BITCOIN PRICE PREDICTION FOR 14 DAYS

```
#importing essential libraries
import pandas as pd
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
```

```
#loading dataset
from google.colab import files
files.upload()
```

```
Choose Files No file chosen    Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving BTC-USD.csv to BTC-USD (1).csv
{'BTC-USD.csv': b'Date,Open,High,Low,Close,Adj Close,Volume\n2020-07-07,9349.161133,9
```

```
#show the data
df=pd.read_csv('BTC-USD.csv')
df
```

	Date	Open	High	Low	Close	Adj Close	
0	2020-07-07	9349.161133	9360.617188	9201.815430	9252.277344	9252.277344	1.383
1	2020-07-08	9253.020508	9450.335938	9249.500000	9428.333008	9428.333008	1.970
2	2020-07-09	9427.994141	9431.378906	9234.999023	9277.967773	9277.967773	1.800
3	2020-07-10	9273.357422	9287.471680	9118.001953	9278.807617	9278.807617	1.686
4	2020-07-11	9277.511719	9293.532227	9199.485352	9240.346680	9240.346680	1.324
...	...	...	...	...	...	...	
361	2021-07-03	33854.421875	34909.261719	33402.695313	34668.546875	34668.546875	2.438
362	2021-07-04	34665.566406	35937.566406	34396.476563	35287.781250	35287.781250	2.492

```
#checking for null values
df.isnull().any()
```

```
Date         False
Open         False
High         False
Low          False
```

```
Close        False
Adj Close    False
Volume       False
dtype: bool
```

```
#replacing null values with some random values
df = df.fillna(method='ffill')
```

```
#create a variable for predicting 'n' days out in the future
projection = 14
#create a new column called prediction
df['Prediction'] = df[['Close']].shift(-projection)
df
```

	Date	Open	High	Low	Close	Adj Close	
0	2020-07-07	9349.161133	9360.617188	9201.815430	9252.277344	9252.277344	1.383
1	2020-07-08	9253.020508	9450.335938	9249.500000	9428.333008	9428.333008	1.970
2	2020-07-09	9427.994141	9431.378906	9234.999023	9277.967773	9277.967773	1.800
3	2020-07-10	9273.357422	9287.471680	9118.001953	9278.807617	9278.807617	1.686
4	2020-07-11	9277.511719	9293.532227	9199.485352	9240.346680	9240.346680	1.324
...	...	...	...	...	...	...	
361	2021-07-03	33854.421875	34909.261719	33402.695313	34668.546875	34668.546875	2.438
362	2021-07-04	34665.566406	35937.566406	34396.476563	35287.781250	35287.781250	2.492

```
#create the independent data set (X)
X = np.array(df[['Close']])
#remove the last 14 rows
X = X[:-projection]
print(X)
```

```
[[ 9252.277344]
 [ 9428.333008]
 [ 9277.967773]
 [ 9278.807617]
 [ 9240.34668 ]
 [ 9276.5     ]
 [ 9243.614258]
 [ 9243.213867]
 [ 9192.836914]
 [ 9132.227539]
 [ 9151.392578]
 [ 9159.040039]
```

```
[ 9185.817383]
[ 9164.231445]
[ 9374.887695]
[ 9525.363281]
[ 9581.072266]
[ 9536.892578]
[ 9677.113281]
[ 9905.166992]
[10990.873047]
[10912.823242]
[11100.467773]
[11111.213867]
[11323.466797]
[11759.592773]
[11053.614258]
[11246.348633]
[11205.892578]
[11747.022461]
[11779.773438]
[11601.472656]
[11754.045898]
[11675.739258]
[11878.111328]
[11410.525391]
[11584.93457 ]
[11784.137695]
[11768.871094]
[11865.698242]
[11892.803711]
[12254.402344]
[11991.233398]
[11758.283203]
[11878.37207 ]
[11592.489258]
[11681.825195]
[11664.847656]
[11774.595703]
[11366.134766]
[11488.363281]
[11323.397461]
[11542.5     ]
[11506.865234]
[11711.505859]
[11680.820313]
[11970.478516]
[11414.03418 ]
[10245.296875]
```

```
#create the dependent data set (y)
y = df['Prediction'].values
y = y[:-projection]
print(y)
```

```
[ 9374.887695  9525.363281  9581.072266  9536.892578  9677.113281
  9905.166992 10990.873047 10912.823242 11100.467773 11111.213867
 11323.466797 11759.592773 11053.614258 11246.348633 11205.892578
 11747.022461 11779.773438 11601.472656 11754.045898 11675.739258
 11878.111328 11410.525391 11584.93457  11784.137695 11768.871094
 11865.698242 11892.803711 12254.402344 11991.233398 11758.283203
 11878.37207  11592.489258 11681.825195 11664.847656 11774.595703
 11366.134766 11488.363281 11323.397461 11542.5      11506.865234
```

```
11711.505859 11680.820313 11970.478516 11414.03418  10245.296875
10511.813477 10169.567383 10280.351563 10369.563477 10131.516602
10242.347656 10363.138672 10400.915039 10442.170898 10323.755859
10680.837891 10796.951172 10974.905273 10948.990234 10944.585938
11094.34668  10938.271484 10462.259766 10538.459961 10225.864258
10745.548828 10702.290039 10754.4375   10774.426758 10721.327148
10848.830078 10787.618164 10623.330078 10585.164063 10565.493164
10684.428711 10804.000977 10621.664063 10679.136719 10923.62793
10923.62793  11296.361328 11384.181641 11384.181641 11384.181641
11429.506836 11495.349609 11322.123047 11358.101563 11483.359375
11742.037109 11916.334961 12823.689453 12965.891602 12931.539063
13108.0625   13031.173828 13075.248047 13654.21875  13271.285156
13437.882813 13546.522461 13780.995117 13737.109375 13550.489258
13950.300781 14133.707031 15579.848633 15565.880859 14833.753906
15479.567383 15332.31543  15290.902344 15701.339844 16276.34375
16317.808594 16068.138672 15955.587891 16716.111328 17645.40625
17804.005859 17817.089844 18621.314453 18642.232422 18370.001953
18364.121094 19107.464844 18732.121094 17150.623047 17108.402344
17717.414063 18177.484375 19625.835938 18802.998047 19201.091797
19445.398438 18699.765625 19154.230469 19345.121094 19191.630859
18321.144531 18553.916016 18264.992188 18058.904297 18803.65625
19142.382813 19246.644531 19417.076172 21310.597656 22805.162109
23137.960938 23869.832031 23477.294922 22803.082031 23783.029297
23241.345703 23735.949219 24664.791016 26437.037109 26272.294922
27084.808594 27362.4375   28840.953125 29001.720703 29374.152344
32127.267578 32782.023438 31971.914063 33992.429688 36824.363281
39371.042969 40797.609375 40254.546875 38356.441406 35566.65625
33922.960938 37316.359375 39187.328125 36825.367188 36178.140625
35791.277344 36630.074219 36069.804688 35547.75     30825.699219
33005.761719 32067.642578 32289.378906 32366.392578 32569.849609
30432.546875 31649.605469 34316.386719 34269.523438 33114.359375
33537.175781 35510.289063 37472.089844 36926.066406 38144.308594
39266.011719 38903.441406 46196.464844 46481.105469 44918.183594
47909.332031 47504.851563 47105.515625 48717.289063 47945.058594
49199.871094 52149.007813 51679.796875 55888.132813 56099.519531
57539.945313 54207.320313 48824.425781 49705.332031 47093.851563
46339.761719 46188.453125 45137.769531 49631.242188 48378.988281
50538.242188 48561.167969 48927.304688 48912.382813 51206.691406
52246.523438 54824.117188 56008.550781 57805.121094 57332.089844
61243.085938 59302.316406 55907.199219 56804.902344 58870.894531
57858.921875 58346.652344 58313.644531 57523.421875 54529.144531
54738.945313 52774.265625 51704.160156 55137.3125   55973.511719
55950.746094 57750.199219 58917.691406 58918.832031 59095.808594
59384.3125   57603.890625 58758.554688 59057.878906 58192.359375
56048.9375   58323.953125 58245.003906 59793.234375 60204.964844
59893.453125 63503.457031 63109.695313 63314.011719 61572.789063
60683.820313 56216.183594 55724.265625 56473.03125  53906.089844
51762.273438 51093.652344 50050.867188 49004.253906 54021.753906
55033.117188 54824.703125 53555.109375 57750.175781 57828.050781
56631.078125 57200.292969 53333.539063 57424.007813 56396.515625
57356.402344 58803.777344 58232.316406 55859.796875 56704.574219
```

```
#split the data into 85% training and 15% testing data sets
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size = .15)


#create and train model
linReg = LinearRegression()
#train the model
linReg.fit(x_train, y_train)
```

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

```
#test the model using score
linReg_confidence = linReg.score(x_test, y_test)
print('Linear regression Confidence:', linReg_confidence)
```

```
Linear regression Confidence: 0.8620951654398986
```

```
#create a variable called x_projection and set it equal to the last 14 rows of data from t
x_projection = np.array(df[['Close']])[-projection:]
print(x_projection)
```

```
[[34662.4375  ]
 [31637.779297]
 [32186.277344]
 [34649.644531]
 [34434.335938]
 [35867.777344]
 [35040.835938]
 [33572.117188]
 [33897.046875]
 [34668.546875]
 [35287.78125 ]
 [33746.003906]
 [34235.195313]
 [34809.742188]]
```

```
#print the linear regression models predictions for the next 14 days
linReg_prediction = linReg.predict(x_projection)
print(linReg_prediction)
```

```
[35327.40042769 32557.86951946 33060.10223215 35315.68653452
 35118.5390345  36431.07090178 35673.8812858  34329.04768788
 34626.56983774 35332.99448221 35899.99697502 34488.2672121
 34936.19574842 35462.28009308]
```