Advanced Data Science & Architecture

Midterm Project

- *Under the guidance of Sri Krishnamurthy*

Compiled By,

Mohit Mittal,
Sneha Ravikumar,
Taj Poovaiah

CONTENTS

STEPS TO RUN THE CODES

USING DOCKER IMAGE

Use the link below, find the image pptaj/ads:latest

https://cloud.docker.com/app/pptaj/repository/docker/pptaj/ads/general

RUN $docker pull pptaj/ads:latest

RUN $docker run –i –t pptaj/ads:latest to run the image

Part1:
1. Open the terminal in the directory "1_Final_Code"
2. To run the program enter following command without quotes:
   "python part1.py Summarize_data --local-scheduler"
   Pre-requisites to run the program:
   - python3.x
   - pip
   - python libraries: luigi, beautifulsoup, mechanicalsoup, glob, pandas, numpy
   The program will ask you for the username and password for freddiemac.embs.com website to download the data.
3. Enter the username and password and it will run the tasks to download the data, clean the data and summarize it

4. The downloaded files can be found in the "1_Final_Code/downloads" directory, the cleaned files can be found in the "1_Final_Code/cleaned" directory and the summaries can be found in the "1_Final_Code/summary" directory

5. The python notebook for the summary can be found in the "1_Final_Code" directory with the name "New_Summary_Performance". The tableu files can be found in the directory ""1_Final_Code/TableauFiles"

Part2:
1. Open the terminal in the directory "1_Final_Code"
2. To run the program enter following command without quotes:
   "python part2.py Build_prediction_model --local-scheduler"
   Pre-requisites to run the program:
   - python3.x
   - pip
   - python libraries: luigi, beautifulsoup, mechanicalsoup, glob, pandas, numpy
   The program will ask you for the username and password for freddiemac.embs.com website to download the data.

3. Enter the username and password. Enter the year and quarter you want to run the prediction model for. it will run the tasks to download the data, clean the data and summarize it.

4. The downloaded files can be found in the "1_Final_Code/downloads" directory and the cleaned files can be found in the "1_Final_Code/cleaned" directory.

5. Run the "Classification_Logistic_Regression.R" in RStudio to run the logistic regression for Delinquent, "neuralnet.R" in RStudio to run the neural network for Delinquent.

Programming Language used : Python

Workflow Manager User: Luigi

Tasks

- **A. Downloading Data**
- **B. Clean Origination Data**
- **C. Clean Performance Data**
- **D. Summarizing Origination Data**
- **E. Summarizing Performance Data**

DOWNLOADING DATA:

File Location : Classes/Part1/Download_sf_loan.py

Task Requires no prior tasks to be completed.
Output of the task are all the sample origination and performance files.
Process:

- Asking user for username and password.
- Creating a browser agent (using the mechanicalsoup library) to store and pass the cookies
- Logging in with the user's credentials
- Checking if the user is succesfuly logged in or not.
- Landing to the page that contains the list of files and download links
- Putting the table of files in a dataframe
- Iterating through the rows in dataframe for the links that contain sample files and downloading them to a newly created (if it doesn't already exist) "Downloads" directory
- The program also checks if the files are already present in the "Downloads" directory. It skips the downloading if the file already exists.
- Unzipping the downloaded file.

ORIGINATION FILE OBSERVATION AND CLEANING

CREDIT SCORE: DELETED THE ROWS THAT HAD MISSING CREDIT SCORE

a. Cannot replace missing values as it is explicitly specified that credit score can be either less than 301 or greater than 850.
b. Number of such instances is very less (0.002% in 2016 to 1.242% in 2000)
c. Removing the rows that have blank values and nulls for credit score.

CREDIT SCORE	COUNT OF BLANKS	YEAR
	362	1999
	621	2000
	274	2001
	201	2002
	33	2003
	42	2004
	24	2005
	39	2006
	29	2007
	29	2008
	1	2009
	1	2011
	3	2013
	2	2014
	1	2016

FIRST PAYMENT DATE: NO MISSING VALUES IN SAMPLE FILES

FIRST TIME HOMEBUYER FLAG:

a. If blank it can be replaced by NA if Occupancy Status is either "I" or "S" (Investement property or Second Home)
b. If blank it can replaced by NA if Loan Purpose is either "C" or "N" (Refinance)
c. If blank, then replace it with NA
d. Created three columns for – First Time HomeBuyer Flag YES (1,0) , NO(1,0) and NA(1,0)

MATURITY DATE:

d. No missing values in the sample files
e. Splitting Maturity year and month

METROPOLITAN STATISTICAL AREA(MSA) OR METROPOLITAN DIVISION:

f. Replaced missing values with zero.
g. Derived a new column for Metropolitan Area Flag, that had values in it
h. Future Scope: Compare the values of zip codes, if the zip code belongs to a MSA or MD, then map the msa or md code in the data.

YEAR	COUNT OF BLANKS
1999	7640

2000	7542
2001	6978
2002	7309
2003	7182
2004	7844
2005	7913
2006	8209
2007	8671
2008	7729
2009	7528
2010	7022
2011	6944
2012	6593
2013	5475
2014	5030
2015	4845
2016	1184

MORTGAGE INSURANCE PERCENTAGE (MI%):

MORTGAGE INSURANCE PERCENTAGE (MI %)	COUNT	YEAR	Percentage
	9026	1999	18.052
0	21885	1999	43.77
	44	2000	0.088
0	32764	2000	65.528
	58	2001	0.116
0	36990	2001	73.98
	11	2002	0.022
0	38304	2002	76.608
	10	2003	0.02
0	40083	2003	80.166
	9	2004	0.018
0	40437	2004	80.874
	57	2005	0.114
0	43136	2005	86.272
0	43086	2006	86.172
0	39839	2007	79.678
0	40958	2008	81.916
0	46460	2009	92.92

0	46266	2010	92.532
0	44985	2011	89.97
0	43711	2012	87.422
0	40459	2013	80.918
0	36478	2014	72.956
0	37309	2015	74.618
0	9481	2016	18.962

i. Zero means No Mortgage insurance
j. Blanks Means either less than 1% or greater than 55%, so the replacement cannot be generalized in this case. Also, such cases are ~18% in 1999 and ~0.01% in until 2005 and 0 in the later years.
k. Deriving a new column for mortgage insurance flag is done, where the value is kept No if MI% is zero, otherwise it is made Yes

NUMBER OF UNITS:

	1	2000
	7	2004

l. No missing values for most sample files. Only 1 in the year 2000 and 7 cases in 2004 where number of units is missing
m. Replaced it with the mode OR Discard the row

OCCUPANCY STATUS:

n. No missing values in the sample files.
o. Handled the missing value by replacing it by mode or discarding the rows

ORIGINAL COMBINED LOAN-TO-VALUE(CLTV):

ORIGINAL COMBINED LOAN-TO-VALUE (CLTV)	COUNT	Year
	0	1999
	3	2000
	2	2001
	4	2002
	3	2003
	3	2004
	6	2005
	1	2006

	2	2007
	0	2008
	0	2009
	1	2010
	0	2011
	2	2012
	2	2013
	1	2014
	2	2015
	0	2016

   p. ~0.01% missing values in the sample files.
   q. If the LTV is less than 80 or greater than 200 or unknown, then this column is unknown. Also if CLTV is less than LTV then, CLTV is set to unknown.
   r. This value is dependent on each individual case, so may not be replaced by mean, median or mode.

2. ORIGINAL DEBT-TO-INCOME (DTI) RATIO:
   a. Ratio greater than 65% are represented as spaces. We replaced it by 70.
   b. Unknowns are represented by null, which we replaced by the median.
3. ORIGINAL UPB:
   a. No missing values in the sample files
   b. If value is missing then discard the rows.

4. ORIGINAL LOAN-TO-VALUE:
   a. Ratios below 6% and greater than 105% are unknown.

ORIGINAL LOAN-TO-VALUE (LTV)	COUNT	YEAR
	0	1999
	2	2000
	1	2001
	1	2002
	3	2003
	3	2004
	6	2005
	1	2006
	2	2007
	0	2008
	0	2009
	1	2010
	0	2011
	2	2012
	2	2013
	1	2014

	2	2015
	0	2016

b. Close to zero percent of such occurrence. But, replacing of the values with mean/median cannot be justified as it is specifically said that these values are either less than 6 or greater than 105. So, discarding such rows.

ORIGINAL INTEREST RATE:

c. No missing values
d. If value is missing then replace by median

CHANNEL:

e. No missing values in sample files
f. If values are missing then replace by mode

PREPAYMENT PENALTY MORTGAGE (PPM) FLAG:

PREPAYMENT PENALTY MORTGAGE (PPM) FLAG	COUNT	YEAR
	1247	1999
	236	2000
	122	2001
	171	2002
	198	2003
	73	2004
	49	2005
	65	2006
	113	2007
	1039	2008
	317	2009
	336	2010
	580	2011
	39	2012
	4	2013
	11	2014
	41	2015
	7	2016

g. Most number of blanks (unknown) in the year 1999 -> 2.49%, 2008 -> 2.078%

1999	**48753**
N	48491

Y	262
2000	**49764**
N	49737
Y	27
2001	**49878**
N	49867
Y	11
2002	**49829**
N	49784
Y	45
2003	**49802**
N	49652
Y	150
2004	**49927**
N	49752
Y	175

h. Maximum are “N” throughout the years. 97.5% in 1999, 99.5% in 2000…
i. We are replacing unknown(blanks) values by mode as it wouldn’t affect the distribution.

PRODUCT TYPE:

j. No missing values found in the observations
k. If there are any missing values, then it is replaced with “FRM”

PROPERTY STATE:

l. No missing values found in the observations
m. If there are any missing values, then it is replaced with “Unknown”

PROPERTY TYPE:

PROPERTY TYPE	COUNT	YEAR
	8	2000
	11	2001
	3	2002
	14	2004

PROPERTY TYPE	COUNT	YEAR
	8	2000
CO	4090	2000
CP	74	2000
LH	15	2000

MH	244	2000
PU	6531	2000
SF	39038	2000
	11	2001
CO	3546	2001
CP	45	2001
LH	22	2001
MH	181	2001
PU	5470	2001
SF	40725	2001
	3	2002
CO	3399	2002
CP	48	2002
LH	12	2002
MH	274	2002
PU	5053	2002
SF	41211	2002
	14	2004
CO	3616	2004
CP	210	2004
LH	35	2004
MH	529	2004
PU	6829	2004
SF	38767	2004

n. No missing values for most of the years.
o. Very few missing values observed for years 2000, 2001, 2002 and 2004.
p. Replaced the missing values with the mode (“SF” as observed) because most number of records are categorized as Single Family Home (77% to 82%)

POSTAL CODE:

POSTAL CODE	COUNT	YEAR
	1	1999
	72	2000
	1	2001
	1	2002
	0	2003
	0	2004

	1	2005
	0	2006
	0	2007
	0	2008
	0	2009
	0	2010
	0	2011
	0	2012
	0	2013
	0	2014
	0	2015
	0	2016

q. 72 of 50000 unknowns in 2000, 1 row each in 1999, 2001,2002 and 2005 of unknowns
r. Replaced the blanks with 99999 as unknown value
s. Future Scope: Get a complete dictionary of Metropolitan Statistical Area or Metropolitan Division codes and map the MSA or MD for the row to the dictionary to find the missing postal code

LOAN SEQUQENCE NUMBER:

t. Unique Identifier Column.
u. No missing values. If the value is missing for a row, then replace by random Loan sequence number the complete row or generating a unique identifier UUID
v. Derived two new columns for origination year and origination quarter

LOAN PURPOSE:

w. No missing values in the sample files.
x. If the values are missing then, loan purpose is unknown. Assuming that the percentage of such occurrence in the yearly data would be close (if not equal to) 0%, and it wouldn't affect the distribution of the data, we replaced it by the mode of the column

ORIGINAL LOAN TERM:

y. No missing values observed.

NUMBER OF BORROWERS:

NUMBER OF BORROWERS	COUNT	YEAR
	30	1999
	20	2000
	11	2001
	9	2002
	7	2003

	14	2004
	17	2005
	17	2006
	23	2007
	19	2008
	6	2009
	0	2010
	0	2011
	0	2012
	0	2013
	0	2014

z. 0% to 0.6% Missing values found.
aa. Replacing missing values with the mode.

SELLER NAME:

bb. No missing values found in the sample files.
cc. Replacing missing values by “Unknown”

SERVICES NAME:

dd. No missing values found in the sample files.
ee. Replacing missing values by “Unknown”

SUPER CONFORMING FLAG:

SUPER CONFORMING FLAG	COUNT	YEAR
Y	80	2008
Y	1236	2009
Y	1364	2010
Y	1967	2011
Y	2189	2012
Y	1718	2013
Y	1995	2014
Y	2223	2015
Y	492	2016

a. Per the data dictionary, all the missing values are Not super conforming, so replaced the missing values by “N”

PERFORMANCE FILE

LOAN SEQUENCE NUMBER:

a. Derived two new columns for origination year and origination quarter

MONTHLY REPORTING PERIOD:

b. Derived two new columns for monthly reporting period year and month

CURRENT ACTUAL UPB:

CURRENT LOAN DELINQUENCY STATUS:

c. No Missing values observed in the sample files.
d. Replacing missing values with “XX” which is also used for unknown.

LOAN AGE:

e. No missing values observed.

REMAINING MONTHS TO LEGAL MATURITY:

f. No missing values found.

REPURCHASE FLAG:

g. This field is only populated at loan termination. For all others the value is not applicable.
h. Replacing nulls with NA.

MODIFICATION FLAG:

i. Replacing nulls with “NO” (Not modified)

ZERO BALANCE CODE:

j. Replacing nulls and spaces with “NA” as it is not applicable if the balance is not reduced to zero.

ZERO BALANCE EFFECTIVE DATE:

k. Replacing missing values with 999999, which will denote not applicable.
l. Deriving 2 new columns for zero balance effective year and month.

CURRENT INTEREST RATE:

m. Replacing empty values with 0.

DUE DATE OF LAST PAID INSTALLMENT:

n. Replacing missing values with 999999.
o. Deriving 2 new columns for due year and month of last paid installment.

REPLACING MISSING VALUES WITH 0 FOR THE FOLLOWING COLUMNS

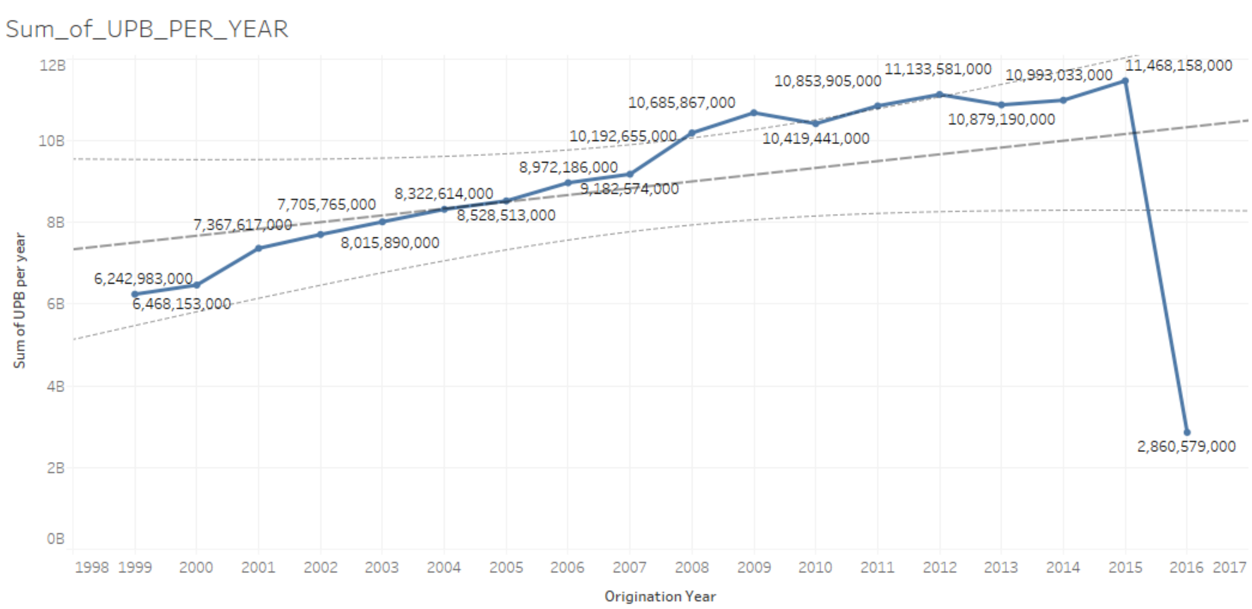
p. MI RECOVERIES
q. NET SALES PROCEEDS
r. NON MI RECOVERIES
s. EXPENSES
t. LEGAL COSTS
u. MAINTENANCE AND PRESERVATION COSTS:
v. TAXES AND INSURANCE:
w. MISCELLENEOUS EXPENSES:
x. ACTUAL LOSS CALCULATION:
y. MODIFICATION COST
z. CURRENT DEFERRED UPB

EXPLORATORY DATA ANALYSIS

SUMMARIES OF SAMPLE ORIGINAL FILES- 1999

SUM OF ORIGINAL UPB PER YEAR (1999)

ORIGINATION YEAR Sum_of_UPB_per_year
0 99 6242983000



The trend of sum of Sum of UPB per year for Origination Year.

OBSERVATION

We see an inclination for the sum of UPB each year, which is natural considering the inflation and demand each year. We also observe a decline during 2009, which would be the effect of the financial crisis of ‘08 and ‘09.

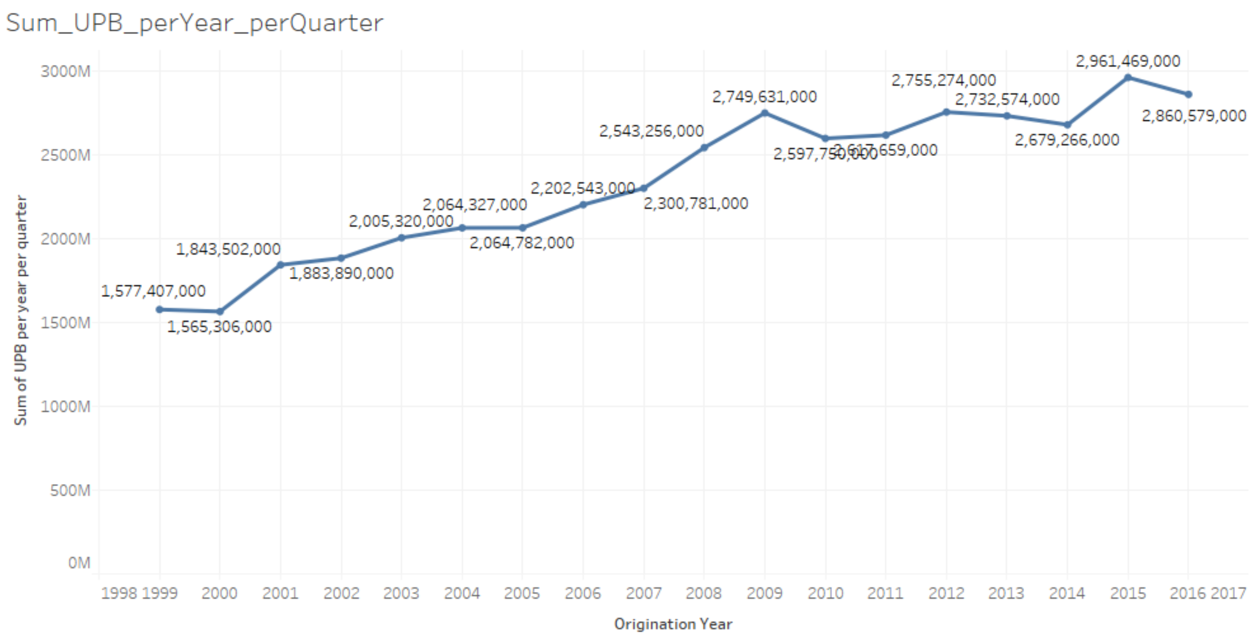
SUM OF ORIGINAL UPB PER QUARTER

	ORIGINATION QUARTER	Sum of UPB per Quarter
0	1	1577407000
1	2	1584374000
2	3	1541836000
3	4	1539366000

SUM OF ORIGINAL UPB PER YEAR PER QUARTER

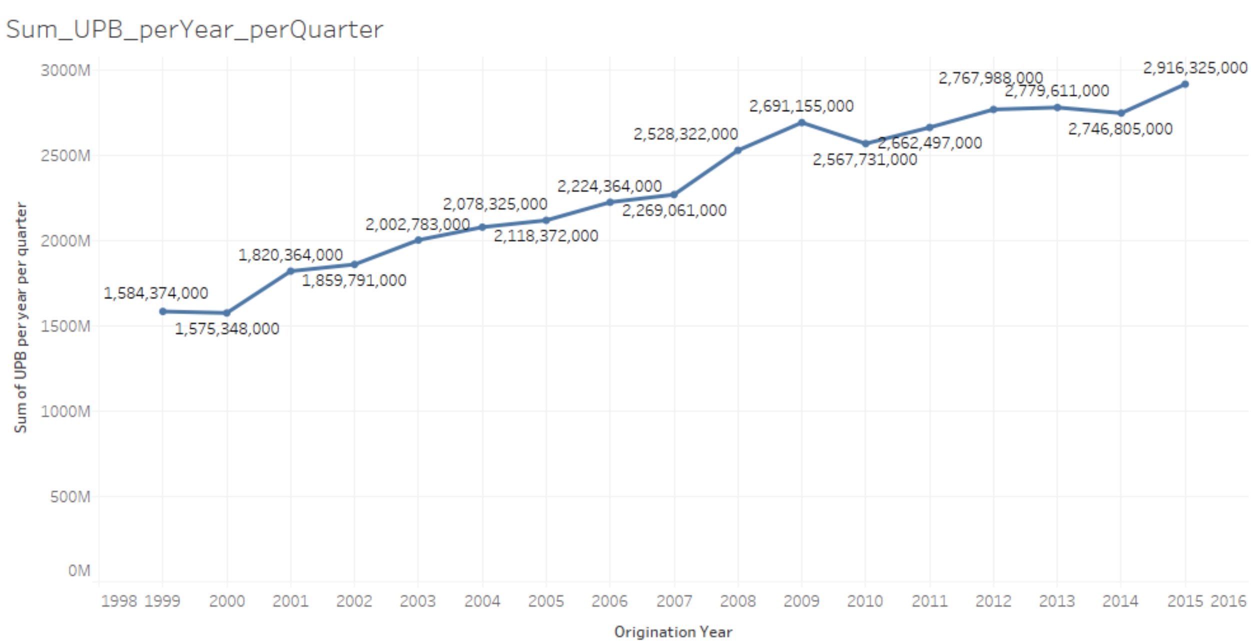
	ORIGINATION YEAR	ORIGINATION QUARTER	Sum of UPB /year /quarter
0	99	1	1577407000
1	99	2	1584374000
2	99	3	1541836000
3	99	4	1539366000

QUARTER 1



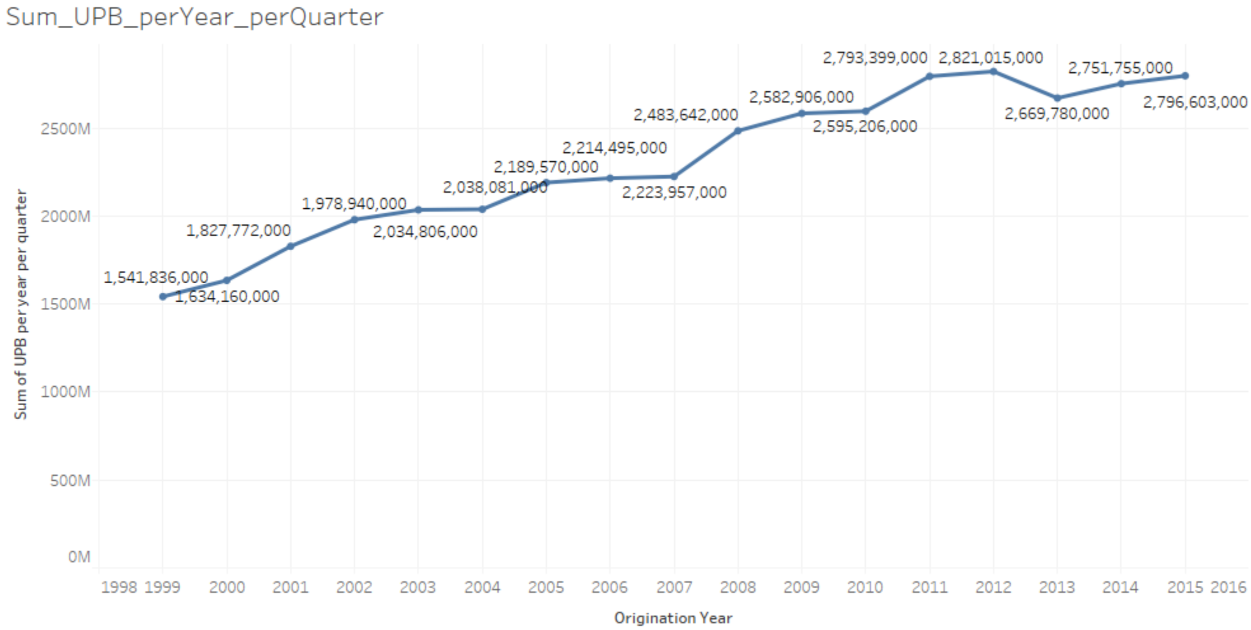
The trend of sum of Sum of UPB per year per quarter for Origination Year. The data is filtered on Origination Quarter, which ranges from 1 to 1.

QUARTER 2



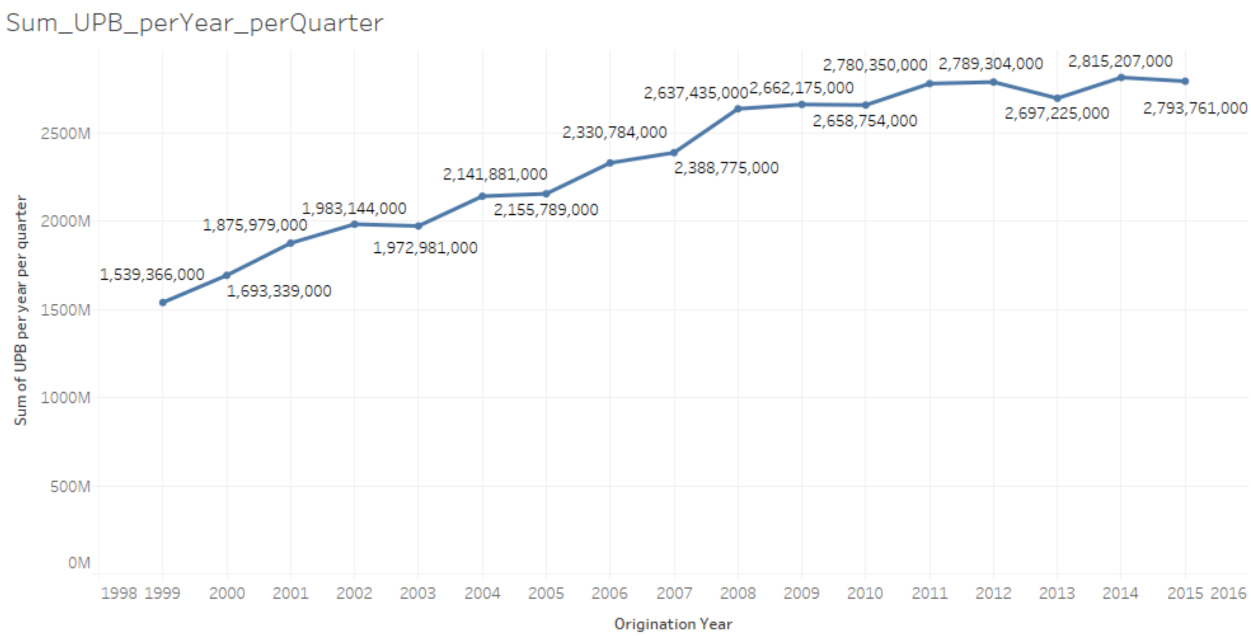
The trend of sum of Sum of UPB per year per quarter for Origination Year. The data is filtered on Origination Quarter, which ranges from 2 to 2.

QUARTER 3



The trend of sum of Sum of UPB per year per quarter for Origination Year. The data is filtered on Origination Quarter, which ranges from 3 to 3.

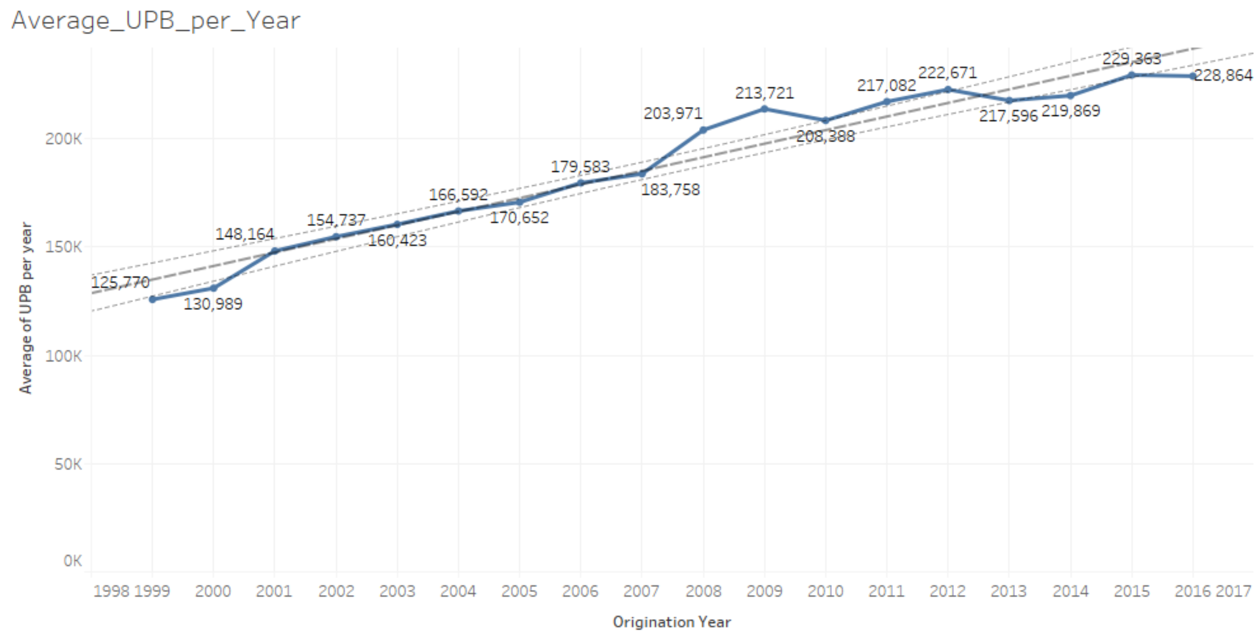
QUARTER 4



The trend of sum of Sum of UPB per year per quarter for Origination Year. The data is filtered on Origination Quarter, which ranges from 4 to 4.

AVERAGE OF ORIGINAL UPB PER YEAR

	ORIGINATION YEAR	Average of UPB per year
0	99	125770



The trend of sum of Average of UPB per year for Origination Year.

Similar observation as the sum of UPB. Inclination in initial years and a declination in 2009

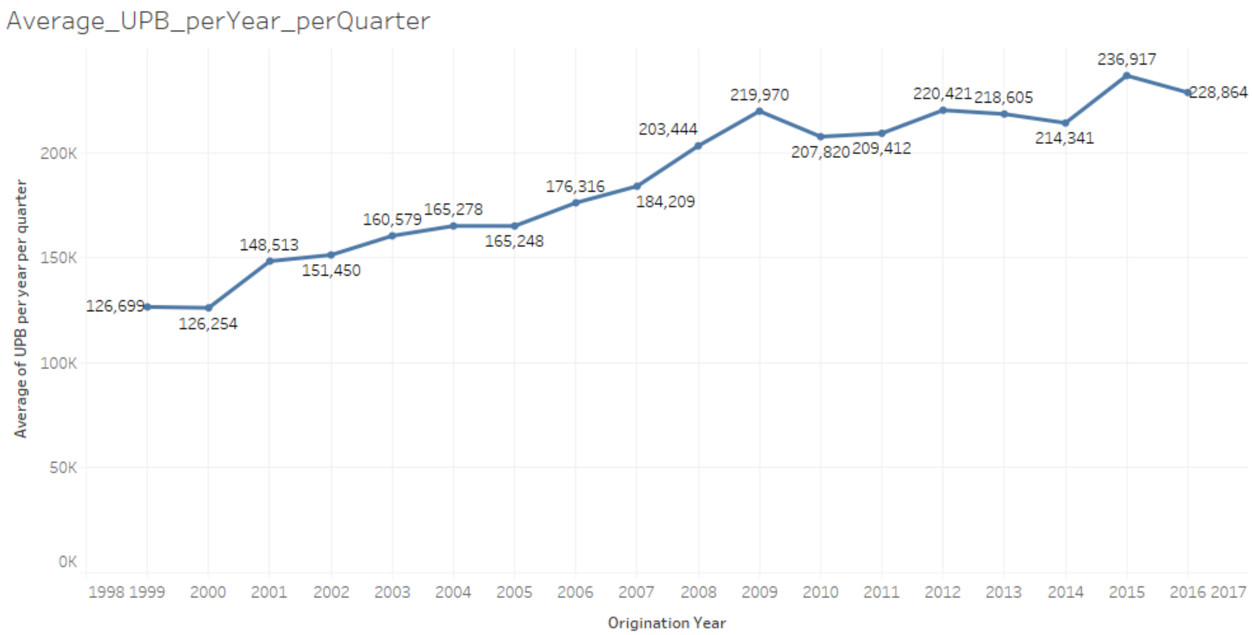
AVERAGE OF ORIGINAL UPB PER YEAR PER QUARTER

	ORIGINATION QUARTER	Average of UPB per quarter
0	1	126699
1	2	127607
2	3	124492
3	4	124272

AVERAGE OF ORIGINAL UPB PER YEAR PER QUARTER

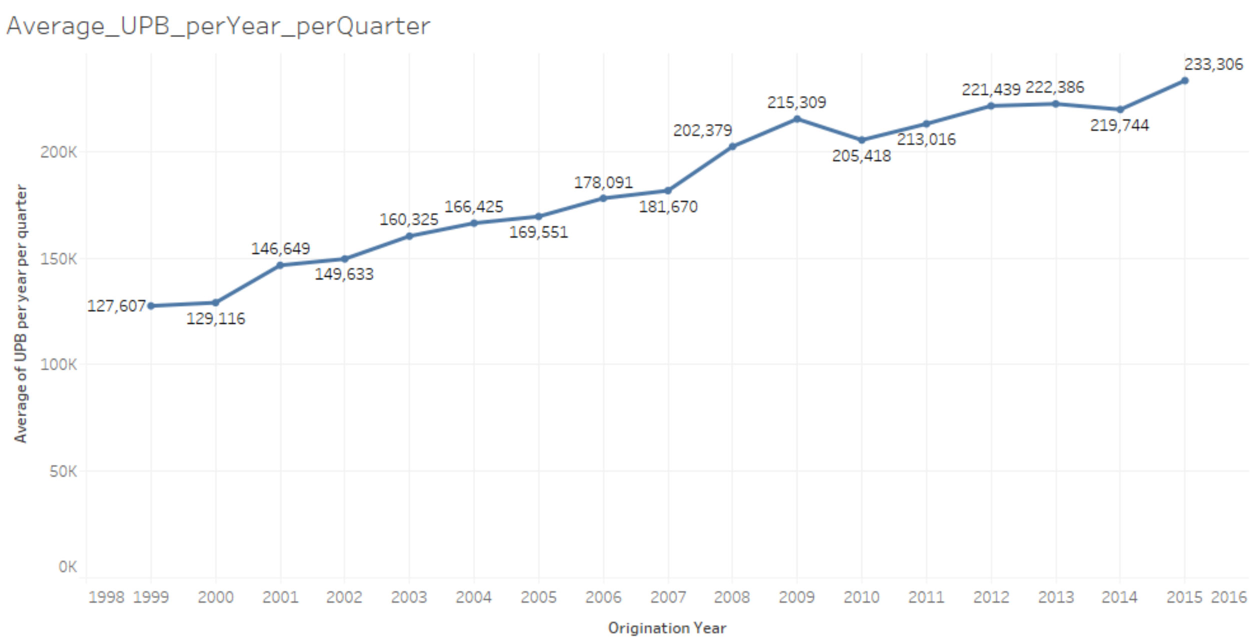
	ORIGINATION YEAR	ORIGINATION QUARTER	Average of UPB per year per quarter
0	99	1	126699
1	99	2	127607
2	99	3	124492
3	99	4	124272

QUARTER 1



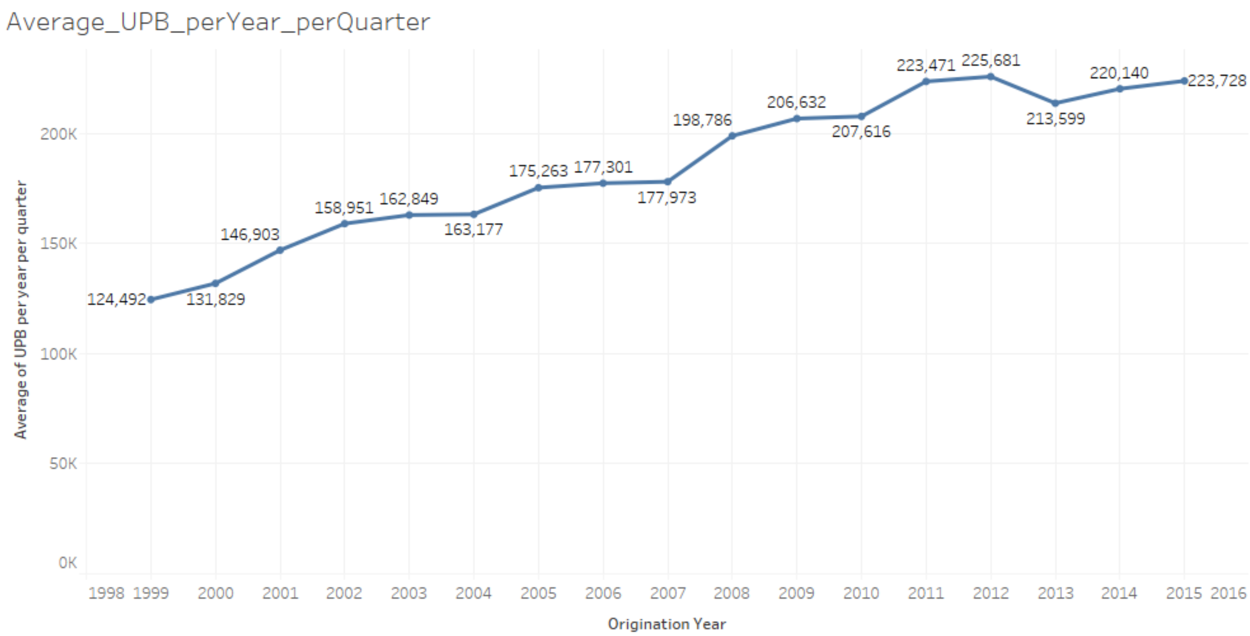
The trend of sum of Average of UPB per year per quarter for Origination Year. The data is filtered on Origination Quarter, which ranges from 1 to 1.

QUARTER 2



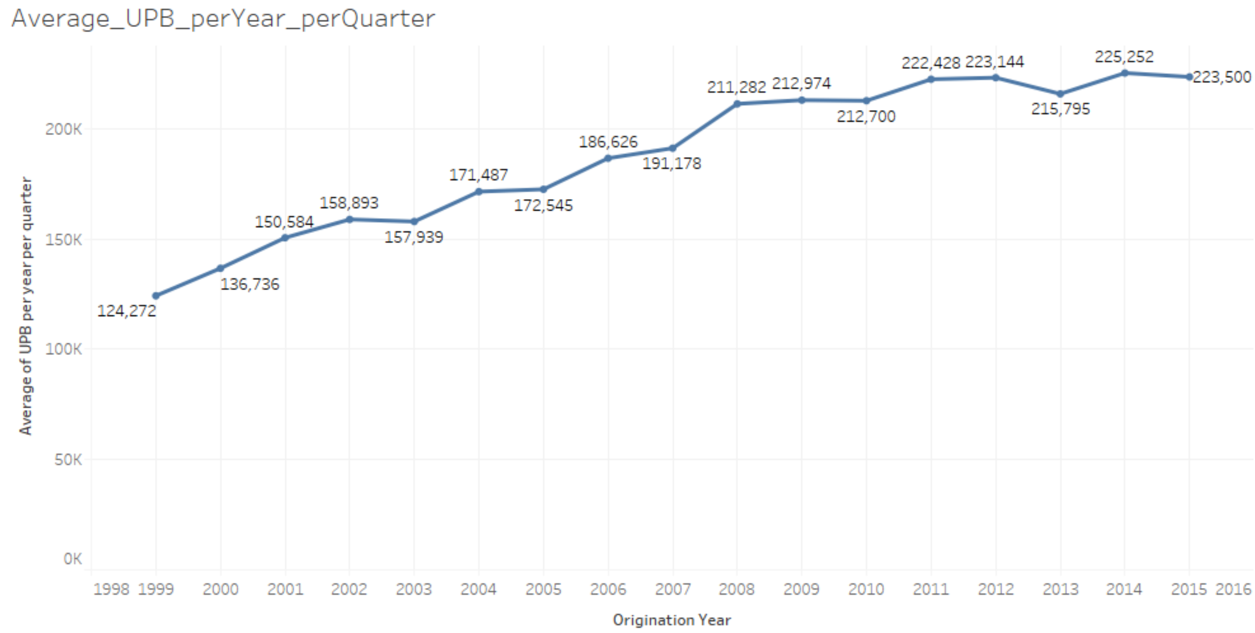
The trend of sum of Average of UPB per year per quarter for Origination Year. The data is filtered on Origination Quarter, which ranges from 2 to 2.

QUARTER 3



The trend of sum of Average of UPB per year per quarter for Origination Year. The data is filtered on Origination Quarter, which ranges from 3 to 3.

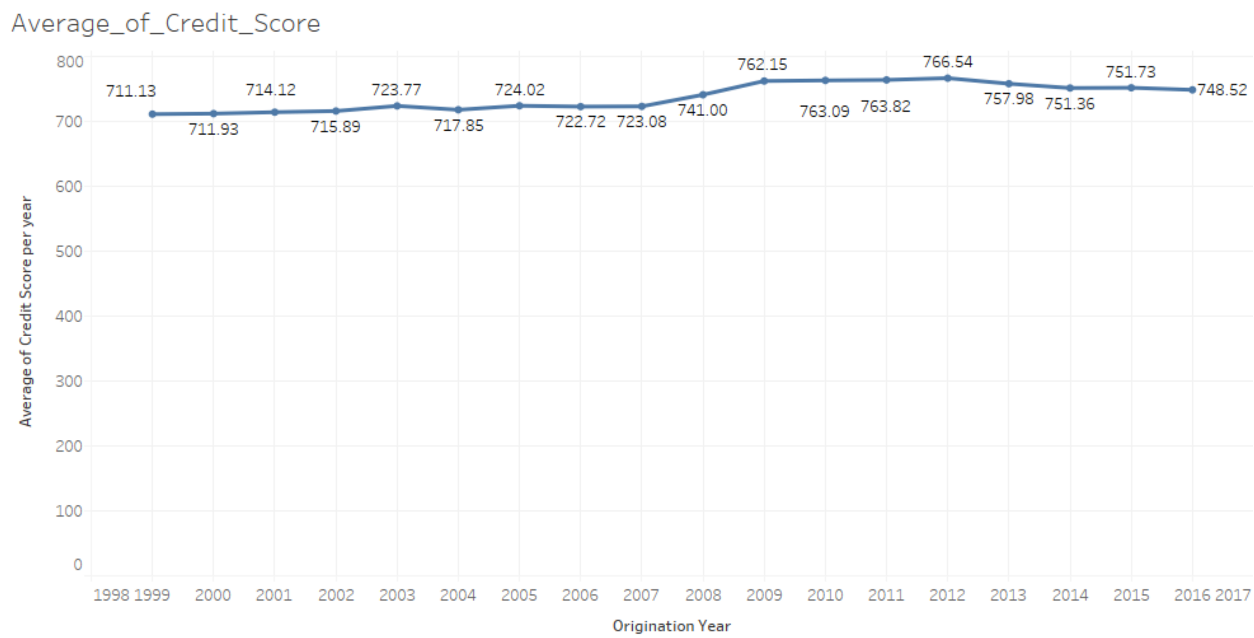
QUARTER 4



The trend of sum of Average of UPB per year per quarter for Origination Year. The data is filtered on Origination Quarter, which ranges from 4 to 4.

AVERAGE OF CREDIT SCORE PER YEAR

ORIGINATION	YEAR	Average of Credit Score per year
0	99	711.133748



The trend of sum of Average of Credit Score per year for Origination Year.

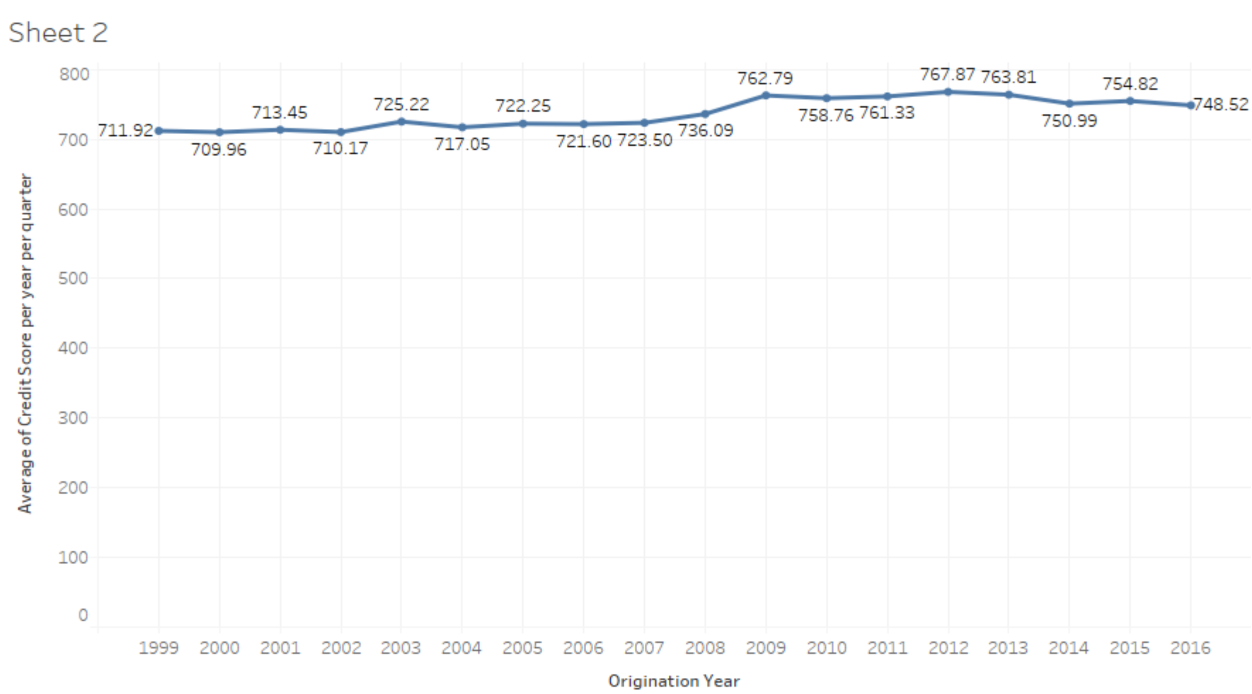
OBSERVATION

Initially we observe that the loan is given to people with avg. credit score of 711-741, but after fi nancial crisis of '08 and '09 we observe a significant increase in the credit score, which means th at the banks were not willing to take high risk

AVERAGE OF CREDIT SCORE PER QUARTER

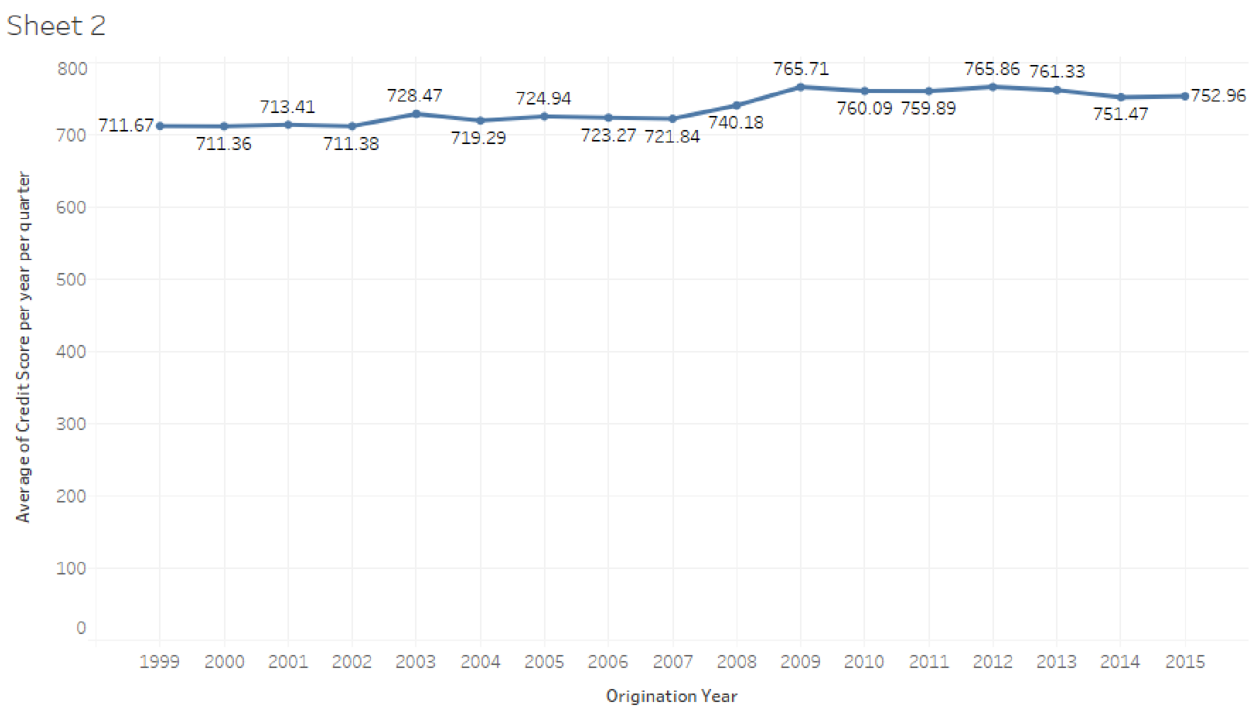
	ORIGINATION QUARTER	Average_Credit_Score_per_quarter
0	1	711.919036
1	2	711.671472
2	3	711.183044
3	4	709.756196

QUARTER 1



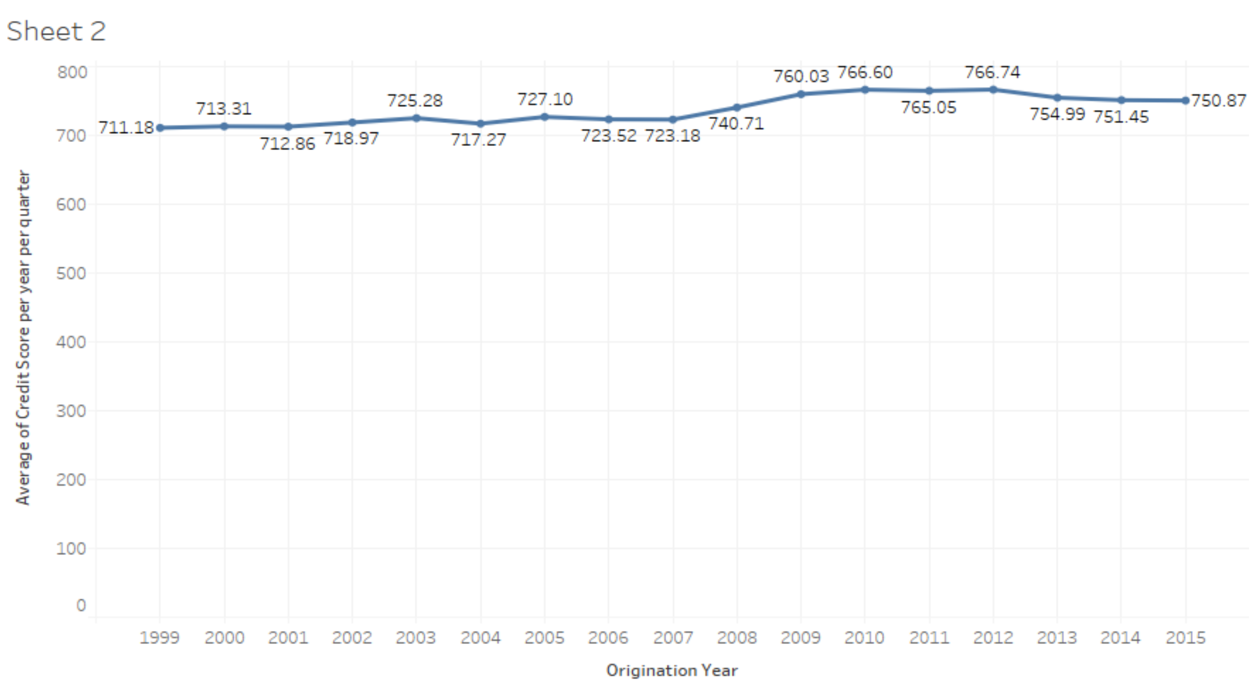
The trend of sum of Average of Credit Score per year per quarter for Origination Year. The data is filtered on Origination Quarter, which ranges from 1 to 1.

QUARTER 2



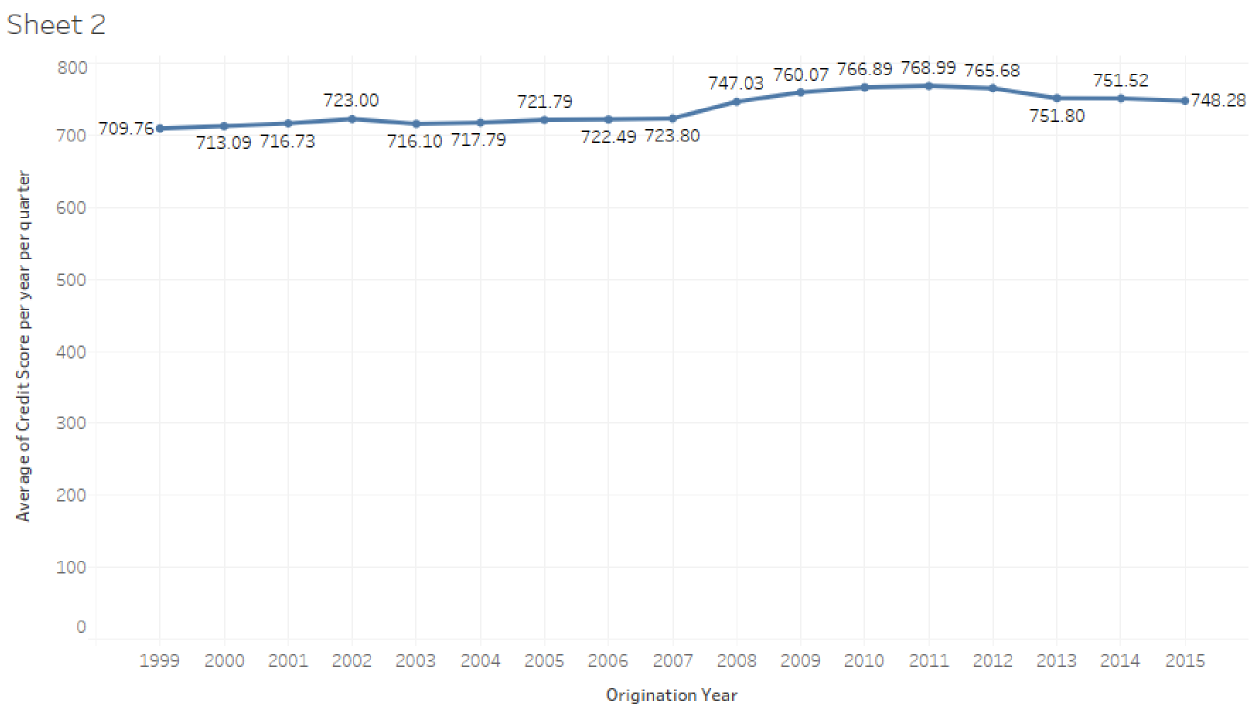
The trend of sum of Average of Credit Score per year per quarter for Origination Year. The data is filtered on Origination Quarter, which ranges from 2 to 2 and keeps Null values.

QUARTER 3



The trend of sum of Average of Credit Score per year per quarter for Origination Year. The data is filtered on Origination Quarter, which ranges from 3 to 3 and keeps Null values.

QUARTER 4



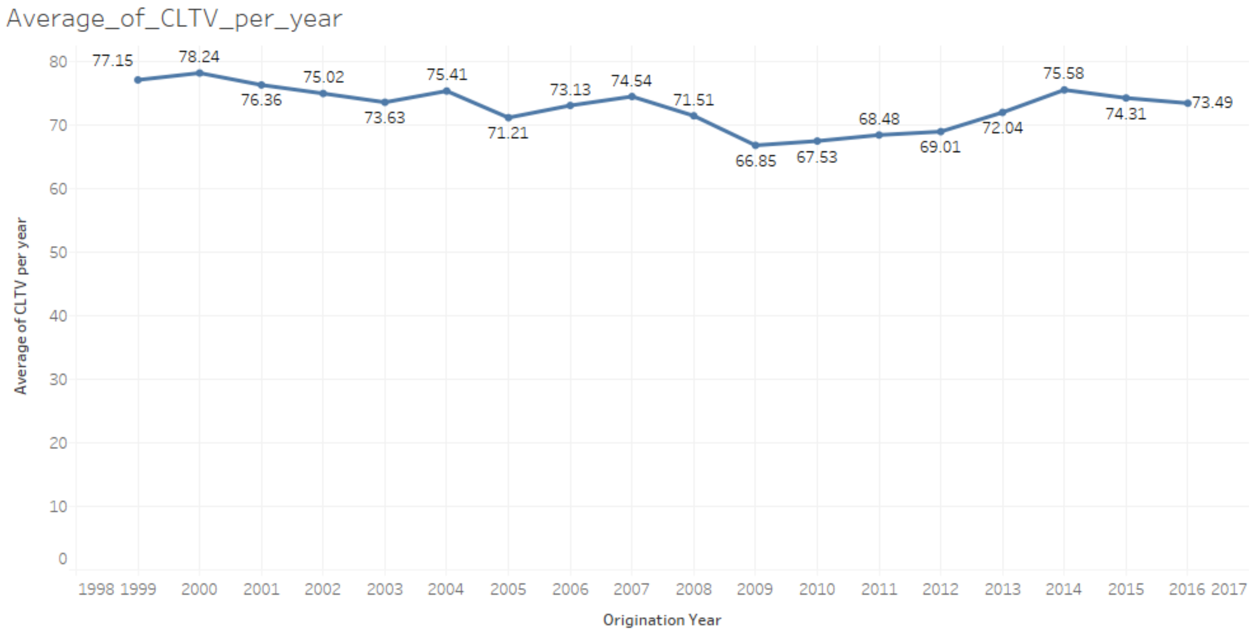
The trend of sum of Average of Credit Score per year per quarter for Origination Year. The data is filtered on Origination Quarter, which ranges from 4 to 4 and keeps Null values.

AVERAGE OF CREDIT SCORE PER YEAR PER QUARTER

	ORIGINATION_YEAR	ORIGINATION_QUARTER	Average_Credit_Score_year_quarter
0	99	1	711.919036
1	99	2	711.671472
2	99	3	711.183044
3	99	4	709.756196

AVERAGE OF CLTV PER YEAR

	ORIGINATION_YEAR	Average of CLTV per year
0	99	77.153753



The trend of sum of Average of CLTV per year for Origination Year.

AVERAGE OF CLTV PER QUARTER

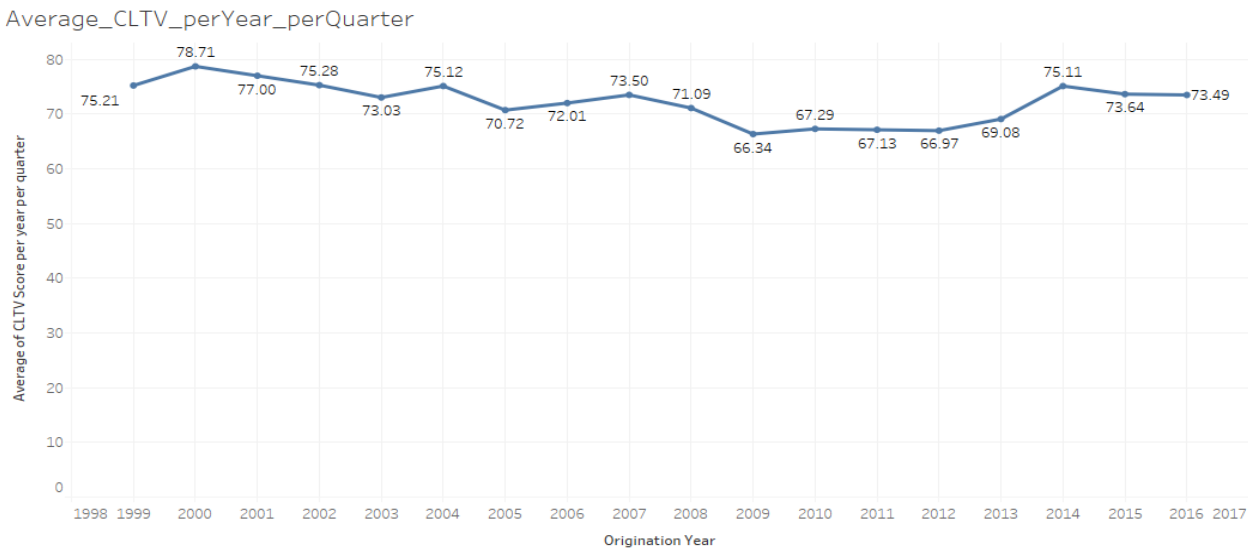
	ORIGINATION_QUARTER	Average_CLTV_quarter
0	1	75.212610
1	2	77.176627
2	3	77.813726
3	4	78.421975

ORIGINATION_YEAR	ORIGINATION_QUARTER
99	1
99	2
99	3
99	4

AVERAGE OF CLTV PER YEAR PER QUARTER

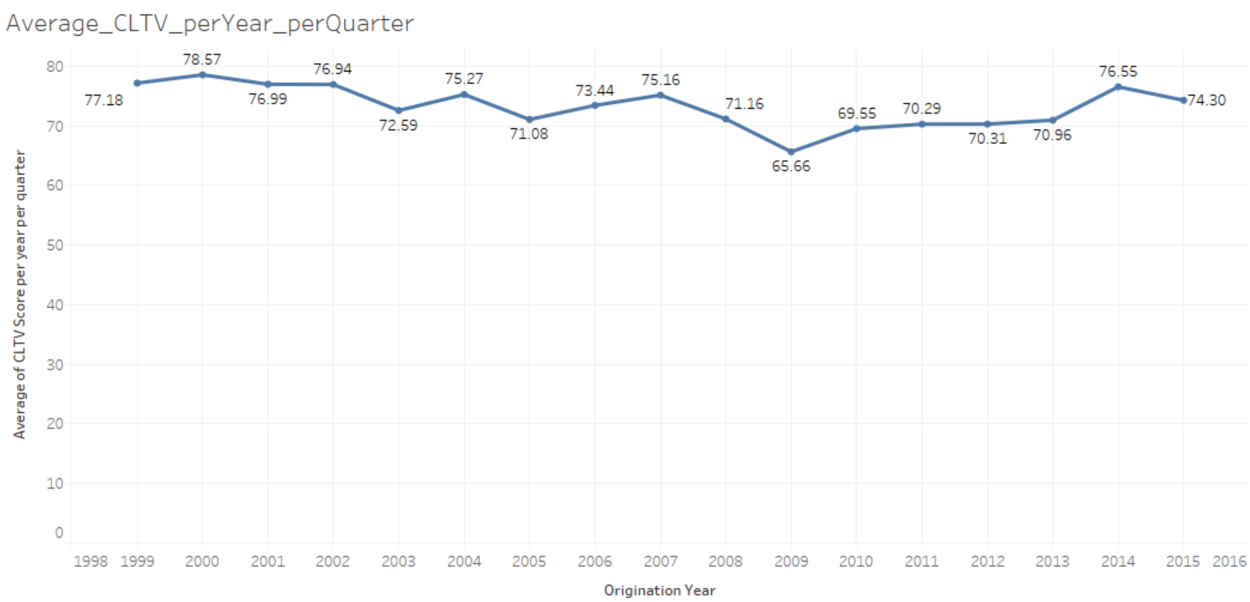
	Average of CLTV Score per year per quarter
0	75.212610
1	77.176627
2	77.813726
3	78.421975

QUARTER 1



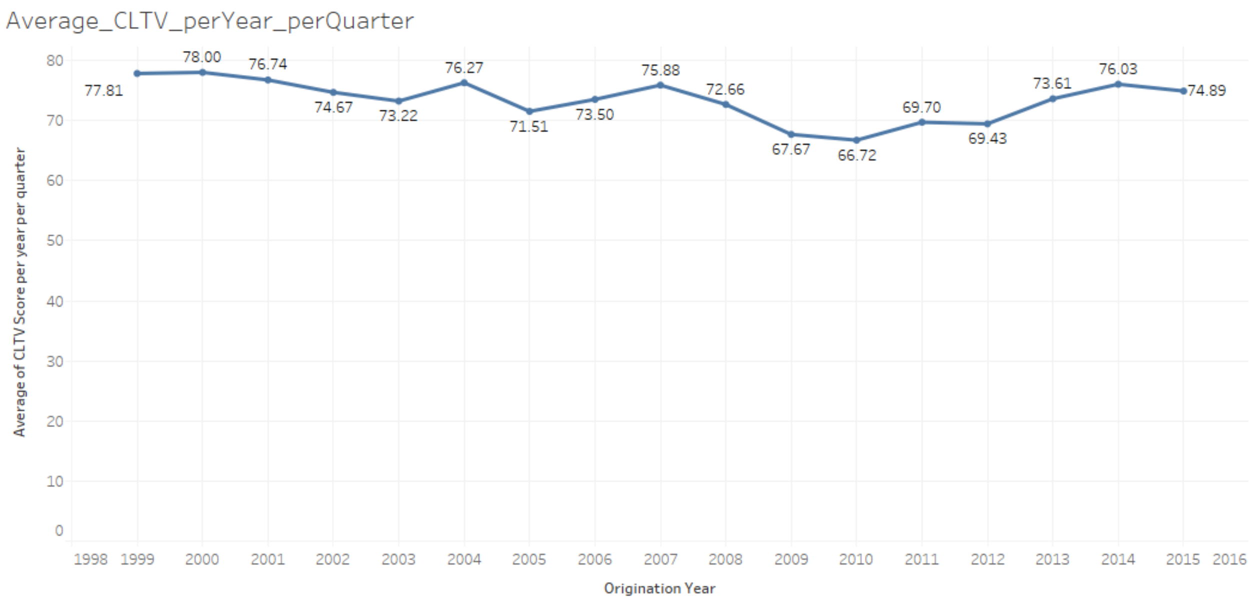
The trend of sum of Average of CLTV Score per year per quarter for Origination Year. The data is filtered on Origination Quarter, which ranges from 1 to 1.

QUARTER 2



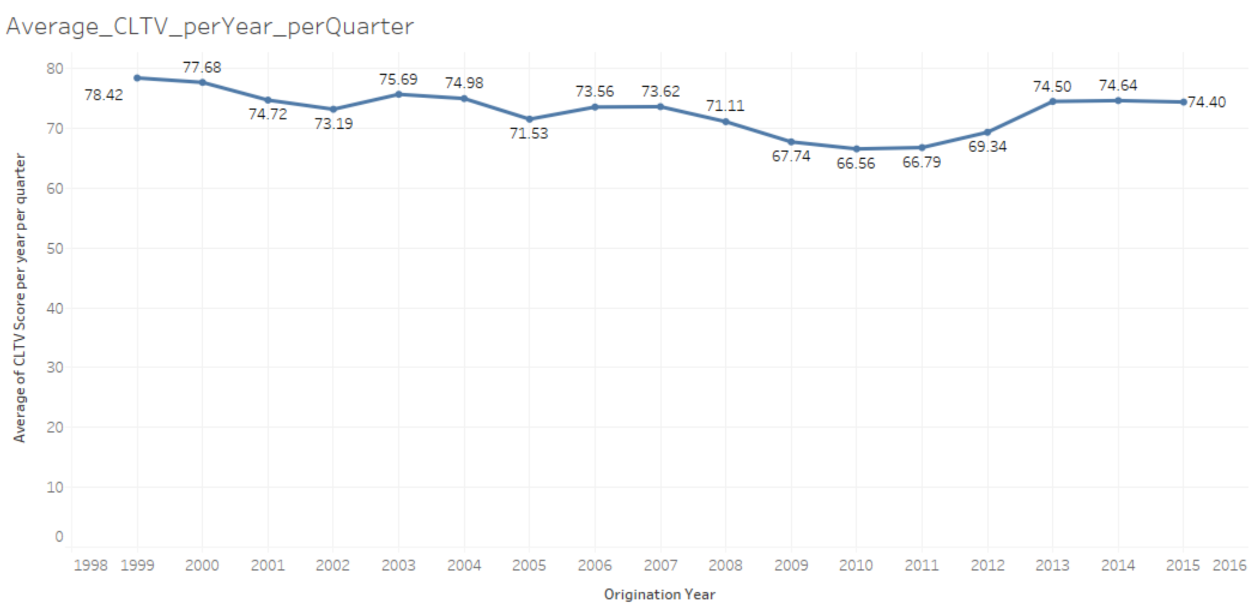
The trend of sum of Average of CLTV Score per year per quarter for Origination Year. The data is filtered on Origination Quarter, which ranges from 2 to 2.

QUARTER 3



The trend of sum of Average of CLTV Score per year per quarter for Origination Year. The data is filtered on Origination Quarter, which ranges from 3 to 3.

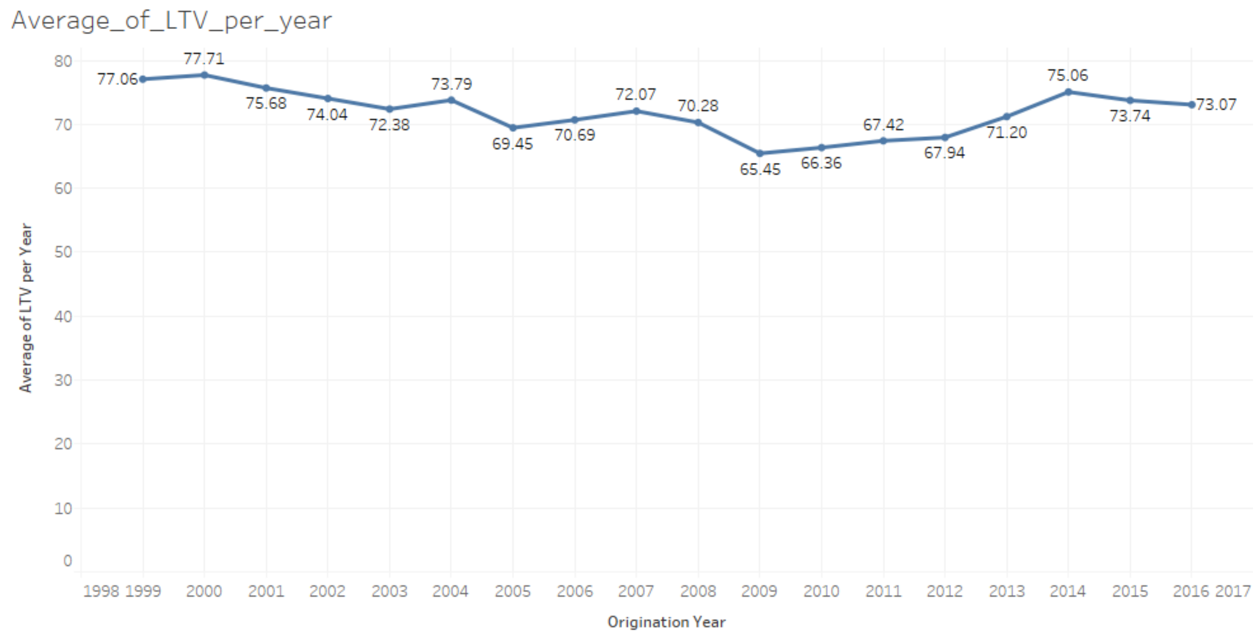
QUARTER 4



The trend of sum of Average of CLTV Score per year per quarter for Origination Year. The data is filtered on Origination Quarter, which ranges from 4 to 4.

AVERAGE OF LTV PER YEAR

	ORIGINATION_YEAR	Average_LTV_per_Year
0	99	77.056509



The trend of sum of Average of LTV per Year for Origination Year.

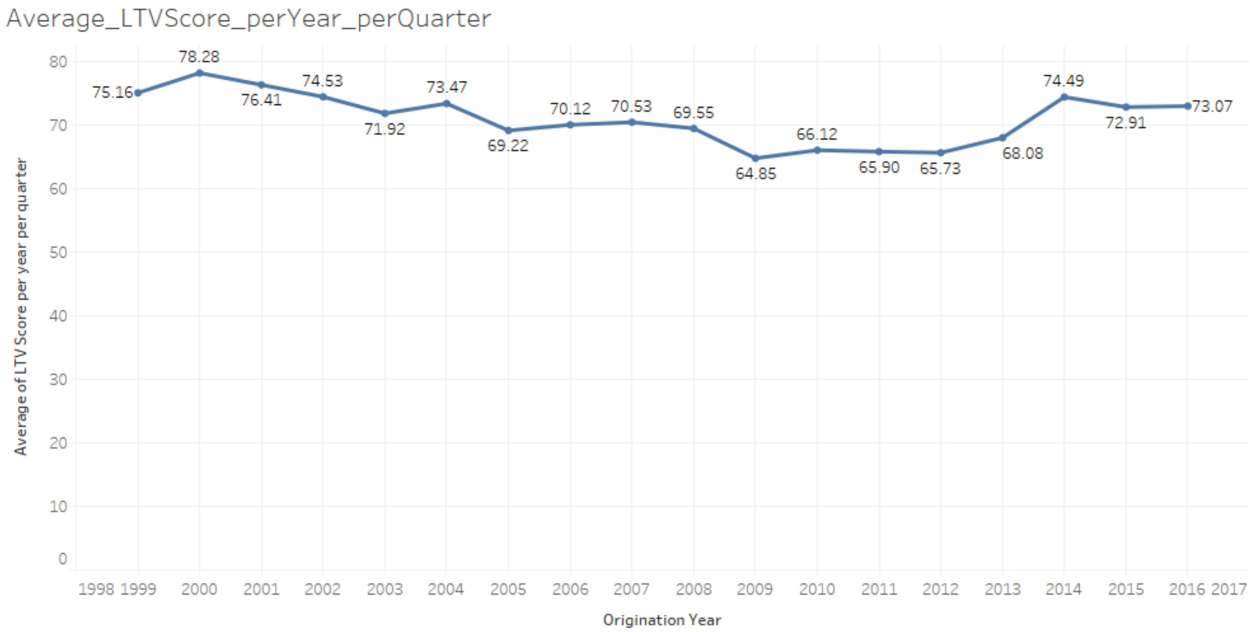
AVERAGE OF LTV PER QUARTER

	ORIGINATION_QUARTER	Average_LTV_Quarter
0	1	75.155020
1	2	77.108811
2	3	77.715382
3	4	78.256479

AVERAGE OF LTV PER YEAR PER QUARTER

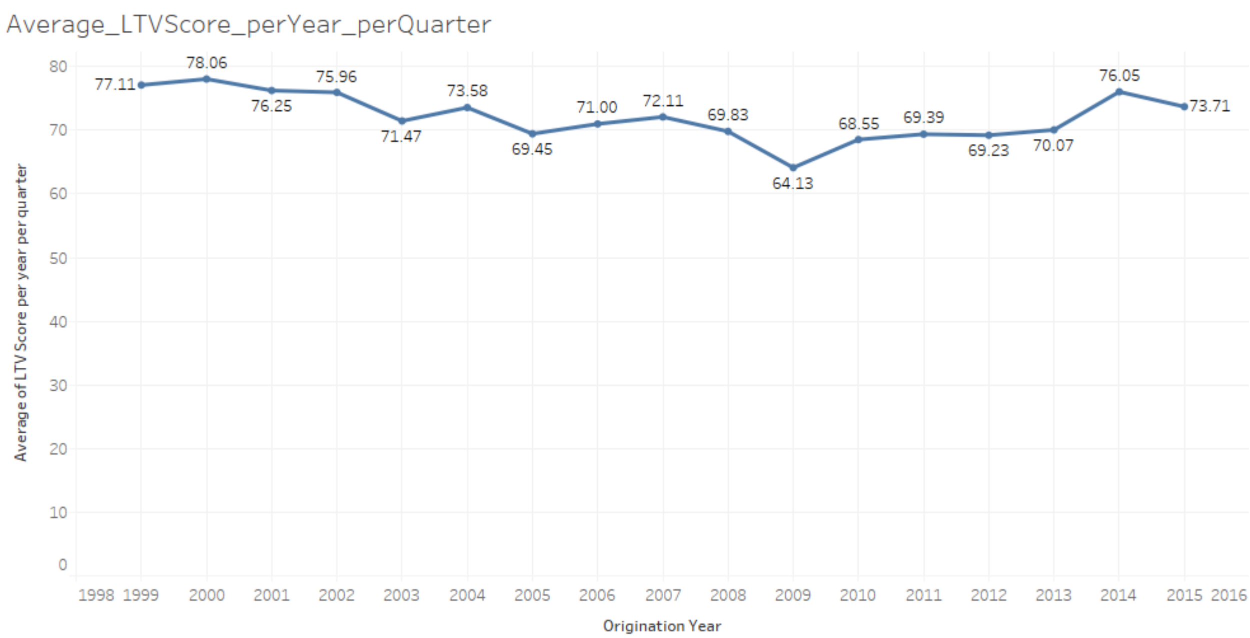
	ORIGINATION_YEAR	ORIGINATION_QUARTER	Average_LTV_Score_year_quarter
0	99	1	75.155020
1	99	2	77.108811
2	99	3	77.715382
3	99	4	78.256479

QUARTER 1



The trend of sum of Average of LTV Score per year per quarter for Origination Year. The data is filtered on Origination Quarter, which ranges from 1 to 1.

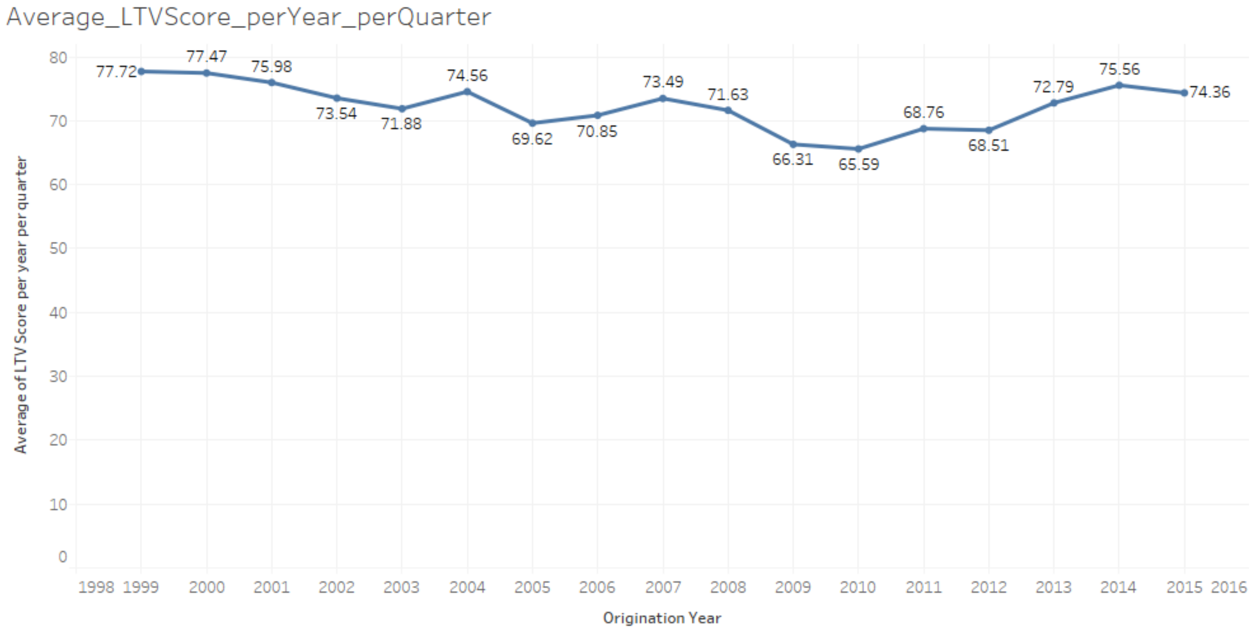
QUARTER 2



The trend of sum of Average of LTV Score per year per quarter for Origination Year. The data is filtered on Origination Quarter, which ranges from 2 to 2.

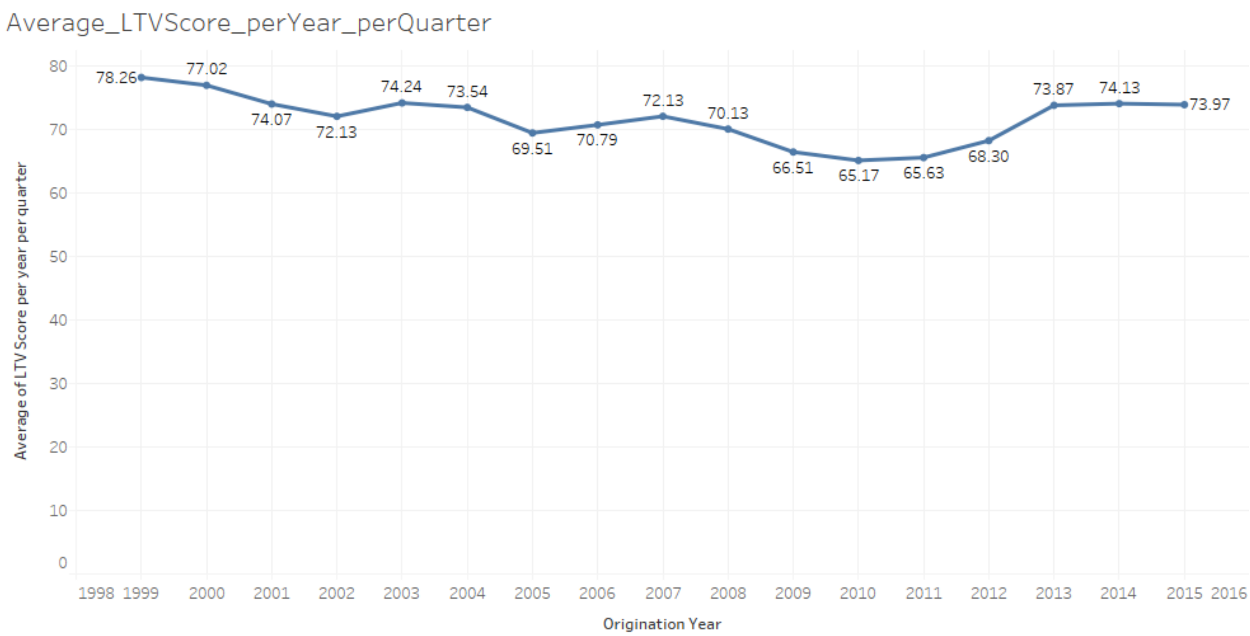
Advanced Data Science/Architecture Mid-Term Project Team 7

QUARTER 3



The trend of sum of Average of LTV Score per year per quarter for Origination Year. The data is filtered on Origination Quarter, which ranges from 3 to 3.

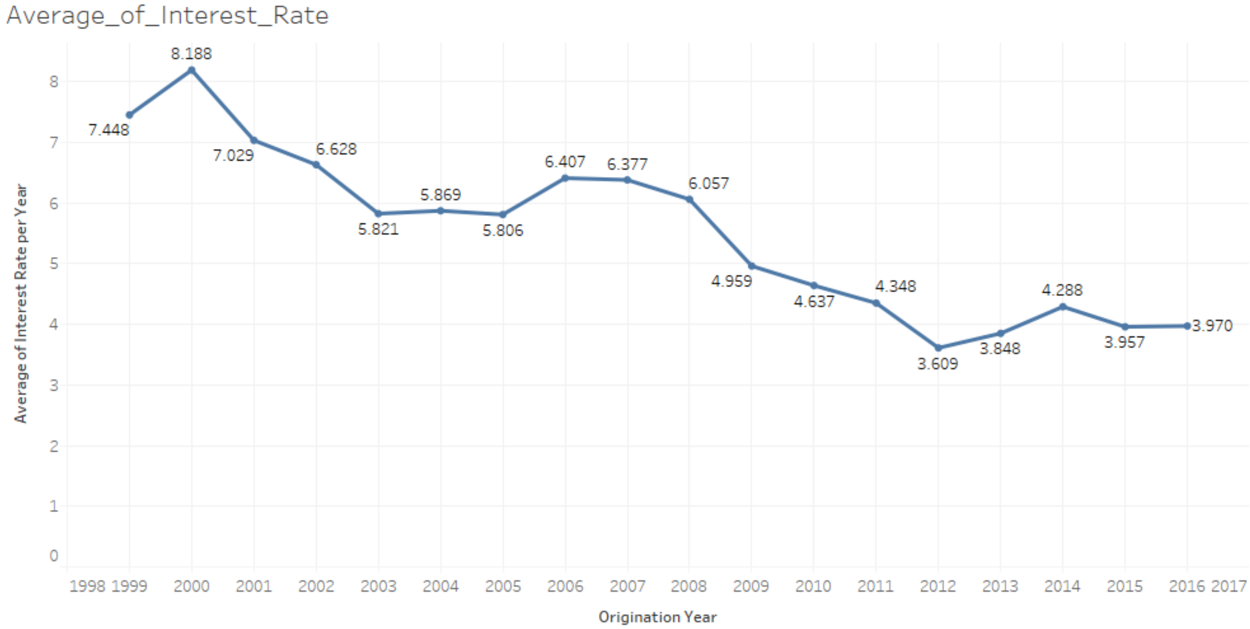
QUARTER 4



The trend of sum of Average of LTV Score per year per quarter for Origination Year. The data is filtered on Origination Quarter, which ranges from 4 to 4.

AVERAGE OF INTEREST RATE PER YEAR

	ORIGINATION_YEAR	Average_Interest_Rate_per_Year
0	99	7.447759



The trend of sum of Average of Interest Rate per Year for Origination Year.

OBSERVATION

We see that the interest rate is declining over the years. After looking at the statistics of the financial years, we can say that the interest rate is the higher when the economy is good, and declines when bad. Hence, 2012 has the least and improves during 2013.

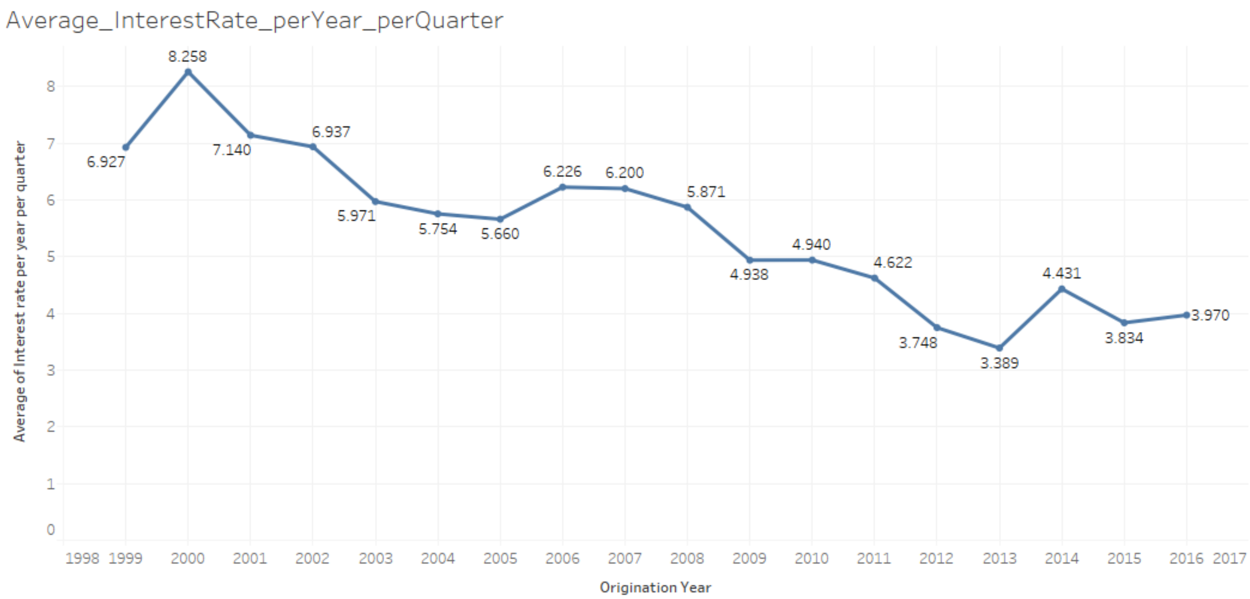
AVERAGE OF INTEREST RATE PER QUARTER

	ORIGINATION_QUARTER	Average_Interest_Rate_per_Quarter
0	1	6.927040
1	2	7.155169
2	3	7.777791
3	4	7.934422

AVERAGE OF INTEREST RATE PER YEAR PER QUARTER

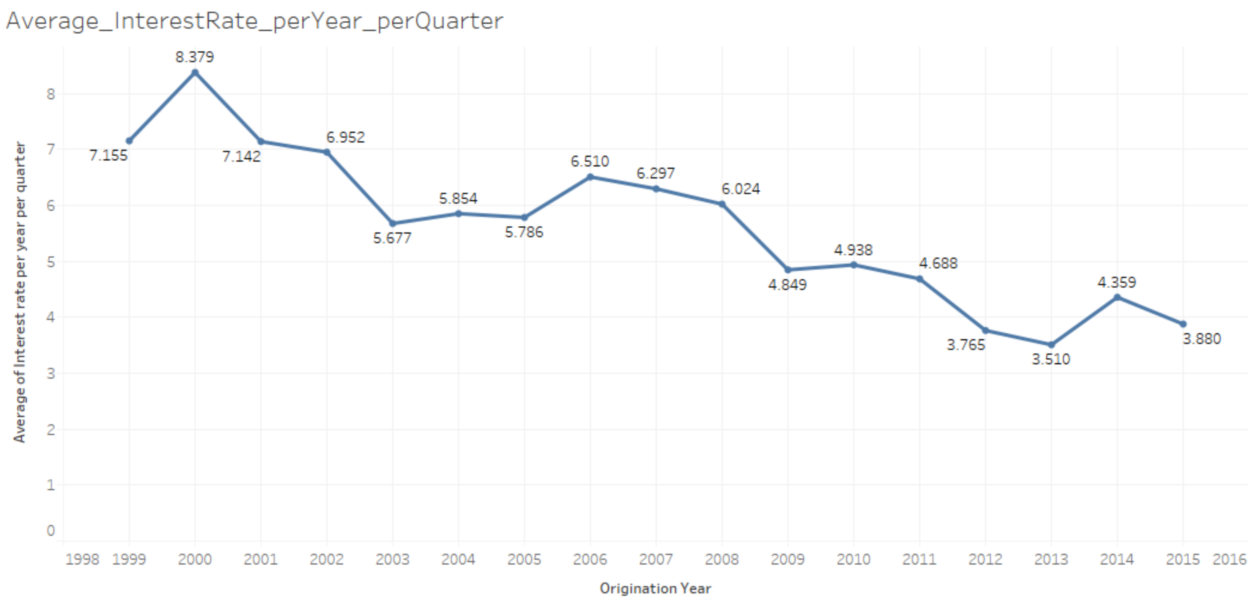
	ORIGINATION_YEAR	ORIGINATION_QUARTER	Average_Interest_rate_year_quarter
0	99	1	6.927040
1	99	2	7.155169
2	99	3	7.777791
3	99	4	7.934422

QUARTER 1



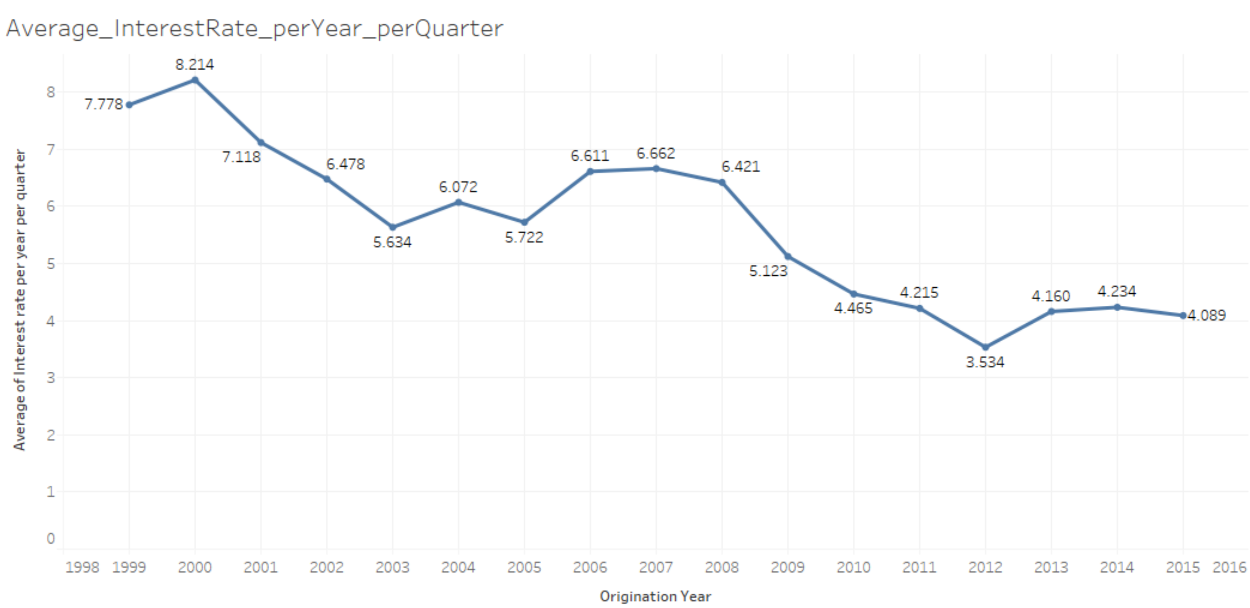
The trend of sum of Average of Interest rate per year per quarter for Origination Year. The data is filtered on Origination Quarter, which ranges from 1 to 1.

QUARTER 2



The trend of sum of Average of Interest rate per year per quarter for Origination Year. The data is filtered on Origination Quarter, which ranges from 2 to 2.

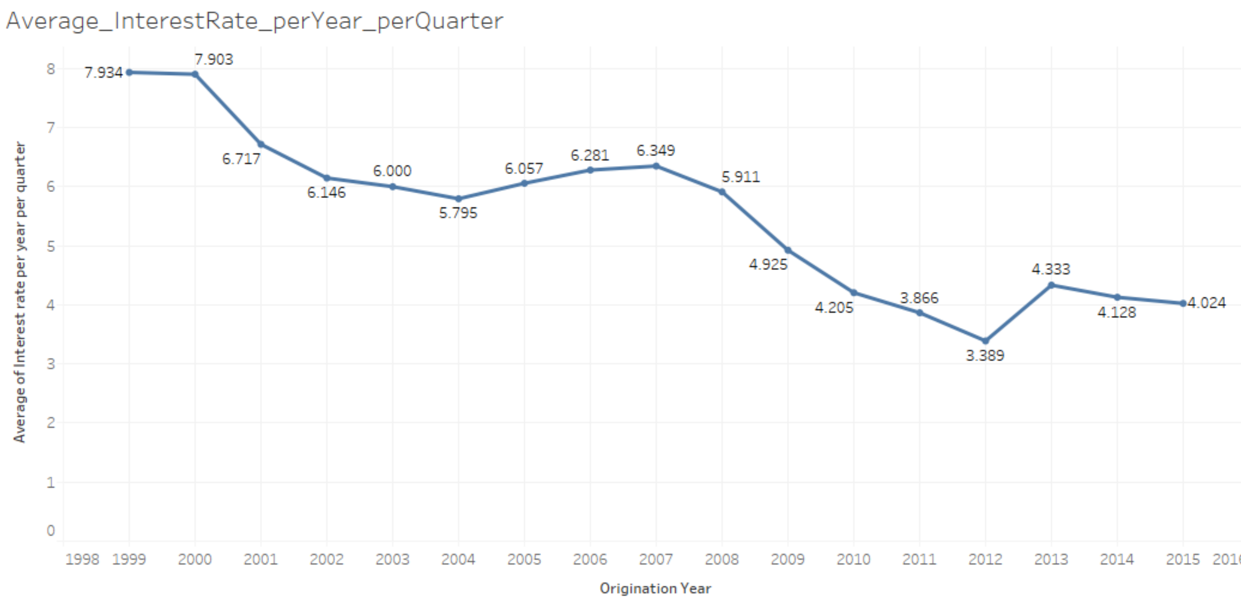
QUARTER 3



The trend of sum of Average of Interest rate per year per quarter for Origination Year. The data is filtered on Origination Quarter, which ranges from 3 to 3.

QUARTER 4

Advanced Data Science/Architecture Mid-Term Project Team 7

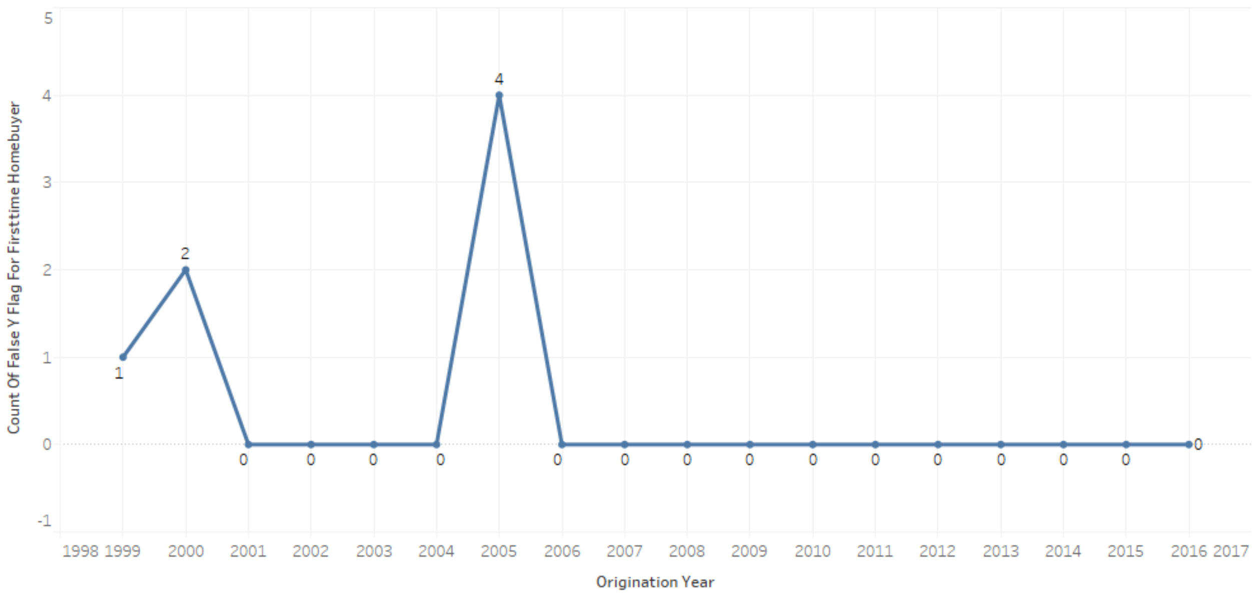


The trend of sum of Average of Interest rate per year per quarter for Origination Year. The data is filtered on Origination Quarter, which ranges from 4 to 4.

COUNT OF LOANS WITH FIRST TIME HOME BUYER EQUAL TO "Y", OCCUPANCY EQUAL TO "I" OR "S" AND LOAN PURPOSE EQUAL TO "C" AND "N"

1

Count_of_FirstTimeBuyers_OccupancyStatusIorS_LoanPurposeCorN



The trend of sum of Count Of False Y Flag For Firsttime Homebuyer for Origination Year.

Trend Lines Model

A linear trend model is computed for sum of Count Of False Y Flag For Firsttime Homebuyer given Origination Year.

Model formula:	(Origination Year + intercept)
Number of modeled observations:	18
Number of filtered observations:	0
Model degrees of freedom:	2
Residual degrees of freedom (DF):	16
SSE (sum squared error):	15.9615
MSE (mean squared error):	0.997592
R-Squared:	0.126728
Standard error:	0.998795
p-value (significance):	0.147084

Individual trend lines:

Panes		Line		Coefficients				
Row	Column	p-value	DF	Term	Value	StdErr	t-value	p-value
Count Of False Y Flag For Firsttime Homebuyer	Origination Year	0.147084	16	Origination Year	-0.0691434	0.0453764	-1.52378	0.147084
				intercept	139.194	91.0933	1.52804	0.146029

COUNT OF LOANS WITH MSA FLAG EQUAL TO "YES"

42069

COUNT OF LOANS WITH MSA FLAG EQUAL TO "NO"

7569

AVERAGE ORIGINAL UPB WHERE MSA FLAG EQUAL TO "YES"

129213.17359575935

AVERAGE ORIGINAL UPB WHERE MSA FLAG EQUAL TO "YES"

106634.1656757828

AVERAGE CREDIT SCORE WHERE MSA FLAG EQUAL TO "YES"

711.3050464712734

AVERAGE CREDIT SCORE WHERE MSA FLAG EQUAL TO "NO"

710.1816620425419

AVERAGE INTEREST RATE WHERE MSA FLAG EQUAL TO "YES"

7.434357864460766

AVERAGE INTEREST RATE WHERE MSA FLAG EQUAL TO "NO"

7.522243361078074

COUNT OF LOANS WHERE PPM FLAG EQUAL TO "Y"

262

SUMMARIES OF SAMPLE PERFORMANCE FILES- 2016

DISTINCT COUNT OF LOAN

	LOAN SEQUENCE NUMBER	Distinct count of loan
0	F116Q1000017	8
1	F116Q1000025	5
2	F116Q1000034	8
3	F116Q1000051	8
4	F116Q1000076	8
5	F116Q1000093	8
6	F116Q1000110	7
7	F116Q1000186	8
8	F116Q1000203	8
9	F116Q1000212	8
10	F116Q1000220	7
11	F116Q1000229	8
12	F116Q1000245	8
13	F116Q1000262	8
14	F116Q1000271	8
15	F116Q1000288	8
16	F116Q1000296	7
17	F116Q1000330	8
18	F116Q1000381	8
19	F116Q1000389	8
20	F116Q1000423	8
21	F116Q1000440	5
22	F116Q1000508	8
23	F116Q1000525	8
24	F116Q1000559	8
25	F116Q1000584	8

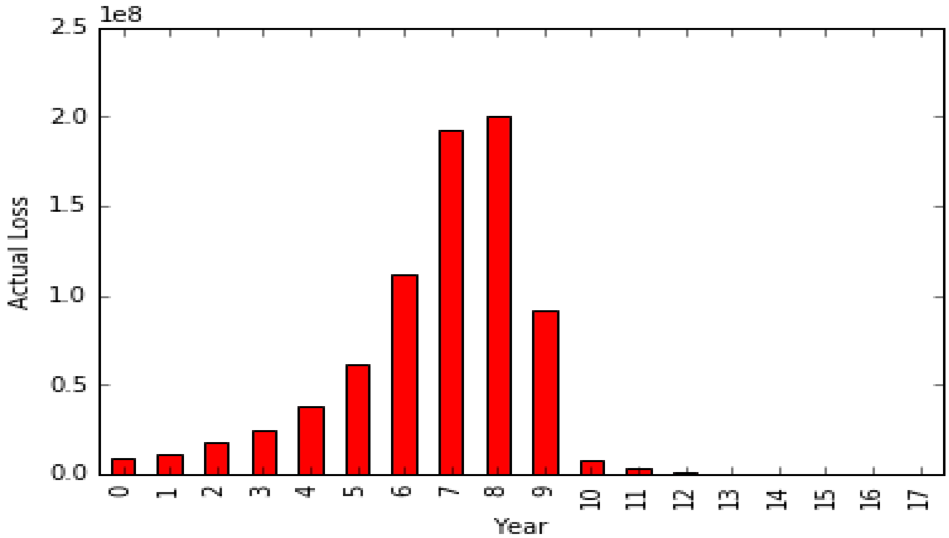
26	F116Q1000609	8
27	F116Q1000660	8
28	F116Q1000669	8
29	F116Q1000686	8
...	...	...
12444	F116Q1276390	3
12445	F116Q1276415	3
12446	F116Q1276424	3
12447	F116Q1276441	3
12448	F116Q1276449	3
12449	F116Q1276458	3
12450	F116Q1276475	3
12451	F116Q1276492	3
12452	F116Q1276500	3
12453	F116Q1276509	3
12454	F116Q1276517	4
12455	F116Q1276526	4
12456	F116Q1276542	4
12457	F116Q1276551	4
12458	F116Q1276559	4
12459	F116Q1276568	4
12460	F116Q1276585	3
12461	F116Q1276602	3
12462	F116Q1276610	3
12463	F116Q1276619	3
12464	F116Q1276627	3
12465	F116Q1276644	3
12466	F116Q1276652	3
12467	F116Q1276669	3
12468	F116Q1276686	3
12469	F116Q1276695	2
12470	F116Q1276729	2
12471	F116Q1276779	1
12472	F116Q1276813	1
12473	F116Q1276864	1

COUNT OF LOANS WITH CURRENT UPB EQUAL TO 0, AND ZERO BALANCE CODE EQUAL TO 1 OR 6

424

COUNT OF ACTUAL LOSS CALCULATION WITH CURRENT UPB EQUAL TO 0, AND ZERO BALANCE CODE EQUAL TO 9

0



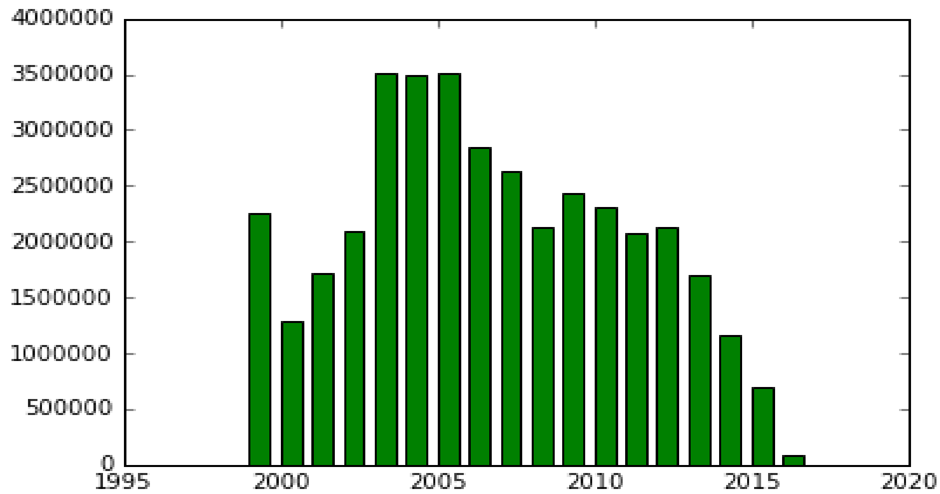
OBSERVATION

Here, we are calculating the actual losses for the loans that have been closed due to non-performance, and sold or auctioned by the lender.

We see that there is a significant increase from 2006 through 2008

COUNT OF LOANS WITH CURRENT UPB NOT EQUAL TO 0

83058



OBSERVATION

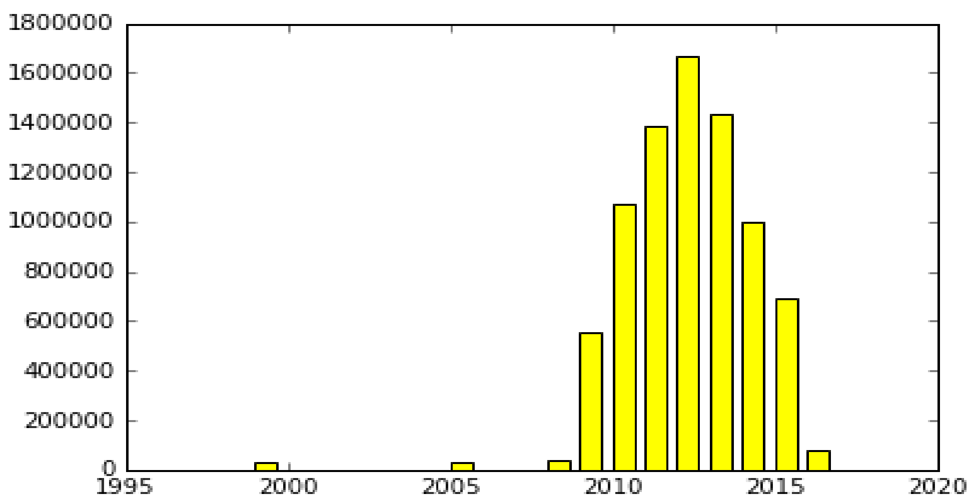
These are the current active loans

COUNT OF LOANS WITH DELINQUENCY STATUS >= 5

0

COUNT OF LOANS WITH CURRENT UPB NOT EQUAL TO 0 AND DELINQUENCY STATUS >= 0

82917



OBSERVATION

With this observation, we can infer that the loans are active and has been delinquent at least once

COUNT OF MODIFICATION FLAG GROUPED BY LOAN SEQUENCE NUMBER

```
  LOAN SEQUENCE NUMBER  \
0          F116Q1000017
1          F116Q1000025
2          F116Q1000034
3          F116Q1000051
4          F116Q1000076
5          F116Q1000093
6          F116Q1000110
7          F116Q1000186
8          F116Q1000203
9          F116Q1000212
10          F116Q1000220
11          F116Q1000229
12          F116Q1000245
13          F116Q1000262
14          F116Q1000271
15          F116Q1000288
16          F116Q1000296
17          F116Q1000330
18          F116Q1000381
19          F116Q1000389
20          F116Q1000423
21          F116Q1000440
22          F116Q1000508
23          F116Q1000525
24          F116Q1000559
25          F116Q1000584
26          F116Q1000609
27          F116Q1000660
28          F116Q1000669
29          F116Q1000686
...               ...
12444        F116Q1276390
12445        F116Q1276415
12446        F116Q1276424
12447        F116Q1276441
12448        F116Q1276449
12449        F116Q1276458
12450        F116Q1276475
12451        F116Q1276492
12452        F116Q1276500
12453        F116Q1276509
12454        F116Q1276517
12455        F116Q1276526
12456        F116Q1276542
12457        F116Q1276551
12458        F116Q1276559
12459        F116Q1276568
```

12460	F116Q1276585
12461	F116Q1276602
12462	F116Q1276610
12463	F116Q1276619
12464	F116Q1276627
12465	F116Q1276644
12466	F116Q1276652
12467	F116Q1276669
12468	F116Q1276686
12469	F116Q1276695
12470	F116Q1276729
12471	F116Q1276779
12472	F116Q1276813
12473	F116Q1276864

Number of Modification flag grouped by Loan Sequence Number

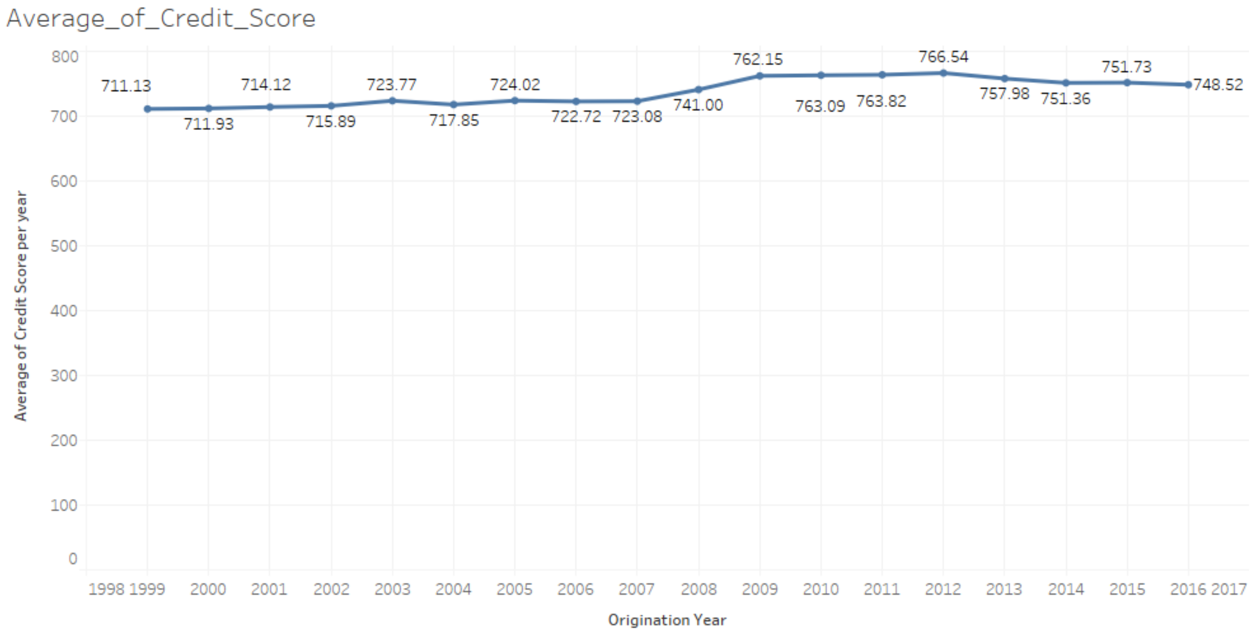
0	8
1	5
2	8
3	8
4	8
5	8
6	7
7	8
8	8
9	8
10	7
11	8
12	8
13	8
14	8
15	8
16	7
17	8
18	8
19	8
20	8
21	5
22	8
23	8
24	8
25	8
26	8
27	8
28	8
29	8
...	...
12444	3

12445	3
12446	3
12447	3
12448	3
12449	3
12450	3
12451	3
12452	3
12453	3
12454	4
12455	4
12456	4
12457	4
12458	4
12459	4
12460	3
12461	3
12462	3
12463	3
12464	3
12465	3
12466	3
12467	3
12468	3
12469	2
12470	2
12471	1
12472	1
12473	1

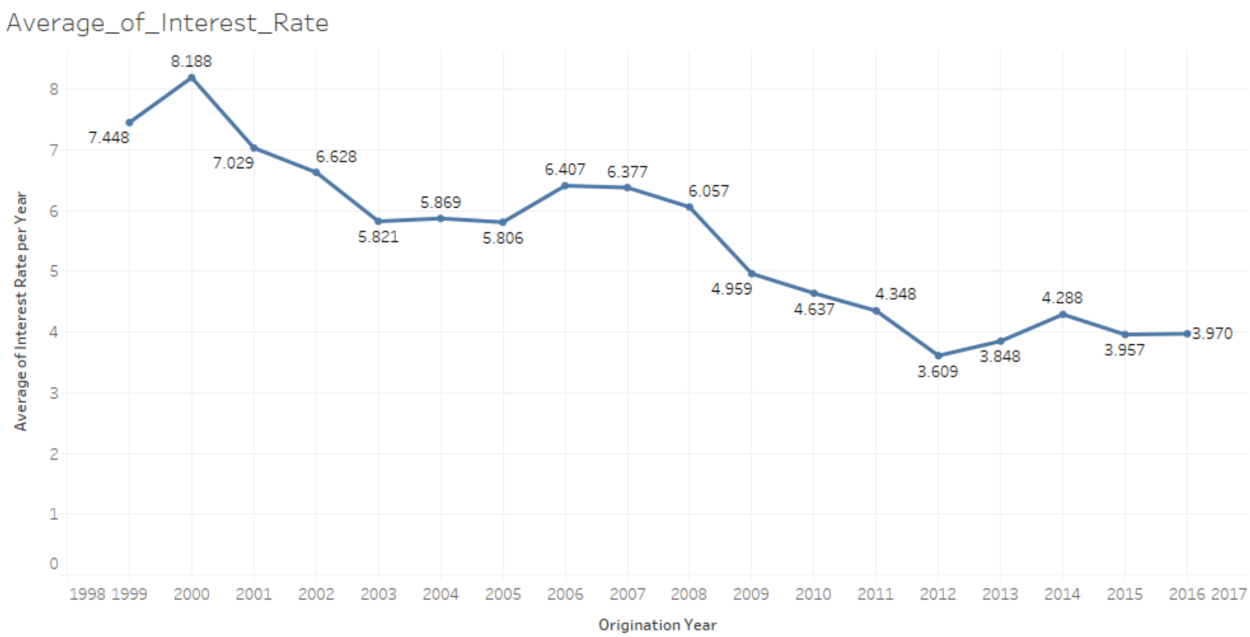
[12474 rows x 2 columns]

OBSERVATIONS

1) The Trend Lines for Average Credit Score Shows that 2012 has the highest Credit Score (766.54). Meanwhile, the Trend Lines for The Average Interest Rate is the least in 2012 (3.609)
Hence, it can be inferred that the Credit Score is the highest in 2012, since the Interest rate is the least

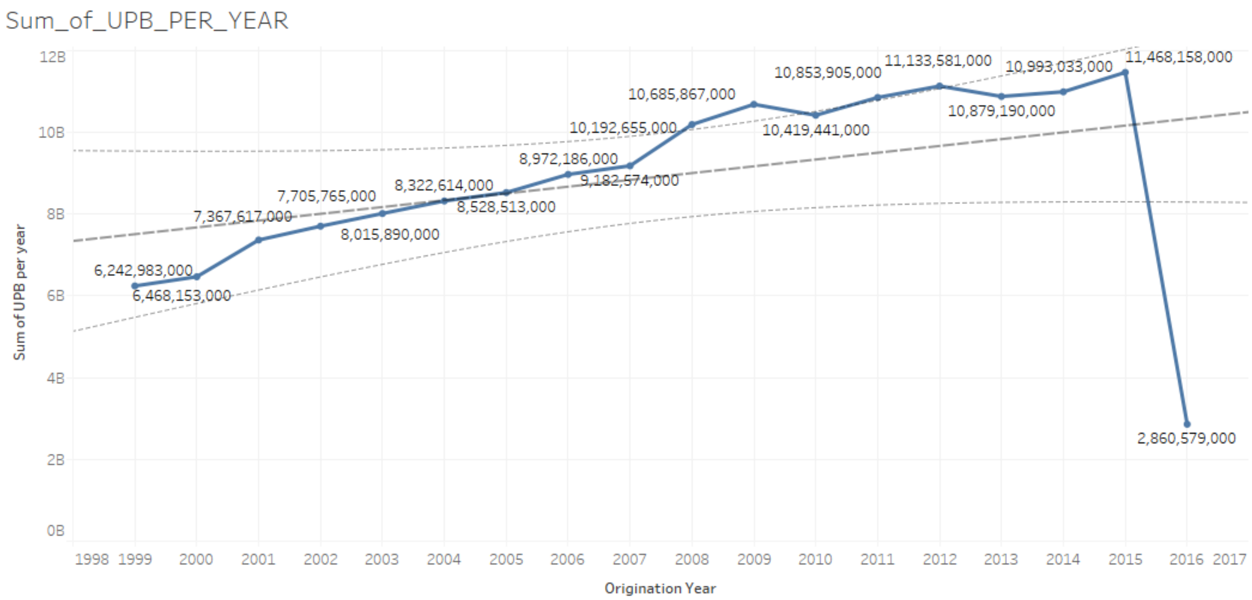


The trend of sum of Average of Credit Score per year for Origination Year.



The trend of sum of Average of Interest Rate per Year for Origination Year.

2) We observed that the Sums and Average values for the year 2016 is much lesser than the rest of the years, since only the data for the first quarter is available
For Example: The Sum of UPB for 2016 is much lesser than the rest



The trend of sum of Sum of UPB per year for Origination Year.

Programming Language used : Python

Workflow Manager User: Luigi

Tasks

- **F. Downloading Data**
- **G. Preprocessing Origination Data**
- **H. Preprocessing Performance Data**
- **I. Building prediction model**

DOWNLOADING DATA(PART 2):

File Location : Classes/Part1/Download_sf_loan.py

Task Requires no prior tasks to be completed.
Output of the task are all the sample origination and performance files.
Process:

- Asking user for username and password.
- Creating a browser agent (using the mechanicalsoup library) to store and pass the cookies
- Logging in with the user's credentials.
- Checking if the user is succesfuly logged in or not.
- Landing to the page that contains the list of files and download links

- Asking user for the year and the quarter file to run prediction model.
- Putting the table of files in a dataframe
- Iterating through the rows in dataframe for the links that contain sample files and downloading them to a newly created (if it doesn't already exist) "Downloads" directory
- The program also checks if the files are already present in the "Downloads" directory. It skips the downloading if the file already exists.
- Unzipping the downloaded file.

ORIGINATION FILE OBSERVATION AND CLEANING

5. Credit Score: Deleted the rows that had missing credit score
    a. Cannot replace missing values as it is explicitly specified that credit score can be either less than 301 or greater than 850.
    b. Number of such instances is very less (0.002% in 2016 to 1.242% in 2000)
    c. Removing the rows that have blank values and nulls for credit score.

CREDIT SCORE	COUNT OF BLANKS	YEAR
	362	1999
	621	2000
	274	2001
	201	2002
	33	2003
	42	2004
	24	2005
	39	2006
	29	2007
	29	2008
	1	2009
	1	2011
	3	2013
	2	2014
	1	2016

FIRST PAYMENT DATE:

No missing values in sample files

FIRST TIME HOMEBUYER FLAG:

e. If blank it can be replaced by NA if Occupancy Status is either "I" or "S" (Investment property or Second Home)
f. If blank it can replaced by NA if Loan Purpose is either "C" or "N" (Refinance)
g. If blank, then replace it with NA
h. Created three columns for – First Time Homebuyer Flag YES (1,0) , NO(1,0) and NA(1,0)

MATURITY DATE:

d. No missing values in the sample files
e. Splitting Maturity year and month

METROPOLITAN STATISTICAL AREA(MSA) OR METROPOLITAN DIVISION:

f. Replaced missing values with zero.
g. Derived a new column for Metropolitan Area Flag, that had values in it
h. Future Scope: Compare the values of zip codes, if the zip code belongs to a MSA or MD, then map the msa or md code in the data.

YEAR	COUNT OF BLANKS
1999	7640
2000	7542
2001	6978
2002	7309
2003	7182
2004	7844
2005	7913
2006	8209
2007	8671
2008	7729
2009	7528
2010	7022
2011	6944
2012	6593
2013	5475
2014	5030
2015	4845
2016	1184

MORTGAGE INSURANCE PERCENTAGE (MI%):

MORTGAGE INSURANCE PERCENTAGE (MI %)	COUNT	YEAR	Percentage
	9026	1999	18.052
0	21885	1999	43.77
	44	2000	0.088
0	32764	2000	65.528
	58	2001	0.116
0	36990	2001	73.98
	11	2002	0.022
0	38304	2002	76.608
	10	2003	0.02

0	40083	2003	80.166
	9	2004	0.018
0	40437	2004	80.874
	57	2005	0.114
0	43136	2005	86.272
0	43086	2006	86.172
0	39839	2007	79.678
0	40958	2008	81.916
0	46460	2009	92.92
0	46266	2010	92.532
0	44985	2011	89.97
0	43711	2012	87.422
0	40459	2013	80.918
0	36478	2014	72.956
0	37309	2015	74.618
0	9481	2016	18.962

i. Zero means No Mortgage insurance
j. Blanks Means either less than 1% or greater than 55%, so the replacement cannot be generalized in this case. Also, such cases are ~18% in 1999 and ~0.01% in until 2005 and 0 in the later years.
k. Deriving a new column for mortgage insurance flag is done, where the value is kept No if MI% is zero, otherwise it is made Yes

NUMBER OF UNITS:

	1	2000
	7	2004

l. No missing values for most sample files. Only 1 in the year 2000 and 7 cases in 2004 where number of units is missing
m. Replaced it with the mode OR Discard the row

OCCUPANCY STATUS:

n. No missing values in the sample files.
o. Handled the missing value by replacing it by mode or discarding the rows

ORIGINAL COMBINED LOAN-TO-VALUE(CLTV):

ORIGINAL COMBINED LOAN-TO-VALUE (CLTV)	COUNT	Year
	0	1999
	3	2000
	2	2001

	4	2002
	3	2003
	3	2004
	6	2005
	1	2006
	2	2007
	0	2008
	0	2009
	1	2010
	0	2011
	2	2012
	2	2013
	1	2014
	2	2015
	0	2016

p. ~0.01% missing values in the sample files.
q. If the LTV is less than 80 or greater than 200 or unknown, then this column is unknown. Also if CLTV is less than LTV then, CLTV is set to unknown.
r. This value is dependent on each individual case, so may not be replaced by mean, median or mode.

ORIGINAL DEBT-TO-INCOME (DTI) RATIO:

s. Ratio greater than 65% are represented as spaces. We replaced it by 70.
t. Unknowns are represented by null, which we replaced by the median.

ORIGINAL UPB:

u. No missing values in the sample files
v. If value is missing then discard the rows.

ORIGINAL LOAN-TO-VALUE:

w. Ratios below 6% and greater than 105% are unknown.

ORIGINAL LOAN-TO-VALUE (LTV)	COUNT	YEAR
	0	1999
	2	2000
	1	2001
	1	2002
	3	2003
	3	2004
	6	2005
	1	2006

	2	2007
	0	2008
	0	2009
	1	2010
	0	2011
	2	2012
	2	2013
	1	2014
	2	2015
	0	2016

x. Close to zero percent of such occurrence. But, replacing of the values with mean/median cannot be justified as it is specifically said that these values are either less than 6 or greater than 105. So, discarding such rows.

ORIGINAL INTEREST RATE:

y. No missing values
z. If value is missing then replace by median

CHANNEL:

aa. No missing values in sample files
bb. If values are missing then replace by mode

PREPAYMENT PENALTY MORTGAGE (PPM) FLAG:

PREPAYMENT PENALTY MORTGAGE (PPM) FLAG	COUNT	YEAR
	1247	1999
	236	2000
	122	2001
	171	2002
	198	2003
	73	2004
	49	2005
	65	2006
	113	2007
	1039	2008
	317	2009
	336	2010
	580	2011
	39	2012
	4	2013
	11	2014
	41	2015
	7	2016

cc. Most number of blanks (unknown) in the year 1999 -> 2.49%, 2008 -> 2.078%

1999	**48753**
N	48491
Y	262
2000	**49764**
N	49737
Y	27
2001	**49878**
N	49867
Y	11
2002	**49829**
N	49784
Y	45
2003	**49802**
N	49652
Y	150
2004	**49927**
N	49752
Y	175

dd. Maximum are “N” throughout the years. 97.5% in 1999, 99.5% in 2000…

ee. We are replacing unknown(blanks) values by mode as it wouldn’t affect the distribution.

6. PRODUCT TYPE:
   a. No missing values found in the observations
   b. If there are any missing values, then it is replaced with “FRM”
7. PROPERTY STATE:
   a. No missing values found in the observations
   b. If there are any missing values, then it is replaced with “Unknown”

PROPERTY TYPE:

PROPERTY TYPE	COUNT	YEAR
	8	2000
	11	2001
	3	2002
	14	2004

PROPERTY TYPE	COUNT	YEAR
	8	2000
CO	4090	2000
CP	74	2000

LH	15	2000
MH	244	2000
PU	6531	2000
SF	39038	2000
	11	2001
CO	3546	2001
CP	45	2001
LH	22	2001
MH	181	2001
PU	5470	2001
SF	40725	2001
	3	2002
CO	3399	2002
CP	48	2002
LH	12	2002
MH	274	2002
PU	5053	2002
SF	41211	2002
	14	2004
CO	3616	2004
CP	210	2004
LH	35	2004
MH	529	2004
PU	6829	2004
SF	38767	2004

c. No missing values for most of the years.
d. Very few missing values observed for years 2000, 2001, 2002 and 2004.
e. Replaced the missing values with the mode (“SF” as observed) because most number of records are categorized as Single Family Home (77% to 82%)

POSTAL CODE:

POSTAL CODE	COUNT	YEAR
	1	1999
	72	2000
	1	2001
	1	2002
	0	2003
	0	2004
	1	2005

	0	2006
	0	2007
	0	2008
	0	2009
	0	2010
	0	2011
	0	2012
	0	2013
	0	2014
	0	2015
	0	2016

f. 72 of 50000 unknowns in 2000, 1 row each in 1999, 2001,2002 and 2005 of unknowns
g. Replaced the blanks with 99999 as unknown value
h. Future Scope: Get a complete dictionary of Metropolitan Statistical Area or Metropolitan Division codes and map the MSA or MD for the row to the dictionary to find the missing postal code

LOAN SEQUQENCE NUMBER:

i. Unique Identifier Column.
j. No missing values. If the value is missing for a row, then replace by random Loan sequence number the complete row or generating a unique identifier UUID
k. Derived two new columns for origination year and origination quarter

LOAN PURPOSE:

l. No missing values in the sample files.
m. If the values are missing then, loan purpose is unknown. Assuming that the percentage of such occurrence in the yearly data would be close (if not equal to) 0%, and it wouldn't affect the distribution of the data, we replaced it by the mode of the column

ORIGINAL LOAN TERM:

n. No missing values observed.

NUMBER OF BORROWERS:

NUMBER OF BORROWERS	COUNT	YEAR
	30	1999
	20	2000
	11	2001
	9	2002
	7	2003
	14	2004
	17	2005
	17	2006

	23	2007
	19	2008
	6	2009
	0	2010
	0	2011
	0	2012
	0	2013
	0	2014

o. 0% to 0.6% Missing values found.
p. Replacing missing values with the mode.

SELLER NAME:

q. No missing values found in the sample files.
r. Replacing missing values by "Unknown"

SERVICES NAME:

s. No missing values found in the sample files.
t. Replacing missing values by "Unknown"

SUPER CONFORMING FLAG:

SUPER CONFORMING FLAG	COUNT	YEAR
Y	80	2008
Y	1236	2009
Y	1364	2010
Y	1967	2011
Y	2189	2012
Y	1718	2013
Y	1995	2014
Y	2223	2015
Y	492	2016

b. Per the data dictionary, all the missing values are Not super conforming, so replaced the missing values by "N"

PERFORMANCE FILE

2. LOAN SEQUENCE NUMBER:
   a. Derived two new columns for origination year and origination quarter
3. MONTHLY REPORTING PERIOD:
   a. Derived two new columns for monthly reporting period year and month
4. CURRENT ACTUAL UPB:

5. CURRENT LOAN DELINQUENCY STATUS:
    a. No Missing values observed in the sample files.
    b. Replacing missing values with “XX” which is also used for unknown.
6. LOAN AGE:
    a. No missing values observed.
7. REMAINING MONTHS TO LEGAL MATURITY:
    a. No missing values found.
8. REPURCHASE FLAG:
    a. This field is only populated at loan termination. For all others the value is not applicable.
    b. Replacing nulls with NA.
9. MODIFICATION FLAG:
    a. Replacing nulls with “NO” (Not modified)
10. ZERO BALANCE CODE:
    a. Replacing nulls and spaces with “NA” as it is not applicable if the balance is not reduced to zero.
11. ZERO BALANCE EFFECTIVE DATE:
    a. Replacing missing values with 999999, which will denote not applicable.
    b. Deriving 2 new columns for zero balance effective year and month.
12. CURRENT INTEREST RATE:
    a. Replacing empty values with 0.
13. DUE DATE OF LAST PAID INSTALLMENT:
    a. Replacing missing values with 999999.
    b. Deriving 2 new columns for due year and month of last paid installment.
14. Replacing missing values with 0 for the following columns
    a. MI RECOVERIES
    b. NET SALES PROCEEDS
    c. NON MI RECOVERIES
    d. EXPENSES
    e. LEGAL COSTS
    f. MAINTENANCE AND PRESERVATION COSTS:
    g. TAXES AND INSURANCE:
    h. MISCELLENEOUS EXPENSES:
    i. ACTUAL LOSS CALCULATION:
    j. MODIFICATION COST
    k. CURRENT DEFERRED UPB

CLASSIFICATION (LOGISTIC REGRESSION)

SUMMARY OF THE LOGISTIC REGRESSION

```
> summary(modelLogit)


Call:
glm(formula = DELINQUENT ~ ., family = binomial(link = "logit"), 
   data = train)
```

```
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.4397  -0.3161  -0.2212  -0.1563   3.5311

Coefficients: (6 not defined because of singularities)
                                     Estimate Std. Error  z value Pr(>|z|)
(Intercept)                        -1.289e+00  5.889e-02  -21.882  < 2e-16 ***
CURRENTACTUALUPB                    3.625e-06  3.433e-08  105.574  < 2e-16 ***
LOANAGE                             2.221e-02  8.232e-05  269.852  < 2e-16 ***
REMAININGMONTHSTOLEGALMATURITY      2.346e-03  4.261e-05   55.061  < 2e-16 ***
ZEROBALANCECODE                    -4.294e-03  2.531e-04  -16.964  < 2e-16 ***
CURRENTINTERESTRATE                 2.093e-02  4.267e-03    4.905 9.36e-07 ***
MONTHLYREPORTINGYEAR                       NA         NA       NA       NA
MONTHLYREPORTINGMONTH                      NA         NA       NA       NA
REPURCHASEFLAGYES                   2.890e+00  1.895e-01   15.255  < 2e-16 ***
MODIFICATIONFLAGYES                 5.272e+00  1.060e-01   49.717  < 2e-16 ***
ZEROBALANCEEFFECTIVEYEAR                   NA         NA       NA       NA
ZEROBALANCEEFFECTIVEMONTH                  NA         NA       NA       NA
DUEDATEOFLASTPAIDINSTALLMENTYEAR           NA         NA       NA       NA
DUEDATEOFLASTPAIDINSTALLMENTMONTH          NA         NA       NA       NA
CREDIT_SCORE                       -1.025e-02  3.998e-05 -256.290  < 2e-16 ***
NUMBER_OF_UNITS                    -1.825e-01  1.235e-02  -14.777  < 2e-16 ***
ORIGINAL_INTEREST_RATE              5.930e-01  8.312e-03   71.348  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1354776  on 3551480  degrees of freedom
Residual deviance: 1155220  on 3551470  degrees of freedom
AIC: 1155242

Number of Fisher Scoring iterations: 6

> pR2(modelLogit)
          llh       llhNull            G2      McFadden          r2ML          r2CU
-5.776098e+05 -6.773880e+05  1.995563e+05  1.472984e-01  5.464012e-02  1.722893e-01
```

CONFUSION MATRIX FOR LOGISTIC REGRESSION

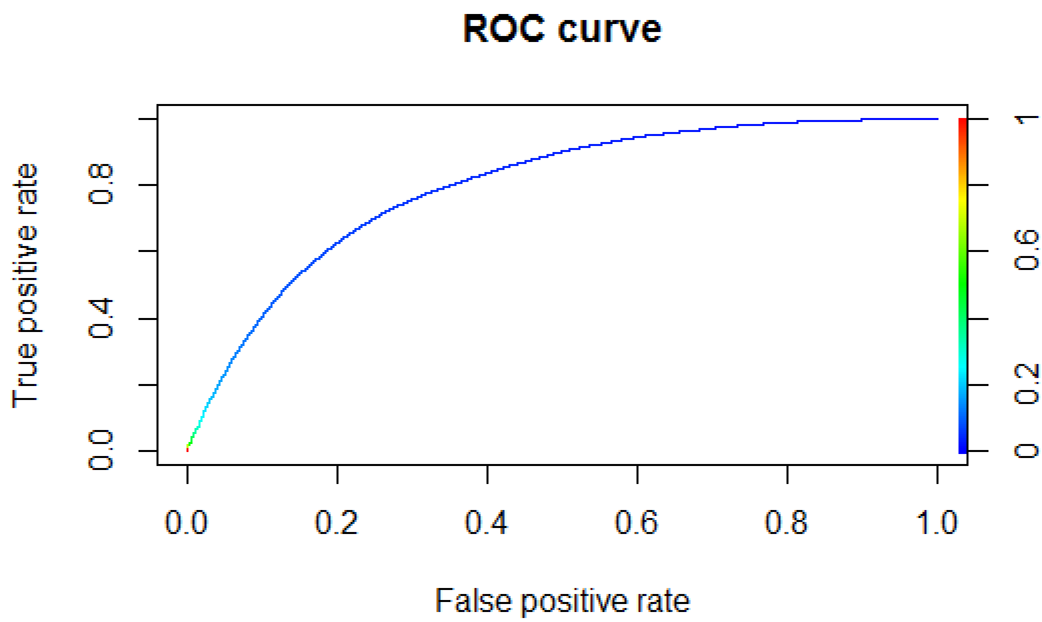
```
> confusionMatrix(data=factor(pred.resp.level),reference=factor(test$DELINQUENT),positive='1')
Confusion Matrix and Statistics
```

PREDICTION

Reference		
Prediction	0	1
0	2665523	209453
1	6246	5076

```
            Accuracy : 0.9253
              95% CI : (0.925, 0.9256)
 No Information Rate : 0.9257
 P-Value [Acc > NIR] : 0.9957

               Kappa : 0.0378
Mcnemar's Test P-Value : <2e-16

         Sensitivity : 0.023661
         Specificity : 0.997662
      Pos Pred Value : 0.448331
      Neg Pred Value : 0.927146
          Prevalence : 0.074327
      Detection Rate : 0.001759
Detection Prevalence : 0.003923
   Balanced Accuracy : 0.510662

    'Positive' Class : 1

> delinquent.logistic.error
[1] 0.07473206
>
```

ROC CURVE



NEURAL NETWORK

```
fitnn <- nnet(DELINQUENT ~ ., traindatannet, size=20,
+         maxit = 90, entropy = TRUE, softmax = FALSE,censored = FALSE, skip = FALSE,
+         rang = 0.7, Hess = FALSE, trace = TRUE, MaxNWts = 1000, abstol = 1.0e-4,
+         decay = 15e-4, reltol = 1.0e-8, hidden = 2,threshold = 0.01,act.fct="tanh")
```

```
# weights:  361
initial  value 30046.231065
iter  10 value 10542.487357
iter  20 value 10494.425353
iter  30 value 10461.239244
iter  40 value 10452.427974
iter  50 value 10450.919183
iter  60 value 10450.736090
iter  70 value 10450.191369
iter  80 value 10449.676547
iter  90 value 10448.726339
…
…
..
```

final value 10448.726339

> fitnn

a 16-20-1 network with 361 weights

inputs: CURRENTACTUALUPB LOANAGE REMAININGMONTHSTOLEGALMATURITY ZEROBALANCECODE CURRENTINTERESTRATE MONTHLYREPORTINGYEAR MONTHLYREPORTINGMONTH REPURCHASEFLAGYES MODIFICATIONFLAGYES ZEROBALANCEEFFECTIVEYEAR ZEROBALANCEEFFECTIVEMONTH DUEDATEOFLASTPAIDINSTALLMENTYEAR DUEDATEOFLASTPAIDINSTALLMENTMONTH CREDIT_SCORE NUMBER_OF_UNITS ORIGINAL_INTEREST_RATE

output(s): DELINQUENT

options were - entropy fitting decay=0.0015

CONFUSION MATRIX FOR NEURAL NETWORK

> confusionMatrix(data=pred.resp.nnet.factor,reference=factor(testdatannet$DELINQUENT), positive='1')

Confusion Matrix and Statistics

PREDICTION

	Reference	
Prediction	0	1
0	26	140
1	46450	2764

```
            Accuracy : 0.0682
              95% CI : (0.066, 0.0704)
 No Information Rate : 0.9419
 P-Value [Acc > NIR] : 1
               Kappa : -0.0041
Mcnemar's Test P-Value : <2e-16
         Sensitivity : 0.95179
         Specificity : 0.01372
      Pos Pred Value : 0.05616
      Neg Pred Value : 0.82188
          Prevalence : 0.05808
      Detection Rate : 0.05528
Detection Prevalence : 0.98428
   Balanced Accuracy : 0.48275
    'Positive' Class : 1
```

> #computing the overall error - 0.32

> delinquent.neural.error <- 1- sum(pred.resp.nnet.factor==testdatannet$DELINQUENT)/length(testdatannet$DELINQUENT)

> delinquent.neural.error

[1] 0.9318

PREDICTION

Regression model for the interest rate.

LINEAR REGRESSION

ALL VARIABLES CONSIDERED (45)

The dataset 'train' consists of all the columns from the cleaned sample origination files.

OUTPUT:

```
+                        'SUPERCONFORMINGFLAGYES','SUPERCC
> library(forecast)
> lm.fit = lm(ORIGINALINTERESTRATE ~ .,data = train)
> summary(lm.fit)

Call:
lm(formula = ORIGINALINTERESTRATE ~ ., data = train)

Residuals:
     Min       1Q   Median       3Q      Max
-2.68723 -0.16869 -0.00276  0.16019  2.83441

Coefficients: (13 not defined because of singularities)
```

The coefficients are given for the columns:

13 columns were removed as they were singular i.e, removing these columns will not affect the coefficients of the model

```
                                                          Estimate Std. Error  t value Pr(>|t|)    
(Intercept)                                             -2.146e+02  7.638e+00  -28.103  < 2e-16 ***
CREDITSCORE                                             -7.178e-04  8.079e-06  -88.856  < 2e-16 ***
FIRSTPAYMENTYEAR                                         8.531e-02  3.811e-03   22.387  < 2e-16 ***
FIRSTPAYMENTMONTH                                       -5.135e-02  1.023e-03  -50.209  < 2e-16 ***
MATURITYYEAR                                             2.485e-02  8.331e-05  298.318  < 2e-16 ***
MATURITYMONTH                                           -2.845e-02  1.002e-03  -28.384  < 2e-16 ***
METROPOLITANSTATISTICALAREA.MSA.ORMETROPOLITANDIVISION   6.971e-07  4.360e-08   15.990  < 2e-16 ***
MORTGAGEINSURANCEPERCENTAGE.MI..                         1.775e-04  2.286e-05    7.764 8.23e-15 ***
NUMBEROFUNITS                                            1.218e-02  2.285e-03    5.328 9.95e-08 ***
ORIGINALCOMBINEDLOAN.TO.VALUE.CLTV.                      1.931e-04  4.525e-05    4.266 1.99e-05 ***
ORIGINALDEBT.TO.INCOME.DTI.RATIO                         3.862e-04  3.464e-05   11.149  < 2e-16 ***
ORIGINALUPB                                             -7.267e-07  5.952e-09 -122.102  < 2e-16 ***
ORIGINALLOAN.TO.VALUE.LTV                                8.779e-04  5.645e-05   15.552  < 2e-16 ***
POSTALCODE                                              -3.974e-07  1.572e-08  -25.276  < 2e-16 ***
ORIGINALLOANTERM                                                NA         NA       NA       NA    
NUMBEROFBORROWERS                                       -1.734e-02  9.485e-04  -18.284  < 2e-16 ***
FIRSTTIMEHOMEBUYERFLAGYES                               -1.095e-02  2.282e-03   -4.798 1.60e-06 ***
FIRSTTIMEHOMEBUYERFLAGNO                                -1.194e-02  1.549e-03   -7.710 1.26e-14 ***
FIRSTTIMEHOMEBUYERFLAGNA                                        NA         NA       NA       NA    
METROPOLITAN_AREA_FLAG                                  -2.032e-02  1.834e-03  -11.078  < 2e-16 ***
MORTGAGE_INSURANCE_FLAG                                  9.754e-02  1.649e-03   59.153  < 2e-16 ***
OWNEROCCUPIEDFLAG                                       -2.552e-02  2.162e-03  -11.808  < 2e-16 ***
INVESTMENTPROPERTYFLAG                                   3.073e-01  3.388e-03   90.708  < 2e-16 ***
SECONDHOMESPACEFLAG                                             NA         NA       NA       NA    
RETAILCHANNELFLAG                                        8.852e-02  9.502e-04   93.161  < 2e-16 ***
BROKERCHANNELFLAG                                        2.485e-01  1.755e-02   14.161  < 2e-16 ***
CORRESPONDENTCHANNELFLAG                                 4.064e-01  1.159e-02   35.056  < 2e-16 ***
TP0NOTSPECIFIEDCHANNELFLAG                                      NA         NA       NA       NA    
PREPAYMENTPENALTYMORTGAGE.PPM.FLAGYES                    1.120e-01  1.127e-02    9.944  < 2e-16 ***
PREPAYMENTPENALTYMORTGAGE.PPM.FLAGNO                            NA         NA       NA       NA    
FIXEDRATEMORTGAGEPRODUCTTYPEFLAGYES                             NA         NA       NA       NA    
FIXEDRATEMORTGAGEPRODUCTTYPEFLAGNO                              NA         NA       NA       NA    
CONDOPROPERTYTYPEFLAG                                   -1.678e-03  8.071e-03   -0.208 0.835264    
LEASEHOLDPROPERTYTYPEFLAG                                6.743e-02  1.871e-02    3.603 0.000314 ***
PUDPROPERTYTYPEFLAG                                     -2.637e-02  8.000e-03   -3.296 0.000982 ***
MANUFACTUREHOUSINGPROPERTYTYPEFLAG                       2.415e-01  8.846e-03   27.301  < 2e-16 ***
FREESIMPLEHOUSINGPROPERTYTYPEFLAG                       -1.379e-02  7.912e-03   -1.743 0.081327 .  
COOPHOUSINGPROPERTYTYPEFLAG                                     NA         NA       NA       NA    
ORIGINATIONYEAR                                                 NA         NA       NA       NA    
ORIGINATIONQUARTER                                              NA         NA       NA       NA    
LOANPURPOSEISPURCHASEFLAG                               -5.893e-02  1.316e-03  -44.771  < 2e-16 ***
LOANPURPOSEISCASHOUTREFINANCEFLAG                        1.613e-02  1.232e-03   13.086  < 2e-16 ***
```

PREDICTION ON TEST DATA

```
In [10]: pred = predict(lm.fit, test)
         accuracy(pred, train$ORIGINALINTERESTRATE)

Warning message in predict.lm(lm.fit, test):
"prediction from a rank-deficient fit may be misleading"
```

	ME	RMSE	MAE	MPE	MAPE
Test set	0.1685502	0.3997372	0.3092227	2.573145	5.243713

MODEL TWO – SELCTING STATISTICALLY SIGNIFICANT COLUMNS (P <0.05) –

Select the columns that were considered significant by the previous model.

```
:rain2 <- subset(train, select =  c('CREDITSCORE','FIRSTPAYMENTYEAR','FIRSTPAYMENTMONTH',
      'MATURITYYEAR','MATURITYMONTH',
      'METROPOLITANSTATISTICALAREA.MSA.ORMETROPOLITANDIVISION',
      'MORTGAGEINSURANCEPERCENTAGE.MI..','NUMBEROFUNITS',
      'ORIGINALCOMBINEDLOAN.TO.VALUE.CLTV.','ORIGINALDEBT.TO.INCOME.DTI.RATIO',
      'ORIGINALUPB','ORIGINALLOAN.TO.VALUE.LTV','ORIGINALINTERESTRATE',
      'POSTALCODE','NUMBEROFBORROWERS','FIRSTTIMEHOMEBUYERFLAGYES',
      'FIRSTTIMEHOMEBUYERFLAGNO','METROPOLITAN_AREA_FLAG','MORTGAGE_INSURANCE_FLAG','OWNEROCCUPIEDFLA
     'RETAILCHANNELFLAG','BROKERCHANNELFLAG','CORRESPONDENTCHANNELFLAG',
      'TPONOTSPECIFIEDCHANNELFLAG','PREPAYMENTPENALTYMORTGAGE.PPM.FLAGYES',
     'CONDOPROPERTYTYPEFLAG','LEASEHOLDPROPERTYTYPEFLAG',
      'PUDPROPERTYTYPEFLAG','MANUFACTUREHOUSINGPROPERTYTYPEFLAG','FREESIMPLEHOUSINGPROPERTYTYPEFLAG',
      'LOANPURPOSEISPURCHASEFLAG',
      'LOANPURPOSEISCASHOUTREFINANCEFLAG'))
```

PREDICTION ON TEST DATA

```
#Removing the insignificant columns (Singularity)
pred = predict(lm.fit, test)
accuracy(pred, train2$ORIGINALINTERESTRATE)
```

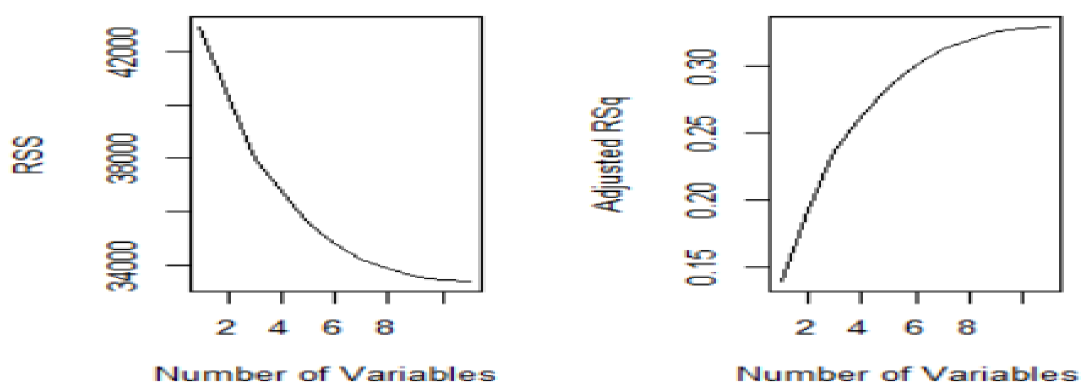
The accuracy measure do not change as expected

VALIDATION WITH OBSERVED DATA

```
> accuracy(pred, train2$ORIGINALINTERESTRATE)
                ME      RMSE       MAE      MPE     MAPE
Test set 0.1685502 0.3997372 0.3092227 2.573145 5.243713

> head(res)
  values      ind
1  6.875 Observed
2  5.625 Observed
3  6.125 Observed
4  6.125 Observed
5  6.125 Observed
6  6.125 Observed
> print(RMSE)
[1] 0.3500248206
```

EXHAUSTIVE SEARCH

```
# display results
#**********************************************************************************************
#EXHAUSTIVE SEARCH
#install.packages('ISLR',repos = "http://cran.us.r-project.org")
#install.packages("leaps", repos = "http://cran.us.r-project.org")
library(leaps)
library(ISLR)
regfit.full = regsubsets(ORIGINALINTERESTRATE ~ .,data = train2,method = "exhaustive",nvmax = 10)
reg.summary = summary(regfit.full)
names(reg.summary)
reg.summary$rss
```



Since RSS seems to flatten at about 10 variables, we can conclude that these ten variables will influence my model the most. We obtain the 10 columns and their corresponding co-efficients.

```
> coef(regfit.full ,10)
                       (Intercept)                       CREDITSCORE                FIRSTPAYMENTMONTH
               -41.3369085816142174               -0.0008061022807977              -0.0774275355854282
                      MATURITYYEAR                        ORIGINALUPB       MORTGAGE_INSURANCE_FLAG
                0.0238010222965116               -0.0000008177401646               0.1212752164660597
            INVESTMENTPROPERTYFLAG          CORRESPONDENTCHANNELFLAG PREPAYMENTPENALTYMORTGAGE.PPM.FLAGYES
                0.3320840177423762                0.3605368513953445               0.1494393065342177
 FREESIMPLEHOUSINGPROPERTYTYPEFLAG    LOANPURPOSEISCASHOUTREFINANCEFLAG
               -0.0089520528252344                0.0417240429487903
```

TRAINING ON THE COLUMNS SELECTED USING EXHAUSTIVE SEARCH

```
#Choosing exhaustive search columns
train_exhaustive = subset(train2, select =  c('CREDITSCORE','FIRSTPAYMENTMONTH',
                                'MATURITYYEAR','ORIGINALUPB',
                                'MORTGAGE_INSURANCE_FLAG',
                                'INVESTMENTPROPERTYFLAG','CORRESPONDENTCHANNELFLAG','ORIGINALINTERESTRATE',
                                'PREPAYMENTPENALTYMORTGAGE.PPM.FLAGYES','FREESIMPLEHOUSINGPROPERTYTYPEFLAG','LOANPURPOSEISCASHOU
lm.fit_exhaustive = lm(ORIGINALINTERESTRATE ~ .,data = train_exhaustive)
summary(lm.fit_exhaustive)
```

SUMMARY OF THE MODEL

```
(Intercept)                          -41.336908581684867   0.164943526595522 -250.61249 < 0.000000000000000222 ***
CREDITSCORE                           -0.000806102280797   0.000008025975234 -100.43668 < 0.000000000000000222 ***
FIRSTPAYMENTMONTH                     -0.077427535584556   0.000477504054074 -162.15053 < 0.000000000000000222 ***
MATURITYYEAR                           0.023801022296543   0.000080980977288  293.90881 < 0.000000000000000222 ***
ORIGINALUPB                           -0.000000817740165   0.000000005662563 -144.41166 < 0.000000000000000222 ***
MORTGAGE_INSURANCE_FLAG                0.121275216466159   0.001362062508303   89.03792 < 0.000000000000000222 ***
INVESTMENTPROPERTYFLAG                 0.332084017742415   0.002700869563483  122.95448 < 0.000000000000000222 ***
CORRESPONDENTCHANNELFLAG               0.360536851395320   0.011851171152688   30.42204 < 0.000000000000000222 ***
PREPAYMENTPENALTYMORTGAGE.PPM.FLAGYES  0.149439306534013   0.011515421246847   12.97732 < 0.000000000000000222 ***
FREESIMPLEHOUSINGPROPERTYTYPEFLAG     -0.008952052825319   0.001148398848642   -7.79525   0.0000000000000064434 ***
LOANPURPOSEISCASHOUTREFINANCEFLAG      0.041724042949117   0.001006152040951   41.46892 < 0.000000000000000222 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.2925614 on 405411 degrees of freedom
Multiple R-squared:  0.3034926, Adjusted R-squared:  0.3034754
F-statistic: 17665.17 on 10 and 405411 DF,  p-value: < 0.000000000000000222
```

PREDICTING

```
#Test Exhaustive subset
test_exhaustive = subset(test, select =  c('CREDITSCORE','FIRSTPAYMENTMONTH',
                                'MATURITYYEAR','ORIGINALUPB',
                                'MORTGAGE_INSURANCE_FLAG',
                                'INVESTMENTPROPERTYFLAG','CORRESPONDENTCHANNELFLAG','ORIGINALINTERESTRATE',
                                'PREPAYMENTPENALTYMORTGAGE.PPM.FLAGYES','FREESIMPLEHOUSINGPROPERTYTYPEFLAG','LOANPURP(

#Predict Exhaustive
pred_exhaustive = predict(lm.fit_exhaustive, test_exhaustive)
#Accuracy Exhaustive
accuracy(pred_exhaustive, train_exhaustive$ORIGINALINTERESTRATE)
```

RESULTS

```
> #Predict Exhaustive
> pred_exhaustive = predict(lm.fit_exhaustive, test_exhaustive)
> #Accuracy Exhaustive
> accuracy(pred_exhaustive, train_exhaustive$ORIGINALINTERESTRATE)
                    ME         RMSE          MAE         MPE         MAPE
Test set 0.1632722895 0.3957662754 0.3056607124 2.478774059 5.184506586
>
```

VALIDATION

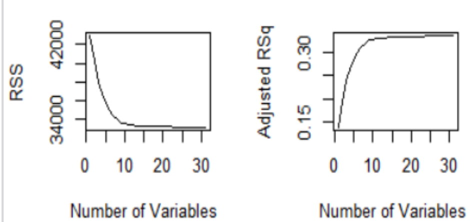
```
> pred_exhaustive_interest <- as.data.frame(pred_exhaustive)
> #VALIDATION EXHAUSTIVE
> x <- (test_interest - pred_exhaustive_interest)
> MSE <- sum((x^2))/nrow(test_interest)
> RMSE<-sqrt(MSE)
> print(RMSE)
[1] 0.3506538842
```

FORWARD SELECTION

```
#FORWARD SELECTION

library(leaps)
library(ISLR)
regfit.forward = regsubsets(ORIGINALINTERESTRATE ~ .,data = train2,method = "forward",nvmax = 30)
reg.fwd.summary = summary(regfit.forward)
names(reg.fwd.summary)
print(reg.fwd.summary)
```

CHECKING CURVES FOR COLUMN SELECTION

```
#Plotting
par(mfrow=c(2,2))
plot(reg.fwd.summary$rss ,xlab="Number of Variables ",ylab="RSS", type="l")
plot(reg.fwd.summary$adjr2 ,xlab="Number of Variables ", ylab="Adjusted RSq",type="l")
coef(regfit.forward ,10)
```



RSS flattens out at 10 variables. This means that we don't need more than ten significance columns to explain the model

We move ahead to pick these ten columns and their co-efficients

```
> par(mfrow=c(2,2))
> plot(reg.fwd.summary$rss ,xlab="Number of Variables ",ylab="RSS", type="l")
> plot(reg.fwd.summary$adjr2 ,xlab="Number of Variables ", ylab="Adjusted RSq",type="l")
> coef(regfit.forward ,10)
                        (Intercept)                        CREDITSCORE                  FIRSTPAYMENTMONTH
               -41.3369085816294373             -0.0008061022807973               -0.0774275355844824
                       MATURITYYEAR                        ORIGINALUPB         MORTGAGE_INSURANCE_FLAG
                  0.0238010222965157             -0.0000008177401646                0.1212752164661729
              INVESTMENTPROPERTYFLAG         CORRESPONDENTCHANNELFLAG PREPAYMENTPENALTYMORTGAGE.PPM.FLAGYES
                  0.3320840177424165              0.3605368513954511                0.1494393065342023
    FREESIMPLEHOUSINGPROPERTYTYPEFLAG   LOANPURPOSEISCASHOUTREFINANCEFLAG
                 -0.0089520528253193              0.0417240429491150
>
```

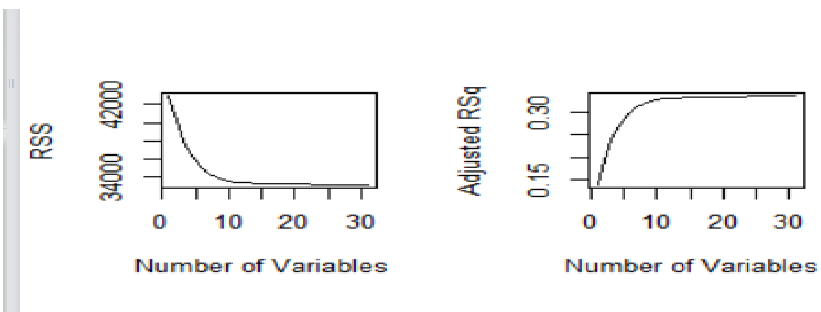
These columns are exactly the same as we got for exhaustive search. Hence, we expect the same accuracy

BACKWARD SELECTION

```
#BACKWARD SELECTION

library(leaps)
library(ISLR)
regfit.backward = regsubsets(ORIGINALINTERESTRATE ~ .,data = train2,method = "backward",nvmax = 30)
reg.bcwd.summary = summary(regfit.backward)
names(reg.bcwd.summary)
#Plotting
par(mfrow=c(2,2))
plot(reg.bcwd.summary$rss ,xlab="Number of Variables ",ylab="RSS", type="l")
plot(reg.bcwd.summary$adjr2 ,xlab="Number of Variables ", ylab="Adjusted RSq",type="l")
coef(regfit.backward ,10)
```

A similar curve, flattening at 10 variables is observed.

And the following ten varaibles are chosen



```
> coef(regfit.backward ,10)
                    (Intercept)                       CREDITSCORE                  FIRSTPAYMENTMONTH
             -43.6044644306088500                -0.0008107461254836               -0.0779461357198873
                   MATURITYYEAR                        ORIGINALUPB         MORTGAGE_INSURANCE_FLAG
               0.0248944890879210                -0.0000007545784086                0.1249299086553009
         INVESTMENTPROPERTYFLAG                  RETAILCHANNELFLAG        CORRESPONDENTCHANNELFLAG
               0.3279751543040528                 0.0894127439457250                0.4614354783359391
FREESIMPLEHOUSINGPROPERTYTYPEFLAG LOANPURPOSEISCASHOUTREFINANCEFLAG
              -0.0098336400214939                 0.0441510749684636
>
```

SUBSET COLUMNS AS PER BACKWARD SELECTION AND PREDICTION

```
#Choosing backward search columns
train_backward  = subset(train, select =  c('CREDITSCORE','FIRSTPAYMENTMONTH',
                                             'MATURITYYEAR','ORIGINALUPB',
                                             'MORTGAGE_INSURANCE_FLAG',
                                             'INVESTMENTPROPERTYFLAG','RETAILCHANNELFLAG','CORRESPONDENTCHANNELFLAG',
                                             'FREESIMPLEHOUSINGPROPERTYTYPEFLAG','ORIGINALINTERESTRATE','LOANPURPOSEISCASHOUTREFI
lm.fit_backward = lm(ORIGINALINTERESTRATE ~ .,data = train_backward)
summary(lm.fit_backward)
#Test Exhaustive subset
test_backward = subset(train, select =  c('CREDITSCORE','FIRSTPAYMENTMONTH',
                                           'MATURITYYEAR','ORIGINALUPB',
                                           'MORTGAGE_INSURANCE_FLAG',
                                           'INVESTMENTPROPERTYFLAG','RETAILCHANNELFLAG','CORRESPONDENTCHANNELFLAG',
                                           'FREESIMPLEHOUSINGPROPERTYTYPEFLAG','LOANPURPOSEISCASHOUTREFINANCEFLAG'))
#Predict Exhaustive
pred_backward = predict(lm.fit_backward, test_backward)
#Accuracy Exhaustive
accuracy(pred_backward, train_backward$ORIGINALINTERESTRATE)
#*******************************************************************************************************
```

```
> #Predict Exhaustive
> pred_backward = predict(lm.fit_backward, test_backward)
> #Accuracy Exhaustive
> accuracy(pred_backward, train_backward$ORIGINALINTERESTRATE)
                              ME      RMSE       MAE        MPE     MAPE
Test set 0.000000000000005198129646 0.2893635542 0.2157720901 -0.2492952937 3.738865316
>
```

ALGORITHM 2 – RANDOM FOREST

```
#RANDOM FOREST
#*******************************************************************************************
#install.packages("randomForest")
#install.packages("MASS")
library(randomForest)
library(MASS)
randomForestfit <- randomForest(ORIGINALINTERESTRATE ~ .,data = train_backward, n_tree=20)
```

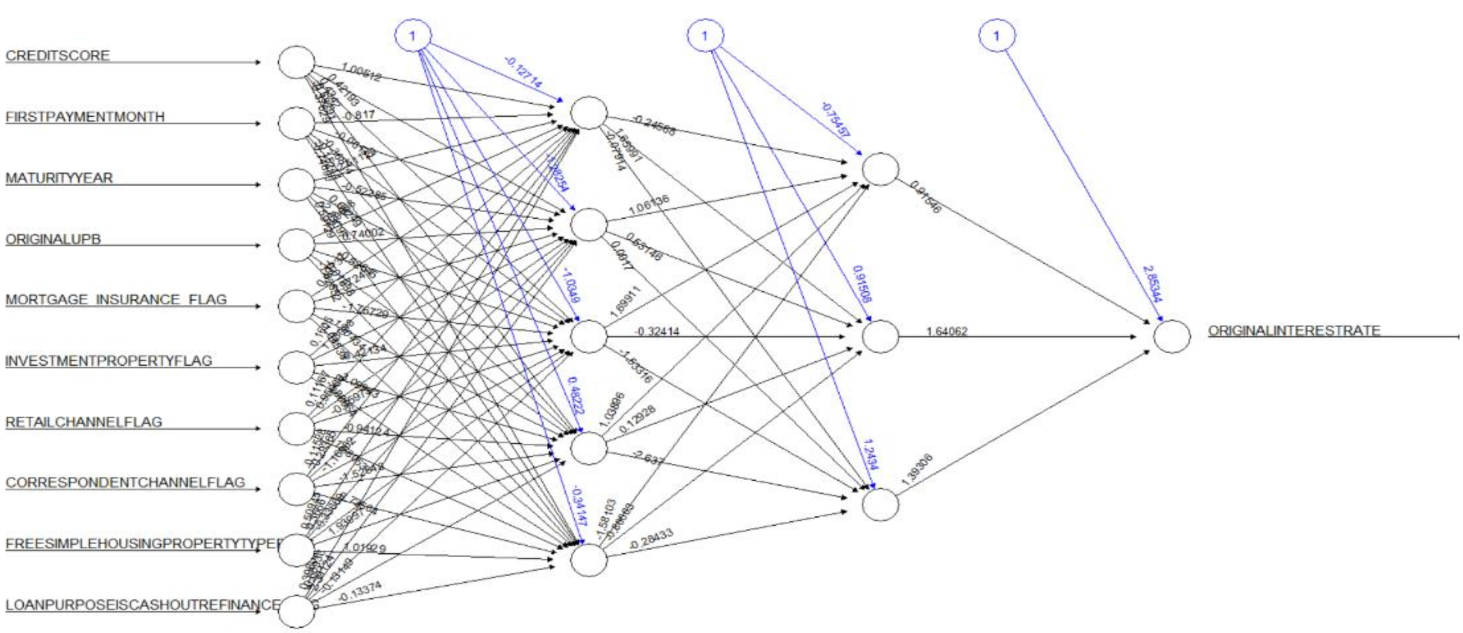
PREDICTION

```
proximity               0 -none- NULL
ntree                   1 -none- numeric
mtry                    1 -none- numeric
forest                 11 -none- list
coefs                   0 -none- NULL
y                  263804 -none- numeric
test                    0 -none- NULL
inbag                   0 -none- NULL
terms                   3 terms  call
> predictedrandomForest = predict(randomForestfit,test_r)
> accuracy(predictedrandomForest, train_r$ORIGINAL_INTEREST_RATE)
                    ME     RMSE       MAE        MPE     MAPE
Test set -0.0001373272 0.444603 0.3437845 -0.4251312 6.121744
>
```

ALGORITHM 3- NEURAL NETWORK

CREATING THE NETWORK

```
n <- names(train_backward)
f <- as.formula(paste("ORIGINALINTERESTRATE~", paste(n[!n %in% "ORIGINALINTERESTRATE"],collapse="+")))

net.interest <- neuralnet(f, data = train_backward, hidden=c(5,3),linear.output=T)

plot(net.interest)
```



PREDICTION

```
#PREDICTING USING THE NEURAL NETWORK
predicted.nn.values <- compute(net.interest,test_backward)
#Caculating MSE
test_nn_values <- as.data.frame(test$ORIGINALINTERESTRATE)
pred_df <- as.data.frame(predicted.nn.values$net.result)
x <- (test_nn_values - pred_df)
sum((x^2))/405422
```

CHECK MEAN SQUARE ERROR

```
> test_nn_values <- as.data.frame(test$ORIGINALINTERESTRATE)
> sum((x^2))/nrow(test_nn_values)
[1] 0.1229720542
>
```

> RMSE = sqrt(0.1220720542)

> print(RMSE)

[1] 0.3506

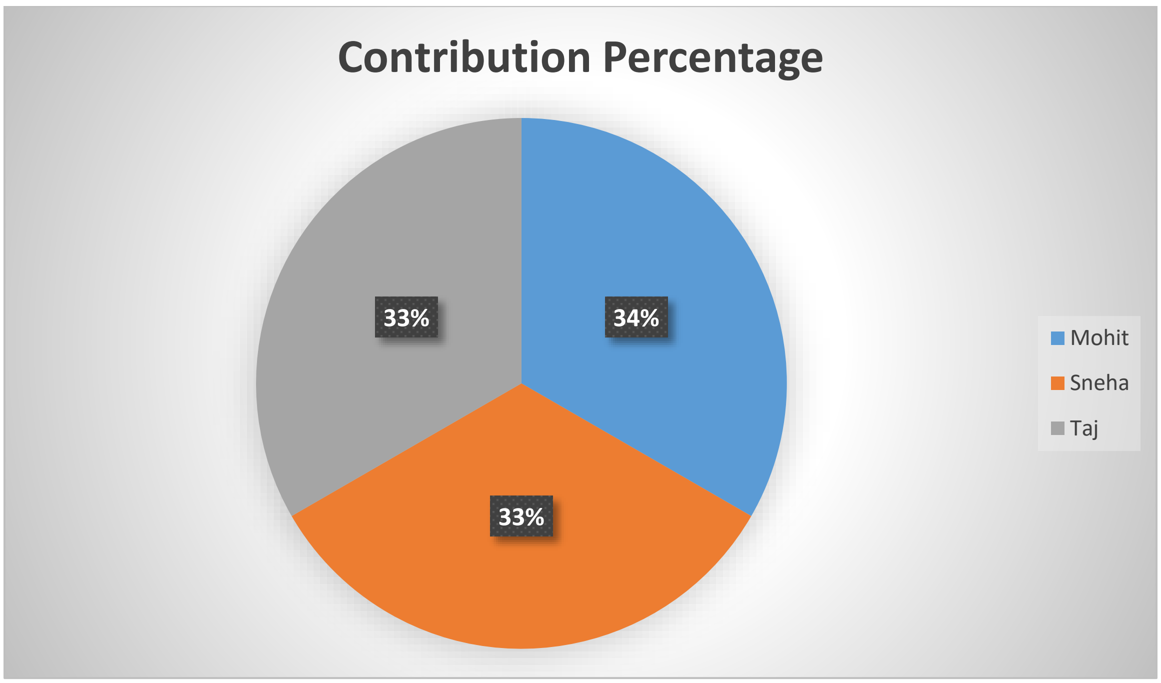
ALGORITHM 4 -KNN

```
install.packages("FNN")
install.packages("caret")
library(caret)
library(FNN)
fit.knn<-knn.reg(train2, test = NULL, train2$ORIGINALINTERESTRATE, k = 3, algorithm=c("kd_tree", "cover_tree", "brute"))
x = fit.knn
set.seed(3333)
knn_fit <- train(ORIGINALINTERESTRATE ~., data = train, method = "knn", tuneLength = 10)

test_pred <- predict(knn_fit, newdata = test)

confusionMatrix(test_pred, test_full$ORIGINALINTERESTRATE )
```

COMPARISON OF MODELS

The RMSE values across different models , it was evident that the backward selection method gave the best set of variables for a highly accurate model with RMSE of 0.2893 compared to the other models:

	RMSE
Linear Regression with all variables	0.3996
Exhaustive and Forward Selection with Linear Regression:	0.3957
Backward Selection	0.2893
Random Forest	0.44
Neural Network	0.35



END OF REPORT