HarvardX Data Science Program

CYO Capstone Project - COVID19 Total Cases Global Forecasting

Le Cong Binh

2021-12-11

Contents

1 Introduction

This is a report of HarvardX Data Science Program - CYO - Capstone Project (PH125.9x).

In this project, We will be predicting the cumulative number of confirmed COVID19 cases in various locations across the world, for future dates. To achieve our goal the project is divided in the following phases: Data exploration; Model identification; Build and improve the model; Limitations and conclusion.

Corona Virus are zoophytic viruses (means transmitted between animals and people).Symptoms include from fever, cough, respiratory symptoms, and breathing difficulties. In severe cases, it can cause pneumonia, severe acute respiratory syndrome (SARS), kidney failure and even death. Corona Virus are also asymptomatic, means a person can be a carrier for the infection but experiences no symptoms.

Dataset:

The data repository for the Novel Corona Virus operated by the Johns Hopkins University Center for Systems Science and Engineering (JHU CSSE). Also, Supported by ESRI Living Atlas Team and the Johns Hopkins University Applied Physics Lab (JHU APL).

It is available here: https://github.com/CSSEGISandData/COVID-19

Before building a model we have to perform Exploratory data analysis (EDA) and select metric for model estimation. RMSE is our metric for this project. It can be calculated by equation:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i}(\hat{y}_i - y_i)^2},$$

where N is size of test-set, y_i is the true rating given by user i day. Root mean squared error (RMSE) is reported in the same units as the outcomes, which makes understanding what is large and what is small enough RMSE more intuitive.

Final RMSE estimation will be performed on the final hold-out validation test set, which we will not use for any other purposes, neither for training model nor for model selection.

2 Data preparation

```r
# loading libraries
if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
if(!require(lubridate)) install.packages("lubridate", repos = "http://cran.us.r-project.org")
if(!require(forecast)) install.packages("forecast", repos = "http://cran.us.r-project.org")
if(!require(knitr)) install.packages("knitr", repos = "http://cran.us.r-project.org")
```

The Code bellow will download data from github

```r
# download data
url = 'https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19
df_confirmed <- read.csv(url)
url = 'https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19
df_death <- read.csv(url)
```

In the original data, a day stands for a variable(column), but they should be placed by row. So we have to get all the days together and create a variable “Date” to store them (per day per row format).

```r
# Reshape data
df_confirmed <- df_confirmed %>%
  gather(key = Date, value = TotalCase,-Province.State,-Country.Region,-Lat,-Long)
df_confirmed$Date <- as.Date(strptime(substr(df_confirmed$Date,2,length(df_confirmed$Date)-1),format="%

df_death <- df_death %>%
  gather(key = Date, value = TotalDeath,-Province.State,-Country.Region,-Lat,-Long)
df_death$Date <- as.Date(strptime(substr(df_death$Date,2,length(df_death$Date)-1),format="%m.%d.%y"))

df <- left_join(df_confirmed,df_death,by=c("Country.Region","Province.State","Lat","Long","Date"))
df <- df %>% group_by(Country.Region) %>% arrange(Date) %>%
  mutate(NewCase = (TotalCase - lag(TotalCase,1))) %>%
  mutate(NewDeath = (TotalDeath - lag(TotalDeath,1)))
```

3 Exploratory data analysis

3.1 First look on dataset

```r
class(df)
```

```
## [1] "grouped_df" "tbl_df"     "tbl"        "data.frame"
```

Class of our dataset is data.frame, we can work with this data class as is. Let's see on first 6 records in the dataset:

Table 1: The last records of dataset

Province.State	Country.Region	Lat	Long	Date	TotalCase	TotalDeath	NewCase	NewDeath
	Venezuela	6.42380	-66.58970	2021-12-10	437113	5229	511	6
	Vietnam	14.05832	108.27720	2021-12-10	1382272	27402	14839	216
	West Bank and Gaza	31.95220	35.23320	2021-12-10	463573	4830	296	4
	Yemen	15.55273	48.51639	2021-12-10	10056	1962	9	5
	Zambia	-13.13390	27.84933	2021-12-10	210724	3668	162	0
	Zimbabwe	-19.01544	29.15486	2021-12-10	155817	4723	0	0

After data wrangling, now we have a dataset with: Province, Country, Lat, Long, Total cases, Total death, New cases, New death and Date. In this project We focus on total case forecast only.

We have 196 country and region; from 2020-01-22 to 2021-12-10.

Now we look about statistics of total case group by country:

```r
summary(df %>% group_by(Country.Region) %>% summarise(TotalCase = sum(TotalCase)) %>% pull(TotalCase))
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 4.300e+01 4.830e+06 4.444e+07 3.516e+08 1.641e+08 1.390e+10
```

3.2 Country and Date

Number of unique country in dataset: 196; unique date in dataset: 689.

Plot of total case by date:

```r
# Total case by date
df %>% group_by (Date) %>% summarize(TotalCases = sum(TotalCase)) %>%
    ggplot(aes(Date,TotalCases)) +
    geom_point() +
    geom_smooth()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

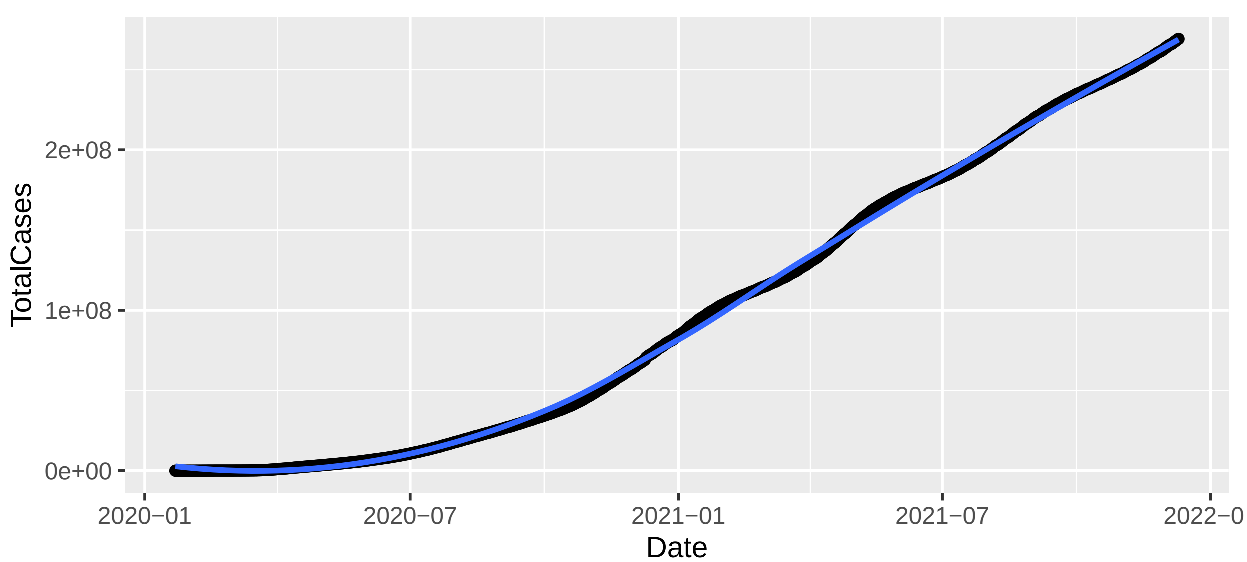


Figure 1: Distibution of total cases

The trend of total case is fit with exponential smoothing. So, We should be use some algorithm from forecast package.

3.3 Country and total cases

First, let's look on the top-10 and bottom 10 movies:

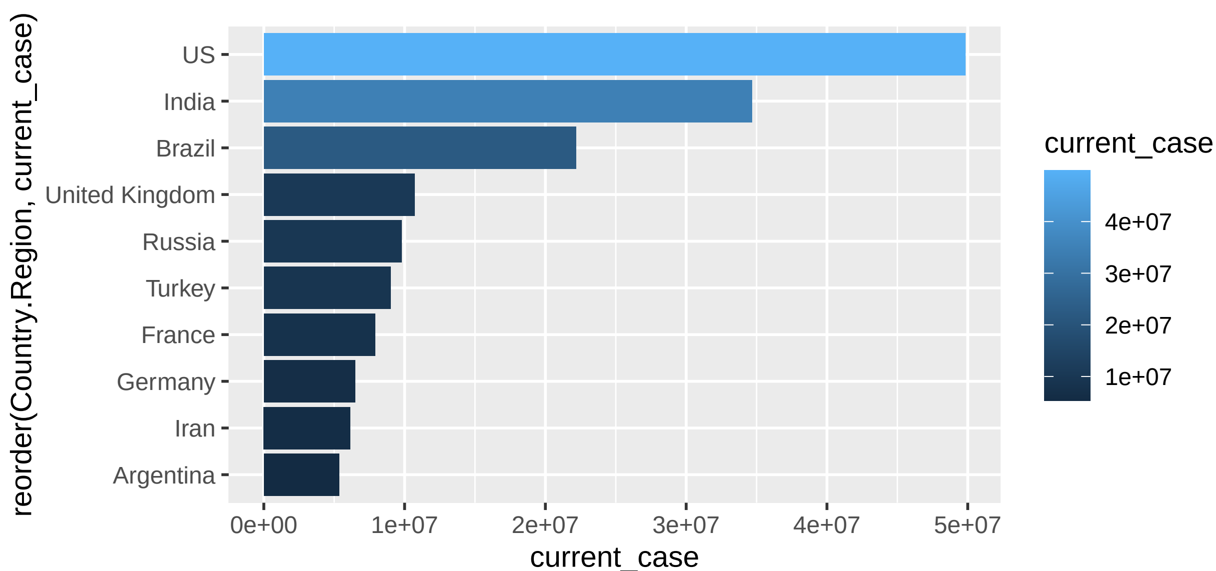
```r
# Top Country overview
knitr::kable(df %>% group_by(Country.Region) %>%
  summarise(current_case = max(TotalCase)) %>%
```

```
  arrange(desc(current_case)) %>%
  ungroup() %>%
  head(10),
  caption = "Top country by case")
```

Table 2: Top country by case

Country.Region	current_case
US	49833439
India	34674643
Brazil	22177059
United Kingdom	10719165
Russia	9782723
Turkey	9004938
France	7912490
Germany	6496142
Iran	6150843
Argentina	5354440

```
df %>% group_by(Country.Region) %>%
  summarise(current_case = max(TotalCase)) %>%
  arrange(desc(current_case)) %>% head(10) %>%
  ggplot(aes(reorder(Country.Region, current_case), current_case, fill = current_case)) +
  geom_bar(stat = "identity") +
  coord_flip()
```




```
knitr::kable(df %>% group_by(Country.Region) %>%
  summarise(current_case = max(TotalCase)) %>%
  arrange(current_case) %>%
  ungroup() %>%
  head(10),
  caption = "Bottom country by case")
```

Table 3: Bottom country by case

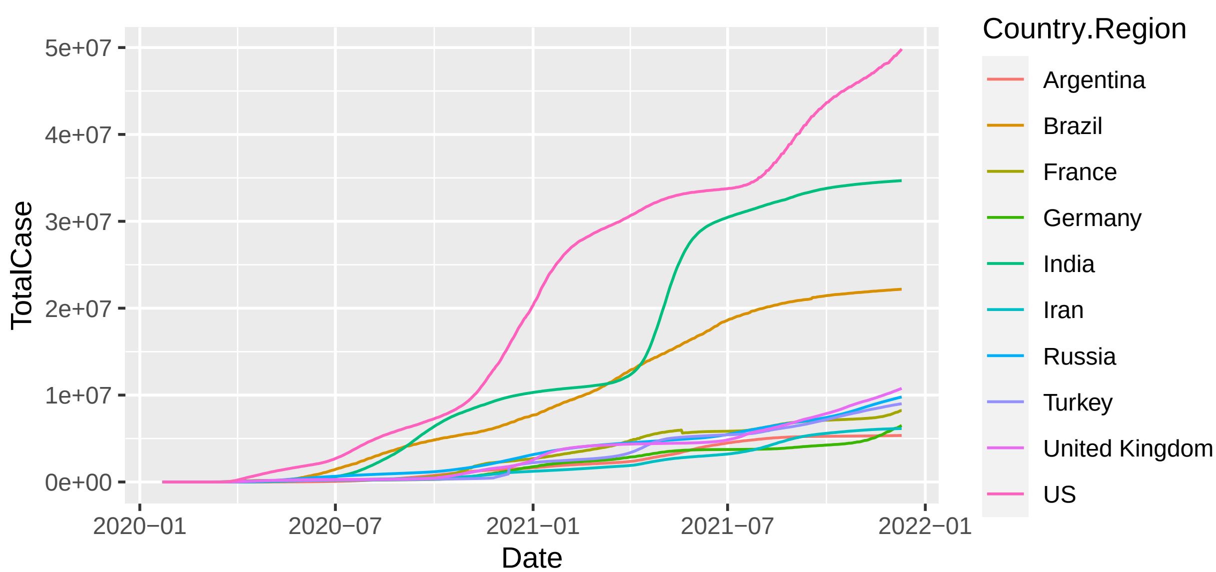
Country.Region	current_case
Micronesia	1
Tonga	1
Kiribati	2
Samoa	3
Marshall Islands	4
Vanuatu	6
Palau	8
MS Zaandam	9
Solomon Islands	20
Holy See	27

Looking on the tables, we see that the top-10 and bottom-10 have big different. The top country will be more effect to prediction results. This give us a ideal to predict on each of country.

We will look the trend of total case in top country

```
# Top country trend
top_country <- df %>% group_by(Country.Region) %>%
  summarise(current_case = max(TotalCase)) %>%
  arrange(desc(current_case)) %>%
  head(10)

inner_join(top_country, df %>% group_by(Date, Country.Region) %>% summarise(TotalCase = sum(TotalCase),
```



3.4 Summary

The data is clean. Because the data set is limitation of feature so we will forecast using various methods, namely: naive approach, caret package (glm, knn, rf, ...) and forecast package (Holt linear, exponential smoothing, ARIMA,...)

We don't use strong tree base algorithm like Light GBM, XGB,.. or deep learning in this project.

4 Methods of model building

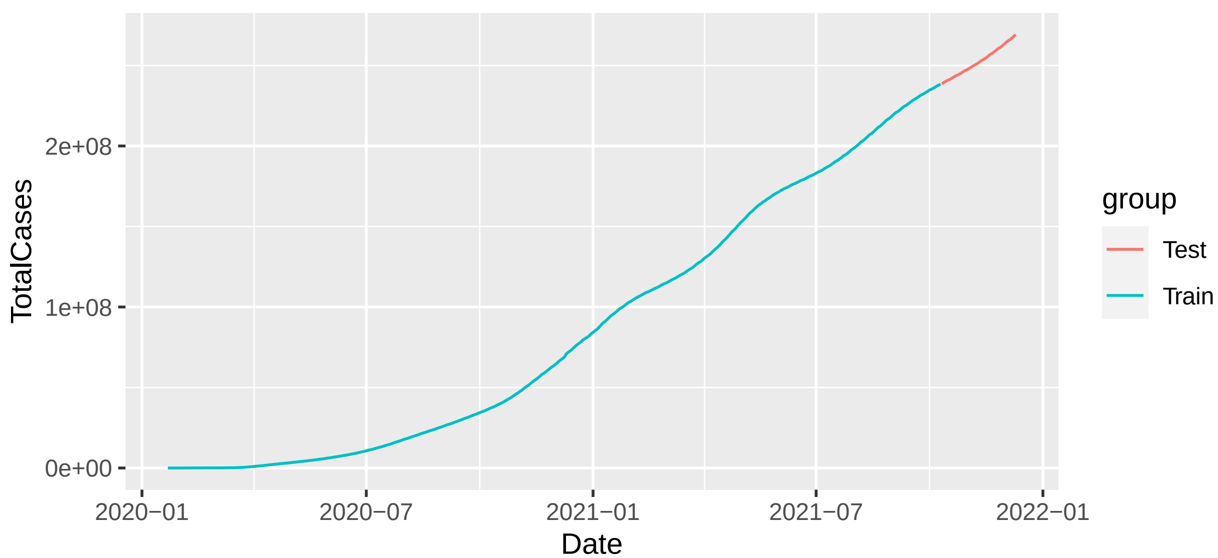
In this chapter we will try different approaches to build prediction model.

4.1 Validation technique

First, we need to create training and validation sets to train and validate our models. I will take the last 2 months as the validation data.

```
# Train. test split
date_index <- max(df$Date) %m-% months(2)
train <- df %>% filter(Date <= date_index) %>% group_by(Date) %>% summarize(TotalCases = sum(TotalCase))
test <- df %>% filter(Date > date_index) %>% group_by(Date) %>% summarize(TotalCases = sum(TotalCase))

df %>% group_by (Date) %>% summarize(TotalCases = sum(TotalCase)) %>%
  mutate(group = ifelse(Date <= date_index,'Train','Test')) %>%
  ggplot(aes(Date,TotalCases,col = group)) + geom_line()
```



Check datasets dimensions: dimensions of dataset are 628, 2 and dimensions of validation dataset are 61, 2 .

Function of the RMSE is defined by code:

```
# function to estimate RMSE
RMSE <- function(true_ratings, predicted_ratings){
  sqrt(mean((true_ratings - predicted_ratings)^2))
  }
```

4.2 Linear model

In order to have some baseline, we will implement the simplest model. Assume, that the total case depend on Date only

```r
# Base line model
fit <- lm(TotalCases ~ Date, data = train)
y_hat <- predict(fit,test)
rmse <- RMSE(test$TotalCases,y_hat)
```

With just predicting the linear model we have RMSE $= 2.3552899 \times 10^7$. And save RMSE to the list:

```r
rmse_results <- tibble(method = "lm", RMSE = rmse)
```

4.3 Caret model selection

We will scan some basic model using caret package and save predict results

```r
# Caret model scan
set.seed(1, sample.kind="Rejection")
models <- c("glm", "svmLinear", "knn", "rf")
fits <- lapply(models, function(model){
    print(model)
    train(TotalCases~Date, method = model, data =train)
    #y_hat <- predict(fit, newdata = test)
    #y_hat[is.na(y_hat)] = 0
})
```

```
## [1] "glm"
## [1] "svmLinear"
## [1] "knn"
## [1] "rf"
```

```r
# save predict data
pred <- sapply(fits, function(object)
    predict(object, newdata = test))

pred[is.na(pred)] = 0

#Let show predict result on test set

caret_pred <- test %>% select(Date, TotalCases)
caret_pred <- cbind(caret_pred, pred)
colnames(caret_pred) <- c("Date","TotalCases",models)
knitr::kable(caret_pred %>% head(10), caption = "Top of result in caret model")
```

Table 4: Top of result in caret model

Date	TotalCases	glm	svmLinear	knn	rf
2021-10-11	238716090	216618667	221685792	237606446	237699546
2021-10-12	239149400	217036537	222126994	237606446	237699546
2021-10-13	239612736	217454407	222568196	237606446	237699546
2021-10-14	240057301	217872278	223009398	237606446	237699546
2021-10-15	240514378	218290148	223450599	237606446	237699546
2021-10-16	240851652	218708018	223891801	237606446	237699546

Date	TotalCases	glm	svmLinear	knn	rf
2021-10-17	241164937	219125888	224333003	237606446	237699546
2021-10-18	241579698	219543759	224774205	237606446	237699546
2021-10-19	242021852	219961629	225215406	237606446	237699546
2021-10-20	242493057	220379499	225656608	237606446	237699546

```
# Save RMSE result
rmse <- sapply(seq(1,length(models)), function(i) sqrt(mean((pred[,i] - test$TotalCases)^2)))
names(rmse) <- models

for (model in models){
    rmse_results <- bind_rows(rmse_results, tibble(method=model, RMSE = rmse[model] ))
}

knitr::kable(rmse_results, caption = "RMSE of caret model")
```

Table 5: RMSE of caret model

method	RMSE
lm	23552899
glm	23552899
svmLinear	17774485
knn	17460251
rf	17380090

The smallest RMSE is c(rf = 17380089.7956975) with algorithm rf It's seems very high. We will try with some algorithm in forecast package

4.4 Forecast model selection

```
library(forecast)
forcast_model <- c("arima","ets","bats")
set.seed(2021)
forcast_pred <- sapply(forcast_model, function(model){
    y = train$TotalCases
    if (model == "arima"){
        fit <- auto.arima(y)
    }
    else if (model == "ets"){
        fit <- ets(y)
    }
    else{
        fit <- tbats(y)
    }
    y_hat <- forecast(fit, h=nrow(test)) %>% .$mean
    #rmse <- sqrt(mean((y_hat - test$TotalCases)^2))
})
pred_result <- cbind(caret_pred,forcast_pred)
```

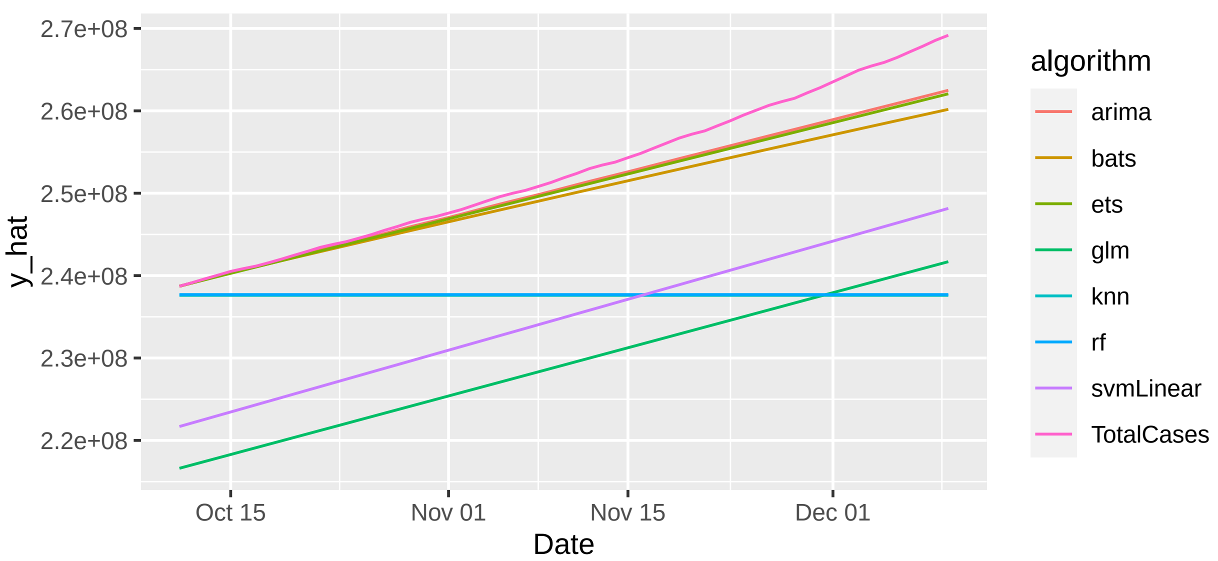
```
rmse_results <- sapply(3:ncol(pred_result), function(i){sqrt(mean((pred_result[,i] - pred_result$TotalCa
names(rmse_results) <- colnames(pred_result)[3:ncol(pred_result)]
knitr::kable(tibble(Algorithm = names(rmse_results), RMSE=unlist(rmse_results)) %>% arrange(RMSE), capti
```

Table 6: RMSE includes forecast model

Algorithm	RMSE
arima	2833689
ets	3078612
bats	4013273
rf	17380090
knn	17460251
svmLinear	17774485
glm	23552899

4.5 Model Selection

```
# Model compare
pred_result %>%
    pivot_longer(!c('Date'), names_to = "algorithm", values_to = "y_hat") %>%
    ggplot(aes(Date,y_hat, col = algorithm)) +
    geom_line()
```



This result confirm the ideal the best model is exponential smooth. Because we don't have many parameter of est to tuning, we will try est for each of country to final tuning.

4.6 Predict by each of country

```
#Fine tuning for est
## Predict by Country
```

```r
train_country <- df %>% filter(Date <= date_index) %>%
    group_by(Country.Region,Date) %>%
    summarize(TotalCases = sum(TotalCase),.groups = 'drop')
#train_country

pred_country <- sapply(unique(train_country$Country.Region), function(country){
    # train with country have covid already
    y <- train_country %>% filter(Country.Region == country & TotalCases > 0) %>% pull(TotalCases)
    if (length(y) ==0){
        y_hat = 0
    } else if (length(y) < 100){
        y_hat = max(y)
    } else {
        fit <- ets(y)
        y_hat <- forecast(fit, h=nrow(test)) %>% .$mean
    }
    y_hat
})

df_pred_country <- pred_country %>% as_tibble() %>%
    mutate(Date = test$Date) %>%
    pivot_longer(!c('Date'), names_to = "Country.Region", values_to = "y_hat")

y_hat <- df_pred_country %>% group_by(Date) %>% summarize(y_hat = sum(y_hat)) %>% pull(y_hat)
#sqrt(mean((y_hat - test$TotalCases)^2))
```

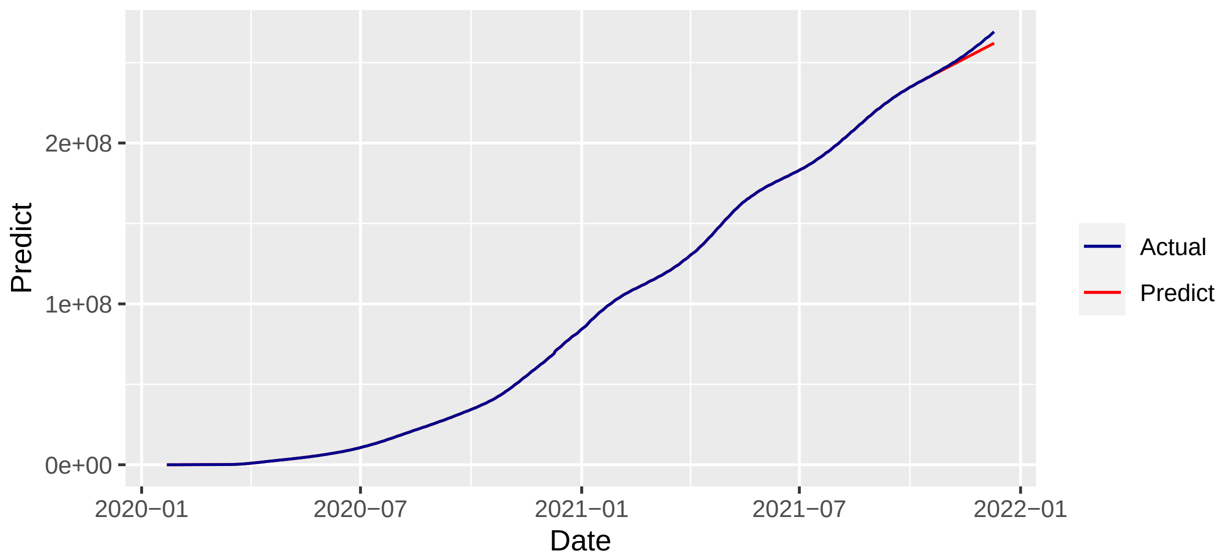
Base on country the RMSE is 4.0897509×10^6 It's not better than predict by all country.

4.7 Final Result

In a week training model We saw the Auto ARIMA and est have the similar RMSE. But all most of case the RMSE score of est algorithm is better when we update data daily. So the final model is est in forecast package for all country.

4.7.1 Recheck model on train / test set

```r
y_hat <- train %>% pull(TotalCases) %>% ets() %>% forecast(nrow(test))
rbind(train %>% mutate(Actual = TotalCases, Predict = TotalCases) %>% select(Date, Actual, Predict),
test %>% mutate(Actual = TotalCases, Predict = y_hat$mean) %>% select(Date, Actual, Predict)) %>%
  ggplot(aes(x = Date)) +
  geom_line(aes(y = Predict, col = 'Predict')) +
  geom_line(aes(y = Actual, col = 'Actual'))+
  scale_color_manual(name = "", values = c("Actual" = "darkblue", "Predict" = "red"))
```



4.7.2 Final prediction for next month

The model predict lower than actual but the chart is smoothly. So we have some ideal to choose the range to predict in the future (in this case we will choose the higher band of final forecast). Let's rerun model for all data to forecast the total case in next month

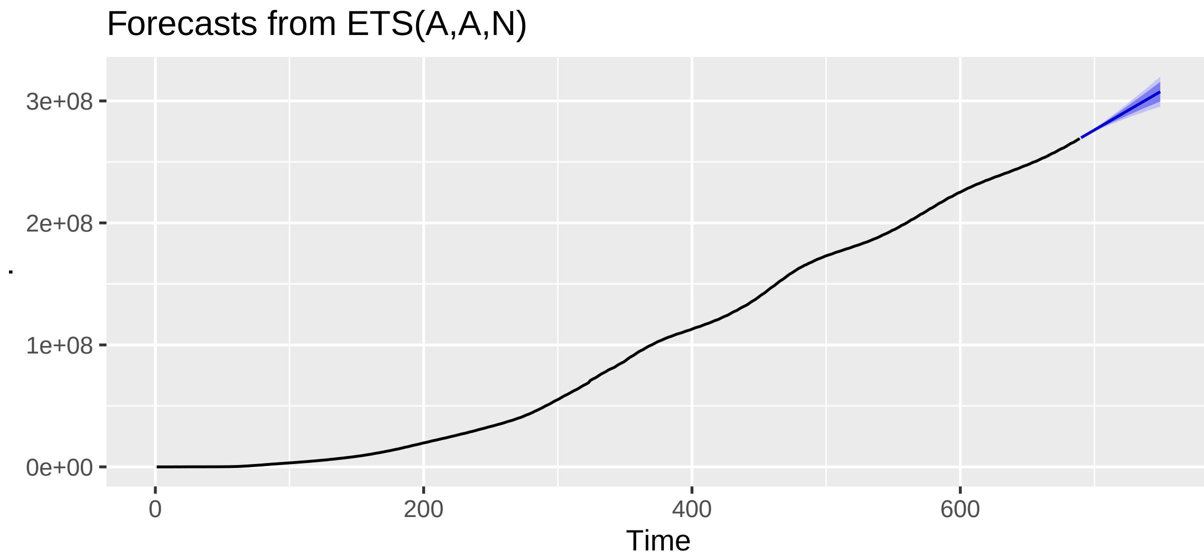
```
finalResult <- df %>% group_by(Date) %>% summarize(TotalCases = sum(TotalCase)) %>%
  pull(TotalCases) %>%
  ets() %>% forecast(10) %>% summary() %>% as_tibble() %>%
  mutate(Date = max(df$Date))

finalResult$Date <- seq(max(df$Date)+1,max(df$Date)+10,by="days")
knitr::kable(finalResult %>% select('Date','Point Forecast','Lo 80','Hi 80','Lo 95','Hi 95'), caption =
```

Table 7: Next 10 days global total cases forecast

Date	Point Forecast	Lo 80	Hi 80	Lo 95	Hi 95
2021-12-11	269807927	269705042	269910812	269650578	269965276
2021-12-12	270447558	270281332	270613784	270193338	270701778
2021-12-13	271087189	270857686	271316693	270736194	271438184
2021-12-14	271726820	271431455	272022185	271275098	272178543
2021-12-15	272366451	272001937	272730965	271808975	272923927
2021-12-16	273006082	272568939	273443225	272337529	273674635
2021-12-17	273645713	273132445	274158981	272860737	274430689
2021-12-18	274285344	273692509	274878179	273378681	275192007
2021-12-19	274924975	274249213	275600737	273891486	275958464
2021-12-20	275564606	274802647	276326565	274399290	276729921

```
df %>% group_by(Date) %>% summarize(TotalCases = sum(TotalCase)) %>%
  pull(TotalCases) %>%
  ets() %>% forecast(60) %>%
  autoplot()
```



5 Conclusion

In this project we have built a model to predict global total case of corona virus. We scan many basic model and found that the best algorithm is exponential smooth. This dataset don't have many feature, that need complex machine learning algorithm like XGB, Light GBM or Cat-boots or Deep learning model.

Possible future development of the model can be:

- Try some booting tree with lag and average feature from time series
- Use deep learning algorithms: LSTM or basic network;

This project only shows my skill about Data Visualization, Data Wrangling, Data Modeling that I learnt in this program. It also gives the model to define the trend of covid 19 in the world. To predict Corona Pandemic in fact We need many job to do: Additional reference data: vaccinate rate, population per country, the policy about tourist, transportation in each country, . . . Which need much time to complete. So, this is the limitation of project.

6 Literature

1. Rafael A. Irizarry, Introduction to Data Science
2. HarvardX Data Science Program