<u>Bitcoin Price Prediction using Time Series analysis</u>

Name: Debangan Mandal

Problem Statement:

This caters to the age old inquisitiveness, our power to predict the future, and also now, it's the topic of money, that many people want to have as much as they can. Here the model created will predict the closing value of the Bitcoin cryptocurrency, thus it will help in predicting when to buy and when to sell, thus getting the utmost profit.

Abstract:

In this project we try to implement Time Series Regression Analysis, to predict the closing values for the few months ahead of the present. This type of analysis has worked greatly for predicting certain values for the currency.

Most importantly, if we want the correct prediction, we need to have huge amounts of "good" data. But in the real world, as we all know, "good" data is quite impossible to find. Here "good" is meant by, the data should not have sudden spikes, or any kind of misinformation data, to keep all at bay.

Cryptography - a combination of advanced mathematics and computer science – connects the blocks. Any effort to change data breaks the cryptographic linkages between blocks, and computers in the network can rapidly identify it as false.

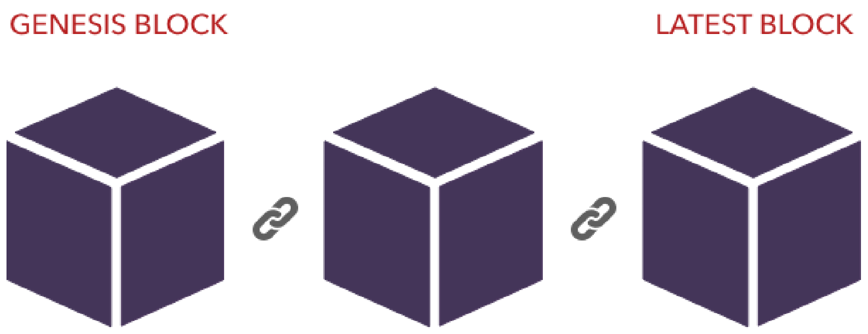
What is the crypto-currency market?

A cryptocurrency is a digital or virtual currency that is protected by encryption, making counterfeiting and double-spending practically impossible. Many cryptocurrencies are built on blockchain technology, which is a distributed ledger enforced by a global network of computers. Cryptocurrencies are distinguished by the fact that they are not issued by any central authority, making them potentially impervious to government intervention or manipulation.

If we get to think about it, is it actually quite similar to the currency we use in everyday life

To understand how trading works, we need to perceive what blockchain is and how it is so closely related to cryptocurrency's development.

A blockchain is a decentralised digital ledger of data. This is the transaction history for each bitcoin unit, showing how ownership has changed over time. Blockchain records transactions in 'blocks,' with fresh blocks added to the chain's front end.



What is Crypto-trading?

The act of speculating on cryptocurrency price fluctuations via a CFD trading account, or buying and selling the underlying coins via an exchange, is known as cryptocurrency trading.

CFDs trading are derivatives, which enable you to speculate on cryptocurrency price movements without taking ownership of the underlying coins. You can go long (‘buy’) if you think a cryptocurrency will rise in value, or short (‘sell’) if you think it will fall.

Cryptocurrency markets are decentralised, meaning they are neither issued or supported by a central authority like a government. Instead, they're distributed throughout a computer network. Cryptocurrencies, on the other hand, may be purchased and traded on exchanges and held in 'wallets.'

Cryptocurrencies, unlike traditional currencies, only exist as a shared digital record of ownership maintained on a blockchain. A user sends bitcoin units to another user's digital wallet. The transaction isn't deemed complete until it's validated and added to the blockchain, which is done through a process known as mining. New cryptocurrency tokens are frequently produced in this manner.

Market/Customer/Business Need Assessment

- We analyze the predictability of the bitcoin market across prediction horizons in the coming few months based on the data provided to us.
- I have made an ML model that will help us get through with the task, and give quite an accurate result.
- Using this model, people and also some of the traders can invest money in the crypto industry to gain some juicy profit.

Target Specifications and Characterization

- Here, the model that I have made, many novice traders, or some traders who would like to make easy money can use this model.
- We can make a website out of it, and also provide information on how the prediction was made, such that students can also do it easily.
- Many interested individuals can provide us the work for predicting market prices.
- All other cryptocurrencies that have been in the trading market for so long, can be predicted, and can help us with a variety of fruitful predictions.

External Search

The dataset I used was taken from:
https://www.kaggle.com/jessevent/all-crypto-currencies

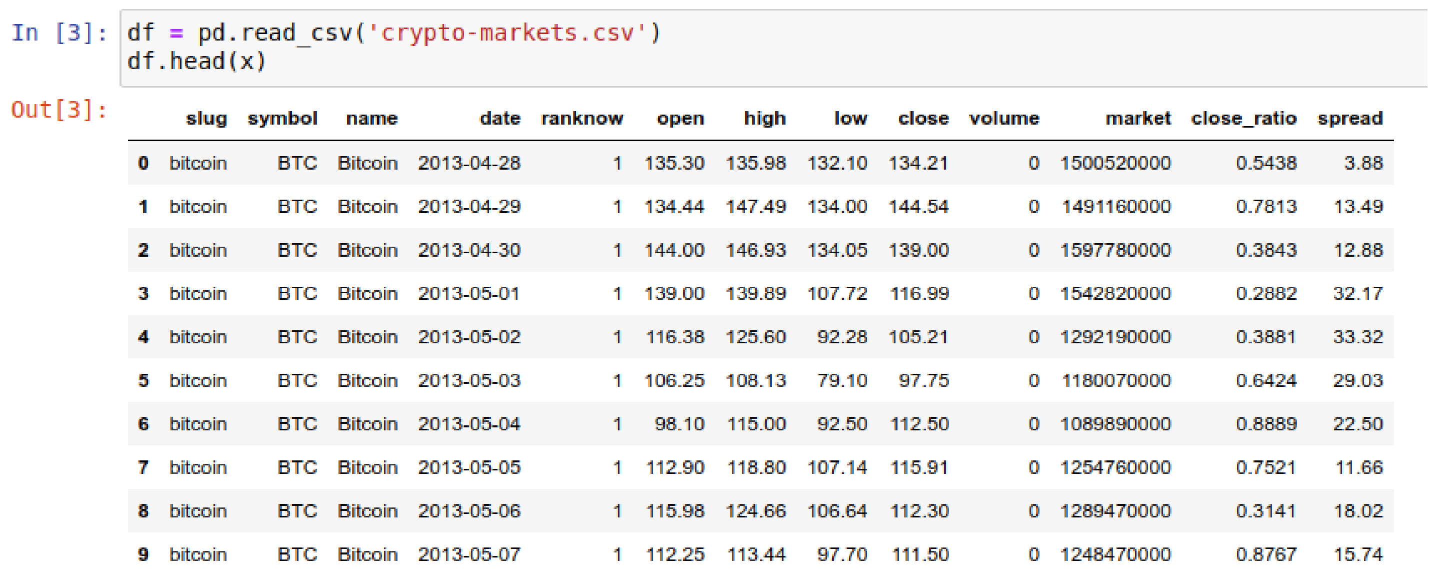
In the dataset, we have been provided with data of many cryptocurrencies, from dated 2013 to 2018. In this data we get to know a lot about various cryptocurrencies, that will help us in understanding the ranking of the currencies and the best crypto to trade on.
Research Paper regarding this:

Time-Series Prediction of Cryptocurrency Market using Machine Learning Techniques

- July 2021
- EAI Endorsed Transactions on Creative Technologies 4(4):6

DOI:10.4108/eai.7-7-2021.170286

The Dataset:



```
In [3]: df = pd.read_csv('crypto-markets.csv')
        df.head(x)
```

Out[3]:

	slug	symbol	name	date	ranknow	open	high	low	close	volume	market	close_ratio	spread
0	bitcoin	BTC	Bitcoin	2013-04-28	1	135.30	135.98	132.10	134.21	0	1500520000	0.5438	3.88
1	bitcoin	BTC	Bitcoin	2013-04-29	1	134.44	147.49	134.00	144.54	0	1491160000	0.7813	13.49
2	bitcoin	BTC	Bitcoin	2013-04-30	1	144.00	146.93	134.05	139.00	0	1597780000	0.3843	12.88
3	bitcoin	BTC	Bitcoin	2013-05-01	1	139.00	139.89	107.72	116.99	0	1542820000	0.2882	32.17
4	bitcoin	BTC	Bitcoin	2013-05-02	1	116.38	125.60	92.28	105.21	0	1292190000	0.3881	33.32
5	bitcoin	BTC	Bitcoin	2013-05-03	1	106.25	108.13	79.10	97.75	0	1180070000	0.6424	29.03
6	bitcoin	BTC	Bitcoin	2013-05-04	1	98.10	115.00	92.50	112.50	0	1089890000	0.8889	22.50
7	bitcoin	BTC	Bitcoin	2013-05-05	1	112.90	118.80	107.14	115.91	0	1254760000	0.7521	11.66
8	bitcoin	BTC	Bitcoin	2013-05-06	1	115.98	124.66	106.64	112.30	0	1289470000	0.3141	18.02
9	bitcoin	BTC	Bitcoin	2013-05-07	1	112.25	113.44	97.70	111.50	0	1248470000	0.8767	15.74

Additional Information:

```
In [4]: df.isnull().sum()

Out[4]: slug               0
        symbol             0
        name               0
        date               0
        ranknow            0
        open               0
        high               0
        low                0
        close              0
        volume             0
        market             0
        close_ratio    13257
        spread             0
        dtype: int64
```

```
In [5]: df.info()

        <class 'pandas.core.frame.DataFrame'>
        RangeIndex: 679183 entries, 0 to 679182
        Data columns (total 13 columns):
         #   Column       Non-Null Count   Dtype  
        ---  ------       --------------   -----  
         0   slug         679183 non-null  object 
         1   symbol       679183 non-null  object 
         2   name         679183 non-null  object 
         3   date         679183 non-null  object 
         4   ranknow      679183 non-null  int64  
         5   open         679183 non-null  float64
         6   high         679183 non-null  float64
         7   low          679183 non-null  float64
         8   close        679183 non-null  float64
         9   volume       679183 non-null  int64  
         10  market       679183 non-null  int64  
         11  close_ratio  665926 non-null  float64
         12  spread       679183 non-null  float64
        dtypes: float64(6), int64(3), object(4)
        memory usage: 67.4+ MB
```

```
In [8]: # How many various cryptocurrency informations are present on the dataset
        df['symbol'].value_counts()
```

```
Out[8]: BTC    1745
        NMC    1745
        LTC    1745
        PPC    1744
        NVC    1744
               ...
        ADB       3
        DMT       3
        ZLA       2
        ING       2
        JNT       1
        Name: symbol, Length: 1479, dtype: int64
```

```
In [9]: list_crypto = df['symbol'].value_counts()
        max_crypto = np.where(list_crypto == list_crypto.max(), df['symbol'].unique(), 'nil')
        print("The coins with maximum data")
        for coin in max_crypto:
            if coin != 'nil':
                print(coin)
```

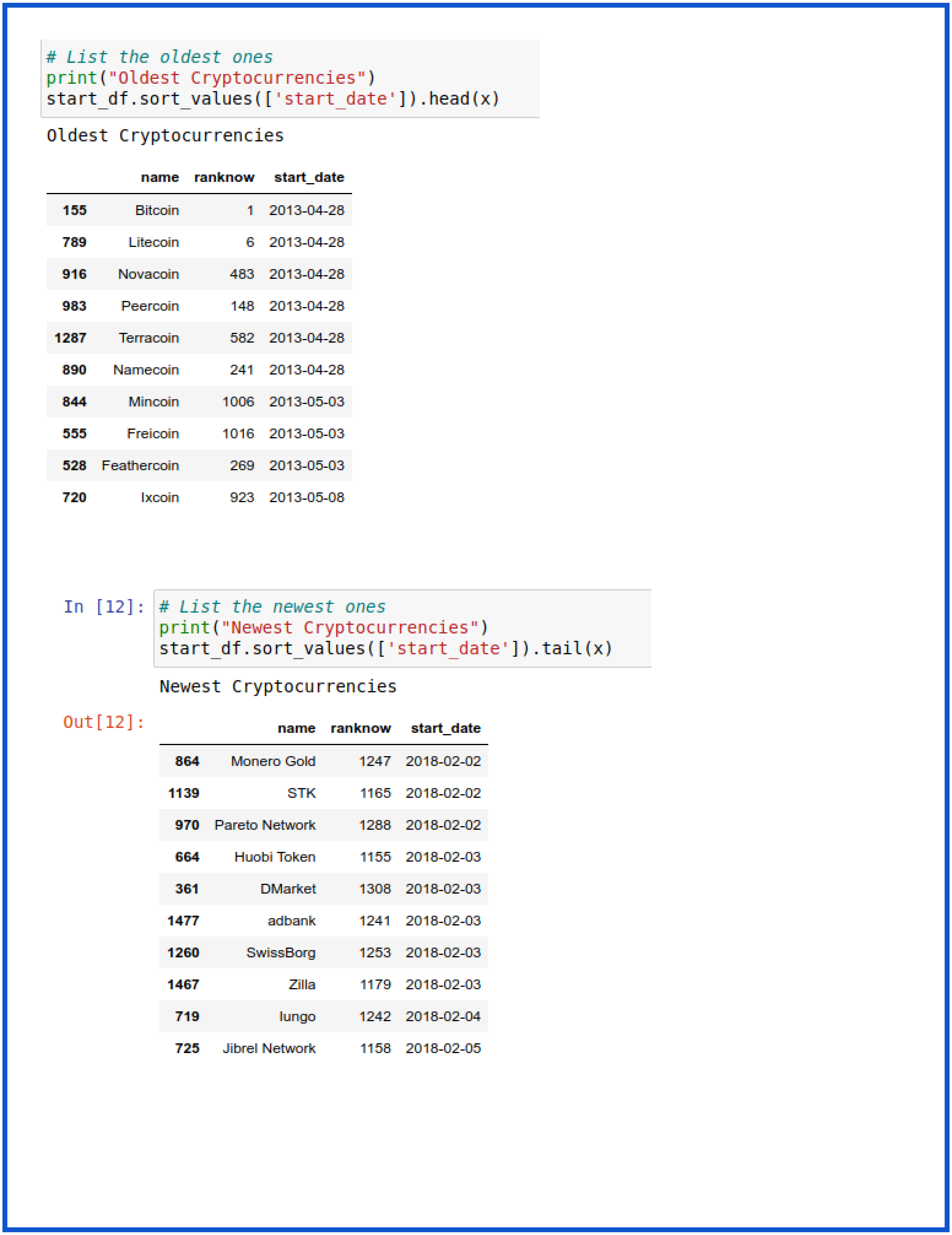
```
        The coins with maximum data
        BTC
        ETH
        XRP
```

```
In [10]: # Get the oldest and the newest cryptocurrencies
         start_df = pd.DataFrame({'start_date' : df.groupby( [ "name", "ranknow"] )['date'].min()}).reset_index()
         start_df
```

Out[10]:

	name	ranknow	start_date
0	0x	40	2017-08-16
1	10M Token	1424	2017-10-24
2	2GIVE	565	2016-05-16
3	300 Token	841	2017-07-25
4	42-coin	615	2014-01-14
...	...	...	...
1508	netBit	1442	2016-11-14
1509	onG.social	470	2017-12-01
1510	ugChain	1207	2018-01-13
1511	vSlice	922	2016-12-19
1512	vTorrent	504	2014-12-25

1513 rows × 3 columns



```
# List the oldest ones
print("Oldest Cryptocurrencies")
start_df.sort_values(['start_date']).head(x)
```

```
Oldest Cryptocurrencies
```

	name	ranknow	start_date
155	Bitcoin	1	2013-04-28
789	Litecoin	6	2013-04-28
916	Novacoin	483	2013-04-28
983	Peercoin	148	2013-04-28
1287	Terracoin	582	2013-04-28
890	Namecoin	241	2013-04-28
844	Mincoin	1006	2013-05-03
555	Freicoin	1016	2013-05-03
528	Feathercoin	269	2013-05-03
720	Ixcoin	923	2013-05-08

In [12]:

```
# List the newest ones
print("Newest Cryptocurrencies")
start_df.sort_values(['start_date']).tail(x)
```

```
Newest Cryptocurrencies
```

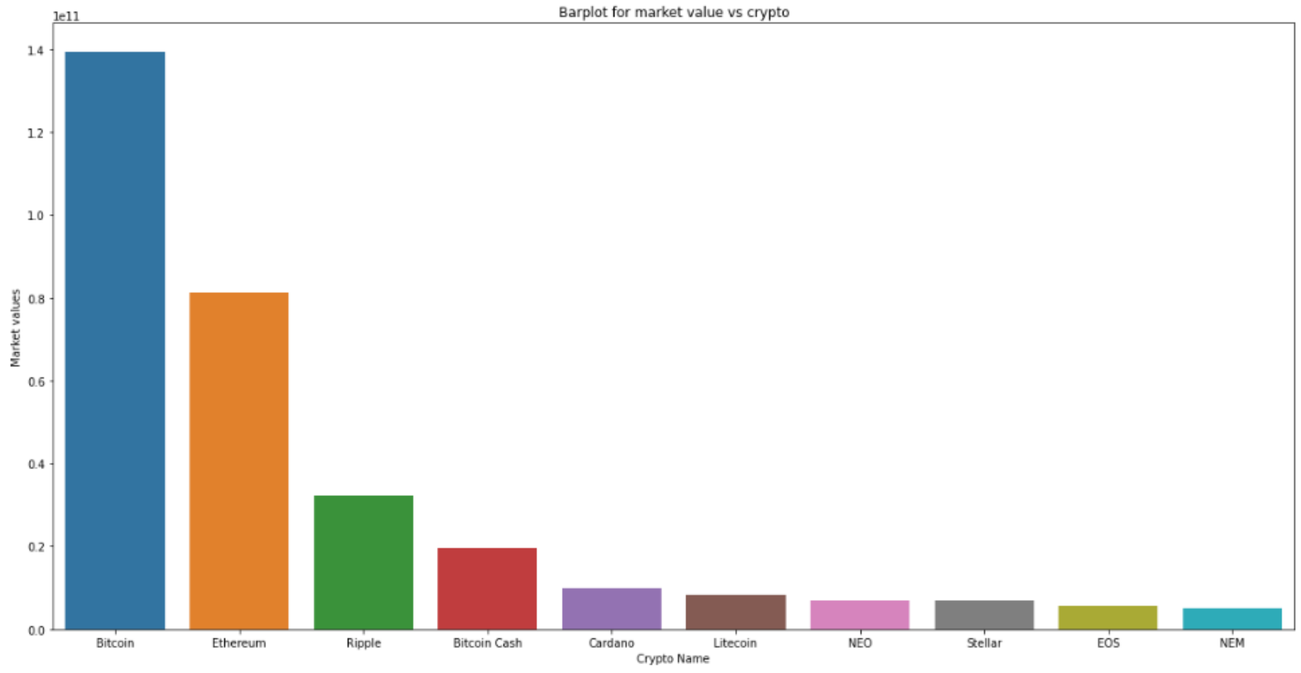
Out[12]:

	name	ranknow	start_date
864	Monero Gold	1247	2018-02-02
1139	STK	1165	2018-02-02
970	Pareto Network	1288	2018-02-02
664	Huobi Token	1155	2018-02-03
361	DMarket	1308	2018-02-03
1477	adbank	1241	2018-02-03
1260	SwissBorg	1253	2018-02-03
1467	Zilla	1179	2018-02-03
719	Iungo	1242	2018-02-04
725	Jibrel Network	1158	2018-02-05

```
In [15]: # Get the list of top 10 ranked cryptocurrencies
         latest_df[latest_df['ranknow'] <= x].groupby('ranknow').name.unique()

Out[15]: ranknow
         1           [Bitcoin]
         2          [Ethereum]
         3            [Ripple]
         4      [Bitcoin Cash]
         5           [Cardano]
         6          [Litecoin]
         7               [NEO]
         8           [Stellar]
         9               [EOS]
         10              [NEM]
         Name: name, dtype: object
```

Benchmarking

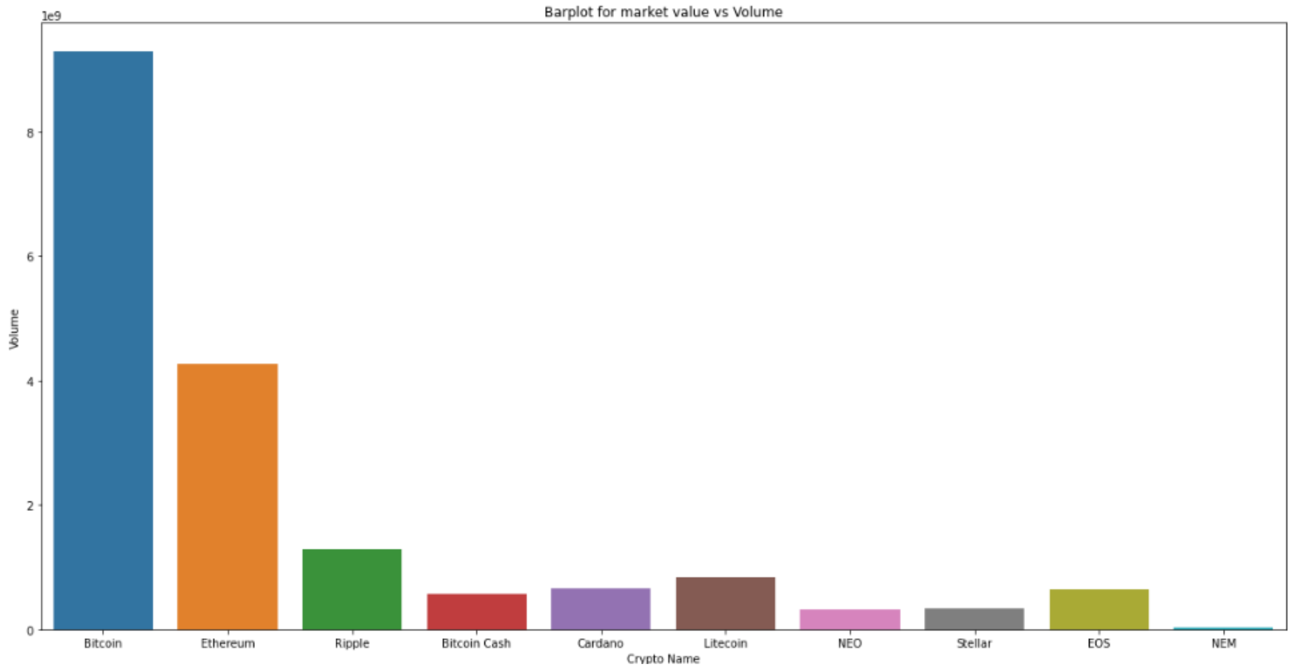
```
In [17]: plt.figure(figsize=(20,10))
         sns.barplot(x=latest_df['name'][:x], y=latest_df['market'][:x])
         plt.xlabel('Crypto Name')
         plt.ylabel('Market values')
         plt.title('Barplot for market value vs crypto')

Out[17]: Text(0.5, 1.0, 'Barplot for market value vs crypto')
```



Volume

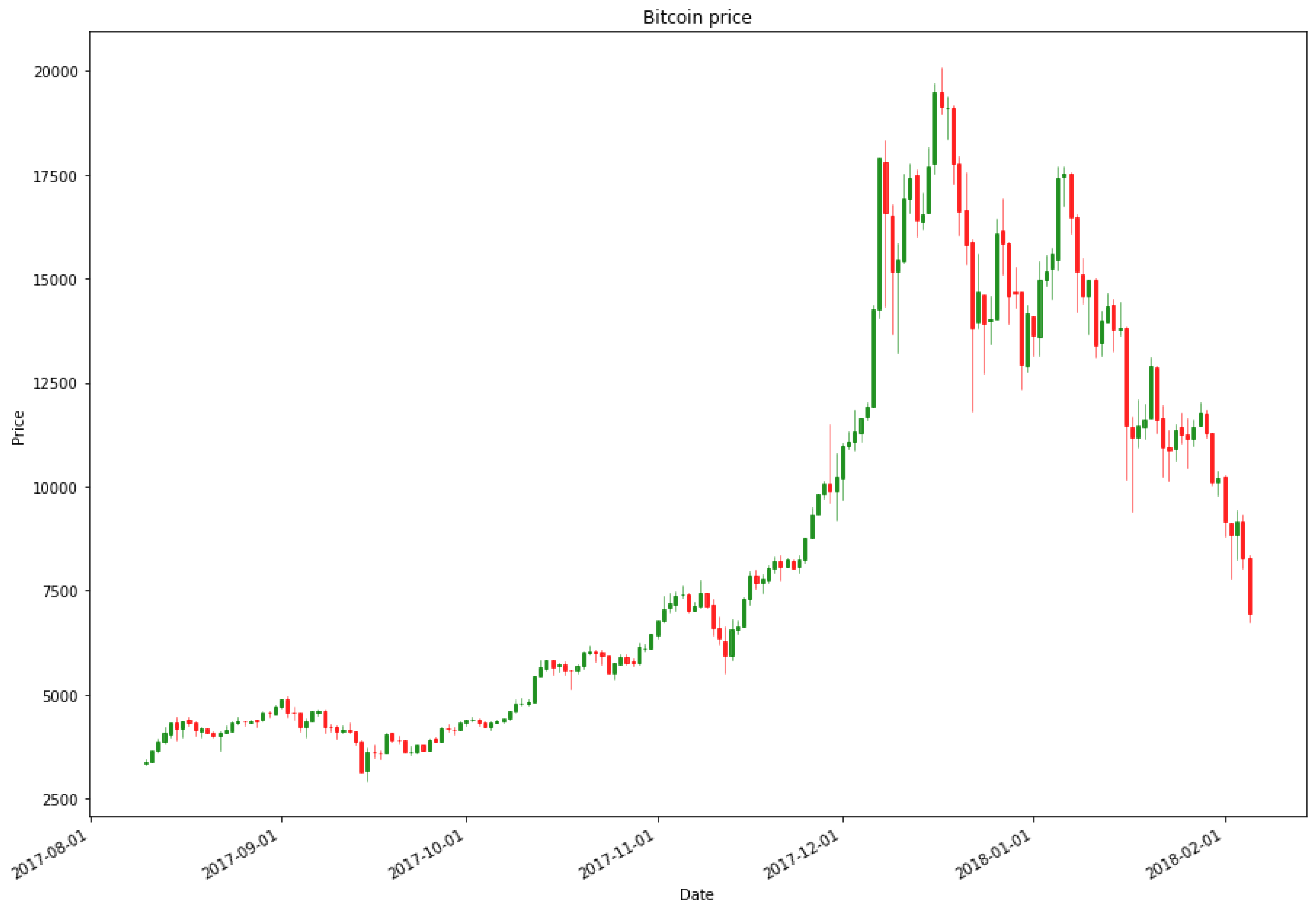
Volume is a measure of how much of a given financial asset has been traded in a given period of time and even though so simple, it can be a powerful indicator for trading.

```
In [18]: plt.figure(figsize=(20,10))
         sns.barplot(x=latest_df['name'][:x], y=latest_df['volume'][:x])
         plt.xlabel('Crypto Name')
         plt.ylabel('Volume')
         plt.title('Barplot for market value vs Volume')

Out[18]: Text(0.5, 1.0, 'Barplot for market value vs Volume')
```



As we can see, the volume of Bitcoin is the highest, and thus the data regarding bitcoin is the highest. Crypto currency has evolved a lot since its advent, and this type of digital currency can help us grow in many ways.

Candle Stick Graph

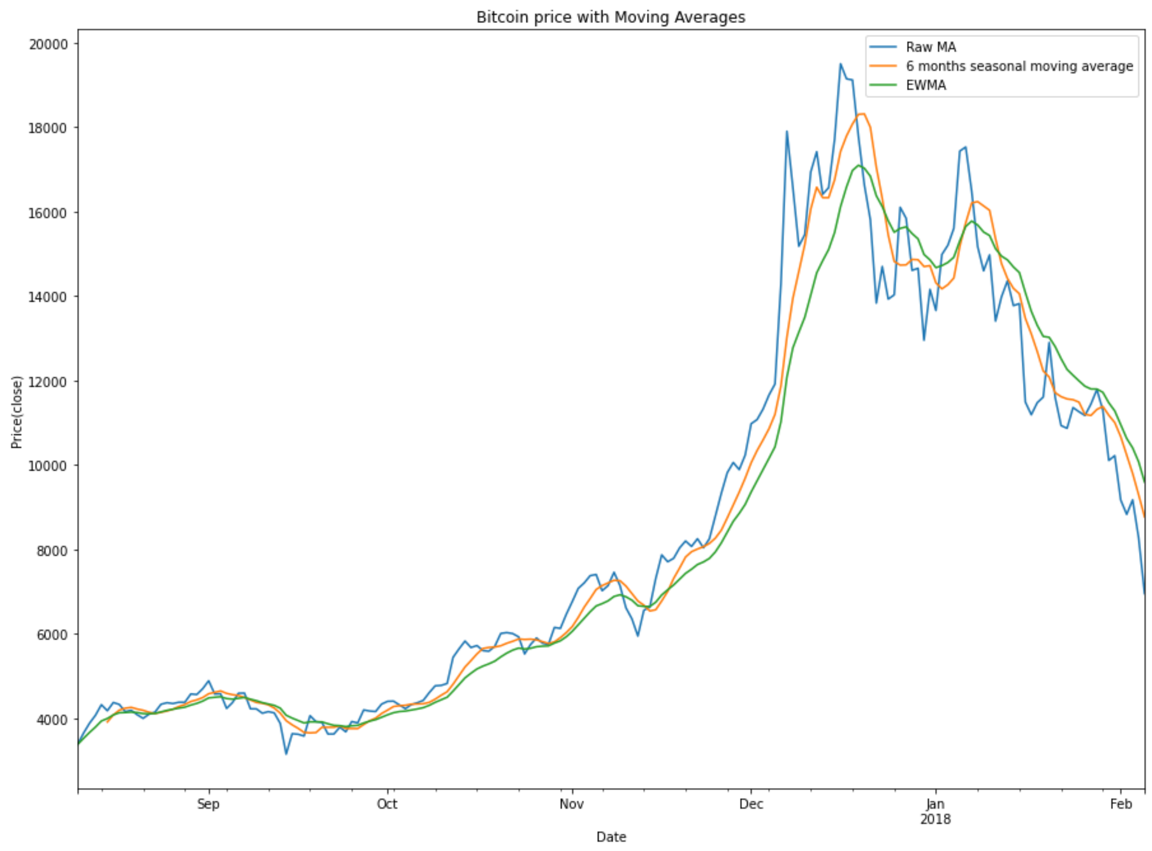


The above graph is limited to 6 months

Traders use candlestick charts to forecast price movement based on previous patterns. Candlesticks are important in trading because they display four price points (open, close, high, and low) across the time period specified by the trader.

Moving Averages

Now let's look at how we can spot stock trends. Moving averages are one of the most basic and oldest trading "tools" available. A n-day moving average is the average of the previous n days for a series and a point in time. Moving averages smooth out a series and aid in the detection of patterns. A moving average procedure is less susceptible to short-term variations in the series as n increases. The premise is that moving average procedures can assist separate "noise" from patterns. The Exponentially Weighted Moving Average (EWMA) is a statistic for monitoring a process that averages data in a way that provides less and less weight to data as it gets further away in time.



Application Regulation

If the algorithms are not created and optimised specifically and for our needs, the patents indicated above may claim the technology. If a patent claim is filed, using a pre-existing model is out.

We need to consistently look at our model and keep the dataset updated, the model parameters also will be updated with the advent of better algorithms and will create newer methods for predicting values.

Applicable Constraints

1. Data checking and updation
2. The model needs to be updated based on the dataset
3. Market knowledge and trends that is extraordinary

Business Opportunity

1. Crypto lending is a one-of-a-kind side-hustle business option that is thriving in today's economy. It will yield more profits than holding and exchanging bitcoin.
   Cryptocurrency investments are becoming increasingly popular, and if you have a cryptocurrency exchange account, you may start the crypto loaning process. You may make money by lending your bitcoin or other currencies to others who engage in speculative trading or other types of trade. Traders can benefit from such possibilities on a number of bitcoin trading platforms. As a consequence, pick the lowest interest rate possible to attract more investors.

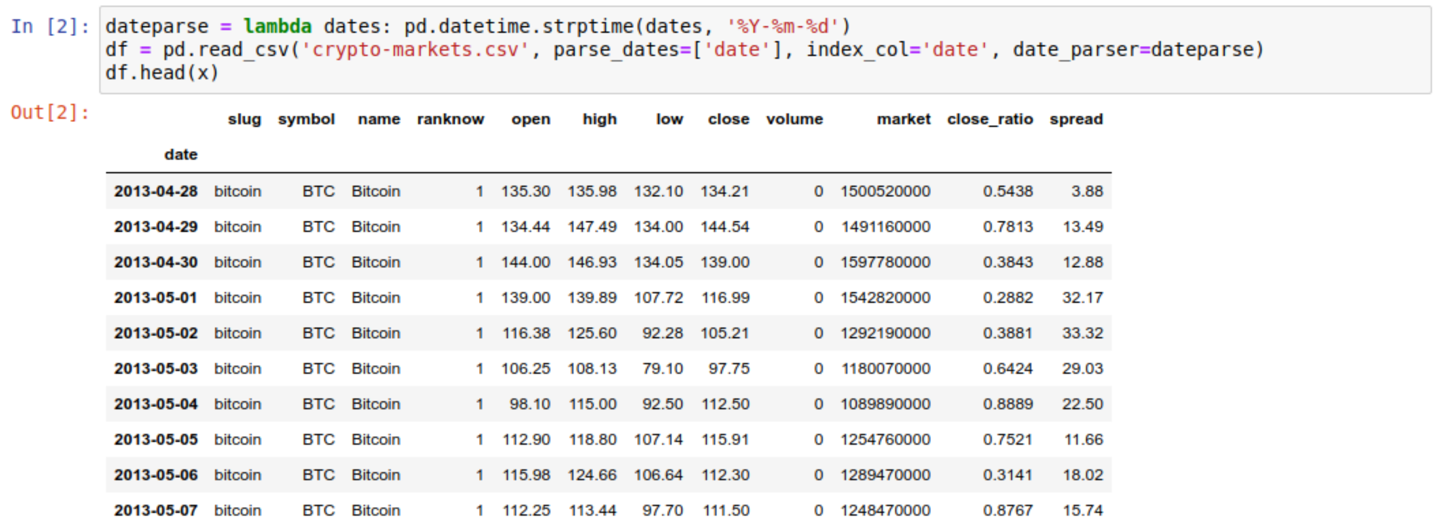
2. Crypto Payment Gateway: Many businesses have adopted bitcoin payments by incorporating a cryptocurrency payment gateway into their POS terminals and delivery check-ins, such as Walmart and Amazon. Businesses want a trustworthy and dependable multi-bitcoin payment system to succeed in the cryptocurrency payment gateway sector. What if you could provide them one-of-a-kind bitcoin payment gateway solutions at a low cost? Establish yourself as a crypto entrepreneur by starting your bitcoin payment gateway

firm.

3. CryptoCurrency ATM: Many businesses have adopted bitcoin payments by incorporating a cryptocurrency payment gateway into their POS terminals and delivery check-ins, such as Walmart and Amazon. Businesses want a trustworthy and dependable multi-bitcoin payment system to succeed in the cryptocurrency payment gateway sector. What if you could provide them one-of-a-kind bitcoin payment gateway solutions at a low cost? Establish yourself as a crypto entrepreneur by starting your bitcoin payment gateway firm.

Concept Generation and Development

The ARIMA model is a famous and commonly used statistical approach for time series forecasting. One of the most often used models for predicting linear time series data is this one. This model has been widely utilised in banking and economics since it is known to be reliable, efficient, and capable of predicting short-term share market movements. The two most generally used techniques to time series forecasting are exponential smoothing and ARIMA models, which give complimentary approaches to the problem. While exponential smoothing methods try to explain the data's trend and seasonality, ARIMA models strive to characterise the data's auto-correlation.

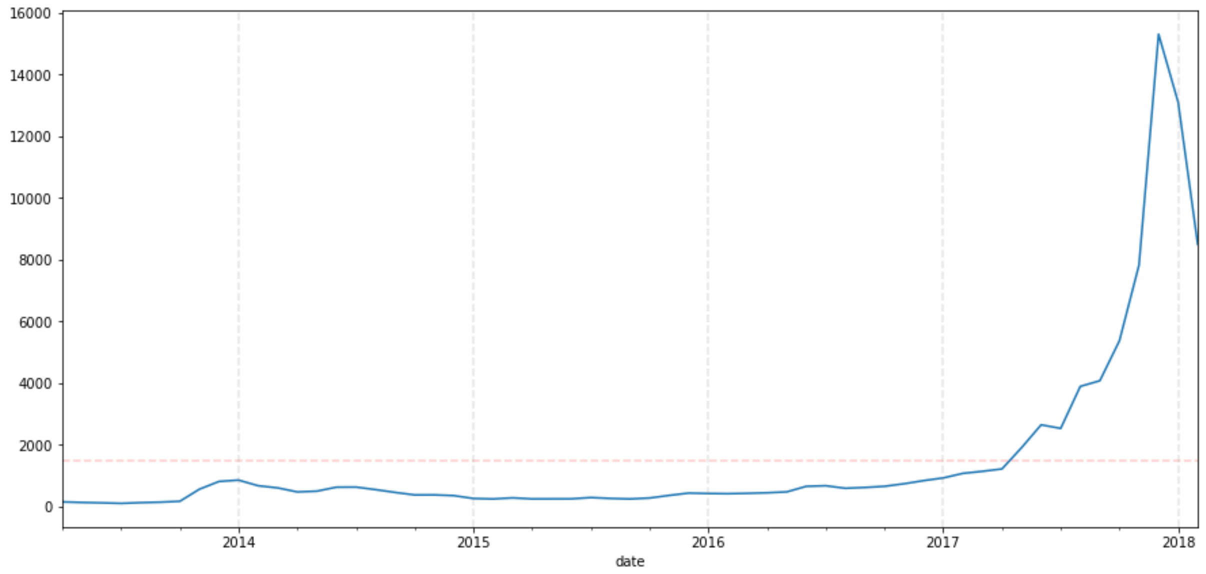


```
In [2]: dateparse = lambda dates: pd.datetime.strptime(dates, '%Y-%m-%d')
        df = pd.read_csv('crypto-markets.csv', parse_dates=['date'], index_col='date', date_parser=dateparse)
        df.head(x)
Out[2]:
```

date	slug	symbol	name	ranknow	open	high	low	close	volume	market	close_ratio	spread
2013-04-28	bitcoin	BTC	Bitcoin	1	135.30	135.98	132.10	134.21	0	1500520000	0.5438	3.88
2013-04-29	bitcoin	BTC	Bitcoin	1	134.44	147.49	134.00	144.54	0	1491160000	0.7813	13.49
2013-04-30	bitcoin	BTC	Bitcoin	1	144.00	146.93	134.05	139.00	0	1597780000	0.3843	12.88
2013-05-01	bitcoin	BTC	Bitcoin	1	139.00	139.89	107.72	116.99	0	1542820000	0.2882	32.17
2013-05-02	bitcoin	BTC	Bitcoin	1	116.38	125.60	92.28	105.21	0	1292190000	0.3881	33.32
2013-05-03	bitcoin	BTC	Bitcoin	1	106.25	108.13	79.10	97.75	0	1180070000	0.6424	29.03
2013-05-04	bitcoin	BTC	Bitcoin	1	98.10	115.00	92.50	112.50	0	1089890000	0.8889	22.50
2013-05-05	bitcoin	BTC	Bitcoin	1	112.90	118.80	107.14	115.91	0	1254760000	0.7521	11.66
2013-05-06	bitcoin	BTC	Bitcoin	1	115.98	124.66	106.64	112.30	0	1289470000	0.3141	18.02
2013-05-07	bitcoin	BTC	Bitcoin	1	112.25	113.44	97.70	111.50	0	1248470000	0.8767	15.74

Plot Close vs Date:

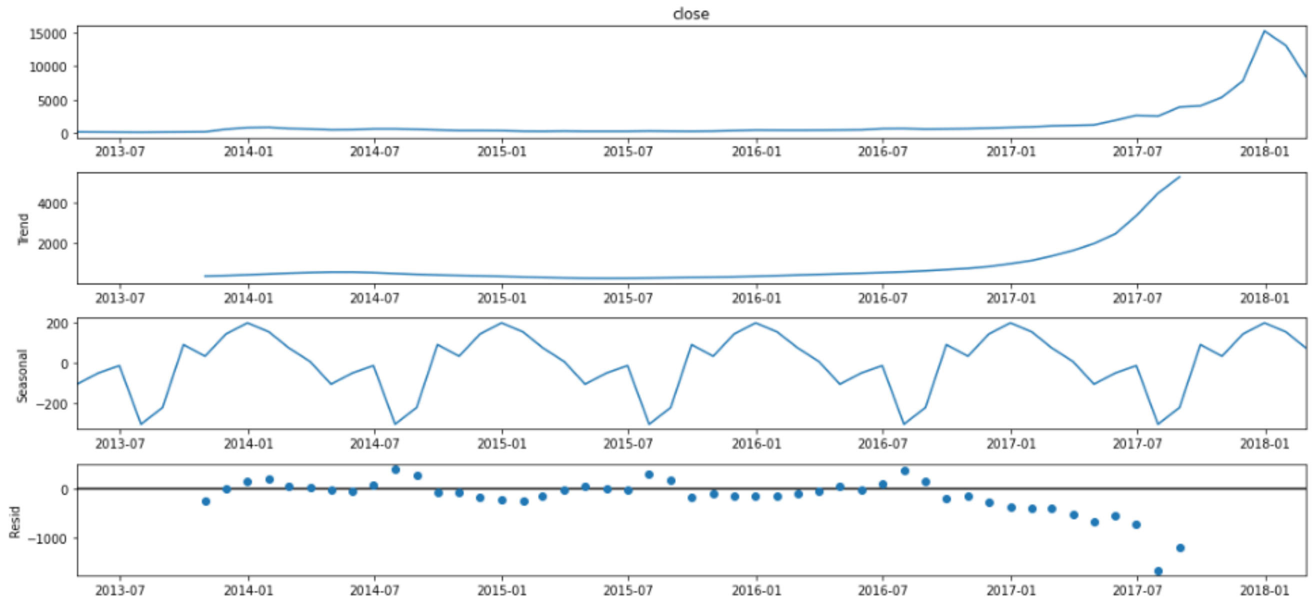
```
In [6]: btc_month.close.plot()
        plt.axhline(btc_month.close.mean(), color='r', alpha=0.2, linestyle='--')

        for year in range(2014, 2019):
            plt.axvline(pd.to_datetime(str(year)+'-01-01'), color='gray', alpha=0.2, linestyle='--')
```



Dickey-Fuller test for close values

```
In [7]: seasonal_decompose(btc_month.close).plot()


        print("Dickey–Fuller test: p=%f" % adfuller(btc_month.close)[1])


        if(adfuller(btc_month.close)[1] <= 0.05):
            print("strong evidence aginst null hypothesis, the data is stationary")
        else:
            print("weak evidence aginst null hypothesis, time series has a unit root, the data is non stationary")
        plt.show()
```

```
Dickey–Fuller test: p=0.998803
weak evidence aginst null hypothesis, time series has a unit root, the data is non stationary
```



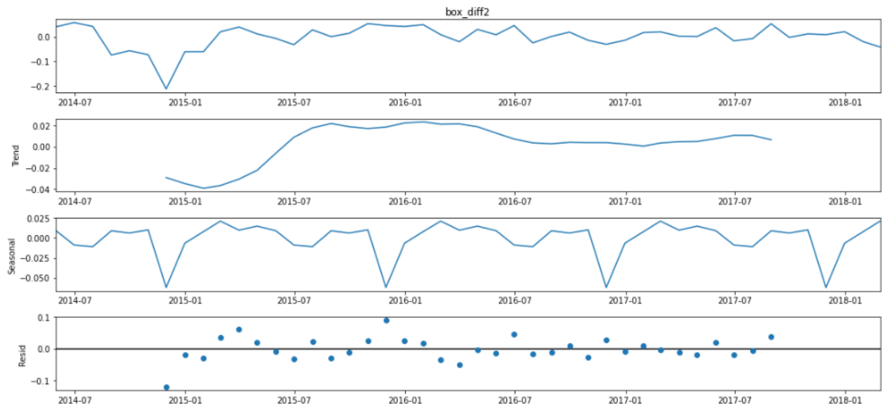
Transformations to make data stationary

```
In [8]: # Box-Cox Transformations
        btc_month['close_box'], lmbda = stats.boxcox(btc_month.close)


        print("Dickey–Fuller test: p=%f" % adfuller(btc_month.close_box)[1])


        if(adfuller(btc_month.close_box)[1] <= 0.05):
            print("strong evidence aginst null hypothesis, the data is stationary")
        else:
            print("weak evidence aginst null hypothesis, time series has a unit root, the data is non stationary")

        Dickey–Fuller test: p=0.494475
        weak evidence aginst null hypothesis, time series has a unit root, the data is non stationary
```

```
In [9]: # Seasonal differentiation (12 months)
        btc_month['box_diff_seasonal_12'] = btc_month.close_box - btc_month.close_box.shift(12)


        print("Dickey–Fuller test: p=%f" % adfuller(btc_month.box_diff_seasonal_12[12:])[1])


        if(adfuller(btc_month.box_diff_seasonal_12[12:])[1] <= 0.05):
            print("strong evidence aginst null hypothesis, the data is stationary")
        else:
            print("weak evidence aginst null hypothesis, time series has a unit root, the data is non stationary")

        Dickey–Fuller test: p=0.661446
        weak evidence aginst null hypothesis, time series has a unit root, the data is non stationary
```

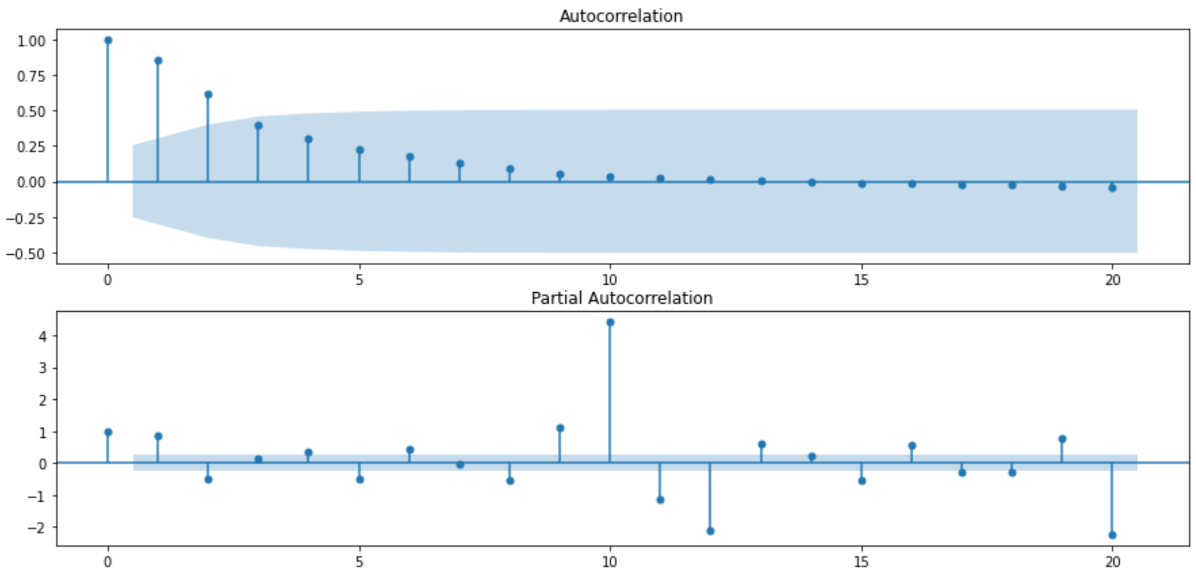
```
In [10]: btc_month['box_diff_seasonal_3'] = btc_month.close_box - btc_month.close_box.shift(3)


         print("Dickey–Fuller test: p=%f" % adfuller(btc_month.box_diff_seasonal_3[3:])[1])


         if(adfuller(btc_month.box_diff_seasonal_3[3:])[1] < 0.05):
             print("strong evidence aginst null hypothesis, the data is stationary")
         else:
             print("weak evidence aginst null hypothesis, time series has a unit root, the data is non stationary")

         Dickey–Fuller test: p=0.018033
         strong evidence aginst null hypothesis, the data is stationary
```

```
In [11]: # Regular differentiation
         btc_month['box_diff2'] = btc_month.box_diff_seasonal_12 - btc_month.box_diff_seasonal_12.shift(1)

         # STL-decomposition
         seasonal_decompose(btc_month.box_diff2[13:]).plot()
         print("Dickey–Fuller test: p=%f" % adfuller(btc_month.box_diff2[13:])[1])

         if(adfuller(btc_month.box_diff2[13:])[1] < 0.05):
             print("strong evidence aginst null hypothesis, the data is stationary")
         else:
             print("weak evidence aginst null hypothesis, time series has a unit root, the data is non stationary")

         Dickey–Fuller test: p=0.002424
         strong evidence aginst null hypothesis, the data is stationary
```

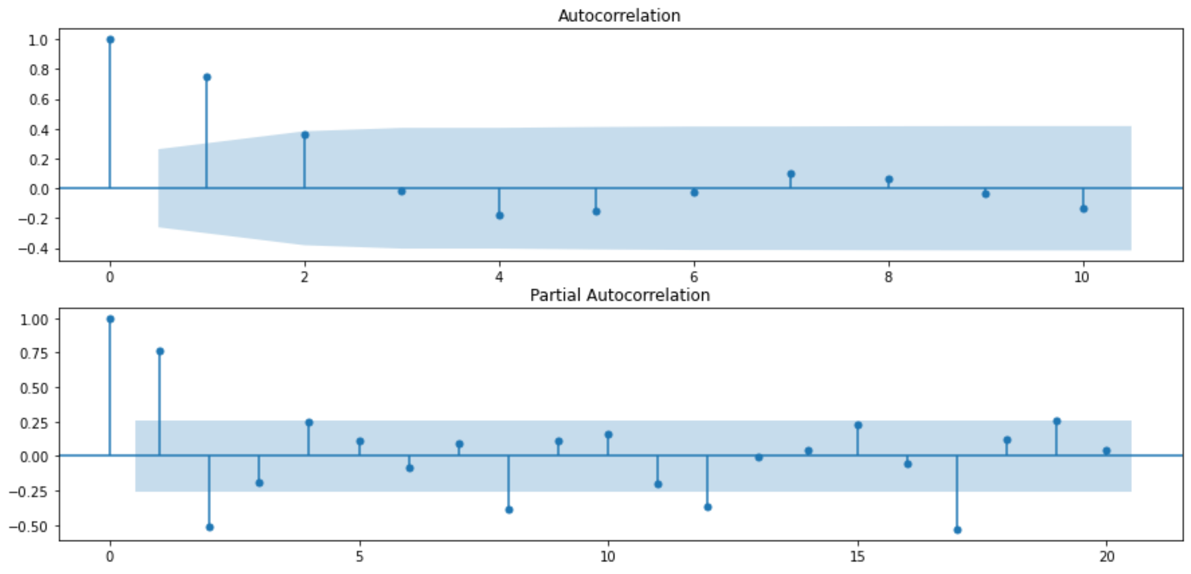


PACF and ACF plots

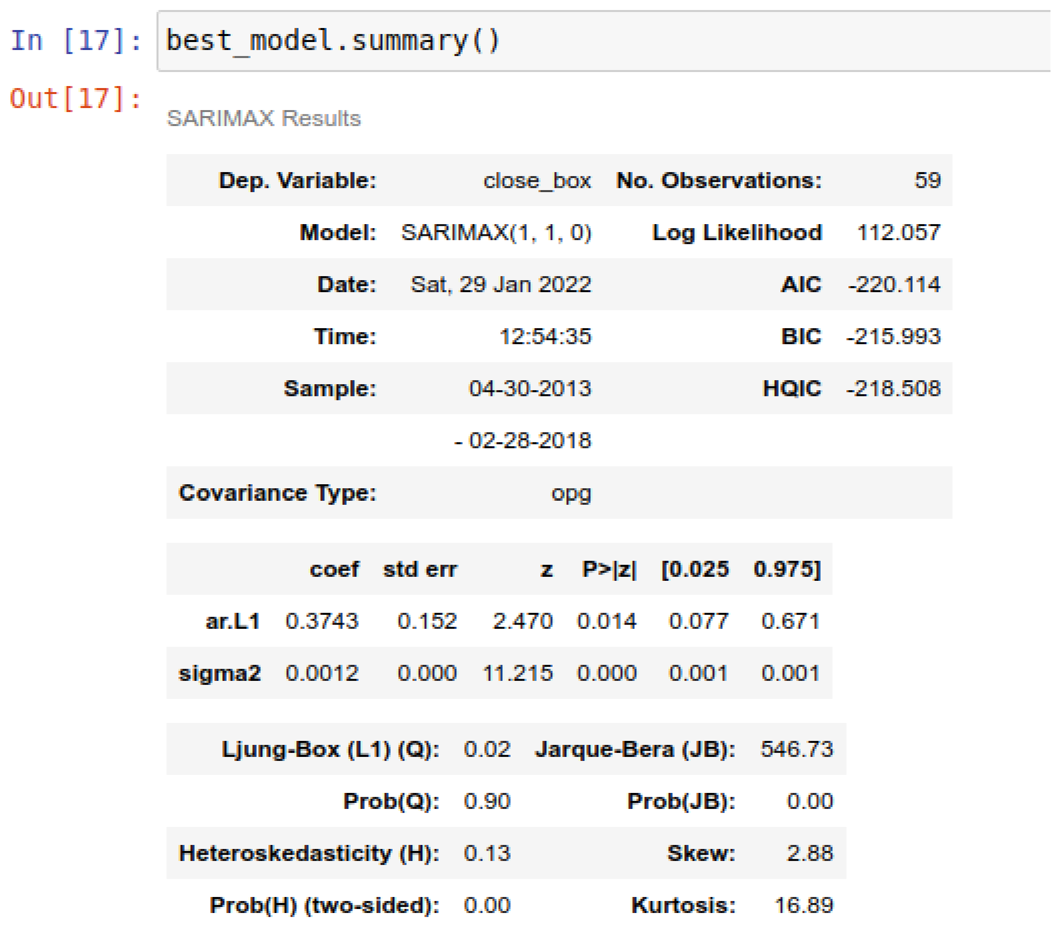
In [12]:
```
ax = plt.subplot(211)
plot_acf(btc_month.close, lags=20, ax=ax)


ax = plt.subplot(212)
plot_pacf(btc_month.close, lags=20, ax=ax)
plt.show()
```



In [13]:
```
ax = plt.subplot(211)
plot_acf(btc_month.box_diff_seasonal_3[3:], lags=10, ax=ax)


ax = plt.subplot(212)
plot_pacf(btc_month.box_diff_seasonal_3[3:], lags=20, ax=ax)
plt.show()
```



ARIMA MODEL



```
In [17]: best_model.summary()
```

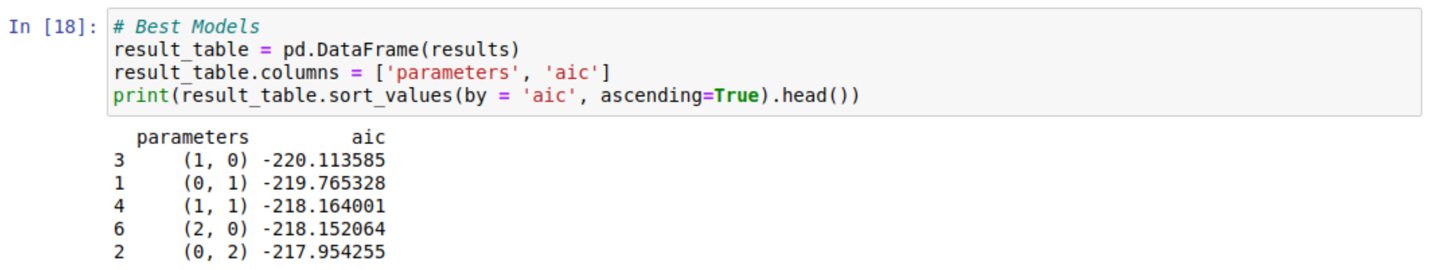
Out[17]:

SARIMAX Results

Dep. Variable:	close_box	**No. Observations:**	59
Model:	SARIMAX(1, 1, 0)	**Log Likelihood**	112.057
Date:	Sat, 29 Jan 2022	**AIC**	-220.114
Time:	12:54:35	**BIC**	-215.993
Sample:	04-30-2013	**HQIC**	-218.508
	- 02-28-2018		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	0.3743	0.152	2.470	0.014	0.077	0.671
sigma2	0.0012	0.000	11.215	0.000	0.001	0.001

Ljung-Box (L1) (Q):	0.02	**Jarque-Bera (JB):**	546.73
Prob(Q):	0.90	**Prob(JB):**	0.00
Heteroskedasticity (H):	0.13	**Skew:**	2.88
Prob(H) (two-sided):	0.00	**Kurtosis:**	16.89

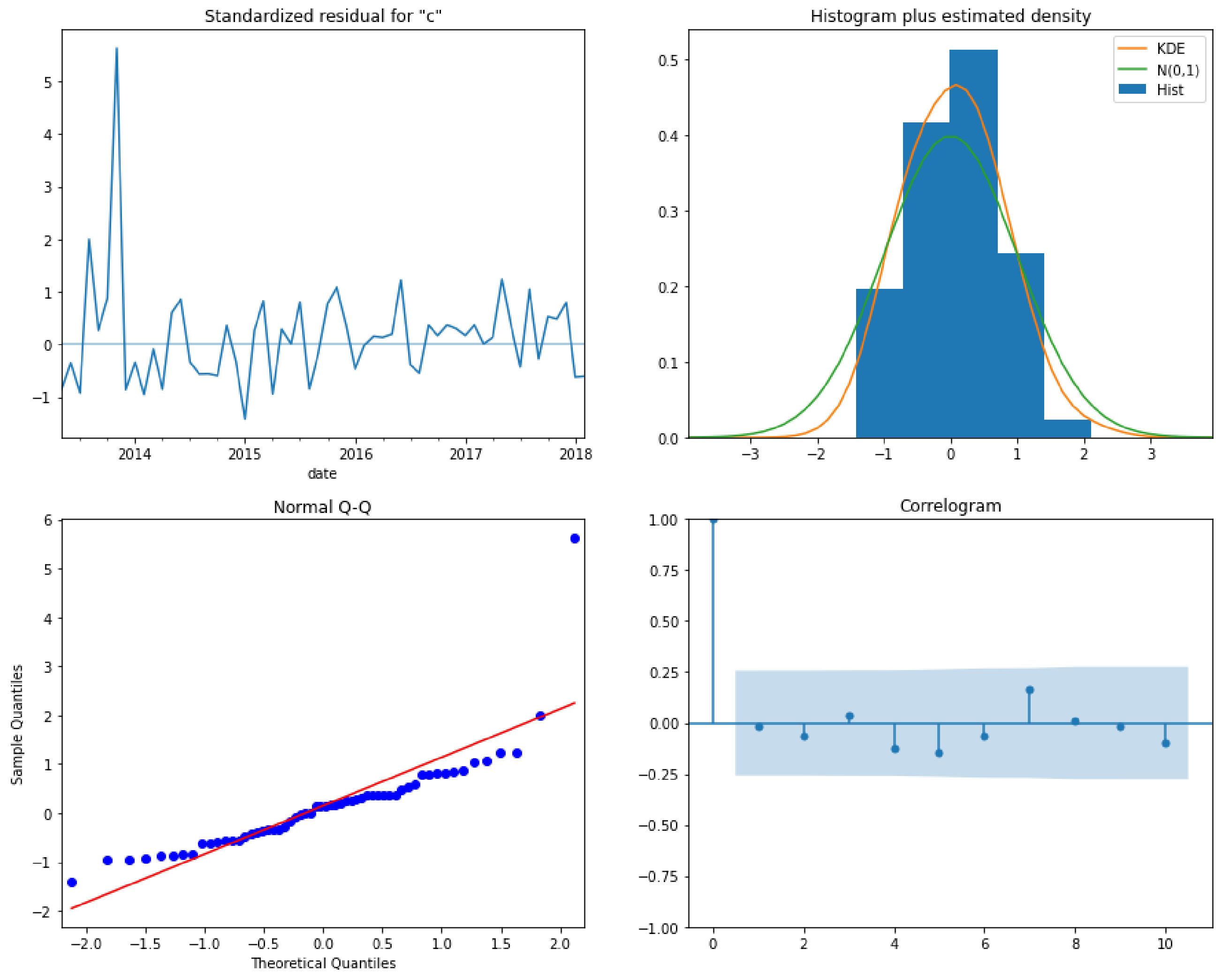


```
In [18]: # Best Models
         result_table = pd.DataFrame(results)
         result_table.columns = ['parameters', 'aic']
         print(result_table.sort_values(by = 'aic', ascending=True).head())
```

```
  parameters         aic
3     (1, 0) -220.113585
1     (0, 1) -219.765328
4     (1, 1) -218.164001
6     (2, 0) -218.152064
2     (0, 2) -217.954255
```

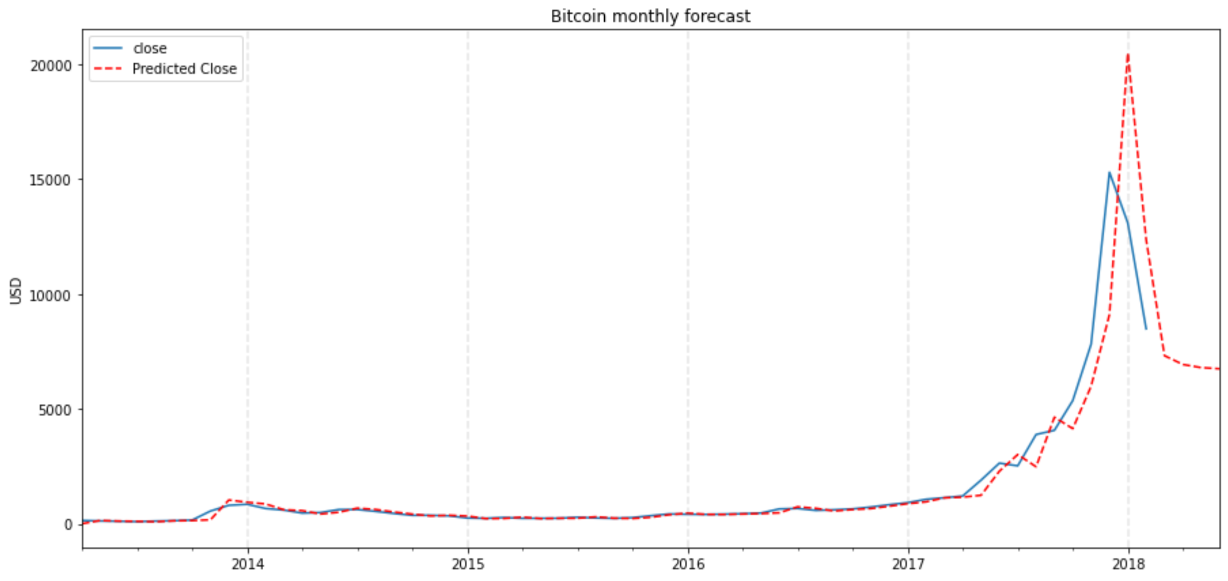
RESIDUALS plot for verifications

```
best_model.plot_diagnostics(figsize=(15, 12))
plt.show()
```



Prediction Plots for the future

```
In [25]: plt.figure(figsize=(15,7))
         prediction_data.close.plot()
         prediction_data.pred_close.plot(color='r', ls='--', label='Predicted Close')
         for year in range(2014,2019):
             plt.axvline(pd.to_datetime(str(year)+'-01-01'), color='gray',alpha=0.2, ls='--')
         plt.legend()
         plt.title('Bitcoin monthly forecast')
         plt.ylabel('USD')
         plt.show()
```



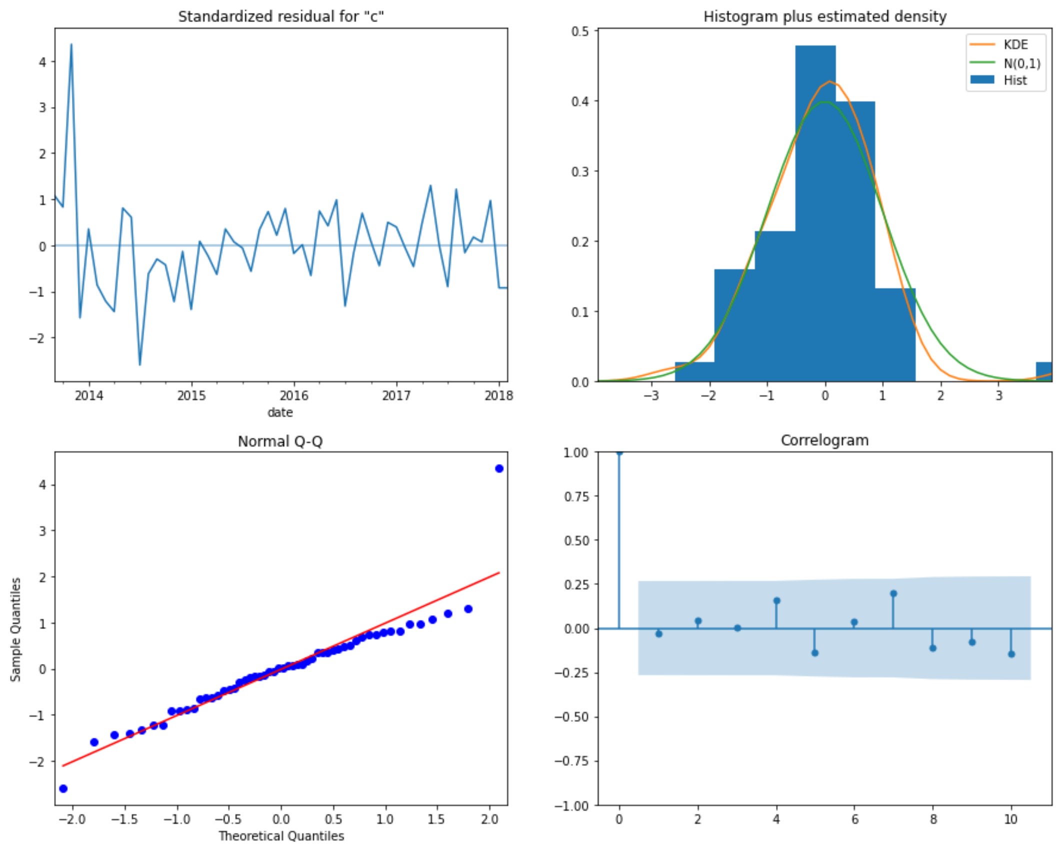
```
In [26]: y_forcasted = prediction_data[datetime(2013,8,31):datetime(2018,2,28)].pred_close
         y_truth = prediction_data[datetime(2013,8,31):datetime(2018,2,28)].close

         rmse = np.sqrt(((y_forcasted - y_truth) ** 2).mean())
         print('Mean Squared Error: {}'.format(round(rmse, 6)))

         Mean Squared Error: 1455.934514
```

SARIMAX Model

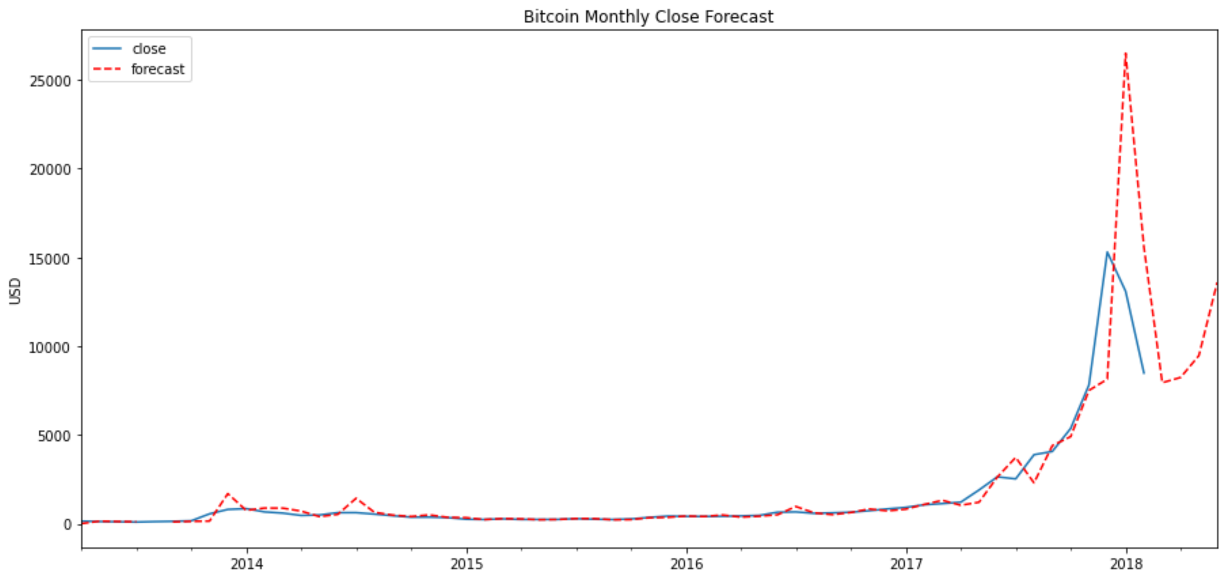
In [29]:
```
# Best Models
result_table = pd.DataFrame(results)
result_table.columns = ['parameters', 'aic']
print(result_table.sort_values(by = 'aic', ascending=True).head())
print(best_model.summary())
```

```
      parameters         aic
21  (1, 0, 1, 1) -193.013122
49  (2, 2, 0, 1) -192.919337
19  (1, 0, 0, 1) -192.627054
7   (0, 1, 0, 1) -192.226372
9   (0, 1, 1, 1) -192.174973
                                     SARIMAX Results                                     
=========================================================================================
Dep. Variable:                          close_box   No. Observations:                   59
Model:             SARIMAX(1, 1, 0)x(1, 1, [1], 4)   Log Likelihood                 100.507
Date:                            Sat, 29 Jan 2022   AIC                           -193.013
Time:                                    12:55:04   BIC                           -185.057
Sample:                                04-30-2013   HQIC                          -189.945
                                     - 02-28-2018                                         
Covariance Type:                              opg                                         
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.3773      0.129      2.934      0.003       0.125       0.629
ar.S.L4       -0.3083      0.087     -3.560      0.000      -0.478      -0.139
ma.S.L4       -0.7753      0.217     -3.566      0.000      -1.201      -0.349
sigma2         0.0013      0.000      4.768      0.000       0.001       0.002
===================================================================================
Ljung-Box (L1) (Q):                   0.05   Jarque-Bera (JB):                72.60
Prob(Q):                              0.82   Prob(JB):                         0.00
Heteroskedasticity (H):               0.21   Skew:                             1.08
Prob(H) (two-sided):                  0.00   Kurtosis:                         8.25
===================================================================================
```

In [30]:
```
best_model.plot_diagnostics(figsize=(15, 12))
plt.show()
```



```
In [31]: btc_month2 = btc_month[['close']]
         date_list = [datetime(2018, 3, 31), datetime(2018, 4, 30), datetime(2018, 5, 31), datetime(2018, 6, 30)]
         future = pd.DataFrame(index=date_list, columns= btc_month.columns)
         btc_month2 = pd.concat([btc_month2, future])

         btc_month2['forecast'] = invboxcox(best_model.predict(start=0, end=75), lmbda)

         plt.figure(figsize=(15,7))
         btc_month2.close.plot()
         btc_month2.forecast.plot(color='r', ls='--', label='forecast')
         plt.legend()
         plt.title('Bitcoin Monthly Close Forecast')
         plt.ylabel('USD')
         plt.savefig('bitcoin_monthly_forecast.png')
         plt.show()
```



```
y_forecasted = btc_month2.forecast
y_truth = btc_month2['2015-01-01':'2017-01-01'].close

# Compute the root mean square error
rmse = np.sqrt(((y_forecasted - y_truth) ** 2).mean())
print('Mean Squared Error: {}'.format(round(rmse, 2)))
```

```
Mean Squared Error: 85.24
```

Code Implementation:
https://github.com/DebanganMandal/Crypto-Market-Prediction

Conclusion

Many prediction making models exist, but they are unable to reach their customers with their correct predictions. They have no idea which crypto to buy that will provide them with high closing or any values. I've made a ML model. The crypto market is now flourishing, and now we just have to juice out the best out of it. The way many have done with the stock market, we should do the same with the stock market and make the currency digital.