# AI Project: Stock Prediction Using Machine Learning

016072910024 (Shansi Dong)董善思 henrich0122@protonmail.com

517021910612(Yuanhao Feng)冯元昊 fyhisme@sjtu.edu.cn

# 1. Introduction

In this project, we mainly handled several actual business problems about financial transactions. There are a lot of factors we should take into consideration when we try to forecast stock prices such as policy, enterprise revenue and so on. In this case, machine learning would be an effective method to predict stock price change. And in this project, we are supposed to implement supervised learning, unsupervised learning and reinforcement learning to forecast the stock price. In task one, our main goal is to learn a certain model and do regression according to the features and labels in the dataset. And in task two, we mainly add generated features to the model of task one and test whether the model performance is improved in the testing set. All the implementation performs well.

# 2. Experiments

## 2.1 Data Reconstruction and Cleaning

### 2.1.1 Configuration

We implement data reconstruction and clean in windows 10 system using python3.7.0 and the main library is pandas.

### 2.1.2 Analysis

In this part, we should reconstruct and clean the raw data for our following parts. Since all the indicators have been provided properly, what we should do is calculate labels and clean the data we do not need. We choose the midprice change after 5 ticks, 10 ticks,20 ticks and 50 ticks as our labels and abandon all the other raw data.

At the very start, we misunderstood the meaning of tick. We equate “tick” to “second” which would make this problem quite confusing as there are no date stamps in the data set so we cannot figure out the time difference. After searching some referenced material, we realize that tick is a snapshot of stock status and each item in the data set represents one tick, which makes us figure out our labels easily.

### 2.1.3 Code

(Part of critical code as following)

```
rows, cols = data.shape

data.drop(data.columns[109:], axis=1, inplace=True)

midPrice_plus_5 = []
midPrice_plus_10 = []
midPrice_plus_20 = []
midPrice_plus_50 = []
for idx in range(rows):
    if idx + 50 < rows:
        midPrice = data.iloc[idx]['midPrice']
        midPrice_plus_5.append(midPrice - data.iloc[idx+5]['midPrice'])
        midPrice_plus_10.append(midPrice - data.iloc[idx+10]['midPrice'])
        midPrice_plus_20.append(midPrice - data.iloc[idx+20]['midPrice'])
        midPrice_plus_50.append(midPrice - data.iloc[idx+50]['midPrice'])

data.drop(list(range(rows-50, rows)), axis=0, inplace=True)
data.drop(['midPrice'], axis=1, inplace=True)

data['midPrice_plus_5'] = midPrice_plus_5
data['midPrice_plus_10'] = midPrice_plus_10
data['midPrice_plus_20'] = midPrice_plus_20
data['midPrice_plus_50'] = midPrice_plus_50
```

### 2.1.4 Evaluation

As a result, we get a suitable dataset for the learning of task one and task and lay a solid foundation for the following part.

## 2.2 Task One

### 2.2.1 Configuration

We implement this part in windows 10 system using python3.7.0 and the main library is numpy, pandas, matplotlib and sklearn .

### 2.2.2 Analysis

In this part, we are supposed to learn a certain model and do regression according to the features and labels in the dataset. The features of learning method are indicators 1-108 and the labels are what we have figured out in data reconstruction and clean part.

In the beginning, we try to use KNN regression to train the prediction model. And 108 dimensions are too much for KNN so that we use PCA to reduce dimensions for better performance. However, the result is disastrous. The bias proportion is more than fifty percent which means our prediction is meaningless. After the failure of KNN, we also tried GRBT regression, decision tree regression and so on. Unfortunately, all the methods did not perform well and the least bias proportion is about thirty percent.

Finally, we try linear regression for targeted therapies but in this time it performs well in test set. We summarize that the problem may arise in PCA process. We use PCA process to reduce dimensions except for linear regression because the model of linear regression is simple enough so that the dimension could be more complex. And the PCA process may indistinct the feature of data, as a result, the ability of

expression of the model may be weak. And in linear regression, though higher dimension could increase complexity, all the features of data would be expressed in the model so the result is better.

### 2.2.3 Code

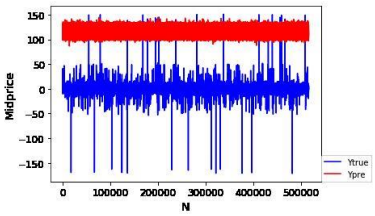
(Part of critical code as following)

```
y_train=np.array(y_train)
y_test=np.array(y_test)
lab_enc = preprocessing.LabelEncoder()
y_train = lab_enc.fit_transform(y_train)
```

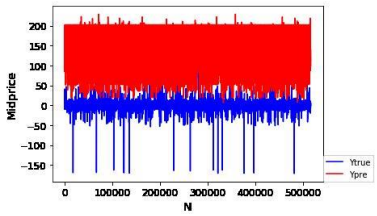
```
from sklearn.linear_model import LinearRegression
linreg = LinearRegression()
model=linreg.fit(X_train1, y_train)
#print(model)
print(linreg.intercept_)
print(linreg.coef_)
```

### 2.2.4 Evaluation

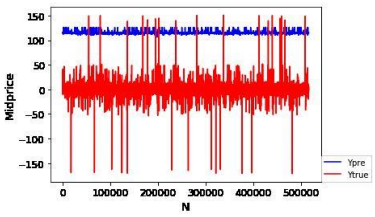
Here are some terrible results we got by KNN, DT and GRBT.



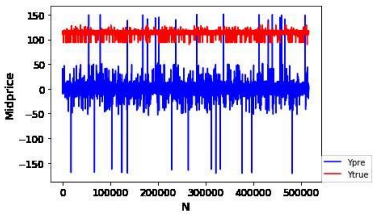

*KNN*




*Decision Tree*

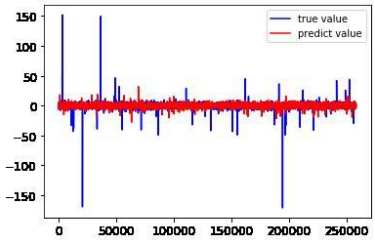



*Logistic Regression*




*GRBT*

And here are the final result using linear regression.




*Linear Regression For 5 ticks*

## 2.3 Task Two

### 2.3.1 Configuration

We implement this part in windows 10 system using python3.7.0 and the main libraries are numpy, pandas, matplotlib and sklearn .

### 2.3.2 Analysis

In this part, we mainly add generated features to the model of task one and test whether the model performance is improved in the testing set. We choose several ways including ICA dimension reduction, con features and t-SNE dimension reduction to reduce dimensions and create new features.

### 2.3.3 Code

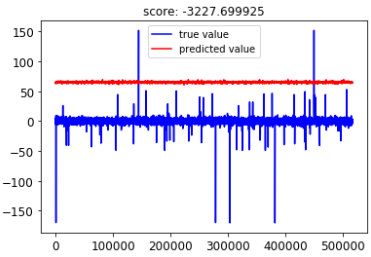
(Part of critical code as following)

```
y_train=np.array(y_train)
y_test=np.array(y_test)
lab_enc = preprocessing.LabelEncoder()
y_train = lab_enc.fit_transform(y_train)
```

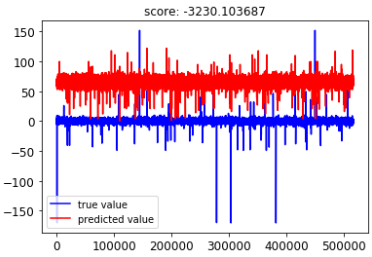
```
from sklearn.manifold import TSNE
X_train2 = TSNE(n_components=2, n_iter=300).fit_transform(X_train1)
X_test2 = TSNE(n_components=2, n_iter=300).fit_transform(X_test1)
```

```
from sklearn.decomposition import FastICA
ICA = FastICA(n_components=10, random_state=0)
X_train2=ICA.fit_transform(X_train1)
X_test2=ICA.fit_transform(X_test1)
```

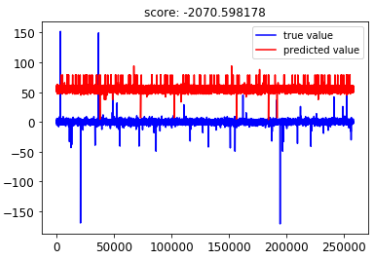
### 2.3.4 Evaluation

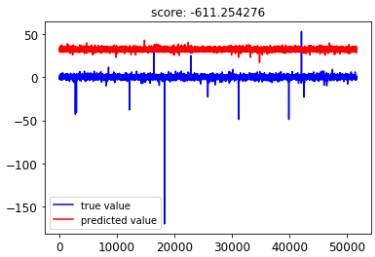



*con features + Linear Regression*




*con features+ Decision Tree Regressor:*




*（ICA）dimension reduction + Extra Tree Regressor*　　*t-SNE dimension reduction + Bagging Regressor*

As we can see in the figure, all of them do not perform as well as linear regression without dimension reduction. In our opinion, the reason may be that the generation of new features may indistinct the structure and trait of original data. As the result, all the method performance lagged behind.

# 3. Conclusion

## 3.1 Problems

Frankly speaking, we met a lot of problems with this project. However, what makes things better was that, once we got the project handout, we began our work immediately so that we had enough time to solve all the problems by acting with united strength. In the following part, we would expatiate these problems specifically.

Firstly, we had a problem with project content understanding. Our group mates both do not have financial background, as a result, we spend a lot of time learning related financial knowledge and we also make some mistakes on basic stock concepts that make our project once go astray. What makes things better is that after searching for some referenced materials, we finally handle all these problems.

We also had a lot of other problems during the learning process. For example, we used PCA to reduce dimensions which makes our model underfitting. There are amount of similar problems that occurred during the learning process. What's worse, these problems are extremely time-consuming. Fortunately, after a long time suffering, we solved all the problems via searching related documents, asking TA for help and many other ways.

## 3.2 Achievements

In this project, we've successfully overcome all kinds of problems we encountered to satisfy all the requirements and get a relatively good result.

We've worked out a functionally complete code, but we know it still has a fairly great room of improvement but isn't further optimized due to the time limit.

Through this project, we mastered several machine learning methods and

get a basic understanding of artificial intelligence. The timely help from our teacher and TA makes the whole process is substantial and pleasant. As a whole, we've progressed successfully and learned a lot. Heartfelt thanks to the guidance of our teacher and assistants!

## 3.3 Team Contribution

In our team, Dong mainly takes responsibility of writing machine learning codes while Feng mainly takes responsibility of data processing and report writing. Both of us contribute to learning model optimization and parameter adjustion.