**Midterm- Project**

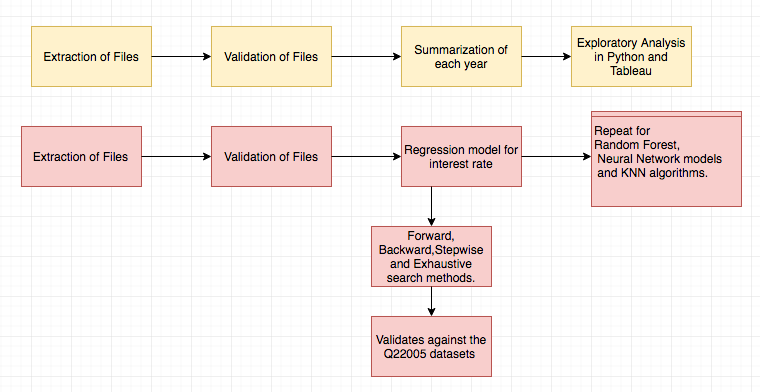
**(Advanced Data Science)**

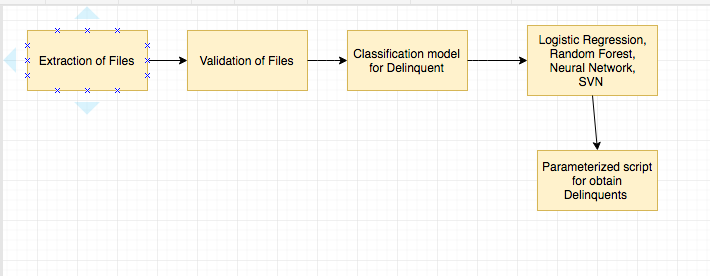
**TEAM 8  
- Antriksh Antriksh  
- Sweta Bajaj**

**- Yogita Jain**

**Freddie Mac Data Wrangling and Data Analysis**

Flow of events:





**Part 1: Data Wrangling**

First Step:

Assuming the user has already created an account and has a valid password sent to his email.

Second step:

Programmatically downloaded the sample data from: https://freddiemac.embs.com/FLoan/Data/download.php

It takes two user arguments: username and password. We have created a session that saves the cookies. In that session the user logs into the website and fetch the sample files from 1999-2016 and extract the csv files and stores them in the current directory.



Third step:

* Sample Performance Files (1999-2016):

Read files into a list. We created separate functions to load data into the data frame file by file, add column names to each file, and validate each of the 23 columns.

For date field we replaced null values with ‘199701’ since this is not included in the range of (1999-2016). For numeric and float columns we replaced ‘nan’ and spaces with 9999 which is not in the range of values in the columns. For certain Boolean columns we did bfill making sure it doesn’t no impact the analysis.

At the end we wrote a for loop looping through the length each file in the range and called each function to perform the validations on every column and store the result into different csv files.

Below are the screenshots of a couple of validations performed.





Summary for Performance Files:

* Read each validated file from current directory and stored into a list. Created separate functions for loading the data, and for each summarization such as loan paid, default loan, terminated loans, remaining loans, loans that were 180 days old and so on.
* Performed computation such as terminated loans, remaining loans, loans that were 180/120 days old total expenses, total loss. (1999-2016).



**DATA EXPLORATION IN PYTHON:**

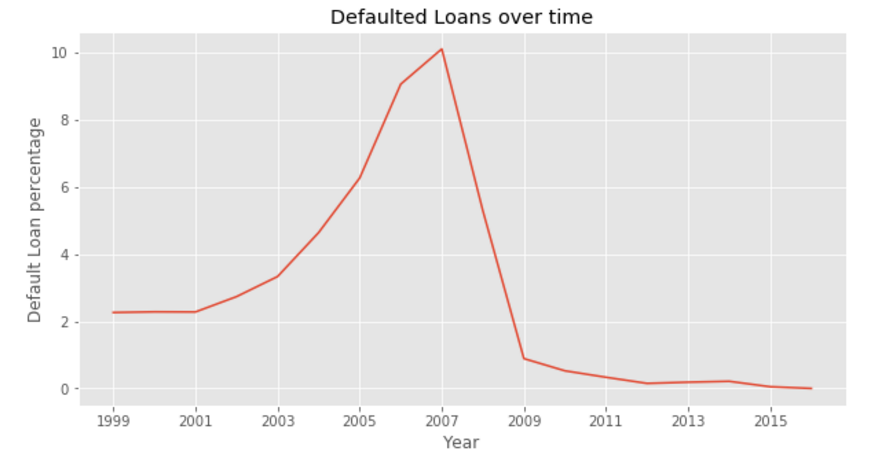
Read the Summary files created in the above steps for Sample Original and Performance files for years 1999-2016 and plot graphs, charts to show the trends of the data.

1. **Defaulted Loans over Time:**

Created a new derived attribute named Loans Defaulted Percentage by calculating

CURRENT\_LOAN\_DELINQUENCY\_STATUS > 2 and setting it to 1.

**Findings**: Default Loan Percentage was plotted for all the years. There is a peak in the defaulted loans for the year 2006-2007 which shows economic slowdown in that time.

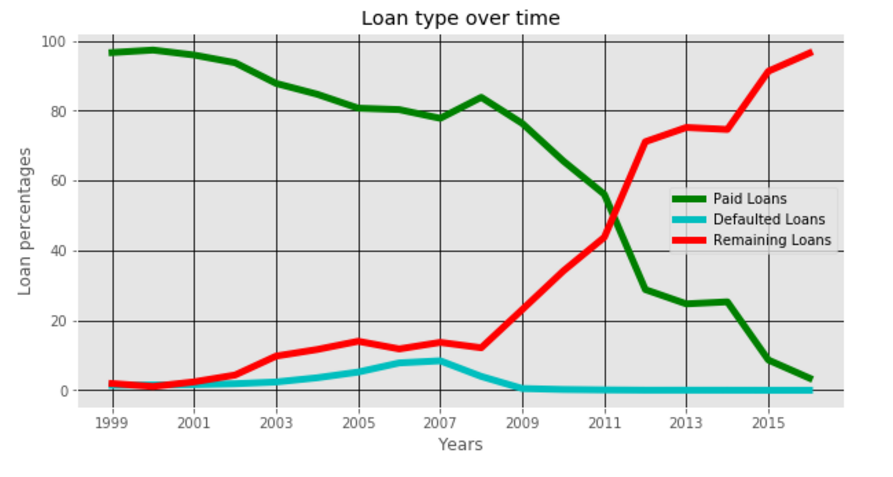


PEAK IN 2007 FOR DEFAULTED LOANS

1. **Loan Type percentage over time:**

Plotted different type of loan percentages over time.  
a) Paid Loans  
b) Defaulted Loans  
c) Remaining Loans (Total Loans – Paid Loans – Defaulted Loans)

**Findings**: Over years, Paid Loans were at the minimum in years 2006-07 and defaulted loans had peaks at the same time . At the time of economic slowdown.

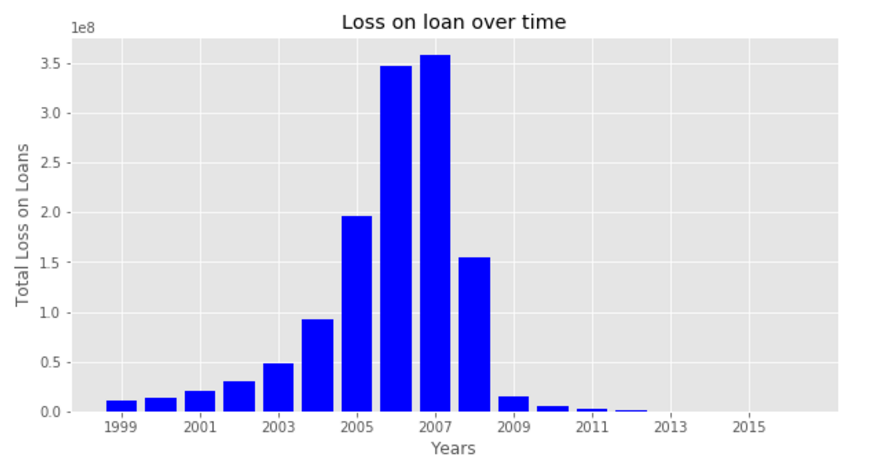


Economic Slowdown period

1. **Loss on Loans:**

Plotted Total Loss incurred by the loan companies over time.

**Findings:** During the economic boom time of 2013 and 1999 there were negligible losses shared by the loan companies, whereas the loss is highest during slowdown period from 2006-07



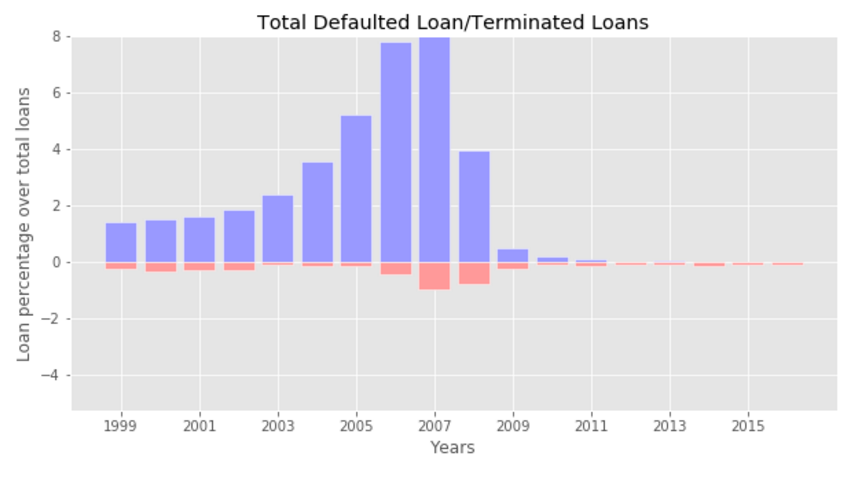
ECONOMIC BOOM

ECONOMIC SLOWDOWN

1. **Defaulted Loans vs Terminated Loans:**

Plot was made of Total percentage of Defaulted Loans over the percentage of loans that were terminated for the years.

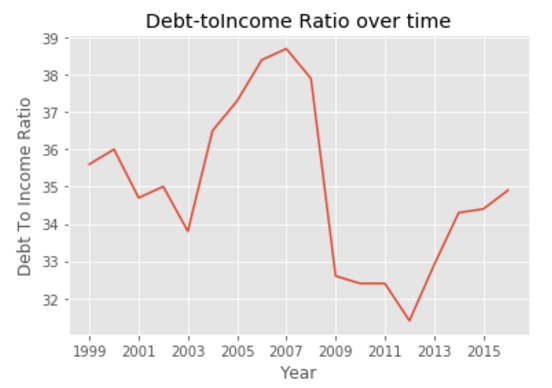
**Findings**: Looking at the trend. There is a peak in both the percentage of defaulted loans and the percentage of terminated loans for years 2006-07 and a gradual low for period of economic boom in 2013.



1. **Debt To Income Ratio over Time:**

Aggregated column for origination file, Debt to Income Ratio, telling the percentage of debt per person vs their total income over the years.

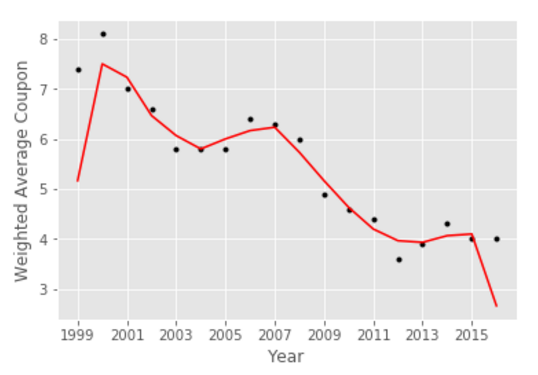
**Findings**: There is higher debt count per person owning to economic slowdown leading to higher debt vs income ratio.



High Debt Peak

1. **Weighted Average Coupon:**

Weighted Average Coupon against running average.   
Findings : No considerable outliers.



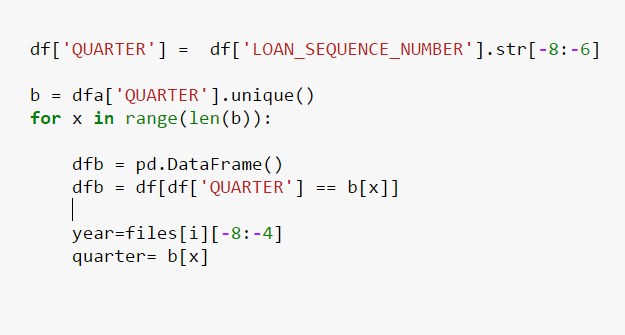
**TABLEAU ANALYSIS:**

1. Sample Original and Performance files were downloaded and validated to remove the null and erroneous records.

**Creating Files:**

**Performance File:**

1. Performance files were divided into quarters using quarter information in Loan Sequence number in the files.



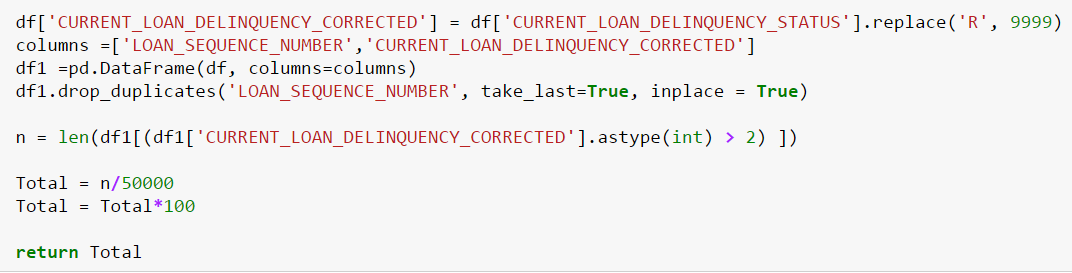
1. Summarization was done for every yearly file and consequently on different quarters, so as to have summation of key values per quarter per year

Following key fields were chosen for summation:

* Percentage of Paid Loans
* Percentage of Defaulted Loans
* Percentage of Repurchased Loans
* Percentage of Modified Loans
* Percentage of loans with Delinquency status 2, 4, 6, R
* Total losses on loans incurred

1. **Derived Attribute: 'Loans Defaulted Percentage'**

To calculate the percentage of defaulted loans over total loans, a new derived attribute was created which calculated all the loans with delinquency status >2 and set it as 1.  
We then calculate sum of all the loans with delinquency status >2 and calculated the defaulted loans.

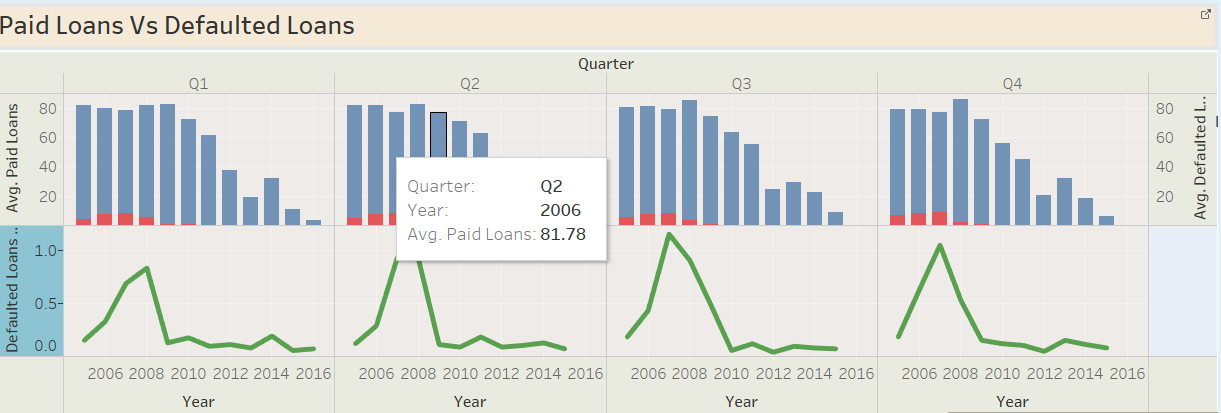


1. Created excel sheet for the data frame and used Tableau to open the csv summation file created.

**Analysis for Performance File:**

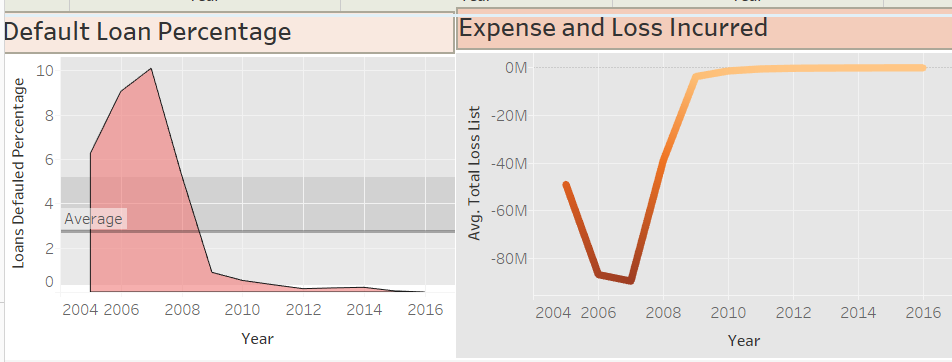
Parameters and analysis Performed:

1. **Paid Loads Vs Defaulted Loans (with Defaulted Loans repurchased percentage) over time**:



**Findings**:

1. We calculated average paid loans for every year and summed for all the quarters.
2. The values in red show the % of loans defaulted for every year in comparison to paid loans.
3. The green lines show the % of defaulted loans which were repurchased
4. Hike in value of defaulted loans in 2007-2008 suggest economic slowdown.
5. **Total Expenses over time plus Loan Defaulted Percentage over time:**

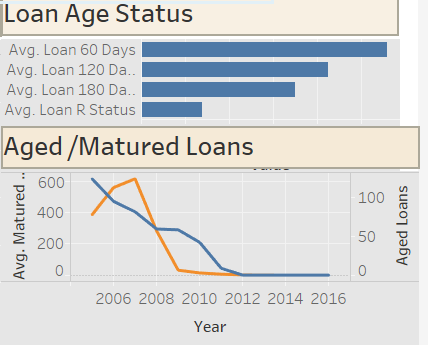


Defaulted loan percentage (derived attribute) showing % of loans which defaulted over time.

Total Loss which is also a derived attribute calculates the total losses incurred (sum of different losses incurred by the insurance company for all the loans)

**Findings:**

1. The default loan percentage is very high for period around economic slowdown i.e 2006-07 but lower for other values
2. The total losses incurred due to defaulted loans is highest for period of economic slowdown.
3. **Loan age status and Aged/Matured loans status over time:**

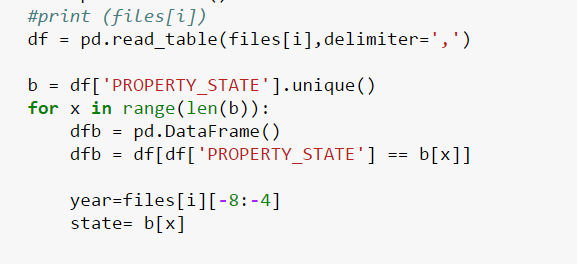


To see the loan status , we plotted the loans based on its delinquency status to see which loans stand where based on their loan age status.  
  
We also looked at Aged/Matured Loans to see trends over time for both.

**Origination File**

Sample Origination files were summarized based on different property states to calculate summarized information relative to each state.

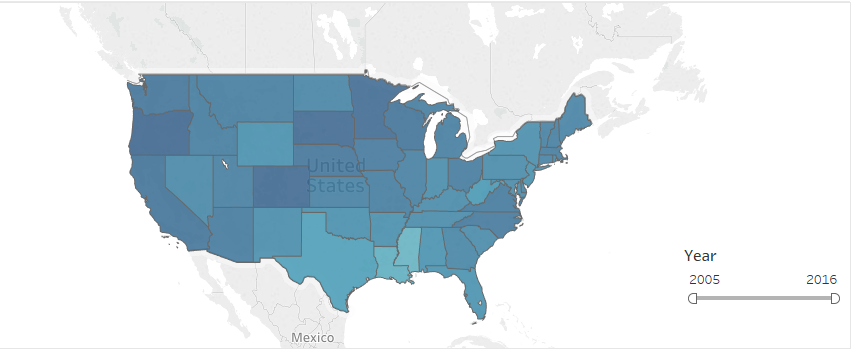
1. Sample origination files were divided into different states:



1. Summarized files was done for every year and for every quarters to calculate the specific fields for analysis :
2. Credit score average for every state
3. Interest rate over time and for every state
4. No of loans tagged as first time buyers
5. No of loans tagged as repeat buyers
6. Units of property taken on loan
7. Debt to income ratio over time
8. Loan to value ratio over time

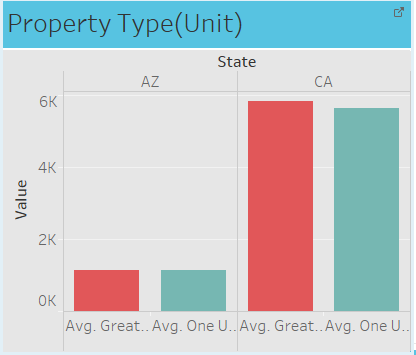
**Analysis done:**

1. **State visualization based on credit score:**



**Findings**:

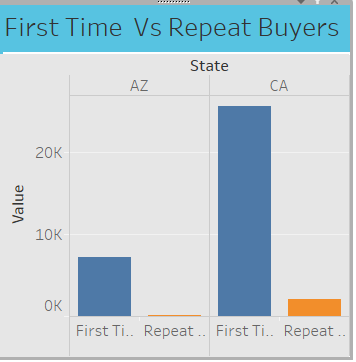
1. The statues with darker tones are years with higher credit scores compared to others.
2. We can check by filtering to each year to see state with respect to values to that individual year.
3. **Property Type Analysis:**



1. Shows the number of units for specified states the no of units for which loan was purchased

No of loans with unit equal to 1  
No of loans with unit greater than 1

1. Could be used for finding the states where loan is taken for bigger units
2. **First Time Buyers vs Repeat Buyers:**

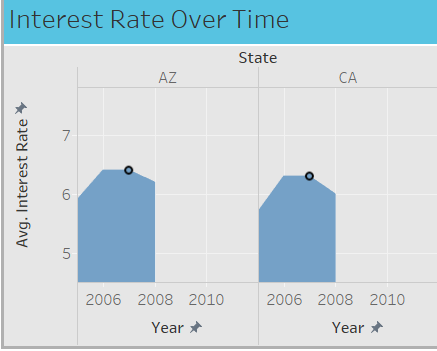


First Time home buyers were calculated for each state and for every year and plotted against repeat buyers who have bought homes many times.

**Findings:**

The first time loan buyers increased in many states considerably before the economic slowdown.

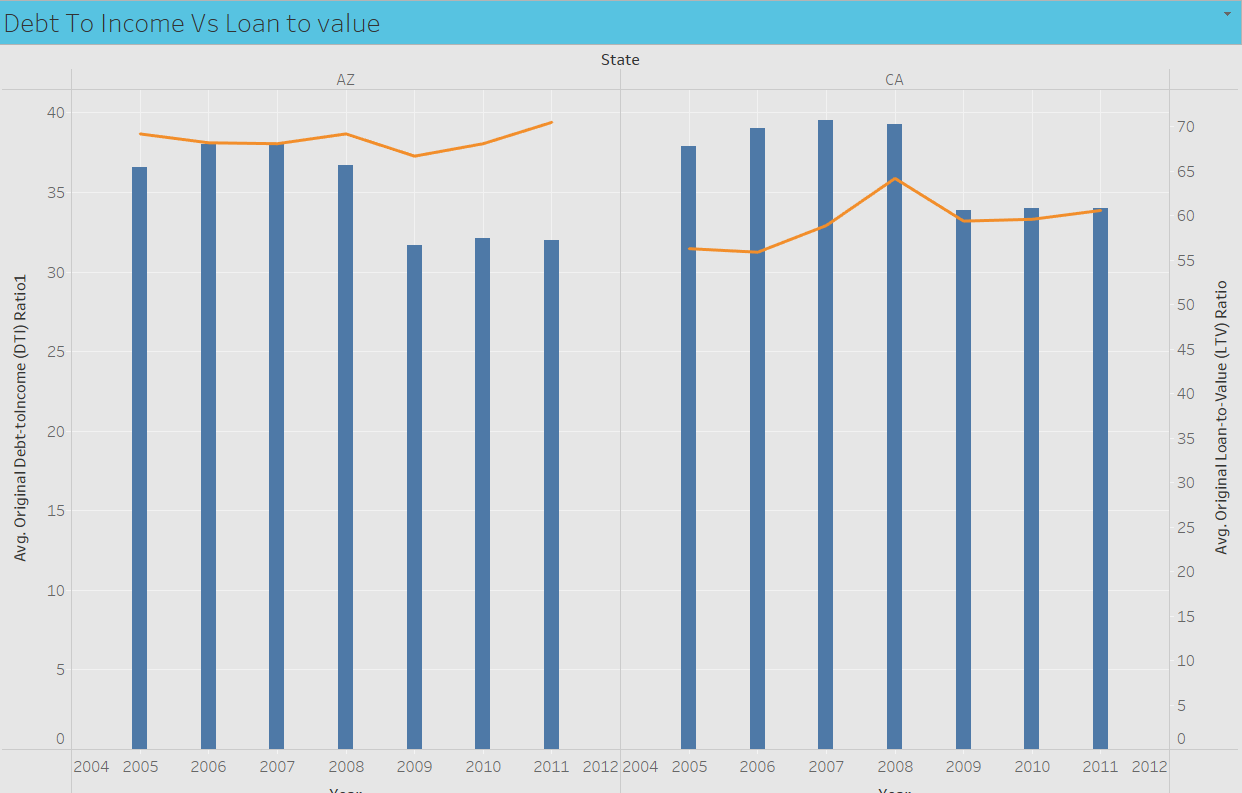
1. **Interest Rate over time:**



Interest rate was plotted for every year for each state to check interest rate trend for each state over time.

**Findings:**

1. Interest rate was considerable higher during the times of economic slowdown.
2. Interest rate decreased gradually to the time of economic boom around 2013
3. **Debt to Loan Ratio plotted against Loan to Value:**



Average Debt to loan ratio was plotted with respect to loan to value ratio for every state over time.

**Findings:**

1. **The average debt to loan ratio is highest for years of economic slowdown.**
2. **Average Linear loan to value is plotted relative to thesedebt to lone value**

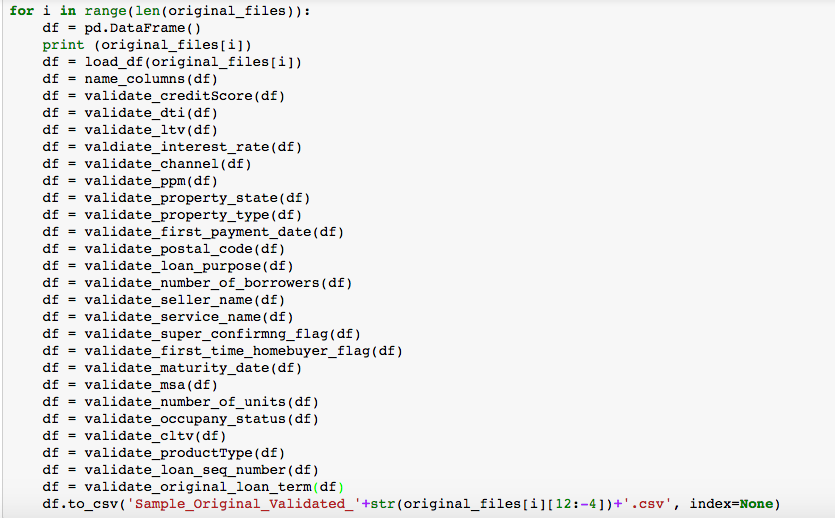
* Sample Origination Files (1999-2016)
* Read files into a list. We created separate functions to load data into the data frame file by file, add column names to each file, and validate each of the 23 columns.

For date field we replaced null values with ‘199701’ since this is not included in the range of (1999-2016). For numeric and float columns we replaced ‘nan’ and spaces with 9999 0r 999 based on the field length, which is not in the given range of values in the columns.

At the end we wrote a for loop that loops through the length each file in the range and called validation function for each and store the result into different csv files.

Below are the screenshots of a couple of validations performed.



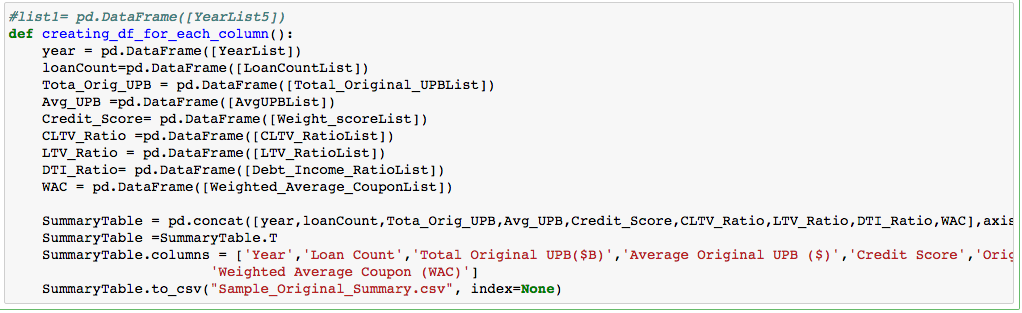


Summary for Origination Files:

Read each validated origination file from the current directory and stored them in a list.

Performed computation such as average UPB, total UPB, credit score, cltv ratio, dti ratio etc for every year’s origination file(1999-2016).

The below function includes code for creating a data frame for each computed column and concatenates them to create a Summary table and stores in into a csv file.



We have also handle scenario where if the code is run again it should empty the already existing list so that it doesn’t not append to the previous values in the list.



Next, we created list for each and every column that we will be creating for summarization. The for loop loops through each and every file and read the data in a data frame and computes the summarization and adds it to the list before the next iteration begins.

Once it is over the creating\_df\_for\_each\_column() function is called to create the data frame and store them in a csv file.





**Prediction** – **Regression Models**

The main goal of regression algorithms is the predict the discrete or a continues value. In some cases, the predicted value can be used to identify the linear relationship between the attributes

The files are loaded and validated in the previous script. Here we have parameterized the input. The user gives two input on the console one for train data and the other for test data. We have built four regression models namely: Random Forrest, Neural Network, KNN and Linear Regression.

a) **Linear Regression:**

Regression analysis is a form of predictive modelling technique which investigates the relationship between a **dependent**(target) and **independent variable (s)** (predictor). This technique is used for forecasting, time series modelling and finding the casual effect relationship between the variables.

Linear Regression establishes a relationship between **dependent variable (Y)** and one or more **independent variables (X)** using a **best fit straight line** (also known as regression line).

It is represented by an equation **Y=a+b\*X + e**, where a is intercept, b is slope of the line and e is error term. This equation can be used to predict the value of target variable based on given predictor variable(s).

In the Linear regression model, we first try and understand what variables should be chosen for regression.

We also use the search criteria to check the importance of the variables and include them accordingly in the model to improve the r-square value

# **Backward Search**

regfit\_backward = regsubsets(ORIGINAL\_INTEREST\_RATE~., data = QuarterDataFeature, really.big=TRUE, method="backward")

regsummary\_backward = summary(regfit\_backward)

# **Forward Search**

# nvmax is the maximum size of subsets

regfit\_forward = regsubsets(ORIGINAL\_INTEREST\_RATE~., data = QuarterDataFeature, really.big=TRUE, method="forward")

regsummary\_forward = summary(regfit\_forward)

A combination of both forward and backward would be the step-wise serach

We look at the continuous variables to regress against the target variable. If we know that the continuous variables are on different scale, we then look for normalizing the variables on one scale.

#**Normalized Data**

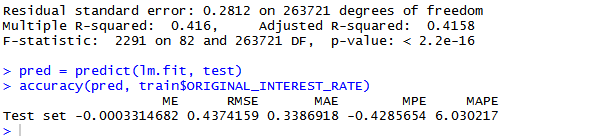
normalized <- function(x) {

(x-min(x))/(max(x)-min(x))

}

Once we get the normalized variables, we then can add our combinational variables to reduce the RMSE of the mode

**Accuracy Parameters:**



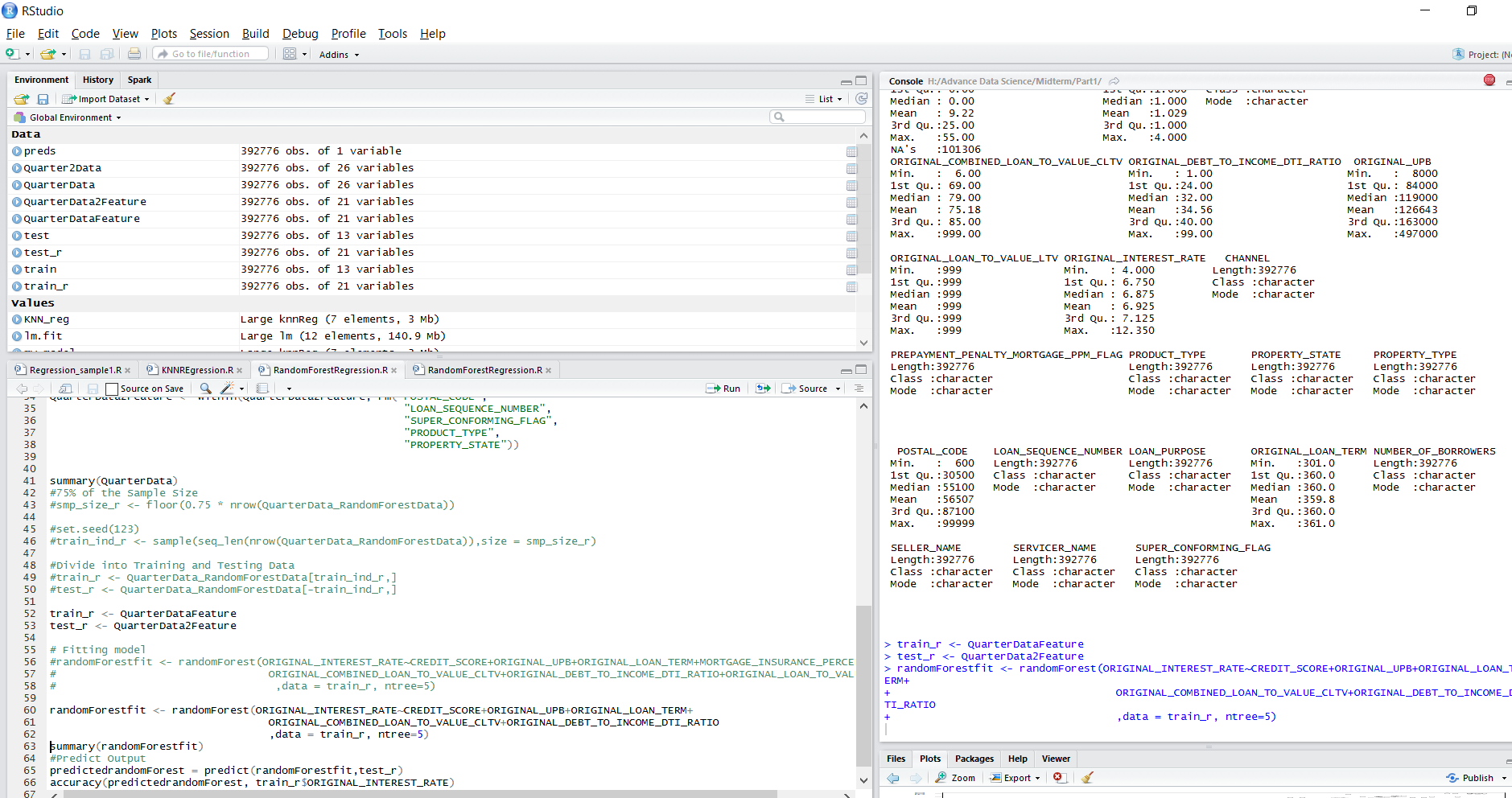
With a suitable adj. r square value and low rmse, we then take the other quarter file to predict the interest rates.

b) **Random Forrest**:

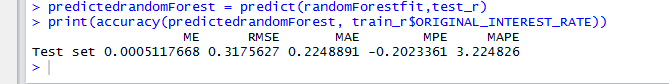
**Random forests** or random decision forestsare an ensemble learning method for regression and other tasks, that operate by constructing a multitude of decision trees at training time and outputting mean prediction (regression) of the individual trees.

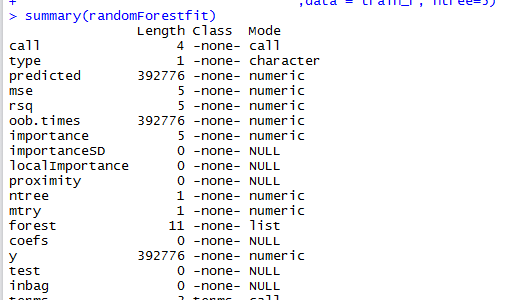
Random Forest regresses over the plane to find the best ensemble model to use. For regression

We have used the randomForest Package and then used the two quarters to make the model and predict the value



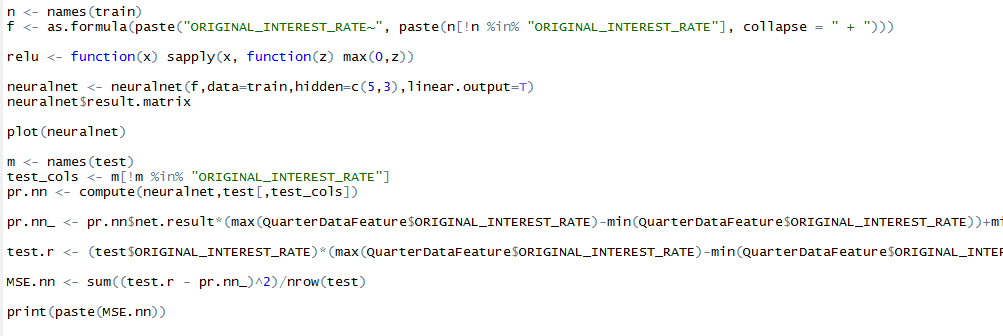
**Accuracy parameters:**



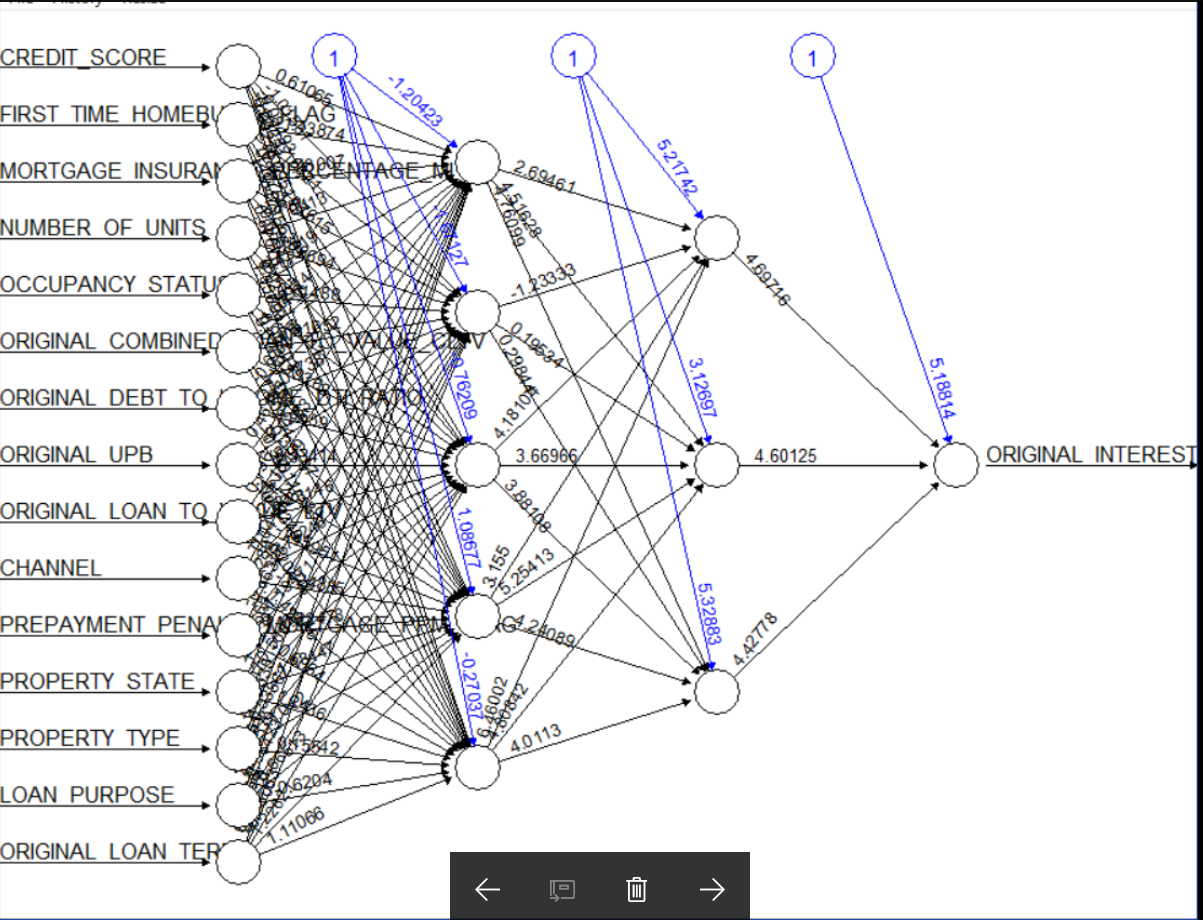


c) **Neural Network:**

An [artificial neural network](https://en.wikipedia.org/wiki/Artificial_neural_network) (ANN) learning algorithm, usually called "neural network" (NN), is a learning algorithm that is inspired by the structure and functional aspects of [biological neural networks](https://en.wikipedia.org/wiki/Biological_neural_networks). Computations are structured in terms of an interconnected group of [artificial neurons](https://en.wikipedia.org/wiki/Artificial_neuron), processing information using a [connectionist](https://en.wikipedia.org/wiki/Connectionism) approach to [computation](https://en.wikipedia.org/wiki/Computation). Modern neural networks are [non-linear](https://en.wikipedia.org/wiki/Non-linear) [statistical](https://en.wikipedia.org/wiki/Statistical) [data modeling](https://en.wikipedia.org/wiki/Data_modeling) tools. They are usually used to model complex relationships between inputs and outputs, to [find patterns](https://en.wikipedia.org/wiki/Pattern_recognition) in data, or to capture the statistical structure in an unknown [joint probability distribution](https://en.wikipedia.org/wiki/Joint_probability_distribution) between observed variables.



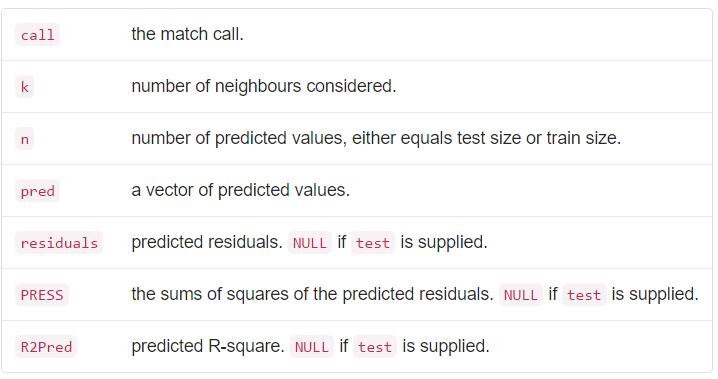
**Neural Network :**

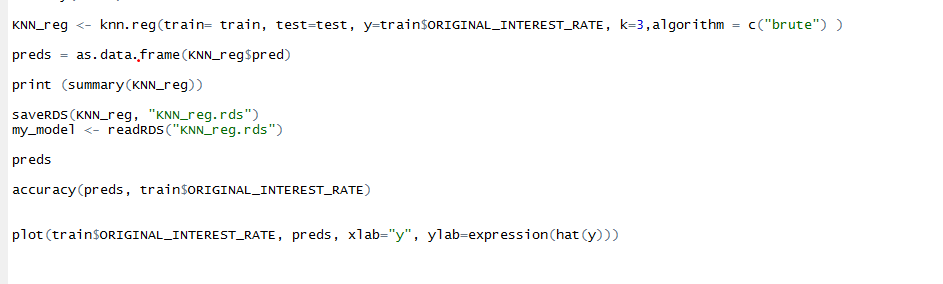


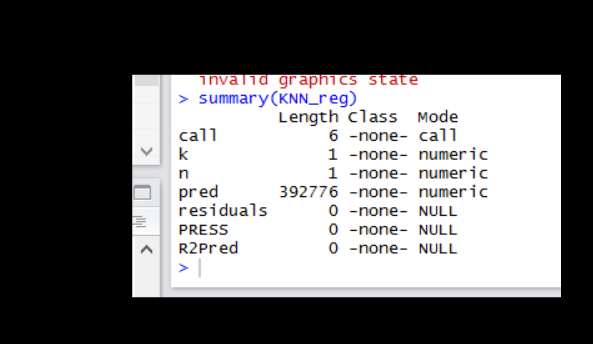
d) **KNN Regression:**

In pattern recognition, the ***k*-nearest neighbor’s algorithm** (***k*-NN**) is a non-parametric method used for classification and regression In *k-NN regression*, the output is the property value for the object. This value is the average of the values of its *k* nearest neighbors.

In KNN we used the FNN library, to regress with three nearest centers and used these parameters to predict







**WHAT-IF ANALYSIS FOR ECONOMIC SLOWDOWN (2007-2009):**

1. **Ran Neural net Regression Model for every quarter from 2007 to 2009**.

Below are the accuracy parameters for every quarter for year 2007 and predicted for next subsequent quarter (4 runs)

The same accuracy parameter was run for every quarter for year 2009 and  
predicted for next subsequent quarter (4 runs)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **TRAIN/TEST DATA** | **ME** | **RMSE** | **MAE** | **MPE** | **MAPE** |
| **YEAR 2007** |  |  |  |  |  |
| Q12007/Q22007 | 0.1241467 | 0.4293535 | 0.3269922 | 1.599496 | 5.103408 |
| Q22007/Q32007 | 0.3922983 | 0.5822884 | 0.4730778 | 5.561261 | 6.926472 |
| Q32007/Q42007 | -0.3612773 | 0.6120397 | 0.5010782 | -6.192616 | 8.147171 |
| Q42007/Q12008 | -0.4245339 | 0.7078234 | 0.5848542 | -8.003672 | 10.3618 |
|  |  |  |  |  |  |
| **YEAR 2009** |  |  |  |  |  |
| Q12009/Q22009 | -0.0442108 | 0.3682598 | 0.2763404 | -1.39357 | 5.679372 |
| Q22009/Q32009 | 0.2983488 | 0.5332135 | 0.4216045 | 5.222166 | 7.940233 |
| Q32009/Q42009 | -0.1532992 | 0.4995381 | 0.3986837 | -3.678846 | 8.298317 |
| Q42009/Q12010 | -0.0158694 | 0.4610557 | 0.3688077 | -0.875003 | 7.582225 |
|  |  |  |  |  |  |

Graph plot for every year for RMSE values for our model over all quarters.

**Findings**:

1. There was linear increase in RMSE for year 2007, with highest shift between Q1 and Q2.
2. The model showed no high deviation for the quarters.
3. There was a similar increase in RMSE for year 2009, in quarter Q1 and Q2.
4. The model deviated from its prediction/accuracy values for the quarter Q1 and Q2 for 2009.
5. **Ran Neural net Regression Model for every quarter for economic boom (1999 and 2013)**.

Below are the accuracy parameters for every quarter for year 1999 and predicted for next subsequent quarter (4 runs)

The same accuracy parameter was run for every quarter for year 2013 and  
predicted for next subsequent quarter (4 runs)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **TRAIN/TEST DATASET** | **ME** | **RMSE** | **MAE** | **MPE** | **MAPE** |
| **YEAR 1999** |  |  |  |  |  |
| Q11999/Q21999 | 0.2243714 | 0.4689823 | 0.3540942 | 2.863739 | 4.822734 |
| Q21999/Q31999 | 0.5186405 | 0.7229008 | 0.5911169 | 6.356283 | 7.430162 |
| Q31999/Q41999 | 0.05270924 | 0.5073762 | 0.3779834 | 0.3841281 | 4.761562 |
| Q41999/Q12000 | 0.00983388 | 0.47325 | 0.3293976 | -0.139271 | 4.153841 |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
| **YEAR 2013** |  |  |  |  |  |
| Q12013/Q22013 | 0.1036262 | 0.6016836 | 0.4758356 | 1.166784 | 13.84661 |
| Q22013/Q32013 | 0.5935808 | 0.9307163 | 0.7744326 | 12.381 | 18.01515 |
| Q32013/Q42013 | 0.1294591 | 0.663996 | 0.5093443 | 1.677615 | 12.08458 |
| Q42013/Q12014 | 0.08086357 | 0.6037905 | 0.4532798 | 0.6159199 | 10.65134 |

Graph plot for every year for RMSE values for our model over all quarters.

Findings:

1. There is a considerable difference in RMSE again for Q1 and Q2 prediction for both 1999 and 2013.
2. The shift in RMSE was due to the shift in the one of the input variables.

**Classification :**

The main goal of classification is to predict the target class (Yes/ No). If the trained model is for predicting any of two target classes. It is known as binary classification. Here we are predicted the derived column Delinquent which is the target class.

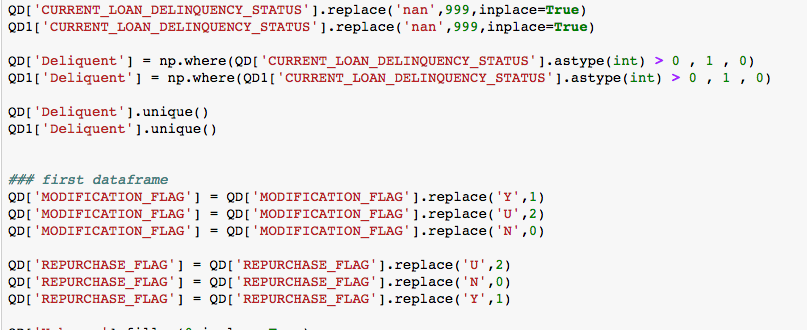
We Programmatically downloaded files from the freddiemac website. The input is parameterized. The user provides two inputs one for test data and the other for train data. We have built four models namely: Random Forest, Neural Network, SVN and Logistic Regression.



We created a file that contains quarters for each year. Once the user provides the input such as Q12005 and Q12006, it will calculate the number of quarters between the range. Using this range we have pulled the required number of files. For now we are reading just two files as we need just two files one for train data and the other for test data, but this can be changed.

We did not select

We created two data frames two read both the files. We make a small check to double check our validation. We created a new column as ‘Delinquent’ which we will be predicting in our all four models. The Delinquent column has 0 and 1 which is based on the column CURRENT LOAN DELIQUENCY STATUS. Whenever the value of the column has a value > 0 we added it as 1 in the delinquent column. Replaced R and XX with 999 which was then converted to 1 in the Delinquent column. We modified the existing columns: Modification and Repurchase flag to have values 0,1,2 for N, Y and U. U is the value we provided during validation for empty spaces and null values.



Once this is done we have our two data frames ready to be trained and tested against.

The column to be predicted is ‘Delinquent’ and this is our y variable ( train\_y and test\_y)

The columns that will contributed to the model our stored in train\_X and test\_X

train\_y and train\_X are the variable of our train data.

test\_y and test\_X are the variables of our test data.

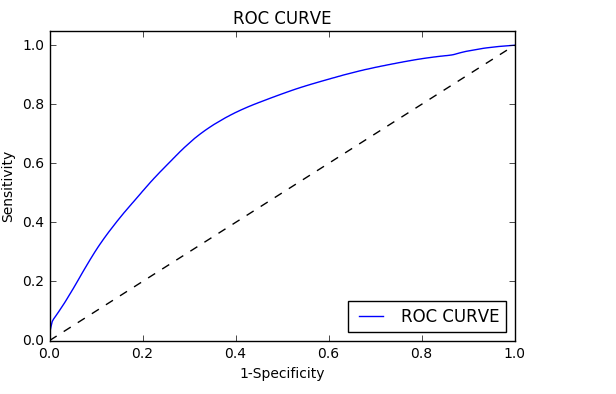


Now, we build the logistic Regression model on X\_train and y\_train variable. And once this is done we use the model to predict X\_test and y\_test. We calculate the accuracy rate of the model using model.score() and predict the accuracy of the next quarter using pd.crosstab() function.



The confusion matrix indicates that: 27649937, 31855 cases were classified correctly and 1314489 and 1071 these were incorrectly classified

The ROC curve for Logistic Regression is:



The model is saved as: LogisticRegression\_model.sav

2) **Random Forest:**

Random Forest is a trademark term for an ensemble of decision trees. In Random Forest, we’ve collection of decision trees (so known as “Forest”). To classify a new object based on attributes, each tree gives a classification and we say the tree “votes” for that class.

The steps to fetching the validated files, taking user input, creating the y variable ‘Deliquent’ and converting the Boolean columns to numeric is same as previous model.

Once the data frames are ready set the Xtrain, y\_train, Xtest and y\_test variables.

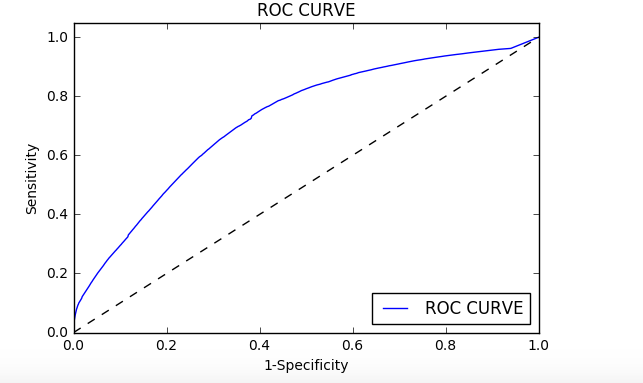
Once the train and test data are ready, cross\_val\_score is calculated and mean of it to check its accuracy. The accuracy for the next quarter is predicted using clf.predict and accuracy\_score() functions.



The confusion matrix is as follows: 2375995, 7009 cases were classified correctly and 8615

and 108381 these were incorrectly classified.

The ROC curve:



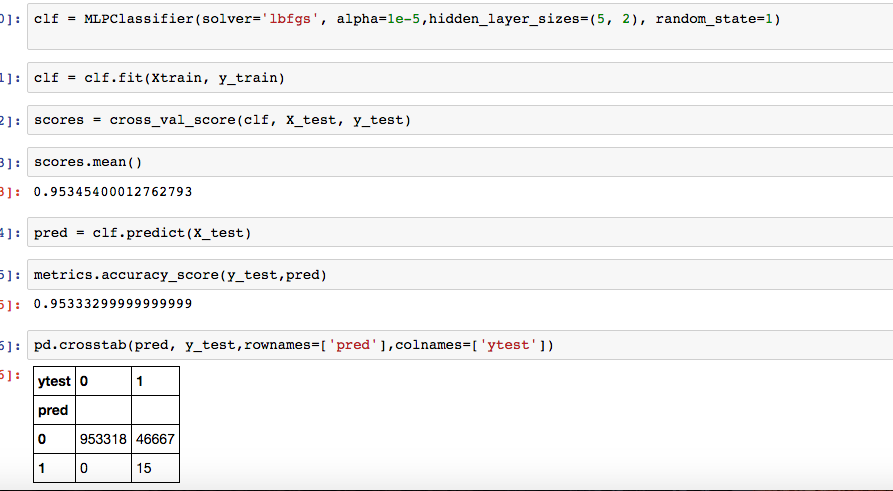
The model is saved as RandomForest\_model.sav

**Neural Network:**

The steps to fetching the validated files, taking user input, creating the y variable ‘Deliquent’ and converting the Boolean columns to numeric is same as previous model.

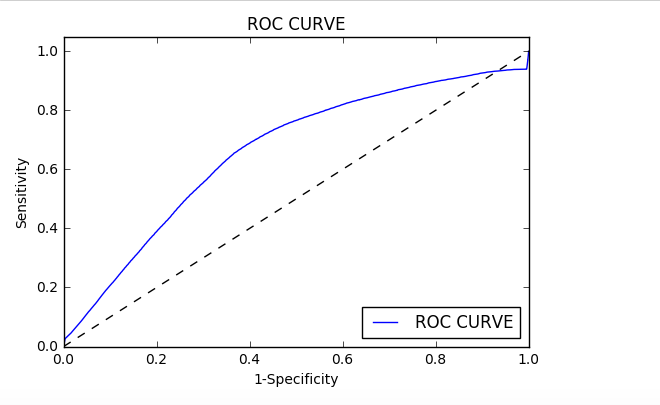
Once the data frames are ready, set the Xtrain, y\_train, Xtest and y\_test variables.

Once the model is ready, we used MLPClassifier to train our model for neural network. Fit the model using clf.fit() and pass Xtrain and ytrain to it. Its mean is calculated and prediction for next quarter is done using clf.predict() and calculates its accuracy.



The confusion matrix: The confusion matrix is as follows: 953318, 15 cases were classified correctly and 46667 these were incorrectly classified.

The ROC curve:



**SVM model:**

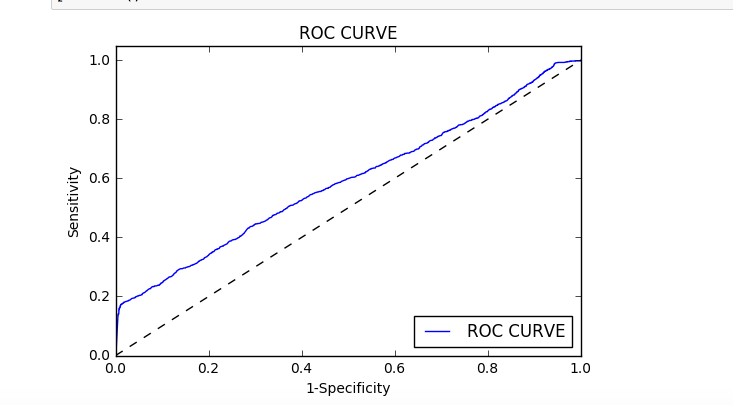
The steps to fetching the validated files, taking user input, creating the y variable ‘Deliquent’ and converting the Boolean columns to numeric is same as previous model.

Once the data frames are ready, set the Xtrain, y\_train, Xtest and y\_test variables.

Once the test and train data ser is ready, we used svm library to train our model. Fit the model using clf.fit() and pass train\_X and train\_Y to it. The prediction for the next quarter is done on X\_test which contains data for next quarter.



The ROC curve:



Comparison: ROC order is as: logistic Regression, Random Forest, Neural, SVN.

* We have parameterized the input to the script. The user gives a range say: Q12005 and Q22005. The script fetches the files.

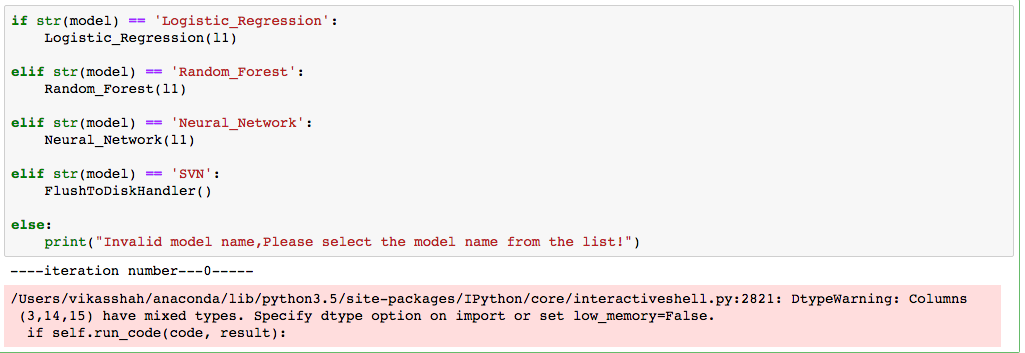
The next user is asked for an input to run a specific model. The model names are prompted on the screen and user can choose any one of them to run. If the model name doesn’t match with anyone then the invalid model name is displayed on the screen. If user has to run another model than re run the docker image and give the model name specified in the list.

Once a valid model name is received the equivalent model will be run and the deliquents for that will be stored in a csv file obtained from the confusion matrix.

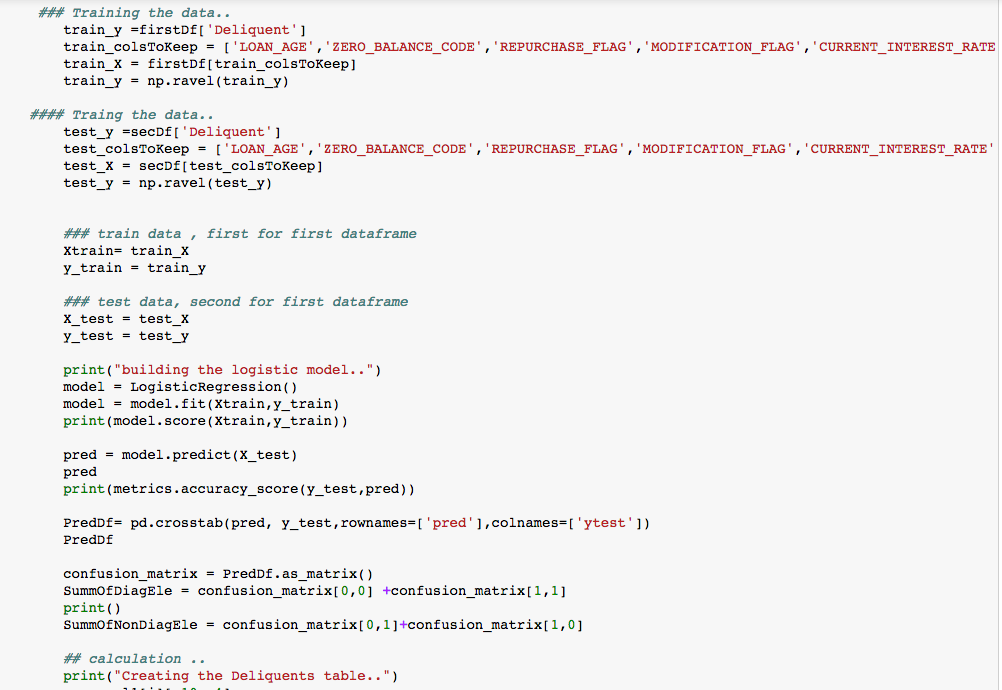
1) prompts for a user input for a model.

../Screen%20Shot%202017-03-17%20at%205.21.29%20PM.png

2) Check if the model exists and if does it will run that model.



3) Glimpse of the Logistic Regression script.



4) Script that will create a deliquients table, stores the values in the columns and outputs the result in a csv file



The same process repeats for any other model that is run.

The same script can be used for running the any number of quarters or for all quarters i.e. Q11999-Q42015. The user will select the model and the script will run for provided number of years and quarters and outputs the result in a csv file.

