Developing a Credit Spread Trading Strategy using Tree-Based Machine Learning Methods

Abstract
We propose a trading strategy on Investment Grade Corporate Bond ETFs based on a signal of the credit spread increasing or decreasing. Using various machine learning methods ranging from a single classification tree to a whole ensemble of black-box models, we aim to fit models that generalize well to out-of-sample data. By using a time-series oriented forward chaining cross-validation scheme, we tune our various models accordingly. Additionally, we never use future information to predict past information. Our results indicate modest accuracy improvements relative to a baseline Logistic LASSO model. Further, by implementing a simple long-short strategy that is self-financing with various Bond ETFs, we show it is possible to achieve very appealing returns for the risk taken over a three year time period in out-of-sample testing.

Contents

1. Introduction

Credit spreads for corporate bonds have long been viewed as a fundamental dynamic, where investors are compensated for taking additional credit risk relative to default-free government securities. But, Longstaff (2005) has argued against attributing default risk to existing spreads. Premiums demanded for high grade bonds cannot be explained solely by differences in default risk – many corporations hold capital reserves which more than cover for unlikely credit market events. We branch out from this fundamental view and explore credit spreads as a picture of two moving components, focusing on factors affecting treasury notes and corporate bonds on individual bases. Our approach integrates both global and endogenous factors that encompass equity market risk and signal future corporate performance.

There has been growing popularity in utilizing machine learning methodologies for financial time series prediction. Various leading investment management firms have began investing resources either in new hires or on data competitions. Two Sigma has recently finished hosting a stock prediction competition on Kaggle using news and market data. Citadel routinely hosts datathons across major US universities scouring for talent in statistics and machine learning. AQR recently hired Marcos Lopez De Prado as head of Machine Learning, and in fact in 2018, they published a white paper detailing the estimation of equity risk premia using machine learning methodologies.

The ability to accurately predict credit spreads using machine learning is extremely valuable for multiple reasons. Credit spreads tend to signal the health of an economy. When spreads are tight, it signals economic growth and prosperity. The opposite is true if wide credit spreads are observed. Having accurate predictions of credit spreads corresponds to possible trading strategies that can be quite profitable. Given predictions on whether credit spreads will widen or tighten, one can formulate a simple trading strategy as follows:

- For widening credit spreads, long an ETF that tracks the top end (i.e. the BAML AAA corporate bond index) and short an ETF that tracks the bottom end (10 year treasuries).
- For tightening credit spreads, do the opposite trades

Our paper is structured as follows: in section 2, we introduce a wide range of features that have been collected and generated for this project. In sections 3 and 4, we describe the machine learning models used on this data and report the results. In section 5, we expand on how we use the predictions to create a profitable trading strategy. Lastly, in section 6, we summarize our findings.

2. Data and Methodology

2.1 Variables

To represent the credit spread, we used 10 year constant-maturity US treasuries and the ICE BAML US Corporate Effective Yield. The BAML index tracks the performance of US investment-grade rated corporate debt issued in the US market.

Because the credit spread consists of two moving parts, it is important to consider what factors cause each yield to move separately and what can cause the spread to narrow or widen. Therefore, we incorporated new features that can roughly be characterized into three groups:

Macroeconomic The risk free rate can be viewed as a function of crucial economic variables, as lower maturity treasury notes and bills are disproportionately influenced by movements in the federal funds rate (FFR). The FFR is also determined by macroeconomic indicators paramount to FOMC decisions. The macroeconomic variables we consider are:

- NAIRU (non-accelerating inflation rate of unemployment) - a key metric that illustrates levels of economic activity and consumer spending.
- Unemployment rate (including discouraged workers) - provides information about the state of the economy, and how easy it is to find jobs.
- Balance of Payments - a measure of US export strength and weakness
- Manufacturing - captures the state of the US economy
- Gold Prices - often viewed as a safe asset class alternative to treasuries for investors to flee towards in times of crisis.

Interest Rate Since we are predicting corporate yield spreads, it is important to try to capture information from the term structure of interest rates. For example, an inverted yield curve can often signal impending market downturns. Thus, the term structure of US treasuries is incorporated using various "DGS" (constant maturity treasury yield) variables from the FRED database. A time series for each of 1, 2, 5, 7, 10, 20 year maturities are included to capture as much of the trends as possible. The 10 year treasury yield is inherently accounted for in our spread variables, but we also generate various lag features on these variables.

Additionally, we also look at the TED spread, which is the difference between the three month treasury and three month libor rates. This helps us capture additional macroeconomic and interest rate information from international fixed income markets. The LIBOR and US treasuries are often used for pricing of more complex financial derivative products, and thus the difference of the two is an important predictive variable.

Foreign Exchange Prevailing economic theory such as Uncovered Interest Rate Parity suggests there should be an empirical relationship between exchange rates and interest rates. Given exchange rate fluctuations, foreign investors will see different levels of attractiveness in investing in US corporate bonds. Additionally, the selected exchange rates reflect countries that the US engages heavily in trade with. All of these factors are also important for interest rate decisions made by the FED.

Stock Market One motivating factor is evidence by Kwan (1996) and Gebhardt et al. (2005b) suggesting a strong relationship between past stock market returns and future bond returns for a given company. We include various metrics of the SP for proxying equity returns, used as an indicator of aggregate corporate bond performance. In addition, corporate bonds being correlated with both the stock and bond markets hint that yields have exposure to liquidity shocks in either asset class. Thus, we have incorporated the daily volume of the SP as a measure for corporate bond appetite. We also added VIX, the implied volatility of the SP.

2.2 Generated Features

Our lag features all consist of looking backwards in time for 3 rolling windows; 5,20 and 60 trading days back in time. For each rolling window, we compute simple moving averages, exponential moving averages, and standard deviations for the chosen features. Additionally, we also process sharpe and tail ratio features for the spread variable looking 60 trading days back in time. The calculations are as follows:

Let X_t denote a variable at time t. For $w \in \{5, 20, 60\}$

Simple Moving Average

$$MA = \frac{\sum_{i=1}^{w} X_{t-i}}{w}$$

Exponential Moving Average

$$EMA = \sum_{i=1}^{w} (1 - \frac{2}{w+1})^{i-1} X_{t-i}$$

For y_t denote the spread at time t.

Sharpe Ratio of Spread

$$Sharpe = \frac{\frac{1}{60}\sum_{i=1}^{60} y_{t-i}}{\sqrt{\frac{1}{60-1}\sum_{i=1}^{60}(y_{t-i} - \frac{1}{60}\sum_{i=1}^{60} y_{t-i})^2}}$$

Tail Ratio of Spread (Looking 60 days back)

$$Tail = \frac{\text{95th percentile value}}{|\text{ 5th percentile value }|}$$

Investors tend to view bonds as substitute investments when the stock market is performing poorly. By generating additional features from the SP variables, we extract market trends/signals and incorporate this information into our models. In particular, we implement real quant trading alpha signals from Kakushadze (2016) on the SP variables along with the same lag features previously described.

Let $\mu(x,t)$ be the trailing mean function looking t day back for variable x. Let $\rho(x,y,t)$ denote the rolling correlation looking t days back in time for variables x and y. Let $\phi(x,t)$ be the time series rank function, such that looking t days back for the variable x, return the rank of the computed value which can range from 1 to t. Further, let $min(X,t)$ and $max(x,t)$ denote the historical min/max in the last t days for variable x. Given Open, Close and Volume for some underlying X_t at time t, we compute the following alpha signals:

Alpha 6

$$-1 * \rho(X_t(Open), X_t(Volume), 10)$$

Alpha 7

$$\begin{cases} -\phi(X_t(Close) - X_{t-7}(Close), 60) \\ *sign(X_t(Close) - X_{t-7}(Close)) \\ \\ \text{ for } \mu(X_t(Volume), 20) < X_t(Volume) \\ \\ -1 \text{ otherwise} \end{cases}$$

Alpha 9

$$\begin{cases} X_t(Close) - X_{t-1}(Close) \\ \\ \text{for } min(X_t(Close) - X_{t-1}(Close), 5) > 0 \\ \\ \text{or } max(X_t(Close) - X_{t-1}(Close), 5) < 0 \\ \\ -1 * (X_t(Close) - X_{t-1}(Close)) \text{ otherwise} \end{cases}$$

Alpha 12

$$-(X_t(Close) - X_{t-1}(Close))$$

$$*sign(X_t(Volume) - X_{t-1}(Volume))$$

A summary of all of the variables can be found in Table 1.

2.3 Response Variable

Instead of predicting a value for the spread in the next trading day, we decide to predict whether the spread will increase or decrease in the next trading day. Thus, this becomes a binary classification problem.

2.4 Validation Strategy

Our data ranges from 1998 to 2019. We hold out the last three years of data to use as a test set. With the rest of the data, we use cross-validation to evaluate the predictive power of various models and fine-tune each model's hyper-parameters. Traditional cross-validation techniques such as K-fold do not typically work well on time-series data, as data is often serially correlated. Additionally, data leakage must be prevented.

We use a rolling cross-validation technique [?] that ensures that each training set consists only of observations occurring prior to each test set observation. Therefore, no future information is used to train the models. Figure 1 shows how the data is partitioned. The cross validation accuracy is computed by averaging across all folds.

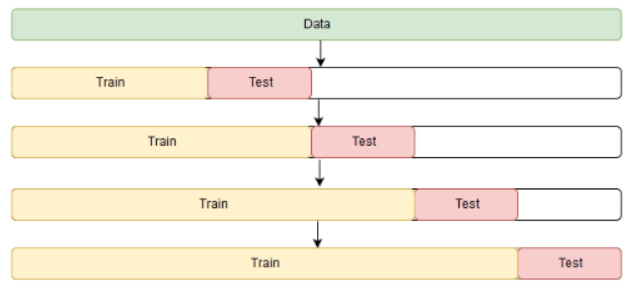

Figure 1. Cross-Validation Scheme for Time Series

3. Models

The goal of a tree-based method is to predict the value of a target variable by learning simple decision rules inferred from the data features. Tree-based methods are flexible, relatively interpretable, and have great predictive power. Because of these aspects, we decided to compare a series of tree-based models ranging in their black-box nature.

3.1 Classification Tree

Classification trees take a top-down, greedy approach to searching for the optimal split at each step of the tree. Using the CART algorithm, we select the feature and threshold that yields the largest information gain at each node.

Let data at node m be represented by Q. For each candidate split $\theta = (j, t_m)$, where j is the feature and t_m is the threshold, partition the data in to $Q_{left}(\theta)$ and $Q_{right}(\theta)$ based on the threshold.

Impurity measures how well the two response classes are separated. Ideally, want nodes to perfectly separate the 0 and 1 class; in this case, the impurity would be 0.

In CART, we use the GINI index to represent impurity. Thus, the impurity of the node is:

$$G(Q,\theta) = \frac{n_{left}}{N_m} H(Q_{left}(\theta)) + \frac{n_{right}}{N_m} H(Q_{right}(\theta))$$

Then we select the parameters to minimize impurity.

$$\theta^* = argmin_\theta G(Q,\theta)$$

Table 1. Variables
(*) indicates that MA, EWA, STD were generated on the variable

Type of Variable					
Spread	**Macroeconomic**	**Interest Rate**	**Foreign Exchange**	**Stock Market**	**Generated**
Risk-Free Rate	NAIRU	DSG (*)	USD/CAD (*)	S&P Open (*)	Alpha 6
BAML index	Balance of Payments	(1, 2, 5, 7, 10, 20 year	USD/EURO (*)	S&P Close (*)	Alpha 7
Spread Lag 1	Manufacturing	constant maturities)	USD/JPY (*)	S&P Adj Close	Alpha 9
Spread Lag 2	Unemployment Rate	TED Rate (*)	USD/GBP (*)	S&P Volume (*)	Alpha 12
Spread Lag 3	Gold Prices (*)			S&P High (*)	Spread Sharpe Ratio(*)
				S&P Low (*)	Spread Tail Ratio (*)
				VIX (*)	

In this fashion, we continue splitting the tree until the maximum depth is reached. To prevent overfitting as much as possible, we fine-tune the maximum depth and minimum sample size of a split and leaf to encourage shorter trees. Decision trees are highly unstable so we incorporate a minimum impurity decrease parameter; we only allow a node to be split if the reduction in impurity is at least as large as the parameter.

3.2 Ensemble Methods

A single classification tree is prone to overfitting and forming biased models. To improve on the decision tree, we apply a series of ensemble methods that combine individual models and incorporate features such as bagging and boosting.

Bagging Bagging methods seek to build several independent estimators and then average their predictions. This reduces the variance of the combined estimator.

3.2.1 Random Forest

A random forest has several key differences from a decision tree classifier. Firstly, when splitting a node, we no longer select from the best possible split among all features. Instead, we choose a random subset of features and pick the best split from that smaller group. Moreover, each tree is built from a bootstrap sample of the training set, with data points drawn with replacement.

The individual classifiers are combined by averaging their probabilistic predictions.

We tune the same parameters as the classification tree, but add an additional parameter, *number of estimators* that controls the number of trees in the forest. Increasing the number of trees helps the classifier, but the generalization ability decreases past a center point.

Boosting Boosting methods try to reduce the bias of the combined estimator by combining several weak models.

3.2.2 Gradient Tree Boosting

Gradient Tree Boosting generalizes boosting to arbitrary differentiable loss functions. They have good predictive power and are robust to outliers in output space.

The gradient boosting classifier is an additive model using classification trees of a fixed size as weak learners $h_m(x)$.

$$F(x) = \sum_{m=1}^{M} \gamma_w h_m(X)$$

When the $m-th$ learner is added to the model, it becomes:

$$F_m(x) = \sum F_{m-1}(x) + \gamma_m h_m(x)$$

Each new learner tries to minimize the loss L, given the previous ensemble of trees $F_{m-1}(x)$. Thus

$$h_m = \underset{h}{argmin} \sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) + h(x_i)$$

The optimization problem can be solved using gradient descent. For our classification problem, we use the negative binomial log-likelihood loss function. The main parameters we tune are the number of learners as well as the learning rate. The learning rate is a regularization parameter and scales back the step length in gradient descent. There is a trade-off between number of learners and the learning rate, as a smaller learning rate typically requires more weak learners.

3.2.3 XGBoost and LightGBM

XGBoost and LightGBM are the two most black-box models we consider. These algorithms have been dominating Kaggle competitions over the past few years because of their incredible predictive power.

XGBoost stands for Extreme Gradient Boosting. XGBoost implements gradient boosted decision trees, but with deep considerations in terms of systems optimization in order to provide scalable, portable, and accurate predictions. Besides traditional gradient boosting, the algorithm includes stochastic gradient boosting with sub-sampling at the row, column, and column per split levels, and regularized gradient boosting with both L1 and L2 regularization.

The training objective is to minimize the following objective function:

$$Obj = L + \Omega$$

L is the loss function, which in this case is binary classification log loss:

$$L = -\frac{1}{N}\sum_{i=1}^{N}(y_i log(p_i)) + (1-y_i)log(1-p_i)$$

Ω is the regularization term, which controls the complexity of the model.

$$\Omega = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$$

where T is the number of leaves and w_j^2 is the score on the $j-th$ leaf.

LightGBM is another black-box gradient boosting method. It boasts a reduced cost of calculating many of the tree computations and overall memory usage. While traditional tree-based methods are top-down (growing by level), LightGBM grows tree leaf-wise, choosing the leaf with the maximum reduction in loss to grow.

Because neither model appears to have an advantage of the other, we decided to combine LightGBM and XGBoost into one classifier. We hope that even by averaging the predictions for these two models, that we can get a lower-variance, more stable estimator.

4. Model Results

The results for all of the models can be found in table 2.

We create a "ensemble model" that averages the predictions for Decision Tree, Gradient Boosting, and XGBoost & LightGBM.

We use a regularized logistic regression model as a baseline to compare against. Because of the number of features we are considering, using a LASSO penalty helps with feature selection and prevent overfitting. This results in a simple but effective baseline model, that yielded a 59.6% test set accuracy and test set Logloss of 0.696.

For our other models, we can see that the random forest did not perform too well on the test set which could be a sign of overfitting. The Decision Tree model performed surprisingly well for again a relatively simple model, though it did not beat the Logistic LASSO in terms of test set accuracy. Finally, the various gradient boosting models and the ensemble model performed exceptionally well, with very good accuracy and much lower test set Logloss. As we may want to know which features contribute the most to model performance, we have shown a feature importance chart for the LightGBM and XGBoost models in Figures 2 and 3.

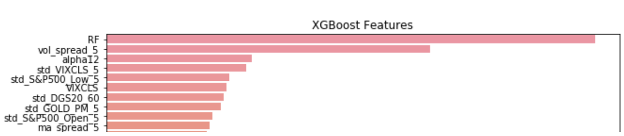

Figure 2. Top 10 features for XGBoost Model

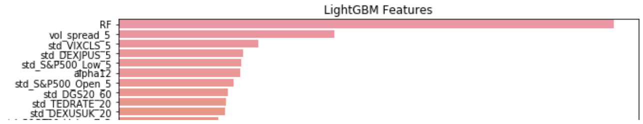

Figure 3. Top 10 features for LightGBM model

We can see that in both models, the Risk Free Rate, volatility of the spread over 5 days, alpha12, standard deviation of the VIX, standard deviation of the S&P500, and standard deviation of DGS 20 over 60 days were among the top 10 features. In XGBoost, the standard deviation of gold over 5 days was in the top 10, while for LightGBM the standard deviation of the TED rate over 20 days was in the top 10. This shows that changes in the spread are extremely affected by volatility features.

Accuracy is not a great measurement of predictive performance in a finance setting, because we are more interested in making good bets with events with low risk. Thus, looking at the Logloss on the test set may be a better indicator of model performance. As we will expand upon in future sections of this paper, we are very interested in trading performance, which is related to the annualized return you can extract from these predictive models, as well as risk quantified by annualized volatility of a trading strategy based on these models.

5. Trading Strategy

5.1 Methodology

We employ ETFs that track various bond indices. For the top leg of the spread, we use QLTA (For predictions from the AAA, AA and A indices) and PBBBX (For predictions from the BBB index). For the bottom leg, we use IEF. Predictions for yield directions are inversely related to bond prices and correspondingly also inversely related to our ETF prices. If we predict spreads will widen, we short the top leg ETF and long the bottom leg ETF. If we predict spreads will tighten, we long the top leg ETF and short the bottom leg ETF. The nominal amounts are scaled such that the strategy will be self financing on any given day and we use notional amounts $1 on both the long and short side.

Our predictions come from probability estimates which allows us to refine the strategy further. Instead of predicting a binary class, our models will return a probability score p, which we will use to calculate a confidence measure $2*p-1$. If this value has absolute value near 1, we are very confident in our predictions of increase or decrease. However, if the value is near 0, we are not very confident in our predictions.

We multiply this confidence measure to the $1 amounts on the long and short trades to further scale our trading strategy. The profit of the trading strategy is computed in the following way:

Let T_t be the price of an ETF for the top leg and B_t be the price of an ETF for the bottom leg. Let X_t denote the value of the trading strategy at time t. Consider just two time periods,

Table 2. Model Results

Result	Logistic LASSO (Base Line)	Decision Tree (1)	Random Forest (2)	Gradient Boosting (3)	XGBoost & LightGBM (4)	Ensemble (1),(3) and (4)
CV Accuracy	Fold 1: 0.533 Fold 2: 0.600 Fold 3: 0.556 Fold 4: 0.605 Fold 5: 0.585	Fold 1: 0.488 Fold 2: 0.593 Fold 3: 0.537 Fold 4: 0.606 Fold 5: 0.570	Fold 1: 0.558 Fold 2: 0.672 Fold 3: 0.597 Fold 4: 0.674 Fold 5: 0.701	Fold 1: 0.587 Fold 2: 0.625 Fold 3: 0.552 Fold 4: 0.652 Fold 5: 0.657	Fold 1: 0.580 Fold 2: 0.620 Fold 3: 0.547 Fold 4: 0.617 Fold 5: 0.622	Fold 1: 0.513 Fold 2: 0.614 Fold 3: 0.541 Fold 4: 0.615 Fold 5: 0.597
CV Logloss	Fold 1: 0.840 Fold 2: 0.666 Fold 3: 1.015 Fold 4: 0.664 Fold 5: 0.667	Fold 1: 2.644 Fold 2: 0.674 Fold 3: 0.800 Fold 4: 0.675 Fold 5: 0.680	Fold 1: 0.665 Fold 2: 0.620 Fold 3: 0.656 Fold 4: 0.621 Fold 5: 0.590	Fold 1: 0.667 Fold 2: 0.636 Fold 3: 0.671 Fold 4: 0.635 Fold 5: 0.624	Fold 1: 0.671 Fold 2: 0.651 Fold 3: 0.682 Fold 4: 0.651 Fold 5: 0.648	Fold 1: 0.687 Fold 2: 0.651 Fold 3: 0.687 Fold 4: 0.648 Fold 5: 0.648
Test Accuracy	0.596	0.589	0.536	0.595	0.607	0.605
Test Logloss	0.696	0.682	0.688	0.676	0.673	0.673

where we enter a position at time t and exit at time $t+1$ for demonstration purposes:

$$C_t = 2 * p_t - 1$$

$$\Delta_{T_t} = \frac{1}{T_t}$$

$$\Delta_{B_t} = \frac{1}{B_t}$$

$$X_t = \begin{cases} (T_t * \Delta_{T_t} - B_t * \Delta_{B_t}) * C_t & p_t > 0.5 \\ (B_t * \Delta_{B_t} - T_t * \Delta_{T_t}) * C_t & p_t \leq 0.5 \end{cases}$$

$$\implies X_{t+1} = \begin{cases} (T_{t+1} * \Delta_{T_t} - B_{t+1} * \Delta_{B_t}) * C_t & p_t > 0.5 \\ (B_{t+1} * \Delta_{B_t} - T_{t+1} * \Delta_{T_t}) * C_t & p_t \leq 0.5 \end{cases}$$

Then, profit $\pi_{t+1,t}$ between trading days can be easily computed on a daily level as $X_{t+1} - X_t$, for all days t we in the test set. Since the strategy is self financing, and at most can be long and short \$1, the daily profits in fact are simply the daily returns of the strategy. We can thus get annualized estimates of average return and volatility as follows (for T being a set of times that we consider in the test set):

$$\mu_{\text{annualized}} = 252 * \frac{1}{|T|} \sum_{t \in T} \pi_{t+1,t}$$

$$\sigma_{\text{annualized}} = \sqrt{252} * \sqrt{\frac{1}{|T|-1} \left(\pi_{t+1,t} - \frac{1}{|T|} \sum_{t \in T} \pi_{t+1,t} \right)^2}$$

5.2 Trading Strategy Results

Table 3 details the trading strategy results. From our modes, there is a clear consensus that predictions from A rated bonds do not perform well with the trading strategy that uses the ETF QLTA (which is comprised of AAA to A rated bonds). However, the other spread types perform extremely well. The Decision Tree performed the worst, but this is to be expected as it is the simplest model we considered. Surprisingly, Gradient Boosting by itself performed exceptionally well, even with using predictions from the A rated spread. Additionally, a striking feature is that predictions on the A rated spreads all have volatility around 10.4 to 10.5 percent, and for the BBB rated spread a volatility of around 12.5 percent. The ensemble model does exceptionally well for the BBB spread predictions, and this strategy in particular produces a very appealing risk to reward profile.

In figures 4 - 7, we show a time series of the trading performance on the test set dates. The ensemble method is superior in the AAA, AA, and BBB bonds. However, Gradient Tree Boosting does the best for the A bonds and all of the models are much more volatile for these bonds.

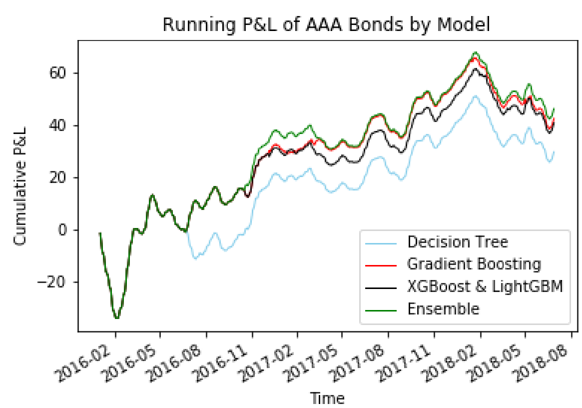

Figure 4. Trading Performance for AAA Bonds

5.3 Conclusion

We investigated various tree-based machine learning methodologies to predict yield curve spread movements, and examined the profitability of trading strategies based on these predictions. We find that while it is extremely difficult to get a very high out of sample accuracy, it is possible to ex-

Table 3. Trading Strategy Results

Classifier	Spread Type	Annualized Return	Annualized Volatility
Gradient Boosting	AAA	17.14%	10.39%
Gradient Boosting	AA	19.35%	10.38%
Gradient Boosting	A	8.80%	10.48%
Gradient Boosting	BBB	25.77%	12.50%
Decision Tree	AAA	11.96%	10.42%
Decision Tree	AA	11.96%	10.42%
Decision Tree	A	4.04%	10.49%
Decision Tree	BBB	18.42%	12.55%
XGBoost&LightGBM	AAA	16.47%	10.40%
XGBoost&LightGBM	AA	16.43%	10.40%
XGBoost&LightGBM	A	2.58%	10.49%
XGBoost&LightGBM	BBB	24.68%	12.50%
Ensemble	AAA	18.65%	10.38%
Ensemble	AA	18.65%	10.38%
Ensemble	A	2.26%	10.50%
Ensemble	BBB	28.06%	12.48%

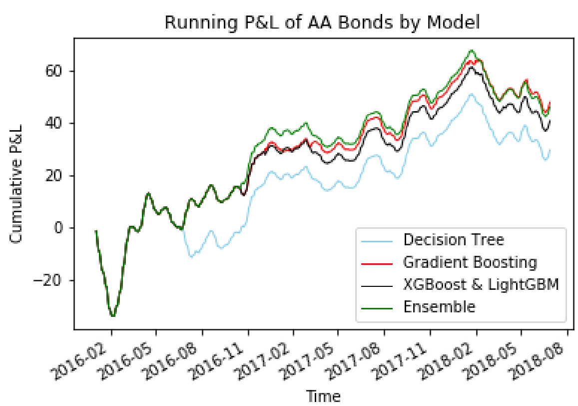

Figure 5. Trading Performance for AA Bonds

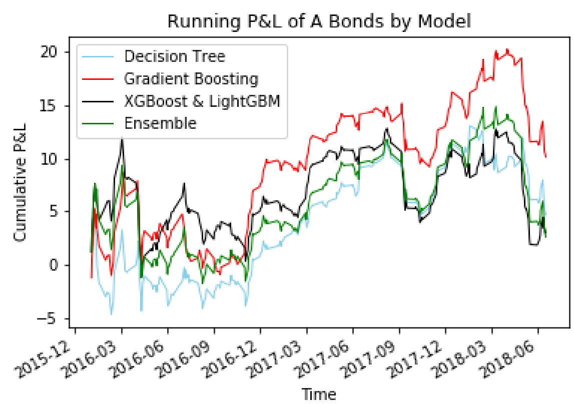

Figure 6. Trading Performance for A Bonds

tract extremely lucrative and profitable trading strategies from modestly accurate predictions. By scaling predictions by the confidence of our predictions, we are able to achieve great risk to return profiles trading the ETFs: QLTA, PBBBX and IEF. By constructing long short portfolios that are self-financing, and scaling the amount by our predictive confidence, we can trade on a daily timescale by making bets on spreads widening or tightening. In the future, to get even better results, we believe researchers can investigate at the individual bond level and consider a set of liquid instruments. By building a model at the individual bond level, more data points will be available to train and test on, yielding even more robust results. With our conservative approach to cross validation, we still believe that our results generalize well out of sample, as shown by the performance of the trading strategies we tested.

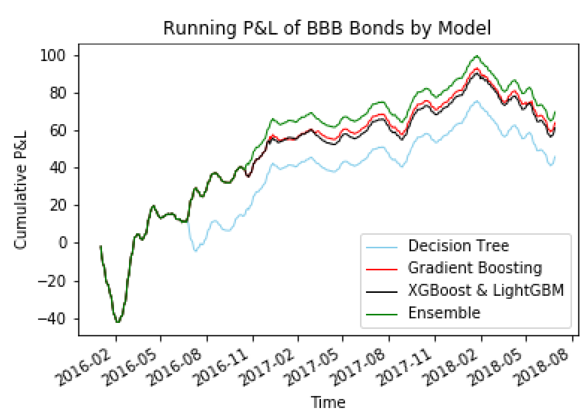

Figure 7. Trading Performance for BBB Bonds

References

Longstaff: Corporate Yield Spreads: Default Risk or Liquidity? New Evidence from the Credit Default Swap Market
`http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.337.1116rep=rep1type=pdf`

Kwan: Firm-specific information and the correlation between individual stocks and bonds
`https://www.sciencedirect.com/science/article/pii/0304405X95008364?fbclid=IwAR1wdRP-nw6jyEBh-`

101 Formulaic Alphas
`https://arxiv.org/pdf/1601.00991.pdf`

Hyndman: Cross-validation for time series
`https://robjhyndman.com/hyndsight/tscv/`

Ensemble Methods
`https://scikit-learn.org/stable/modules/ensemble.html`

Hauskrecht: Decision Trees
`https://people.cs.pitt.edu/ milos/courses/cs2750-Spring03/lectures/class19.pdf`

Machine Learning Methods for Better Models for Predicting Bond Prices
`https://arxiv.org/pdf/1705.01142.pdf`

The Cross-Section of Expected Corporate Bond Returns: Betas or Characteristics?
`http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.198.5763rep=rep1type=pdffbclid=IwAR1`

LightGBM
`https://lightgbm.readthedocs.io/en/latest/`

A Gentle Introduction to XGBoost for Applied Machine Learning
`https://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/`

Decision Trees
`https://scikit-learn.org/stable/modules/tree.htmltree`

Empirical Asset Pricing via Machine Learning `https://papers.ssrn.com/sol3/papers.cfm?abstract`$_id =$ 3159577