RISK OF MORTGAGE LOAN DEFAULT

PROBLEM STATEMENT

Determine how to to assess the risk of mortgage loan application by categorizing into one of two categories:

- Default (Likely to default)
- Non-Default (Not likely to default)

DATA SOURCES

Fannie Mae has made available on their website a subset of the single-family loans that were acquired by the agency since 2000.

The dataset is available in .txt format by quarter and each quarter has the following two types of datasets:

- Acquisition Data (identifying data i.e. loan type, borrower credit score, original interest rate)
- Performance Data (monthly data i.e. current loan balance, delinquency status, loan age)

DATA WRANGLING

The following steps were taken to clean up the data:

- Adding header names
- Removing rows with missing data for borrower credit score
- Filling in number of borrowers with either 1 or 2 depending on credit score data
- Combining the two dataset files into one DataFrame

EXPLORATORY DATA ANALYSIS

Exploratory Data Analysis (EDA) was conducted by visually exploring the data to gather insights and trends from summary statistics.

Below are the features that were initially reviewed:

- Borrower Credit Score, Co-Borrower Credit Score
- Zero Balance Code
- Original Interest Rate
- Original Unpaid Balance
- Original Loan to Value
- Original Debt-to-Income Ratio

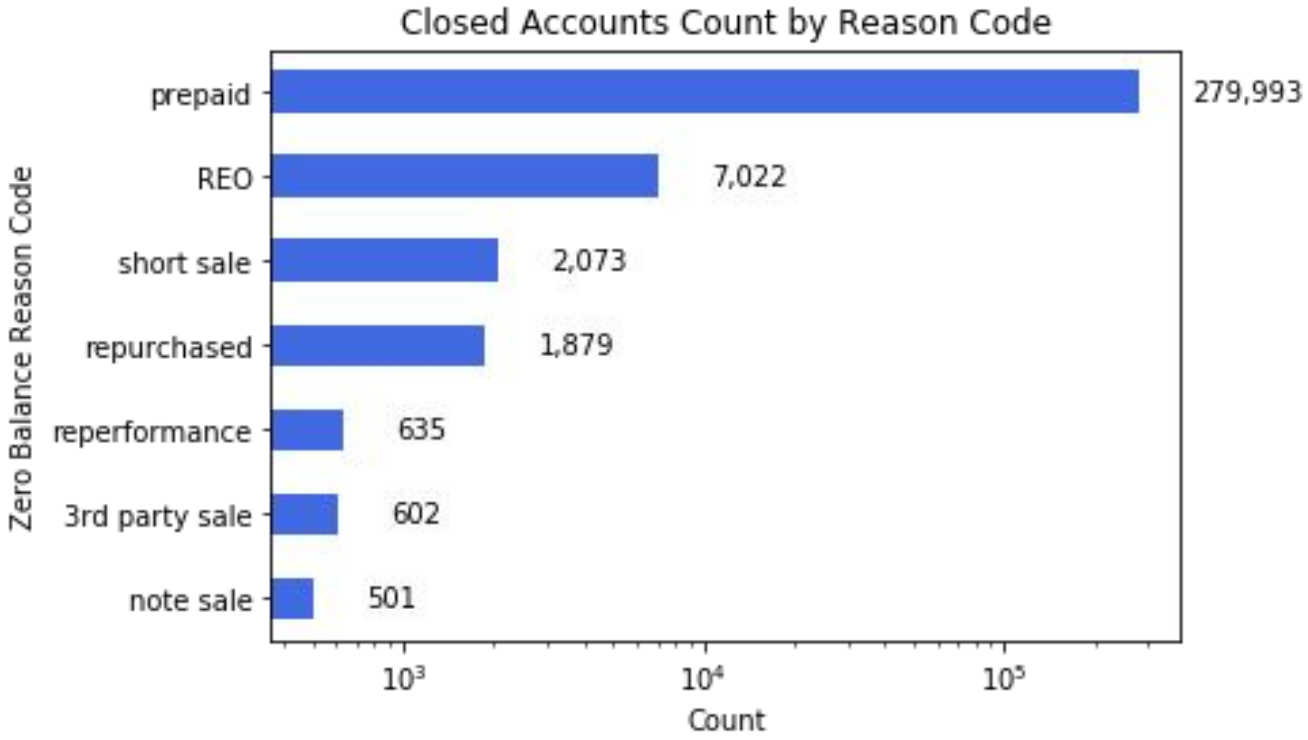
EXPLORATORY DATA ANALYSIS

The loans originating in the 4th quarter of 2008 were used for this analysis.

315k total mortgage loan accounts

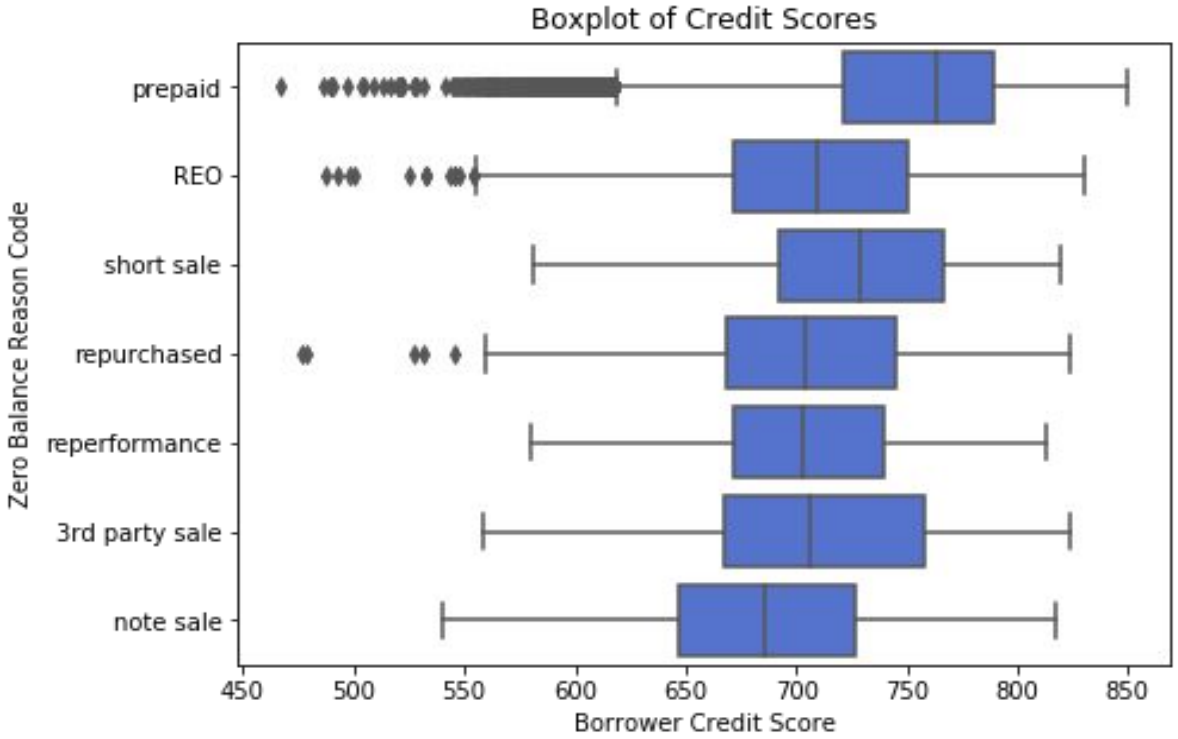
93% of accounts are closed

EXPLORATORY DATA ANALYSIS



EXPLORATORY DATA ANALYSIS

- With a larger portion of the data, the prepaid accounts have a credit scores with a larger variance and higher medians.
- Accounts closed coded as REO had significantly lower median.

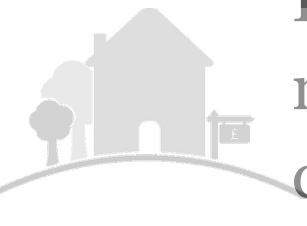


INFERENTIAL STATISTICS

Inferential statistics use random sampling from a population to make deductions about the dataset.

Bootstrap sampling, which involves random sampling with replacement, was employed to test the null hypothesis.

In general, the **Null hypothesis** states that there is not anything significantly different between groups of data.



INFERENTIAL STATISTICS

The null hypothesis used for this dataset is that there is not a difference between credit scores of REO accounts and prepaid accounts.

$\overline{\mu_P}$ = population mean of prepaid loans

$\overline{\mu_R}$ = population mean of REO loans

$\overline{x_P}$ = sample mean of prepaid loans

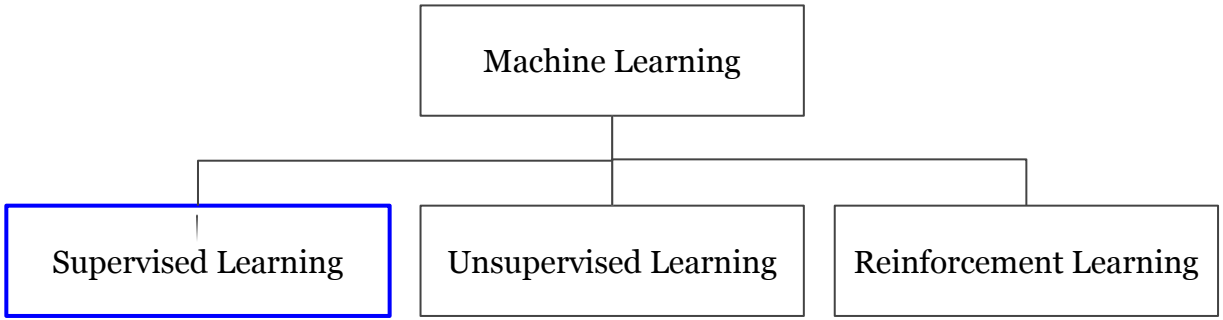
$\overline{x_R}$ = sample mean of REO loans

$$p\ value = \sum \left[(\overline{\mu_P} - \overline{\mu_R}) >= (\overline{x_P} - \overline{x_R}) \right] / 10,000$$

$$p\ value = 0$$

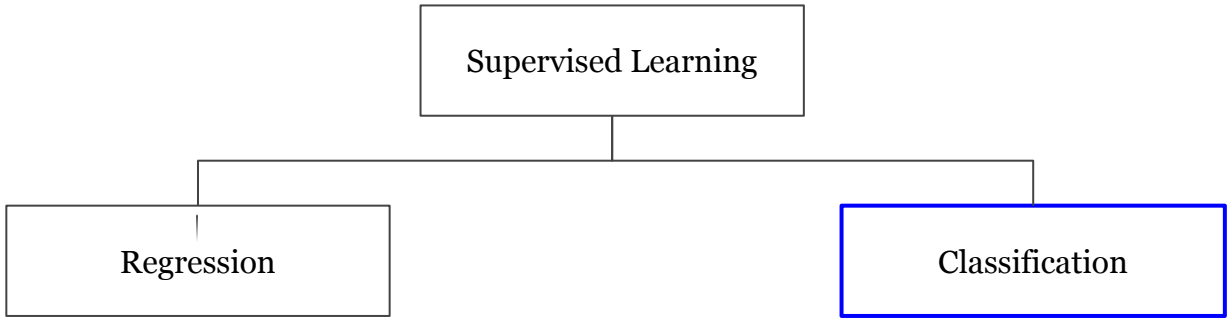
MACHINE LEARNING

Machine learning is a branch of artificial intelligence that uses algorithms to learn without explicitly being programmed.



MACHINE LEARNING

Of the two types of supervised learning, classification is the appropriate choice for placing each loan into two categories.



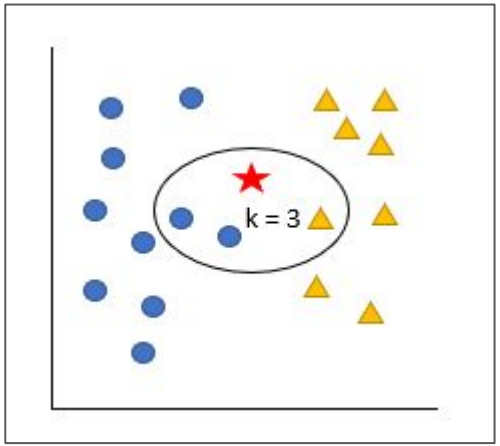
MACHINE LEARNING

Three classification models were considered and metrics compared for this analysis.

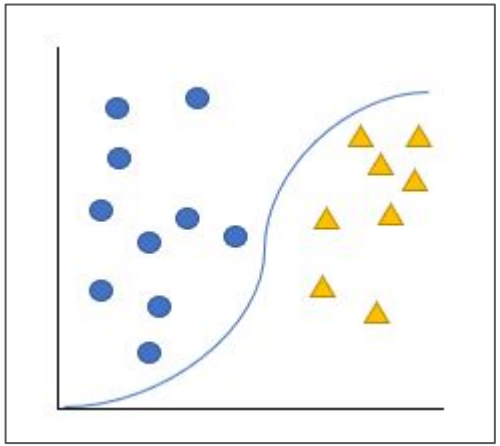
- K Nearest Neighbors - predicts the class of new data by using its proximity to labeled data
- Logistic Regression - uses the logistic function, an S-shaped curve that can take any value between 0 and 1, for binary classification
- Random Forest - aggregates multiple decision trees (makes classification based on a series of rules from the features of the dataset) and uses the majority vote to determine the class

MACHINE LEARNING

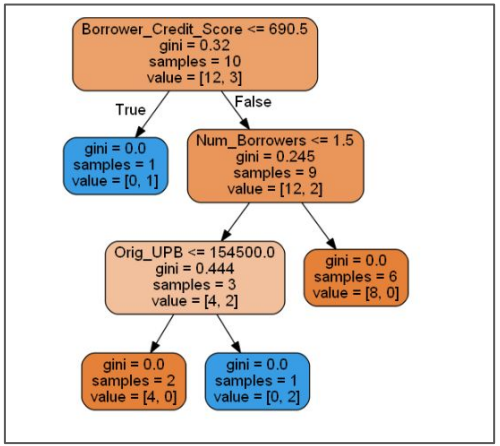
K Nearest Neighbors



Logistic Regression



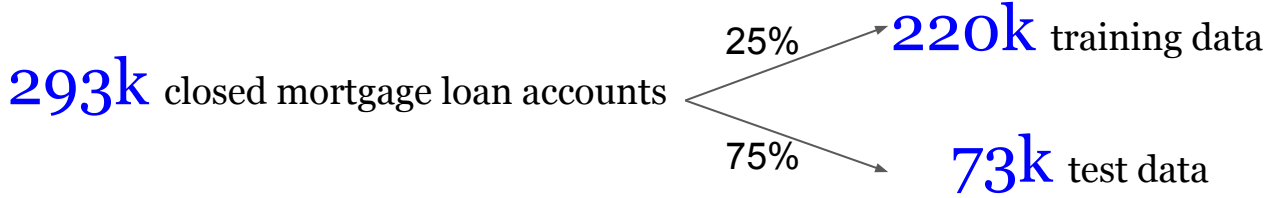
Random Forest



MACHINE LEARNING

In supervised machine learning, the dataset is split into two separate subsets, training and test.

The training subset is used to learn the data and make future predictions on the test dataset. 75% of the data was assigned to the training set and the remaining 25% assigned for the testing.



MACHINE LEARNING

The F1 score was used to measure and compare each of the three algorithms' test data results. F1 is a composite score of the precision and recall scores.

Precision measures the proportion of positive identifications that are correct: True Positive /(True Positive + False Positive)

Recall measures the proportion of actual positives correctly identified: True Positive / (True Positive + False Negative)

F1 Score is the weighted average of the precision and recall scores: 2 * (Precision * Recall) / (Precision + Recall)

MACHINE LEARNING

The features included in the models were reduced to 11 after reviewing the summary from the function pandas_profiling.ProfileReport().

- ‘Borrower_Credit_Score' (Borrower Credit Score)
- 'Coborrower_Credit_Score' (Co-borrower Credit Score)
- 'Orig_Debt_Inc_Ratio' (Original Debt to Income Ratio)
- 'Orig_LTV' (Original Loan to Value)
- 'Orig_UPB' (Original Unpaid Balance)
- 'First_Time_Home_Buyer_Ind' (First Time Home Buyer Indicator)
- 'Loan_Purpose' (Loan Purpose)
- 'Property_Type' (Property Type)
- 'Num_Borrowers' (Number of Borrowers)
- 'Occupancy_Type' (Occupancy Type)
- 'Zip_Code' (Zip Code)

MACHINE LEARNING

The Random Forest classifier was the first to be executed and measured.

```python
# Random forest classifier
from sklearn.ensemble import RandomForestClassifier
def random_forest(n_estimator, max_feature, min_samples_split):
    global y_test, y_predict, model
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state=0)
    model = RandomForestClassifier(n_estimators=n_estimator, random_state=0, max_features=max_feature,
                                   min_samples_split=min_samples_split)
    model.fit(X_train, y_train)
    y_predict = model.predict(X_test)
    return y_test, y_predict, model
```

```python
# Initial random forest run of model using default hyperparameters
random_forest(100, int(np.sqrt(len(features))), 2)
print(classification_report(y_test, y_predict))
```

```
             precision    recall  f1-score   support

          0       0.97      1.00      0.98     70885
          1       0.68      0.02      0.04      2292

avg / total       0.96      0.97      0.95     73177
```

MACHINE LEARNING

As shown in the previous slide, the total F1 score for the Random Forest classifier was quite high. However, when taking a closer look, there was an imbalance between the two classifications. 0 represents non-defaults, whereas 1 represent defaults. Since there were so many more non-defaults, the composite score was very high, however the f1 score for the defaults was unacceptably low at a mere 4%.

	precision	recall	f1-score	support
0	0.97	1.00	0.98	70885
1	0.68	0.02	0.04	2292
avg / total	0.96	0.97	0.95	73177

MACHINE LEARNING

To mitigate the imbalance, resampling in the form of random undersampling was performed to reduce the count of non-defaults equal to the number of defaults.

```
# Resample data for undersampling
from imblearn.under_sampling import NearMiss
nr = NearMiss()
X, y = nr.fit_sample(X, y)
```

Original Data		Resampled Data
9.7k default loans	→	9.7k default loans
283k non-default loans	→	9.7k non-default loans

MACHINE LEARNING

F1-score for defaults improved after undersampling the going from from 4% to 79%.

Before Sampling

	precision	recall	f1-score	support
0	0.97	1.00	0.98	70885
1	0.68	0.02	0.04	2292
avg / total	0.96	0.97	0.95	73177

After Sampling

	precision	recall	f1-score	support
0	0.79	0.77	0.78	2402
1	0.78	0.79	0.79	2430
avg / total	0.78	0.78	0.78	4832

MACHINE LEARNING

K Nearest Neighbors was executed next and the F1 score for defaults was calculated as 75%.

```
# k nearest neighbors
from sklearn.neighbors import KNeighborsClassifier
X_train, X_test, y_train, y_test = train_test_split(
            X, y, test_size = 0.25, random_state=0)

knn = KNeighborsClassifier(n_neighbors=5)

knn.fit(X_train, y_train)

y_predict = knn.predict(X_test)
print(classification_report(y_test, y_predict))
```

	precision	recall	f1-score	support
0	0.72	0.89	0.80	2402
1	0.86	0.66	0.75	2430
avg / total	0.79	0.78	0.77	4832

MACHINE LEARNING

Finally, Logistic Regression was executed and the F1 score for defaults was calculated as 66%.

```
# Logistic regression
from sklearn.linear_model import LogisticRegression
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25,
                                                    random_state=0)
logreg = LogisticRegression()
logreg.fit(X_train, y_train)
y_predict = logreg.predict(X_test)
print(classification_report(y_test, y_predict))
```

	precision	recall	f1-score	support
0	0.65	0.59	0.62	2402
1	0.63	0.69	0.66	2430
avg / total	0.64	0.64	0.64	4832

MACHINE LEARNING

The Random Forest model having the highest F1 score of 79% for default loans was chosen as the best choice for this classification problem.

Three of the hyperparameters were tuned to achieve a slightly better F1 score:

- n_estimators=[10, 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000], the number of trees in forest
- max_features=[2, 3, 4, 5, 6, 7, 8, 9, 10, 11], number of features to consider when looking for the best split
- min_sample_split= [2, 4, 8, 16, 32, 64, 128, 256], minimum number of samples required to split an internal node

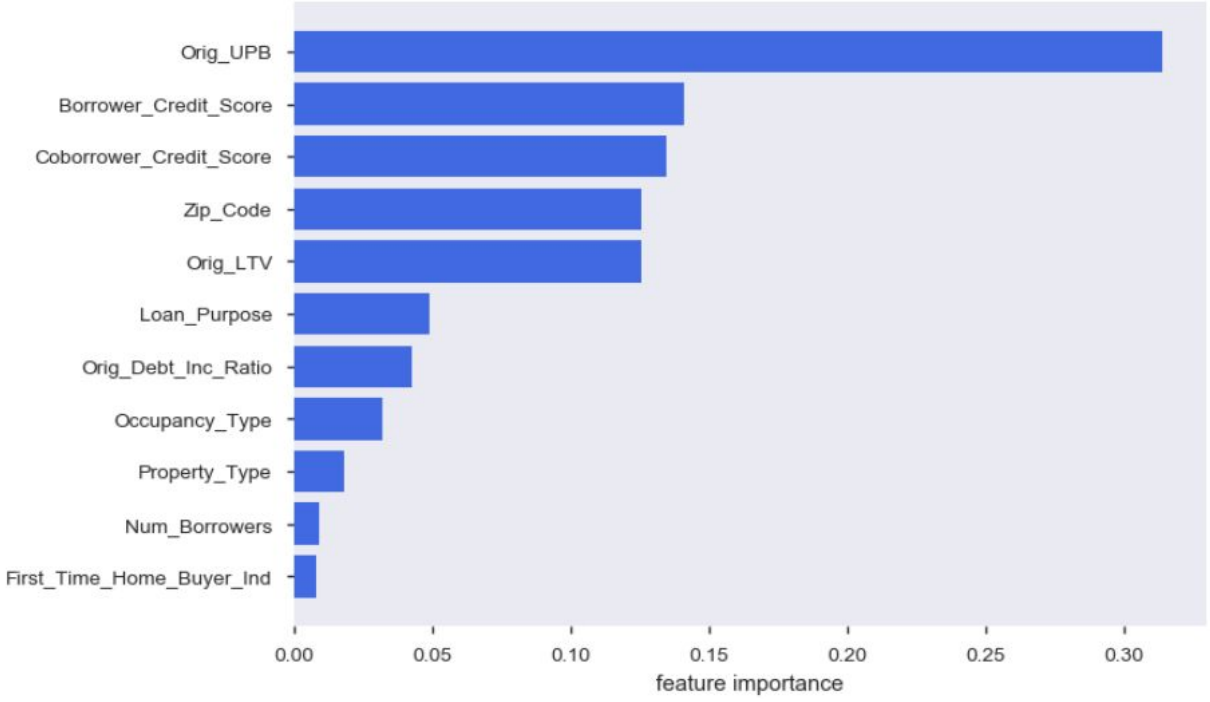
MACHINE LEARNING

The final Random Forest F1 score for predicting loan defaults (with n_samples=400, max_features=2, and min_sample_split=64) was calculated as 80%.

	precision	recall	f1-score	support
0	0.81	0.75	0.78	2402
1	0.77	0.82	0.80	2430
avg / total	0.79	0.79	0.79	4832

MACHINE LEARNING

Percent of importance by feature for Random Forest model



CONCLUSION

- Using the Random Forest classifier on the Fannie Mae dataset provided a fairly good model, with a F1 score of 80%, to predict the loans that may default with the test sample of 19k records.
- This model could be approved by reducing the number of features and further tuning of the hyperparameters.
- Pertinent data, such as borrower income was not included in the dataset and probably would have been useful in this analysis.