Save and package your model for deployment

Given the strength of the model generated in the last analysis, this model will be chosen as the best model going forward, and is thus ready for saving and deployment. This report will discuss deployment methodology, performance metrics monitoring, data drift, or concept drift.

1. Deployment Methodology

When deploying a model, there are several methods regarding use cases and availability. This model will utilize batch reporting as its preferred method. The initial goal of this analysis was to provide market participants with statistical insight for the upcoming Federal Reserve monetary policy action in advance of the FOMC so as to be able to hedge against the rest of the market, usually the Short-Term Treasury Bond market. As such, this model will ideally be run some time—roughly a week—before each FOMC meeting, which happens eight times a year during normal years (exceptions were made during the Covid-19 pandemic, which required the Federal Reserve to meet to discuss the providing of emergency stimulus to American citizens and how to adjust interest rates accordingly). Batch prediction also has the benefit of delayed latency, as downloading and utilizing real-time data would be time-consuming and impractical.

2. Model Performance Tracking

Tracking the performance of this model post-launch will be very importance given the degree of overfitting in the final model. A model that has only a moderate amount of predictive success is not worth publication, or could lead to some adverse effects in public usage. In a worst case scenario, though unlikely, a great deal of money is hedged against Short-Term Treasury Rate price changes using this model, and a market participant loses a good sum of money. As such, tracking model performance is incredibly important.

To do this, the predicted values will be calculated for ordinal log-loss. This is a modified version of the standard log-loss that is designed for ordinal data, like the monetary policy action predicted. This performance metric seeks to penalize a model's predictions based on the degree of misclassification. Ordinal log-loss also benefits from being able to return the direction of error. So, if the model is routinely predicting too high, then ordinal log-loss will be able to capture that. While simple accuracy sufficed for model creation, when considering the gradual degradation of model accuracy over time, the degree to which the model incorrectly predicts monetary policy action will be necessary. Selecting a baseline value for log-loss error will help automatically flag when the model is no longer meeting required performance standards.

For performance tracking, it is best practices to identify a green light (all is well with the model), yellow light (errors require tracking), and a red light (model should be refit) thresholds. Given that ordinal log loss takes directional error into account, the absolute value of ordinal log loss will be considered. These thresholds will be [0.0 to 0.2], [0.2 to 0.5], and [0.5<] for green, yellow, and red lights respectively. Because of the low frequency of prediction (8 observations every year), unless the model dips egregiously in predictability, it is not likely that the model will immediately reach critical.

That said, there are certain ways to prevent this from happening. During the “yellow light” period, it could be useful to analyze the actual residuals rather than just using the log-loss value as a signal. Should the loss values be negative, this can indicate that the model is over- or underestimating. If multiple iterations and observations in a row are incorrect in a single direction, this may be the result of bias, which would prompt an investigation into further variables in the training set that may cause bias.

Similar things can be said for ordinal Mean Average Error. This is a much more straightforward performance metric that can show the magnitude and direction of prediction error versus real values. It also has a much more straightforward interpretation, as it would represent the average degree and direction to which the model incorrectly predicts monetary policy action. Similarly, the thresholds for yellow and red lights will be the absolute value of 0.5 and 0.75 respectively.

The model will also go through a natural retraining lifecycle. Retraining would feel necessary, as the limited nature of the dataset would call for a more accurate model when given more datapoints. Further, naturally pushing the training/validation set cutoff further into Jay Powell’s tenure would strongly reduce the bias in the training set, and could result in a more generalizable model. Eventually including the tumultuous period during the early months of the Covid-19 pandemic into the training set would further allow the model to be able to capture exigent scenarios and make for a more malleable model. Because the original dataset was already so limited, retaining may result in a wildly different model, even with the same model and features. Retraining every four years, in accordance with the general length of tenure for a Fed Chair as well as President, would be practical, but could lead to overfitting and bias in the training set. That said, retraining it every year or every eight meetings would feel redundant, and any inherent seasonality in the model would be reset each time. Thus, retraining the model every two years or 16 meetings would be apt for this model.

3. Data Drift Tracking

Should the model begin to be less and less reliable, retraining will become necessary, requiring updating data inputs and refitting the model. Increasing error could be the result of the natural relation between variables changing over time, a key aspect of data drift. For example, it could be that the FOMC Minutes change in their intention, and thus naturally alter the Net Sentiment Score without intending to alter their monetary policy decisions. It could also be that in 2024, following an election, a new Fed Chair may take over for Jay Powell, who could have wildly different vernacular in their speeches, and could alter the effect of Net Sentiment Score.

For performance metrics, Kolmogorov-Smirnov Testing will help detect if the distributions of specific features have changed over time. This will help determine if the model requires reformatting. One benefit of using Kolmogorov-Smirnov Testing (KS Testing) is its non-parametric nature, which makes it versatile and applicable to all sorts of data distributions. It also is widely known, and has an easy interpretation, with a significance level that can be flagged should the data drift outside of preferred distributions.