Project Overview

Kolten Safron – DTSC 691

Note that most of these details are all included in the Juypter notebook as well so there will be some duplication.

Project Scope and Description

The stock market has been around for decades and has proven to be an enormous opportunity for generating wealth. Through the years data has been maintained on all the transactions that have occurred, generally summarized in time intervals (Ex yearly, monthly, daily). The markets themselves consist of trending markets, whether that is upwards, sideways or downwards. Given these trends and the significant amount of data available I am hoping to build a supervised machine learning model to predict the direction of the overall stock market. By overall stock market I am referring to the S&P 500 and more specifically the ES (S&P 500 mini futures). While the data maintained for the basics of the stock market is fairly simple, I believe with strong feature engineering it will be possible to create a model to take advantage of this opportunity.

To take it a step further, not only do I want to predict the direction of the stock market but I want to determine if you should buy, and subsequently what price you should sell at. To do so a strategy will need to be developed for when to enter and exit the market. The base strategy will include the market direction prediction from the machine learning model as well as a few other criteria and filters. Additionally, 3 key price points will need to be determined for the strategy (Note that these prices will be determined using a general algorithm and not machine learning):

1) The price to enter the market
2) The price to exit the market if your position is winning (Referred to as the "target")
3) The price to exit the market if your position is losing (Referred to as the "stop").

By determining these price points, the hope is to build an application that will determine if you should enter into the market at any given time to generate a profit. Due to this using intraday data (see below in data description) the trades will be short term, likely opened and closed within an hour.

In other words, with the above information I hope to create a bot that will be able to day trade the market. I believe this will be beneficial as it will help empower the individuals to make consistent profits off the market, without being reliant on a trained professional broker.

Data Description

For this project I focused on predicting the S&P 500 futures prices, or more specifically the E-mini S&P 500 (symbol ES). Futures contracts work a bit different than a typical security on the

markets. They do have prices that trend up and down just like normal securities however there are a few key differences:

- They have expiry dates which are rolled over every quarter. This won't affect this project much but just a key difference to note. There is an adjustment that is needed because of this, however this is already completed by the data provider. See the last paragraph of this section for more details on this.
- They prices change in increments of $0.25. That is the price won't go from $4,000.00 to $4,000.01, rather it will go to $4,000.25 - increase by $0.25 at a time.
- Futures work in contracts rather than shares or units. For all intents and purposes they behave the same.
- The profit or loss from these works different. Each $0.25 increment change in price represents $12.50 gain or loss. If you bought the ES at $4,000 and sold at $4,001.50, you would have profited 6 ticks ($1.50 / $0.25) or a gain of $75 (6 * $12.50). Vice versa is true if the ES dropped in price since you bought it. As a result, the profit and loss calculations done in this notebook will be a bit different then you expect. This of course is affected by the number of contracts you hold in the same way it would if you held shares.
- When trading futures, you are trading on a margin account. If the price of the ES is $4,000 and you want to purchase it, the amount of cash you need does not necessarily relate to the $4,000. Rather your broker will have set margin requirements for each contract of futures you wish to purchase. This means you will need X cash in your account to purchase one contract which is essentially used as collateral if the future goes in the wrong direction. The required margin per contract varies greatly by broker and their risk tolerance, ranging anywhere from $500 to $10,000 per contract. Note that you are purchasing the same instrument either way and it works the same, it is just a risk management tool done by the brokers.
- Due to how the profit and loss (PNL) is calculated as well as the margins, it allows you to have significantly more leverage and profit potential then just trading an index fund on the S&P while also still trading the same underlying instrument with high volume. Because of these reasons I believe it will be an ideal instrument for a day trading bot.

The data source for this project is FirstRate Data. They are a online company specializing in providing historical data for the stock market over a variety of instruments, symbols and timeframes. I do have access to an API through my broker to access this information, however I am limited in how far I can go back as such FirstRate Data was used instead.

ES prices are available in a variety of time intervals (Ex. 5 mins, 1 hour, daily, weekly). Each interval represents the summary of all the individual buys and sells that occurred over that time period. For this project 5 minute intervals were used as it will provide the highest sample count for both training and testing and verifying the strategy's profitability at the end. For this project each instance/time interval summary will be referred to as a "bar".

In our initial data we have the following features for each bar in the market:

- date - This will be the date and the 5 minute interval that the bar covers
- open - This is the price that the “date” opened at. In other words, this is the price of the first transaction in that time frame.
- high - This is the highest price achieved during the timeframe.
- low - This is the lowest price achieved during the timeframe.
- close - This is the price that the “date” closed at. In other words, this is the price of the last transaction in that time frame.
- volume - This is the number of units transferred or bought/sold during that time frame.

The data acquired goes from January 2008 to September 2023. Given changing market conditions throughout time and the large amount of data provided, I filtered the data to only include data from 2013 on. After the data is filtered to be only 2013 on, there is still over 700k samples.

The expiration of futures contracts cause a gap (significant change) in the price of the prior to the new contract. This raises issues as we will have abrupt significant changes in price. However, FirstRate Data provides an adjusted version of futures prices where they are adjusted/smoothed to remove these significant changes during contract expiration. This provides a smoother trend and will make it easier for the models to predict accurately.

Approach

In order to create this day trading bot, the following will be completed during the project:

- Data will be explored to ensure it is clean
    - Given the nature of the data there isn’t much to clean up
    - Outliers were identified and removed

- Data will be adjusted for stationarity given it contains significant trends
    - Different approaches were looked into
        - Differencing – Difference between the current instance and the prior instance.
        - Log return – A common financial metric, this is the log of the current bar divided by the previous bar.
        - Trend residuals – This involves fitting a linear regression to the data, and then calculating the residuals.

- Various features will be created to improve the predictive ability
    - Certain features consist of parameters you can tune. As such explored various parameters through a function to find the optimal parameters. This was decided based on correlation with the target direction

- The optimal features were selected

    - To determine the optimal features random forest importance was used. An initial model was run to get all features importance. From here a function was created to run a random forest with different groups of features based on an importance threshold set. The threshold that gave the optimal model is used to determine the final features.

- Data was split into train and tests sets
    - The standard train and test sets were created from data before July 2023
    - An additional evaluation set called the walk forward set was created for data from July 2023 till Sept 2023. This set is created to evaluate the model and strategy on current market conditions. It is another test to ensure that it is robust.

- Various machine learning models will be created and explored to determine the best model
    - SVC, random forest, gradient boosting, XGBoost, neural network, long short-term network, hard voting classifier
    - Grid search was done where possible, however there was computational constraints due to them often taking more than 8 hours to run. See below in complications section.

- Strategies will be created to try and profit off the predictions from the machine learning model
    - 4 different strategies were created ranging from more basic to advanced.
        - Advanced strategies contained things such as:
            - Several criteria/conditions that need to be met in order to enter a trade
            - Entry price
            - Target price
            - Stop price
            - Max loss price
    - Strategies were evaluated off a number of metrics to determine which strategy to use but annualized PNL and Sharpe ratio were the primary metrics.
        - Annualized PNL is our PNL adjusted for a single year
        - Sharpe ratio is a financial metric often used for evaluating investments and trading bot performance. Generally, a ratio >1 is acceptable, >2 is good, and >3 is exceptional.

- The optimal model and strategy will be utilized in a user-interface web application to output the results to the user
    - See below for more information on the user interface

Model & Strategy Evaluation

Predicting the movement of stock prices is a complex, difficult problem. As such when models are being evaluated we would not anticipate a high accuracy (Ex. 80 or 90%). For this project a model is determined to be successful if it achieves an accuracy above 50% or superior to a random guess. Even a small edge of a few % can allow a person to generate profits over a large enough sample size.

When evaluating the strategy's performance, we will look at a few different metrics such as profit and loss, win %, and Sharpe ratio. Ultimately, the performance of the model will be compared to if you just bought and held the S&P 500 over that period. Note that it is not compared to buying and holding the ES. This is because as mentioned above there is extreme leverage in the ES, so if you bought and held there is a strong chance that your account would be bankrupted on some of the downward corrections in the market.

User Interface

As the intended use of the application is to assist someone with day trading (show when to enter and exit trades) the user interface was created to provide them the necessary details. This was created through Flask and will be hosted locally.

The following will be the main inputs and outputs for the user interface:

Inputs:

- The user will input historical ES data into the application with it the data summarized in 5 minute bars.
- This will need to include the open, high, low, close and volume for that bar.

Outputs:

- The primary output of the user interface will be if they should take a trade at the most current price/bar. Hence, if all the criteria were met.

- In the case it recommends to take a trade, in order to assist the trader, it will then output the following:
    - The price to enter at
    - The target price to exit the trade at
    - The stop price to exit the trade if it is losing

Complications

During the project certain complications and difficulties arose that ultimately affected the final performance of the project. Had these not been present, I believe the performance could increase a fair amount more.

- Computational issues
    - - Given the amount of data being used and the complexities of the models, determining the optimal hyperparameters to use was constrained significantly. Performing grid search on the hyperparameters took a significant amount of

time, often times not finishing overnight. As a result, certain models were only able to have a coarse search or a search by trial and error.
    - This affected the feature selection process. Different approaches were looked into such as forward and backwards sequential feature selection, however due to the time to run this searches over my data I elected for a simpler approach.
    - This affected feature parameter optimization process. Initially I hoped to use random forest importance to adjust my features parameters to be optimal as this is the method used to select the final features. However due to the constraints of running a large number of random forest models this wasn't possible and a simpler approach was again used.
- Missing data - Although the data I used was complete, predicting the stock market is incredibly complex and a wide variety of data can be useful for improving the model. These could include such things as economic data or indicators and significant news events (financial or political). This information can help explain some of the moves in the markets at give times. This information was outside the scope of this project due to the financial costs of acquiring it.
- An alternative approach could of been looked into for training the models and implementing the strategies. A custom loss function could have been created to maximize the strategy metrics (Ex. profit) when training the model. This would have the benefit of implementing the strategy into the machine learning and as such optimizing the strategy which is our final product. However, due to limitations in sklearn's models this was not possible. This is something I would like to explore at a further data using another modeling package.

Conclusion

In this project we have created a day trading bot that appears to have some validity. In doing so we looked at a variety of features to try and encapsulate the direction of the market. We leveraged these features to create a variety of models, and ultimately an ensemble of them all in a hard voting classifier ended up being optimal. This classifier was combined with additional criteria and risk management tools to increase its statistical edge (strategy win % of 57%) and generate greater profits then a passive buy and hold strategy (Buy and hold: $4,066, day trading bot: $11,200).

It is important to note that although there is some validity to the strategy, its practical use has not been confirmed. Additional research should be done over this to ensure this isn't just overfitting the train or test data. Completing something similar to cross fold validation may provide additional confidence over the strategy.

Given my computational constraints throughout the project, it would be interesting to see how much we could increase performance and profitability but this is something to be explored at a later day.