ML Final Report

Introduction/Background:

The stock market produces abundant and ever-changing data, making it a challenge to effectively and accurately analyze market trends. Within these trends, often exist market anomalies which are inconsistent time series patterns in security returns that contradict the efficient market hypothesis (EMH) [1]. This hypothesis states that outperforming the financial market using historical data is highly unlikely because the information about prices speaks to everything that is already observed. However, anomalies displaying the existence of deviations from. Identifying these patterns is incredibly valuable to investors and institutions as it aids them in generating higher yields and market returns, by avoiding major financial risks. Anomaly detection has become an important data analysis tool in the financial domain, and the recent adoption of machine learning in this field allows for more accurate and efficient analysis of market data. [3] Machine learning algorithms can be trained to recognize abnormal data from a given dataset by learning to recognize patterns in normal data [2]. These practices are used to detect widespread anomalies through the market, whether it's recognizing patterns in historical data to predict future market behaviors, to identifying data points deviating from the norm to determine unusual market events.

This project runs such machine learning algorithms on the S&P 500 dataset (over a 10 year period) in order to identify common market anomalies such as price reversals, small-cap, overselling, underselling, etc. Through repeated identification, the algorithms learn abnormal data trends and are able to predict future anomalies. This dataset is temporal and contains five features and 2736 records.

Problem Definition:

Anomalies in the stock market can often cost investors and financial institutions significant and unexpected financial losses. In order to prevent this, our project aims to evaluate market anomalies and identify them ahead of time to protect traders, investors and institutions from financial crises. We will examine three existing detection algorithms (Long Short-Term Memory, Support Vector Machine & Principal Component Analysis) to identify anomaly-indicating stock market trends and enhance them in order to produce models that are capable of making highly accurate anomaly predictions to minimize losses.

Data Collection:

We collected our data from the S&P 500 Dataset that covers 10 years and has features: open, high, low, close, adjusted close and volume. The dataset we’re working with has changed since the proposal due to the need for more features specific to the criteria with which we will be identifying anomalies. Adjusted close and volume are the two new added features that are key to

the building of our models. LSTM is currently using closing prices to detect anomalies (more details on this can be found in the Methods section), while volume will be used during the implementation of the SVM model. To perform anomaly detection with the original four features would be a challenge as trends in open, high, low and close are not accurate predictors of possible anomalies. Here is the new and currently in use dataset:

CS 4641: S&P 500 Dataset 10Y (Group 20)

Visualization of the Dataset:

Date	Open	High	Low	Close	Adj Close	Volume
02/01/2013	145.110001	146.149994	144.729996	146.059998	119.976509	192059000
03/01/2013	145.990005	146.369995	145.339996	145.729996	119.705376	144761800
04/01/2013	145.970001	146.610001	145.669998	146.369995	120.231102	116817700
07/01/2013	145.850006	146.110001	145.429993	145.970001	119.902542	110002500
08/01/2013	145.710007	145.910004	144.979996	145.550003	119.557571	121265100

Methods & Analysis:

We will employ One Class Support Vector Machine (SVM), Long Short-Term Memory (LSTM) and PCA (Principal Component Analysis) techniques to enhance our data analysis, enabling us to identify anomalies and subsequently make predictions for future occurrences. These algorithms are particularly well-suited for handling time series data and are commonly used in the stock market for price prediction and anomaly detection, making them ideal for analyzing the S&P 500 dataset.

LSTM, a type of RNN, uses its ability to capture temporal dependencies and reconstruction errors measured by functions such as MSE to model the dynamic nature of the market. In addition to these capabilities, an established anomaly threshold score helps LSTM efficiently detect deviations. In our implementation of this model, we created the training set for LSTM by normalizing the closing prices and after training, the model's predictions were inverse-transformed to de-normalize the closing prices. We used several error metrics, including Root Mean Squared Error (RMSE), upon which the anomaly threshold was set, Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE) to evaluate the model's performance. When training the model, we defined an anomaly as an instance in a time step where the absolute difference between actual and predicted closing prices is greater than the threshold (1.5 times the RMSE). As the model picks a time step with the largest differences, 1.5 was an appropriate multiplier as it is sensitive to anomalies in the dataset.

One Class SVM, a variation of SVM utilizes the concept of hyperplanes to distinguish anomalies from normal patterns in the market. The hyperplane in this algorithm is a decision boundary that accounts for normal instances of data in the dataset and assigns decision scores to the new instances based on their distance from the boundary. The data points that deviate the

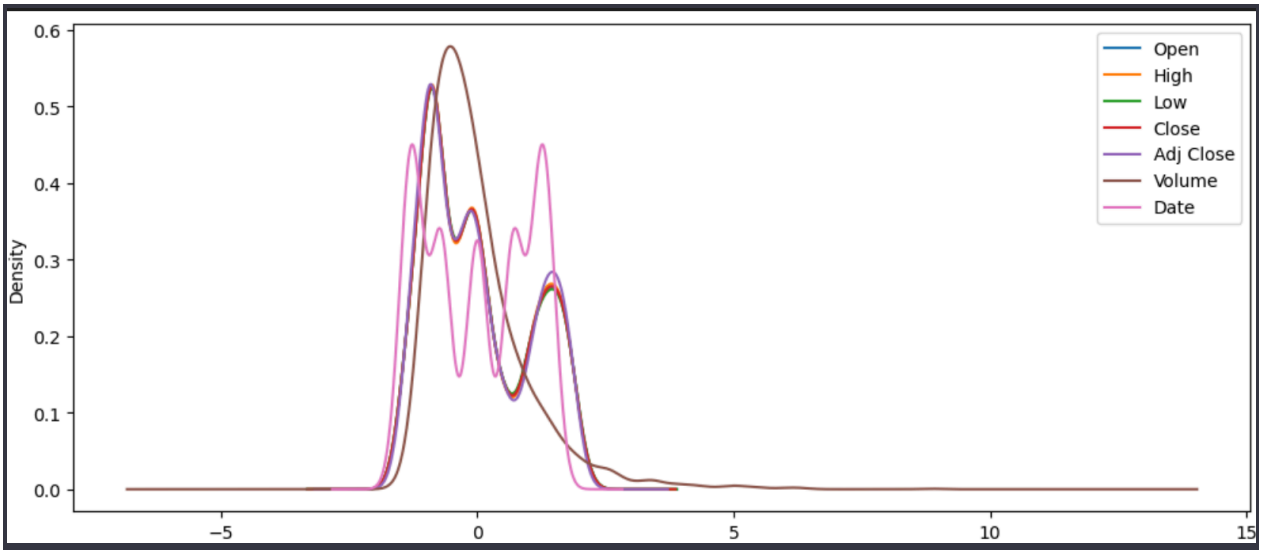
most from the boundary will have higher scores, indicating an anomaly in the data. This method of evaluation is used in all three models of SVM we have implemented, where each model differs in the magnitude of the hyperparameters used. Model 1 uses a gamma = 0.1 which will result in a board decision boundary and therefore a larger range of data points will be considered when the support machine is being created. It also uses a nu = 0.2 that allows for a higher fraction of training errors and a larger amount of support vectors which will result in a model that will tolerate a larger range of deviations as well. Model 2 uses a gamma = 1 which shortens the boundary compared to Model 1 and the focus for individual neighboring data points increases. It uses nu = 0.2 carrying the same effects as mentioned for Model 1. Model 3 uses a gamma = 10 which makes SVM highly sensitive to these individual data points and the boundary becomes a lot more narrow in what it accepts. It uses nu = 0.1 which implements more restrictive constraints on training errors and anomalies are more strictly defined. Overall, we see the influence of gamma in changes of boundary size and nu in changes of error tolerance and sensitivity to deviations. These hyperparameters influence the decision score whose threshold is set to 0 which gives rise to binary anomaly predictions.

We also used PCA to see how our data would be projected in a different type of algorithm. However, we realized very soon that PCA works best with tabular, static data where each observation is independent of others. Time series data has dependencies and a natural order that PCA doesn't expect. Even when using the sin transformer to preprocess and give the "Dates" a numerical value that shows periodic growth and dips, the PCA model can't group the data to be able to detect anomalies.

```
# Load the data from the csv file into a Pandas Dataframe
original_data = pd.read_csv('S&P_final.csv')
original_data.head()
X = original_data.drop(columns=['Date'])
X['Date'] = range(1, len(X) + 1)
X['Date'] = 2 * np.sin(2*pi*X['Date']/12)
# Standardize the data
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

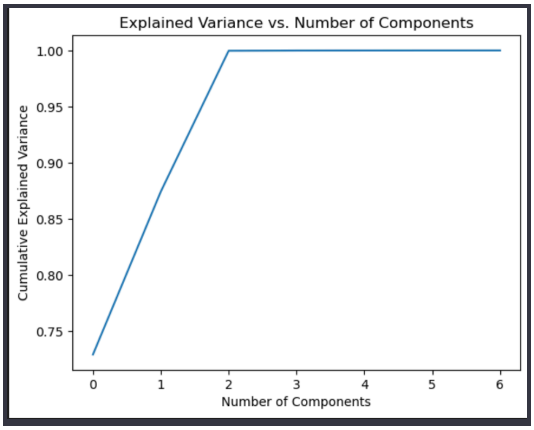
This is loading and preprocessing data by dropping "Date" as a string and putting in a number instead and scaling it with sin transformer (line 6). Then we preprocess with a standard scaler to standardize the values.

This is how the data looks at first: (Date is in monthly periodic format)




```
num_components = 7
# Create the PCA model
pca = PCA(n_components=num_components)
# Fit the model with the standardised data
pca.fit(X_scaled)
PCA(copy=True, iterated_power='auto', n_components=7, random_state=None, svd_solver='auto', tol=0.0, whiten=False)
transformed_data = pca.transform(X_scaled)
```

Next we fit the model with PCA from sklearn library and use variance to see which features have the most influence and then reduce the number of features.



This shows us at what number of components you achieve an optimal solution. And with our S&P data it has 3 features.

Leveraging the capabilities of LSTM, SVM and PCA for sequence prediction and classification, we were able to detect potential outliers that do not fall under a classified group [4]. We have refined and updated these algorithms to enhance their anomaly detection capabilities and assess

their accuracy using a dedicated testing dataset. We utilized sklearn and PyTorch to implement the SVM, LSTM and PCA algorithms.

Results and Discussion

Data Preprocessing

The data preprocessing techniques we used were sin transformer and forward selection.

sin_transformer: This type of transformation is often used when dealing with periodic data, such as dates like in our dataset. The sine transformation can help capture periodic patterns in the data by representing them as oscillations in the sine wave. It does math to convert dates into numbers in sequential order. We needed this on our data because it is in the form of dates so we needed to convert it.

```
def sin_transformer(period):
    return FunctionTransformer(lambda x: np.sin(x / period * 2 * np.pi))
```

forward_selection: In forward selection, we start with a null model, start fitting the model with one individual feature at a time, and select the feature with the minimum p-value. We continue to do this until we have a set of features where one feature's p-value is less than the confidence level. We chose this because the stepwise nature of forward selection allowed us to stop the process when a certain criterion is met (e.g., when model performance plateaus or starts to degrade). This flexibility is beneficial for avoiding unnecessary complexity.

```
##############################
### DO NOT CHANGE THIS CELL ###
##############################
bc_dataset = load_S&P_500()
bc = pd.DataFrame(bc_dataset.data, columns=bc_dataset.feature_names)
bc["Anomaly"] = bc_dataset.target
# print(bc)
X = bc.drop("Anomaly", axis=1)
y = bc["Anomaly"]
featureselection = FeatureReduction()
# Run the functions to make sure two lists are generated, one for each method
print(
    "Features selected by forward selection:", FeatureReduction.forward_selection(X, y)
)
print(
    "Features selected by backward elimination:",
    FeatureReduction.backward_elimination(X, y),
)
```

Metrics:

LSTM:

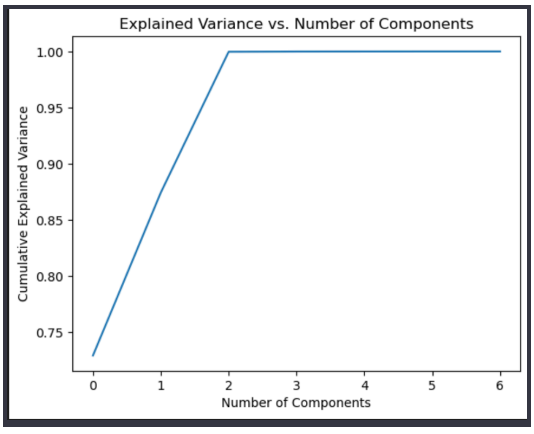
The metrics used to test our models are root mean square error, mean absolute error, and mean absolute percentage error. The root mean squared error indicates a relatively good performance from the model. The range of closing prices, which is our target variable in our dataset, ranges from 145.55 to 477.71, and a rmse of 5.417, is a relatively small percentage of the range of target variable. This means the predictions were close to the actual values. Similarly, the mean absolute error of 3.65 shows the model is a very good predictor of anomalies in the data, given how large the range of the closing prices are. A mean absolute percentage error of 0.012 is similarly a good score, which indicates that the values from the model's predictions deviated from the actual values by only around 1.2%.

```
Root Mean Squared Error:  5.417069159796916
Mean absolute error:  3.6506982867525686
Mean absolute percentage error:  0.012427814117573661
```

PCA:

We used pca.explained_variance_ratio_ to find the optimum number of features to use. The figure below helps visualize this:

```
# Step 2: Access explained variance ratio
explained_variance_ratio = pca.explained_variance_ratio_
print("Explained Variance Ratio:")
print(explained_variance_ratio)
```



It is evident here that after about ~2.3 features, adding more doesn't add to the variance of the data, and would therefore be a waste.

SVM:

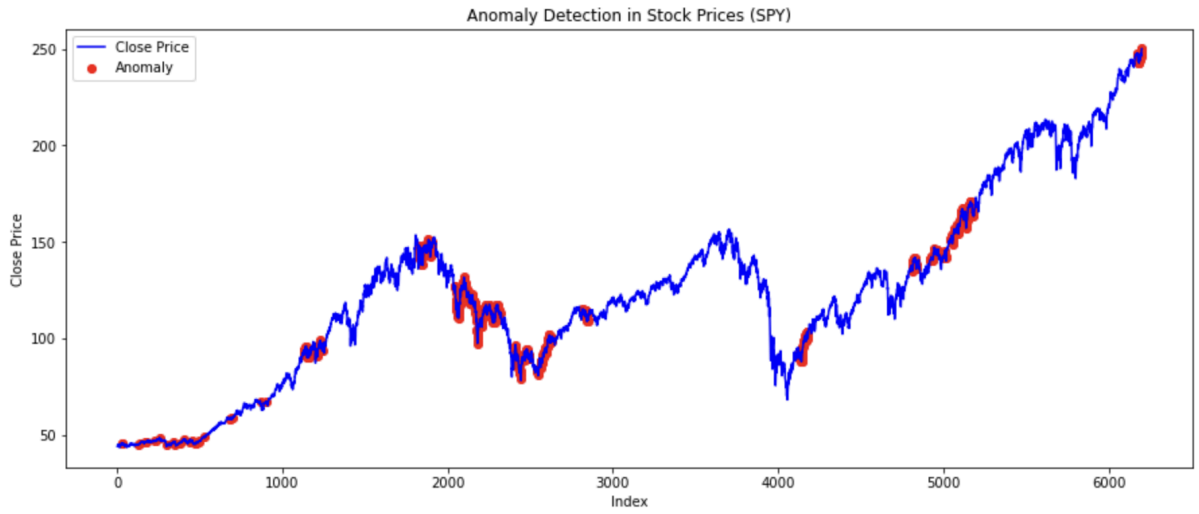
Because of the lack of true labels in our data, the accuracy of the SVM model was calculated by dividing the total number of normal instances (values equal to 1 in the predicted model) divided by the total number of instances. The labeled counts show the distribution of the number of instances that were predicted as an outlier (value of -1), compared to normal instances (value of 1).

```
Label Counts: {-1: 881, 1: 5321}
Accuracy: 0.8579490486939697
```

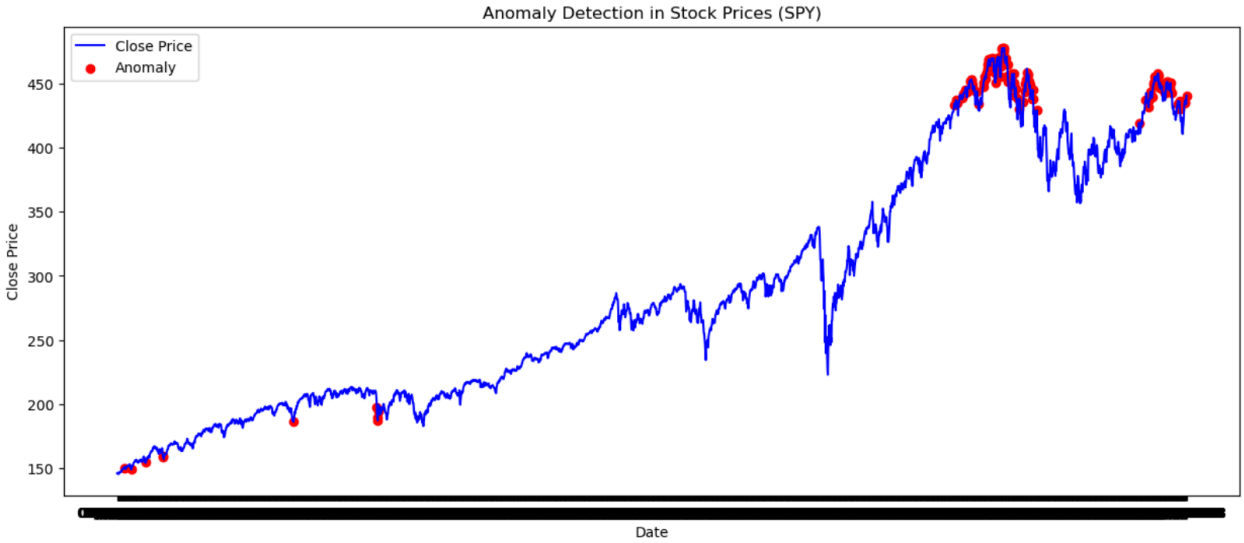
Visualizations:

This is a visualization of the time series LSTM analysis. Because of LSTM's ability to learn observation over a long term, time series forecasting is a useful visualization to analyze this algorithm. This chart maps out the closing prices of the S&P every day over the past 10 years. Closing prices deemed to be anomalous are mapped out in red, and determined based on a threshold. Data points with longer time steps are deemed outliers, and are the points that are greater than the threshold, calculated as the root mean squared error * 1.5, where 1.5 is the sensitivity to anomalies.

This is a visualization of the One Class Support Vector Machine model. Since it is very difficult to obtain labeled data for anomaly detection, One Class SVM is beneficial in that it does not require labeled data to determine normal and anomalous data. This algorithm characterizes normal behavior of the data and identifies anomalies, and is useful for anomaly detection by using the kernel trick to capture trends that are non-linear. Because it focuses on finding regions of normal behavior, regions that show a general increase or decrease could be entirely marked as anomalies, which is the case in some regions marked in red in the visual below.



This is a visualization of the Principal Component Analysis model. PCA works best on static data with dimension reductions and pattern recognitions. It captures patterns on non-linear data over a close price vs index comparison running on the dataset. Though it is capable of anomaly detection, it only does so for subgroups of the dataset. Unlike the other models, it is not spread out and doesn't detect spikes or drops when the data around it seems stable. This visualization will be further analyzed in the next section.



Analysis of Visualization Results:

LSTM: In the visualization, it can be seen that the red highlighted region representing the anomalies is consistent right before the close price dips drastically. This drastic decrease is in fact a huge fluctuation that affects the market and is defined as an anomaly. With the sensitivity of the threshold being high, the LSTM accurately predicts an upcoming anomaly in the market. This predictive nature can be observed in other areas in the visualization where we see anomaly markers highlighted before drastic spikes or drops in the closing prices.

SVM: In the visualization, red highlighted regions indicate anomalies, and many of these regions that are highlighted demonstrate sharper increases or decreases in their closing price. While many of these regions are able to predict anomalies that occur, many significantly isolated outliers are missed by this model, because of the nature of this model in capturing regions of normal instances, sharp and drastic outliers are missed.

PCA: In the visualization, it was noted that the anomaly markers are in chunks of subgroups of the dataset and not spread out. Unlike our first assumption PCA only best works on standalone or static data. Due to this, we see that it is only able to detect multiple anomalies in the same few regions. If the pattern is stable before a spike or drop, PCA is unable to detect an anomaly like LSTM due to its sensitivity. Therefore, PCA is not suitable in detecting anomalies with subtle trends but can be efficient at detection when data is erratic like the group of markers at the highest peak.

Strengths, Limitations, Trade Offs:

LSTM displayed its strengths with the ability to capture temporal dependencies and the use of an accurate threshold (1.5) that leads to very precise anomaly prediction as explained above. Though the LSTM performed with high accuracy, it comes with limitations of high complexity

during training, sensitivity to hyperparameter tuning that can lead to inaccurate detection, and handling extreme anomalies when sensitivity must be controlled. The sensitivity issue was overcome by testing the correct threshold level using trial and error and comparisons. Due to some of the limitations that are difficult to overcome, a tradeoff would be being able to capture complex temporal patterns and the sensitivity of LSTM to noisy data.

One Class SVM was able to effectively identify anomalies using boundaries and hyperparameters that added more flexibility to the sensitivity and ranges. Decision sources used with boundary comparison is a key strength of SVM as it allows to accurately quantify the deviation from normal patterns, rather than giving estimates or larger ranges with uncertainty. Despite the strengths, the model has limited ability to capture complex temporal patterns like LSTM, it can overlook anomalies that are on the lower extreme (hard to identify) and be sensitive to parameters depending on the decision boundary that is defined. Though the flexibility of hyperparameter usage is beneficial it comes with the tradeoff between error tolerance and sensitivity.

PCA has strengths in using distance metrics to identify anomalies and efficient pattern handling, however from our results we have concluded that it is overcome by the weaknesses. It is limited in their abilities due to not being able to handle time series data as effectively as we first assumed. This is due to the dependencies it comes with and the need to make independent observations which challenges consistent anomaly detection. Though we believe it would perform well for stand alone cases where investors want to analyze a controlled static set of stock market data, it will still have a tradeoff accurate detection of data with temporal dependencies.

Comparative Analysis (Metrics):

While comparing the performance and result of the models, new discoveries of strengths and limitations of the algorithm were recorded under various criteria. LSTM and SVM were highly applicable due to their parameters of boundaries and dependencies. However, like mentioned above, PCA was found to be comparatively less effective when analyzing time series data. As seen in the Analysis Of Results (Visualizations) section, LSTM was the model with the most accurate anomaly detection as the markers were present ahead of peaks and drops were more frequent. Metrics also prove this analysis as LSTM was calculated to have a 1.2% margin of error which is relatively low when considering factors of subtle anomalies and false positives that LSTM limits on. SVM was calculated to have an accuracy of 85% which is relatively high but can still be considered an optimal anomaly detection algorithm. PCA returned an explained variance ratio of 3 which is unusual to the standard of 0.8 to avoid overfitting. This once again confirms the claim that PCA isn't best suited for anomaly detection. By comparing these metrics, we can claim that LSTM is the best in performance for detection and SVM follows it, while PCA cannot be accounted for due to reasons stated above.

Conclusions:

Through initial research on the popularity, need and impact of anomaly detection in the stock market, we narrowed down three models that would best address the problem: LSTM, One Class SVM and PCA. These models were run on a 10 year times series data set extracted from the S&P 500 Index and the results were that LSTM performed the best in accuracy out of the three, One Class SVM was second best and PCA faced challenges with time series data analysis. This conclusion was made from the extensive comparative analysis of visualizations, metrics and strengths, limits and tradeoffs above. The results for LSTM and SVM were satisfactory as the anomaly detection had a low margin of error and the visualizations indicated accurate predictions before spikes or dips occurred. To further improve these results and lower the margin of error, closer fine tuning of hyperparameters (gamma, nu, threshold) and more concise and accurate decision boundaries can be implemented. The PCA results were not satisfactory due to the limitations of its independent observation behavior. It is difficult to improve this result, as it was proved in the metrics and visualizations that it will inherently perform as observed. Due to this, it is best to use a different model than PCA for time series anomaly detection such as k-NN.

Overall we can attribute each model's performance due to its characteristics, hyperparameters, dependencies and boundaries used. These performances (LSTM and SVM) make a positive impact in the field of anomaly detection due to the accurate predictive natures observed. The next steps to be considered are to further train and test the models to decrease error and to consider a new model better aimed for time series data in replacement of PCA. A great replacement model we discovered is addressed below.

Next Steps:

Overall, there were many limitations for each of our algorithms used in our project. In order to improve upon our project, we would look to implement models that can be trained on a wider range of features such as real-time market news or economic indicators that would provide a more accurate detection of anomalies in the stock market. One instance that we would explore moving forward is the implementation of the Black-Scholes model. This model takes into account several different variables such as volatility, the time to expiration, current stock price, risk-free rate, and strike price. This model is generally used to determine fair prices of stock options, and it can certainly be used to determine an abnormal change in the price of a stock.