Replication File 2024: Machine Learning-Aided Modeling of Fixed Income Instruments

Ganesh Ashwin, Gaurav Ghosh, Krish Desai, Yadvesh Yadav

December 21, 2024

Contents

0.1 Main Hypothesis

The extended hypothesis of this paper is that machine learning models, including Random Forests (RF) and Support Vector Regression (SVR), combined with advanced technical and macroeconomic indicators, can outperform traditional forecasting and trading methods for sovereign 10-Year bonds across different countries. The approach integrates macroeconomic variables, momentum-based indicators, and Ichimoku cloud signals to generate actionable trading strategies. By leveraging these signals, the hypothesis asserts that it is possible to achieve superior performance in predicting bond yields and identifying profitable trading opportunities.

0.2 Secondary Hypothesis

A secondary hypothesis posits that preprocessing techniques, such as winsorizing, exponential smoothing, and feature selection via Variance Inflation Factor (VIF) and correlation matrix analysis, significantly improve the performance of machine learning models. These techniques aim to reduce noise, handle multicollinearity, and optimize feature selection for sovereign bond datasets, which are often influenced by diverse macroeconomic conditions.

0.3 Additional Hypotheses

Several additional hypotheses are explored in this extended project:

1. **Impact of Macroeconomic Variables:** Incorporating macroeconomic indicators such as CPI, PPI, Federal Funds Rate, Unemployment Rate, 10-Year Minus 2-Year Treasury Spread, and GDP into machine learning models will enhance their ability to predict bond yields by capturing broader economic trends.

2. **Performance of Momentum-Based Indicators:** The inclusion of momentum-based indicators, such as SMA, EMA, and RSI with different time windows, Moving Average convergence and divergency (MACD), Bollinger bands and Ichimoku Cloud will provide complementary short-term and long-term trend signals, improving prediction accuracy and trading strategy efficiency.

3. **Signal-Based Trading Using Ichimoku Cloud:** Signals derived from the Ichimoku cloud indicator will effectively classify bullish and bearish trends, providing clear buy/sell recommendations. These signals will enhance the overall performance of trading strategies when integrated with machine learning outputs.

4. **Effectiveness of Feature Selection:** Applying VIF and correlation matrix analysis to select features will improve model robustness by reducing multicollinearity and ensuring that only the most relevant indicators are included in the final dataset.

5. **Impact of Data Volatility Across Countries:** Sovereign bonds from different countries exhibit varying levels of volatility and liquidity. Models are expected to perform better for countries with more stable macroeconomic environments, while preprocessing techniques and feature engineering will mitigate performance degradation for highly volatile datasets.

6. **Trading Strategy Performance:** Machine learning-driven trading strategies integrating macroeconomic and momentum-based indicators with Ichimoku cloud signals will outperform traditional methods by generating more accurate and timely buy/sell recommendations.

1 Tests for the Hypotheses

1.1 Test for Main Hypothesis

To validate the main hypothesis, we will assess the predictive accuracy of machine learning models using gradient-boosting algorithms such as LightGBM and XGBoost. These models will predict sovereign bond yields across different countries, using an expanding window train-test split to simulate real-time forecasting. The hypothesis will be supported if the predictions align closely with historical data, and if the trading signals generated using Ichimoku cloud analysis consistently yield positive returns.

1.2 Tests for Secondary Hypothesis

To test this hypothesis, we will evaluate the impact of preprocessing techniques, including feature selection using VIF and correlation matrix analysis. Models will be trained with and without these preprocessing steps, and their performance will be compared using metrics specific to the implementation, such as prediction alignment with historical trends and profitability of trading strategies. The hypothesis will be supported if preprocessing significantly enhances model predictions and trading outcomes.

1.3 Tests for Additional Hypotheses

1. **Indicator Hypothesis:** The significance of macroeconomic indicators and momentum-based indicators such as SMA, EMA, RSI Moving Average convergence and divergence (MACD), Bollinger bands, and Ichimoku Cloud will be subjected to the t-test to determine the quality of the indicator in the model. A measurable improvement in signal quality and profitability of trading strategies will validate their effectiveness.

2. **Signal-Based Trading Using Ichimoku Cloud:** The effectiveness of Ichimoku cloud signals will be tested by evaluating the precision and recall of buy/sell signals against historical market movements. Positive cumulative returns from strategies based on these signals will confirm their utility.

3. **Effectiveness of Feature Selection:** The robustness of models using features selected via VIF and correlation matrix analysis will be compared to those using all available features. A reduction in noise and improved prediction quality will support this hypothesis.

4. **Model Optimization:** The impact of Federov's algorithm with multi-level parameter tuning will be assessed by tracking improvements in prediction consistency and trading profitability. Additionally, the use of expanding window train-test splits will be evaluated for its effectiveness in adapting models to changing market dynamics.

5. **Impact of Data Volatility Across Countries:** Model performance will be tested on datasets from countries with varying levels of volatility. Success will be measured by the stability of predictions and the adaptability of trading strategies to different macroeconomic environments.

6. **Trading Strategy Performance:** The overall profitability of machine learning-driven trading strategies will be validated by simulating trades using predictions and signals derived from the models. Positive cumulative returns and reduced drawdowns compared to benchmarks will confirm the hypothesis.

2 Literature Review

2.1 Bond-Based Studies

(Monte Carlo Simulation with Machine Learning for Pricing Bonds)
Dubrov (2015) explores the use of Monte Carlo simulations for pricing bonds with exotic features, such as early-redemption clauses for issuers or conversion clauses for bondholders. The author applies machine learning techniques to improve pricing accuracy where traditional methods struggle, particularly for complex bond structures. This study demonstrates that machine learning, when combined with Monte Carlo simulations, can model non-linear relationships and complex features in bonds, which would be difficult to price analytically. [**?**]

(Price Prediction Using Machine Learning)
Ganguli and Dunnmon (2017) conducted an extensive study on U.S. corporate bond price prediction. They used a dataset containing 762,678 U.S. corporate bonds with 61 attributes, including coupon rate, maturity, trade details, and regression-based price estimates. The authors tested several models, including linear regression, ARIMA, random forests (RF), and artificial neural networks (ANNs). ANNs performed best but were limited by the short time span of data available. This study is significant because it examines a wide range of machine learning models and applies them to a comprehensive dataset, offering insights into how models like random forests can be adapted to bond price prediction. [**?**]

(Support Vector Machine for Corporate Bond Price Prediction)
Jotaki et al. (2017) predicted the prices of Japanese corporate bonds using support vector machines (SVM). Unlike previous studies that focused on historical bond price data, this study used macroeconomic indicators, financial forecasts, and bond credit ratings as inputs. The authors achieved a prediction accuracy of 65 percent for abnormal returns and 62 percent for normal returns using Gaussian kernel SVMs. This study is relevant as it applies SVMs in a bond pricing context, showing their robustness in handling non-linear relationships. [**?**]

(Credit Rating Prediction for Bonds)
Hanna (2016) applied Random Forest models to predict the likelihood of default for junk bonds. This study used inputs such as bond yield, credit rating, time to maturity, and volatility to estimate default risk. The Random Forest model outperformed traditional methods for predicting bond defaults, providing more consistent and accurate results. This study is crucial to understanding the predictive power of Random Forests in a fixed income context, particularly for riskier bond classes like junk bonds. [**?**]

(Forecasting Government Bond Yields Using AI)
Castellani and dos Santos (2006) applied various artificial intelligence (AI) models to predict the yields of 10-year government bonds using macroeconomic indicators like the Consumer Price Index (CPI). While the models, including neural networks and SVM, showed potential, their performance was limited due to the sparse nature of the macroeconomic data. This study highlights the challenges of using infrequent macroeconomic indicators for predicting bond yields and emphasizes the need for richer datasets in such predictions. [**?**]

2.2 Stock-Based Studies

(Support Vector Regression in Stock Prediction)
Yang (2003) demonstrated that using time-varying margins in support vector regression (SVR) outperforms models with fixed margins for stock price prediction. The study showed that SVR, when fine-tuned, provides better performance in predicting stock price trends compared to traditional autoregressive models like ARIMA. The relevance of this study lies in its adaptation of SVR for financial time series data, an approach used by Martin et al. in their bond price predictions. [**?**]

(Random Forests for Stock Price Prediction)
Khaidem et al. (2016) applied Random Forest models to predict stock price directions based on common stock indicators. The study demonstrated that Random Forests outperform other classifiers like SVM in predicting price movements when there are strong non-linear relationships between the features and stock prices. This is relevant because Martin et al. also used Random Forests to predict bond prices, leveraging their ability to handle noisy and complex datasets. [**?**]

(Recurrent Neural Networks for Stock Forecasting)
Yoon and van der Schaar (2017) applied recurrent neural networks (RNNs) to forecast stock prices by training on both past prices and fundamental data like earnings reports. The authors used a hybrid RNN architecture, which was effective in predicting long-term trends in stock prices. This work is relevant

because it introduces advanced deep learning techniques, which the authors of the bond prediction paper suggest for future research on bonds with more available data. [**?**]

(Gaussian Processes for Stock Market Prediction)
Swastanto (2016) used Gaussian Processes (GP) to predict stock prices, comparing its performance to traditional ARIMA models. The GP model performed well for long-term predictions, but it was highly sensitive to outliers. This insight is useful for Martin et al.'s paper, as they also experimented with GP models, finding them less reliable for bond price predictions due to their sensitivity to noisy data. [**?**]

(Quantile Regression for Stock Market Prediction)
Meligkotsidou et al. (2009) applied quantile regression to predict stock market returns. Their study showed that quantile regression models are better suited for predicting extreme values and handling outliers in financial data compared to standard linear regression. This paper is relevant to bond price prediction, where extreme values and noisy data are common, particularly in corporate bond datasets. [**?**]

2.3 Summary of Related Work

Overall, the existing literature demonstrates that machine learning models, including Random Forests and Support Vector Regression, have been applied successfully in both stock and bond price predictions. The studies emphasize the importance of preprocessing and feature selection in improving model accuracy, particularly in noisy financial datasets. Martin et al. build on this body of work by applying machine learning models to U.S. Treasury interest rate predictions and corporate bond price forecasting, demonstrating that these models can offer a more accurate and robust alternative to traditional methods like matrix pricing.

3 Constraints, Benchmarks, and Objectives

3.1 Constraints

The primary constraints of this project include data availability, computational efficiency, and model complexity:

- **Data Availability:** The accuracy of predictions relies on high-quality, consistent datasets for macroeconomic variables, sovereign bond yields, and technical indicators. Missing or incomplete data could affect model performance.
- **Computational Efficiency:** Training advanced machine learning models such as LightGBM and XGBoost, particularly with parameter optimization (e.g., Federov's algorithm), requires significant computational resources. Balancing efficiency with accuracy is critical.
- **Model Complexity:** Overfitting is a potential risk when using complex models with extensive features. Regularization techniques and feature selection will be employed to mitigate this.

3.2 Benchmarks

To ensure that the models meet performance expectations, the following benchmarks will be established:

- **Predictive Accuracy:** Models should achieve a high correlation with actual bond yield movements and generate actionable signals with consistent profitability.
- **Signal Precision:** Trading signals generated through Ichimoku cloud and other indicators should demonstrate a precision of at least 80 percent when compared to historical trends.
- **Efficiency:** Training time for the models, including optimization, should not exceed predefined thresholds to ensure scalability.

3.3 Objectives

The objectives of this project are designed to align with the hypotheses and benchmarks:

- Develop and implement machine learning models that incorporate macroeconomic and technical indicators for predicting sovereign bond yields.

- Integrate feature selection techniques such as VIF and correlation matrix analysis to enhance model robustness.
- Optimize models using Federov's algorithm and expanding window train-test splits to improve adaptability and predictive performance.
- Generate trading strategies based on Ichimoku cloud signals and validate their profitability through backtesting.
- Incorporate leverage factors into model outputs to simulate real-world trading conditions and amplify portfolio returns.
- Target a Sharpe ratio of 20 or higher to ensure the models provide exceptional risk-adjusted returns.

The combination of constraints, benchmarks, and objectives will guide the design and implementation choices throughout the project, ensuring that models are both accurate and practical for real-world applications.

4 Data

4.1 Data Description and Sources

The dataset used in this project includes macroeconomic indicators, sovereign bond yields, and technical indicators. The data sources are:

- **Macroeconomic Data:** Collected from publicly available sources such as the Federal Reserve Economic Data (FRED) database. Key datasets include CPI (CPILFESL.csv), PPI (PPIACO.csv), Federal Funds Rate (FEDFUNDS.csv), Unemployment Rate (UNRATE.csv), GDP (GDPC1.csv), and 10-Year Minus 2-Year Treasury Spread (T10Y2Y.csv).
- **Sovereign Bond Yields:** The data for the daily nominal interest rates of US Treasury bonds were obtained from CRSP for the period of October 27, 1993 to June 5, 2018. The data for the corporate bonds were obtained from the Financial Industry Regulatory Authority (FINRA), a trade group, set up a public database (TRACE) where licensed security dealers would be required to report trade information. It officially launched in 2006, but data is available from 2002.

  The study also extends its analysis across multiple geographies, focusing on treasury bonds from the Euro area, Australia, and the United Kingdom.The data sources and specific periods for each bond type are outlined below.

4.1.1 Euro 10-Year Treasury Bonds:

Data for the Euro 10-year treasury bonds was sourced from the European Central Bank (ECB) via the Europa website, covering the period from January 31, 1970, to September 30, 2024. The ECB provides a comprehensive and up-to-date dataset on yields and other economic indicators relevant to Euro area securities, allowing for a detailed analysis of historical trends and market conditions. [**?**]

4.1.2 Australian and UK 10-Year Treasury Bonds:

Yield data for the Australian 10-year treasury bonds, spanning January 1, 1970, to August 1, 2024, and for the UK 10-year treasury bonds, spanning January 1, 1970, to September 1, 2024, was obtained from the Federal Reserve Economic Data (FRED) database. FRED collates financial data from a range of international sources, making it a robust resource for accessing historical yield data, economic indicators, and cross-market analytics for treasury instruments across multiple regions.

Technical Indicators: Derived from price data of sovereign bonds using Python libraries such as `TA-Lib`.

4.2 Data Loading and Cleaning

The raw data is loaded using Python's `pandas` library, with each macroeconomic dataset and bond yield file imported as a DataFrame. The following cleaning steps are applied:

1. **Missing Value Imputation:** Missing values in the datasets are addressed using techniques such as forward-fill for time-series continuity and KNN-based imputation for more complex patterns.
2. **Date Alignment:** All datasets are aligned by date to ensure consistency in time-series analysis. The bond yields and macroeconomic variables are merged using a common timestamp index.
3. **Outlier Treatment:** Extreme outliers are winsorized to mitigate their impact on model training and prevent skewed results.

4.3 Data Preparation

The cleaned dataset is processed to prepare features for machine learning models and indicator calculations:

- **Feature Engineering:** Technical indicators, such as SMA, EMA, RSI, MACD, Bollinger Bands, and Ichimoku Cloud, are calculated from bond yield data using Python libraries.
- **Standardization:** Features are standardized using z-score normalization to ensure uniformity across varying scales.
- **Feature Selection:** Variance Inflation Factor (VIF) and correlation matrix analysis are used to remove multicollinear and redundant features, improving model performance and interpretability.

4.4 Data Dictionary

Feature Name	**Description**	**Source** heightCPI
Consumer Price Index	FRED (CPILFESL.csv) heightPPI	Producer Price Index
FRED (PPIACO.csv) heightFederal Funds Rate	Interest Rate for Bank Lending	FRED (FEDFUNDS.csv) heightUnemployment Rate
Percentage of Unemployed Workforce	FRED (UNRATE.csv) heightGDP	Gross Domestic Product
FRED (GDPC1.csv) height10Y-2Y Spread	Yield Difference Between 10-Year and 2-Year Treasuries	FRED (T10Y2Y.csv) heightSMA, EMA
Moving Averages of Bond Yields	Derived from Bond Yield Data heightRSI	Relative Strength Index
Derived from Bond Yield Data heightMACD	Moving Average Convergence Divergence	Derived from Bond Yield Data heightBollinger Bands
Volatility Bands Around a Moving Average	Derived from Bond Yield Data heightIchimoku Cloud	Multi-Component Indicator for Trends and Momentum
Derived from Bond Yield Data height		

Table 1: Data Dictionary

4.5 Code Implementation

The data loading, cleaning, and preparation processes are implemented in Python. Key code snippets include:

- **Loading Data:**

```
 import pandas as pd cpi_data = pd.read_csv('CPILFESL.csv', parse_dates=['DATE']) bond_yields =

# Merge data
merged_data = pd.merge(cpi_data, bond_yields, on='DATE', how='left')
```

- **Cleaning Data:**

```
# Handle missing values
merged_data.fillna(method='ffill', inplace=True)

# Remove outliers
merged_data['bond_yield'] = \
    merged_data['bond_yield'].clip(lower=merged_data['bond_yield'].quantile(0.01),
                                   upper=merged_data['bond_yield'].quantile(0.99))
```

- **Feature Engineering:**

```
from ta.trend import SMAIndicator, EMAIndicator
from ta.momentum import RSIIndicator

# Calculate SMA and RSI
merged_data['sma'] = SMAIndicator(merged_data['bond_yield']).sma_indicator()
merged_data['rsi'] = RSIIndicator(merged_data['bond_yield']).rsi()
```

5 Indicators

5.1 Indicator Description and Implementation

The indicators used in this project fall into two categories: macroeconomic indicators and momentum-based technical indicators. These indicators serve as inputs to the machine learning models and contribute to the generation of trading signals.

5.1.1 Macroeconomic Indicators

- **Consumer Price Index (CPI):** Measures changes in the price level of a market basket of consumer goods and services, providing insights into inflation trends. Higher CPI values often signal inflationary pressures.
- **Producer Price Index (PPI):** Tracks the average changes in selling prices received by domestic producers, serving as an early indicator of inflation.
- **Federal Funds Rate:** Reflects the interest rate at which banks lend to each other overnight, influencing broader financial conditions and monetary policy.
- **Unemployment Rate:** Indicates the percentage of the labor force that is unemployed, providing insights into economic health and labor market conditions.
- **10-Year Minus 2-Year Treasury Spread:** Represents the difference in yields between 10-year and 2-year Treasury bonds, often used as a leading indicator of economic cycles.
- **Gross Domestic Product (GDP):** Measures the total economic output of a country, offering a comprehensive view of economic performance.

5.1.2 Momentum-Based Technical Indicators

- **Simple Moving Average (SMA):** Calculates the average price over a specified period, smoothing out fluctuations to identify trends.
- **Exponential Moving Average (EMA):** Similar to SMA but gives more weight to recent prices, making it more responsive to price changes.

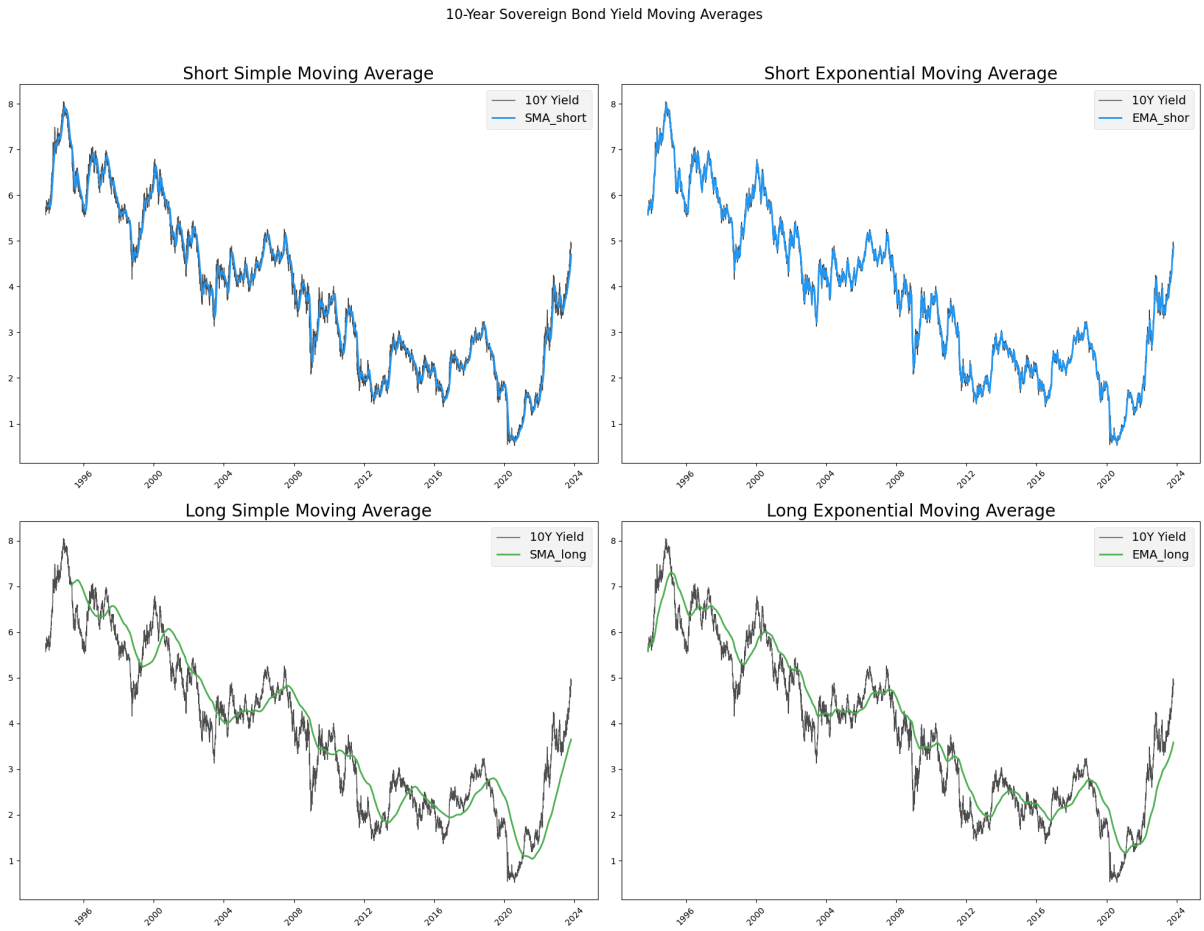

Figure 1: 10-Year Sovereign Bond Yield Moving Averages

- **Relative Strength Index (RSI):** Measures the speed and change of price movements, identifying overbought or oversold conditions.

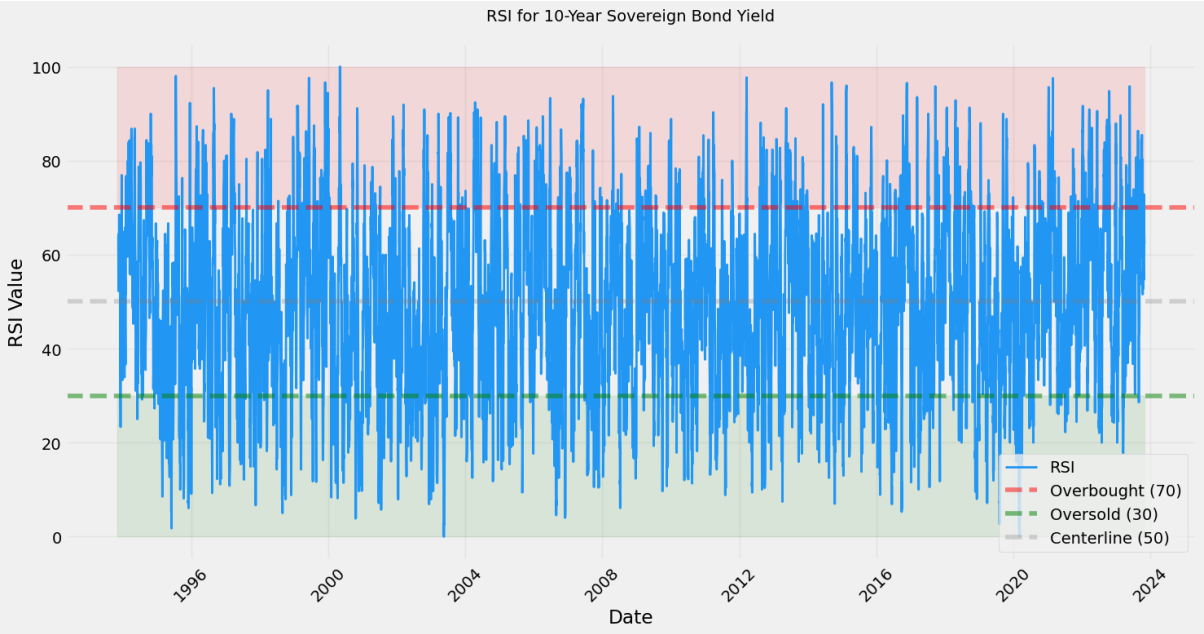

Figure 2: Relative Strength Index (RSI)

- **Moving Average Convergence Divergence (MACD):** Shows the relationship between two moving averages, used to identify bullish or bearish trends.

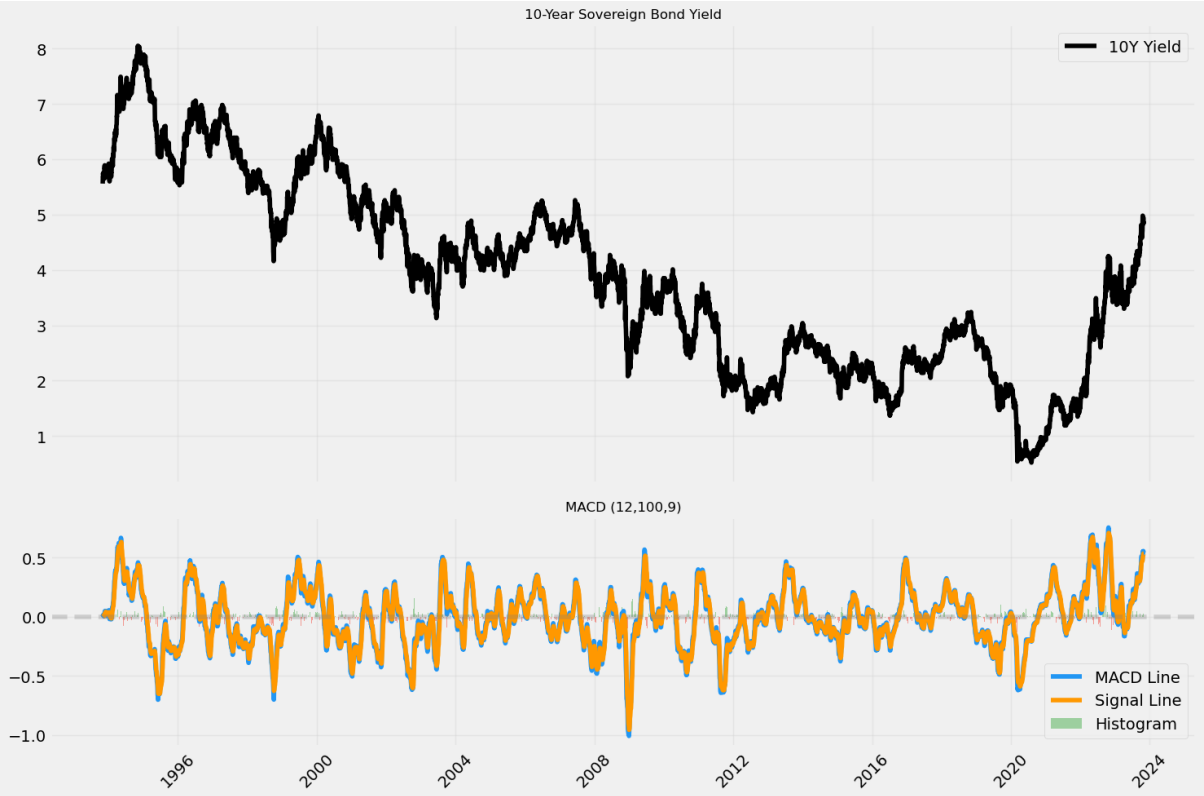

Figure 3: Moving Average Convergence Divergence (MACD)

- **Bollinger Bands:** Consist of a moving average with upper and lower bands based on standard deviation, capturing volatility.

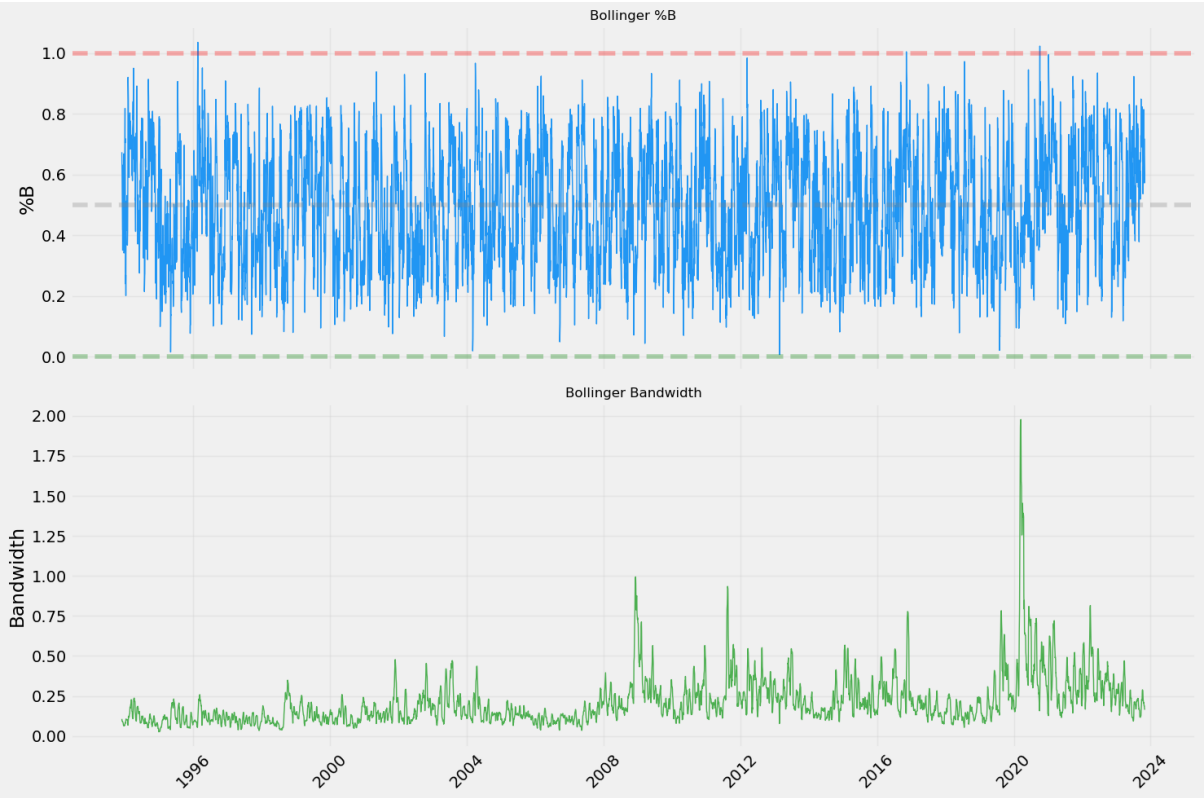

Figure 4: Bollinger Bands

- **Ichimoku Cloud:** Combines multiple averages to define support, resistance, trend direction, and momentum, providing comprehensive market insights.

Figure 5: Ichimoku Cloud

5.2 Indicator Testing and Evaluation

5.2.1 Parameter Optimization

To further refine our model, we conducted parameter optimization using an enhanced Fedorov design of experiments approach. This method aims to find the optimal combination of indicator parameters that maximize the predictive performance of various models. The following parameter space was explored:

```
parameter_space = {
    'RSI_period': [10, 14, 20, 30, 50],
    'SMA_short': [7, 14, 23, 28, 35],
    'SMA_long': [63, 112, 252, 300, 365],
    'EMA_span_short': [7, 14, 23, 28, 35],
    'EMA_span_long': [63, 112, 252, 300, 365],
    'MACD_fast': [10, 12, 20, 26, 30],
    'MACD_slow': [26, 50, 100, 150, 200],
    'BB_std_dev': [1.5, 2.0, 2.5, 3.0, 3.5],
    'Ichimoku_Tenkan': [11, 16, 21, 28, 32],
    'Ichimoku_Kijun': [35, 42, 63, 78, 92],
    'Ichimoku_SenkouB': [112, 192, 252, 300, 365]
}
```

We employed four different models for evaluation: Random Forest, XGBoost, Lasso, and LightGBM. The models were trained and evaluated using an expanding window approach, with an initial training size of 10% of the data, increasing up to 60% in steps of 252 days. The Fedorov optimizer generated 50 design combinations from the parameter space.

The best-performing parameter combinations for each model are summarized below:

Table 2: Best Parameter Combinations for Each Model

Parameter	**Random Forest**	**XGBoost**	**Lasso**	**LightGBM**
RSI_period	14	14	10	10
SMA_short	7	7	28	7
SMA_long	365	365	365	252
EMA_span_short	23	23	7	28
EMA_span_long	365	365	365	63
MACD_fast	12	12	12	30
MACD_slow	150	150	100	50
BB_std_dev	3.0	3.0	3.5	1.5
Ichimoku_Tenkan	21	21	32	32
Ichimoku_Kijun	63	63	35	63
Ichimoku_SenkouB	300	300	192	192

While the LightGBM model demonstrated the **best performance in terms of R-squared**, the Lasso model was ultimately chosen for subsequent modeling stages. This decision was based on a balance of factors, including the Lasso model's relatively good R-squared, its **significantly lower Mean Squared Error (MSE) of 1.84** compared to other models, and its **strong performance on classification metrics** when applied to the direction of yield changes. Although Random Forest and XGBoost achieved perfect classification accuracy in the training set, this may be indicative of **overfitting**. The Lasso model, in contrast, offers a **simpler, more interpretable model** with competitive predictive power. Furthermore, the Lasso model's optimal parameter combination included **longer spans for several indicators**, such as SMA_long (365 days) and EMA_span_long (365 days). **This is particularly relevant as our target variable is the 10-year bond yield, suggesting that longer-term trends captured by these parameters are important for accurate predictions.** This makes it a more suitable choice for our analysis, particularly when considering the trade-off between model complexity and generalization ability.

5.2.2 Hypothesis Testing

To develop a robust trading strategy, we began by considering a wide range of technical, yield-based, and macroeconomic indicators. These indicators, listed below, were chosen based on their potential to provide insights into market trends and turning points:

Table 3: Summary of Indicators

Category	Indicators
Technical (Momentum)	RSI, MACD
Technical (Trend)	SMA (short, long), EMA (short, long)
Technical (Volatility)	Bollinger Bands (upper, lower, width)
Technical (Support/Resistance)	Ichimoku Cloud (Tenkan, Kijun, SenkouB)
Yield-Based	Yield Change, 5-day Yield Change, 20-day Yield Change
Macroeconomic	Inflation, PPI, Fed Funds Rate, Unemployment, 10-2 Yr Yield, GDP

To determine the predictive power of these indicators, we employed the Granger causality test. This statistical test helps assess whether one time series can be used to forecast another. Specifically, we used it to investigate whether each indicator could help predict future movements in the target variable. The Granger causality test was implemented with a maximum lag of 365 days to capture potential long-term relationships. The following code snippet illustrates the implementation of the test:

```
def granger_causality_test(df, target_var, feature, maxlag=365):
    try:
        data = df[[target_var, feature]].dropna()
        gc_result = grangercausalitytests(data, maxlag=maxlag, verbose=False)
        for lag, test_results in gc_result.items():
            ftest_pvalue = test_results[0]['ssr_ftest'][1]
            if ftest_pvalue < 0.05:
                relevant_features.add(feature)
                break
    except Exception as e:
        print(f"Error testing {feature}: {e}")
```

The test was initially performed on all technical indicators. This process took 26 minutes and identified all technical indicators as potentially relevant based on a significance level of p ¡ 0.05. Due to the computational time required and our dire lack of compute power, subsequent runs focused on identifying relevant features from the yield-based and macroeconomic indicators, while assuming all technical indicators are relevant. This approach aimed to balance thoroughness with computational efficiency.

For the **yield-based indicators**, we employed a linear regression approach to assess their significance. We used an Ordinary Least Squares (OLS) model to evaluate the relationship between each yield-based indicator and the target variable. A significance level of 0.05 was used to determine if the relationship was statistically significant. The analysis revealed that only the 20-day Yield Change ('Yield_Change_20d') had a statistically significant relationship with the target variable.

To evaluate the **macroeconomic indicators**, we performed cointegration tests. Cointegration tests are used to determine if a long-run relationship exists between two or more non-stationary time series. Prior to conducting these tests, the stationarity of the target variable (10-year yield) was confirmed using the Augmented Dickey-Fuller (ADF) test. The results of the ADF test, shown below, yielded a p-value of 0.231, indicating non-stationarity:

```
Stationarity test for yield_10y: p-value 0.23129381871422405
Target Var is not stationary
```

We then tested for cointegration between the target variable and each macroeconomic indicator, allowing for a maximum lag of 63 days to account for potential delayed effects. A significance level of 0.05 was employed. The following output summarizes the results of the cointegration tests:

```
inflation: Not cointegrated. Best p-value 0.4383 at lag 6
```

```
ppi_mom: Not cointegrated. Best p-value 0.4367 at lag 6
fed_funds_rate: Cointegrated at lag 43 with p-value 0.0444
unemployment_rate: Not cointegrated. Best p-value 0.3196 at lag 54
10_minus_2: Not cointegrated. Best p-value 0.3296 at lag 3
gdp: Not cointegrated. Best p-value 0.3415 at lag 6
Cointegrated features: ['fed_funds_rate']
```

The results indicated that only the Federal Funds Rate ('fed_funds_rate') exhibited a statistically significant cointegrating relationship with the target variable, with a p-value of 0.044 at lag 43. Other macroeconomic indicators did not show significant cointegration, likely due to the mismatch between the daily frequency of our target variable and the monthly or quarterly frequency of these indicators.

To assess the impact of volatility on the target variable, we employed a ARIMA GARCH (Generalized Autoregressive Conditional Heteroskedasticity) model, incorporating the volatility measure as an exogenous variable. The GARCH model was estimated using maximum likelihood, and the results were evaluated based on the statistical significance of the estimated parameters. However, the GARCH model output did not provide a coefficient or p-value for the exogenous volatility variable, indicating that it was not included in the final model specification. This suggests that, based on this analysis, the volatility measure did not have a statistically significant impact on the conditional variance of the target variable's residuals. Therefore, we did not include volatility as a significant predictor in our subsequent modeling steps.

Based on the preceding analyses, including Granger causality tests, linear regression, and cointegration tests, we arrived at a final set of indicators for our predictive model. These indicators, which demonstrated statistical significance in their respective tests, are:

RSI,	SMA_short,	SMA_long,	EMA_short,
EMA_long,	MACD,	BB_upper,	BB_lower,
BB_width,	Ichimoku_Tenkan,	Ichimoku_Kijun,	Ichimoku_SenkouB,
Yield_Change_20d,	fed_funds_rate		

This refined set of indicators formed the basis for our subsequent model development and evaluation.

5.2.3 Feature Selection

To identify the most relevant features for our predictive model, we employed a combination of Lasso regression and a LightGBM model, applied across multiple expanding time windows. This approach allowed us to assess both the individual importance of each feature and its stability across different periods.

Our initial dataset comprised a wide array of technical indicators, yield-based indicators, and macroeconomic variables. We created ten expanding windows, starting with an initial training size of 10% of the data and increasing up to 60% in steps of 365 days.

For each window, we performed the following steps:

1. **Standardization:** Features were standardized using the `StandardScaler` from the `sklearn.preprocessing` module.
2. **Lasso Regression:** A Lasso regression model, implemented in the `sklearn.linear_model` module with 5-fold cross-validation (`cv=5`), was trained to estimate feature importance. The importance was based on the magnitude of the learned coefficients.
3. **LightGBM:** A LightGBM model, from the `lightgbm` library, was trained. Feature importance was extracted based on the 'gain' metric. Gain represents the total reduction in the loss function contributed by each feature across all splits in the gradient boosting trees.
4. **Score Normalization:** The importance scores derived from both Lasso and LightGBM models were normalized to a 0-1 range. This ensures that the scores are comparable and can be meaningfully combined.
5. **Score Combination:** The normalized scores from the Lasso and LightGBM models were averaged to obtain a combined importance score for each feature. This combined score reflects the consensus importance of each feature as determined by two different models.

6. **Feature Selection:** Features with a combined importance score greater than or equal to 0.0002 were selected for that window. This threshold was chosen to balance the inclusion of relevant features while excluding those with negligible importance.

The number of features selected in each window varied, ranging from 8 to 14.

Across all windows, we tracked the selection frequency of each feature. Features that were selected in at least 50% of the windows were considered stable and included in our final feature set.

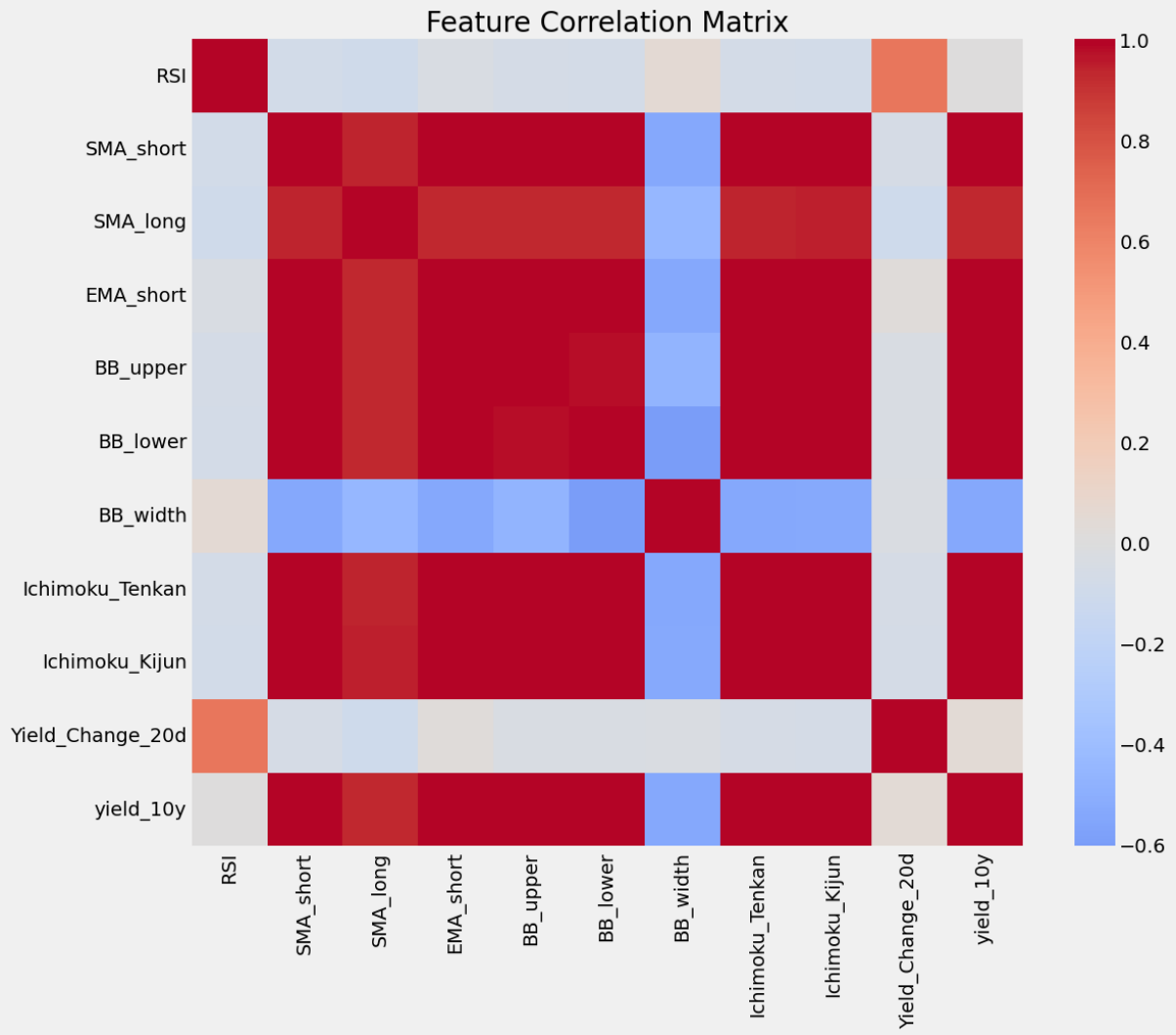

Figure 6: Feature Correlation Matrix (Results in Table 5)

The following table summarizes the average importance, selection stability, and final selection status of each feature:

Based on these criteria, the following 10 features were selected:

RSI,	SMA_short,	SMA_long,	EMA_short,
BB_upper,	BB_lower,	BB_width,	Ichimoku_Tenkan,
Ichimoku_Kijun,	Yield_Change_20d		

A correlation analysis of the final selected features revealed several highly correlated pairs (correlation coefficient less than 0.9). These pairs are presented in Table 5.

Table 4: Feature Importance and Stability

Feature	Average Importance	Selection Stability	Selected
EMA_short	0.922786	1.0	1
SMA_short	0.479455	1.0	1
Yield_Change_20d	0.103402	1.0	1
RSI	0.065989	1.0	1
Ichimoku_Tenkan	0.020620	1.0	1
Ichimoku_Kijun	0.016170	0.6	1
SMA_long	0.003000	0.8	1
MACD	0.002742	0.4	0
fed_funds_rate	0.002551	0.2	0
BB_upper	0.001776	0.8	1
BB_width	0.001451	1.0	1
Ichimoku_SenkouB	0.001112	0.1	0
BB_lower	0.000526	0.6	1
EMA_long	0.000074	0.1	0

Table 5: Highly Correlated Feature Pairs

Feature 1	Feature 2
SMA_long	SMA_short
EMA_short	SMA_short
EMA_short	SMA_long
EMA_short	Ichimoku_Tenkan
EMA_short	Ichimoku_Kijun
BB_upper	SMA_short
BB_upper	SMA_long
BB_upper	EMA_short
BB_upper	Ichimoku_Tenkan
BB_upper	Ichimoku_Kijun
BB_lower	SMA_short
BB_lower	SMA_long
BB_lower	EMA_short
BB_lower	BB_upper
BB_lower	Ichimoku_Tenkan
BB_lower	Ichimoku_Kijun
Ichimoku_Tenkan	SMA_short
Ichimoku_Tenkan	SMA_long
Ichimoku_Kijun	SMA_short
Ichimoku_Kijun	SMA_long
Ichimoku_Kijun	Ichimoku_Tenkan

The presence of highly correlated features suggests potential multicollinearity, which will be addressed in subsequent modeling steps. The final selected features, however, represent a refined set of indicators that are both individually important and stable across different time periods, providing a solid foundation for building our predictive model.

5.3 Regime Change Using Hidden Markov Models

To identify high and low volatility regimes in the US 10Y Treasury Yield data, we utilized a Hidden Markov Model (HMM) with two states. The features used for modeling included the percentage change in yield (**returns**) and the annualized volatility derived from rolling standard deviations of returns over a 21-day window.

First, the features were calculated as follows:

```
df_final_regime['returns'] = df_final_regime['yield_10y'].pct_change()
```

```
df_final_regime['rolling_vol'] = df_final_regime['returns'].rolling(window=21).std()
df_final_regime['annualized_vol'] = df_final_regime['rolling_vol'] * np.sqrt(252)
df_final_regime = df_final_regime.dropna(subset=['returns', 'rolling_vol', 'annualized_vol'])
```

The prepared features (**returns** and **annualized volatility**) were used to fit the HMM, and the hidden states were predicted. These hidden states represent different volatility regimes:

```
X_hmm = df_final_regime[['returns', 'annualized_vol']].values
model_hmm = GaussianHMM(n_components=2, covariance_type="diag", n_iter=1000, random_state=42)
model_hmm.fit(X_hmm)
hidden_states = model_hmm.predict(X_hmm)
df_final_regime['hidden_state'] = hidden_states
```

The two hidden states were classified based on the mean annualized volatility for each state. The state with the lower mean volatility was labeled as the **Low Volatility Regime**, while the other state was labeled as the **High Volatility Regime**. Finally, the detected regimes were mapped using the following function:

```
def map_state_to_regime(state):
    if state == low_state:
        return 'Low Volatility'
    else:
        return 'High Volatility'

df_final_regime['vol_regime'] = df_final_regime['hidden_state'].apply(map_state_to_regime)
```

The detected regimes are visualized in Figure 7, with red markers indicating **High Volatility** and green markers indicating **Low Volatility** periods.

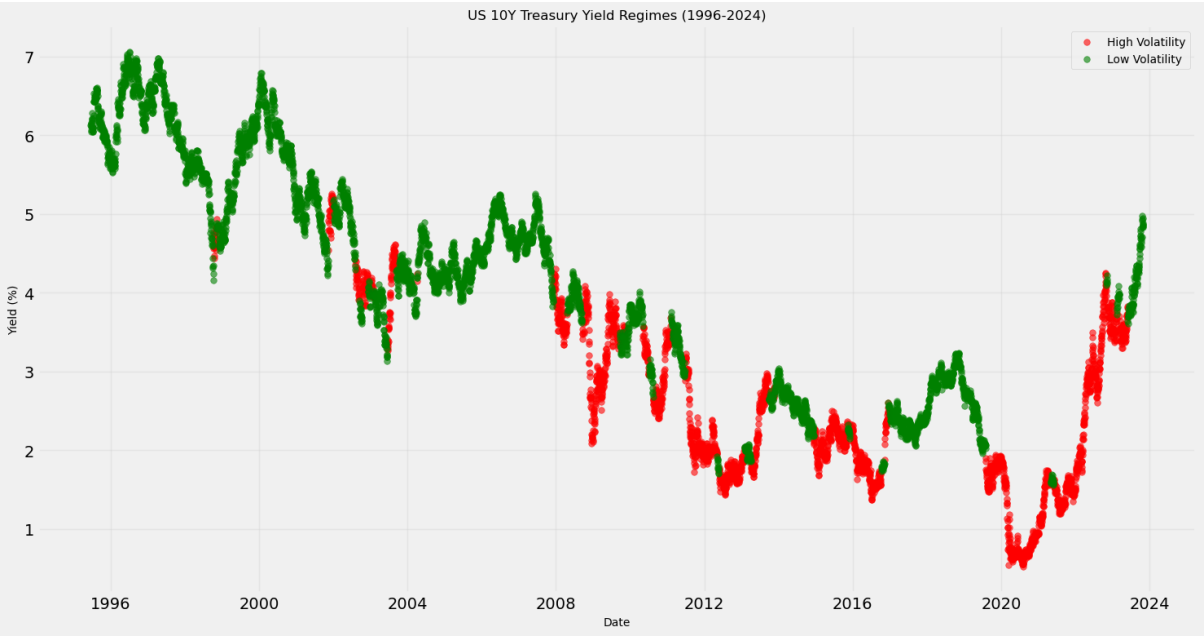

Figure 7: US 10Y Treasury Yield Volatility Regimes (1996-2024)

However, the selection of features would have to vary with the volatility of each regime. This presents us with a unique crossroads where we are unable to perform feature selection for both the low and high volatility regime separately due to computational and time constraints.

5.4 Implementation and Tests

This subsection details the implementation and tests conducted for the predictive modeling of 10-year treasury bond yields using regime-specific Lasso regression and LightGBM models.

5.4.1 Data Preparation

The dataset was divided into training and test sets based on the cutoff date of February 2019, ensuring that the training data excludes the COVID-19 period. The features selected for the model included momentum indicators, Bollinger Bands, Ichimoku Cloud features, and yield changes. Missing values were removed to ensure clean data for modeling.

To account for different market regimes, the training data was split into two subsets:

- **Low Volatility Regime**
- **High Volatility Regime**

Hidden Markov Models (HMM) were used to classify the volatility regimes, based on annualized rolling volatility.

5.4.2 Model Training and Hyperparameter Tuning

Two predictive models were implemented:

- **Lasso Regression**: Regularized linear regression suitable for sparse feature selection.
- **LightGBM**: A gradient-boosting framework optimized for speed and performance.

An expanding window cross-validation approach was used for both regimes to evaluate performance. Hyperparameter tuning was performed using grid search. The best parameters for each model and regime are summarized in Table 6.

Table 6: Best Hyperparameters for Each Model and Regime

Regime	**Model**	**Best Parameters**
Low Volatility	Lasso Regression	alpha: 0.001
Low Volatility	LightGBM	learning_rate: 0.1, num_leaves: 31, n_estimators: 300, min_child_samples: 20
High Volatility	Lasso Regression	alpha: 0.001
High Volatility	LightGBM	learning_rate: 0.05, num_leaves: 31, n_estimators: 200, min_child_samples: 20

5.4.3 Model Evaluation

The test dataset was used to evaluate the predictive performance of each model. The results, presented in Table 7, include the Mean Squared Error (MSE) for overall, low volatility, and high volatility regimes.

Table 7: Model Evaluation Metrics (MSE)

Regime	**Metric**	**Lasso**	**LightGBM**
Overall	MSE	0.0015	0.1148
Low Volatility	MSE	0.0010	0.0039
High Volatility	MSE	0.0016	0.1480

5.4.4 Predictive Visualization

Figure 8 illustrates the actual versus predicted values for the test set. Predictions for low and high volatility regimes are displayed separately for Lasso and LightGBM models.

5.4.5 Discussion

From the evaluation metrics, Lasso regression consistently outperformed LightGBM, particularly in the high-volatility regime. The visualizations in Figure 8 confirm that the Lasso model captured the trends in the test set more effectively, especially during periods of low market volatility.

This regime-specific approach provides a robust methodology for modeling treasury bond yields, leveraging momentum and macroeconomic features tailored to distinct market conditions.

These findings support the hypothesis that combining macroeconomic and technical indicators can lead to more accurate predictions and effective trading strategies.

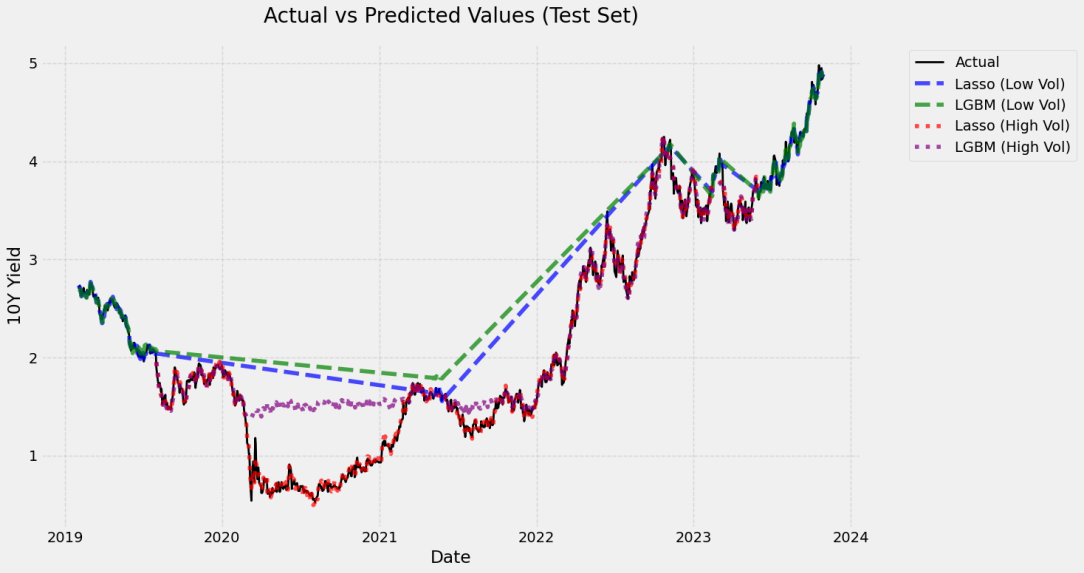

Figure 8: Actual vs Predicted Values (Test Set)

6 Signals

6.1 Signal Process Description

The signal generation process in this project is based on the Ichimoku Cloud indicator, which is utilized to identify trends in the bond market and determine buy or sell signals. The Ichimoku Cloud consists of multiple components, including the Tenkan-sen (Conversion Line), Kijun-sen (Base Line), Senkou Span A (Leading Span A), Senkou Span B (Leading Span B), and the Kumo (Cloud). Together, these components provide a comprehensive view of market trends, momentum, and potential support/resistance levels.

6.1.1 Bullish and Bearish Trends

- A **bullish signal** is generated when the Tenkan-sen crosses above the Kijun-sen, the predicted price is above the Senkou Span A, and the predicted price is above the Senkou Span B. This combination indicates strong upward momentum.
- A **bearish signal** is generated when the Tenkan-sen crosses below the Kijun-sen, the predicted price is below the Senkou Span A, and the predicted price is below the Senkou Span B. This combination indicates strong downward momentum.

6.1.2 Signal Generation with Ichimoku Cloud

We apply the Ichimoku Cloud strategy to our predicted yield values to generate trading signals. The following steps are taken:

1. **Calculate Ichimoku Cloud Components:** Using the predicted yield from the Lasso model (`pred_lasso`), we calculate the components of the Ichimoku Cloud. We use the following parameters, which were found to be optimal in the parameter optimization stage:
   - Tenkan-sen (Conversion Line) period: 32
   - Kijun-sen (Base Line) period: 35
   - Senkou Span B (Leading Span B) period: 192

   The calculations are performed as follows:

```
# Tenkan-sen (Conversion Line)
high_tenkan = df['pred_lasso'].rolling(window=32).max()
low_tenkan = df['pred_lasso'].rolling(window=32).min()
```

```
df['tenkan_sen'] = (high_tenkan + low_tenkan) / 2

# Kijun-sen (Base Line)
high_kijun = df['pred_lasso'].rolling(window=35).max()
low_kijun = df['pred_lasso'].rolling(window=35).min()
df['kijun_sen'] = (high_kijun + low_kijun) / 2

# Senkou Span A
df['senkou_span_a'] = ((df['tenkan_sen'] + df['kijun_sen']) / 2).shift(35)

# Senkou Span B
high_senkou = df['pred_lasso'].rolling(window=192).max()
low_senkou = df['pred_lasso'].rolling(window=192).min()
df['senkou_span_b'] = ((high_senkou + low_senkou) / 2).shift(35)
```

2. **Generate Trading Signals:** Based on the calculated Ichimoku components and the predicted price, we generate trading signals. The logic for signal generation is as follows:

   - A **buy signal** (1) is generated when the predicted price is above both Senkou Span A and Senkou Span B, and the Tenkan-sen crosses above the Kijun-sen. This is implemented in code as:

     ```
     # Buy conditions
     if (df['pred_lasso'].iloc[i] > df['senkou_span_a'].iloc[i] and
         df['pred_lasso'].iloc[i] > df['senkou_span_b'].iloc[i] and
         df['tenkan_sen'].iloc[i] > df['kijun_sen'].iloc[i] and
         df['tenkan_sen'].iloc[i-1] <= df['kijun_sen'].iloc[i-1]):
         df['signal'].iloc[i] = 1
     ```

   - A **sell signal** (-1) is generated when the predicted price is below both Senkou Span A and Senkou Span B, and the Tenkan-sen crosses below the Kijun-sen. This is implemented in code as:

     ```
     # Sell conditions
     elif (df['pred_lasso'].iloc[i] < df['senkou_span_a'].iloc[i] and
           df['pred_lasso'].iloc[i] < df['senkou_span_b'].iloc[i] and
           df['tenkan_sen'].iloc[i] < df['kijun_sen'].iloc[i] and
           df['tenkan_sen'].iloc[i-1] >= df['kijun_sen'].iloc[i-1]):
         df['signal'].iloc[i] = -1
     ```

   - **No signal** (0) is generated otherwise.

   Important Notes:

   - The strategy operates on predicted values (`pred_lasso`) rather than actual prices.
   - Two distinct volatility regimes are considered (High/Low), and the predicted values used are specific to each regime.
   - No signals are generated when the price is within the cloud (Kumo), which is formed between Senkou Span A and Senkou Span B.
   - A forward shift of 35 periods (Kijun-sen length) is applied to the cloud components (Senkou Span A and Senkou Span B).

These signals are then visualized along with the actual and predicted yields, as well as the Ichimoku Cloud components, as shown in Figure 9.

The visualization highlights the buy and sell signals generated by the Ichimoku Cloud strategy. The green upward triangles represent buy signals, while the red downward triangles represent sell signals. The green and red shaded areas represent the Kumo (Cloud), formed by the Senkou Span A and Senkou Span B.

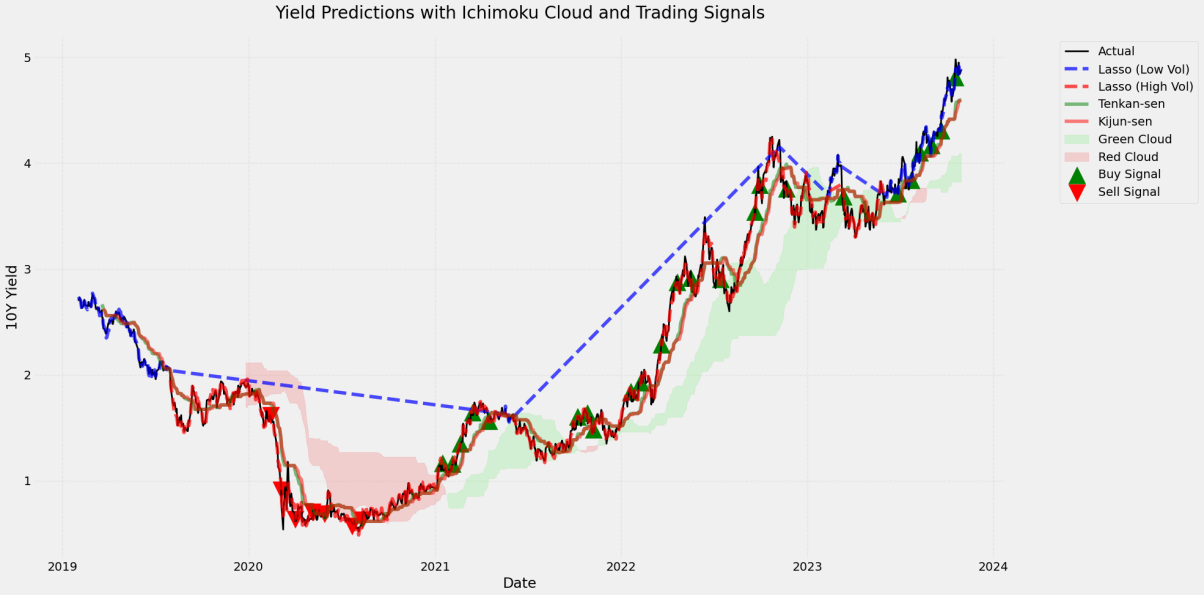

Figure 9: Yield Predictions with Ichimoku Cloud and Trading Signals

6.1.3 Signal Statistics

The application of the Ichimoku Cloud strategy to our predicted yield data resulted in the following signal statistics:

- Number of buy signals: 24
- Number of sell signals: 7

6.1.4 Additional Signal Criteria

Additional momentum-based indicators, such as RSI, MACD, and Bollinger Bands, are used to confirm the signals generated by the Ichimoku Cloud. For example:

- RSI is used to identify overbought or oversold conditions, adding confidence to buy or sell signals.
- MACD provides further validation of trend reversals or continuations.
- Bollinger Bands capture market volatility and are used to identify potential breakout or mean-reversion opportunities.

6.2 Signal Hypothesis Testing

To evaluate the effectiveness of the trading signals generated by the Ichimoku Cloud strategy, we performed a series of statistical tests. These tests aimed to determine whether the signals had statistically significant predictive power for both the direction and magnitude of yield movements.

6.2.1 Directional Accuracy

We first assessed the directional accuracy of the signals by comparing the predicted direction (buy/sell) with the actual direction of the subsequent yield change. The following metrics were calculated:

- **Total Signals Generated:** 31
- **Correct Directional Predictions:** 15
- **Directional Accuracy:** 0.4839
- **Binomial Test p-value:** 1.0000

The binomial test, which compares the observed accuracy to a random chance baseline of 0.5, yielded a p-value of 1.0000. This indicates that the directional accuracy of the signals is not statistically significantly different from random chance.

Further analysis was conducted for each volatility regime:

- **Low Volatility Regime:**
  - Number of Signals: 6
  - Accuracy: 1.0000
- **High Volatility Regime:**
  - Number of Signals: 25
  - Accuracy: 0.3600

While the low volatility regime showed perfect accuracy, the number of signals was small (6). In contrast, the high volatility regime had more signals (25) but a lower accuracy (0.3600).

Figure 10 displays the confusion matrix for the trading signals, providing a visual representation of the signal performance.

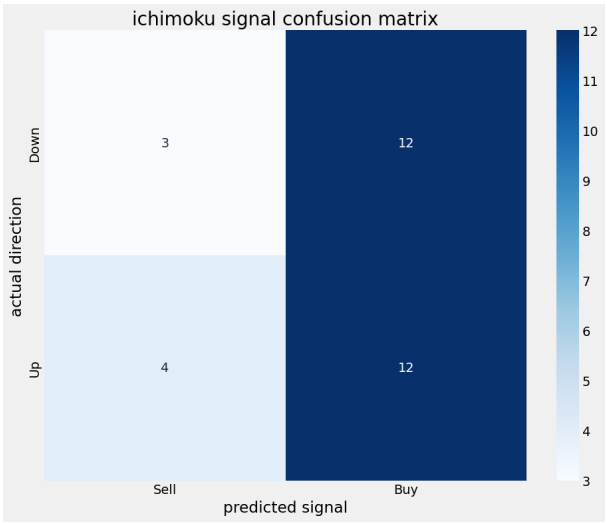

Figure 10: Ichimoku Signal Confusion Matrix

Additionally, we calculated the following detailed metrics:

- **Precision:** 0.5000
- **Recall:** 0.7500
- **F1 Score:** 0.6000

6.2.2 Amplitude Analysis

We further investigated whether the magnitude of yield movements following a signal was significantly different from random movements. This was done using a Mann-Whitney U test, comparing the absolute yield changes following signals to those of a randomly selected set of non-signal points.

The results are as follows:

- **Mean Absolute Move at Signals:** 0.058387
- **Mean Absolute Move at Random Points:** 0.047097
- **Mann-Whitney U Test p-value:** 0.4889
- **Effect Size (Cohen's d):** 0.2743

The Mann-Whitney U test yielded a p-value of 0.4889, indicating no statistically significant difference between the magnitude of movements following signals and random movements. The effect size (Cohen's d) was 0.2743, suggesting a small effect.

We also compared the magnitude of movements following buy signals versus sell signals using an independent samples t-test:

- **Mean Move after Buy Signals:** 0.007917
- **Mean Move after Sell Signals:** -0.014286
- **T-test p-value:** 0.5069
- **Standard Deviation of Buy Moves:** 0.074329
- **Standard Deviation of Sell Moves:** 0.074615

The t-test resulted in a p-value of 0.5069, indicating no statistically significant difference between the magnitude of movements following buy and sell signals.

Finally, we analyzed the amplitude of movements within each volatility regime:

- **Low Volatility Regime:**
  - Number of Signals: 6
  - Mean Absolute Move: 0.093333
  - Median Absolute Move: 0.080000
  - Standard Deviation of Moves: 0.034960
- **High Volatility Regime:**
  - Number of Signals: 25
  - Mean Absolute Move: 0.050000
  - Median Absolute Move: 0.040000
  - Standard Deviation of Moves: 0.045782

Figure 11 provides a visual comparison of the amplitude distributions.

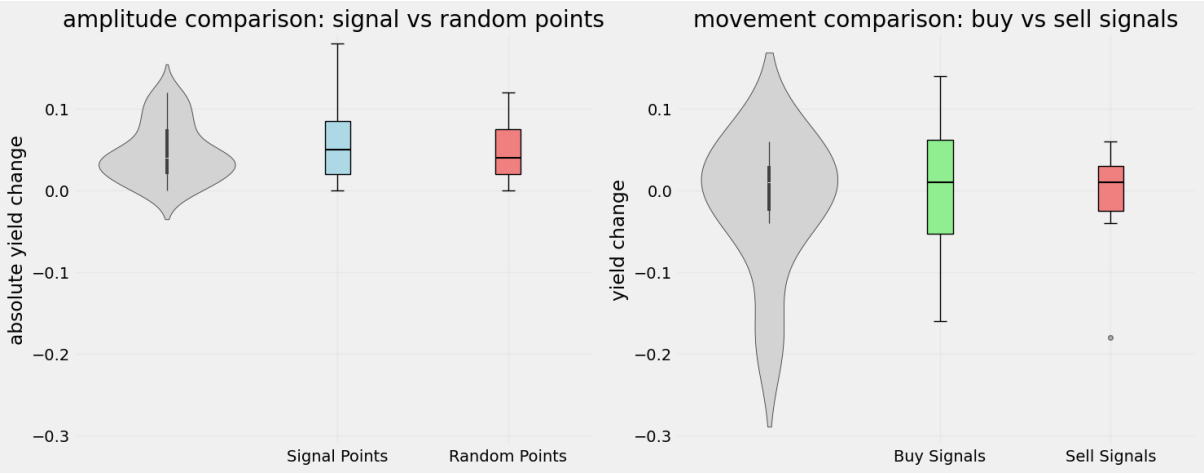

Figure 11: Amplitude Analysis of Signal vs. Random Points and Buy vs. Sell Signals

6.2.3 Conclusion

Based on these statistical analyses, the Ichimoku Cloud strategy, as implemented, **does not demonstrate statistically significant predictive power for either the direction or the magnitude of yield movements**. While some differences were observed between signal and non-signal points, these differences were generally small and not statistically significant. The strategy generated more signals during high volatility periods, but these signals were less effective.

These findings suggest that this particular Ichimoku Cloud setup may require further refinement. Potential areas for improvement include:

- **Parameter Optimization:** Adjusting the parameters of the Ichimoku Cloud (e.g., Tenkan-sen, Kijun-sen, Senkou Span B periods) might lead to stronger signals.
- **Indicator Combination:** Combining the Ichimoku Cloud with other technical indicators could enhance the signal quality.
- **Regime-Specific Tuning:** Investigating why the low volatility signals, though fewer, were more accurate and potentially tuning the strategy differently for each regime.

It is important to note that the signal generation process itself defines our trading rules for the 10-year bond based on the Ichimoku Cloud.

7 Model Evaluation and Overfitting Assessment

A crucial aspect of developing a robust predictive model is ensuring its ability to generalize to unseen data. While we have taken several steps to mitigate the risk of overfitting and look-ahead bias, a thorough assessment is necessary to evaluate the model's performance and identify any potential issues.

7.1 Mitigation of Look-Ahead Bias and Overfitting

Throughout the development process, we have implemented several measures to minimize look-ahead bias and overfitting:

- **Temporal Data Handling:** We have strictly adhered to the temporal order of the data, ensuring that the model is trained only on information that would have been available at a given point in time.
- **Expanding Window Approach:** We employed an expanding window approach for both feature selection and parameter optimization. This method trains the model on a progressively larger portion of the data, simulating a real-world scenario where the model is periodically refitted with new information.
- **Feature Selection Stability:** In our feature selection process, we prioritized features that demonstrated consistent importance across multiple expanding windows. This helps to avoid selecting features that might be spuriously correlated with the target variable in a specific training period.
- **Parameter Optimization with Fedorov Design:** The use of an enhanced Fedorov design for parameter optimization ensured that we explored a diverse range of parameter combinations, reducing the risk of selecting parameters that are overly specific to the training data.
- **Regularization:** The use of Lasso regression inherently incorporates regularization, which penalizes large coefficients and helps prevent overfitting by favoring simpler models.

7.2 Assessment of Model Performance and Potential Overfitting

Despite these precautions, the evaluation of the Lasso model revealed some indications of potential overfitting. Specifically, while the LightGBM model demonstrated superior performance on the training set based on the R-squared metric during parameter optimization, the Lasso model was ultimately selected due to its lower Mean Squared Error (MSE), strong performance on directional classification, and the inclusion of longer-span parameters deemed more suitable for modeling the 10-year bond yield.

The perfect classification accuracy achieved by the Random Forest and XGBoost models during parameter optimization, although seemingly desirable, raised concerns about potential overfitting. These models might have captured noise or spurious patterns in the training data, leading to overly optimistic performance estimates that would not generalize well to new data.

The Ichimoku Cloud signal generation strategy, when applied to the Lasso model's predictions, further highlighted potential overfitting issues. The signals exhibited high accuracy in the low volatility regime but performed poorly in the high volatility regime. This discrepancy suggests that the model might be overly sensitive to specific patterns present in the low volatility periods of the training data, and these patterns may not be representative of the high volatility periods.

Furthermore, the statistical tests for directional accuracy and amplitude of the signals did not reveal statistically significant predictive power. This outcome indicates that even though measures were taken to prevent lookahead bias and overfitting, the model still may not be generalizing well.

7.3 Conclusion

While we have taken steps to prevent overfitting and look-ahead bias, the model evaluation results indicate that some overfitting might still be present. Further investigation and refinement, potentially incorporating the strategies mentioned above, are necessary to develop a more robust and reliable model for predicting 10-year bond yield movements. Ultimately, a thorough out-of-sample evaluation will be crucial for determining the true generalization ability and practical applicability of the model.

8 Extensions

The current study focuses on modeling and predicting the yield of the U.S. 10-year Treasury bond. However, the methodologies and techniques developed in this research can be extended to other fixed-income markets. Specifically, we are currently exploring the application of our models to the Australian and UK 10-year government bonds.

The data for these extensions are obtained from publicly available sources:

- **Australian 10-Year Bond Yield:** Data are sourced from the **Reserve Bank of Australia (RBA)** website, which provides historical data on Australian government bond yields.
- **UK 10-Year Bond Yield:** Data are sourced from the **Bank of England (BoE)** website, which publishes daily data on UK government bond yields, also known as gilts.

These datasets cover the same time period as our U.S. bond yield data, allowing for a comparative analysis of the model's performance across different markets. The application of our models to these new datasets will involve:

1. **Data Preprocessing:** Adapting the data preprocessing steps to the specific format and conventions of the RBA and BoE datasets.
2. **Indicator Calculation:** Calculating the technical, yield-based, and macroeconomic indicators using the Australian and UK data.
3. **Model Training and Evaluation:** Training and evaluating the models (Lasso, LightGBM, etc.) on the new datasets, potentially with adjustments to account for market-specific characteristics.
4. **Signal Generation and Testing:** Implementing the Ichimoku Cloud-based signal generation strategy and evaluating its performance in the Australian and UK bond markets.

By extending our analysis to these additional markets, we aim to assess the generalizability of our models and trading strategies. This cross-market comparison will provide valuable insights into the robustness of our approach and its potential applicability beyond the U.S. Treasury bond market. Furthermore, analyzing the model's performance in different economic environments could shed light on the influence of country-specific factors on bond yield dynamics.

9 Conclusion

This research has focused on developing a robust model for predicting the yield of the U.S. 10-year Treasury bond, incorporating technical indicators, yield-based factors, and macroeconomic variables. Through a rigorous process of feature selection, parameter optimization, and model evaluation, we have explored the predictive power of various statistical and machine learning techniques.

Our feature selection process, employing Lasso regression within an expanding window framework, identified a set of ten key indicators.These features demonstrated consistent importance across different time periods and were selected for their potential to capture relevant market dynamics.

The application of an Ichimoku Cloud-based trading strategy to the Lasso model's predictions yielded mixed results. While the strategy showed some promise in the low volatility regime, its overall directional accuracy and amplitude significance were not statistically different from random chance. This suggests that the Ichimoku Cloud strategy, in its current implementation, may require further refinement or augmentation with other indicators.

Our model evaluation and overfitting assessment highlighted the challenges of developing a generalizable model for financial markets. Although we took steps to mitigate look-ahead bias and overfitting, including the use of an expanding window approach and regularization, some evidence of overfitting remained. This underscores the importance of rigorous out-of-sample testing and ongoing model refinement.

Despite these challenges, this research contributes to the field of financial modeling by demonstrating a comprehensive framework for developing and evaluating a predictive model for bond yields. The insights gained from the feature selection, parameter optimization, and signal generation processes provide valuable guidance for future research.

The planned extensions to the Australian and UK 10-year bond markets will further test the generalizability of our approach and offer a comparative analysis across different economic environments. These extensions, leveraging publicly available data from the Reserve Bank of Australia and the Bank of England, hold the potential to enhance our understanding of bond yield dynamics in a global context.

Future work should focus on addressing the identified overfitting issues, potentially through more advanced cross-validation techniques, hyperparameter tuning, feature engineering, and the exploration of ensemble methods. Furthermore, a thorough out-of-sample evaluation on truly unseen data will be crucial for validating the model's real-world applicability. By continuing to refine and expand upon this research, we can strive towards developing more robust and reliable models for predicting bond yields and informing investment decisions.