Loan Portfolio Optimization for Nonprofit Lenders

David Sewall, Xingyao He, Brad Franklin, Susan Yu, Oscar Hernandez

ABSTRACT

Nonprofit lenders are a unique type of financial institution that focus on providing capital to individuals for socially beneficial causes rather than traditional ones such as purchasing a car, buying a home, etc. These types of lenders do not have the resources that traditional banks have such as being able to accept deposits. Therefore, optimizing a loan portfolio is of greater importance for nonprofit lenders since they are faced with greater financial constraints.

The purpose of this research is to provide a framework that nonprofit lenders can follow so that they can maximize the returns of their loan portfolio. Nonprofit lenders typically deal directly with individuals. Therefore, evaluating credit quality is done with information about the individual. This characteristic makes using peer-to-peer lending (P2P) data ideal for this research. A dataset from a popular P2P platform, Prosper, was used.

The first part of this research consists of conducting Exploratory Data Analysis on the P2P lending data to determine which features would be most useful in predicting the return of a loan. Determining these features allowed for the development of several machine learning regression models. Next, a utility function was determined that measures the expected return of a loan portfolio. Lastly, a nonlinear optimization algorithm was used to decide how to segment the loan portfolio to achieve the highest expected return. This research would provide a user the ability to input their own borrower data, risk tolerance and loan portfolio return goal to determine how to best allocate their loan capital.

Keywords: P2P Lending, Credit Scoring, Portfolio Optimization, Loan Distribution, Default Risk

---

I. INTRODUCTION

Peer-to-peer lending is an online microfinance service that allows lenders to connect directly with borrowers to minimize transaction costs that are normally collected by an intermediary like a bank or a credit union. Nonprofit lenders, a subset of lenders utilizing this platform, specialize in providing capital to individuals that work on socially beneficial causes. Since nonprofit lenders do not typically have the full resources that more common financial institutions possess (e.g. large IT budgets) and do not take on typical banking activities (e.g. accepting deposits), they need to ensure that they are able to evaluate the credit-worthiness of each of their prospective borrowers in a cost-effective manner. Furthermore, their primary goal is to maximize loan portfolio returns which will increase their ability to continue lending. The borrowers that nonprofit lenders serve vary in the scope of their projects and the communities that they impact.

The purpose of this research is to develop a model that nonprofit lenders can use to maximize their loan portfolio returns by properly allocating their capital. A successful implementation of this model would help nonprofit lenders issue more loans so that more borrowers can have a greater social impact in their communities.

The remainder of this paper is organized as follows: Section II is the literature review that analyzes existing research related to this topic. Section III goes over our methodology for this research. Section IV details our computation and analyzes the results. Section V provides an overview of what was learned and discusses next steps.

II. LITERATURE REVIEW

Researchers have worked on many related projects that can be used to help nonprofit lenders reduce risk and maximize return. In his research, Rivera (2018) examines a data set of P2P loans in Mexico and uses multivariate regression to identify the most influential identifiers of risk of default. Through this analysis, he (Rivera, 2018) finds that the strongest predictors are the payment-to-income ratio, whether someone has refinanced the terms of the loan on the same platform, loan purpose, and gender.

Rivera's research has guided some of our work in identifying variables we potentially want to study. However, there are some differences in the two projects. First, Rivera (2018) focuses only on risk of default, whereas we want to help nonprofit lenders maximize their returns on loans dispersed. Second, he (Rivera, 2018) uses a dataset that is geographically centralized only on Mexico, which can introduce cultural differences in borrowing and lending.

Regarding gender, Rivera (2018) finds that a female borrower tends to have higher loan survival rates. Other research (Miller, 2015) finds that gender is a strong determinant for higher risk loans but not all loans. However, it's possible that many of the loans in our data set are issued based on the main borrower's attributes, but the loan would fund a project for an organization whose members can be of either or both genders. Therefore, for our purposes, gender will be excluded from our variables analyzed.

Looking at the Loan Purpose variable from Rivera's (2018) research, we noticed that he categorized a variety of purposes such as home improvement, consolidating debt and education. However, his (Rivera, 2018) research does not look at loans issued for socially beneficial projects, which is the purpose of our

study. Therefore, this specific data around loan purpose isn't relevant to our analysis of nonprofit lenders.

Looking at portfolio optimization and return maximization, there have been several researchers who have created different models. When analyzing P2P data, Guo, Zhou, Luo, Liu, and Xiong (2016) used a machine learning model where loans were assessed using kernel weights and historical performance of similar loans. This model has been shown to outperform some ratings-based models that are more traditionally used (Guo et al., 2016).

III. METHODOLOGY

The decision was made to utilize a P2P lending data set from the popular platform, Prosper, to complete our research. This dataset is a good fit for the analysis because nonprofit lenders would also be working directly with individuals seeking capital for socially beneficial projects. One fault with using this data is that there are a lot of different types of borrowers represented, many of whom are not borrowing for social projects.

The analysis was divided into three sections. First, we conducted Exploratory Data Analysis on the P2P lending data set to isolate influential features affecting the success of a loan. In this first part, we used the results from our EDA to corroborate the variables identified as most important by various research papers. For example, **Table 1** outlines the variables that Chi, Ding, and Peng (2019) determined to be important. We made sure to include the DebtToIncomeRatio variable since this was identified as an important varible by Rivera (2018). Unfortunately, the other feature identified by Rivera (2018) such as whether someone has refinanced on this platform was not available in our data set. We then created a new variable, Return APR, from the data set which is the dependent variable that our research aims to maximize.

Table 1: Important features from Chi, Ding and Peng's research

Variable	Description
X_1	FICO score of the borrower
X_2	The number of inquiries of the borrower in the last 6 months
X_3	The monetary amount of the loan
X_4	The homeownership status of the borrower (0 rent, 1 = own)
X_5	The debt-to-income ratio of the borrower
X_6	The number of accounts delinquent
X_7	The number of public records in the past 10 years
Y	Dependent variable (0 = completed, 1 = default)

Second, we used Python to build several models to predict Return APR based on our variables. For this exercise, we built two different models – Gradient Boosted Regression Trees (GBRT) and Random Forest Regression (RF). For both models, we used K-Fold Cross Validation to determine their predictive accuracy on the training segment of the data set.

Third, after choosing the appropriate model from step two, we used ASPE in Excel along with the Standard Nonlinear GRG engine to build an optimization model that maximizes the expected return of a loan portfolio given one variable. Using this optimization model, nonprofit lenders would be able to

input the variable they deem important, tolerance for average portfolio variance and lowest expected loan portfolio return to determine how to properly allocate lending capital.

IV. COMPUTATION AND ANALYSIS OF RESULTS

1. Exploratory Data Analysis

The Prosper lending data set has 113,937 observations and 81 variables. Using Python, we dropped the duplicate loans, selected observations with a Prosper Score greater than 0 and removed active loans. Active loans were not of value for us since we ultimately were looking to study the % return (Return APR) from each loan. Therefore, only Completed, Defaulted and Charged Off loans were considered to build our predictive models. Lastly, we created the Return APR variable which will serve as our dependent variable (see A1 in Appendix).

Figure 1 shows the correlation heat map between variables that were strongly considered to predict Return APR. The map illustrates the strong correlation that exists between Return APR and Loan Status. This correlation made sense to us after reviewing the map output. For example, if a loan has defaulted, then we automatically know that the Return APR is 0%.

Figure 1: Correlation heat map of potential predictor variables for Return APR

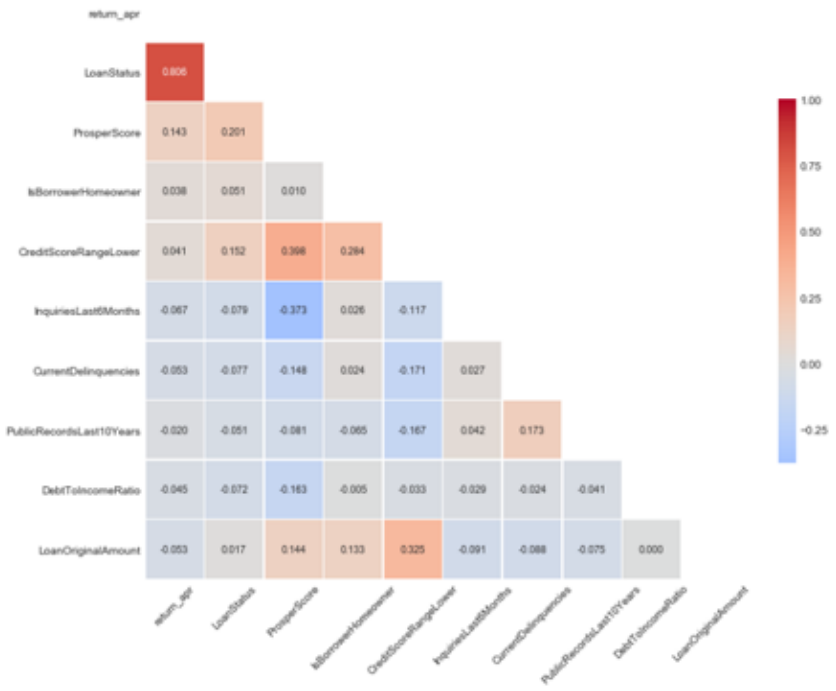
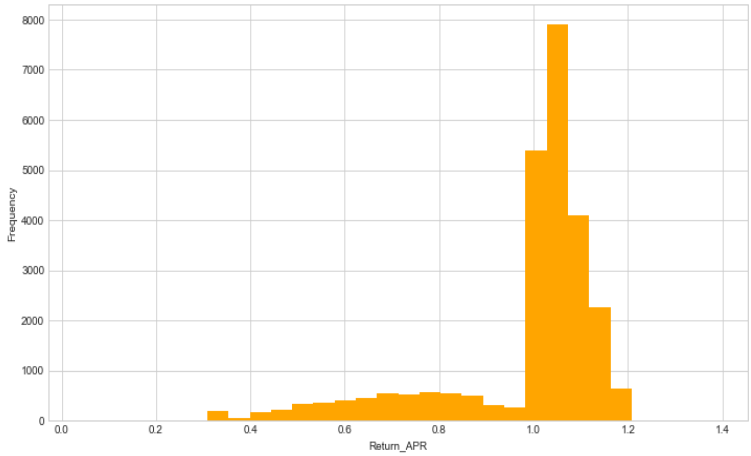


Figure 2 illustrates the distribution of our dependent variable, Return APR. The distribution highlights a large spike between 1% and 1.2% Return APR, which means that most of the loans have a positive return. Also, the negative skew of the graph highlights the wide distribution for loans that did not have a positive Return APR. Presumably, these represent the loans that have defaulted or were charged off.

Figure 2: Histogram of Return APR distribution



After reviewing previous research and finishing our EDA, we selected 8 features that we believe would be good predictors of Return APR which are outlined in **Table 2**. The original data set was reduced to 22,871 observations.

Table 2: Predictor Variables for Return APR

Variable	Variable Name
X_1	ProsperScore
X_2	IsBorrowerHomeOwner
X_3	CreditScoreRangeLower
X_4	InquiriesLast6Months
X_5	CurrentDelinquencies
X_6	PublicRecordsLast10Years
X_7	DebtToIncomeRatio
X_8	LoanOriginalAmount

The next step in our research is constructing a model that will predict Return APR given the variables listed in **Table 2**. Ultimately, the goal is to construct a loan portfolio optimization model that will allow nonprofit lender to better understand how to maximize a portfolio's return.

2. Python Predictive Models

In this section, our goal was to predict the Return APR for each loan in the newly defined data set. Therefore, the decision was made to build two different machine learning models – Gradient Boosted Regression Trees (GBRT) and Random Forest Model (RF). These models were created using GradientBoostingRegressor and RandomForestRegressor from scikit-learn in Python. Scikit-learn is a popular machine learning library that allows for easy implementation of models with the ability to do extensive hyperparameter tuning if necessary. Furthermore, these models allow us to load in the data set without having to complete any regularization and are noted for their ability to avoid overfitting a data set.

Next, we used the cross_val_score function from scikit-learn to compute accuracy scores (RMSE) for each model. Specifically, this function allowed us to use k-fold cross-validation (eight folds) to compute

eight different Root Mean Squared Errors for each of our models. Both models performed similarly on the training set. We were comfortable with the RMSE scores on the training set because they were less than 0.25 which was our consensus cutoff value that would alert us to conduct more hyperparameter tuning. Both models were then used to make predictions on the test set and compute one final RMSE calculation. Finally, we outputted the feature importance using the feature_importances_ attribute from each model (**Table 3**).

Table 3: GBRT Model (left) and RF Model (right) Feature Importance

	Feature Importance
LoanOriginalAmount	0.272029
DebtToIncomeRatio	0.183021
InquiriesLast6Months	0.144708
CurrentDelinquencies	0.121222
ProsperScore	0.114102
CreditScoreRangeLower	0.079409
PublicRecordsLast10Years	0.045554
IsBorrowerHomeowner	0.039955

	Feature Importance
DebtToIncomeRatio	0.291816
LoanOriginalAmount	0.175867
CreditScoreRangeLower	0.159747
InquiriesLast6Months	0.118147
ProsperScore	0.111071
CurrentDelinquencies	0.055467
PublicRecordsLast10Years	0.050751
IsBorrowerHomeowner	0.037134

Table 4 displays the RMSE scores for the GBRT and RF models. The testing and training accuracy for the RF model was slightly worse than that of the GBRT model. This was an expected result given that the GBRT model used an optimized number of decision trees as one of its hyperparameters. **Table 3** illustrates that the most important feature in the GBRT model is LoanOriginalAmount, whereas the most important feature in the RF model is DebtToIncomeRatio. This was an interesting observation but it's also worth noting that both models included it in their respective top three features overall.

Table 4: Training and Testing Model Accuracy

Model	Training Accuracy (Avg.8 Folds)	Testing Accuracy
GBRT	0.166	0.163
RF	0.174	0.171

After reviewing the accuracy scores and the feature importance for each model, we determined that the RF model is the best one. The RF model performed similarly as the GBRT model with respect to RMSE. Furthermore, the RF model listed DebtToIncomeRatio as its most important feature. Intuitively, this made more sense to us as a better predictor of Return APR when compared to LoanOriginalAmount. Our consensus thought was that knowing the amount of debt to income for a borrower would provide more information in their ability to pay back a loan. The secondary research that we conducted as part of the Literature Review validates that the DebtToIncomeRatio feature has been instrumental in predicting loan default (i.e. a proxy dependent variable for Return APR).

1. Loan Portfolio Optimization

The next step in our research involved using the RF model to predict Return APR for our entire data set. This was a necessary step to set up our loan portfolio optimization model correctly. Our goal was to build a model that considers one feature (e.g. LoanOriginalAmount, DebtToIncomeRatio) to create bins that nonprofit lenders will use to segment their lending capital. The decision was made to create four

bins. The bins were based on the DebtToIncomeRatio variable since this was deemed our most important feature. These bins contain the average of all the loans that fell within a given DebtToIncomeRatio threshold (Bin 1: <0.13, Bin 2: ≥ 0.13 and <0.2, Bin 3: ≥ 0.2 and <0.3, Bin 4: ≥ 0.3). **Table 5** shows the average predicted Return APR for each bin. The table can be interpreted as, when grouped by DebtToIncomeRatio, the average predicted return for loans in all those bins is negative (i.e. losing money).

Table 5: Average Predicted Return APR for each bin

Bins	Avg. Predicted Return APR
1	0.985
2	0.985
3	0.979
4	0.96

As discussed earlier, our goal is to develop a model that would provide the optimal bin proportions of a loan portfolio which results in the highest Return APR possible. In other words, given any capital amount (e.g. $100,000) available to lend, how much of that capital amount should a nonprofit lender allocate to each bin (e.g. 25% in Bin 1, 12.5% in Bin 2, etc.). **Table 6** describes all the necessary variables needed to construct our optimization model.

Table 6: Description of optimization model variables

Variable Name	Description
$p_k (\forall i \in \{1,2,3,4\})^{(1)}$	Each bin's proportion as a %
r_k	Average predicted return for bin k
σ_{kj}	Covariance for all combinations of bins

The average loan portfolio variance (Risk) can be written as:

$$\sigma^2 = \sum_j \sum_k p_j \, p_k \, \sigma_{kj}$$

(1) p_J takes on the same values as p_k; it is not a separate variable; it is used here to illustrate all combinations of bins

Therefore, the objective function can be written as:

$$\text{MAXIMIZE } R = \sum_{k=1}^{4} p_k \, r_k$$

where R equals average return of the loan portfolio

Lastly, we defined the constraints of the model in **Table 7**. The values of the constraints were decided by group consensus based on secondary research and realistic goals.

Table 7: Description of constraints

Description
Average loan portfolio variance (σ^2) <= 0.125
Average loan portfolio return (R) >= 0.025
Sum of all decision variables (p_i) = 1

After executing the model in ASPE (see Appendix A2 for Excel screenshots), we determined that approximately 22% of the portfolio should be allocated to Bin 3 and approximately 73% should be allocated to Bin 4. This resulted in a maximum loan portfolio return of 0.96%. Since the Return APR is normalized to 1, the optimal return % that resulted means this loan portfolio had a loss of 0.04%.

V. CONCLUSION AND NEXT STEPS

The results of this project indicate that we were not able to optimize a positive Return APR using only the DebtToIncomeRatio variable. Although debt-to-income is an effective predictor of Return APR, it's not the only one. Moving forward, we can use multiple variables to generate a more robust model. Furthermore, Return APR has a significant negative skew (as shown in the **Figure 2**). The loans on the left side of the distribution, which represent default/charged off loans, dragged down the average Return APR significantly. As such, one key learning is that minimizing return loss may be more important than maximizing return for nonprofit lenders.

When considering further research, there are a few methods that can potentially improve the entire process. First, we can use a data set of nonprofit or socially conscious borrowers. This kind of dataset may contain information about the size of the organization the loan is associated with and the type of project the loan is being used to fund. All of those can be potentially important predictors that we did not have access to in the P2P data set from Prosper. Also, we can change our predictive models to predict Return APR based on the performance of similar loans in the past. This methodology was used successfully by Guo et al. (2016) and should be tested on a data set of nonprofit borrowers. Overall, the results were not as expected but it does provide a preliminary framework to continue studying the topic.

References

Chi, G., Ding, S., & Peng, X. (2019). Data-driven robust credit portfolio optimization for investment decisions in P2P lending. *Mathematical Problems in Engineering, 2019*, 1-10.

Guo, Y., Zhou, W., Luo, C., Liu, C., & Xiong, H. (2016). Instance-based credit risk assessment for investment decisions in P2P lending. *European Journal of Operational Research, 249*(2), 417-426.

Miller, S. (2015). Information and default in consumer credit markets: Evidence from a natural experiment. *Journal of Financial Intermediation, 24*(1), 45-70.

Rivera, C. (2018). Determinants of Default in P2P Lending: The Mexican Case. *Independent Journal of Management & Productions, 9*(1), 1-24.

Appendix

A1. Calculate Return APR

In [25]:
```
#Calculate returned amount, which is the principle amount plus interest received minus fees
loans['return_amount'] = (
    loans.LoanOriginalAmount +
    loans.LP_InterestandFees -
    loans.LP_ServiceFees -
    loans.LP_CollectionFees -
    loans.LP_NetPrincipalLoss)

#Calculate percent return
loans['return_pct'] = (
    loans.LoanOriginalAmount +
    loans.LP_InterestandFees -
    loans.LP_ServiceFees -
    loans.LP_CollectionFees -
    loans.LP_NetPrincipalLoss)/(
    loans.LoanOriginalAmount)

#Calculate annualized percent return
loans['return_apr'] = (
    loans.return_pct**
    (12/loans.Term))

#limit to only loans with valid return data
loans = loans[loans.return_apr > 0]
```

A2. Optimization Model

	Debt to Income Ratio Bin			
Values	1	2	3	4
Average of pred_return_apr	0.98500	0.98500	0.97964	0.96033
StdDev of pred_return_apr	0.12046	0.11554	0.12235	0.13671
Var of pred_return_apr	0.01451	0.01335	0.01497	0.01869

Legend	
Bin 1	[0,0.13)
Bin 2	[0.13,0.2)
Bin 3	[0.2,0.3)
Bin 4	[0.3, 999)

Covariance	1	2	3	4	Proportions, p_j
1	0.01451	0.74210	0.66334	0.40098	0.00000
2	0.74210	0.01335	0.60235	0.36261	0.00000
3	0.66334	0.60235	0.01497	0.32699	0.22199
4	0.40098	0.36261	0.32699	0.01869	0.77801
Proportions, p_k					
Decision Variables	0.000%	0.000%	22.199%	77.801%	1.00

$p_k p_j \sigma_{kj}$	0.00000	0.00000	0.00000	0.00000
	0.00000	0.00000	0.00000	0.00000
	0.00000	0.00000	0.00074	0.05648
	0.00000	0.00000	0.05648	0.01131

				Constraints	
Portfolio Variance	RISK	0.1250		Risk Ceiling	0.125
Portfolio Return	RETURN	0.9646		Return Floor	0.025

A3. Training/Test Split

In [34]:
```
#Create training and validation data sets

#Split loans_df into X and y variables
X = loans_df.drop("return_apr", axis =1)
y = loans_df["return_apr"].copy()

#Split into training and validation sets
X_train, X_val, y_train, y_val = train_test_split(X,y, random_state =10)
```

A4. GBRT Model

In [37]:
```
#Build Gradient Boosted Regression Trees (GBRT)
gbrt = GradientBoostingRegressor(max_depth=2, n_estimators=120, random_state =50)
gbrt.fit(X_train, y_train)

errors = [mean_squared_error(y_val, y_pred)
          for y_pred in gbrt.staged_predict(X_val)]
bst_n_estimators = np.argmin(errors)

gbrt_best = GradientBoostingRegressor(max_depth=2,n_estimators=bst_n_estimators, random_state = 50)
gbrt_best.fit(X_train, y_train)
```

A5. RF Model

In [45]:
```
#Build Random Forest Regression Model (RF)
rf_reg = RandomForestRegressor(n_estimators=500, max_features=6, bootstrap = True, n_jobs=-1, random_state = 99)
rf_reg.fit(X_train, y_train)
```

A6. K-Fold Cross Validation and Test Set Accuracy (GBRT)

In [38]:
```
#Use K-Fold CV on the training data set
gbrt_scores = cross_val_score(gbrt_best, X_train, y_train,
                              scoring="neg_mean_squared_error", cv=8)

gbrt_rmse_scores = np.sqrt(-gbrt_scores)
```

In [39]:
```
#Create function to display the RMSE score statistics for each model
def display_scores(scores):
    print ("Scores:", scores)
    print ("Mean:", scores.mean())
    print ("Standard Deviation:", scores.std())
```

In [40]:
```
print("GBRT RMSE for Training Data Set")
display_scores(gbrt_rmse_scores)
```

```
GBRT RMSE for Training Data Set
Scores: [ 0.16930808  0.16432865  0.17095977  0.1697881   0.16397783  0.16950068
  0.15682411  0.1629212 ]
Mean: 0.165951050952
Standard Deviation: 0.0045095228334
```

In [41]:
```
#Make predictions using GBRT model using test set
gbrt_predictions = gbrt_best.predict(X_val)
```

In [42]:
```
#Output test set RMSE scores for GBRT model
gbrt_test_mse = mean_squared_error(y_val, gbrt_predictions)
gbrt_test_rmse = np.sqrt(gbrt_test_mse)
print("gbrt_test_rmse:", gbrt_test_rmse, "\n")
```

```
gbrt_test_rmse: 0.163085160765
```

A7. K-Fold Cross Validation and Test Set Accuracy (RF)

In [46]:
```
#Use K-Fold CV on the training data set
rf_scores = cross_val_score(rf_reg, X_train, y_train,
                                scoring="neg_mean_squared_error", cv=8)

rf_rmse_scores = np.sqrt(-rf_scores)
```

In [47]:
```
print("RF RMSE for Training Data Set")
display_scores(rf_rmse_scores)
```

```
RF RMSE for Training Data Set
Scores: [ 0.17654976  0.17126966  0.17709611  0.17682814  0.17141403  0.17761182
  0.16736773  0.1701328 ]
Mean: 0.173533755628
Standard Deviation: 0.00368232243503
```

In [48]:
```
#Make predictions using RF model using test set
rf_predictions = rf_reg.predict(X_val)
```

In [49]:
```
#Output test set RMSE scores for RF model
rf_test_mse = mean_squared_error(y_val, rf_predictions)
rf_test_rmse = np.sqrt(rf_test_mse)
print("rf_test_rmse:", rf_test_rmse, "\n")
```

```
rf_test_rmse: 0.171205051469
```

A7. GBRT Feature Importance

In [43]:
```
#Create object to hold feature importance for GBRT model
gbrt_feature_importances = pd.DataFrame(gbrt_best.feature_importances_,
                                    index = X_train.columns,
                                     columns=['Feature Importance']).sort_values('Feature Importance', ascending=False)
```

In [44]:
```
#Print out the feature importance from the GBRT model
gbrt_feature_importances
```

A8. RF Feature Importance

In [50]:
```
#Create object to hold feature importance for RF model
rf_feature_importances = pd.DataFrame(rf_reg.feature_importances_,
                                    index = X_train.columns,
                                     columns=['Feature Importance']).sort_values('Feature Importance', ascending=False)
```

In [51]:
```
#Print out the feature importance from the RF model
rf_feature_importances
```

A9. Flag Variables

In [23]:
```
#Drop duplicate listing keys
loans = loans.drop_duplicates(subset='ListingKey', keep='last')

#Remove loans pre-2009 (those with a Prosper score of 0)
loans = loans[loans.ProsperScore > 0]


#Remove active loans since those are not necessary for building models
loans = loans[(loans.LoanStatus == 'Completed') |
              (loans.LoanStatus == 'Chargedoff') |
              (loans.LoanStatus == 'Defaulted') ]

#Create flag for Completed=1, Defaulted and Chargedoff = 0
loans.replace(['Completed', 'Defaulted', 'Chargedoff'],
                        [1, 0, 0],
                        inplace=True)
```

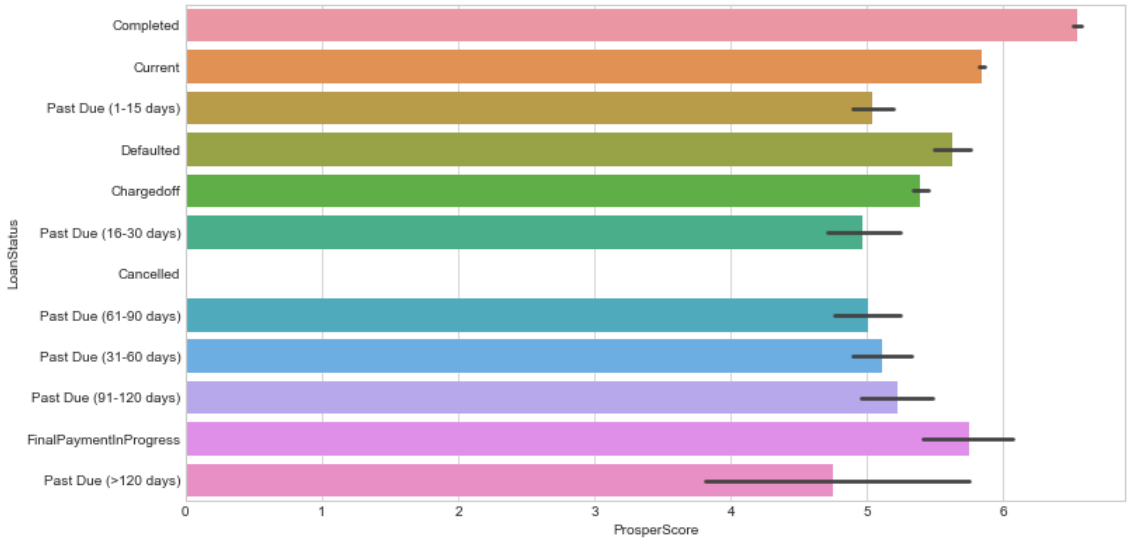
A10. Heat Map

In [36]:
```
#Create correlation heat map
loans_features = pd.DataFrame([loans.return_apr,
    loans.LoanStatus,
    loans.ProsperScore,
    loans.IsBorrowerHomeowner,
    loans.CreditScoreRangeLower,
    loans.InquiriesLast6Months,
    loans.CurrentDelinquencies,
    loans.PublicRecordsLast10Years,
    loans.DebtToIncomeRatio,
    loans.LoanOriginalAmount,]).transpose()

corr = (loans_features.corr())
def corr_chart(df_corr):
    corr=df_corr.corr()
    #screen top half to get a triangle
    top = np.zeros_like(corr, dtype=np.bool)
    top[np.triu_indices_from(top)] = True
    fig=plt.figure()
    fig, ax = plt.subplots(figsize=(12,12))
    sns.heatmap(corr, mask=top, cmap='coolwarm',
        center = 0, square=True,
        linewidths=.5, cbar_kws={'shrink':.5},
        annot = True, annot_kws={'size': 9}, fmt = '.3f')
    plt.xticks(rotation=45) # rotate variable labels on columns (x axis)
    plt.yticks(rotation=0) # use horizontal variable labels on rows (y axis)
    plt.title('Correlation Heat Map')
    plt.savefig('plot-corr-map.pdf',
        bbox_inches = 'tight', dpi=None, facecolor='w', edgecolor='b',
        orientation='portrait', papertype=None, format=None,
        transparent=True, pad_inches=0.25, frameon=None)

corr_chart(df_corr = loans_features)
plt.show()
```

A11. Prosper vs. LoanStatus

In [8]:
```
sns.set_style("whitegrid")
sns.set_context({"figure.figsize": (12, 6.5)})
sns.barplot(x="ProsperScore", y="LoanStatus", data=loans)
```

Out[8]: <matplotlib.axes._subplots.AxesSubplot at 0x1c7b4461d30>



A12. Cross Table of IsBorrowerHomeowner vs. LoanStatus

In [17]:
```
loans.groupby(['IsBorrowerHomeowner', 'LoanStatus',])['LoanStatus'].size().unstack(fill_value=0)
```

Out[17]:

LoanStatus	Cancelled	Chargedoff	Completed	Current	Defaulted	FinalPaymentInProgress	Past Due (1-15 days)	Past Due (16-30 days)	Past Due (31-60 days)	Past Due (61-90 days)	Past Due (91-120 days)	Past Due (>120 days)
IsBorrowerHomeowner												
False	3	6661	19794	26098	2744	95	429	137	179	134	177	8
True	2	5331	18280	30478	2274	110	377	128	184	179	127	8