Project Report

DATA1030 – Fall 2020

Chris Rohlicek *github.com/crohlicek/data1030_financial_forecasting* December 2, 2020

1 Introduction

My project analyzes a different approaches to solving the regression problem of predicting the daily change in the adjusted close price of the S&P 500 index. Our dataset is a time series comprising features for historical S&P 500 values[4], date, weather patterns [2], consumer confidence [3], and market volatility [1].

Our dataset was processed beyond initial construction and preprocessing to be suited for use in time series regression by including features for information from the previous 14 days, as well as any information that would be available at the time of the market's opening that day. We will describe this process more in the EDA section.

In this project we consider random forest regressors, support vector regressors, and XGBoost regressors. In order to tailor our data to these models, the target variable is the change in adjusted close price from one day to the next, otherwise known as the *delta*. Our analysis is carried out on these delta values, but we have also stored figures and data of the reconstructed adjusted close price predictions.

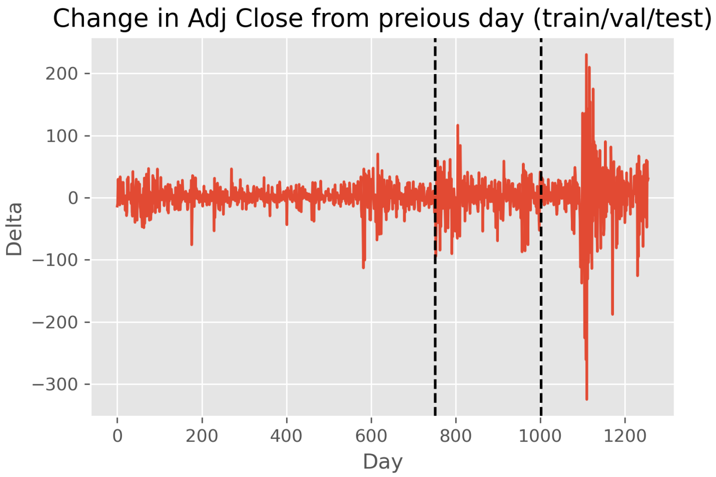

Figure 1: The target variable z, the daily change in adjusted close price of the S&P 500 index

2 EDA

The main change in our data that deviates from our midterm project proposal is the change of our target variable from adjusted close price to the related delta values describing change in adjusted close price. This change was made because random forest regressors are unable to extrapolate beyond the range of target variable values that were given in training. Given that the S&P index generally rises over time, this posed a large problem for our data. To avoid this, we changed our target variable to be the value by which the adjusted close price changed relative to the previous day's value (Fig. 1).

Another change to our data was our windowing strategy which bases predictions on the past two weeks' data. After windowing and removing data from after market open, our data have 379 features. After dropping rows from the first two weeks of data (incomplete given the windowing strategy) there are 738 training samples, 236 validation samples, and 240 testing samples.

3 Methods

Splitting Strategy

Our splitting strategy is designed such that it will maintain the temporality of our data, ensuring that we always train our model on data that occurred before any evaluation data. Our data spans five years, so our splitting separates the first three years' data into the training set, the fourth year into validation, and the fifth into testing.

Data Processing

As we mentioned above, the windowing strategy was the primary addition made to our data processing. In order to cast our prediction problem as a supervised regression problem, we added features to our original datasets which gave data from the past 14 days to be used for the current day's prediction. Each feature in our datasets have a suffix indicating the number of days by which that column's data was delayed (for example, at a given row t, the column "Open_5" has the open price from day $t-5$, and "Open_0" has the open price from day t).

Another additional data processing step was the creation of our target variable, delta, from the values of our old one. We did this by subtracting each adjusted close price value from the following one. Of course this creates a missing value in the index 0 slot, but that day is already being removed along with the first two weeks of data that is missing from each set due to the windowing.

Finally, in the spirit of taking all information into account that would be available at the opening of the market on a given day, we include past days' delta values in the 14-day windows in the feature matrices (that is, there are features given by "z_i" which are the target variables from i days before the current day).

Models & Hyperparameters

The three types of model we applied to our problem were random forest regressors, support vector regressors, and XGBoost regressors. We evaluated all three models using root mean squared error and compared them to the regression baseline model which predicts the mean target value for all time steps. We fixed the random states in the random forest and XGBoost models to ensure the reproducability of our results (this was not necessary in the SVR because that model is deterministic).

The hyperparameters we used for our random forest regressors were 'max_depth': [1, 6, 11, 16, 21, 26, 31, 36, 41, 46], and 'max_features': [0.5,0.6,0.7,0.8,0.9,1.0]. We chose these depth values because while we had many more features, these only included data from 15 different days, so allowing the depth to reach values much greater would result in overfitting on arbitrary past values that could not logically have independent predictive power on the future. The feature proportions were given anywhere from 50% to 100% again because of the high redundancy possible in a 14-day window.

For our support vector regressor we used hyperparameters 'gamma': [1e-4, 1e-3, 1e-2, 1e-1, 1e0, 1e1, 1e2], and 'C': [0.1, 1, 10, 100]. Because these values are less concretely tied to qualities of the features, we simply chose these because of their logarithmic spacing, which allow us to easily capture a broad range of model behavior.
For our XGBoost regressor we tuned over the parameters 'max_depth': [1,3,10,20,30,100], and 'colsample_bytree': [0.5, 0.6, 0.7, 0.8, 0.9, 1.0], and the choices of these values followed from the same reasoning that went into the random forest tuning.

4 Results

The models that were chosen through hyperparameter tuning are given by the following sets of hyperparameters:

- Random forest: {'criterion': 'mae', 'max_depth': 1, 'max_features': 0.5}
- SVR: {'C': 10, 'gamma': 0.01}
- XGBoost: {'colsample_bytree': 0.6, 'learning_rate': 0.03, 'max_depth': 10, 'missing': nan, 'n_estimators': 10000, 'random_state': 42, 'subsample': 0.66}

The results found from these three models are summarized in the following table:

Model Results			
Model	Training RMSE	Validation RMSE	Testing RMSE
RF regressor	17.142	26.740	60.695
SVR	11.860	21.208	60.529
XGB	16.795	27.577	60.548
Baseline	17.326	27.555	60.529

From the similarity of the three models' performance to the baseline, we can tell that three models failed to improve on the prediction achieved by simply guessing that the mean delta value will occur on every day. While this is clear evidence that these models' failed to find meaningful patterns in the data, that alone is a highly nontrivial result, and we will take a closer look at our models' behaviors on the data to get a better sense for how the models decided to optimize themselves.

Random forest results

By looking at the true versus predicted delta values we see that not only is the random forest achieving similar RMSE to the baseline, but it is doing so as a result of mimicking its predictive behavior.

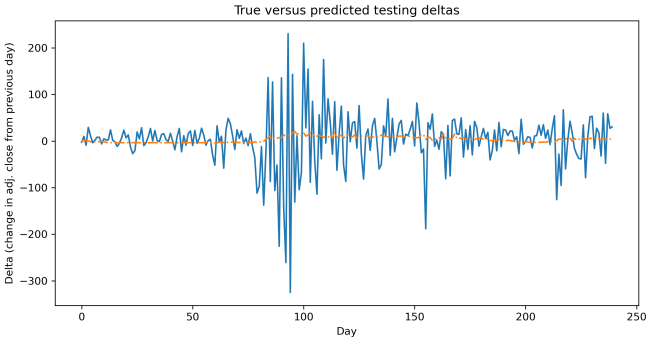

Figure 2: True versus predicted testing delta values

As we can see, the random forest is making all of its predictions at values very much near a delta value of zero. While the predictions are very far from the actual delta values, we can check whether the model is correctly predicting the sign of the deltas (predicting whether a stock price will increase or decrease is a common related problem to predicting the actual value).

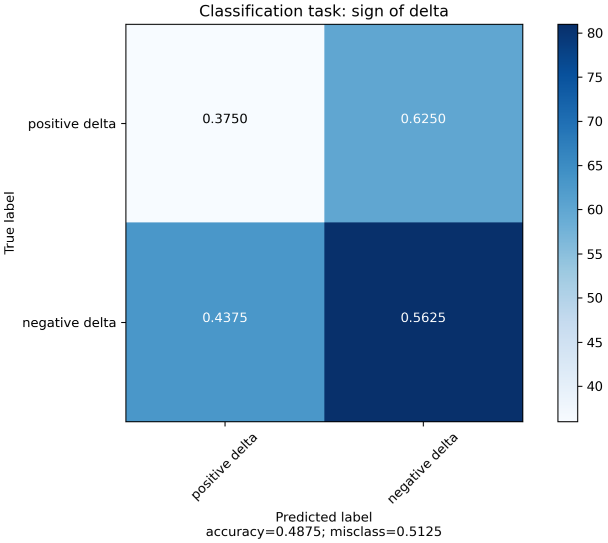

Figure 3: Confusion matrix for classification task of predicting the sign of delta on testing data

From considering the classification analogue of our regression problem we can see that through trying to get as close to baseline behavior as possible, our random forest only reaches a 48.75% accuracy on the task of correctly predicting the direction of change of the S&P price.

Random forest feature importances

Using SHAP we calculated global and local feature importance values for our random forest model. We include all of the findings in our notebook and figures folder in the attached GitHub repository, but we will summarize the findings here with the global feature importance. The top 10 most important features are given (in order):

- VIXCLS_4 (volatility index from four days ago)
- VIXCLS_3 (volatility index from three days ago)
- High_13 (market high from 13 days ago)
- Close_5 (closing price from five days ago)
- Volume_2 (stock volume traded two days ago)
- Low_5 (market low five days ago)
- Close_12 (closing price from 12 days ago)
- Close_13 (closing price from 13 days ago)

- VIXCLS_1 (volatility index from one day ago)

Overall we see an unsurprisingly broad assortment of past days' data included, but from a reasonable assortment of concrete factors that have very intuitive ties to the day-to-day change in index price.

SVR results

Our support vector regressor displayed interesting behavior which contrasted with our random forest, and which made a lot of intuitive sense given the nature of the classification executed by a support vector machine. From looking at the SVR's behavior on the validation data we see that the model focused its predictions in a narrow band centered near zero, and in doing so was able to get a high validation classification accuracy on the task of predicting the sign of the delta values.

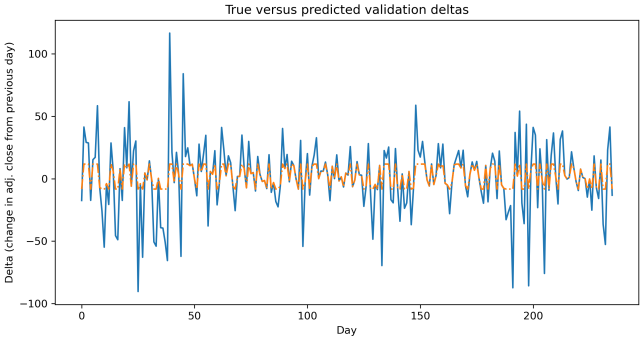

Figure 4: True versus predicted validation delta values

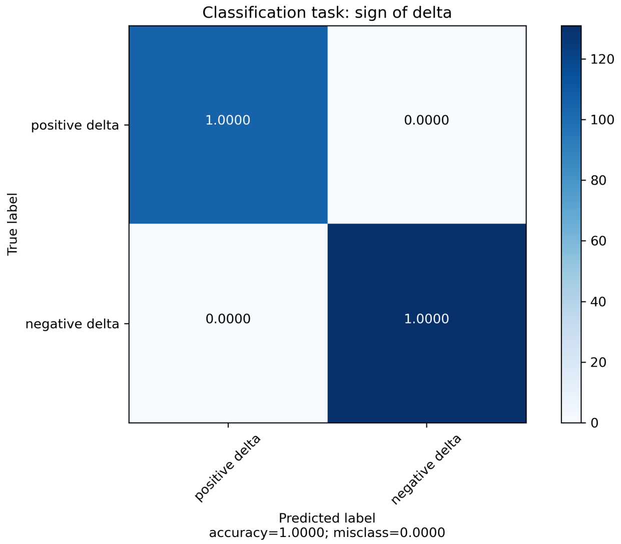

Figure 5: Confusion matrix for classification task of predicting the sign of delta on validation data

However, the model's behavior flattens out and becomes consistently negative on the test set:

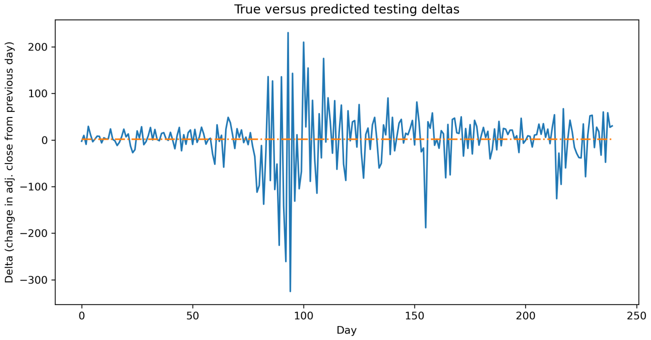

Figure 6: True versus predicted testing delta values

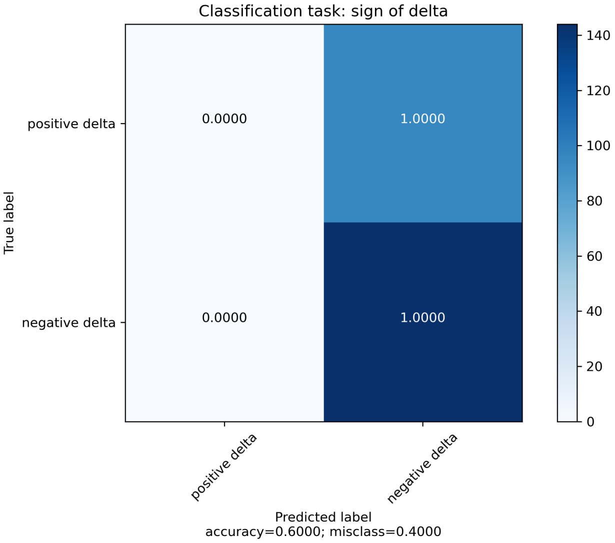

Figure 7: Confusion matrix for classification task of predicting the sign of delta on testing data

SVR feature importances

Using a permutation feature importance test we find the 10 most important features to the SVR to be:

- Volume_2 (stock volume traded two days ago)
- VIXCLS_3 (volatility index from three days ago)
- z_14 (delta from 14 days ago)
- VIXCLS_1 (volatility index from one day ago)
- VIXCLS_4 (volatility index from four days ago)
- VIXCLS_7 (volatility index from seven days ago)
- z_2 (delta from two days ago)
- Low_5 (market low from five days ago)
- z_13 (delta from 13 days ago)
- z_12 (delta from 12 days ago)

The notable result here is the focus of the SVR on volatility values and past delta values.

XGBoost results

The notable difference that is visible in the XGBoost model's performance was its unique ability to fit its training predictions such that there is a strong correlation between the true and predicted values:

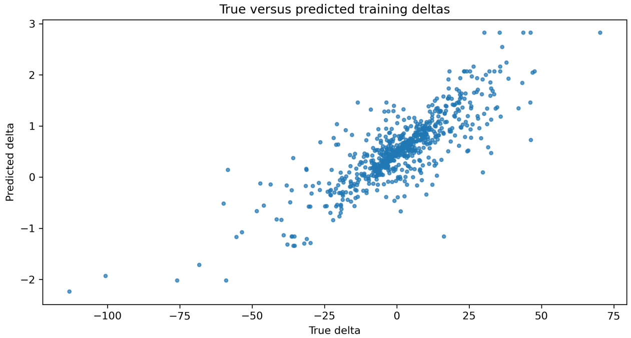

Figure 8: Correlation between the XGBoost model's training predictions and the true training values

This result is a testament to the error-correcting capacity of gradient boosting. However, on the testing data, we find similar results as with the other two models:

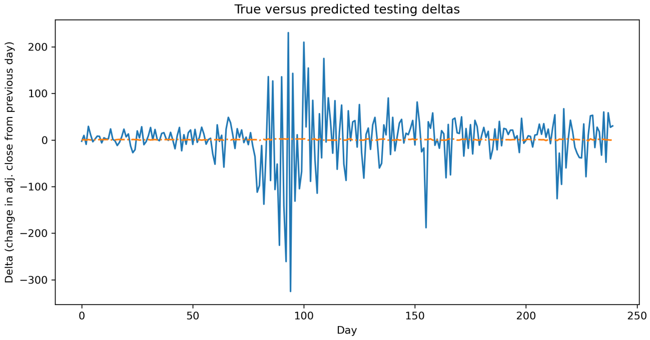

Figure 9: True versus predicted testing delta values

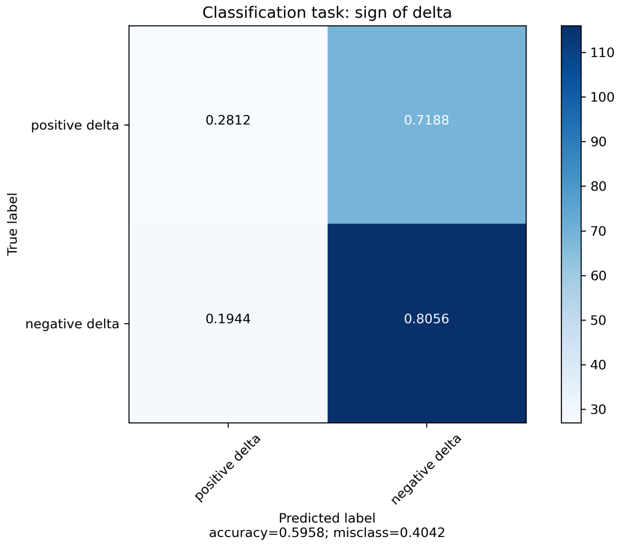

Figure 10: Confusion matrix for classification task of predicting the sign of delta on testing data

XGBoost feature importances

Using SHAP we find the 10 most important features to the XGBoost model to be:

- VIXCLS_3 (volatility index from three days ago)
- VIXCLS_1 (volatility index from one day ago)
- VIXCLS_4 (volatility index from four days ago)
- z_1 (delta from one day ago)
- z_2 (delta from two days ago)
- VIXCLS_2 (volatility index from two days ago)
- z_8 (delta from eight days ago)
- z_9 (delta from nine days ago)
- z_3 (delta from three days ago)
- Open_0 (opening prince from current day)

Again we see a wide variety of days accounted for in this set, but it is interesting to notice that the XGBoost model made its nine most important features all measures of either volatility or previous days' delta values.

The results included here provide an overview of the experiments conducted, but further results and figures can be found in our notebook and figures folder on the attached code repository.

5 Outlook

The results we found from our experiments here provide thorough evidence for these models being ill-suited for the regression task of predicting values of a time series - especially one as volatile as the S&P 500 index. However, these models were able to provide a great deal of insight to the reasons and nature of their malfunctions by nature of the models' high level of general interpretability.

Overall, the obvious shortcomings of the modeling approaches taken in this project are the result of these models inability to extrapolate from given data. Considering the nature of tree-based methods, the results we saw here are highly unsurprising especially when given the volatile data that these models were being tasked with predicting (that being time series data that included that greatest incidence of stock market volatility in modern history - the start of the current pandemic).

However, the varied regression approaches of our three models did provide a look at a couple of techniques that could be useful in different time series prediction problems. First, we saw in the training behavior of the XGBoost model the ability to formulate predictions that were highly correlated with the ground truth. Second, we found that the SVR has strong classification capacity which proved to have some success in the classification analogue to our regression task.

References

[1] Federal Reserve Economic Data. Cboe volatility index: Vix, 2020. Data retrieved from FRED, `https://fred.stlouisfed.org/series/VIXCLS`.

[2] National Oceanic and Atmospheric Administration. Climate at a glande - national time series, 2020. Data retrieved from NOAA, `https://www.ncdc.noaa.gov/cag/national/time-series/110/tavg/all/10/2015-2020?base_prd=true&begbaseyear=2015&endbaseyear=2020`.

[3] Organisation for Economic Cooperation and Development. Consumer confidence index (cci), 2020. Data retrieved from OECD, `https://data.oecd.org/leadind/consumer-confidence-index-cci.htm`.

[4] Yahoo Finance. Sp 500 historical data, 2020. Data retrieved from Yahoo Finance, `https://finance.yahoo.com/quote/\%5EGSPC/history?p=\%5EGSPC`.