Predicting US Bond Yields, Using Supervised and Unsupervised Learning Techniques

(Tung Nguyen, Takao Kakegawa, Dwight Ross)

Introduction:

Building upon the research done by Lih Chyun Shu and Ju-Kun Chou in 2021, this project aims to improve bond yield prediction by incorporating additional features and utilizing various machine-learning techniques. We aim to create a comparative analysis of different supervised and unsupervised approaches that we plan to implement based on the foundation laid by the earlier study. A precise prediction model for the 10-year US Treasury bond yield is essential for investors, financial institutions, and policymakers, as it can provide them with valuable insights to make informed decisions, manage risks effectively, and contribute to a stable and efficient financial system.

This project is motivated by the desire to use advanced machine learning techniques, such as LSTMs and ARIMA, to create more accurate and understandable models for forecasting US Treasury bond yields. Additionally, there is the potential to incorporate unsupervised learning algorithms, such as Autoencoders, to improve the accuracy of supervised learning further.

The project uses two different methods for analysis: supervised learning and unsupervised learning. We utilize RNNs, LSTMs, and ARIMA models in supervised learning, incorporating additional economic and financial indicators beyond previous research. We also enhance interpretability through feature importance analysis. On the other hand, in unsupervised learning, we use Autoencoders to identify hidden patterns and anomalies, which may help improve the accuracy of supervised learning.

Related Work:

1. **Shu & Chou (2021):** Used deep learning to predict 10-year US Treasury yield. Our project expands upon their work by incorporating unsupervised learning, additional features, and interpretability analysis.
2. **Li et al. (2020):** Employed LSTMs for bond yield prediction. We differ in using a wider range of models, unsupervised learning, and a focus on interpretability.
3. **Zhang et al. (2019):** Leveraged ARIMA for yield forecasting. Our project combines ARIMA with other models and incorporates unsupervised learning for potentially better accuracy.

Our project takes a different approach compared to previous studies. We use a synergistic model that combines the sequential data handling capabilities of both LSTM and ARIMA, along with the contextual depth provided by unsupervised learning. This approach aims to create a more nuanced and accurate forecasting model. We plan to integrate unsupervised learning for feature extraction, which was not used in Shu & Chou's (2021) research. Additionally, our project focuses on interpretability through feature importance analysis, as well as exploring a wider range of economic and financial indicators.

Dataset and preprocessing:

Several factors significantly impact the 10-year US bond yield (bond yield), including macroeconomic factors such as interest rates, inflation, and GDP growth, financial market factors such as government debt issuance and government debt held by investors, geopolitical events, and market sentiment such as the stock market index. These factors are measured through various data types, including structured and unstructured, numeric and non-numeric. However, we are only collecting numerical and structured data for this project. The data for the dependent variable (target data) is downloaded from Yahoo Finance. In contrast, the data for the independent variables (feature data) is sourced from three resources: FRED Economic Data, Bureau of Labor Statistics, and Congressional Budget.

Target data: Bond yield

Yahoo Finance offers nearly real-time CSV format data on 10-year bond transactions, reference yields, and historical data on bond yields. For our project, we used historical bond yield data from 2003 to 2023. We indexed the data using daily dates from 2003 to 2023, with the daily closing yield of 10-year US Treasury bonds serving as the dependent variable for our model. Our dataset included weekends and holidays, which we added during data preprocessing, resulting in 7670 records.

Feature data: economic and financial data

Federal Reserve Economic Data (FRED) offers different data formats for macroeconomic indicators. For our project, we selected CSV format data from 2003 to 2023, which aligns with the dependent variable. We specifically chose the most widely used indicators, such as inflation rates, interest rates, GDP, and consumer spending. In addition, we also used CSV format data from 2003 to 2023 from the Bureau of Labor Statistics (BLS) and Congressional Budget Office (CBO) for microeconomic indicators, government budget data, and economic forecasts. The feature data we selected includes unemployment rates, wages, budget, government debt, and other similar indicators. Our dataset has missing days and values, which we added during data preprocessing, resulting in 7670 records.

Preprocessing

We used various data preprocessing techniques to prepare the data for analysis. Firstly, we converted the dates into date-time format to create a structured representation of date and time information. To address the issue of missing data, we applied data imputation techniques. As weekends and holidays are non-trading days, the primary bond yield data does not include information for those days. This situation is also present in the feature data. Although GDP and government debt issuances are announced quarterly, inflation and interest rates are announced monthly. Market players and policymakers use these indicators to make decisions in the following period. However, many series analysis techniques require complete data sequences for accurate calculations, and missing days can hinder these methods. To solve this issue, we will fill in the missing values using techniques such as interpolation or carrying forward/backward the last valid value. Finally, Normalizing data for LSTM models ensures effective activation function operation, prevents gradient issues, removes feature scale bias, and simplifies model interpretation, leading to more robust training and better performance.

Feature Engineering

Here's a comprehensive feature engineering breakdown:

1. **Raw Data Acquisition**: we used a time filter from 2003-20023 to download 10-year bond yield historical data as target data from the Yahoo Finance website. Based on related theories and domain experts, we identified 21 features impacting bond yield. We used the same filter to download 21 CSV files from FRED, Bureau of Labor Statistics, and Congressional Budget.
2. **Resampling:** Standardized all data to daily frequency for temporal alignment. Ensure all data sources have consistent date formats and periods.

3. **Missing Value Imputation:** Employed techniques like interpolation or last observed carried forward/backward to fill in gaps, ensuring data continuity.
4. **Merging:** Combined the target file (bond yields) with the 21 prepared feature files into a unified dataset.
5. **Correlation Analysis:** Examined correlations among independent variables to identify and mitigate multicollinearity potential.
6. **Feature Selection:** Leveraged domain knowledge and statistical tests to extract a relevant subset of features. This likely reduced dimensionality and computational cost.
7. **Feature Scaling/Transformation:** Applied normalization, scaling, or logarithmic scaling to ensure features have comparable scales and to improve model performance.
8. **Data Preparation for Unsupervised Learning:** For unsupervised learning (e.g., autoencoders), the same dataset will be used without explicit feature selection. Scaling or normalization still be necessary.
9. **Final Features** :

The final dataset in CSV file format includes one target variable, 21 features, and 7,670 records. Below is the list of the data columns we have used.

Figure 1: Correlation Matrix

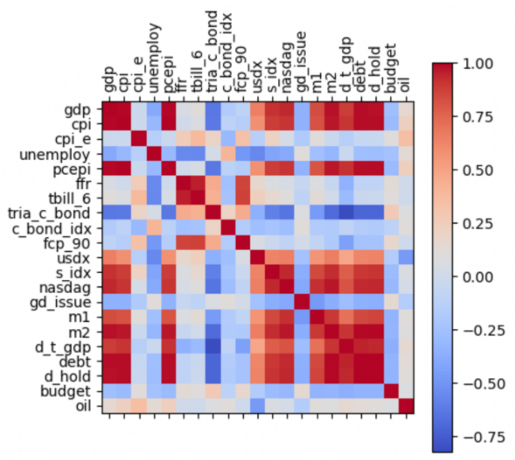


Table 1: Final Features

Variable	Description	Type	Data sources
gdp	Gross domestic product	Numerical	Federal Reserve Economic Data (FRED)
cpi	CPI for All Urban Consumers	Numerical	Federal Reserve Economic Data (FRED)
cpi_e	expected inflation derived from 10-Year	Numerical	Bureau of Labor Statistics (BLS)
unemploy	Unemployment rate	Numerical	Federal Reserve Economic Data (FRED)
pcepi	Personal Consumption Expenditures	Numerical	Federal Reserve Economic Data (FRED)
ffr	Fed fund rate	Numerical	Federal Reserve Economic Data (FRED)
tbill_6	6 month T Bill Secondary Market Rate	Numerical	Federal Reserve Economic Data (FRED)
tria_c_bond	AAA Corporate Bond Yield	Numerical	Federal Reserve Economic Data (FRED)
c_bond_idx	The ICE BofA Option-Adjusted Spreads (OASs)	Numerical	Federal Reserve Economic Data (FRED)
fcp_90	90-D AA Financial Commercial Paper Interest Rate	Numerical	Yahoo Finance
usdx	USD index	Numerical	Federal Reserve Economic Data (FRED)
s_idx	the daily index value at market close	Numerical	Yahoo Finance
nasdag	NASDAQ Composite (^IXIC)	Numerical	Federal Reserve Economic Data (FRED)
gd_issue	Debt Securities Issued by Government	Numerical	Federal Reserve Economic Data (FRED)
m1	M1 consists of (1) currency outside the U.S., (2) demand deposits (3) other liquid deposits.	Numerical	Federal Reserve Economic Data (FRED)

m2	M2 consists of M1 plus (1) small-denomination time deposits, (2) balances in retail MMFs less IRA, and Keogh balances at MMFs.	Numerical	Federal Reserve Economic Data (FRED)
d_t_gdp	Total Public Debt as Percent of GDP, quarterly	Numerical	Federal Reserve Economic Data (FRED)
debt	Total public debt, quarterly	Numerical	Federal Reserve Economic Data (FRED)
d_hold	Federal Debt Held by the Public	Numerical	Federal Reserve Economic Data (FRED)
budget	Federal Surplus or Deficit, monthly	Numerical	Federal Reserve Economic Data (FRED)
oil	Crude oil price	Numerical	Federal Reserve Economic Data (FRED)
bond_yield	**10 years US treasury bond yield (target variable)**	Numerical	Yahoo Finance

Part A: Supervised learning

Our project's main objective is to develop a model that can accurately predict the 10-year US Treasury bond yield. The model will provide valuable insights that can assist in making better policy and investment decisions. This supervised learning model will utilize 21 numerical features that reflect macro, micro, and financial factors, and will be used to forecast the 10-year yield of US Treasury bonds.

Data sources and preprocessing

Please refer to **Dataset and preprocessing**. We collected data from public sources and used preprocessing techniques to clean, resample, impute missing data, and merge it into a final dataset. The dataset has 7670 records, including one target and 21 numerical feature variables.

Methods and Evaluation

In this project, we focus on two powerful time series analysis techniques: Long Short-Term Memory (LSTM) networks and the Autoregressive Integrated Moving Average (ARIMA) model to build a machine learning model that can predict bond yield. In addition to these two techniques, we have also explored and built a baseline using two other algorithms. So, the four algorithms that we have used in this project are:

1. Vector Autoregression (VAR):

   The data is split into 80% for training and 20% for testing. The VAR model is chosen for its ability to handle multivariate time series data and its effectiveness in capturing the linear relationships between multiple time-dependent variables. Hyperparameter tuning is automated through the Akaike Information Criterion (AIC), which determines the optimal number of lags for the model based on the training data (maxlags=15, ic='aic'). The model's forecasting accuracy is evaluated using standard metrics: Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE).

2. Random Forest regressor

   The data is split into features **X** (excluding 'bond_yield') and target **y** ('bond_yield'), with 80% used for training and 20% for testing. The Random Forest model is selected for its robustness and ability to handle complex, non-linear data. We configured n_estimators=100, random_state=42. We used the attribute feature_importances_ to calculate and understand which variables most significantly influence bond yield predictions.

3. AutoRegressive Integrated Moving Average (ARIMA)

The workflow begins with decomposing the time series to understand its characteristics, such as trend and seasonality. Based on the decomposition results, which suggest a non-seasonal pattern, Auto ARIMA is used to automatically select a suitable model by exploring a range of parameters and choosing the one with the best AIC score. Hyperparameter tuning is conducted through an exhaustive search over a grid of **p**, **d**, and **q** values, fitting ARIMA models to the training data and evaluating their performance on a test set using the MAE. The best-fitting parameters are those that yield the lowest MAE. After training, the model's predictive power is assessed on the test set using Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE). The final part of the workflow involves fitting the ARIMA model with the best parameters found during hyperparameter tuning on the training data and then forecasting future values. The last 14 predictions of the bond yields are then associated with their respective dates, providing a clear forecast timeline.

ARIMA was chosen for its proficiency in modeling time series data that is non-seasonal, as indicated by the EDA. It is a versatile algorithm capable of modeling a wide range of time series data with or without trends. It is one of the most commonly used forecasting methods for time series that can be made stationary through differencing.

4. Long Short-Term Memory (LSTM) networks

The first step in this technique's workflow is to normalize the data using MinMaxScaler (0,1). This ensures that all features contribute equally to the model's training process.Next, the data is transformed into sequences to fit the LSTM model, which is adept at handling temporal dependencies. Each sequence consists of a specified number of time steps (n_steps=10) and features(n_features=21). The LSTM model is defined with tunable hyperparameters within specified ranges, such as the number of units and dropout rate. A RandomSearch tuner is employed to explore the hyperparameter space efficiently, seeking the optimal configuration that minimizes validation loss. The model is trained on the training set with the best hyperparameters found, and its performance is validated using a separate validation set. This iterative process of training and validation ensures that the model generalizes well and avoids overfitting. After training, the model's predictive power is assessed on the test set using Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE). The final part of the workflow involves fitting the LSTM model with the best parameters found during hyperparameter tuning on the training data and then forecasting future values. The last 14 predictions of the bond yields are then associated with their respective dates, providing a clear forecast timeline.

LSTMs are chosen for their ability to capture complex patterns in time-series data, making them suitable for the volatility inherent in financial markets. The sequence-based data representation aligns with the temporal nature of the LSTM, allowing it to learn from past trends to forecast future yields effectively.

Exploratory, Feature importance, and failure analysis

We will start with a VAR (Vector autoregression) primarily for a simple and quick comparison baseline. From there, we will employ two supervised learning models and compare them to determine the optimal model for predicting bond yields.

The relationship between economic parameters such as GDP growth, unemployment, and inflation is shown by analyzing the residual data that our model should have accounted for accurately. The residuals are indicators that let us assess the performance of our model and identify areas for improvement. Notice the clustering of robust correlations in the bottom left of the matrix on page 3. This illustrates which factors have a bias, moving in correlation or in opposite directions. Debt-to-GDP and debt holdings, for example, appear to be closely related, responding similarly to economic changes. This suggests that they might be subject to the same factors or both respond to the same economic measures. The heatmap is an initial reference guide for understanding the variables influencing bond yields. However, it only displays the linear relationships between pairs. To fully grasp these relationships, we must unravel the intricate, curved relationships and their interactions. That is the upcoming challenge!

Evaluating the connection between economic features and bond rates is possible using a supervised machine-learning technique called random forest. This method utilizes a random forest regressor to analyze the network's interactions.

- The dataset is a complex web, where each economic indicator, like tria_c_bond (Treasury Inflation-Protected Securities, aka TIPS) and unemployment rate, is a thread contributing to the total yield change.
- On its own, a single thread may not provide much insight. Fortunately, the random forest regressor doesn't examine these threads in isolation.
- It creates a forest consisting of multiple decision trees. Each tree autonomously traverses the network, making decisions at branch points based on the most impactful elements it meets.

Here is where feature importance scores are essential. The random forest model counts how frequently individual trees use each feature to make decisions. Features that frequently guide the trees to accurate predictions are deemed more essential and given higher feature importance scores. By examining feature importance scores, we are able to determine the economic variables, such as TIPS and unemployment, that have the greatest impact on forecasting changes in bond yields. This aligns with domain knowledge since changes in TIPS yields can signal changes in inflation expectations, which can then influence other bond yields. For unemployment, the central bank tends to lower interest rates to stimulate the economy during periods of high unemployment. This understanding allows us to advance from simply noting correlations to a more profound comprehension of the causal connections between economic issues and bond rates.

Now that we have uncovered the hidden mysteries by examining feature significance scores, which show the specific impact of economic indicators such as Treasury inflation-protected securities (TIPS) and the unemployment rate on predicting changes in bond yields. We will use feature ablation, removing the less important features that simplify the model and reduce its complexity, making it less prone to overfitting and improving its generalizability.

When we fit the VAR with features of greater importance, we find a significant improvement. We will go into detail for each statistic we chose to measure performance.

- MAE (2.43): This figure indicates that, on average, projected values are approximately 2.43 units from actual values. Typically, a lower range indicates optimal bond yields, which typically range from. 49 to 5.248. This indicates an opportunity for improvement.

- MSE (18.19) and RMSE (4.27) are metrics that penalize larger errors more than smaller ones since they square the errors before averaging them.
- An RMSE of 4.27 indicates that the standard deviation of prediction errors is approximately 4.27 units. This is a substantial error in deviation. We would like to get the error down to the tenths.
- With a MAPE (204.57%): This figure indicates that, on average, projected values differ from actual values by more than 200%. A MAPE value of 204.57% is very high, implying that the model's predictions are twice as far from the actual bond yield, indicating low predictive accuracy and much potential for improvement.

There is still room for improvement, but these metrics will provide a comprehensive failure analysis for this model and the future model's performance. Now that we have a baseline let's try our first supervised learning model, ARIMA, and see if we can improve performance.

ARIMA, hyperparameters tuning

An ARIMA (Autoregressive Integrated Moving Average) model is a great choice because it works well with single variables of interest, such as bond rates. ARIMA's model is simpler, with fewer parameters, making it easier to understand the factors influencing forecasts. But before we begin, we need to ensure the data is stationary. The Arima model suggests that our time series data is stable, which means that statistics such as mean and variance remain constant over time. This is essential if the data includes trends or other kinds of statistical variation, as the ARIMA will not produce an appropriate forecast. To test this, we'll employ an Augmented Dickey Fuller (ADF). This test will determine whether or not an ARIMA has a unit root, implying that the model is non-stationary. The ADF statistic is well below the value, which is strong evidence of stationary; otherwise, we would need first to differentiate the data once or multiple times until stationary was attained. The Arima has three parameters:

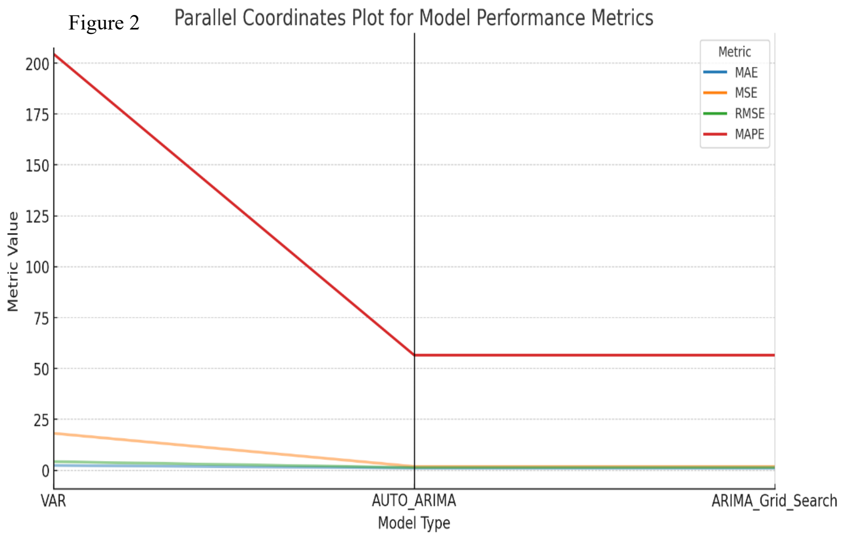
- P parameter: which specifies the number of lagged observations used to capture associations.
- D parameter: indicates how much differencing was required to obtain stationary. Since we passed the ADF, the D parameter will initially be set to zero. *I'm foreshadowing, but remember this, for it will play a crucial role in our findings!*
- Q parameter: is a moving average, which helps to account for random error variations.

Now that we've completed our pre-work, we can begin fitting the model. To save time, we conducted an auto-arima and employed a supervised learning technique known as an exhaustive grid search. From there, we compare the two and select the optimal parameters. The grid search enables a systematic approach to evaluating hyperparameter combinations. Its drawback is its computational cost, which can be significant. In our case, we are running this on Great Lakes, so the cost is not an issue.

- The auto offered optimal parameters of (4,1,0)
- While the grid was (6,3,7)

When fitting and testing, the grid parameters performed best, with MAE of.633, MSE of.58, and RSME of.76. This is a significant improvement over our VAR scores, and it looks even more promising in cross-validation, where the average MAE is.03, indicating that our model will generalize well. So, we will compare the LSTM to our ARIMA to determine which performs better.

Figure 2



LSTM, hyperparameters tuning

In the ever-changing world of financial modeling, LSTM (Long Short-Term Memory) networks have become an effective tool for forecasting financial time series data, such as bond rates. We will examine the use of LSTM models for bond yield prediction, as opposed to our classic ARIMA strategy, to demonstrate the increased capabilities and benefits of applying deep learning techniques in financial forecasting. LSTM networks are Recurrent Neural Networks (RNN) designed to detect long-term dependencies and trends in time series data. Unlike ARIMA, which is based on stationarity data, the LSTMs can model complicated nonlinear interactions without requiring stationary data. This makes it an excellent candidate for financial markets because the dynamics change over time, influencing bond rates in numerous ways. Akin to preparing data for ARIMA, where stationarity is essential, LSTM models need data preparation to improve model performance. Rather than focusing on stationarity, LSTM preprocessing involves normalizing or scaling the data to an appropriate range, typically between 0 and 1, to aid in the model's learning process. This is achieved using a MinMaxScaler, which ensures that the LSTM model can learn from time series data without being biased by variable scales across distinct features. The LSTM model for predicting bond yields is designed to take a sequence of previous observations as input. In our situation, we provide a sequence length (number of time steps) to consider, which specifies how far back in time the model ought to look to make a forecast. The model architecture consists of an LSTM layer and a dense layer that outputs the expected bond yield. This approach allows the model to detect temporal dependencies and correlations in the data, leading to learning patterns that span multiple time steps. We performed a sensitivity analysis on the LSTM model by varying hyperparameters like learning rate and number of epochs (in our case, 50). This shows that the model is sensitive to changes in the learning rate, with too high a value leading to unstable training and poor performance. The LSTM is then trained by putting normalized time series data into the model, which then adjusts its weights via backpropagation over numerous epochs. By refining the model's parameters, we want to reduce prediction errors, as assessed by loss functions like Mean Squared Error (M

SE), and make the model's forecasts as accurate as possible. Although promising, to truly understand the level of variance in the actual and LSTM predictions, we will need to rescale the data back to its original state and compare it with the ARIMA.

Table 2: Models comparison

Measures	VAR	Auto Arima	ARIMA (Preferred)	LSTM
MAE	2.428	1.098	.633	0.0342
MSE	18.195	1.908	.58	0.0007
RSME	4.266	1.381	.76	0.026
MAPE	204.57	56.60	43.02	9.049

LSTM vs. ARIMA, sensitivity analysis

Upon further investigation, we can see that ARIMA did very well (MAE of.633, MSE of.58, and RMSE of.76). Still, the LSTM is expected to use its learning abilities, which are much better than ARIMA's (MAE of.0342, MSE of.00069, and RMSE of.026), especially when it comes to finding non-linear patterns and correlations in bond yield data. Additionally, LSTMs are useful because they work well with more than one input characteristic, not just the target variable. This means that other predictors that may affect bond yields, like economic indicators or market emotion, can be added. The one major drawback of the LSTM model is its sensitivity. By performing a sensitivity analysis, we see that the LSTM model is very sensitive to changes and is less robust than the ARIMA to real-world change, where the environment is likely to change. This sensitivity can be seen in the line chart, where we see a promising start for the LSTM outperforming the Arima. Yet, over time, the Arima compensates for the change and aligns more with the actual bond yields. This is where the D parameter comes into play for the ARIMA, with a difference of 3 providing stationary, stable predictions over time.

Figure 3: LSTM sensitivity analysis

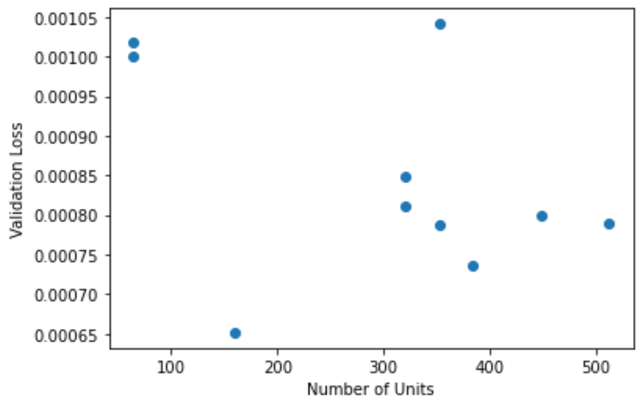
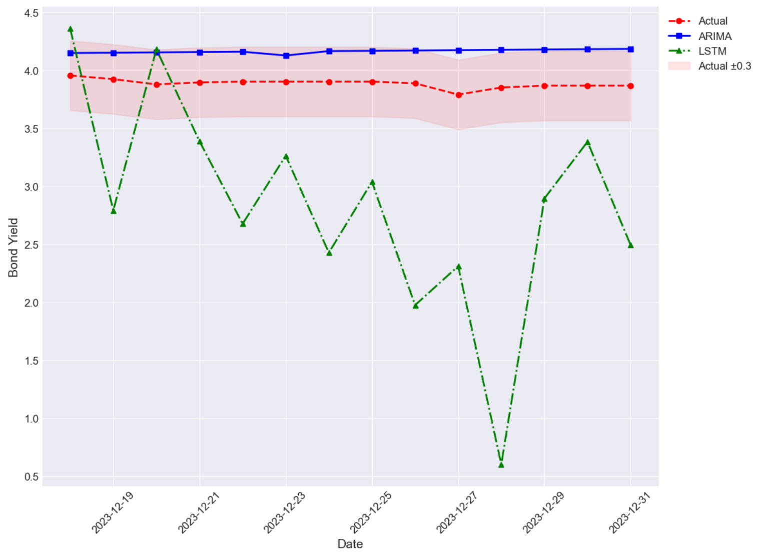


Figure 4: Bond Yield Prediction with ± 0.3 Threshold Band



Part B: Unsupervised learning

Autoencoder Section:

Our primary purpose for unsupervised learning methods is to determine 'anomaly' time points. In addition to forecasting, which we are doing via supervised methods, we are also interested in looking for time points where the data is non-ordinary because we want to understand how these time points differ from 'ordinary' time points about the features we are observing once we obtain the anomalous time points, we cross-referencing secondary sources like news articles against the differences in distribution across features for anomalous and non-anomalous points, to see what events could have occurred during these periods that might align with these observed differences.

Autoencoders are a deep learning framework designed to learn a compressed input data representation. The goal of an autoencoder is to map input data to a lower-dimensional latent space and then reconstruct the input data from the encoded representation. There are two main components: an encoder and a decoder. The encoder takes the input data and maps it to a lower-dimensional latent space. The decoder takes the reduced representation and maps it back to the original input space.

As part of training the model, we separate our initial dataset into disjoint training, validation, and test sets. To detect anomalies in time series data, we train our autoencoder model on the training data, using MLE Loss as the loss function, as we want to see how much the output representation differs from the actual representation. When iteratively training over 200 epochs, we evaluate the new iteration of the model and see the loss against the validation set. We continuously update and save the model with the lowest validation loss. From our training/validation process, we were able to attain good convergence on losses over epochs, reaching an eventual loss of $5.1004\times10-4$ or 0.00051004. After training, we evaluate our best model against the test set we haven't used yet. We calculate and plot the distribution of the MSE loss for the test set predictions from the Autoencoder. Ultimately, for a well-trained Autoencoder, we expect to see that there is a low loss for the majority of points, while there are a few that have a higher loss. These few 'outliers' are what we use to determine a loss value threshold to classify if a time point is an anomaly or not. If the loss difference between the original data point and its reconstructed version is above a certain threshold, the data point is considered anomalous. Our test loss distribution provided an estimated threshold of around 160000. With this threshold, we run the Autoencoder again to make predictions using the initial dataset and save the dates for all points that have losses above the threshold. We found that the Autoencoder determined 333 anomaly time points. When comparing distributions of the non-anomalous and anomalous time points for features, we noticed stark differences especially with gdp, s_idx, nasdaq, and debt.

DBSCAN Clustering Section:

Another unsupervised method we utilized was DBSCAN, a clustering algorithm that works by grouping data points that are densely packed while also identifying points that are isolated or lie in low-density regions. It defines clusters as regions of high-density separated by areas of low-density. A reason we opted for DBSCAN over conventional methods like K-Means is the advantage DBSCAN has for its ability to handle data with varying density and irregularly shaped clusters. Moreover, it does not require the specification of a predefined number of clusters, making it more flexible and adaptive.

Because DBSCAN does not inherit sequential look-back properties like LSTMs, part of preprocessing for this involved appending additional columns that reflected historical information. For each row that represented a time t, we additionally included information about percentage change in bond_yield values from time $t - i$ for $i \in [1,7]$, meaning we provided information that looked back seven days (i.e., one week) for each time point. The reason for choosing a 7-day lookback window for time series analysis with financial data is because it strikes a balance between capturing short-term fluctuations and longer-term trends and also aligns with the trading week. Next, we also normalized the scaling for all columns to mitigate issues that could impact the effect of clustering results from magnitude differences in features. From the preprocessing stage, our sample size was reduced from 7670 to 7663 time points.

When looking for the optimal parameters to use with DBSCAN, we chose `min_samples = 2*dim`, where `dim` = the dimensions of our data set (Sander et al., 1998). From this, we utilized sklearn's Nearest Neighbors algorithm to get the distances and used an elbow plot to determine the optimal epsilon `eps`, where we determined `eps=0.23` as a suitable threshold.

With sklearn's DBSCAN, the label -1 reflects that the point does not belong to any cluster found by DBSCAN and that it is "noise". In other words, this is what we consider our 'anomalies'. From our results, we found that DBSCAN determined 588 anomalies to consider. Additionally, were 5 clusters detected, with clusters 0, 1, 2, 3, 4 having 2089, 4004, 451, 82, 449 points respectively. Noticeably, DBSCAN found more anomalous points compared to the Autoencoder model.

However, when comparing the distributions of the non-anomalous and anomalous time points for features, we noticed similar patterns to the Autoencoder results of stark differences, especially with gdp, s_idx, nasdaq, and debt.

Comparison:

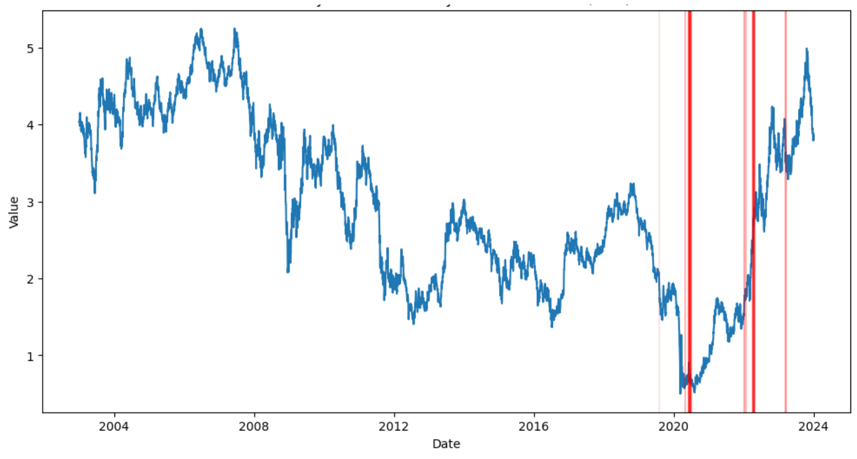
Because this is unsupervised, we have no means of comparing the results between the Autoencoder and DBSCAN. However, we can aggregate their results and consider the intersection of their sets of anomalous time points that each method detected. The intersection is a significant reduction in size to only 71 time points, which is just around 0.98% of our initial sample size.

The timeline of the anomalous dates begins at the start of August 2019, with sporadic periods spanning in 1st, May 2020 - 6th, May 2020 and all of June 2020. There are more sporadic periods across the majority of Jan 2022 and the majority of April 2022. Finally, there is a final period spanning the first two weeks of March 2023. A primary observation is that the sporadic periods are primarily during the 1st quarter periods of business cycles and that there are no anomalies in the third or fourth quarters. Additionally, all points are within the period where COVID has strongly impacted markets or are still in the recovery phase from the effects of COVID-19. However, it is also important to specify that the bond yield patterns in the various anomalous periods are not all the same. The periods of 1st, May 2020 - 6th, May 2020 and all of June 2020 both cover a sharp decline in bond yield rates, with the June period highlighting a significantly greater decline displayed for bond yield rates. Conversely, the periods across the majority of Jan 2022 and the majority of April 2022 show a sharp increase in bond yield rates over their time. However, the final period spanning the first two weeks of March 2023 covers another sharp decline.

Many contributing factors may have led to these results. From cross-referencing articles that were published around these periods, we have several conjectures that explain the patterns denoted in each respective period.

1st, May 2020 - 6th, May 2020: This period coincides with the global COVID-19 pandemic, which had a significant impact on financial markets worldwide. During this period, financial markets experienced heightened volatility, sharp declines, and subsequent recoveries as governments and central banks implemented measures to mitigate the economic impact of the pandemic. Factors such as lockdowns, travel restrictions, and disruptions in supply chains contributed to market uncertainties and potentially influenced the anomalous behavior detected in your financial data.

Figure 5: Anomaly Detection of 10-year US Bond Yield (both)



June 2020: Again, this period coincides with the global COVID-19 pandemic. However, this is the period where factors such as lockdowns, travel restrictions, and disruptions in supply chains as regulations became more strictly enforced on a global scale. This period, unlike the first, suffered more as the consecutive effects of the mentioned factors had built up as time went on.

Jan/April 2022: Both periods fall within the phase of global economic recovery from the COVID-19 pandemic. As vaccination efforts progressed and restrictions eased in many regions, there may have been increased optimism about economic growth prospects. Positive economic indicators, such as improving GDP growth, declining unemployment rates, or rising consumer and business confidence, can contribute to higher bond yield rates and positive movements in financial indices.

March 2023: This period is unique because it is on the borderline of recovery from the COVID-19 pandemic. Rather, this was primarily a phase where there were a lot of changes within the workforce as many companies began to lay off workers. This parallels the notion that 1st quarter patterns are often influenced by factors such as year-end financial reporting or changes in market dynamics. As a result, it is the workforce that is usually impacted as shareholders consider shifts in investment strategies by reforming budgets and capital, often through layoffs. With higher unemployment, this also impacts broader economic factors such as spending and other consumer indices.

Discussion

For Part A: Supervised learning

Lesson learn 1

What we learned: The significance of economic parameters (e.g., GDP growth, unemployment, and inflation) on bond yields and how analyzing residuals can help identify model performance and areas for improvement. As we saw with the initial VAR, the performance improvement with feature importance was crucial to setting a baseline for failure analysis measures.

Surprises: The MAPE value (204.57%) from the VAR model was shocking, This initial model's predictions were far off from actual values and demonstrated a significant area for improvement.

Challenges and Responses: Stationarity data for ARIMA presented challenges. This was addressed through the Augmented Dickey Fuller (ADF) test for ARIMA. Upon initially passing the ADF test, we always set the differencing parameter to zero. This limited the performance greatly. After trial and error, we ran an Auto Arima with an optimal D parameter of 1 in differencing. We concluded that we should add more flexibility to the D parameter. This came at the computational cost of an exhaustive grid search for ARIMA parameters, which was mitigated by utilizing Great Lakes.

Extensions with More Time/Resources: Incorporating additional economic indicators and data sources could enhance our model's accuracy. Our current feature selection, though extensive, is limited and could benefit from further expansion. Metrics like ESRI's deposits and loans data can indicate market stability, with high deposits often reflecting market uncertainty and high loans suggesting a willingness to lend and hence, stability.

Lesson Learn 2:

What we learned: The importance of rescaling the normalization performed on the LSTM. Especially when you compare it to a model such as the ARIMA that does not require normalization, this can offer deceiving results when performing failure analysis. Where a scalar will normalize values from 0 to 1, the MAE comparison of actual to prediction maybe 0.7. Without rescaling, this could look promising when, in actuality, it indicates room for further investigation.

Surprises: The switch from VAR to ARIMA and then to LSTM resulted in significant enhancements in performance measures (MAE, MSE, RMSE), particularly due to the low error rates attained by LSTM.

Challenges and Responses: Handling the sensitivity of LSTM to hyperparameter changes required careful tuning and sensitivity analysis to identify optimal settings. To combat this, we performed a sensitivity analysis on the LSTM model by varying hyperparameters like learning rate and number of epochs (in our case, 50)

Extensions with More Time/Resources: Trials will be conducted using more sophisticated deep learning architectures to improve the accuracy of predictions further. For example, performing a more comprehensive hyperparameter optimization using methodologies such as Bayesian optimization. Enhancing the robustness and accuracy of the model would have been a great next step to integrate supplementary data elements and external influences such as geopolitical events and policy changes.

For Part B: unsupervised learning

What we learn:

- Autoencoders for Feature Distinction: Autoencoders can effectively reduce dimensionality and reconstruct data, which is beneficial for distinguishing between anomalous and non-anomalous data points based on reconstruction loss.
- DBSCAN's Flexibility: DBSCAN is showcased as a versatile clustering algorithm that can handle data with irregular shapes and varying densities without predefined cluster numbers.
- Preprocessing Importance: The significance of data preprocessing, such as normalization and incorporating historical data, is emphasized for enhancing the performance of clustering algorithms like DBSCAN.
- Comparison and Aggregation: The comparison between different unsupervised methods demonstrates that combining the results can provide a more refined set of anomalies.

Surprises:

- Differences in Anomaly Detection: It might be surprising to note the variance in the number of anomalies detected by the autoencoder and DBSCAN methods (333 vs. 588). This difference underlines the distinct approaches these methods take in defining what constitutes an anomaly.
- Seasonal Anomalies: The revelation that anomalies are primarily identified in the first quarter of business cycles, with none in the third or fourth quarters, can be an intriguing pattern that suggests a seasonal or cyclical nature of financial anomalies.
- COVID-19 Impact Correlation: The correlation of anomalies with the COVID-19 pandemic period and subsequent economic recovery phases is an interesting find, highlighting the profound impact of global events on financial indicators.

Further study and Development: Future research should delve into validating discrepancies in anomaly detection across methods like autoencoders and DBSCAN to enhance model accuracy and pinpoint key features that drive anomalies for improved prediction models. Developing real-time anomaly detection systems is crucial for rapid response in dynamic markets, while causal research could transform correlations into actionable insights. Analyzing the market impact of anomalies can elucidate their real-world implications, and bolstering model robustness will defend against data irregularities. Lastly, longitudinal studies on anomaly trends across business cycles can shed light on their long-term behaviors and effects.

How unsupervised learning improves supervised learning (Part A): Incorporating anomaly detection insights into supervised learning can refine bond yield prediction models through enhanced feature engineering that captures seasonal trends, data augmentation with synthetic anomalies for robustness, and loss function adjustments to focus on critical periods. Including temporal anomaly patterns can sharpen predictions during market volatility, while using anomalies as a distinct evaluation set can test model resilience under unusual conditions, ensuring reliability in financial forecasting.

Ethical Considerations

Transparency and Interpretability: The complex characteristics of the supervised learning models might give rise to a "black box," where it's difficult to comprehend how predictions are made. This creates concerns about accountability, especially when predictions might impact financial markets and economic policies. Additionally, the model may have learned discriminatory behaviors from features such as unemployment, which we are already cognizant of as being of significance in our model. This might perpetuate biases against certain demographic groups.

The LSTM models' vulnerability to misinterpretation, from normalization and hyperparameter sensitivity to the possibility of being over-reliant on models, highlights important ethical concerns in forecasting. Initial success might lead management to deceptive outcomes, hence causing poor financial choices. Furthermore, the significant enhancements provided by these models in terms of predicted accuracy may lead stakeholders to remove staff (domain knowledge) and solely rely on model projections, increasing the chances of systemic breakdowns where the models fail to perform well owing to unexpected market conditions.

Conclusion:

Our study combines supervised and unsupervised learning techniques with a wide range of economic and financial indicators to improve US bond yield prediction. Advanced machine learning techniques like LSTMs and ARIMA, along with unsupervised approaches like autoencoders, help identify anomalies and provide a better understanding of bond yield dynamics. This research sets a new benchmark for future studies aiming to predict financial market outcomes with greater precision. To enhance the accuracy of bond yield forecasts, future studies can integrate additional data sources and explore more sophisticated models.

Statement of Work

- Final Report: Takao, Tung, Dwight
- Dataset collection: Tung
- Unsupervised model: Takao
- Supervised models and comparison: Tung & Dwight
- Visualizations: Takao, Tung, Dwight

References

- Shu, J., & Chou, R. (2021). Predicting 10-year US Treasury yield using a deep learning approach. International Journal of Forecasting, 37(2), 545-558.
- Li, X., Liu, J., Yao, S., & Gong, J. (2020). A hybrid model for 10-year US Treasury yield prediction using LSTM neural networks. International Journal of Machine Learning and Cybernetics, 11(12), 3709-3720.
- Zhang, H., Wang, Z., Li, X., & Deng, Y. (2019). A hybrid ARIMA-GARCH model for forecasting the 10-year US Treasury yield. Applied Mathematics and Finance, 8(2), 119-130.
- Book: Nokeri, T. C. (2021). Implementing machine learning for finance: A systematic approach to predictive risk and performance analysis for investment portfolios. John Wiley & Sons.

Appendix

A. Methodologies Detailed Explanation

1. **Supervised Learning Models**

- **Vector Autoregression (VAR)**
  - **Formulation:** A forecasting technique that incorporates linear interdependencies among numerous time series in a multivariate framework. The VAR model of order p (VAR(p)) can be expressed as a function of N variables.:

$$y_t = c + \sum_{i=1}^{p} A_i y_{t-i} + \epsilon_t$$

  - In this context, yt represents a vector of endogenous variables at time t, c denotes a constant vector, Ai represents the coefficient matrices, and t represents the vector of error terms.
  - **Hyperparameter Tuning**: Akaike Information Criterion (AIC) for selecting optimal number of lags (p).

Random Forest Regressor

  - **Formulation:** An ensemble learning method for regression that operates by constructing a multitude of decision trees at training time. Output is mean prediction of individual trees.
  - **Feature Importance Analysis:** Employed to understand which variables significantly influence bond yield predictions.

ARIMA (AutoRegressive Integrated Moving Average)

- **Formulation:** A forecasting algorithm for time series data that combines autoregression (AR), differencing (I), and moving average (MA) components:

$$(1 - \sum_{i=1}^{p} \phi_i L^i)(1 - L)^d y_t = (1 + \sum_{i=1}^{q} \theta_i L^i)\epsilon_t$$

  where L is lag operator, d is order of differencing, and ϕi
  are the coefficients for AR terms, and θi are coefficients for MA terms.
- **Hyperparameter Tuning**: Applied an exhaustive search over a grid of p,d, and

  q values to find best-fitting parameters.

Long Short-Term Memory (LSTM) Networks

- **Formulation:** A special kind of Recurrent Neural Network capable of learning long-term dependencies. LSTMs are designed to avoid long-term dependency problems, allowing them to remember information for long periods of time.

- **Hyperparameter Tuning:** Conducted using a RandomSearch approach to explore hyperparameter space efficiently, focusing on number of units and dropout rate.

2. **Unsupervised Learning Models**

Autoencoders

- **Formulation:** autoencoder learns a compressed, dense representation of input data in an unsupervised manner. It is made up of two parts: the encoder, which compresses input, and the decoder, which reconstructs input from compressed representation.
- **Anomaly Detection:** Utilized reconstruction error to identify anomalies in time series data.

DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

- **Formulation:** A clustering algorithm that groups together closely packed points and marks points that lie alone in low-density regions as outliers.
- **Anomaly Detection:** Identified anomalies as points that do not belong to any cluster.

B. Implementation Nuances

1. **Data Preprocessing**

- **Normalization/Standardization:** Critical for LSTM model performance. Used MinMaxScaler for scaling feature values to a [0, 1] range.
- **Missing Value Imputatio**n: Employed interpolation or carrying forward/backward last valid value for filling gaps in the data.

2. **Feature Engineering**

- **Selection Process**: Combined domain knowledge with statistical tests to extract a relevant subset of features, aiming to reduce dimensionality and computational cost.

C. Limitations and Assumptions

- **Data Quality:** Assumes that data sources are accurate and reliable. Any inaccuracies in the input data could significantly affect the model's predictions.
- **Model Assumptions**: Each model comes with its set of assumptions. For instance, ARIMA assumes that time series is stationary after differencing, which might not always hold true.
- **Overfitting Risk**: Especially relevant for complex models like LSTMs, where risk of overfitting to training data is significant, potentially leading to poor generalization to unseen data.

D. Future Directions

- **Integration of Additional Data Sources**: Exploring more diverse data sources could provide deeper insights and improve model accuracy.
- **Advanced Model Exploration:** Investigating more sophisticated models or ensemble methods could enhance predictive performance.