# ADS ASSIGNMENT – 2

TEAM 2

PRIYANJNA SHARMA
SUMIT DEO
GAUTAM PAWAR

# 1 TABLE OF CONTENTS

# 2 DOCKER

1.Pull the docker image from the docker hub:

docker pull sumit91188/assignment2part1

2.Create an image on the local docker terminal:

docker run -d sumit91188/assignment2part1 tail -f /dev/null

3.Check the container created from the above query

docker ps -a

4.Run the pipeline for loan data  on the docker container

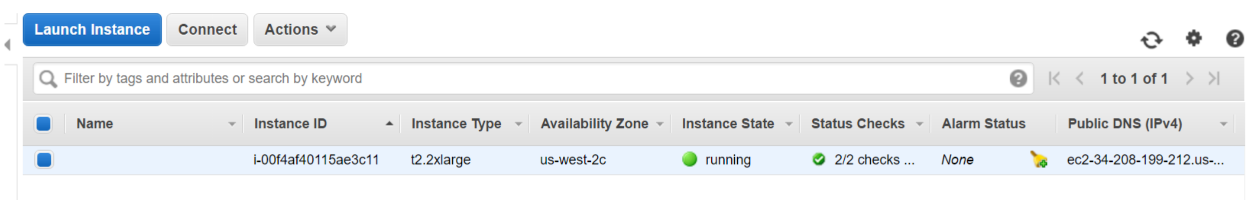
docker exec -it <container_id> python pipeline_loan_data.py start_task --local-scheduler

5.Run the pipeline for declined loan data on the docker container

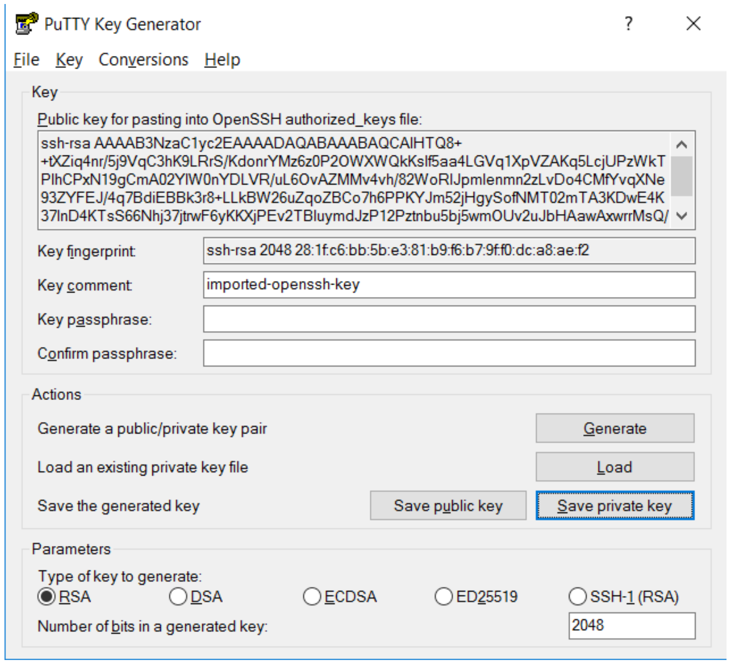
docker exec -it <container_id> python pipeline_declined_loan_data.py start_task --local-scheduler

# 3 RUNNING PIPELINE ON CLOUD

## 1: CREATE AN EC2 INSTANCE: WE SHOULD CREATE AN AMAZON EC2 INSTANCE TO RUN ON CLOUD WHERE THE APPLICATION CAN BE HOSTED.

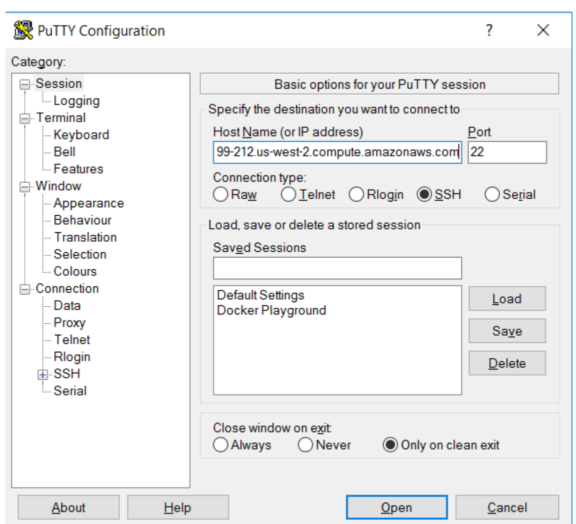



2. Use Putty to connect to the instance: Use puttygen to convert the .pem file generated from the instance and convert it to .ppk (private key) file.




## 3. Connect to EC2 instance:

Enter the public domain DNS for that particular instance and use the private key generated to connect to the EC2 instance.



## 4. Download Docker on the EC2 Instance:

## 5. Pull and Run the Docker Image:

Below screenshots shows the status of the completed job and we can check the S3 bucket for the data uploaded.

```
[ec2-user@ip-172-31-12-71 ~]$ docker pull sumit91188/assignment2part1
Using default tag: latest
latest: Pulling from sumit91188/assignment2part1
6d827a3ef358: Pull complete
2726297beaf1: Pull complete
7d27bd3d7fec: Pull complete
44ae682c18a3: Pull complete
824bd01a76a3: Pull complete
68fe59875298: Pull complete
9ca1d7ae0c4b: Pull complete
3aa0ec115b5a: Pull complete
09351c96a5f0: Pull complete
de4d1ef3301f: Pull complete
d799aa302b6c: Pull complete
Digest: sha256:a65ee34c410dd566911e89f0c54ec27fc878a54d5c1e223040ef18fa61e68482
Status: Downloaded newer image for sumit91188/assignment2part1:latest
[ec2-user@ip-172-31-12-71 ~]$ docker run -d sumit91188/assignment2part1 tail -f
/dev/null
9225564a12364e4e4e7357a5c454874624b983cf87aa066fbef0c4bf26773f5b
[ec2-user@ip-172-31-12-71 ~]$ docker ps -a
CONTAINER ID        IMAGE                         COMMAND               CREATED
            STATUS              PORTS                 NAMES
9225564a1236        sumit91188/assignment2part1   "tail -f /dev/null"   5 second
s ago       Up 4 seconds                              serene_gates
```

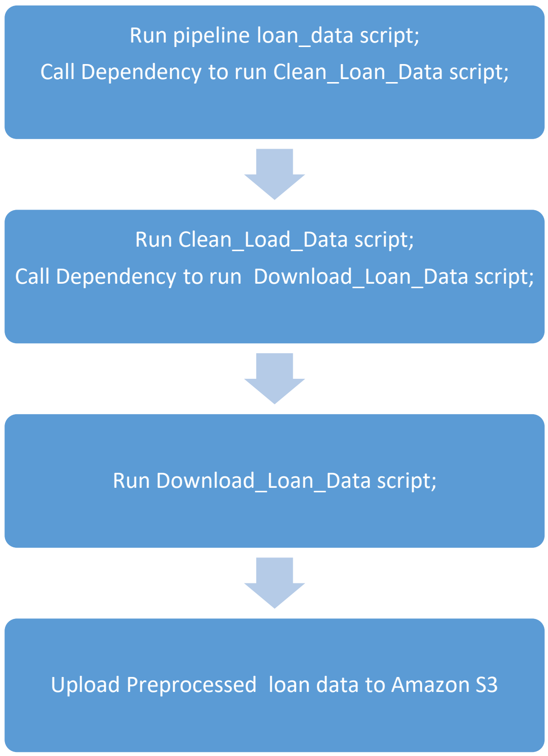
```
[ec2-user@ip-172-31-12-71 ~]$ docker exec -it b052351c3901 python pipeline_loan_data.py start_task --local-scheduler
DEBUG: Checking if start_task() is complete
DEBUG: Checking if Clean_loan_data() is complete
INFO: Informed scheduler that task   start_task__99914b932b   has status   PENDING
DEBUG: Checking if Download_loan_data() is complete
INFO: Informed scheduler that task   Clean_loan_data__99914b932b   has status   PENDING
INFO: Informed scheduler that task   Download_loan_data__99914b932b   has status   PENDING
INFO: Done scheduling tasks
INFO: Running Worker with 1 processes
DEBUG: Asking scheduler for work...
DEBUG: Pending tasks: 3
INFO: [pid 6] Worker Worker(salt=114030799, workers=1, host=b052351c3901, username=root, pid=6) running   Download_loan_data()
INFO:root:Application started....
INFO:root:Opening an url and creating a soup
INFO:root:Working on Loan Stats Data
INFO:root:Total number of Loan Stats Data files : 8
INFO:root:Downloading Loan Stats Data...
INFO:root:Downloading data for 2007 - 2011
INFO:root:Downloading data for 2012 - 2013
INFO:root:Downloading data for 2014
INFO:root:Downloading data for 2015
```

```
DEBUG:luigi-interface:Asking scheduler for work...
INFO:luigi.scheduler:Starting pruning of task graph
INFO:luigi.scheduler:Done pruning task graph
DEBUG: Done
DEBUG:luigi-interface:Done
DEBUG: There are no more tasks to run at this time
DEBUG:luigi-interface:There are no more tasks to run at this time
INFO: Worker Worker(salt=114030799, workers=1, host=b052351c3901, username=root, pid=
INFO:luigi-interface:Worker Worker(salt=114030799, workers=1, host=b052351c3901, user
INFO:
===== Luigi Execution Summary =====

Scheduled 3 tasks of which:
* 3 ran successfully:
    - 1 Clean_loan_data()
    - 1 Download_loan_data()
    - 1 start_task()
```

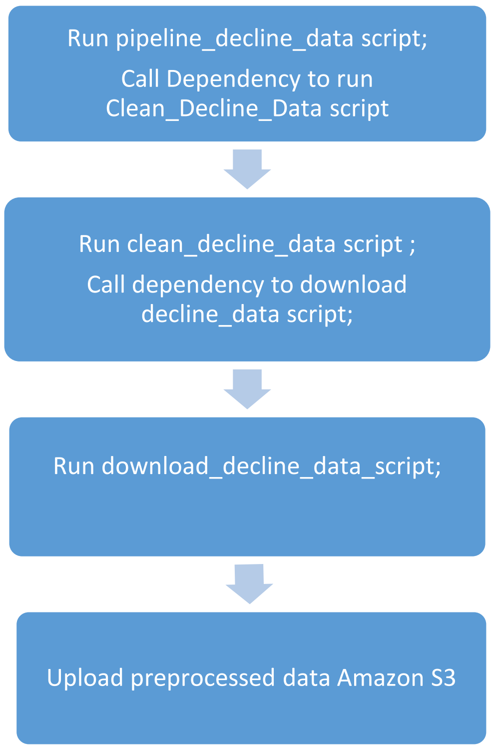
# 4 PIPELINING – LUIGI

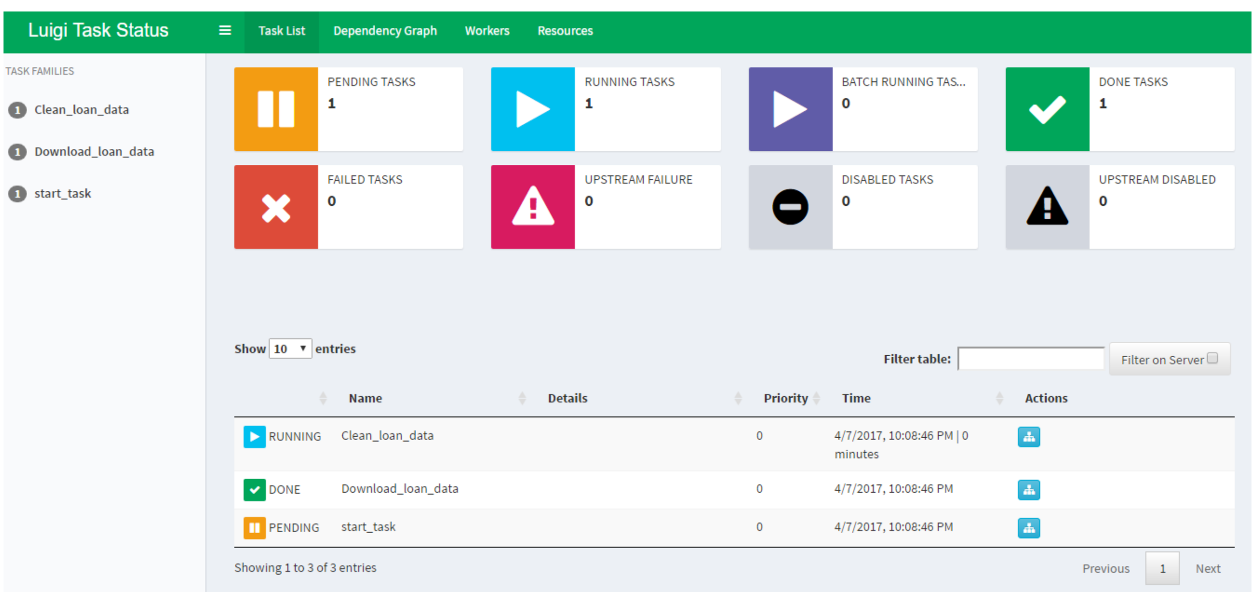
We have used Luigi to pipeline our tasks of Data Download, Data Preprocessing and Saving data to AWS S3. It helped us manage the workflow of our application.

## 4.1 LOAN DATA FLOW



## 4.2 Decline Data Flow



# 5 DATA DOWNLOAD

## 5.1 DOWNLOADING LOAN DATA

- Traversed to the link https://www.lendingclub.com/info/download-data.action
- Selected the loan data files to be downloaded by finding the div that stores it
- Extracted the files and stored them in the home folder dynamically if not already present at the below mentioned location:
  Downloads/LoanData/
- Logged each of the steps in log files

## 5.2 DOWNLOADING DECLINE DATA

- Traversed to the link https://www.lendingclub.com/info/download-data.action
- Selected the loan data files to be downloaded by finding the div that stores it
- Extracted the files and stored them in the home folder dynamically if not already present at this location: Downloads/DeclinedLoans/
- Logged each of the steps in log files

# 6 DATA PREPROCESSING AND FEATURE ENGINEERING

## 6.1 LOAN DATA

We created the below mentioned derived columns that we thought would directly impact the interest rate

### 6.1.1 Preprocessing

- We assumed columns which have >= 90% data as null will not aid our analysis
- So, we dropped columns which have >=90% data as null which gave us 86 columns out of 113 columns
- Missing Values :
  - Replace missing values in **annual_inc** by mean
  - For other columns replaced missing values by their respective modes
- Dropped column zipCode as it was encrypted
-

### 6.1.2 Derived Columns

- **cat_loan_amnt :** Bins of loan_amount based on quantiles

- **cat_annual_inc:** created bins of annual_inc based on quantiles
- **Loan_Status_Binary**: This column indicates if the borrower is a defaulter or not
  **Defaulter :** if the loan_status has any of the following values the value for Loan_Status_Binary will be 1 else 0
  - Charged Off
  - Default
  - Does not meet the credit policy. Status:Charged Off
  - In Grace Period
  - Default Receiver
  - Late (16-30 days)
  - Late (31-120 days)

- **cr_line_history :** this column gives the number of years of the borrower's credit history
  We calculated this column by subtracting issue year and earliest_cr_line value of the borrower
  **Calculation :** ['issue_d'].dt.year - ['earliest_cr_line'].dt.year
- **Verification Status** : if the verification_status is not verified then 0 else 1
  For this we assumed that 'Verified' and 'Verified Source' means the same

#### 6.1.3 Storing on S3

- Finally stored the clean and preprocessed data on S3 using Luigi Pipeline

### 6.2 Declined Loans Data

- Created bins for Risk score using the below code according to FICO score ranges:

```
groupNames = ['Invalid','Very High','High','Moderate', 'Low', 'Very Low']
bins = [110, 299, 400, 600, 700, 800, 991]
df['RiskCategories'] = pd.cut(pd.to_numeric(df['Risk_Score'], errors='coerce'), bins,
labels=groupNames)
```

- Replaced missing values in Risk score by an invalid value '111'

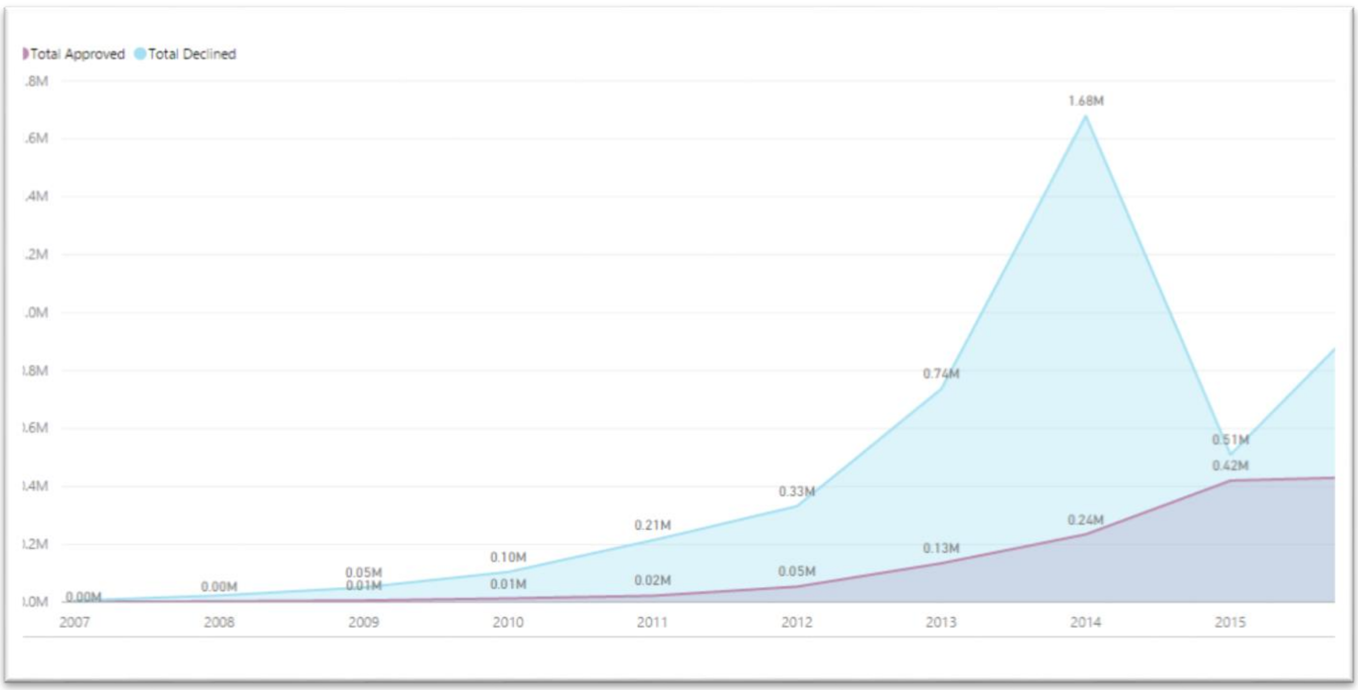
#### 6.2.1 Storing on S3

- Finally stored the clean and preprocessed data on S3 using Luigi Pipeline

# 7 EXPLORATORY DATA ANALYSIS

We performed exploratory data analysis by using our derived columns and the provided columns. Our analysis for Loan Data and Decline Loan data is as follows:

**POWER BI LINK:** https://app.powerbi.com/groups/me/dashboards/c6745b00-47ab-4d32-8de6-a570d13902d7




Observations :

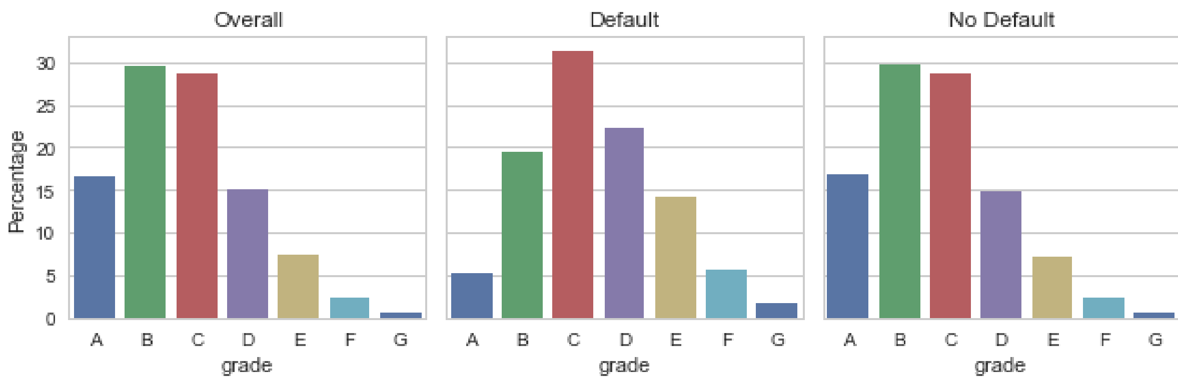
- From the graph we can see a hike in the number of declined loans as well as the number of approved loans. Which implies an increase in the number of applications in the year 2014.
- This supports the fact that in Year 2014, Lending Club went public and started providing loans to small businesses. It also partnered with Union Bank and in the end of 2014, Lending club raised $900 million which we can see with the spike in the number of applications.

## 7.1 USING PYTHON

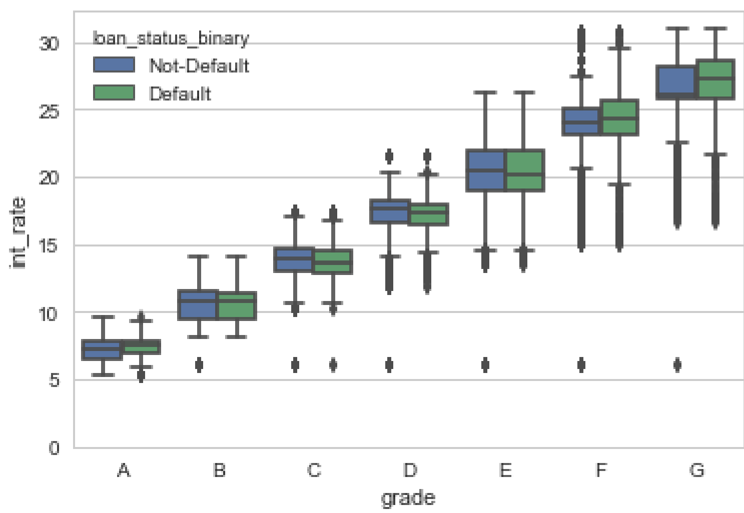
### 7.1.1 LOAN DATA

*1. Exploring the relationship of variables to late payment*

The grade of the loan is the companies estimate of the likelihood of default for the loan. As should probably be expected the best graded loans (A and B) have a higher percentage of loans with no default than with a default. C is approximately the same percentage across no default and default and the worst graded loans (D, E, F and G) have a higher percentage of loans with default than with no default.

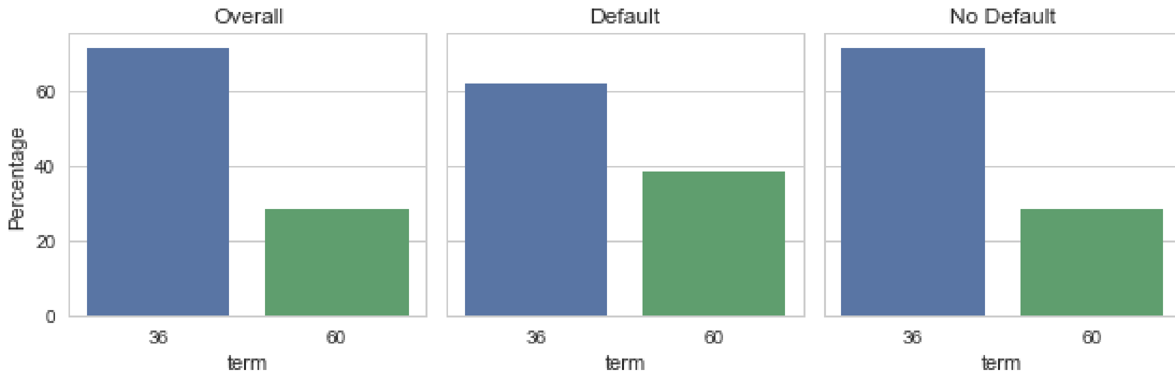



Even controlling for the grade of the loan (as this will be used to calculate the interest rate) the defaulting loans still have a higher interest rate than non-defaulting loans for most of the grades.

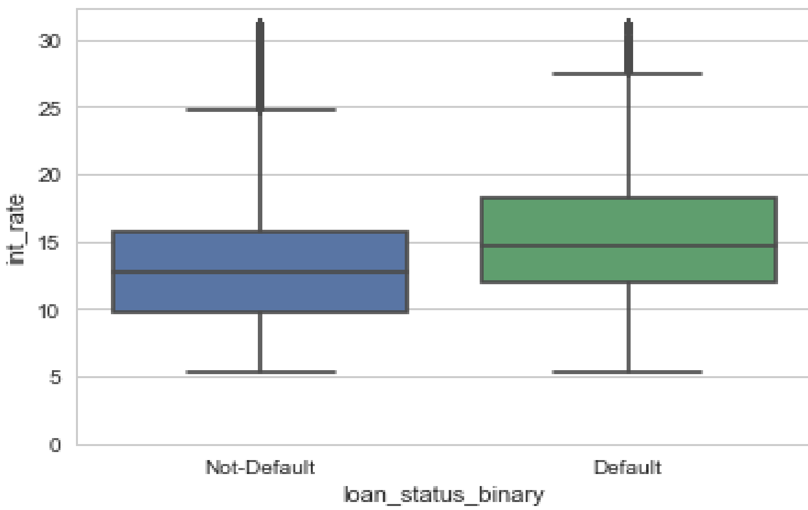



*2. Loan Term:*

The longer-term loans (60 months) make up a higher percentage of the defaults than the non-defaulting loans.

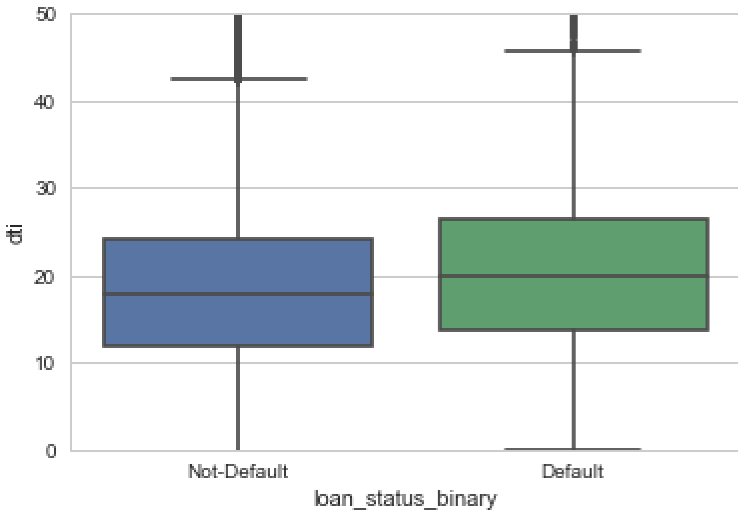



3. *Interest Rate:*
   The defaulting loans have a higher interest rate than non-defaulting loans.



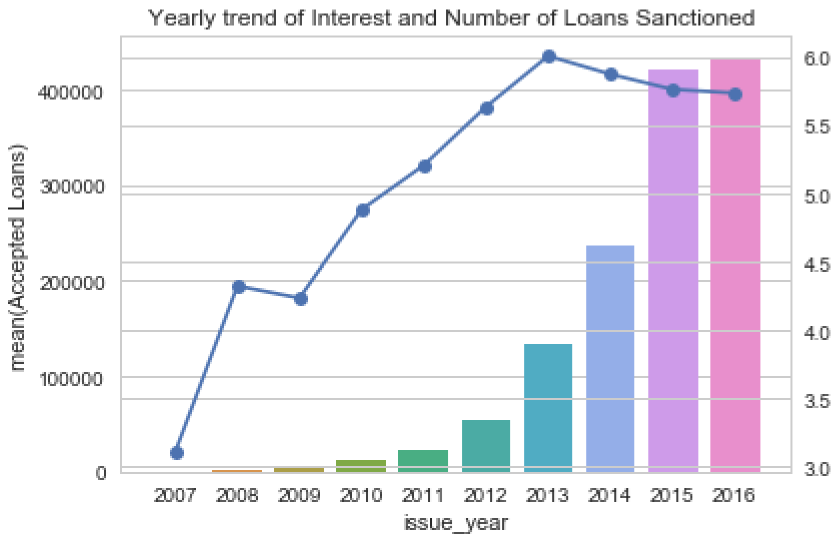

4. *Debt to Income Ratio:*

Defaulting loans have a higher DTI.

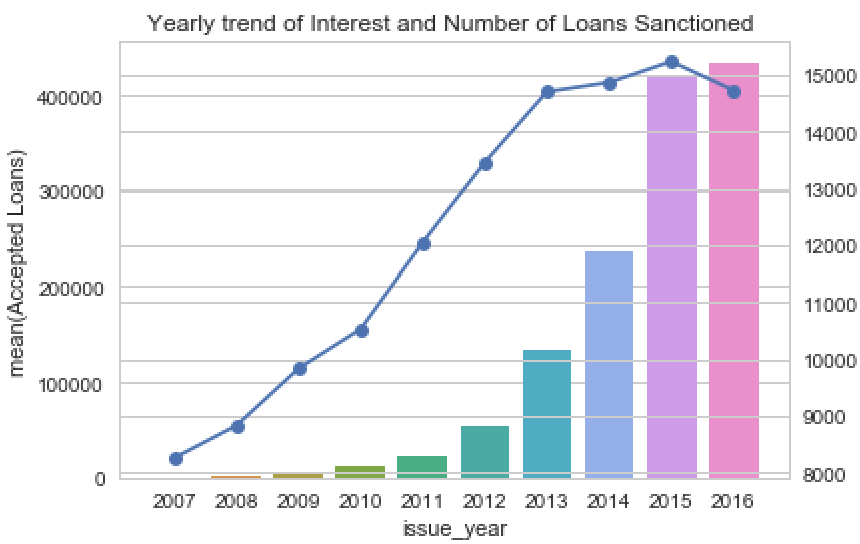


5. *Summarizing the data by vintage:*

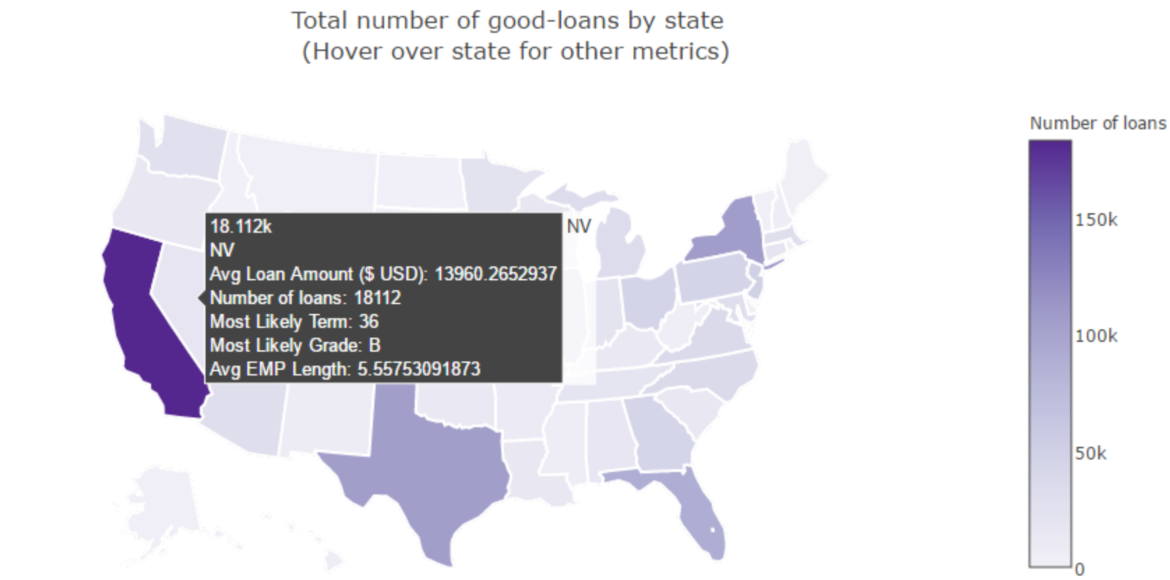
***Summary Chart for Year Wise Accepted Data and Approved Loans:***



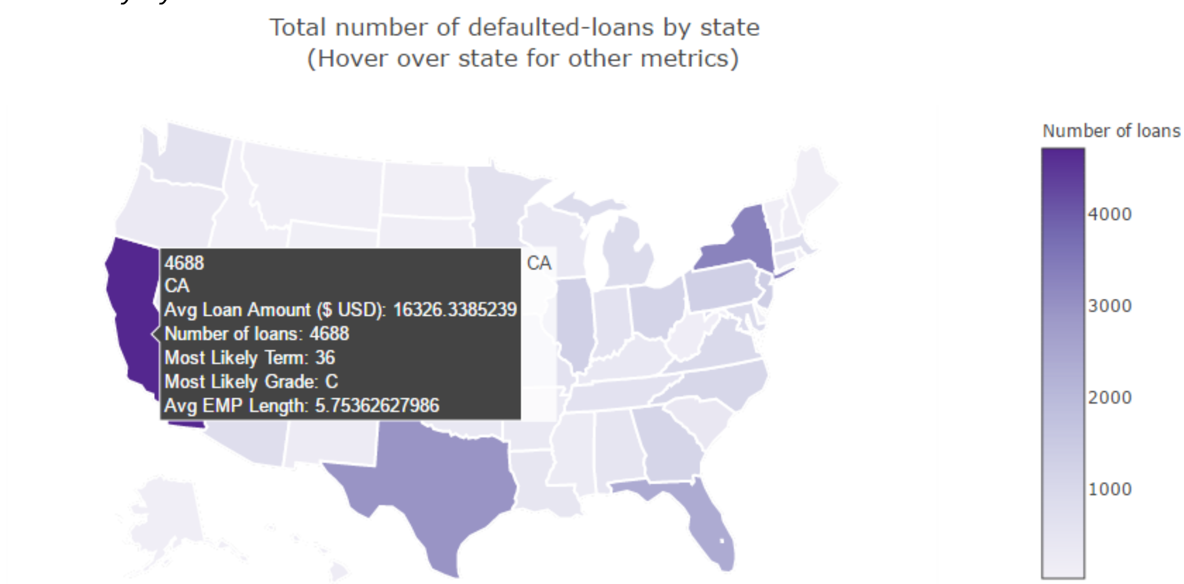

6. *Summarizing the data by vintage:*



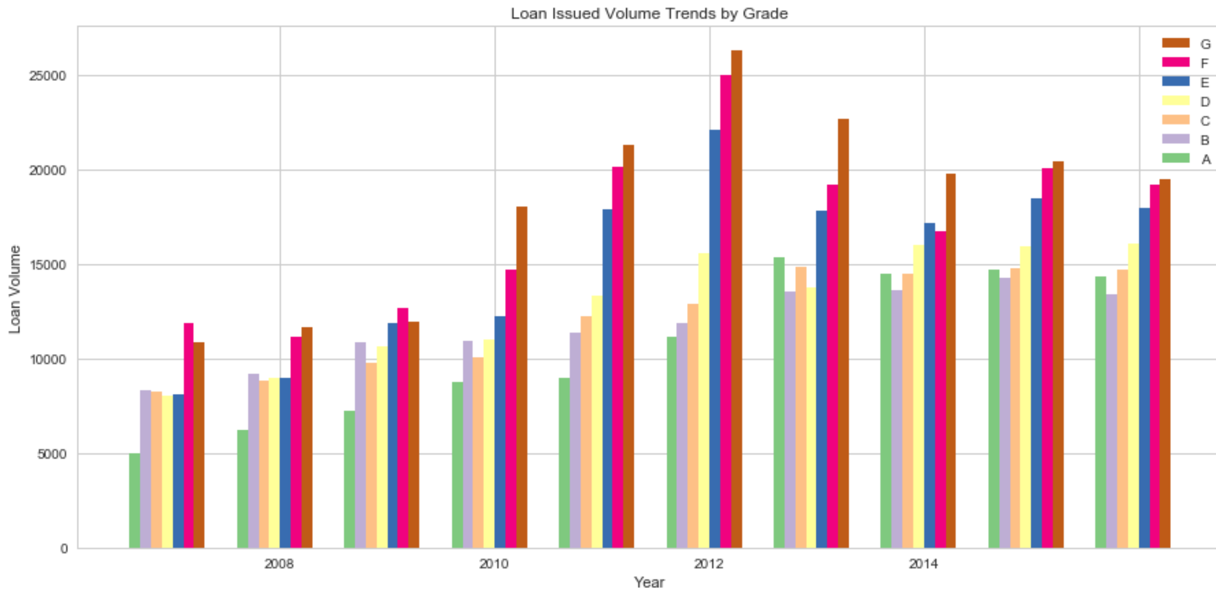
***7.*** *Summarizing good & default loans by state:*
***Summary by state for Good loans:***




***8.*** *Summary by state for default loans:*



*9. Summarizing loans by Grade:*




*10. Summarizing loans by Year & Loan-Status:*

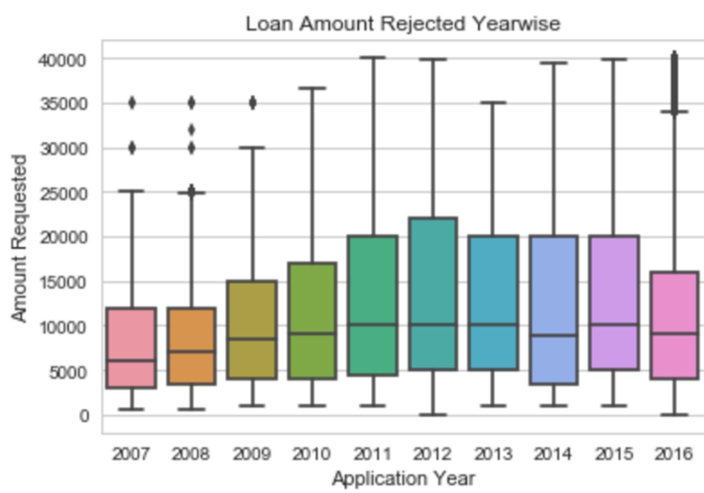
### 7.1.2 DECLINED LOAN DATA

Used seaborn, matplotlib, plotly libraries for plotting the graph.

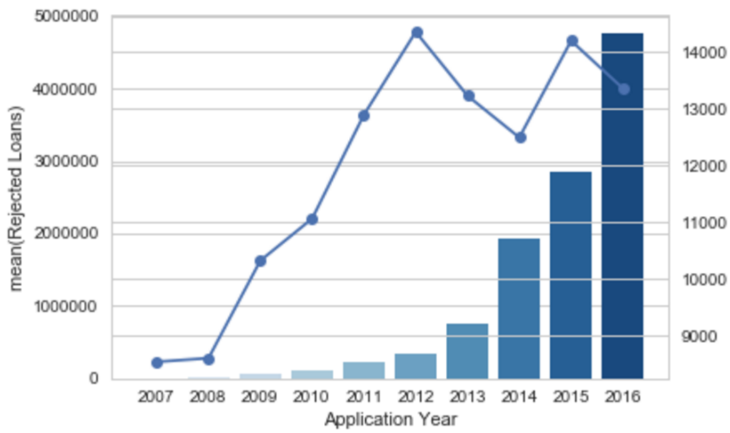
Year wise Analysis:

Created a summary data frame dfSummary to analyze the yearly trend for the loan volume and the total loan amount.

```
#Summary by Year
#by year and counting the total number of rejected loans
seriesCount = dfDecLoans['Amount Requested'].groupby(dfDecLoans['Application Year']).count()

#by year and counting the total of loan-amount
seriesTotAmt = dfDecLoans['Amount Requested'].groupby(dfDecLoans['Application Year']).mean()

#combining seriesCount and seriesTotAmt into summary Metrix data frame
columns=['Application Year', 'Rejected Loans', 'Avg Amount Requested']
dfSummary = pd.DataFrame({'Application Year':seriesCount.index,'Rejected Loans': seriesCount,'Avg Amount Requested':seriesTotAmt}

dfSummary[columns].head()
```



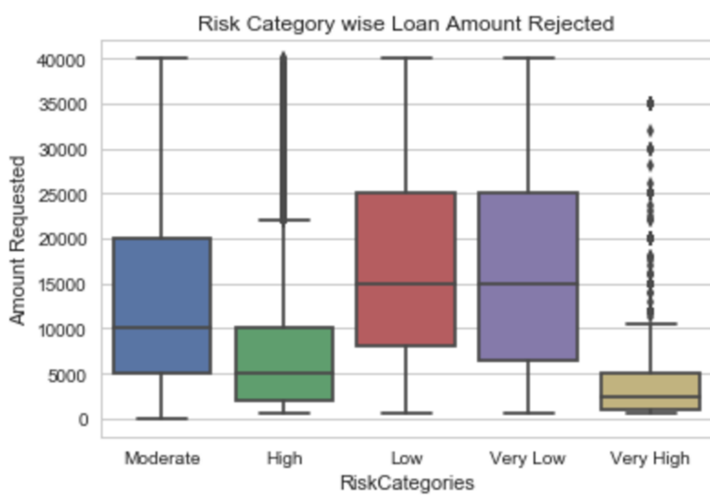

1. As we can see in the above chart, the total amount rejected has **increased** from **2008-2012** and is steady after that.
2. As we can see below the hike in average amount requested from the year **2010-12 got a huge hike** and then suddenly got declined from **2012- 2014** with the total count of rejected loans being increased constantly over years.



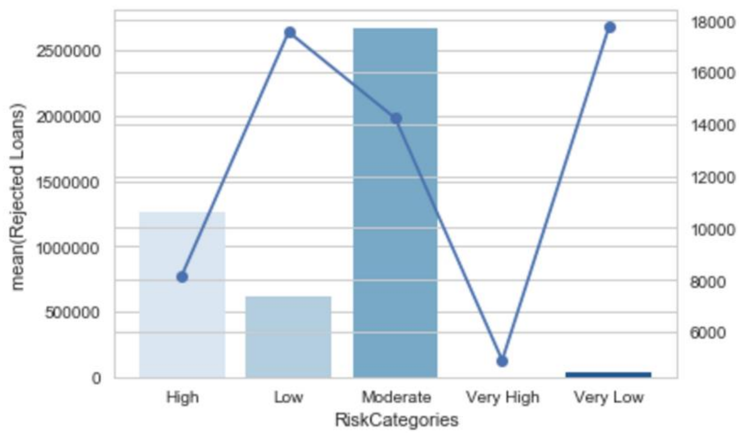

## Risk Categories wise Analysis:

We have created bins on the basis of Risk Scores as: Very High, Very Low, Low and High.

1. As we can see in the below box plot, the people lying in **"Very High"** bucket have the least amount of loan rejected as they apply for less amount of loan.

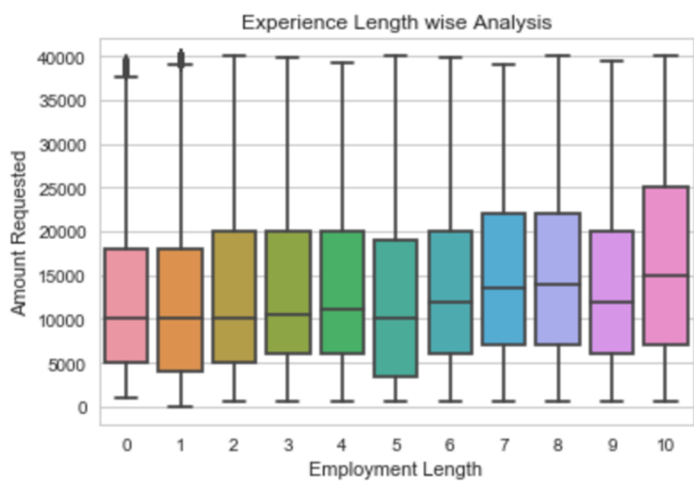



2. People falling in "**Very Low"** i.e Very Low Risk bucket have the least count of rejection and the highest in the amount requested.




## Employee Experience Length wise Analysis:

1. People with **>10 years of experience** have the most loan amount requested.

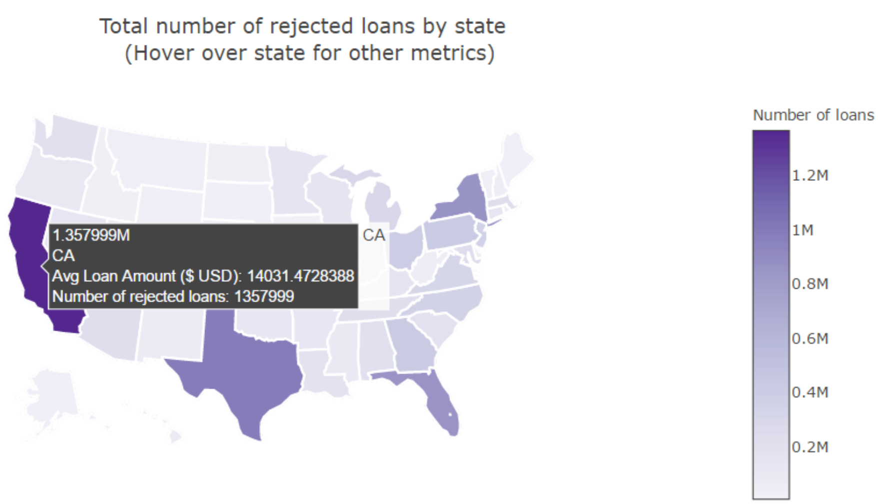


2. Population with 5 years of experience got their loan rejected most.


## State wise Analysis:

We have performed different analysis based on state. As we can see below:

1. California has the highest number of loan amount requested and rejected both.



## 7.2 Power BI

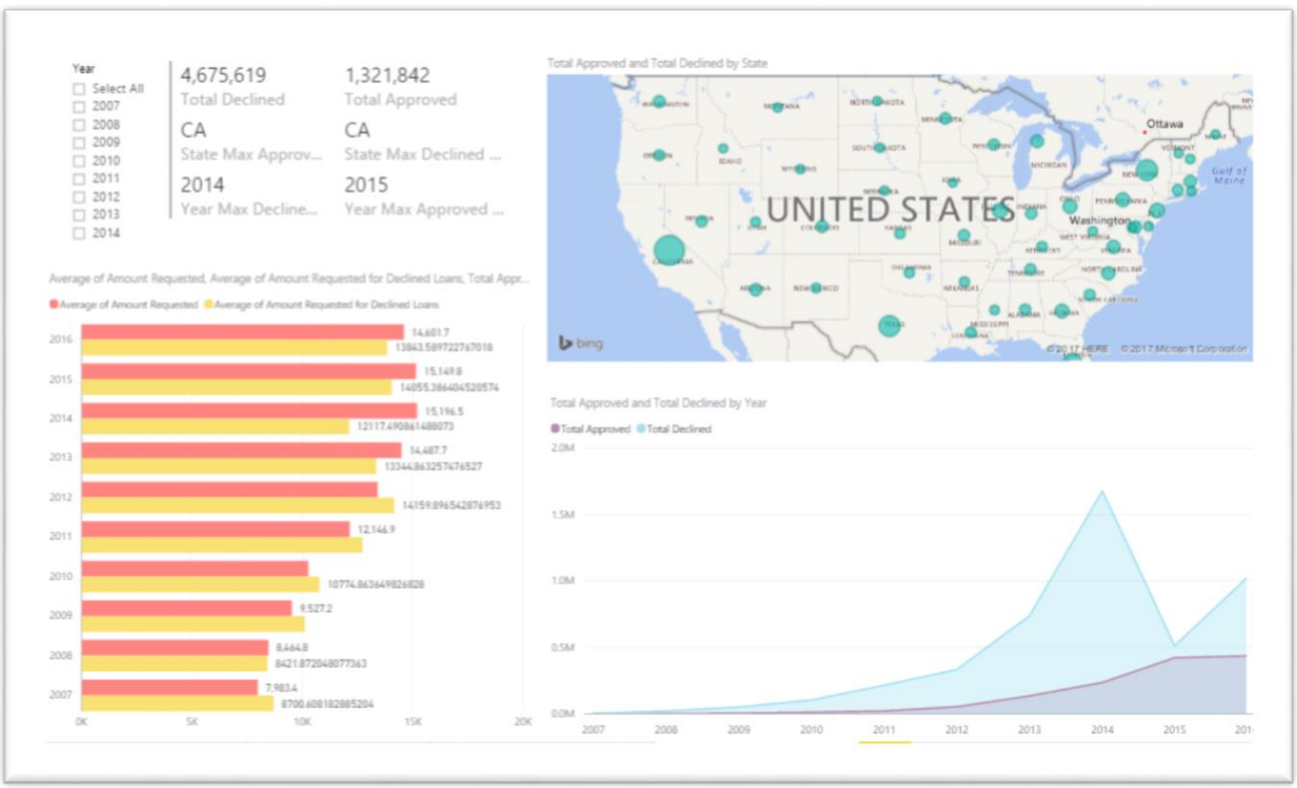
We analyzed the trends in Approved Loans and Declined Loans on the Lending Club data using Power BI

### 7.2.1 Lending Loan Club Analysis for Approved and Declined Loans

- Summarized the loans and declined loans data based on Year and State Using group by in MDX(Power BI)
- Joined the two summaries into one table using DAX

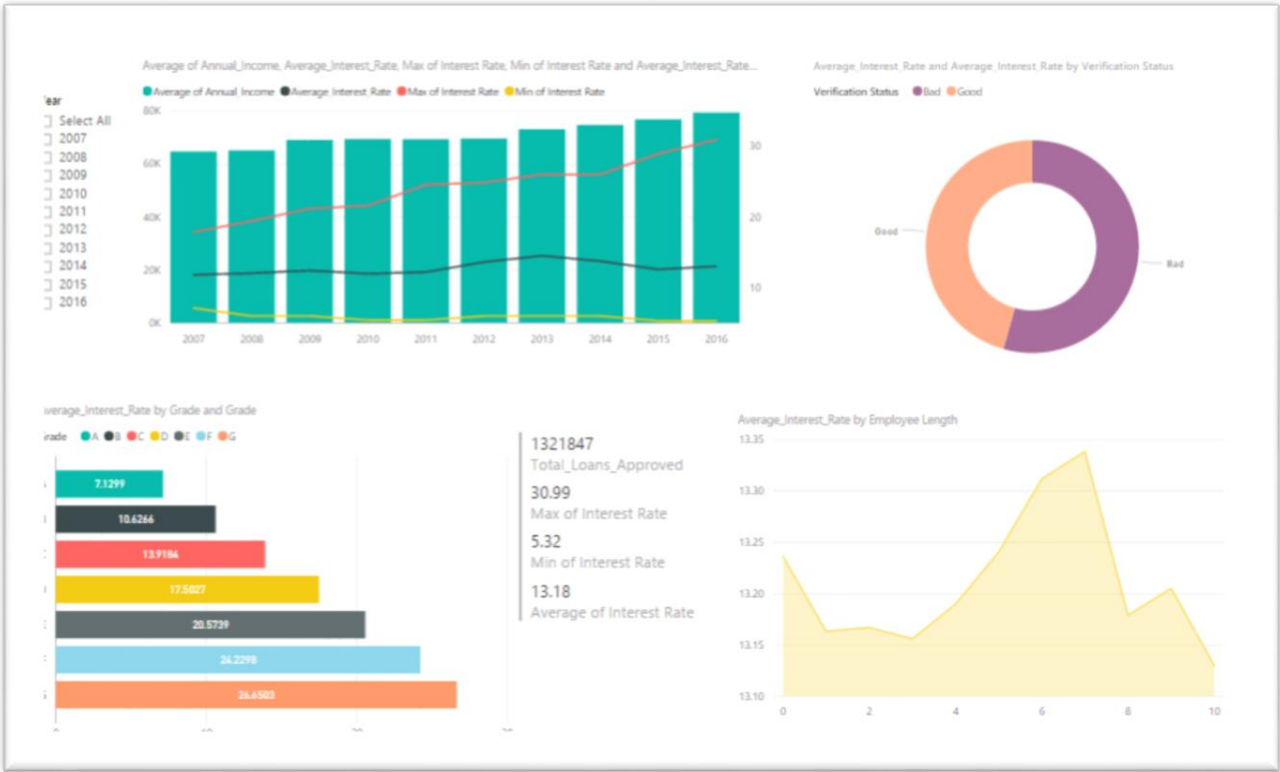
```
LendingClubAnalysis = FILTER (
    CROSSJOIN ( LendingClub, 'Table' ),
    LendingClub[Year] = 'Table'[ApplicationYear] && LendingClub[State] = 'Table'[State1]
)
```

- Lending Loan club

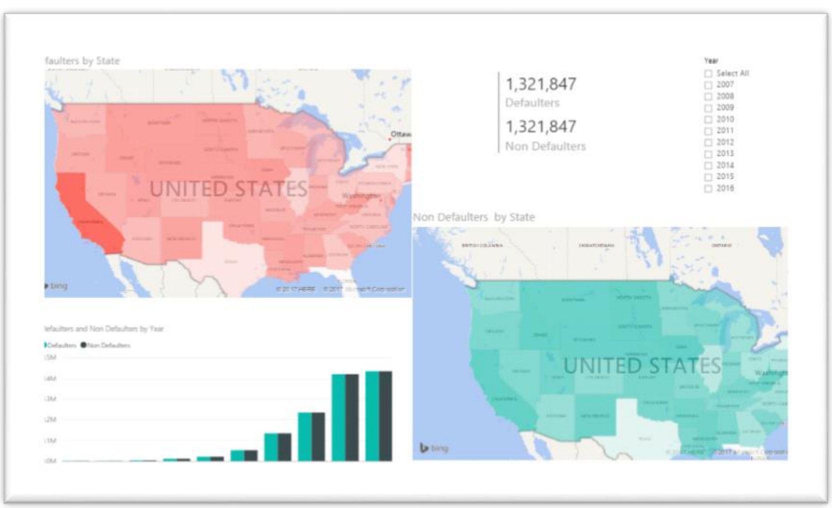


### 7.2.2 Lending Loan Club Analysis for Approved Loans

- Analyzed approved loans based on state, year , grade , verification status , interest rate
- Grade G has the highest interest rates

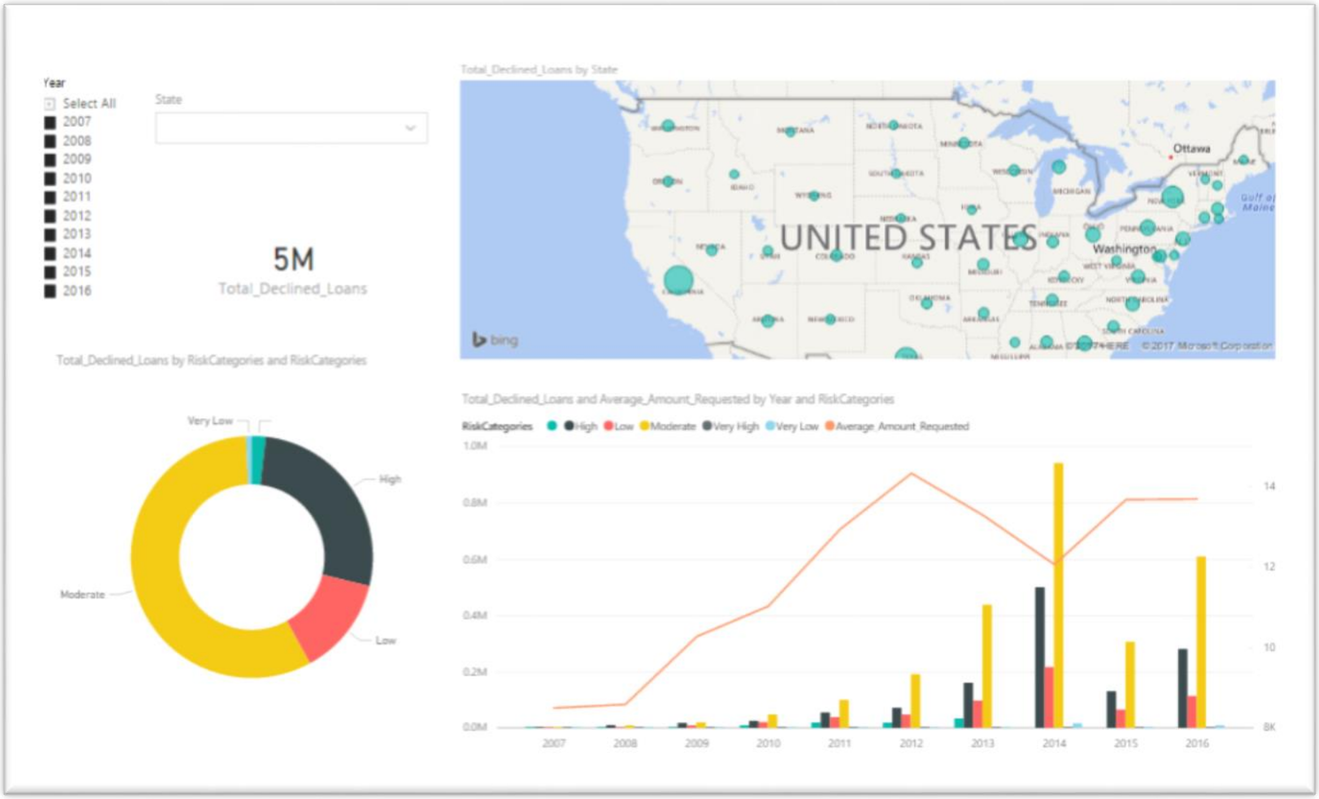



- Defaulter Analyses : analyzed the total number of defaulters by using a calculated column : loan_status_binary for different states

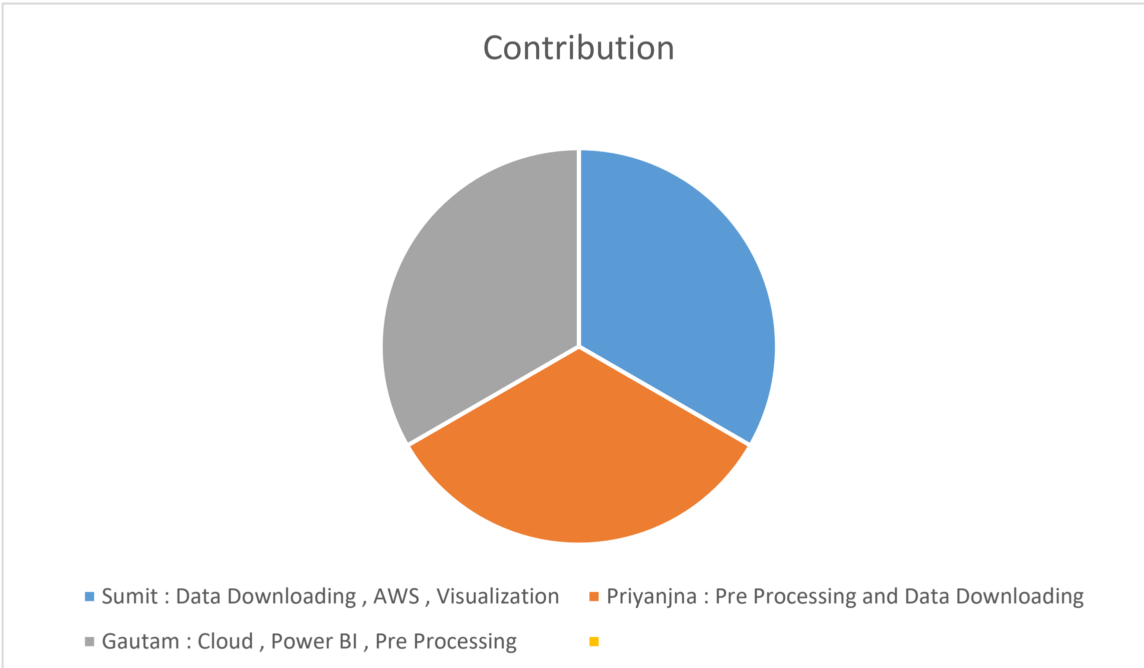


### 7.2.3 Lending Loan Club Analysis for Declined Loans

- Analyzed declined loans based on state, year and different risk score buckets we created.
- Total declined loans were the most for moderate range of risk score
- Total declined loans were less for high risk score ranges



# 8 CONTRIBUTION



# 9 REFERENCES


1. https://en.wikipedia.org/wiki/Lending_Club
2. https://www.lendingclub.com/info/download-data.action
3. https://www.credco.com/assets/pdfs/datasheets/FICO-booklet.pdf
4. http://www.lendingmemo.com/average-investor-return-lending-club-dropping/