**Project Title: Bitcoin Price Prediction using LSTM**

**Overview:**

Cryptocurrencies, particularly Bitcoin, exhibit highly volatile price movements, making them an intriguing subject for predictive modeling. This project focuses on developing a machine learning model, specifically a Long Short-Term Memory (LSTM) neural network, to forecast Bitcoin prices. LSTMs are well-suited for time-series data, making them effective for capturing patterns and dependencies in cryptocurrency price fluctuations.

**Objectives:**

1. **Data Collection:**
    - I have used historical Bitcoin price data, including daily or hourly prices, trading volumes, and other relevant metrics from YAHOO finance website live data

      https://finance.yahoo.com/quote/BTC-USD/history/?fr=sycsrp_catchall

2. **Data Preprocessing:**

   After that I have Clean and preprocess the data, handling missing values, normalizing prices, and create sequences suitable for LSTM input.

3. **Exploratory Data Analysis (EDA):**

   Then I have Explore the dataset to understand trends, seasonality, and other patterns that may influence Bitcoin prices.

4. **Feature Engineering:**

   Then I did some feature engineering  to derive additional features or transformations the data that may enhance the model's ability to capture relevant patterns.

5. **Model Architecture:**

   Then I Design and implement an LSTM neural network architecture suitable for time-series prediction.

6. **Model Training:**

   After that Split the dataset into training and testing sets, and train the LSTM model using historical price sequences.

7. **Hyperparameter Tuning:**

   Next i optimize hyperparameters such as the number of LSTM layers, neurons, learning rate, and dropout to improve the model's performance.

8. **Model Evaluation:**

   After that I assess the model's performance on the testing set using metrics like Mean Squared Error (MSE) or Root Mean Squared Error (RMSE).

9. **Visualization:**

Finally I used visualization tools to visualize the model predictions against actual Bitcoin prices to analyze the accuracy and potential areas for improvement.

**Outcomes:**

- A trained LSTM model capable of predicting future Bitcoin prices based on historical data.
- Visualization tools for assessing the model's accuracy and understanding its predictions.
- Real-time prediction capabilities for users interested in short-term Bitcoin price forecasts.

Accurate cryptocurrency price predictions can be valuable for traders, investors, and other stakeholders in the cryptocurrency market, aiding in decision-making and risk management.

+ Code + Text

[1]
```
import tensorflow as tf
import math
import numpy as np
import pandas as pd
import pandas_datareader as web
from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras.layers import Dense, LSTM
import matplotlib.pyplot as plt
```

[2]
```
import pandas_datareader.data as pdr
from datetime import datetime
```

[9]
```
df= pd.read_csv("/content/BTC-USD (1).csv")
```

```
df.head()
```

| | Date | Open | High | Low | Close | Adj Close | Volume |
|---|---|---|---|---|---|---|---|
| 0 | 2019-12-02 | 7424.036133 | 7474.818848 | 7233.399414 | 7321.988281 | 7321.988281 | 17082040706 |
| 1 | 2019-12-03 | 7323.975586 | 7418.858887 | 7229.356934 | 7320.145508 | 7320.145508 | 14797485769 |
| 2 | 2019-12-04 | 7320.125000 | 7539.784668 | 7170.922852 | 7252.034668 | 7252.034668 | 21664240918 |

0s completed at 11:12 AM

+ Code + Text

[23]
```
# Compile the model
model_1.compile(optimizer='adam', loss='mse')
```

```
# Train the model
callback = tf.keras.callbacks.EarlyStopping(monitor='loss', patience=2)
history = model_1.fit(X_train, y_train, batch_size=1, epochs=10)
```

```
Epoch 1/10
1110/1110 [==============================] - 42s 32ms/step - loss: 0.0030
Epoch 2/10
1110/1110 [==============================] - 38s 34ms/step - loss: 0.0016
Epoch 3/10
1110/1110 [==============================] - 36s 32ms/step - loss: 0.0011
Epoch 4/10
1110/1110 [==============================] - 31s 28ms/step - loss: 9.5320e-04
Epoch 5/10
1110/1110 [==============================] - 36s 32ms/step - loss: 8.9537e-04
Epoch 6/10
1110/1110 [==============================] - 31s 28ms/step - loss: 7.6707e-04
Epoch 7/10
1110/1110 [==============================] - 30s 27ms/step - loss: 7.3314e-04
Epoch 8/10
1110/1110 [==============================] - 31s 27ms/step - loss: 7.8679e-04
Epoch 9/10
1110/1110 [==============================] - 31s 28ms/step - loss: 7.8725e-04
Epoch 10/10
1110/1110 [==============================] - 30s 27ms/step - loss: 6.7640e-04
```

[25]

✓ 0s completed at 11:12 AM

# Data

```
last_60_days = data[-60:].values
last_60_days_scaled = scaler.fit_transform(last_60_days)
new_X_test = []
new_X_test.append(last_60_days_scaled)
# Convert the X_test data set to a numpy array
new_X_test = np.array(new_X_test)
# Reshape the data
new_X_test = np.reshape(new_X_test, (new_X_test.shape[0], new_X_test.shape[1], 1))
# Get the predicted scaled price
pred_price = model_1.predict(new_X_test)
# Undo the scaling
pred_price = scaler.inverse_transform(pred_price)
print(pred_price)
```

```
1/1 [==============================] - 0s 237ms/step
[[39108.727]]
```

| Date | Open | High | Low | Close | Adj Close | Volume |
|---|---|---|---|---|---|---|
| 12/2/2019 | 7424.036133 | 7474.818848 | 7233.399414 | 7321.988281 | 7321.988281 | 17082040706 |
| 12/3/2019 | 7323.975586 | 7418.858887 | 7229.356934 | 7320.145508 | 7320.145508 | 14797485769 |
| 12/4/2019 | 7320.125 | 7539.784668 | 7170.922852 | 7252.034668 | 7252.034668 | 21664240918 |
| 12/5/2019 | 7253.241699 | 7743.431641 | 7232.676758 | 7448.307617 | 7448.307617 | 18816085231 |
| 12/6/2019 | 7450.561523 | 7546.996582 | 7392.175293 | 7546.996582 | 7546.996582 | 18104466307 |
| 12/7/2019 | 7547.265625 | 7589.95166 | 7525.711426 | 7556.237793 | 7556.237793 | 15453520564 |
| 12/8/2019 | 7551.338867 | 7634.606445 | 7476.091309 | 7564.345215 | 7564.345215 | 15409908086 |
| 12/9/2019 | 7561.79541 | 7618.091797 | 7365.985352 | 7400.899414 | 7400.899414 | 17872021272 |
| 12/10/2019 | 7397.134277 | 7424.022949 | 7246.043945 | 7278.119629 | 7278.119629 | 18249031195 |
| 12/11/2019 | 7277.197754 | 7324.15625 | 7195.527344 | 7217.427246 | 7217.427246 | 16350490689 |
| 12/12/2019 | 7216.73877 | 7266.639648 | 7164.741211 | 7243.134277 | 7243.134277 | 18927080224 |
| 12/13/2019 | 7244.662109 | 7293.560547 | 7227.122559 | 7269.68457 | 7269.68457 | 17125736940 |
| 12/14/2019 | 7268.902832 | 7308.836426 | 7097.208984 | 7124.673828 | 7124.673828 | 17137029730 |
| 12/15/2019 | 7124.239746 | 7181.075684 | 6924.375977 | 7152.301758 | 7152.301758 | 16881129804 |
| 12/16/2019 | 7153.663086 | 7171.168945 | 6903.682617 | 6932.480469 | 6932.480469 | 20213265950 |
| 12/17/2019 | 6931.31543 | 6964.075195 | 6587.974121 | 6640.515137 | 6640.515137 | 22363804217 |
| 12/18/2019 | 6647.698242 | 7324.984863 | 6540.049316 | 7276.802734 | 7276.802734 | 31836522778 |
| 12/19/2019 | 7277.59082 | 7346.602539 | 7041.381836 | 7202.844238 | 7202.844238 | 25904604416 |
| 12/20/2019 | 7208.636719 | 7257.921875 | 7086.124023 | 7218.816406 | 7218.816406 | 22633815180 |
| 12/21/2019 | 7220.59375 | 7223.226074 | 7112.73584 | 7191.158691 | 7191.158691 | 19312552168 |
| 12/22/2019 | 7191.188477 | 7518.033203 | 7167.179199 | 7511.588867 | 7511.588867 | 23134537956 |
| 12/23/2019 | 7508.902344 | 7656.17627 | 7326.192383 | 7355.628418 | 7355.628418 | 27831788041 |
| 12/24/2019 | 7354.393066 | 7535.716797 | 7269.528809 | 7322.532227 | 7322.532227 | 22991622105 |
| 12/25/2019 | 7325.755859 | 7357.02002 | 7220.991211 | 7275.155762 | 7275.155762 | 21559505149 |
| 12/26/2019 | 7274.799316 | 7388.302734 | 7200.386719 | 7238.966797 | 7238.966797 | 22787010034 |
| 12/27/2019 | 7238.141113 | 7363.529297 | 7189.934082 | 7290.088379 | 7290.088379 | 22777360996 |
| 12/28/2019 | 7289.03125 | 7399.041016 | 7286.905273 | 7317.990234 | 7317.990234 | 21365673026 |

BTC-USD (1)