



FALL 2017 WAF DATA CHALLENGE

Eric Zeng

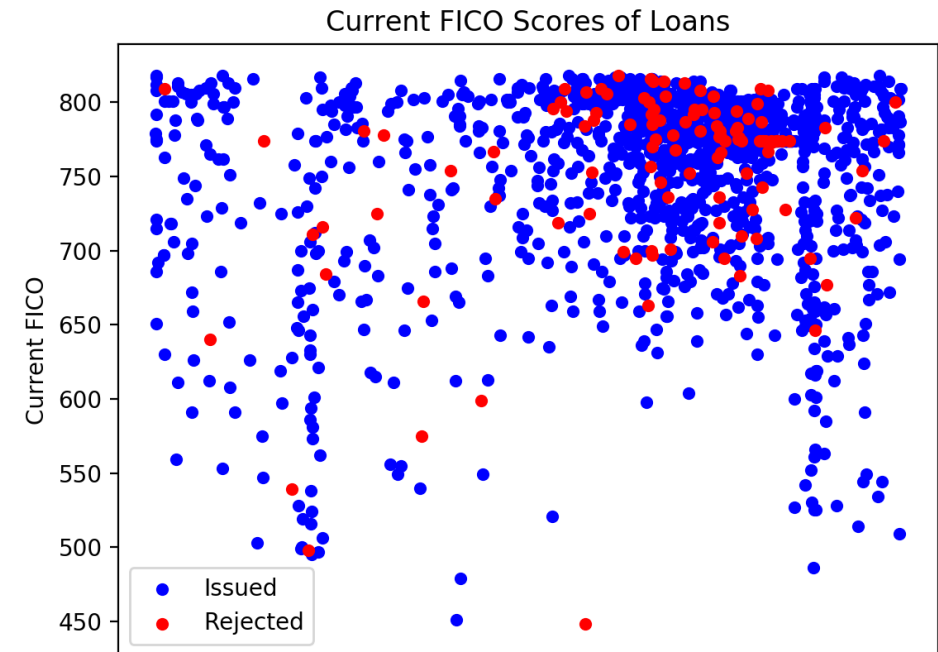
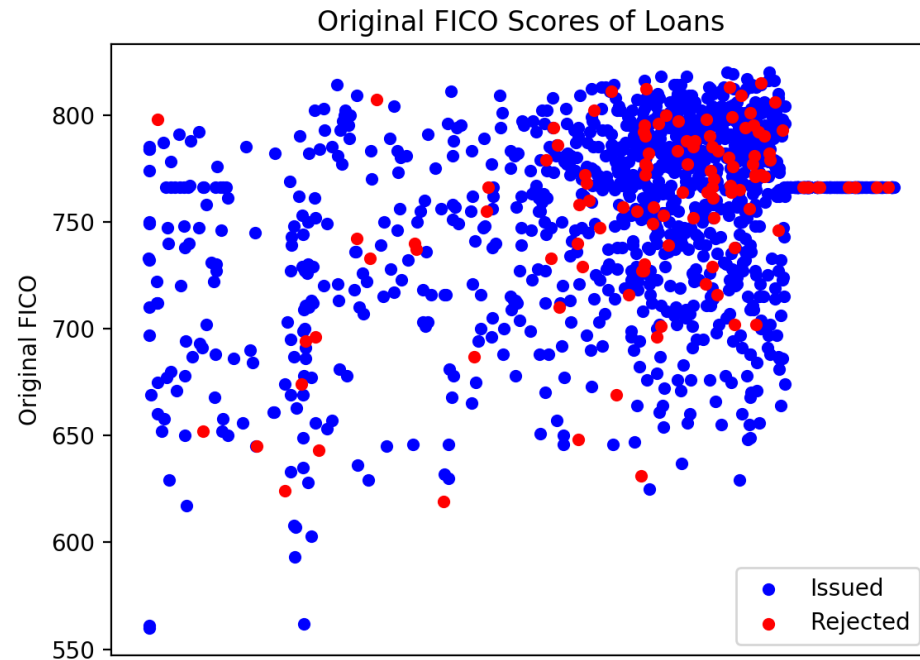


Recommendation

Do not take LoanID 73622

- Low employment/High unemployment
- High crime rate
- High delinquency rate
- High percentage of free cash on payments
- Pattern of rejection in neighborhood
- Zip in question: 10466

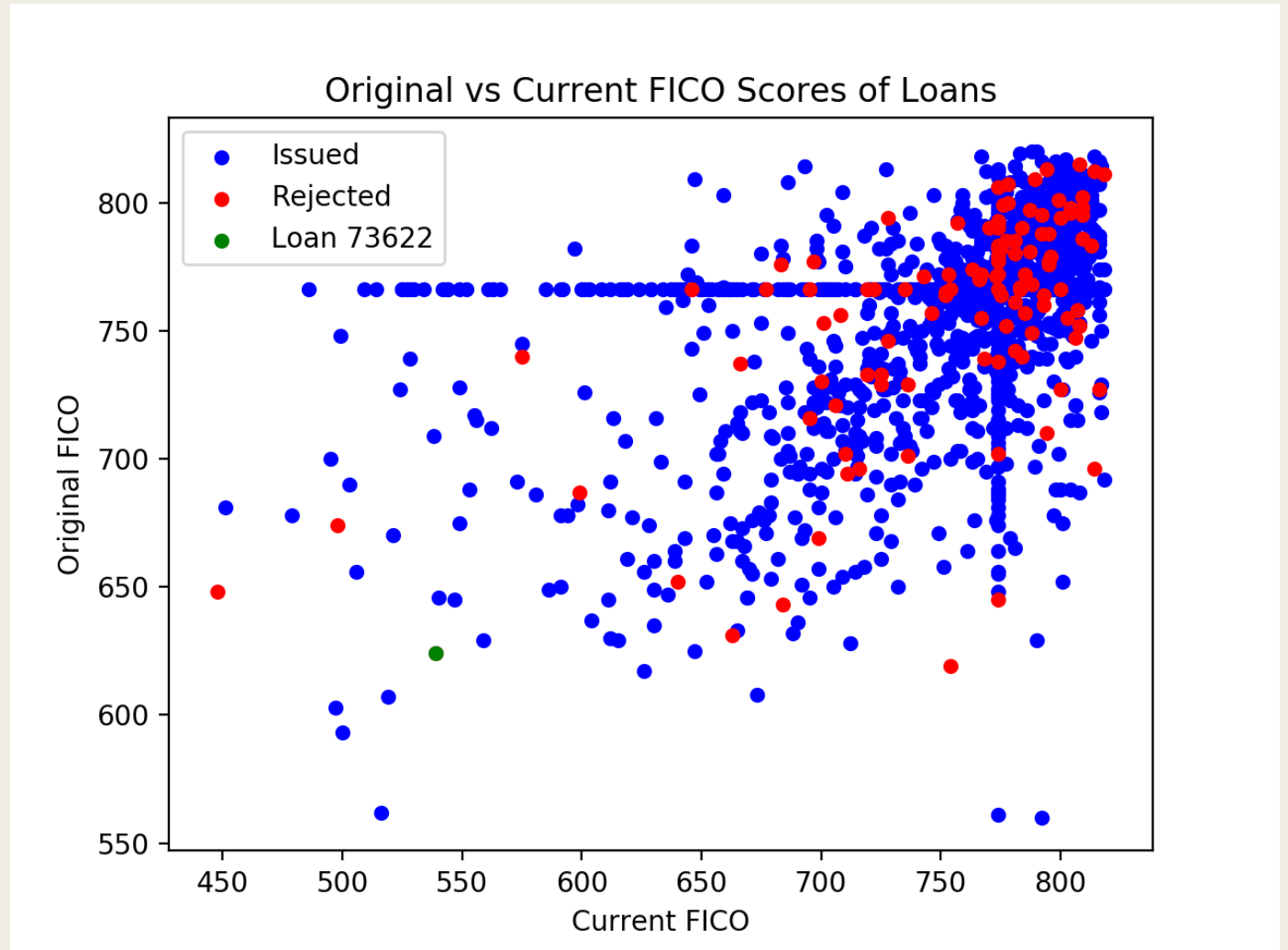
FICO Scores?



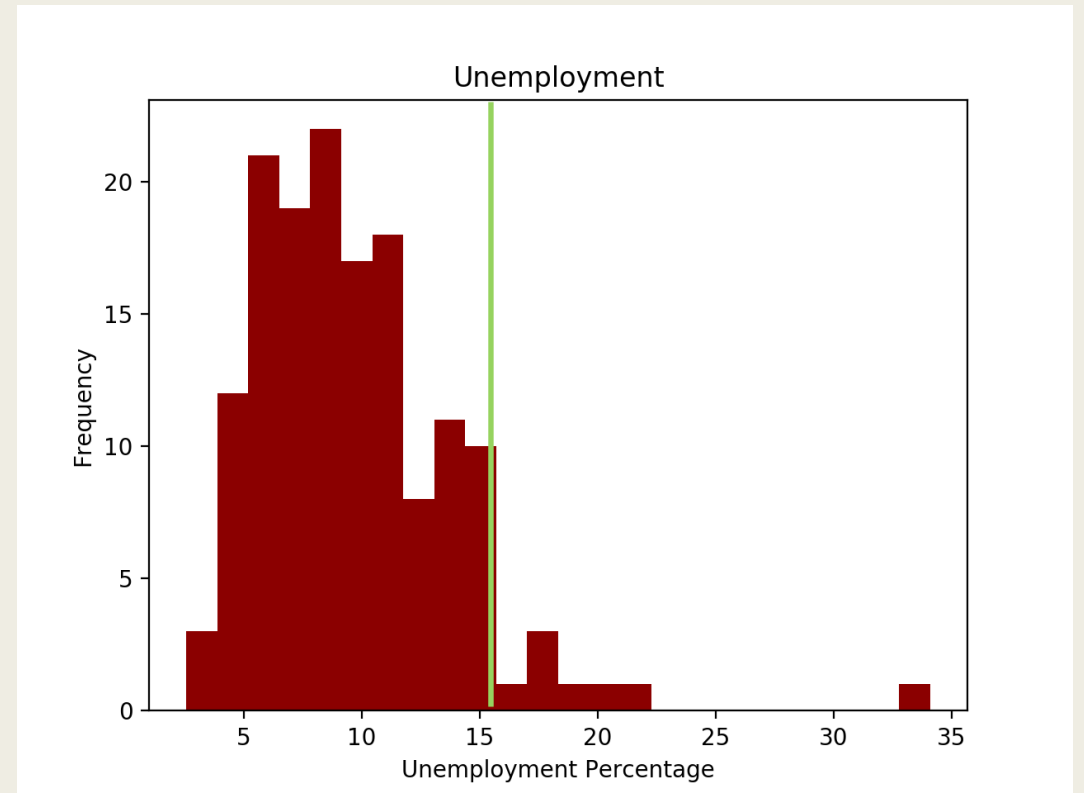
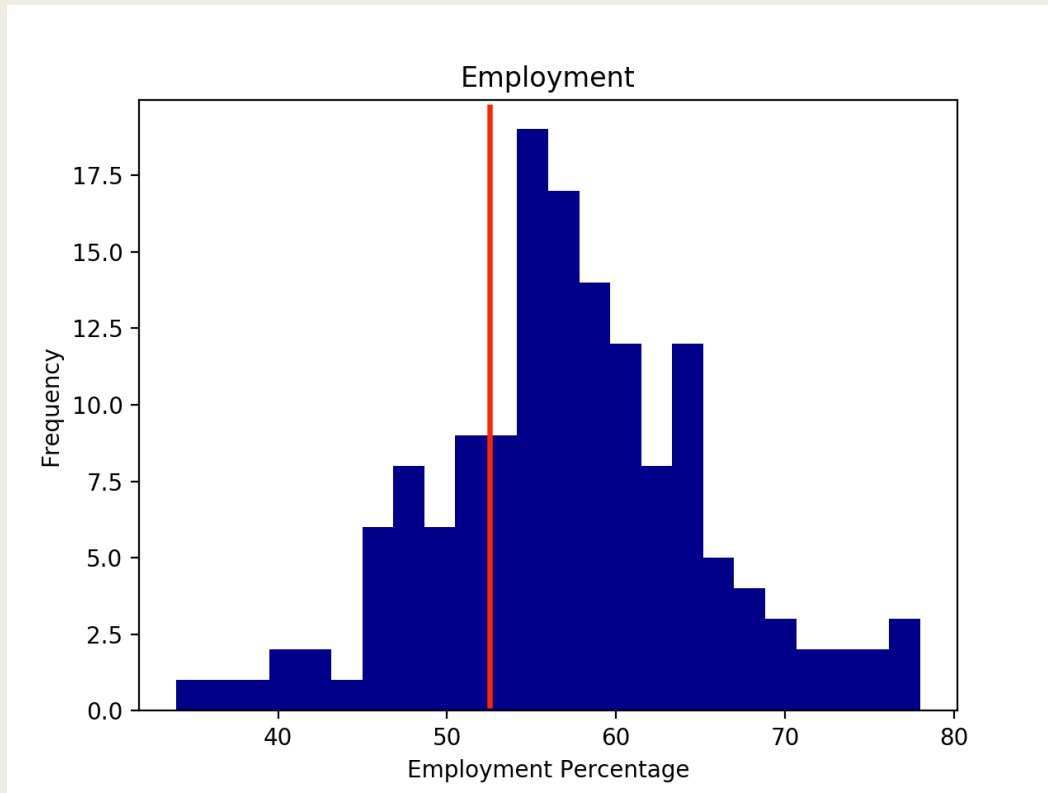
- FICO scores seem to measure credit
- But no correlation between issuing and FICO score

FICO Scores?

- Change in FICO score
 - Separated by Issued/Rejected
- Once again, no correlation
- 73622's FICO decrease is NOT out of place...
- 'Lines' maybe due to bad data?
- Better metrics needed!

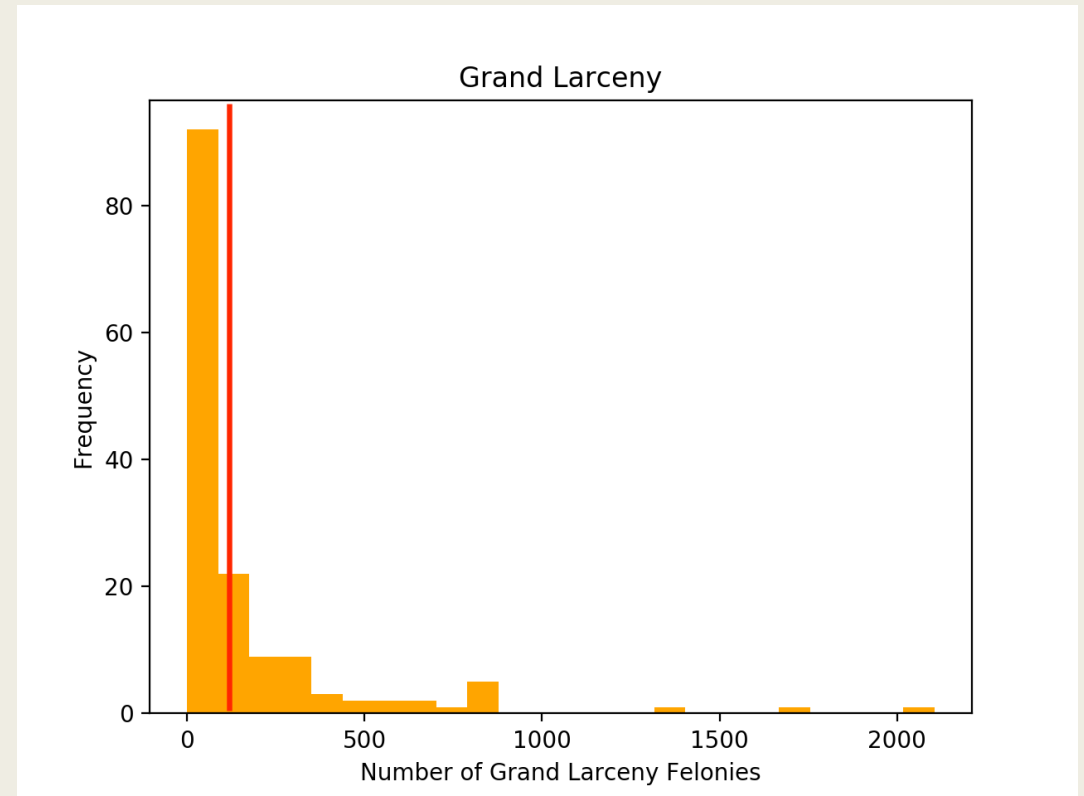
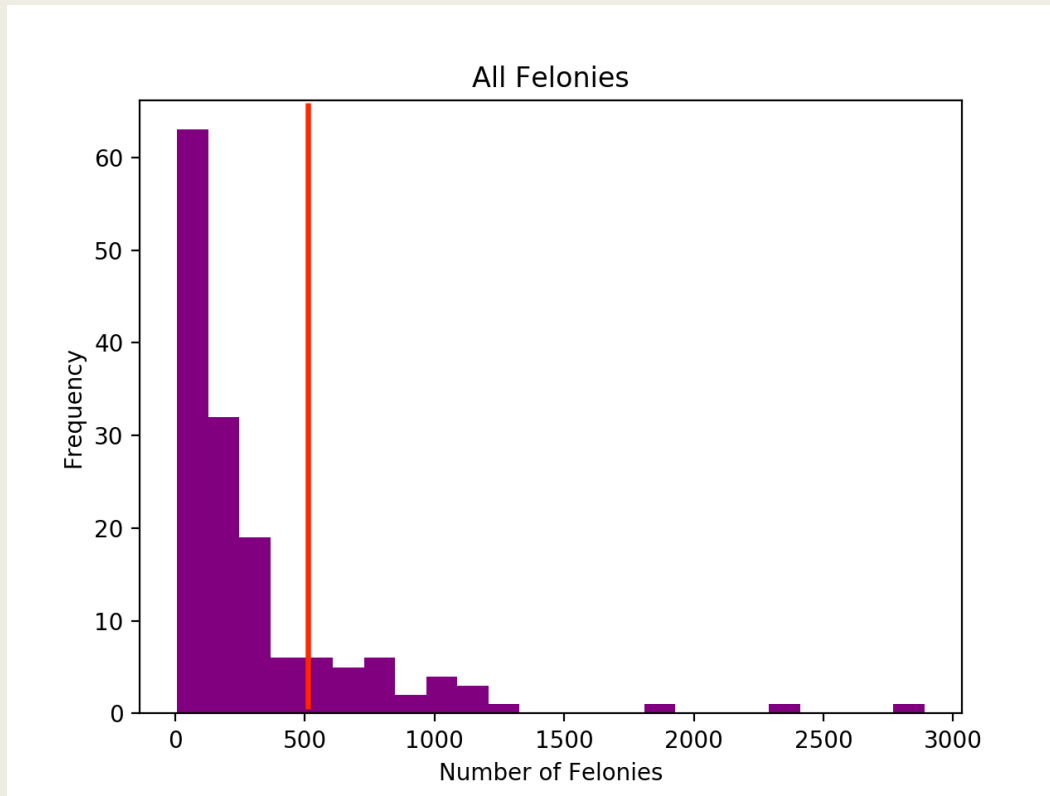


Employment / Unemployment



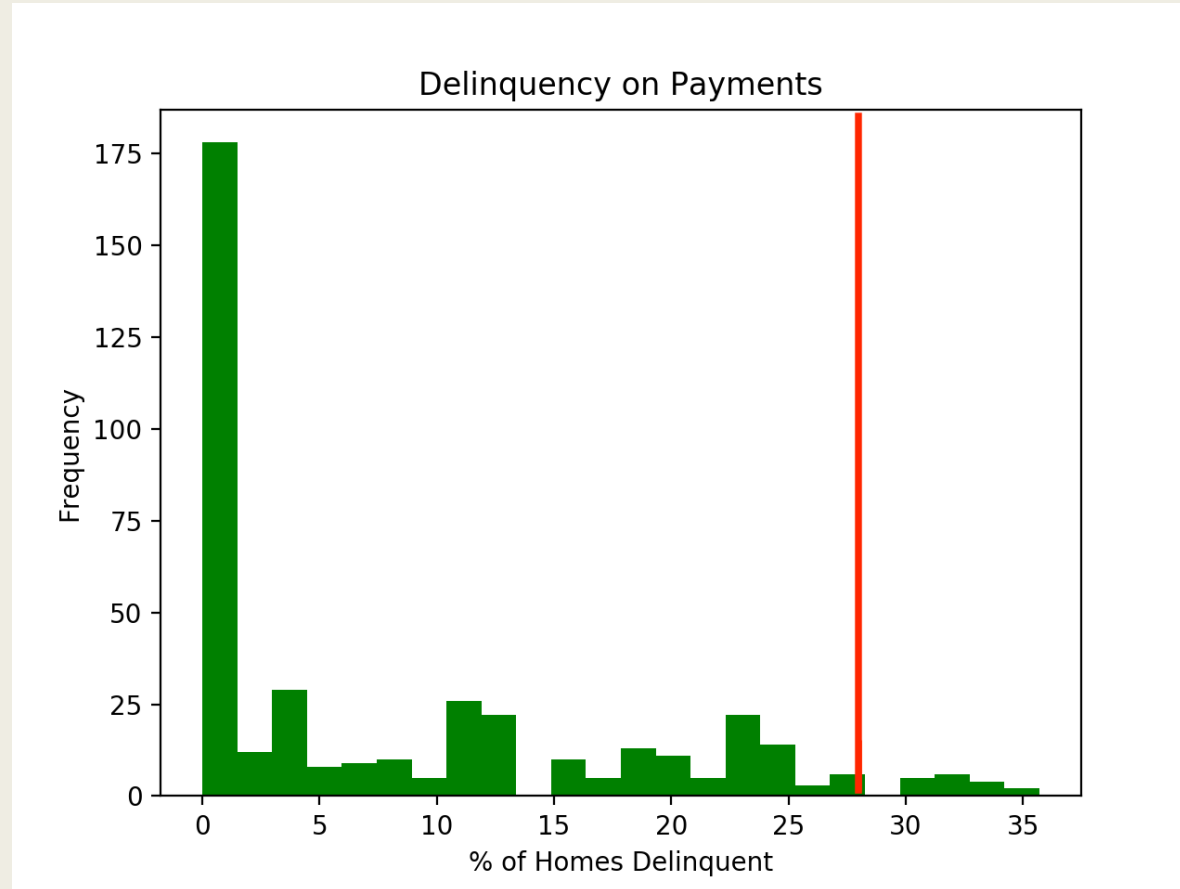
- Employment: 52.2% | ~50 percentile
- Unemployment: 15.2% | >75 percentile
- Large unemployment may be a sign of economic stress

Crime (All Felonies and Grand Larceny)



- All Felonies: 503 | >75 percentile
- Grand Larceny: 135 | ~70 percentile
- Theft crimes lead to more property damage/financial burden

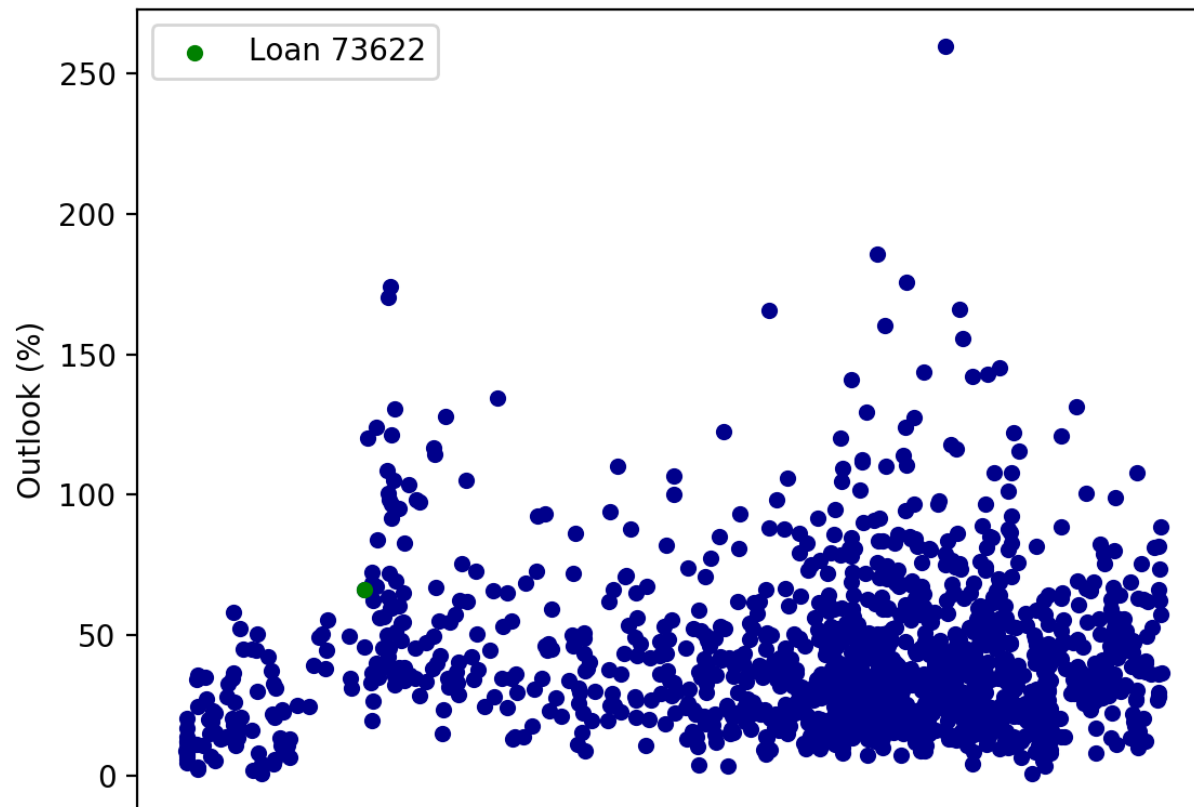
Delinquency on Payments



- Late payment on monthly dues
- 27.7%
 - >75% percentile
- Exposure to risk of default

Percentage of Free Cash on Payments

Percent of Monthly Free Cash Spent on Loan Payment (Lower is better)



- Free cash found using median income and expenditure data by ZIP
- Loan's monthly payment based on balance, APR, and term length
- 73622 is at 66.2%
 - About 2/3 of median free cash will be spent on payments
 - Relatively high
 - >75 percentile

Concluding Thoughts

- Due to the previous metrics
 - *73622 is risky*
 - *FICO scores do not reflect real risk*
 - *Concerns regarding unemployment, crime, delinquency, and median free cash*
 - *Pattern of rejection in Eastchester neighborhood: 359878 and 312537 rejected*

Resources

- Python/Scipy
- Pandas documentation
- Investopedia
- Time
- Monthly payment formula:

$$P = \frac{A}{[(1 + i)^n - 1] / [n(1 + i)^n]}$$

- P payment, A balance, i rate, n terms

```
loans.py x
130 plt.title('Delinquency on Payments')
131 plt.xlabel('% of Homes Delinquent')
132
133
134 def pay(payments, incomes):
135     # determine monthly payment needed at given APR, current balance, and term
136     monthlyPayments = pd.DataFrame({'Loan ID': payments['Loan ID'],
137                                     'ZIP': payments['ZIP'],
138                                     'Monthly Due': payments['Balance'] /
139                                     (((1 + payments['APR'] / 12)
140                                      ** payments['Term Length']) - 1) /
141                                     (payments['APR'] / 12 *
142                                      (1 + payments['APR'] / 12) **
143                                      payments['Term Length']))})
144
145     monthlyPayments = monthlyPayments[monthlyPayments['Loan ID'] != 349324]
146
147     # Series containing matching median FC in each loan's ZIP
148     incomesSorted = pd.Series(name='FC')
149     for i in monthlyPayments.iterrows():
150         incomesSorted.set_value(i[0], incomes.loc[
151             incomes['PROP_ZIP'] ==
152             int(i[1]['ZIP']))['Free Cash'].values[0])
153
154     # add to monthlypayments, divide incomesSorted by 12 to get Monthly FC
155     monthlyPayments['Monthly FC'] = incomesSorted / 12
156     # Outlook: percentage of monthly FC used to pay loan
157     monthlyPayments['Outlook'] = 100 * monthlyPayments[
158         'Monthly Due'] / monthlyPayments['Monthly FC']
159
160     # describe and plots
161     print(monthlyPayments.describe())
162     print(monthlyPayments.loc[monthlyPayments['Loan ID'] == 73622])
163
164     all = monthlyPayments.plot()
```