



PREDICTING LOAN DEFAULTS USING LOGISTIC REGRESSION

Sonja Tilly

November 17

Approach and method

- Project
- Dataset
- Data exploration
- Feature engineering
- Classification
 - Building basic logistic regression model
 - Optimising model parameters
- Results
- Conclusion
- Appendix

Project

- Predict customers' creditworthiness
(i.e. whether they will complete or default on a loan)
- Build and test a logistic regression model
- Use 70% of the data to build the model and then test the model using the remaining 30%

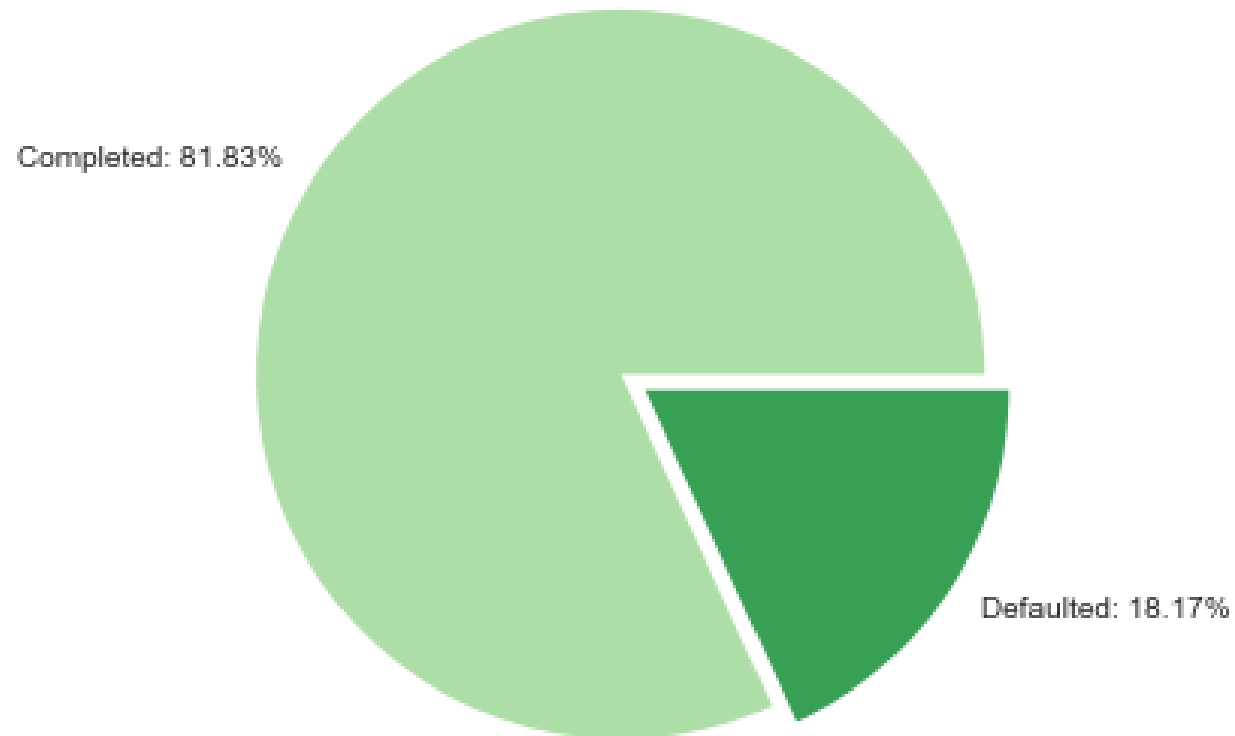
Lending Club dataset

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 24776 entries, 0 to 24775
Data columns (total 21 columns):
Loan Amount                24776 non-null int64
Loan Term                  24776 non-null object
Employment Length          24776 non-null float64
Home Ownership             24776 non-null object
Annual Income              24776 non-null float64
Loan Purpose               24776 non-null object
Address State              24775 non-null object
Debt To Income Ratio       24775 non-null float64
No. Delinquencies In Last 2 Years 24775 non-null object
Earliest Credit Line Opened 24776 non-null float64
FICO Credit Score          24775 non-null float64
No. Inquiries In Last 6 Months 24775 non-null float64
Months Since Last Delinquency 24776 non-null int64
No. Of Credit Lines        24775 non-null float64
No. Adverse Public Records 24775 non-null object
Total Credit Balance       24775 non-null float64
Use Of Credit Line         24724 non-null float64
Total Number Of Credit Lines 24775 non-null float64
Loan Application Description 24776 non-null int64
No. Of Public Record Bankruptcies 24776 non-null object
Class                      24776 non-null object
dtypes: float64(10), int64(3), object(8)
memory usage: 4.0+ MB
```

The data contains rows of customers, with each column showing the features for loan applications that have been approved, together with outcomes of the loans (in the final column 'Class').

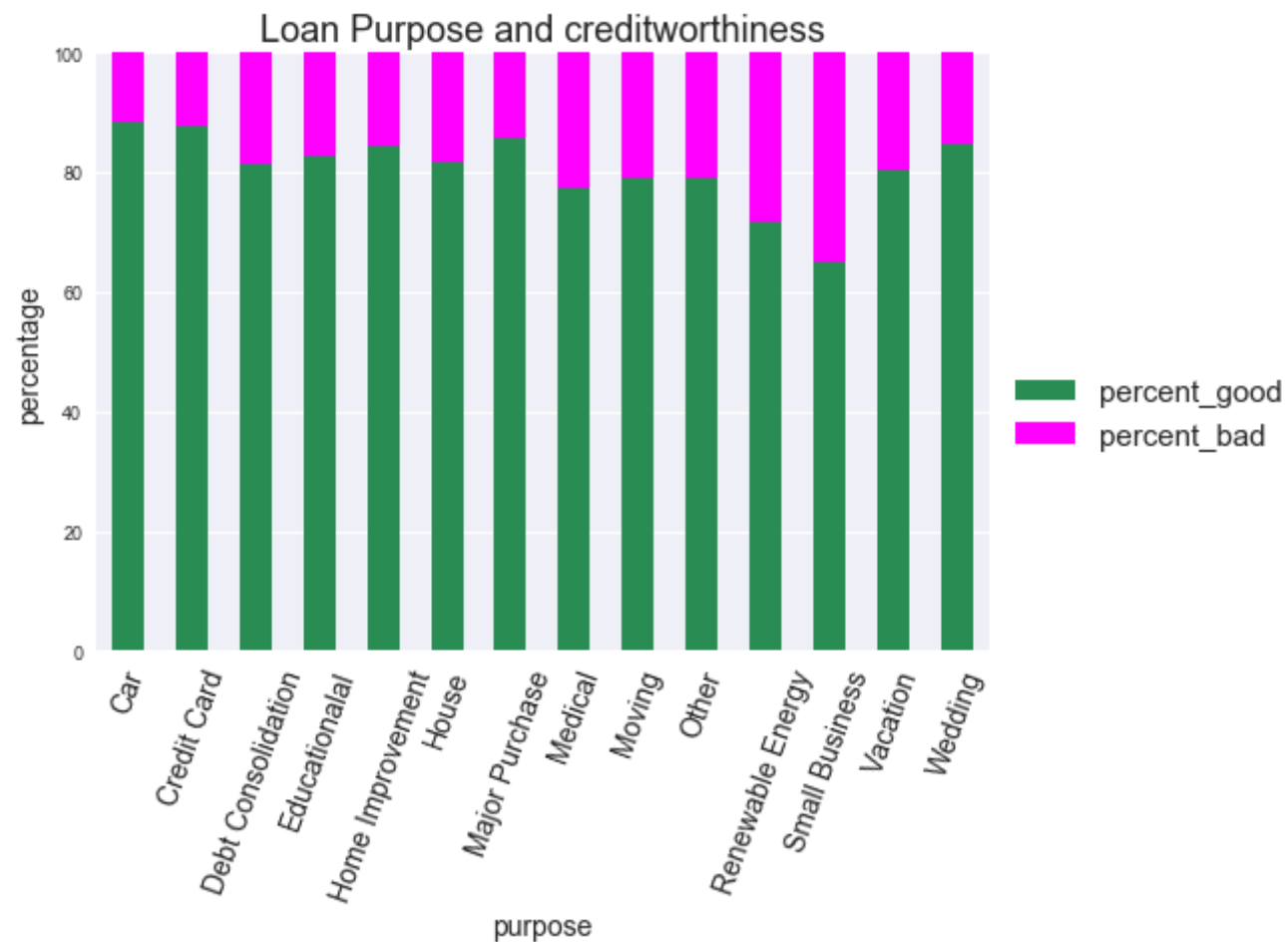
The outcomes show that each customer has either defaulted or completed their loan.

Data exploration



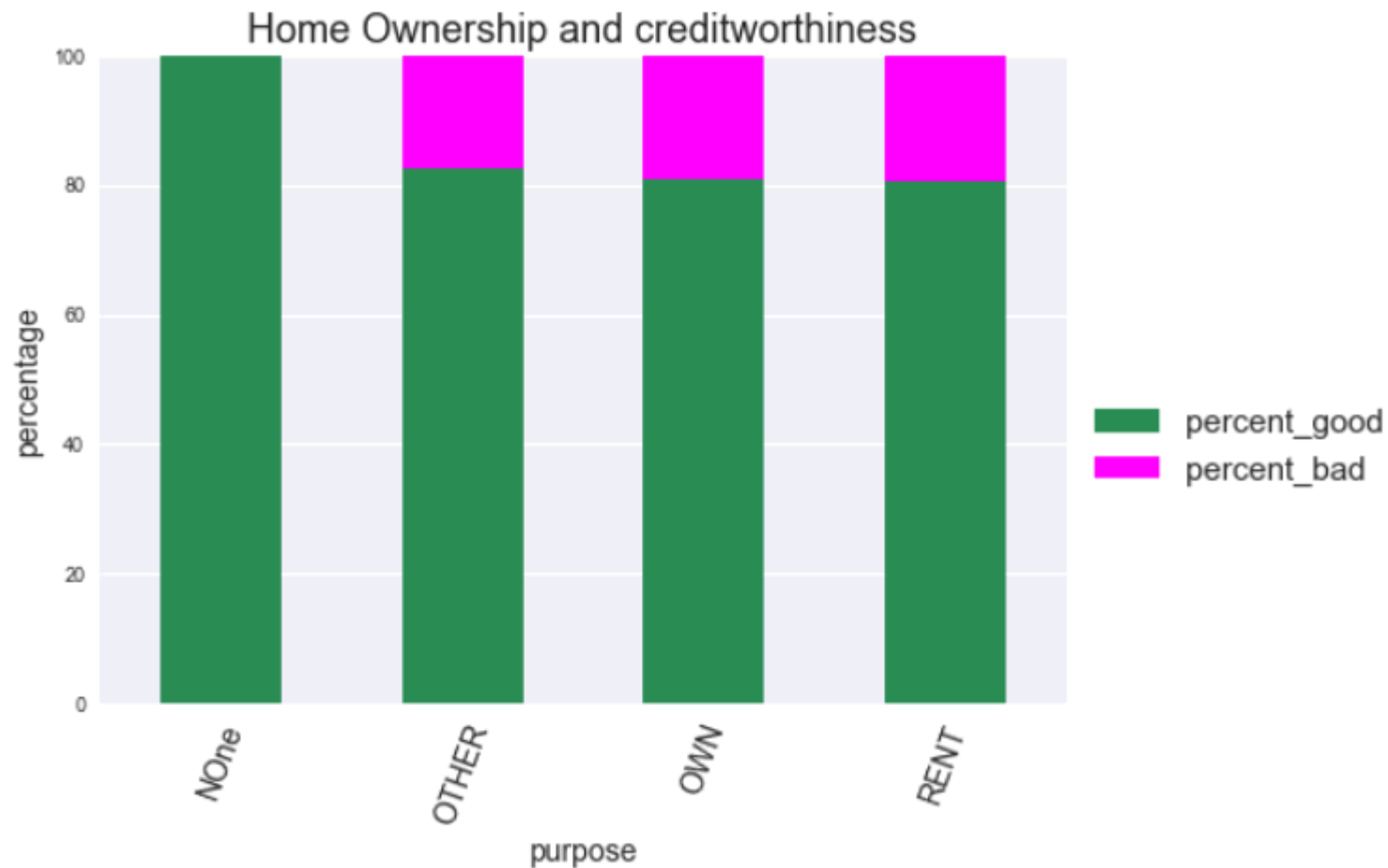
- The dataset is unbalanced, with defaulted loans only accounting for 18.17%.
- The unbalanced nature of the data will have to be considered when making predictions.

Data exploration



Loan purposes 'Renewable Energy' and 'Small Business' have the highest percentage of bad loans.

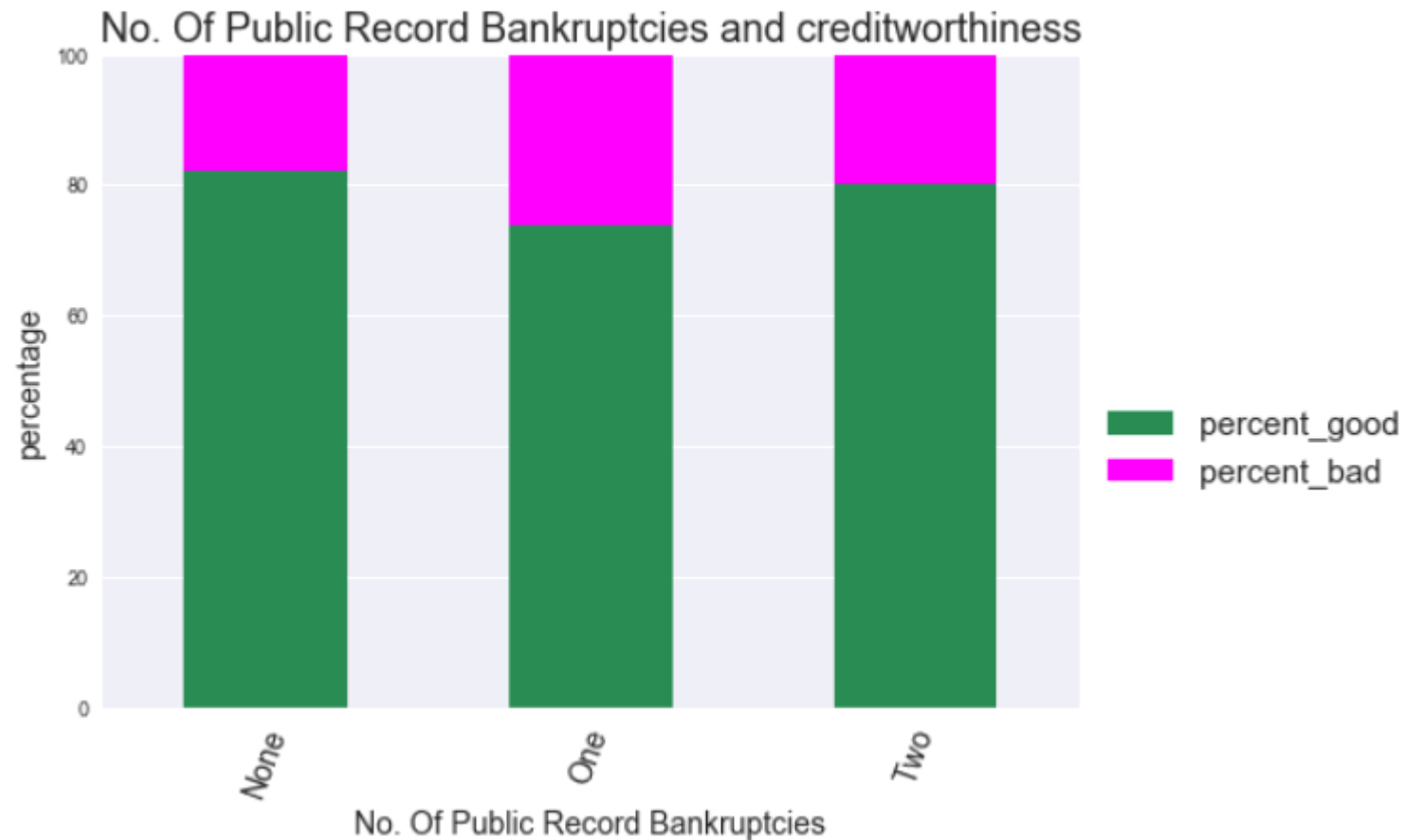
Data exploration



Category 'None' only contains completed loans.

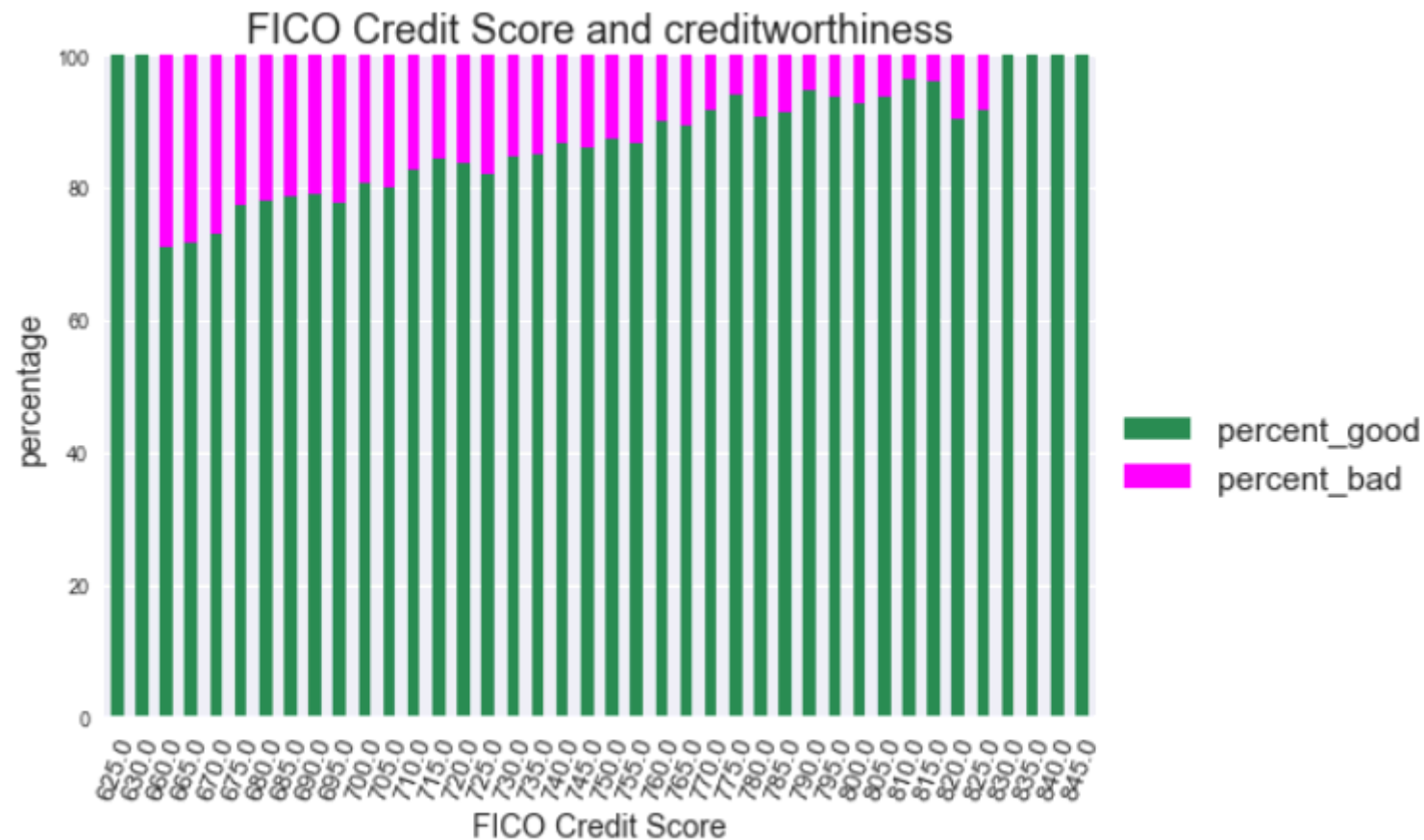
Defaults in category 'Rent' are marginally higher than in 'Other and 'Own'.

Data exploration



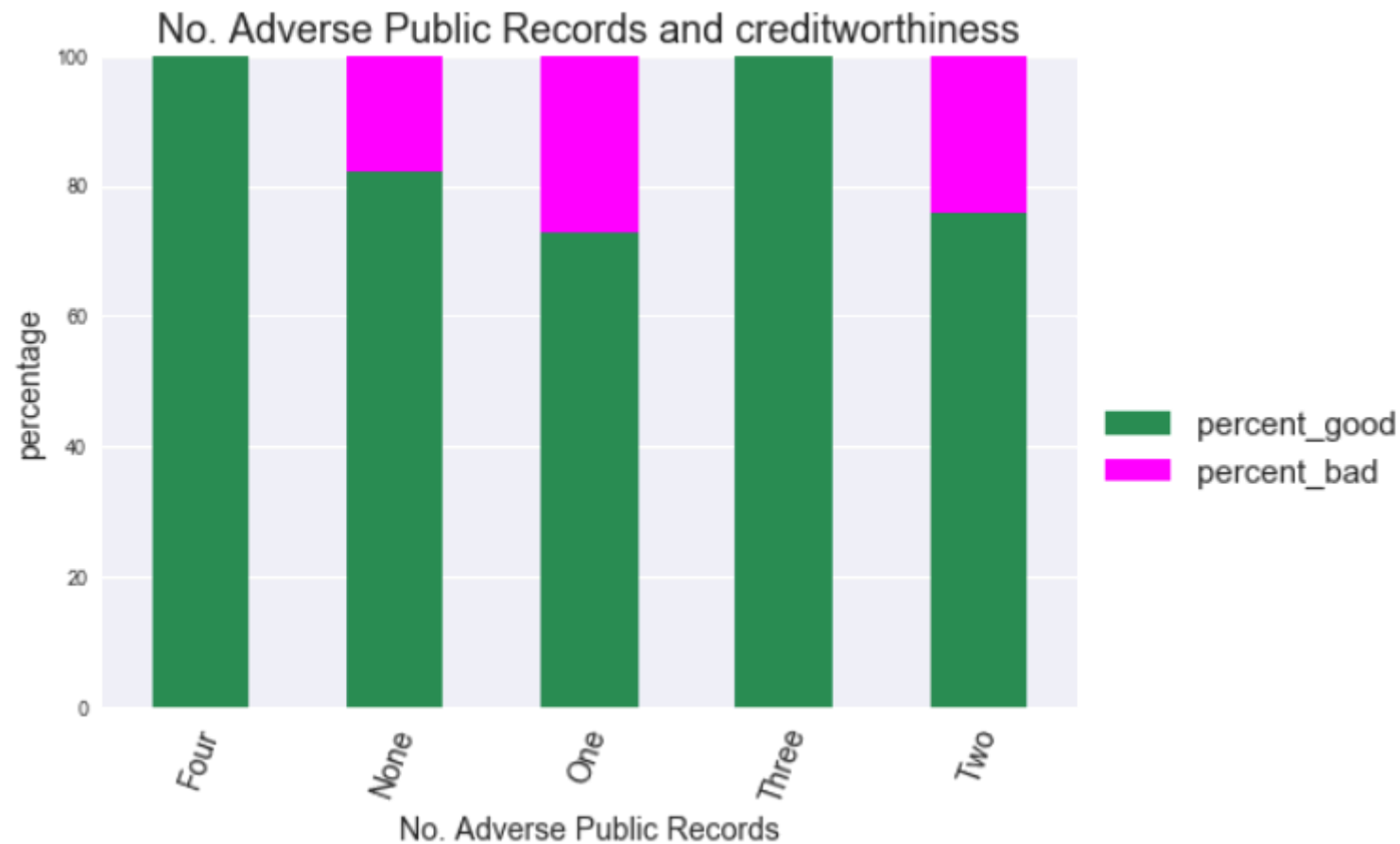
One public record bankruptcy has the highest percentage of bad loans.

Data exploration



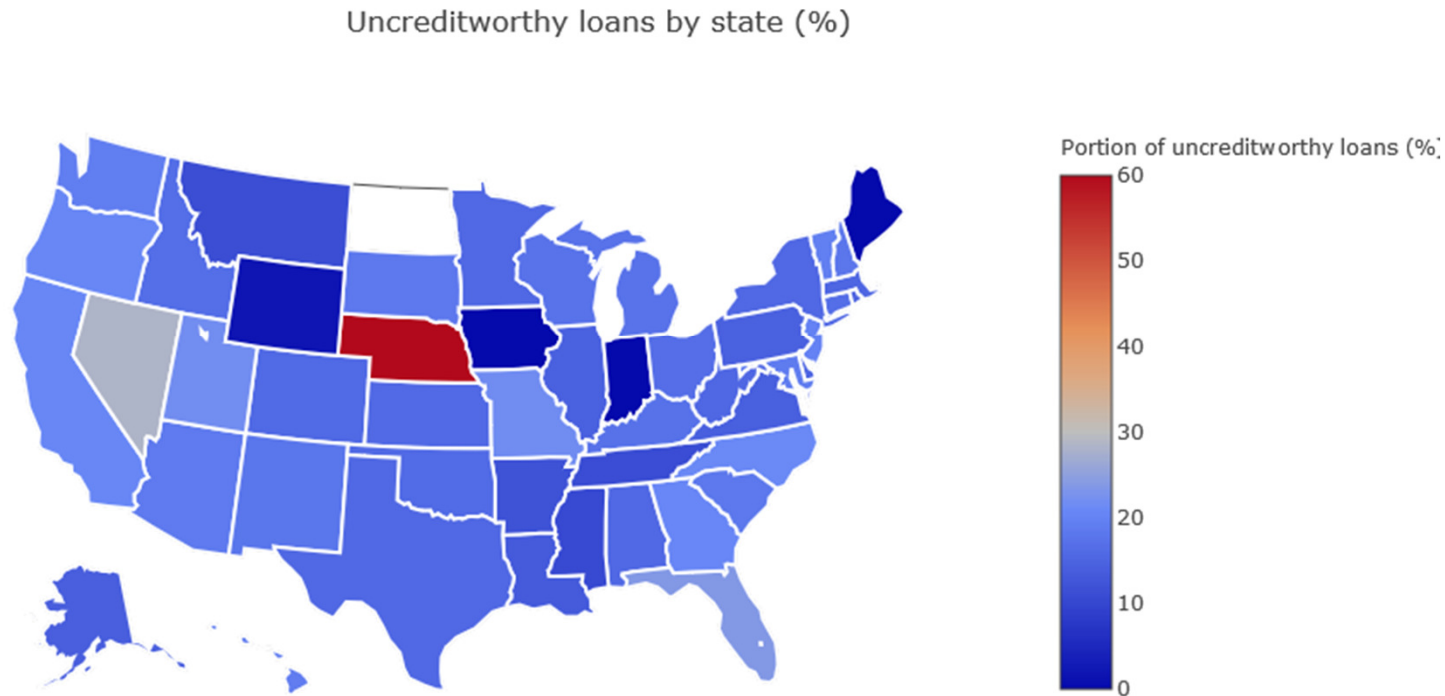
The lowest and the highest FICO scores do not contain bad loans, otherwise bad loans decrease with increase in score.

Data exploration



One adverse public record has the highest percentage of bad loans, while three only contain good loans.

Data exploration



- NE stands out as having 60% of bad loans, or 5 bad loans in absolute numbers.
- NV has 28.43% of bad loans, or 306 bad loans in absolute numbers.

Data exploration: Initial insights

- Relationships between predictor and target variable are not always linear.
- Observations of features 'FICO Credit Score' and 'Address State' can be grouped in order to increase predictive power.

Feature engineering

- convert 'Class' feature (=target) into numerical values
 - Assign '0' to 'creditworthy'
 - Assign '1' to 'uncreditworthy'

Feature engineering

- summarise address states into geographic regions:
 - East_Coast: 7731 observations
 - South: 6885 observations
 - West_Coast: 4485 observations
 - Middle: 3461 observations
 - North: 2213 observations
- summarise FICO scores into categories
 - Very Good: 15950 observations
 - Exceptional: 5208 observations
 - Good: 3337 observations

Feature engineering

- create dummy variables for categorical features
 - 'Loan Term',
 - 'Home Ownership',
 - 'Loan Purpose',
 - 'FICO Credit Score',
 - 'Address State'

Feature engineering

- encode remaining columns containing object data types
 - 'No. Delinquencies In Last 2 Years'
 - 'No. Adverse Public Records'
 - 'No. Of Public Records'
- The above columns contain strings (e.g. one, two, three, ...) describing numbers from 0 to 9
- Change strings into numerical format (e.g. 1, 2, 3, ...)

Feature engineering

- drop redundant feature
 - Drop 'Class' feature
- deal with nan values
 - Replace 'nan' with median values for each feature
- remove outliers
 - See appendix

Classification

- Define predictor variables (X) and target variable (y)
 - y = 'Class_new' column containing '0' for completed and '1' for defaulted loan
 - X = transformed features excluding y
- Implement train test split on X and y
- 70% train, 30% test
- Stratify target y as data may not be normally distributed

Classification: Evaluation metric

- AUC ('area under the curve') score
 - The AUC score equals the probability that a randomly chosen positive example ranks above (is deemed to have a higher probability of being positive than) a randomly chosen negative example.
- TPR (also called Recall or Sensitivity) = $\frac{\text{sum total positives}}{\text{sum of condition positive}} = \frac{\text{TP}}{\text{TP} + \text{FN}}$
- FPR (also called Fall-out) = $\frac{\text{sum of false positives}}{\text{sum of condition negative}} = \frac{\text{FP}}{\text{FP} + \text{TN}}$
- The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings.

Source: www.wikipedia.com

Classification: Build model

```
# build logistic regression model
# address unbalanced classes with parameter 'class_weights'

from sklearn import linear_model
from sklearn.metrics import roc_auc_score

estimator = linear_model.LogisticRegression(random_state=42, class_weight={0:0.1817, 1:0.8183})
estimator.fit(X_train, y_train)

y_pred_regr = estimator.predict(X_test)

# assess model performance using area under the curve (AUC)

from sklearn.metrics import roc_auc_score

AUC = roc_auc_score(y_test, y_pred_regr)
print("The AUC score for a simple Logistic Regression model is {:.4f}.".format(AUC))
```

Add class weights
to account for
unbalanced classes

The AUC score for a simple Logistic Regression model is 0.6584.

Initial AUC score:
0.6584

Classification: Optimise model

```
# tuning model parameters

from sklearn.model_selection import GridSearchCV

param_grid = {'C': [0.1, 1, 2, 3], 'penalty': ['l1', 'l2']}

gsregr = GridSearchCV(estimator, param_grid, cv=8)

gsregr.fit(X_train, y_train)

y_pred_opt = gsregr.predict(X_test)

# how does the optimised model perform?

AUC = roc_auc_score(y_test, y_pred_opt)
print("The AUC score for a simple Logistic Regression model is {:.4f}.".format(AUC))
```

Use GridSearchCV
to find best model
parameters

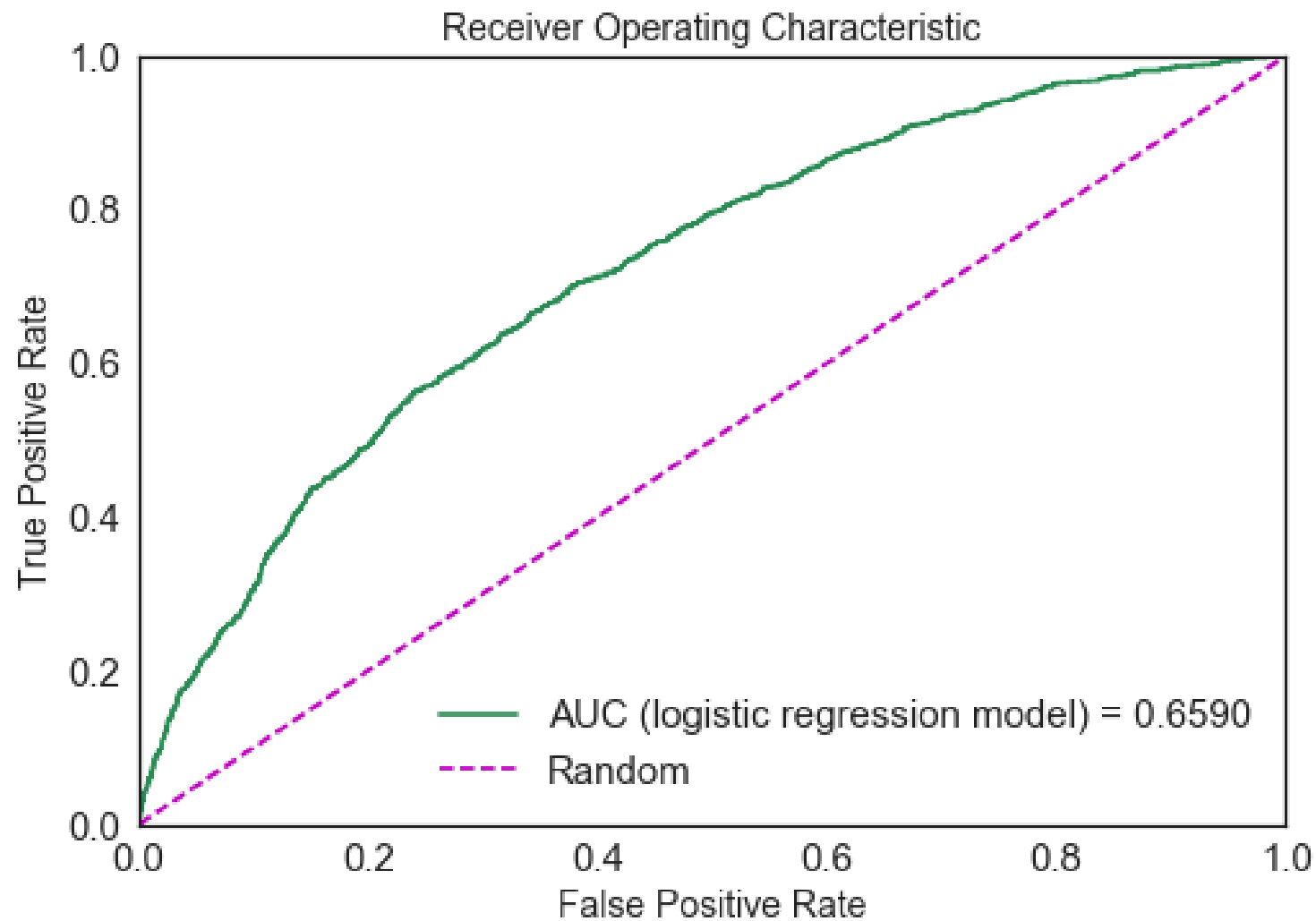
The AUC score for a simple Logistic Regression model is 0.6590.

The optimised
model generates an
AUC of 0.6590

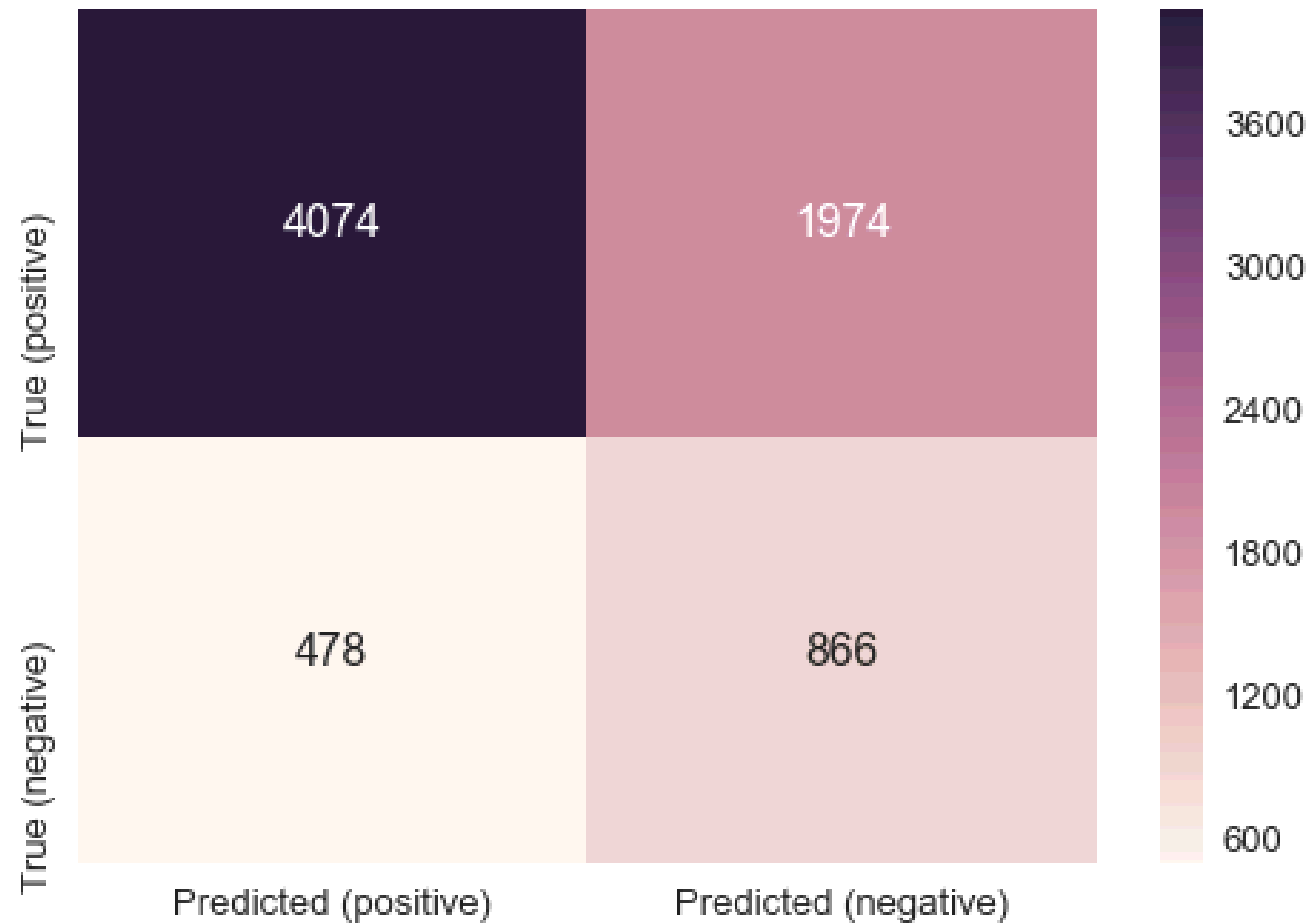
Results

- The optimised model delivered an AUC score of 0.6590.
- This can be interpreted as the probability that a randomly chosen positive example is deemed to have a higher probability of being positive than a randomly chosen negative example.
- Parameter tuning improved the score.

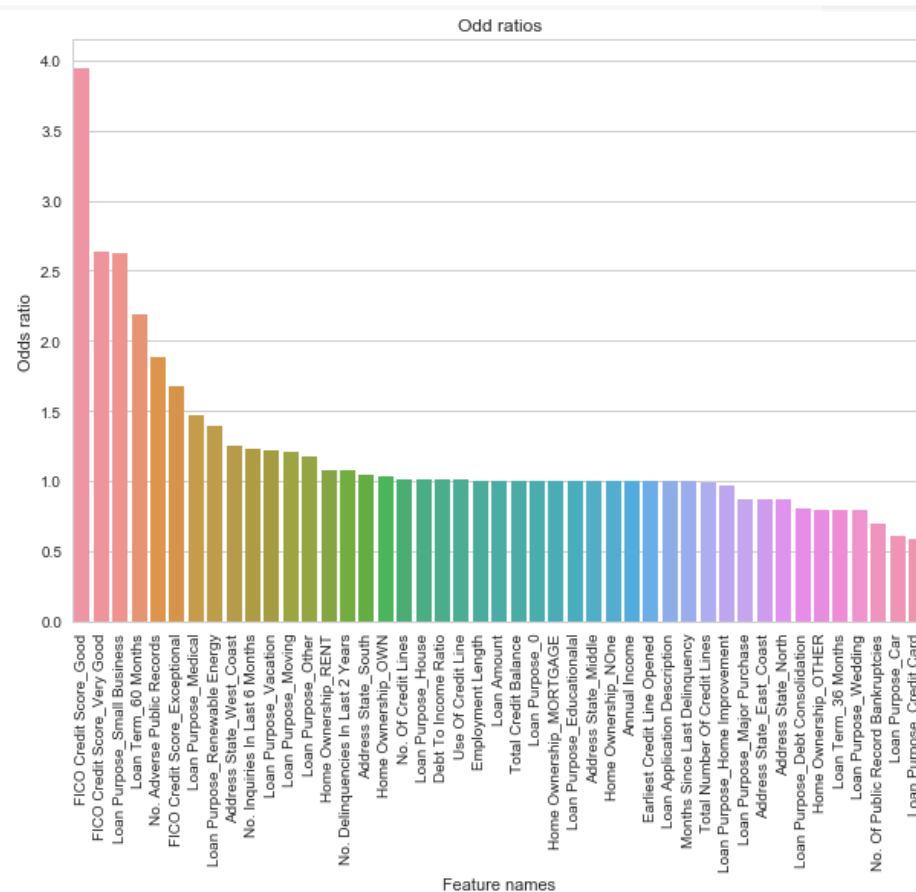
Results: Receiver Operating Characteristic



Results: Confusion matrix



Results: odd ratios for bad loans



How to interpret the odds ratio:

A score over 1 indicates a positive relationship with the target variable, a score below 1 a negative one.

Odds of success = ratio of probability of success / probability of failure

If the probability of success is 50/50, the odds of success are $1/1 = 1$

For example, FICO Credit Score_Good has an odd ratio of 3.948234. This means that the odds for a defaulted loan are 295% higher than the odds for a completed one.

Conclusion

The model outperforms random guessing in predicting loan defaults.

Highest odds for bad loans:

- FICO Credit Score Good
- FICO Credit Score Very Good
- Loan Purpose Small Business
- Loan Term 60 Months
- No. Adverse Public Records

Lowest odds for bad loans:

- Loan Purpose Credit Card
- Loan Purpose Car
- No. Of Public Record Bankruptcies
- Loan Purpose Wedding
- Loan Term 36 Months

Conclusion

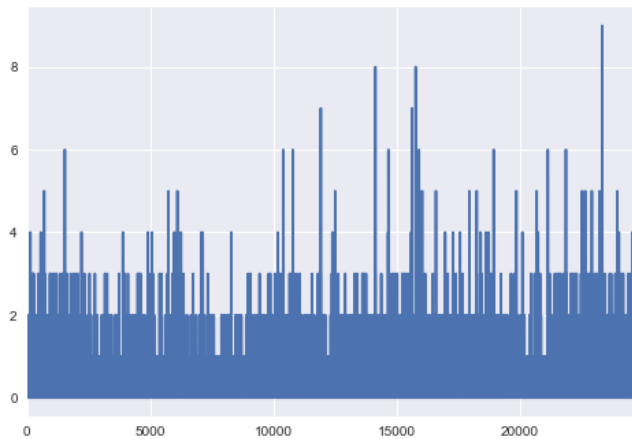
Limitations

- Relationships between predictor variables and target are not necessarily linear.
- The amount of features and observations is not very large.

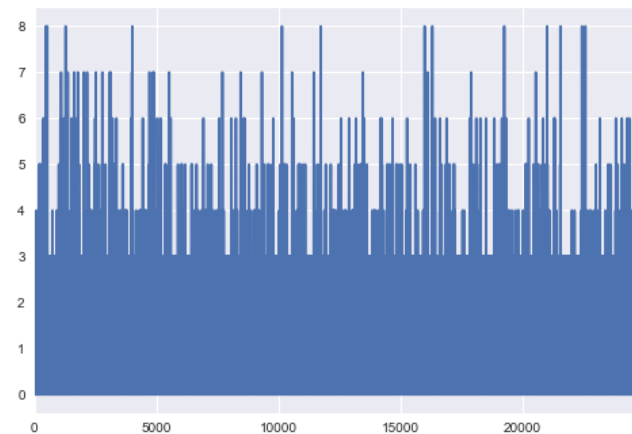
Recommendations

- Get more data.
- Build model using XG Boost, as this algorithm tends to perform well when variables exhibit non-linear relationships.

Appendix: Dealing with outliers

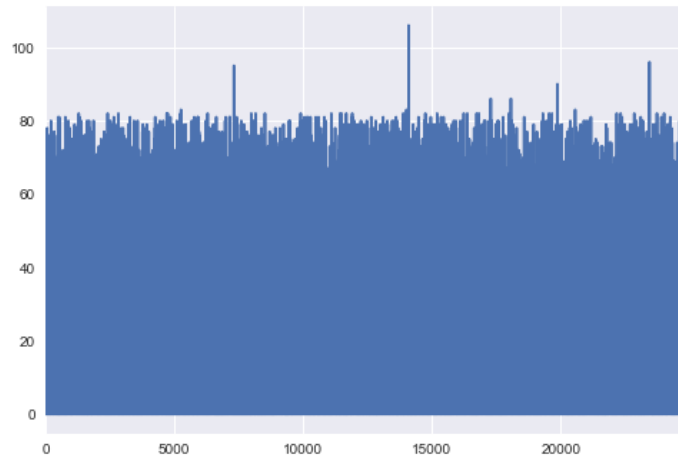


No. Delinquencies In Last 2 Years:
Take out values above 4

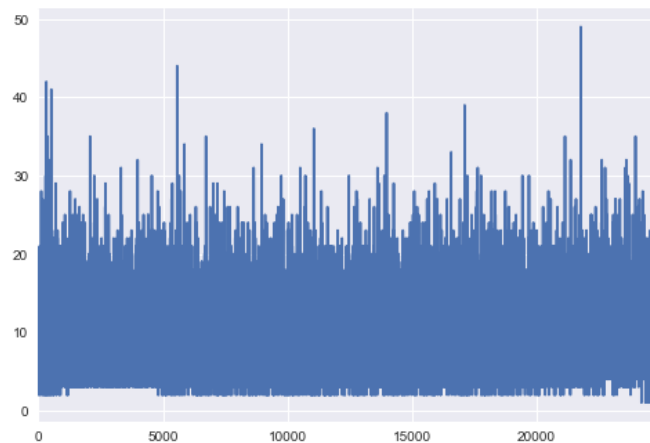


No. Inquiries In Last 6 Months:
Take out values above 7

Appendix: Dealing with outliers

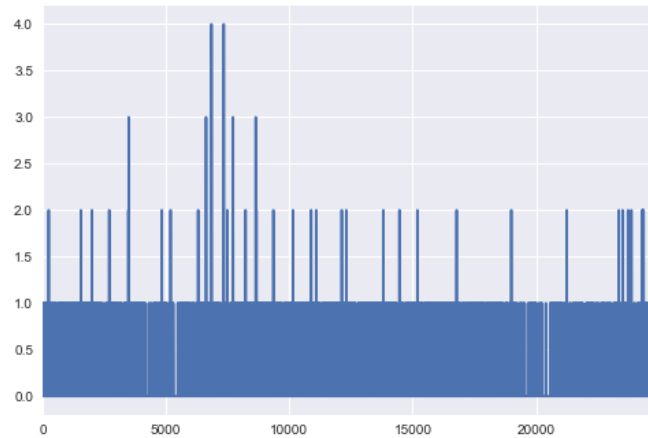


Months Since Last Delinquency:
Take out values above 85

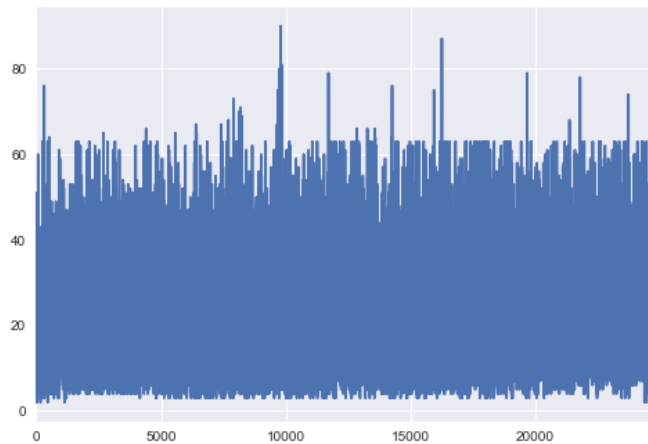


No. Of Credit Lines:
Take out values above 30

Appendix: Dealing with outliers

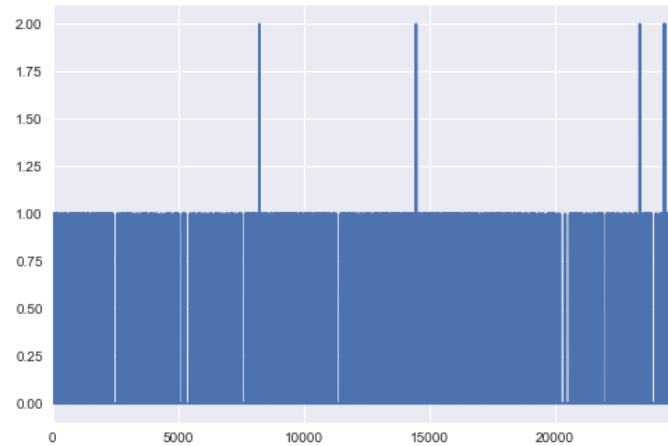


No. Adverse Public Records:
Take out values above 1



Total Number Of Credit Lines:
Take out values above 65

Appendix: Dealing with outliers



No. Of Public Record Bankruptcies:
Take out values above 1