Insider Project

Description

ABOUT US

At Meson Gravity, we are dedicated to fundamental investing on a modern software foundation. We have built an asset management platform that utilizes data science, machine learning and software-defined business processes to automate financial analysis, risk management and trading.

That's why:

- We use open-source modern tools like Pandas, sklearn and Jupyter to build models that make predictions about what stock is likely to outperform the market over the next 12 months while respecting what fundamental stock analysts do.
- We use 1000s of portfolio simulations run in the cloud to test hypotheses around potential risks embedded in our portfolios while respecting hard-won portfolio manager experience.
- We use AWS for data and compute and run a server-less architecture to handle our trading execution while respecting the intricacies of traditional trading operations.

INVESTMENT IS PREDICTION

If you invest, then you are squarely in the prediction business. That's why we are always looking for high-quality research teams to continuously improve upon our ability to make predictions. We look forward to getting to know you and the high-quality research you are capable of. In fact, this project is designed to get to know each other better.

PREDICTION MODEL OVERVIEW

This task is to develop a prediction model of a stock's binary outcome 12-months in the future given known insider buying and selling activity. The best way to think of this is looking for signal to separate winners from losers among potential investments.

PREDICTION MODEL DETAILS

The universe we are studying consists of stocks of publicly traded companies. Each company is identified by a **tradingitemid**. The prediction model we are building will answer the following question about companies in the universe:

Given features for a specific company and date, what is the probability output of a binary classifier?

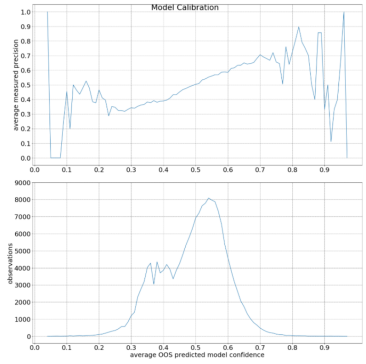
Feature data is a time time series of weekly snapshots of insider trading statistics indexed on **(date, tradingitemid)**. Feature names are preserved as column names for your convenience and are easily interpreted. Please use your domain knowledge when preparing this data and choosing how to impute, normalize etc. Always feel free to ask questions about the data.

Outcome data is a time series of binary outcomes encoded as 1 or 0. You can join outcomes with features on **(date, tradingitemid)**. Each outcome represents a measurement made in the *future* and that is important to keep in mind to prevent *data leakage* when splitting up the data for training and testing.

Test data is a time series of samples organized by **(date, tradingitemid)**. We hold out outcome data from 2016-01-01 and on and will use that data for evaluation of submitted solutions.

Evaluation

Model evaluation is intentionally not a single score. When evaluating your model the team will first use the holdout set to evaluate model calibration. Our metrics are sensitive to how well your model can estimate probability of the given outcome and how many high-confidence predictions your model can make. Further we will feed probabilities from your model to a market simulator to measure hypothetical returns, volatility and other characteristics of the generated signal. You are free to propose any additional metrics you've found useful when designing your solution.



It is helpful to check your model output using calibration plots where a linear fit closer to y=x indicates a better calibrated model.

A calibration plots y as function of x:

- x=bins of out-of-sample predicted model confidence i.e. output of "predict_proba" if using sklearn classifiers.
- y=average measured accuracy of predictions for a given x (bin).

Not too familiar with model calibration? A helpful reference on model calibration can be found here: https://scikit-learn.org/stable/modules/calibration.html. Brier score or a similar metric can be used to rank sets of predictions to find better calibrated models.

You will want to generate multiple train/test splits from the data provided for training. Keep in mind that most common cross-validation techniques are not suitable for time series data. See a more detailed discussion here: https://towardsdatascience.com/time-series-nested-cross-validation-76adba623eb9. Cumulative walk-forward training (aka forward-chaining) with 1-year steps and a fixed starting date is what we commonly find acceptable within the team for model scoring and comparison.

Data

The following data files are included.

train_features.pkl - features of the training set. The file contains time series of statistics for insider trading activity at public companies. Each company has a persistent unique identifier. Indexed by (**date**, **tradingitemid**).

train_outcomes.pkl - binary outcomes are represented in columns and class membership is encoded as 1 or 0. Classes are balanced, so expect an equal number rows labeled as 1 and 0 for a given date. Join with features on (**date**, **tradingitemid**).

- **g__12m_binary** - for a given sample this column indicates a binary outcome looking *12 months into the future*. To prevent data leakage, when performing train/test splits the earliest test sample must be dated not earlier than 12 months after the latest training sample.

test_features.pkl - the test set for generating out-of-sample predictions for evaluation. Indexed on (**date**, **tradingitemid**).

example_predictions.pkl - example of a predictions file. This file uses **train_features.pkl** and was produced using walk-forward training discussed above. Generate a similar file using **test_features.pkl** and submit with your solution.

Download link:
https://mesoncapital-shared-data.s3-us-west-2.amazonaws.com/research/projects/insider-1/handout.zip

Solution

Please submit your solution as a ZIP archive containing data files with test predictions and Jupyter notebooks with all the code and comments necessary to reproduce your results.

Timeline

Step 1: Receive project description and associated notebook. Ask questions! Set up research environment to start studying the problem.

Step 2: Study the problem. Explore the data, engineer your features, write training and validation code, train and tune models, plot calibration etc. This is where we hope to see you spend the bulk of your time, but be mindful of the assignment's time constraints.

Step 3: Generate predictions from the best model you found. Share your predictions file with us and include the notebook(s) used to generate the predictions, prepare the features and train the model. We encourage you to include any additional notebooks or other documents that explain how you arrived at your solution and how you dealt with various challenges that arose along the way.