**Lending Club Analysis**

**Advance in Data Sciences and Architecture**

**INFO 7390 - SPRING 2017**

PROFESSOR:

**SRIKANTH KRISHNAMURTHY**

TEAM MEMBERS:

**HARSHAD PARDESHI (**[**www.linkedin.com/in/harshad-pardeshi/**](file:///C:\Data\Important\Study\ADS\Presentations\Kubernete\kubernets_datascience-master\kubernets_datascience-master\www.linkedin.com\in\aashritandon\)**)**

**PRITHVI KAMATH (**[**www.linkedin.com/in/**](http://www.linkedin.com/in/)**prithvipkamath/)**

**TEJESH RATHI (www.linkedin.com/in/tejesh-rathi/)**

Table of Contents

[Problem Statement 3](#_Toc481448732)

[Part 1: Data wrangling and exploratory data analysis 5](#_Toc481448733)

[Data downloading and storing 5](#_Toc481448734)

[Data cleaning 7](#_Toc481448735)

[Exploratory data analysis in Power BI 17](#_Toc481448736)

[Storing clean data on Amazon S3 bucket 21](#_Toc481448737)

[Part 2: Building and Evaluating Models 25](#_Toc481448738)

[Classification 25](#_Toc481448739)

[Clustering 28](#_Toc481448740)

# **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.

Your first challenge is to programmatically download the data from https://www.lendingclub.com/info/download-data.action

Your goal is to download the data programmatically from the website and create one dataset for the entire database and perform the following tasks:

* Data download: How will you download all loan data and create one dataset
* Missing data analysis: How will you handle missing data?
* Feature engineering: What variables do you need to predict interest rates? Ensure users would be able to give you that information to help you predict rates
* Pipeline: Using Luigi/Pinball/Airflow automate the above 3 steps.
* You need to create one more pipeline to do this for the “Declined Loan data”. Repeat above steps

You should dockerize the whole project (excluding the Power BI dashboard) and write clear instructions on how to run the docker image. Research how you would schedule this on Amazon/Azure/BlueMix so that this image would run and execute the Luigi/Pinball/Airflow pipeline. After running this pipeline, the clean pre-processed data should be stored on S3/Object Storage/Blob storage(AWS/BlueMix/Azure)

* Write a Jupyter notebook using R/Python to graphically represent different summaries of data. Summarize your findings in this notebook.
* Summarize your key insights about different user profiles, states, loan amounts etc.
* Create a Data scientist view of Power BI dashboards to illustrate your key insights

Your next goal is to build a model to predict interest rates. You will get leads from people with different profiles and you must decide if you will give loans or not and if you will give a loan, how much interest you would charge for those loans.

**Classification**

Use the “Loan Data” and the “Declined Loan Data” datasets to build classification models that will generate a flag whether to give a loan or not.

Start with logistic regression using Jupyter and Python/R

Compute ROC curve and Confusion matrices for training and testing datasets and comment on the results.

Repeat this using Random Forest, Neural Network models and SVN algorithms.

Choose one model you will deploy and implement this model on the Microsoft azure machine learning studio and create a REST API

You should be able to a new record (You can define what features you will use) and the result will be a flag whether you would give a loan or not.

**Clustering**

Once you have decided to give a loan, you should build models to decide what interest rate to give. You are debating whether to create one model for all customer prospects or segment data into clusters and then build prediction models specific to each cluster. You think of creating segments or clusters and build models one for each cluster. Your brainstorm with your team and come up with 3 possibilities

1. Segment data into clusters (you define how many) manually using categorical or numerical features. For example, you can segment by state, by ownership of home, by average dti or a combination of features.

2. You use a clustering algorithm (that can factor both numerical and categorical variables) and segment data into k clusters. You will then build prediction models for each cluster.

3. No clusters; Just use data as is

Once you do the clustering use t-sne to visualize your clusters for some sample test data. See http://distill.pub/2016/misread-tsne/ for guidance on using t-sne

**Prediction**

Write a prediction script in a Jupyter notebook in R/Python that builds a Regression model for the interest rate using data from the 3 clustering methodologies you worked.

Try variable selection and build the best model for each segment/cluster (Note: You may have many segments and each model may have different coefficients based on the clusters used to train. You should automate it. Try parallel computing libraries to make things go faster)

o Compute MAE, RMS, MAPE for training and testing datasets

o Repeat this using Random Forest, Neural Network models and KNN algorithms.

o Choose the best model amongst the 4 types of algorithms.

o Deploy the best algorithm/algorithms on Azure ML studio

o You will have a bunch of Rest APIs you should be able to choose from based on the cluster the record belongs to

**Deployment**

Design the following workflow:

Given a record, use a pre-trained clustering model to cluster the record to a segment.

You will have 3 cluster assignments (1-manual, 2-based on your clustering algorithm, 3-default 1- cluster for all data)

For each cluster, there should be a RestAPI which is linked to a chosen prediction model. Look up that API and use it to predict the 3 distinct interest rates.

Select the highest interest rate and return it as your prescribed interest rate.

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

In this section, we will perform the data the following operations:

* Data downloading and storing
* Data cleaning
* Exploratory data analysis in Python
* Exploratory data analysis in Power BI
* Storing clean data on Amazon S3 bucket

## **Data downloading and storing**

Our first step is to get the Lending club dataset from the Lending Club web site. To do this we have two options.

1. To download the data that is available publicly from [lending club site](https://www.lendingclub.com/info/download-data.action).
2. Login to the lending club website by providing all personal details in apply for loan section and then download the complete data set.

We are considering the first option to download both approved loans data as well as rejected loan data. We have gone through the second step to download which requires all your personal details (including your Social Security Number) as the terms and conditions of the site nowhere mentions that they are obliged to not share any personal information to any other third party and does not clearly specify how they will use our data.

The data is divided into multiple files based on the year in which loan was issued/rejected. To download multiple files in python we first need to get the URLs of all the files. For this we use *urllib3* library. File name of all the files is fetched from the a hidden div variable. Hidden div for Accepted loan data is ‘*loanStatsFileNamesJS’* and for rejected loan data is *‘rejectedLoanStatsFileNamesJS’*. This variable contains all the file names separated by ‘|’. From this names we form the URL and download the file using *requests.get(url)* method. Then we extract the downloaded zip files using the *extractall()* function *zipfile* library.

We have implemented this part by writing a single function to download data based on the parameter we pass. To download Accepted loan data we pass ‘loanStatsFileNamesJS’ as parameter and to download Rejected loan data we pass ‘rejectedLoanStatsFileNamesJS’ parameter.

Below is the code snippet of the function.

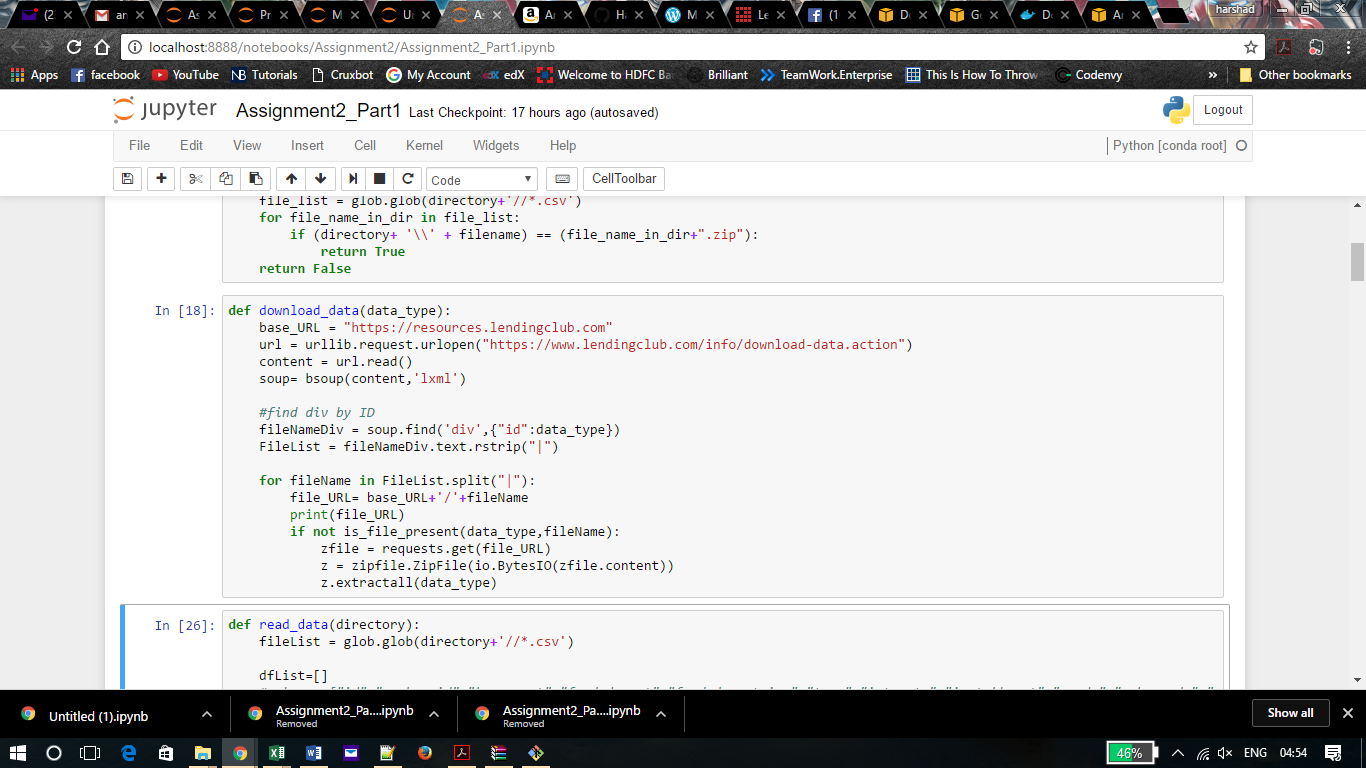


Figure 1: Download Accepted loan and Rejected loan data

Further we read these files in individual Dataframes and add all the dataframes in a list. After reading all the files in list we concatenate all the dataframes to get the combined data.

We have implemented this part by writing a single function to read the data from both loan (accepted/rejected) data and store separately. We call this function with ‘*loanStatsFileNamesJS’* parameter for loan dataset and with parameter ‘*rejectedLoanStatsFileNamesJS’ for rejected loan dataset*.

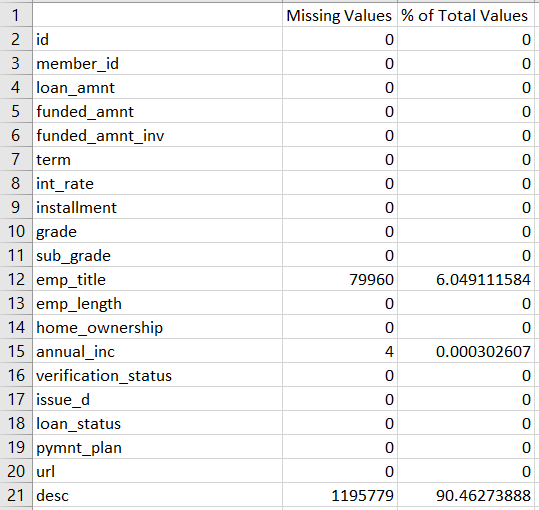
The combined dataframe is then written into a .CSV file.

This entire process is considered as a single Luigi task. This process will give output as the two folders that will contain all the downloaded csv files.

## **Data cleaning**

This section comprises of handling missing data. We get the concatenated .CSV file from the previous part, on which we perform data cleaning steps.

We first analyzed all the columns having empty values as shown below:



We have implemented this by writing a function, which gives us a .CSV file having the count of null values and percentage of empty rows for each variable.

We further moved on to feature selection and found columns which do not have any impact on the prediction of interest rates. If some of these columns were null, we directly removed the column as they were of no use in determining the interest rates.

We have also ignored variables with 70% and above null values.

**Columns deleted:**

* emp\_title
* desc
* last\_pymnt\_d
* next\_pymnt\_d
* last\_credit\_pull\_d
* url
* desc
* title
* emp\_title
* chargeoff\_within\_12\_mths
* mths\_since\_last\_record
* mths\_since\_last\_major\_derog
* open\_il\_12m

As part of feature selection, we are selecting all features except the one mentioned above.

Another document named “FeatureSelection.docx” has been added for better explanation.



**Columns Updated and cleaned:**

* **emp\_length** – As per the Data Dictionary, emp\_length should have only values ranging from 0 to 10, whereas there were few records with value of “10+ years” and “<1 year”, which we replaced with 10 and 0 respectively.
* **Title** – Title is the reason for a person to borrow money. There are various categories based on different reasons provided by the users; since blank means they haven’t specified the reason for borrowing the money, we assumed that it was something they didn’t wanted anybody to know and had some personal reasons, so we assigned them to the “personal” category.
* **mths\_since\_last\_delinq** – This field shows the number of months passed since a person was marked delinquent. Here, we made use of another column which is correlated to this column i.e. delinq\_2yrs. If the value in delinq\_2yrs = 0, that means there has been no delinquencies recorded for a person in the last 2 years. So, making use of this analysis, we replaced all the empty data of mths\_since\_last\_delinq which has delinq\_2yrs as 0, as 24 (resembling 2 years).
* **mths\_since\_last\_record** – This field indicates the number of months passed by since a person had any public record. This value is provided by the Lender’s based on the data they have. All public record is noted and present in their database. Since, the field is left blank means, that a person doesn’t have any public record yet. So, we marked it as -1 to show that a person doesn’t have any public record.
* **annual\_inc** – Annual income is empty just for 4 records in the data. On analyzing we found that all these 4 records don’t have any employer, which means that they don’t earn so we replaced all blanks by 0.
* **delinq\_2yrs** – delinq\_2yrs column has maximum of 39 and minimum of 0. On reading the data, we found that all the empty columns are for people who are not valid and does not meet the credit policy, so we replaced them by the mean.
* **revol\_util –** These are the percent values, so we first stripped all the “%” and then replaced it by the mean percentage and processed it in a new column “derived\_revol\_util”.
* **delinq\_amnt** – 99.98% of the data is zero. Only negligible records have any other values, so we replaced it by zero in a new column “derived\_delinq\_amnt”.
* **pub\_rec\_bankruptcies** – All the empty rows are the ones which do not meet the credit policy and all are non-verified. With only negligible blanks, we replaced it by mean.
* **tax\_liens** – These are the same 29 empty records as pub\_rec\_bankruptcies and so we processed a new column as “derived\_tax\_liens”.
* **Interest\_rate** – Stripped the “%”, so that analysis can be done on top of that and put the values in a new column “derived\_interest\_rate”

**Data pre-processing**

We are excluding the below rows that summarize the dataset.

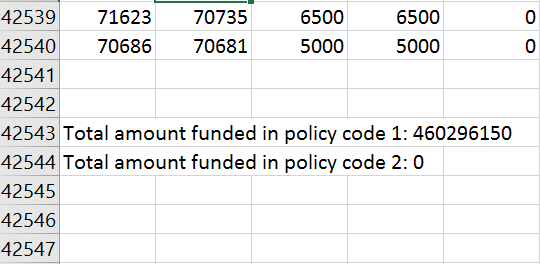


Figure 2: Summary rows that are excluded from the dataframe.

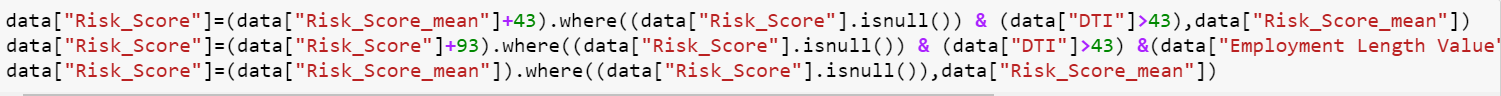
Converting ‘issue\_d’ to datetime format using pd.to\_datetime

We are creating new columns in the dataframe to store the below values:

* Interest rate value as Interest\_Rate
* Issue month as issue\_month
* Issue year as issue\_year

In Rejected Loan Dataset

* For the rejected file, we observe that comparable number of observations have Null in their Risk Score.
* We are filling the null values by considering the Debt-To-Income-Ratio, Employment Length and the mean Risk Score for the individual’s state.
* We have learnt that Debt-To-Income-Ratio above 43% leads to higher risk score and hence leads to loan rejection.
* Hence for employers, who don’t have Risk Score we have replaced them by the addition of mean Risk Score for the state and increased it by 43 if Debt-To-Income-Ratio is above 43%.
* We have also added 50 points to the risk score if the employment length of an individual is less than the average employment rate for that state.
* Hence the lower limit for the value would be the mean of the state-wise risk score and higher limit would be mean of state-wise risk score + 93



We are deriving two variables:

* Lending\_Club\_Interest: This is the amount Lending Club makes on every transaction. It is calculated by subtracting the amount paid back to the investor from the total amount paid by the loan borrower.
* credit\_age: This variable shows the duration since when the user is using credit. It is calculated by subtracting the first date when credit was checked from the date when credit/loan was issued.

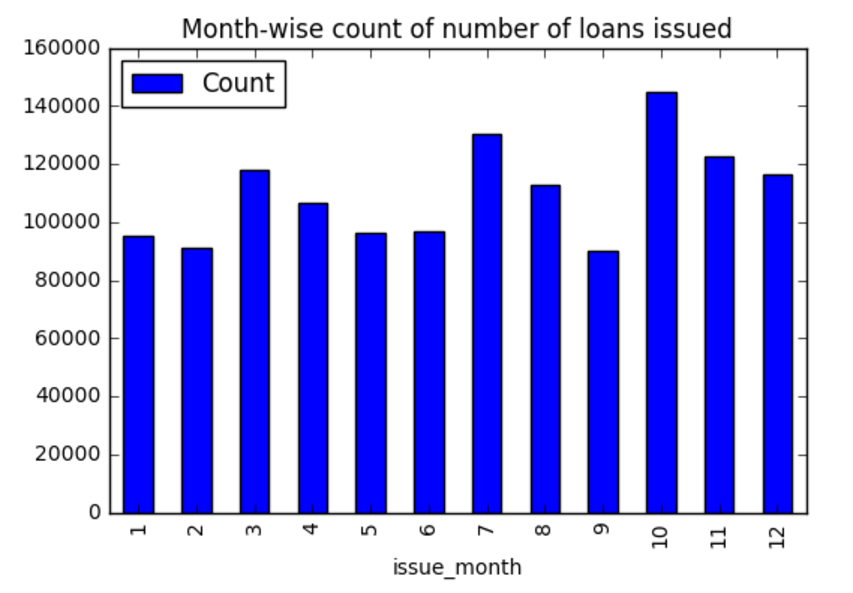
**Exploratory data analysis in Python**

**Analysis 1:** Find the month-wise count of number of loans issued

**Approach:**

Group the above dataframe for every month and count the number of loans issued in each month.

The result is saved in Analysis\_1.csv and a graph is drawn which represent this flow.



**Conclusion:** This graph shows us the month wise distribution on the number of loan issued.

**Analysis 2:** State-wise count of number of loans issued

**Approach:**

Group the dataframe for every state and count the number of loans issued in each state.

The result is saved in Analysis\_2.csv and a graph is drawn which represent this flow.

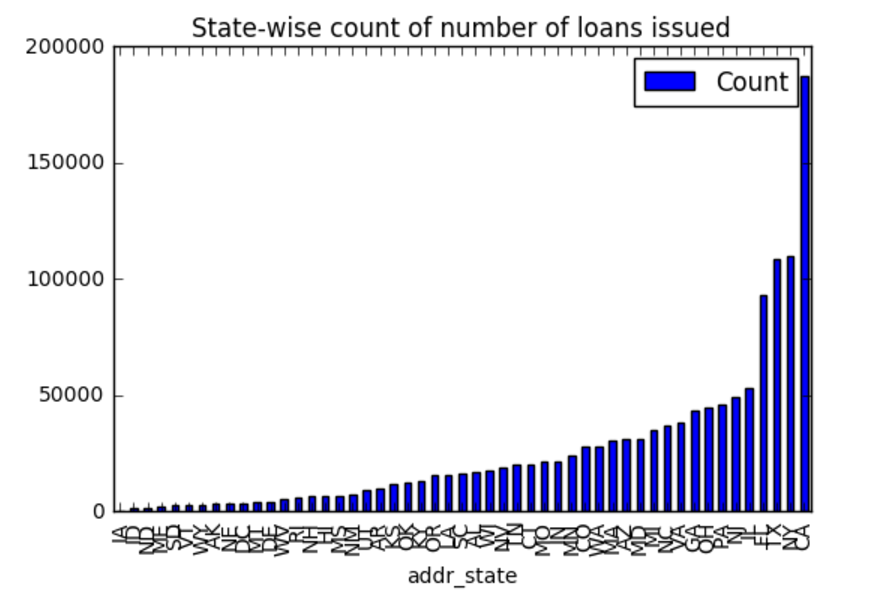
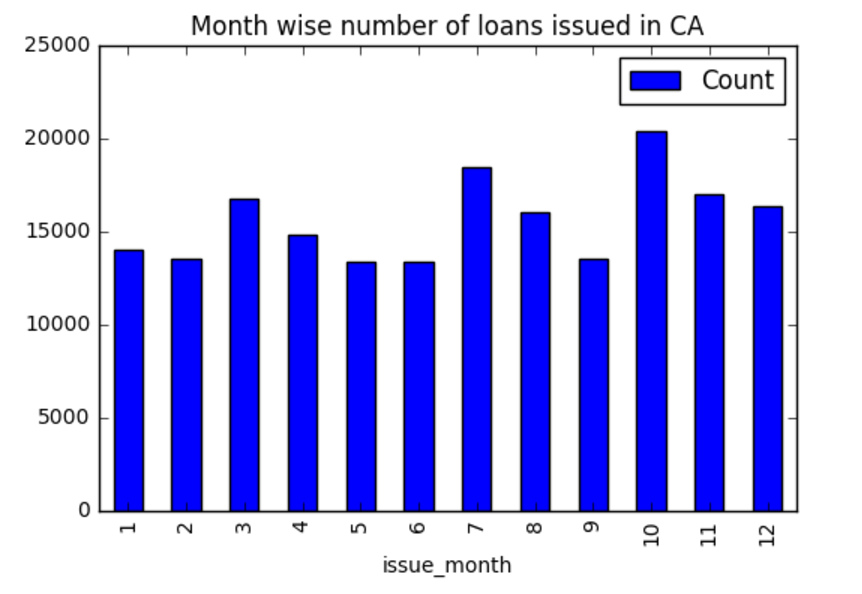
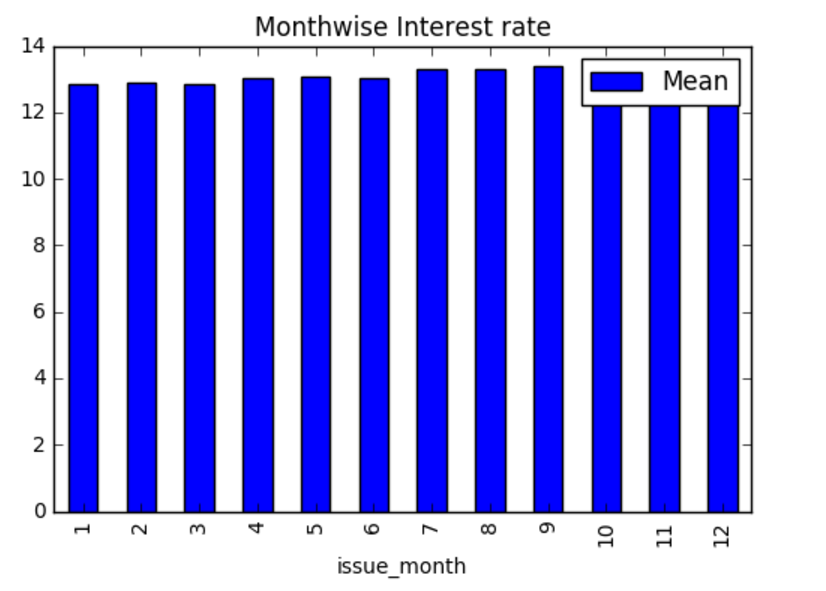
****

Figure 3: State wise loan issued distribution.

Here we see that CA has the highest count compared to other states and hence we dive into CA state issuers. On getting a month-wise graph, we observe that it is similar to the overall month-wise distribution.



Since there is no anomalies or sudden rise or fall in the graph we extract the interest rate in CA for each month.



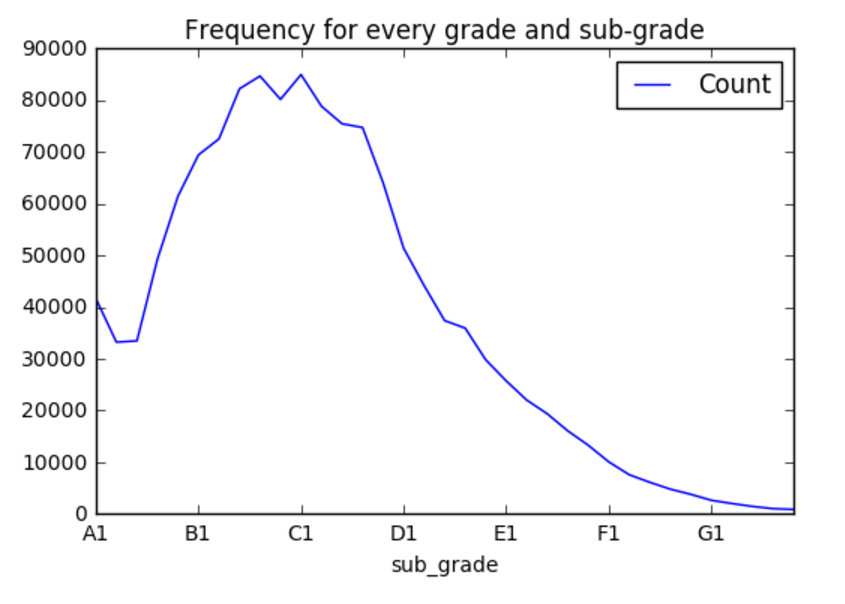
**Conclusion:** We observe that even though the interest rate is increasing over the months, there seems to be a rise in the number of people who are issued loans. This concludes that the Lending Club is targeting the right market in CA.

**Analysis 3:** Find the frequency for every grade and sub-grade

**Approach:**

Group the above dataframe for every grade and sub-grade and count the number of loans issued.

The result is saved in Analysis\_3.csv and a graph is drawn which represent this flow.

****

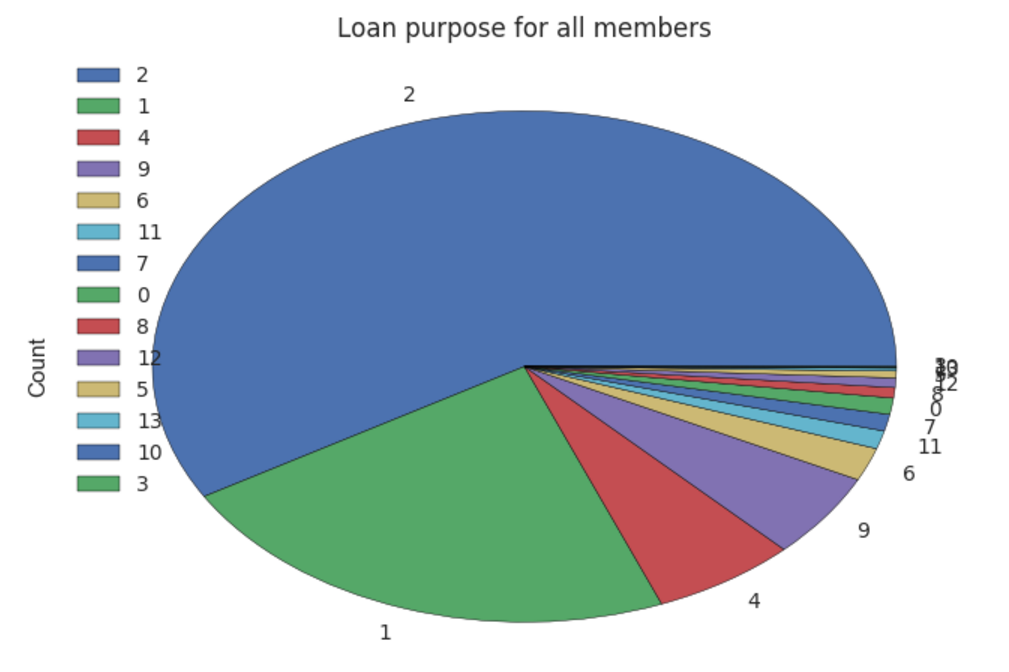
**Conclusion:** We observe that the maximum number of loan issuers lie in grades B3, B4, B5, C1 and C2.

**Analysis 4:** Find the loan purpose for all members

**Approach:**

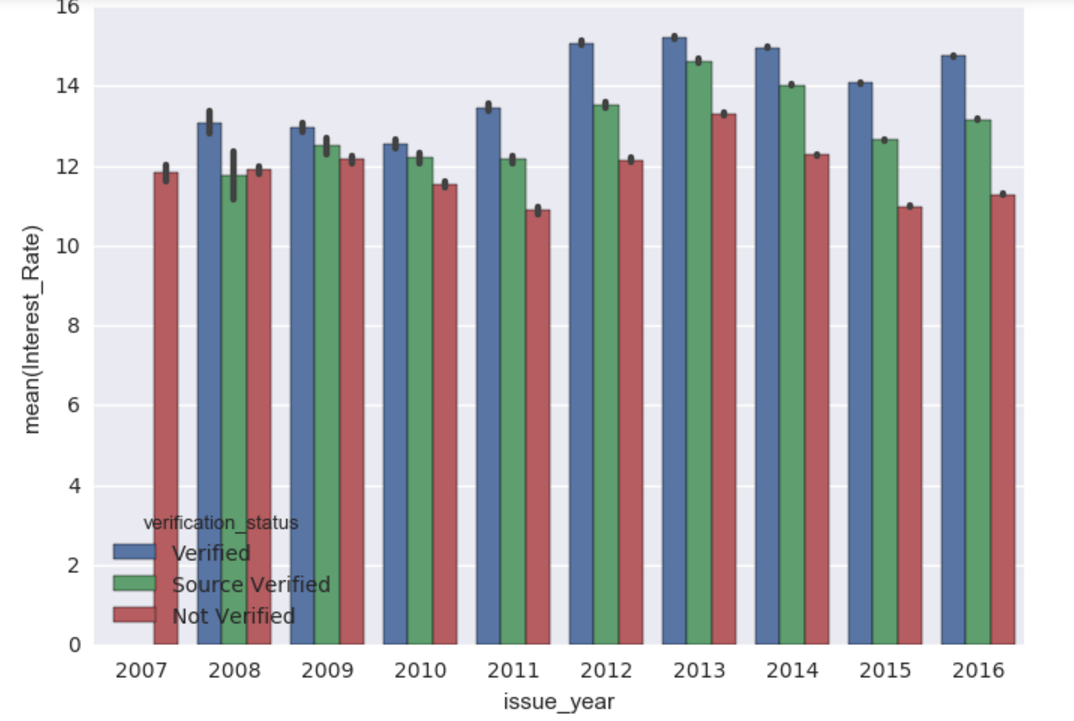
Group the above dataframe for every purpose and count the number of loans issued.

The result is saved in Analysis\_4.csv and a graph is drawn which represent this flow.



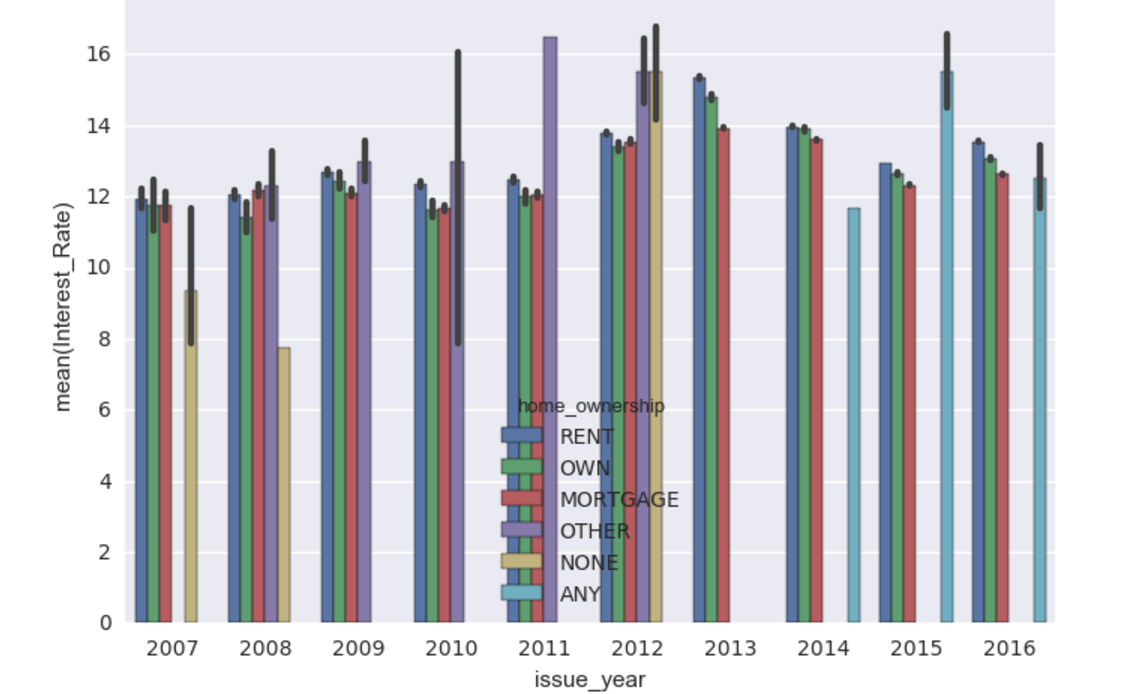
**Conclusion:** We observe that the most mentioned reason for issuing a loan is debt consolidation which is referred by the number 2.

**Analysis 5:** Find the mean Interest Rate for every Verification status over all the years.

****

**Conclusion:** We observe that the difference between various groups of Verification Status grows over the years from 2007 to 2016. Hence we can conclude that the verification team is working efficiently and processing more

**Analysis 6:** House ownership status for all members

****

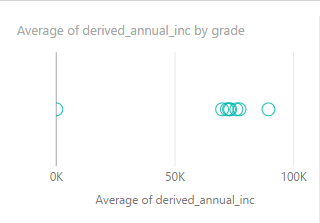
**Conclusion:** This graph represents the home ownership distribution for each year from 2007 to 2016. As the mean interest rate for people renting, owning and mortgaging a house is almost the same across all the years, we can conclude that this attribute does not make a huge difference to the credit score and interest rate.

## **Exploratory data analysis in Power BI**

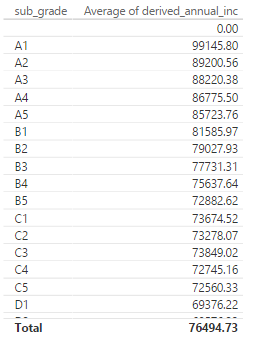
Once we have extracted the data, cleaned and pre-processed it, analyzed the data, completed the feature selection we come to the final task of making the data insights visually appealing using PowerBI.

Analysis is done for two reasons –

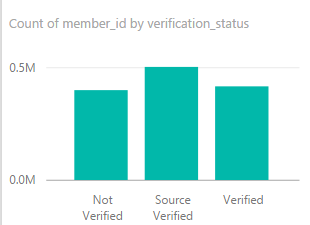
1. to get some favorable insights
2. for statistical purpose
3. Average annual income based on the Grade a person belongs to.
   * This gives the data of the people having an average salary based on the Grades they belong. By looking at the data, we get to know that only Grade A people are class apart and have way better salary than any other Grades. Any Grade other than A, has more or less a similar average salary.



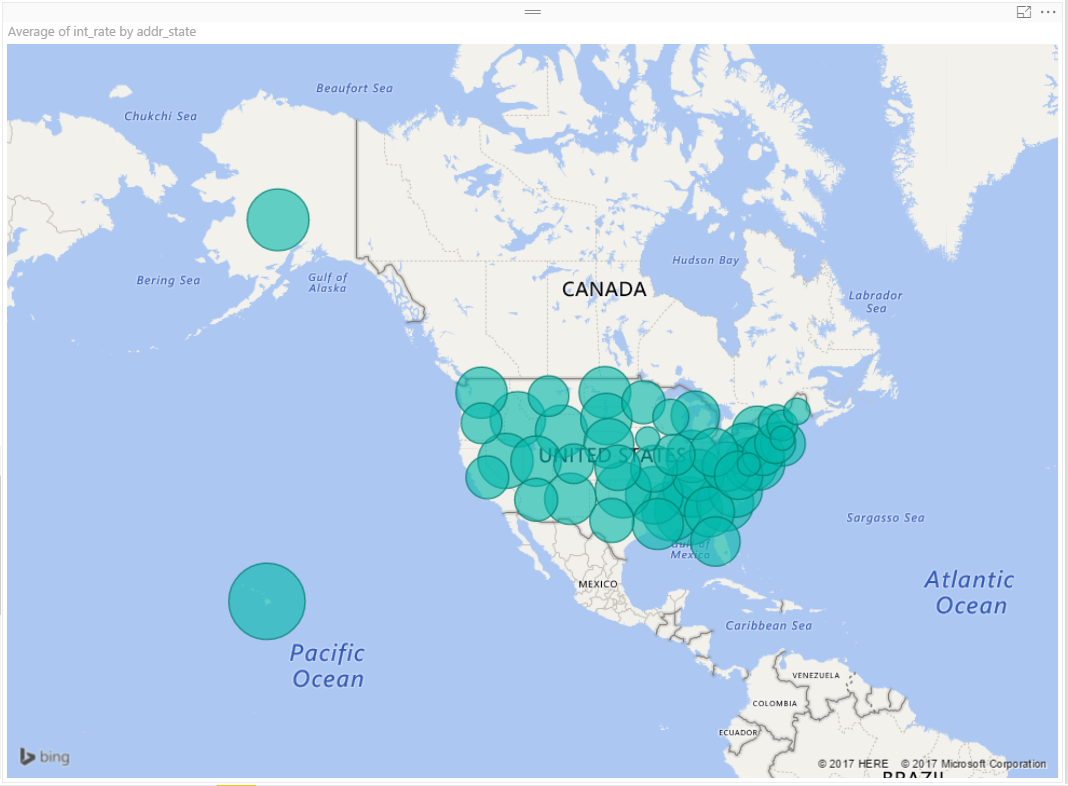
1. Getting even deeper and drilling down the hierarchy, the average salary based on the sub-grades of an individual.



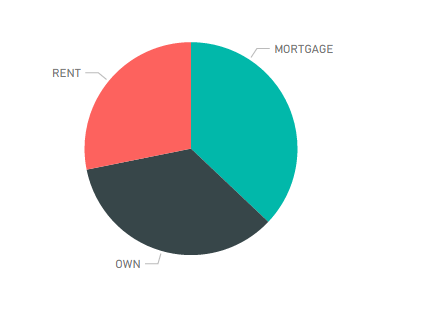
1. Statistical data on count of verified, not verified and Source verified individuals.



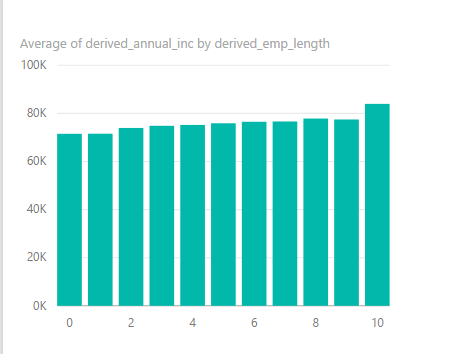
1. The average interest rate based on each state. On visualizing the data from the given link, we concluded that the closer the western part of the States has comparatively less interest rates.



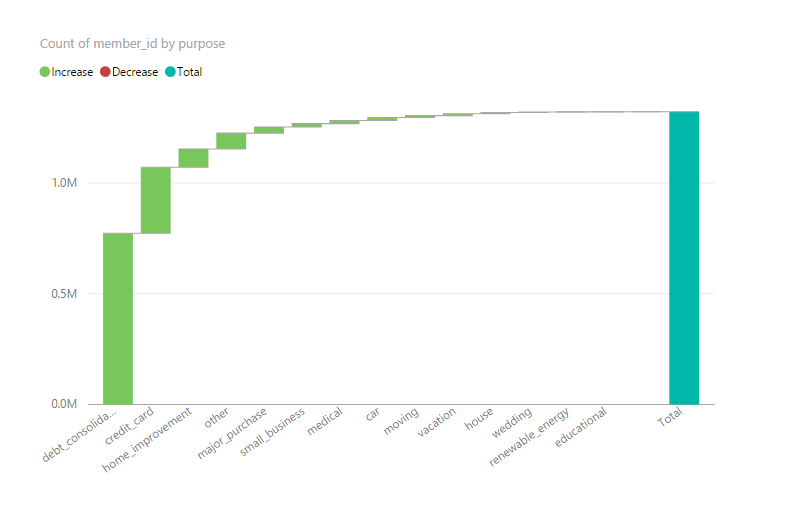
1. The average employee work length to acquire different types of home ownership.



1. Salary to number of years a person has worked. This shows that salary increases with amount of experience.



1. This analysis is probably the most interesting which helps us find the most popular reason for a person to land into Lender’s Club to borrow money.



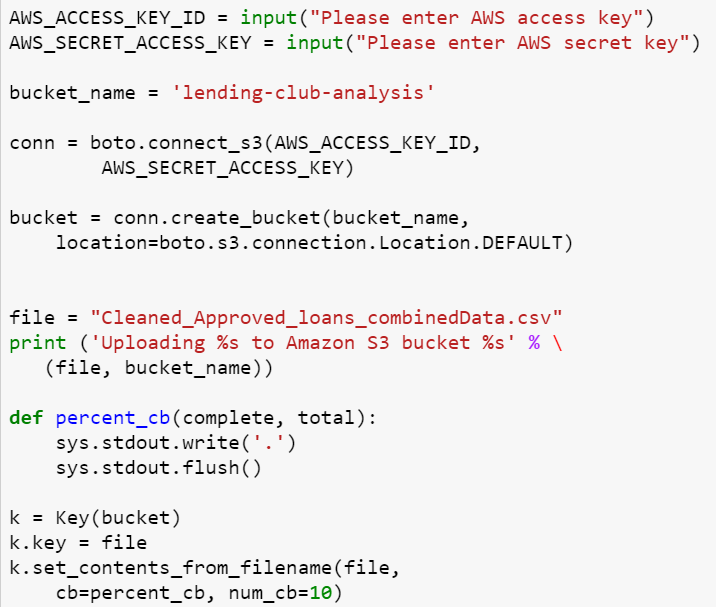
PowerBI dashboards are available in the below link:

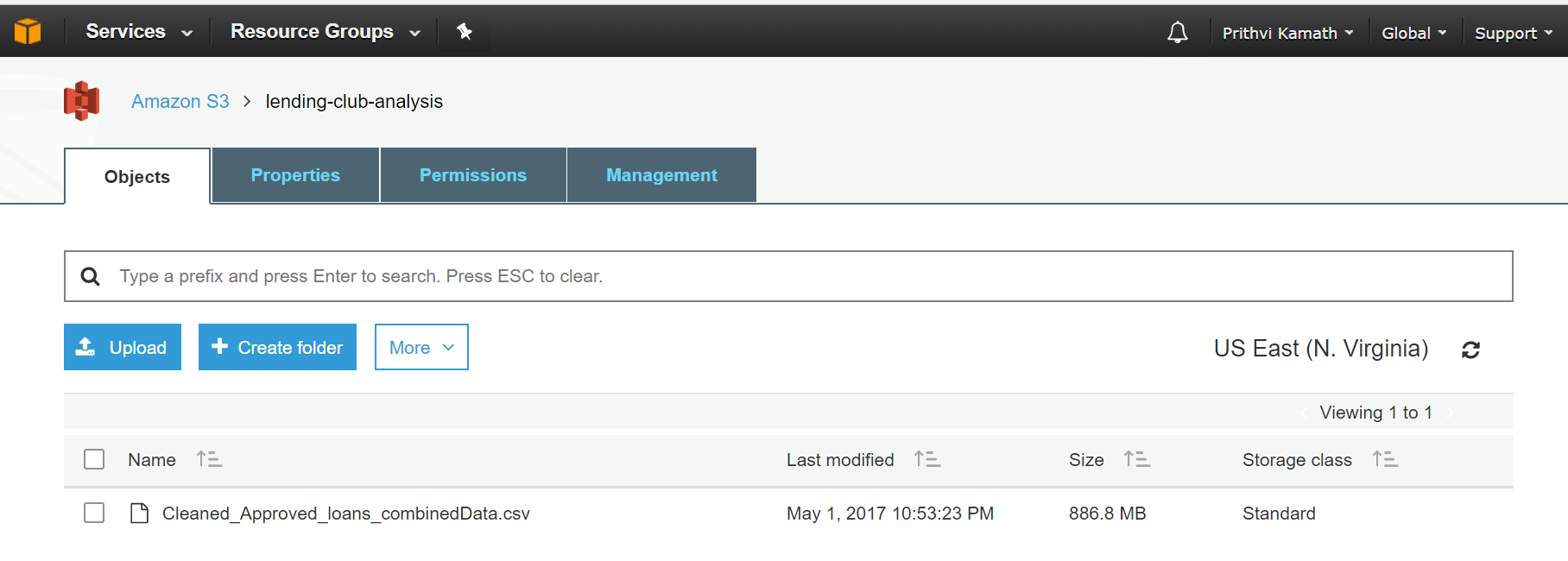
<https://app.powerbi.com/groups/me/reports/6a3b7b14-1a9c-44b2-8cf5-9a7db45bb3bf/ReportSection>

## **Storing clean data on Amazon S3 bucket**

Once data cleaning and preprocessing is complete we will push the final clean file to the Amazon S3 bucket. We are using *boto* library to archive this task. We first get the Amazon S3 connection using the AWS\_KEY and AWS\_SECRET\_KEY. These two keys are passed when liuigi scrip is called.

After getting the connection se simply create a new bucket with team name and assignment2 (team-10-assignment2). After this we simply push the clea file to S3 bucket





**Scheduling and pipelining**

For pipelining the process, we have used Luigi that will run the series of tasks. We have created 4 tasks for each pipeline:

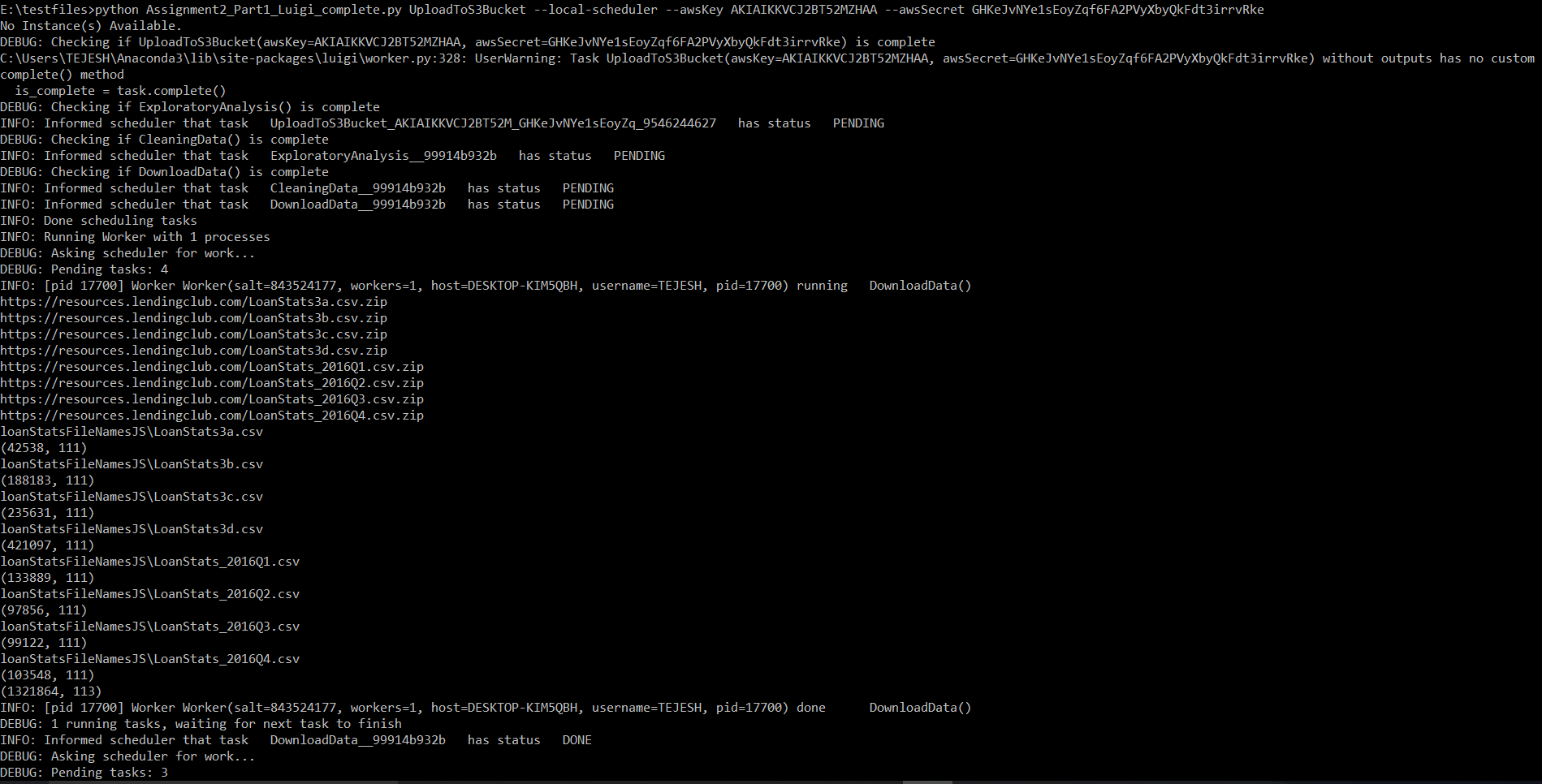
1. DownloadData
2. CleaningData
3. ExploratoryAnalysis
4. UploadToS3Bucket

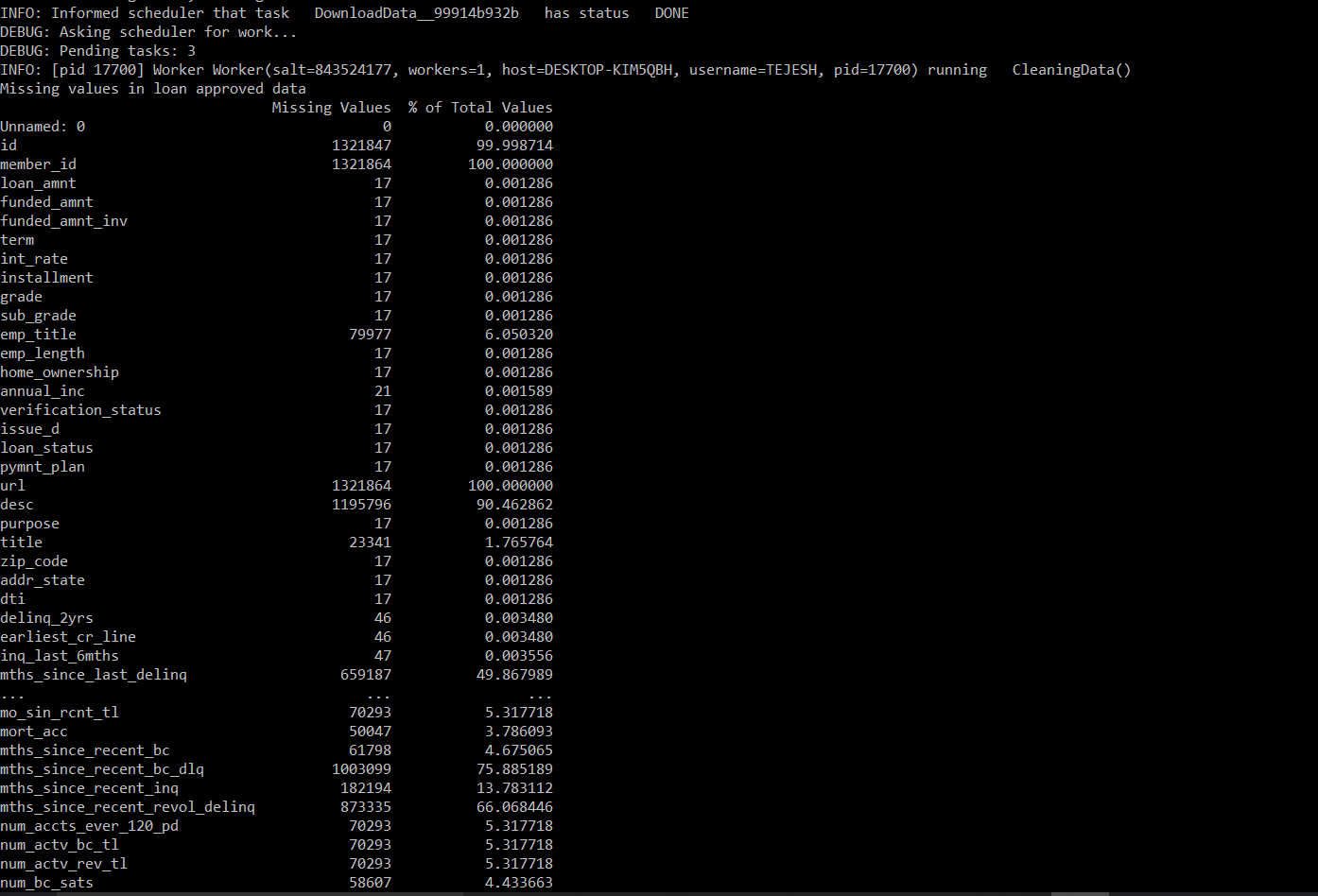
Output of each task is an intermediate processed file which is the output the next task.

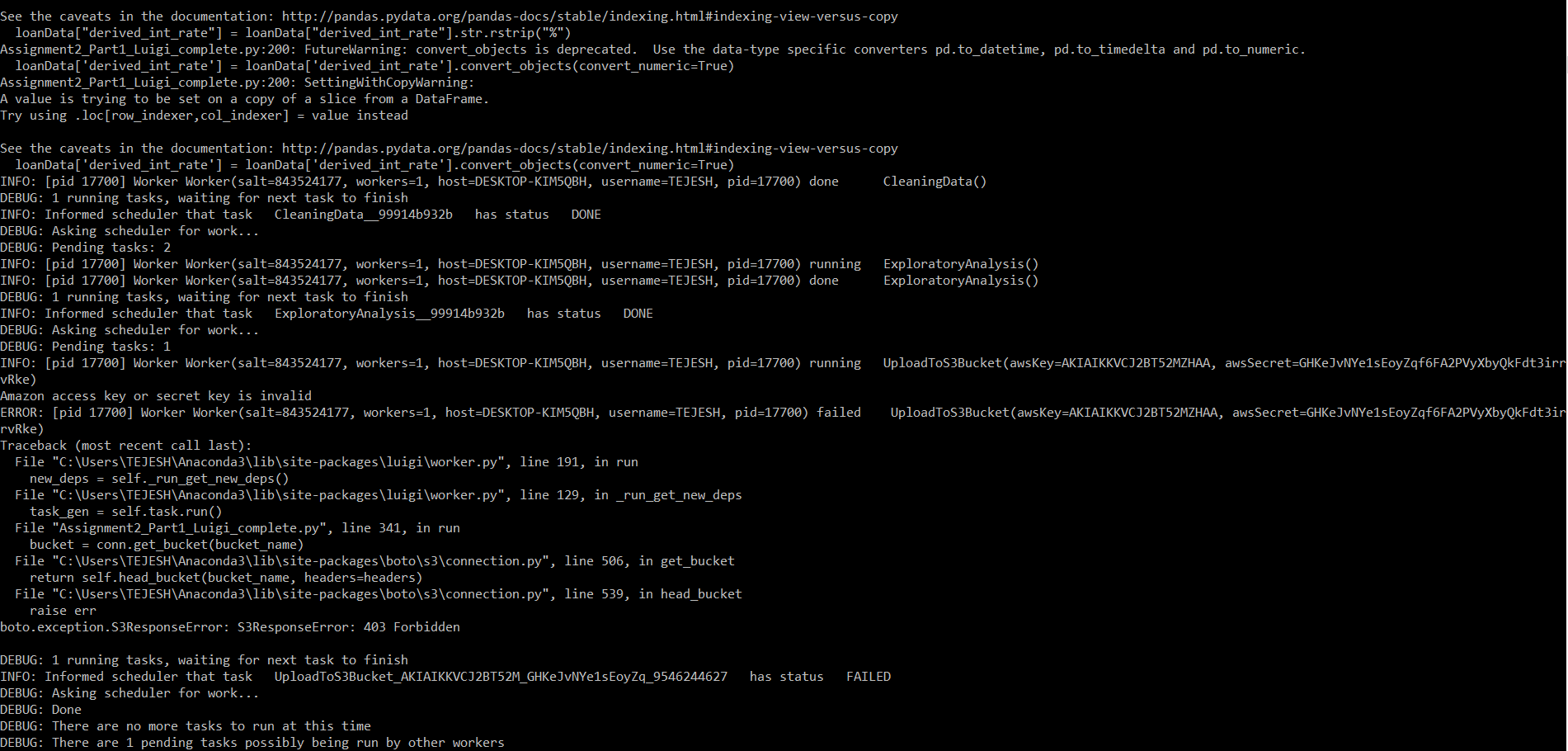
The final task does not have any output as it does not have any other task dependent on it. It directly uploads the data to S3 bucket.

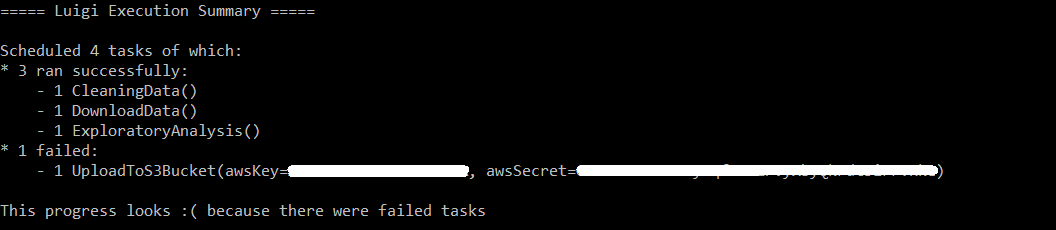
**Luigi Execution:**

We ran the entire process using Luigi workflow. Please find below the screenshots of the same showing the full execution of Luigi.









Steps followed to perform scheduling on Amazon AWS:

* Created an EC2 instance.
* Installed Docker on EC2 instance.
* Created repositories that can hold docker images on AWS.
* Created Task Definitions to run our pipeline.
* Created scheduling tasks to run the above task definitions.

Steps to be followed to replicate the above assignment:

docker pull harshadpardeshi/assignment2

docker run -ti harshadpardeshi/assignment2

**Part 2: Building and Evaluating Models**

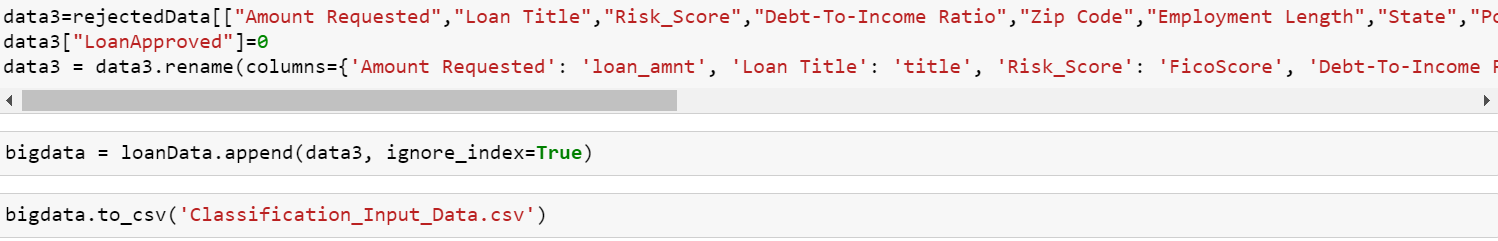
**Classification**

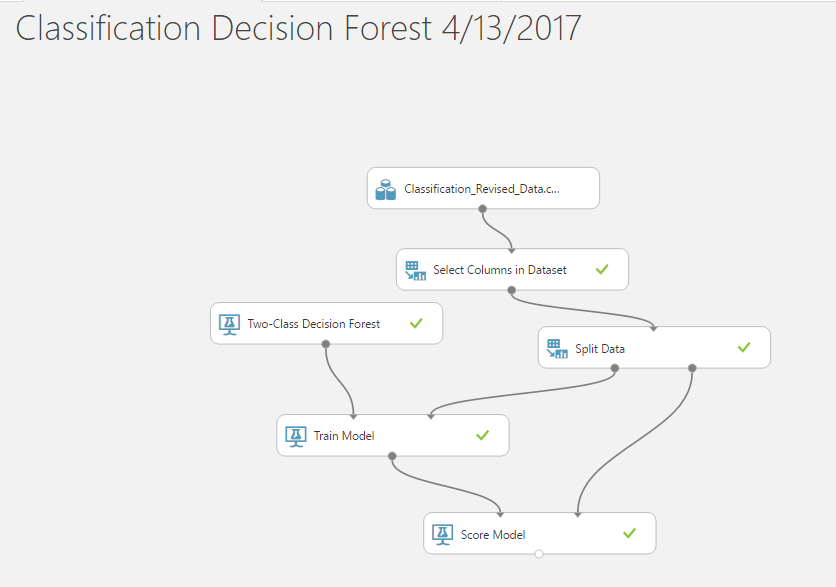
Our first step in classification is to obtain a dataset which has accepted and rejected observations.

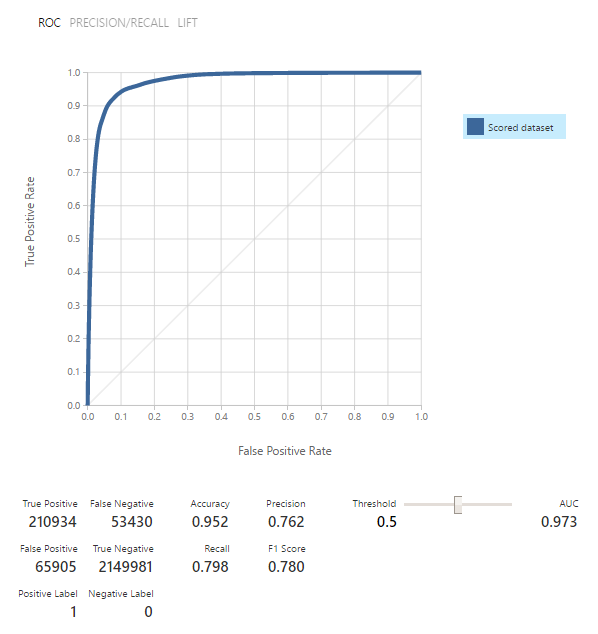
To get this combined data we are concatenating loan accepted data with the rejected data based on common columns in both the files. The selected columns are as follows:

|  |  |
| --- | --- |
| **Accepted Loan Data** | **Rejected Loan Data** |
| Loan\_amnt | Amount Requested |
| title | Loan Title |
| FicoScore (derived attribute) | Risk\_Score |
| dti | Debt-To-Income Ratio |
| zip\_code | Zip Code |
| emp\_length | Employment Length |
| addr\_state | State |
| policy\_code | Policy Code |
| LoanApproved | LoanApproved (derived attribute) |

Below is the code snippet for the same.







Based on the combined file of rejected and accepted records, we trained our model based on few variables, whether to give loan to a person or not. We took the common columns between both the rejected and accepted files, combined it together and moved the data with policy code 2 into the rejected file, as they will not be getting the loan henceforth.

Among the 8 columns, we figured out few columns which are most relevant and gives us higher accuracy of output based on backward selection. We then trained those 8 columns and predicted whether a person should be given loan or not.

We implemented various classification algorithms like Logistic Regression, Random Forest, KNN, Neural Network using iPython Notebook.

Among these, we figured out the best model based on the Accuracy rate we obtained.

Random Forest – 95.2%

SVN – 92.8 %

Logistic Regression – 92.8%

Neural Network – 94.9%

As per the accuracy rates, Random Forest has the highest accuracy and hence that is our best model for Classification. We then implemented Random Forest on Microsoft Azure Machine Learning and the output is as shown in the above image.

**Clustering**

In this, we finalized 13 variables to train the data. The 13 features of our model are as below:

sub\_grade,

derived\_int\_rate,

annual\_inc,

Derived\_term,

FICO,

derived\_emp\_length,

addr\_state,

loan\_amnt,

verification\_status,

home\_ownership,

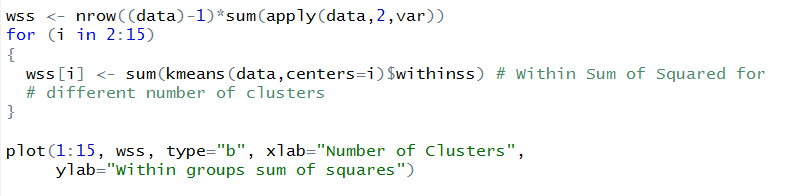
derived\_mths\_since\_last\_delinq,

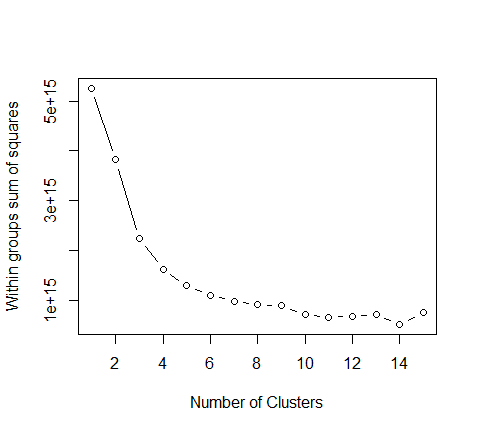
issue\_year,

application\_type

**Algorithm Clustering**

We have implemented Bend Graph to get the number of clusters using the below code snippet.



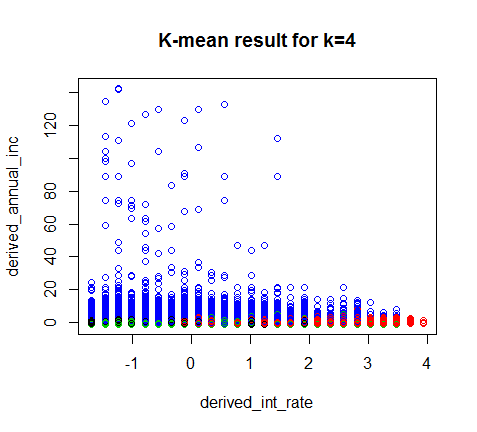
Our bend curve is as follows:

From this Bend Graph, we understand that the line seems to be flat after 4. Hence we chose 4 clusters to implement our clustering algorithms.

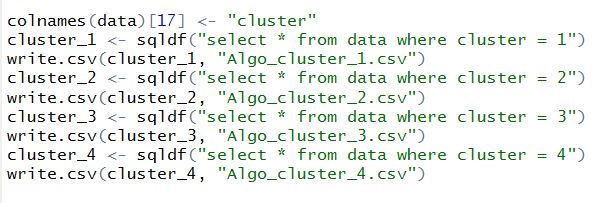
The Clustering algorithm we have chosen is KMeans. To implement KMeans algorithm we have converted the selected columns to integer format and scaled them.

Then we have plotted a scatter plot to show the distribution across each cluster.

The scatter plot is as follows:

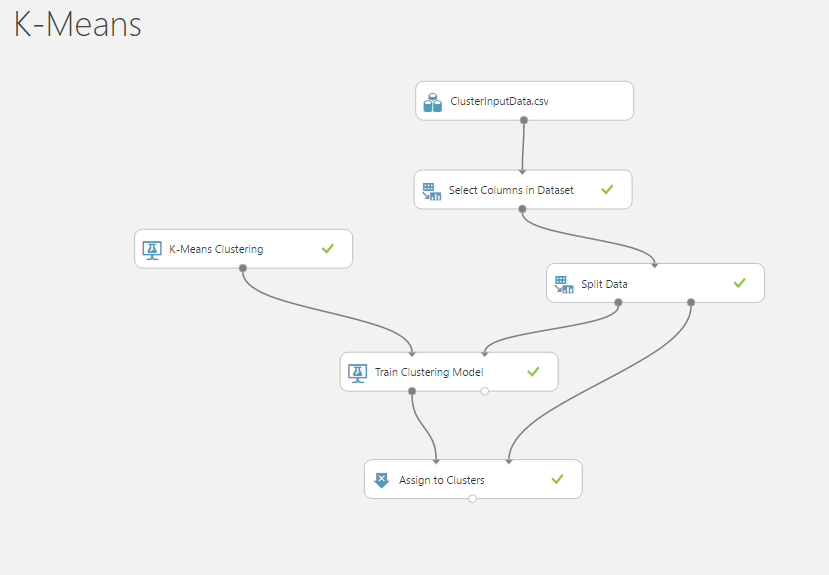


Once we get the scatter plot, we use the cluster information and append it for each observation in the input dataframe and create files for each cluster. These files are cluster specific and are used as input to the prediction algorithms that run on each cluster.



The following prediction algorithms are implemented on each cluster:

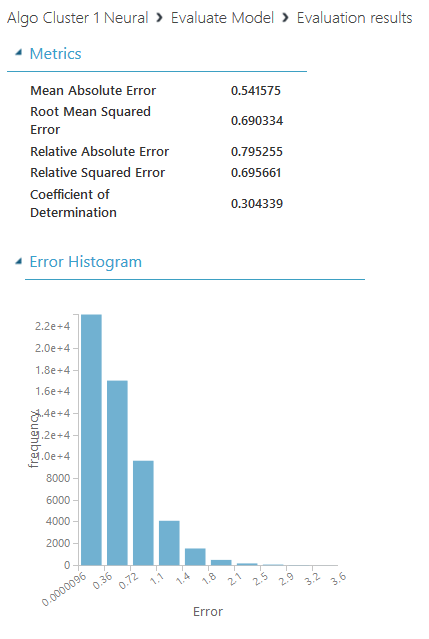
* Linear Regression
* Random Forest Regression
* Neural Network

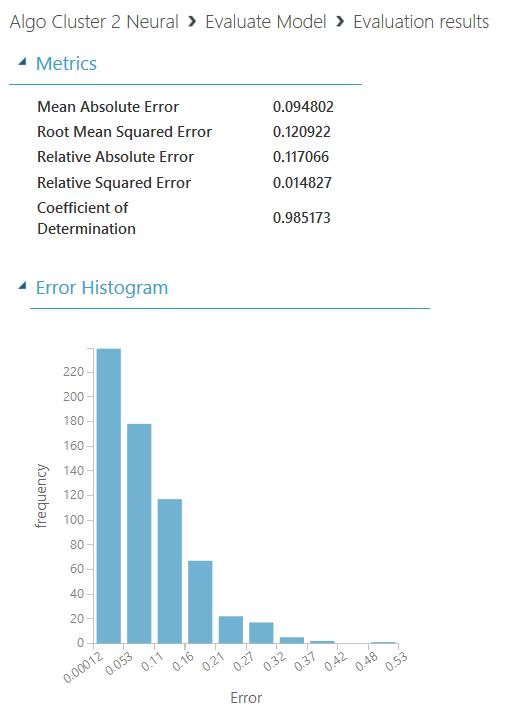


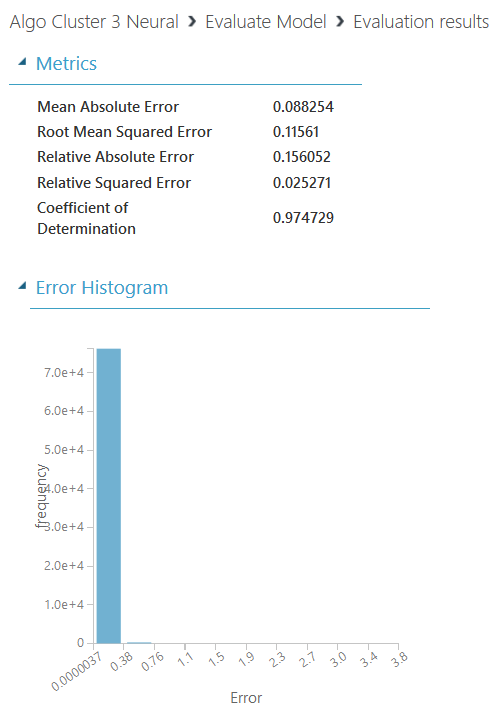
K-Means is a clustering method where in the data is divided into k number of clusters, in our case it is 4 clusters.

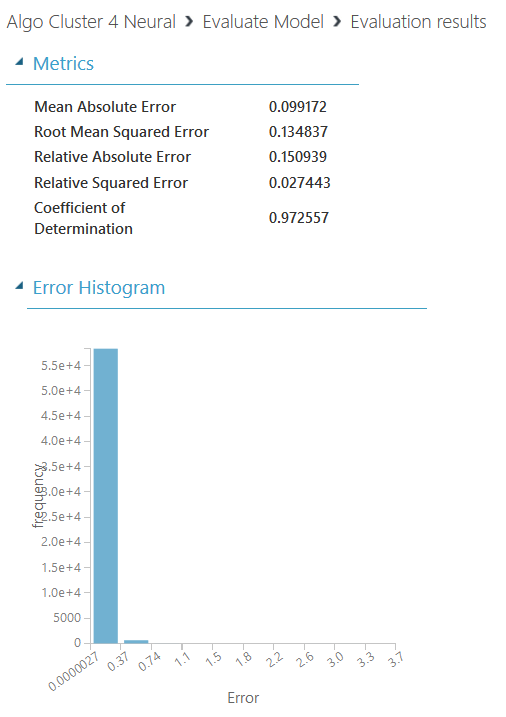
The above model shows the implementation of k-means in Microsoft Azure Machine Learning. This divides the data into 4 clusters, based on the user input in the Web UI created.

We have selected Neural Network Regression to be our best prediction model.

****







**Manual Clustering**

We are dividing the entire dataset into 3 clusters while implementing manual clustering based on the below factors:

* State
* FICO Score
* Employment Length
* Term

For each of the above factors, we have grouped each observation into 3 groups and each group has been assigned a specific weight. Each observation is later assigned a value based on the range of these weights.

Observations with weights 0-5 form the first cluster.

Observations with weights 6-9 form the second cluster.

Observations with weights 10-12 form the third cluster.

Below is the explanation on the distribution of weights for each observation.

**State Distribution:**

States have been divided into 3 groups based on cash solvency distribution.



|  |  |
| --- | --- |
| State | Weights |
| State1 | 1 |
| State2 | 2 |
| State3 | 3 |

**FICO score:**

Observations with FICO score in the range 300-449, 450-649 and 650-850 have been divided into 3 groups.

|  |  |
| --- | --- |
| FICO | Weights |
| 300-449 | 1 |
| 450-649 | 2 |
| 650-850 | 3 |

**Employment Length:**

Observations with Employment length 0-3, 4-7 and 8-10 have been grouped together into 3 groups.

|  |  |
| --- | --- |
| Employment Length | Weights |
| 0-3 | 1 |
| 4-7 | 2 |
| 8-10 | 3 |

**Loan Term:**

We only have 2 values for loan term, but we have divided them into 3 groups.

Observations with loan term as 36 months are clubbed together.

Observation with 60 months are further divided into 2 groups based on FICO score.

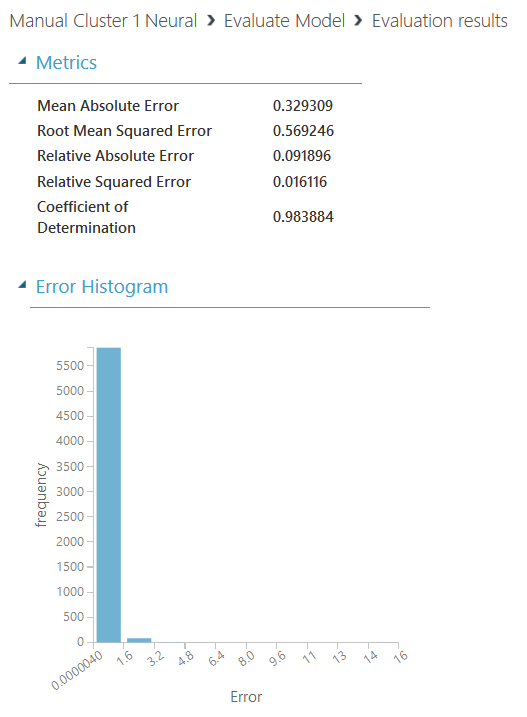
|  |  |
| --- | --- |
| Loan Term | Weights |
| 36 months | 1 |
| 60 months and FICO score 450-649 | 2 |
| 60 months and FICO score 650-850 | 3 |

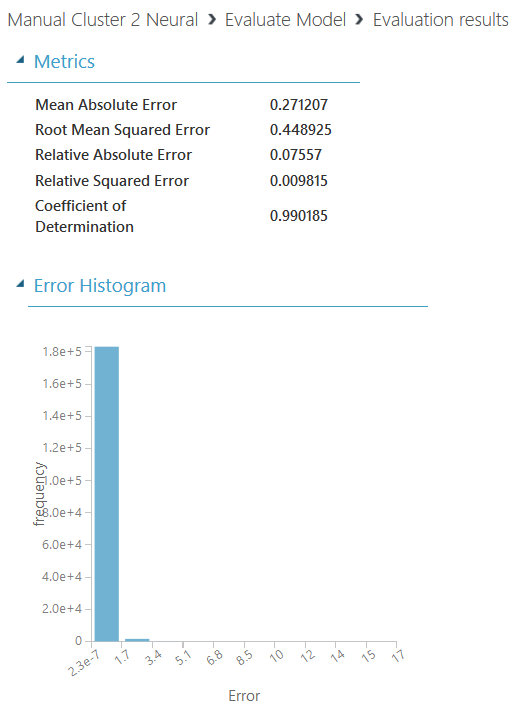
Below prediction algorithms are implemented on each of these 3 clusters.

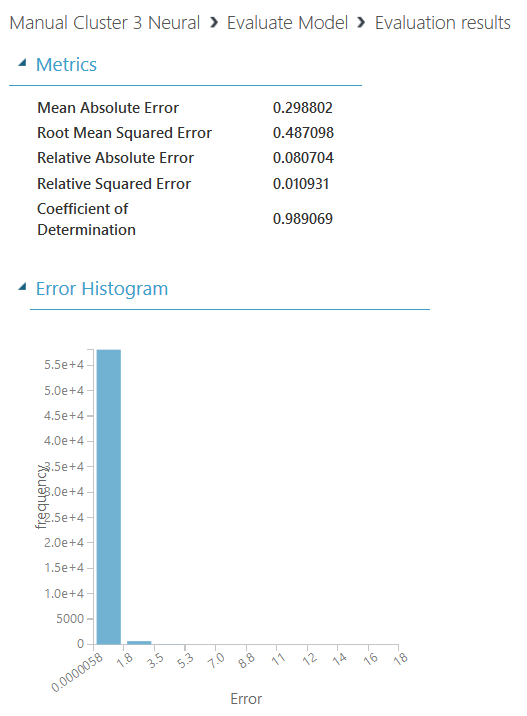
* Linear Regression
* Random Forest Regression
* Neural Network

We have selected Neural Network Regression to be our best prediction model based on RMSE value.

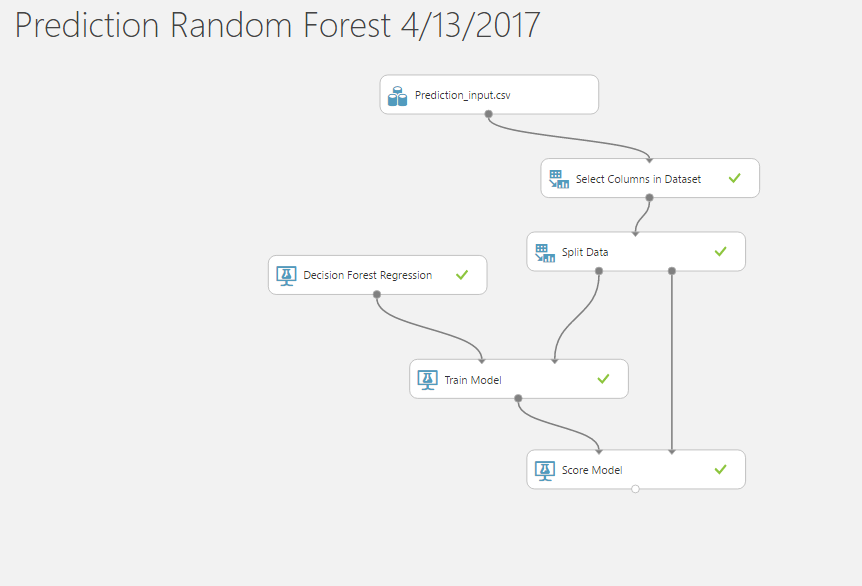
Below are the screen shots for the same.

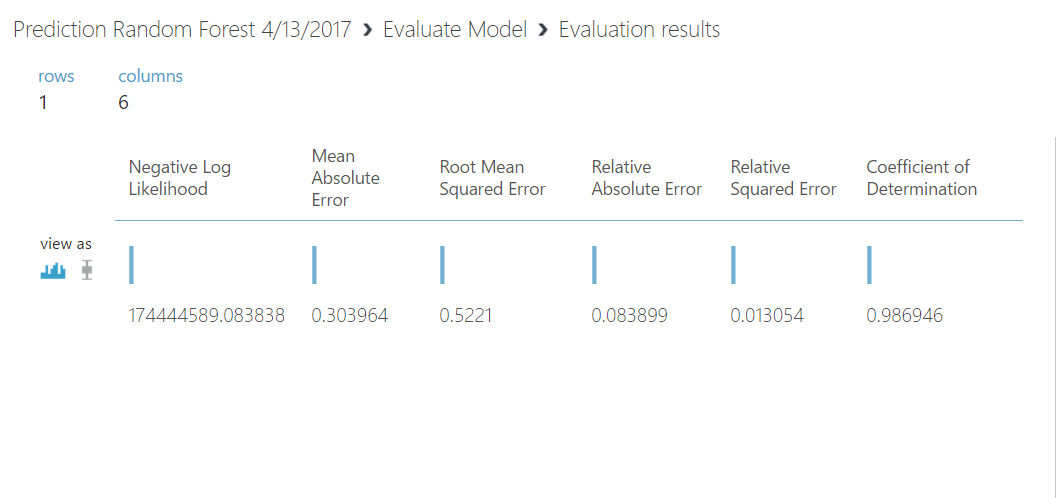






**No Clustering**





The next aim was to predict the interest rate by training the data set using various Regression models. We started off this section by considering few shortlisted features from part 1. And then feature engineered those columns by using Hypothesis testing, generated the p-values for each variables of the data. We repeated the hypothesis testing and finally got a few variables to be considered for the best output. Later, we worked on Backward variable selection method to finalize the final set of columns which we need as an input from the user and based on those values, we will be calculating the interest rate.

As per the requirement, we were supposed to run Regression Algorithm like KNN, Random Forest, Neural Network and Linear Regression. We chose the best method among these 4 by considering the smallest Root Mean Squared Error (RMSE) value. We implemented the algorithms with the help of python code using ipynb files. There were various parameters we generated like MAPE, MAE, RMSE and R-Squared, but we considered RMSE as the main parameter on which we decide the accuracy and correctness of our Algorithms.

The value with the minimum Root Mean Squared Error is considered the best and based on all the values, we chose Random Forest to be the best model with RMSE of 0.52

|  |  |  |  |
| --- | --- | --- | --- |
| Algorithm | RMSE | R-Squared | MAE |
| Linear Regression | 0.908 | 0.039 | 0.633 |
| Random Forest | 0.5221 | 0.013 | 0.303 |
| Neural Network | 0.5666 | 0.016 | 0.306 |

Based on the above RMSE value, we concluded that Random Forest is the best algorithm.

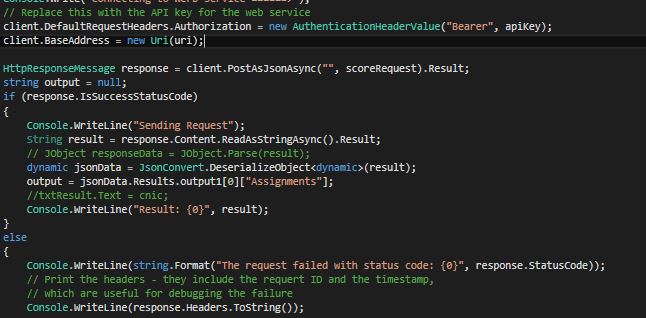
**Deployment**

Once the best prediction model was selected for every cluster, we deployed them in Azure Machine Learning Studio by following the below steps:

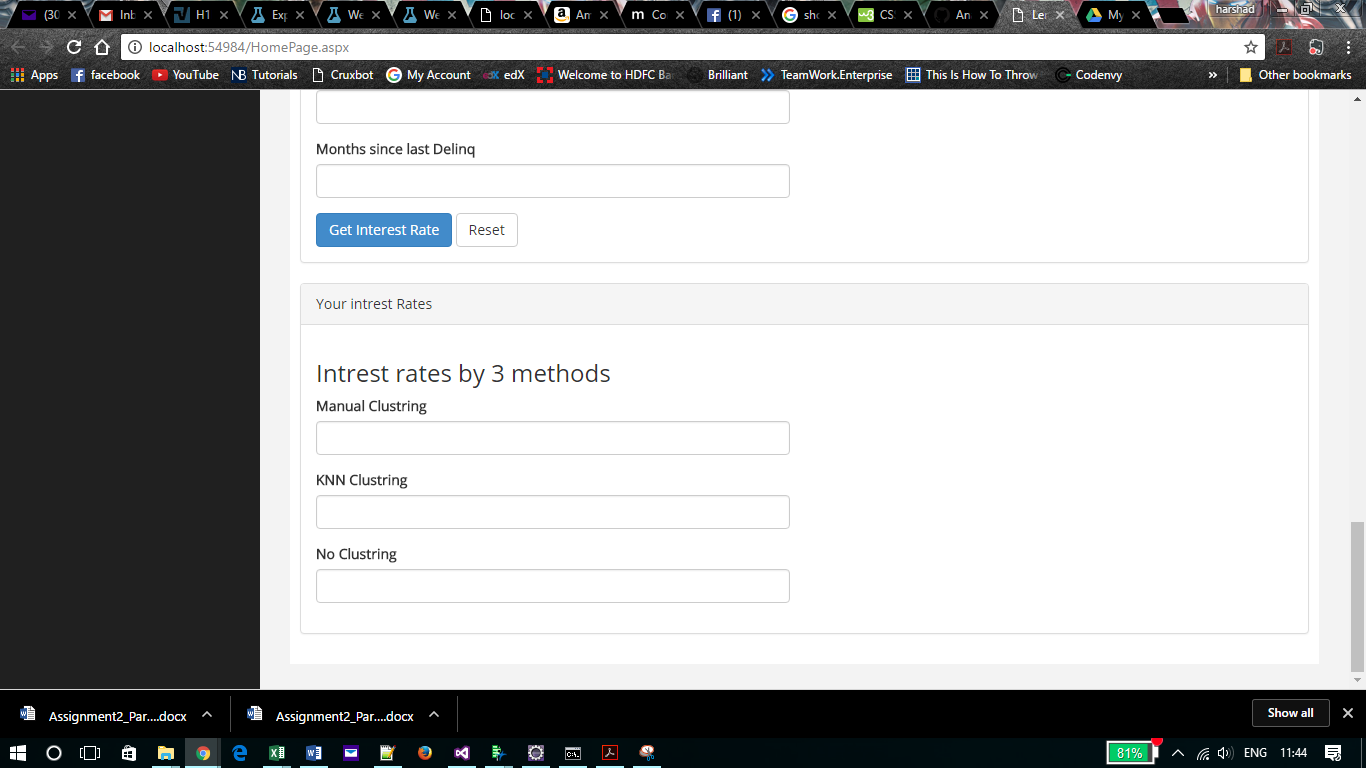
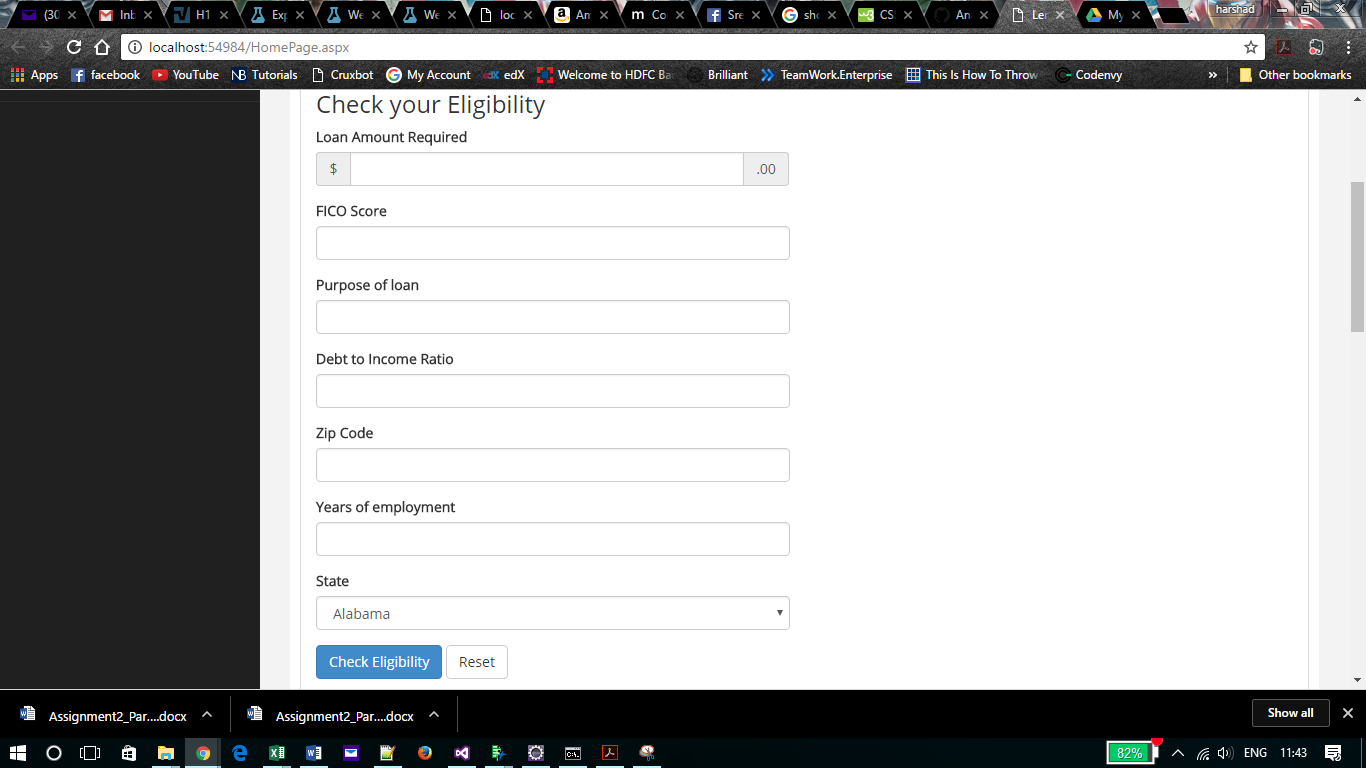
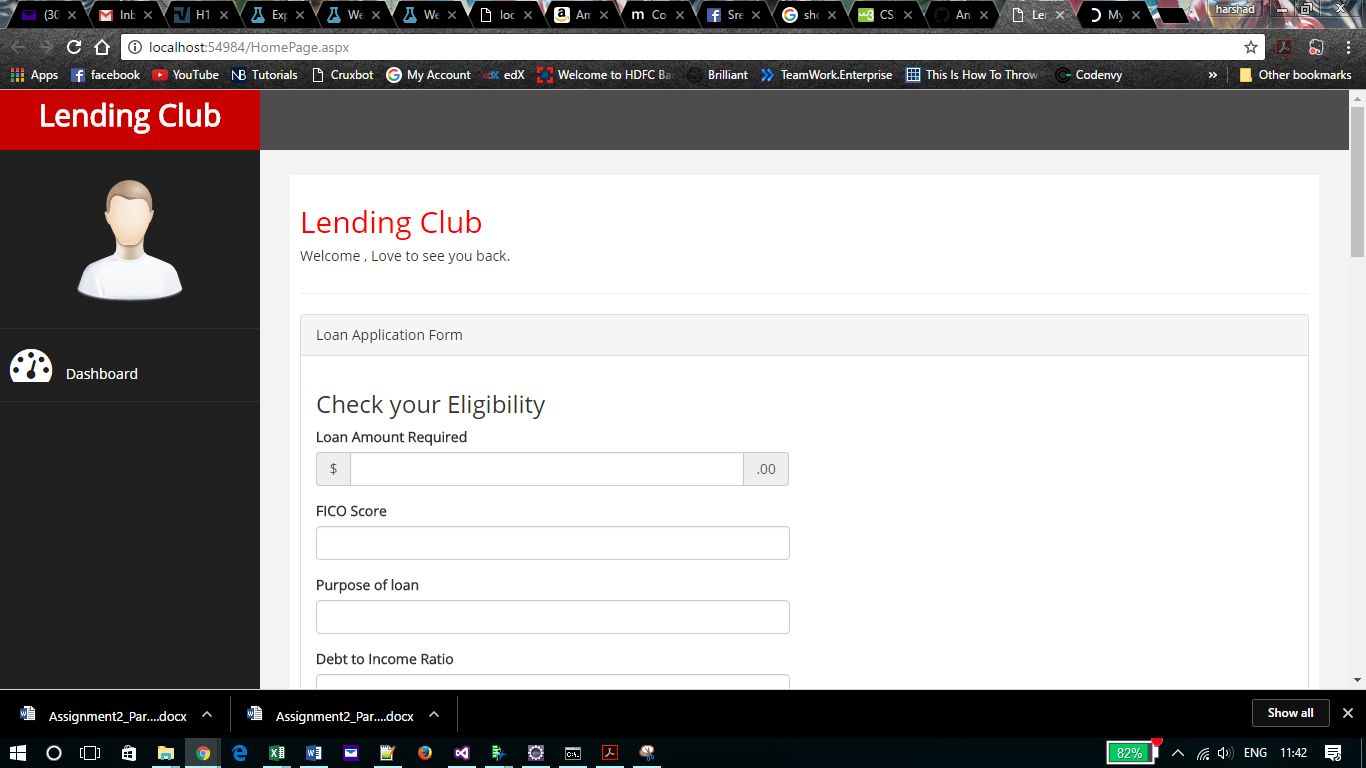
* Uploaded the datasets for each cluster
* Created a new experiment for each prediction model
* Imported the uploaded input file o the experiment
* Selected required columns from the file
* Split the input data into 80-20 format for training and testing the model
* Trained the best model.
* Tested the model by using the Score Model
* Evaluated the model using Evaluation Model
* Create a Predictive Web Service for the model
* Edit the Select Columns from Dataset Model to remove the column to be predicted
* Run the model again and deploy it to web service
* Test the model for an observation
* Consume this model using API key and Request-Response URL

**Consuming the Web service using C#**

We have used C# and asp.net to form the request, call the API and read the response sent by the Rest API.



The input is given by a asp.net form which is designed to take all the inputs and show the output.



We had to implement this web service using some synchronous method so we have changed the request to Synchronous method.