MACHINE LEARNING FOR THE SPREAD OF COVID-19

Drew Tada
School of Arts and Sciences
University of Virginia
dkt9nv

Shivaen Ramshetty
School of Engineering
University of Virginia
sr3hd

August 12, 2020

1 Abstract

COVID-19 is a coronavirus identified in late 2019, which emerged in China and rapidly spread across the world in a matter of months. The impact of this virus has been severe, as significant parts of the North America, Europe, and East Asia are under lock-down. Because of this, the spread of the virus is of great interest to governments, businesses, and citizens alike. Our project aims to use Machine Learning to predict the spread of this virus across the world. In doing so, we hope to give power to those in charge to act early and prevent greater damage to our economies, societies, and lives.

2 Introduction

Many have previously tried to model the spread of COVID-19 using mathematical models. Using publicly available data, a team at MIT used a neural network to model the spread of the virus [2]. They used a neural net to estimate the parameters of a standard epidemiological model, and then extrapolated the results. In addition, they added a regularization for "with quarantine" and "without quarantine," so that the impacts of social distancing could be visualized when extrapolating. In addition, Neil Ferguson, whose model is now informing the UK's response, created a more sophisticated model, with many parameters [4]. In addition to past data, his model uses detailed population statistics to estimate susceptibility to the disease. It is incredibly complex, written in 1000s of lines of code, which has drawn criticism for being overly complex.

What we wanted to know was: what patterns will a machine pick up on? The previous models were constructed logically from how we believe viruses spread, and then used those parameters to predict further spread. However more complex models don't always lead to more accurate predictions. We wanted to investigate the accuracy of a simple machine learning model to pick up basic patterns, fitting a reasonably accurate function to the spread of the virus, based solely on past data. We believe a simple model could be of great use for countries taking proactive measures before the virus' detrimental impacts become irreversible. The setting of such a problem is global; there hasn't been a day in the last month where the virus hasn't dominated the news cycle. It has changed everyone's life, and without proactive measures, could get a lot worse.

3 Method

The spread of COVID was modeled using supervised learning methods, with numerical labels. We were originally unsure of which regression model would best fit the data, so we tried many. The data we used was the total number of confirmed cases every day [1]. This data was very fine grained, and for some countries like the US, got down to the county and city level. Further, with the help of Virginia L. Berry, we modified the existing data of confirmed COVID-19 cases to be more easily trainable. The label was the total number of cases on day

$$x_0$$

while the features used to predict the label is the sequence of total cases from the previous ten days,

$$[x_{-10}, x_{-9}, ..., x_{-1}]$$

By splitting the data like this, we were able to extract much more training data from our data set. For every location, and for almost every day recorded, we had a feature vector and label.

In addition, we split the data into the standard train, validation, and test sets, which allowed us to evaluate our models later and ensure that overfitting didn't take place. Our splits were 65 percent train, 15 percent validation, and 20 percent test. Furthermore, the data was shuffled in order to avoid the model just memorizing simple patterns based on location and/or the values of the previous data point.

Next, we tried to find the best model using a greedy approach. We ran a plethora of models (without hyper-parameter tuning), and then hypertuned the highest performing ones. Some models we chose include: Linear Regression, Elastic Net, SVR, BayesianRidge Regression, Decision Tree, Random Forest, and even a Recurrent Neural Network.

Initially, the performance of most of the models was mediocre and sometimes awful, even with some models we thought would perform very well, such as SVR. It was then recommended to us that transform our data to fit a log scale. This single transformation resulted in huge improvements across the board, as can be seen in Table 1. With these new error rates, we hyper-tuned two chosen models.

4 Experiments

Each model's baseline RMSE is recorded in Table 1 and we used Linear Regression as our baseline.

Labels Expected: Sample [0.0, 0.0, 0.0, 0.0, 1.301, 0.0, 0.477, 0.0, 0.0, 2.053]

SVR: The initial trial of our SVR was a failure, but it improved greatly upon being fed the log scaled data.

SVR gave us the predictions - [0.100, 0.100, 0.100, 0.100, 0.201 ,0.101, 0.261, 0.100, 0.100, 1.197]

These results gave us an RMSE of .367 and cross validation scores of: [0.478, 0.452, 0.528, 0.349, 0.501, 0.392, 0.384, 0.440, 0.503, 0.448]

Linear Regression: Simple linear regression model gave us the predictions-
[0.010, 0.011, 0.015, 0.005, 1.402, 0.020, 0.523, 0.010, 0.010, 2.05]

These results gave us an RMSE of .080 and cross validations scores of:
[0.061, 0.111, 0.059, 0.063, 0.106, 0.061, 0.075, 0.099, 0.068, 0.078]

Elastic Net: Elastic Net model gave us the predictions-
[0.044, 0.056, 0.10, -0.0291, 0.959, 0.169, 0.516, 0.038, 0.044, 1.910]

These results gave us an RMSE of 0.130 and cross validations scores of:
[0.148, 0.145, 0.123, 0.104, 0.167, 0.109, 0.126, 0.138, 0.140, 0.131]

The score was worse than that of Linear Regression, so have ruled this model out of our next step.

Bayesian Ridge Regression: Bayesian Ridge regression model gave us the predictions-
[0.010, 0.011, 0.015, 0.005, 1.402, 0.020, 0.523, 0.010, 0.010, 2.05]

These results gave us an RMSE of .080 and cross validations scores of:
[0.148, 0.145, 0.123, 0.104, 0.167, 0.109, 0.126, 0.138, 0.140, 0.131]

The differing cross validation scores for this regression was certainly interesting, considering the other data suggested it was almost identical to Linear Regression. However, these results don't give us a good reason to move forward with this model.

Decision Tree: A Decision Tree model gave us the predictions-
[0., 0.019, 0., 0.018, 1.568, 0.011, 0.778, 0.025, 0.017, 2.04]

These results gave us an RMSE of 0.098 and cross validations scores of:
[0.093, 0.128, 0.108, 0.070, 0.114, 0.077, 0.092, 0.124, 0.110, 0.104]

We have to ensure that our hyper parameters like maximum depth allow us to avoid the problem of overfitting, but this can be cumbersome and would rather spend time with other models with less risk of overfitting.

Random Forest: Random Forest model gave us the predictions-
[0., 0.021, 0., 0.022, 1.463, 0.011, 0.604, 0.025, 0.015, 2.06]

These results gave us an RMSE of 0.084 and cross validations scores of:
[0.074, 0.116, 0.072, 0.063, 0.111, 0.073, 0.082, 0.106, 0.080, 0.091]

Recurrent Neural Net: RNN gave us a RMSE of 0.090 and when evaluated on the test set gave us a score of 0.008. With loss as mean squared error, our train loss converged near .0093, while validation loss at around .0065.

With these results, we decided to hyper-tune the Random Forest and Recurrent Neural Net models. The decision tree also performed very well, but we suspected that this was due to overfitting. In addition, we thought it would be more interesting to test the neural net against the random forest because they are very different models underneath, while decision trees and random forests are similar.

Table 1.

Model	RMSE (unscaled data)	RMSE (on log data)
Linear Regression	115	.080
Elastic Net	186	.130
Bayesian Ridge	115	.080
Decision Tree	2.46	.098
Random Forest	42.4	.084
Recurrent NN	n/a	.090
SVR	788	.367

Table 2.

Model	Mean Cross-Validation (on log data)
Linear Regression	.078
Elastic Net	.133
Bayesian Ridge	.133
Decision Tree	.102
Random Forest	.087
SVR	.448

We used a random search to hypertune our forest and found that the optimal number of estimators was 436 with max number of features at 7. With these parameters we were able to get RMSE of .093 on our test set. The preliminary RMSE we had was for our validation set, so this increased RMSE gives us no concern. Though our number of estimators is high, we found that a large number of estimators does not cause overfitting[2]. At the end, these results were satisfactory and gave us enough confidence to move onto the RNN.

On our Recurrent Neural Network, we knew we could do three things: extend the number of epochs, tune the optimizer, and experiment with the number of LSTM neurons. After a plethora of tests, we saw our best results with epochs set to 10, our optimizer as SGD(lr=.15, momentum=.3, nesterov=True, decay=.001), and the number of LSTM neurons as 10. 10 epochs worked just fine as we noticed how fast the model was converging, so more epochs made little difference to our model's performance on the test set. With these optimizations, we were able to get our RMSE to .090. Although, tuning did not seem to vastly improve our RNN's RMSE, its losses were minutely better, with training loss at .0090 and validation remaining at around .0065.

With what we knew, we tried to minimize over fitting and allowed our model to fit the natural phenomenon as tightly as possible. However, even still there was a slight gap between our testing and validation accuracy.

5 Results

Having finally tuned the models, they were run against the test sets. The results for the Random Forest trial gave a RMSE of .092, while the RNN outputted a RMSE of .090, as seen in Table 3. These results were stated above in the experiments section, but must be included here to detail the following patterns within the data.

Table 3.

Model	Test-set RMSE
Random Forrest	.092
Recurrent NN	.090

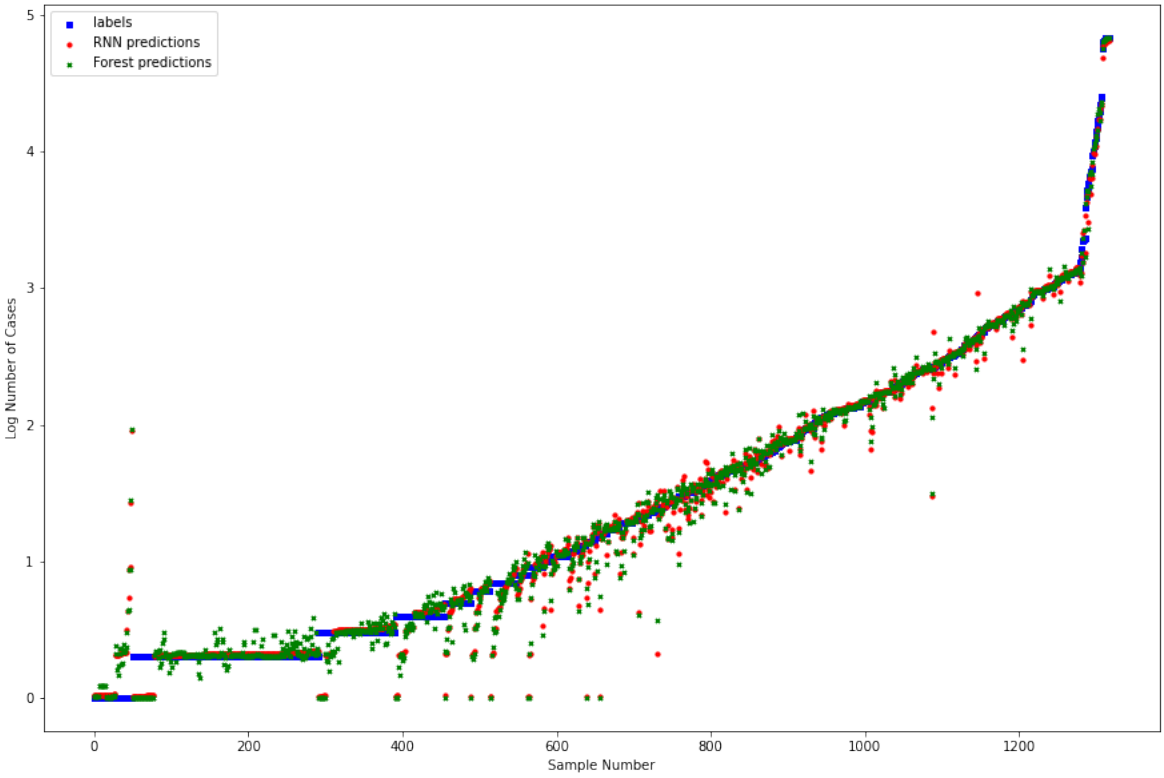
The RMSE's only provide a surface level understanding of what the model does, so we created a graph of the predictions, Figure 1. To create this graph, first each feature vector was grouped with its corresponding label and predictions from both the random forest and neural net. Then, they were sorted into a list of tuples so that similar labels would be grouped together. This was then all graphed together. Each blue line represents a series of similar valued labels, while the red and green points on the same vertical axis represent the different predictions made by the two models, red for the RNN and green for the Forest. The figure gives a good picture of where the model over/under estimates total confirmed cases.

At certain intervals of the graph, large errors are exacerbated by this sorting approach. This is because, the sorting function sorts by the index and when values are equal it looks down the line of elements to compare. This leads to outlier data being pushed to the front or end of a segment, since these data points are the ones whose predictions vary furthest from the expected/label value.

Given the data and how it shapes, various patterns are immediately clear. Firstly, data prediction grows increasingly accurate the larger the number of cases there are. As seen in cases beyond 800, the errors between prediction and label begin to drop drastically compared to cases before. Suggesting that the model better predicts data when a certain trend is clear. Next, errors tend to be either greater than or less than a certain segment of labels. This arises in part due to learned model being one that takes off earlier but grows slower. This is represented, again, by the greater errors in the earlier samples. Finally, there is no correlation between the model chosen and how it predicts the cases, in fact, many of predictions for either model line up. This hints at the fact that both models learned the same function or underlying pattern.

Lastly, the results of the test data can be further explored in respect to the numbers. Beginning with the Forest model, the Randomized Search converged at 7 max features and 436 estimators. Therefore, the model saw that fewer features were necessary than provided and that a conclusion could be drawn from just those seven. The higher number of estimators provides insight into how the trees are prioritizing the features or the amount of depth the trees are going for these features. The lower the number of estimators, the higher the likelihood to miss features, so random search saw great importance in these seven features[2]. On the other hand, the Recurrent neural network provides little additional information into what is under the "hood." This is due to neural nets being black box architectures; however, the loss trends reported in the experiments section gives some information. Focusing on the validation data reveals that the model avoided overfitting, a huge concern, especially when attempting to model such a problem. With validation losses hovering around .0065, there is strong attestation to the model converging and steering away from memorizing the data. Since both models were noted to be similar, there is evidence that the forest model also avoided overfitting, on top of its RMSE.

Figure 1.



6 Conclusion

Our model performed very well when the following day followed a the same pattern as the previous days. However, when the features were very low, close to zero, our model tended to over estimate the growth of the virus. This is because the model is unsure of when the numbers will start to "take-off," and so sometimes will predict a rapid increase, when in fact it is under control. At the very high end of numbers, our model tended to be very accurate. We believe this is because at high numbers, growth becomes more regular. However, when you are at very low or medium number of cases, it can over estimate when there are no takeoffs, and underestimate when there are big takeoffs.

Predicting extreme events is notoriously difficult - and some argue even impossible. Understandably, our model occasionally had very large errors. Because we are dealing with a viral phenomenon, on any given day, it is possible that the number of cases could jump by an order of magnitude. Our model seems to fit according to what would happen "normally," which is reasonable. This definitely limits the use of our model, as lawmakers *should* act as if the number of cases will jump by a large amount. Acting in a precautionary way will stimulate action, but our model may bias someone to think that the future is brighter than it really is, and cause people to under react. However, from a raw accuracy standpoint, our model was able to meet the standard we had in mind. Building the model's rigidity and robustness could come from larger data sets and more complex training; which definitely illuminates the scope/importance of all facets such a project. From the data collection to analysis to action, each holds great significance in our state/country/world and a large section of that process relies on the models we create.

Our models give reasonable predictions for future corona virus cases, but it is important to note that projecting further into future means more uncertainty. If we use our model's predictions to make new feature vectors, extrapolating many days into the future, the error would grow significantly. This underscores that the growth of the virus is dependent upon how citizens and lawmakers react. Our model was only concerned with tomorrow, but the real concerns should be set on the weeks, months, and even years in advance to make sure this virus is contained, and its overall impacts mitigated.

7 References

[1] Adam, David. "The simulations driving the world's response to COVID-19", April 2, 2020
https://www.nature.com/articles/d41586-020-01003-6

[2] Breiman, Leo and Cutler, Adele. "Random Forests"
https://www.stat.berkeley.edu/ breiman/RandomForests/cc_home.htmremarks

[3] Brownlee, Jason. "Time Series Prediction with LSTM Recurrent Neural Networks in Python with Keras", August 5, 2019
https://machinelearningmastery.com/time-series-prediction-lstm-recurrent-neural-networks-python-keras/

[4] Dandekar, Raj et. al. "Quantifying the effect of quarantine control in Covid-19 infectious spread using machine learning", April 6, 2020
https://www.medrxiv.org/content/10.1101/2020.04.03.20052084v1.full.pdf+html

[5] "Novel Coronavirus (COVID-19) Cases Data"
https://data.humdata.org/dataset/novel-coronavirus-2019-ncov-cases/resource/4cd2eaa1-fd3e-4371-a234-a8ef2b44cc1f

8 Contributions

Drew: I drafted and refined the paper, conducted the SVR trial, and handled the data preprocessing issues. The data was originally preprocessed using Virginia V. Berry's code, but more was required. This code can be seen in section "Data Preparation (2)" of the collab notebook.

Shivaen: I began by runninents using various models (Linear Regression, Elastic Net, Bayesian Ridge Regression, Decision Tree, Random Forest, and Recurrent Neural Network). Then I tuned the Random Forest with Random Search and the Recurrent Neural Net with its own hyperparameters. After the tuning, I wanted to visualize our predictions relative to the labels and the other model, so I made multiple graphs to do so. Finally, I helped write parts of the report that intertwined with our data and its results.

Layne V. Berry: helped write some of the code for transforming the data into something trainable.