COVID-19 Case Incidence in US Counties

Thomas Le Menestrel
ICME Department
Stanford University
tlmenest@stanford.edu

German Kolosov
MS&E Department
Stanford University
gkolosov@stanford.edu

Mahammad Shirinov
MS&E Department
Stanford University
shirinov@stanford.edu

1 Dataset

The dataset consists of 31 signals that may be useful in predicting COVID-19 cases in the United States, for the top 100 counties in terms of overall case counts, between June and November 2020. These counties are identified by a five-digit FIPS (Federal Information Processing Standard) code. The exact objective of interest (`response`) is COVID-19 case incidence counts per 100,000 people, 14 days ahead.

The features included in our dataset come from a variety of sources; some were collected by facebook surveys on COVID symptoms, mask wearing and other questions, others come from hospitalization numbers and other indicators, and yet others come from SafeGraph and are about the number of visits to bars and restaurants.

2 Approach

2.1 Task

The goal of this project is to build a regression model to forecast COVID-19 case incidence in 100 US counties.

2.2 Metrics

The error will be measured using a custom loss function defined as:

$$l(y, \hat{y}) = |\log(1+y) - \log(1+\hat{y})| \tag{1}$$

The model will make predictions for the test set, which runs from 01 Dec to 31 Dec for the 100 counties.

2.3 Exploratory Data Analysis

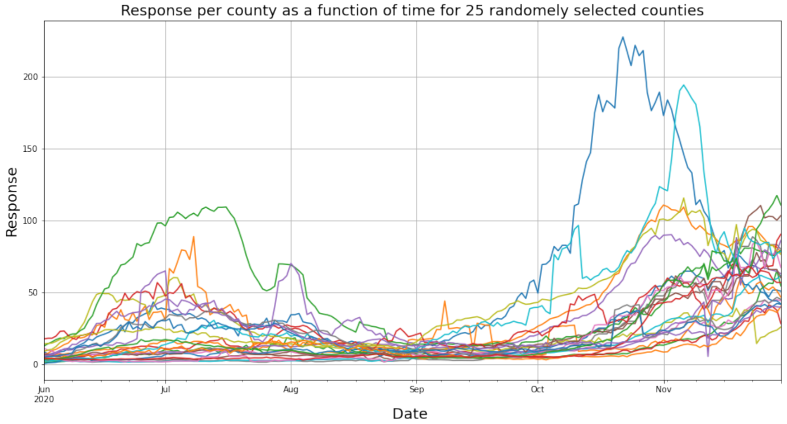


Figure 1: Response per county as a function of time for 25 randomly selected counties

Taking an initial look at the evolution of cases over different counties, we can tell that they are very different. The period from mid-August to late September seems to be similar for most of them, but at all the other periods of time the disease evolves very differently and has peaks at different times. This gives us an initial suggestion that our model should take into account the county information as well as the date information. However, using date is not trivial, especially because the case movement changes for different counties at different times.

2.4 Feature Engineering

There are two features in the dataset that are extremely useful but not immediately usable: `date` and `county`. Since the `date` column contains the dates of the observations, and there is nothing inherent to the date that might help us predict the case counts better, we use this column only to transform our rows. Specifically, we used date to augment each datapoint by adding to it the features of observations from k previous days for different values of k, mimicking an autoregressive model. We experimented with different features and different numbers of days. Once we have these features, our models then drop the date information. We refer to these features from previous days as *lagged features*. In addition to the lagged features themselves, we also included the difference of these with the corresponding features of the current date, and also the percentage difference. To make these lagged features also available on the test set, we drew upon the necessary dates in the training data and used them to augment test datapoints.

Additionally, we also tried to make use of the `date` information in a way inspired by Question 5 on HW 3. Specifically, since our data is temporal and calls for temporally changing models, we modeled the coefficients of our features as functions of time, by using natural cubic splines. Our intuition was that this would let the model learn time-varying relations between the response and features, instead of having a static relation coefficient for all times over the course of half a year. However, this didn't show promising results in practice, so we decided against using it.

For `county`, we took three different approaches. In our first approach, we partitioned our data by county, and fit a model separately to each one, effectively resulting in 100 models. However, this approach has the obvious drawbacks that the amount of data for each model is 1/100 of the original data. Furthermore, the models do not learn the "global trends" that might benefit them all, and this reduces the efficiency of learning further.

Our second approach was to encode county as a one-hot vector. This improved the fit across the board for all the models we tried, so we kept it in our final model.

Our final approach for `county` was similar to the one discussed before; we transformed this data into one-hot encodings, and then we took interaction terms with these encodings and other features. The idea here was to let the linear models learn to differentiate importance and weights of features across counties – without these interaction terms, the model learns a single coefficient for every feature, but fails to take into account how different features might be of different importance across different counties. However, to our surprise, this worsened the performance instead of increasing it, so we opted against this approach in our final model.

2.5 Feature Selection

In this part we will define a procedure to create meaningful features that our model will use to predict the **response**. Our main intuitions are : First, and most importantly, to predict the number of COVID cases two weeks after, the model needs to capture not only the situation at the time **t**, but also the overall trend of the indicators. For example if some indicators show an increasing trend, the model should use it to predict the the response. T

Given this, we think that we need to add to the model some lagged features, in order to incorporate the past behaviour. To do so we will define 3 types of lagged features:

For every feature β, at the time t, for the county c, we will define:

- $\beta_{shift}(n) := \beta(t-n)$
- $\beta_{diff}(n) := \beta(t) - \beta(t-n)$
- $\beta_{pct\_change}(n) := [\beta(t) - \beta(t-n)]/\beta(t-n)$

The intuition behind the three types of lagged features is the following:

- β_{shift} captures what happened in the past
- β_{diff} captures what the changes between the past and today.
- β_{diff} captures what the return between the past and today, which is the relative changes.

As these features are computed per county, they incorporate the county information as well. The only thing that we need to do is to define the appropriate lag days n, for these features. Intuitively, some of the features are more short-term and the optimal lags for these would be few days, whereas some others, gain in predictive power when are computed with higher lags.

To find the optimal lags we will use the following technique :

$$\forall \beta, \in H, n^*(\beta) = argmax_{n \in S} corr(\beta(n), y)$$

Hence, we for every raw feature and every type in $\{shift, diff, pct_change\}$, we define the lag which maximizes the correlation of lagged the feature with the response.

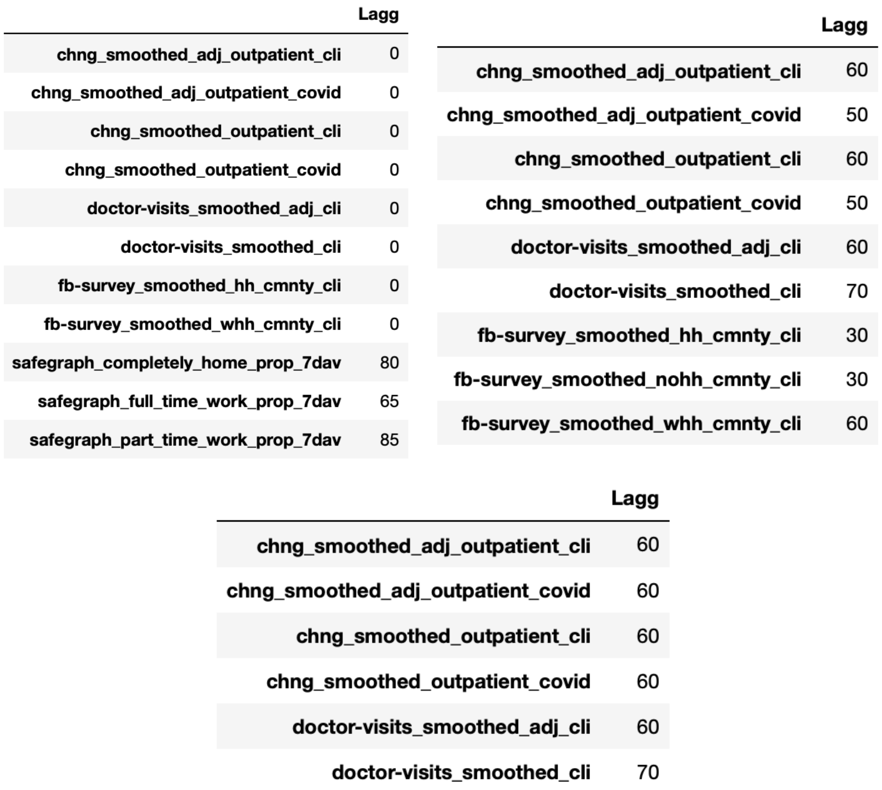
2.6 Models

Our final model is ElasticNet on our augmented data matrix. Since it's not immediately clear which features are important to include and exclude, we took the approach of including a lot of features and letting the model to shrink and exclude features itself.

2.7 Results

Our model reaches a **validation score of 0.24**, which we computed by splitting our training data into training and validation sets. With all our experimentation with splines, one-hot encoding interactions, this is the maximum score that we attained.

Below (fig. 2) is a list of the features that we select with correlation analysis and feed to our model. They consist of the original features, their lagged versions, diffs and pct_changes.



	Lagg
chng_smoothed_adj_outpatient_cli	0
chng_smoothed_adj_outpatient_covid	0
chng_smoothed_outpatient_cli	0
chng_smoothed_outpatient_covid	0
doctor-visits_smoothed_adj_cli	0
doctor-visits_smoothed_cli	0
fb-survey_smoothed_hh_cmnty_cli	0
fb-survey_smoothed_whh_cmnty_cli	0
safegraph_completely_home_prop_7dav	80
safegraph_full_time_work_prop_7dav	65
safegraph_part_time_work_prop_7dav	85

	Lagg
chng_smoothed_adj_outpatient_cli	60
chng_smoothed_adj_outpatient_covid	50
chng_smoothed_outpatient_cli	60
chng_smoothed_outpatient_covid	50
doctor-visits_smoothed_adj_cli	60
doctor-visits_smoothed_cli	70
fb-survey_smoothed_hh_cmnty_cli	30
fb-survey_smoothed_nohh_cmnty_cli	30
fb-survey_smoothed_whh_cmnty_cli	60

	Lagg
chng_smoothed_adj_outpatient_cli	60
chng_smoothed_adj_outpatient_covid	60
chng_smoothed_outpatient_cli	60
chng_smoothed_outpatient_covid	60
doctor-visits_smoothed_adj_cli	60
doctor-visits_smoothed_cli	70

Figure 2: Optimal lags for Shifted, Diff and Pct_change Features

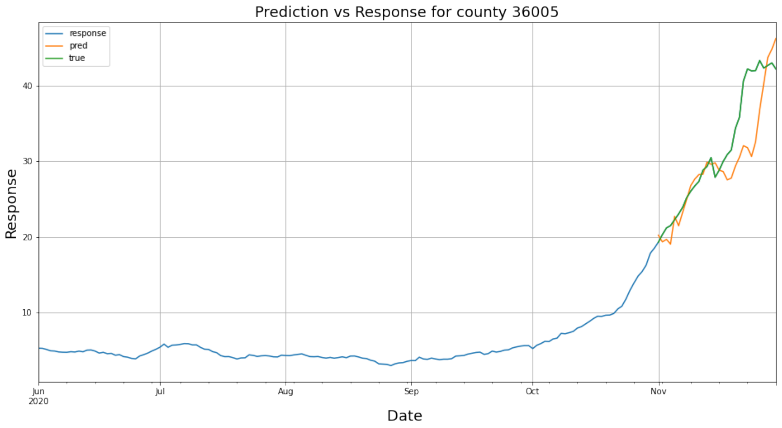
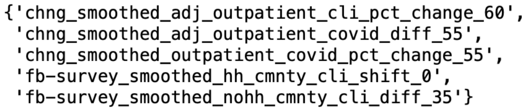


Figure 3: Train on October and predict on November

Figure 4 shows the features that our elasticnet model uses to make predictions at the end. All the other features are discarded. What's interesting about the remaining features is that they contain features of each type – shift, diff, and pct_change. What's more, the model uses the data of some of the features belonging to today, 1 month ago and 2 months ago, effectively selecting the features that matter the most.

Figure 5 shows the performance over different counties. As can be seen, the performance varies a lot.



```
{'chng_smoothed_adj_outpatient_cli_pct_change_60',
 'chng_smoothed_adj_outpatient_covid_diff_55',
 'chng_smoothed_outpatient_covid_pct_change_55',
 'fb-survey_smoothed_hh_cmnty_cli_shift_0',
 'fb-survey_smoothed_nohh_cmnty_cli_diff_35'}
```

Figure 4: Features eventually selected by the model

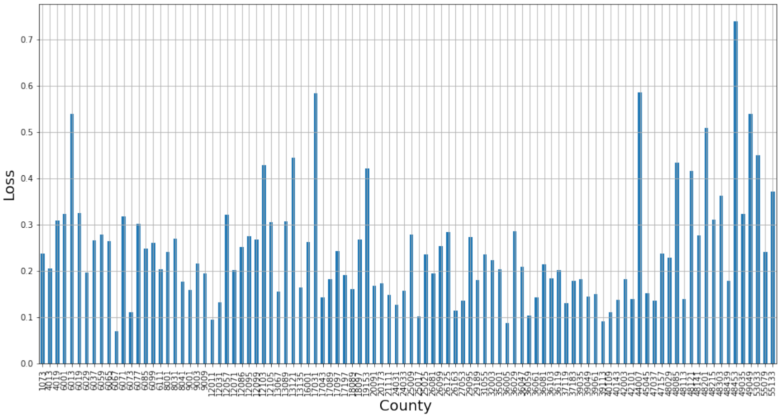


Figure 5: Train on October and predict on November: test error per county

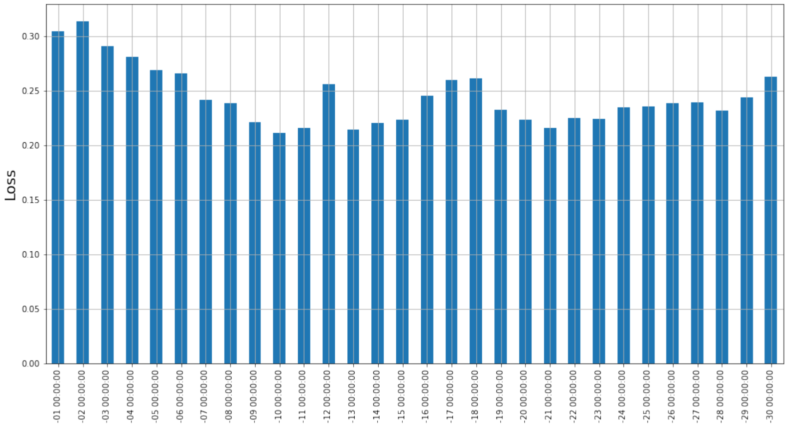


Figure 6: Train on October and predict on November: test error per date

In figure 6, we show the loss over the whole month of December. We see that, the loss is pretty constant during the validation month. What is surprising is that the mean loss doesn't seem to increase with time.

2.8 Cross-Validation

We used cross validation to select the parameters of our ElasticNet model. We didn't base our feature selection entirely on correlation – we divided our training data into train and validation, and used the validation set to select features. We do realize that a proper usage of CV must include the feature selection process we do beforehand in itself, but that is left to future work.

3 Future work

Some of the ideas that we had and that were beyond the scope of the class was to deal with the problem as a dynamic prediction problem where every of our predictions is then verified by the actual response that we observe later. We may use the error as a cost and train an agent to learn to minimize the cost. One additional development would be to combine this supervised learning with auto regressive time series forecasting using SARIMA model in order to incorporate past responses as features to predict the new ones and use autoregressive properties as well as seasonality.