Loan Project

Haitang Wang

**Abstract**

In this report, we evaluate the percentage of charged off loans at the end of 3rd year. The data set is analyzed, and tendency of saturation is fitted using logarithmic and exponential models. The optimization algorithm is based on least square of errors/residual sum of squares. Logarithmic model gives the best result. The same approach is further applied to data set generated using bootstrapping technique was applied. The result shows 6.29 percentage of charged off loans at the end of 3rd. Assumptions and improvements are also discussed in the context.

1. **Data preparation**

Column 1: days elapsed between the origination and the date when the data was collected

Column 2: days from origination to chargeoff, it tells us the essential information when a loan is charged off. Here, I name this column data as a new array = *charged\_off*

Package: Pandas.DataFrame

* 1. **Resampling**

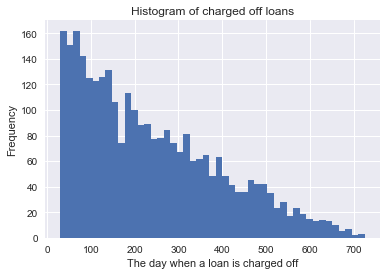
To simplify the problem, I assume every month is 30 days. I initialize **x** in days in the following:

**x**: =np.linspace(30, max(*charged\_off*), int((max(*charged\_off*)- 30)/30))

We then counting the number of charged off loans at time x. So we have:

**y**: = [sum(charged\_off<=i) for i in **x**]

* 1. **Visualization of y:**

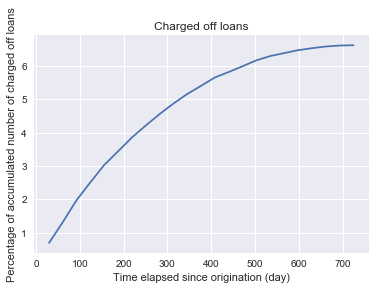


* 1. **Observation**

Borrowers who pay the loans in the beginning tend to continuously pay their loans. As a result, the number of the newly added chargeoff loans decreases over time.

* 1. **Problem discussion**

To find the prediction value, we can either predict the number of newly added the chargeoff loans and then calculate the accumulated percentage of charged off loans, or we can calculate the accumulated percentage of charged off loans and then predict the result. My approach is the second one. The distribution of percentage of accumulated charged off loans is in below:



1. **Modelling** 
   1. **Linear regression model**

The curve of charged loan is non-linear. If we want to use linear regression library, we I will have to expand the order of the input.

* 1. **Multivariate fitting using logarithm with least square**

Package: Scipy.optimize.leastsq

To capture the tendency of saturation, I introduce logarithm term. The function is written:

* 1. **Multivariate fitting using exponential with least square**

* 1. **Bootstrapping**

Input: array of charged off loans, described by the elapsed time between origination and the day when the loan is charged off.

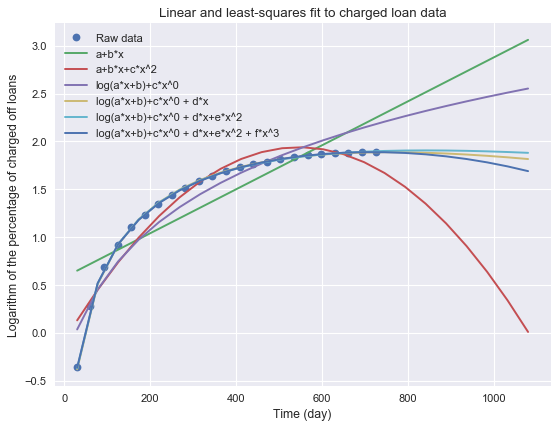
There are a limited number of charged off loans, when compared to the total number of loans investigated. We will resample loans with replacement. We further calculated the accumulated percentage of charged off loans over time, and then build models to predict the percentage of charged off loans at the end of year 3.

The average of the predicted values is used as the final prediction.

* 1. **Discussion** 
     1. **Assumptions**

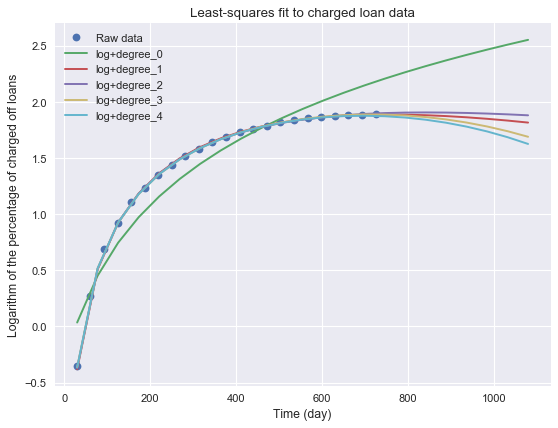
1. For the loans that are charged off, the company won’t receive further payments of those loans. Thus, the number of accumulated charged off loans will only increase over time.
2. Every month has 30 days. Thus, one year has 360 days.
   * 1. **Exploration of linear, logarithmic, and exponential models**

Even though the raw data only covers 30 to about 700 days, one would expect the curve will slightly increases along with increase of time. I didn’t add high orders in the in following plot, since the higher orders introduce overfitting for time higher than 700 days.

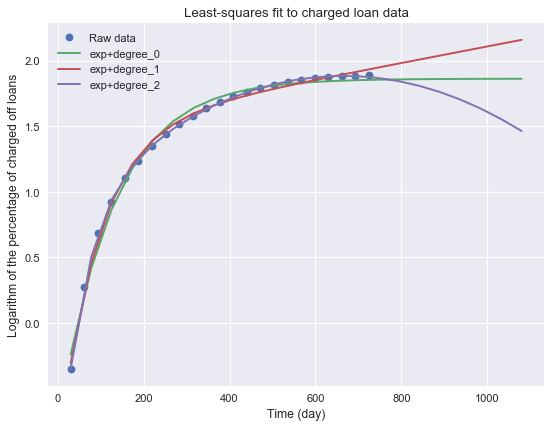


Clearly, linear fitting doesn’t work well with this given dataset. Here, we further focus on fitting functions with exponential terms and/or logarithm terms.

For the logarithmic model, we have the following result, and the one with degree 2 works among other fitted logarithmic models. In the plot, degree 4 means it has *x*4.

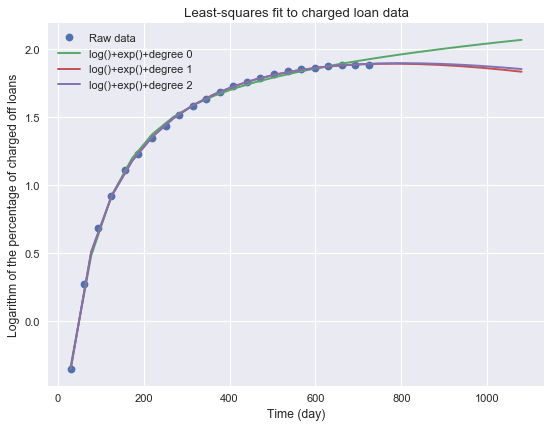


For exponential model, we have the following result, and the one with degree 0 works the best. However, the combination of fitted functions with degree 0 and degree 1 may provide a better solution, given the fact that fitted exponential function with degree 1 (*x*) is slightly over fitting the data points, while the one with degree 0 is slightly underfitting the data points.

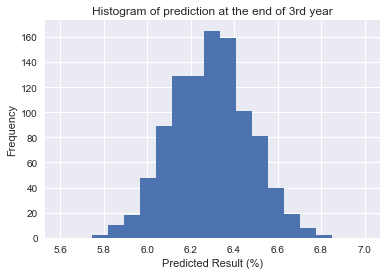


* + 1. **Combing logarithmic and exponential terms**

Unfortunately, it doesn’t really help by fitting logarithmic and exponential terms, despite the model can capture the raw data very well.



1. **Result**



Comparing the functions explored, we are using the logarithmic model with degree of 1, since it is better at handling with variance. The above plot shows 1000 results based on sampling with replacement. Here is a summary:

|  |  |
| --- | --- |
| Properties | Results |
| mean | 6.29%± |
| standard deviation () | 0.18% |
| median | 6.30% |

1. **Further improvement**

As we have mentioned here, we applied logarithmic models for this work, and the model we selected has overfitting the region of time above 730 days. This issue can be improved by adding some tricks, like neglecting some data points or adding more weights to the data close to saturating region, so that our model will propagate better in the end. Another approach would be averaging different models. Additionally, other optimization algorithms, like simplex may also be explored. It’s also important to know that we only sampled 1000 times in order to shorten the time needed for evaluation.