SACRAMENTO STATE
COLLEGE OF ENGINEERING & COMPUTER SCIENCE

CSC 215-01 Artificial Intelligence

Mini-Project 4

Stock Price Prediction using LSTM and CNN

By

Mansi Patel

Sac State ID - 219760009

Megha Mathpal

Sac State ID – 219695880

March 27, 2019

Abstract— **This project report is about the stock prices prediction in which we predict the close of a day by using fully connected neural network, LSTM and CNN.**

I. Problem Statement

The project aims to build a stock price predicting system to predict the close of the day by using fully connected neural network and by implementing models like LSTM and CNN to predict the close of 7th day. We will compare the RMSE score of all models to find the best model of all.

II. Methodology

A. *Understanding the data*

- The dataset downloaded from https://drive.google.com/drive/folders/1Bp4_UvWcfZLK2fwiLCpyDcUO9PL9i-Z5.
- This dataset contains stock prices of various companies. We consider understanding this data by analyzing it.
- The main dataset has 4392 rows and 7 columns in total that has [‘Date’, ‘Open’, ‘High’, ‘Low’, ‘Adj_Close’, ‘Close’, ‘Volume’] fields.
- Visualized the ‘Close’ stock value by plotting it in a Graph.
- In this project, the fields [Open, Low ,High, Volume] that represents each day’s open value, low value, high value, volume of the stock value changed respective, used to predict the ‘Close’ value for that day.

B. *Data Pre processing and Cleaning*

- Dropped the unnecessary columns ‘Date’ and ‘Adj_Close’.
- Removed missing/null values in our data by dropping that row. There were no missing values found in our data.
- Redundant records in the train set will cause learning algorithms to be biased towards the more frequent records. So, we tried to remove all duplicate records, but none were found.
- We perform normalization by z-score on the features.

C. *Models used*

- We split the test and train data. We used the first 70% of the records for training and the remaining 30% of the records for test. We implemented various models to predict the close. Those are
    - Fully-Connected Neural Networks (FCNN)
    - Long Short term Memory (LSTM)
    - Convolutional Neural Networks (CNN)
- For FCNN we z score all columns except close which is the prediction column.
- For LSTM and CNN we take z score of all columns for x (input feature) and for y (output) we take the original close column.
- Compared the RMSE for ALL the models. Showed “regression lift chart” of test data of each model.
- Used Dropout layers, Early Stopping and Model Checkpoint when training neural networks using TensorFlow.
- We did hyper parameters tuning of activation, layer and neuron count, optimizer, kernel number and size when training neural networks using TensorFlow. Also for LSTM we changed the number of layers and neuron count in each layer.

III. Experimental Results and Analysis

- Table I has the comparison of all models variations implemented by FCNN

Optimizer	**Activation and layers**	**Number of neuron count**	**RMSE**
Adam	**(relu, relu)**	**(32,16)**	**0.3723**
Adamax	(tanh,tanh)	(25,25)	0.3921
SGD	(relu, relu)	(32,16)	0.4490
Adamax	(relu, relu)	(50,25)	0.3980
Adam	(sigmoid, relu, tanh)	(100,50,25)	0.5835
Adagrad	(relu, relu, relu, relu)	(25,25,25,25)	0.3962
Adamax	(sigmoid,sigmoid)	(50,30)	0.5807

Table I. FCNN different optimizers, layers and neuron count comparison

- We had a loop in our code that code has 4 optimizers ‘SGD’, 'Adagrad', 'Adam', 'Adamax'.
- In these table we have the best optimizer’s RMSE after comparing it with all four optimizers.
- As we can see in table I ‘Adam’ optimizer is performing best as compared to the other optimizers.
- Table II shows the various combinations tried for LSTM.

Optimizer	Activation and layers	Number of neuron count	RMSE
SGD	(relu, relu, sigmoid)	(128,128, 128)	34.7011
Adam	**(relu, relu)**	**(128,64)**	**3.0068**
Adam	(relu)	(128)	6.4358
Adam	(relu)	(64)	5.4238
Adam	(relu)	(32)	5.5125

Table II. LSTM different combinations

- Table III shows different combinations used in CNN.

Optimize r	Kernel size	Kernel number	Activation and layers	RMSE
Adagrad	(1,3) (1,1)	64 128	(relu, relu)	5.8705
Adamax	**(1,3) (1,1)**	**64 128**	**(relu, relu) (without dropout)**	**1.9849**
Adam	(1,3) (1,1)	128, 64	(relu, relu)	3.8398
Adamax	(1,3) (1,3)	64 128	(relu, relu)	2.4575

Table III. CNN different combinations

IV. Task Division

Mansi Patel

- Data Preprocessing
- Shaping the data ready for model
- best N value
- Bidirectional LSTM

Megha Mathpal

- Data visualization
- Parameter tuning
- LSTM for predicting 5 continuous value
- good model for Google dataset

V. Project Reflection

- Get exposure to LSTM model for Neural Network
- Dealt with the time series data for the first time
- By visualising the time series data for the prediction value ‘Close’ observed the continuous pattern and fluctuations in the Graph.
- For, the FCNN the ‘Close’ which we are predicting is not included in the input features while LSTM & CNN have the last 7 values in the input feature.
- Defining shape for CNN was little tricky, need to understand and visualize the 4D data on paper.
- Processing the given data to be ready for applying model is the toughest part for the machine learning project and the prediction highly depends on that.
- FCNN & CNN includes the prediction column to be included for the input feature thus the input feature for ‘Close’ need to be normalized while output should be remaining as it is.
- Adding the dropout layer with 0.5 or greater decreases the performance of the model for LSTM & CNN
- Adding the more layer to LSTM leads to worst performance sometimes.
- Observed that the 2 initial layers with kernel number 128 & 64 respectively in LSTM & CNN gave our best model performance for the given data.
- For CNN the best model have the combination of Conv2D,Max pooling , Flatten, Dense, Dropout and Dense.
- Learned how to use the keras wrapper to implement the Bidirectional LSTM.
- LSTM took lot of time to run, for N best value function waited half an hour for LSTM.
- CNN was faster for the small set of data.
- Parameter tuning for CNN & LSTM is very tedious requires time to try different combinations.
- Optimizer SGD (customized) sometimes gave nan value during the LSTM execution

VI. Extra Features

A. *Best N value*

1) LSTM

- In the project, we predict [Close] of a day based on the last 7 days’ data. In this part we find the best N value (number of the days we should consider in the past) that yields the best model.
- Fig 1 shows the best N value graph for LSTM in the model.

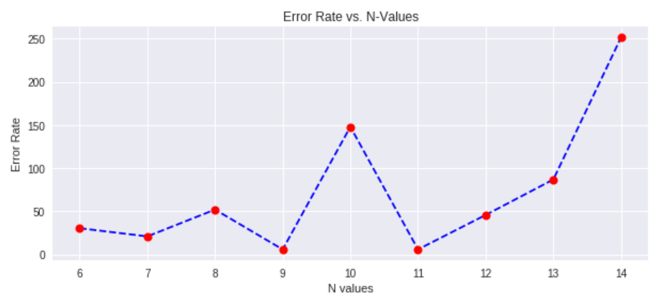

Fig 1. best n value graph for LSTM

- Made a function that takes range and we ran with our best Lstm model.
- Ran this function with several range and several times for the same range.
- LSTM gave good performance with N value 7,9,11 & 13.

2) CNN

- Fig 2 shows the best N value graph for CNN in the model.

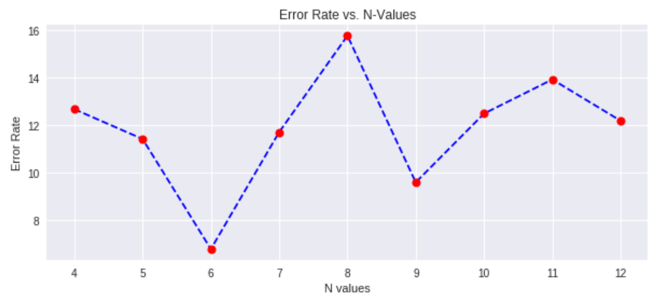

Fig 2. best n value graph for CNN

- Made a function that takes range and we ran with our best CNN model
- Ran this function with several range and several times for the same range
- LSTM gave good performance with N value 6,7 & 9.

B. *LSTM for continuous time*

- We used the JPM data set to work on this extra feature. Previously we tried this with apple dataset but we got nan values, so we continued with different dataset.
- First, we do data preprocessing by removing unrelated and useless data
- We split the data set as x,y training and testing. we do Z score on all columns for x data set and for y we use the original close column.
- By using the data set of last 5 days we predicted the data for next five days
- We used to predict sequences multiple function to predict the prices for continuous time.
- We tried few combinations for LSTM with apple dataset as shown in table IV below.

Optimizer	Activation and layers	Number of neuron count	RMSE
Adam	relu	(128,64)	46.8850

Table IV. different combinations for LSTM continious

C. *Bidirectional LSTM*

- On our original dataset we applied bidirectional LSTM . Bidirectional LSTMs are supported in Keras via the Bidirectional layer wrapper.
- This wrapper takes a recurrent layer (e.g. the first LSTM layer) as an argument.
- It also allows you to specify the merge mode, that is how the forward and backward outputs should be combined before being passed on to the next layer.
- Model are having dropout =0.2 and recurrent dropout=0.2.
- Table V below has summary of observation.

Optimizer	Activation and layers	Number of neuron count	RMSE
Adam	relu, relu	(64,128)	10.0595
Adam	relu	(32)	4.9650

Table V. Bidirectional different combinations

D. *Stock price prediction for Google dataset*

- We used the google data set for this extra feature
- For this we predict the close value by taking last 10 days value and predict the close value for next day after 10 days
- Tried previously obtained best model from the original dataset. Based on our observations we tried get the best performance.
- We got RMSE and regression lift chart for each model.
- We used the best model obtained previously, did not do much parameter tuning for these models. Table VI shows the summary of the best model obtained previously in our project i.e the best combinations obtained are used for this dataset as well.

Model	Optimizer	Activation and layers	Number of neuron count	RMSE
FCNN	Adam	relu,relu	(32,16)	4.5739
LSTM	Adam	relu, relu	(64,128)	52.1711
CNN	Adamax	relu, relu, relu	(64,128, 1024)	18.1740

Table VI. best model form project used for new dataset.