Covid-19 Cases Prediction Midterm Report

Terry Ye (004757414)
Yichen Lyu (004940413)
Lihang Liu (604972806)

1 INTRODUCTION

The Covid-19 coronavirus is striking our society significantly and causing the death of millions of people. This project focuses on predicting future cases and deaths caused by the virus for the fifty states in the United States based on previous data of daily confirmed cases and deaths. Hopefully, it can help us defeat the coronavirus together.

The training dataset we are using is the JHU CSSE COVID-19 Dataset for daily reported cases in the United States combined with the daily mobility data among different states. More specifically, the dataset from JHU records previous confirmed cases from April 12, 2020 to April 31, 2020 and includes other features such as recovery rate and hospitalization rate. For this initial stage of our project, we will solely focus on the newly confirmed and death cases. Since interstate travel might also impact the cases of the related states, the mobility dataset would also be very important for our model in later optimization.

Therefore, the formal definition of the problem is:

Learning Algorithm: Given the set of states S of size 50 and a set of dates T of size 26, for $s_i \in S(i \in [1, 50])$ and $t_j \in T(j \in [1, 26])$, we have $c(s_i, t_j)$ and $d(s_i, t_j)$ denoting the confirmed cases and deaths for state s_i at day t_j.

Assume a function h_c^* where

$$h_c^*(s_i, t_j) = c(s_i, t_j)$$

and h_d^* where

$$h_d^*(s_i, t_j) = d(s_i, t_j),$$

let the previous cases and deaths form the set $C(S, T)$ and $D(S, T)$.

We want to learn the h_c^* and h_d^* function such that given the training dataset:

$$T = ((S, T), C(S, T) \& D(S, T)),$$

the hypothesis class:

$$H = h : (S, T) \to C(S, T) \& D(S, T),$$

and the loss function L defined by Mean Absolute Percentage Error:

$$\text{APE} = \frac{|Predicted - Truth|}{|Truth|},$$

we have

$$h^* = argmin_{h \in H} L_T(h).[4]$$

The prediction task is defined as:

Prediction task: for the fifty states $s_i^{test} \in S$, and a set of dates $t_j test$ for the date from September 1st to September 26th, given $26 * 50 * 2$ pairs of unseen data T_{test}, predict $c(s_i^{test}, t_j^{test})$ and $d(s_i^{test}, t_j^{test})$ using our h^* functions.

2 DATA PROCESSING

For the data preprocessing section, we look into the given training dataset, extract features related to the task, and transform it into desirable patterns to fit our training model.

At this initial stage of model development, we focus only on the dataset of confirmed cases and deaths provided by JHU CSSE. More specifically, we look at the daily cases and deaths number for each state, ignoring other features such as tested or recovered cases. Therefore, our data would be a labeled training set of size 26 * 50 * 2 for 26 days, 50 states and 2 values for cases and deaths. Since our prediction will be from September 1st to 26th, the unlabeled test set is also of size 26 * 50 * 2.

To further extract the data's features, we break down the training data set into 50 separate csv files, each containing the relevant data for a single state. For this first stage of the model's development, we assumed a linear pattern for the confirmed cases and deaths of the states. Therefore, we also calculated the differential values for the confirmed and death cases in our data set, which will be the main feature to train our model later as we are using AR model which requires stationary data.

In addition, since kaggle only allows three submission per day, our team also needed to generate the labeled testing set on our own to test our model as many times as we want for parameter tuning. To do this, we downloaded 26 files from JHU CSSE's dataset for September, removed districts that are not in the 50 states, sorted the files and concatenated them, such that they appear in the exact same pattern as kaggle's testing data.

3 MODEL SELECTION

We start off with a simple AutoRegressive(AR) model with lag of k days. The autoregressive model is a powerful algorithm for predicting future behavior based on the past when there is a correlation between the values in time series. It depends on the linear relation of previous values and follows a stochastic term, and therefore the algorithm appears in a form of stochastic difference equation. We chose this model because it is simple and light-weighted, so it is highly scalable. It does not require many feature labels to predict relatively reliable trends.[5] Also, as the algorithm predicts the future based on a linear combination of the past values, it trains a flexible model that covers a wide range of data patterns. [3] The model is also very sensitive to recurring patterns, which are commonly seen in epidemic progressions.

4 MODEL TRAINING

In this section, we will discuss about the algorithm selection and detailed training process of our model.

To train our model with AR, for every state, we put the difference values of deaths and confirmed cases into two separate AutoRegression model built using linear regression from SKLearn. This is done by selecting the difference values after the lag k days as y values and for every y the previous k days of difference values as x variables. So the target regression model we wants to train for is

$$v_t = b + \alpha_1 v_{t-1} + \ldots + \alpha_k v_{t-k} + \epsilon_t$$

We also split the data and did not include last 10 days of data intro training but as quick validation of the model. We then tried to tune the model with different values of k from 5 to 30. After putting in all the different k and comparing the validation error of the model, we found out that a lag of 10 days looks the best for the model. So we uses difference values and trained a linear model of

$$v_t = b + \alpha_1 v_{t-1} + \ldots + \alpha_{10} v_{t-10} + \epsilon_t [1]$$

which means that the value will be predicted based on previous 10 days of values. The model will then predict the future difference values and calculate the actual numbers based on the predicted difference values by starting from the last day's value and moving forward.

5 MODEL PERFORMANCE

Our current model splits the last 10 days of data as validation data and achieves a validation MAPE of approximately 1.2. For the test cases on Kaggle, we got a score of 3.556 and range in the middle of the leaderboard. We think this performance is generally not bad, but still needs some improvement to get a higher score.

6 EVALUATION AND DISCUSSION

For this stage of the project, our main goal was to build a rough model to familiarize ourselves with the project and the dataset.

Based on the result in the above section, we see that the performance of our current model is 3.5 for the MAPE score on the leaderboard. By using an Autoregressive model we assumed that the confirmed and death numbers depend on their previous values and follow a linear pattern. However, this might not be the case, and the performance is therefore not as promising as we thought. A simple AR model might not be the best options as we only consider linear relationship of difference values, so a more complex model will be needed to improve the performance.

In future optimizations, what we can do is to train separate models on the dataset using algorithms, for example, K nearest neighbours, Random Forest, Convolutional Neural Network and Multi-layer Perception, and to ensemble those models together to form a final model, where the sub-models contribute to it with different weights. The weights for every sub-model could be assigned based on the accuracy of those models.

7 FUTURE PLANS

As mentioned above, there are still a lot can be done to do to improve our current model. In the coming weeks, we will continue with our previous works and optimize the model with this proposed timeline:

Date	Results
Week 6	New Model Researching
Week 7	Model Development
Week 8	Parameter Tuning
Week 9	Report Writing

A more detailed plan for each stage is discussed in the below sections:

7.1 Model Researching

Our goal is to go through some research papers related with covid-19 predictions to get some ideas about what models might be better. One idea is to add Moving Average(MA) model on top of our current AR models to build an ARMA or ARIMA model. This can make our model more precise by adding the MA model to incorporate the possible white noise error terms but we are afraid that the dataset is still too complex that even adding MA the model will not be good enough. Another idea is to incorporate Neural Network, specifically RNN which can help taking into account of historical information and building a more complex model to explain and predict the dataset. We are also considering XGBoost, as it is one of the most popular ML models in the industry. Although it is more powerful on classification and regression on tubular datasets, it has strong support from the community, which should speed up the implementation process, leaving us with more time to tune hyper-parameters[2]. This part will be done by all three of us. We will all try to look up some possible model ideas, and after a week we will brainstorm and discuss what the best idea we have and try to decide on one to go.

7.2 Model Development

We plan to spend a week to develop the basic structure of a model. This will be a bit challenging as we only have three people available. But we think that we can try to break the model into different parts and to work it parallelly for better efficiency. For example if we are building RNN, we can break the model into different layers and can have members working on different layer at the same time.

7.3 Parameter Tuning

We plan to spend another week tuning the parameters and make the model have the best result possible. This is an important step and we plan to have everyone participate by copying the model and trying different parameters or setting on its own. We will then what the best test accuracy is and what parameter setting produces that.

7.4 Report Writing

Lihang will be the main contributer of the report with his great experience in paper writing. He will write the basic structure and distribute different parts to ours

REFERENCES

[1] Jason Brownlee. [n.d.]. Autoregression Models for Time Series Forecasting With Python. https://machinelearningmastery.com/autoregression-models-time-series-forecasting-python/

[2] Hourly Energy Consumption. [n.d.]. [Tutorial] Time Series forecasting with XGBoost. https://www.kaggle.com/robikscube/tutorial-time-series-forecasting-with-xgboost

[3] Rob J Hyndman and George Athanasopoulos. 2018. *Forecasting: Principles and Practice*. Monash University, Australia, Melbourne, Australia.

[4] Regression Methods PennState STAT 501. [n.d.]. Autoregressive Models. https://online.stat.psu.edu/stat501/lesson/14/14.1

[5] The Business Professor. [n.d.]. Autoregression – Definition. https://thebusinessprofessor.com/lesson/autoregression-definition/