

Project no :1

Name : CHO VATIYA JAY MAHESHBHAI

Student ID : 1161980

Course: COMP5313 Artificial Intelligence

Topic: Stock Prices Prediction Using AI and Machine Learning

Stock Prices Prediction:

In order to forecast the prices of stock, I have gathered all the data from financial websites such as Yahoo Finance in which "Adani Enterprises Limited" ticker is selected and downloaded five years historical data. After obtaining all the necessary information, according to develop the model's required features, we sort the Date column by Year, Month, and Day, as well as computing the daily returns and moving average.

In the following stage, the data is separated into training and testing sets when the features are designed as well as the model may then be trained on the training set and tested on the testing set. Once the model is built up then we are going to use neural network because it is capable of recording complicated nonlinear interactions. Moreover, I have also defined the model's architecture and constructed the model, which includes the number of layers, neurons in each layer as well as include deciding on the loss function, optimizer, and assessment criteria respectively.

Last but not least, data may be used to train it which needs running the training data set through the model and modifying the weights in order to minimize loss. To elaborate, the model is then verified by running it over the testing set and comparing predicted to actual values. It gives an indication of how good the model is performing.

Outputs :1. Describe the adani shares stock data according to the date.

Jupyter Project_1_stock prediction(1161980) Last Checkpoint: an hour ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Kernel

```
data_file.read()
```

Out[3]:

	Date	Open	High	Low	Close	Adj Close	Volume
0	2018-02-12	115.083138	119.496841	115.083138	118.379791	116.667130	13340428
1	2018-02-14	119.060921	121.785423	118.107338	119.578575	117.848579	12691224
2	2018-02-15	120.477661	121.322258	114.783440	117.017540	115.324593	9129656
3	2018-02-16	117.589684	119.033676	111.595772	113.230476	111.592316	10018309
4	2018-02-19	111.704750	113.039757	109.252693	110.560455	108.960930	8936687

In [4]: `#understanding the data`
`data_file.describe()`

Out[4]:

	Open	High	Low	Close	Adj Close	Volume
count	1236.000000	1236.000000	1236.000000	1236.000000	1236.000000	1.236000e+03
mean	965.759690	983.825570	945.414674	965.176069	964.225123	6.642477e+06
std	1117.636842	1134.035374	1096.252517	1115.259350	1115.613560	8.687182e+06
min	67.113579	69.263008	64.707497	67.402306	66.427162	2.482490e+05
25%	143.000000	146.974998	140.337505	143.124996	142.032997	2.294392e+06
50%	252.250000	256.625000	249.949997	253.250000	251.137329	4.052000e+06
75%	1622.650025	1660.112488	1587.100037	1621.450012	1620.491760	7.129097e+06
max	4175.000000	4190.000000	4066.399902	4165.299805	4165.299805	8.885864e+07

Outputs 2: By using describe function i showed all the values which is given in below image.

Jupyter Project_1_stock prediction(1161980) Last Checkpoint: an hour ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Kernel

```
X_data.describe()
```

In [12]:

Out[12]:

	Open	High	Low	Adj Close	Volume
count	927.000000	927.000000	927.000000	927.000000	9.270000e+02
mean	1133.918708	1151.557705	1114.880484	1133.007711	3.325535e+06
std	1191.948484	1208.651896	1173.453329	1193.066144	1.667494e+06
min	69.327171	69.327171	67.081497	66.964645	2.482490e+05
25%	151.900002	154.599998	148.474998	150.132462	1.898094e+06
50%	447.250000	452.250000	438.049988	445.464844	3.129269e+06
75%	1753.000000	1771.849976	1723.224976	1750.680786	4.496028e+06
max	4175.000000	4190.000000	4066.399902	4165.299805	7.124786e+06

In [13]: `Y_data.describe()`

Out[13]:

count	927.000000
mean	1133.872482
std	1192.705098
min	67.947685
25%	151.425003
50%	445.950012
75%	1751.424988
max	4165.299805
Name:	Close, dtype: float64

Outputs 3:

```
jupyter Project_1_stock prediction(1161980) Last Checkpoint: an hour ago (autosaved) Logout
```

```
File Edit View Insert Cell Kernel Widgets Help Trusted Kernel
```

```
tf.keras.layers.Dense(64, activation='relu', input_shape=(X_train.shape[1],)),
tf.keras.layers.Dense(1)
])
model.compile(optimizer='adam', loss='mean_squared_error')

In [18]: # Train the model on the training data
model.fit(X_train, y_train, epochs=500, batch_size=32, validation_data=(X_test, y_test))
Epoch 492/500
29/29 [=====] - 0s 3ms/step - loss: 99.9828 - val_loss: 14.3356
Epoch 493/500
29/29 [=====] - 0s 3ms/step - loss: 69.1078 - val_loss: 30.9435
Epoch 494/500
29/29 [=====] - 0s 3ms/step - loss: 103.5536 - val_loss: 30.7697
Epoch 495/500
29/29 [=====] - 0s 3ms/step - loss: 87.1688 - val_loss: 14.0169
Epoch 496/500
29/29 [=====] - 0s 3ms/step - loss: 78.5974 - val_loss: 15.1579
Epoch 497/500
29/29 [=====] - 0s 3ms/step - loss: 83.7125 - val_loss: 13.9162
Epoch 498/500
29/29 [=====] - 0s 3ms/step - loss: 75.4881 - val_loss: 17.2868
Epoch 499/500
29/29 [=====] - 0s 3ms/step - loss: 83.3743 - val_loss: 19.9780
Epoch 500/500
29/29 [=====] - 0s 4ms/step - loss: 79.8720 - val_loss: 18.2156
```

Outputs 4:

```
jupyter Project_1_stock prediction(1161980) Last Checkpoint: an hour ago (autosaved) Logout
```

```
File Edit View Insert Cell Kernel Widgets Help Trusted Kernel
```

```
In [19]: from sklearn.linear_model import LinearRegression
logi = LinearRegression()
logi.fit(X_train, y_train)
pred = logi.predict(X_test)

In [20]: mean_absolute_error = tf.keras.losses.mean_absolute_error(y_test, pred)
mean_absolute_error = mean_absolute_error.numpy().mean().item()
print(f'your mean absolute error is :{mean_absolute_error:.2f}')

your mean absolute error is :0.46
```

Outputs 5: According to the trained model, the predicated and actual output is shown in the below graph.



