Predicting Missed Payments: Freddie-Mac Fixed-Rate Mortgage Loans

Mitchell Ewing
Thinkful.com
Supervised Learning Capstone

Abstract

- Freddie Mac as a GSE & monopoly
- Wall Street wants in on the profits
- Lower requirements for loan approval
- Housing prices & home equity stagnates
- 2007-2008 Financial Crisis
- Increased transparency
- Investors build more accurate credit performance models

The Data

- Freddie Mac: Single Family Loan-Level Dataset
- Timeframe: 1999-2016
- www.freddiemac.com/research/datasets/sf_loanlevel_dataset.html

Dataset	File Name Format	Contents	File Type	Delimiter
Full	historical_data1_QnYYYY.zip	historical_data1_QnYYYY.txt	Origination Data	Pipe ("|")
		historical_data1_time_QnYYYY.txt	Monthly Performance Data	
Sample	sample_YYYY.zip	sample_orig_YYYY.txt	Origination Data	Pipe ("|")
		Sample_svcg_YYYY.txt	Monthly Performance Data	

Data Wrangling

- Combining Origination & Monthly Performance Files
- Data-types
- Nulls

```
# Merge origination files.
frames_orig = [orig1999, orig2000, orig2001, orig2002, orig2003, orig2004,
               orig2005, orig2006, orig2007, orig2008, orig2009, orig2010,
               orig2011, orig2012, orig2013, orig2014, orig2015, orig2016]
orig_combined = pd.concat(frames_orig)
print(orig_combined.shape)
orig_combined.head()
```

```
(900000, 26)
```

	creditScore	firstPaymentDate	firstTimeHomebuyerFlag	maturityDate	metroArea	miPercentage
0	799	199903	N	202901	37620.0	0
1	635	200212	N	202904	10420.0	0
2	787	199905	N	202904	12060.0	999
3	726	199904	N	202903	28140.0	0
4	748	199905	9	202904	17140.0	999

5 rows × 26 columns

```
# Merge performance files.
frames_perf = [perf1999, perf2000, perf2001, perf2002, perf2003, perf2004,
               perf2005, perf2006, perf2007, perf2008, perf2009, perf2010,
               perf2011, perf2012, perf2013, perf2014, perf2015, perf2016]
perf_combined = pd.concat(frames_perf)
print(perf_combined.shape)
perf_combined.head()
```

```
(42282074, 25)
```

	lsn	monthlyReportingPeriod	currentActualUpb	dlq	loanAge	remainMthsToMaturity
0	F199Q1000012	200208	42058.58	0	42	317
1	F199Q1000012	200209	42011.81	0	43	316
2	F199Q1000012	200210	41964.77	0	44	315
3	F199Q1000012	200211	41917.46	0	45	314
4	F199Q1000012	200212	41869.88	0	46	313

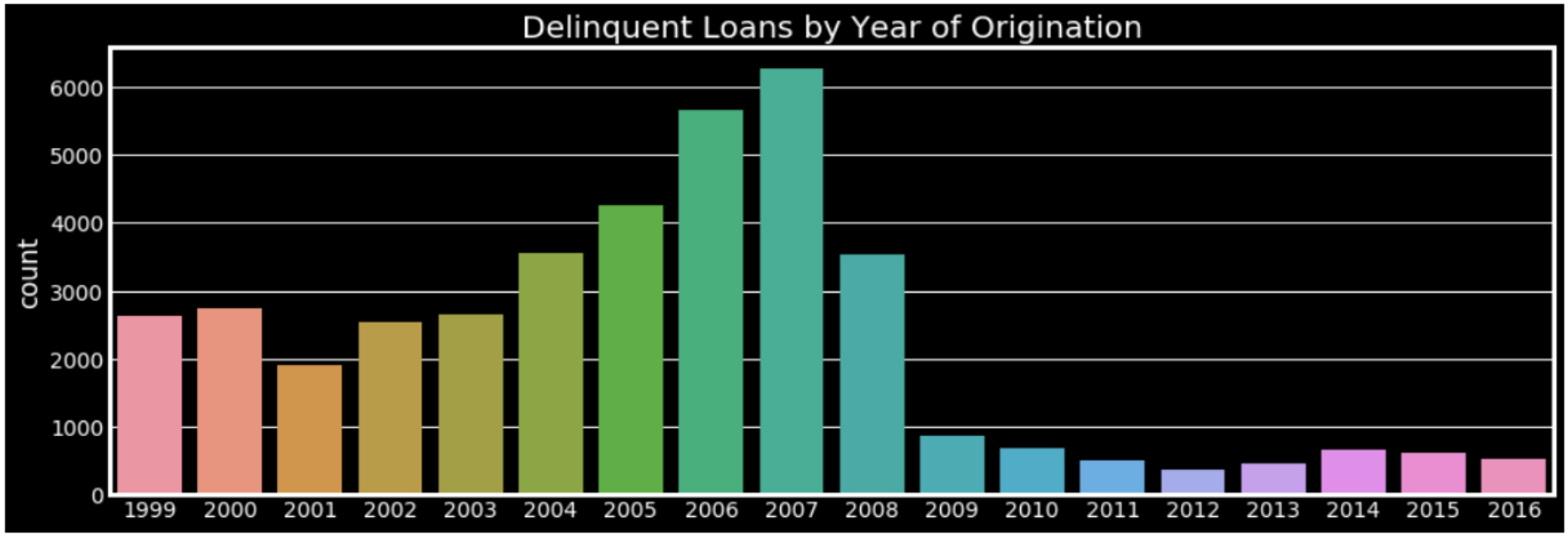
5 rows × 25 columns

Features

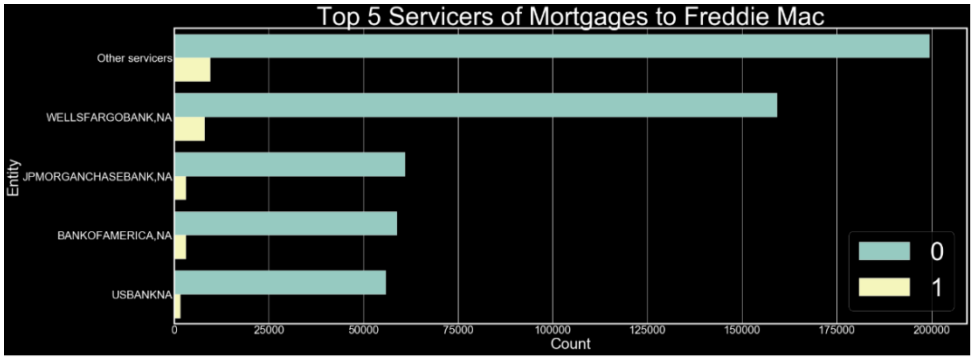
- Numeric:
  - Delinquent (missed payment)
  - Credit Score
  - Interest Rate
  - Loan-to-Value, Combined Loan-to-Value
  - Debt-to-Income
  - Mortgage Insurance %
  - Unpaid Principal Balance

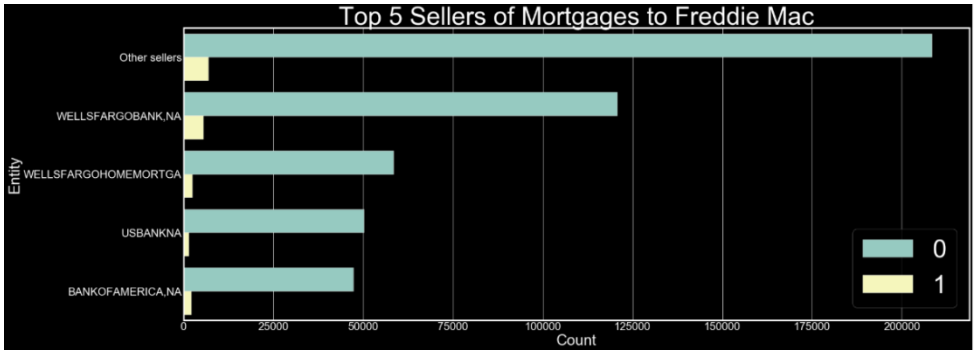
- Categorical:
  - Loan Purpose
  - Number of Units
  - Occupancy Status
  - Number of Borrowers
  - First-Time Homebuyer Flag
  - Super-Conforming Flag
  - Prepayment Penalty Flag
  - Geographical State
  - Property Type
  - Loan Channel
  - Loan Seller, Servicer
  - Term
  - Maturity Date
  - First Payment Date

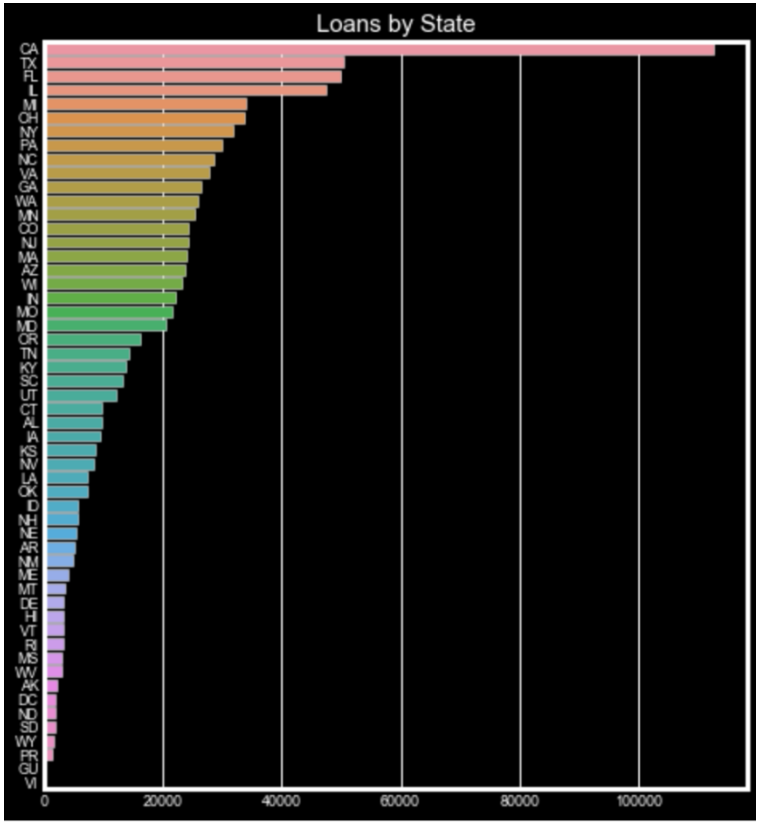
Delinquent Loans



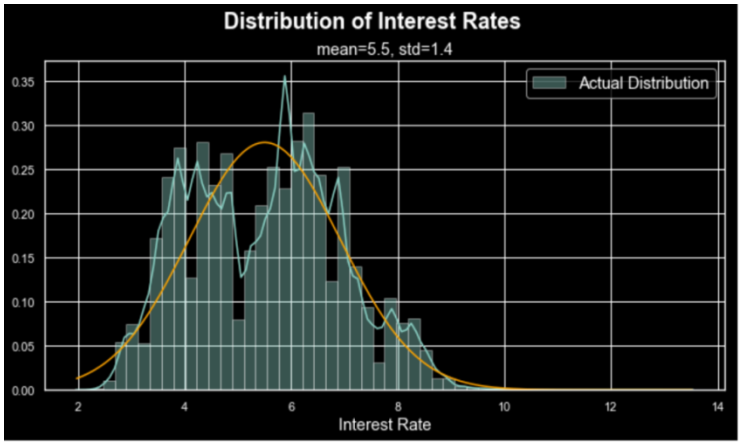
Loans by Seller/Servicer/State

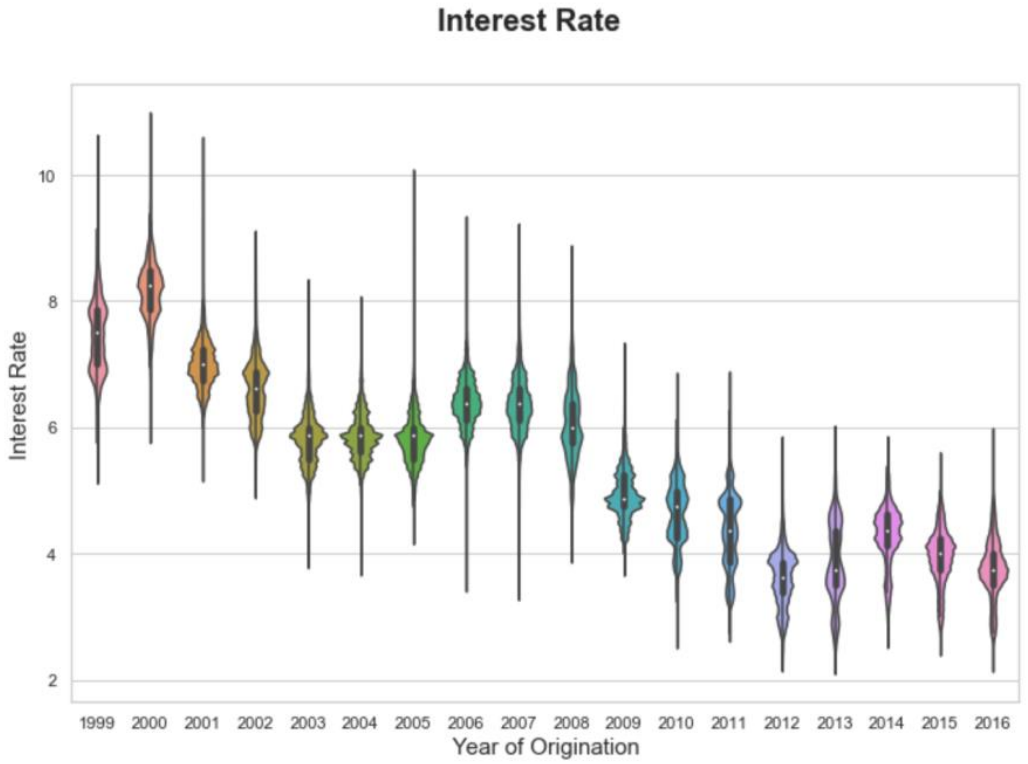




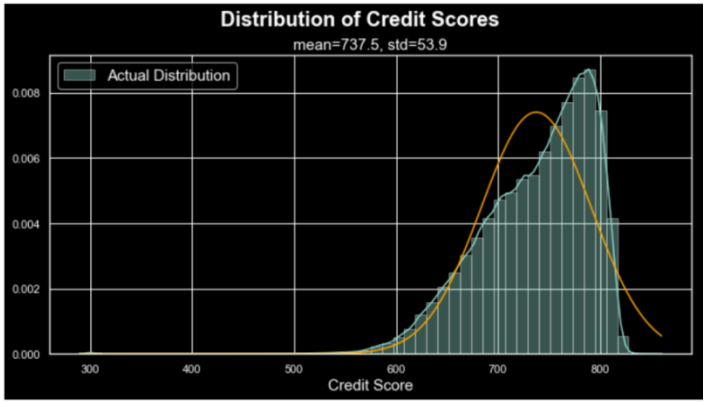


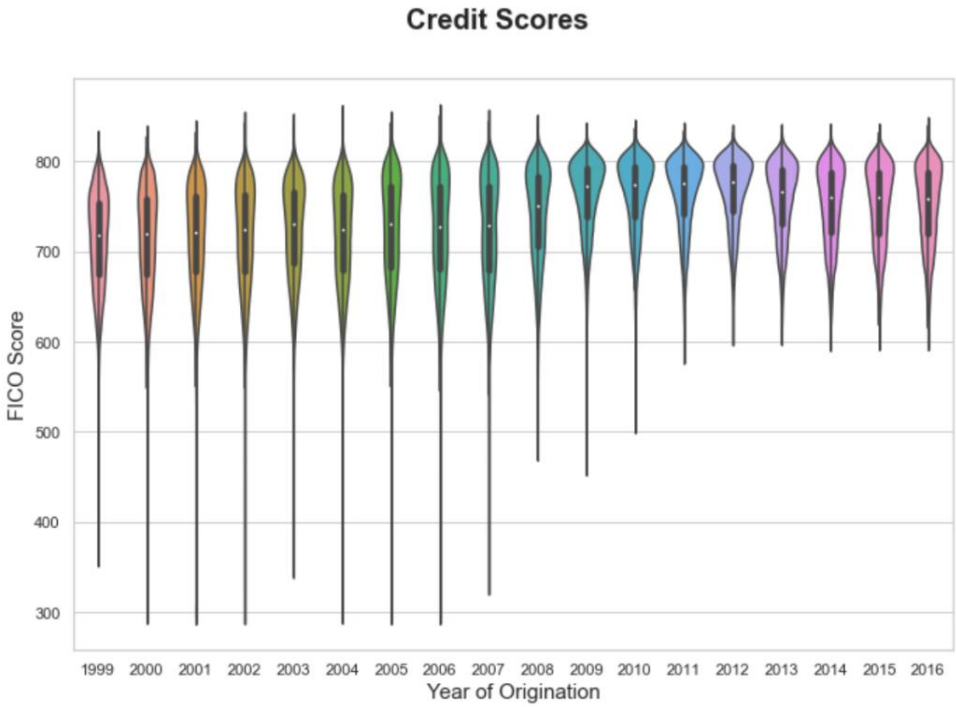
Interest Rates





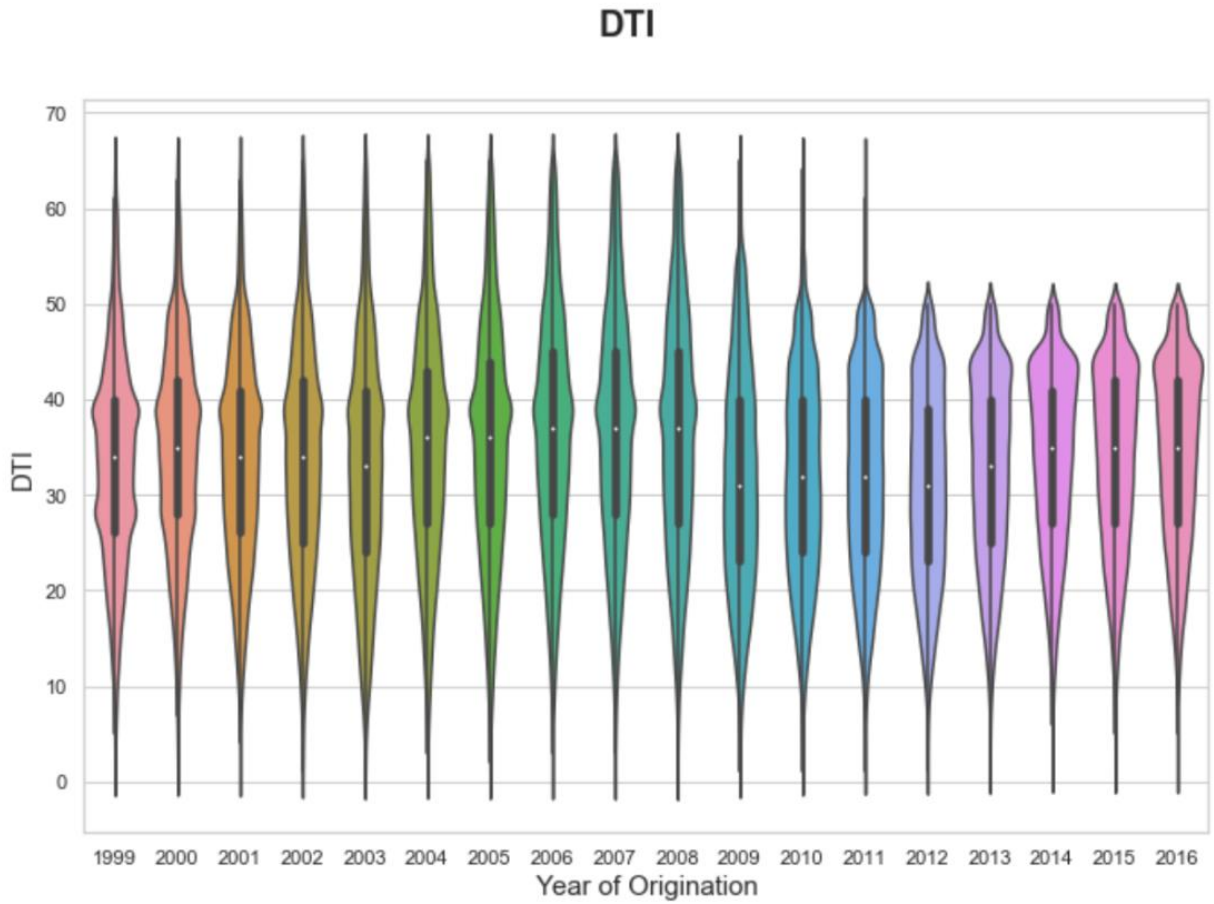
Credit Scores



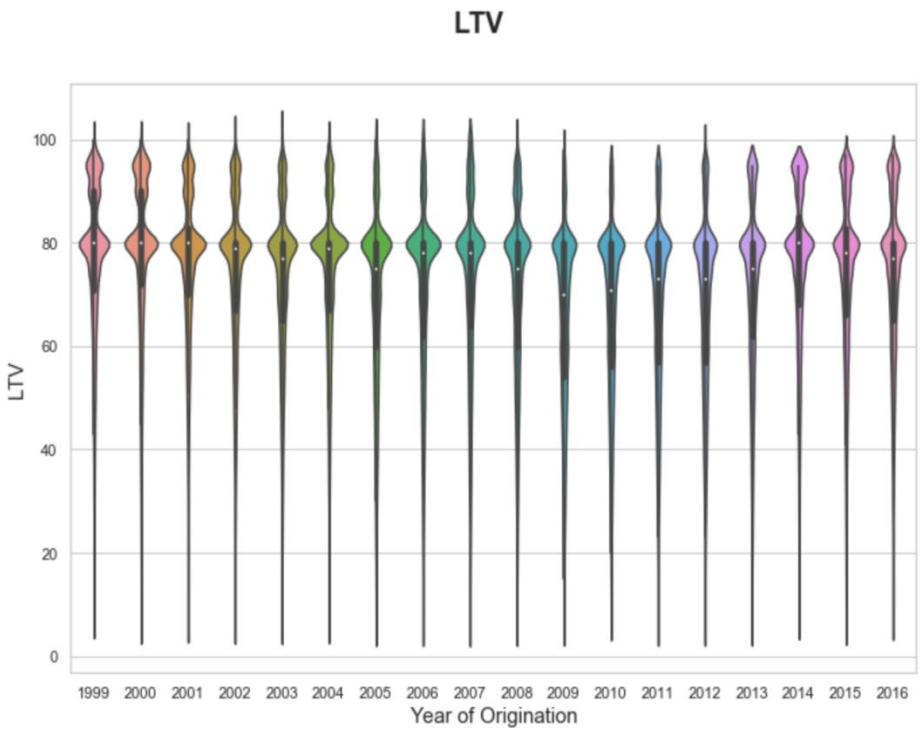


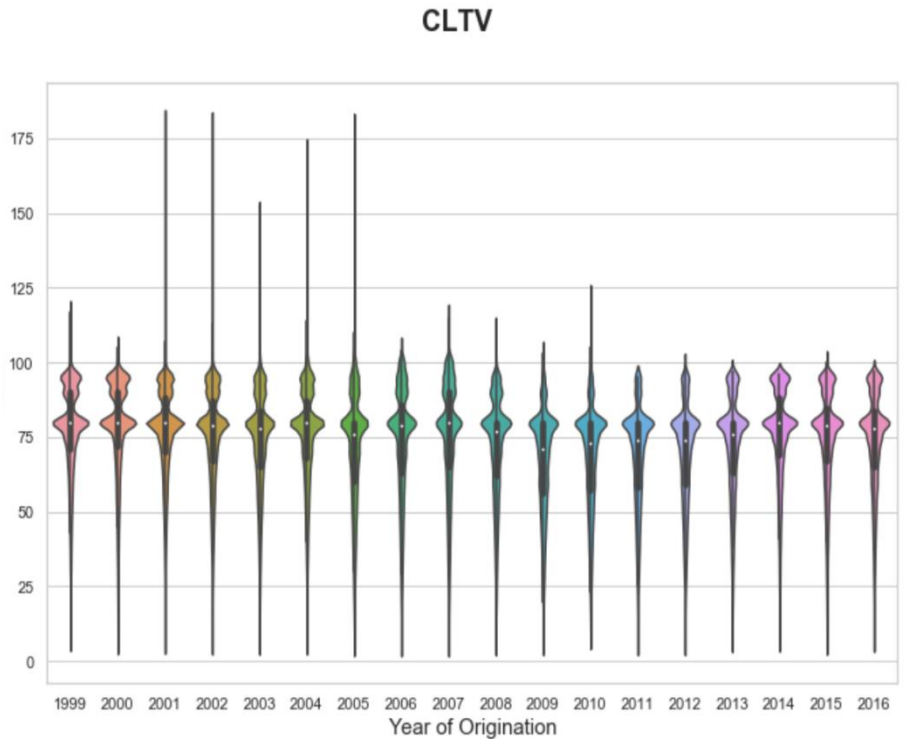
Debt-to-Income

- Higher requirements for loan post-crisis
- Key takeaway: lower debt-to-income trend post-crisis

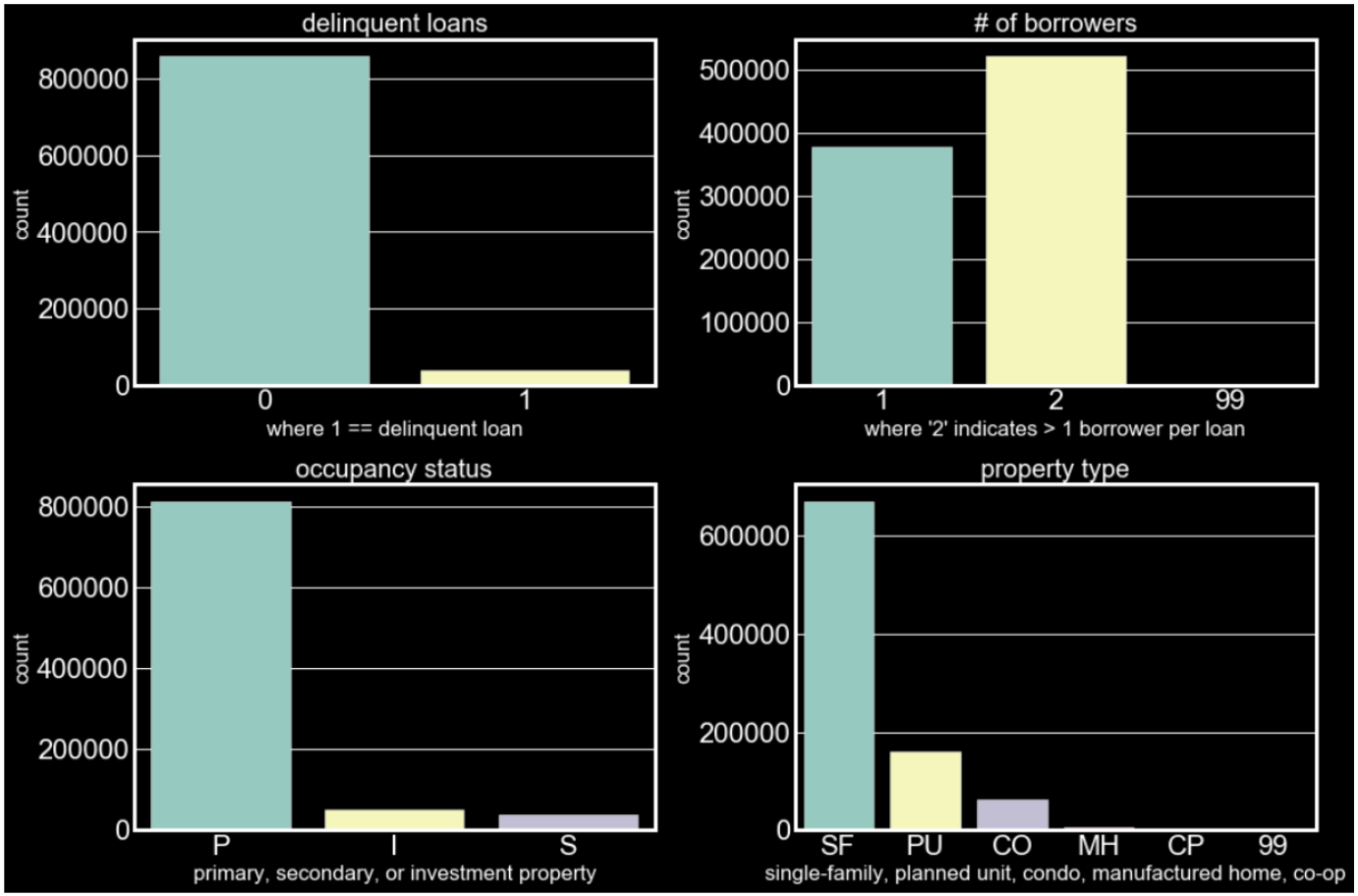


Loan-to-Value, Combined-Loan-to-Value

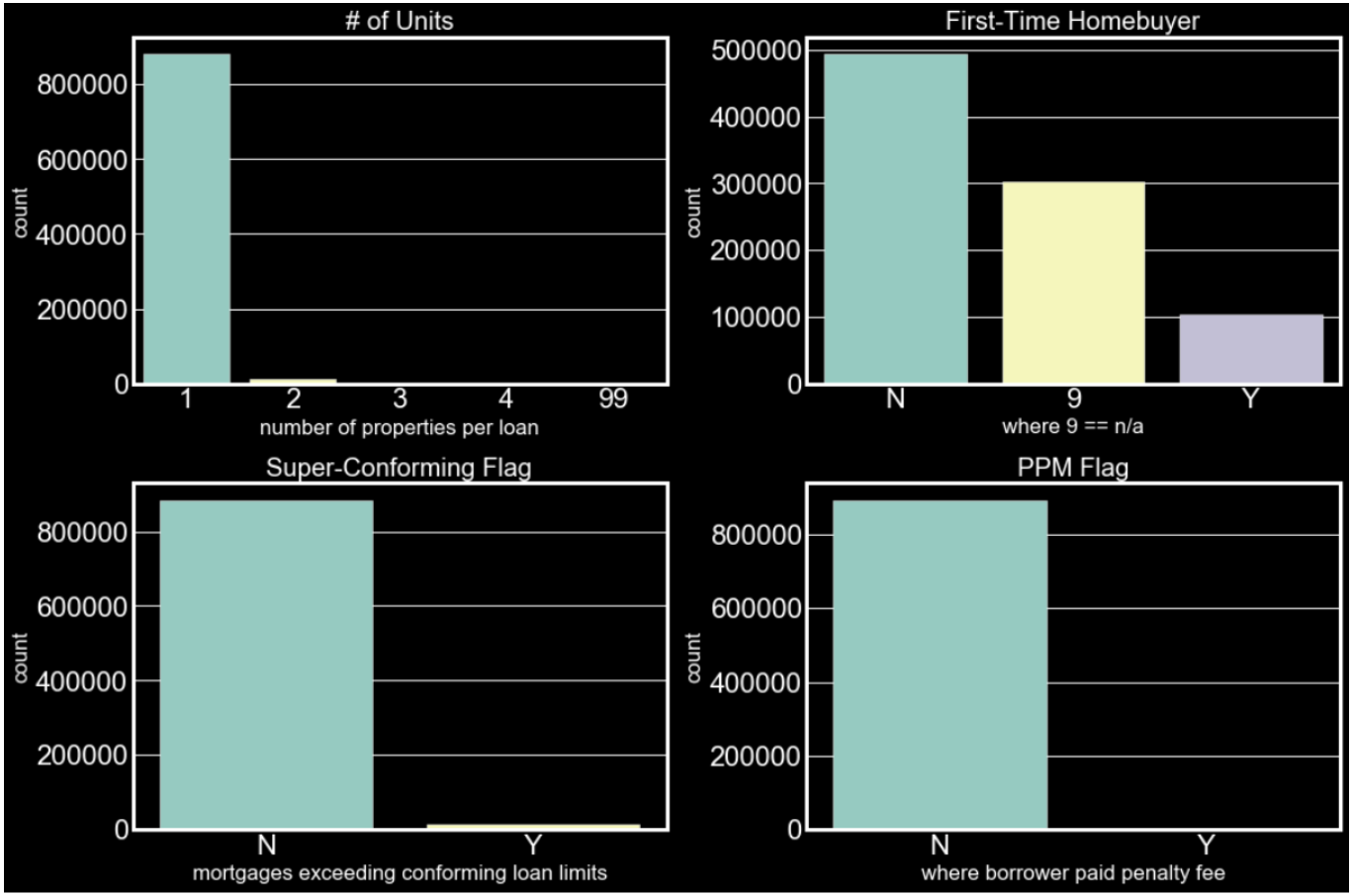




Categorical Features



Categorical Features



Features (continued)

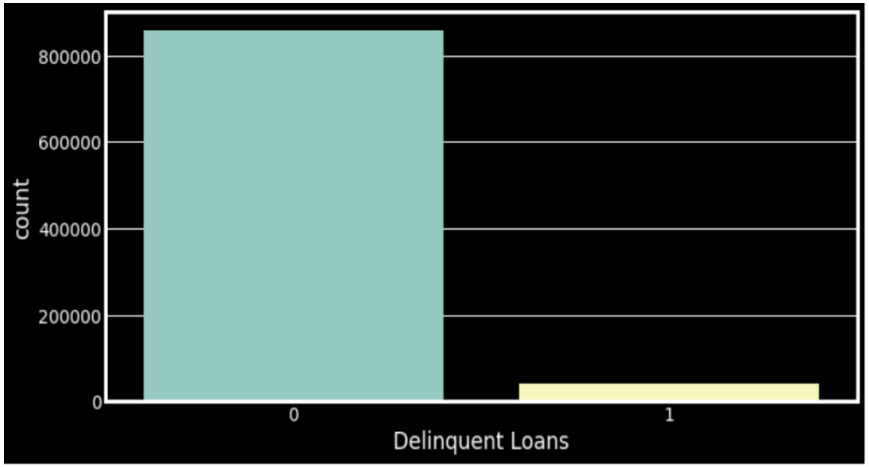
- Creating dummies: Categorical
- Substantial amount of missing/unavailable features
  - Credit Score, Loan-to-Value, Debt-to-Income, Combined-Loan-to-Value, Mortgage Insurance %
  - Example: Credit Score ranges from 350-800, if info unavailable to Freddie Mac, then its value in dataset represented as “9999”
  - Of the 12,000 loans with unavailable DTI, 3500 ended up missing a payment
  - Unavailable thus represented as “xx_DTI”, “xx_credit_score”, etc
- One-Hot-Encoding: # of units, # of borrowers

```
# Load the sklearn package.
from sklearn.preprocessing import OneHotEncoder
```

```
feature_cols
```

```
Index(['credit_score', 'int_rate', 'LTV', 'DTI', 'CLTV', 'UPB', 'MI',
       'xx_credit_score', 'xx_LTV', 'xx_DTI', 'xx_CLTV', 'xx_MI',
       'loan_purpose_C', 'loan_purpose_N', 'loan_purpose_P',
       'occupancy_status_I', 'occupancy_status_P', 'occupancy_status_S',
       'first_home_flag_9', 'first_home_flag_N', 'first_home_flag_Y',
       'super_conform_flag_N', 'super_conform_flag_Y', 'ppm_flag_N',
       'ppm_flag_Y', 'prop_type_99', 'prop_type_CO', 'prop_type_CP',
       'prop_type_MH', 'prop_type_PU', 'prop_type_SF', 'channel_B',
       'channel_C', 'channel_R', 'channel_T', 'units_1', 'units_2', 'units_3',
       'units_4', 'borrow_1', 'borrow_2', 'borrow_99'],
      dtype='object')
```

Class Imbalance




```
# Review the imbalance.
print('What % of loans end up in default?\n',
      scaled_df['dlq'].value_counts(normalize=True))
print('\nNumerical count:\n', scaled_df['dlq'].value_counts())
```

```
What % of loans end up in default?
 0    0.955078
1    0.044922
Name: dlq, dtype: float64

Numerical count:
 0    859570
1     40430
Name: dlq, dtype: int64
```

- Random under-sampling

```
print('Where 0 == non-delinquent')
print('and 1 == delinquent:\n')
print('The number of indexes within the model_df is:')
print(sample_model_df['dlq'].value_counts())
```

```
Where 0 == non-delinquent
and 1 == delinquent:

The number of indexes within the model_df is:
1    40430
0    40430
Name: dlq, dtype: int64
```

Model DataFrame

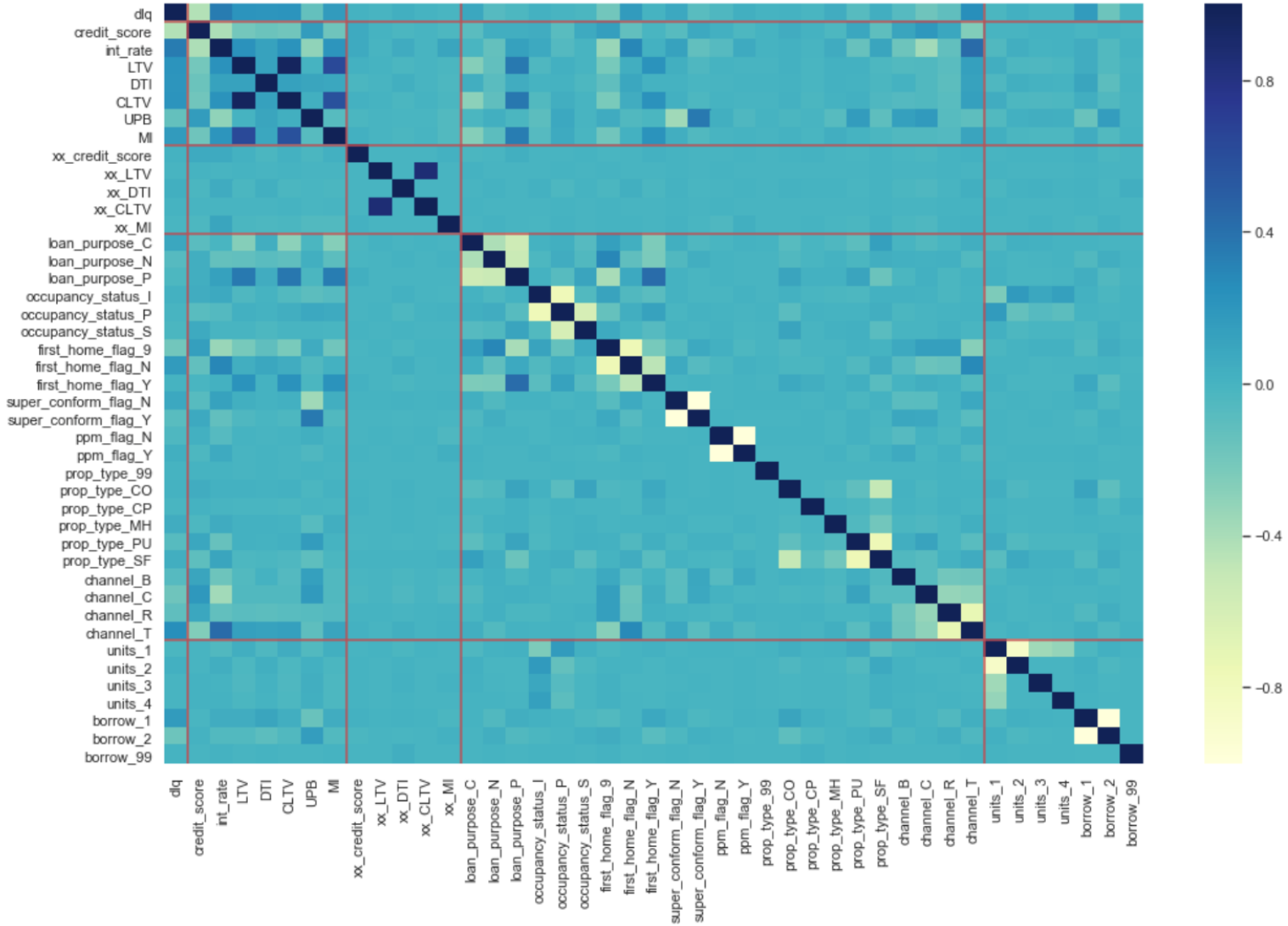
- Target Variable -> dlq
- dlq == borrower missed at least 1 payment
- This does NOT imply a full default

- 42 Features
- 80,860 loans

	dlq	credit_score	int_rate	LTV	DTI	CLTV	UPB	MI	xx_credit_
48109	0	0.807104	0.349784	0.127492	-1.960438	0.062194	0.155798	-0.479279	
30732	1	-1.642736	0.349784	1.373676	1.456013	1.291912	-0.061386	2.407046	
25837	1	-2.292315	0.965371	0.364860	-0.342119	0.296426	-0.354113	-0.479279	
37217	0	0.602951	0.613607	-0.643956	0.377134	-0.699061	1.033976	-0.479279	
30772	1	0.640070	0.701548	0.186834	1.635827	0.120752	0.391867	-0.479279	

...	channel_C	channel_R	channel_T	units_1	units_2	units_3	units_4	borrow_1	borrow_2	borrc
...	0	0	1	1.0	0.0	0.0	0.0	1.0	0.0	
...	0	1	0	1.0	0.0	0.0	0.0	0.0	1.0	
...	0	0	1	1.0	0.0	0.0	0.0	0.0	1.0	
...	0	1	0	1.0	0.0	0.0	0.0	0.0	1.0	
...	1	0	0	1.0	0.0	0.0	0.0	1.0	0.0	

Visualized Correlations



Feature Correlations:

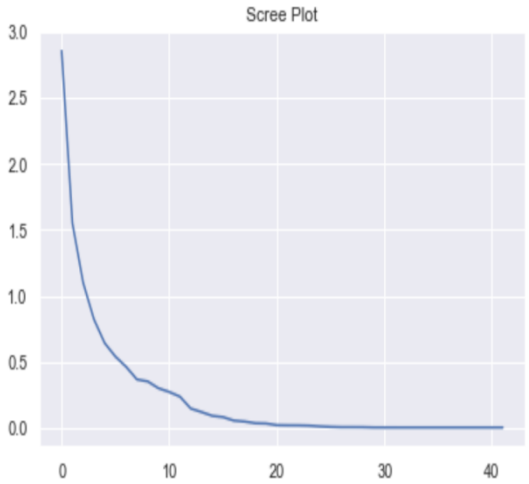
Positive Correlation:

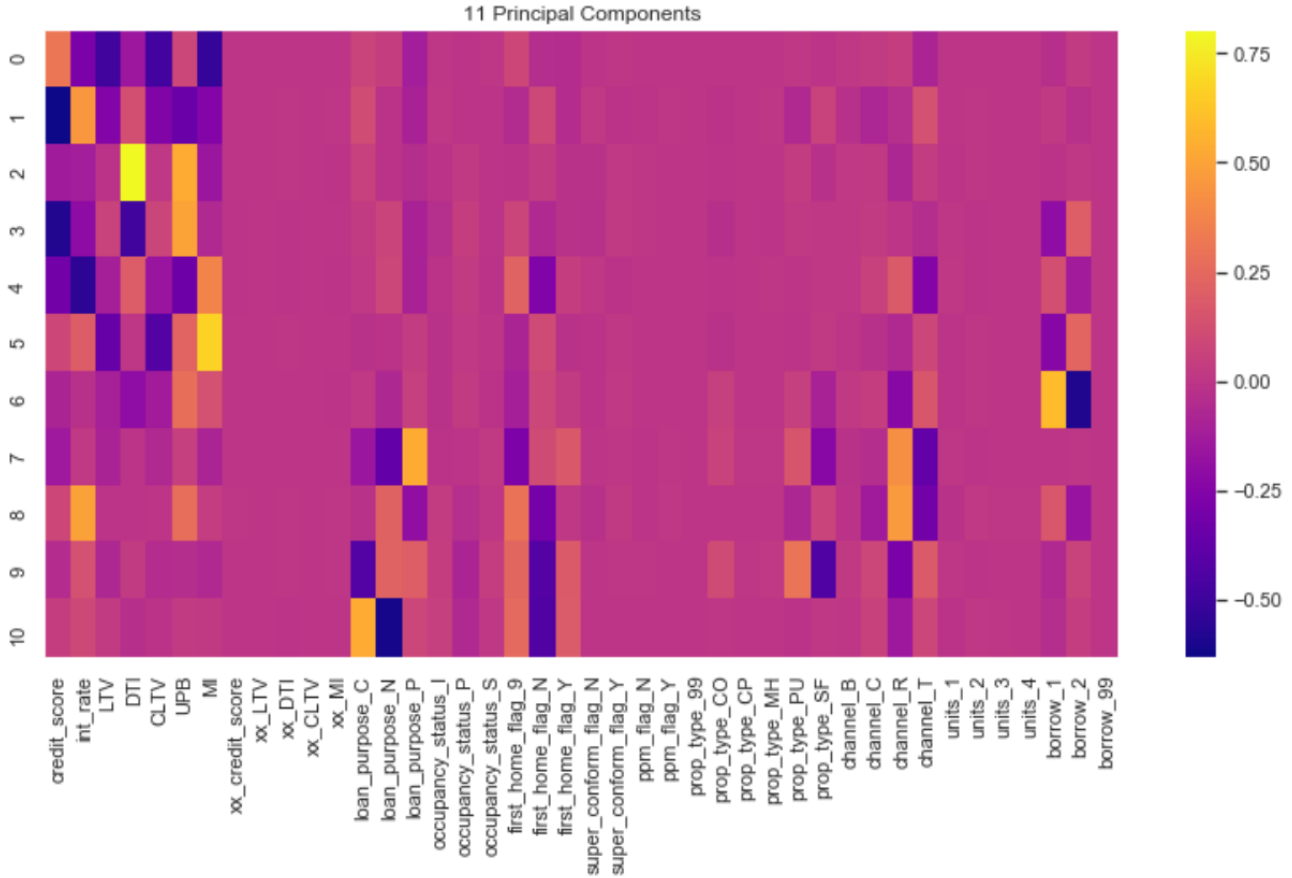
- Interest Rate
- TPO Channel
- Combined LTV
- LTV
- Debt-to-Income
- Returning Home Buyer
- 1 Borrower on the Loan

Negative Correlation:

- Credit Score
- Unknown whether First-Time-Homebuyer
- Correspondent Channel
- 2 Borrowers on the Loan
- Unpaid Principal Balance
- Retail Channel
- Super Conforming Loan == Yes

PCA


Scree Plot
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0
10
20
30
40


11 Principal Components
0
1
2
3
4
5
6
7
8
9
10
credit_score
int_rate
LTV
DTI
CLTV
UPB
MI
xx_credit_score
xx_LTV
xx_DTI
xx_CLTV
xx_MI
loan_purpose_C
loan_purpose_N
loan_purpose_P
occupancy_status_I
occupancy_status_P
occupancy_status_S
first_home_flag_9
first_home_flag_N
first_home_flag_Y
super_conform_flag_N
super_conform_flag_Y
ppm_flag_N
ppm_flag_Y
prop_type_99
prop_type_CO
prop_type_CP
prop_type_MH
prop_type_PU
prop_type_SF
channel_B
channel_C
channel_R
channel_T
units_1
units_2
units_3
units_4
borrow_1
borrow_2
borrow_99
0.75
0.50
0.25
0.00
−0.25
−0.50

Results

- By F1 score, SVC best performing model
- Gradient Boost highest r-squared

Rank:	Model:	R-Squared:	CV Score:	CV StdDev:	Sensitivity:	Specificity:	F1 Score:	Time (sec):
1	Support Vector Classifier	0.76	0.75	0.02	81.5	69.6	0.77	2987.9
2	Gradient Boost	0.76	0.76	0.02	79.4	72.7	0.77	225.2
3	Logistic Regression	0.75	0.75	0.02	76.0	73.2	0.75	9.4
4	Lasso Regression	0.75	0.75	0.02	76.0	73.2	0.75	59.3
5	Ridge Regression	0.75	0.75	0.02	76.0	73.1	0.75	7.1
6	K-Nearest Neighbors	0.73	0.73	0.02	70.1	75.1	0.72	128.8
7	Random Forest Classifier	0.72	0.71	0.02	69.3	74.8	0.71	9.3

Future Extensions

- Time Series Analysis
- Regression: $ amount for delinquent borrowers
- Number of payments to reach full default
- Scenario Analysis
- Home Affordable Refinance Program (HARP) dataset

Questions/Comments/Critiques