# Lending Club Loan

## Team: 8

**Lending Club** is a US peer-to-peer lending company, headquartered in San Francisco, California.It was the first peer-to-peer lender to register its offerings as securities with the Securities and Exchange Commission (SEC), and to offer loan trading on a secondary market.

Lending Club enables borrowers to create unsecured personal loans between $1,000 and $40,000. The standard loan period is three years. Investors can search and browse the loan listings on Lending Club website and select loans that they want to invest in based on the information supplied about the borrower, amount of loan, loan grade, and loan purpose. Investors make money from interest. Lending Club makes money by charging borrowers an origination fee and investors a service fee.

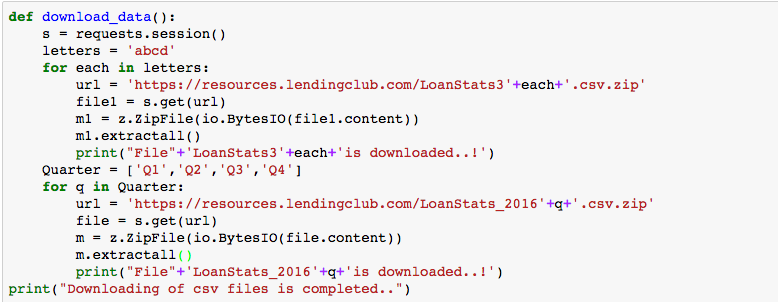
**Problem Statement:**

You are working at a bank and you are considering investing in Lending club. Since there are no standard models, you are expected to build prediction models that will help you predict the interest rates based on various parameters users would input.

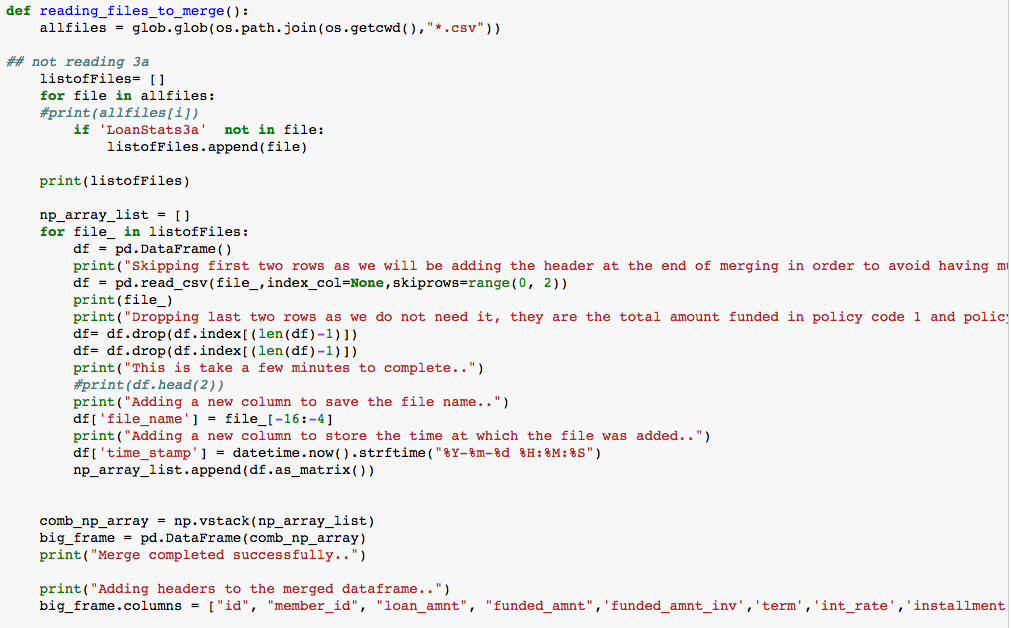
**Part 1: Data wrangling and exploratory data analysis:**

* Programmatically downloaded all the loan data csv files from <https://www.lendingclub.com/info/download-data.action>

inside the docker container.

****

* Once all the csv files are downloaded, read all the files into a list except LoanStats3a as it has loans that did not meet the credit policy. We will read that separately and then merge it with the rest of the loan data.
* We are skipping the headers, row: ‘Notes offered by Prospectus (https://www.lendingclub.com/info/prospectus.action)’and , last two rows as they are not required and we will add header column after merging the data in order to avoid having duplicate headers in the file.
* While reading the files into the data frame as one, we added a timestamp and also the filename in order to when the data was recorded and belongs to which loan data file.

****

* Read LoanStats3a csv file and divided into by reading the loan status column. If the loan status column does not have values equal to **Does not meet the credit policy. Status: Charged Off** and **Does not meet the credit policy. Status: Fully Paid** then store the rows in a new data frame (Loan3aDf). And then we merge it with the rest of the files and store it in a csv file(Combined\_data\_file.csv).
* And If the loan status column has values equal to **Does not meet the credit policy. Status: Charged Off** and **Does not meet the credit policy. Status: Fully Paid** then store the rows in a new data frame (Loan3a\_not\_meet\_status\_DF) and store it in csv file(Loan3aDf\_not\_meet\_status.csv)



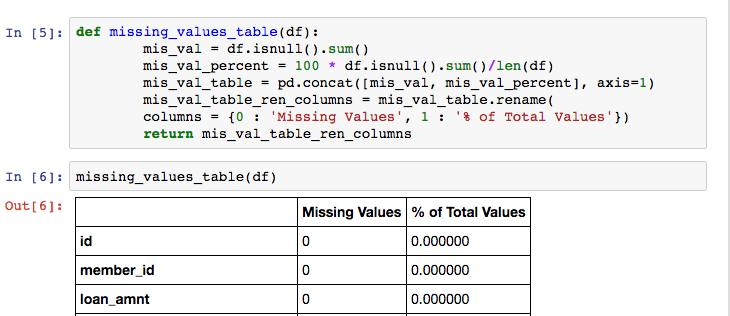
* Once a single dataset is created, we call the validate function to validate all the columns.
* After checking the percentage of null/nans in each column, we dropped a few columns and filled nulls with unknown in categorical columns and interpolation on a few columns.

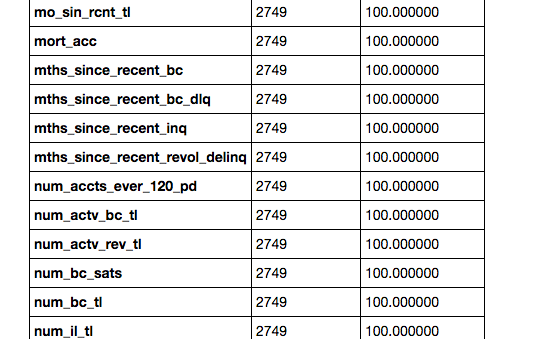
#### add a screenshot for checking nulls

* We removed % from **interest rate** & **revolving utilization** and drop the word **months** from the column term

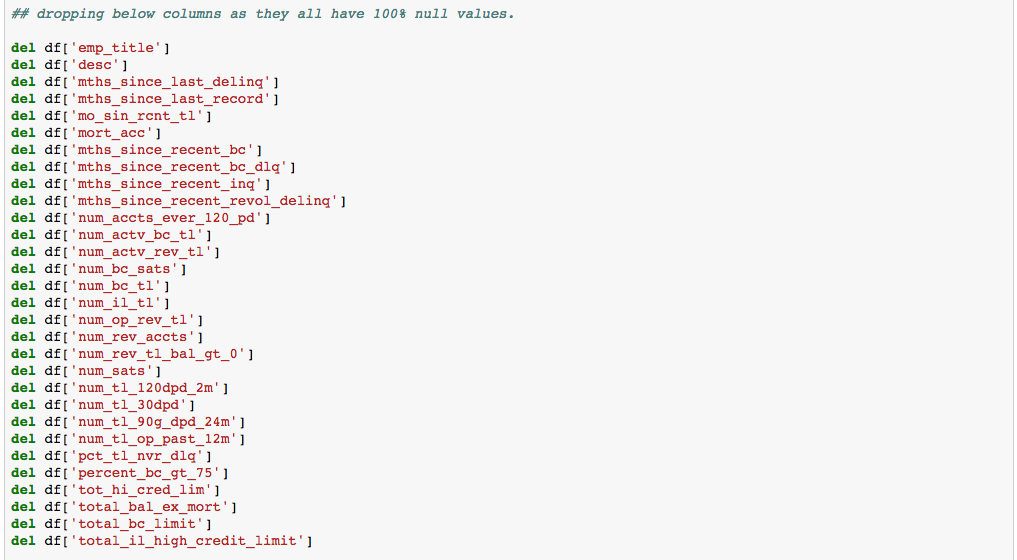


* Validating the file that has loans data that did not meet credit policy.
* After checking the percentage of nulls in each column, we dropped all those columns that has 100% null values.
* Below is the function to calculate percentage of nulls in each column.

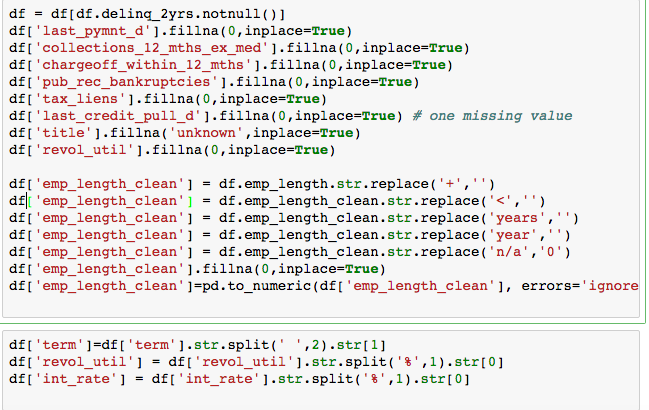




* Below is the screenshot of the function that drops the columns that has null values.



* Once that is done, we cleaned few columns and removed ‘% , + , < , months, years from the column values to make them complete numeric.



15240

**Pipeline:**

We used Luigi to schedule the task.

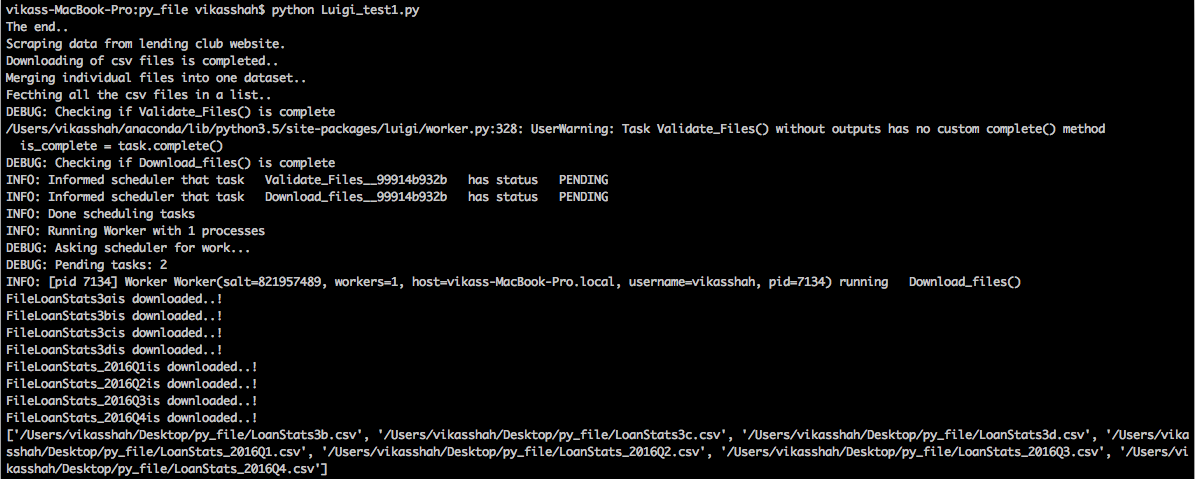


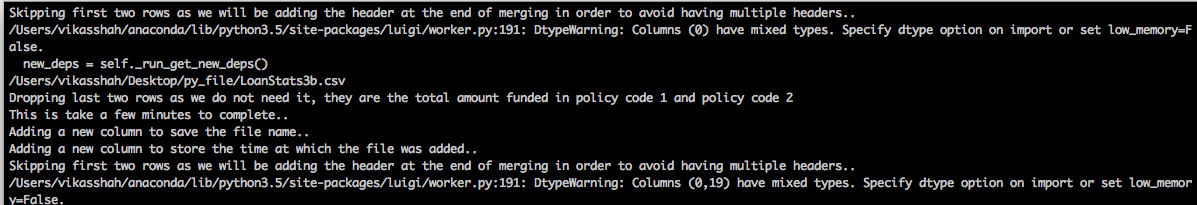
Steps to run Luigi:

To run loan data script: run the below command on the terminal:

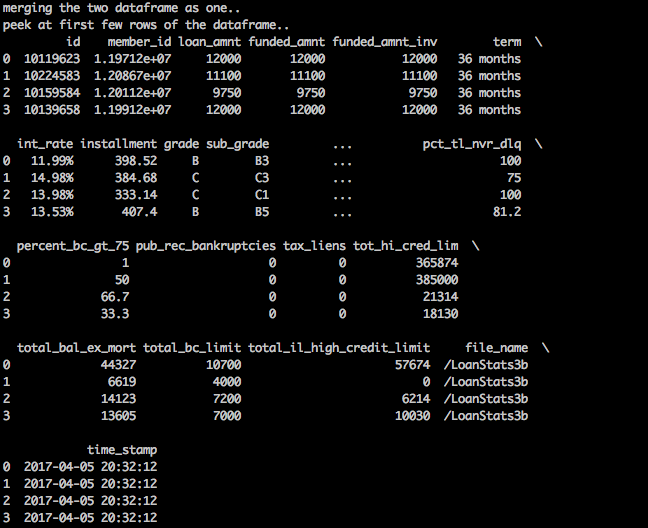
$ python [Luigi\_test1.py](https://github.com/bajajsweta/Lending-Club-Analysis/blob/master/Loan_Data/Luigi_Files/Luigi_test1.py)

Screen%20Shot%202017-04-05%20at%208.36.24%20PM.png

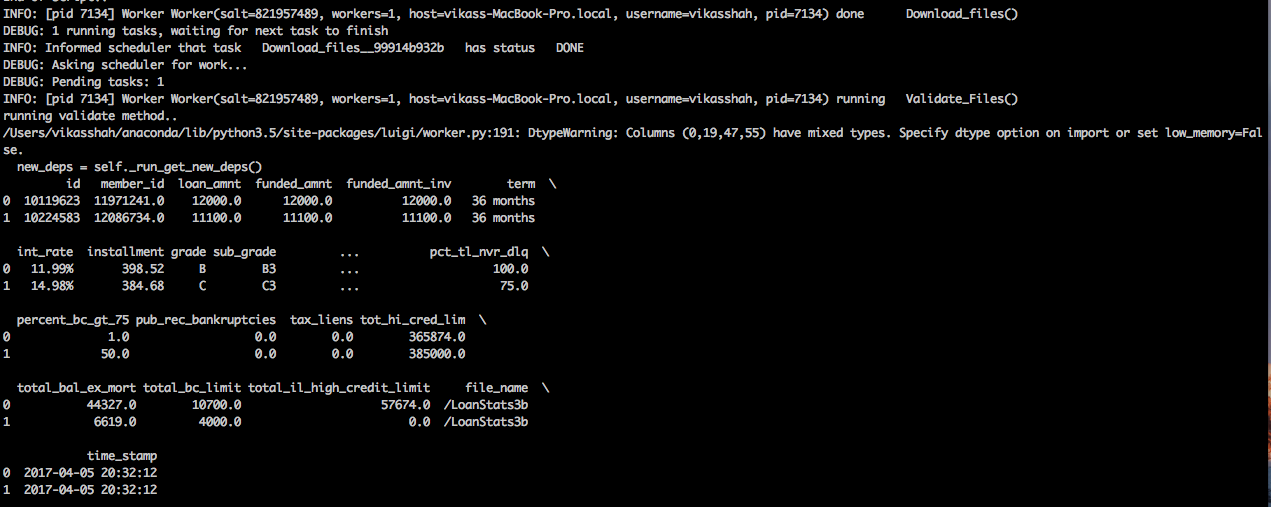




merging data with timestamp and the file name.



the job next runs the dependent job that is the validating the combined data file.



The task completes successfully!

