In [1]:
```python
import numpy as np
import pandas as pd
df = pd.read_csv("bitcoin.csv")
df.head()
```

Out[1]:

	Date	Price
0	5/23/2019	7881.846680
1	5/24/2019	7987.371582
2	5/25/2019	8052.543945
3	5/26/2019	8673.215820
4	5/27/2019	8805.778320

In [2]:
```python
df.drop(['Date'],1,inplace=True)
```

```
C:\Users\USER\AppData\Local\Temp\ipykernel_10104\2060810189.py:1: FutureWarning: In a future version of pandas all arguments of DataFrame.drop except for the argument 'labels' will be keyword-only.
  df.drop(['Date'],1,inplace=True)
```

In [3]:
```python
predictionDays = 30
# Create another column shifted 'n'  units up
df['Prediction'] = df[['Price']].shift(-predictionDays)
# show the first 5 rows
df.head()
```

Out[3]:

	Price	Prediction
0	7881.846680	10701.69141
1	7987.371582	10855.37109
2	8052.543945	11011.10254
3	8673.215820	11790.91699
4	8805.778320	13016.23145

In [4]:
```python
df.tail()
```

Out[4]:

	Price	Prediction
362	9729.038086	NaN
363	9522.981445	NaN
364	9081.761719	NaN
365	9182.577148	NaN
366	9180.045898	NaN

In [5]:
```python
# Create the independent dada set
# Here we will convert the data frame into a numpy array and drp the prediction column
x = np.array(df.drop(['Prediction'],1))
# Remove the last 'n' rows where 'n' is the predictionDays
x = x[:len(df)-predictionDays]
print(x)
```

```
[[ 7881.84668 ]
 [ 7987.371582]
 [ 8052.543945]
 [ 8673.21582 ]
 [ 8805.77832 ]
 [ 8719.961914]
 [ 8659.487305]
 [ 8319.472656]
 [ 8574.501953]
 [ 8564.016602]
 [ 8742.958008]
 [ 8208.995117]
 [ 7707.770996]
 [ 7824.231445]
 [ 7822.023438]
 [ 8043.951172]
 [ 7954.12793 ]
 [ 7688.077148]
 [ 8000.32959 ]
 [ 7927.714355]
 [ 8145.857422]
 [ 8230.923828]
 [ 8693.833008]
 [ 8838.375   ]
 [ 8994.488281]
 [ 9320.352539]
 [ 9081.762695]
 [ 9273.521484]
 [ 9527.160156]
 [10144.55664 ]
 [10701.69141 ]
 [10855.37109 ]
 [11011.10254 ]
 [11790.91699 ]
 [13016.23145 ]
 [11182.80664 ]
 [12407.33203 ]
 [11959.37109 ]
 [10817.15527 ]
 [10583.13477 ]
 [10801.67773 ]
 [11961.26953 ]
 [11215.4375  ]
 [10978.45996 ]
 [11208.55078 ]
 [11450.84668 ]
 [12285.95801 ]
 [12573.8125  ]
 [12156.5127  ]
 [11358.66211 ]
 [11815.98633 ]
 [11392.37891 ]
 [10256.05859 ]
 [10895.08984 ]
 [ 9477.641602]
 [ 9693.802734]
 [10666.48242 ]
 [10530.73242 ]
 [10767.13965 ]
 [10599.10547 ]
 [10343.10645 ]
 [ 9900.767578]
 [ 9811.925781]
 [ 9911.841797]
 [ 9870.303711]
 [ 9477.677734]
 [ 9552.860352]
 [ 9519.145508]
 [ 9607.423828]
 [10085.62793 ]
 [10399.66895 ]
 [10518.17481 ]
 [10821.72656 ]
 [10970.18457 ]
 [11805.65332 ]
 [11478.16895 ]
 [11941.96875 ]
 [11966.40723 ]
 [11862.93652 ]
 [11354.02441 ]
 [11523.5791  ]
 [11382.61621 ]
 [10895.83008 ]
 [10051.7041  ]
 [10311.5459  ]
 [10374.33887 ]
 [10231.74414 ]
 [10345.81055 ]
 [10916.05371 ]
 [10763.23242 ]
 [10138.04981 ]
 [10131.05566 ]
 [10407.96484 ]
 [10159.96094 ]
 [10138.51758 ]
 [10370.82031 ]
 [10185.5     ]
 [ 9754.422852]
 [ 9510.200195]
 [ 9598.173828]
 [ 9630.664063]
 [ 9757.970703]
 [10346.76074 ]
 [10623.54004 ]
 [10594.49316 ]
 [10575.5332  ]
 [10353.30273 ]
 [10517.25488 ]
 [10441.27637 ]
 [10334.97461 ]
 [10115.97559 ]
 [10178.37207 ]
 [10410.12695 ]
 [10360.54688 ]
 [10358.04883 ]
 [10347.71289 ]
 [10276.79395 ]
 [10241.27246 ]
 [10198.24805 ]
 [10266.41504 ]
 [10181.6416  ]
 [10019.7168  ]
 [10070.39258 ]
 [ 9729.324219]
 [ 8620.566406]
 [ 8486.993164]
 [ 8118.967773]
 [ 8251.845703]
 [ 8245.915039]
 [ 8104.185547]
 [ 8293.868164]
 [ 8343.276367]
 [ 8393.041992]
 [ 8259.992188]
 [ 8205.939453]
 [ 8151.500488]
 [ 7988.155762]
 [ 8245.623047]
 [ 8228.783203]
 [ 8595.740234]
 [ 8586.473633]
 [ 8321.756836]
 [ 8336.555664]
 [ 8321.005859]
 [ 8374.686523]
 [ 8205.369141]
 [ 8047.526855]
 [ 8103.911133]
 [ 7973.20752 ]
 [ 7988.560547]
 [ 8222.078125]
 [ 8243.720703]
 [ 8078.203125]
 [ 7514.671875]
 [ 7493.48877 ]
 [ 8660.700195]
 [ 9244.972656]
 [ 9551.714844]
 [ 9256.148438]
 [ 9427.6875  ]
 [ 9205.726563]
 [ 9199.584961]
 [ 9261.104492]
 [ 9324.717773]
 [ 9235.354492]
 [ 9412.612305]
 [ 9342.527344]
 [ 9360.879883]
 [ 9267.561523]
 [ 8804.880859]
 [ 8813.582031]
 [ 9055.526367]
 [ 8757.788086]
 [ 8815.662109]
 [ 8808.262695]
 [ 8708.094727]
 [ 8491.992188]
 [ 8550.760742]
 [ 8577.975586]
 [ 8309.286133]
 [ 8206.145508]
 [ 8027.268066]
 [ 7642.75     ]
 [ 7296.577637]
 [ 7397.796875]
 [ 7047.916992]
 [ 7146.133789]
 [ 7218.371094]
 [ 7531.663574]
 [ 7463.105957]
 [ 7761.243652]
 [ 7569.629883]
 [ 7424.29248 ]
 [ 7321.988281]
 [ 7320.145508]
 [ 7252.034668]
 [ 7448.307617]
 [ 7546.996582]
 [ 7556.237793]
 [ 7564.345215]
 [ 7400.899414]
 [ 7278.119629]
 [ 7217.427246]
 [ 7243.134277]
 [ 7269.68457 ]
 [ 7124.673828]
 [ 7152.301758]
 [ 6932.480469]
 [ 6640.515137]
 [ 7276.802734]
 [ 7202.844238]
 [ 7218.816406]
 [ 7191.158691]
 [ 7511.588867]
 [ 7355.628418]
 [ 7322.532227]
 [ 7275.155762]
 [ 7238.966797]
 [ 7290.088379]
 [ 7317.990234]
 [ 7422.652832]
 [ 7292.995117]
 [ 7193.599121]
 [ 7200.174316]
 [ 6985.470215]
 [ 7344.884277]
 [ 7410.656738]
 [ 7411.317383]
 [ 7769.219238]
 [ 8163.692383]
 [ 8079.862793]
 [ 7879.071289]
 [ 8166.554199]
 [ 8037.537598]
 [ 8192.494141]
 [ 8144.194336]
 [ 8827.764648]
 [ 8807.010742]
 [ 8723.786133]
 [ 8929.038086]
 [ 8942.808594]
 [ 8706.245117]
 [ 8657.642578]
 [ 8745.894531]
 [ 8680.875977]
 [ 8406.515625]
 [ 8445.43457 ]
 [ 8367.847656]
 [ 8596.830078]
 [ 8909.819336]
 [ 9358.589844]
 [ 9316.629883]
 [ 9508.993164]
 [ 9350.529297]
 [ 9392.875   ]
 [ 9344.365234]
 [ 9293.521484]
 [ 9180.962891]
 [ 9613.423828]
 [ 9729.801758]
 [ 9795.943359]
 [ 9865.119141]
 [10116.67383 ]
 [ 9856.611328]
 [10208.23633 ]
 [10326.05469 ]
 [10214.37988 ]
 [10312.11621 ]
 [ 9889.424805]
 [ 9934.433594]
 [ 9690.142578]
 [10141.99609 ]
 [ 9633.386719]
 [ 9608.475586]
 [ 9686.441406]
 [ 9663.181641]
 [ 9924.515625]
 [ 9650.174805]
 [ 9341.705078]
 [ 8820.522461]
 [ 8784.494141]
 [ 8672.455078]
 [ 8599.508789]
 [ 8562.454102]
 [ 8869.669922]
 [ 8787.786133]
 [ 8755.246094]
 [ 9078.762695]
 [ 9122.545898]
 [ 8909.954102]
 [ 8108.116211]
 [ 7923.644531]
 [ 7909.729492]
 [ 7911.430176]
 [ 4970.788086]
 [ 5563.707031]
 [ 5200.366211]
 [ 5392.314941]
 [ 5014.47998 ]
 [ 5225.629395]
 [ 5238.438477]
 [ 6191.192871]
 [ 6198.77832 ]
 [ 6185.066406]
 [ 5830.254883]
 [ 6416.314941]
 [ 6734.803711]
 [ 6681.062988]
 [ 6716.44043 ]
 [ 6469.79834 ]
 [ 6242.193848]
 [ 5922.042969]
 [ 6429.841797]
 [ 6438.644531]
 [ 6606.776367]
 [ 6793.624512]
 [ 6733.387207]
 [ 6867.527344]
 [ 6791.129395]
 [ 7271.78125 ]
 [ 7176.414551]
 [ 7334.098633]
 [ 7302.089355]
 [ 6865.493164]
 [ 6859.083008]
 [ 6971.091797]
 [ 6845.037598]
 [ 6842.427734]
 [ 6642.109863]
 [ 7116.804199]
 [ 7096.18457 ]
 [ 7257.665039]
 [ 7189.424805]
 [ 6881.958496]
 [ 6880.323242]
 [ 7117.20752 ]
 [ 7429.724609]]
```

```
C:\Users\USER\AppData\Local\Temp\ipykernel_10104\1732545242.py:3: FutureWarning: In a future version of pandas all arguments of DataFrame.drop except for the argument 'labels' will be keyword-only.
  x = np.array(df.drop(['Prediction'],1))
```

In [6]:
```python
# Create the dependent data set
# convert the data frame into a numpy array
y = np.array(df['Prediction'])
# Get all the values except last 'n' rows
y = y[:-predictionDays]
print(y)
```

```
[10701.69141  10855.37109  11011.10254  11790.91699  13016.23145
 11182.80664  12407.33203  11959.37109  10817.15527  10583.13477
 10801.67773  11961.26953  11215.4375   10978.45996  11208.55078
 11450.84668  12285.95801  12573.8125   12156.5127   11358.66211
 11815.98633  11392.37891  10256.05859  10895.08984   9477.641602
  9693.802734 10666.48242  10530.73242  10767.13965  10599.10547
 10343.10645   9900.767578  9811.925781  9911.841797  9870.303711
  9477.677734  9552.860352  9519.145508  9607.423828 10085.62793
 10399.66895  10518.17481  10821.72656  10970.18457  11805.65332
 11478.16895  11941.96875  11966.40723  11862.93652  11354.02441
 11523.5791   11382.61621  10895.83008  10051.7041   10311.5459
 10374.33887  10231.74414  10345.81055  10916.05371  10763.23242
 10138.04981  10131.05566  10407.96484  10159.96094  10138.51758
 10370.82031  10185.5       9754.422852  9510.200195  9598.173828
  9630.664063  9757.970703 10346.76074  10623.54004  10594.49316
 10575.5332   10353.30273  10517.25488  10441.27637  10334.97461
 10115.97559  10178.37207  10410.12695  10360.54688  10358.04883
 10347.71289  10276.79395  10241.27246  10198.24805  10266.41504
 10181.6416   10019.7168   10070.39258   9729.324219  8620.566406
  8486.993164  8118.967773  8251.845703  8245.915039  8104.185547
  8293.868164  8343.276367  8393.041992  8259.992188  8205.939453
  8151.500488  7988.155762  8245.623047  8228.783203  8595.740234
  8586.473633  8321.756836  8336.555664  8321.005859  8374.686523
  8205.369141  8047.526855  8103.911133  7973.20752   7988.560547
  8222.078125  8243.720703  8078.203125  7514.671875  7493.48877
  8660.700195  9244.972656  9551.714844  9256.148438  9427.6875
  9205.726563  9199.584961  9261.104492  9324.717773  9235.354492
  9412.612305  9342.527344  9360.879883  9267.561523  8804.880859
  8813.582031  9055.526367  8757.788086  8815.662109  8808.262695
  8708.094727  8491.992188  8550.760742  8577.975586  8309.286133
  8206.145508  8027.268066  7642.75      7296.577637  7397.796875
  7047.916992  7146.133789  7218.371094  7531.663574  7463.105957
  7761.243652  7569.629883  7424.29248   7321.988281  7320.145508
  7252.034668  7448.307617  7546.996582  7556.237793  7564.345215
  7400.899414  7278.119629  7217.427246  7243.134277  7269.68457
  7124.673828  7152.301758  6932.480469  6640.515137  7276.802734
  7202.844238  7218.816406  7191.158691  7511.588867  7355.628418
  7322.532227  7275.155762  7238.966797  7290.088379  7317.990234
  7422.652832  7292.995117  7193.599121  7200.174316  6985.470215
  7344.884277  7410.656738  7411.317383  7769.219238  8163.692383
  8079.862793  7879.071289  8166.554199  8037.537598  8192.494141
  8144.194336  8827.764648  8807.010742  8723.786133  8929.038086
  8942.808594  8706.245117  8657.642578  8745.894531  8680.875977
  8406.515625  8445.43457   8367.847656  8596.830078  8909.819336
  9358.589844  9316.629883  9508.993164  9350.529297  9392.875
  9344.365234  9293.521484  9180.962891  9613.423828  9729.801758
  9795.943359  9865.119141 10116.67383   9856.611328 10208.23633
 10326.05469  10214.37988  10312.11621   9889.424805  9934.433594
  9690.142578 10141.99609   9633.386719  9608.475586  9686.441406
  9663.181641  9924.515625  9650.174805  9341.705078  8820.522461
  8784.494141  8672.455078  8599.508789  8562.454102  8869.669922
  8787.786133  8755.246094  9078.762695  9122.545898  8909.954102
  8108.116211  7923.644531  7909.729492  7911.430176  4970.788086
  5563.707031  5200.366211  5392.314941  5014.47998   5225.629395
  5238.438477  6191.192871  6198.77832   6185.066406  5830.254883
  6416.314941  6734.803711  6681.062988  6716.44043   6469.79834
  6242.193848  5922.042969  6429.841797  6438.644531  6606.776367
  6793.624512  6733.387207  6867.527344  6791.129395  7271.78125
  7176.414551  7334.098633  7302.089355  6865.493164  6859.083008
  6971.091797  6845.037598  6842.427734  6642.109863  7116.804199
  7096.18457   7257.665039  7189.424805  6881.958496  6880.323242
  7117.20752   7429.724609  7550.900879  7569.936035  7679.867188
  7795.601074  7807.058594  8801.038086  8658.553711  8864.766602
  8988.59668   8897.46875   8912.654297  9003.070313  9268.761719
  9951.518555  9842.666016  9593.896484  8756.430664  8601.795898
  8804.477539  9269.987305  9733.72168   9328.197266  9377.013672
  9670.739258  9726.575195  9729.038086  9522.981445  9081.761719
  9182.577148  9180.045898]
```

In [7]:
```python
# Split the data into 80% training and 20% testing
from sklearn.model_selection import train_test_split
xtrain, xtest, ytrain, ytest = train_test_split(x,y, test_size = 0.2)
# set the predictionDays array equal to last 30 rows from the original data set
predictionDays_array = np.array(df.drop(['Prediction'],1))[-predictionDays:]
print(predictionDays_array)
```

```
[[7550.900879]
 [7569.936035]
 [7679.867188]
 [7795.601074]
 [7807.058594]
 [8801.038086]
 [8658.553711]
 [8864.766602]
 [8988.59668 ]
 [8897.46875 ]
 [8912.654297]
 [9003.070313]
 [9268.761719]
 [9951.518555]
 [9842.666016]
 [9593.896484]
 [8756.430664]
 [8601.795898]
 [8804.477539]
 [9269.987305]
 [9733.72168 ]
 [9328.197266]
 [9377.013672]
 [9670.739258]
 [9726.575195]
 [9729.038086]
 [9522.981445]
 [9081.761719]
 [9182.577148]
 [9180.045898]]
```

```
C:\Users\USER\AppData\Local\Temp\ipykernel_10104\3783758658.py:5: FutureWarning: In a future version of pandas all arguments of DataFrame.drop except for the argument 'labels' will be keyword-only.
  predictionDays_array = np.array(df.drop(['Prediction'],1))[-predictionDays:]
```

In [8]:
```python
from sklearn.svm import SVR
# Create and Train the Support Vector Machine (Regression) using radial basis function
svr_rbf = SVR(kernel='rbf', C=1e3, gamma=0.00001)
svr_rbf.fit(xtrain, ytrain)
```

Out[8]:

```
SVR
SVR(C=1000.0, gamma=1e-05)
```

In [9]:
```python
svr_rbf_confidence = svr_rbf.score(xtest,ytest)
print('SVR_RBF accuracy :',svr_rbf_confidence)
```

```
SVR_RBF accuracy : 0.17259267409914747
```

In [10]:
```python
# print the predicted values
svm_prediction = svr_rbf.predict(xtest)
print(svm_prediction)
print()
print(ytest)
```

```
[ 9572.18127581  8141.35685273  8097.24067385  8752.51259175
  9272.22097482  9345.47579352  8805.10379747  7048.02805505
  9900.58317489  8116.53270868  8861.41163305  8428.16836183
  8000.16705693  6826.47123038  8360.13291761  8802.88736232
  7726.04422864  8709.9942654   8417.85449479  8816.58666548
  8850.72628285  8777.31455494  9144.05875148 10348.32298471
 10525.19033445  9325.46783168  8561.97015295  8216.17054822
  8071.15800252  7688.31557373  7081.71757867  8930.62397651
  9802.77510907 10798.75225848  8846.21071498  8515.61199943
  8134.68223517  8520.82415812  7996.42518872  7593.95083756
  7665.58723006  8330.5802504  10313.86406468  8118.94615387
  8103.74395696  8136.07860329  8230.67079429  8086.28186245
  7792.24259078  7944.89906904  7295.1342098   7803.21023303
  9019.6741485   8436.61827244  7823.9416972   7531.59503544
  8017.97239386  8328.95273602  9865.06821054  9175.11509704
  9398.70215218 10366.54241856 10694.87359635  7612.46147478
  8110.66901246  8182.47848925  8188.76979175  9329.05998066]

[11450.84668   8620.566406 10276.79395   8336.555664  8206.145508
  9412.612305  8813.582031  6845.037598 11862.93652   9842.666016
  9003.070313 11182.80664  10407.96484   7807.058594  9608.475586
  9508.993164  7411.317383  8706.245117  9889.424805  9729.038086
  7550.900879  9341.705078  9663.181641 10266.41504  10115.97559
 10116.67383   5922.042969  8078.203125  7911.430176  9358.589844
  7116.804199  7275.155762 12285.95801   9477.677734  8897.46875
  8486.993164 10599.10547   7117.20752   8680.875977  7317.990234
  9180.045898  7217.427246 10517.25488   9729.324219  8321.756836
  7988.560547 10360.54688   9510.200195  6867.527344  8251.845703
  7189.424805  7320.145508  7152.301758  8929.038086  7278.119629
  7795.601074  7934.433594  7511.588867  7302.089355  8808.262695
  9244.972656 10198.24805  11382.61621   8745.894531  7546.996582
  8869.669922 10185.5       9729.801758]
```

In [11]:
```python
# Print the model predictions for the next 30 days
svm_prediction = svr_rbf.predict(predictionDays_array)
print(svm_prediction)
print()
#Print the actual price for bitcoin for last 30 days
print(df.tail(predictionDays))
```

```
[7743.66344439 7843.21151692 8727.72457548 9619.71698685 9676.35441473
 8233.07938223 8602.22563753 8116.53190449 7927.17825473 8065.33077708
 8042.45456498 7904.47932126 7871.32847628 8213.68943633 8047.60738119
 8112.63586669 8333.89613556 8763.7667418   8226.01186853 7874.17620971
 7940.06218847 8025.22065494 8148.90707515 7980.43972141 7940.48919691
 7940.21888175 8232.33677917 7788.48359433 7742.41972189 7741.0996575 ]

        Price  Prediction
337  7550.900879         NaN
338  7569.936035         NaN
339  7679.867188         NaN
340  7795.601074         NaN
341  7807.058594         NaN
342  8801.038086         NaN
343  8658.553711         NaN
344  8864.766602         NaN
345  8988.596680         NaN
346  8897.468750         NaN
347  8912.654297         NaN
348  9003.070313         NaN
349  9268.761719         NaN
350  9951.518555         NaN
351  9842.666016         NaN
352  9593.896484         NaN
353  8756.430664         NaN
354  8601.795898         NaN
355  8804.477539         NaN
356  9269.987305         NaN
357  9733.721680         NaN
358  9328.197266         NaN
359  9377.013672         NaN
360  9670.739258         NaN
361  9726.575195         NaN
362  9729.038086         NaN
363  9522.981445         NaN
364  9081.761719         NaN
365  9182.577148         NaN
366  9180.045898         NaN
```

In [ ]: