In [1]:
```
import os
os.chdir('../')
```

Package Overview

Pip install the package (git clone)

In [5]:
```
!pip install pybcoin
```

```
Collecting pybcoin
  Downloading https://files.pythonhosted.org/packages/0b/57/ead41e3ac4fcd91df18f4c8f0867e6
0a19a3554f3b6d73e0683843118f53/pybcoin-0.14-py3-none-any.whl
Installing collected packages: pybcoin
Successfully installed pybcoin-0.14

You are using pip version 9.0.1, however version 10.0.1 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
```

After installation the user needs to configure the API keys in the config.ini file.

- The user needs to register an app in twitter applications inorder to use tweepy.
- Needs to create an account in Quandl to be able to retreive various day level data.

In [4]:
```
!cat 'pybcoin/config/config.ini'
```

```
[Twitter]
consumer_key =
consumer_secret =
access_token =
access_token_secret =
tweet-count-url = https://bitinfocharts.com/comparison/tweets-btc.html

[Reddit]
api-uri = https://elastic.pushshift.io/rc/comments/_search?source={"query":{"bool":{"must"
:[{"simple_query_string":{"query":"bitcoin|btc|crypto","fields":["body"],"default_operator
":"and"}}],"filter":[{"range":{"created_utc":{"gte":START_UTC,"lte":END_UTC}}},{"terms":{"
subreddit":["bitcoin","btc","cryptocurrency"]}}],"should":[],"must_not":[]}},"size":10000,
"sort":{"created_utc":"desc"}}
data_path = ./data/latest/

[Collector]
in_path_btc = ./data/btc/
in_path_comm = ./data/commodity/
in_path_gtrends = ./data/gtrends/
out_path = ./data/

[Forecast]
in_path_btc = ./data/btc/
in_path_comm = ./data/commodity/
in_path_gtrends = ./data/gtrends/
out_path = ./data/
in_path_social = ./data/latest/
path_time_pred = ./data/pred/

[Quandl]
quandl-key =

[Sentiment]
text_csv_path = ./data/latest/
wc_path = ./pybcoin/static/
```

Triggering data collection

In [8]:
```
from pybcoin.DataCollector.controller_collector import ControllerCollector
```

In [11]:
```
config_path = './pybcoin/config/config.ini'
config = SafeConfigParser()
config.read(config_path)
collector = ControllerCollector(config_path)
collector.data_collection_pipeline()
```

```
C:\Users\vivan\Anaconda3\lib\site-packages\ipykernel_launcher.py:2: DeprecationWarning: Th
e SafeConfigParser class has been renamed to ConfigParser in Python 3.2. This alias will b
e removed in future versions. Use ConfigParser directly instead.


Twitter error response: status code = 400
Failure while collecting tweets.
Data Collection complete
```

Out[11]:
```
True
```

This will start the entire data collection pipeline. You can also trigger individual data collection modules as needed. For example you can use BtcDataCollector module to collect day level data for btc prices, number of btc related tweets and the total volume of btc transactions in USD.

In [10]:
```
from pybcoin.DataCollector.controller_collector import BtcDataCollector
```

In [12]:
```
collector  = BtcDataCollector(config)
```

In [13]:
```
collector.fetch_btc_price()
```

In [14]:
```
collector.fetch_transaction_volume()
```

In [15]:
```
collector.fetch_tweet_counts()
```

The data will be appended to the corresponding files in **data/btc/** folder.

Similarly you can retrieve data from various other sources. See documentation for details.

Analysis overview

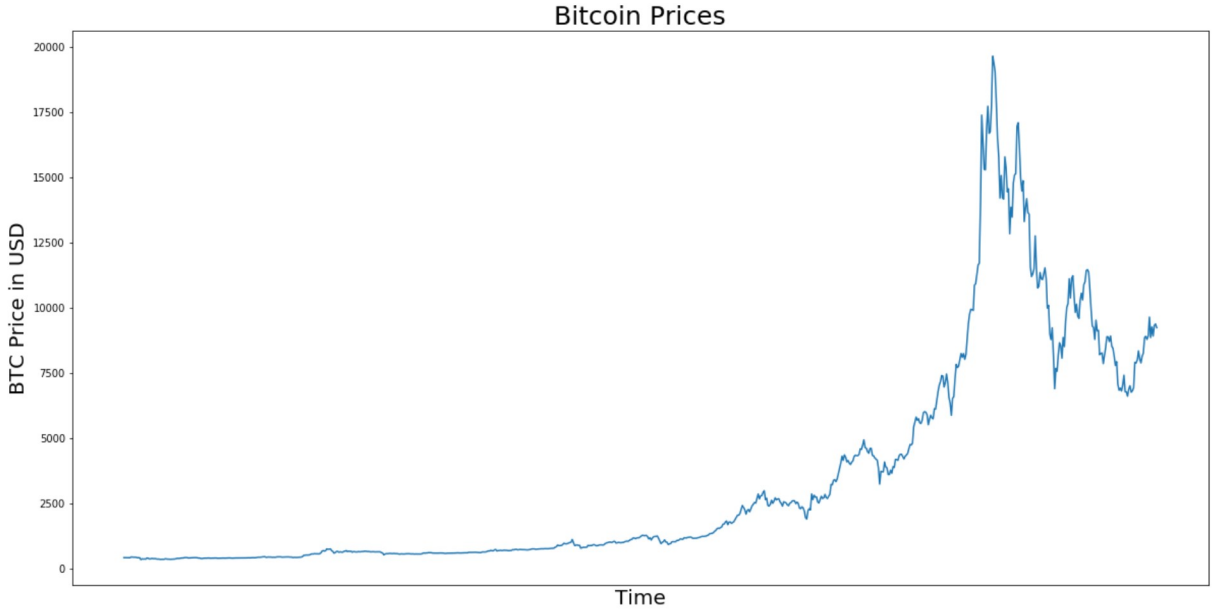
In [17]:
```
# import packages
import matplotlib.pyplot as plt
import pandas as pd
import warnings
from fbprophet import Prophet
import seaborn as sns
```

In [18]:
```
# suppressing warnings for now
warnings.filterwarnings("ignore")
```

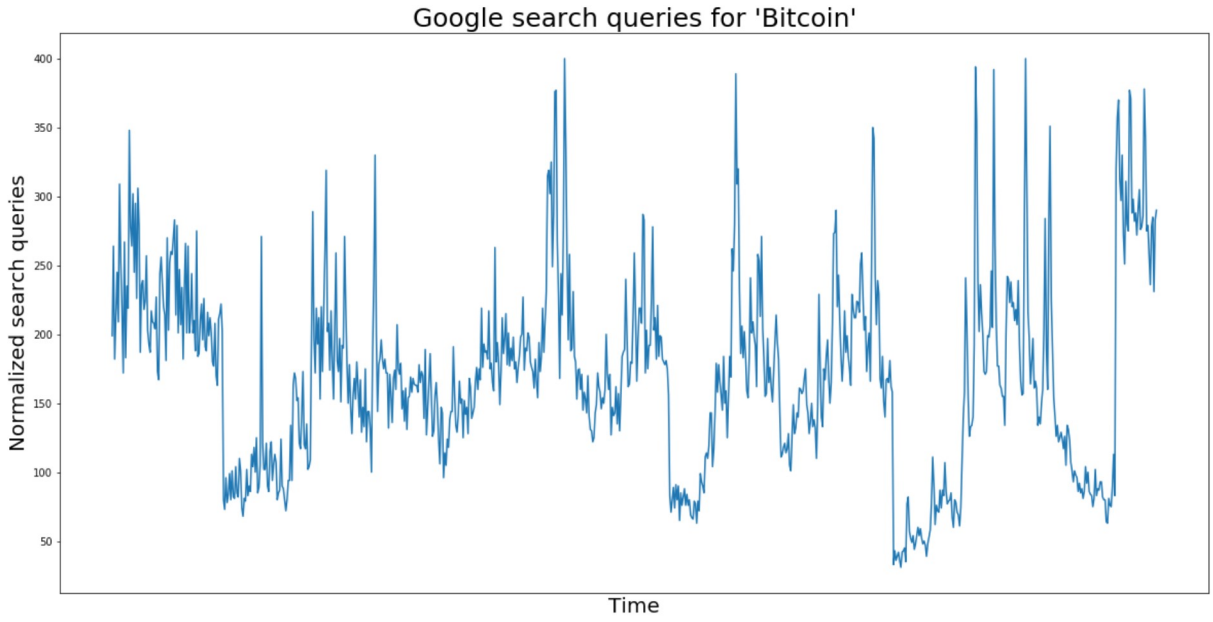
In [19]:
```
# reading data
btc_data = pd.read_csv('data/btc/btc_prices.csv')
oil_data = pd.read_csv('data/commodity/oil_price.csv')
google_data = pd.read_csv('data/gtrends/GTrendsData.csv')
twitter_data = pd.read_csv('data/latest/tweets_sentiment.csv')
reddit_data = pd.read_csv('data/latest/reddit_comments_sentiment.csv')
```

In [4]:
```
# getting relevant sum from the google trends data
google_data['google_hits'] = google_data['btc'] + google_data['bitcoin'] + google_data['btc
usd'] + google_data['btcusd']
```

In [5]:
```
# visualizing the various prices
# plotting BTC prices
plt.figure(figsize=(20,10))
plt.plot(btc_data['Date'], btc_data['btc_price'])
plt.xticks([])
plt.title('Bitcoin Prices', fontsize=25)
plt.xlabel('Time', fontsize=20);
plt.ylabel('BTC Price in USD', fontsize=20);
```



In [6]:
```
# plotting google search queries
plt.figure(figsize=(20,10))
plt.plot(google_data['Date'], google_data['google_hits'])
plt.xticks([])
plt.title("Google search queries for Bitcoin", fontsize=25)
plt.xlabel('Time', fontsize=20);
plt.ylabel('Normalized search queries', fontsize=20);
```



In [7]:
```
# demonstration to use the time series package: fbprophet
# prophet module requires columns ds (date) and y (value)
time_series = btc_data.rename(columns={'Date': 'ds', 'btc_price': 'y'})

# parameters can be changed based on the seasonality that needs to be incorporated
# the parameter changepoint_prior_scale can be tuned to give the best results for specific u
se-cases
m = Prophet(yearly_seasonality=True, daily_seasonality=False, changepoint_prior_scale=0.001)

# fitting the time series time model
m.fit(time_series);
```

In [8]:
```
# making the future predictions for 1 day excluding history
future = m.make_future_dataframe(periods=1, include_history=False)
future = m.predict(future)
```

In [9]:
```
# the result contains various trend and seasonality components
# the result also contains the confidence levels for the various components
future
```

Out[9]:

	ds	trend	yhat_lower	yhat_upper	trend_lower	trend_upper	seasonal	seasonal_lower	se
0	2018-05-01	7763.114318	4535.96216	10037.568007	7763.114318	7763.114318	-544.975864	-544.975864	-5

In [10]:
```
# computing the correlation-matrix between the variables
# joining the various data-frames
df = pd.merge(btc_data, oil_data, on='Date')
del df['dates_s']
df = pd.merge(df, google_data, on='Date')
del df['date']
df = pd.merge(df, twitter_data, on='Date')
del df['Date']

# renaming columns name for twitter
df = df.rename(columns = {'Negative':'twitter_negative'})
df = df.rename(columns = {'Positive':'twitter_positive'})
df = pd.merge(df, reddit_data, left_on = 'utc_time', right_on = 'Date')
del df['Date']

# renaming columns name for reddit
df = df.rename(columns = {'Negative':'reddit_negative'})
df = df.rename(columns = {'Positive':'reddit_positive'})

# creating various ratios
df['twitter'] = df['twitter_positive']/(df['twitter_positive'] + df['twitter_negative'])
df['reddit'] = df['reddit_positive']/(df['reddit_positive'] + df['reddit_negative'])
```

In [11]:
```
# creating a correlation matrix between various variables
# selecting the relevant variables
df_cor = df[['oil_price','google_hits','twitter','reddit','btc_price']]
```

In [12]:
```
# plotting the correlation matrix
corr = df_cor.corr()
fig, ax = plt.subplots(figsize=(10, 8))
sns.heatmap(corr,
            xticklabels=corr.columns.values,
            yticklabels=corr.columns.values, ax=ax);
```