A Machine Learning Approach to Credit Spread Forecasting

Abstract

Credit Spread indicates the risk premium of a risky security, which can serve as a measure of economic uncertainty. Successful prediction of movements in credit spread makes it possible for investors to develop profitable trading strategies.

In this report, we selected various features with high predictive power on economic uncertainty, to be able to predict credit spread. We developed a pipeline for data preprocessing and feature engineering. We also used Hidden Markov Model to detect market regime shifting. Then, we developed an XGBoost model that proved to be more effective when compared to our linear regression benchmark model. Based on our prediction, we built a trading strategy on the correlated ETF.

Content

1. Data Exploration 1
1.1 Data Gathering 1
1.2 Data Preprocessing 2
1.3 Feature Engineering 3
1.4 Feature Selection 5
1.5 Distribution Analysis 5
1.6 Correlation Analysis 6
2. Model selection 6
2.1 Linear Model 6
2.2 Tree Model 7
3. Results 7
4. Model Explanation 8
5.Trading Strategy 8
6. Conclusion 9
Reference 10
Appendix 10

1. Data Exploration

1.1 Data Gathering

We define credit spread as the *DAAA* index (Moody's AAA Corporate Bond yield) minus the 10-year treasury rate. This is our prediction target, and can be accessed directly in FRED by the name *AAA10Y*.

For the independent variables, we included macro-level and bond-level features as well as sentiment indicators in our dataset.

According to many research papers which focused on bond spread and bond markets, macroeconomic indicators are widely used and proved to be helpful. Indicators of the overall economy such as GDP, CPI and unemployment rate were added to the dataset.

Features such as note issuance volume were used in our models since they represent US bond markets. In addition, our project also used macro political uncertainty indicators that reflect the overall US policy stability such as Financial Regulation and Fiscal Policy (Categorical EPU Data).

For bond-level technical indicators, limited bonds were ranked as AAA so the list of these companies could be easily collected. Therefore, some Greeks of these corporate bonds and related asset swap spreads were used.

This project also made use of data collected from Google trend. Google trend is developed by Google and shows word frequencies for selected word and phrases. This dataset serves as a sentiment indicator of the markets.

In total, we aggregated about 160 features with 6940 observations (time span from 01/03/2000 to 12/31/2018).

1.2 Data Preprocessing

The aggregated dataset is quite messy with various issues like missing data, different scales, non-stationarity and skewed distributions. To feed our model with data of better quality, we followed the following procedure to clean up and preprocess the dataset.

1.2.1 Missing data handling

There are different sources for missing data in our dataset, such as data from non-trading day and mismatch of different frequencies.

Firstly, we applied a time window to slice the data since some features didn't exist in early period of time. For features with data from non-trading dates, we dropped all feature values on those dates. As for scaling different data frequencies, we employed a forward filling method (using values in time t-1 to fill values in time t) to convert monthly features to daily ones. And we finally drop the features which have too many null values.

After solving the missing data problems, we then split the whole dataset into three parts: a training part for model learning, contains first 70% of whole dataset, range from 01/2005 to 10/2014; a development set with the following 15% data from 11/2014 to 11/2016 for tuning parameters of models; a Test Set with the rest 15% data for evaluating the performance of our model, which contains data from 12/2016 to 12/2018. Based on three different parts of the data, we then did the following data preprocessing and model training.

1.2.2 Normalize

The distributions of financial data tend to be right-skewed, and the skewness property will make credit spread prediction easily affected by asymmetry and outliers in distribution. At the same time, normal distribution is more robust to predict distributions with symmetry property. To make our dataset closer to normal distribution, we applied log transformation to adjust features with skewed distributions. As our feature values are not all positive, we applied the following formula to transform our data:

$$f(x) = log(|x| + 1) \cdot sign(\mathrm{x})$$

For our target variable, we simply used $f(x) = log(x + 1)$ since credit spread is always larger than 0 in our dataset.

Table I   Feature skewness before/after transformation

Feature Symbol	Skew before transformation	Skew after transformation
^VIX_TR	40.48910	7.888031
meltdown	13.393359	0.674479
investment-grade	8.136946	5.331867
bond broker	7.551093	5.331867
sell-off	6.769123	3.730037

After log transformation, the skewness was reduced significantly. Especially for some features like meltdown, a word frequency from google trend, the skewness was reduced to near zero after transformation.

1.2.3 Stationarilize

When predicting credit spreads, we want to keep the similarity between training set and Test Set, which increases the chance for the model to perform well in both sets. Thus, we applied augmented Dicker-Fuller test to check the stationarity and differentiate all the features until stationary.

One thing to mention is that we applied a log-transformation to the dataset first, then differentiate the dataset. We would get the log return of each feature which has economic sense.

1.2.4 Standardize

Since our features are different in scale, standardization was applied to make data consistent in scale. Also, consistent scaling can make algorithm converge much faster in training set.

Since the distributions of most features are highly concentrated, we can fairly assume these data are not subject to outliers. So, we can conduct min-max scaling method to scale data.

$$X_{scaled} = \frac{X - min(X)}{max(X) - min(X)}$$

1.3 Feature Engineering

1.3.1 Multicollinearity handling

For machine learning algorithms, in general, multicollinearity would not be an issue for prediction, but it may cause inaccurate estimation of parameters.

We used Principal Component Analysis (PCA) to handle this problem. Instead of using PCA on the whole dataset, we applied PCA to a certain group of highly correlated variables, for example, various kinds of price data of VIX (i.e. OHLC). We can retain most of the variance as well as remaining high explanation on new features after PCA.

1.3.2 Lagged Credit Spread Features

Consistent with the idea of time series analysis, the lagged credit spread was also introduced to our features set. We used one lagged credit spread to help with prediction in our models.

1.3.3 Interest Rate Term Structure Features

We also computed the difference of Treasury yield with different maturities as our features.

1.3.4 Hidden Markov Model for regime detection

One implicit assumption of machine learning is the training set and Test Set come from the same distribution. However, financial markets can change their behaviors and the change often persists for several periods. This is often known as market regime shifting. Here we employed Hidden Markov Model (HMM) to detect market regime, which can make our machine learning algorithms more robust over different regimes.

Here we defined market regime as different states and used the level of credit spread as our observation sequence. Our target is using the time series observations to infer the hidden underlying states sequence. That is the decoding problem in HMM, for a given Hidden Markov Model λ and observation sequence $O = (o_1, o_2, \dots, o_T)$, find the states sequence $I = (i_1, i_2, \dots, i_T)$ that maximize the conditional probability $P(I|O)$.

First, we looked at our credit spread training set.

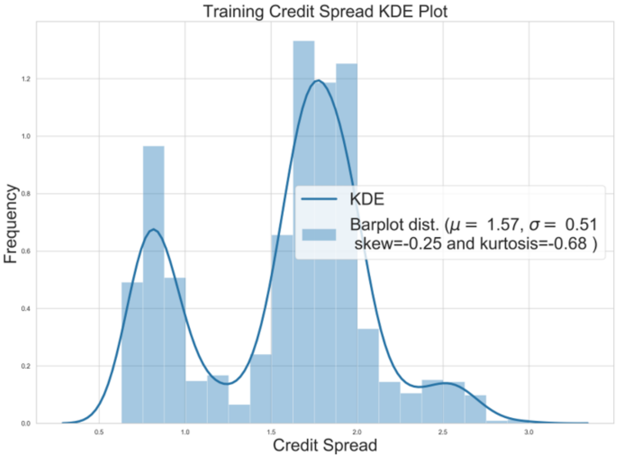

Figure 1 Train set Credit Spread Distribution

It is apparently not normal, and we observed three peaks in the distribution. Thus, we started to analysis HMM with three states. We used hmmlearn package available in python to fit the data, and we selected normal distribution as the emission distribution.

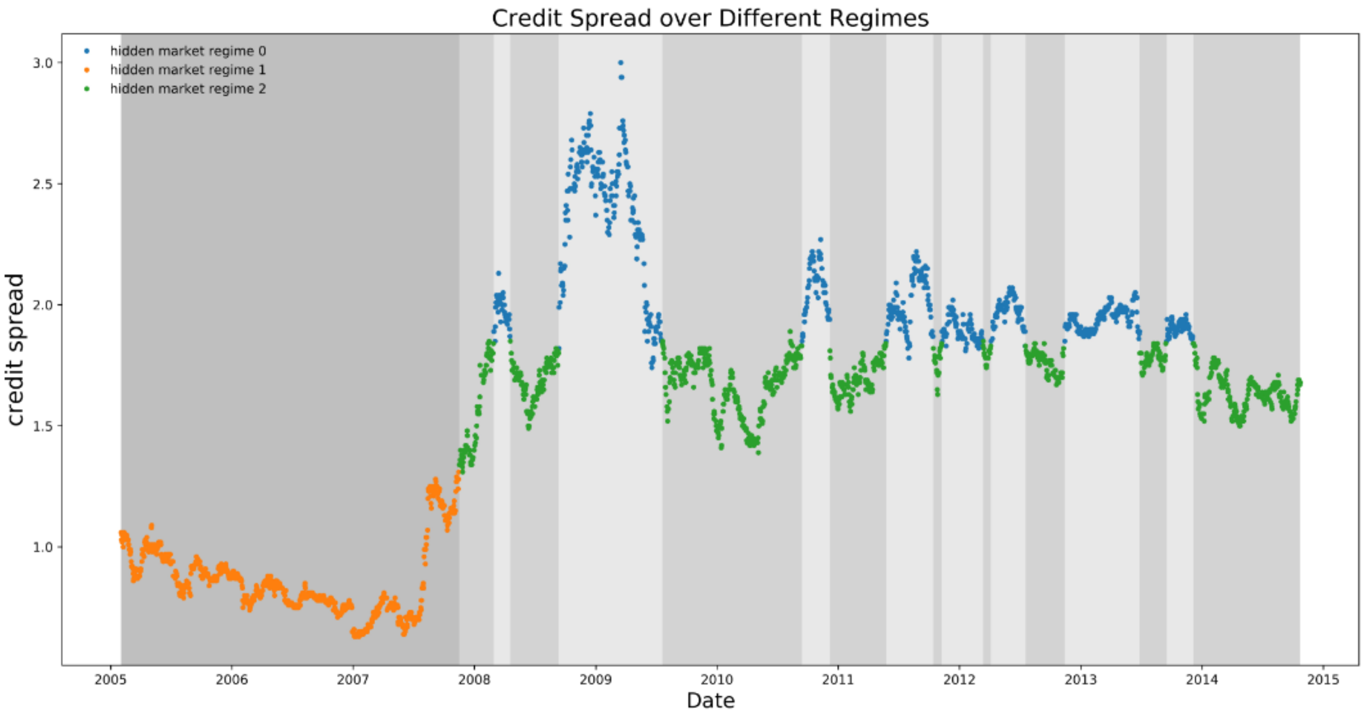

Figure 2 Credit Spread Time Series over Different Regimes

Table II Market regime transition matrix

State i, j	0	1	2
0	0.999	0.001	0.
1	0.001	0.992	0.007
2	0	0.013	0.987

Table III Central moments of credit spread distribution in different market regime

Market Regime	Mean	Standard Deviation	Skewness	Excess Kurtosis
0	2.08	0.25	1.37	0.83
1	0.86	0.14	0.99	0.64
2	1.67	0.11	-0.76	0.11

As shown in Figure 2, HMM separates the credit spread sequence into three regimes: Market Regime 1 with mean 0.86, which is the regime with low credit spread; Market Regime 2 with mean 1.67, which is the regime with medium credit spread; Market Regime 3 with mean 2.08, which is the regime with extreme high credit spread. The three regimes separation aligns with our intuition that market can be separate to bull, bear and somewhere in the middle.

Later we used this model to generate 50000 samples, then compared the samples distribution with our observation sequence.

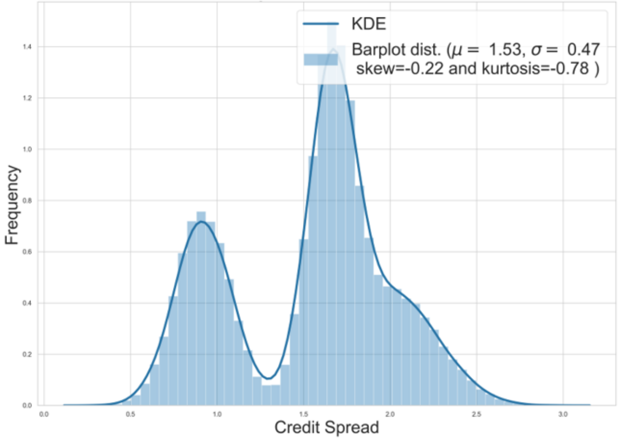

Figure 3 Generated Sample Distribution by HMM

We generated similar distribution with our observation sequence, which meant that Hidden Markov Model with three states can provide us a pretty good perspective about market regimes. We used the output as one of our features for machine learning algorithms.

1.4 Feature Selection

After data collection and feature engineering, we then did feature selection based on feature importance which is measured by decrease in impurity (variance) under the framework of Random Forest. 1000 decision trees were created and trained to get feature importance. We then ranked features according to the average impurity reduction of each feature across decision trees and selected features whose importance was greater than 0.005.

Table IV Top 5 Feature Importance

Feature Symbol	Importance
y_lag1	0.158222
market regime_cs	0.139979
GDP	0.112010
Fiscal Policy	0.011196
leverage	0.057937

As showed in Figure 4, the selected features have good economic explanations. One-day lagged credit spread (*y_lag1*) showed top importance because as time-series data, credit spread has highly auto-correlation effect.

Macroeconomic features including *Fiscal Policy* and *GDP* outline the overall macro-economy condition.

Also, the excellent performance of market regime indicator (*Market regime_cs*) means that in different market states, credit spread represents different data patterns in which features work differently.

Google trend words including *Leverage* reflect the sentiment situations of the market which indicates whether the overall expectation to the market is deteriorating or not.

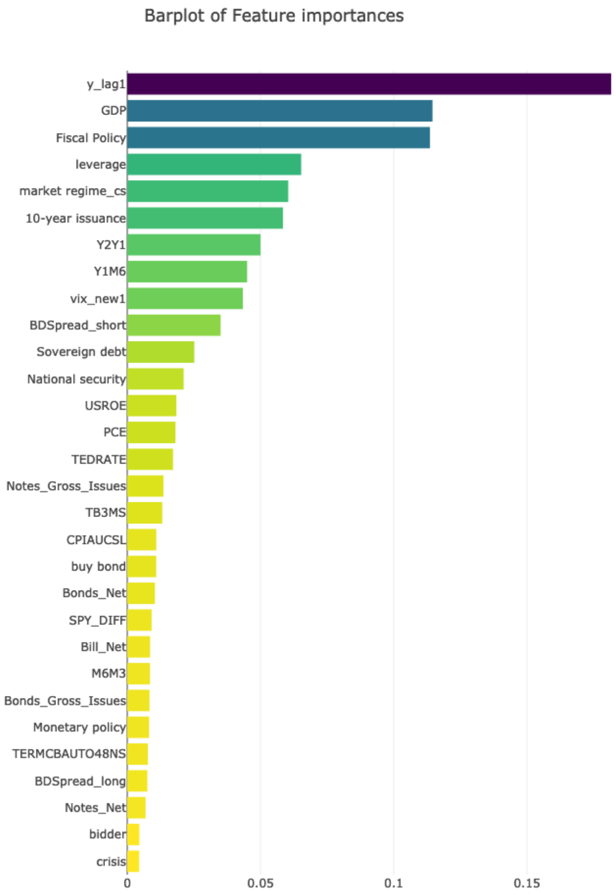

Figure 4 Bar Graph of Feature Importance

Market technical indicators including PCA of VIX features (*vix_new1*) and *SPY-DIFF* indicate that some effective transformation of market data may give better information.

1.5 Distribution Analysis

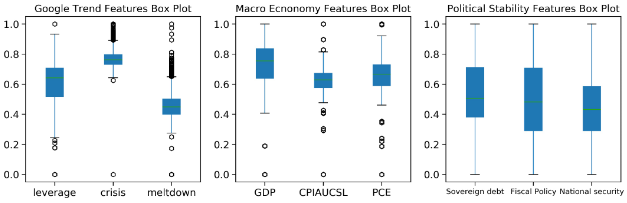

Figure 5 Boxplot of selected features

After log transformation and differentiation, we expected our data to be closer to normal distribution.

For google trend features (Figure 5 left 1), they are centered and have more outliers than other features. Since we did the *MinMax* scaling, those outliers would have limited effect on our model.

For Macro economy features and Political Stability features (Figure 5 right 2 plots), after transformation, they do become more symmetry, which is the normal property we want.

1.6 Correlation Analysis

As we explained before, multicollinearity can be a problem for machine learning algorithms to estimate suitable parameters.

As shown in Figure 6, there is no high correlation between our features. Firstly, because we applied PCA to certain group of features that are highly correlated (VIX OHLC data to vix_new1 feature in Figure 9). Also, because we differentiated the features in dataset which can eliminate the common trend in correlated features.

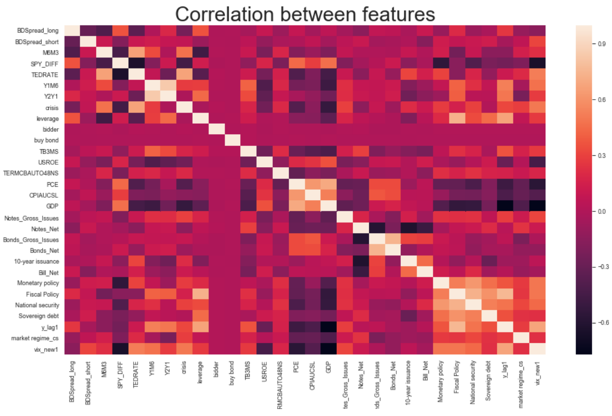

Figure 6 Correlation Heat Map between Features

2. Model selection

2.1 Linear Model

2.1.1 Linear Regression

For all the models below, we used mean squared error (MSE) as performance metrics. We would apply different regularization to linear models to avoid over-fitting. First, we trained our model in training set, then used the development set to tune the model, finally utilized the Test Set to evaluate model performance.

The first model we used is linear regression without regularization. This model would serve as a benchmark model to evaluate another models' performance.

$$minimize\ \frac{1}{2M}\sum_{i=1}^{M}(y_i - w_i x_i)^2$$

We also tried ridge regression on our dataset. The ridge model has ability to shrink the coefficients, further to make our model results more stable.

$$minimize\ \frac{1}{2M}(\sum_{i=1}^{M}(y_i - w_i x_i)^2) + \lambda\sum_{i=1}^{M} w_i^2$$

Thirdly, we tried lasso regression. The lasso regression would give a sparser result. Some unnecessary features' coefficients would be set to zero in lasso.

$$minimize\ \frac{1}{2M}(\sum_{i=1}^{M}(y_i - w_i x_i)^2) + \lambda\sum_{i=1}^{M}|w_i|$$

Table V Linear model performance

Model	MSE for Different Sets		
	Train Set	Dev Set	Test Set
Linear Regression	0.001926	0.002132	0.002768
Lasso Regression	0.002553	0.002627	0.001925
Ridge Regression	0.001925	0.002132	0.002768

By comparing the linear model results, those three models yield to almost the same result in the Test Set. This is a sign indicating the data preprocessing above successfully reduced redundant features.

On the other hand, financial markets are also famous with its non-linearity. We believe a simple linear model may fail to capture some deep patterns in the financial data. Later we would use tree-based models to enhance our prediction.

2.2 Tree Model

2.2.1 Random Forest

Random Forest constructs a multitude of decision trees at training time and utilizes bagging method to randomly select features and observations for each tree training. As for regression, Random Forest outputs mean prediction of the individual trees.

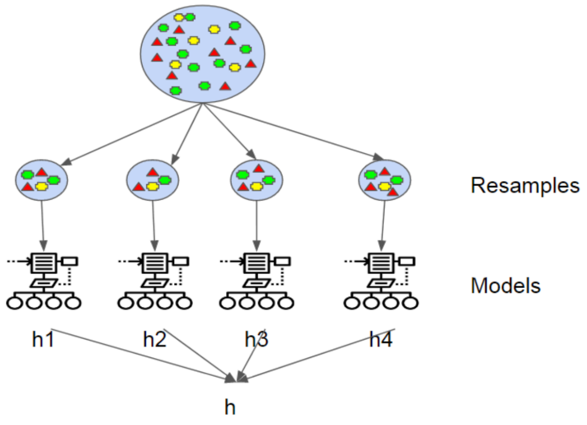

Figure 7 Random Forest Simplified

Here we still used training set to train Random Forest model, development set to tune the model, Test Set to evaluate. Since Random Forest has more hyper parameters to tune, we tuned those parameters based on their importance step by step.

First, we tuned the number of trees used, then found the optimal level is 300. Next, we selected 70% as the maximum features ratio for each tree which minimize MSE. After that, we found an optimal max depth which equals to 5.

2.2.2 XGBoost

Boosting is an ensemble learning algorithm which improves weak prediction models gradually. We trained XGBoost models for prediction.

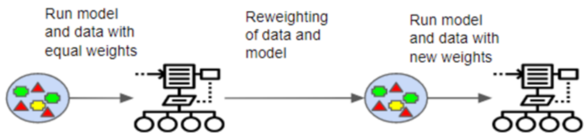
Run model and data with equal weights
Reweighting of data and model
Run model and data with new weights

Figure 8 Boosting Simplified

We used the similar method to tune the boosting model. And we ran the well-tuned model in the Test Set.

3. Results

The results of our tree models can be seen in Table VI. The XGBoost has the lowest MSE in the Test Set. The model performance was greatly enhanced by introducing boosting method. Within each iteration, boosting would increase the weight for mis-predicted data in the previous iteration. Thus, the boosting model reduced the bias of our prediction, and represented a more accurate result.

By comparing tree models and linear models, the tree models proved to be more accurate because they can capture the non-linearity in our data.

By comparing Random Forest and XGBoost results, it is easy to observe that overfitting in this Random Forest model, resulting in a not so good performance in Test Set even we used Develop Set to validate that model. This indicates tree model is more capable of catching the non-linearity pattern, while Random Forest model failed to adapt itself between different market environments. On the contrary, XGBoost is smarter through different periods, so we will use this model for prediction and further use the results for trading.

Table VI All Model Performance

Models	MSE for Different Sets		
	Train Set	Dev Set	Test Set
Linear Regression	0.001926	0.002132	0.002768
Lasso Regression	0.002553	0.002627	0.001925
Ridge Regression	0.001925	0.002132	0.002768
Random Forest	0.000149	0.000883	0.002100
XGBoost	0.000643	0.000803	0.000854

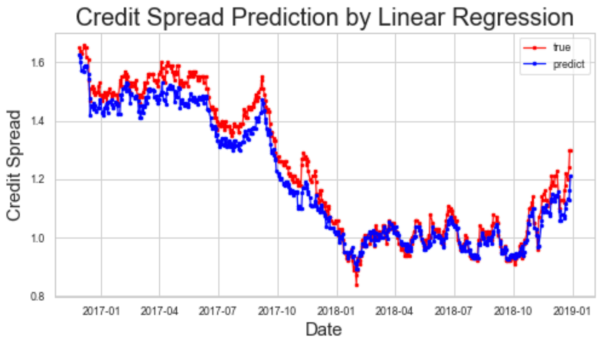

Figure 9 Linear regression out-of-sample Results

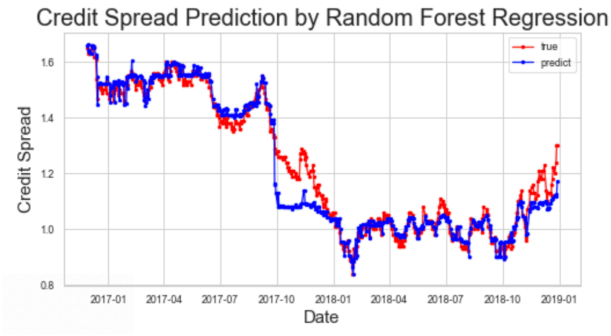

Figure 10 Random Forest out-of-sample Results

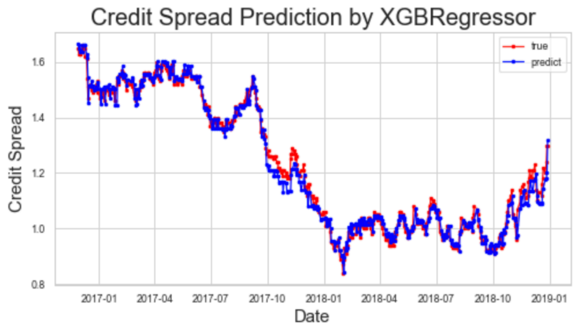

Figure 11 XGBoost out-of-sample Results

4. Model Explanation

Traditional machine learning for financial time-series prediction have certain problems including non-transparency and overfitting, and more specifically for financial market, the market regime shift.

Part of the non-transparency problem could be well explained by our feature importance graph. The important features selected to our model have strong and clear economic explanation to our target as we have explained in 1.4 Feature Selection. As for our model itself, the XGBoost model and Random Forest model use classic ensemble method. Decision trees are connected in parallel or in sequence for the two different algorithms. Tree nodes split when information entropy has maximum gain. This helps explain why and how the features contribute to the prediction ability of our model.

To solve overfitting problem, we kept our model pretty simple. There are no hidden structures in our model. Only a few important features were selected to our model and we remain economics explanation towards them. We also split the dataset into three parts, and only use the Develop Set to tune the model; the Test Set was only used once to evaluate the generalization ability of our model, which prevents overfitting. We also strictly controlled the parameters including max depth of trees to reduce the complexity of the individual tree.

We use Hidden Markov Model (HMM) to deal with regime shift issues which has been discussed in 1.3.3 HMM for regime detection.

5.Trading Strategy

Now the model shows a pretty good predictive power towards credit spread. It is possible for us to take the advantage of the predictions to develop trading ideas.

Since credit spread is not traded directly, we used 'LQDH' ETF as our trading security. LQDH stands for the Interest Rate Hedged Corporate Bond ETF, it is negative correlated to the credit spread with correlation equal to -0.96.

The average daily volume of the ETF is 34,742, which is quite liquid compared to other credit spread ETF. If we only trade in a small volume, the liquidity problem can be ignored. We also assume the trading cost is zero for simplicity.

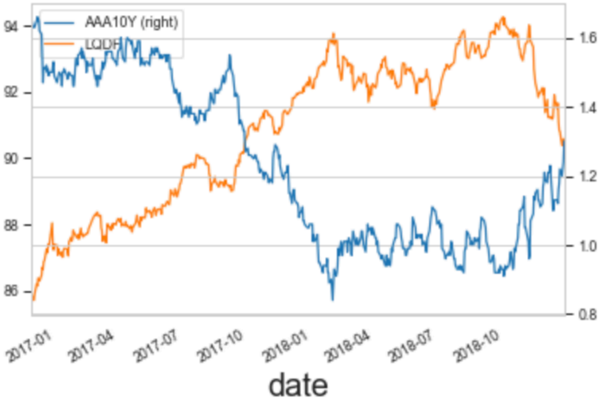

Figure 12 Correlation between ETF and credit spread

Our strategy is pretty simple, when the predicted credit spread tomorrow is larger than the actual credit spread today, we would close our position if we have some and then short the ETF; when the predicted credit spread is lower, we would long the ETF.

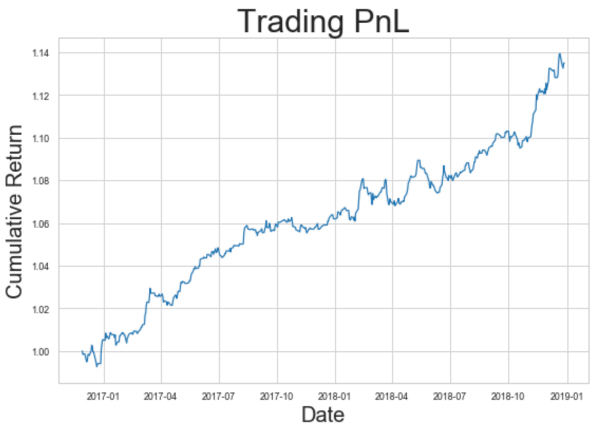

Figure 13 Cumulative PnL from 2017-2018

We used the test period from 2016/12-2018/12 as our trading period. And we rebalanced our portfolio on a daily basis. For this strategy, we achieved an annualized information ratio of 2.5 and an annualized return of 6.67%.

6. Conclusion

In this project, we predicted credit spread using a variety of macro-level, bond-level and sentiment indicators. We also built a trading strategy based on our prediction.

For data preprocessing, we developed a pipeline for handling missing data and removing negative data properties such as asymmetry, non-stationarity and different scaling in the dataset.

We also conducted feature engineering like adding lagged credit spread, using “local PCA” of a group of highly-correlated features, and using Hidden Markov Model to detect market regimes.

Next, we selected features based on Feature Importance Analysis in Random Forest model. Some features such as lagged credit spread and market regime indicator significantly stand out with good economic reasons. In the end, we retained 30 features to feed into our model.

Using mean squared error (MSE) as performance metrics, we first trained linear model in the model selection, whose MSE also served as a benchmark for later improvements. The close similarity of MSE between three linear models reflects that we performed well in data preprocessing. Later, we trained tree models, in which the MSE was greatly reduced in all sets. Particularly, XGBoost model worked well with our data, reaching a satisfying MSE of 0.0008 in Test Set.

Finally, we used our prediction on credit spread to trade ‘LQDH’ ETF. With a simple long-short strategy, we achieved a high information ratio of 2.5 and total return of 6.67%.

Reference

[1] C.N.V. Krishnan. Predicting credit spreads, 2007.

[2] Bruno Miranda Henrique. Machine Learning Techniques Applied to Financial Market Prediction, 2019.

[3] Mahsa S. Kaviania. Policy Uncertainty and Corporate Credit Spreads, 2018.

[4] Stephen H.T. Lihn. Hidden Markov Model for Financial Time Series and Its Application to S&P 500 Index, 2017.

[5] Leung et al. Forecasting Stock Indices: A Comparison of Classification and Level, 2000.

[6] Barak et al. Fusion of Multiple Diverse Predictors in Stock Market, 2017.

[7] Peter Martey Addo. Credit Risk Analysis Using Machine and Deep Learning Models, 2018.

Appendix

Features	Data Source	Description
Credit Spread	FRED	Moody's AAA bond yield minus 10-year treasury yield
vix_new1	Yahoo Finance	The first principal component of different VIX prices features
SPY_DIFF	Yahoo Finance	The difference of SPY 12-day EMA and 26-day EMA
BDSpread_long	CRSP	The bid-ask spread of The Center for Research in Security Prices's 1-year (short-term) and 10-year(long-term) Fixed Income Indices.
BDSpread_short	CRSP	
TERMCBAUTO48NS	FRED	Finance Rate on Consumer Installment Loans at Commercial Banks, New Autos 48 Month Loan
TEDRATE	FRED	TED Spread
USROE	FRED	Return on Average Equity for all U.S. Banks
PCE	FRED	Personal Consumption Expenditures
CPIAUCSL	FRED	Consumer Price Index for All Urban Consumers: All Items
TB3MS	FRED	3-Month Treasury Bill: Secondary Market Rate
GDP	FRED	Nominal Gross Domestic Product in US
Y1M6, Y2Y1, M6M3	FRED	Treasury yield difference between 1 year and 6 months.
y_lag1	FRED	Lagged credit spread
market regime_cs	N/A	market regime generated by HMM
crisis,leverage, bidder, buy bond	Google Trend	Word Searching Frequency over time
National security	http://www.policyuncertainty.com/methodology.html	News data from the Access World News database of over 2,000 US newspapers
Sovereign debt		
Monetary policy		
Fiscal Policy		
Bills_Net	SIFMA	U.S treasury (Notes, Bonds) issuance gross amount for Gross, U.S treasury (Bills, Notes, Bonds) outstanding amount for Net
Notes_Net		
Notes_Gross_Issues		
Bonds_Net		
Bonds_Gross_Issues		
10-year issuance		10-year U.S treasury notes issuance amount