Domestic Stock Market Analysis

Amy Yucus, Annie Kittendorf, Dalton Bode, Ted Korby

Introduction

The United States domestic stock market holds enormous value and influences the US economy. In March 2021, its market capitalization was calculated to be over $49 trillion (GorillaTrades, 2021). With such value, it is important to understand what affects the domestic stock market and how this information may be used to form predictions.

This project uses Vanguard Index ETFs as proxies to analyze the domestic stock market and its top five dominant sectors. Sector dominance is determined by market capitalization. The individual ETFs examined are VTI, VGT, VCR, VIS, VHT, and VFH. They serve as proxies for the whole domestic stock market, as well as the following sectors: technology and information, consumer discretionary, industrials, healthcare, and finance. The project combines data for each of these funds with domestic economic data to model the market and analyze investment risk compared to expected returns and how historical prices compare to national historical revenue. It aims to answer the following questions: How do different machine learning algorithms compare when modeling the domestic stock market? Which economic features are the most predictive of market price? How does investment risk vary across sectors? How does return on investment vary across sectors? How do historical market prices compare to national historical revenue?

Understanding what factors influence the domestic stock market and how it can be modeled is crucial to making educated decisions. Businesses may incorporate this knowledge when deciding whether to go public and share prices, and investors may use this information to strategize trading. Analytical sector comparisons provide insight into what industries are favored in the economy, which supplies legislators information on how to shape economic policies.

Methods and Technologies

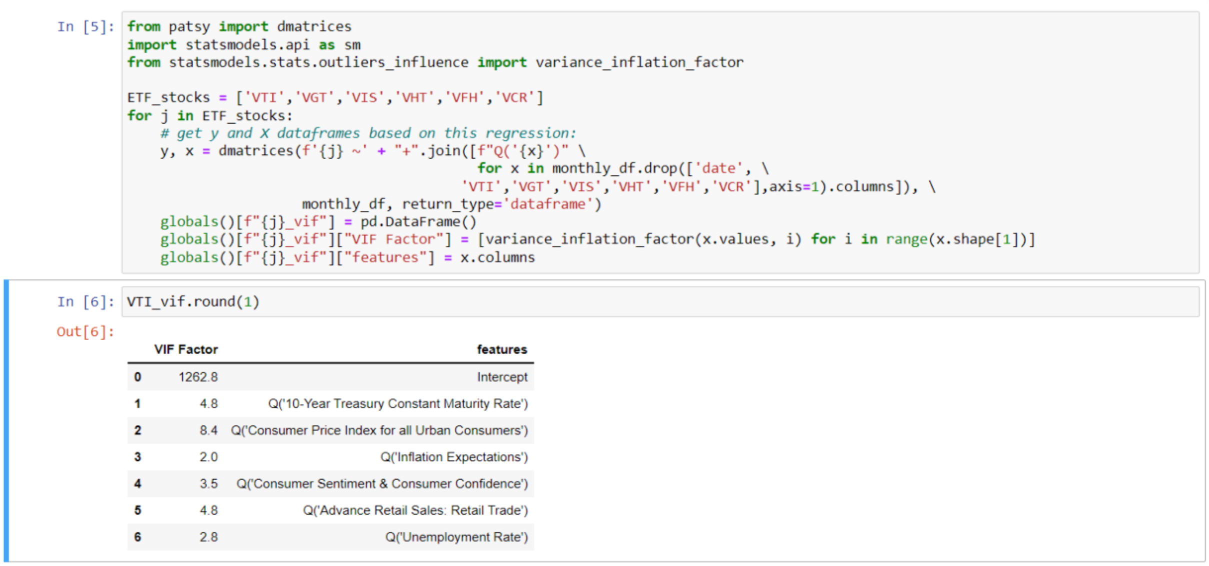
Data

We downloaded XSLX files containing yearly US revenue data from the US Census Bureau's website (United States Census Bureau, 2021a-k, 2022a-d). Each CSV file stood for a year between 2006 and 2020. The values for each CSV file were aggregated by summing the revenue values into a grand total and

The data for the monthly stock prices and the economic features was collected from the Alpha Vantage API (Alpha Vantage, 2022). We chose the monthly timescale as the economic features most granular timescale was monthly, making it unreasonable to combine the economic features data with daily stock prices. Each of the ETF stocks and the economic features were called from the API in the producer of our data factory. We had to implement a 12-second delay between each call in order to meet the API's request limit of 5 API calls per minute.

The economic features investigated in this report include: 10-Year Treasury Constant Maturity Rate, Inflation Expectations, Consumer Sentiment & Consumer Confidence, Advance Retail Sales: Retail Trade, and Unemployment Rate. We had initially decided to include two additional economic features, Real Gross Domestic Product (GDP) and Consumer Price Index (CPI) for all Urban Consumers. We used the Variance Inflation Factor (VIF) method to determine if our features were multicollinear. CPI had a VIF value above five, which showed that it was highly correlated with one or more features (see Figure 1). The Real GDP variable had 2/3 of its data imputed, so we all agreed it would be best to remove the feature. With those features removed, all other features were considered for investigation in this report.

Code Showing VIF Process



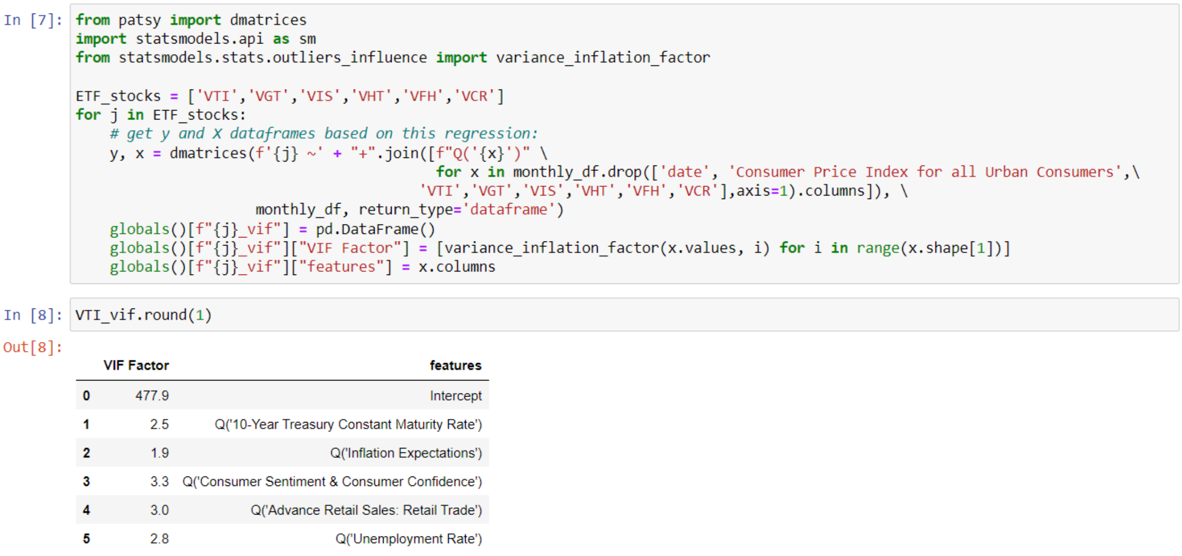
```
In [5]: from patsy import dmatrices
        import statsmodels.api as sm
        from statsmodels.stats.outliers_influence import variance_inflation_factor

        ETF_stocks = ['VTI','VGT','VIS','VHT','VFH','VCR']
        for j in ETF_stocks:
            # get y and X dataframes based on this regression:
            y, x = dmatrices(f'{j} ~' + "+".join([f"Q('{x}')" \
                                                  for x in monthly_df.drop(['date', \
                                                'VTI','VGT','VIS','VHT','VFH','VCR'],axis=1).columns]), \
                             monthly_df, return_type='dataframe')
            globals()[f"{j}_vif"] = pd.DataFrame()
            globals()[f"{j}_vif"]["VIF Factor"] = [variance_inflation_factor(x.values, i) for i in range(x.shape[1])]
            globals()[f"{j}_vif"]["features"] = x.columns

In [6]: VTI_vif.round(1)

Out[6]:
```

	VIF Factor	features
0	1262.8	Intercept
1	4.8	Q('10-Year Treasury Constant Maturity Rate')
2	8.4	Q('Consumer Price Index for all Urban Consumers')
3	2.0	Q('Inflation Expectations')
4	3.5	Q('Consumer Sentiment & Consumer Confidence')
5	4.8	Q('Advance Retail Sales: Retail Trade')
6	2.8	Q('Unemployment Rate')



```
In [7]: from patsy import dmatrices
        import statsmodels.api as sm
        from statsmodels.stats.outliers_influence import variance_inflation_factor

        ETF_stocks = ['VTI','VGT','VIS','VHT','VFH','VCR']
        for j in ETF_stocks:
            # get y and X dataframes based on this regression:
            y, x = dmatrices(f'{j} ~' + "+".join([f"Q('{x}')" \
                                                  for x in monthly_df.drop(['date', 'Consumer Price Index for all Urban Consumers',\
                                                'VTI','VGT','VIS','VHT','VFH','VCR'],axis=1).columns]), \
                             monthly_df, return_type='dataframe')
            globals()[f"{j}_vif"] = pd.DataFrame()
            globals()[f"{j}_vif"]["VIF Factor"] = [variance_inflation_factor(x.values, i) for i in range(x.shape[1])]
            globals()[f"{j}_vif"]["features"] = x.columns
```

```
In [8]: VTI_vif.round(1)
```

```
Out[8]:
     VIF Factor                                          features
  0       477.9                                         Intercept
  1         2.5          Q('10-Year Treasury Constant Maturity Rate')
  2         1.9                          Q('Inflation Expectations')
  3         3.3  Q('Consumer Sentiment & Consumer Confidence')
  4         3.0             Q('Advance Retail Sales: Retail Trade')
  5         2.8                             Q('Unemployment Rate')
```

Figure 1. VIF factors for each feature used in our machine learning models. Before the "CPI for all Urban Consumers" feature was removed (top), it had a VIF factor of 8.4, signifying high correlation with other features. Once the feature was removed, no other feature had a VIF factor above 5 (bottom), indicating all other features could be used for our machine learning models.

Technologies

In the project we utilized several technologies to simulate streaming data, implement machine learning models, and visualize our base and predicted data. The technologies we worked with are as follows: *Azure, Apache Kafka, SQL, Python, and Power BI*.

We used *Azure* for cloud computing and used services such as *Azure Resource Groups*, *Azure Databricks, Azure Data Factories, and Azure Data Lakes*. We worked within Dev10's Cohort Resource Group, using *Azure Databricks* as the integrated development environment for our Producer and Consumer within the cloud. This is greatly important since we will later run these files within an *Azure Data Factory* to simulate streaming and storing data in a Data Lake for preservation. The Data Factory is designed to run the Producer and Consumer Databricks once every 12 hours until stopped, regardless of an error being returned.

Since we are visualizing monthly data, the only values that will update in our desired input data frame are the current or most recent month's value of a given index in a set of stock indexes which will occur once every 24 hours. Due to the nature of this data stream, Apache Kafka was the streaming service used for streaming data between our producer and consumer which allowed us to automate the ETL process. We sent the API data via messaging from the producer to the consumer as JSON objects to replicate streaming data. The JSON objects were combined into two separate Pandas data frames and loaded into our SQL Database using PySpark.

After processing the data, we set up an Azure SQL Database and tables to store the structured data in for visualization purposes. Before starting the project, our team agreed to load the data into the database utilizing the PySpark library, as the dataset is miniscule in size

(< 1 MB). In the future, we will make sure to use DDL to create SQL tables for consistent loading between multiple consumers.

Most of the coding in this project revolves around the application of machine learning models such as Long-Short Term Memory, ARIMAX, SARIMAX, Linear Regression and Lasso CV, which are all supported models within the Python community. Due to this fact and our experience with the language, Python was the primary language used for this capstone project. A special mention to the following libraries used that made this project possible: *Pandas, Matplotlib, NumPy, Tensorflow/Keras and SKlearn*.

Lastly, we used Power BI for visualizations and to create reports replicating a dashboard. A portion of the visualizations in our report are Python generated.

Machine Learning Models

Several machine learning algorithms were implemented to address the question of how different machine learning algorithms compare when modeling the stock market. These include Linear Regression, LASSO CV Regression, ARIMAX, SARIMA, and LSTM. Each model used VTI stock price as the target variable to predict. Each model was trained on 95% of the dataset and tested on the remaining 5%. Further details about each model are below.

Linear Regression and Lasso CV

Linear regression is a commonly used machine learning model that calculates a line that best fits a distribution of data points representing a target variable and a feature of interest. The formula used to calculate the line of best fit is the least square fit method. This method sums up the distance from each point to the predicted line, squares those distances, and tries to find the minimum value of that summation. The formula used to calculate this line can use a single variable or multiple variables, depending on how many features you want to consider for the machine learning model. In our case, we investigated 5 features in both the Linear Regression and Lasso CV models. Lasso CV is also a linear regression model, but it has additional regularization parameters that are multiplied to each variable. With these, any features that show no relationship with the target variable are shrunk in magnitude. (Kumarl, 2020) The Lasso CV model conducts cross-validation by splitting up data based on the hyperparameter, α. For that reason, it is the only model in this report that doesn’t require splitting up datasets into training and testing sets.

For large datasets, Lasso CV is typically used over linear regression for quicker processing speed, thanks to features being filtered out. It can have higher accuracy than linear regression when both models are fit to data that rapidly fluctuates, as well. In most other cases, datasets that can be expressed by a linear relationship and are small (< 1 MB of data) use linear regression as the model of choice for accuracy.

ARIMAX

Auto Regressive Integrated Moving Average with eXogenous features (ARIMAX) is an algorithm commonly used with time series data to better understand the data and predict future values (The Pennsylvania State University, 2022). The model is auto regressive (AR) in that it predicts future values based on the data's earlier values (The Pennsylvania State University, 2022). It is integrated (I) in that it uses order differencing of the time series data to make it stationary (The Pennsylvania State University, 2022). Moving average (MA) references the model's incorporation of past errors when making predictions (The Pennsylvania State University, 2022). Exogenous features (X) references the use of features other than the target variable to predict future values (The Pennsylvania State University, 2022).

ARIMAX takes in three parameter s, *p*, *d*, and *q. p* corresponds to the AR part of the model and is decided by the data's partial autocorrelation (The Pennsylvania State University, 2022). *d* corresponds to the I part of the model and is the order of differencing at which the data is stationary (The Pennsylvania State University, 2022). *q* corresponds to the MA part of the model and is decided by the data's autocorrelation (The Pennsylvania State University, 2022).

The ARIMAX model for this project used 7, 1, and 7 as the *p*, *d, q* values, respectively. The original VTI prices were not stationary, as shown by the positive linear trend in price as time progresses (Figure 2, Augmented Dickey Fuller test p-value = 0.97). However, when the data is differenced by an order of 1, it is stationary (Figure 2, Augmented Dickey Fuller test p-value = 0.0). For both the autocorrelation and partial autocorrelation plots of the first-differenced VTI prices, 7 is the only number of lags other than 0 where there is statistical significance with alpha=0.05 (Figure 3, Figure 4). The model used the VTI stock price as the endogenous feature and 10-Year Treasury Constant Maturity Rate, Inflation Expectations, Consumer Sentiment & Consumer Confidence, Advance Retail Sales: Retail Trade, and Unemployment Rate as exogenous features.

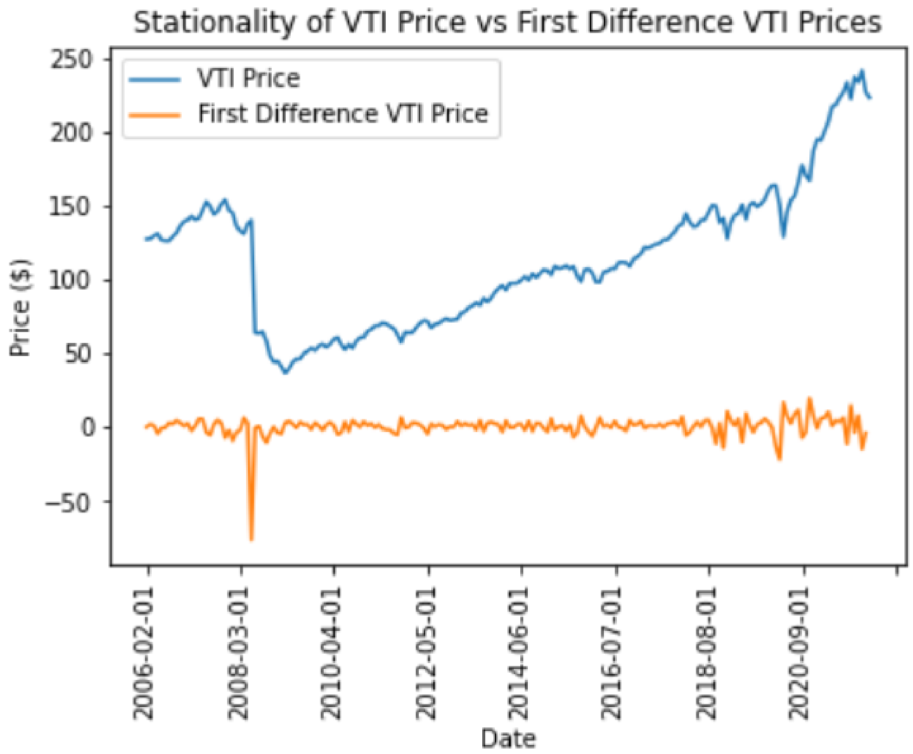


Figure 2. Graph of the VTI prices (blue) and first difference VTI prices (orange) across time. Augmented Dickey Fuller tests for stationality yield p-values of 0.97 and 0.0, respectively.

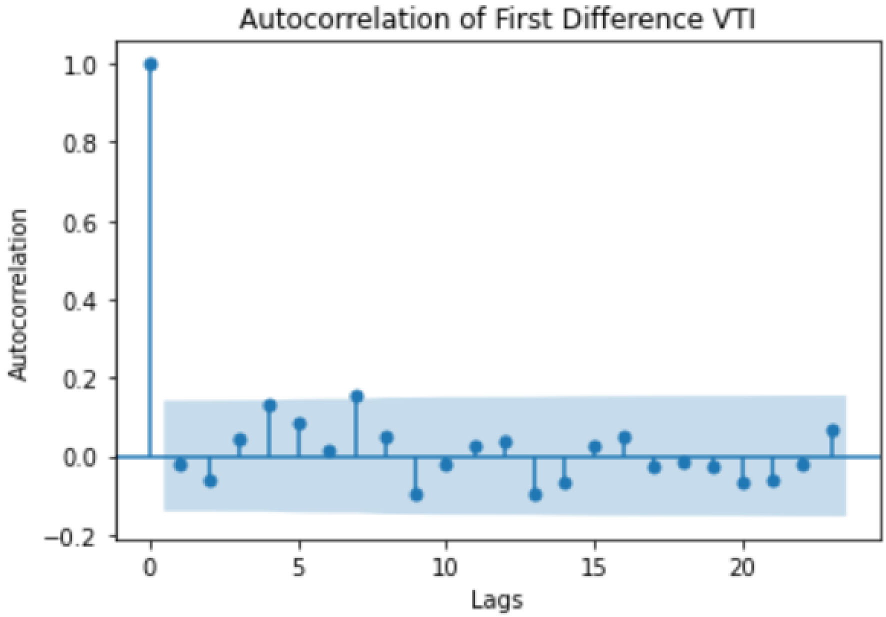


Figure 3. Autocorrelation of the first difference VTI prices. The blue band indicates the confidence band of 95%. Values outside the band are statistically significant and indicate appropriate values for the q parameter of ARIMAX.

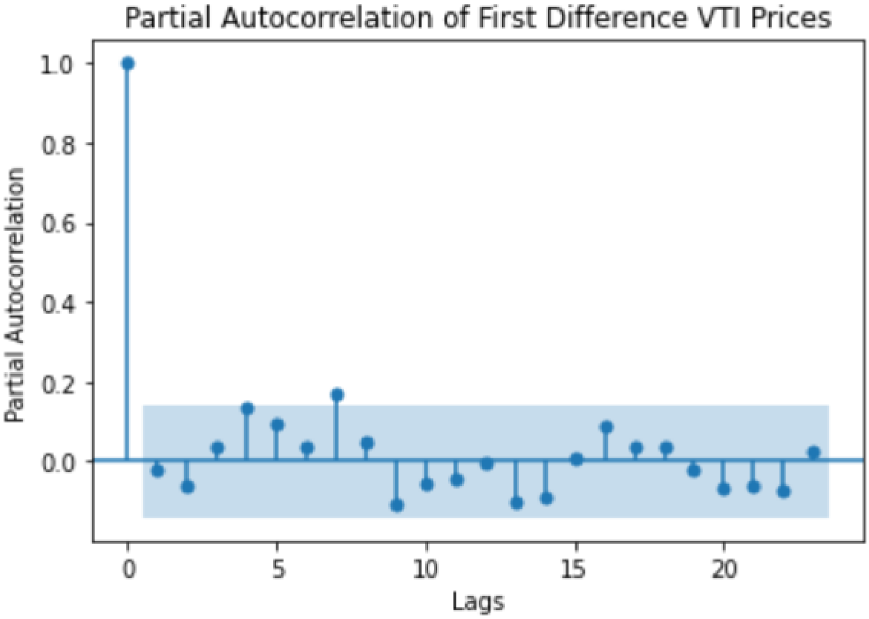


Figure 4. Partial autocorrelation of the first difference VTI prices. The blue band indicates the confidence band of 95%. Values outside the band are statistically significant and indicate appropriate values for the p parameter of ARIMAX.

SARIMAX

The SARIMAX model is an auto-regressive moving average model that takes seasonality and exogenous factors into account. It assumes that the data is seasonal, so we had to run a seasonal decomposition to do a preliminary check for seasonality. The third graph in the figure below shows the basic cyclical pattern of the VTI prices.

Seasonal Decomposition

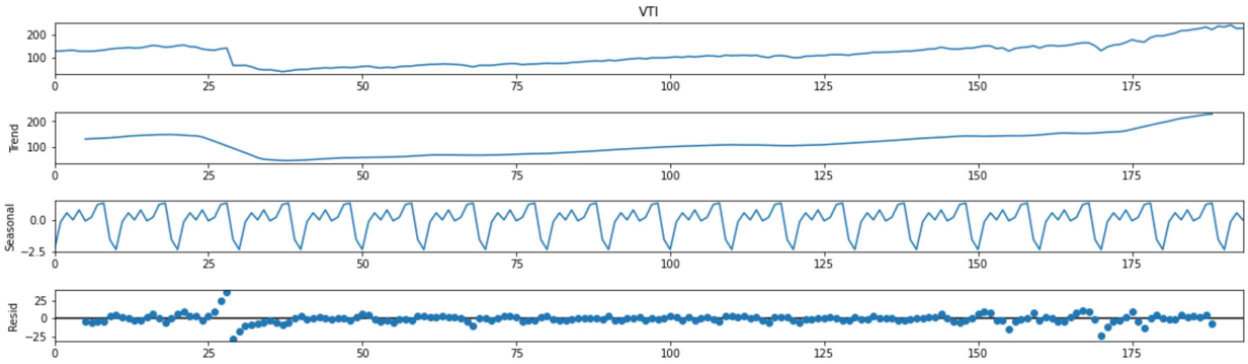


Figure 5. Seasonal decomposition visualizations for exploring the dataset. Note the distinct seasonality in the third visualization. This pattern is welcome during data exploration as it hints at seasonality playing a role in the data, which is a key component of SARIMAX.

SARIMAX takes in seven parameters which are grouped into two: order and seasonal order. The first order is (p, d, q), and is the same as the ARIMAX model mentioned above. The seasonal order is displayed as (P, D, Q, m) and the parameters represent Seasonal AR

Specification, Seasonal Integration Order, Seasonal MA, and Seasonal Periodicity respectively (Verma, 2021).

Similar to ARIMAX, we ran autocorrelation and partial autocorrelation functions to determine the *p* and *q* values, and the first significant changes for both were one, as shown in the below figures representing the ACF and PACF functions. The data alone was not stationary, and had to be differenced twice for this model, making the *d* value two. The last parameter in the seasonal order is twelve to portray that our data is monthly. In this instance we tested *P, D,* and *Q* variables to fine tune the model until we could decrease the mean squared error as much as possible.

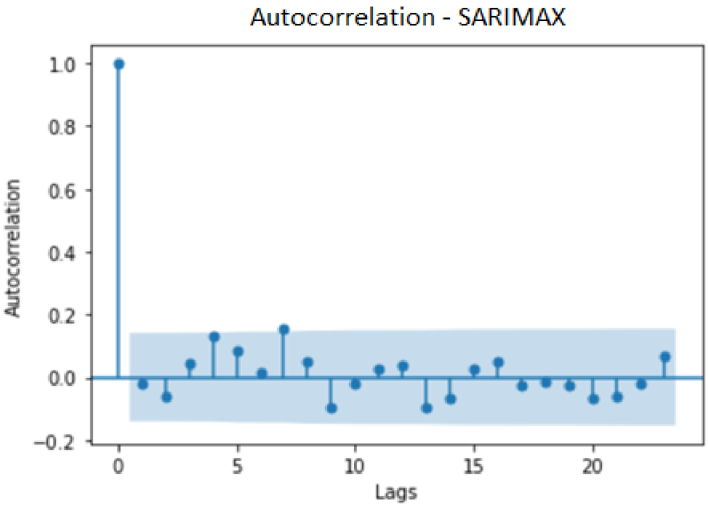


Figure 6. Autocorrelation graph, lag values that are outside of the blue shaded area are significant and are used for SARIMAX variables.

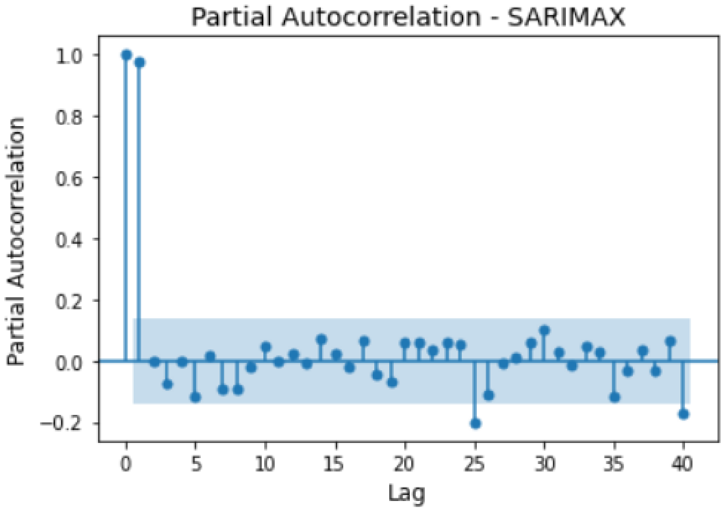


Figure 7. Partial autocorrelation graph, lag values that are outside of the blue shaded area are significant and are used for SARIMAX variables.

The SARIMAX model generally followed actual stock prices, but after the sharp decrease around 2008, it began to overcompensate and veer far from the actual prices. This difference is seen in the groups of years 2009-2012 and 2016-2020 in the figure below, which are either above or below the actual price of VTI by around 50 points each time the model overestimates the price. As the model learned and corrected itself through the training data, it eventually began to predict numbers that more closely followed the actual stock price.

SARIMAX Model vs VTI Price

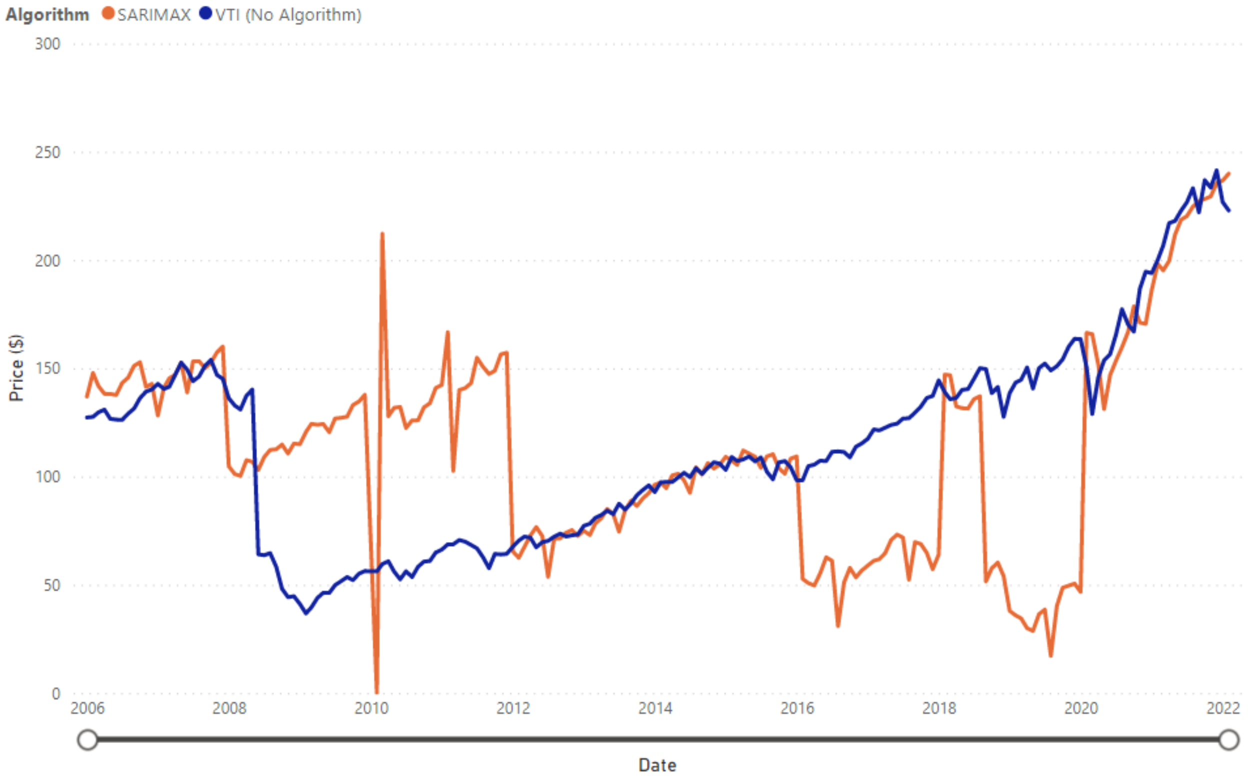


Figure 8. Line graph visualization with two lines representing the SARIMAX values (orange) and the actual VTI price values (blue).

LSTM

Long-Short Term Memory (LSTM) is an artificial recurrent neural network (RNN) architecture which has the capability of predicting values in data structures as complex as audio and video as well as simple data structures such as stock prices or room temperatures. What is interesting about this architecture is that it is well-suited for predicting future values based on timeseries data since it contains feedback connections allowing the model to reinforce itself during training. (Wikimedia Foundation, 2022)

Before training our model, we must note that the model highly benefits from the data to be normalized to values between 0 and 1 since it utilizes geometric functions such as hyperbolic tangent and sigmoid, which cause large values to yield slightly different results when determining the impact on a given layers node weight and increases the overall computation time when training the model. To achieve this, we use the “MinMaxScaler” from the sklearn preprocessing library which fits itself to a given data structure on a per column basis. This is important since the data structure returned after creating predictions isn’t the same shape as when the scaler was fitted so we need to append original data or repeat determined values to match the shape for inverse transforming. (AIEngineering, 2020)

When training our model, we can return data on the training per epoch or iteration of testing. This is useful when determining whether the model is overfitting itself to the training data or improving over iterations. Ideally, we want to see a validation loss lower than the training loss but when working with complex multivariate data structures, this can likely happen.

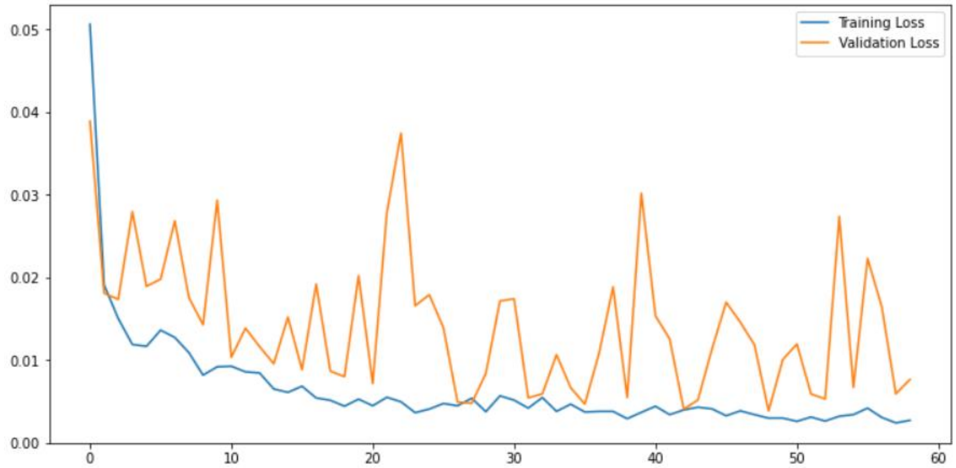


Figure 9. LSTM Training Loss compared to the Validation loss, decreasing trend suggests model is not overfitting

The LSTM model has the capability of producing an accurate trend when trained on multivariate data with few data spikes occurring which quickly correct themselves. This is due to the model's ability to understand abnormal data and compensate for it during training. In the given stance that the model is only exposed to values ranging from the minimum to maximum of the training set, determining values near the outside of that range may be more difficult as seen between the group of years 2017-2020 and 2021-2022. The reason this model is useful in a stock analysis perspective is that it doesn’t overextend its predictions by a large margin on observed extremes such as in the 2008 market crash and the most recent market peak in late 2021 and provides a consistently accurate estimation of price movement utilizing multivariate data.

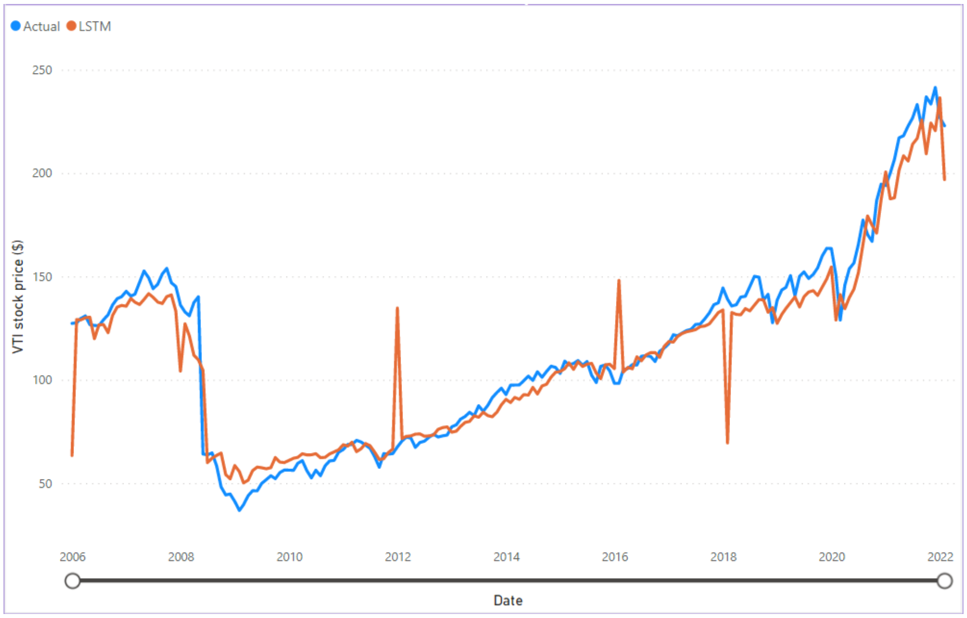


Figure 10. LSTM predicted values (orange) compared to actual values of VTI Index price (light blue)

Results and Discussion

Historical Prices and Total US Revenue by Year

Choosing VTI to predict wasn't random, there were two main levels of checking that it went through to be selected. The first check was examining VTI's composition to see if it had any holdings outside of the United States. Only 0.1% of VTI's holdings are foreign holdings (Vanguard, 2022). That passed the first check and established VTI as a majority domestic ETF. The second check to see if VTI could act as a proxy for the state of the United States' economy was to compare VTI's historical prices to U.S. Total Revenue. The following figure shows yearly VTI prices and U.S. Total Revenue. The slope of total revenue is generally positive when the price for VTI is increasing, and it is negative when VTI prices decrease.

United States Total Revenue vs Historical VTI Prices

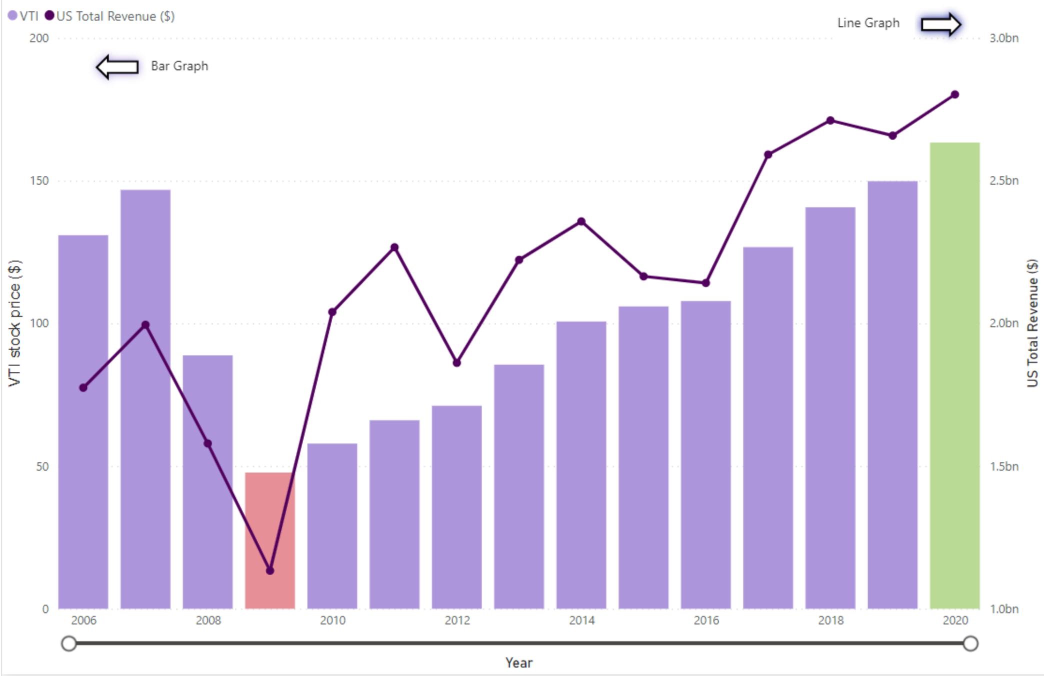


Figure 11. A combined bar (VTI stock prices) and line (U.S. total revenue) graph by year.

Machine Learning Models

Linear Regression

Looking at the Q-Q plot in Figure 10, it is immediately apparent that the linear regression and Lasso CV models are not effective models to predict VTI stock prices. The trend of the data is non-linear, resulting in the poorest performances in terms of accuracy when compared to the root squared mean error values of the ARIMAX, SARIMA and LSTM models. The residual plots in Figures 13 and 14 seem to result in a structured pattern, which suggests these models aren't accurately predicting the test sets. This likely arises from the time-dependence of the stock prices and economic features. With a test size of 25%, the linear regression model and Lasso CV model overpredict the steep drop in stock price that occurs at the beginning of 2020, significantly affecting predictions thereafter.

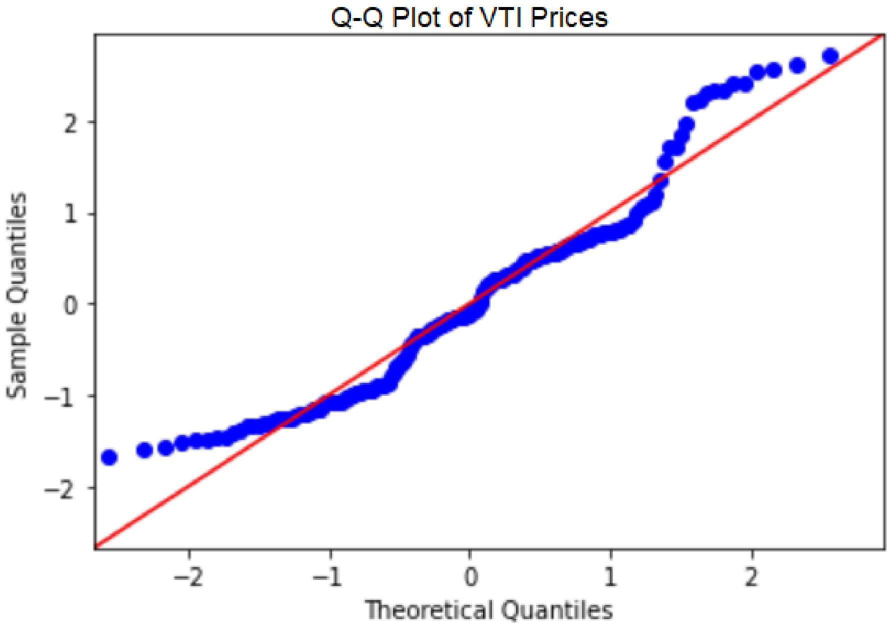


Figure 12. Q-Q plot of VTI Stock Prices. The trend formed from the sample quantiles and the theoretical quantiles represent a non-linear relationship.

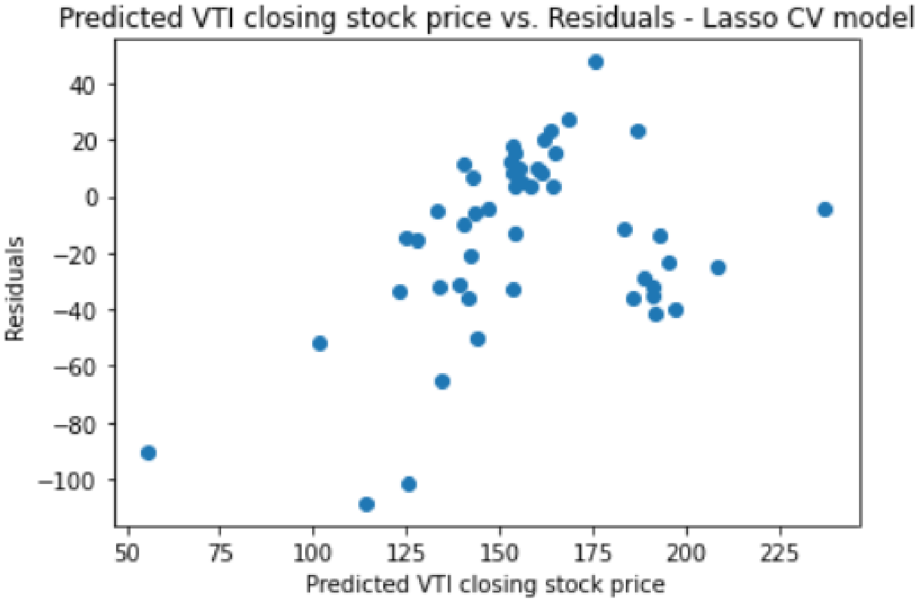


Figure 13. Residual Plot of the LassoCV model predictions.

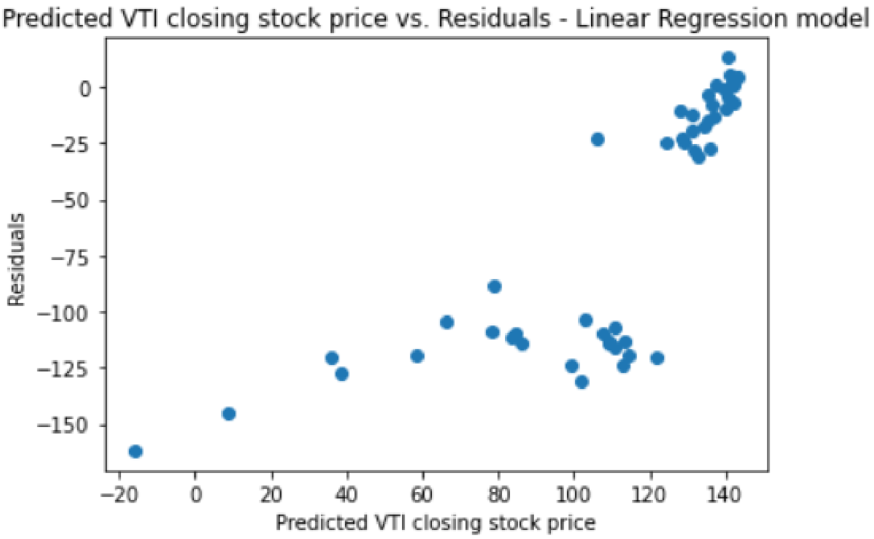


Figure 14. Residuals Plot of the Linear Regression model predictions. Both the Lasso CV and Linear Regression model's residual plots have a structured, linear pattern, which arises from the features and target variable being time dependent.

ARIMAX

The ARIMAX model performed well on the test dataset and followed the general upward trend but did not account for the smaller peaks and valleys.

SARIMAX

After the process of fitting and training the SARIMAX model, the predictions made close predictions to the actual ETF price. It kept a conservative upward slope with slight dips in a seasonal pattern. One opportunity for further exploration with this model is using it to predict values that fall drastically, like the drop in price noted around 2008.

LSTM

The LSTM predicted VTI prices well and followed the general upward trend. However, all predictions were less than the actual price and there seemed to be a downward trend toward the end of the data that was not matched by the actual price.

Overall

Overall, the ARIMAX, SARIMAX, and LSTM models made accurate predictions. The SARIMAX was the most accurate (MSE = 65.72), followed by ARIMAX (MSE = 87.9) and LSTM (MSE = 287.11) (Figure 15). Linear regression and Lasso CV regression were the least accurate (MSE = 4157.01, MSE = 5156.71), but this is explained by the fact that time series data are not good candidates for these algorithms (Figure 15). One reason why SARIMAX may have had the most accurate predictions is because VTI prices are seasonal, and this is the only model that incorporates seasonality.

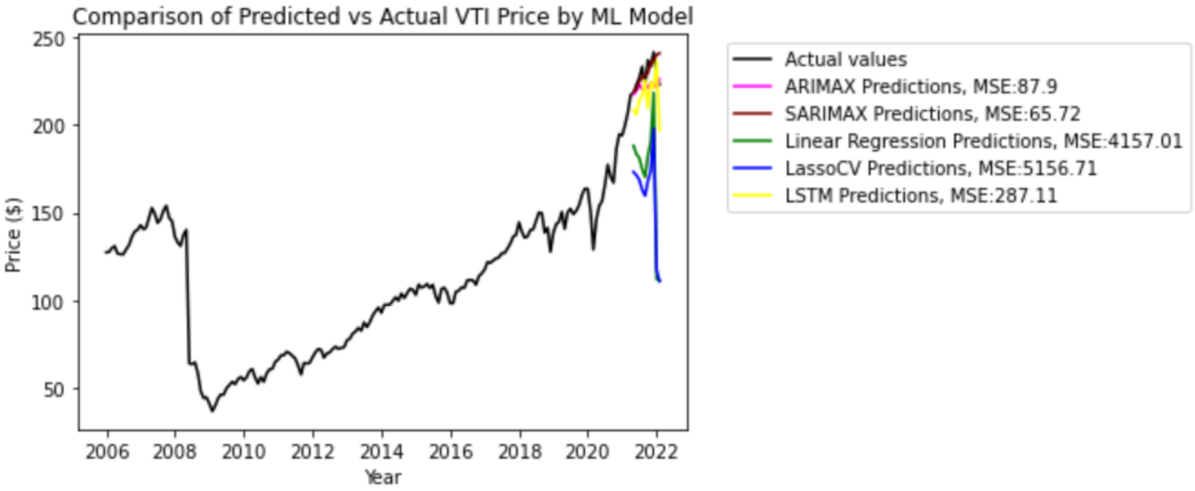


Figure 15. A comparison of several machine learning models' predictions to the actual VTI stock prices and their mean squared error values (MSE).

Risk and Expected Returns on Investment by Sector

To quantify the risk and expected returns, the percentage change in the stock price for each month was calculated. The standard deviation of these values represents the risk of each ETF stock, while the mean of these values represents the expected returns for each ETF stock. Figure 16 shows the risk and expected returns for the top 6 Vanguard ETF stocks. Considering all stocks standard deviations fall below a tenth of a percentage, and their expected returns fall within the range of 1/1000th of a percent and 3/200ths of a percent of your initial investment, these would be considered low risk, low reward investments, intended for long-term investment plans. VGT would be the index fund (out of these 6) that has the highest expected return on investment, while VHT would be the safest index fund to invest in.

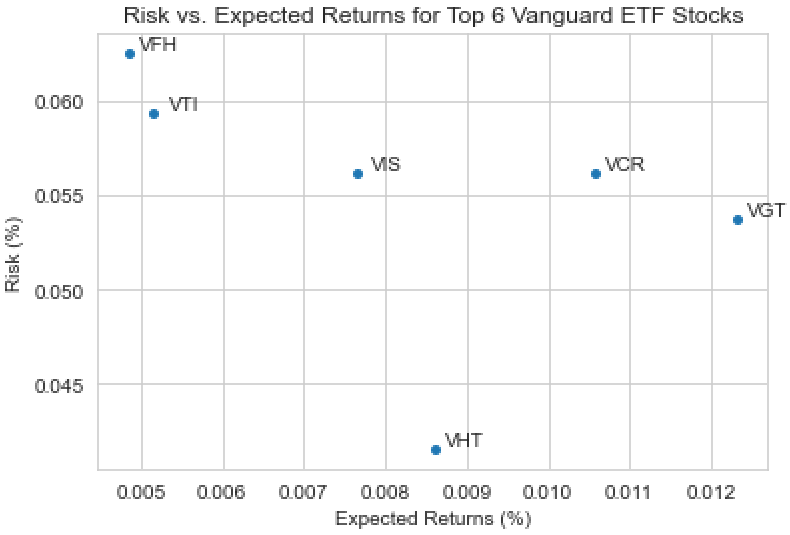


Figure 16. Risk vs. Expected Returns for the top 6 Vanguard ETF stocks.

Conclusion

This project examined the domestic stock market and its top five sectors through Vanguard index ETF proxies. It found that the SARIMAX algorithm yields the most accurate machine learning model to predict VTI prices, followed by ARIMAX and LSTM, and that Linear Regression and LASSO Regression are inappropriate algorithms for datasets containing time dependent features. The results also indicate that the features “Advanced Retail Sales: Retail Trade”, “Unemployment Rate” and “10-Year Treasury Constant Maturity Rate” are the most important features when predicting VTI prices, but that feature importance is not consistent across sectors. Additionally, the healthcare sector has the lowest investment risk and the technology and information sector has the highest investment return. Lastly, historical VTI prices trend similarly with the national total revenue. This suggests VTI can be used as a reliable indicator of the national economy.

The findings of this project contribute to the greater understanding of the domestic stock market by providing insight into effective modeling algorithms and features, sector trends, and context for how the stock market relates to the national economy. Businesses, investors, and legislators all benefit from this knowledge and can use it to better inform their financial decisions.

References

AIEngineering (2020, October 11) *End to End Multivariate Time Series Modeling using LSTM.* Youtube.com https://youtu.be/4FmVIpcwl4k

Alpha Vantage. (2022). Alpha Vantage API Documentation. Alpha Vantage. https://www.alphavantage.co/documentation/.

Brownlee, J. (2020, August 28). *How to Grid Search SARIMA Hyperparameters for Time Series Forecasting*. Machine Learning Mastery. Retrieved February 13, 2022, from https://machinelearningmastery.com/how-to-grid-search-sarima-model-hyperparameters-for-time-series-forecasting-in-python/

How much is the stock market worth? GorillaTrades. (2021, May 19). Retrieved February 15, 2022, from https://www.gorillatrades.com/how-much-is-the-stock-market-worth/

KumarI, A. (2020, October 8). *Lasso regression explained with python example*. Data Analytics. Retrieved February 15, 2022, from https://vitalflux.com/lasso-ridge-regression-explained-with-python-example/

The Pennsylvania State University. (2022). *2.1 moving average models (MA models): Stat 510*. PennState: Statistics Online Courses. Retrieved February 16, 2022, from https://online.stat.psu.edu/stat510/lesson/2/2.1

United States Census Bureau. (2021a). *2006 Annual Survey of State Government Finances Tables* [Data set]. Retrieved January 31, 2022, from 2006 Annual Survey of State Government Finances Tables (census.gov).

United States Census Bureau. (2021b). *2007 Annual Survey of State Government Finances Tables* [Data set]. Retrieved January 31, 2022, from 2007 Annual Survey of State Government Finances Tables (census.gov).

United States Census Bureau. (2021c). *2008 Annual Survey of State Government Finances Tables* [Data set]. Retrieved January 31, 2022, from 2008 Annual Survey of State Government Finances Tables (census.gov).

United States Census Bureau. (2021d). *2009 Annual Survey of State Government Finances Tables* [Data set]. Retrieved January 31, 2022, from 2009 Annual Survey of State Government Finances Tables (census.gov).

United States Census Bureau. (2021e). *2010 Annual Survey of State Government Finances Tables* [Data set]. Retrieved January 31, 2022, from 2010 Annual Survey of State Government Finances Tables (census.gov).

United States Census Bureau. (2021f). *2011 Annual Survey of State Government Finances Tables* [Data set]. Retrieved January 31, 2022, from 2011 Annual Survey of State Government Finances Tables (census.gov).

United States Census Bureau. (2021g). *2012 Annual Survey of State Government Finances Tables* [Data set]. Retrieved January 31, 2022, from 2012 Annual Survey of State Government Finances (census.gov).

United States Census Bureau. (2021h). *2013 Annual Survey of State Government Finances Tables* [Data set]. Retrieved January 31, 2022, from 2013 Annual Survey of State Government Finances Tables (census.gov).

United States Census Bureau. (2021i). *2014 Annual Survey of State Government Finances Tables* [Data set]. Retrieved January 31, 2022, from 2014 Annual Survey of State Government Finances Tables (census.gov).

United States Census Bureau. (2021j). *2015 Annual Survey of State Government Finances Tables* [Data set]. Retrieved January 31, 2022, from 2015 Annual Survey of State Government Finances Tables (census.gov).

United States Census Bureau. (2021k). *2016 Annual Survey of State Government Finances Tables* [Data set]. Retrieved January 31, 2022, from 2016 Annual Survey of State Government Finances Tables (census.gov).

United States Census Bureau. (2022a). *2017 Annual Survey of State Government Finances Tables* [Data set]. Retrieved January 31, 2022, from 2017 Annual Survey of State Government Finances Tables (census.gov).

United States Census Bureau. (2022b). *2018 Annual Survey of State Government Finances Tables* [Data set]. Retrieved January 31, 2022, from 2018 Annual Survey of State Government Finances Tables (census.gov).

United States Census Bureau. (2022c). *2019 Annual Survey of State Government Finances Tables* [Data set]. Retrieved January 31, 2022, from 2019 Annual Survey of State Government Finances Tables (census.gov).

United States Census Bureau. (2022d). *2020 Annual Survey of State Government Finances Tables* [Data set]. Retrieved January 31, 2022, from 2020 Annual Survey of State Government Finances Tables (census.gov).

Vanguard. (2022). *Vanguard Total Stock Market ETF (VTI): Portfolio & Management*. Retrieved February 16, 2022, from https://investor.vanguard.com/etf/profile/portfolio/vti

Verma, Y. (2021, July 30). Complete Guide To SARIMAX in Python for Time Series Modeling. Analytics India Mag. https://analyticsindiamag.com/complete-guide-to-sarimax-in-python-for-time-series-modeling/#:~:text=SARIMAX-,SARIMAX(Seasonal%20Auto%2DRegressive%20Integrated%20Moving%20Average%20with%20eXogenous%20factors,average%20component%20in%20the%20model

Wikimedia Foundation. (2022, January 30). *Long short-term memory*. Wikipedia. Retrieved February 16, 2022, from https://en.wikipedia.org/wiki/Long_short-term_memory