Mortgage Loan's Ever Delinquent Probability Prediction

Sunny Liu
Dec 23, 2017

1. Introduction

1.1 Problem Statement

Private mortgage insurance, or PMI, is typically required with most conventional (non-government backed) mortgage programs when the down payment is less than 20% of the property value. In other words, when purchasing or refinancing a home with a conventional mortgage, if the loan-to-value (LTV) is greater than 80% (or equivalently, the equity position is less than 20%), the borrower will likely be required to carry private mortgage insurance.

Company ACME is one of the top players in PMI industry. Similar to other insurance companies, the way ACME makes money is by collecting PMI premium from the lenders (ultimately from home buyers). However, if a home buyer misses multiple mortgage payments, the loan will fall under "default/delinquent" status. If it is never cured within required time frame, the lender will file a claim to ACME which pays agreed-upon coverage amount if the claim gets ultimately approved.

The goal of this project is to predict mortgage loan's ever-delinquent probability by using machine learning approach, and assist ACME make decisions on whether to insure a given mortgage loan sent from the lender.

1.2 Metrics

The key metrics used in this project to determine whether ACME insures a loan are expected premium received amount and expected claim paid amount. See the calculation of each below:

- **Expected Premium Received** = Premium Amount **X** Payment Frequency **X** Expected Life of PMI
- **Expected Claim Paid** = Loan Amount **X** Coverage % **X** ***Ever Delinquent Probability*** **X** Claim Rate
- If Expected Premium Received >= Expected Claim Paid, then insure the loan
- If Expected Premium Received < Expected Claim Paid, then DO NOT insure the loan

1.3 Glossary

The following terms are important concepts in the mortgage insurance industry.

- **LTV**: Loan to Value. It represents the mortgage portion of the property.
- **DTI**: Debt to Income ratio. The number is one way lenders measure your ability to manage

the payments you make every month to repay the money you have borrowed.

- **FICO**: Credit score used in US. The number represents the creditworthiness of a person, the likelihood that person will pay his or her debts.
- **Loan Purpose**: It is a term in United States mortgage industry to show the underlying reason an applicant is seeking a loan. The purpose of the loan is used by the lender to make decisions on the risk and may even impact the interest rate that is offered. Possible loan purposes are "Purchase", "Refinance with Cash-Out" (higher risk) and "Refinance Pay-off Existing Lien".
- **Property Type**: Types of property is also one of the factors when considering default risks. For instance, ACME does not insure investment property housing since 2008. Property types that ACME insures are "Single-Family Home", "Condo" and "Manufactured Housing".
- **Origination Channel**: The channel which a mortgage loan is originated from are "Retail", "Broker" or "Correspondent".
    - *Retail:* A mortgage loan for which the mortgage loan seller takes the mortgage loan application and then processes, underwrites, funds and delivers the mortgage loan to Fannie Mae.
    - *Correspondent:* A mortgage loan that is originated by a party other than a mortgage loan seller and is then sold to a mortgage loan seller.
    - *Broker:* A mortgage loan that is originated under circumstances where a person or firm other than a mortgage loan seller or lender correspondent is acting as a “broker” and receives a commission for bringing together a borrower and a lender.
- **Occupancy Status**: The status includes "Principal Residence", "Second Home" and "Investment".
    - *Principal Residence:* A principal residence is a property that the borrower occupies as his or her primary residence.
    - *Second Home:* Second home is a property that the borrower occupies as his or her secondary residence.
    - *Investment:* An investment property is owned but not occupied by the borrower.

2. Data

2.1 Data Description and Wrangling

1) Data Population: The data used in this project are mortgage loans insured by ACME and were originated during 2010-2013. I purposely avoided any pre-2009 loans in order to minimize the impact from 2008 financial crisis. Using 2013 as the upper limit cut-off is based on the consideration of giving loans enough time (more than 4 years) to stabilize the status.
2) Data Size: The total number of records for the above-mentioned population is 478,262.
3) Attributes: 27 attributes are included in the dataset. Some key loan characteristics are FICO, LTV, DTI, Occupancy Status, Loan Purpose, First-time Home Buyer Indicator, Number of Borrowers, Number of Units, Property Type and Origination Channel. I also included loan amount and premium related information for later dollar amount calculation purpose.
4) Missing Data: If the missing data belongs to FICO, LTV or DTI, it gets filled with the most risky bucket to be conservative. For other attributes, the missing data gets replaced with the value based on "majority rule".
5) Data Bucketing: FICO, LTV and DTI values have been transformed into buckets in the dataset following the historically established bucketing rules.

2.2 Exploratory Data Analysis

Although there are 27 attributes originally included in the dataset, only **10** are selected as **features** mainly due to one of following reasons:
1) The attribute does not represent unique record ID in the dataset.
2) Number of distinct values under an attribute should be more than 1. In addition, the distribution of distinct values should not be extremely unbalanced (eg. if more than 99.5% of records share the same value, then the corresponding attribute should be excluded).
3) Should make business sense when considering the possible impact/cause to loan's ever delinquent status.

See below regarding selected feature's distribution as well as the target (ever delinquent flag) distribution

Orig Yr	% to All	Claim %
2010	10.6%	0.6%
2011	15.0%	0.3%
2012	34.5%	0.1%
2013	40.0%	0.1%

Loan Purp	% to All
Refi Cash Out	2.6%
Refi Payoff Lien	32.3%
Purchase	65.2%

Property Type	% to All
Co-op or Condo	9.8%
Manufacutre Housing	0.3%
Single Fam	89.9%

Occupancy Status	% to All
Primary Resident	96.5%
Secondary Resident	3.5%

First Time Home Buyer	% to All
Y	31.7%
N	68.3%

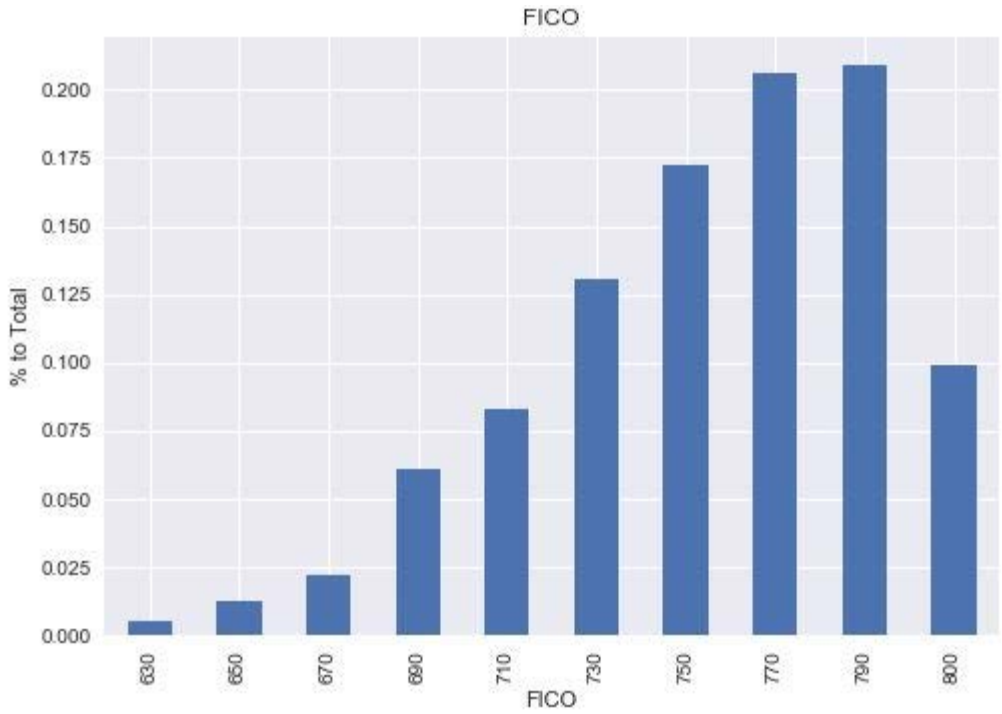
Multi Borrower	% to All
Y	49.2%

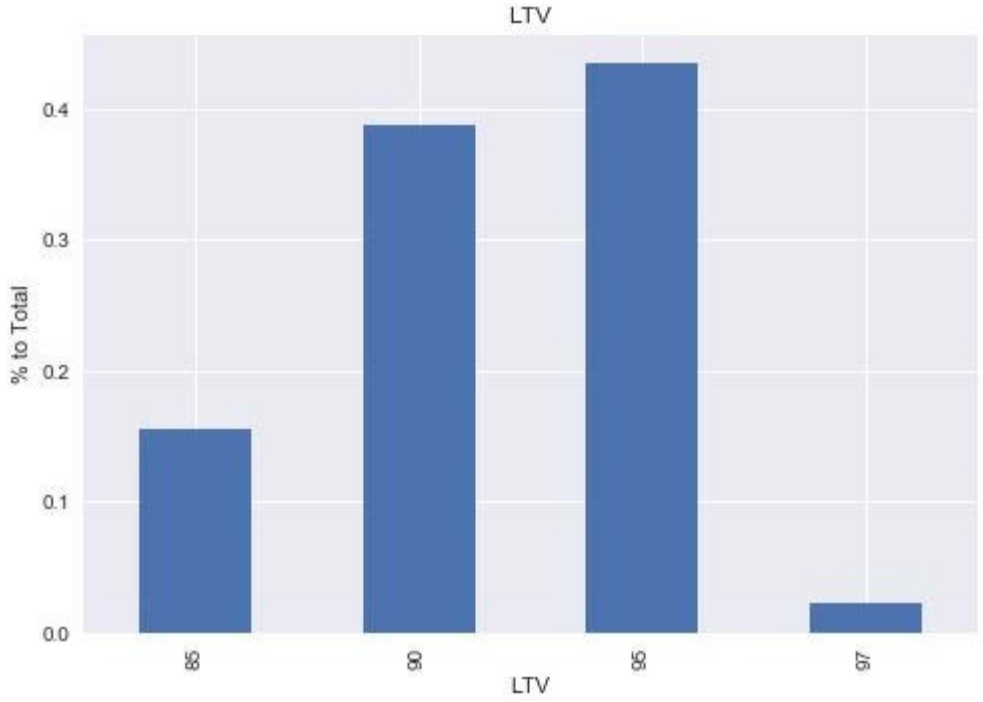
N	50.8%

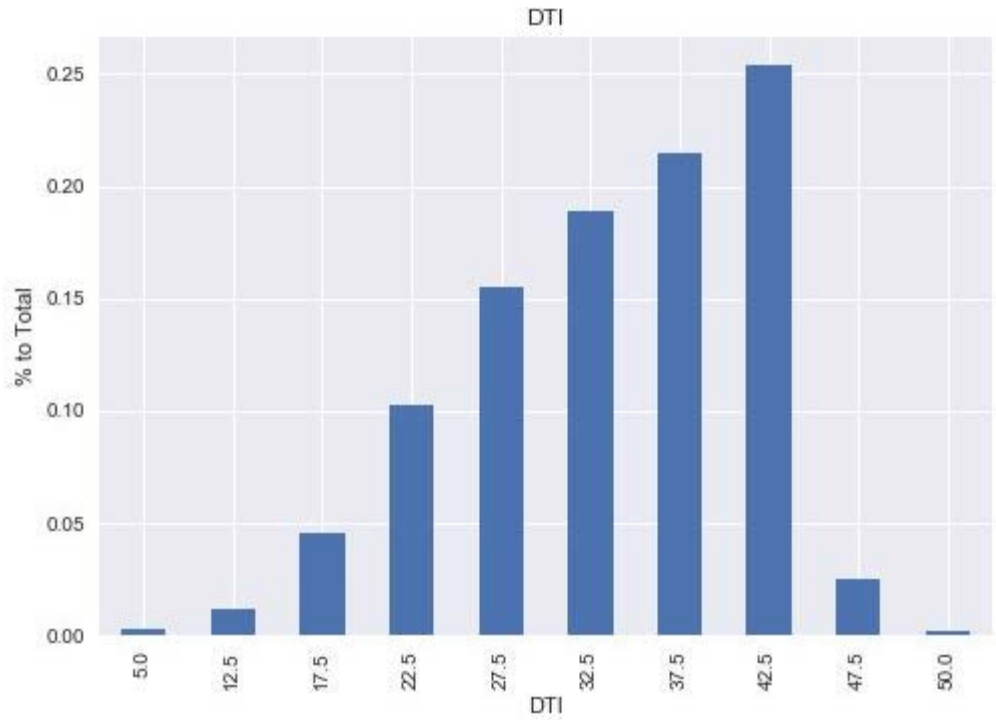
MI Channel	% to All
Delegated	66.7%
Non-Delegated	33.3%

Ever Delinquent	% to All
Y	2.7%
N	97.3%

Features such as FICO, LTV and DTI have been transformed into corresponding buckets. The distribution of each is as below:







In addition, due to unbalanced class distribution of the target (2.7% delinquent loans vs. 97.3% never delinquent loans), I downsampled the "never delinquent" population. At end of the data exploration step, 10,000 loans were sampled for modeling: **5,000 never delinquent and 5,000 ever delinquent**.

3. Training and Modeling

3.1 Objective

1) Target: The target of this project is a given mortgage loan's ever delinquent probability.

2) Features: 10 chosen features are: FICO, DTI, LTV, Origination Year, Loan Purpose, Property Type, Number of Borrower, Origination Channel, First Time Homebuyer Indicator, Occupancy Status.

3) Loss Functions:

- In machine learning, loss function is used to measure the degree of fit. It represents the price paid for inaccurately predicting a class(s).
- Below are some commonly used loss functions for classification problems:

  Logistic Loss: Log Loss measures the performance of a classification model whose output is a probability value between 0 and 1. The loss increases as the predicted probability diverges from the actual label. For binary classification problem, the log loss function is $-(y \quad \log(p) + (1-y) \quad \log(1-p))$. The example of algorithm which is based on log loss function is Logistic Regression.
  Hinge Loss: The hinge loss is used for "maximum-margin" classification, most notably for support vector machines (SVMs). For an intended output $t = \pm 1$ and a classifier score y, the hinge loss of the prediction y is defined as $\ell(y) = \max(0, 1 - t \quad y)$.
  0/1 Loss: This is one of the loss function which does not have convex shape $\min_\theta \sum_i L_{0/1}(\theta^T x)$. We define $L_{0/1}(\theta^T x) = 1$ if $y = \theta^T x$, and $L_{0/1}(\theta^T x) = 0$ if $y \neq \theta^T x$.

Since the target of this project is to predict probability, I tried different classification algorithms based on loss functions such as log loss or hinge loss.

3.2 Model Selection

1) Classification Models: Since this project is a typical classification problem, I tried 8 different classification models on the training set during "Model Selection" step. The models are Logistic Regression(LR), Linear Discriminant Analysis(LDA), K-NN, Decision Tree(CART), Naive Bays(NB), SVM, Kernel SVM and Random Forest(RF).

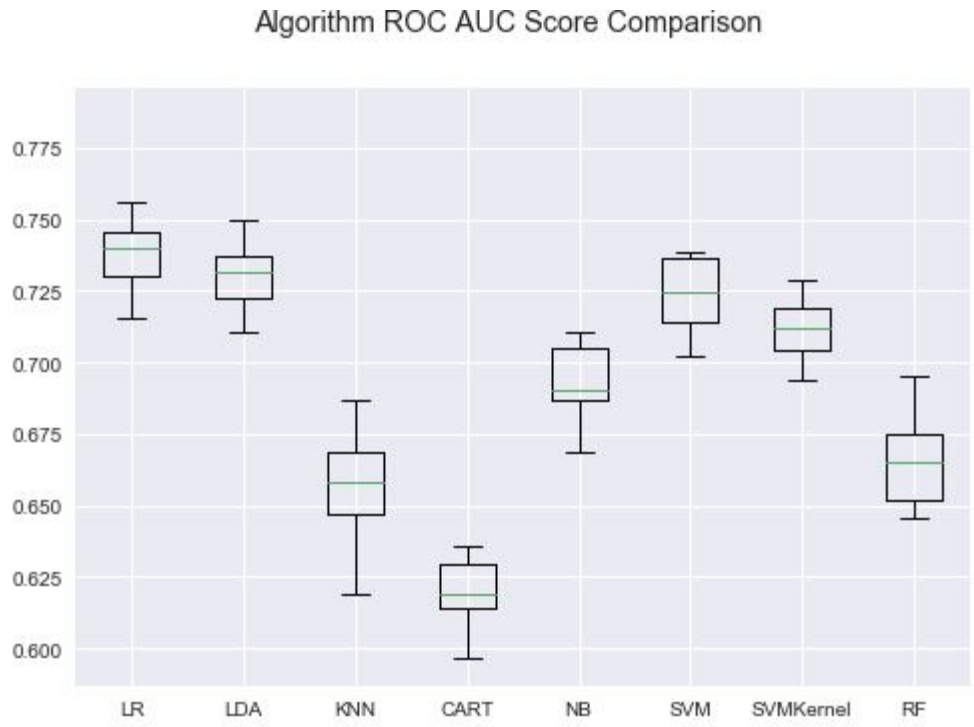
```
# prepare models
models = []
models.append(('LR', LogisticRegression(random_state=0)))
models.append(('LDA', LinearDiscriminantAnalysis(solver='eigen', shrinkage='auto')))
models.append(('KNN', KNeighborsClassifier(n_neighbors=5, metric='minkowski', p=2)))
models.append(('CART', DecisionTreeClassifier(criterion='entropy', random_state=0)))
models.append(('NB', GaussianNB()))
models.append(('SVM', SVC(kernel="linear", random_state=0)))
models.append(('SVMKernel', SVC(kernel="poly", random_state=0)))
models.append(('RF', RandomForestClassifier(n_estimators=100, criterion='entropy', random_state=0)))
```

2) ROC Curve and ROC AUC Score: ***ROC Curve*** is the plot of TPR (True Positive Rate) and FPR (False Positive Rate) when comparing predicted target outcome vs. actual considering all possible 0 to 1 threshold. ***ROC AUC*** is the area under the ROC curve which represents the performance of the classifier. If ***ROC AUC score*** is 0.5 then the prediction is as good as random guessing; if ROC AUC score is 1 then prediction is perfect. The more ROC AUC score is closer to 1, the better the model prediction is.

Below is the comparison among above-mentioned 8 classification models in terms of the mean of ROC AUC score as well as the standard deviation of ROC AUC score after cross validation step:

```
# prepare configuration for cross validation test harness
seed = 7

# Compare AUC score among different models
results = []
names = []
scoring = 'roc_auc'
for name, model in models:
        kfold = model_selection.KFold(n_splits=10, random_state=seed)
        cv_results = model_selection.cross_val_score(model, X_train[:, 1:], y_train, cv=kfold, scoring=
scoring)
        results.append(cv_results)
        names.append(name)
        msg = "%s: %f (%f)" % (name, cv_results.mean(), cv_results.std())
        print(msg)

# boxplot algorithm comparison
fig = plt.figure()
fig.suptitle('Algorithm ROC AUC Score Comparison')
ax = fig.add_subplot(111)
plt.boxplot(results)
ax.set_xticklabels(names)
plt.show()
```

```
LR: 0.741572 (0.018733)
LDA: 0.733755 (0.017979)
KNN: 0.656545 (0.017935)
CART: 0.621446 (0.015338)
NB: 0.695944 (0.019894)
SVM: 0.728301 (0.021162)
SVMKernel: 0.713313 (0.014212)
RF: 0.665516 (0.015130)
```



As the boxplot shows, ***LR outperforms all other models*** - it has the highest mean of ROC AUC score as well as lowest standard deviation of ROC AUC score. LDA and SVM have the 2nd and 3rd best performance based on ROC AUC score compared to other non-linear models.

It is also noticeable that LR and LDA have very similar performance. This is because they have no difference in model function but assumptions on feature distribution and the estimation of the

coefficients. In general, LR is the more flexible and more robust method in case of violations of these assumptions.

3) Hyperparameter Optimization and Regularization:

- ***Hyperparameter Optimization:*** This is an important step during model selection. The same kind of machine learning model can require different constraints, weights or learning rates to generalize different data patterns. These measures are called hyperparameters, and have to be tuned so that the model can optimally solve the machine learning problem.
- ***Regularization:*** Regularization refers to the method of preventing overfitting, by explicitly controlling the model complexity. It leads to smoothening of the regression line and thus prevents overfitting. It does so by penalizing the bent of the regression line that tries to closely match the noisy data points. There are couple of techniques to achieve regularization such as L1 and L2 based on different cost functions.

  The step below shows how to get the best regularization technique as well as the best hyperparameter C selected for the LR model.

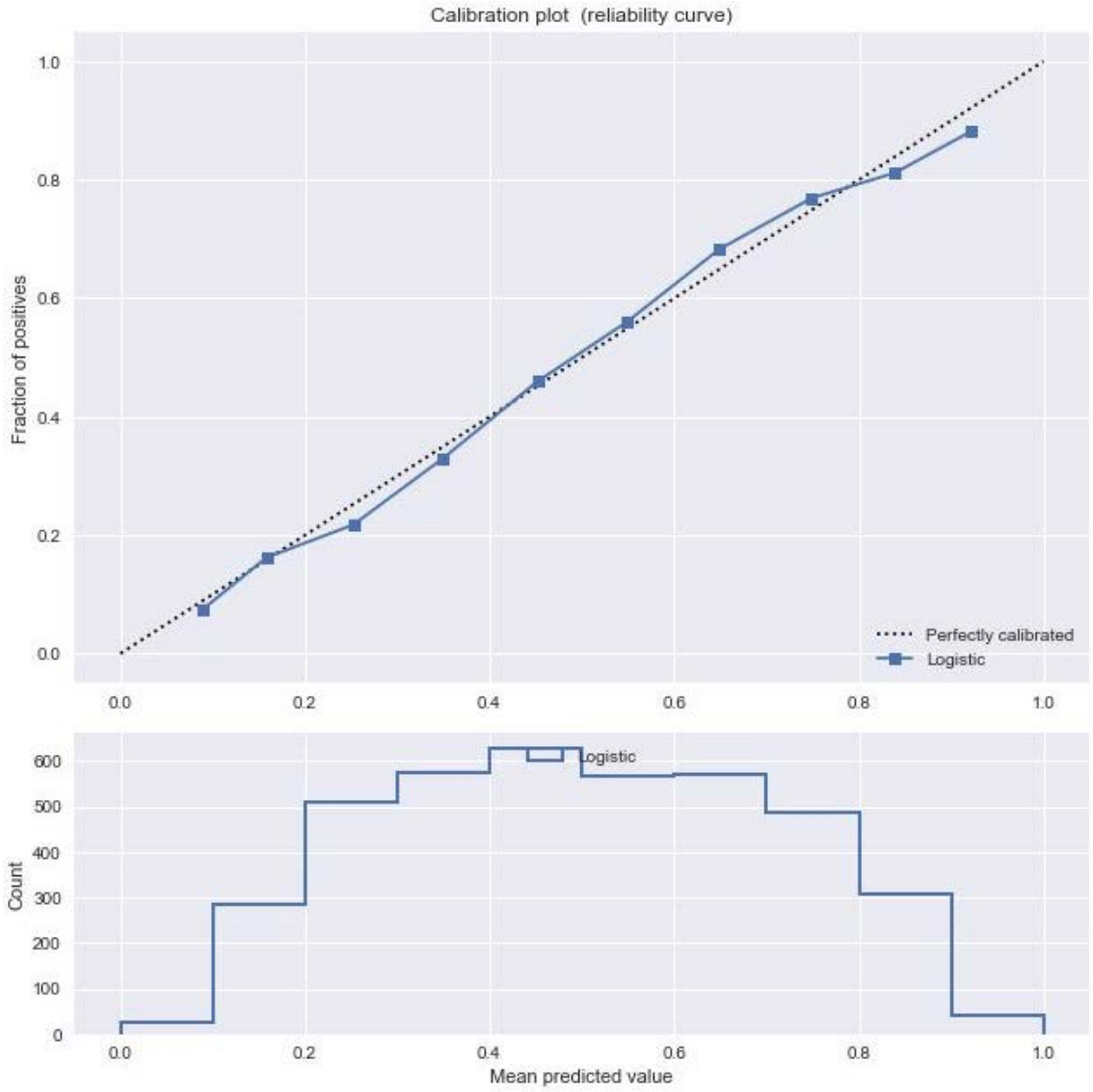
```
# Create logistic regression
lr = LogisticRegression(random_state=0)

# Create regularization penalty space
penalty = ['l1', 'l2']

# Create regularization hyperparameter space
C = np.logspace(0, 4, 10)

# Create hyperparameter options
hyperparameters = dict(C=C, penalty=penalty)

# Create grid search using 10-fold cross validation
clf = GridSearchCV(lr, hyperparameters, cv=10, verbose=0)
```

```
# Fit grid search
best_model = clf.fit(X_train[:, 1:], y_train)
```

```
# View best hyperparameters
print('Best Penalty:', best_model.best_estimator_.get_params()['penalty'])
print('Best C:', best_model.best_estimator_.get_params()['C'])
```

```
Best Penalty: l2
Best C: 1.0
```

4) Model Calibration: Since the goal of this project is to predict the probability of ever delinquency on a given loan, we want to evaluate how closely the model outcome (probability score) and the actually predicted probability. This is how calibration comes into play. It is used to improve probability estimation or error distribution of an existing model. Below is the calibration plot of Logistic Regression based on test dataset.



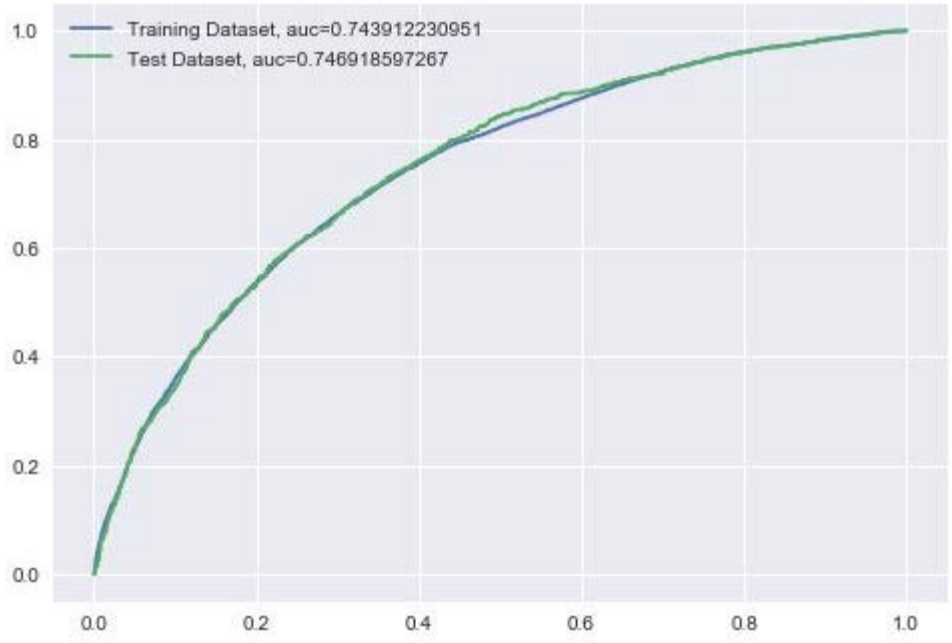
The plot tells us that the model is well-calibrated. In order words, we could use the probability score calculated from the model as the probability of target class - "ever delinquency" in this project.

3.3 Model Evaluation

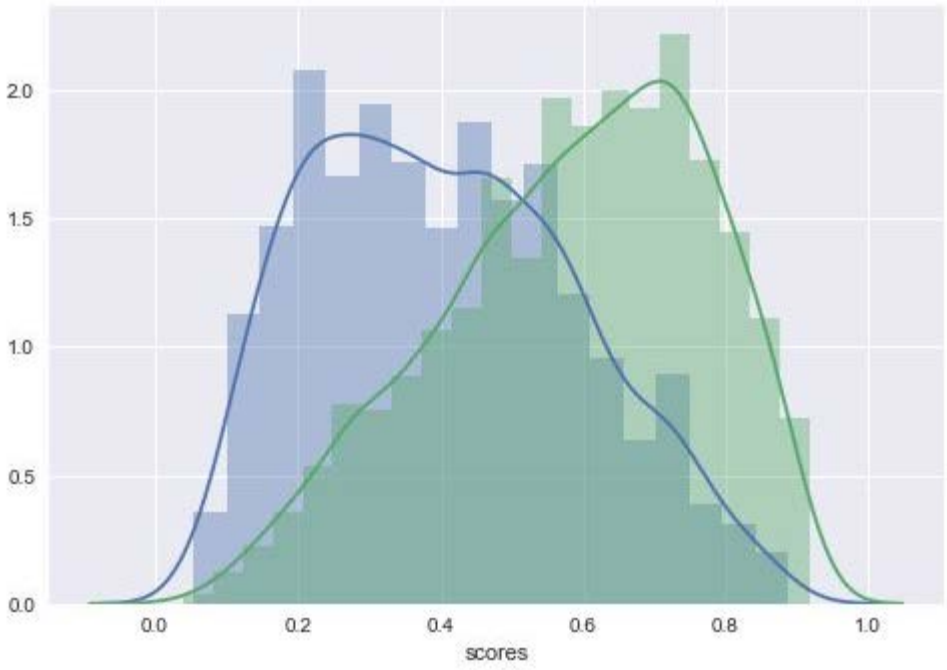
1) Model Coefficients: The coefficients are helpful to provide high level direction on what features are more sensitive and significant in terms of impacting model predictions. As shown below, we see high DTI bucket, low FICO score and high LTV bucket are likely to drive a loan getting into delinquent status. This can be verified from the business perspectives as well.

	Coefficients	Feature
0	-0.561749	loan_purp_N
1	-0.621293	loan_purp_P
2	0.248021	prop_type_MH
3	0.266889	prop_type_SF
4	0.416829	occ_stat_S
5	0.026384	dti_new_12.5
6	-0.136451	dti_new_17.5
7	0.082136	dti_new_22.5
8	0.073897	dti_new_27.5
9	0.260876	dti_new_32.5
10	0.433512	dti_new_37.5
11	0.754123	dti_new_42.5
12	0.901008	dti_new_47.5
13	-0.132746	dti_new_50.0
14	0.221742	FICO_new_650
15	-0.442506	FICO_new_670
16	-0.828015	FICO_new_690
17	-1.135426	FICO_new_710
18	-1.591250	FICO_new_730
19	-1.893020	FICO_new_750
20	-2.299009	FICO_new_770
21	-2.595515	FICO_new_790
22	-2.705185	FICO_new_800
23	0.184131	ltv_new_90
24	0.509485	ltv_new_95
25	0.750395	ltv_new_97

2) Training vs. Test ROC Curve Comparison: This step is used to evaluate if the training set and test set have similar prediction performance using the trained model which is LR in this project. The ROC curve comparison graph below shows that the model has stable outcomes on both training and test set because the two ROC curves overlap each other.



3) PDF(Probability Density Function) vs. Prediction Score : The graph below is a visual representation on how each class is separated by the probability score. For instance, based on the model outcome, if the probability score is 0.5, we have equally chance of predicting a loan going into delinquent and not going into delinquent status. As the score move towards 1, we predict a loan has higher chance of going into "delinquent" compared to "not delinquent". The overlap area of the plot explains where false categorizations fall into.



4) Business Evaluation: The key of building a model is to make the model useful and solve real world problems. In this project, the goal is to get a given loan's ever delinquent probability hence to calculate expected claim payment. Along with the calculated expected premium received, the business could evaluate if a loan is worth (making money) to insure. See calculated function below:

- **Expected Premium Received** = Premium Amount **X** Payment Frequency **X** Expected Life

of PMI
- **Expected Claim Paid** = Loan Amount **X** Coverage % **X** ***Ever Delinquent Probability*** **X** Claim Rate
- If Expected Premium Received >= Expected Claim Paid, then insure the loan
- If Expected Premium Received < Expected Claim Paid, then DO NOT insure the loan

Two assumptions here in the calculation:

- Assume PMI lasts about 3 years for every MI eligible loan - from the time the PMI is activated till it is canceled. This is based on the historical data along with applying weighted average method on loan's PMI duration.
- Assume the claim rate on a delinquent loan is 8.5%.

By applying the calculation method and assumptions above, we can conclude that 86.4% of test set population have expected premium payment exceeding expected claim payment and 13.6% of population have expected premium payment less than expected claim payment. In order words, although ACME is insuring the 13.6% of the population at the moment, it will help the company save more money if they choose not insure those loans.

As for the equivalent dollar amount, ACME is expectedly making 8.51 million insuring all the loans on its book. However, if excluding loans which have expected claim amount exceeding expected premium collected, ACME could save 0.33 million, hence expectedly making 8.84 million in total as the revenue. The model actually results in 3.8% additional revenue.

```
# Calculate expected premium paid
def expected_prem(row):
    if row['paymentplan'] == 'Monthly':
        val = row['renewal_premium'] * 12 *3
    elif row['paymentplan'] == 'Annual':
        val = row['renewal_premium'] *3
    else:
        val = row['initial_premium']
    return val

# Calculate expected claim paid
def expected_claim_paid(row):
    val = row['Initial_Loan_Amt'] * (row['coverage']/100) * row['delinquent_proba'] * 0.085
    return val

def claim_exceed_prem(row):
    if row['expected_claim_paid'] > row['expected_prem']:
        val = 1
    else:
        val = 0
    return val
```

```
df_test_new['expected_prem'] = df_test_new.apply(expected_prem, axis=1)
df_test_new['expected_claim_paid'] = df_test_new.apply(expected_claim_paid, axis=1)
df_test_new['claim_exceed_prem'] = df_test_new.apply(claim_exceed_prem, axis=1)
df_test_new['net'] = df_test_new['expected_prem'] - df_test_new['expected_claim_paid']
```

4. Conclusions

- After all the steps mentioned above, from Data Exploratory Analysis, to Model Selection, and then to Model Evaluation, it appears that Logistic Regression is the best fitted algorithm for this machine learning problem.
- Selecting multiple features instead of using only FICO score improved the model prediction

power based on the AUC score increasing from 0.65 to 0.74. This also makes business sense since there is more than 1 feature which impact the loan's ever delinquent probability prediction.
- The predictive model created in this project helps ACME determine whether it costs money to insure a mortgage loan. In other words, if the expected claim payment is more than expected premium collected, ACME is making a loss insuring the given loan.
- As a result, if applying the predictive model created in this project, ACME could potentially prevent 0.33 million dollars loss from the overall revenue 8.84 million dollars.

5. Next Steps/Future Works

- ***Include More Features:*** Since mortgage loans' delinquency status can be caused by both loan characteristics and borrower behavior, there might be additional features which help model prediction power. So one of the future works is to collect data on new features such as borrower credit history, borrower occupancy, location of the house, self-employed Y/N etc.
- ***Investigate RF and CART Performance:*** Investigate on why Random Forest and Decision Tree models underperform linear classification models in this project. Also performing parameter optimization for these models might yield better results.
- ***Redefine Target:*** In the real world, mortgage loans' have different level of delinquent status. For example, some loans might only miss one month payment or 30-day delinquent, some might miss four months payment or 120-day delinquent. Currently I am treating all level of delinquency the same, should we treat them differently? Will that improve the predicting power?
- ***Improve Assumption under Metric Calculation:*** The current assumption in calculating expected claim payment is using 8.5% as the expected claim rate applying on all ever-delinquent loans. But in reality, claim rates differ based on level of delinquency. The more payments missed or more days of delinquency a loan has, the higher chance the loan will go from delinquent status to claim.