Feature Selection

Given that this dataset was relatively high-dimensional, I wanted to first use filtering techniques to reduce the number of features independent from any particular downstream model. The choices I tested were mutual information, correlation between features, and correlation between features and the target variable.

Y1 and Y2 seemed to have low signal-to-noise ratio and as such no feature had high correlation with the target or large mutual information. Thus, I chose a set of features for which no two variables had correlation greater than or equal to .3, which would indicate moderate correlation. The logic was that including a set of highly correlated features is unnecessary and likely provides little more information than including a single one of the features.

This resulted in about 23 uncorrelated features.

Because the feature selection technique should align with the downstream model, and the feature set was now relatively small, I settled on two possibilities: (1) Use LASSO for feature selection with an alpha = .01 so as not to be overly aggressive and then train a linear model with the features given non-zero weight or (2) Use a shallow (depth = 2, number of estimators = 50) xgboost model and select the top ten features and train a more complex xgboost model with these features.

I tried both methods. The approach using LASSO and linear regression performed better and is more interpretable, which is why I chose it.

Calibration Method

I tried all three permitted methods since it was unclear a priori which would perform best.

The neural networks I trained using Keras seemed very prone to overfit unless they were extremely small, in which case using neural networks seemed excessive.

Gradient-boosted trees performed better than neural networks but similarly to linear regression in terms of MSE and worse on directional accuracy.

Thus, I settled on linear regression because it was the most interpretable and yielded the best results amongst the models I tried.

Data Sampling

I did not use any data sampling for this task. If time constraints were more pressing then sampling could help by reducing the size of the datasets. Another possibility is data sampling could reduce the noise in the data - related to the concept of aliasing in signal processing.

However, I decided to not sample because choosing how exactly to sample the data without introducing bias or reducing an already small signal seemed like a dead-end. Given a broader scope of methods, I would attempt smoothing and de-noising techniques.

In-sample and out-of-sample-periods

It seemed reasonable to have the out-of-sample periods be the most recent data. The logic being that if there are continuous trends in these data with respect to time then model performance on the most recent data would be more indicative of future model performance. Thus, I chose in-sample to be 20220103 - 20220930 and out-of-sample to be 20221001-20230331.

Other Thoughts

I used the augmented Dickey-Fuller test and the original Mann-Kendall test along with plotting Y1 and Y2 against time (average Y1 or Y2 value for each day against the day). Each of these established that Y1 and Y2 appeared to have no statistically significant trend over time and they are essentially stationary. Indeed, in the process of choosing a model and a set of relevant features, I noticed that most models performed very similarly. Although it is not clear what exactly Y1 and Y2 are, they are clearly difficult to predict and may be related to stock returns, which are notoriously noisy.

It made sense then to establish a baseline performance and useful metrics by which to judge model performance. I chose the mean value of the training set as the baseline predictor and I used root mean squared error (RMSE) and mean directional accuracy (MDA) as metrics. For a model to be better than baseline it should have lower RMSE than the mean predictor and greater than 50% MDA.

If Y1 and Y2 represent stock returns, which they may not, an MDA over 50% is useful even if only by a percentage point. The linear model I trained for Y1 had about a 51% MDA on the out-of-sample data and a slightly lower RMSE than the baseline mean predictor and the linear model for Y2 had about a 50.6% MDA on out-of-sample data and a slightly lower RMSE than the baseline mean predictor.

For feature selection, I tried both forward selection and recursive feature elimination using the uncorrelated features mentioned before with the linear model. Neither of these yielded better results than simply using the LASSO to wither down the uncorrelated features.

I noticed that Y1 and Y2 are very highly correlated. In fact, they are about 95% correlated. I believe in practice the calibrations would depend on more context. For example, if you want a balance of (1) your calibrations being as uncorrelated as possible and (2) minimizing RMSE on

the individual tasks, this would yield different calibrations than if you solely wanted to minimize RMSE on the individual tasks.

Calibrations

For both calibrations, afternoon = 1 when the time is *strictly* after 12:00 PM and afternoon = 0 when the time is before or equal to 12:00 PM.

Calibration for Y1:

Y1 = -0.09907126 + 0.1568341314837869*X1 - 0.037324892051129425*X10 + 0.11098108743690752*X114 + 0.09553339455383426*X116 - 0.015637364884467157*X118 + 0.1375949409034615*X143 + 0.15323034278719955*X147 - 0.05874139139016094*X372 - 0.151608163335215*X44 + 0.025911287285092403*afternoon

Calibration for Y2:

Y2 = -0.23209854 + 0.4449436772263295*X1 + 0.06006251850779655*X10 + 0.07903327626190287*X110 + 0.160591162622354*X114 + 0.10806743486272212*X116 - 0.048542652223079526*X118 + 0.6347399136750168*X20 - 0.4108515271210851*X335 - 0.11927884632174064*X372 - 0.35644407503369435*X44 - 0.06072720349867292*X74 + 0.13130361914398447*afternoon