillion. So, if a small country like the Western Sahara has missing GDP data assigning it an average GDP in the billions would not suffice. Another option was

median, because it less susceptible to skewing, but it fails to be solution to the problem just mentioned. In order to keep the data as accurate as possible the best solution was to search the internet for these missing values. Naturally, this disrupted the singularity of merging the data, because not all of the data came from the same source. However, the benefits of getting the most accurate data for each country and not having missing data, outweighed the risks of not having data come from a single source.

After the datasets were combined extra columns were added in order to label each country because the machine learning techniques used are for classification type problems. The most important thing these models could predict is the death rate for each country. The reason death rate was chosen, rather than case rate, was because the importance of accurately predicting death rate is much more paramount than case rate. For example, a country with a case rate of 1000 per million and a death rate of 2 per million is much better off than a country with a case rate of 1000 per million and a death rate of 500 per million. Basically, the probability of death is a significantly more important value than the probability of infection. The reason for such values cannot be determined alone by the coronavirus dataset, it could be based on a country's GDP or population and or population density, and that was one of the reasons those datasets were combined. Each row was given two labels: above or below the average and above or below the median. These labels allow the classifier models to run and try to determine whether a country will be above or below the world's threshold. The reason both median and average is used is because average is usually skewed by outliers, but those outliers could potentially not be outliers, so it is important to use just in case. Median is used to ensure that balance of datapoints is even, in that half of the data will be on and the other half will be the other.

Accurately testing models is a vital component of machine learning and there is still ongoing research into determining the best way to test the machine learning models. The standard amount of data set aside for training is 20 percent and 80 percent on testing. However, due to the relatively small size of the dataset, there are only around 200 countries, the ratio had to be higher just so that the models can have more models to train on. The trade off of course is the testing accuracy, but the intention of these models is to give some insight into the coronavirus data and not to be used to determine policy in any manner. The way the data was split into such a ratio was the first 40 percent was set aside for training and the rest for testing. Critics for this method might suggest that the models will be skewed because the data is separated in an ordered fashion. However, the data is sorted alphabetically and the countries close together alphabetically is no way correlated to their attributes. If it were sorted geographically or by GDP or population then of course there would be a problem but alphabetically does not matter. For example, Spain will not be related to Senegal just because they are alphabetically similar.

The first machine Learning Technique used was a simple and naïve K-nearest neighbors approach. The testing method to find the most optimal K-nearest neighbors was 10-fold cross validation.

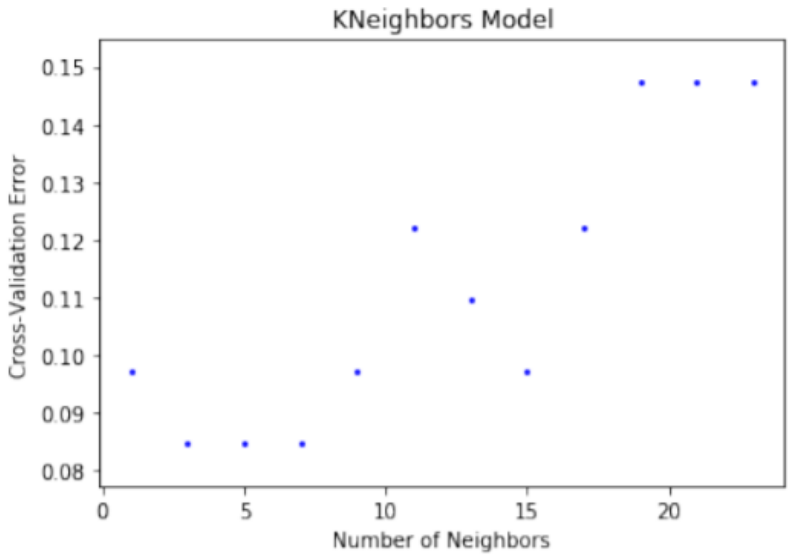


Figure 1

After creating models where the number of neighbors varies from one to twenty-five the above figure was created for models predicting if a country's death per million will be above or below the global average. The figure clearly shows an increase in error after K equals 7. The figure shows that the optimal model for predicting if a country will be above or below the average death per million is 3. The reason there is a slight parabolic pattern to the cross-validation error is the complexity that this data requires for K-nearest neighbors. One neighbor is not enough because a point is going to be closest to itself if points do not overlap, which is not guaranteed here but highly unlikely. This most likely will cause overfitting because the model is highly dependent on each data point for its training. Increasing the neighbors past seven increases error as well because the model will suffer from underfitting. If the number of neighbors is equal to the number of data points then each prediction will just be whatever the majority is which is underfitting.

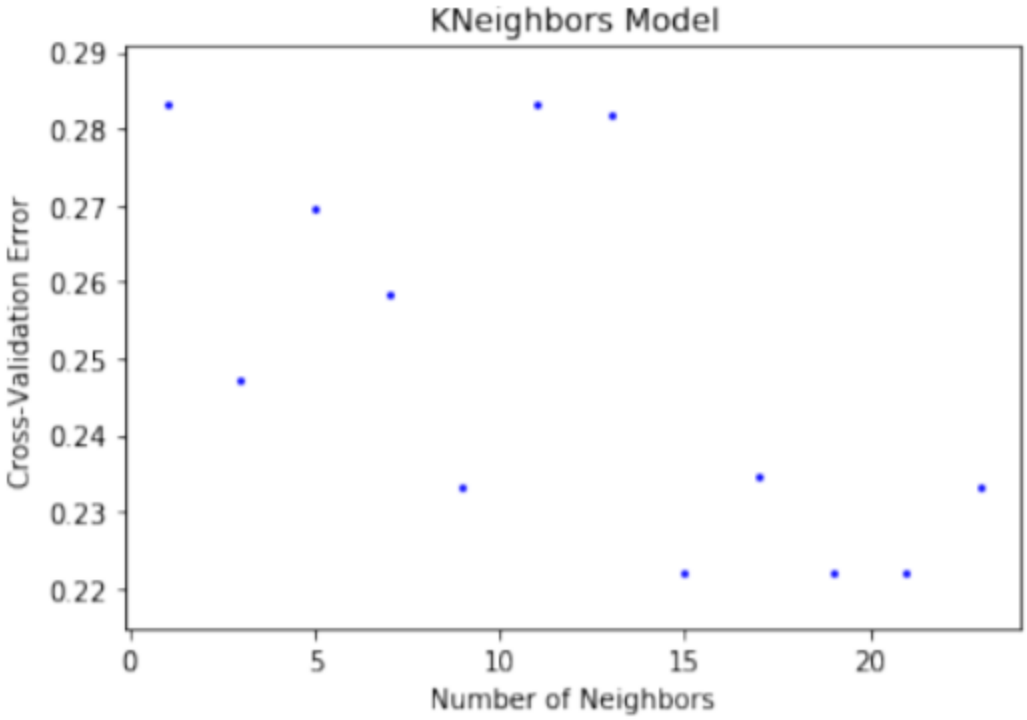


Figure 2

Figure 2 is for the same machine learning technique of K-nearest neighbors however the labels have changed from if country is below or above an average to now below or above a median. The change in label has also resulted in a change of the optimal model. According to figure 2 the optimal K value is 15. Unlike Figure 1, Figure 2 does not have as smooth parabolic shape. Clearly for these labels the models need more neighbors than before as the "local minimum" for this graph is shifted quite considerably compared to Figure 1. The reason for this is most likely due to the variance of the label. It is most likely that since average is more skewed it could require less neighbors. However, the median labels are equally balanced in that each label will have the same amount of data points. Since the median data points are more balanced

this could result in points in the space be varying and so 7 was not enough to determine its label and more was needed.

Another very important thing to note is the y axis bounds in Figure 2 to compared Figure 1. Figure 1's error value was upper bounded to 0.15 while Figure 2 lower bound is 0.22 which is higher than Figure 1's upper bound. The fact that labels that are so similar have a relatively significant difference in error is quite fascinating. The reason for the change in error is most likely again due to the nature of each label's frequency. Since average can be skewed so much it will drastically change the average label's frequency. However median always splits the data into a fifty percent ratio. This ratio will cause points in the data space be much harder to differentiate because they are more likely to be close together. The closer and varying data points in the space will cause the error to go up.

The next machine learning technique was to do some feature extraction on the dataset. While the dimensionality of this dataset is already low, there could be a possibility that the data heavily relies one just one or two, though most likely the models need all the attributes given except the name of the country. The feature extraction technique used was the Kernel principal component analysis.

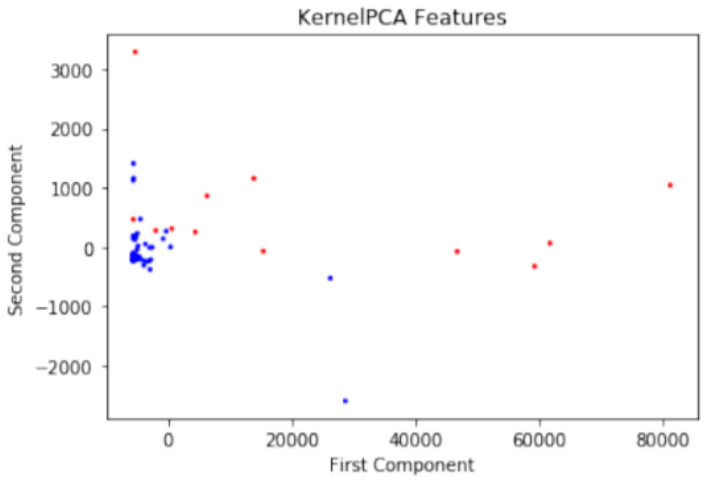


Figure 3

Figure 3 is the Kernel PCA on the label if a country is below or above the global average death rate per million. If this data was dependent on a subset of the features, then these data points would be more separated, and the space would be more defined. As expected, it seems that most of the columns of the data is used to calculate the data point location in the space.

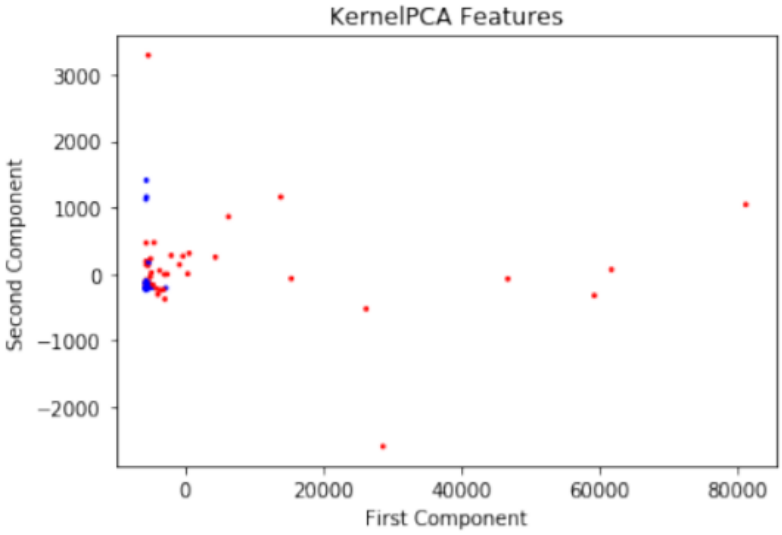


Figure 4

While Figure 4 is the same graph but on the median labels it is important to note the similarities and differences of figure 3 and figure 4. The second and first component have the same range, so the features extracted are most likely the same, which means that the median label also most likely uses all the features of the data, which was expected. However, the separability of points in figure 4 compared to figure 3 is much less. There is much more overlap in the space of figure 4 compared to figure 3. Since median labels have more balance it results in equal amount of data points for each side and this causes the points to have much more overlap than before. The overlap proves that reducing the dimensionality of the data will not create a better model that separates the data, and it also proves that the machine learning models will need all the features to better separate the space because reducing dimensions causes overlap.

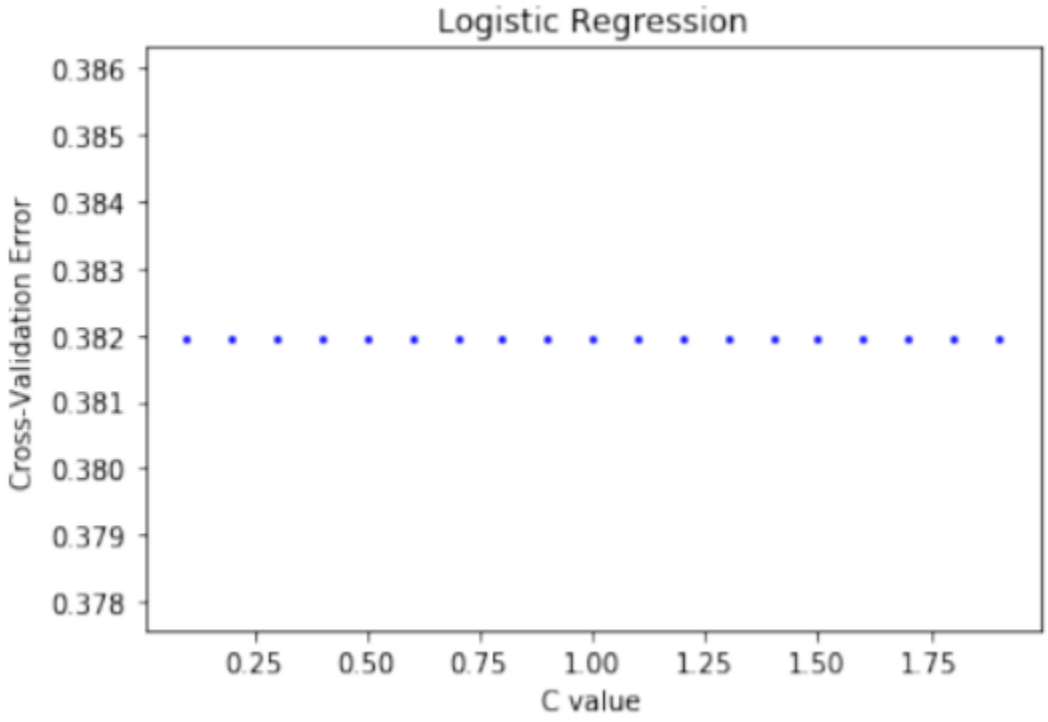


Figure 5

Figure 5 is another piece of evidence that using a KernelPCA will not benefit the models. Figure 5 is the cross-validation error on a logistic regression model on the median label data. The cross-validation error is stable at 0.382 which means that the C value in this range will not matter. However, an error of .382 is very high. The worst error one can possible get is 0.5, because that is just choosing randomly, so being only 12 percent off means that logistic regression using a KPCA is not very good at predicting whether a country will be above or below the average median.

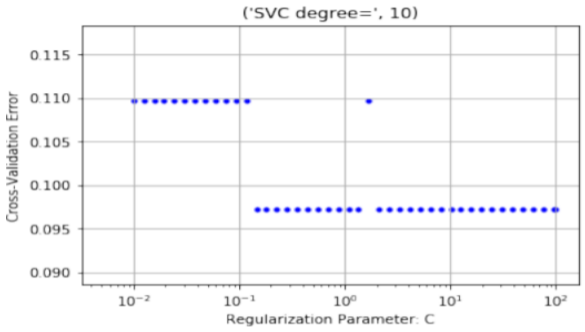


Figure 6

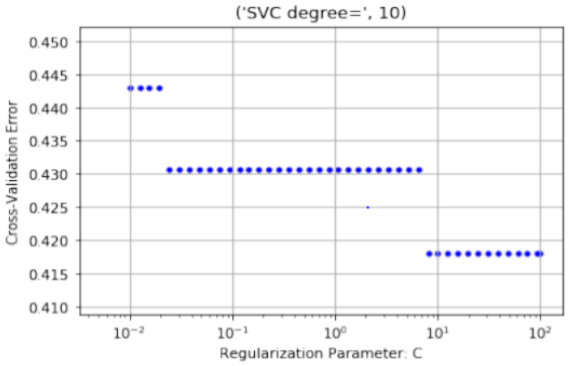
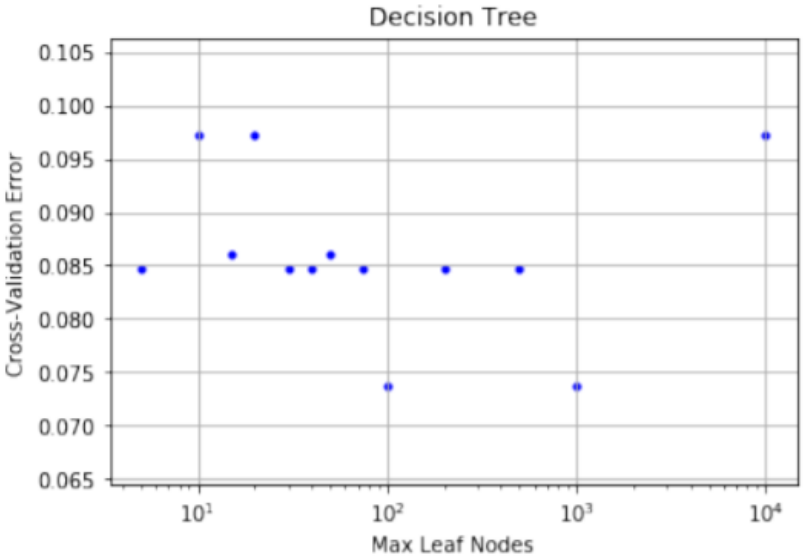


Figure 7

Figures 6 and 7 are for the average and median labels used on support vector machines, respectively. While figure 5 already shows that using models that require dimensionality

reduction is not a good fit for this data, figures 6 and 7 confirm it. These two models have higher cross-fold validation error than the models seen above. Even the K-nearest neighbor models had better cross fold validation errors than the support vector machine models. The skew in average can be seen again in figures 5 and 6 as median labels have higher error than the models predicting average.

Another machine learning technique used to try to predict whether a country will be above or below average and median was decision trees.

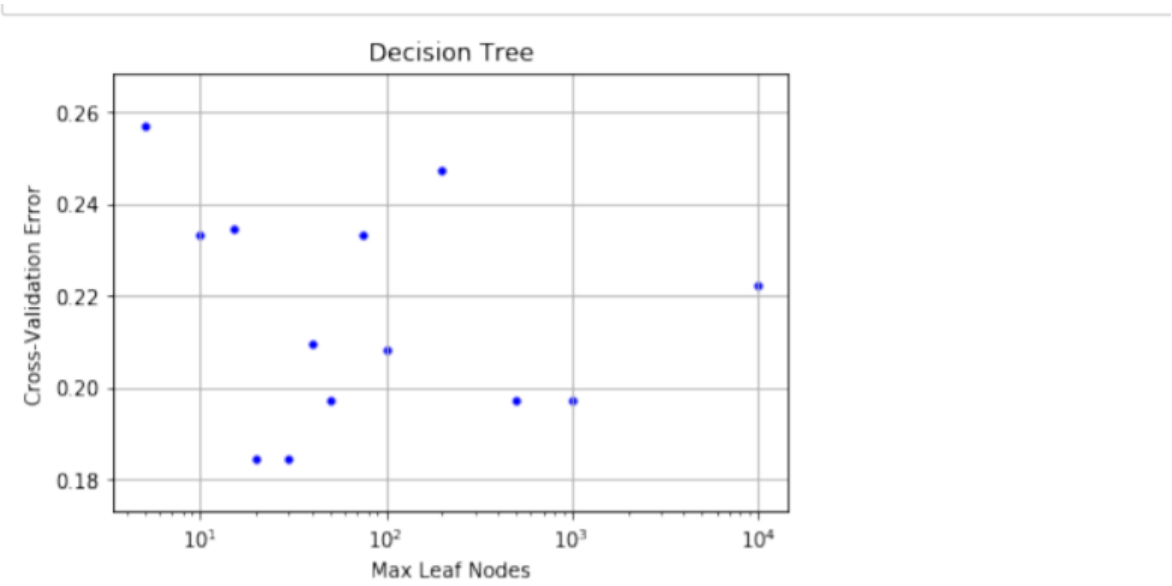



```
Lowest Cross-Validation Error = 0.07361111111111111 at max_leaf_node
s = 100
```

Figure 8

Figure 8 is another graph of the cross-validation error on the decision tree average data. Of the models seen so far this model had the best error. The best configured decision tree was at max leaf nodes equal to 100. This goes back again to the idea of finding the middle ground of underfit and overfit. Nodes less than 100 most likely result in underfitting because the tree isn't

complex enough for the data while nodes greater than 100 result in the model being too reliant on what it is trained on.




```
Lowest Cross-Validation Error = 0.18472222222222223 at max_leaf_node
s = 20
```

Figure 9

Figure 9 is the same as figure 8 but using the labels for median. As usual the cross-validation error is higher for this model but the chosen leaf node here is 20 while in figure 9 it was 100. Since for figure 9's decision tree it chose a lower leaf node this decision tree is less complex. However, the median labeled decision tree only allows so much accuracy compared to the average. All the models seen so far have this disparity between accuracy and median which is explained below. Another difference between figure 9 and figure 8 is the variance in error in figure 9 compare to figure 8. This again is due to the frequency of labels in average compared to the frequency in median. Since median is more balanced the decision tree is deciding on points much closer together in space compared to average.

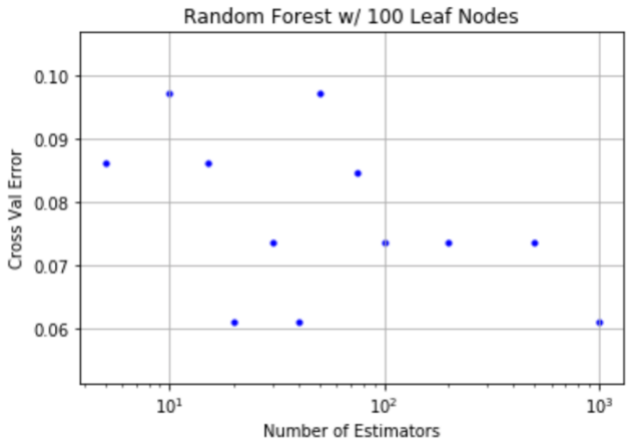


Figure A

Figure A demonstrates the cross-validation errors for the optimal found number of leaf nodes for a random forest classifier on the dataset using the average labels. The optimal model for this classifier seems to be attained using around 20 – 40 estimators which produces a quite low error of slightly over 0.06. This is likely the number of estimators that most nicely fits to the data as we can see that if we decrease the number of estimators, error increases likely due to the model not being complex enough (underfitting), but if we increase the number of estimators, the error still rises—likely due to overfitting to the data.

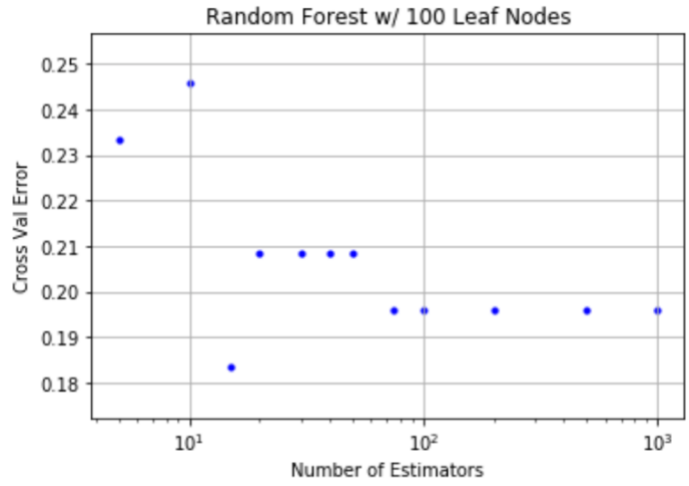


Figure B

Figure B demonstrates the cross-validation errors for the optimal found number of leaf nodes for a random forest classifier on the dataset using the median labels. As like all the previous models, the error is higher when using the median labels. We can also observe that as we increase the number of estimators from the optimal number found in figure A, the error still continues to decrease. This is expected, since unlike the data using the average labels, this data has more overlap and so it is more difficult to fit, thus fitting it more tightly using more estimators should increase its accuracy.

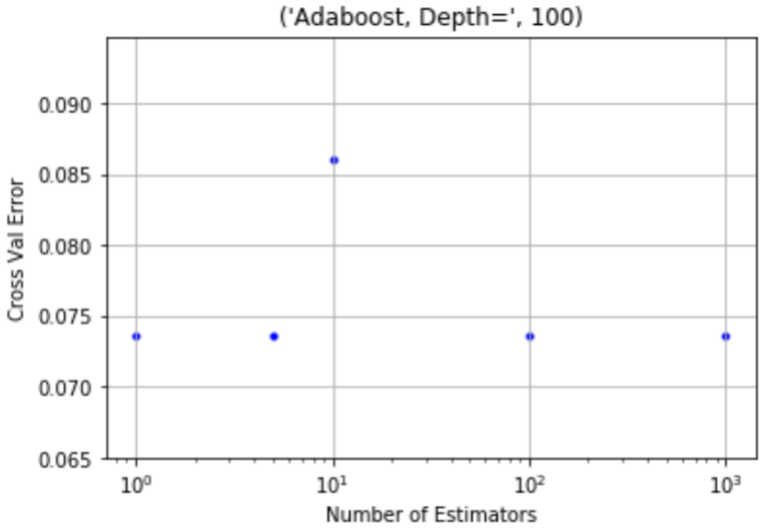


Figure C

Figure C demonstrates the cross-validation errors that result from performing AdaBoost using decision trees as estimators with a depth of 100 (optimal number found). As can be seen by comparing this figure with figure 8, the cross-validation error is reduced when using AdaBoost, though very slightly. It is evident that the accuracy is not changing much with a change in the number of estimators, indicating that the model is already as fit as it can be to this data, likely since the depth size is already quite large at the value of 100, so many estimators are not needed. However, when using the smaller max depth value, we were able to observe that the cross-

validation error decreases as more estimators are used, as the model becomes more fitted to the data.

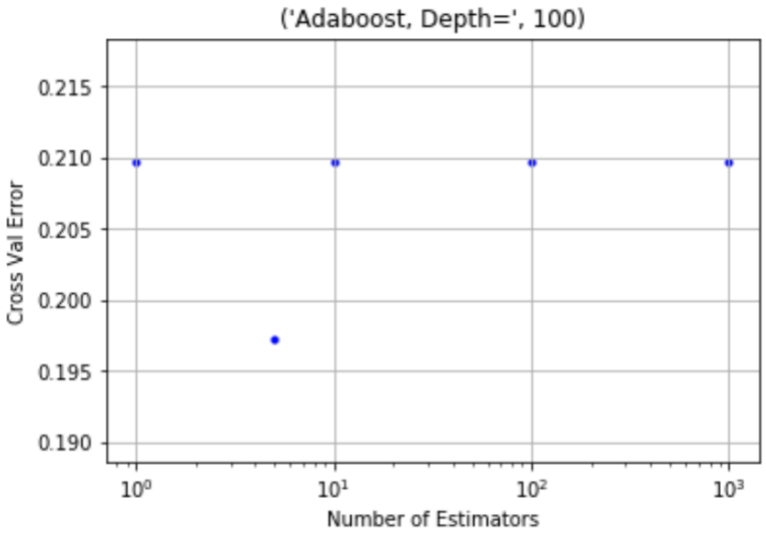


Figure D

Figure D demonstrates cross-validation errors of running the same model from figure C but on the dataset using median labels. It follows the same trends of the accuracy not changing much with a change in estimators. However, similar to figure C, running this model on a smaller depth size does decrease error as estimators increase, indicating that the depth of 100 is causing the model to already be as fitted as it can. Just like all the previous models, this model has a lower accuracy when being run on the median-labeled data opposed to the average-labeled data.

```
Markov Bound at 75.0 %
        K-Neighbors Bound: 0.540983606557377
        Logistic Regression Bound: 0.6065573770491803
        Support Vector Machine Bound: 0.540983606557377
        Decision Tree Bound: 0.6065573770491803
        Random Forest Bound: 0.639344262295082
        AdaBoost Bound: 0.6065573770491803

Markov Bound at 95.0 %
        K-Neighbors Bound: -1.2950819672131129
        Logistic Regression Bound: -0.9672131147540965
        Support Vector Machine Bound: -1.2950819672131129
        Decision Tree Bound: -0.9672131147540965
        Random Forest Bound: -0.8032786885245884
        AdaBoost Bound: -0.9672131147540965

Markov Bound at 99.0 %
        K-Neighbors Bound: -10.475409836065566
```



```
        Logistic Regression Bound: -8.836065573770483
        Support Vector Machine Bound: -10.475409836065566
        Decision Tree Bound: -8.836065573770483
        Random Forest Bound: -8.016393442622942
        AdaBoost Bound: -8.836065573770483


Chebyshev Bound at 75.0 %
        K-Neighbors Bound: 0.11475409836065575 +- 0.090535746042518
53
        Logistic Regression Bound: 0.09836065573770492 +- 0.0905357
4604251853
        Support Vector Machine Bound: 0.11475409836065575 +- 0.0905
3574604251853
        Decision Tree Bound: 0.09836065573770492 +- 0.0905357460425
1853
        Random Forest Bound: 0.0901639344262295 +- 0.09053574604251
853
        AdaBoost Bound: 0.09836065573770492 +- 0.09053574604251853

Chebyshev Bound at 95.0 %
        K-Neighbors Bound: 0.11475409836065575 +- 0.202444082544728
93
        Logistic Regression Bound: 0.09836065573770492 +- 0.2024440
8254472893
        Support Vector Machine Bound: 0.11475409836065575 +- 0.2024
4408254472893
        Decision Tree Bound: 0.09836065573770492 +- 0.2024440825447
2893
        Random Forest Bound: 0.0901639344262295 +- 0.20244408254472
893
        AdaBoost Bound: 0.09836065573770492 +- 0.20244408254472893

Chebyshev Bound at 99.0 %
        K-Neighbors Bound: 0.11475409836065575 +- 0.452678730212592
5
        Logistic Regression Bound: 0.09836065573770492 +- 0.4526787
302125925
        Support Vector Machine Bound: 0.11475409836065575 +- 0.4526
787302125925
        Decision Tree Bound: 0.09836065573770492 +- 0.4526787302125
925
        Random Forest Bound: 0.0901639344262295 +- 0.45267873021259
25
        AdaBoost Bound: 0.09836065573770492 +- 0.4526787302125925


Hoeffding Bound at 75.0 %
        K-Neighbors Bound: 0.11475409836065575 +- 0.075375952842301
09
```



```
        Logistic Regression Bound: 0.09836065573770492 +- 0.0753759
5284230109
        Support Vector Machine Bound: 0.11475409836065575 +- 0.0753
7595284230109
        Decision Tree Bound: 0.09836065573770492 +- 0.0753759528423
0109
        Random Forest Bound: 0.0901639344262295 +- 0.07537595284230
109
        AdaBoost Bound: 0.09836065573770492 +- 0.07537595284230109

Hoeffding Bound at 95.0 %
        K-Neighbors Bound: 0.11475409836065575 +- 0.110804292719448
99
        Logistic Regression Bound: 0.09836065573770492 +- 0.1108042
9271944899
        Support Vector Machine Bound: 0.11475409836065575 +- 0.1108
042927194 4899
        Decision Tree Bound: 0.09836065573770492 +- 0.1108042927194
4899
        Random Forest Bound: 0.0901639344262295 +- 0.11080429271944
899
        AdaBoost Bound: 0.09836065573770492 +- 0.11080429271944899

Hoeffding Bound at 99.0 %
        K-Neighbors Bound: 0.11475409836065575 +- 0.137381397224041
52
        Logistic Regression Bound: 0.09836065573770492 +- 0.1373813
9722404152
        Support Vector Machine Bound: 0.11475409836065575 +- 0.1373
8139722404152
        Decision Tree Bound: 0.09836065573770492 +- 0.1373813972240
4152
        Random Forest Bound: 0.0901639344262295 +- 0.13738139722404
152
        AdaBoost Bound: 0.09836065573770492 +- 0.13738139722404152
```

Figure E: Confidence bounds for dataset using the average labels

```
Markov Bound at 95.0 %
        K-Neighbors Bound: -4.737704918032781
        Logistic Regression Bound: -6.704918032786878
        Support Vector Machine Bound: -5.557377049180323
        Decision Tree Bound: -6.868852459016386
        Random Forest Bound: -5.065573770491798
        AdaBoost Bound: -6.704918032786878

Markov Bound at 99.0 %
        K-Neighbors Bound: -27.688524590163908
```



```
        Logistic Regression Bound: -37.52459016393439
        Support Vector Machine Bound: -31.786885245901615
        Decision Tree Bound: -38.34426229508193
        Random Forest Bound: -29.32786885245899
        AdaBoost Bound: -37.52459016393439


Chebyshev Bound at 75.0 %
        K-Neighbors Bound: 0.28688524590163933 +- 0.090535746042518
53
        Logistic Regression Bound: 0.38524590163934425 +- 0.0905357
4604251853
        Support Vector Machine Bound: 0.3278688524590164 +- 0.09053
574604251853
        Decision Tree Bound: 0.39344262295081966 +- 0.0905357460425
1853
        Random Forest Bound: 0.30327868852459017 +- 0.0905357460425
1853
        AdaBoost Bound: 0.38524590163934425 +- 0.09053574604251853

Chebyshev Bound at 95.0 %
        K-Neighbors Bound: 0.28688524590163933 +- 0.202444082544728
93
        Logistic Regression Bound: 0.38524590163934425 +- 0.2024440
8254472893
        Support Vector Machine Bound: 0.3278688524590164 +- 0.20244
408254472893
        Decision Tree Bound: 0.39344262295081966 +- 0.2024440825447
2893
        Random Forest Bound: 0.30327868852459017 +- 0.2024440825447
2893
        AdaBoost Bound: 0.38524590163934425 +- 0.20244408254472893

Chebyshev Bound at 99.0 %
        K-Neighbors Bound: 0.28688524590163933 +- 0.452678730212592
5
        Logistic Regression Bound: 0.38524590163934425 +- 0.4526787
302125925
        Support Vector Machine Bound: 0.3278688524590164 +- 0.45267
87302125925
        Decision Tree Bound: 0.39344262295081966 +- 0.4526787302125
925
        Random Forest Bound: 0.30327868852459017 +- 0.4526787302125
925
        AdaBoost Bound: 0.38524590163934425 +- 0.4526787302125925


Hoeffding Bound at 75.0 %
        K-Neighbors Bound: 0.28688524590163933 +- 0.075375952842301
09
```



```
        Logistic Regression Bound: 0.38524590163934425 +- 0.0753759
5284230109
        Support Vector Machine Bound: 0.3278688524590164 +- 0.07537
595284230109
        Decision Tree Bound: 0.39344262295081966 +- 0.0753759528423
0109
        Random Forest Bound: 0.30327868852459017 +- 0.0753759528423
0109
        AdaBoost Bound: 0.38524590163934425 +- 0.07537595284230109

Hoeffding Bound at 95.0 %
        K-Neighbors Bound: 0.28688524590163933 +- 0.110804292719448
99
        Logistic Regression Bound: 0.38524590163934425 +- 0.1108042
9271944899
        Support Vector Machine Bound: 0.3278688524590164 +- 0.11080
429271944899
        Decision Tree Bound: 0.39344262295081966 +- 0.1108042927194
4899
        Random Forest Bound: 0.30327868852459017 +- 0.1108042927194
4899
        AdaBoost Bound: 0.38524590163934425 +- 0.11080429271944899

Hoeffding Bound at 99.0 %
        K-Neighbors Bound: 0.28688524590163933 +- 0.137381397224041
52
        Logistic Regression Bound: 0.38524590163934425 +- 0.1373813
9722404152
        Support Vector Machine Bound: 0.3278688524590164 +- 0.13738
139722404152
        Decision Tree Bound: 0.39344262295081966 +- 0.1373813972240
4152
        Random Forest Bound: 0.30327868852459017 +- 0.1373813972240
4152
        AdaBoost Bound: 0.38524590163934425 +- 0.13738139722404152
```

Figure F: Confidence bounds for dataset using the median labels

From figure E, we can observe that all the optimal models perform fairly well on the average-labeled data, as all of their accuracies are greater than or equal to 0.8 with 75% confidence. All of the models had a score of about 0.9 on the testing data, but the random forest classifier model had a slight performance edge over the remaining models.

From figure F, we can observe that the best of the optimal models performed with an accuracy of 0.6 or greater with 95% confidence. The random forest classifier had a score of 0.7 on the testing data, performing quite better than most of the other models.

Not all machine learning techniques we used were for generating predictions. Using KMeans clustering, we were able to identify how easy it was to group datapoints into clusters and visualize where those clusters were centered. In Figure 10 we establish that the dataset could be grouped relatively accurately into clusters using just 5 cluster centers (identified by the elbow point in the inertia graph). Using the optimal Kmeans cluster value of 5, Figures 11 and 12 show where the clusters would be located in 2-d space using two components from the dataset: GDP by billion and total cases per million. Figure 11 colors points based on if a country's death rate is above or below the average of all countries, and Figure 12 colors based on the median of all countries.

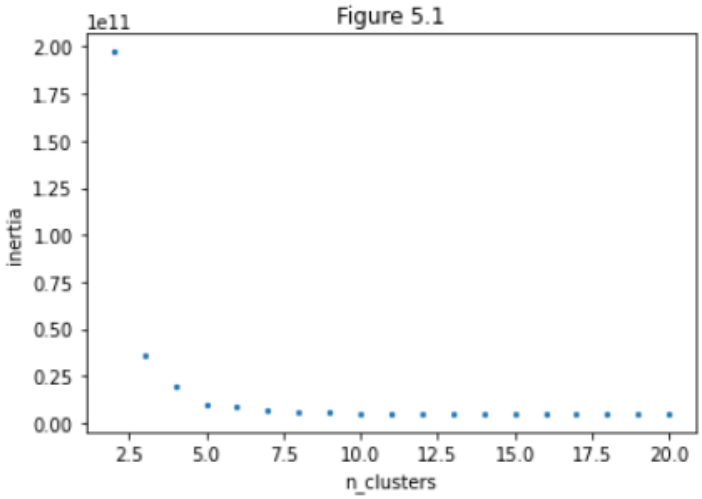


Figure 5.1
1e11
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
inertia
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
n_clusters

Figure 10

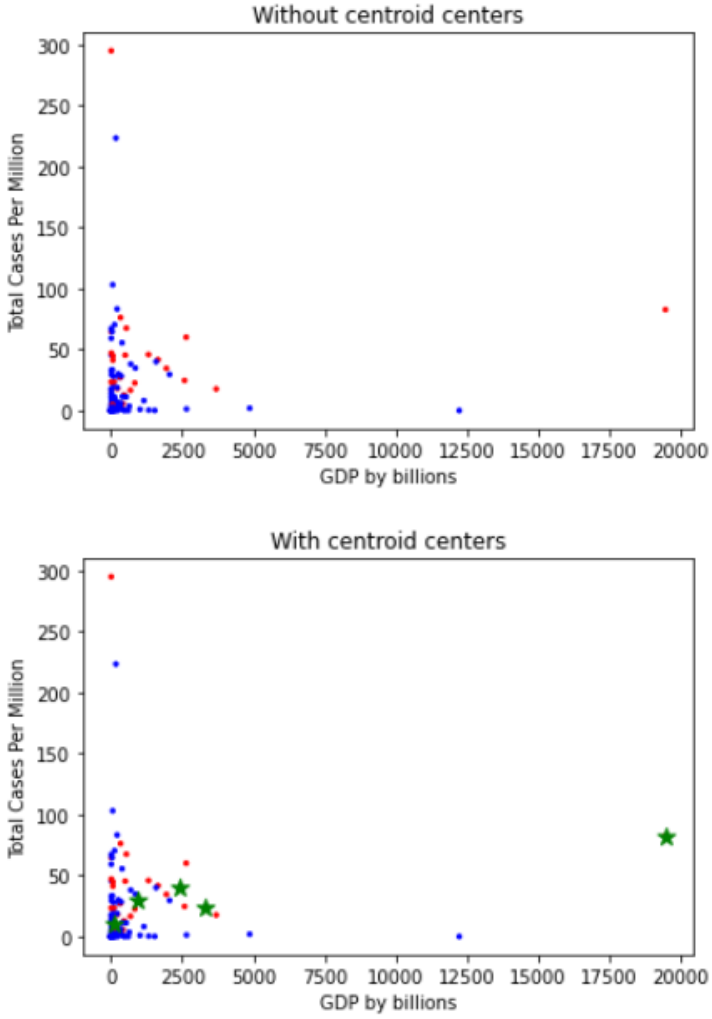

Without centroid centers
Total Cases Per Million
GDP by billions
With centroid centers
Total Cases Per Million
GDP by billions

Figure 11

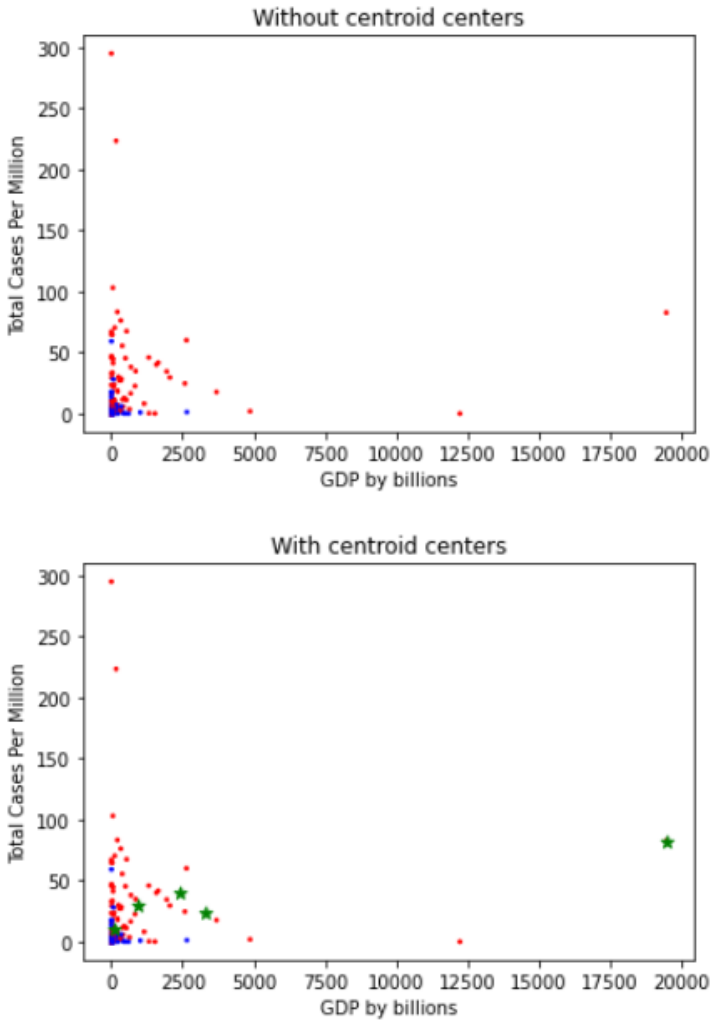


Figure 12

Although these two components picked may not show the full extent of why the clusters converged where they did, it is clear that many datapoints exist near the lower X and Y values.

Surprisingly, at least for the chosen components as the x and y axis, it appears that clustering is more effective with the median classification than with the mean classification. For the mean graph, the blue and red points are much more interspersed in the clusters, while the median graph more clearly has the below median values near the origin of the graph, with red points (above median) further away. Higher GDP and Total Cases Per Million may not be the only reasons for a higher death rate; however, higher x and y values are correlated with more red points. Potentially this shows that as GDP and cases per million increase, there is a higher chance that more people in a country will die, with the task of our models being to predict just how much the death percentage is increased by such factors.

In general, it does not seem like this is a dataset that lends itself very well to clustering. There do not appear to multiple, distinct clusters, just one main cluster centered at the origin, fading out as the x and y axis increase. To get the best understanding of the dataset, we turned to both regular Neural Networks and a Convoluted Neural Network to try to create the most advanced model we could for the dataset. Using the TensorFlow tutorials at https://www.tensorflow.org/tutorials/, we were able to create two different neural networks and assess their accuracy using both the mean and median classification methods. To fully test these networks, we used 40% of the dataset for testing and 60% for training and reported the testing accuracy for the models in Figures 13 – 16. In our simple Neural Network, we used 5 layers of 100 nodes, using the ReLu function to pass information forward. Our Convoluted Neural Network used 2 Convolution Layers with 16 nodes and a convolutional window size of 3, followed by 3 regular neural network layers with 100 nodes. Our reasoning behind using the CNN was that there might have been features in the dataset that would only become apparent

when combining multiple components together, causing the CNN to perform better than a typical Neural Network.

```
Epoch 200/200
4/4 [==============================] - 0s 5ms/step - loss: 0.9588 - accuracy: 0.9504
3/3 [==============================] - 0s 1ms/step - loss: 3.3282 - accuracy: 0.8537
Regular NN, mean data
test loss:  3.328240394592285
test accuracy:  0.8536585569381714
```

Figure 13

```
Epoch 200/200
4/4 [==============================] - 0s 9ms/step - loss: 2.2946 - accuracy: 0.8512
3/3 [==============================] - 0s 5ms/step - loss: 1.6930 - accuracy: 0.8902
CNN, mean data
test loss:  1.6929821968078613
test accuracy:  0.8902438879013062
```

Figure 14

```
Epoch 200/200
4/4 [==============================] - 0s 6ms/step - loss: 0.1650 - accuracy: 0.9174
3/3 [==============================] - 0s 1ms/step - loss: 0.4219 - accuracy: 0.8049
Regular NN, median data
test loss:  0.42189455032348633
test accuracy:  0.8048780560493469
```

Figure 15

```
Epoch 200/200
4/4 [==============================] - 0s 6ms/step - loss: 1.3286 - accuracy: 0.7769
3/3 [==============================] - 0s 5ms/step - loss: 1.1872 - accuracy: 0.7683
Regular NN, median data
test loss:  1.187229037284851
test accuracy:  0.7682926654815674
```

Figure 16

As can be seen in the graphs, in general, the CNN performed better than the regular NN for the mean data and the regular NN outperformed the CNN in the median dataset. There can be many explanations for why this turned out to be the case, but it is likely true that the mean dataset better lent itself to a Neural Network finding similarities within a convolutional window, and making predictions based partly on how certain components of the dataset related to each other. Additionally, because the median dataset was generally harder to classify, both models scored lower on the test accuracy, showing it is more difficult for essentially all of our models to predict if a country will have above median deaths than above average deaths. These Neural Networks are also likely not perfect, and by adjusting the parameters even more, and possibly through bagging, a more accurate Convoluted or regular Neural Network could be generated.

The reason that there is such a higher error for median compared to average labels is because of average's susceptibility to outliers. The coronavirus data is very prone to outliers because certain countries will have way higher deaths compared to large percent of the world's other countries. For example, China and the U.S will completely dominate one side of the scale just by sheer volume therefore skewing the average. Since the models are predicting whether a country will be above or below the median and average the disparity in these frequencies is a key reason that there is higher error in median compared to average. The median ration for above and

below is guaranteed to be around 50/50 because that is how median works. However, average has no constraint allowing the ratio to be whatever it wants. Since average allows a majority, and by nature of the coronavirus data a huge majority, this will cause the models to train on the majority and the features allowing the models to be more accurate in predicting average because the ratio becomes an added feature. The median ratio is not a feature so the models themselves are constrained to the given features and must try to accurately predict above or below on only the features given not on the frequency of the data.

Machine learning is a great technological advancement and using it on the coronavirus data can potentially lead to some way to combat the epidemic. However, there are always ethical issues that need to be addressed whenever using machine learning.

This biggest ethical issue faced in this paper is most likely on the data. The models created can only predict above and below the average of confirmed deaths by coronavirus and should not be used to estimate actual deaths because the models only trained on labels of confirmed deaths. Confirmed deaths and actual deaths are two different things. This is because of the lack of testing in countries. Even the U.S has problems testing people in a timely manner so expecting other 3rd world countries to test their citizens in way that allows scientists to estimate actual numbers is not worthwhile. Of course, one can use the confirmed deaths to try and estimate the actual number of deaths, but the data so far does not allow to make that estimation. Even though the dataset comes from a reputable source this data contains data bias. One reason is a social and political problems of lack of transparency in a country's policy. For example, North Korea and Russia are countries that lack transparency and so with that the data received from countries with low transparency should be taken with a grain of salt.

As said in the above pages these models should not be used for any type of decision making. One might think to use this model to gauge a country's preparedness for an epidemic, which is fine just to see, but should never be used for any type of policy creating or lawmaking. If such a model were used, even an accurate one used by the most reputable scientists, for decision making there could be unmitigated catastrophes. For example, say the model was ran for a given country and said the rates were going to be below the global average that country might reduce funding for pandemic prevention which, as the U.S has seen, has grave consequences. This will also allow those in power who do not care about the country's preparedness to use this as an excuse to reduce funding and, as seen in the past, will blame the "computer" rather than take responsibility.

While this data has no issues of privacy it is important to reiterate, in this day and age, to assure readers that these models will not affect them in anyway. The data itself deals with large order features such as GDP and population. The death rates, case rates are aggregation of hospital data which cannot be accessed using the dataset. None of the datapoints will ever be able to distinguish a single person or group in a country. The dataset contains only the statistics of a country as a whole.

Through the data collected in this paper, and by analyzing the accuracies of the models tested, we conclude that the decision tree is the best way to model our dataset. The three most accurate models are shown below:

Name of model	Mean test error	Median test error
Decision Tree	0.0736	0.1847

Convoluted Neural Network	0.111	0.232
Regular Neural Network	0.137	0.196

Therefore, based solely on the accuracy of the models, the Decision Tree model is the preferred choice for this dataset, while also having the advantage of being the most intuitive model to understand and explain to parties concerned. Using the decision tree allows us to convey the concept of how the model was created by demonstrating a decision tree on a piece of paper, showing how the lines are drawn to section off parts of the graph with many points of one label. Especially for a model that could have very significant effects on the real world, we want to pick a model that minimizes the amount of decision making that is abstracted away into something as abstract as a Neural Network. It is much easier to validate a decision tree than a Neural Network, especially a CNN, where it is impossible to know how the network arrived at the conclusion it came to. Admittedly, the accuracy of the Decision Tree model is still not incredibly high, but that is likely because this dataset is far from ideally separated, especially because it is pulled from the real world. It is impossible to predict the amount of deaths in a country purely from some data about a country, with noise and variation always being a factor. Regardless, because the Decision Tree model is the most accurate and comprehensible, it is the optimal model.

With respect to whether a country should be classified by its relation to the mean or median amount of deaths from COVID-19 is much more up for debate. While predicting the mean statistic is much easier, and results in a lower error percentage, it can be argued that this is a less useful metric for countries that don't have very large populations and GDPs. Predicting the median statistic, though it might be a less accurate prediction, can give a country more insight

into how serious COVID-19 is within their borders. If a country that has a relatively small economy and population has above the median amount of deaths, that country can immediately recognize that something is going wrong and needs immediate attention. False positives have a much lower significance than False negatives: it is better to think your country is being hit by COVID-19 harder than it actually is, than to be unprepared for the terrible effects of the pandemic. Due to this, we conclude that countries should primarily be concerned with the median statistic and should not take being below the median as a sure sign that nothing is wrong. As stated before, being overprepared is significantly better than being underprepared. We hope this model can help countries realistically estimate how their fatality rate compares to that of the world's average or median, and to adjust their preparations and planning accordingly.

Bibliography

"Convolutional Neural Network (CNN): TensorFlow Core." *TensorFlow*, www.tensorflow.org/tutorials/images/cnn. Accessed May 5th, 2020.

"Coronavirus Pandemic (COVID-19) – the data." *Our World In Data*, https://ourworldindata.org/coronavirus-data. Accessed May 2nd, 2020.

"Countries in the world by population (2020)" *worldometer*, https://www.worldometers.info/world-population/population-by-country/. Accessed May 2nd, 2020.

"GDP by Country" *worldometer*, https://www.worldometers.info/gdp/gdp-by-country/. Accessed May 2nd, 2020.

All packages downloaded: sklearn, numpy, matplotlib, tensorflow, tensorflow.keras.