

Capstone Project - CYO

Jean-Claude de Villeres

March 19, 2020

Contents

1	Introduction	1
2	Methods and Analysis	2
2.1	Data Exploration	2
2.1.1	Statistics as of March 11	3
2.1.2	Plot number of cases over time	3
2.1.3	Countries with most cases	8
2.1.4	Spatial analysis using ggplot and map_data	9
2.2	Data Modeling	12
2.2.1	Naive	12
2.2.2	Simple Exponential Smoothing	13
2.2.3	HoltWinters	14
2.2.4	ARIMA	15
3	Results	16
3.1	Model performance	20
3.2	Model forecast	20
4	Conclusion	20
4.1	Summary	21
4.2	Recommendations	21

1 Introduction

This capstone report aims to explore the Coronavirus dataset from Kaggle and do a straight-forward forecast on the number of cases for the next 50 days. We will explore the data to find various trends and patterns and also to forecast using different models future Coronavirus cases. The dataset consists of around 4,935 rows of records of daily cases from January 22 to March 11. The data is mainly sourced out from Johns Hopkins University for education and academic research purposes, where they aggregated data from various global resources such as, World Health Organization, US Center for Disease Control, Canadian Government, China CDC, Italy Ministry of Health and others.

The following are the column descriptions:

- * Sno - Serial number
- * ObservationDate - Date of the observation in MM/DD/YYYY
- * Province/State - Province or state of the observation (Could be empty when missing)
- * Country/Region - Country of observation
- * Last.Update - Time in UTC at which the row is updated for the given province or country
- * Confirmed - Cumulative number of confirmed cases till that date
- * Deaths - Cumulative number of deaths till that date
- * Recovered - Cumulative number of recovered cases till that date

Here is the link for the dataset <https://www.kaggle.com/sudalairajkumar/novel-corona-virus-2019-dataset>

2 Methods and Analysis

We will explore the data using dplyr summarize / groupby methods and ggplot to analyze patterns in the dataset. We will also try to print a global map of cases before and after the pandemic. Equipped with the resulting information, we will generate various models to forecast the number of cases for the next 50 days.

2.1 Data Exploration

We will group data to find various patterns as well as wrangle with the dataset and perform summarisations to find relevant trends.

```
head(covid_data)
```

```
##   Sno ObservationDate Province.State Country.Region      Last.Update Confirmed
## 1   1     2020-01-22       Mainland China 1/22/2020 17:00         1
## 2   2     2020-01-22       Mainland China 1/22/2020 17:00        14
## 3   3     2020-01-22       Mainland China 1/22/2020 17:00         6
## 4   4     2020-01-22       Mainland China 1/22/2020 17:00         1
## 5   5     2020-01-22       Mainland China 1/22/2020 17:00         0
## 6   6     2020-01-22       Mainland China 1/22/2020 17:00        26
##   Deaths Recovered
## 1      0        0
## 2      0        0
## 3      0        0
## 4      0        0
## 5      0        0
## 6      0        0
```

```
summary(covid_data)
```

```
##      Sno      ObservationDate      Province.State      Country.Region
## Min.   : 1   Min.   :2020-01-22   :1815   Mainland China:1548
## 1st Qu.:1234 1st Qu.:2020-02-11   Gansu    : 51   US          :1003
## Median :2468 Median :2020-02-26   Hebei    : 51   Australia   : 225
## Mean   :2468 Mean   :2020-02-22   Anhui    : 50   Canada      : 146
## 3rd Qu.:3702 3rd Qu.:2020-03-06   Beijing   : 50   France       : 50
## Max.   :4935  Max.   :2020-03-11   Chongqing: 50   Japan        : 50
##                   (Other) :2868   (Other)   :1913
##      Last.Update      Confirmed      Deaths      Recovered
## 2020-02-01T19:43:03: 63   Min.   : 0.0   Min.   : 0.00   Min.   : 0
## 2020-02-01T19:53:03: 63   1st Qu.: 1.0   1st Qu.: 0.00   1st Qu.: 0
## 2020-02-24T23:33:02: 63   Median : 9.0   Median : 0.00   Median : 1
## 1/31/2020 23:59     : 62   Mean   : 577.6   Mean   : 17.69   Mean   : 201
## 1/30/2020 16:00     : 58   3rd Qu.: 93.0   3rd Qu.: 1.00   3rd Qu.: 14
## 1/29/2020 19:30     : 54   Max.   :67773.0   Max.   :3046.00   Max.   :49134
## (Other)           :4572
```

2.1.1 Statistics as of March 11

```
# Filter data for March 11
covid_data_recent <- covid_data %>% filter(ObservationDate == "2020-03-11")
head(covid_data_recent)

##   SNo ObservationDate Province.State Country.Region           Last.Update
## 1 4720      2020-03-11        Hubei Mainland China 2020-03-11T10:53:02
## 2 4721      2020-03-11          Italy
## 3 4722      2020-03-11          Iran
## 4 4723      2020-03-11    South Korea
## 5 4724      2020-03-11       France
## 6 4725      2020-03-11       Spain
##   Confirmed Deaths Recovered
## 1     67773    3046    49134
## 2     12462     827    1045
## 3      9000    354    2959
## 4      7755     60    288
## 5      2281     48     12
## 6      2277     54    183

# Show number of confirmed
sum(covid_data_recent$Confirmed)

## [1] 125865

# Show number of recovered
sum(covid_data_recent$Recovered)

## [1] 67003

# Show number of deaths
sum(covid_data_recent$Deaths)

## [1] 4615
```

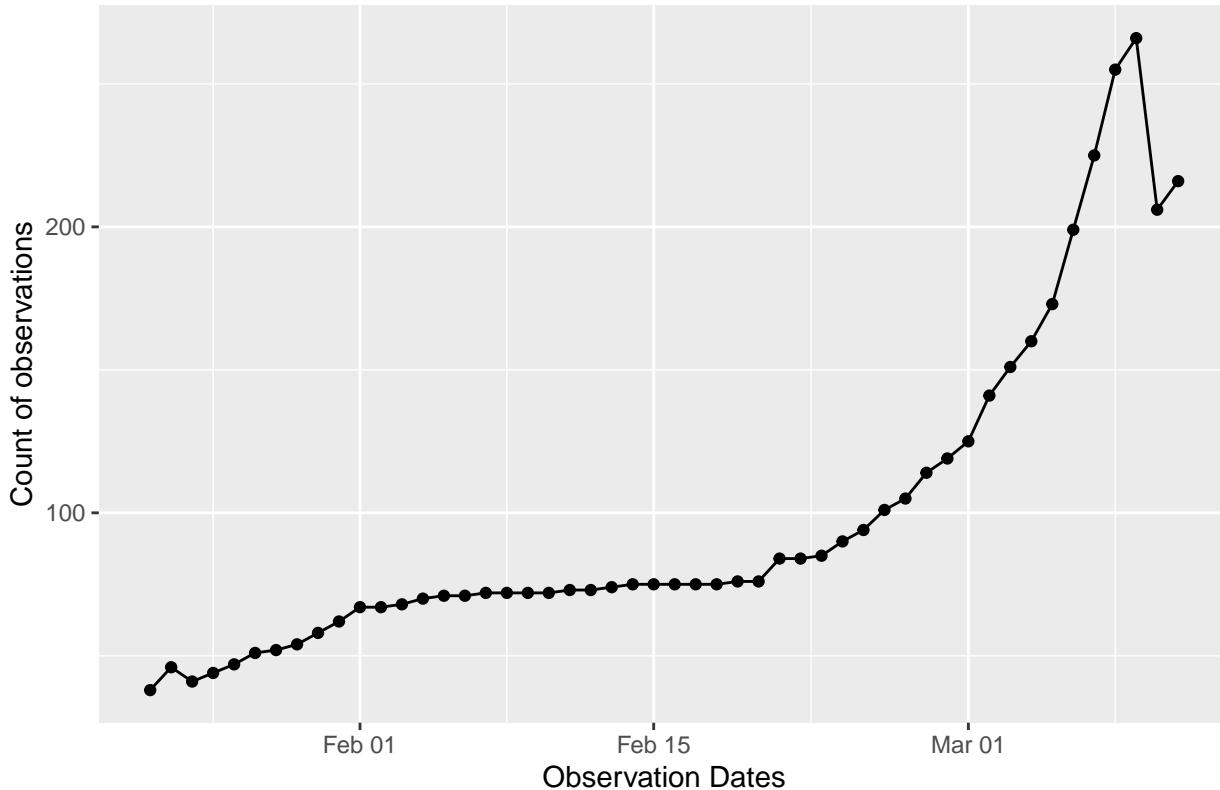
2.1.2 Plot number of cases over time

```
# Show number of observations over time
covid_observations <- covid_data %>%
  group_by(ObservationDate) %>%
  summarize(count = n())
head(covid_observations %>% arrange(desc(count)))

## # A tibble: 6 x 2
##   ObservationDate count
##   <date>           <int>
## 1 2020-03-09       266
## 2 2020-03-08       255
## 3 2020-03-07       225
## 4 2020-03-11       216
## 5 2020-03-10       206
## 6 2020-03-06       199
```

```
# Plot the number of observations over time
covid_observations %>% ggplot(aes(x=ObservationDate, y=count, group=1)) +
  geom_point() + geom_line() + labs(x = "Observation Dates", y = "Count of observations",
  title = "Count of observations over time")
```

Count of observations over time

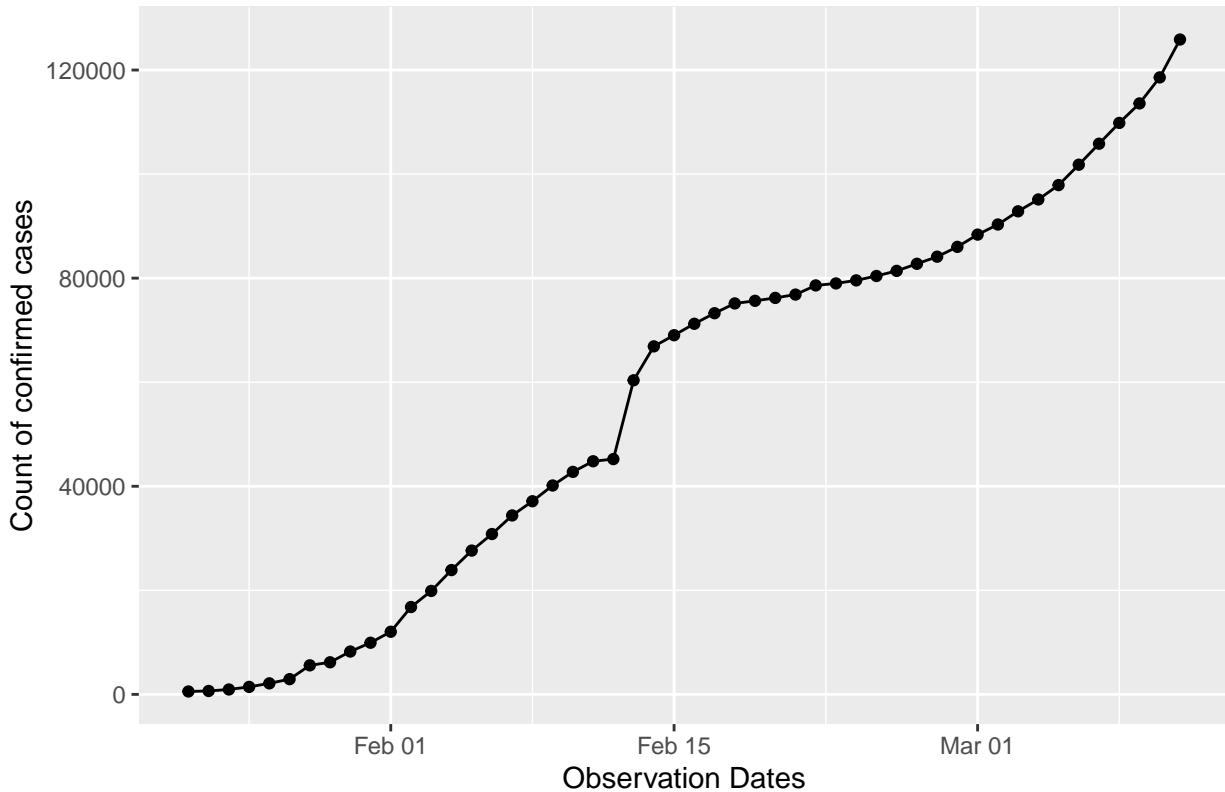


```
# Show number of confirmed cases over time
covid_confirmed <- covid_data %>%
  group_by(ObservationDate) %>%
  summarize(total_confirmed = sum(Confirmed))
head(covid_confirmed %>% arrange(desc(total_confirmed)))
```

```
## # A tibble: 6 x 2
##   ObservationDate total_confirmed
##   <date>                <int>
## 1 2020-03-11            125865
## 2 2020-03-10            118582
## 3 2020-03-09            113582
## 4 2020-03-08            109835
## 5 2020-03-07            105836
## 6 2020-03-06            101800
```

```
# Plot the number of confirmed cases over time
covid_confirmed %>% ggplot(aes(x=ObservationDate, y=total_confirmed, group=1)) +
  geom_point() + geom_line() + labs(x = "Observation Dates", y = "Count of confirmed cases",
  title = "Count of confirmed cases over time")
```

Count of confirmed cases over time

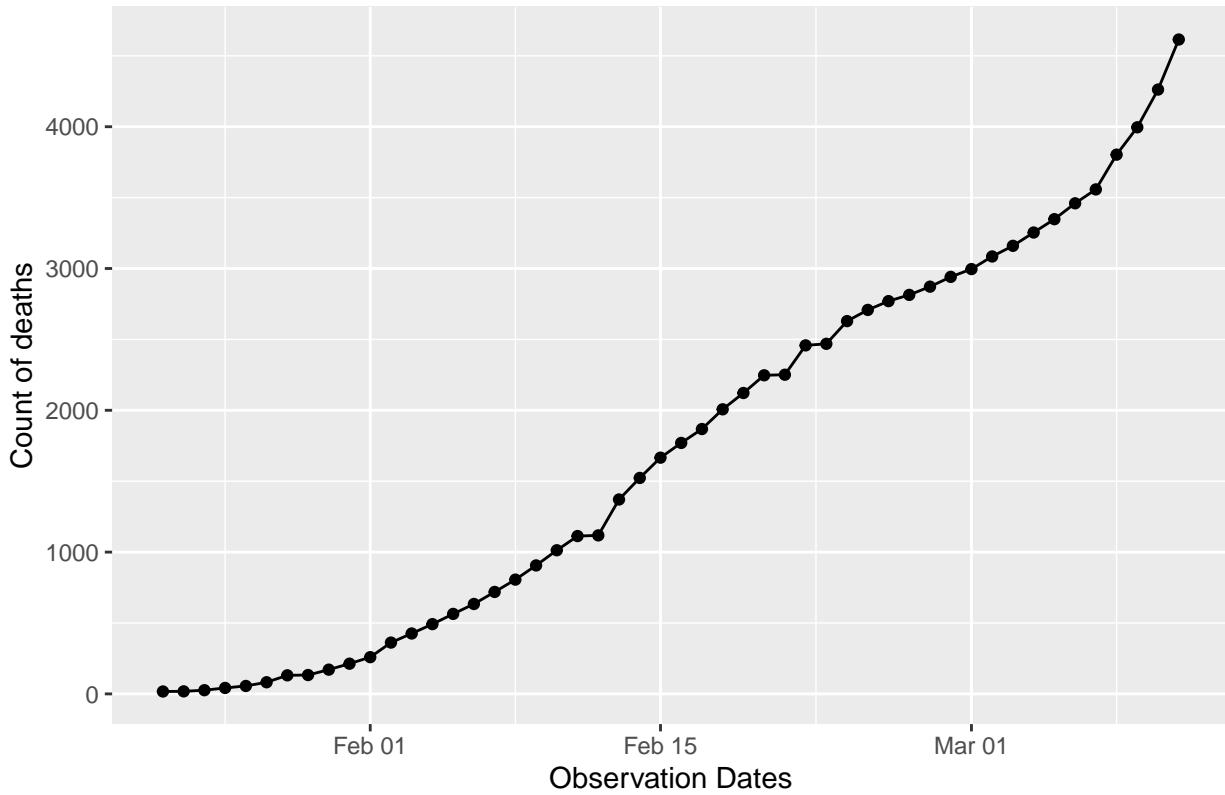


```
# Show number of deaths cases over time
covid_deaths <- covid_data %>%
  group_by(ObservationDate) %>%
  summarize(total_deaths = sum(Deaths))
head(covid_deaths %>% arrange(desc(total_deaths)))

## # A tibble: 6 x 2
##   ObservationDate total_deaths
##   <date>           <int>
## 1 2020-03-11        4615
## 2 2020-03-10        4262
## 3 2020-03-09        3996
## 4 2020-03-08        3803
## 5 2020-03-07        3558
## 6 2020-03-06        3460

# Plot the number of deaths cases over time
covid_deaths %>% ggplot(aes(x=ObservationDate, y=total_deaths, group=1)) +
  geom_point() + geom_line() + labs(x = "Observation Dates", y = "Count of deaths",
  title = "Count of deaths over time")
```

Count of deaths over time



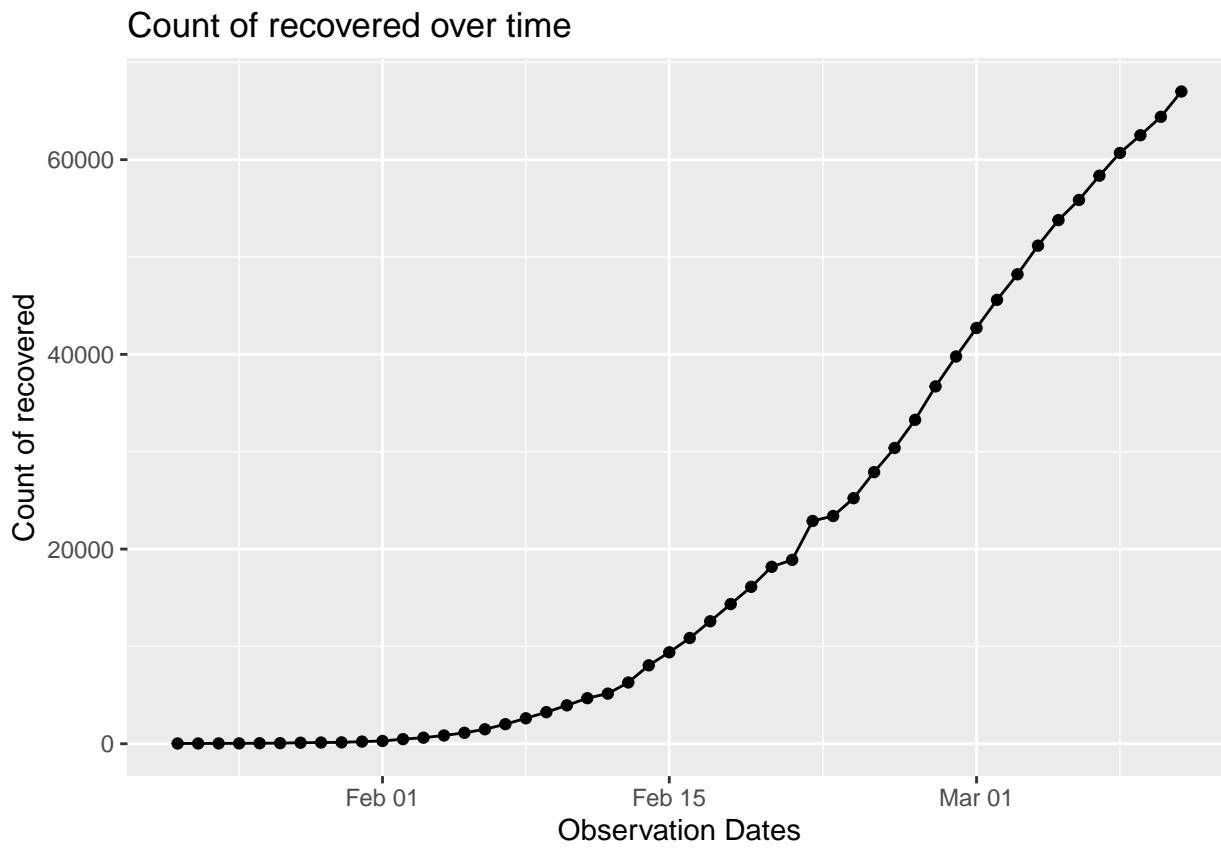
```
# Show number of recovered cases over time
covid_recovered <- covid_data %>%
  group_by(ObservationDate) %>%
  summarize(total_recovered = sum(Recovered))
head(covid_recovered %>% arrange(desc(total_recovered)))
```



```
## # A tibble: 6 x 2
##   ObservationDate total_recovered
##   <date>                <int>
## 1 2020-03-11            67003
## 2 2020-03-10            64404
## 3 2020-03-09            62512
## 4 2020-03-08            60695
## 5 2020-03-07            58359
## 6 2020-03-06            55866
```



```
# Plot the number of recovered cases over time
covid_recovered %>% ggplot(aes(x=ObservationDate, y=total_recovered, group=1)) +
  geom_point() + geom_line() +
  labs(x = "Observation Dates", y = "Count of recovered",
       title = "Count of recovered over time")
```



Let us combine our confirmed, deaths, and recovered.

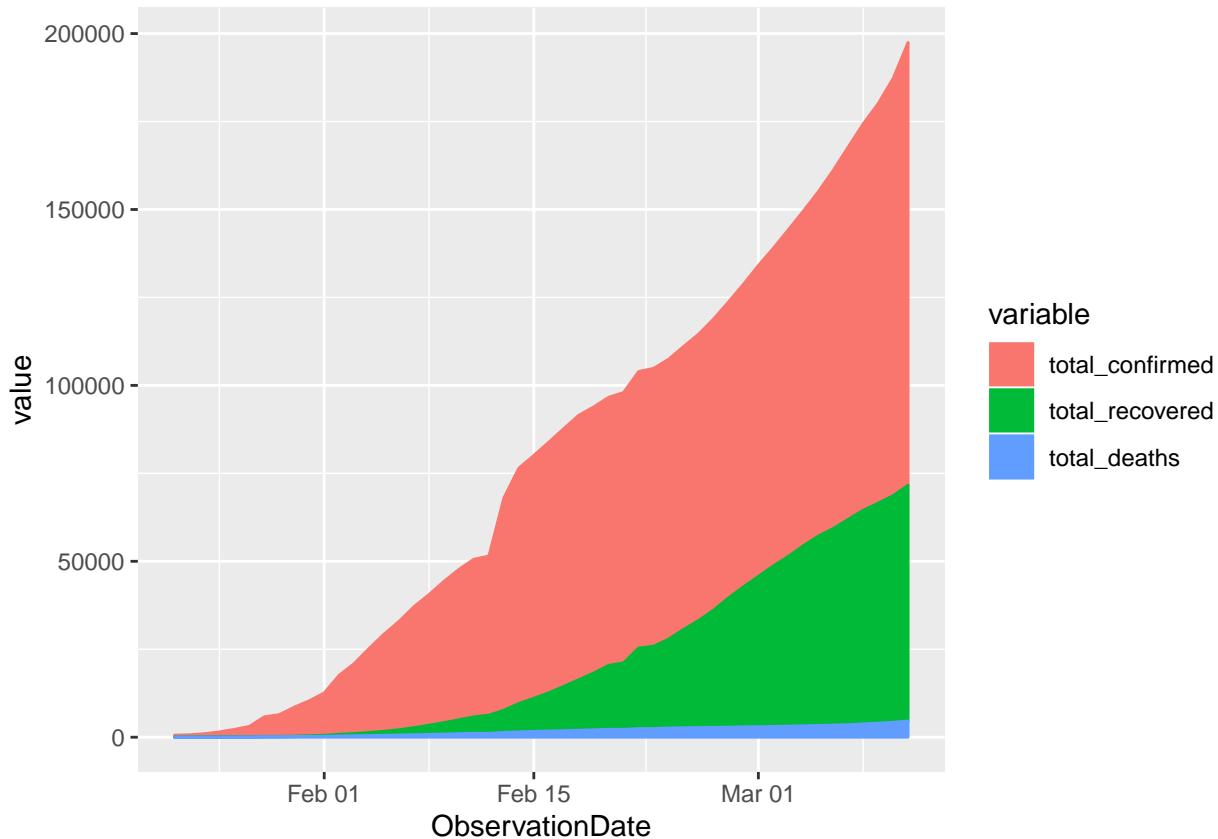
```
# Show number of all cases over time
covid_all <- covid_data %>%
  group_by(ObservationDate) %>%
  summarize(total_confirmed=sum(Confirmed), total_recovered=sum(Recovered), total_deaths=sum(Deaths))
head(covid_all)

## # A tibble: 6 x 4
##   ObservationDate  total_confirmed total_recovered total_deaths
##   <date>                <int>            <int>          <int>
## 1 2020-01-22              555              28            17
## 2 2020-01-23              653              30            18
## 3 2020-01-24              941              36            26
## 4 2020-01-25             1438              39            42
## 5 2020-01-26             2118              52            56
## 6 2020-01-27             2927              61            82

covid_all_melted <- melt(covid_all, id.var='ObservationDate')
head(covid_all_melted)

## # A tibble: 6 x 4
##   ObservationDate variable value
##   <date>        <chr>   <dbl>
## 1 2020-01-22    total_confirmed 555
## 2 2020-01-23    total_confirmed 653
## 3 2020-01-24    total_confirmed 941
## 4 2020-01-25    total_confirmed 1438
## 5 2020-01-26    total_confirmed 2118
## 6 2020-01-27    total_confirmed 2927
```

```
covid_all_melted %>% ggplot(aes(x=ObservationDate, y=value, col=variable)) +
  geom_area(aes(fill=variable))
```



Based on the plots generated we can see that:

- There has been a large spike in the number of confirmed cases between February 8 to 10. The trajectory is also continually increasing.
- The number of deaths increased daily. We can also see the a smaller version of the spike aforementioned.
- On the number of recovered cases, there has been a minimal number from January 22 to February 3 despite the increase in the number of infected cases. Bare in mind that the incubation period is 14-days.

2.1.3 Countries with most cases

Which 10 countries have the most confirmed cases?

```
# Show top 10 Country.Region with cases
covid_country_top10 <- covid_data_recent %>%
  group_by(Country.Region, Province.State, Confirmed) %>%
  summarize(total_confirmed = sum(Confirmed)) %>% arrange(desc(total_confirmed))
covid_country_top10 <- covid_country_top10[1:10,1:3]
covid_country_top10
```

```
## # A tibble: 10 x 3
## # Groups:   Country.Region, Province.State [10]
##   Country.Region Province.State Confirmed
##   <fct>          <fct>           <int>
## 1 Mainland China "Hubei"        67773
```

```

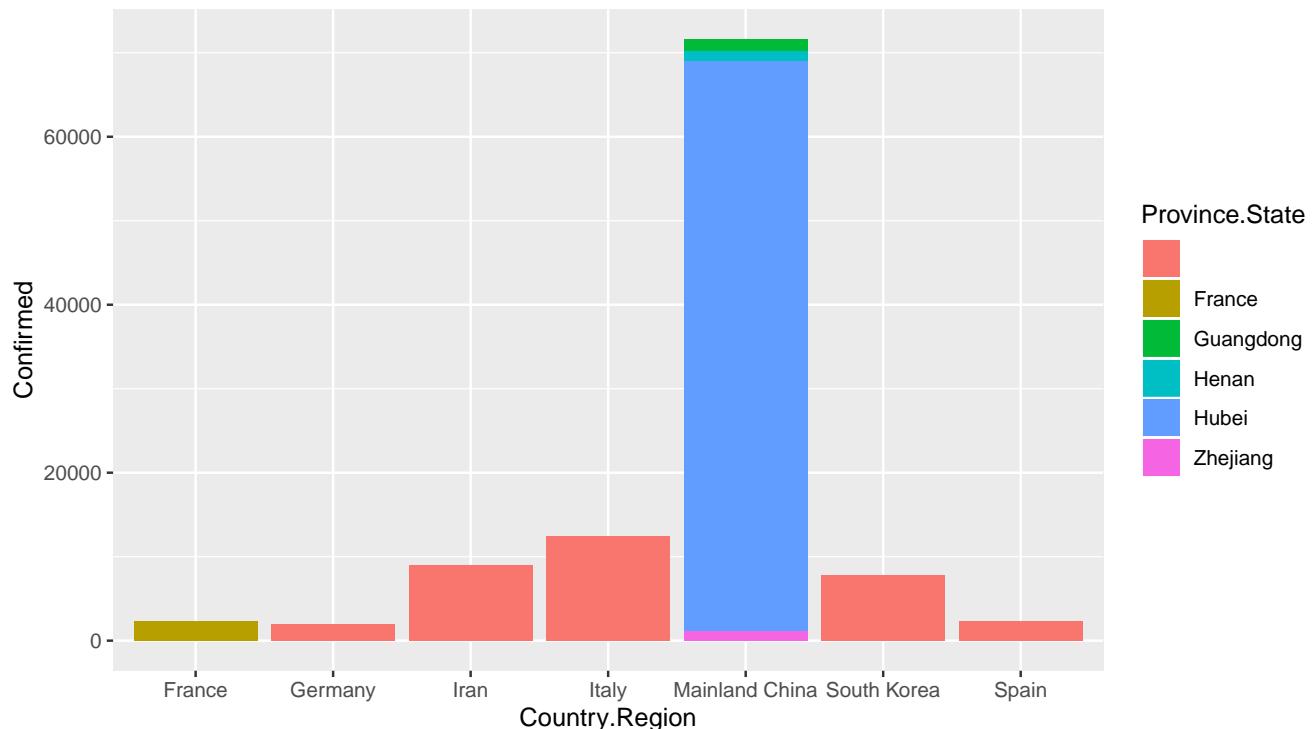
## 2 Italy      ""
## 3 Iran       ""
## 4 South Korea ""
## 5 France     "France"
## 6 Spain      ""
## 7 Germany    ""
## 8 Mainland China "Guangdong"
## 9 Mainland China "Henan"
## 10 Mainland China "Zhejiang"

```

```

# Plot the number of country cases over time
covid_country_top10 %>% ggplot(aes(Country.Region, Confirmed)) +
  geom_bar(stat="identity", aes(fill=Province.State)) +
  theme(legend.position = "right")

```



Majority of the cases in China are from Hubei. Italy, Iran and South Korea also succumbed to the pandemic. France, Germany and Spain also reports few hundreds of cases.

2.1.4 Spatial analysis using ggplot and map_data

Note that the following plot: * It is much better viewed in .html version (uploaded in Github) or upon running the .Rmd file * If both fails, you can still use the PDF file but zoom in on the map to detect the changes.

```

# Load our time series data
covid_data_timeseries <- read_csv(str_c('time_series_covid_19_confirmed.csv'))

```

```

## Parsed with column specification:
## cols(
##   .default = col_double(),
##   `Province/State` = col_character(),
##   `Country/Region` = col_character()
## )

```

```

## See spec(...) for full column specifications.

# Set to 50 daily entries count
n_times <- 50
# Prepare timeseries data, parse latitude longitude data
covid_data_ts_latlong <- covid_data_timeseries
colnames(covid_data_ts_latlong) <- c(colnames(covid_data_ts_latlong)[1:4],
  str_c('D1: ', str_pad(as.character(1:n_times), 2, 'left', '0'),
    str_c(' - ', colnames(covid_data_ts_latlong)[5:(4 + n_times)])))
# Pivot our latitude longitude data for ggplot use
covid_data_ts_latlong_pivot <- covid_data_ts_latlong %>% pivot_longer(names_to = 'Confirmed.Time',
  values_to = 'Confirmed',
  cols = colnames(covid_data_ts_latlong)[5:(4 + n_times)])
head(covid_data_ts_latlong_pivot)

## # A tibble: 6 x 6
##   `Province/State` `Country/Region`  Lat Long Confirmed.Time   Confirmed
##   <chr>           <chr>        <dbl> <dbl> <chr>          <dbl>
## 1 Anhui           Mainland China  31.8  117. D1: 01 - 1/22/20      1
## 2 Anhui           Mainland China  31.8  117. D1: 02 - 1/23/20      9
## 3 Anhui           Mainland China  31.8  117. D1: 03 - 1/24/20     15
## 4 Anhui           Mainland China  31.8  117. D1: 04 - 1/25/20     39
## 5 Anhui           Mainland China  31.8  117. D1: 05 - 1/26/20     60
## 6 Anhui           Mainland China  31.8  117. D1: 06 - 1/27/20     70

# Use ggplot map_world, to display a global map
world <- map_data('world')
ggplot(legend = FALSE) +
  # Plot our map
  geom_polygon(data = world, aes(x = long, y = lat, group = group),
    color = 'black', fill = 'antiquewhite') + xlab('') + ylab('') +
  # Plot points of our confirmed cases using latitude/longitude data
  geom_point(data = covid_data_ts_latlong_pivot, fill = 'red', color = 'black',
    shape = 21, alpha = 0.6,
    aes(x = Long, y = Lat, fill = Confirmed, size = Confirmed)) +
  theme_minimal() +
  # Set the scale from 0 to 30 for the shapes in the map
  scale_size_continuous(range = c(0, 30)) + ggtitle('Occurrences Map - COVID19') +
  theme(text = element_text(size = 12), legend.position = 'top',
    panel.background = element_rect(fill='lightblue', colour='blue')) +
  facet_wrap(~Confirmed.Time, ncol = 2)

## Warning: Removed 188 rows containing missing values (geom_point).

```



2.2 Data Modeling

For this exercise we will use various timeseries forecasting methods such as Naive, HoltWinters, ARIMA and Simple Exponential Smoothing. We will evaluate each of their performance and compare predictions made. Mainly, we will be predicted the confirmed number of cases for the next 50 days since March 11, 2020.

```
# Since the default ts() function is not a good tool to use for daily sampled data,
# - we will use 'zoo' package for irregular time series data. This takes care of-
# indexing for time-series data. Create a daily Date object - helps our work on dates
inds <- seq(as.Date("2020-01-22"), as.Date("2020-03-11"), by = "day")
# We link the time-series with our confirmed cases
covid_confirmed_ts <- zoo(covid_confirmed$total_confirmed, inds)
```

2.2.1 Naive

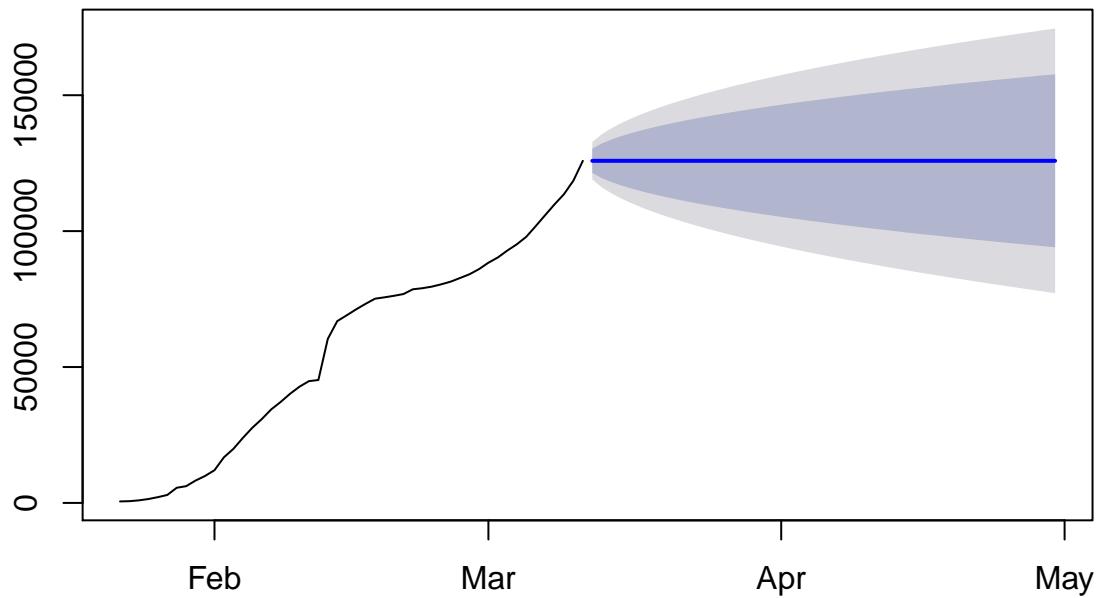
One of the simplest forecasting is to reuse recent observations for predict the next one. We start with this, the naive forecast which can be generated using the naive() function.

```
# Initialize our Naive model. Forecast for the next 50 days
model_naive <- naive(covid_confirmed_ts, h = 50)
naive_forecast_max <- max(model_naive$mean)
naive_forecast_max

## [1] 125865

# The plot though will cause an issue as the x-axis is in days since the epoch (1970-01-01)
# So we need to suppress the auto plotting of this axis and then plot our own
plot(model_naive, xaxt ="n")
Axis(inds, side = 1)
```

Forecasts from Naive method



Using naive forecast, there is a more steady rate and highest number of predicted cases is 1.25865×10^5

2.2.2 Simple Exponential Smoothing

As opposed to naive which just uses the recent observations, exponential smoothing looks at older observations and has several configurable parameters that affect forecasting such as:

- alpha - Value of smoothing parameter for the level (0 - 1)
- lambda - Box-Cox transformation parameter.

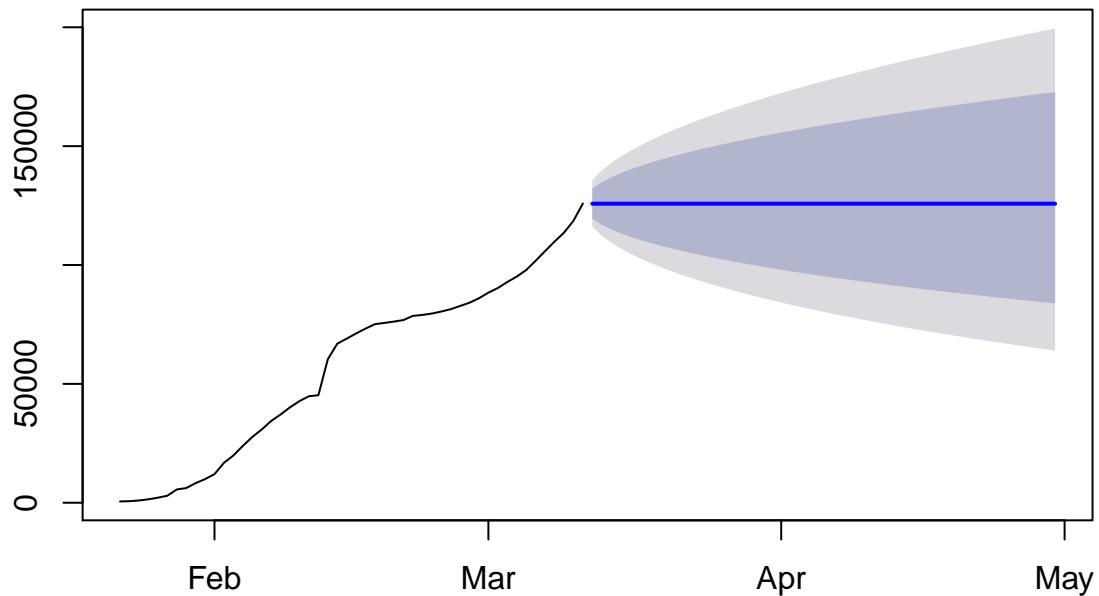
```
# For simple exponential smoothing we also plug in our time series data and predict for the next 50 days
```

```
model_ses <- ses(covid_confirmed_ts, h = 50, alpha = 0.99, lambda="auto")
ses_forecast_max <- max(model_ses$mean)
ses_forecast_max
```

```
## [1] 125791
```

```
plot(model_ses, xaxt ="n")
Axis(inds, side = 1)
```

Forecasts from Simple exponential smoothing



Using Simple Exponential Smooth the maximum predicted number of cases is 1.25791×10^5

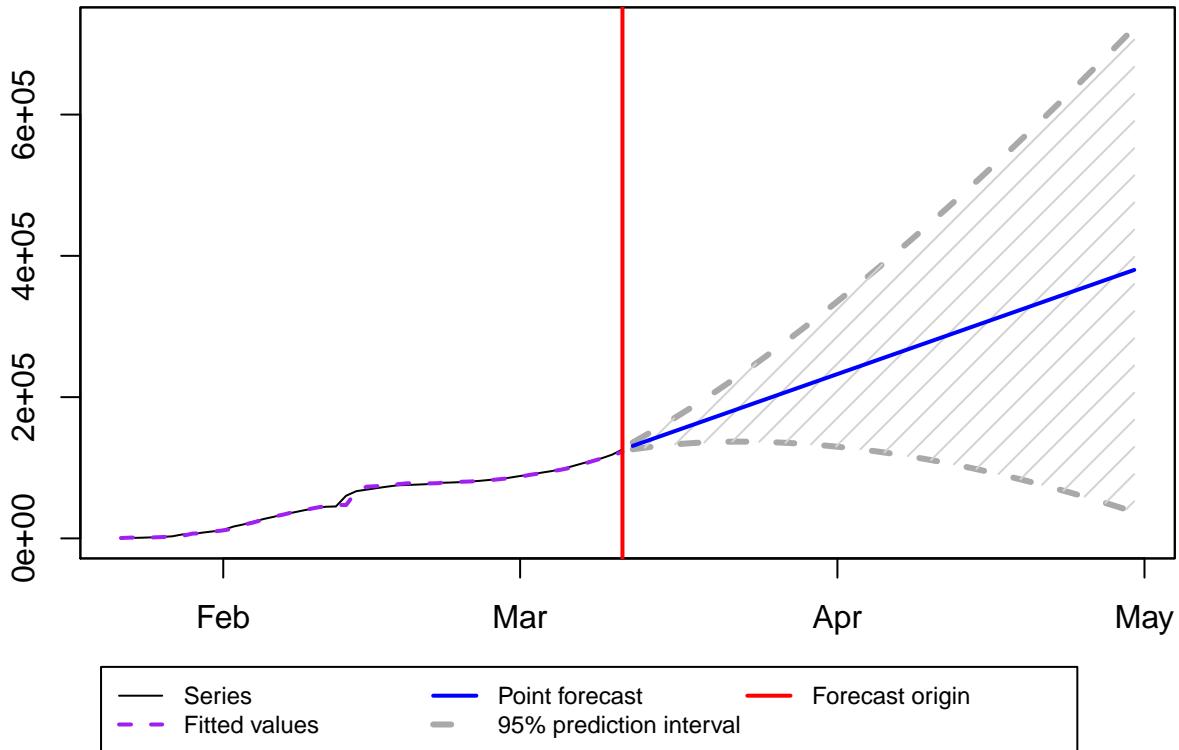
2.2.3 HoltWinters

Using HoltWinters method we can factor in the Trends, Seasonability and Level. These effects factor in questions such as annual sales trends (Black Friday Sales), weekends inactivity, and other recurring effects. It can be used to model complex seasonal and trend patterns.

```
# Initialize our HoltWinters model, the es() function constructs a model and returns forecast and fitted values
# "AAM" denotes HoltWinters model, h is our forecast length, interval allows for prediction intervals
model_AAM <- es(covid_confirmed_ts, "AAM", h=50, interval=TRUE, silent = "none")
```

```
## The provided data is not ts object. Only non-seasonal models are available.
```

ETS(AAN)



```
holtwinters_forecast_max <- max(model_AAM$forecast)
holtwinters_forecast_max
```

```
## [1] 380108.8
```

Based on above forecast, with our current trend the highest number of confirmed cases with me 3.80109×10^5 there is also an evident steady increase.

2.2.4 ARIMA

Auto Regressive Integrated Moving Average models factor in lags and lagged errors. It has 3 terms: p (autoregression), d(moving average), q(difference) The model also describes autocorrelations of the observations.

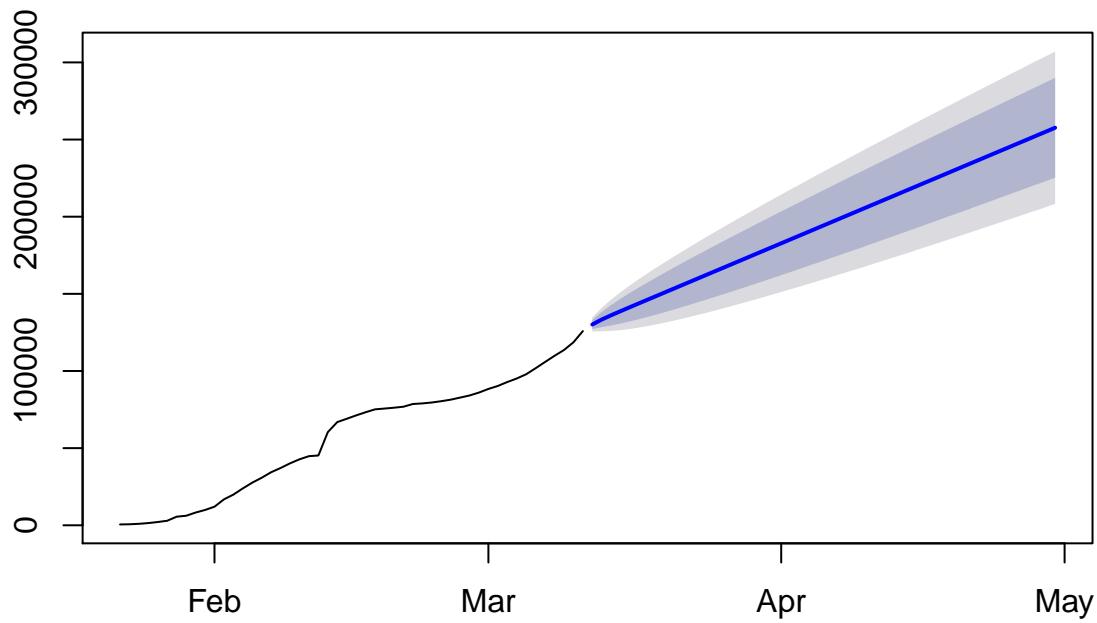
```
# Initialize our ARIMA model. It is straightforward as we only have to plug in our time series
model_arima <- auto.arima(covid_confirmed_ts)
```

```
# Forecast next 50 days for the ARIMA model
forecast_arima <- forecast::forecast(model_arima, h=50)
arima_forecast_max <- max(forecast_arima$mean)
arima_forecast_max
```

```
## [1] 257657.5
```

```
plot(forecast_arima, xaxt = "n")
Axis(inds, side = 1)
```

Forecasts from ARIMA(1,1,0) with drift



Using the ARIMA model this is the forecast 2.57657×10^5 of cases for the next 50 days

3 Results

Now let us tabulate our models' performance and maximum forecast for Confirmed cases.

```
# Look at the Naive model performance and maximum forecast
summary(model_naive)
```

```
## 
## Forecast method: Naive method
## 
## Model Information:
## Call: naive(y = covid_confirmed_ts, h = 50)
## 
## Residual sd: 2431.0251
## 
## Error measures:
##               ME      RMSE      MAE      MPE      MAPE MASE      ACF1
## Training set 2557.347 3511.31 2557.347 9.697016 9.697016     1 0.3318278
## 
## Forecasts:
##      Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 18333      125865 121365.08 130364.9 118982.96 132747.0
## 18334      125865 119501.15 132228.9 116132.32 135597.7
## 18335      125865 118070.90 133659.1 113944.96 137785.0
## 18336      125865 116865.15 134864.8 112100.92 139629.1
```

```

## 18337      125865 115802.86 135927.1 110476.29 141253.7
## 18338      125865 114842.48 136887.5 109007.51 142722.5
## 18339      125865 113959.32 137770.7 107656.83 144073.2
## 18340      125865 113137.29 138592.7 106399.65 145330.4
## 18341      125865 112365.23 139364.8 105218.88 146511.1
## 18342      125865 111634.99 140095.0 104102.08 147627.9
## 18343      125865 110940.44 140789.6 103039.85 148690.1
## 18344      125865 110276.80 141453.2 102024.91 149705.1
## 18345      125865 109640.29 142089.7 101051.45 150678.6
## 18346      125865 109027.82 142702.2 100114.76 151615.2
## 18347      125865 108436.87 143293.1 99210.97 152519.0
## 18348      125865 107865.30 143864.7 98336.84 153393.2
## 18349      125865 107311.34 144418.7 97489.62 154240.4
## 18350      125865 106773.44 144956.6 96666.97 155063.0
## 18351      125865 106250.28 145479.7 95866.88 155863.1
## 18352      125865 105740.73 145989.3 95087.58 156642.4
## 18353      125865 105243.76 146486.2 94327.53 157402.5
## 18354      125865 104758.48 146971.5 93585.37 158144.6
## 18355      125865 104284.12 147445.9 92859.89 158870.1
## 18356      125865 103819.96 147910.0 92150.02 159580.0
## 18357      125865 103365.38 148364.6 91454.80 160275.2
## 18358      125865 102919.80 148810.2 90773.34 160956.7
## 18359      125865 102482.71 149247.3 90104.87 161625.1
## 18360      125865 102053.64 149676.4 89448.66 162281.3
## 18361      125865 101632.16 150097.8 88804.08 162925.9
## 18362      125865 101217.90 150512.1 88170.51 163559.5
## 18363      125865 100810.48 150919.5 87547.42 164182.6
## 18364      125865 100409.58 151320.4 86934.30 164795.7
## 18365      125865 100014.90 151715.1 86330.69 165399.3
## 18366      125865 99626.16 152103.8 85736.15 165993.8
## 18367      125865 99243.09 152486.9 85150.30 166579.7
## 18368      125865 98865.45 152864.5 84572.76 167157.2
## 18369      125865 98493.03 153237.0 84003.18 167726.8
## 18370      125865 98125.60 153604.4 83441.25 168288.7
## 18371      125865 97762.98 153967.0 82886.67 168843.3
## 18372      125865 97404.98 154325.0 82339.15 169390.8
## 18373      125865 97051.42 154678.6 81798.44 169931.6
## 18374      125865 96702.16 155027.8 81264.28 170465.7
## 18375      125865 96357.02 155373.0 80736.44 170993.6
## 18376      125865 96015.88 155714.1 80214.71 171515.3
## 18377      125865 95678.59 156051.4 79698.87 172031.1
## 18378      125865 95345.03 156385.0 79188.73 172541.3
## 18379      125865 95015.07 156714.9 78684.11 173045.9
## 18380      125865 94688.61 157041.4 78184.82 173545.2
## 18381      125865 94365.53 157364.5 77690.72 174039.3
## 18382      125865 94045.73 157684.3 77201.62 174528.4

```

```

# Residual sd: 2431.0251
# Error measures:
#               ME      RMSE      MAE      MPE      MAPE MASE      ACF1
# Training set 2557.347 3511.31 2557.347 9.697016 9.697016     1 0.3318278

# Look at the SES model performance and maximum forecast
summary(model_ses)

```

##

```

## Forecast method: Simple exponential smoothing
##
## Model Information:
## Simple exponential smoothing
##
## Call:
## ses(y = covid_confirmed_ts, h = 50, alpha = 0.99, lambda = "auto")
##
## Box-Cox transformation: lambda= 0.6811
##
## Smoothing parameters:
## alpha = 0.99
##
## Initial states:
## l = 107.2787
##
## sigma: 117.4361
##
##      AIC      AICc      BIC
## 674.1495 674.4048 677.9736
##
## Error measures:
##      ME      RMSE      MAE      MPE      MAPE      MASE      ACF1
## Training set 2530.505 3499.967 2530.536 9.58068 9.586328 0.9895163 0.3540446
##
## Forecasts:
##      Point Forecast      Lo 80      Hi 80      Lo 95      Hi 95
## 18333 125791 119479.90 132204.7 116181.22 135640.8
## 18334 125791 116940.13 134845.0 112338.88 139718.5
## 18335 125791 114997.33 136888.5 109409.91 142883.0
## 18336 125791 113365.60 138620.8 106956.97 145571.5
## 18337 125791 111933.25 140153.8 104809.24 147954.8
## 18338 125791 110642.78 141545.0 102878.74 150121.1
## 18339 125791 109459.96 142828.5 101113.09 152122.6
## 18340 125791 108362.43 144026.7 99478.11 153993.4
## 18341 125791 107334.67 145155.2 97950.02 155757.5
## 18342 125791 106365.34 146225.2 96511.49 157432.2
## 18343 125791 105445.89 147245.4 95149.44 159030.4
## 18344 125791 104569.68 148222.4 93853.70 160562.4
## 18345 125791 103731.42 149161.5 92616.15 162036.3
## 18346 125791 102926.80 150066.9 91430.24 163458.7
## 18347 125791 102152.31 150942.2 90290.55 164834.9
## 18348 125791 101404.96 151790.3 89192.56 166169.5
## 18349 125791 100682.28 152613.9 88132.43 167466.4
## 18350 125791 99982.11 153414.9 87106.89 168728.7
## 18351 125791 99302.61 154195.3 86113.10 169959.4
## 18352 125791 98642.17 154956.6 85148.62 171160.9
## 18353 125791 97999.38 155700.3 84211.29 172335.3
## 18354 125791 97373.00 156427.6 83299.20 173484.5
## 18355 125791 96761.94 157139.6 82410.66 174610.4
## 18356 125791 96165.19 157837.3 81544.16 175714.2
## 18357 125791 95581.89 158521.6 80698.36 176797.5
## 18358 125791 95011.23 159193.3 79872.03 177861.3
## 18359 125791 94452.49 159853.0 79064.08 178906.9
## 18360 125791 93905.02 160501.5 78273.49 179935.2
## 18361 125791 93368.23 161139.3 77499.36 180947.2

```

```

## 18362      125791  92841.57 161767.0  76740.85 181943.6
## 18363      125791  92324.53 162385.2  75997.20 182925.3
## 18364      125791  91816.67 162994.1  75267.70 183892.9
## 18365      125791  91317.55 163594.3  74551.70 184847.1
## 18366      125791  90826.78 164186.2  73848.60 185788.5
## 18367      125791  90343.99 164770.2  73157.84 186717.6
## 18368      125791  89868.86 165346.4  72478.91 187635.0
## 18369      125791  89401.06 165915.4  71811.31 188541.2
## 18370      125791  88940.29 166477.4  71154.61 189436.6
## 18371      125791  88486.30 167032.6  70508.37 190321.6
## 18372      125791  88038.81 167581.3  69872.22 191196.6
## 18373      125791  87597.59 168123.7  69245.77 192062.0
## 18374      125791  87162.42 168660.2  68628.67 192918.2
## 18375      125791  86733.06 169190.8  68020.62 193765.4
## 18376      125791  86309.34 169715.8  67421.28 194604.0
## 18377      125791  85891.05 170235.4  66830.39 195434.2
## 18378      125791  85478.02 170749.7  66247.65 196256.4
## 18379      125791  85070.08 171259.0  65672.82 197070.8
## 18380      125791  84667.07 171763.4  65105.64 197877.6
## 18381      125791  84268.84 172263.0  64545.88 198677.1
## 18382      125791  83875.23 172758.0  63993.32 199469.5

```

```

#      AIC      AICc      BIC
# 674.1495 674.4048 677.9736
#
# Error measures:
#               ME      RMSE      MAE      MPE      MAPE      MASE      ACF1
# Training set 2530.505 3499.967 2530.536 9.58068 9.586328 0.9895163 0.3540446

# Look at the HoltWinters model performance and maximum forecast
summary(model_AAM)

```

```

## Time elapsed: 0.07 seconds
## Model estimated: ETS(AAN)
## Persistence vector g:
##   alpha   beta
## 1.0000 0.3243
## Initial values were optimised.
##
## Loss function type: MSE; Loss function value: 5235844.5326
## Error standard deviation: 2385.61
## Sample size: 50
## Number of estimated parameters: 4
## Number of provided parameters: 1
## Number of degrees of freedom: 46
## Information criteria:
##      AIC      AICc      BIC      BICc
## 923.4458 924.3347 931.0939 932.8326
##
## 95% parametric prediction interval was constructed

```

```

# Loss function type: MSE; Loss function value: 5235844.5326
# Error standard deviation: 2385.61
# Information criteria:
#      AIC      AICc      BIC      BICc
# 923.4458 924.3347 931.0939 932.8326

```

```

# Look at the ARIMA model performance and maximum forecast
summary(model_arima)

## Series: covid_confirmed_ts
## ARIMA(1,1,0) with drift
##
## Coefficients:
##             ar1      drift
##            0.3606  2582.8334
## s.e.    0.1390   498.1569
##
## sigma^2 estimated as 5297239:  log likelihood=-447.9
## AIC=901.81  AICc=902.34  BIC=907.48
##
## Training set error measures:
##               ME      RMSE      MAE      MPE      MAPE      MASE      ACF1
## Training set 21.2316 2231.458 1176.584 -13.61523 16.47069 0.46008 -0.01570852

# AIC=901.81  AICc=902.34  BIC=907.48
#
# Training set error measures:
#               ME      RMSE      MAE      MPE      MAPE      MASE      ACF1
# Training set 21.2316 2231.458 1176.584 -13.61523 16.47069 0.46008 -0.01570852

```

3.1 Model performance

For evaluation, there is a combination of error metrics available: ME, RMSE, MAE, MAPE. Additionally we are introduced to AIC and BIC which are penalized-likelihood criteria to predict best subsets. In general the lower the AIC and BIC, the better the model performance.

If we compare our Naive and SES models, they have similar error metrics, and SES has better AIC/BIC values compared to our HoltWinters and ARIMA model.

On the other hand if we look at HoltWInter and ARIMA, ARIMA has better AIC/BIC values and it performed relatively better compared to our Naive and SES models.

3.2 Model forecast

Forecast model	Maximum forecasted cases
Naive	1.25865×10^5
Simple Exponential Smoothing (SES)	1.2579097×10^5
Holt-Winters	3.8010878×10^5
ARIMA	2.5765749×10^5

We can see that the highest forecast came from HoltWinters model, while the lowest came from SES.

4 Conclusion

In this short exercise we were able to explore the Coronavirus dataset and do a forecast on the number of cases for the next 50 days.

4.1 Summary

- The dplyr functions, including group_by, and summarize worked splendidly for the data set. We have explored various ways of data exploration with dplyr.
- We were able to practice a lot with timeseries data, i.e., ‘zoo’ time series, ts() function, as well as plotting these series.
- We were also able to display a global map using ggplot of Confirmed cases. (Note: view plot as .html, run .Rmd, or zoom in on the PDF page)
- We also observed different forecast behaviors using our four models. Some have linear predictions whereby others were more non-linear.
- We still have not explored trends, seasonality, and recurring patterns since we only have 50-days worth of datapoints.

4.2 Recommendations

- A deeper look on the Province.State and Country.Region to customize the forecast models and parameters.
- Particular attention to recent events, lock-downs, and bans, and how they relate with the forecasting.
- Finally, humanity as a whole is ever-hopeful and optimistic in finding a vaccine. A model forecasting the decline in Confirmed cases as a product of developing a vaccine will be useful for future consideration.