Final Project

Part A Algorithmic Trading System using Machine Learning

Objective

Develop a trading algorithm that uses machine learning to predict stock price movements and make trading decisions.

Rules:

- You start wiith a portfolio worth 1Million (in Cash)
- Interest rate = 0
- You can only go long on an integer number of stocks.
- You cannot borrow , neither go short.
- Before buying new stock, you need to first sale some other to have the cash (no direct stock conversion)
- There is only one price per day. At which you sell and buy.
- The last day you sell all your assets and get the cash.

Data.

The data will be given to you (created by me). All of you will have the same data

It will consist on 3 years of stock prices for 10 stocks (ficticious prices) that you can use as training-validation, and 6 months that you need to use as test. (don't forget to have validatin set!)

Additionally I will have extra 6 months data (test2) to verify that that your est performance indicators also hold with my data.

Data Preprocessing

You should handle missing data, bad data or outliers in training data before anything.

You can create additional features from the prices if you wish including correlations, covariances, volatilities, or any addirional variale that you consider interesting;

Or, on the contrary sense, you can use PCA to reduce the dimensionality

Model Development:

Decide which model to use (supervised or unsupervised, regression or classification, neural network, logistic regression, or simply rules based approach)

Hyperparameter Tuning: find the best parameters for your model.

Trading Strategy Development:

Signal Generation: Define rules when to buy, sell, or hold based on model predictions. For instance, if the model predicts the price will go up, the signal is to buy. (signal can consists on several stocks at the same time)

Backtesting: Simulate the trading strategy using validation data to assess performance on new data

Use validation data while finetunng the model, and test data obly for the winning model

Evaluation:

There are a number of performance indictors that tell s if the strategy is good or not. These should be computed always for training, validation and test data.

- Average annualized expected return
- Annualized sharpe ratio on training
- Maximum drawdown on 6 months
- Cumulative return (on validation and test)
- correlation of my portfolio with equally weighted market portfolio

Documents to present.

You need to present 2 documents.

- A word/pdf file explaining all you have done, your decision, methodology used, result of algorithm for different performance measures, explain differences between training and validation and test performance, etc. I expect this document to be well written and presented, almost as en executive report tring to convince a Hedge Fund Manager to implement your strategy.
- A Jupyter notebook that works if I restar kernel and run all cells, in which all the calculations you discuss in your report can be verified.
- The application f the strategy to the test set should be in such a way that if I substitute your file for mu test2 set, all run smoothly and I get agaiall the performance indicators.

Deliverable

I expect a zip file containing

- The word/pdf document (please, write ALL your names in the document)
- The jupyter notebook (or python project in pycharm/vstudio code if you prefer)
- The data

Part B Fast price approximator to time dependent derivatives

Objective:

Develop a rapid approximation function for pricing time-dependent derivatives using a neural network. The network will be trained on prices generated from Monte Carlo simulations to provide a quick and efficient pricing mechanism.

Background:

Time-dependent derivatives, such as Asian and barrier options, have payoffs that depend on the entire path of the underlying asset. Traditional pricing methods, such as the Monte Carlo method, can be computationally expensive and time-consuming, especially for real-time or high-frequency trading environments. This project aims to approximate the pricing function for these derivatives, providing rapid estimates with reasonable accuracy.

Steps:

Data Generation: Use Monte Carlo simulations to generate a datasets (training validation test) of derivative prices based on various initial conditions and parameters.

Design a neural network architecture suitable for regression tasks.

Train the neural network on the generated dataset. Validate its performance and adjust hyperparameters as necessary.

Test the neural network's pricing capabilities against test set. Measure the speed of the approximator and compare accuracy/speed to traditional montecarlo methods

Conclusion and Recommendations:

Assess the viability of the neural network approximator for real-world applications. Discuss potential improvements, and limitations

Documents to present.

You need to present 2 documents.

- A word/pdf file explaining all you have done, your decision, methodology used, results and recommendations. I expect this document to be well written and presented, that could be used at a financial insitituion as support to implement or not the ‘fast pricer’
- A Jupyter notebook that works if I restar kernel and run all cells, in which all the calculations you discuss in your report can be verified.

Deliverable

I expect a zip file containing

- The word/pdf document (please, write ALL your names in the document)
- The jupyter notebook (or python project in pycharm/vstudio code if you prefer)