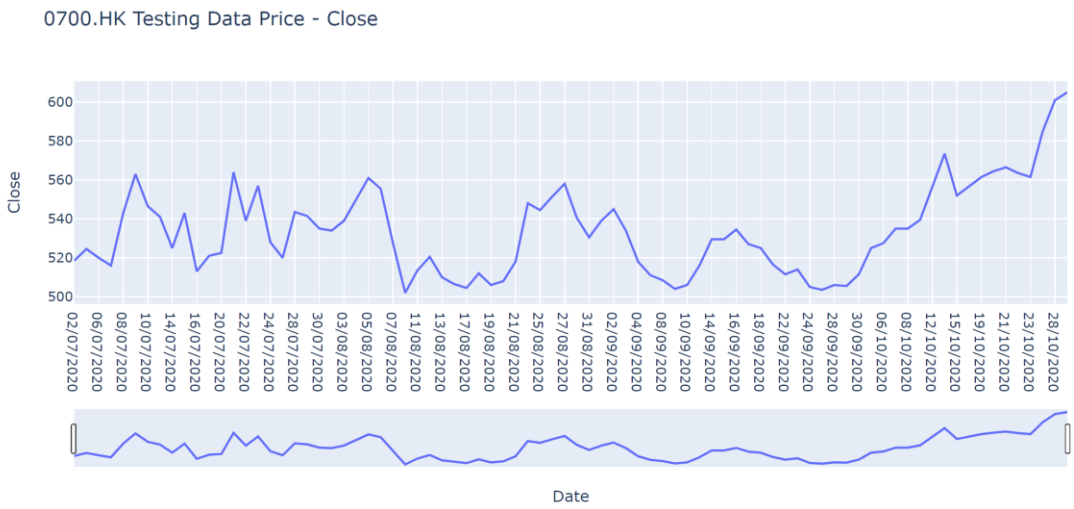
COMP1901 Project 2 --- Research Project: Predicting Stock Prices Report

JIANG Guanlin (21093962D)

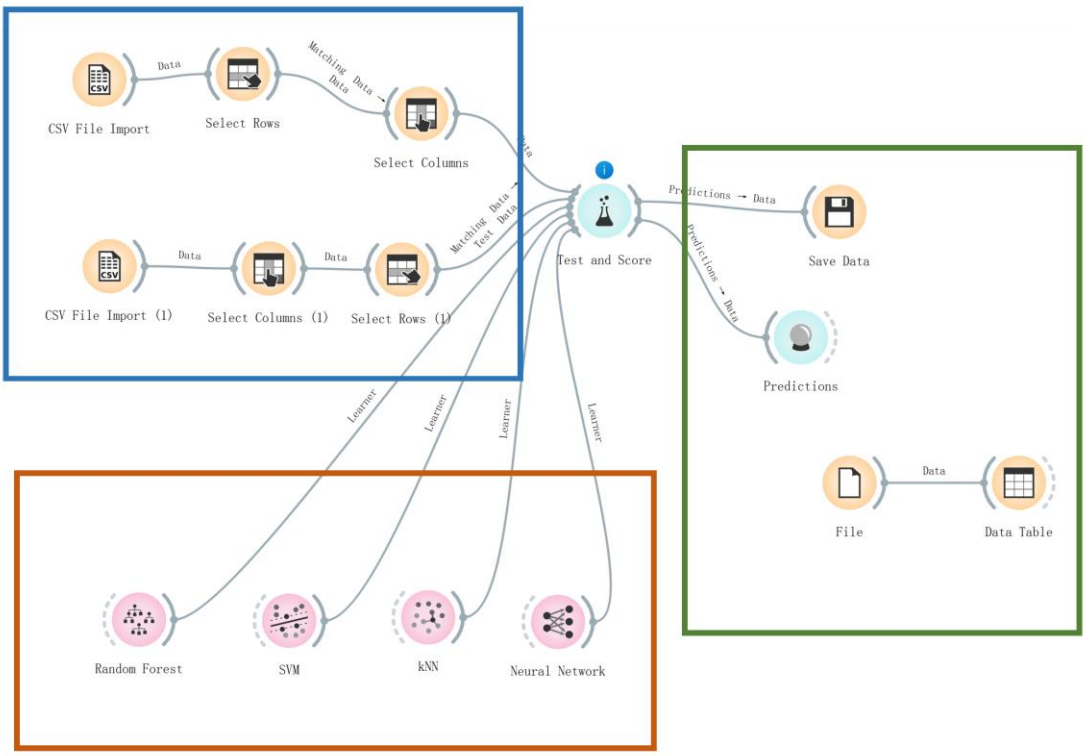
This research project is talking about using Orange Data Mining Tools to train the data using some models to analyze those models, which is more efficient.

Data & Training Preparation

In this project, I will predict stock prices that need using Orange Data Mining Tools, and the data need to use 0700 Tencent Holdings Limited (0700.HK) previous price data to be the predicted stock in this research project. The dataset format I choose to use CSV format file to use because the data have good formatting, so in the dataset preparation, I choose to download the datasets from Yahoo Finance [1], which is the format by CSV file. The time for training data use from January 3rd in 2017 to June 30th in 2020, which file named 0700.HK_training.csv, and the testing data from the range July 2nd in 2020 to October 30th in 2020 which file named 0700.HK_testing.csv. The graphs below are the price trend for training data and testing data:



According to the experiment, I used Orange Data Mining Tools to be the training platform, and I chose to use four training models methods, Random Forest, Neural Network, SVM, kNN model methods to train models by using training data and predict the result compare with testing data. Below is the training process: (Blue: Data Preparation, Orange: Model Training & Testing, Green: Predict & Saving Data)



Data Formatting

In this Orange Data Mining platform, I use three kinds of way to be from the data and let models to training those data. First, I just use the everyday price data to train it. The result show below:

Test and Score Mon Nov 08 21, 15:46:28

Settings

Sampling type: Stratified 5-fold Cross validation

Scores

Model	MSE	RMSE	MAE	R2
kNN	199.4861277153878	14.123955809736442	10.052674362790698	0.9508391021505863
SVM	4082.560156111301	63.894914947210786	50.10813981395349	-0.006096639887769317
Random Forest	1.5508510949706782	1.2453317208562056	0.8371093450601815	0.9996178118592373
Neural Network	19289685.72934866	4392.002473741181	2822.0370671305354	-4752.7053355447615

Second data formatting I use five days average price to train the models, also use next price to predict the result, which calculated by Excel, the data Day_n-1, Day_n-2, Day_n-3, Day_n-4, Day_n-5, and Close to be the Filters, and the Next Price to be the Target, Date to be the meta. Using those data to training, the result shows below:

Test and Score Mon Nov 08 21, 17:42:42

Settings

Sampling type: Stratified 5-fold Cross validation

Scores

Model	MSE	RMSE	MAE	R2
kNN	4.1959400344346705	2.048399383527214	1.579813048895642	-0.17823734205282138
SVM	4.436919315400715	2.10639960961844	1.6531033301514895	-0.24590532233021034
Random Forest	4.284368300465725	2.069871566176444	1.5803284572555933	-0.20306836544107787
Neural Network	3.7879282006907125	1.9462600547436388	1.4714644124614213	-0.06366592907471635

Third data formatting I used calculate the percentage of 5 days average by Excel and list to be the training data. the data n-1 %, n-2 %, n-3 %, n-4 %, n-5 %, and Close to be the Filters, and the Next Price % to be the Target, Date to be the meta. Using those data to training, the result shows below:

Test and Score Mon Nov 08 21, 17:55:50

Settings

Sampling type: Stratified 5-fold Cross validation

Scores

Model	MSE	RMSE	MAE	R2
kNN	62.4700331608791	7.903798653867588	5.867753020823247	0.9839755486271949
SVM	1550.8594135353048	39.38095242036821	31.73577730401294	0.6021825185484875
Random Forest	56.1257875944556	7.491714596436226	5.7644797753964605	0.9856029377837603
Neural Network	36283.640822554575	190.4826522876941	171.6748735543076	-8.30726955902206

The data shows here:

Date	Open	High	Low	Close	Adj Close	Volume	Day_n-1	n-1 %	Day_n-2	n-2 %	Day_n-3	n-3 %	Day_n-4	n-4 %	Day_n-5	n-5 %	Next Price	Next Price
10/01/2017	197.5	198	196.1	198	195.4995	15628780	195.6	-1.21212	195.1	-1.46464	193.3	-2.37374	189	-4.54545	189.4	-4.34344	200.8	1.414143
11/01/2017	200	201.4	198.7	200.8	198.2642	20703850	198	-1.39442	195.6	-2.58964	195.1	-2.83864	193.3	-3.73506	189	-5.8765	198.8	-0.99602
12/01/2017	201	201	196.6	198.8	196.2895	16362476	200.8	1.006036	198	-0.40242	195.6	-1.60966	195.1	-1.86117	193.3	-2.7666	199.7	0.452713
13/01/2017	199	200.2	198.7	199.7	197.1781	10155934	198.8	-0.45067	200.8	0.550829	198	-0.85128	195.6	-2.05308	195.1	-2.30345	196.7	-1.50225
16/01/2017	199.7	199.7	196	196.7	194.216	11600274	199.7	1.525165	198.8	1.067619	200.8	2.084396	198	0.660906	195.6	-0.55922	197.7	0.508388
17/01/2017	199.9	199.9	197.5	197.7	195.2033	8845604	196.7	-0.50582	199.7	1.011634	198.8	0.556402	200.8	1.568035	198	0.151747	199.7	1.011634
18/01/2017	198.6	199.8	198	199.7	197.1781	10962405	197.7	-1.0015	196.7	-1.50225	199.7	0	198.8	-0.45067	200.8	0.550829	199.5	-0.10015
19/01/2017	200.6	200.6	198.3	199.5	196.9806	9664250	199.7	0.100249	197.7	-0.90226	196.7	-1.40351	199.7	0.100249	198.8	-0.35088	197.7	-0.90226
20/01/2017	199	199.4	197.7	197.7	195.2033	7686725	199.5	0.910472	199.7	1.011634	197.7	0	196.7	-0.50582	199.7	1.011634	197.8	0.050585
23/01/2017	196.4	199.3	196.4	197.8	195.3021	7059398	197.7	-0.05056	199.5	0.859452	199.7	0.960563	197.7	-0.05056	196.7	-0.55612	197.8	0
24/01/2017	198.9	198.9	196.9	197.8	195.3021	8825373	197.8	0	197.7	-0.05056	199.5	0.859452	199.7	0.960563	197.7	-0.05056	200.8	1.516683
25/01/2017	201	201	199.2	200.8	198.2642	17017199	197.8	-1.49402	197.8	-1.49402	197.7	-1.54383	199.5	-0.64741	199.7	-0.54781	205	2.091632
26/01/2017	202.4	205.4	201.8	205	202.4111	21037878	200.8	-2.04878	197.8	-3.51219	197.8	-3.51219	197.7	-3.56098	199.5	-2.68293	204.4	-0.29269
27/01/2017	205.6	205.6	204.2	204.4	201.8187	8717396	205	0.293545	200.8	-1.76125	197.8	-3.22896	197.8	-3.22896	197.7	-3.27789	206.6	1.076327
01/02/2017	202	206.8	200.8	206.6	203.991	19494245	204.4	-1.06487	205	-0.77445	200.8	-2.80736	197.8	-4.25944	197.8	-4.25944	205.2	-0.67764
02/02/2017	207	208.4	204.4	205.2	202.6086	14245064	206.6	0.682266	204.4	-0.38987	205	-0.09746	200.8	-2.14425	197.8	-3.60623	205	-0.09746
03/02/2017	205.4	206.4	202.4	205	202.4111	12525535	205.2	0.09756	206.6	0.780491	204.4	-0.29269	205	0	200.8	-2.04878	207.4	1.170729
06/02/2017	206.4	207.4	204.4	207.4	204.7808	9244792	205	-1.15718	205.2	-1.06075	206.6	-0.38572	204.4	-1.44648	205	-1.15718	205.4	-0.96432
07/02/2017	206	207.4	204.4	205.4	202.8061	11540961	207.4	0.97371	205	-0.19474	205.2	-0.09737	206.6	0.584232	204.4	-0.48685	204	-0.68159
08/02/2017	204.4	205.2	202.4	204	201.4238	15978068	205.4	0.686272	207.4	1.666664	205	0.490196	205.2	0.588234	206.6	1.274513	204.2	0.098038
09/02/2017	205.6	207	204.2	204.2	201.6213	12967234	204	-0.09794	205.4	0.587658	207.4	1.56709	205	0.391774	205.2	0.489716	202.6	-0.78354
10/02/2017	205.4	206	202.4	202.6	200.0415	13335776	204.2	0.789729	204	0.691014	205.4	1.382028	207.4	2.369194	205	1.184597	203.6	0.493583
13/02/2017	203	205	203	203.6	201.0288	11469706	202.6	-0.49116	204.2	0.294691	204	0.196461	205.4	0.884081	207.4	1.866399	202.8	-0.39293
14/02/2017	203.2	203.8	202.8	202.8	200.2389	11741212	203.6	0.394479	202.6	-0.09862	204.2	0.690332	204	0.591714	205.4	1.282047	206.6	1.873769
15/02/2017	202.2	207.8	202.2	206.6	203.991	20967951	202.8	-1.8393	203.6	-1.45208	202.6	-1.93611	204.2	-1.16167	204	-1.25847	212	2.613743
16/02/2017	210	213	208	212	209.3228	30654514	206.6	-2.54717	202.8	-4.33962	203.6	-3.96226	202.6	-4.43396	204.2	-3.67925	212.2	0.094338
17/02/2017	213.6	215	211.8	212.2	209.5202	18573582	212	-0.09425	206.6	-2.63902	202.8	-4.42978	203.6	-4.05278	202.6	-4.52403	214	0.848258
20/02/2017	213.2	215.2	212.6	214	211.2975	14388585	212.2	-0.84112	212	-0.93458	206.6	-3.45794	202.8	-5.23364	203.6	-4.85981	211	-1.40187
21/02/2017	213.6	214.8	210	211	208.3354	14978211	214	1.421801	212.2	0.568719	212	0.473934	206.6	-2.08531	202.8	-3.88625	215	1.895735
22/02/2017	212	215	212	215	212.2849	14282531	211	-1.86047	214	-0.46512	212.2	-1.30233	212	-1.39535	206.6	-3.90697	214.8	-0.09302
23/02/2017	215.6	216.4	214	214.8	212.0874	15728062	215	0.093108	211	-1.76909	214	-0.37244	212.2	-1.21043	212	-1.30354	211.2	-1.67598

According to those format data, training use everyday data's model have the smallest MAE value, and the model training by five days average data which get the second prize for less MAE result, the data using percentage get last.

Models

The model I used for the first time is Random Forest, which is an approach created by a number of decision trees and uses an integrated learning method to train and learn [2]. Using the close stock value, which is when the stock market closed on that day, that value of 0388 HKEX this stock. The mean absolute error (MAE) value shows the

difference between testing results generated by model and testing database, if the model has low MAE value, which is more accurate under the same situation. In this model the MAE value has about 0.8371 in whole testing, which is a good start for model prediction.

The second model I use to train is SVM, which is the method to convert some things that cannot be spilled to be separate [3]. The MAE value from the testing result using the SVM model training is slightly higher than Random Forest, reaching the 50.1081 error rate of the whole testing prediction.

The third model I use to train is the KNN model, which is an algorithm that uses the data that is already featured in processing and learning [4]. Using the model training by KNN, which is better than the SVM method but still not good compared with the Random Forest approach. The MAE result reached 10.0527 for this testing prediction.

The last model I use to train the stock data is the Neural Network approach [5], which needs to use features to process classification, and every neural point can be input and output pass to next points, also to analyze the parts of the high feature. The MAE result generated from the model training by Neural Network to testing is a huge difference from other models, which reached 2822.0371, and this method is the most not accurate in these four models.

In this project, I also used Excel to predict the price of 0700.HK this stock, which uses an average function in Excel to calculate the five days average prices data (10 days average price predict is the same way), and calculate MAE using the function:

$$\text{n Days ABS} = \text{abs}\left[(\text{n Days MA} - \text{Close}) / \text{Close}\right]$$

Also, I use five days and ten days forecast to predict, and use the forecast function to calculate the MAE of this way:

n Forecast ABS = abs [(n Days Forecast – Next Price) / Next Price]

The Excel table graph show below:

Date	Open	High	Low	Close	Adj Close	Volume	Next Price	5 Days MA	5 Days ABS	5 Days Forecast	5 Forecast ABS	10 Days MA	10 Days ABS	10 Days Forecast	10 Forecast ABS
17/01/2017	199.9	199.9	197.5	197.7	195.2033	8845604	199.7	198.7399994	0.9599976	197.0858452	2.614151849	196.4700012	3.2299958	201.8999397	2.199942712
18/01/2017	198.6	199.8	198	199.7	197.1781	10962405	199.5	198.5199982	0.9800018	196.4738757	3.026124328	197.5400009	1.9599991	200.6540576	1.154057563
19/01/2017	200.6	200.6	198.3	199.5	196.9806	9664250	197.7	198.6599976	0.9600006	198.0776074	0.377610418	198.1600006	0.4600036	199.9658675	2.265870453
20/01/2017	199	199.4	197.7	197.7	195.2033	7686725	197.8	198.2599976	0.4599946	198.743395	0.943391981	198.4199997	0.6199967	199.6554841	1.855481105
23/01/2017	196.4	199.3	196.4	197.8	195.3021	7059398	197.8	198.4799988	0.6799958	200.1599991	2.3599961	198.6399994	0.8399964	198.8494683	1.049465284
24/01/2017	198.9	198.9	196.9	197.8	195.3021	8825373	200.8	198.5	2.300003	197.7943428	3.005660151	198.6199997	2.1800033	197.821648	2.978355029
25/01/2017	201	201	199.2	200.8	198.2642	17017199	205	198.7200012	6.2799988	197.199258	7.800742037	198.6199997	6.3800003	197.4411699	7.558830077
26/01/2017	202.4	205.4	201.8	205	202.4111	21037878	204.4	199.8200012	4.5799928	199.1709	5.229094045	199.2399994	5.1599946	198.7675773	5.632416733
27/01/2017	205.6	205.6	204.2	204.4	201.8187	8717396	206.6	201.1600006	5.4400054	203.322645	3.277360981	199.7099991	6.8900069	201.0250484	5.574957578
01/02/2017	202	206.8	200.8	206.6	203.991	19494245	205.2	202.9200012	2.2799958	215.4399859	10.2399889	200.7	4.499997	205.2326332	0.032636221
02/02/2017	207	208.4	204.4	205.2	202.6086	14245064	205	204.4	0.6	208.6597976	3.659797603	201.45	3.55	206.477099	1.477099045
03/02/2017	205.4	206.4	202.4	205	202.4111	12525535	207.4	205.2399994	2.1599946	206.6736848	0.72630918	201.9800003	5.4199937	206.6249044	0.775089619
06/02/2017	206.4	207.4	204.4	207.4	204.7808	9244792	205.4	205.7199982	0.3200042	205.8751901	0.475196124	202.7699997	2.6299943	208.102419	2.702425029
07/02/2017	206	207.4	204.4	205.4	202.8061	11540961	204	205.9199982	1.9199982	207.0436075	3.043607466	203.5399994	0.4600006	208.7632698	4.763269827
08/02/2017	204.4	205.2	202.4	204	201.4238	15978068	204.2	205.399997	1.2	206.1895436	1.989546575	204.1599991	0.0399979	208.0995259	3.899528854
09/02/2017	205.6	207	204.2	204.2	201.6213	12967234	202.6	205.199997	2.599991	205.2865633	2.686557313	204.7999985	2.1999925	206.9310261	4.331020143
10/02/2017	205.4	206	202.4	202.6	200.0415	13335776	203.6	204.7199982	1.1199922	204.4622602	0.862254189	204.9799988	1.3799928	205.8213761	2.221370134
13/02/2017	203	205	203	203.6	201.0289	11469706	202.8	203.9600006	1.1599976	199.3200117	3.4799913	204.8399994	2.0399964	204.3480045	1.548001514
14/02/2017	203.2	203.8	202.8	202.8	200.2389	11741212	206.6	203.4400024	3.1600036	202.7188774	3.881128623	204.6800003	1.9200057	203.8267577	2.773248256
15/02/2017	202.2	207.8	202.2	206.6	203.991	20967951	212	203.9600036	8.0399964	202.8194091	9.180590948	204.6800003	7.3199997	202.6843865	9.315613505
16/02/2017	210	213	208	212	209.3228	30654514	212.2	205.5200042	6.6799928	204.9865741	7.213422918	205.3600006	6.8399964	203.902334	8.297662968
17/02/2017	213.6	215	211.8	212.2	209.5202	18573582	214	207.4400024	6.5599976	209.9528319	4.047168113	206.0800003	7.9199997	206.558404	7.441596002
20/02/2017	213.2	215.2	212.6	214	211.2975	14388585	211	209.5200012	1.4799988	220.6399919	9.6399919	206.7400009	4.2599991	209.8498296	1.150170396
21/02/2017	213.6	214.8	210	211	208.3354	14978211	215	211.1600006	3.8399994	217.5396201	2.539620066	207.3000015	7.6999985	213.0539747	1.946025331
22/02/2017	212	215	212	215	212.2849	14282531	214.8	212.8399994	1.9600036	213.6925357	1.107467343	208.4000015	6.4000015	213.740716	1.059286981
23/02/2017	215.6	216.4	214	214.8	212.0874	15728062	211.2	213.4	2.200003	213.9402988	2.740301843	209.4600021	1.7399949	215.4637135	4.263716496
24/02/2017	213.4	214.2	211.2	211.2	208.5329	17224457	210.2	213.2	3.000003	214.7792482	4.57925117	210.3200012	0.1200042	216.7466334	6.546636449
27/02/2017	211	213.6	209.6	210.2	207.5455	14141432	207	212.4399994	5.4399994	212.2999985	5.2999985	210.9800003	3.9800003	217.7999973	10.79999726
28/02/2017	210.8	211.6	207	207	204.3859	16362243	206.8	211.6399994	4.8399964	210.2962228	3.496219821	211.4	4.599997	214.8202519	8.02024892
01/03/2017	207	207.4	205	206.8	204.1884	19151883	206.4	210	3.600006	206.505968	0.105974045	211.4199997	5.0200057	211.0414518	4.641457752
02/03/2017	210.6	211	206.4	206.4	203.7935	15808759	207.2	208.3199982	1.1200012	205.4343288	1.765668216	210.8599991	3.6600021	208.3466218	1.146624771

After training the model and testing to get the result, the Random Forest approach can be used in this kind of predict case, which is the lowest error rate in those models. Below is the table to shows the MAE of the stock:

10-day moving average	5-day moving average	10-day forecast	5-day forecast	Machine learning based on previous prices	Machine learning based on previous price change %
10.32494595	7.736287091	9.294608247	8.296353017	1.579813048895642	5.7644797753964605

According to all the results, the machine learning based on previous prices is more accurate, and using the ten days average price data, which have higher MAE that means not that accurate.

New Method (RNN)

The new method I found is the Recurrent neural network (RNN) model, a kind of

Deep Neural Network. The reason I choose RNN method because of the timestamp function is in this Neural Network, which is relative the stock time feature, that time will be connect with the price. Also, RNN model can remember a long-time memoir, and the network allows the parameters storing in model memoirs storing a long time [6].

The timestamp function is like this:

$$\frac{\partial \mathcal{L}^{(T)}}{\partial \mathrm{W}} = \sum_{t=1}^{T} \frac{\partial \mathcal{L}^{(T)}}{\partial \mathrm{W}} \mid (t)$$

Thanks, code from MARCO CARUJO this user on Kaggle platform (https://www.kaggle.com/mcarujo/rnn-tesla-stock-price), I change some parts of the code, training the model and predict by myself. I uploaded the datasets to the Kaggle platform and changed the datasets to mine.

[1]:
```
import numpy as np
import pandas as pd
import plotly.express as px
import plotly.graph_objects as go
import pandas_datareader.data as web
from sklearn.metrics import mean_squared_error
from sklearn.metrics import r2_score
```

[2]:
```
data = pd.read_csv('../input/0700hk-data/0700.HK_training.csv')
data = data.round(3)
data.head()
```

[2]:

	Date	Open	High	Low	Close	Adj Close	Volume
0	2017-01-03	188.0	191.1	188.0	189.4	187.008	9637272
1	2017-01-04	190.4	190.5	188.1	189.0	186.613	11411490
2	2017-01-05	191.0	194.2	190.7	193.3	190.859	20543005
3	2017-01-06	196.2	196.8	194.7	195.1	192.636	20077760
4	2017-01-09	196.7	196.9	195.2	195.6	193.130	13605277

Now the code is displaying the RNN model (in this case, I decide to use LSTM framework, which is a sub-framework under the RNN), LSTM, which can store the last training data, and they have a chain that can transfer the information to other layers, and LSTM network can decide data which forget and member, so which keep

the highly accuracy in the model [7]. The RNN model code shows below:

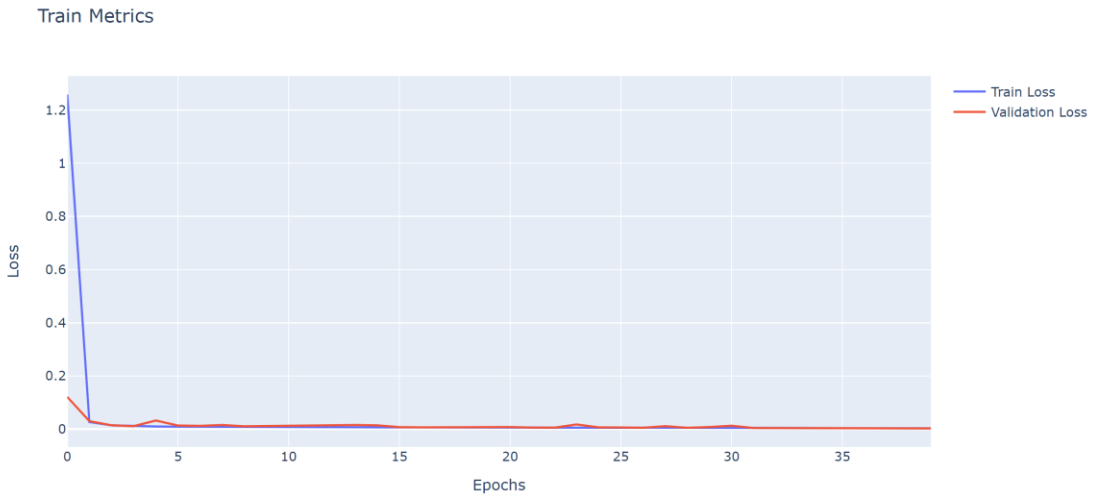
```
%%time

import keras
from keras.models import Sequential
from keras.layers import Dense, LSTM, Dropout

def gen_model():
    rnn = Sequential()
    rnn.add(LSTM(units=360, return_sequences=True, input_shape=(X_train_close_scaled.shape[1], 1)))
    rnn.add(Dropout(0.2))
    rnn.add(LSTM(units=360, return_sequences=True))
    rnn.add(Dropout(0.2))
    rnn.add(LSTM(units=360, return_sequences=True))
    rnn.add(Dropout(0.2))
    rnn.add(LSTM(units=360, return_sequences=True))
    rnn.add(Dropout(0.2))
    rnn.add(LSTM(units=360, return_sequences=True))
    rnn.add(Dropout(0.2))
    rnn.add(LSTM(units=360))
    rnn.add(Dropout(0.2))
    rnn.add(Dense(units=1,activation='relu'))
    rnn.compile(optimizer="adam", loss=["mean_squared_error",'cosine_similarity'])
    callback = keras.callbacks.EarlyStopping(
        monitor="val_loss",
        min_delta=0,
        patience=0,
        verbose=False,
        mode="auto",
        baseline=None,
        restore_best_weights=False,
    )
    return rnn

rnn = gen_model()
history = rnn.fit(X_train_close_scaled, y_train_close_scaled,validation_data=(X_test_close_scaled,y_test_close_scaled), epochs=40, batch_size=32)
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 61, 360)           521280
_________________________________________________________________
dropout (Dropout)            (None, 61, 360)           0
_________________________________________________________________
lstm_1 (LSTM)                (None, 61, 360)           1038240
_________________________________________________________________
dropout_1 (Dropout)          (None, 61, 360)           0
_________________________________________________________________
lstm_2 (LSTM)                (None, 61, 360)           1038240
_________________________________________________________________
dropout_2 (Dropout)          (None, 61, 360)           0
_________________________________________________________________
lstm_3 (LSTM)                (None, 61, 360)           1038240
_________________________________________________________________
dropout_3 (Dropout)          (None, 61, 360)           0
_________________________________________________________________
lstm_4 (LSTM)                (None, 61, 360)           1038240
_________________________________________________________________
dropout_4 (Dropout)          (None, 61, 360)           0
_________________________________________________________________
lstm_5 (LSTM)                (None, 360)               1038240
_________________________________________________________________
dropout_5 (Dropout)          (None, 360)               0
_________________________________________________________________
dense (Dense)                (None, 1)                 361
=================================================================
Total params: 5,712,841
Trainable params: 5,712,841
Non-trainable params: 0
_________________________________________________________________
```

In this RNN model, I use 5 groups of LSTM method to training the model. The training model loss result, which calculate by loss function:

$$\mathcal{L}(y1, y) = \sum_{t=1}^{T} \mathcal{L}(y1^{<t>}, y^{<t>})$$

The Loss graph shows below:



In the end, the predict result is like this:


```
r2_score close : 0.857605538228998
mean_squared_error close: 0.002190527972826093
```

According to the result from RNN model, the MAE is 0.00219, the result is more less than those models that training by orange tool.

Reference

[1] Yahoo Finance, “Tencent (0700.HK) stock historical prices & data,” *Yahoo! Finance*, 08-Nov-2021. [Online]. Available: https://finance.yahoo.com/quote/0700.HK/history. [Accessed: 08-Nov-2021].

[2] A. Dubey, "Feature selection using Random Forest," *Medium*, 15-Dec-2018. [Online]. Available: https://towardsdatascience.com/feature-selection-using-random-forest-26d7b747597f. [Accessed: 08-Nov-2021].

[3] S. Ray, "SVM: Support Vector Machine Algorithm in machine learning," *Analytics Vidhya*, 13-Sep-2017. [Online]. Available: https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/. [Accessed: 08-Nov-2021].

[4] Mahima, "Features of KNN algorithm," *Edureka Community*, 13-May-2019. [Online]. Available: https://www.edureka.co/community/46176/features-of-knn-algorithm. [Accessed: 08-Nov-2021].

[5] C. Nicholson, "A beginner's Guide to Neural Networks and deep learning," *Pathmind*. [Online]. Available: https://wiki.pathmind.com/neural-network. [Accessed: 08-Nov-2021].

[6] A. Amidi and S. Amidi, "Recurrent neural networks cheatsheet star," *CS 230 - Recurrent Neural Networks Cheatsheet*. [Online]. Available: https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks. [Accessed: 09-Nov-2021].

[7] Colah, “Understanding LSTM networks,” *Understanding LSTM Networks -- colah's blog*, 27-Aug-2015. [Online]. Available: https://colah.github.io/posts/2015-08-Understanding-LSTMs/. [Accessed: 09-Nov-2021].