# Team 18: CS 145 Kaggle Competition Final Report

Khoa Le
rubikvn2100@gmail.com
Student ID: 813981162

Steven Luong
trungluong93@gmail.com
Student ID: 605361550

Vaishnavi Tipireddy
tvaishu9@gmail.com
Student ID: 705143552

Danning Yu
danningyu@ucla.edu
Student ID: 305087992

## ABSTRACT

Team 18 participated in the CS 145 Kaggle Competition to predict cumulative COVID-19 confirmed cases and deaths for each state in the United States over 2 time periods, 9/1/2020-9/26/2020 and 12/7/2020-12/13/2020. We explored various time series forecasting models to do this, including Holt's exponential smoothing, vector autoregression, recurrent neural network, and multi-layer perceptron regressor. The model used to submit predictions for both rounds because it performed best was Holt's exponential smoothing. This model was fitted using data from 4/12/2020 to 2 days before the start of the prediction period, and its hyperparameters were tuned using data from the last day before the start of the prediction period. Different hyperparameters were used for each state and confirmed cases versus deaths because of differing magnitudes and trajectories for each prediction. Only confirmed cases or deaths data were used for training because other attributes either had incomplete data or were not well correlated with confirmed cases and deaths. Our predictions achieved a mean average percent error (MAPE) of 2.04460% for 9/1/2020-9/26/2020, which is rank 10 on the Kaggle leaderboard. The unofficial MAPE for our 12/7/2020-12/13/2020 predictions is 1.7494% according to COVID-19 data directly scraped from the Johns Hopkins University COVID-19 dataset on 12/13/2020.

**ACM Reference Format:**
Khoa Le, Steven Luong, Vaishnavi Tipireddy, and Danning Yu. 2020. Team 18: CS 145 Kaggle Competition Final Report. In *CS 145 Kaggle Competition*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1111/11111111.111111111

## 1 INTRODUCTION

In late 2019 and early 2020, COVID-19, a coronavirus, began spreading in Wuhan, China. In a couple months it quickly became a global pandemic, causing the closure of borders, cancellation of flights, and shutdown of restaurants, schools, and other institutions worldwide as in-person activities were curtailed and people transitioned to staying at home [8]. The economic costs have been enormous, with immense sums of money being spent on medical equipment and care and businesses losing customers due to public health regulations and customers' economic struggles [10]. The cost to human life has also been great, with over 70 million infections and over 1.5 million deaths attributed to COVID-19 worldwide. The United States has been hit especially hard, with over 16 million people infected and nearly 300,000 dead. These numbers make it the world leader in confirmed cases and deaths [16].

Computer scientists and statisticians have been employed to help by predicting future confirmed cases and deaths for each state in America. Such forecasts are important because they help states and hospitals plan and prepare their facilities, personnel, and equipment in anticipation of either a surge or decrease in cases [1]. For example, states anticipating increased cases would order more equipment and increase their hospital beds. Finally, with the recently announced vaccine authorizations, COVID-19 forecasts could also be used to direct distribution of the vaccine to states or cities with high expected rates of COVID-19.

As students in a data mining class, we can help contribute to this effort of predicting COVID-19 cases and deaths while simultaneously learning about time series forecasting techniques. To judge our efforts, our models will be used to make predictions for confirmed cases and deaths for every state in America for two time periods: 9/1-9/26 and 12/7-12/13. The goal is to make predictions that match the actual confirmed cases and deaths as closely as possible.

## 2 PROBLEM STATEMENT AND FORMALIZATION

The objective of this competition is to make accurate predictions for COVID-19 confirmed cases and deaths for each state in America for two time periods (rounds): 9/1-9/26 and 12/7-12/13. For each time period, we are allowed to use data from days up until the start of the prediction period to train a time series forecasting model. Each prediction datapoint will be for a specific state, on a specific day, for either confirmed cases and deaths. The accuracy of the predictions will be judged using mean absolute percentage error (MAPE), which is defined below, where $n$ is the number of predictions, $p_i$ is the predicted value, and $a_i$ is the actual value:

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\frac{|p_i - a_i|}{a_i}$$

For example, for round 1 predictions from 9/1 to 9/26, a total of (50 states)(26 days)(2 types) = 2600 predictions will be made, where the two types are confirmed cases and deaths. The goal of the competition is to minimize MAPE for both prediction rounds.


*CS 145, Fall 2020, UCLA*

ACM ISBN 978-0-0000-0000-0/0/00…$0.00
https://doi.org/10.1111/11111111.111111111

## 3 RELATED WORK

When COVID-19 emerged as a global pandemic, understanding its spread and predicting the total number of infections and deaths became necessary for hospitals, policymakers, and governments to make plans and policies. All these models incorporate currently known data about COVID-19 cases and deaths and then make varying assumptions about properties of the virus and pandemic, such as infection rates, death rates, social distancing levels, mask usage, and recently, vaccination rates [1] [2].

The model typically employed by researchers for this task is the susceptible-infected-recovered model (SIR) [1], which models people that are susceptible (not yet infected), infectious (capable of spreading the virus to others), and recovered (can no longer spread the virus to others; either sick, dead, or recovered). Variants of this model have been created, such as SEIR [6] (addition of an exposed stage during which people catch the virus but are not yet infectious). Other models used include logistic curve fitting, exponential curve fitting, autoregression (and its variants), and recurrent neural networks [1]. The curve fitting techniques can be used on any form of data, while autoregression and recurrent neural networks are specifically adapted for time series data. Finally, ensemble methods incorporating predictions from many models are typically used to make the final prediction because this allows for a variety of models to be incorporated and typically results in a higher accuracy; this method is used by organizations such as the US Centers for Disease Control [4].

One example of a SIR related model was implemented by UCLA's Statistical Machine Learning Lab [18]. Their model is called SuEIR, which elaborates upon SEIR by adding an unreported and recovered stage, where people become sick and recover. Figure 1 illustrates their SuEIR model, which was able to make predictions that matched the actual numbers extremely closely.

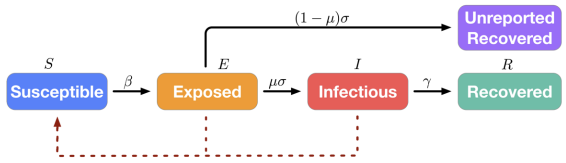


**Figure 1: SuEIR model used by the UCLA Statistical Machine Learning Lab to predict cumulative cases and deaths.**

## 4 DATA SELECTION AND PREPROCESSING

For any data mining project, the selection, preprocessing, visualization, and analysis of data is an important step in helping lay a solid foundation for selecting and training models.

### 4.1 Data Sources

The data we used for our model comes from the Kaggle competition site, which in turn comes from the Johns Hopkins University (JHU) COVID-19 data repository [17]. This data includes COVID-19 confirmed cases, deaths, testing rates, mortality rates, and many other attributes, reported daily for each state. In addition, daily mobility data between the 50 states from Safegraph [14] was also provided through Kaggle. The first dataset that was provided contained data from 4/12 to 8/31, while the second dataset had data from 4/12 to 11/22. Finally, since second round predictions had to be made for 12/7-12/13, we augmented the round 2 data provided to us on Kaggle with data taken directly from the JHU COVID-19 data repository for the time period 11/23-12/6. Since we did not use mobility data in our final model, we did not download any further mobility data from Safegraph beyond what was provided to us on Kaggle.

### 4.2 Data Processing

Since the training data is provided as CSV files with column headers, one can simply use the `pandas.read_csv()` function to load it as a Pandas dataframe in a Python script. To handle missing data values in the CSV that show up as `NaN` in the Pandas dataframe, the `df.dropna()` function can be used. Finally, since we need to predict COVID-19 cases for the 50 states individually, it is helpful to get the data for a particular state. To do this, one can use `df.loc[filter]`, where the filter matches values in the "Province_State" column with the desired state [15].

A transformation was applied to the data to convert dates given in the MM-DD-YYYY format to days since 4/12. This is because for all of our models, we will be using date as the input variable, and so it is easier for the models to handle a numerical input rather than a date input. To do this, each entry in the "Date" column was recalculated to show the number of days since 4/12 by "subtracting" 4/12 from its current value. This simplifies the data, essentially turning it into a time series that starts at time 0, which makes it easier to train models and make forecasts. For example, for the first round of data from 4/12-8/31, the transformed date column has values ranging from 0 to 141. When predictions are made for round 1, it will be made for integer inputs in the range [142, 167] rather than 9/1-9/26.

### 4.3 Data Visualization

It is important to visualize the training data so that one has a sense of what trends, patterns, and/or irregularities may be present. Plots of confirmed cases versus time and deaths versus time using round 1's training data were plotted for each state. Through these plots, it was found that every state has different rates of increase and that trends differ for confirmed cases versus deaths. In terms of modeling, this means we will need to use different hyperparameters or even entirely different models for different states and confirmed cases versus deaths. Finally, some states have days where their confirmed cases or deaths do not increase, and others have sudden spikes or even dips in their total numbers due to reporting irregularities. Our models must be robust enough to handle irregularities like these when being trained.

### 4.4 Correlation Analysis

At the time of our midterm report submission, all our models were univariate and making predictions with only the date and relevant prediction type (confirmed cases or deaths) as an input. There may be other attributes provided to us, such as hospitalization rate and mortality rate, that correlate with confirmed cases and deaths and thus would be useful to include in a multivariate forecasting model. With 10 attributes overall in the training data provided to us, we

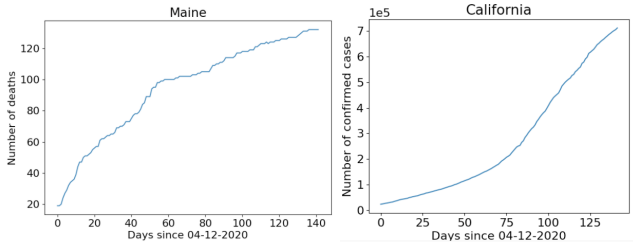


**Figure 2: COVID-19 deaths in Maine and confirmed cases in California from round 1 training data, showing how different states can have very different case curves. In Maine, there was an initial sharp rise followed by a flattened curve, while the opposite is true for California. Also, Maine's data is not very smooth, which can create challenges for models.**

found the correlation between each attribute and confirmed cases or deaths; the results are presented in Figure 3. We used the following formula to determine the correlation $\rho$ between any two variables $x$ and $y$, where $\text{cov}(x, y)$ is the covariance between $x$ and $y$ and $\sigma_i$ is the correlation of variable $i$:

$$\rho = \frac{\text{cov}(x, x)}{\sigma_x \sigma_y}$$

| | Confirmed | Deaths |
|---|---|---|
| Date | 0.141669 | 0.141669 |
| Confirmed | 1 | 0.937184 |
| Deaths | 0.937184 | 1 |
| Recovered | 0.562703 | 0.47416 |
| Active | 0.954674 | 0.902096 |
| Incident_Rate | 0.69047 | 0.552961 |
| People_Tested | 0.682419 | 0.667152 |
| People_Hospitalized | 0.940531 | 0.947172 |
| Mortality_Rate | 0.44059 | 0.555488 |
| Testing_Rate | 0.401912 | 0.400621 |
| Hospitalization_Rate | 0.266107 | 0.289094 |

**Figure 3: Correlation between deaths, confirmed cases, and all other attributes across all 50 states**

After looking at the correlations on a state-by-state basis, we noticed that the correlations varied significantly from state to state. In Arkansas, for example, the correlation between "Incident_Rate" and "Confirmed" is 0.938, which is much higher than the average 0.690. On the other hand, "People_Hospitalized" in Arkansas has a correlation of only 0.118 with "Confirmed" whereas the average is 0.941. This speaks to the different patterns of behavior in each state, as well as the fact that some states are missing data for some of these attributes. After identifying similar variations in most correlated attributes in other states, we considered the idea of training individual models for each state using their most relevant attributes by state, but we ultimately did not have time to try this out. However, we were able to create multivariate models that incorporated

| | Confirmed | Deaths |
|---|---|---|
| Date | 0.034411 | 0.161477 |
| Confirmed | 1 | 0.028188 |
| Deaths | 0.028188 | 1 |
| Recovered | 0.192958 | 0.061849 |
| Active | 0.387691 | -0.046088 |
| Incident_Rate | 0.938292 | 0.028139 |
| People_Tested | 0.206544 | -0.081027 |
| People_Hospitalized | 0.117812 | 0.279735 |
| Mortality_Rate | -0.305465 | 0.498567 |
| Testing_Rate | 0.201747 | -0.074797 |
| Hospitalization_Rate | 0.023636 | 0.115752 |

**Figure 4: Correlation between deaths, confirmed cases, and all other attributes for Arkansas.**

the 5 attributes with correlations across all 50 states: "Confirmed", "Deaths", "Active", "People_Hospitalized" and "Incident_Rate." The relationship of these 5 attributes to confirmed cases or deaths can easily be interpreted. For example, it intuitively makes sense that if there are lots of people hospitalized, then there will be a lot of confirmed cases and deaths. This correlation analysis was very useful when we started building our models, as we saved a lot of time by not having to experiment with all the attributes and focusing on the attributes for which the correlation was high.

## 5 MODELS AND TECHNIQUES

As a team, we approached the problem of forecasting COVID-19 confirmed cases and deaths by trying out a variety of models, starting from the ones taught in class for vector data before realizing that time-series forecasting models would work better. We tried linear regression and kernel regression as our first two models. Then, for time series models we tried recurrent neural networks, multilayer perceptron regressors, autoregression (both single and multivariable versions), and simple exponential smoothing. A variant of simple exponential smoothing, called Holt's exponential smoothing, gave the best results and was used to submit our predictions for both rounds, and so is presented first.

### 5.1 Holt's Exponential Smoothing

Exponential smoothing is a time series prediction technique that predicts outputs for future days based on values of previous days [7]. It is called exponential smoothing because the weight of each previous observation decreases exponentially as the observation gets older. This makes intuitive sense for COVID-19, as the number of confirmed COVID-19 cases from many days ago should have a much smaller effect than confirmed cases from the previous day. Most people can only infect others for a period of around 10 days; active cases from over 14 days ago essentially do not contribute to any more infections. Exponential smoothing makes forecasts according to the following equation, which predicts a value for the $(t + 1)$th day given 2 previous days of observations [7]:

$$y_{t+1} = \alpha y_t + (1 - \alpha) y_{t-1}$$

The variable $\alpha$ is called the smoothing level and is allowed to take on values between 0 and 1, inclusive. If $\alpha = 1$, then this is simply a naive predictor since the equation reduces to $y_{t+1} = y_t$. As the value of $\alpha$ decreases, so does the weight of the more recent observations. Thus, $\alpha$ is an important hyperparameter that needs to be tuned.

The shortcoming of this model is that it cannot handle data where there is a trend; over time, it converges to a particular value. Cumulative COVID-19 cases and deaths will be constantly increasing (for the foreseeable future), and so a slightly more complex model is needed to handle this monotonically increasing trend.

To do this, we can employ a second order simple exponential smoothing model (also called double exponential smoothing or Holt's exponential smoothing) [7], which derives the "second order" in its name by incorporating a trend in its predictions. By capturing the monotonically increasing trend in the COVID-19 data and the rate at which this increase is occurring, this model works much better than simple exponential smoothing. It also makes the model fairly resistant to occasional jumps, dips, or flatlines in the training data, as long as such irregularities do not happen near the end of the training data set. To tune the trend, another hyperparameter called $\beta$ is added, which is called the smoothing trend and the domain $[0, 1]$. The model can be described using the following equations, where $y$ is the prediction, $h$ is how many days into the future we want to predict, $l$ is called the level component, and $b$ is called the trend component. The level component determines the magnitude of the prediction, while the trend component determines the trend of the predictions. The hyperparameter $\alpha$ from simple exponential smoothing is retained in the equation for the level component [7].

$$y_{t+h} = l_t + hb_t$$
$$l_t = \alpha y_t + (1 - \alpha)(l_{t-1} + b_{t-1})$$
$$b_t = \beta(l_t - l_{t-1}) + (1 - \beta)b_{t-1}$$

To create this model, the Python `statsmodels` package [9] was used, which contains an implemented Holt's model. To train the data, first the model was fitted to all but the last day of training data for each of the 2 rounds. Then, the hyperparameters $\alpha$ and $\beta$ were tuned individually for each state and confirmed cases versus deaths by doing a grid search for the optimal combination over the following values using the last day of training data:

$$\alpha = [0.05, 0.1, 0.15, 0.2, 0.25, 0.275]$$
$$\beta = [0.425, 0.45, 0.5, 0.55, 0.6, 0.65, 0.7]$$

This set of values was picked based on training on a wider set of training data (10 days) and recognizing that values outside of these upper and lower bounds resulted in lower validation MAPE scores. After training, forecasts were made for the necessary time interval (9/1-9/26 or 12/7-12/13). The results, which were outpointed as floating point numbers, were then rounded to the nearest integer.

Carrying out the procedure described above resulted in a model that handled irregularities in the training data (days with no increase in cumulative counts, or even a decrease) and made fairly accurate predictions. A desirable property it exhibited was that the predictions it made were always increasing with respect to time, which was something we tried to achieve in some of our other models but had trouble doing so.

When Holt's model was first implemented by tuning one set of hyperparameters for all 50 states' confirmed cases and another for deaths using the procedure described above, we achieved a MAPE of 2.31% for round 1 predictions. Then, by tuning each state's hyperparameters individually, we were able to decrease the MAPE of round 1 predictions to 2.04460%, which gave the team the 10th position on the leaderboard, and so it was decided that we would use this model to make round 2 predictions. For round 2, the same procedure was carried out, but the predictions were all multiplied by 1.005 (an increase of 0.5%) due to the anticipation that after Thanksgiving, there will be a large spike in confirmed cases and deaths due to travel and holiday gatherings. This manual adjustment was done after making predictions for 12/5-12/6 and noticing that in the vast majority of states, the model was underpredicting. A slight upwards adjustment of 0.5% was seen as a small change that we hoped would marginally improve our MAPE.

A possible improvement to explore is using the fact that the model also has exponential and damped variants that forecast higher and lower values compared to the regular variant respectively (but still follow the overall increasing trend). The exponential variant would work well for states in the Midwest like North Dakota and Wisconsin, which saw a spike of cases in September, while the damped variant would work well for states in the Northeast like Maine and Vermont where cases stayed relatively low. However, I was not able to find a way to determine when to use damped, linear, or exponential Holt's models based solely on the training data given to us, and so this is an area I'd like to improve on.

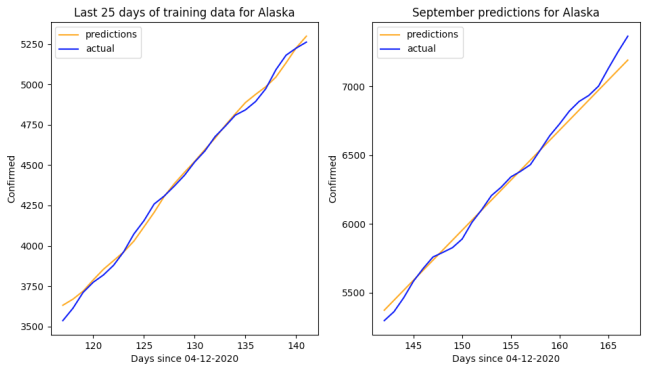


**Figure 5: Holt's exponential model predictions and actual numbers for cumulative COVID-19 cases in Alaska for 9/1-9/26. The model makes extremely accurate predictions.**

## 5.2 Autoregression

Autoregression (AR) [3] is the idea that we can predict outputs for a particular day based on a linear combination of outputs from previous days. That is, for an AR model of order $p$, which means it uses data from the past $p$ days, the output $y$ on day $t$ is determined by the equation:

$$y_t = c + \epsilon + \sum_{i=1}^{p} p_i y_{t-i}$$

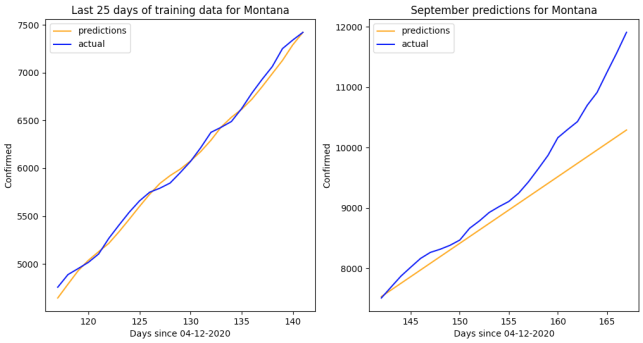


**Figure 6: Holt's exponential model predictions and actual numbers for cumulative COVID-19 cases in Montana for 9/1-9/26. The predictions are accurate until there is a sudden increase in confirmed cases starting around mid-September, which is difficult to foresee given data only up until 8/31.**

where each $p_i$ is a weight that needs to be tuned, $c$ is a constant, and $\epsilon$ is some normally distributed noise. Using the `statsmodels` package, AR models of various orders were created and trained on the first 131 days of data from round 1. Then were then tested on the remaining 10 days of data, and it was found that models of order 4 resulted in the lowest overall error.

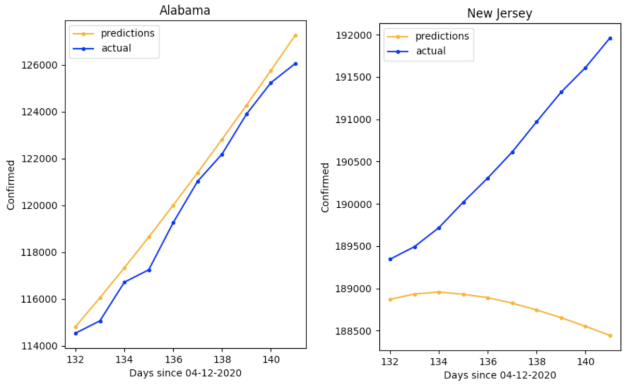


**Figure 7: AR model of order 4 for COVID-19 cumulative confirmed cases in Alabama and New Jersey. This model works well for Alabama, but for New Jersey, it starts predicting a decrease in total cumulative COVID-19 cases, causing it to be highly inaccurate.**

When the AR model of order 4 was applied for all 50 states for both confirmed cases and deaths, it resulted in an MAPE error of 8.872%. This high number is probably due to mispredictions for states like New Jersey, where AR started making the nonsensical prediction that cumulative cases would decrease (see Figure 7). There were also states that had reported no deaths for a couple days, and so for those states, the AR model predicted no deaths for the future, which is also quite inaccurate. Thus, the main improvement that needs to be looked into is how to create an AR model that is monotonically increasing. Furthermore, the AR model is the simplest model in a family of autoregression models, and so to improve its prediction capabilities, we tried its multivariate version, called vector autoregression.

## 5.3 Vector autoregression

Vector autoregression (VAR) [13] is an extension of the autoregression model to multiple attributes (variables) that preserves the idea of using a linear combination of past days, or lags, to forecast future values. We used the Python `statsmodels` package to build the VAR model. The equation of a $p$th order VAR prediction model with $K \times T$ multivariate time series $Y$, where $T$ is the number of observations (days), $K$ is the number of variables, $A_i$ is a $K \times K$ coefficient matrix, $u_t$ is some normally distributed error:

$$Y_t = v + A_1 Y_{t-1} + .... + A_p Y_{t-p} + u_t$$

To determine which attributes to incorporate into the model, we created a correlation matrix (see §4.4 Correlation Analysis) between all the attributes and picked the attributes that have the highest correlation to confirmed cases and deaths. This saves us from having to try out all possible combinations of the 8 attributes given (aside from confirmed cases and deaths), which would be extremely time-consuming. According to the correlation data table, the top 3 attributes with the highest correlation values are "Active," "Incident_Rate," and "People_Hospitalized," and so we incorporated various combinations of these 3 attributes into the VAR model. For the case where some states do not have any data for a particular attribute, we left it out of the VAR model and used the remaining attributes instead.

We tuned the model in 2 ways: through different values of $p$, the order of the VAR, and different combinations of the 3 selected attributes mentioned above. Finally, there were still states for which the cumulative cases or deaths were predicted to decrease, and so if this occurred, during post-processing, the predicted value would simply be set to the previously predicted value. Figure 8 shows the results of this tuning of attributes and lag order, The optimal MAPE of 2.713% was achieved using "Deaths", "Confirmed" and "Active" with $p = 6$.

## 5.4 MLP Regressor

Multi-layer perceptron (MLP) regressor [11] is a type of feed-forward artificial neural network that consists of at least three layers: input, hidden and output layer. Between those layers, there are activation functions like a typical neural network, and so it is well designed to handle nonlinear data and trends. Also, like a typical neural network, back-propagation is used to update the parameters of the network.

To start off, only "Date", "Deaths" and "Confirmed" columns are used. 2 MLP models were built: one for predicted confirmed cases and another for predicting deaths.

To tune parameters, we kept the number of hidden layers at 100. Only solver type, number of iterations, and learning rate were tuned, and this resulted in MAPE for round 1 predictions ranging from 2.41% to 4.65%. The best result of a 2.41% MAPE was obtained with solver "lbfgs", max iterations 10000, and learning rate 0.0001. The MLP regressor works based on the idea of neural networks, so to improve it, it probably needs more data than just confirmed

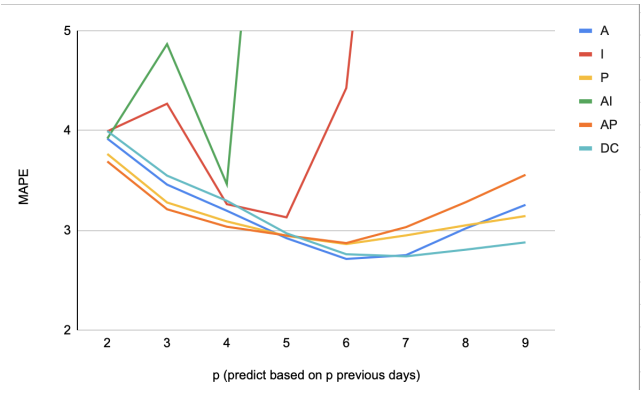


**Figure 8: MAPE results for tuning VAR with different values of $p$ and added attributes. "A" stands for "Active," "I" stands for "Incident_Rate," and "P" stands for "People_Hospitalized." Deaths and confirmed cases were always included in the model.**

| p | No added | A | I | P | AI | AP | IP | AIP |
|---|---|---|---|---|---|---|---|---|
| 2 | 3.995 | 3.919 | 3.992 | 3.765 | 3.92 | 3.69 | inf | inf |
| 3 | 3.548 | 3.458 | 4.269 | 3.279 | 4.866 | 3.211 | inf | inf |
| 4 | 3.296 | 3.196 | 3.262 | 3.089 | 3.463 | 3.036 | inf | inf |
| 5 | 2.97 | 2.921 | 3.13 | 2.944 | 24.313 | 2.948 | inf | inf |
| 6 | 2.76 | **2.713** | 4.426 | 2.862 | 7.098 | 2.872 | inf | inf |
| 7 | 2.739 | 2.751 | 49.187 | 2.949 | 12.79 | 3.032 | inf | inf |
| 8 | 2.806 | 3.019 | inf | 3.05 | inf | 3.282 | inf | inf |
| 9 | 2.879 | 3.255 | inf | 3.143 | inf | 3.556 | inf | inf |

**Figure 9: MAPE results for tuning VAR with different values of $p$ and added attributes in tabular form. "inf" stands for a number greater than 10.**

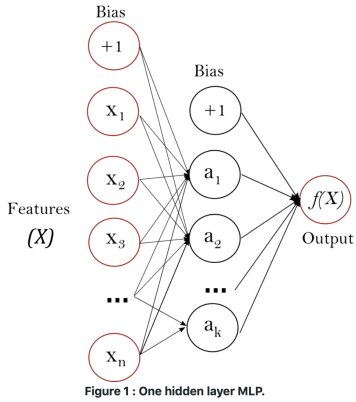


**Figure 10: Multi-layer perceptron with 3 layers: input, output, and hidden layers.**

cases or deaths, as well as more data in general. Attributes such as number of people hospitalized and mortality rate would probably help increase the accuracy of the MLP regressor.

## 5.5 Recurrent Neural Network

Recurrent neural networks (RNN) [5] are a specific type of neural network specially designed to handle time series data such as text

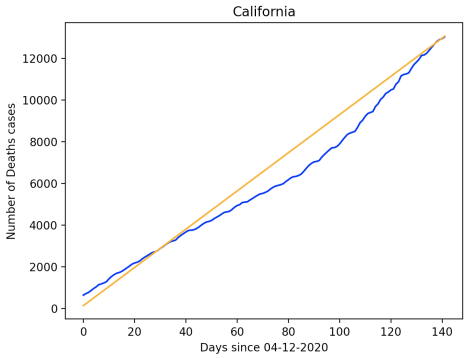


**Figure 11: Deaths due to COVID-19 using MLP regressor model. The prediction (yellow line) closes matches the actual cases (blue line).**

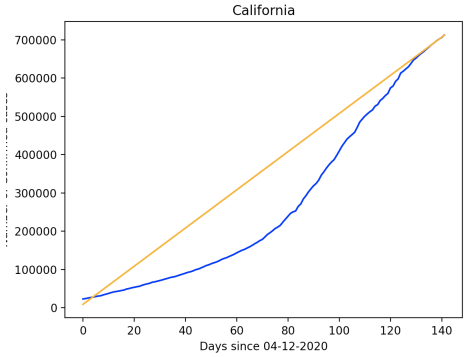


**Figure 12: Confirmed COVID-19 cases using MLP regressor model. The prediction does not work well because the actual number of cases is not perfectly linear. Hence, a non-linear prediction model is needed for better accuracy.**

and speech. They can be used for both classification problems, such as whether it will rain or not given a past time series of temperature, humidity, and wind speed measurements, as well as making predictions, such as cumulative COVID-19 confirmed cases and deaths. An RNN that takes in multiple variables to predict a single attribute is called a many-to-one RNN, and this was used for the project, with multiple attributes being used to predict either confirmed cases or deaths. An attempt to make a many-to-many RNN model to predict confirmed cases and deaths did not work as well so we did not look further into this type of RNN.

During the training process, RNN models can suffer from the vanishing gradient problem, where due to a large number of neurons a backwards pass has to go through, the gradient update becomes very small or even 0, causing cells at the start of the network to receive very small or nonexistent updates. To deal with this problem, the long short term memory (LSTM) model [12] was developed.

Each LSTM cell gets 3 inputs, namely $x_t$, $C_{t-1}$, and $h_{t-1}$, and then returns $C_t$ and $h_t$ according to the following equations. The variables $U$ and $W$ are weight matrices that are optimized during

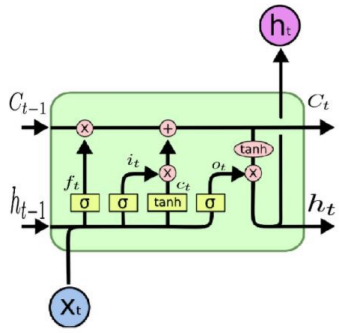


**Figure 13: An LSTM cell diagram that shows the flow of vectors.**

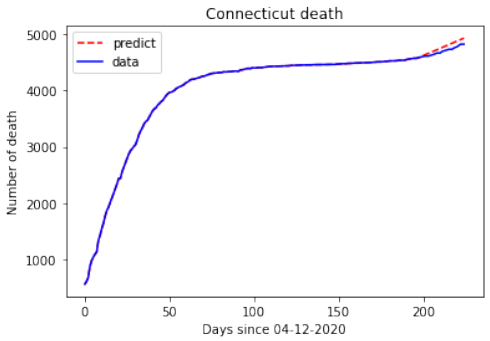


**Figure 14: RNN: with $k = 5$, the model found the trend and the prediction follows the trend with only 1.490% error.**

the training process. The output of the $\sigma(\cdot)$ function lies in the domain $[0, 1]$ and is used to determine how much of the subsequent neuron to activate, just like a traditional neural network. The full RNN is then formed by concatenating multiple LSTM units together and training them.

$$f_t = \sigma(x_t U_f + h_{t-1} W_f)$$
$$i_t = \sigma(x_t U_i + h_{t-1} W_i)$$
$$o_t = \sigma(x_t U_o + h_{t-1} W_o)$$
$$\hat{C}_t = tanh(x_t U_g + h_{t-1} w_g)$$
$$C_t = \sigma(f_t * C_{t-1} + i_t * \hat{C}_t)$$
$$h_t = tanh(C_t) * o_t$$

To train the LSTM, the cumulative confirmed cases and deaths need to first be preprocessed by extracting the last $k$ days of data:

$$x_i = [\text{data}_{i-k}, ..., \text{data}_{i-1}]$$

Each LSTM unit will then be fed with $x_i$ and its output $h_i$ compared with the ground truth for that day. The model was trained with L2 regularization to avoid overfitting. To get 26 days of predictions, predictions were made 1 day at a time, and then the prediction was used as part of the data used to predict the next day.

For a small value of $k$, such as $k = 5$, the model did a good job in finding the recent trend in cumulative cases or deaths and following that trend. However, the downside is that it cannot predict sudden change in trends.

For a higher value of $k$, such as $k = 14$, the model tends to overfit to the training data and then produces predictions that are quite inaccurate with an odd trend, as can be seen in the figures.

Thus, from trying different values of $k$ it is evident that when $k$ becomes too large, the RNN will overfit. On the other hand, if $k$ is small, it simply predicts based on the trend it observed and cannot handle changes in this trend. Thus, $k$ is an important hyperparameter for RNNs. The model gave good predictions for days at the start of the prediction periods when the value of $k$ was low, such as $k = 1$ and $k = 2$, which is probably because data from the most recent days is the best indicator of what future confirmed cases and deaths will be. In the end, it was found that $k = 10$ achieved the lowest MAPE of 6.10%, but this is still quite high, and so we did not use this model for our final predictions.

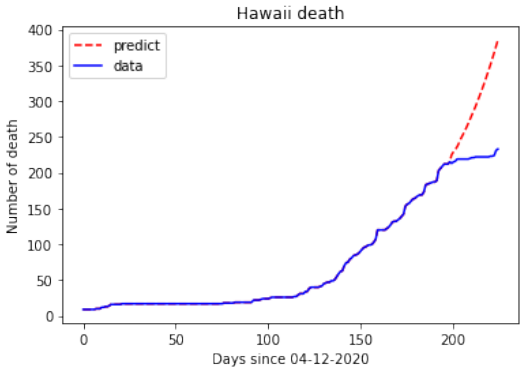


**Figure 15: RNN: With $k = 5$, the model found the trend in this case, but then trend suddenly changed, leading to a high error of 33.151%.**

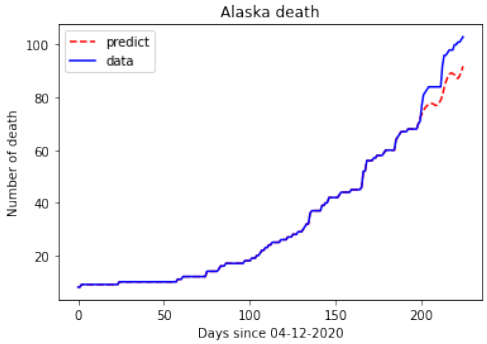


**Figure 16: With $k = 14$, the prediction has the shape similar to the shape of the training data, which indicates overfitting. The error for this state is 9.01%.**

## 5.6 Linear Regression

Linear regression is one of the simplest prediction models available and is well-suited for data where there is a constant rate of increase. Although this trend is obviously not true for cumulative COVID-19 cases and deaths, we chose to build one at the start of

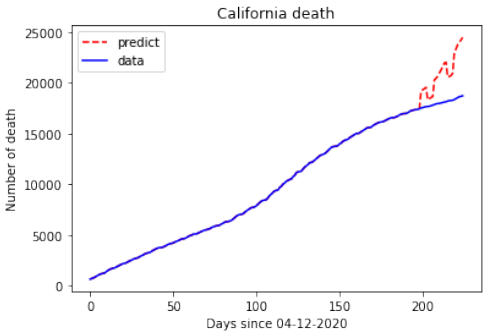


**Figure 17: With $k = 14$, once again the model fits the training data too well, resulting in an abnormal prediction shape and inaccurate predictions that lead to a 16.189% error.**

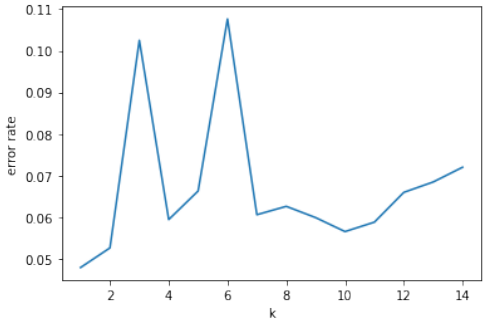


**Figure 18: RNN: MAPE error versus different values of $k$. Ignoring low values of $k$, the optimal value is at $k = 10$.**

the project to get used to manipulating the data and using Pandas dataframes, as well as to evaluate the performance of a simple naive model. We implemented a simple linear regression model using the `scikit-learn` package, and the resulting predictions for round 1 had an MAPE of around 14.57%. As expected, the error for this model is very high because the rate of COVID-19 infections and deaths has not stayed constant since the pandemic started. Furthermore, unlike autoregression or other models, linear regression tries to fit all the data at once instead of using data from only the last couple days. This is not very useful, as data from April should have little influence on case predictions in September. A better model would mainly only consider data from the previous few days.

## 5.7 Kernel Regression

As a natural extension of linear regression, the next model we tried was kernel regression, as it can handle the nonlinear trends seen in the training data through a kernel function. We implemented the model using the `scikit-learn` package by training it on the first 131 days of training data from round 1 and then evaluating it on the remaining 10 days. We tried polynomial kernels of varying degrees, as well as a Gaussian kernel, and it was found that a polynomial kernel of degree 3 seemed to work best. However, despite this, the predictions made for the next 10 days were not very accurate, and furthermore, as predictions were made farther out for September,

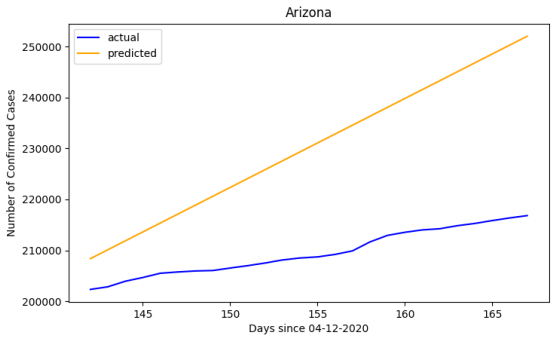


**Figure 19: Number of confirmed COVID-19 cases in Arizona in September predicted using linear regression. The predictions are much higher than the ground truth.**

they actually showed a decreasing trend, which makes no sense for cumulative confirmed cases or deaths. This is most likely due to the fact that a polynomial model fits the training data really well, but then has no guarantees of behavior for predictions outside the domain of the training data. Furthermore, polynomials are not guaranteed to be monotonically increasing. The result was an overall MAPE of 18.44% on the round 1 test data, which is quite high and in fact even worse than linear regression.

We then realized that fitting the kernel regression model to the entire set of data from April to August does not make sense, as case numbers in April have little bearing on the number of cases in September. We tried using differing amounts of previous days' data to train the model and found that using the last 5 days of training data resulted in an optimal MAPE. With this adjustment, the MAPE error was decreased to 9.423%.

The fact that only using the last 5 days of data resulted in more accurate predictions hints at the fact that a good model will probably use a weighted average of the past couple days of data to predict the future, so we switched to using an autoregression model. Also, since COVID-19 total cases and deaths will only ever go up, we should explore ways to add an increasing monotonicity constraint to our future models.

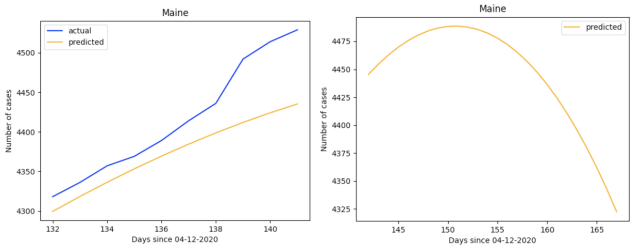


**Figure 20: Kernel regression for COVID-19 cases in Maine with a polynomial kernel of degree 3 trained on the last 5 days of training data. The left graph shows the predictions versus expected results for the last 10 days, while the right graph shows predictions for 9/1-9/26. The predictions take on a downwards trend that results in a high MAPE.**

# 6 RESULTS, OBSERVATIONS, AND INSIGHTS

## 6.1 Model Results Summary

Table 1 shows all the models that were explored and their MAPE for round 1 (9/1-9/26). For reference, the baseline score for the Kaggle competition was 2.93299. Holt's exponential smoothing had the lowest MAPE for round 1, so we used it to make predictions for round 2 as well.

**Table 1: Model Performance**

| Model Name | MAPE Score (%) |
|---|---|
| Holt's Exponential Smoothing | 2.04460 |
| Vector Autoregression | 2.71 |
| Vector Autoregression Variant 2 | 2.92 |
| MLP Regressor | 2.41 |
| Multilayer RNN | 6.10 |
| Autoregression | 8.87 |
| Kernel Regression | 9.42 |
| Linear Regression | 14.57 |

## 6.2 Observations and Insights

This section summarizes the various observations and insights that helped us to select and improve our models.

- Vector data regression methods do not work well because they are better suited for making in-range forecasts (rather than predictions). This relates to the idea of the dangers of extrapolating a regression model.
- Datapoints from April have little bearing on what the case numbers will be for September or December. This is one of the reasons why vector data regression methods don't work very well. Thus, only the last few points of data are the most important to making forecasts.
- Each state has different magnitudes and trajectories for confirmed cases and deaths, so tuning hyperparameters individually for each state and confirmed cases versus deaths results in more accurate predictions.
- Making predictions 26 days into the future for round 1 is difficult because there can be unpredictable factors (such as changes in public health restrictions, a sudden increase or decrease in social distancing) that make it hard for the model to anticipate and know about. Our models can only predict trends based on the trends present within training data.

# 7 CONCLUSION

In this project, we explored a variety of time series forecasting models to make predictions for COVID-19 cumulative cases and deaths from 9/1-9/26 and 12/7-12/13. We tried a variety of models, including Holt's exponential smoothing, single variable and multivariable autoregression, recurrent neural networks, multilayer perceptron regressor, kernel regression, and linear regression. The model that gave the best performance was Holt's exponential smoothing with hyperparameters tuned individually for each state for both confirmed cases and deaths, giving us an MAPE of 2.04460% for round 1 predictions, which is 10th place on the Kaggle leaderboard. The unofficial MAPE for 12/7-12/13 predictions, obtained using the JHU COVID-19 dataset data on 12/13, is 1.7494%.

One improvement to make to our models is to tune them for each state individually like we did with Holt's exponential smoothing. However, doing this can be time-consuming, so a possible way to avoid is to split the states into groups based on similarity in confirmed cases and deaths and then train just a model for each group using its shared highly correlated variables.

Another possible improvement is to incorporate the graph mobility data. The number of people travelling state to state is sure to affect the number of confirmed cases for a particular state, as well as the number of deaths after some lag period. Since COVID-19 symptoms and a positive COVID-19 test do not occur until a few days after infection, there is a delay to consider between the input (number of people travelling) and output (number of cases). We would first find states where the number of incoming people has a high correlation to the number of cases presenting a couple days later, and only then incorporate that attribute into our model.

This project overall was a really good learning experience in terms of working with data, experimenting with various models, and identifying places of improvement for those models. The relevancy of the project to current world affairs made it all the more exciting. We appreciated working on this with a team, as each member could try out different ideas and we could collectively learn from the results. We now have some practical experience with time series forecasting that will prove useful for similar problem statements in the future.

# 8 TASK DISTRIBUTION FORM

See table 2 below for the task distribution table.

**Table 2: Task Distribution Form**

| Task | Participating Individuals |
|---|---|
| Data Preprocessing | Danning |
| Vector Autoregression | Steven |
| Vector Autoregression Variant 2 | Vaishnavi |
| MLP Regressor | Steven |
| Multilayer RNN | Khoa |
| Autoregression | Danning, Steven, Vaishnavi |
| Kernel Regression | Danning, Steven |
| Linear Regression | Vaishnavi, Danning |
| Writing midterm and final reports | All |

# REFERENCES


[1] Aniruddha Adiga, Devdatt Dubhashi, Bryan Lewis, Madhav Marathe, Srinivasan Venkatramanan, and Anil Vullikanti. 2020. Mathematical Models for COVID-19 Pandemic: A Comparative Analysis. *Journal of the Indian Institute of Science* 100 (2020), 14 pages. https://doi.org/10.1007/s41745-020-00200-6

[2] American Hospital Association. 2020. *COVID-19 Models: Forecasting the Pandemic's Spread.* American Hospital Association. Retrieved December 13, 2020 from https://www.aha.org/guidesreports/2020-04-09-compendium-models-predict-spread-covid-19

[3] Jason Brownlee. 2020. *Autoregression Models for Time Series Forecasting With Python.* Machine Learning Mastery. https://machinelearningmastery.com/autoregression-models-time-series-forecasting-python/

[4] CDC. 2020. *Forecasts of COVID-19 Deaths.* Retrieved December 13, 2020 from https://www.cdc.gov/coronavirus/2019-ncov/covid-data/forecasting-us.html

[5] IBM Cloud Education. 2020. *Recurrent Neural Networks.* IBM. Retrieved December 13, 2020 from https://www.ibm.com/cloud/learn/recurrent-neural-networks

[6] Shaobo He, Yuexi Peng, and Kehui Sun. 2020. SEIR modeling of the COVID-19 and its dynamics. *Nonlinear dynamics* (18 Jun 2020), 1–14. https://doi.org/10.1007/s11071-020-05743-y

[7] R.J. Hyndman and G. Athanasopoulos. 2018. *Forecasting: Principles and Practice* (2nd ed.). Otexts: Melbourne, Australia. Retrieved December 13, 2020 from https://otexts.com/fpp2/

[8] T. Ibn-Mohammed, K. B. Mustapha, J. Godsell, Z. Adamu, K. A. Babatunde, D. D. Akintade, A. Acquaye, H. Fujii, M. M. Ndiaye, F. A. Yamoah, and S. C. L. Koh. 2021. A critical analysis of the impacts of COVID-19 on the global economy and ecosystems and opportunities for circular economy strategies. *Resources, conservation, and recycling* 164 (Jan 2021), 105169–105169. https://doi.org/10.1016/j.resconrec.2020.105169

[9] Jonathan Taylor Josef Perktold, Skipper Seabold. 2014. *Exponential smoothing.* Retrieved December 13, 2020 from https://www.statsmodels.org/stable/examples/notebooks/generated/exponential_smoothing.html

[10] World Health Organization. 2020. *Timeline of WHO's response to COVID-19.* World Health Organization. Retrieved December 13, 2020 from https://www.who.int/news/item/29-06-2020-covidtimeline

[11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[12] Michael Phi. 2018. *Illustrated Guide to LSTM's and GRU's: A step by step explanation.* Retrieved December 13, 2020 from https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21

[13] Selva Prabhakaran. [n.d.]. *Vector Autoregression (VAR) – Comprehensive Guide with Examples in Python.* Retrieved December 13, 2020 from https://www.machinelearningplus.com/time-series/vector-autoregression-examples-python/

[14] Safegraph. 2020. *Weekly Patterns (v2).* Safegraph. Retrieved December 13, 2020 from https://docs.safegraph.com/docs/weekly-patterns

[15] Pandas Development Team. 2020. *Pandas Documentation.* Pandas Development Team. Retrieved December 13, 2020 from https://pandas.pydata.org/docs/

[16] The New York Times. 2020. *Coronavirus World Map: Tracking the Global Outbreak.* The New York Times. Retrieved December 13, 2020 from https://www.nytimes.com/interactive/2020/world/coronavirus-maps.html

[17] Johns Hopkins University. 2020. *Johns Hopkins University COVID-19.* Center for Systems Science and Engineering (CSSE) at Johns Hopkins University. Retrieved December 13, 2020 from https://github.com/CSSEGISandData/COVID-19

[18] Difan Zou, Lingxiao Wang, Pan Xu, Jinghui Chen, Weitong Zhang, and Quanquan Gu. 2020. Epidemic Model Guided Machine Learning for COVID-19 Forecasts in the United States. *medRxiv* (2020). https://doi.org/10.1101/2020.05.24.20111989