# Covid Datathon

## Team Quaranteed Success

Francisco Collado & Oriel Kiss

ETH Zurich

June 29, 2020

# Schedule

# SIR Linear model

# SIR model

The Susceptible-Infected-Recovery (SIR) model is a system of coupled differential equations:

$$\frac{dS}{dt} = -\beta SI \tag{1}$$

$$\frac{dI}{dt} = \beta SI - \gamma I - \mu_i I \tag{2}$$

$$\frac{dR}{dt} = \gamma I \tag{3}$$

Here $\beta, \gamma, \mu$ are the transmission, recovery and death rate respectively. So our idea was to use the different features of this model (S,I,R,SI) to build a linear model. We used the scikit-learn package and try a few linear model (Ridge, Lasso, elastic-Net). We then observed which one was the more probable and then tune the hyperparameters.

# Numerical Integration

Actually, things become more clear if we approximate the derivative

$$\frac{dS}{dt}|_t \approx S(t+1) - S(t) \tag{4}$$

. We can then rewrite the SIR equations as:

$$S(t+1) = S(t) - \beta S(t) I(t) \tag{5}$$
$$I(t+1) = I(t) + \beta S(t) I(t) - \gamma I(t) - \mu_i I(t) \tag{6}$$
$$R(t+1) = R(t) + \gamma I(t) \tag{7}$$

We could then guess the parameters $\beta, \gamma$ and $\mu_i$ and perform a numerical integration. Or we could build a linear model with $S(t)$, $I(t)$, $S(t)I(t)$ and $R$ as feature, use machine learning to approximate the parameters and then make prediction for the day $t$+1 given the day $t$. This is the method we have chosen here. The target variable are $S(t+1)$, $I(t+1)$, $R(t+1)$.

# Elastic-Net

At the end we chose the elastic net model, which is a mix of the ridge and lasso regression:

$$\hat{\beta} = \underset{\beta}{\text{argmin}}(||y - X\beta||^2 + \lambda_2||\beta||_2^2 + \lambda_1||\beta||_1) \tag{8}$$

The main part of the code looked like:

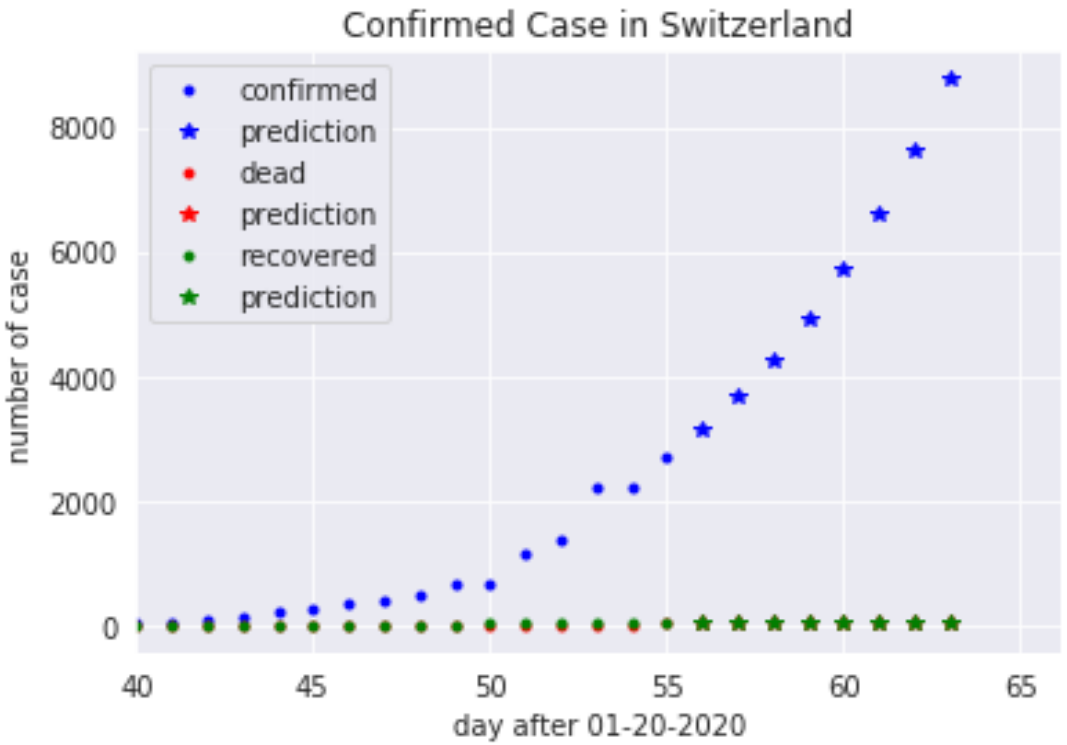
## elastic-net model

1. Define the target as the data for (S,I,R,D) sliced from one day in the future.
2. Train the model to predict the data from tomorrow with the data from today. Use all available data.
3. When the model is trained, then predict the next day and use this prediction to predict the next day and repeat the operation.
4. adjust the regularisation parameter to have a curve which make sens.

We remarked that the model tend to always increase so we had to choose a bigger $\lambda$ over time in order that the curve does not expose.

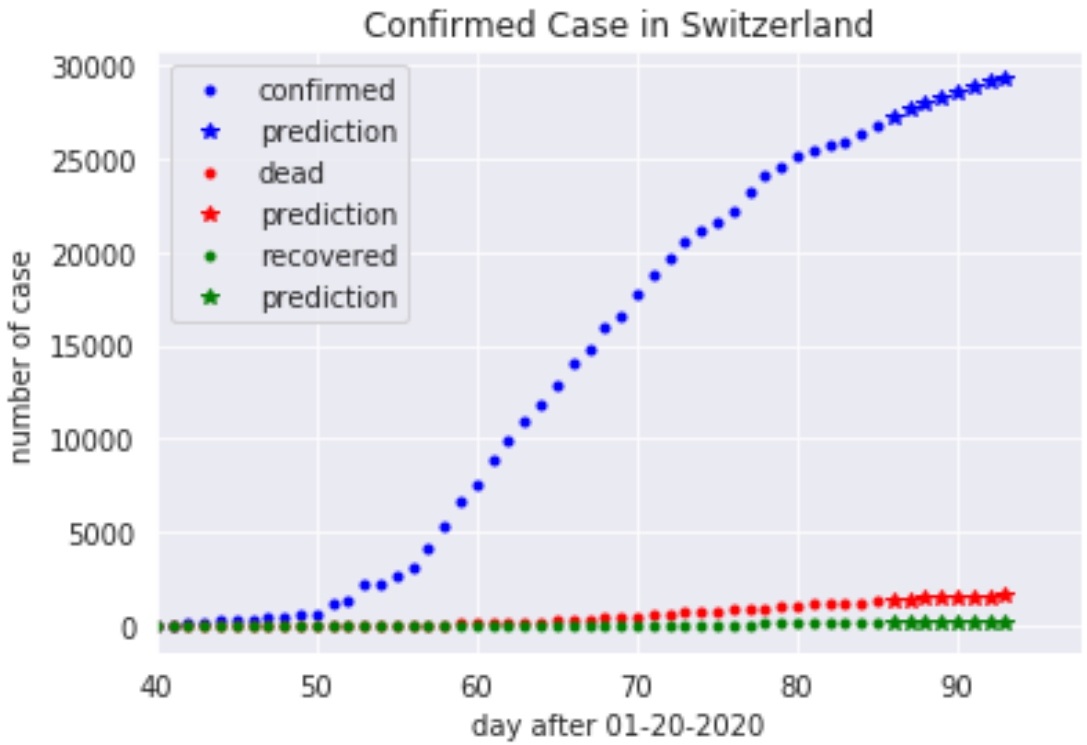
# Results for Switzerland

Prediction for the end of March. Here $\lambda$=100. The points are the data and the stars our predictions. Blue=Confirmed, Red= Dead, Green=Recovered.



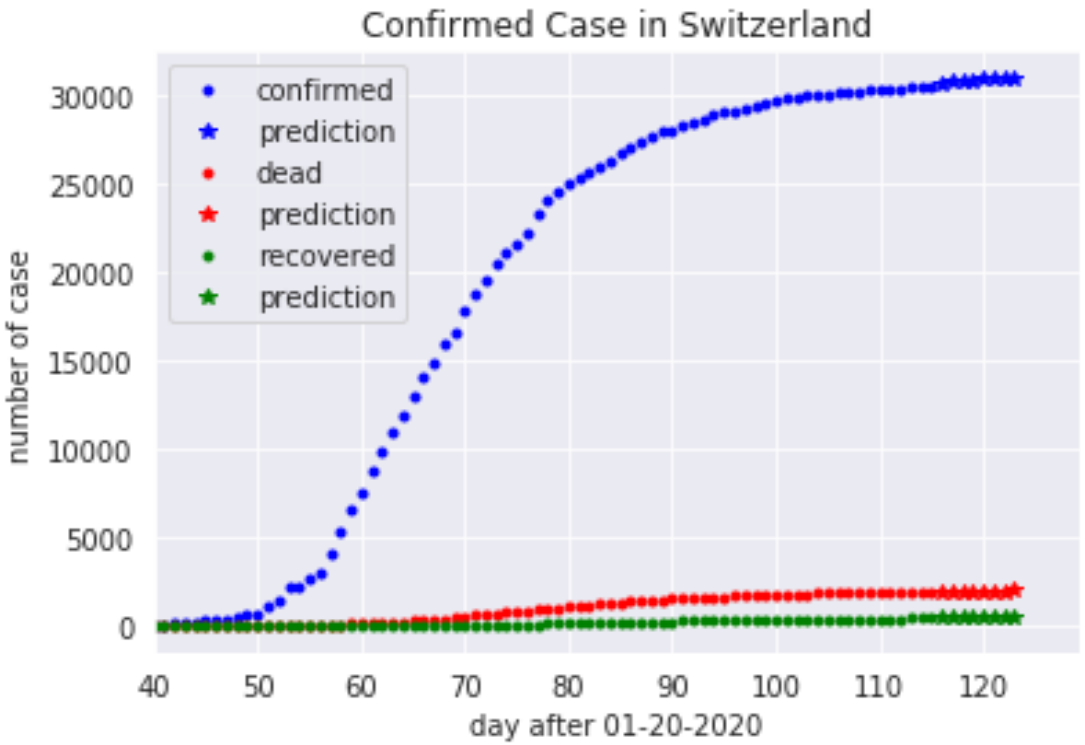
# Results for Switzerland

Prediction for the end of April. Here $\lambda$=0.1. The points are the data and the stars our predictions. Blue=Confirmed, Red= Dead, Green=Recovered.



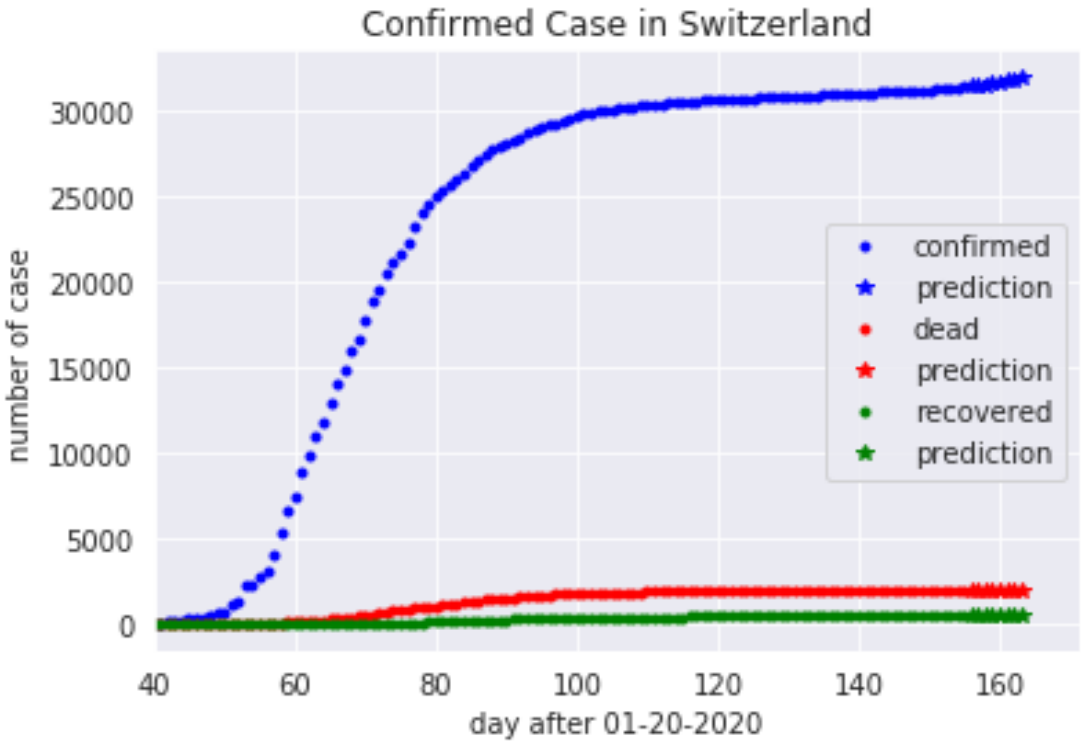
# Results for Switzerland

Prediction for the end of May. Here $\lambda$=1. The points are the data and the stars our predictions. Blue=Confirmed, Red= Dead, Green=Recovered.



# Results for Switzerland

Prediction for the end of June. Here $\lambda$=100. The points are the data and the stars our predictions. Blue=Confirmed, Red= Dead, Green=Recovered. We remark that this model tends to predict an augmentation of the number of infected people.



# Pros and Cons

Pros:

1. quick and easy to train
2. relative good prediction

Cons:

1. model is time independent
2. strong dependencies on hyperparameter

# Logistic Regression

# logistic function

We also tried in parallel to perform a logistic regression. Here the idea is to fit the logistic function:

$$\sigma(x) = \frac{A}{1 + \exp\left(-k(x - x_0)\right)} + C \tag{9}$$

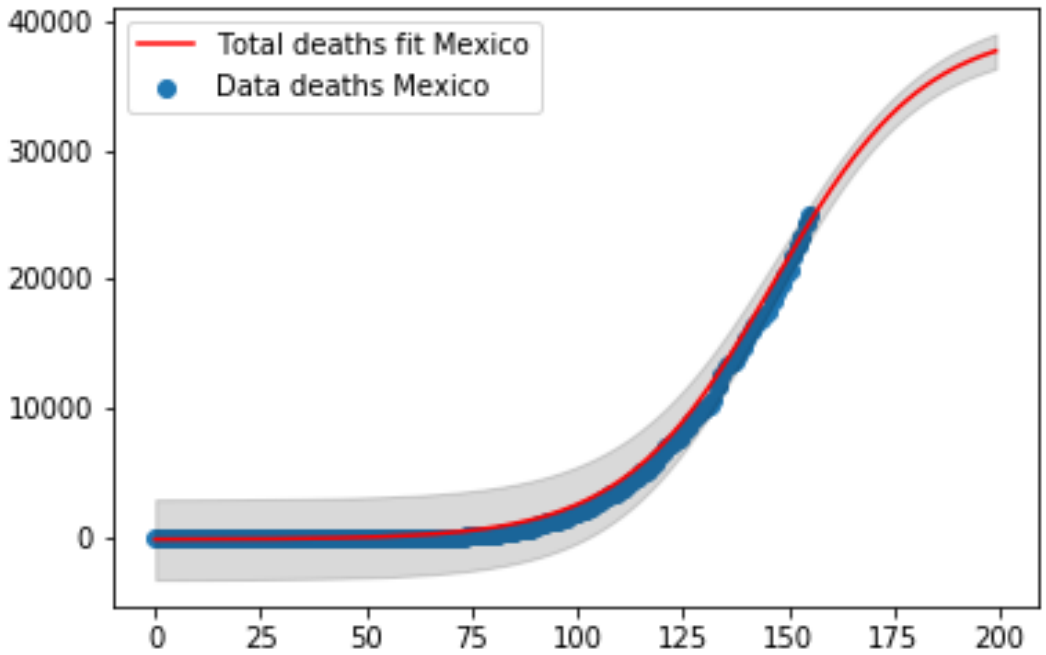
which has parameters $A$, $k$,$x_0$ and $C$. We then used the curvefit function from the scipy.optimize package to fit the logistic function to the data.

We think that this regression is a good idea because at the beginning, nobody is infected and at the end a certain fix number of people (here $A$) is infected. In the time in between, the infected rate grows fast, like a logistic function.

The curvefit function just optimizes the choice of parameters with a least square loss function.

# Results for Mexico

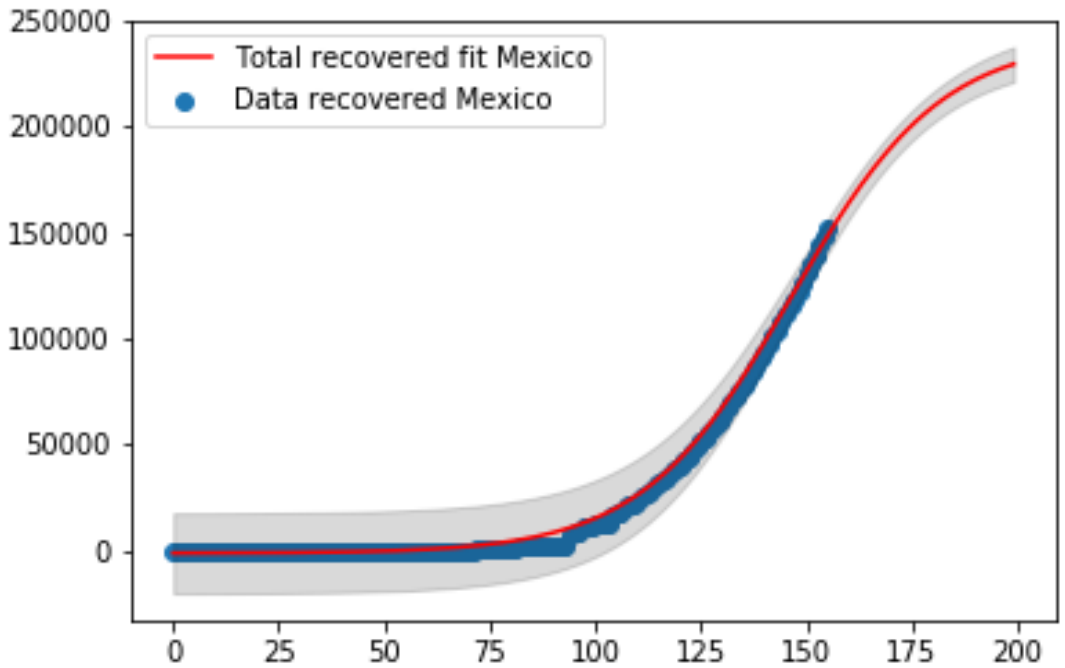
We used the logistic regression to predict the number of dead and recovered cases:
For example for Mexico we have for the number of death:




The gray area is a 95% confidence interval.

# Results for Mexico

We used the logistic regression to predict the number of dead and recovered cases:
For example for Mexico we have for the number of recovered people:




We see that the red curve fits the data well and we are able to make prediction until 30 days in the future.

# Predict the new cases

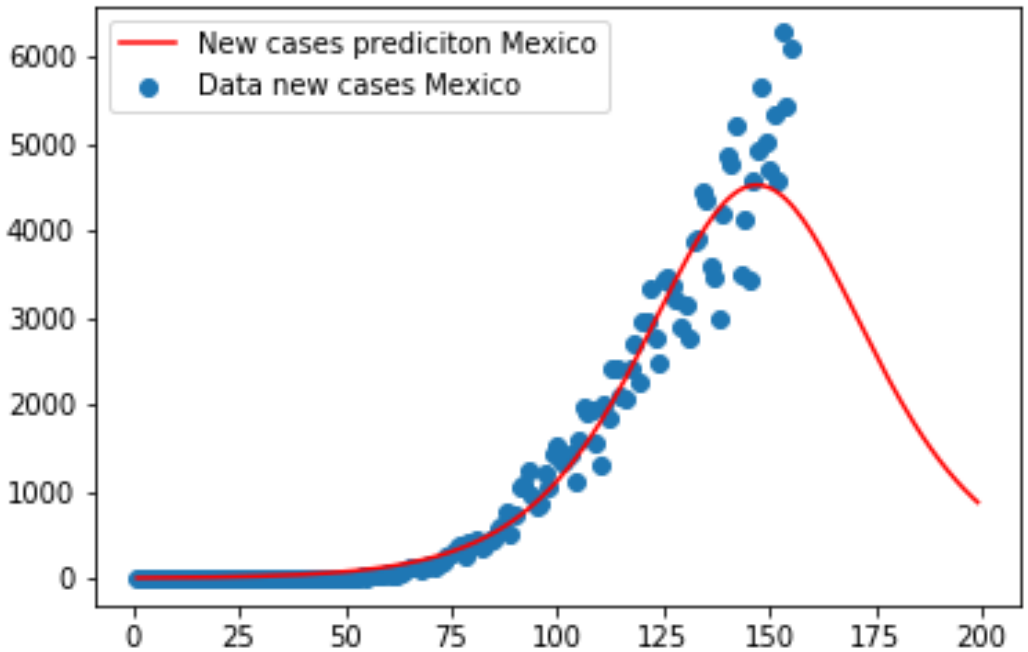
To predict the new cases, we have to use a new technique. Indeed the number of new cases does not go from 0 to $A$ but rather explodes quickly and then goes down. So we assumed that the new cases are normally distributed and fitted

$$f(x) = A \exp\left(-\frac{(x - x_0)^2}{\sigma^2}\right) \tag{10}$$

to the data. We also used the curvefit function from scipy to optimize the choice of $A$, $x_0$ and $\sigma$.

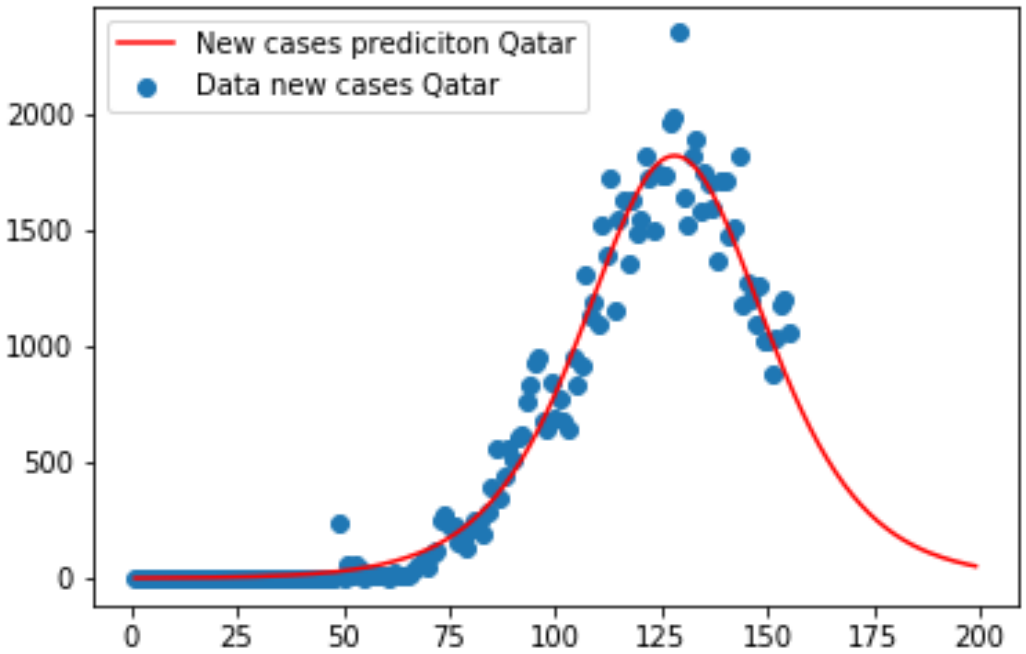
# Results for Mexico

We got for The new cases in Mexico:

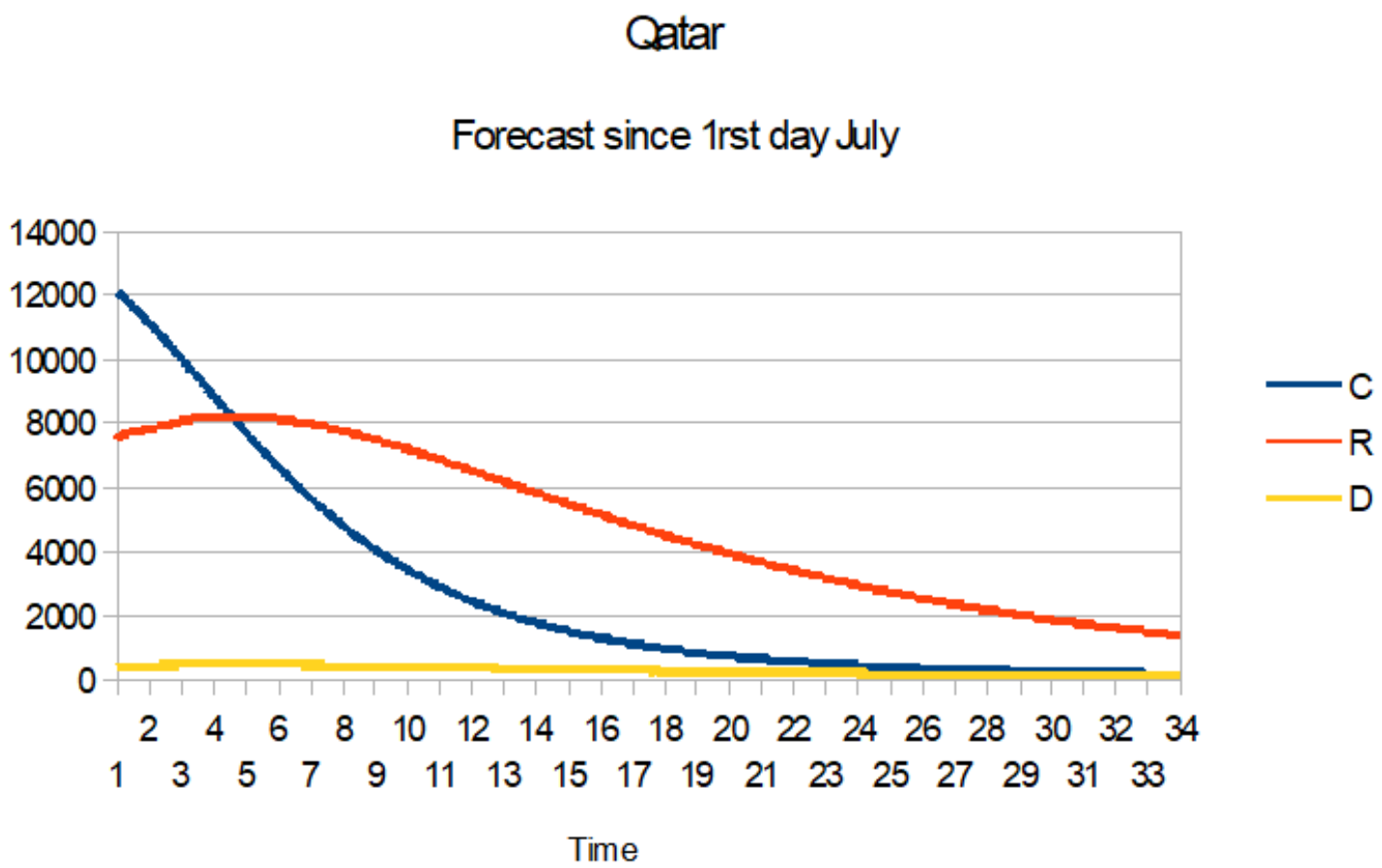



We remark that this does not fit the data so well so maybe this assumption was not accurate enough in this case. However, it works better for other countries, like Qatar.

# results for Qatar

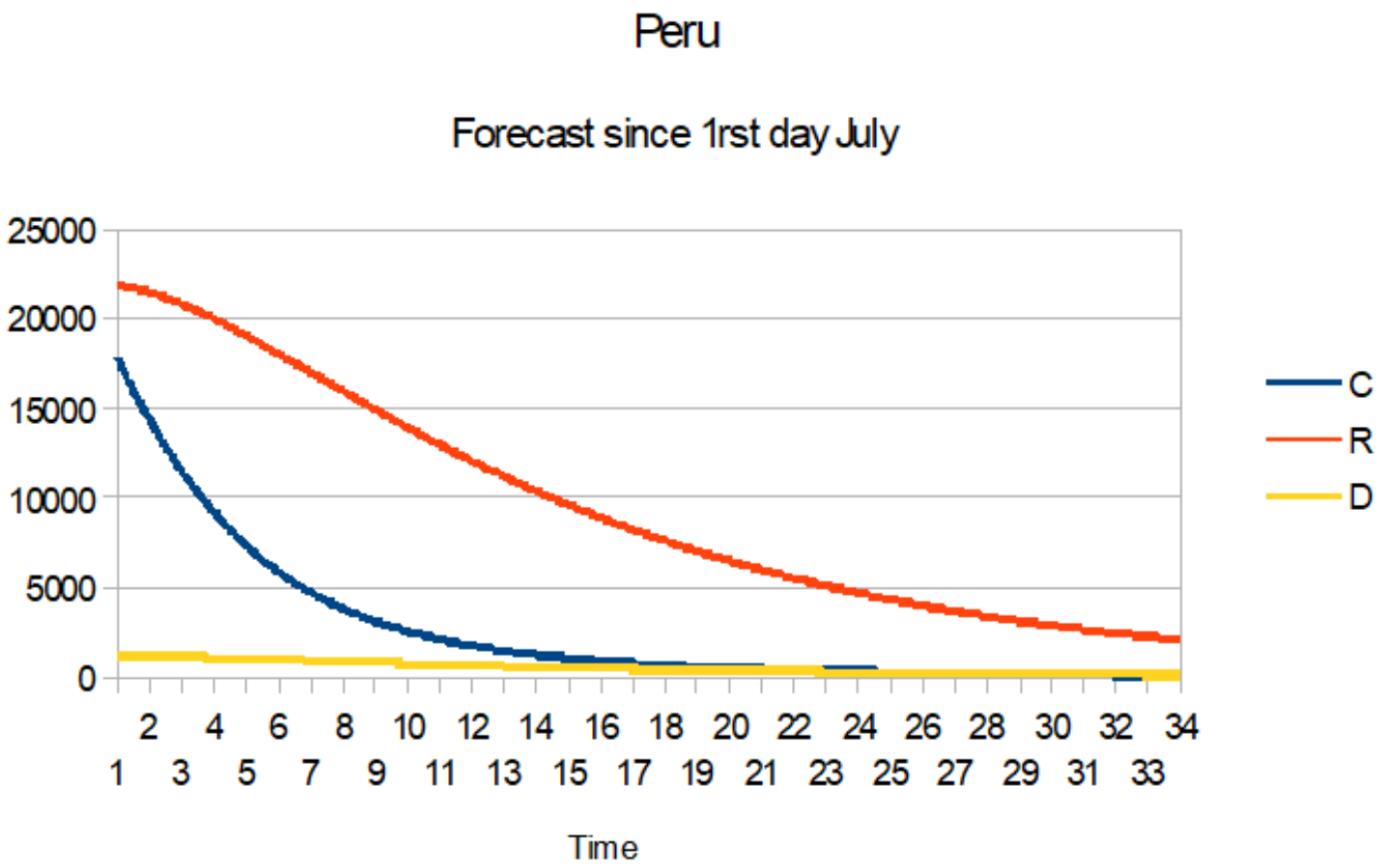


# results for Qatar in July



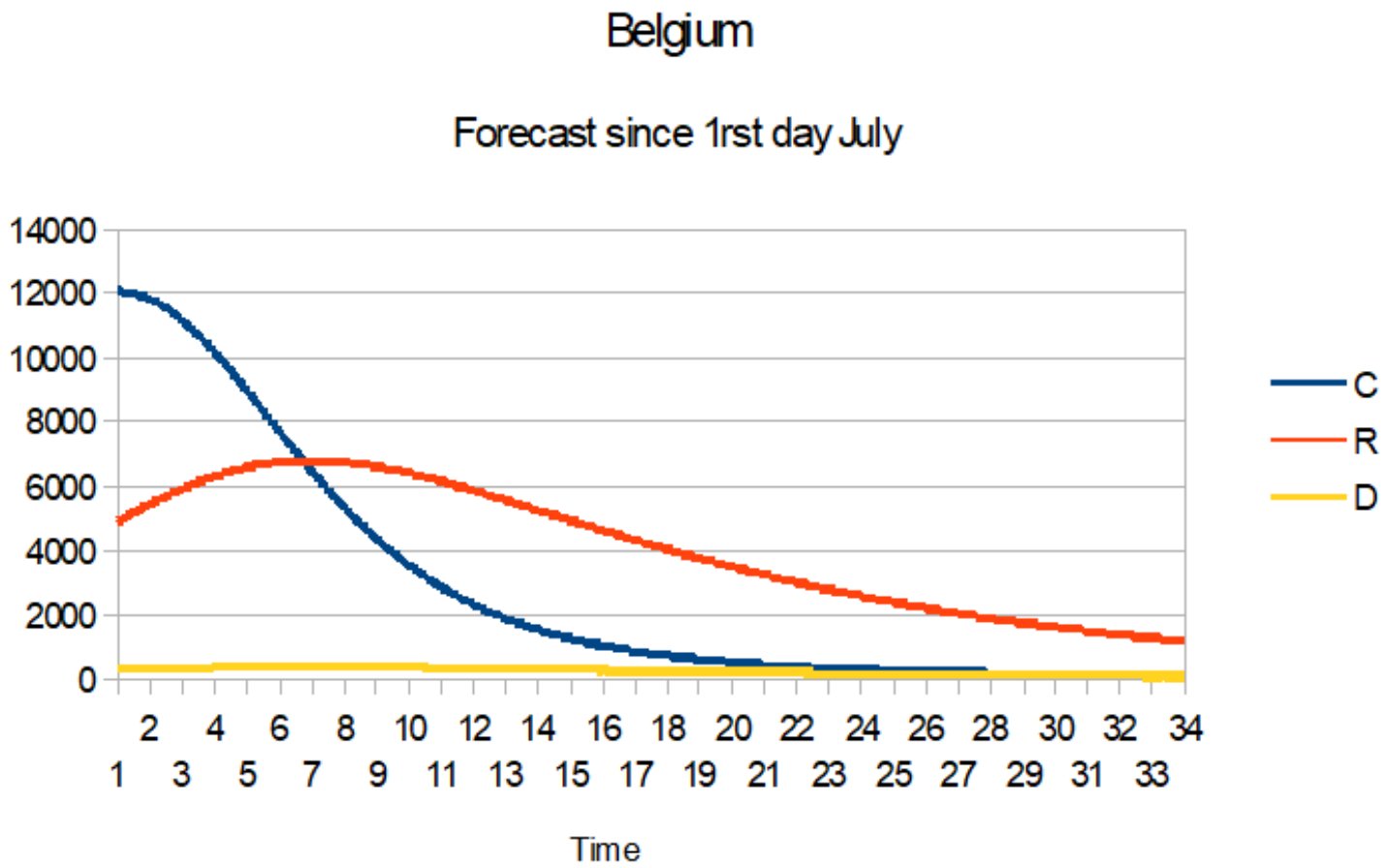

From another perspective to Qatar, taking into account the evolution of all the totals since this week, the fall in Contagion, Recovered and Dead is observed.

# results for Peru in July



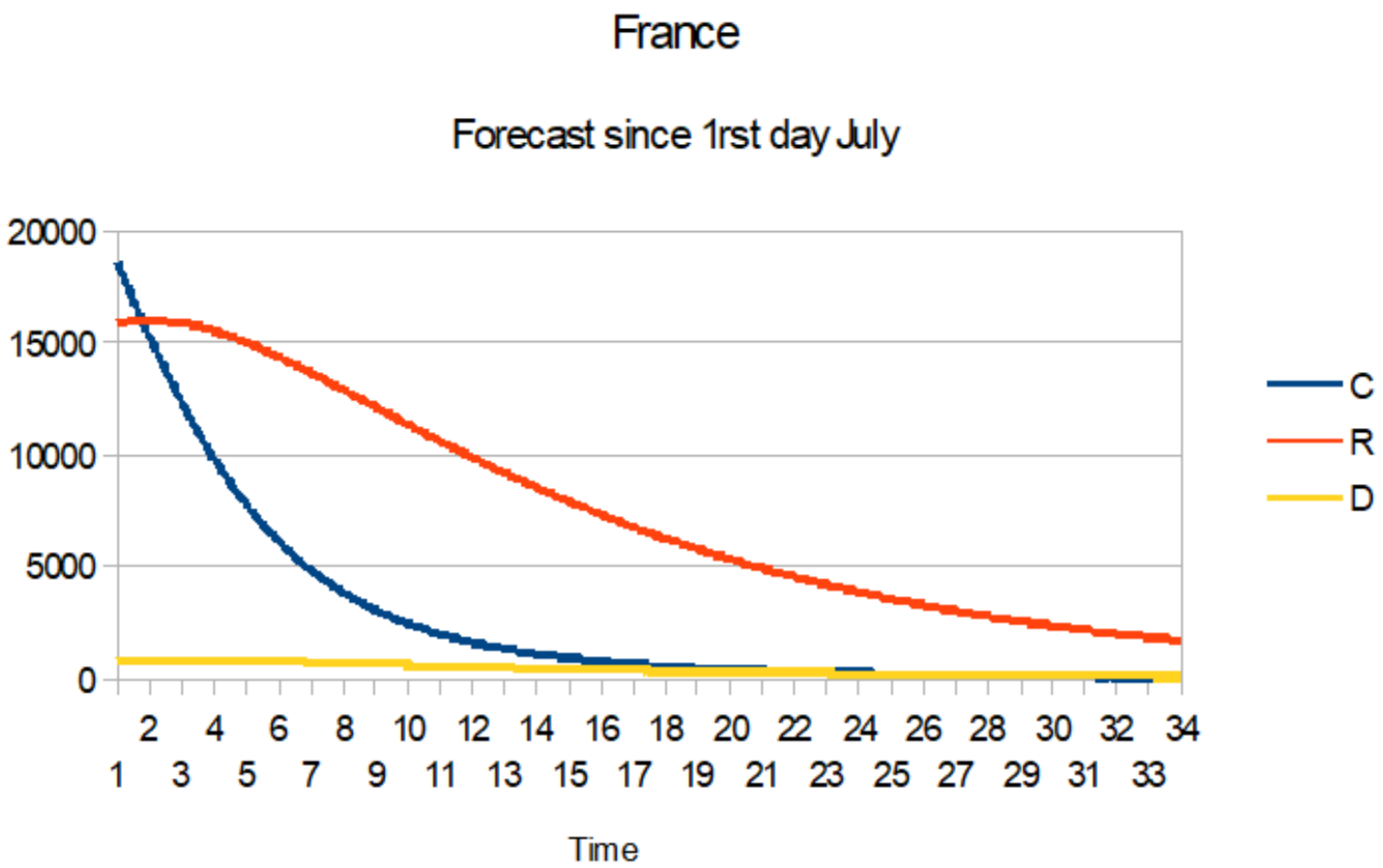

Same perspective for Peru since this week, the fall in Contagion/Cases is better than Qatar, Recovered is slower and Dead is stable.

# results for Belgium in July



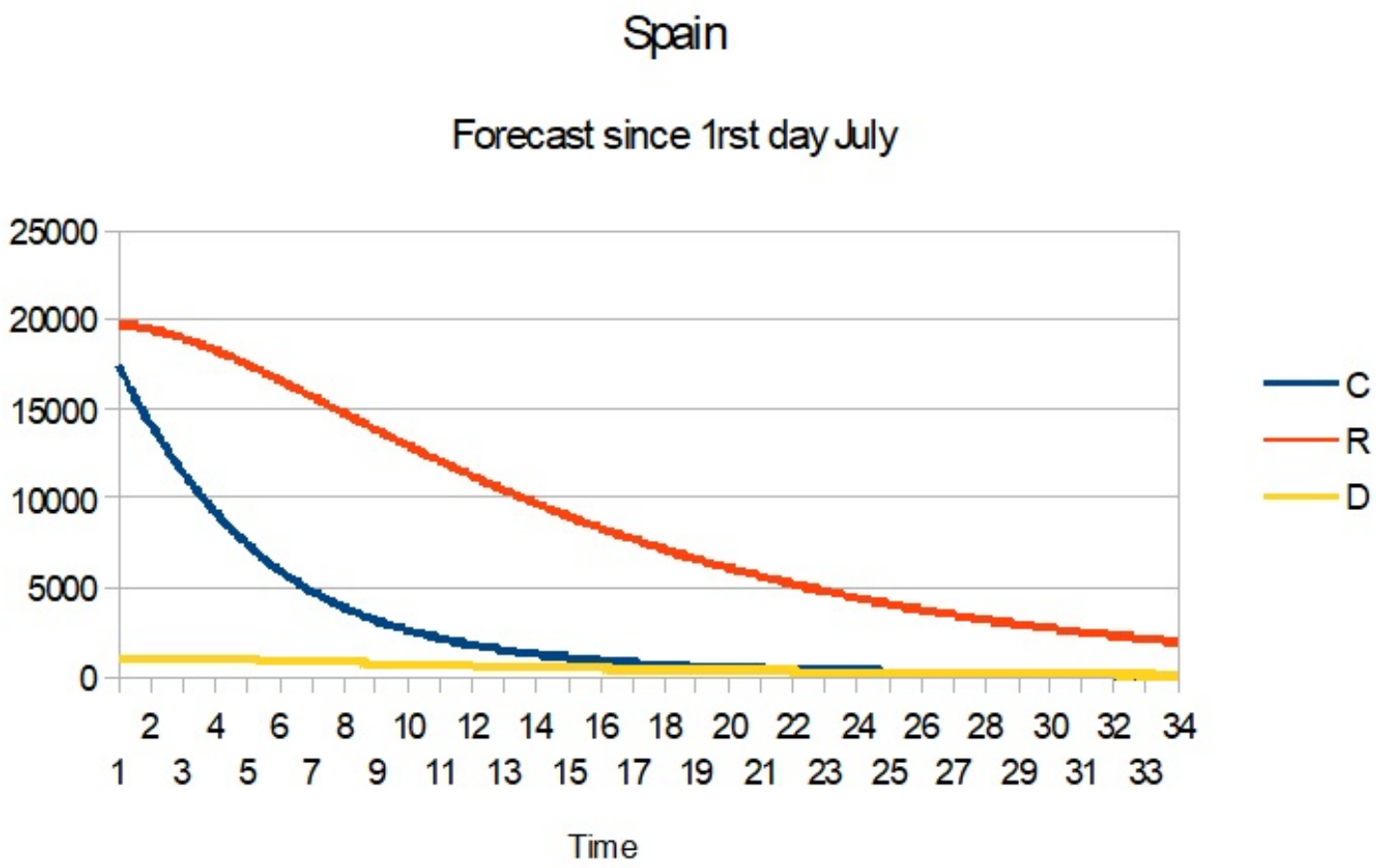

Belgium will increase the number of Recovered at the end of this week.

# results for France in July




France will cross the lines of Contagion/Cases (infected people) and Recovered people in a couple of days.

# results for Spain in July




Spain will have the number of Recovered slower than the other countries due to the high number of Cases.