Using AI to Make Predictions and Decisions on Stock Market
Jack Jin; Alice Zheng

Project Introduction
The problem we want to tackle in the project is to create an automated tool for managing investments in the stock market with a limited amount of selected stocks. Our model comprises of two separate parts. The first part takes in the stock prices in the past and makes a prediction of the future prices, which is then fed into the second part to be used for making decisions on which stocks to buy/sell, and when to buy/sell the shares we hold.

Predictions based on Technical Indicators
Technical indicators are models of processing time series data. After reading several investment education websites, we picked the following 11 most popular technical indicators to work with.

Predictions based on Time Series Data
We started with a simplified problem: predicting whether the prices will increase or decrease in the next n days, using the stock prices and volumes in the past m days. For each stock, we used Alpha Vantage API to access the daily open price, daily high price, daily low price, daily close price and the daily volume from the beginning of 2000.

<table>
<thead>
<tr>
<th>(m, n)</th>
<th>(20, 5)</th>
<th>(20, 3)</th>
<th>(10, 3)</th>
<th>(10, 1)</th>
<th>(5, 1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logistic Regression</td>
<td>51.20%</td>
<td>51.04%</td>
<td>52.37%</td>
<td>47.97%</td>
<td>48.08%</td>
</tr>
<tr>
<td>Bayesian Network</td>
<td>50.84%</td>
<td>50.97%</td>
<td>48.62%</td>
<td>47.09%</td>
<td>46.87%</td>
</tr>
<tr>
<td>Simple Neural Network</td>
<td>47.06%</td>
<td>45.93%</td>
<td>44.66%</td>
<td>42.14%</td>
<td>42.85%</td>
</tr>
<tr>
<td>SVM with rbf kernel</td>
<td>46.22%</td>
<td>44.79%</td>
<td>43.38%</td>
<td>41.83%</td>
<td>41.01%</td>
</tr>
</tbody>
</table>

Table 1. Test error rates for different models on Time Series Data
The high error rate suggests that the underlying process is not well represented by the models we chose, or our choice of predictors is not good.

Making decisions
Our decision part is defined as a search problem with the future stock prices known, and we implemented it as a Markov decision process with the actions only comprising of selling all the stocks we currently hold and investing all the money on one specific stock for each timestep. For each timestep, we look ahead at the next 20 timesteps and find the best policy to achieve the best result in 20 timesteps, and we use the action of the acquired policy in the first timestep as the action of our current timestep. This theoretically should provide a nearly optimal solution if we are sure that the predicted future prices are exact.

Prediction Model and Results
In order to run our decision machine, we need a prediction of the specific price value, so we switch to a regression problem. The response variable we target is the difference between the prices over a n-day period, and our predictors include the stock price at the end of the day at the beginning of the period, the volume of the same day, and the 11 indicators that we chose in the previous section.

We collected a total of 260k data samples from 82 stocks (time series data) we partition the data into training set, dev set, and test set in a ratio of 7:2:1. We used two metrics for evaluating different regression models: the first one is the classification of the price trend (increase/decrease), and the second metric is the mean squared error. After evaluating different models on the data set, we see that support vector regression gives the best result. However, it is also extremely costly to compute, so we only train the model on a subset of all our data. For processing more data, we see linear regression is the best tool we have. By comparing the training MSE and the test MSE, we see that there is not much over-fitting in the model.

To the right is a figure of the correct classification rate versus n for linear regression, plain and with 2 types of regularization (ridge and lasso). We see that as n increases, the correct rate drops, suggesting that time series data do not provide us the required information to make good predictions for over a very long time.

To the left is a figure of the actual difference versus the predicted difference for one of the stocks we reserved as part of the test set. Even though we see that our best prediction correct rate is lower than 70%, we see that we tend to have a much higher prediction correct rate for prices changes that are "bigger". For example, we can achieve a 92.3% correct rate for the trend when the price increases for more than $2, and a correct rate of 72.9% when decreases by more than $2

NOTE: When we ran support vector regression with rbf kernel on 10% of the training data, we were able to achieve a correct rate of 68% on test data. However, the computation is quadratic to the sample size when the sample is small, and it becomes near exponential when the cache is incapable of holding all required data, we hope to find more powerful computation resources and finish SVR before the final report.

References