The aim of the project is to familiarize you with the process of applying data mining or machine learning techniques to solve real-world problems. This project will focus on the tasks of stock price forecasting and trading strategy optimization.

Grading

The final grade will be computed as

30% basic score + 70% competition score + bonus score

The basic score (100 points for graduate students and 150 points for undergraduate students) is based on the completion of the assigned tasks, while the competition score (100 points) is determined by your evaluation metric ranking compared to those of your peers. The evaluation metric used for this competition is **precision in classification, not accuracy.** You also have the opportunity to earn a 20-point bonus for successfully completing Task 7.

Requirements

What you need to submit is **a single well-formatted Colab notebook** (via a sharing link), that is optimized for readability. This notebook should contain **all outputs required in the tasks**, as well as **all training and testing records** of your program.

1) You are required to use Python (sklearn, pandas, numpy, etc) and Pytorch. You are not permitted to use Tensorflow, Keras, Theano, etc. Your code **must be reproducible** on Google Colab **by clicking "Edit->Run All".** Additionally, your code should be well documented with comments explaining what your code is doing and what your outputs represent.
2) You are welcome to communicate with your classmates; you are however **NOT** allowed to share or copy from other people's code. **Your implementation must be your own**.

Suggestions

1. Begin on the first day, as completing the whole project within a month may not be feasible;
2. Learn actively;

3. Grading is based on the outcome, so it is crucial to present the results in a clear and succinct manner;
4. Keep in mind to consistently save your model checkpoints, allowing you to reload your models as needed.

Team

Form a team of 1-2 using this spreadsheet (to be created).

Introduction

Data Description:
You have been provided with historical stock market data and your task is to use the data to train and test various machine learning models for predicting stock prices. Specifically, you have been given three datasets, namely one training set and two testing sets, which can be loaded using the pickle.load() function.The training set is a list of 2000 pandas dataframes, each representing the historical trading data of a single stock. There are a total of 2000 stocks in the dataset. Each pandas dataframe contains 2202 trading records for consecutive time points, each record including 5 features, namely “Open”, “Low”, “High”, “Close”, and “Volume”. Note that all the numbers in the dataset have been normalized.

Task Description:
In this project, you will be working on training a model to predict the future stock prices based on the past 100 trading records. This is a supervised learning problem, where the input features consist of a matrix with 100 rows and m (m>=100) columns, and the label is a 3-dimensional vector. We will provide further details on this later on.

Tasks

Task 1: Data Preparation (5 points for grad, 7.5 points for undergrad)

In this task, you will be responsible for preparing for the labels for the data mining problem by following these steps:

1. Calculate the daily percentage changes of the close prices for each stock.
2. Divide the daily percentage changes into 3 levels, ensuring that the number of the data points in each level is roughly equal. These 3 levels will represent "decrease", "no big change", and "increase" respectively.
3. Output the two thresholds used to divide the price changes in step 2, as well as the number of the data points for each level.

Hints: it may be helpful to sort the daily percentage changes to find the appropriate thresholds.

Task 2: Feature Engineering (15 points for grad, 22.5 points for undergrad)

In this task, you are required to create at least 95 new features for the stock data. However, feel free to create more if you believe it will improve the prediction results. In general, the more features you have, the higher the chances of obtaining accurate predictions.

Suppose that you have m (m>=100) features in total after the feature engineering step. Print the last 5 data points for the first of the 2000 stock frames in the given training dataset. The result should be a dataframe with 5 rows by m columns.

Hints:

You are free to design your own features that you believe will assist in predicting future prices. This can include the difference between "High" and "Close" prices, as well as various statistical indicators such as standard deviation, range, and mode. Additionally, you may also consider financial technical indicators such as Sharpe Ratio, Moving Average Convergence Divergence (MACD), Relative Strength Index (RSI), and Money Flow Index (MFI). Please keep in mind that you should **NOT** share your features with other teams.

Task 3: Training Predictive Models (20 points for grad, 30 points for undergrad)

Divide the training dataset into two subsets: a primary training set and a validation set. Utilize the primary training set to train at least 10 models, with the goal of predicting which of the three levels the next stock price will fall into, based on the past 100 trading records. Keep in mind that the features are represented as a matrix of 100 x m, and the label is a 3-element-vector.

For each of the 10 models, evaluate their performance on both the primary training set and the validation set by measuring the following three metrics:

1. Prediction accuracy: the proportion of correctly predicted outcomes;
2. Classification precision: the proportion of true positive predictions among all positive predictions (precision measures how many of the items that were classified as positive were actually positive);
3. Percentage of positive predictions: the proportion of predictions that fall into the "increase" category (the 3rd level).

Note that the "increase" category (the 3rd level) should be considered as positive, while the "decrease" and "no big change" categories (the 1st and 2nd levels, respectively) should be considered as negative. This will result in a total of 10 (models) x 2 (datasets) x 3 (metrics) = 60 numbers to be reported. Additionally, ensure that the percentage of positive predictions is greater than 10%, as a percentage lower than this would indicate that the model is not acceptable and need to be revised.

Hints:

1. Our ultimate goal is not to predict prices with complete accuracy, but rather to make profitable decisions based on our price prediction model. Therefore, we actually hope that when we predict that a stock will rise, it will actually rise. So, the final evaluation metrics for this project will be the precision and under the condition that the percentage of positive predictions is no lower than 10% rather than the accuracy. To achieve this goal, **you should adjust your loss functions** to achieve high precision, even if it means accepting a decrease in accuracy and a lower percentage of positive predictions.
2. You can train 10 or more models such as random forest, SVM, and neural nets. You can also train neural nets independently for 10 or more times.
3. Keep in mind that since the data dimensionality is high, your models should not be too small. Additionally, it is crucial to utilize the validation set to optimize hyperparameters such as model capacity, learning rates, and the number of epochs to achieve optimal performance.

Task 4: Feature Selection (25 points for grad, 37.5 points for undergrad)

Select a specific subset of features and repeat Task 3. Once completed, present the three evaluation metrics. Remember to provide the 60 numbers as output.

Hints: you are free to utilize any method of feature selection that you prefer. Additionally, consider reducing the number of past trading records by using the past 60 or 30 trading records to predict future prices. Using a subset of the data sets for training the models may also improve performance. Make every effort to improve the results.

Task 5: Model Ensembling (voting & blending) and Adaboosting (25 points for grad, 37.5 points for undergrad)

If the models in tasks 4 and 3 do not meet your expectations, there is a chance to enhance your final results for free. Model ensembling and adaboosting are two major approaches to boost your models' performance, once you have multiple weak models in place.

Utilize the methods of voting, blending, and Adaboosting to enhance the performance of your 10 models.Present the results of the three evaluation metrics on the primary training set and the validation set after implementing these techniques. You should output 3 (methods) x 2 (datasets) x 3 (metrics) = 18 numbers.

Task 6: Model Evaluation (10 points for grad, 15 points for undergrad)

Choose the best model from Task 5 as your final model. For Task 1-5, you should not touch the two testing sets. In this task, evaluate the performance of your final model on the two testing sets. Report the results using the three evaluation metrics. It is expected that the evaluation results on the two testing sets as well as the validation set will be consistent.

Note that your final ranking will be based on the classification precision (rather than accuracy) on **a private testing set** that is not released to you, as well as whether the percentage of positive predictions exceeds 10%. If your evaluation results on the two testing sets are consistent, you can expect consistent good results on the private testing set.

Task 7 (bonus 20 points for all): Trading Strategy Optimization and Backtesting

Suppose that you have been given $10,000 US dollars. Use a reinforcement learning technique such as policy gradient, PPO, or Q-learning to develop a trading policy that maximizes the cumulative returns of your portfolio.

A trading policy, in this context, refers to a function (parameterized model) that maps features (represented by a 100 x m matrix) to a decision, which can be encoded as a 3-element-vector indicating "buy," "sell," or "hold" at each time point. You may choose to train your trading policy from scratch or utilize predictive models from previous Tasks 3-5.

Denote your policy model as f_θ, and your 10 predictive models as p_1 to p_{10}. The action is represented as a 3-element-vector, and the features are represented as X (a 100 x m matrix). One approach to constructing the policy model is as follows:

$$\text{action} = f_\theta(p_1(X), \ldots, p_{10}(X)),$$

where the policy model takes the predictive results as features (similar to the blending technique). If you choose to train your trading policy from scratch, action = $f_\theta(X)$. However, you are free to design your own approach.

The loss function for this task is the negative cumulative money return (the negative reward), rather than the difference between the prediction and the actual outcome as in typical prediction tasks.

After effectively training your policy, you can analyze the 2000 stocks in the training set to identify which to buy, sell, or hold at each time point. To determine the quantity of shares to buy or sell, you can utilize risk management techniques such as the Kelly Formula or the Minimum Variance Portfolio Method.

Output the rate of return, Sharpe Ratio, and maximum drawdown of your portfolio on both the training set and the two testing sets. Additionally, visualize the cumulative monetary returns of your portfolio on the training set and the two testing sets.

Hints:

1. You can learn about policy gradient method through this link: CS 285: Lecture 5, Part 1
2. You can learn about more reinforcement learning algorithms at this link:Spinning Up in Deep RL!