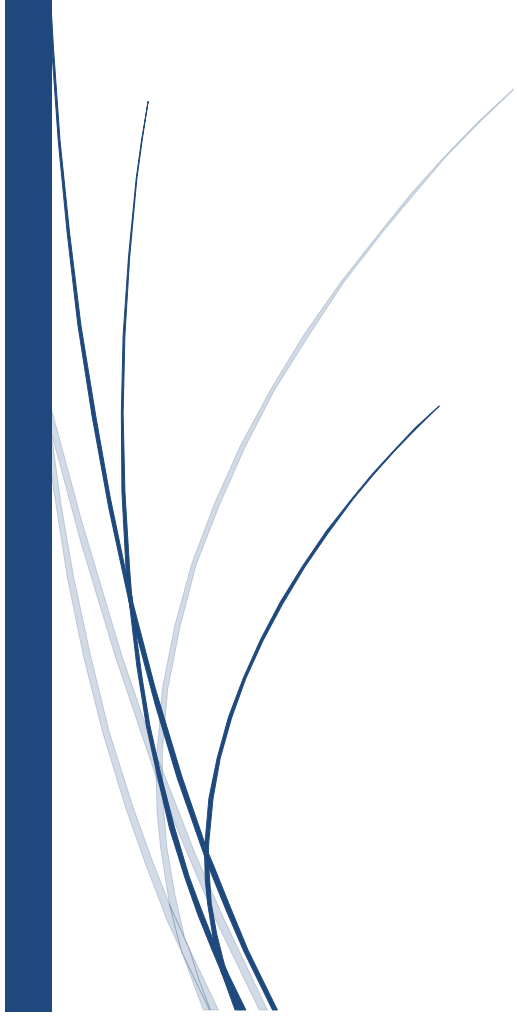
Predicting Loan Default



GROUP PROJECT
ON
PYTHON

Batch: DSP 9 –Mumbai

Group Members

Abhishek Khadse

Disha Lakhani

Nachiket Thakur

Rinkesh Ghadi

INDEX

Chapter 1

Title: Prediction of Loan Defaulter

Objective:

- The purpose is to build a model to predict whether a borrower will default in the future or not, thereby helping the company to make a decision whether to pass the loan or decline.

Introduction:

- The XYZ Corporation is lending company which issues loan based on default, payment information, and credit history of the customer.

- The main use of classification models is to score the likelihood of an event occurring.

- For loan data, the model will be used to predict whether a loan will be paid off in full or the loan needs to be charged off and possibly go into default.

- You can use the model to score the quality of current loans and identify the ones most likely to default.

Chapter 2

2.1 Understanding the Data:

- The dataset consists of 855969 unique observations and 73 variables.

- Out of 855969, 602939 individuals are under current status.
- In the given dataset, 602939 individuals are under current status have been considered as Non-defaulters.
- Building a model considering these observations is of no use, as they may turn out to be defaulters at the end.
- Hence, these need to drop from the dataset and can be used as validation file by dropping the dependent column.
- Some variables leak information regarding the future, i.e, they turn to be irrelevant features when it comes for predicting default.
- Then, there are some variables that have more than 50% NA values.Following table shows variables that are dropped due to their irrelevancy or having more than 50% missing values.

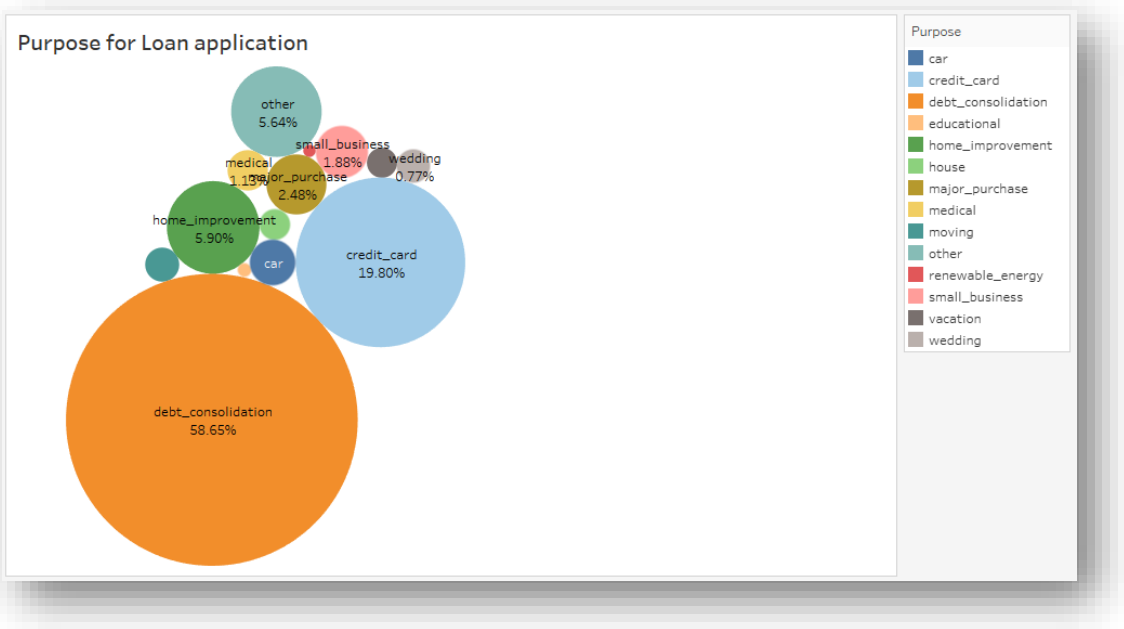
Parameter	Variable
Irrelevant	last_credit_pull_d, title, collection_recovery_fee, collections_12_mths_ex_med, funded_amnt, funded_amnt_inv, earliest_cr_line, id, member_id, sub_grade, pymnt_plan, last_pymnt_d, last_pymnt_amnt, next_pymnt_d, out_prncp, out_prncp_inv,emp_title, zip_code, recoveries, total_pymnt, total_pymnt_inv, total_rec_prncp, total_rec_int, total_rec_late_fee, inq_last_6mths, instalment,int_rate
More than 50% missing values	annual_inc_joint, dti_joint, desc, mths_since_last_delinq (51%MV), mths_since_last_major_derog, mths_since_last_record, verified_status_joint, open_acc_6m, open_il_6m, open_il_12m, open_il_24m, mths_since_rcnt_il, total_bal_il, il_util, open_rv_12m, open_rv_24m, max_bal_bc, all_util, inq_fi, total_cu_tl, inq_last_12m

Chapter 3

Exploratory Data Analysis (EDA):

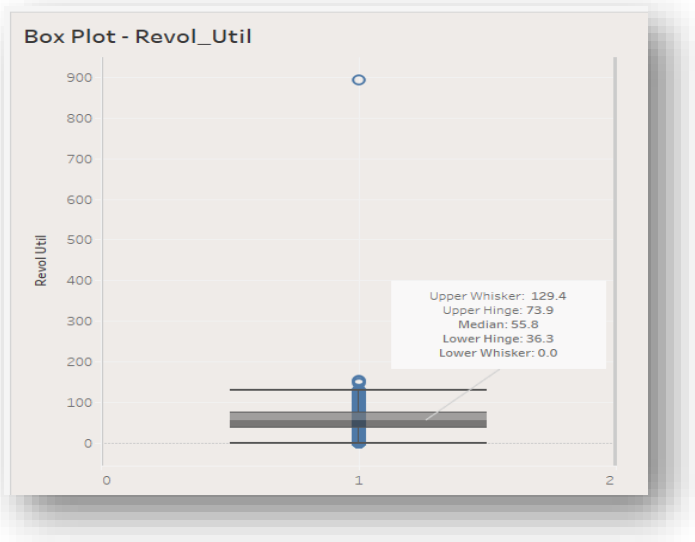
- It is an approach for summarizing, visualizing, and becoming familiar with the important characteristics of a data set.
- It is performed in order to define and refine the selection of variables that will be used for model building.
- It is majorly performed using the following methods:
  - Univariate visualization—provides statistics for a single variable in the data set.

3.1 Purpose for loan application:



- The purpose for individuals applying for loan is shown in the above bubble chart.
- It shows that around 58% individuals applied loan for debt consolidation.
- Credit card constitutes for 19.80% and remaining percent for various purpose

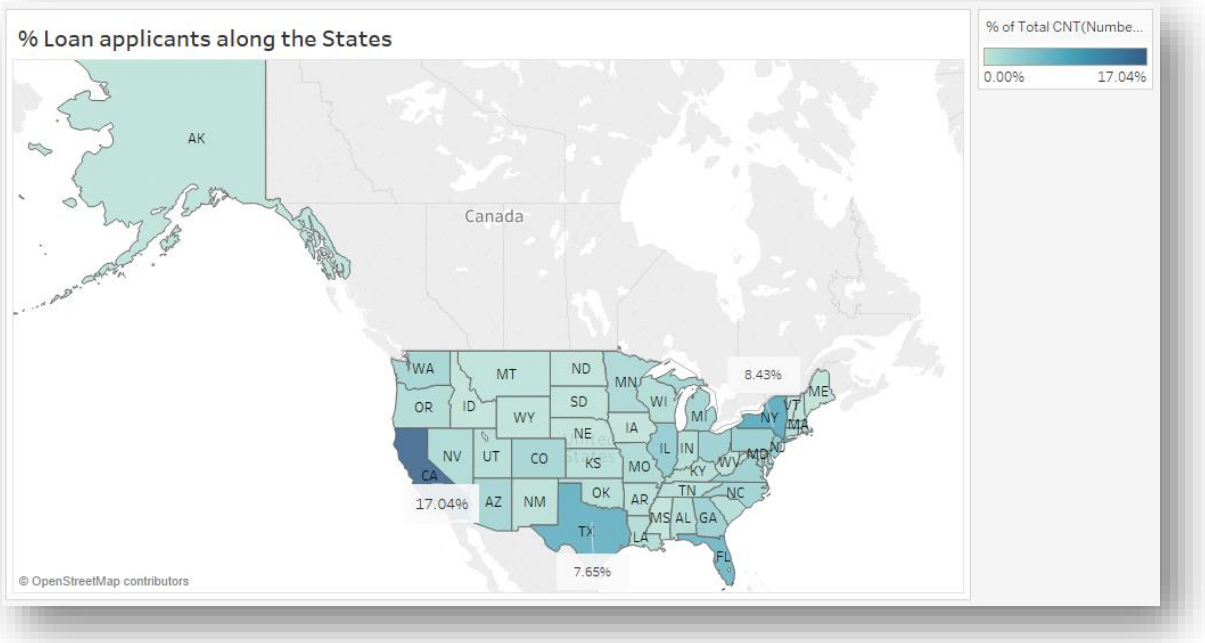
3.2 Outlier and anomalies in Revol_Util Column:



'Revol_Util' –

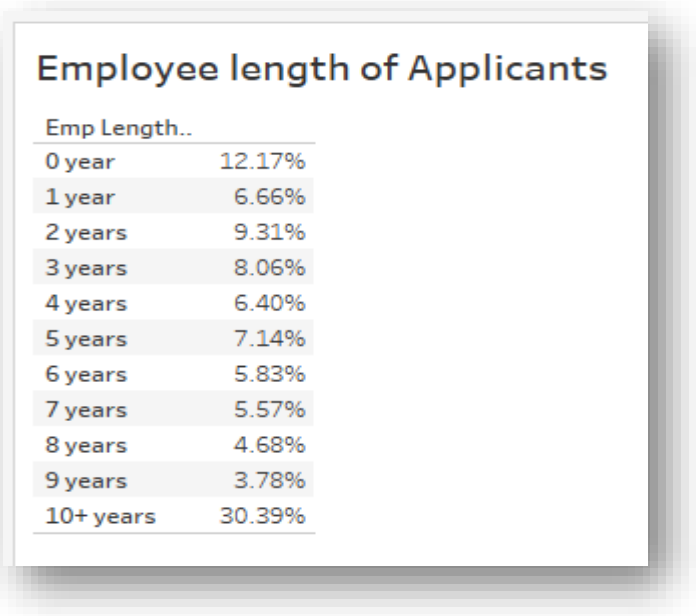
- The amount of credit the borrower is using relative to all available revolving credit.
- This means Revol_util is in percentage.
- It can never be above 100%.
- In our data there are around 300 observations that have value more than 100.
- This is an anomaly in data. Also, an outlier is observed at 892.

3.3 %Loan Applicants along the states:



- The above map shows the percent of applicants who applied for loan State Wise.
- Out of 253030 applicants, majority of loans were applied by individuals residing in California (17.04%).
- The top three States that applied for loan are: California, New York and Texas.
- There were hardly any loan applicants from North Dakota, Nebraska and Iowa (0.00%).

3.4 Experience of Applicants:



Employee length of Applicants

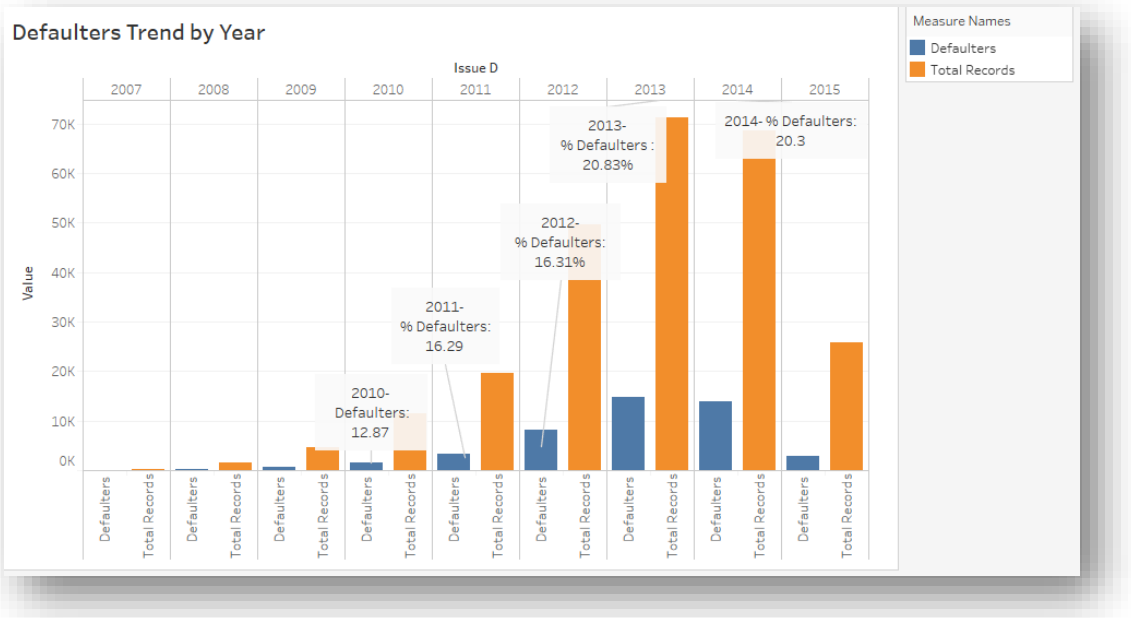
Emp Length..	
0 year	12.17%
1 year	6.66%
2 years	9.31%
3 years	8.06%
4 years	6.40%
5 years	7.14%
6 years	5.83%
7 years	5.57%
8 years	4.68%
9 years	3.78%
10+ years	30.39%

Employee_length column has 12 unique values.:

- n/a – those who don't have any experience.
- >1 years – have less than a year work experience.In the above table, these two values have been included under 0 year.
- The table shows that more than 30% of loan applicants were having 10+ years of experience.

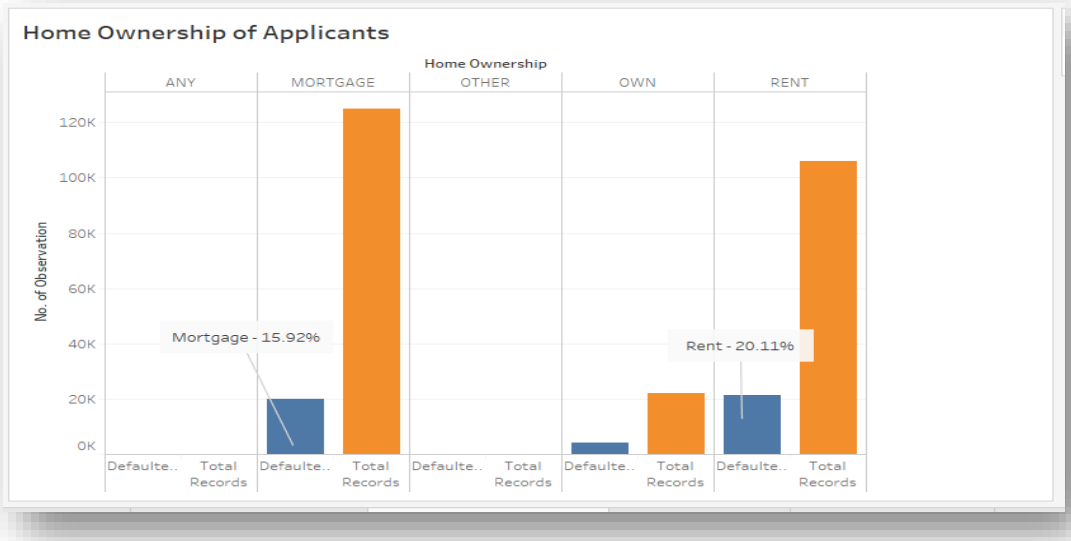
Bivariate visualization- To find the relationship between two variables.

3.5 Defaulters Trend by Issue_d:



- Above graph shows the trend of defaulters with years.
- The defaulter rate seems to increase along the years.
- Looking at this increase, we can say that %Defaulters in 2015 will be equal to or more than 20%.

3.6 Defaulters vs Home Ownership:



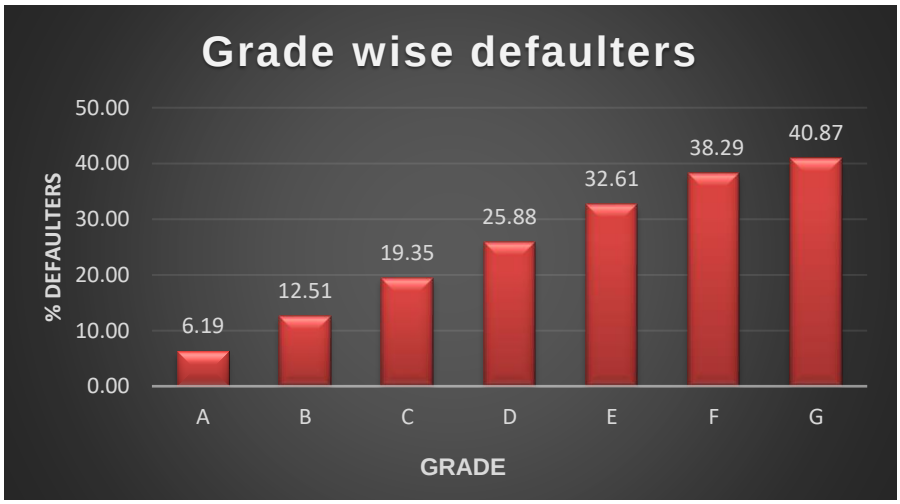
- The above graph shows Home ownership of Loan applicants.

- Majority of individuals who applied for loan belong to ‘Mortgage’ category.
- Total Defaulters in ‘Rent’ category are **20.11%.**

3.7 Relationship between Interest Rate and Grade:

- This graph shows relationship between two independent variables in our dataset, Int_rate and Grade.
- They are linearly related to each other.
- As the Grade moves from A to G, Interest rate increases.
- Maximum interest rate for grade:
  A - 9.63, B – 14.09, C- 17.27, D- 21.31, E-23.4, F-25.99, G-28.99.

3.8 Defaulter rate Grade wise:

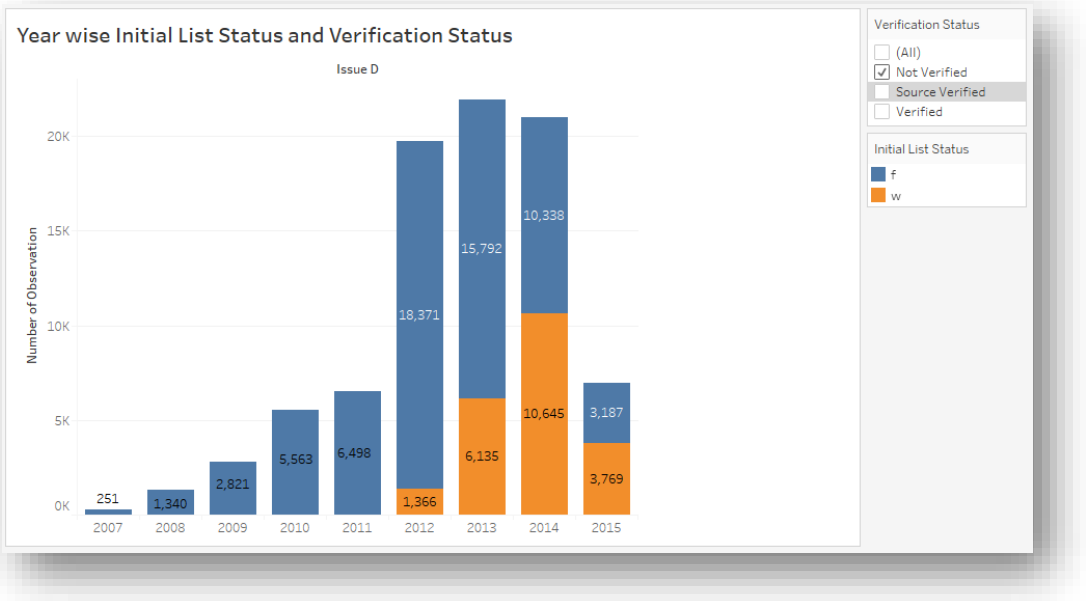


- The above chart shows Grade wise defaulters. As Grade goes from A to G, defaulter rate increases.

40.87% defaulters are observed in grade ‘G’ whereas only 6.19% in grade ‘A’

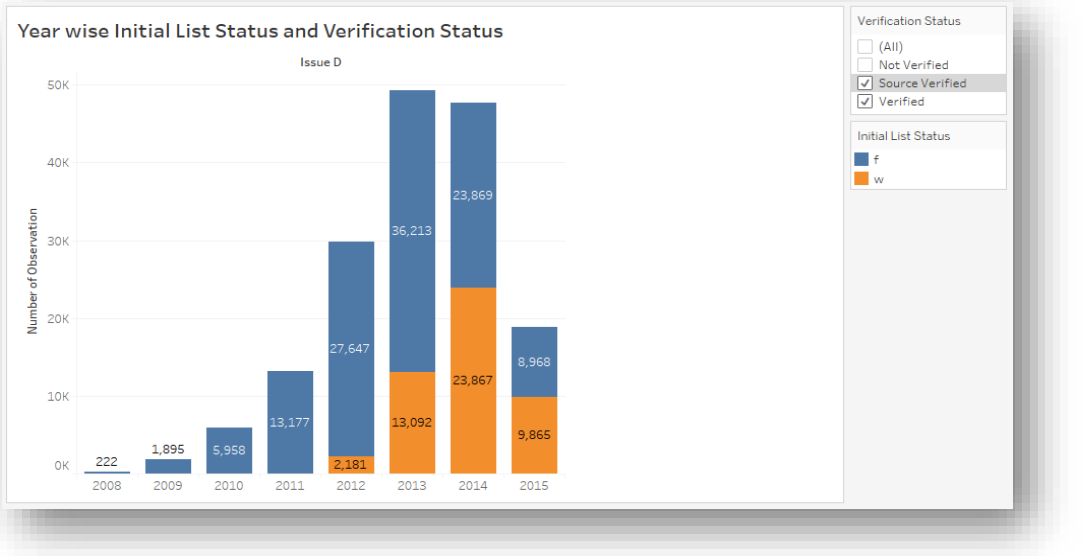
- Multivariate visualization—To understand interactions between different variables in the dataset

3.8 (a) Year wise Initial List Status and Verification Status (Not verified)



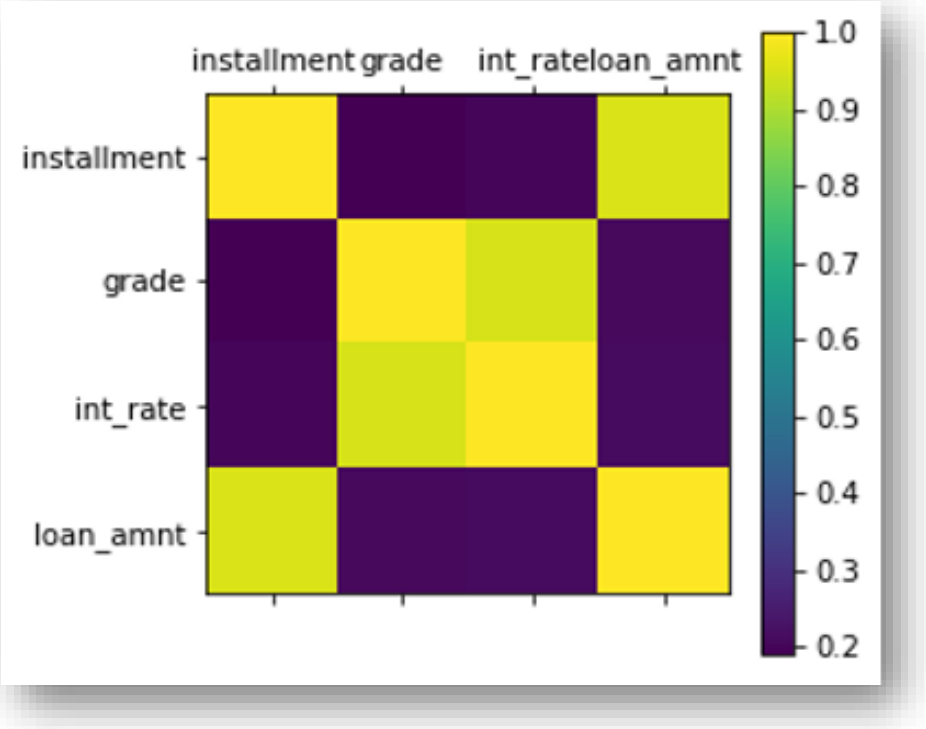
- The above graph shows Year wise distribution of Initial list status with Verification Status as Not verified.
- The range for verification status as ‘Not verified’ is from year 2007 to 2015.
- Initial list status has two values w-whole and f- fractional.
- XYZ Corporation started offering the whole loans only since late 2012.

3.8 (b) Year wise Initial List Status and Verification Status (verified and source verified)



- The above graph shows Year wise distribution of Initial list status with Verification Status as Verified and Source Verified.
- The range for verification status as 'Not verified' is from year 2008 to 2015.
- In 2007, borrower's income was not verified at all.
- In 2013 and 2014, more loans were issued that borrower's income was 'verified' and 'source verified' than the loans with 'not verified' income.
- There is no significant difference in income verification for loans initially listed as fractional or whole.

3.9 Correlation matrix



The above correlation matrix shows a strong positive correlation (approx. 0.9) between two independent Variable:

- Grade and int_rate.
- loan_amnt and Instalment.

Chapter 4

Transforming data:

4.1 Removing columns that have more than 50% missing values:

- 18 variables are having NA values more than 50%. Those are dropped using the code:

```
df2 = df1.dropna (thresh=0.5*len (df1),axis=1)
```

4.2 Removing Irrelevant columns:

Out of 73 variables, 31 are irrelevant for building model.

- **Varibles that leak data from the future:** last_pymnt_d, last_pymnt_amnt, total_rec_prncp, total_rec_int, total_rec_late_fee, recoveries, last_credit_pull_d, collection_recovery_fee, funded_amnt, funded_amnt_inv, issue_d,
- **Variables that contain randomly generated value**: id, member_id, title, earliest_cr_line, pymnt_plan
- **Interrelated variables**: zip_code, out_prncp, out_prncp_inv, total_pymnt, total_pymnt_inv, sub_grade
- **Requires more data to become useful**: emp_title, pymnt_plan, inq_last_6mths

4.3 Treating Outliers:

Outlier is a commonly used terminology by analysts and data scientists as it needs close attention else it can result in wildly wrong estimations. Simply speaking, Outlier is an observation that appears far away and diverges from an overall pattern in a sample.

- Checking all numerical values, we decided to treat (total_rev_hi_lim,tot_coll_amt,tot_cur_bal,revol_bal and revol_util)
- Imputing the Outliers with the 95th percentile value
- Deleting some Outlier Values due to Outlier observation are very small in numbers.

4.4 Treating Missing values:

- There are four numerical type variables (revol_util, total_rev_hi_lim, tot_coll_amt and tot_cur_bal) which have missing values.
- These missing values are to be replaced with mean value. :

```
df1.revol_util.fillna(df1.revol_util.mean(),inplace=True)
df1.total_rev_hi_lim.fillna(df1.total_rev_hi_lim.mean(),inplace=True)
df1.tot_coll_amt.fillna(df1.tot_coll_amt.mean(),inplace=True)
df1.tot_cur_bal.fillna(df1.tot_cur_bal.mean(),inplace=True)
```

4.5 Converting Categorical to numerical:

4.5.1 Ordinal – Grade and Emp_length:

- Emp_length column has 12 unique values.
- Out of 12, one is 'n/a' value which means that the borrower has no experience.
- 'n/a' is different from NA values.
- '< 1 years' & 'n/a' values are replaced with value 0.
- Dictionary is created assigning numerical values to each unique categorical value.
- It is done manually, as Grade and Employee length are ordinal.

4.4.2 Nominal – Home_ownership , Purpose, etc.

- Using Label Encoder function of scikit learn, each value in a column is converted to a number

Chapter 5

Data Preparation:

5.1 Multicollinearity:

- It is a case of multiple regression in which the independent variables are themselves highly correlated.
- When Variance Inflation Factor (VIF) values are high for any of the independent variables, the fit is affected by multicollinearity .
- Using vif technique of scikit learn, two variables ‘loan_amt’ and ‘installment’ were found to be highly correlated.
- Removing either of the variable, will reduce the degree of multicollinearity.

5.2 Dividing the data into active and inactive data:

- The given dataset contains defaulters, non-defaulters and customers who were current during that time.
- The customers under 'current' status are considered as non-defaulters in the dataset.
- The dependent variable ‘default_ind’ has two unique values 0 and 1.
- To find out the customers under ‘current’ status, variable ‘out_prncp’ has been considered.
- This variable stores principal amount that is remaining for total amount funded. Customers having principal amount more than 0 are considered as active and remaining are inactive.
- Model is built on inactive data.
- Using the model that is built on inactive data, ‘default_ind’ for active data is predicted.
- The current loans are also added in the dataset, which is not a good data to get insights from and hence we are removing those rows .

```
df1 = df[df.out_prncp == 0] # 2.5 lacs    Inactive

df1 = df[df.out_prncp > 0] # 6.5 lacs     Active
```

5.3 Dividing into train and test:

- Inactive data is divided into train and test.
- As per the problem statement, we need to split the data into train and test using 'issue_d' variable.
- Train data: From June 2007 to May 2015.
- Test data: From June 2015 to Dec 2015 .

5.4 Data Standardization :

- Standardizing means to rescale the data is such a way that mean is 0 and standard deviation is 1.
- In the data set, we have variables having different ranges.
- Using variables without standardization can give variables with larger ranges greater importance while building model.
- Hence , we are transforming the data to comparable scales that can prevent this problem of building baised model.
- In python, standardization is done using standardscalar function of sklearn.

5.5 Feature selection based on p-value:

- Under null-hypothesis, p-value is the probability of obtaining a result equal to or more extreme than what was actually observed.
- The smaller the p-value, the higher the significance because it tells the investigator that the hypothesis under consideration may not adequately explain the observation.
- Variables having p-value < 0.05 are considered and rest are dropped.
- Three such variables are dropped ('tot_coll_amt', 'revol_bal', 'acc_now_delinq')

5.6 Upsampling:

- It is a process of randomly duplicating the observation of minority class till it becomes equal to majority.
- Since the data is imbalanced, the accuracy would always come around 90%, so we can't consider the accuracy.
- Upsampling can reduce this error to some extent.
- Code :

```
from sklearn.utils import resample

# Separate majority and minority classes

f_majority = df[df.balance==0]

df_minority = df[df.balance==1]

# Upsample minority class

df_minority_upsampled = resample(df_minority,
replace=True,     # sample with replacement
n_samples=576,    # to match majority class
random_state=123) # reproducible results
# Combine majority class with upsampled minority class
df_upsampled = pd.concat([df_majority, df_minority_upsampled])
# Display new class counts
df_upsampled.balance.value_counts()
# 1    576
# 0    576
# Name: balance, dtype: int64
```

6 Model Building:

6.1 Logistic Regression:

6.1.1 Logistic Regression:

```
LOGISTIC REGRESSION on Normal data
CONFUSION MATRIX:
[[7860   80]
 [ 233   11]]
Classification report :
             precision    recall  f1-score   support

          0       0.97      0.99      0.98      7940
          1       0.12      0.05      0.07       244

avg / total       0.95      0.96      0.95      8184

Accuracy of the model: 0.961754643206
AUC: 0.517503200231
```

6.1.2 Logistic Regression with Threshold:

```
Applying Threshold on Logistic with value of 0.4
CONFUSION MATRIX
[[7533  407]
 [ 214   30]]
Accuracy of the model:  0.924120234604
Classification report :
             precision    recall  f1-score   support

          0       0.97      0.95      0.96      7940
          1       0.07      0.12      0.09       244

avg / total       0.95      0.92      0.93      8184

AUC: 0.535845686914
```

6.1.3 Logistic Regression with upsampling :

```
LOGISTIC REGRESSION WITH UPSCALE DATA:
CONFUSION MATRIX:
[[4086 3854]
 [  81  163]]
Classification report :
             precision    recall  f1-score   support

          0       0.98      0.51      0.67      7940
          1       0.04      0.67      0.08       244

avg / total       0.95      0.52      0.66      8184

Accuracy of the model: 0.519183773216
AUC: 0.591321179337
```

➢ 6.2 Decision Tree:

6.2.1 Decision Tree:

```
Decision TREE
Confusion Matrix:
[[6177 1763]
 [ 171   73]]
Accuracy of the model:  0.763685239492
Classification report :
             precision    recall  f1-score   support

          0       0.97      0.78      0.86      7940
          1       0.04      0.30      0.07       244

avg / total       0.95      0.76      0.84      8184

AUC: 0.538570012801
```

6.2.1.1 Decision Tree with Cross Validation:

```
kfold_cv_result 0.712542065239
```

6.2.2 Decision Tree with upsampling:

```
DECISION TREE WITH UPSAMPLING
CONFUSION MATRIX
[[7940    0]
 [ 244    0]]
Accuracy of the model:  0.97018572825
Classification report :
             precision    recall  f1-score   support

          0       0.97      1.00      0.98      7940
          1       0.00      0.00      0.00       244

avg / total       0.94      0.97      0.96      8184

AUC 0.5
```

6.2.2.1 Decision Tree with upsampling using Cross validation:

```
kfold_cv_result 0.712925541317
```

- **6.3 Xgboost:**

6.3 Xgboost:

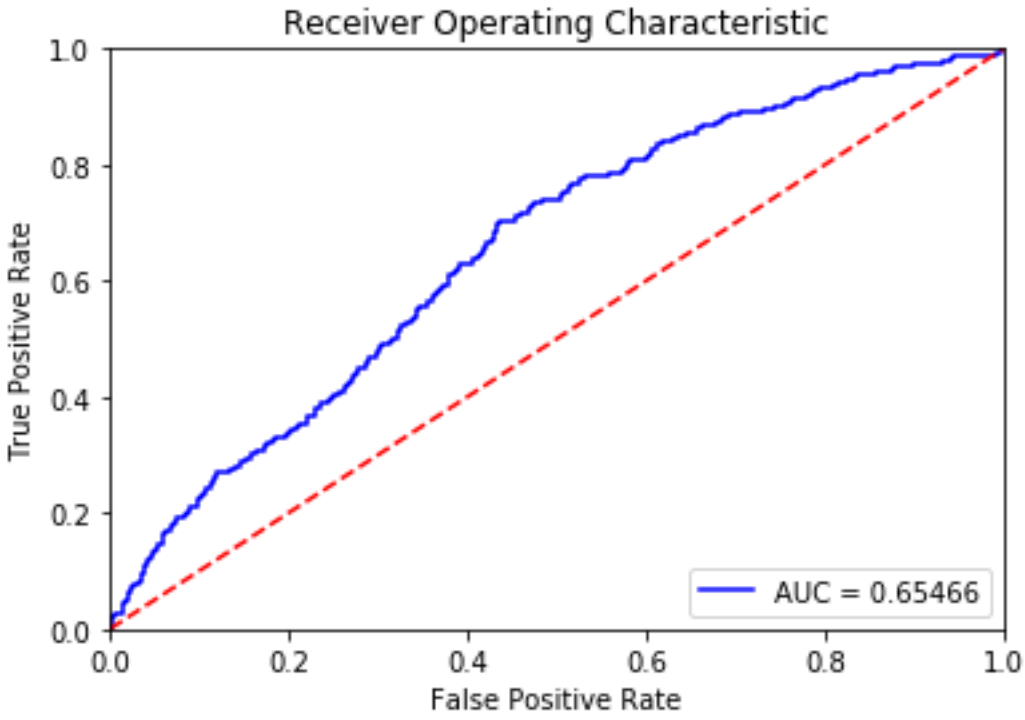
AUC Test:

- AUC TEST USING XGBOOST : 0.654657368791

AUC Train:

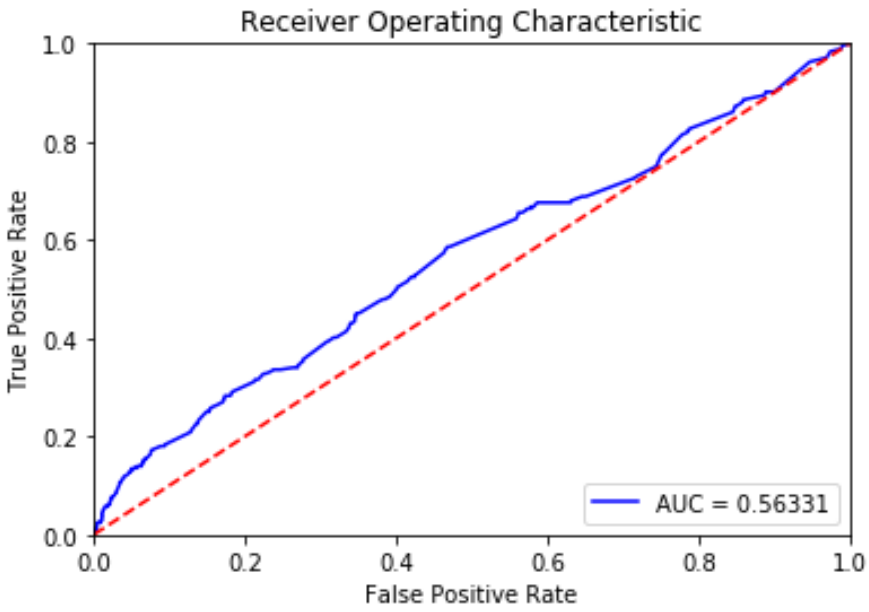
- AUC TRAIN USING XGBOOST : 0.726267091971

AUC GRAPH:



- **6.6 Xgboost upsampling :**

Performance test: 0.563313478135

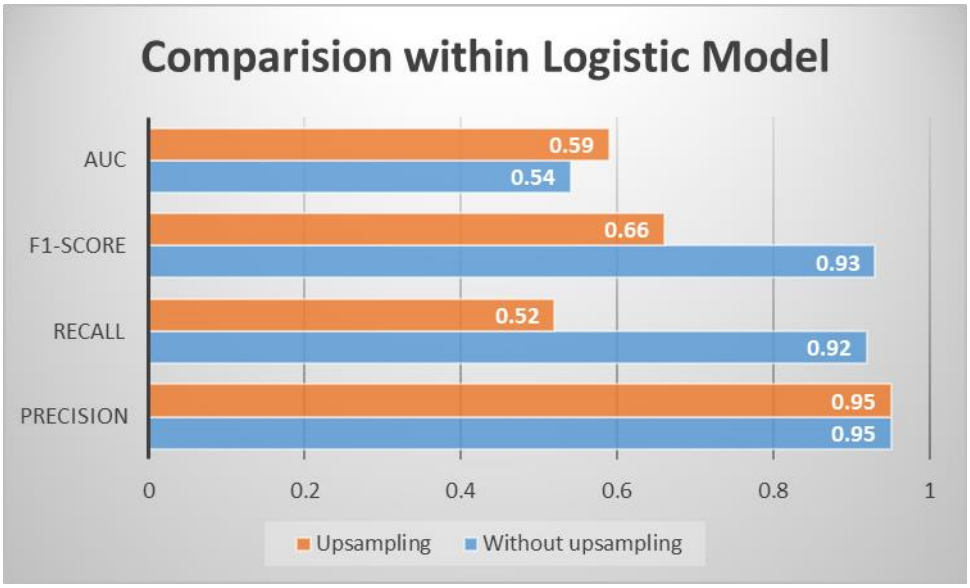

Receiver Operating Characteristic
True Positive Rate
False Positive Rate
AUC = 0.56331
0.0
0.2
0.4
0.6
0.8
1.0

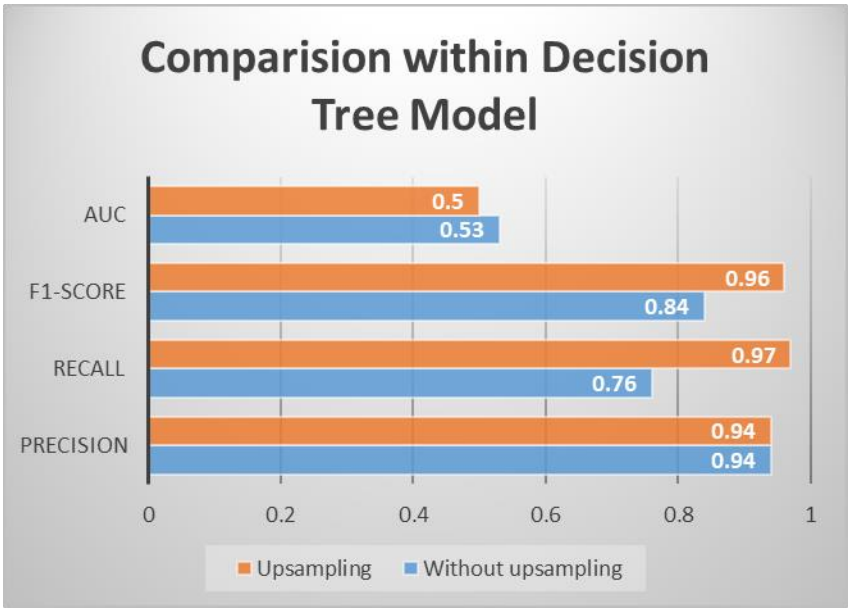
Chapter 7

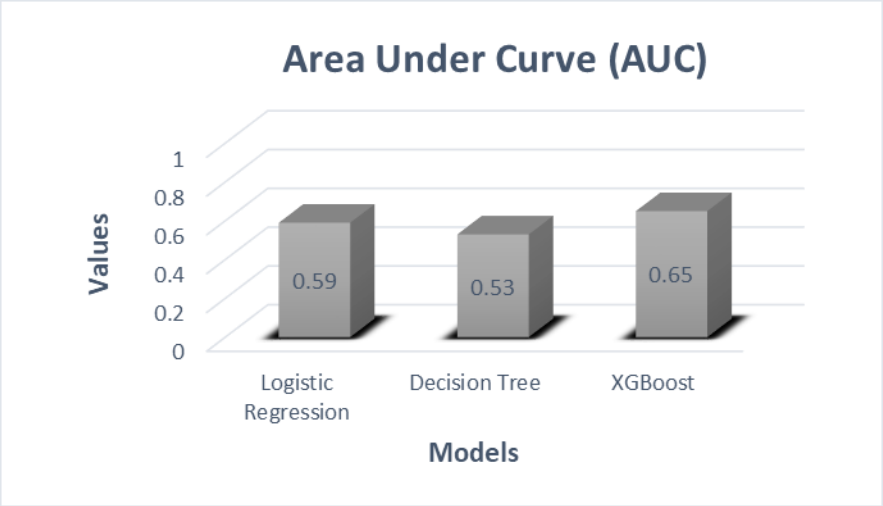
Conclusion:

After applying the models on the Inactive data set we got the vales of AUC

Comparing All the Models:







Going through the Graphs we conclude that XGBOOST (without upsampling) provided the best Model fit so far.

After doing Validation of (Active data) on XGBOOST model the following values were observed:

Class	Number of Records
0	575961
1	19346

➢ References:

- **https://elitedatascience.com/imbalanced-classes**
- **https://machinelearningmastery.com/develop-first-xgboost-model-python-scikit-learn/**
- https://towardsdatascience.com/building-a-logistic-regression-in-python-step-by-step-becd4d56c9c8