CE418: Final Project

Θειου Βασιλειος(2685)

Χαραμης Γεωργιος(2655)

University Of Thessaly

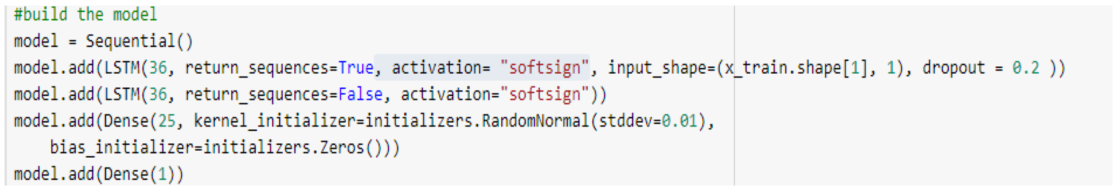
Final Project Description:

The problem was to predict stock's prices for the next 30 days after given a database of stock's prices for some time. So,

we extract the close price from the database and we used that to train our model. More precialy, we took 80% of the given data for training and the remaining 20% to test our predictions.

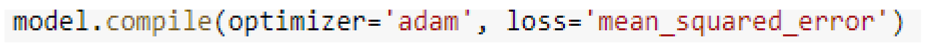
Model Architecture:

We use Long Short Term Memory(LSTM) neural network with 2 hidden layer, 36 neurons each one and added a dropout of 20% to the first one. As activation function we used softsign. We also added 2 dense layers with 25 and 1 neurons each. At the first one, we added some kernel weight initializers as you can see in the next image.



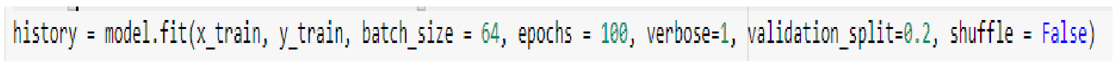
```
#build the model
model = Sequential()
model.add(LSTM(36, return_sequences=True, activation= "softsign", input_shape=(x_train.shape[1], 1), dropout = 0.2 ))
model.add(LSTM(36, return_sequences=False, activation="softsign"))
model.add(Dense(25, kernel_initializer=initializers.RandomNormal(stddev=0.01),
    bias_initializer=initializers.Zeros()))
model.add(Dense(1))
```

For training the above model, we used adam optimizer and as a loss function was mean squared error.



```
model.compile(optimizer='adam', loss='mean_squared_error')
```

For fitting the model we run for 100 epoch and a batch size of 67 . Compilation time is 0,0099 and our fit model run for approximately 107 seconds.



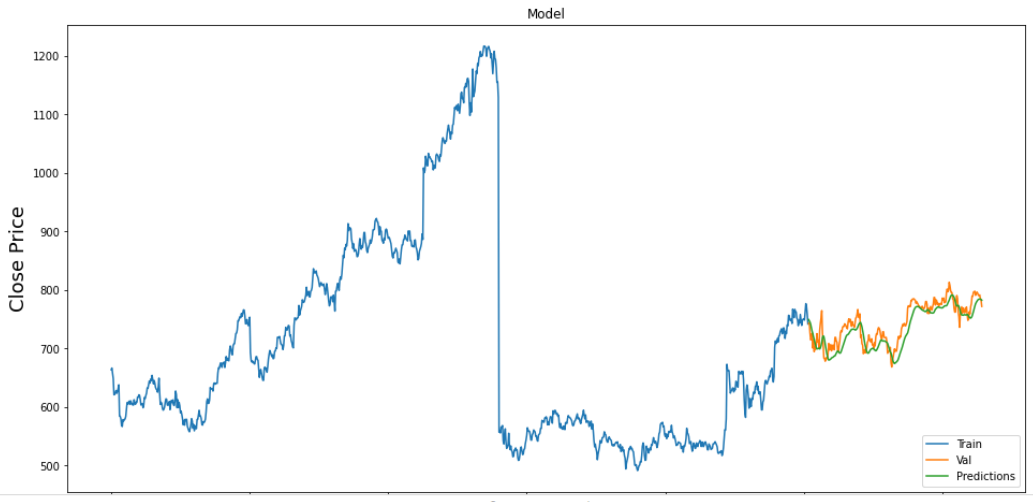
```
history = model.fit(x_train, y_train, batch_size = 64, epochs = 100, verbose=1, validation_split=0.2, shuffle = False)
```

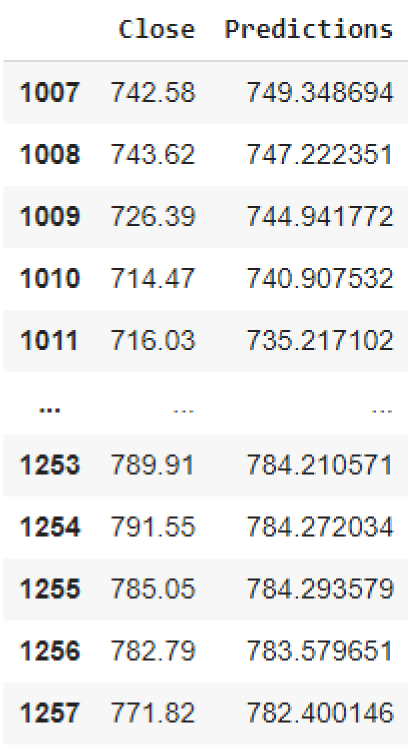
In order to calculate the error of our predictions and used root mean squared error

```
#root mean squared error
rmse = np.sqrt(np.mean(predictions - y_test)**2)
rmse
```

```
10.433841791038969
```

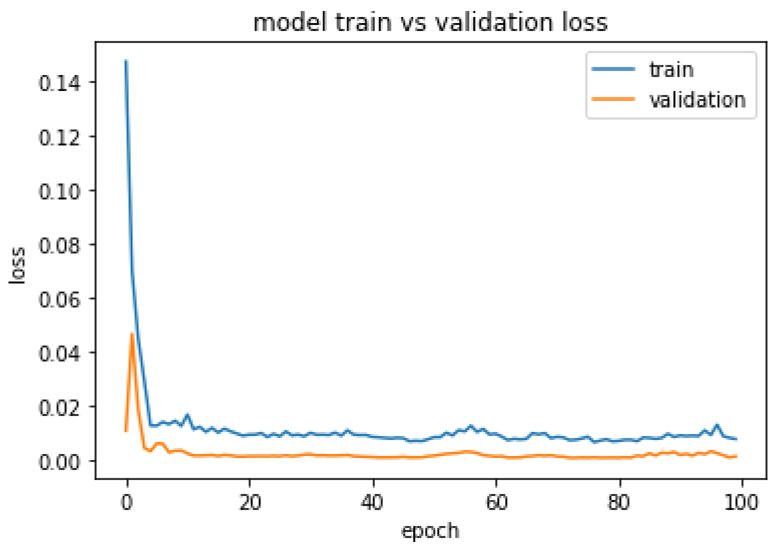
The 2 images bellow show how our model performed.

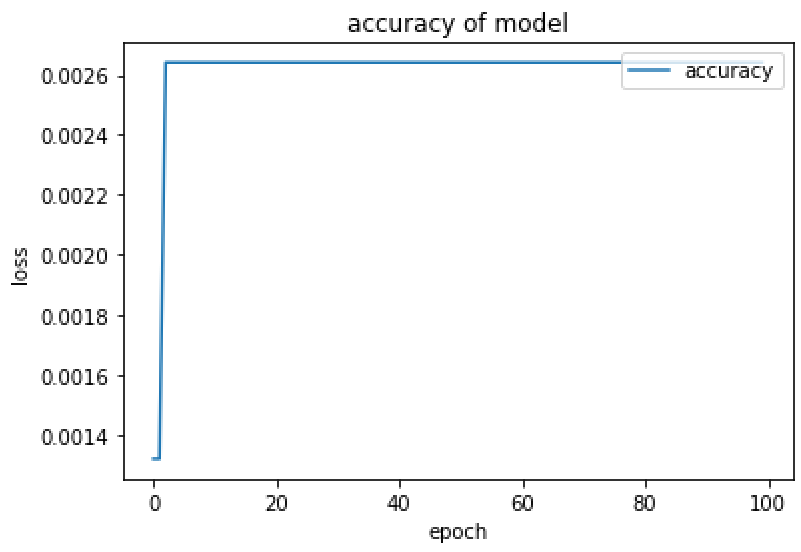




	Close	Predictions
1007	742.58	749.348694
1008	743.62	747.222351
1009	726.39	744.941772
1010	714.47	740.907532
1011	716.03	735.217102
...	...	...
1253	789.91	784.210571
1254	791.55	784.272034
1255	785.05	784.293579
1256	782.79	783.579651
1257	771.82	782.400146

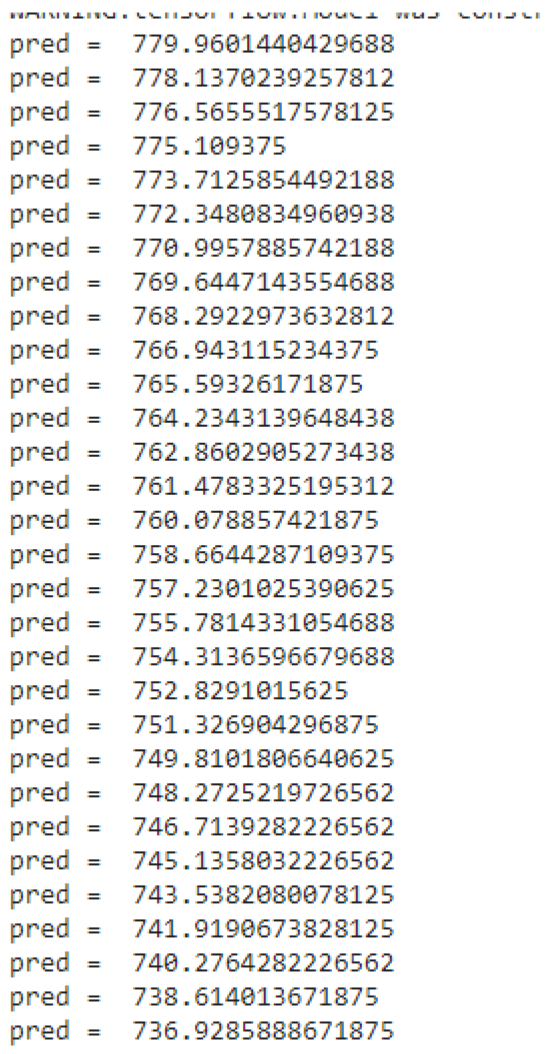
The accuracy of the model is :





Results:

In order to find the 30 asked values, we extract the last 30 stock’s closing prices from our database and predict the next one. Then, append our prediction and again extract 60 values and so on. The results of our prediction can be shown on the image bellow.



```
WARNING:tensorflow:Model was constr
pred =  779.9601440429688
pred =  778.1370239257812
pred =  776.5655517578125
pred =  775.109375
pred =  773.7125854492188
pred =  772.3480834960938
pred =  770.9957885742188
pred =  769.6447143554688
pred =  768.2922973632812
pred =  766.943115234375
pred =  765.59326171875
pred =  764.2343139648438
pred =  762.8602905273438
pred =  761.4783325195312
pred =  760.078857421875
pred =  758.6644287109375
pred =  757.2301025390625
pred =  755.7814331054688
pred =  754.3136596679688
pred =  752.8291015625
pred =  751.326904296875
pred =  749.8101806640625
pred =  748.2725219726562
pred =  746.7139282226562
pred =  745.1358032226562
pred =  743.5382080078125
pred =  741.9190673828125
pred =  740.2764282226562
pred =  738.614013671875
pred =  736.9285888671875
```

Inspiration:

We did not take ideas by a single source but we compined information from the sources bellow and we tried to improve them by searching on the web. After testing a lot of models we came up whith this one that seems to perform better.

1. https://www.datacamp.com/community/tutorials/lstm-python-stock-market
2. https://www.youtube.com/watch?v=SauRW1Vok44&ab_channel=Kite

3. https://www.youtube.com/watch?v=H6du_pfuznE&ab_channel=KrishNaik