Machine Learning Engineer Nanodegree
Project Report

COVID-19 in Spain

Juan Luis Ramírez

April 2020

Content

1. Definition

1.1. Project Overview

COVID-19 or popularly known as “coronavirus” is a disease caused by SARS-CoV-2 virus. Its first appearance was in Wuhan (Hubei), a city in the south of China, in late 2019. With a high infectious rate; in the moment this document is being written 2.7M people around the world have been infected, 188.000 deceased and 738.000 have recovered from the virus. In this moment all the world is paralyzed because of this reason.

In the middle of all of this we need to use all the tool we have available and all our effort to help each others. Because of this in this project we are focusing on Spanish status regarding Covid-19. The idea is to give clarity to the bunch of data we are exposed daily.

To aim this problem we are developing a online real-time web dashboard to show official data but also to compute some daily metrics that can be interesting to understand the status of the pandemic in every moment and each community in Spain.

The next step is to apply machine learning models and deep learning approaches to try to understand how the pandemic is growing or not, how the official actions against the pandemic could have stopped the increase of cases.

During the project we will also learn how to deploy the application and in future blog articles will focus on details of development of a project like this.

2. Creating a real time dashboard

2.1. Getting and processing data

Starting our project, to be in context, we need to know how data is officially released in Spain; everyday at 11 a.m. the technical board created for this creased offer a press conference to announce last day's data. At the same time an official website from the Spanish gobern release this data.

DATADISTA is a Spanish company focus on data-based journalism; they have managed to create a repository in GitHub that contains all important data concerning Covid-19 in Spain, following daily updates. Thanks to this repository we are able to develop lot of powerful solutions to understand how the pandemic is evolving in the spanish territory.

Using the data they provide we developed a first Jupyter Notebook for Exploratory Data Analysis, this notebook is available in the repository of the project following this link. So for more detailed analytics and extended explanation everybody has access to the notebooks.

Basically and summarizing, the relevant task performed over the dataset were:

1. Drop "Hospitalizados" and "UCI" columns: these columns reference people in hospital and people in ICU. The reason to drop it from the dataset is the different criteria depending on the community. To be clear: some communities give this daily data as accumulated totals but some others refers only to people still today in.

   This difference of criteria make these columns useless as we cannot compare the number of people in hospital between Madrid and Cataluña for example.

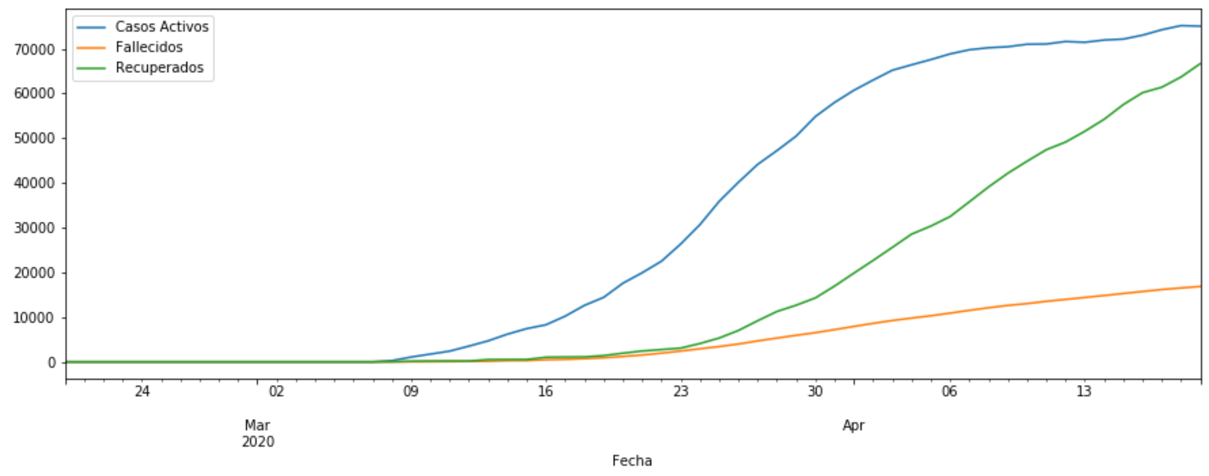
2. Remove rows with zero cases; as we are working with a dataset that contains data from each community since the beginning of the pandemic we have lot of rows with 0 cases at the beginning. In order to save memory and avoid future problems we drop them.
3. Create a column for active cases: more than total cases a metric with the cases that are active today is more defining. We calculate as "cases" - "deceased" - "recovered" daily.

In this point the EDA breaks in two paths: first we analyze the Spain's situation overall, then the most affected communities.

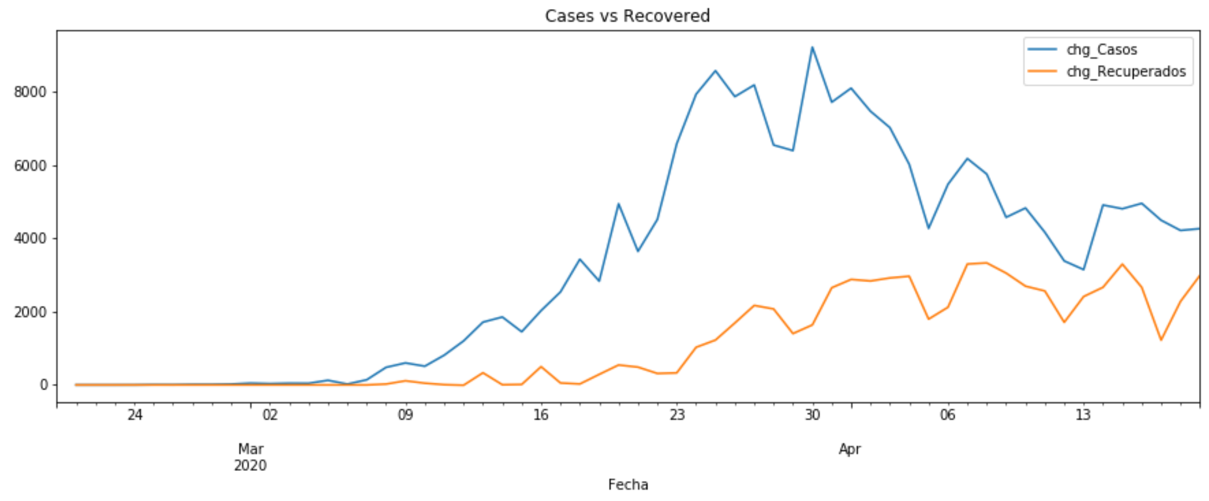
2.1.1. Spain aggregate analysis

First of all, let's group our dataset by date with a sum of columns. So we have daily data in the whole country without community distinction.

Then, for the relevant columns (Cases, Deceased, Recovered and Active), we will calculate daily increment, daily percentage increment and vs 5 days percentage increment. These metrics will help us understand the evolution of the pandemic, the daily increments of every metrics are more relevant than absolute accumulated digit.

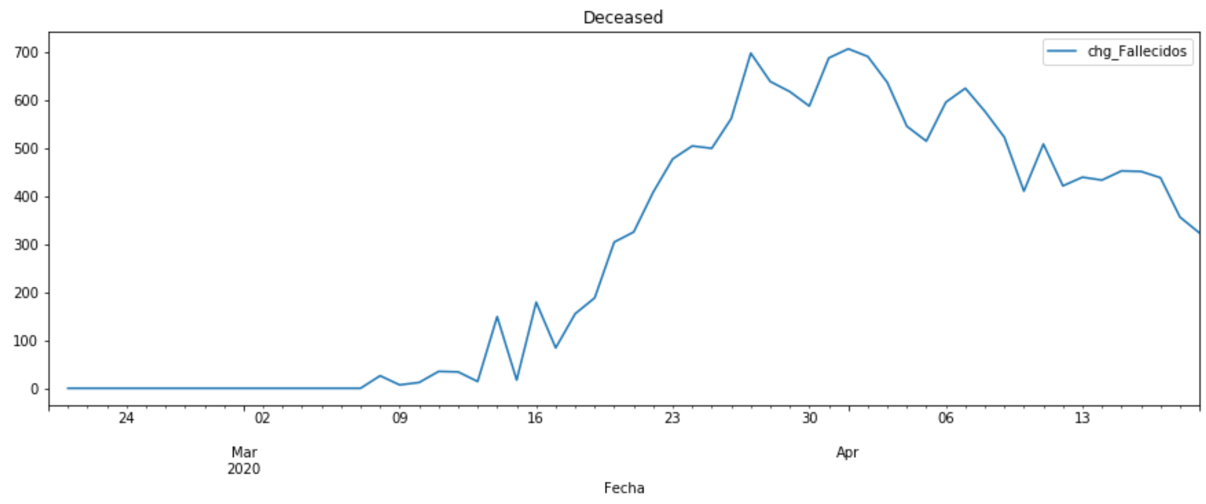


This is the result of plotting the accumulated values, we can see the curve flattening at mid-April. But going deeper if we take a look at the plot of daily increments.



We can clearly see that daily recovered people follows more and less the same pattern than daily cases. We can imagine that the reason is that an infected person need some time to recover from the illness.

If we take a look at deceased data we can see it follows the same pattern.



Some interesting points about this plots:

- New cases started decreasing about 30th March, the strict lockdown started on 29th.
- The first "soft" lockdown started 14th March, taking in mind that the illness can take up to 10-15 days to show effects, we can say that the decreasing is also thanks to this lockdown.

Take into account that we are not going to make a "today" analysis because the pandemic is changing daily. The idea of this project is to find some patterns to understand the pandemic better, for daily metrics we have a real time dashboard.

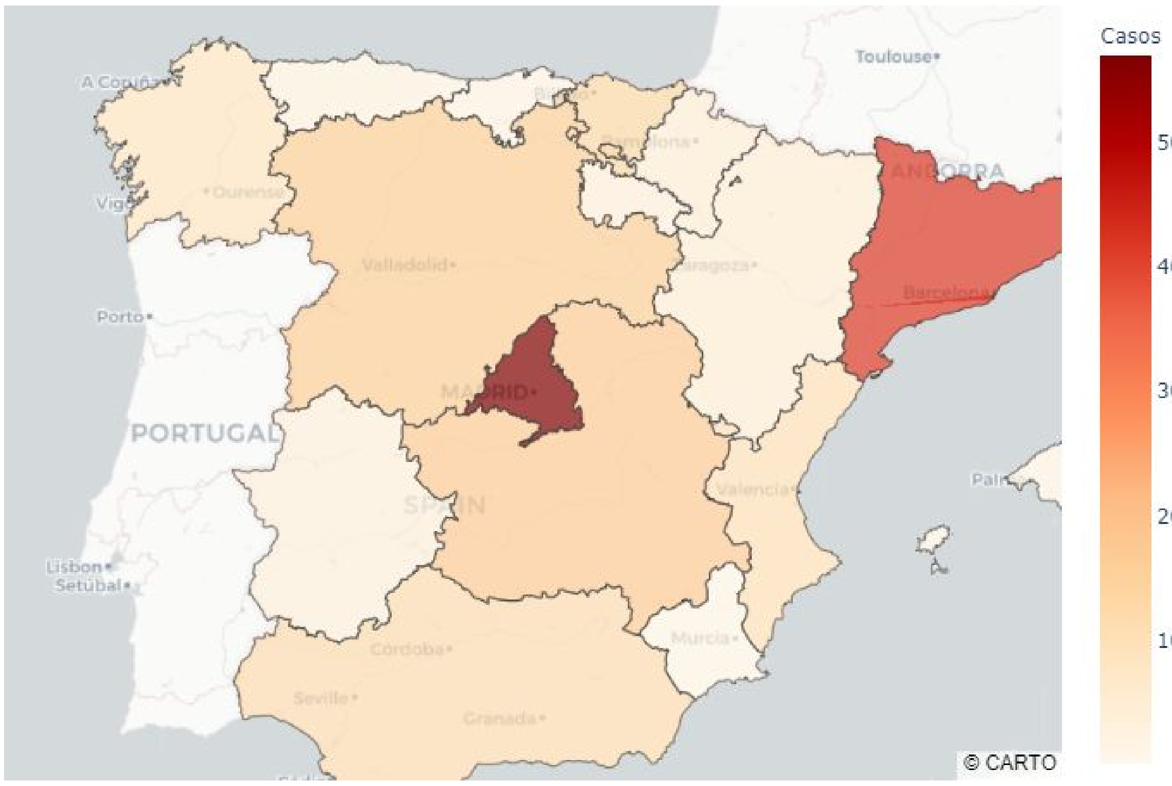
For detailed metrics you can also give a look at the EDA notebook we referred before.

2.1.2. Community analysis

What we want first is to have a map, it is the easiest way to follow the pandemic status all over Spain. For that we are using Choropleth Mapbox by Plotly. The first thing is to have a GeoJSON (a special JSON designed for geographical data) with the information of Spanish communities. Thanks to Albert del Amor for the one he is sharing.

Due to the technology used in the Choropleth Mapbox if you are reading the Jupyter Notebook in Github the maps is not shown. Use nbviewer.

The result (code available in the repository and in the notebook) is:



This is a density map for total cases, in the web dashboard we are going to make some other metrics available (some requested by some users of the website):

- Active cases
- Total cases
- Deceased
- Recovered
- Active cases per million of population
- Total cases per million of population
- Deceased per million of population
- Recovered per million of population

Also same metrics as overall analysis are available for each community (daily increments and historical evolution). In this point we have all our data clearly analyzed.

2.2. Creating a Dash app

Dash is a library created by Plotly, one of the most important charting libraries in Python. The idea is to have well performing, scalable and easy to use. It uses flask as backend. We are not going too much in detail about Dash, basically we created a class in a module for data processing called data.py, this way we can easily work with the same data in all the applications (in Dash we need an app for each page of the web).

All the code is available and documented in the repository of the project. We use the components of Dash Bootstrap Components to make it easier to have good UI and well looking page. It uses Twitter's Bootstrap classes and components.

Brief introduction to app scheme of Dash:

```
# -*- coding: utf-8 -*-
import dash
import dash_core_components as dcc
import dash_html_components as html

external_stylesheets = ['https://codepen.io/chriddyp/pen/bWLwgP.css']

app = dash.Dash(__name__, external_stylesheets=external_stylesheets)

app.layout = #content of the page

if __name__ == '__main__':
    app.run_server(debug=True)
```

For debugging we are running the application in local server running but it is important to not forgive to change debug to false when deploying.

Inside app.layout we have the content of each application, with all the chart, HTML elements and user inputs elements. For more details, please, check the code.

2.3. Deploying

For deploying the dashboard we are using a free-available solution called Heroku. It is a PaaS (Platform as a Solution) that give us cloud computing resources with support of tons of well known programming languages. We have some “free hours” to run our applications and in our case is more than needed.

It is a bit hard to deploy easily a Dash app into heroku, first because the docs are not clear when deploying multi-page applications and second because there is not to much information in the internet.

Basically we need to create a Procfile:

```
web: gunicorn index:server
```

This way we are telling the server that we want to serve gunicorn server contained in index.py file. It is also important to install gunicorn before deploying and make a requirements.txt file (easy if we are using virtual environments).

The next step is to install and configure Heroku CLI with our credentials and creating a new app. As we had a repository already created in the folder of the application the only thing we need to do is, everytime we make a change, push the changes to the heroku repository:

```
git push heroku master
```

The first time it needs a bit more because it has to install all the dependencies from the requirements, but then is fast and easy.

Important: as we are processing our data directly from Datadista's file, our dashboard and our analysis are automatically daily updated. The website reads and process the file each time the user goes in.

3. Applying machine learning

3.1. Feature engineering

At first we thought some additional information could maybe help us improve our results; finally as we are trying to approach the problem as time series prediction the new features result not useful. Nevertheless, this data is public and the dataset can be used by someone to make a new model or predict another target value. The features made were:

- Moving averages each 5 days
- Moving averages each 7 days
- Moving averages each 10 days
- R values, interesting calculated values that shows the ratio of infected people by each infected person. Thanks to Raul Martinez.
- For each day if flights coming/going from/to Italy were available.
- If the lockdown was already on.
- If the strict lockdown was on.
- If the day was inside Holy week (bank holiday in Spain).
- Weekday

This feature engineering is available in this notebook.

3.2. Modeling

The first idea of this project was to apply machine learning and deep learning techniques to predict the evolution of the pandemic. Unfortunately, the results are not good, we have data for only 52 days so it is completely impossible to find any pattern or to discover the trends data is following.

You can find some experiments to apply state-of-art techniques in this notebook, but the results are not satisfying.

3.2.1. Facebook Prophet

The first tried model was Facebook Prophet, a deep learning algorithm made to understand the trends and stationary of human data. Unfortunately, they recommend to have data of almost 1 or 2 years, not our case. Apply the algorithm give us:

```
MAPE 74.16064954939999
MAE 3196.3414328630943
```

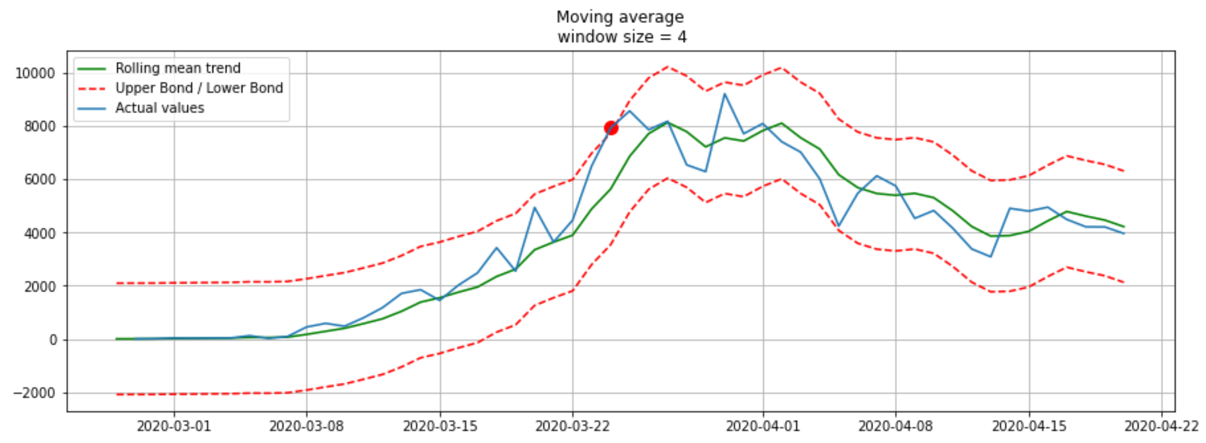
MAPE refers to Mean Absolute Percentage Error and MAE to Mean Absolute Error; so having 74% of error and 3200 cases is completely not good performing.

These results made the decision to finally not try DeepAR for the moment (maybe when we have more data).

3.2.2. Moving averages

The next approach was to use moving averages (we used them in analysis), okay, it is not formally machine learning but the final aim of the project is to understand better the pandemic, so let's give a try.

The idea of using moving or rolling averages is that the virus need some time to affect people, some time for people to recover and so on.



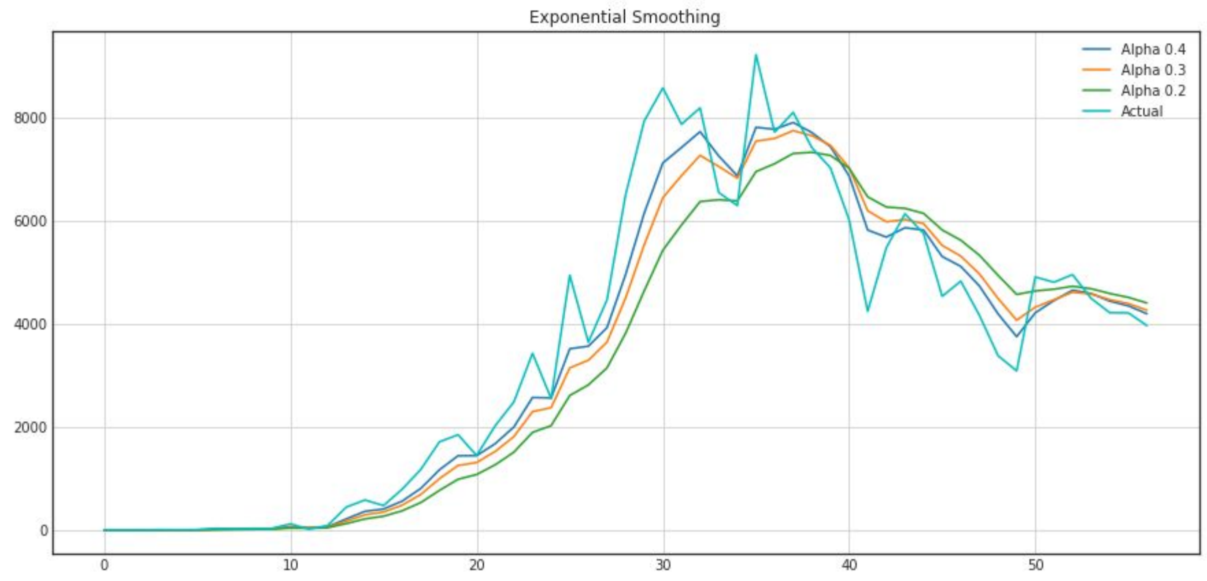
Using a window of 4 days we have this results, we can see that between our bonds the trend is briefly well followed.

So we can say that for a day-by-day prediction we can take a look at last four days average value of new cases. It is a good starting point. This model trends will be also deployed in the website in the upcoming version.

3.2.3. Exponential Smoothing

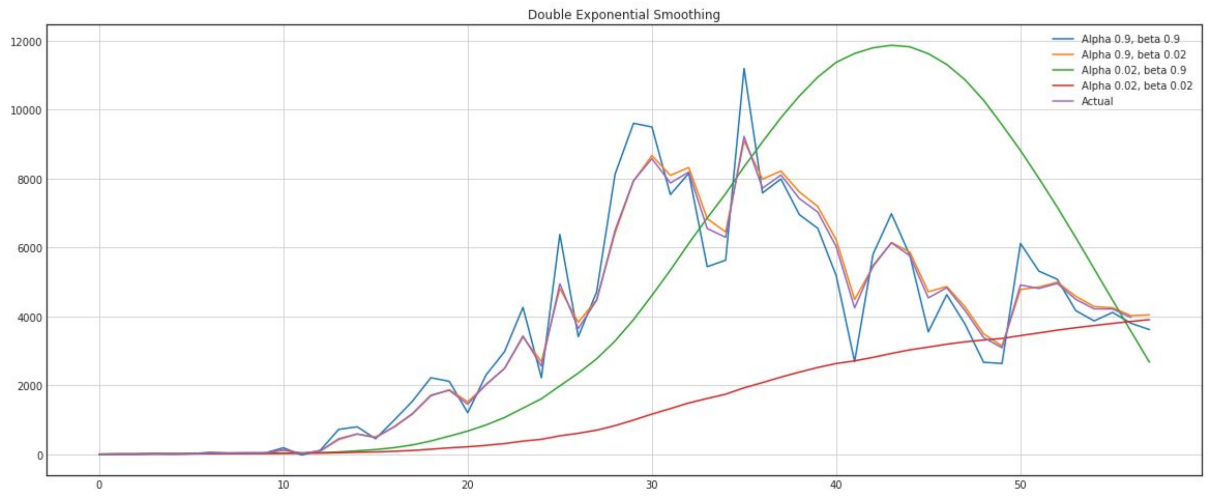
As the results with moving averages were good, we tried another approach commonly used for time series: Exponential Smoothing. It is commonly used in sales to predict the quantity of product to be sell. What it is doing is; for moving averages we were weighting all the past observations equally, in exponential smoothing we are using exponential function to decrease the weights in time.

You can find more information in the notebook but we will be plotting the exponential smoothing with different alpha values for smooth. Thanks MLCourse for the plotting and calculation functions.



We can see some models following the trends of actual data pretty well.

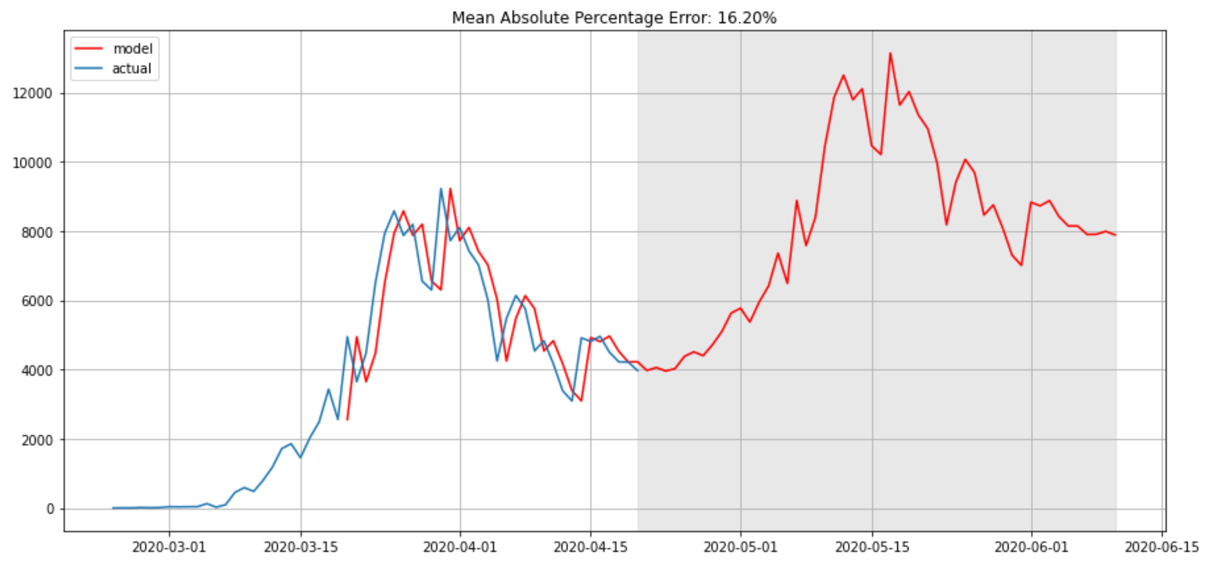
3.2.4. Double Exponential Smoothing



In double exponential we have to values, one called level component and a second one called trend component. Both components are used to update each period value. We can see it follows the test data even better than simple exponential smoothing.

3.2.5. SARIMA

The last tried model was SARIMA, Seasonal Autoregressive Integrated Moving Averages, this model aims to take the stationary component of our data to predict the future evolution.



We had 16.20% MAPE with this model, but it is not a good model, if we take a look at the picture we can see that SARIMA is supposing that our data will follow in the future the same pattern than in the past. Just take a look at the peaks, they are exactly the same.

Fortunately this is not true and we are not getting 12k new cases mid-May. The trend is to have cases going down. So this model is discarded.

3.2.6. Final thoughts

Normally, Machine Learning projects are more “metric” based, the point about this project is that we have only about 60 days data. This way if we have good “scores” it is normally because our model fits our past data, but in this case and in this project the trends of the past are not going to be any trend of future data. That’s why the project was focused more on exploring, dashboarding than modeling. Moving averages and exponential smoothing models will be available soon in the website.

We have found that moving averages can be a good way to predict daily cases but we don’t have enough data to understand the future of the pandemic. Another approach to be tested in the coming days is to take a look at other countries data and try some epidemiologic probed models.

Thanks for reading,
Juan Luis Ramírez