Exploring Credit Spread Opportunities Using Machine Learning

Yuan Han Lim

July 26, 2024

Abstract

This paper leverages machine learning techniques to predict the Option-Adjusted Spread (OAS) for US Corporate Investment-Grade bonds and formulates a systematic trading strategy based on these predictions. By integrating various economic, financial, and technical indicators, and utilizing models such as Logistic Regression, Random Forest, and Gradient Boosting, we aim to demystify the factors influencing credit spreads. Our findings suggest that a strategy based on these predictions can achieve moderate returns, demonstrating the practical applicability and effectiveness of our approach.

1 Introduction

Credit spreads over treasuries compensate investors for risks such as default, market volatility, and liquidity issues. This study explores how these risks, and consequently credit spreads, can be predicted using machine learning models, enabling the development of profitable trading strategies.

2 Market Conditions

The classification of market conditions into buy, sell, and hold signals is based on the log change of spreads, which defines market states as follows:

- **Tighten Signal**: When the log change of spreads is greater than the mean plus 1.5 times the standard deviation.

- **Widen Signal**: When the log change of spreads is less than the mean minus 1.5 times the standard deviation.
- **Hold Signal**: When the log change of spreads is between the mean minus 1.5 times the standard deviation and the mean plus 1.5 times the standard deviation.

Formally, the market condition M_t at time t is defined as:

$$M_t = \begin{cases} \text{Tighten Signal} & \text{if } \frac{\log(\text{Spread}_t)-\mu}{\sigma} > 1.5 \\ \text{Widen Signal} & \text{if } \frac{\log(\text{Spread}_t)-\mu}{\sigma} < -1.5 \\ \text{Hold Signal} & \text{if } -1.5 \leq \frac{\log(\text{Spread}_t)-\mu}{\sigma} \leq 1.5 \end{cases}$$

where $\log(\text{Spread}_t)$ represents the log change in the spread at time t, μ is the mean, and σ is the standard deviation of the log changes in spreads.

This classification helps in understanding the market environment and aids in making informed investment decisions based on the prevailing market conditions.

3 Data

3.1 Credit Spread Data (Response)

The BofAML US Corporate Investment-Grade OAS Index (C0A0) tracks the performance of investment-grade corporate bonds, adjusted for embedded options. Our models predict weekly spread changes across AAA, AA, A, and BBB ratings. Weekly log changes are calculated and we categorize the changes into 3 categories (i.e. 0: Tighten, 1: Hold, 2: Widen)

3.2 Explanatory Variables

- **Credit Spread Slope**: Difference between 10-15 year and 1-3 year OAS.
- **Speculative-Grade OAS (HYM)**: Spread for high-yield bonds.
- **Technical Indicators**: Including trailing 5-week log change

- **Macroeconomic Indicators**: CCPIU, CCPI, and Leading Indicator Index.
- **Financial Situation Indicators**: Chicago Fed's NFCI
- **Financial Market Variables**: US Treasury yields, RUSSELL 2000, VIX, SP500 and crude oil prices.

4 Data Preprocessing

Data preprocessing involves seasonality testing, ensuring time series stationarity through logarithmic transformation and differencing, and performing dimensionality reduction using Principal Components Analysis (PCA) for highly correlated subsets.

4.1 Seasonality Testing

We first detect possible seasonality patterns in the credit spread time series using regression models with dummy variables representing each month.

4.2 Stationarity

Stationarity of time series data is ensured by taking the logarithm of variables resembling financial securities prices (including OASs) and using the augmented Dickey-Fuller test:

$$\Delta y_t = \alpha + \gamma y_{t-1} + \delta_1 \Delta y_{t-1} + \delta_2 \Delta y_{t-2} + \cdots + \delta_{p-1} \Delta y_{t-p+1} + \epsilon_t$$

where the null hypothesis $\gamma = 0$ is tested.

4.3 Feature Correlation and Dimensionality Reduction

We use a correlation heatmap to identify highly correlated features and apply PCA to reduce dimensionality. Specifically, PCA was applied to the technical indicators and the financial market variables to address multicollinearity. The variables include PC_Diffs, derived from features such as Spread Slope, IG_HYM, AAA_HYM, AA_HYM, A_HYM, BBB_HYM, and PC_T5Wks, derived from trailing technical data (T_5WkAAA, T_5WkAA, T_5WkA, T_5WkBBB). Additionally, PC_UST was derived from U.S. Treasury data. By extracting the principal components, we can capture the majority of the variance in the data with fewer variables, thus simplifying the models and improving their robustness.

4.4 Feature Importance

Feature importance is calculated using the XGBoost classifier to determine which variables have the most influence on the predictions. The following features were found to be the most important:

- Spread Slope
- NFCI
- VIX
- Adj Close
- Alpha 12
- PC_UST1
- PC_T5Wk1
- PC_Diff1

The feature importances are visualized in a bar plot, indicating the relative importance of each feature in predicting the market state.

4.5 Alpha Signals

Investors tend to view bonds as substitute investments when the stock market is performing poorly. By generating additional features from the SP500 variables, we extract market trends/signals and incorporate this information into our models. We incorporate market trends/signals from the SP variables into our models. Specifically, we implement real quant trading alpha signals from Kakushadze (2016) on the SP variables. For example, Alpha 12 is computed as follows:

$$\text{Alpha 12} = \{\ -1 \times (X_t(\text{Close}) - X_{t-1}(\text{Close})) \times \text{sign}(X_t(\text{Volume}) - X_{t-1}(\text{Volume}))$$

where X_t(Close) and X_t(Volume) are the close price and volume at time t.

5 Models

The study employs several models:

5.1 Logistic Regression

Logistic Regression is used to predict the probability of a certain class or event. It is particularly useful for binary classification problems.

The Logistic Regression model is defined as follows:

$$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k$$

where p is the probability of the event occurring, β_0 is the intercept, and β_i are the coefficients for the predictor variables X_i.

5.2 Gradient Boosting

Sequentially reduces bias by fitting models to the gradient of the loss function:

$$f_m(X) = F_{m-1}(X) - \gamma_m \sum_{i=1}^{n} \nabla_{f_{m-1}} L(y_i, F_{m-1}(x_i))$$

5.3 Neural Networks

We implemented a neural network model using TensorFlow and Keras with the following architecture:

```
model1 = Sequential()
model1.add(Dense(64, input_dim=X_train.shape[1], activation='relu'))
model1.add(Dense(32, activation='relu'))
model1.add(Dense(16, activation='relu'))
model1.add(Dense(3, activation='softmax'))
model1.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accura

# One-hot encoding of the labels
y_train_encoded = to_categorical(y_train, num_classes=3)
y_test_encoded = to_categorical(y_test, num_classes=3)

# Fit the model with the encoded labels
model1.fit(X_train, y_train_encoded, epochs=10, verbose=False)
```

6 Model Evaluation

We evaluated the models using accuracy, AUC-ROC, precision, and recall. Below is an example of how these metrics were computed for a logistic regression model:

```
model = LogisticRegression(solver='lbfgs', max_iter=1000)
model.fit(X_train, y_train)

# Predict on the test set
y_pred_log = model.predict(X_test)

# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred_log)

# Transform labels for binary classification metrics calculation
lb = LabelBinarizer()
lb.fit(y_train)
y_test_binarized = lb.transform(y_test)
y_pred_binarized = lb.transform(y_pred_log)

# Predict probabilities for AUC-ROC
y_test_prob = model.predict_proba(X_test)

# Compute AUC-ROC - assuming binary classification or one-vs-rest for multi-class
auc_roc = roc_auc_score(y_test_binarized, y_test_prob, average='macro')

# Compute precision and recall - assuming average for multi-class
precision = precision_score(y_test, y_pred_log, average='macro')
recall = recall_score(y_test, y_pred_log, average='macro')

# Print the metrics for each stock
print(f"OAS: COAO")
print(f"Accuracy: {accuracy:.4f}")
print(f"AUC-ROC: {auc_roc:.4f}")
print(f"Precision: {precision:.4f}")
print(f"Recall: {recall:.4f}")
print("-" * 30)
```

Model	**Accuracy**	**AUC-ROC**
Logistic Regression	0.7879	0.6862
Neural Network	0.8030	0.5571
XGBoost	0.7424	0.4650

Table 1: Model Performance Metrics

The key performance metrics are calculated as follows:

- **Accuracy**: The proportion of correct predictions out of total predictions.
- **AUC-ROC**: Area Under the Receiver Operating Characteristic Curve, measuring the model's ability to distinguish between classes.
- **Precision**: The ratio of true positive predictions to the total predicted positives.
- **Recall**: The ratio of true positive predictions to the total actual positives.

7 Trading Strategy

7.1 Trading Signals

The strategy uses the following signals:

- **0 (Tighten)**: Buy and hold until the end of the week, then sell before the next week.
- **1 (Hold)**: No action is taken.
- **2 (Widen)**: Short the price of C0A0 and recover the position next week.

7.2 Strategy Implementation

The implementation of the trading strategy is as follows:

- **Tighten (0)**: If the signal is 0, the strategy buys and holds the position until the end of the week, then sells it before the next week.

- **Hold (1)**: If the signal is 1, no action is taken.
- **Widen (2)**: If the signal is 2, the strategy shorts the price of C0A0 and recovers the position next week.

The formula used to calculate the weekly returns is:

$$\text{Weekly Return} = \begin{cases} \frac{P_{t+1}-P_t}{P_t} & \text{if signal} = 0 \text{ (Buy)} \\ 0 & \text{if signal} = 1 \text{ (Hold)} \\ \frac{P_t - P_{t+1}}{P_{t+1}} & \text{if signal} = 2 \text{ (Sell)} \end{cases}$$

where P_t is the price at week t and P_{t+1} is the price at week $t+1$.

The cumulative return is calculated as:

$$\text{Cumulative Return}_{t+1} = \text{Cumulative Return}_t \times (1 + \text{Weekly Return})$$

8 Conclusions

This study demonstrates the efficacy of machine learning models in predicting credit spreads. Ensemble models combining linear and non-linear approaches show moderate performance.

Table 2: Model Performance Comparison

Metric	Logistic Regression	Neural Network	XGBoost
Cumulative Return	1.020246	1.000000	1.002023
Percentage Return	2.02%	0.00%	0.20%

Based on our statistical models and feature importance analysis, we identified five exceptionally effective explanatory variables: the Adj Close Price of SP500 of the index, NFCI, 3rd Principal componenet of the UST curve (PCUST3), first principal component of difference between HY spread and IG Spread (PCDiff1), and the first principal component from our 5-weeks trailing log change of spread (PCT5Wk1). These variables demonstrated strong predictive power in our analysis.

It is important to note that all machine learning-based strategies are subject to the risk of regime shifts. Caution should be exercised when applying the model to out-of-sample tasks, as unforeseen regime shifts could impact performance.