

CS 450 - Machine Learning & Data Mining

Name: Kwok Moon Ho | Github: <https://github.com/kwokmoonho/Machine-Learning-SP500>

Introduction:

The objective of this project is aimed at using python and different machine learning algorithms(LSTM, KNN, Linear Regression) to conduct predictions on the S&P500 index. This problem is interesting because it can compare the performance of different machine learning algorithms when it applies to the prediction of the S&P500 index. At the end of the project, it should be able to conclude which algorithms are better for the prediction of the S&P500 index. All data are obtained from [Yahoo Finance S&P500 index](#). There are 8850 instances in the data set from the period 2/1/1985 to 3/13/2020. The reason this period is chosen because it contains at least three [economic recession](#) in USA history. The data type of this data only contains Date, Numeric data type.

The project experienced three different machine learning algorithms which are LSTM, KNN, and linear regression. LSTM obviously has a better result than linear regression, and linear regression has a better result than KNN.

Data preparation:

Since stock data does not contain missing value and categorical data, therefore, in the data preprocessing process, this project does not implement labels or one-hot encoding or handling missing value. This project applied feature scaling - "[Scaling features to a range](#)". The purpose of this is to standardize the data so that the maximum absolute value of each feature is scaled to unit size. This is achieved by using '[MinMaxScaler](#)'. The motivation to use this scaling include robustness to very small standard deviations of features and preserving zero entries in sparse data.

```
scaler = MinMaxScaler(feature_range=(0, 1))
scaled_data = scaler.fit_transform(dataset)
```

Mining / Learning from the data:

In this project, three machine learning algorithms are used which included: Linear regression, KNN, and Long short-term memory (LSTM). The process of mining the data for these three different algorithms are: Data Preprocessing → Data Feature Scaling → Splitting 80% of data to train set and 20% to test set.

Linear Regression:

```
#implement linear regression
model = LinearRegression()
```

```
model.fit(x_train,y_train)
```

The code above creates a linear regression model and fits in the x_train and y_train dataset. The root means square (rms) is 727.06. This shows that this is not a good fit for using linear regression to predict the S&P500 index.

KNN:

```
#using gridsearch to find the best parameter
params = {'n_neighbors':[2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20]}
knn = neighbors.KNeighborsRegressor()
model = GridSearchCV(knn, params, cv=5)
.....
```

In order to find the best k for KNN algorithms, the code above used [GridSearchCV](#) to find the best parameter of k from 2 to 20. Then the x_train and y_train data will fit in the KNN model. The root means square (RMS) is 1478.34, which shows that this is not a good fit for using KNN regression to predict the S&P 500 index.

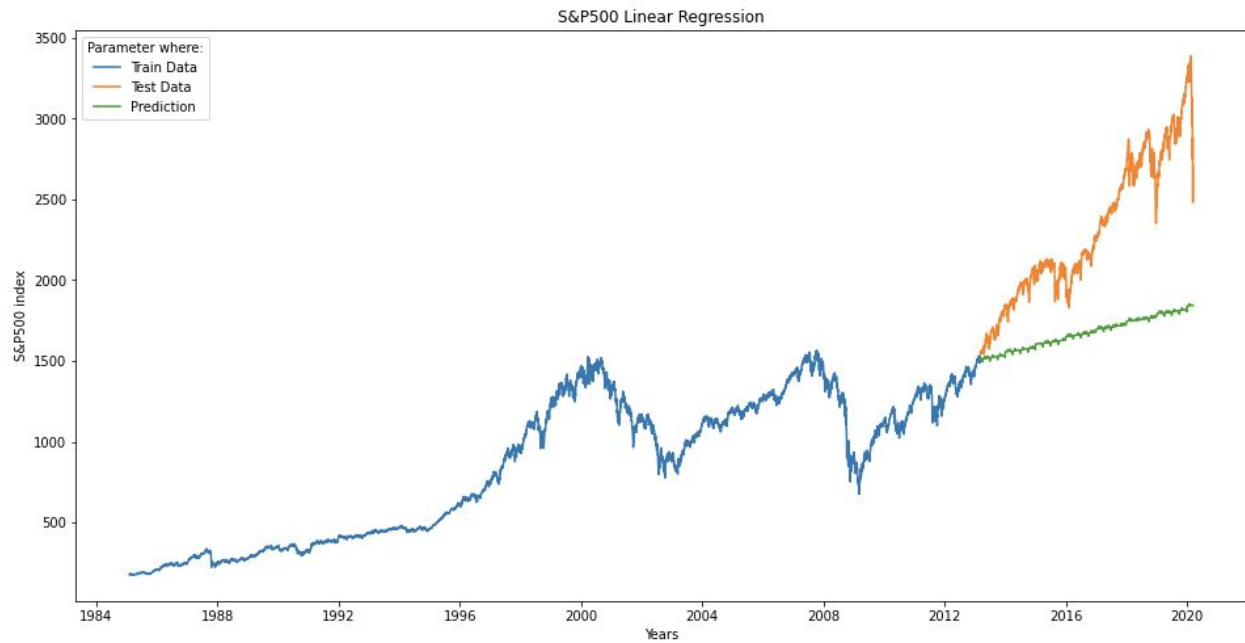
Long short-term memory (LSTM):

```
for i in range(60,len(train)):
    x_train.append(scaled_data[i-60:i,0])
.....
# create and fit the LSTM network
myRMS = []
for p in range (40,60):
    model.add(LSTM(units=p, return_sequences=True, input_shape=(x_train.shape[1],1)))
    model.compile(loss='mean_squared_error', optimizer='adam')
    model.fit(x_train, y_train, epochs=10, batch_size=32, verbose=2)
```

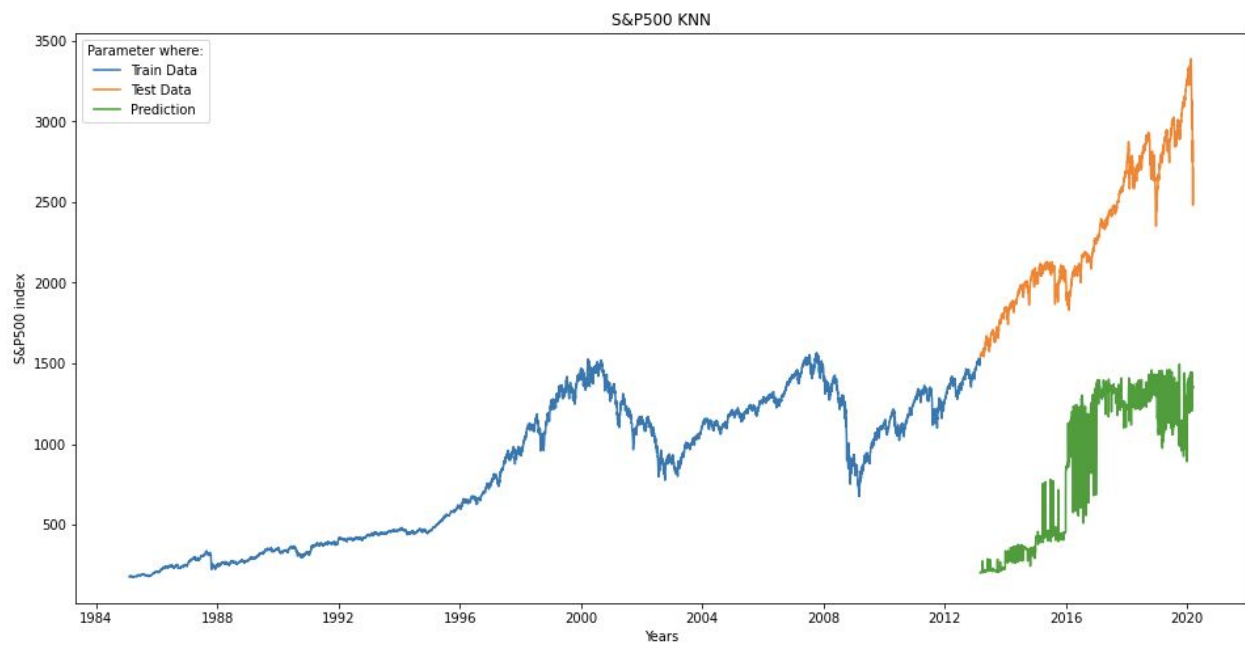
I used the past 60 days from the train data to calculate the predicted value. Therefore, when the data is split into x_train and y_train, the for loop start from 60 to the length of the train data set. In the LSTM function, the unit means the dimensionality of the output space. In this case, units are set into different values inside a “for loop” from 40 to 60, which means that there will be 40 to 60 hidden layers each loop. Since it is a regression problem, this LSTM model implemented the [Mean Squared Error Loss](#), which calculated as the average of the squared differences between the predicted and actual values. Epochs are the number of epochs to train the model. An epoch is an iteration over the entire x_train and y_train. The variable “batch_size” means that the number of samples per gradient update, here this LSTM model uses the default value 32. The variable “verbose” is 2 in this model means that one line per epoch. This LSTM tried different numbers of units, epochs, and batch_size. The challenge for mining or learning from the data is that it takes a lot of time to understand the parameter of the function. I overcome these challenges by looking up into the Kears library document.

Results:

Linear Regression:



KNN:



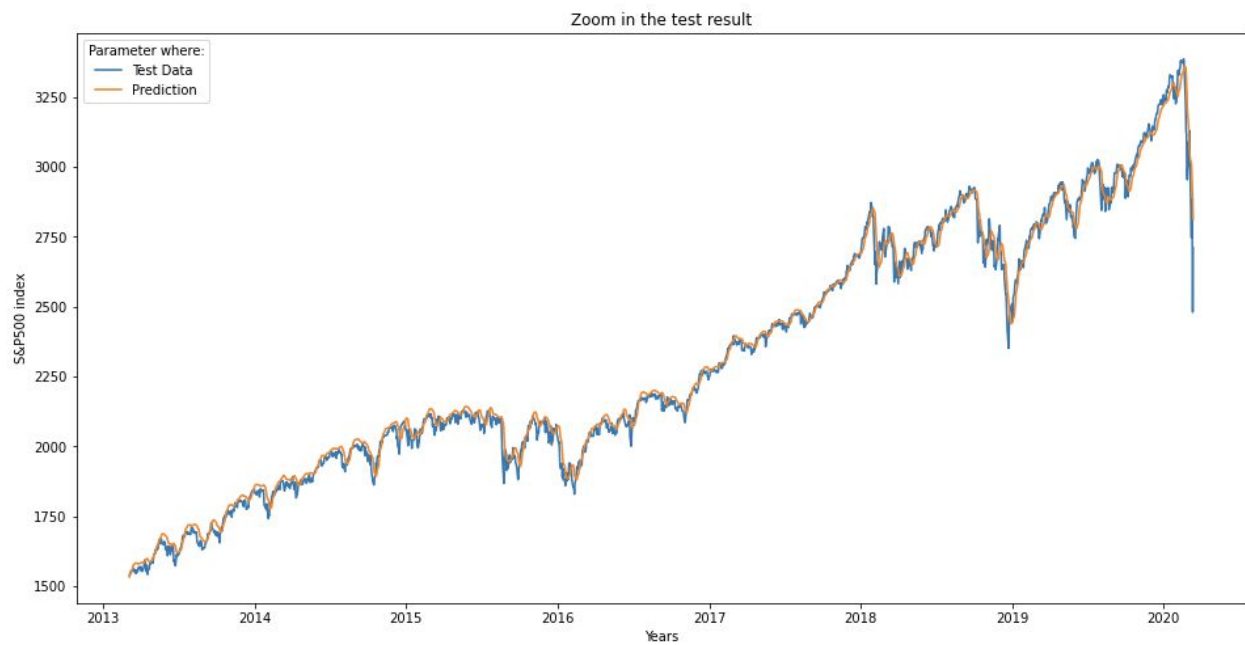
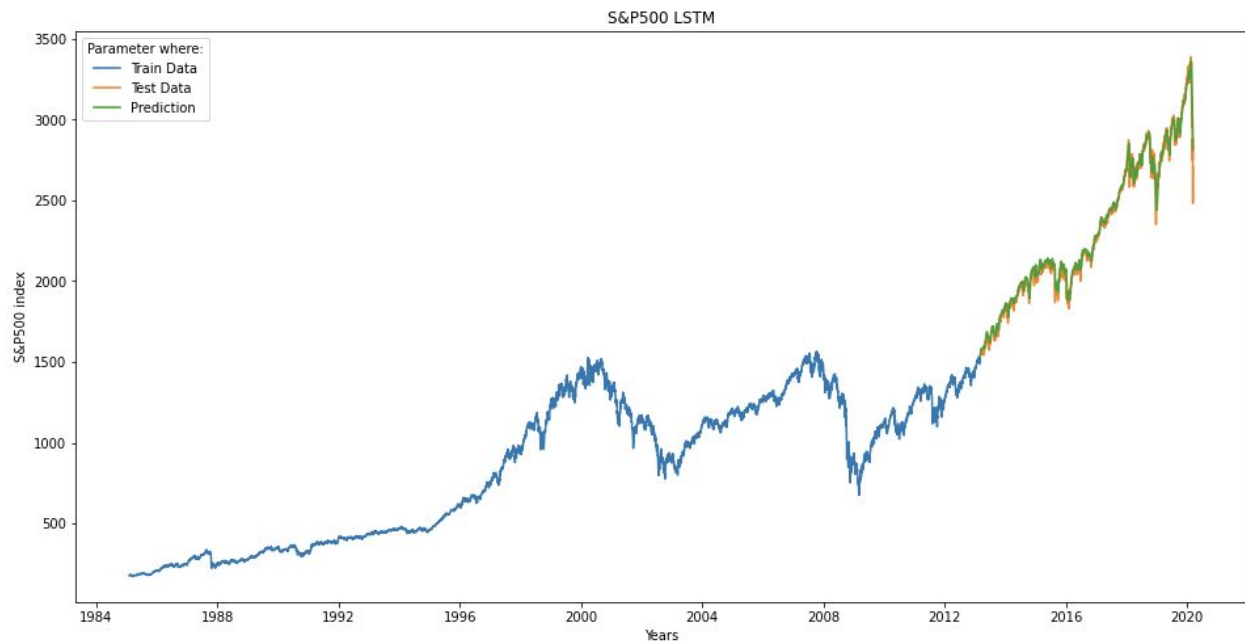
LSTM:

The best result for these variables are units = 57, batch_size = 32, epochs = 10.

```

Dimensionality of the output space for different units values:
units = 40 , rms = 65.37063190091425
units = 41 , rms = 45.89844408012986
units = 42 , rms = 99.70129572746092
units = 43 , rms = 91.17935637518467
units = 44 , rms = 49.89241133645945
units = 45 , rms = 42.66220668280837
units = 46 , rms = 50.128990450202146
units = 47 , rms = 50.95898662952903
units = 48 , rms = 72.05277381527566
units = 49 , rms = 48.408390041762374
units = 50 , rms = 84.87201592263493
units = 51 , rms = 72.5394931071893
units = 52 , rms = 63.56218344248112
units = 53 , rms = 63.06455622111648
units = 54 , rms = 58.643816084347876
units = 55 , rms = 57.950913983878515
units = 56 , rms = 58.79153934974726
units = 57 , rms = 37.21806176578225
units = 58 , rms = 92.66046336243197
units = 59 , rms = 107.44299843097981

```



Conclusions:

Values and outcome of the project:

The best-fit machine learning algorithms for prediction S&P500 index is LSTM > Linear Regression > KNN. Investors or business may consider to include LSTM as one of the trend indicators to help them to make investment judgment or include in their investment strategy plan.

Limitation:

Since the S&P500 index is an integrated index by 500 stocks value, it may not really reflect the limitation of LSTM when predicting non-numerical issues. In another word, the stock price is affected by the news about the company and other factors like demonetization or merger/demerger of companies. There are certain intangible factors as well which can often be impossible to predict beforehand.

Lessons Learned:

I learned how to implemented the concept and function of linear regression, KNN, and LSTM in python to conduct a machine learning prediction. I learned a lot by doing research on the Kears library and look up all the variables and meanings. I will continue my research by conducting “back-testing” to analyze the profit and loss ratio by using LSTM to trade.

If I could start over, I will just implement the LSTM algorithms and try this model in different datasets. Then, I will explore in “machine trading” concept by implementing my model to trade and analyze the profit and loss ratio. Since I do not have any group members, I think I do a great job in this project by myself.