# Introduction

This notebook is an implementation of the twitter sentiment analysis that will be used to predict the price of bitcoin based on the sentiment gathered from the tweets. Data will be saved in an sql database to make it possible to create a datawarehouse that can be used over time for more analysis.

# Importing Libraries

import sys  
import tweepy  
import matplotlib.pyplot as plt  
import pandas as pd  
import numpy as np  
import os  
import nltk  
import pycountry  
import re  
import string  
  
from textblob import TextBlob  
from wordcloud import WordCloud, STOPWORDS  
from PIL import Image  
from nltk.sentiment.vader import SentimentIntensityAnalyzer  
from langdetect import detect  
from nltk.stem import SnowballStemmer  
from sklearn.feature\_extraction.text import CountVectorizer

# Twitter Authentication

import datetime  
import nltk  
nltk.downloader.download('vader\_lexicon')  
client=tweepy.Client(bearer\_token='AAAAAAAAAAAAAAAAAAAAAAfVlQEAAAAA8pctnhfuuAiNvqOj%2FeXaAXzlQ%2Fs%3D9YoRCaJG6cViLtEhDY7oeSMVJrelLm4DC4djXlp9zMBt0i4Icm')

[nltk\_data] Downloading package vader\_lexicon to  
[nltk\_data] C:\Users\kinut\AppData\Roaming\nltk\_data...

# Data Collection

In this step we collect data by retrieving the tweets using the tweepy library.

def percentage(part, whole):  
 return 100 \* float(part)/float(whole)  
  
  
today = datetime.datetime.now()  
today = today.replace(hour=23, minute=59, second=59, microsecond=999999)  
time\_to\_the\_past = 6  
yesterday = today - datetime.timedelta(time\_to\_the\_past)  
  
keyword = "bitcoin btc"  
noOfTweet = 1000  
next\_day = yesterday + datetime.timedelta(time\_to\_the\_past)  
  
query = '#bitcoin lang:en'  
tweets = client.search\_recent\_tweets(query=query, tweet\_fields=['context\_annotations', 'created\_at'], max\_results=100)  
#print(tweets)  
positive = 0  
negative = 0  
neutral = 0  
polarity = 0  
tweet\_list = []  
neutral\_list = []  
negative\_list = []  
positive\_list = []  
for tweet in tweets.data:  
 tweet\_list.append(tweet.text)  
 analysis = TextBlob(tweet.text)  
 score = SentimentIntensityAnalyzer().polarity\_scores(tweet.text)  
 neg = score['neg']  
 neu = score['neu']  
 pos = score['pos']  
 comp = score['compound']  
 polarity += analysis.sentiment.polarity  
  
 if neg > pos:  
 negative\_list.append(tweet.text)  
 negative += 1  
 elif pos > neg:  
 positive\_list.append(tweet.text)  
 positive += 1  
  
 elif pos == neg:  
 neutral\_list.append(tweet.text)  
 neutral += 1  
positive = percentage(positive, noOfTweet)  
negative = percentage(negative, noOfTweet)  
neutral = percentage(neutral, noOfTweet)  
polarity = percentage(polarity, noOfTweet)  
positive = format(positive, '.1f')  
negative = format(negative, '.1f')  
neutral = format(neutral, '.1f')

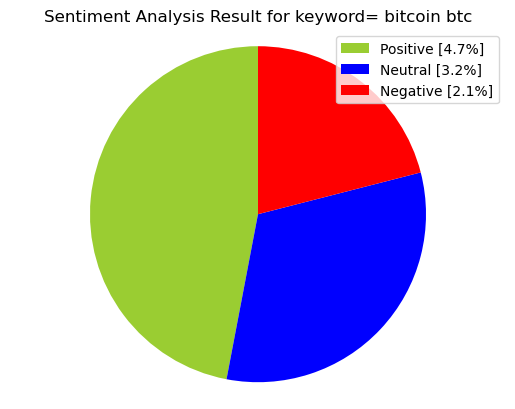
### Results of the retrieved tweets

tweet\_list = pd.DataFrame(tweet\_list)  
neutral\_list = pd.DataFrame(neutral\_list)  
negative\_list = pd.DataFrame(negative\_list)  
positive\_list = pd.DataFrame(positive\_list)  
print("total number of tweets: ",len(tweet\_list))  
print("positive number of tweets: ",len(positive\_list))  
print("negative number of tweets: ", len(negative\_list))  
print("neutral number of tweets: ",len(neutral\_list))

total number of tweets: 100  
positive number of tweets: 47  
negative number of tweets: 21  
neutral number of tweets: 32

## Visualizing the scores of the uncleaned data

labels = ['Positive ['+str(positive)+'%]' , 'Neutral ['+str(neutral)+'%]','Negative ['+str(negative)+'%]']  
sizes = [positive, neutral, negative]  
colors = ['yellowgreen', 'blue','red']  
patches, texts = plt.pie(sizes,colors=colors, startangle=90)  
plt.style.use('default')  
plt.legend(labels)  
plt.title("Sentiment Analysis Result for keyword= "+keyword+"" )  
plt.axis('equal')  
plt.show()



### Displaying the retrieved tweets

tweet\_list

0  
0 RT @BTC\_Archive: ‼️ White House Cryptocurrency...  
1 RT @alanbwt: Nostr feels like a zero to one mo...  
2 RT @CryptoQueenUSA: Just because supply goes d...  
3 @CoinDesk @realMatrixport @godbole17 #bitcoin ...  
4 RT @WatcherGuru: The top 100 richest #Bitcoin ...  
.. ...  
95 RT @saylor: Stocks valued on expected cash flo...  
96 RT @BitcoinMagazine: BREAKING: 🇺🇸 White House ...  
97 RT @Kenslog: Good morning friends, I won 5500 ...  
98 RT @CoinMarketCap: Let's debunked the seven bi...  
99 ➡️ We see the market solid, we will continue t...  
  
[100 rows x 1 columns]

# Data Preparation

The first step of data preparation is dropping any duplicate tweets

### Dropping duplicates

tweet\_list.drop\_duplicates(inplace = True)

The new list after the duplicates are dropped becomes:

tweet\_list

0  
0 RT @BTC\_Archive: ‼️ White House Cryptocurrency...  
1 RT @alanbwt: Nostr feels like a zero to one mo...  
2 RT @CryptoQueenUSA: Just because supply goes d...  
3 @CoinDesk @realMatrixport @godbole17 #bitcoin ...  
4 RT @WatcherGuru: The top 100 richest #Bitcoin ...  
.. ...  
94 RT @Jcastweet: Adopting #Bitcoin ⚡️ payments m...  
95 RT @saylor: Stocks valued on expected cash flo...  
97 RT @Kenslog: Good morning friends, I won 5500 ...  
98 RT @CoinMarketCap: Let's debunked the seven bi...  
99 ➡️ We see the market solid, we will continue t...  
  
[92 rows x 1 columns]

### Removing special characters, RT and punctuations

tw\_list = pd.DataFrame(tweet\_list)  
tw\_list["text"] = tw\_list[0]  
remove\_rt = lambda x: re.sub('RT @\w+: '," ",x)  
rt = lambda x: re.sub("(@[A-Za-z0–9]+)|([0-9A-Za-z \t])|(\w+:\/\/\S+)"," ",x)  
tw\_list["text"] = tw\_list.text.map(remove\_rt).map(rt)  
tw\_list["text"] = tw\_list.text.str.lower()

A preview of the first 5 rows is shown below

tw\_list.head(5)

0 \  
0 RT @BTC\_Archive: ‼️ White House Cryptocurrency...   
1 RT @alanbwt: Nostr feels like a zero to one mo...   
2 RT @CryptoQueenUSA: Just because supply goes d...   
3 @CoinDesk @realMatrixport @godbole17 #bitcoin ...   
4 RT @WatcherGuru: The top 100 richest #Bitcoin ...   
  
 text   
0 ‼️ :\n• ...   
1 ...   
2 , ' ...   
3 # ...   
4 # ...

# Sentiment Analysis

Now that our data is cleaned we can conduct sentiment analysis to obtain sentiments from the data.

tw\_list[['polarity', 'subjectivity']] = tw\_list['text'].apply(lambda Text: pd.Series(TextBlob(Text).sentiment))  
for index, row in tw\_list['text'].iteritems():  
 score = SentimentIntensityAnalyzer().polarity\_scores(row)  
 neg = score['neg']  
 neu = score['neu']  
 pos = score['pos']  
 comp = score['compound']  
 if neg > pos:  
 tw\_list.loc[index, 'sentiment'] = "negative"  
 elif pos > neg:  
 tw\_list.loc[index, 'sentiment'] = "positive"  
 else:  
 tw\_list.loc[index, 'sentiment'] = "neutral"  
 tw\_list.loc[index, 'neg'] = neg  
 tw\_list.loc[index, 'neu'] = neu  
 tw\_list.loc[index, 'pos'] = pos  
 tw\_list.loc[index, 'compound'] = comp  
tw\_list.head(10)

0 \  
0 RT @BTC\_Archive: ‼️ White House Cryptocurrency...   
1 RT @alanbwt: Nostr feels like a zero to one mo...   
2 RT @CryptoQueenUSA: Just because supply goes d...   
3 @CoinDesk @realMatrixport @godbole17 #bitcoin ...   
4 RT @WatcherGuru: The top 100 richest #Bitcoin ...   
5 RT @WatcherGuru: FUN FACT: #Bitcoin miners gen...   
6 RT @TrustWallet: #Bitcoin future price predict...   
7 RT @BitcoinSapiens: "#Bitcoin is the only majo...   
8 RT @saylor: My discussion with @natbrunell on ...   
9 RT @cryptorecruitr: Unpopular opinion: Altcoin...   
  
 text polarity subjectivity \  
0 ‼️ :\n• ... 0.0 0.0   
1 ... 0.0 0.0   
2 , ' ... 0.0 0.0   
3 # ... 0.0 0.0   
4 # ... 0.0 0.0   
5 : # $ ... 0.0 0.0   
6 # : \n\n ... 0.0 0.0   
7 "# ... 0.0 0.0   
8 ... 0.0 0.0   
9 : #... 0.0 0.0   
  
 sentiment neg neu pos compound   
0 neutral 0.0 1.0 0.0 0.0   
1 neutral 0.0 0.0 0.0 0.0   
2 neutral 0.0 0.0 0.0 0.0   
3 neutral 0.0 0.0 0.0 0.0   
4 neutral 0.0 0.0 0.0 0.0   
5 neutral 0.0 0.0 0.0 0.0   
6 neutral 0.0 0.0 0.0 0.0   
7 neutral 0.0 1.0 0.0 0.0   
8 neutral 0.0 1.0 0.0 0.0   
9 neutral 0.0 0.0 0.0 0.0

tw\_list\_negative = tw\_list[tw\_list["sentiment"]=="negative"]  
tw\_list\_positive = tw\_list[tw\_list["sentiment"]=="positive"]  
tw\_list\_neutral = tw\_list[tw\_list["sentiment"]=="neutral"]

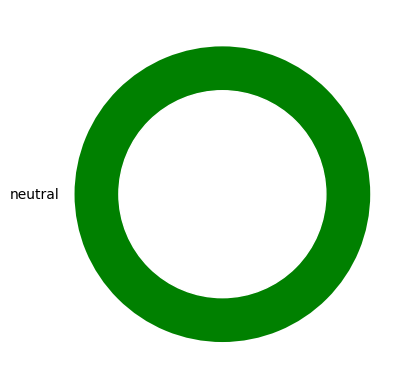
#### Statistics

def count\_values\_in\_column(data,feature):  
 total=data.loc[:,feature].value\_counts(dropna=False)  
 percentage=round(data.loc[:,feature].value\_counts(dropna=False,normalize=True)\*100,2)  
 return pd.concat([total,percentage],axis=1,keys=['Total','Percentage'])  
#Count\_values for sentiment  
count\_values\_in\_column(tw\_list,"sentiment")

Total Percentage  
neutral 92 100.0

## Plotting the cleaned data

pc = count\_values\_in\_column(tw\_list,"sentiment")  
names= pc.index  
size=pc["Percentage"]  
   
# Create a circle for the center of the plot  
my\_circle=plt.Circle( (0,0), 0.7, color='white')  
plt.pie(size, labels=names, colors=['green','blue','red'])  
p=plt.gcf()  
p.gca().add\_artist(my\_circle)  
plt.show()



# Conclusion

The sentiment analysis above can help us understand and predict the price of bitcoin because if the majority of the tweets have a neutral sentiment then the price is not likely to go down or go up but if it was positive then the price is likely to go up and for negative the price is likely to go down