Stock Price Trend Prediction Using Machine Learning Through Python

*Gao Yi (5555 5912), Jiang Feng (5537 1545), Lin Junqi (55577690) , Yang Zhe (5555 9498)*

Using machine learning, we try to predict the future stock price trend and the results seem good.

### Introduction

We mainly conduct our project by proceeding the following four parts: data processing, LSTM model building, training and testing, and data visualization.

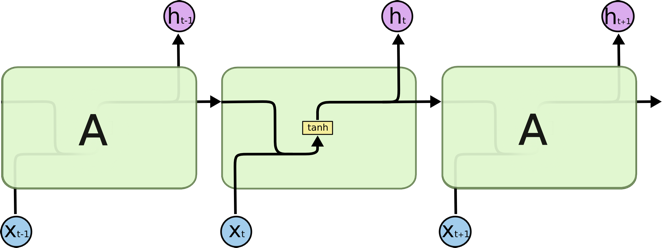
We import TensorFlow to help us .

TensorFlow is an end-to-end open source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries and community resources. We choose TensorFlow because it is easy for us to build a model and we can learn the advanced technique in the process.

LSTM is another vital tool.

LSTM means long short term memory. The key to LSTMs is the cell state, the horizontal line running through the top of the diagram. The cell state is kind of like a conveyor belt. It runs straight down the entire chain, with only some minor linear interactions. It’ s very easy for information to just flow along it unchanged.The LSTM does have the ability to remove or add information to the cell state, carefully regulated by structures called gates. Gates are a way to optionally let information through. They are composed out of a sigmoid neural net layer and a pointwise multiplication operation.

A screen shot of a clock

Description automatically generated

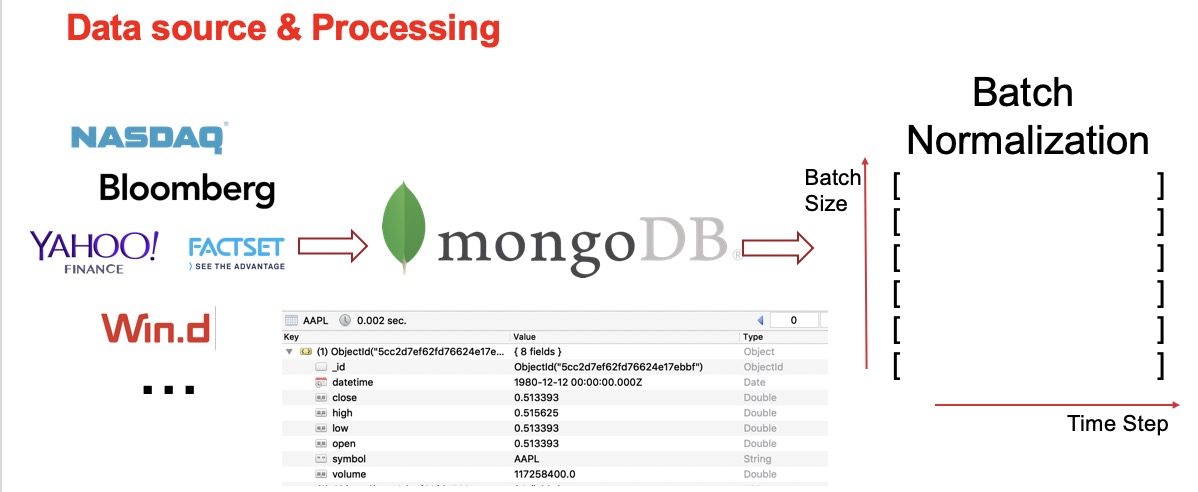
### Python Implement

We import four packages: TensorFlow, NumPy, Pandas, and plotly.

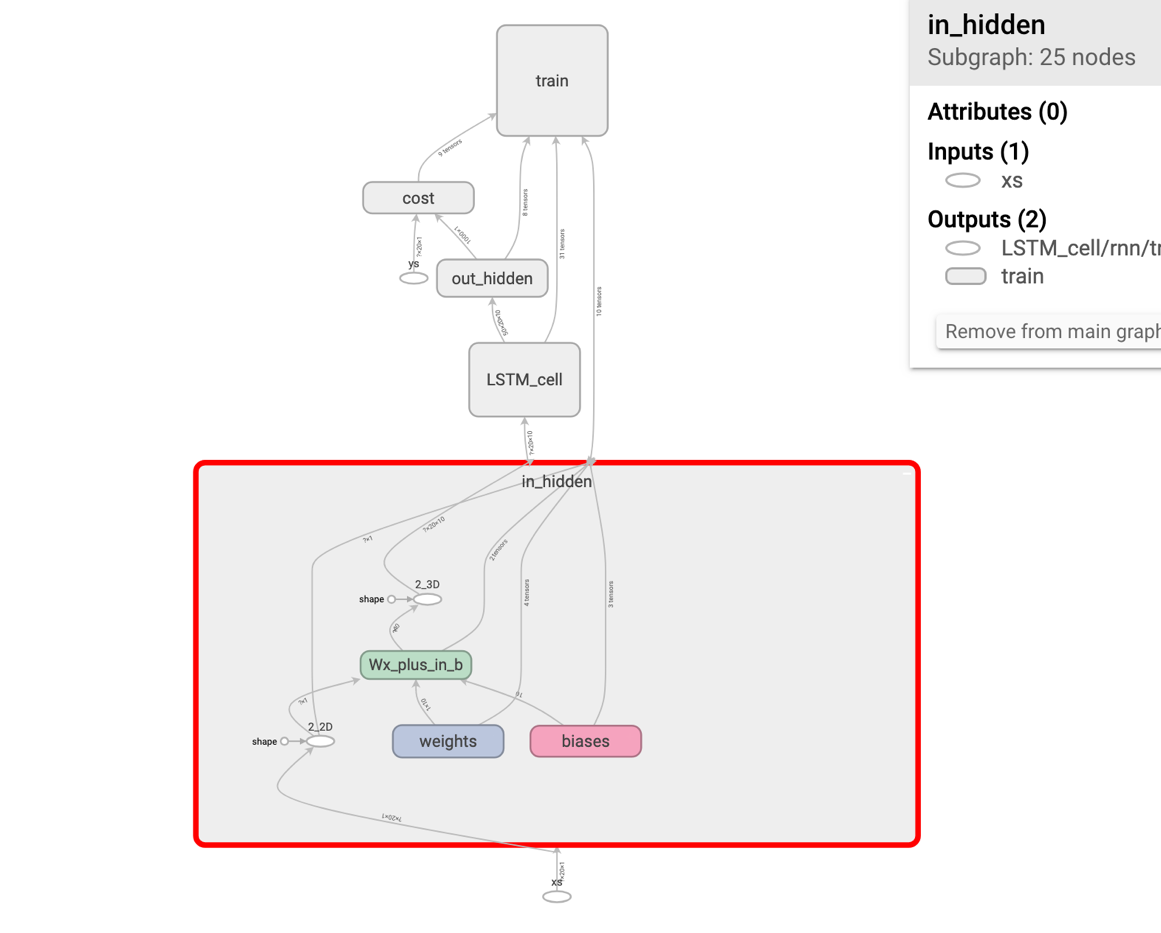
Considering that we may have many kinds of data sources which have different formats, so we developed a data process class to deal with all data sources such as Bloomberg, yahoo finance and so on. Then we stored data in a standard format into MongoDB. The reason we do that because in this way, we don't need to modify our code according different data sources, which's more convenient.

Then we developed a Data\_Generator class to make data got by database into data needed by model learning. The core function is get\_batch(steps), we can get this kind of data. the data shape is batch\_size\* time\_steps .

Tensorflow's lstm cell can read data by one batch instead of one sequence, it's more efficient. At time\_step 1, model read 1 and to predict 2, at time\_step 2, model read 2 and to predict 3 and meanwhile it keeps the effect from time\_step1, this's is how long and short memory works.



At the beginning is our data has been dealt with Data\_generator, its shape is (batch\_size,time\_step,1), one means one factor. Here our factor is the close price of stock. Then our data will go through input hidden layer, then it enters the LSTM\_cell. In the end, it will get the result going out of output hidden layer.



TIME\_STEPS = 20

BATCH\_SIZE = 50

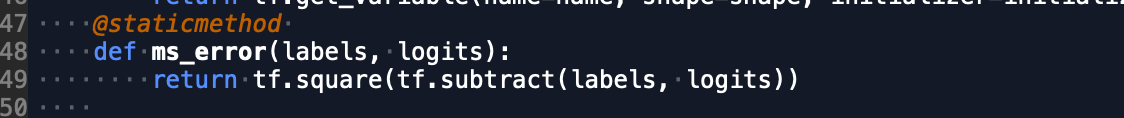
CELL\_SIZE = 10

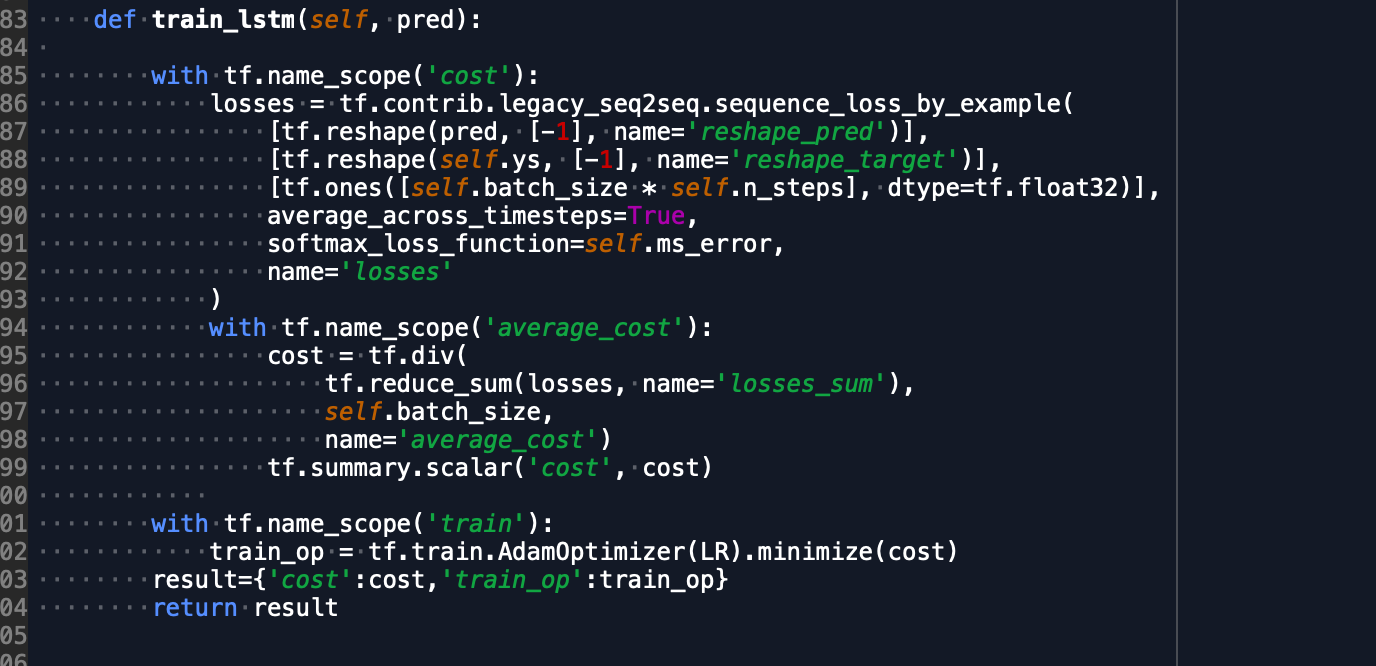
SPLIT = 0.8

LR = 0.006

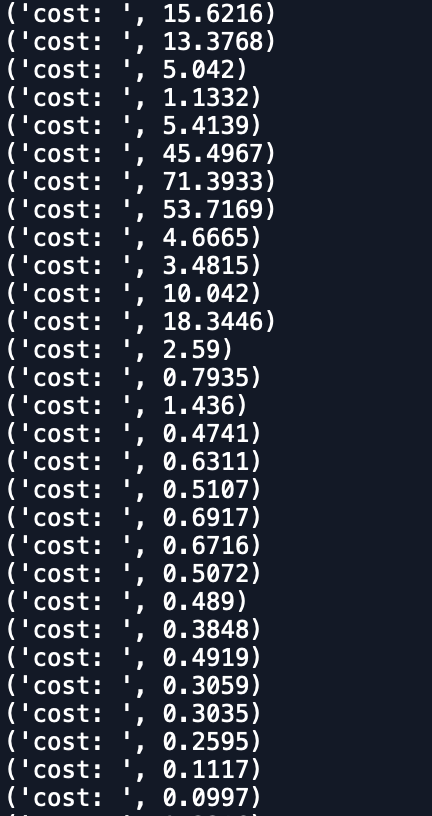
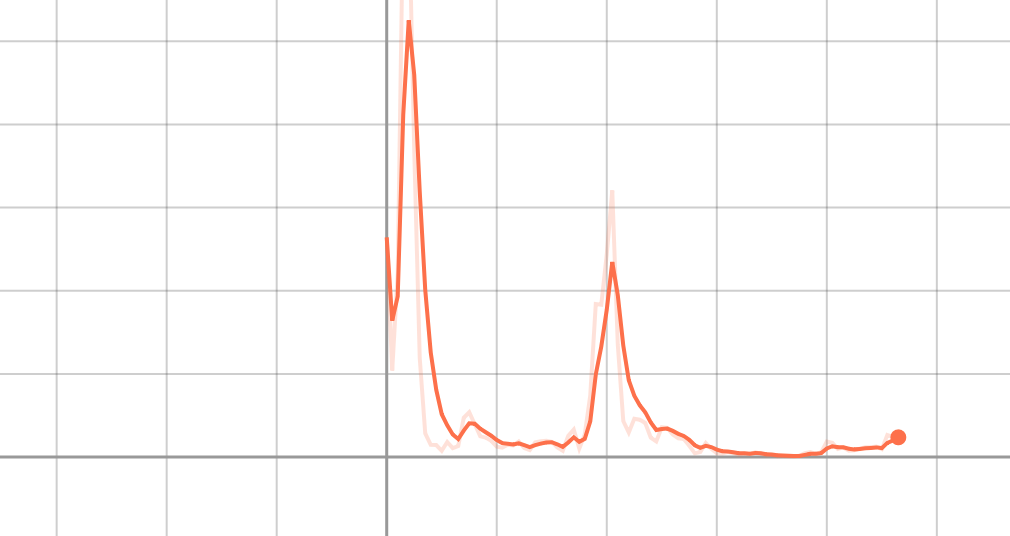
DB\_NAME = "SP500\_TOP10"

Cell\_size is the numbers of cells in each lstm cell. LR is the learning rate which work for model optimizer, we can choose it, but not too large, by several modification. 0.006 can get pretty good result. Split is the coefficient which cut data into train set and test set. Here we 80 percent of data is train set and 20 percent of data is test set. DB\_NAME is our database name.



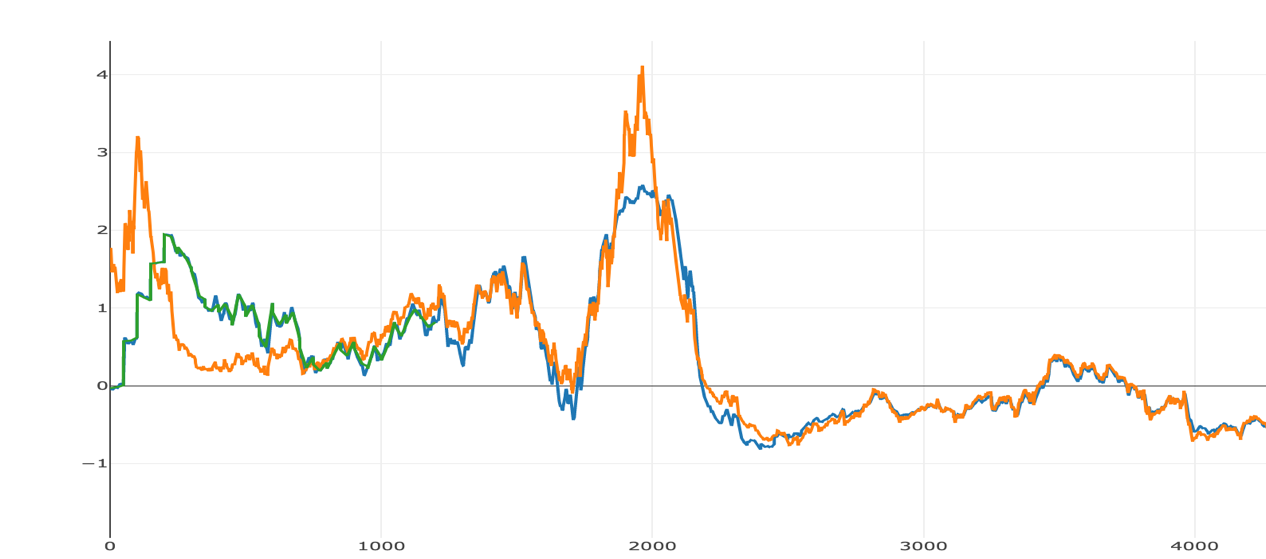


First we make the loss function: mean-square error, it can calculate the error between real data and prediction data. Then we use the AdamOptimizer() to optimize the coefficient of model. Then we can find in each iteration, the error decline, it means that the model is learning something

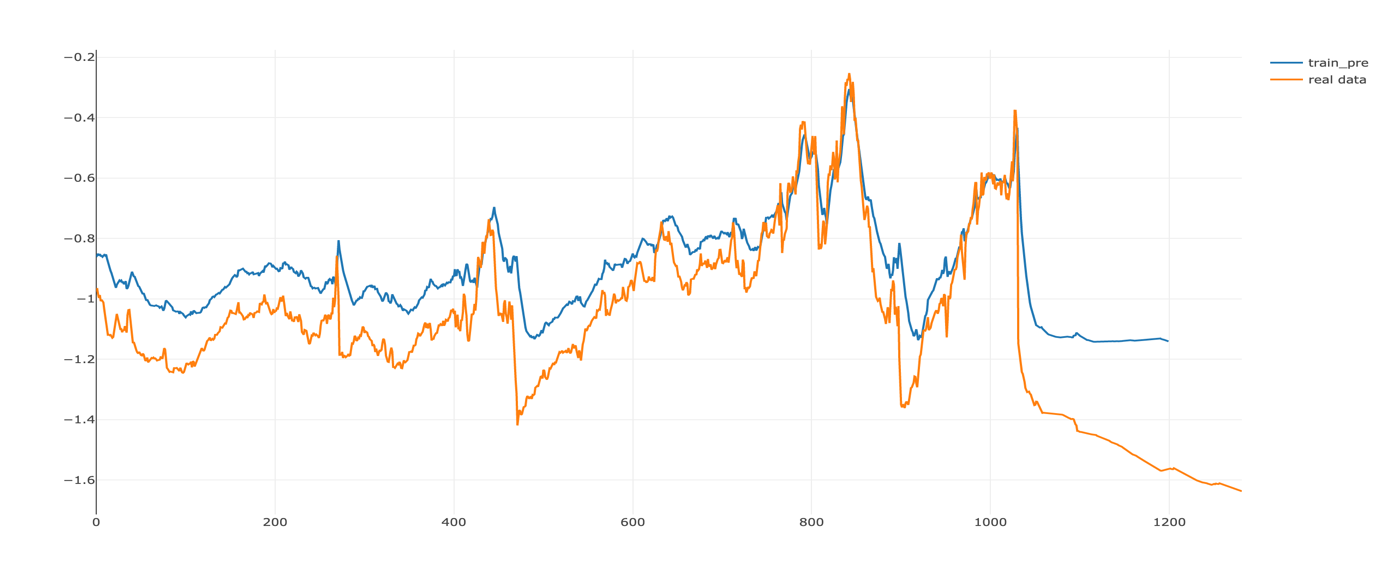


The cost keeps reducing.

### Output



This is our training set result. At the beginning, we can see our prediction curve cannot fit the real data well. But after many times training, we can see the prediction show good result.



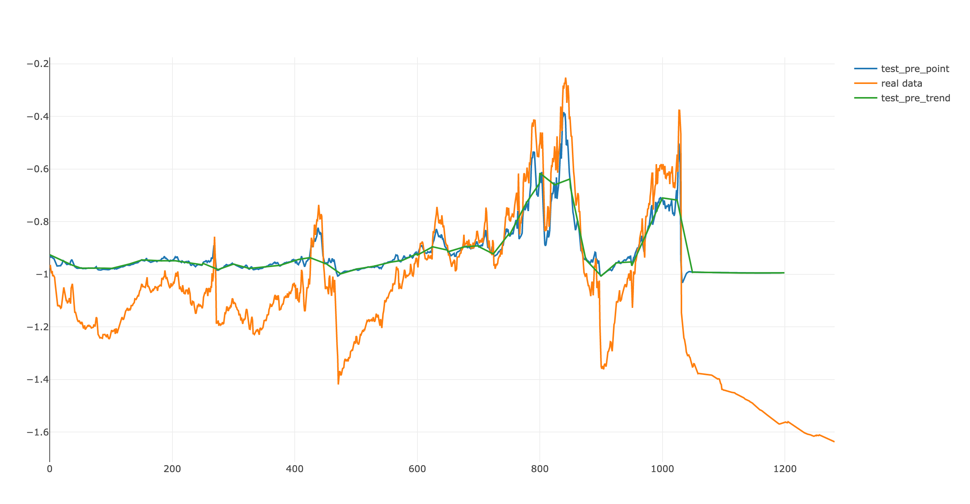
This is our testing set result. Also, the prediction can fit real data curve very well.

### Discussion

There are two points that we can improve.

The first is our data richness and effectiveness. For Apple, the 6,000 data from the beginning is too smooth on the chart, they will be filtered by the filter. The remaining 3,000 to 4,000 trading days of data is enough for us to get a good fit, but for Facebook it is different. He started to market in 2012, with less than 2,000 data, but we Try to introduce day trading data to increase the amount of data, but fluctuations in day trading can lead to over-fitting problems

Secondly, if we want to refer to our results in real stock operations, we should focus on the forecasting trend instead of the precise forecasting point, although there will be some difference between the predicted value and the actual value at a specific point. , but the price trend is a good reference for us in the actual operation.



When we want to go a step further and make predictions over longer intervals, we find that the predictions are not as good as our previous predictions based on one-day intervals. For example, the two are the test results for the 10-day interval and the 50-day interval, which is still somewhat worse than the fit of the first result.

