EX05 Stock Price Prediction

AIM

To develop a Recurrent Neural Network model for stock price prediction.

Problem Statement and Dataset

The given problem is to predict the google stock price based on time.For this we are provided with a dataset which contains features like Date,Opening Price,Highest Price,Lowest Price,Closing Price,Adjusted Closing,Price and Volume.Based on the given features, develop a RNN model to predict the price of stocks in future.

Design Steps

- Step 1: Import the required packages
- Step 2: Load the dataset
- Step 3: Perform the necessary data preprocessing
- Step 4: Build and fit the data in the Learning model
- Step 5: Predict using the fit model
- Step 6: Check the error value of the predicted pricing model

Program

Name: DEEPIKA S
Register Number: 212222230028

Importing Libraries

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from keras import layers
from keras.models import Sequential
```

Loading Dataset

```
dtrain=pd.read_csv('trainset.csv')
dtrain.columns
dtrain.head()
dtrainset=dtrain.iloc[:,1:2].values
```

Scaling the Data

```
sc = MinMaxScaler(feature_range=(0,1))
training_set_scaled = sc.fit_transform(dtrainset)
training_set_scaled.shape
```

Training the Data

```
X_train_array = []
y_train_array = []
for i in range(60, 1259):
    X_train_array.append(training_set_scaled[i-60:i,0])
    y_train_array.append(training_set_scaled[i,0])
X_train, y_train = np.array(X_train_array), np.array(y_train_array)
X_train1 = X_train.reshape((X_train.shape[0], X_train.shape[1],1))
X_train.shape
```

Creating Network Model

```
model = Sequential([layers.SimpleRNN(42,input_shape=(60,1)),layers.Dense(1)])
model.compile(optimizer='adam',loss='mse')
model.summary()
model.fit(X_train1,y_train,epochs=20, batch_size=32)
```

Reading Test Data

```
dataset_test = pd.read_csv('testset.csv')
test_set = dataset_test.iloc[:,1:2].values
test_set.shape
```

Training Test Data

```
dataset_total = pd.concat((dtrain['Open'],dataset_test['Open']),axis=0)
inputs = dataset_total.values
inputs = inputs.reshape(-1,1)
inputs_scaled=sc.transform(inputs)
X_test = []
y_test = []
```

```
for i in range(60,1384):
    X_test.append(inputs_scaled[i-60:i,0])
    y_test.append(inputs_scaled[i,0])
X_test = np.array(X_test)
X_test = np.reshape(X_test,(X_test.shape[0], X_test.shape[1],1))
X_test.shape
```

Ploting Results

```
predicted_stock_price_scaled = model.predict(X_test)
predicted_stock_price = sc.inverse_transform(predicted_stock_price_scaled)
plt.figure(figsize=(8,3))
plt.plot(np.arange(0,1384),inputs, color='red', label = 'Test data')
plt.plot(np.arange(60,1384),predicted_stock_price, color='green',label = 'Predicted stock price')
plt.title('DEEPIKA S - 212222230028\nStock Price Prediction')
plt.xlabel('Time')
plt.ylabel('Stock Price')
plt.legend()
plt.show()
```

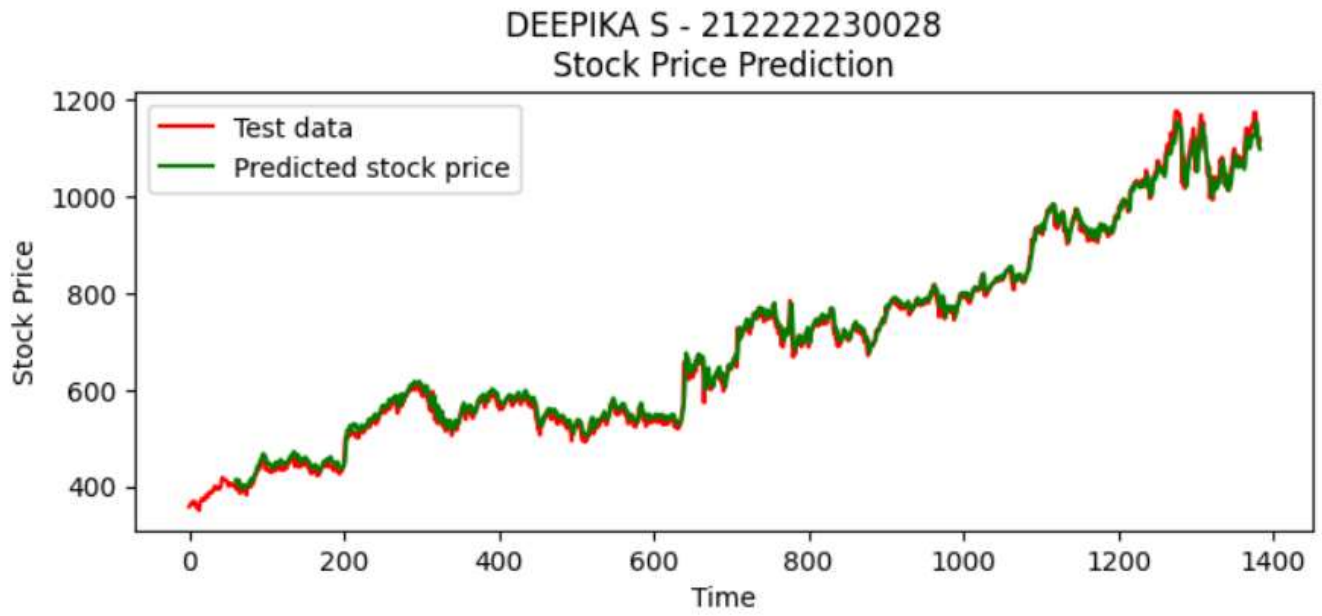
Mean Square Error

```
from sklearn.metrics import mean_squared_error as mse
print('DEEPIKA S')
print(mse(y_test,predicted_stock_price))
```

Output

True Stock Price, Predicted Stock Price vs time

```
42/42 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step
```



Mean Square Error

```
DEEPIKA S
544283.8194571127
```

Result

Thus, a Recurrent Neural Network model for stock price prediction is developed.