Homework 4

Matt Hyatt

November 12, 2021

Predicting the Stock Market with SVM

Include the details on what kind of pre-processing you performed and your choice of model hyper-parameters. This assignments will be evaluated primarily based on your writeup. Please see the general guidelines for homework submission in the syllabus.

Dataset Selection

I used the `yfinance` python module to get 5406 stock values in `IVV`. I intended to classify the dataset according to whether or not the stock gained or lost value on any given day.

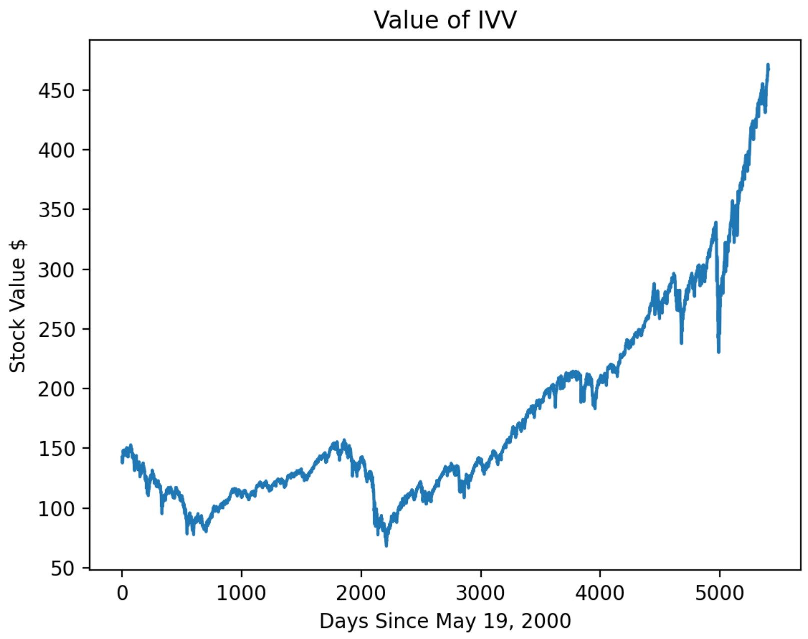

Figure 1: value.png

Score determination

I used `svm.SVC.score()` to score my model. An f1 score did not seem like an important feature for this dataset since mispredicted values are equally detrimental to the performance of the model. In other words, the model will perform best by minimizing misclassified samples, regardless of class.

K-Fold Cross Validation

My kfold CV algorithm is implemented by splitting the training dataset k parts with `sklearn.train_test_split()`. I appended them into a list and used a for loop to successively train on all the folds except for the ith fold. I evaluated the module on the ith fold and averaged its score with that of the rest of the folds.

Grid Search

I used for loops to do my grid. I searched through `C=[0.001, 0.01, 0.1, 1, 10, 100, 1000]`, `kernel=["linear", "poly", "rbf", "sigmoid"]`, `degree=[1, 2, 3, 4, 5, 6, 7]`. I found that models perform with 0.53 accuracy on average, and only linear models ever perform significantly better than 0.53 on the training set. The C value does not seem to make a significant difference in performance as long as C >= 1.

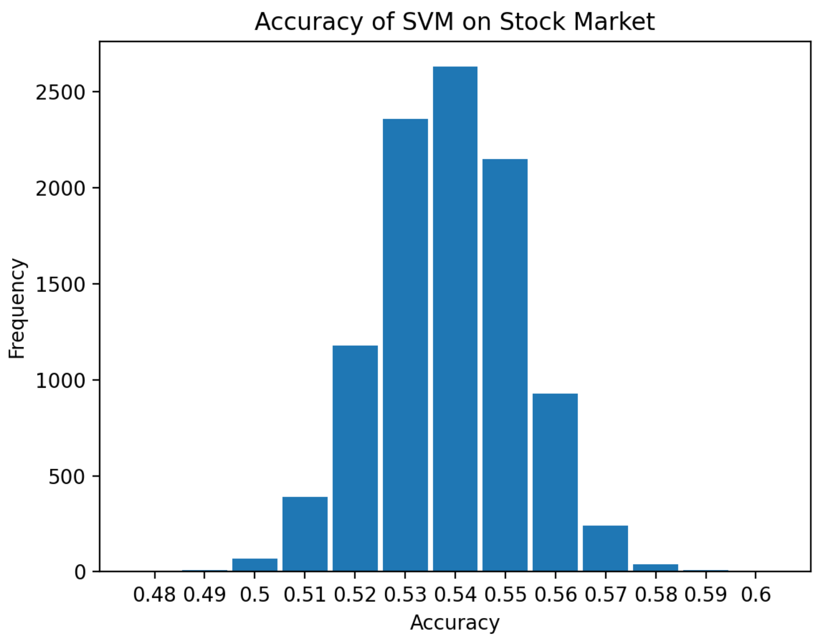

Figure 2: accuracy.png

Results

Over 10 thousand trials on the test set, the models has an expected accuracy of 0.5387 and variance of 0.0002.

I developed a simulation where the program:

- buys a stock at the `Open` value and sells at the `Close` on days where the value is predicted to rise
- shorts the stock at the `Open` value and buys it back at the `Close` value on days where the value is predicted to decrease.

```
net_worth = sum([val * a for val, a in zip(difference, action)]))
```

Over 10 thousand iterations of the simulation, the expected gain of the user is 21.5$ over the span of 21 years. If the user ignores the simulation, and holds the stock continuously for 21 years, he earns 326$.

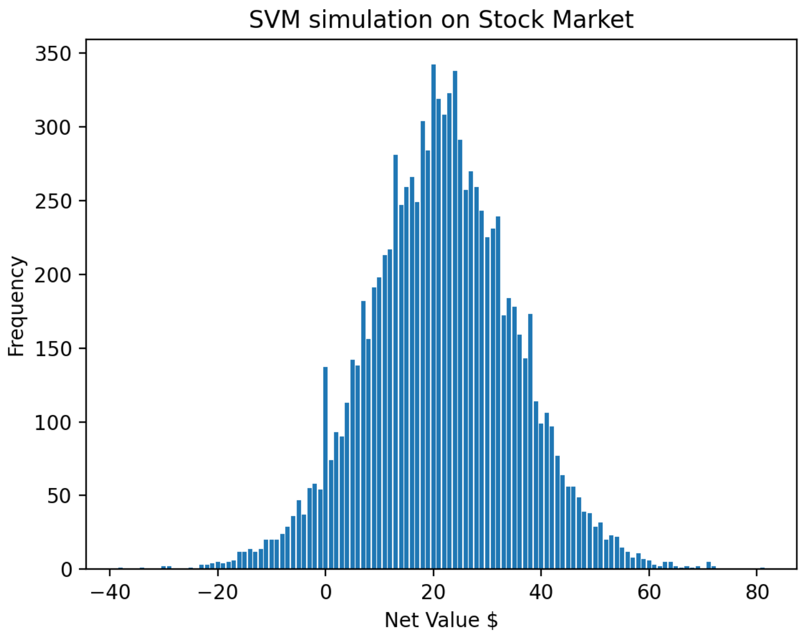

Figure 3: money.png

Concluding Thoughts

It is inherently difficult to predict the direction of a future gradient in financial data since it is largely impacted by events that occur outside of the dataset. I hope to use an RNN, LSTM or traditional DL model to achieve better performance on a similar dataset in my final project. I believe that in collecting more data, One may be able to make more accurate predictions. I intend to spend more time preprocessing and understanding the data in the future in order to minimize the analysis of irrelevant data.