CAPSTONE 1:

BITCOIN PRICE PREDICTION

AUTHOR: Akinkunle Akinola

Table of Contents

DATASETS 3

DATA CLEANING 3

EDA 15

FEATURE ENGINEERING 15

MODELLING 15

TRADING STRATEGY 0: Baseline Model using market data 17

Baseline Model + Model with Alerts Feature 17

MODEL INTERPRETATION 17

BOLLINGER BANDS 17

TRADING STRATEGY #1 Bollinger Band. 20

TRADING STRATEGY #2: BITCOIN PRICE Z-SCORE 20

TRADING STRATEGY #3: LOGISTIC REGRESSION 20

Logistic Model: Model using market data 23

Classification table without alert data 23

RESULTS: QUANTIFIED AUTOMATED TRADING PERFORMANCE 24

FUTURE WORK 1

CREDIT 1

ABSTRACT

Over the last couple years, the cryptocurrency market has been growing tremendously, and gains in popularity almost everyday. The average daily trading in the market (ie: market cap, daily trading volume, etc) is enormous especially for Bitcoin. Despite these success stories about the Bitcoin market, it does have some downfalls. It is well-known that the prices of cryptocurrencies rise and fall over a short period of time. This price

volatility tends to scare some traders who are afraid of losing money invested in the market.

This project will use a time series analysis to predict the price of Bitcoin over a period of time using market data and a proprietary alerts data set. The business goal is to build a predictive model that will give an insight on when, how and what to trade at a particular time to maximize return on investment capital and help traders make better trading decisions.

DATASETS

A market dataset and alert dataset from a proprietary saas website will be used in the course of this project. The market dataset which was found on kaggle, has 942,297 rows of daily observations between 04/28/2013 and 11/29/2018 and 13 features while the alert dataset contains 102,225 rows and 23 features which makes them sufficient for this project to develop a good predictive model.

DATA CLEANING

Data cleaning is one of the most important aspects of a project. The datasets used for this project were rigorously cleaned so as to achieve the ultimate goal. Here are the things done during the data cleaning process:

1. Low signal columns like updated_at, deleted_at, last_checked, and last_sent were removed from the alert dataset.
2. The column “data” which was in JSON format was converted to the appropriate format and splitted to get additional columns like “comparison”, “value”, and “operator”

```
In [9]:
import json
# val = Alert_price_point.iloc[0]["data"]
# print(val)
def convert_str_to_proper_dict(val):
  val = r'"' + val
  val = val[:3] + '"' + val[3:]
  val = val.replace(r'\"\"', r'\",\"')
  val = val.replace('null', 'null,')
  val = "".join(val.split())
  return json.loads(json.loads(val))
```

```
In [10]:
q = list(Alert_price_point["data"])
for idx, val in enumerate(q):
  try:
    convert_str_to_proper_dict(val)
  except:
    print(f"idx is {idx}")
    break
```

```
In [11]:
print(q[23811].strip())
convert_str_to_proper_dict(q[23811])
```

```
{\comparison\":\"buy_price\"\"value\":\"        0.0009\"\"operator\":\">=\"}"
```

```
Out[11]: {'comparison': 'buy_price', 'value': '0.0009', 'operator': '>='}
```

```
In [12]:
Alert_price_point["data"] = Alert_price_point["data"].apply(convert_str_to_proper_dict)
```

```
In [13]:
json_df = pd.json_normalize(Alert_price_point["data"])
# json_df
```

Figure 1: Splitting Json column

Pandas json_normalize function was used to flatten the JSON column into a DataFrame after being splitted and cleaned appropriately.

3. Missing values which were stored as NaN were dropped.
4. Date column was converted to datetime.
5. Market data and Alert data were merged together on date and BTC USD.
6. Since the data sets contain different types of cryptocurrency, I subsetted the dataset over bitcoin to get only the bitcoin data.

```
BTC      0.453064
ETH      0.134895
LTC      0.068649
USDT     0.055205
XRP      0.048074
BCH      0.021152
TRX      0.018861
EOS      0.009903
ADA      0.009743
XVG      0.008237
NEO      0.007435
XLM      0.007067
XMR      0.006650
ETC      0.006506
BNB      0.003445
DASH     0.003221
ZEC      0.003221
OMG      0.003077
ICX      0.003077
ETN      0.002195
Name: symbol, dtype: float64
```

Figure 2: Bitcoin consists of about 45 percent of the Alerts

```
USD     0.427136
BTC     0.175774
EUR     0.056935
ETH     0.048186
USDT    0.037450
XRP     0.018188
AUD     0.011906
NEO     0.010801
LTC     0.008157
BCC     0.006041
SC      0.005673
DGB     0.005096
CAD     0.004775
ADA     0.004631
OMG     0.004487
XVG     0.004455
XMR     0.004391
DASH    0.004086
ETC     0.004006
DOGE    0.003830
```

Figure 3: USD consists of about 45 percent of the Alerts.

Exploratory Data Analysis

Exploratory data analysis was performed on the dataset which results in the below graphs. In addition to the below graphs is data profiling technique to examine, analyze, and create useful summaries of the data. Pandas profiling was used to explore the data.

Pandas profiling

Overview:

Dataset statistics

Number of variables	13
Number of observations	661
Missing cells	0
Missing cells (%)	0.0%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	67.3 KiB
Average record size in memory	104.2 B

Variable types

NUM	12
CAT	1

Variables:

Date:

date
Categorical
UNIQUE

Distinct count	661
Unique (%)	100.0%
Missing	0
Missing (%)	0.0%
Memory size	5.2 KiB

2017-05-26	1
2018-11-14	1
2018-08-28	1
2017-08-23	1
2017-12-09	1
Other values (656)	656

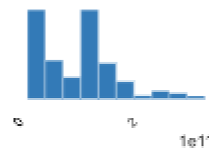
Market Cap:

market_y
Real number ($\mathbb{R}_{\geq 0}$)

HIGH CORRELATION
UNIQUE

Distinct count	661
Unique (%)	100.0%
Missing	0
Missing (%)	0.0%
Infinite	0
Infinite (%)	0.0%

Mean	100889948825.5
Minimum	14284036613.0
Maximum	326502485530.(
Zeros	0
Zeros (%)	0.0%
Memory size	5.2 KiB



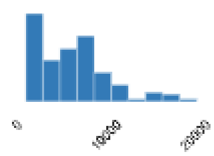
High:

high
Real number ($\mathbb{R}_{\geq 0}$)

HIGH CORRELATION
UNIQUE

Distinct count	661
Unique (%)	100.0%
Missing	0
Missing (%)	0.0%
Infinite	0
Infinite (%)	0.0%

Mean	6160.73313161(
Minimum	899.4
Maximum	20089.0
Zeros	0
Zeros (%)	0.0%
Memory size	5.2 KiB



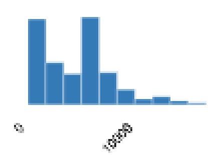
Low:

low
Real number ($\mathbb{R}_{\geq 0}$)

HIGH CORRELATION
UNIQUE

Distinct count	661
Unique (%)	100.0%
Missing	0
Missing (%)	0.0%
Infinite	0
Infinite (%)	0.0%

Mean	5755.50757942(
Minimum	830.80000000(
Maximum	18974.1000000(
Zeros	0
Zeros (%)	0.0%
Memory size	5.2 KiB



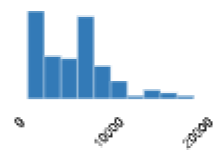
Open:

open
Real number ($\mathbb{R}_{\geq 0}$)

HIGH CORRELATION
UNIQUE

Distinct count	661
Unique (%)	100.0%
Missing	0
Missing (%)	0.0%
Infinite	0
Infinite (%)	0.0%

Mean	5970.28632375
Minimum	830.94999999
Maximum	19475.7999999
Zeros	0
Zeros (%)	0.0%
Memory size	5.2 KiB



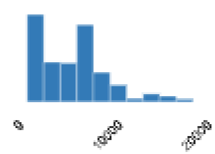
Close:

close
Real number ($\mathbb{R}_{\geq 0}$)

HIGH CORRELATION

Distinct count	659
Unique (%)	99.7%
Missing	0
Missing (%)	0.0%
Infinite	0
Infinite (%)	0.0%

Mean	5976.99381240
Minimum	886.62
Maximum	19497.3999999
Zeros	0
Zeros (%)	0.0%
Memory size	5.2 KiB



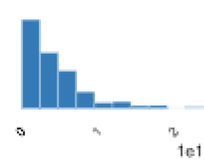
Volume:

volume
Real number ($\mathbb{R}_{\geq 0}$)

HIGH CORRELATION
UNIQUE

Distinct count	661
Unique (%)	100.0%
Missing	0
Missing (%)	0.0%
Infinite	0
Infinite (%)	0.0%

Mean	4258915207.181
Minimum	60851700.0
Maximum	23840899072.0
Zeros	0
Zeros (%)	0.0%
Memory size	5.2 KiB



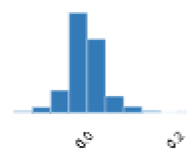
High Daily Percent Change:

high_daily_%…
Real number (ℝ)

UNIQUE

Distinct count	661
Unique (%)	100.0%
Missing	0
Missing (%)	0.0%
Infinite	0
Infinite (%)	0.0%

Mean	0.003227113581
Minimum	-0.15640282884
Maximum	0.24570780355
Zeros	1
Zeros (%)	0.2%
Memory size	5.2 KiB



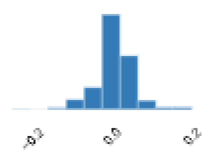
Low Daily Percent Change:

low_daily_%_…
Real number (ℝ)

UNIQUE

Distinct count	661
Unique (%)	100.0%
Missing	0
Missing (%)	0.0%
Infinite	0
Infinite (%)	0.0%

Mean	0.003599757331
Minimum	-0.25266645652
Maximum	0.19650775594
Zeros	1
Zeros (%)	0.2%
Memory size	5.2 KiB



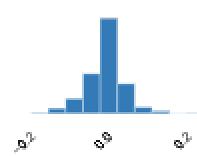
Open Daily Percent Change

open_daily_%…
Real number (ℝ)

UNIQUE

Distinct count	661
Unique (%)	100.0%
Missing	0
Missing (%)	0.0%
Infinite	0
Infinite (%)	0.0%

Mean	0.00356657818
Minimum	-0.18296833592
Maximum	0.25046147610
Zeros	1
Zeros (%)	0.2%
Memory size	5.2 KiB



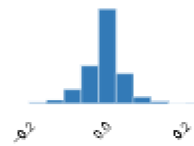
Close Daily Percent Change:

close_daily_%...
Real number (ℝ)

HIGH CORRELATION

Distinct count	660
Unique (%)	99.8%
Missing	0
Missing (%)	0.0%
Infinite	0
Infinite (%)	0.0%

Mean	0.003450633303
Minimum	-0.18741098081
Maximum	0.252471748941
Zeros	2
Zeros (%)	0.3%
Memory size	5.2 KiB



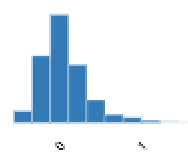
Volume Daily Percent Change:

volume_daily_...
Real number (ℝ)

UNIQUE

Distinct count	661
Unique (%)	100.0%
Missing	0
Missing (%)	0.0%
Infinite	0
Infinite (%)	0.0%

Mean	0.043212925696
Minimum	-0.57684903928
Maximum	1.655869507935
Zeros	1
Zeros (%)	0.2%
Memory size	5.2 KiB



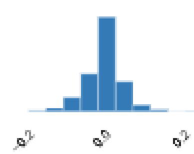
Market Daily Percent Change:

market_daily_...
Real number (ℝ)

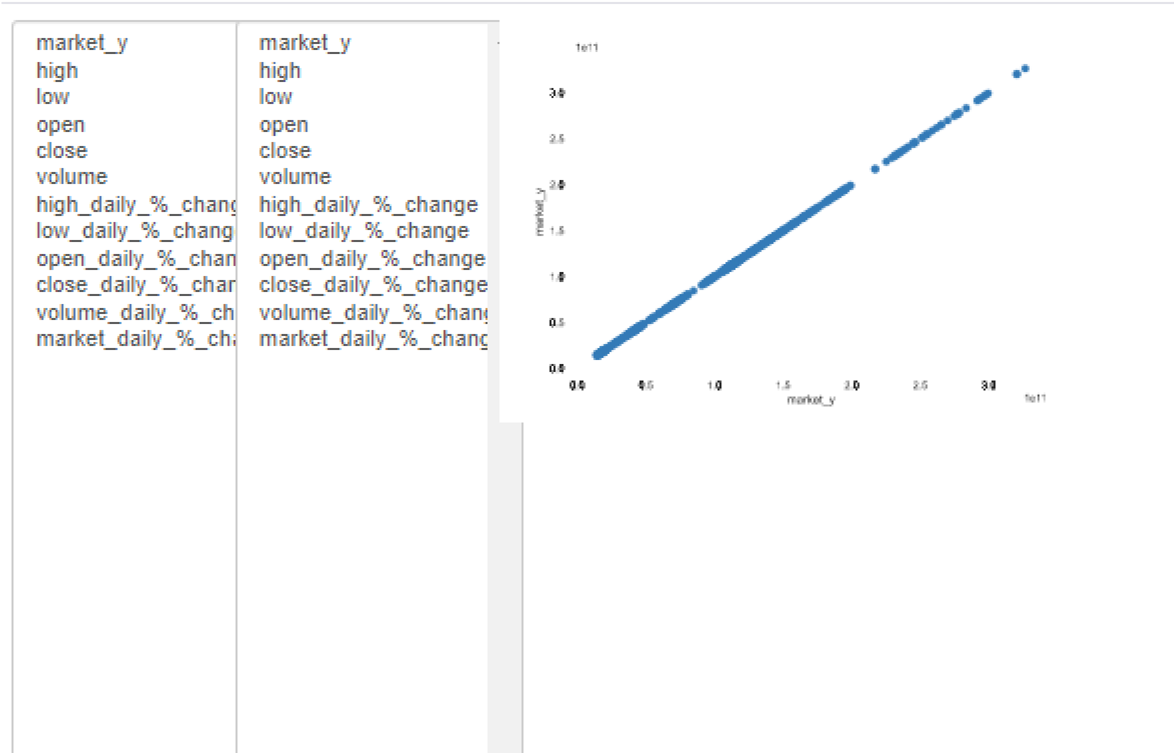
HIGH CORRELATION
UNIQUE

Distinct count	661
Unique (%)	100.0%
Missing	0
Missing (%)	0.0%
Infinite	0
Infinite (%)	0.0%

Mean	0.003567534811
Minimum	-0.18730305407
Maximum	0.252614033261
Zeros	1
Zeros (%)	0.2%
Memory size	5.2 KiB

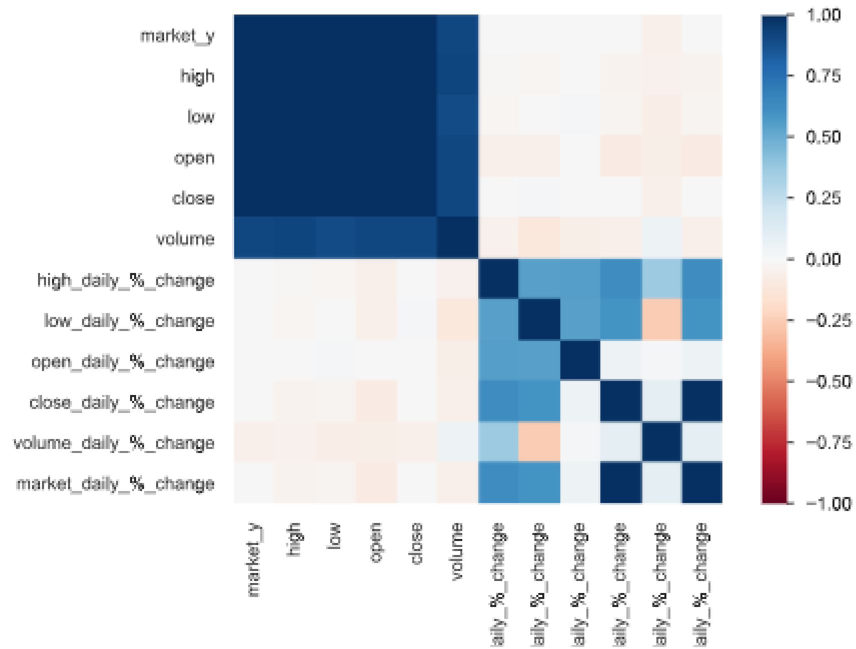


Interaction Plot:

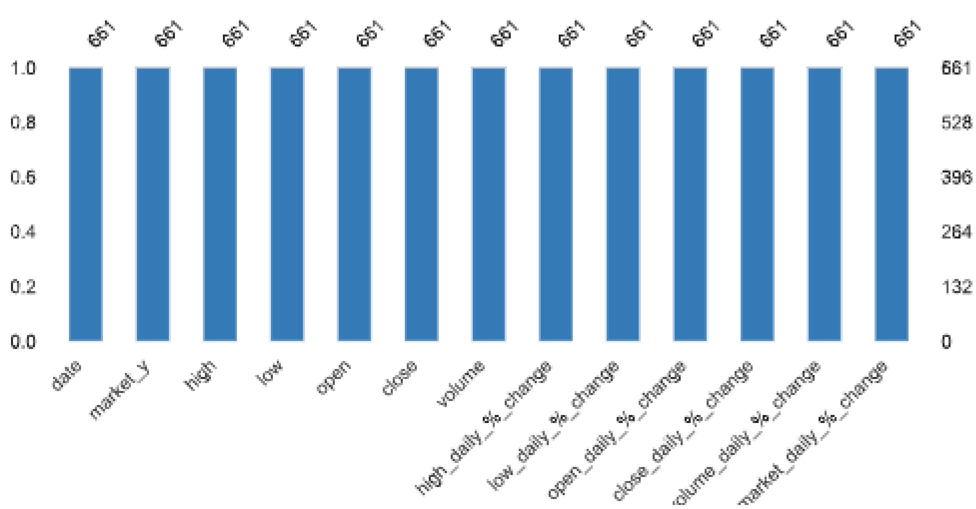


Correlation Plot:

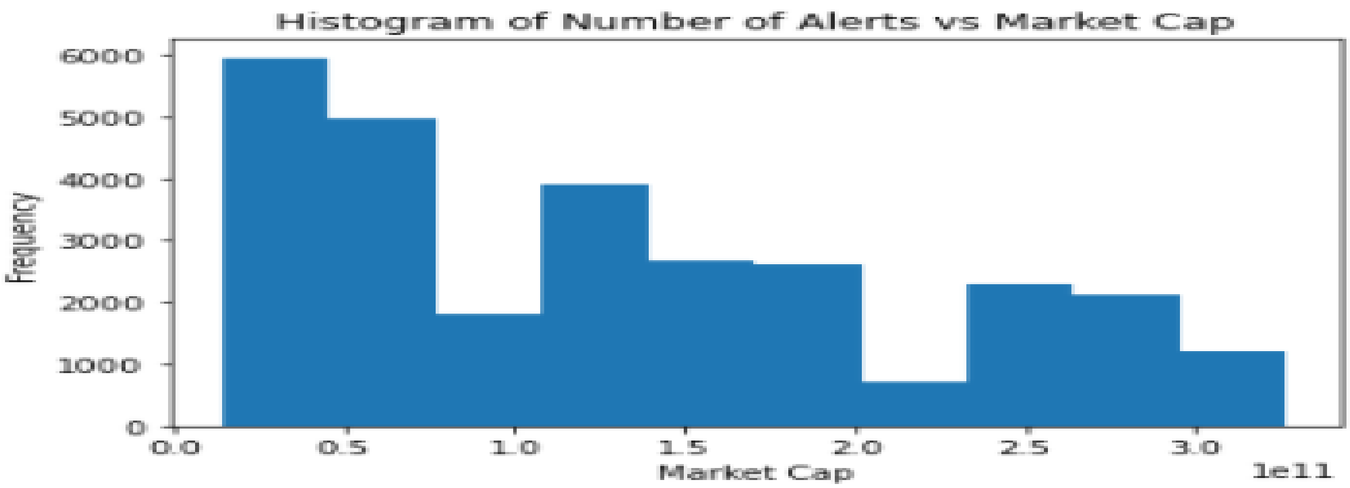
Pearson's r | Spearman's ρ | Kendall's τ | Phik (φk) | Toggle correlation descriptions



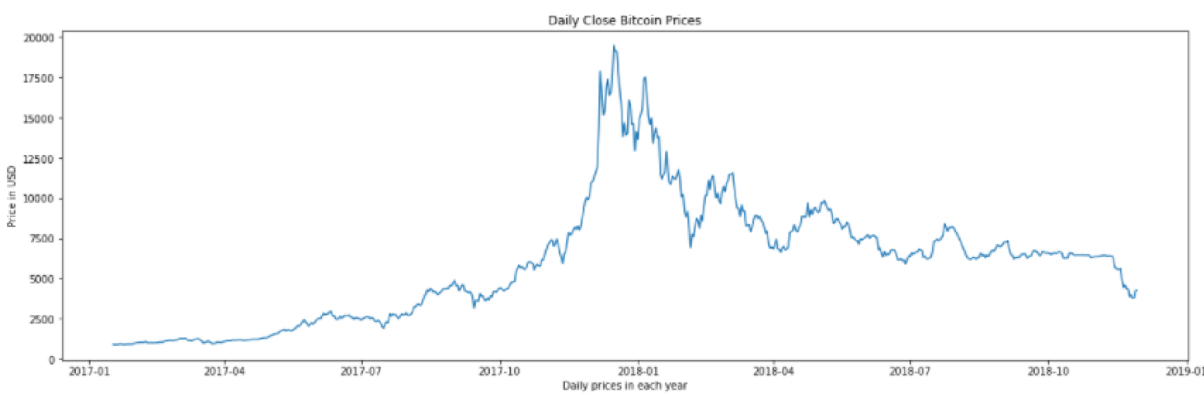
Missing Values:



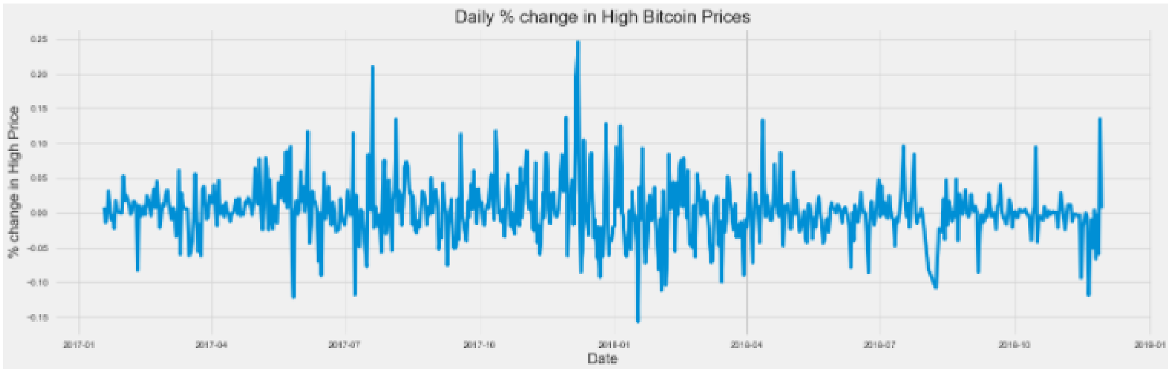
Market Cap Vs Frequency



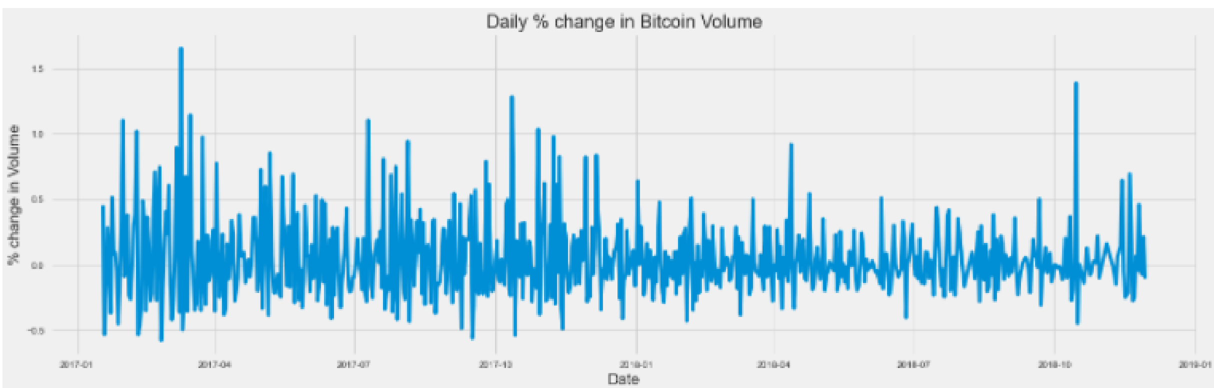
Bitcoin Prices plot:



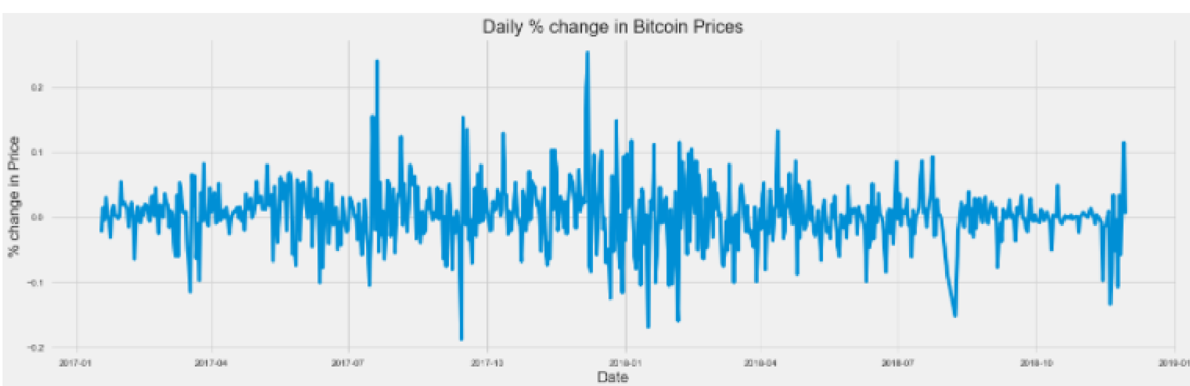
Percent Change in High prices plot:



Percent Change in Volume prices plot:



Percent Change in Close prices plot:



FEATURE ENGINEERING

The below dataframe shows the daily summary of the features as well as the daily percentage change of each feature. In order to be able to predict the estimated future's percentage change in price, an additional column was generated where the percentage change in price was shifted by a day.

```
notification_market_BTC.head().T
```

date	2017-01-17	2017-01-18	2017-01-19	2017-01-20	2017-01-21
market_y	1.462578e+10	1.428404e+10	1.448662e+10	1.442355e+10	1.485691e+10
high	9.105600e+02	9.175000e+02	9.046100e+02	8.994000e+02	9.273700e+02
low	8.308000e+02	8.583000e+02	8.843400e+02	8.870100e+02	8.955300e+02
open	8.309500e+02	9.093700e+02	8.883400e+02	8.981700e+02	8.955500e+02
close	9.079400e+02	8.866200e+02	8.990700e+02	8.950300e+02	9.217900e+02
volume	1.550950e+08	2.256770e+08	1.056250e+08	8.672840e+07	1.111580e+08
high_daily_%_change	0.000000e+00	7.621683e-03	-1.404905e-02	-5.759388e-03	3.109851e-02
low_daily_%_change	0.000000e+00	3.310063e-02	3.033904e-02	3.019201e-03	9.605303e-03
open_daily_%_change	0.000000e+00	9.437391e-02	-2.312590e-02	1.106558e-02	-2.917042e-03
close_daily_%_change	0.000000e+00	-2.348173e-02	1.404209e-02	-4.493532e-03	2.989844e-02
volume_daily_%_change	0.000000e+00	4.550887e-01	-5.319638e-01	-1.789027e-01	2.816794e-01
market_daily_%_change	0.000000e+00	-2.336581e-02	1.418227e-02	-4.353765e-03	3.004572e-02
user_id	3.000000e+01	5.300000e+01	1.650000e+02	8.500000e+01	1.120000e+02
alert_id	3.000000e+01	5.300000e+01	1.650000e+02	8.500000e+01	1.120000e+02
alert__daily_count_%_change	0.000000e+00	7.666667e-01	2.113208e+00	-4.848485e-01	3.176471e-01
change	0.000000e+00	0.000000e+00	1.000000e+00	0.000000e+00	1.000000e+00

MODELLING

Linear Regression: This is a linear approach used to verify the linear relationship among the features. The goal is to see if the alerts dataset is a good predictor of the price of Bitcoin and to also predict the future's percentage change in price . Does the model that include both alerts and crypto market features outperform the model with only crypto market features? To answer this question, an Ordinary Least Square model was implemented to establish a multiple linear regression and the following outcomes were achieved.

TRADING STRATEGY 0: Baseline Model using market data

Added features

high_daily_%_change

volume_daily_%_change
low_daily_%_change

Dependent Variable
close_shift_daily_%_change

OLS Regression Results

Dep. Variable:	y_k	**R-squared:**	0.360
Model:	OLS	**Adj. R-squared:**	0.357
Method:	Least Squares	**F-statistic:**	122.9
Date:	Tue, 28 Sep 2021	**Prob (F-statistic):**	3.91e-63
Time:	23:02:46	**Log-Likelihood:**	1225.7
No. Observations:	659	**AIC:**	-2443.
Df Residuals:	655	**BIC:**	-2425.
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	0.0016	0.002	1.090	0.276	-0.001	0.005
x_k['high_daily_%_change']	0.4728	0.054	8.764	0.000	0.367	0.579
x_k['volume_daily_%_change']	-0.0136	0.007	-2.076	0.038	-0.027	-0.001
x_k['low_daily_%_change']	0.2660	0.045	5.919	0.000	0.178	0.354

Omnibus:	87.319	**Durbin-Watson:**	2.819
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	604.565
Skew:	0.331	**Prob(JB):**	5.25e-132
Kurtosis:	7.645	**Cond. No.**	44.3

Baseline Model + Model with Alerts Feature

Added features
high_daily_%_change
volume_daily_%_change
low_daily_%_change
alert_daily_%_change

Dependent Variable
close_shift_daily_%_change

Dep. Variable:	y_k	R-squared:	0.365
Model:	OLS	Adj. R-squared:	0.361
Method:	Least Squares	F-statistic:	93.90
Date:	Tue, 28 Sep 2021	Prob (F-statistic):	4.30e-63
Time:	23:02:46	Log-Likelihood:	1228.1
No. Observations:	659	AIC:	-2446.
Df Residuals:	654	BIC:	-2424.
Df Model:	4		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	0.0013	0.002	0.874	0.382	-0.002	0.004
x_k['high_daily_%_change']	0.4574	0.054	8.432	0.000	0.351	0.564
x_k['volume_daily_%_change']	-0.0175	0.007	-2.579	0.010	-0.031	-0.004
x_k['low_daily_%_change']	0.2844	0.046	6.239	0.000	0.195	0.374
x_k['alert__daily_count_%_change']	0.0107	0.005	2.200	0.028	0.001	0.020

Omnibus:	90.616	Durbin-Watson:	2.821
Prob(Omnibus):	0.000	Jarque-Bera (JB):	638.101
Skew:	0.356	Prob(JB):	2.74e-139
Kurtosis:	7.768	Cond. No.	45.0

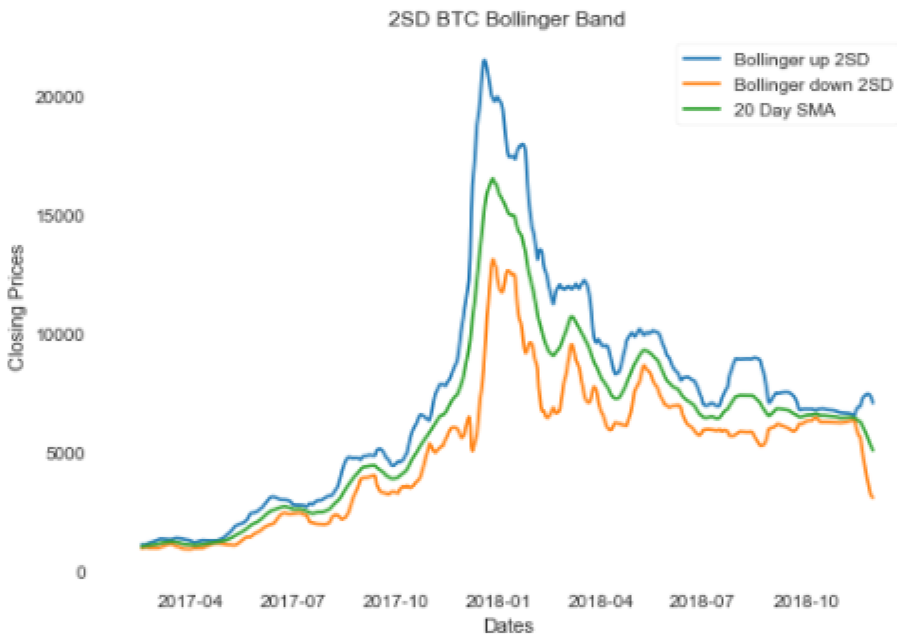
MODEL INTERPRETATION

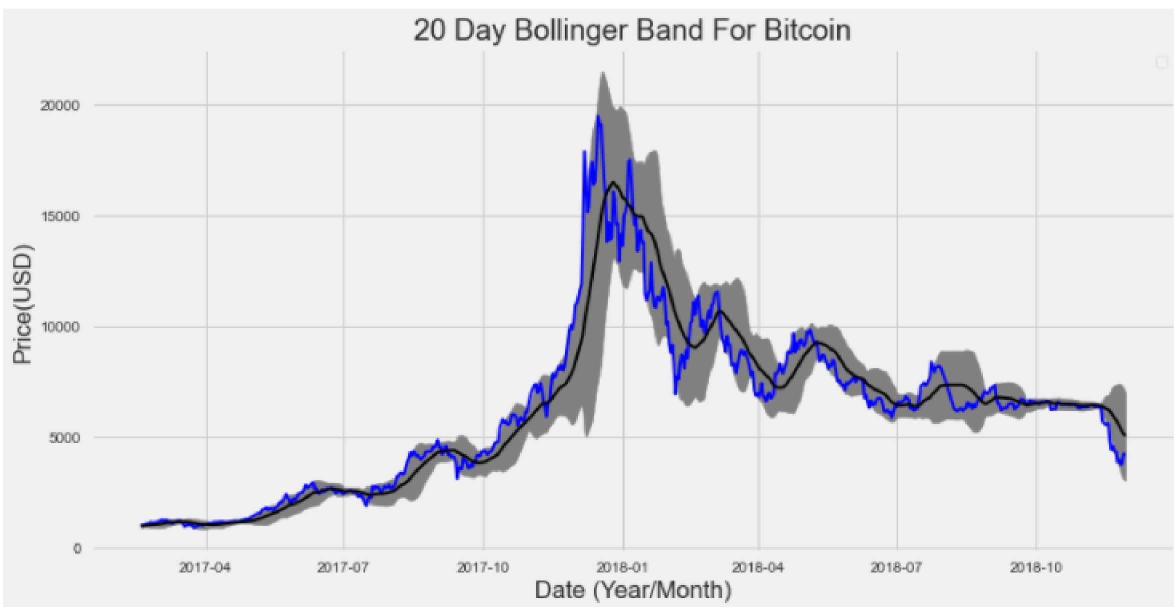
The model with only crypto market features has a R-Squared of 0.365. However, when the alert feature was added to the model, there was a slight increase in R-Squared to 0.360. This is an indication that the alert has a slight impact on the model performance.

BOLLINGER BANDS

This is an indicator designed to provide traders with information regarding price volatility. It is a very popular technique to determine if the market is overbought or oversold. The movement of the prices towards the bands is what decides the condition of the market i.e if the prices move closer to the upper band, the market is considered overbought(when to sell), otherwise, if they are closer to the lower band, the market is oversold(which is when to buy).

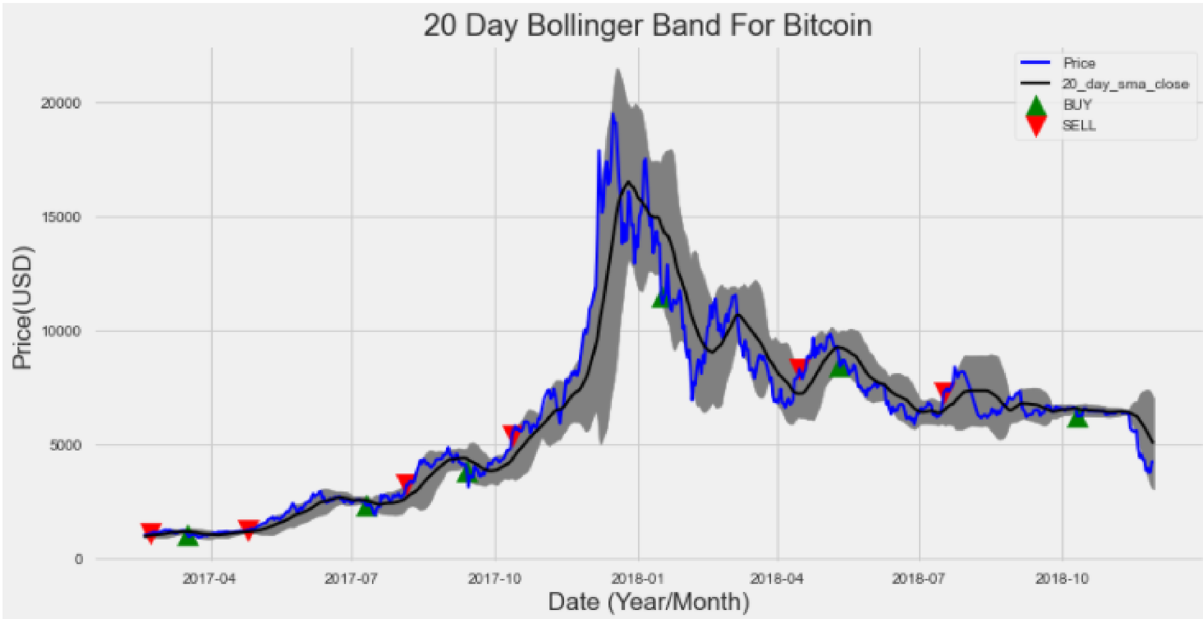
A **standard deviation** in this case is a measure of how far away a stock's price is from its typical/average price.The charts below shows the 2 standard deviation Bollinger band of bitcoin price as well as its 20 trading days moving average





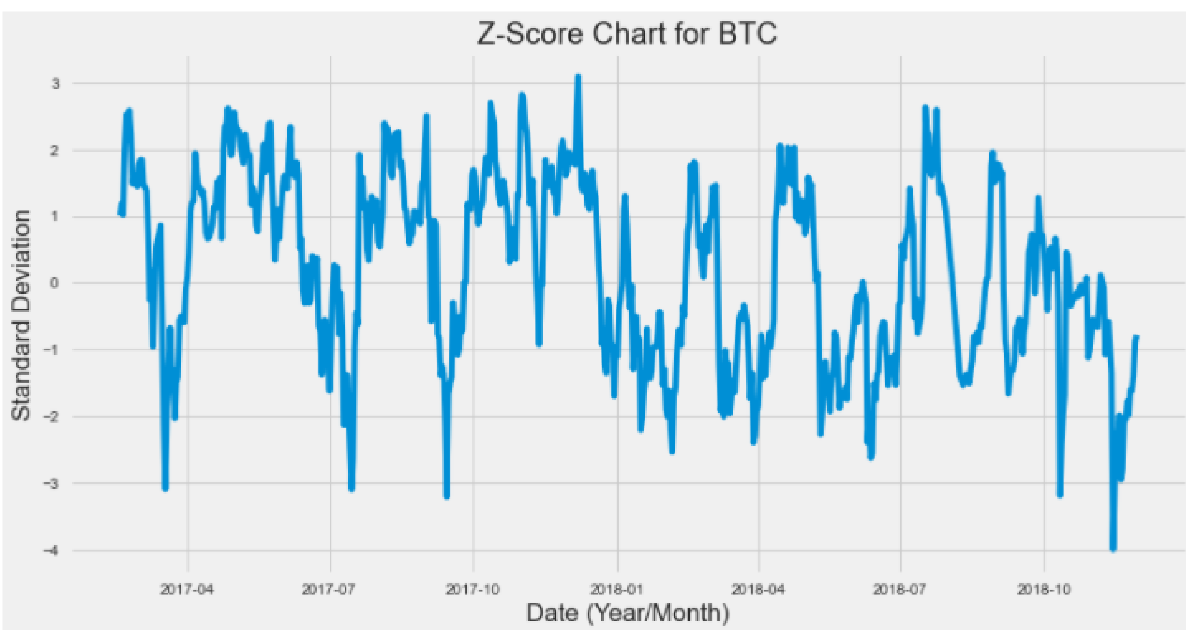
Notice that the stock's prices are within the bands 90% of the time. Most traders tend to buy a stock when it touches the lower band and then sell the stock when it touches the upper band. Oftentimes, the Bollinger band method works better with less volatile stocks i.e stocks that don't make huge moves as much as more likely to remain contained within the bands all though even less volatile stocks can bust right through the bands either to the upside or the downside.

TRADING STRATEGY #1 Bollinger Band.



The chart above shows a better picture of the 20 day Bollinger band for bitcoin where there are indicators that show when to buy or sell stocks. The grey area shows the full squeeze effect of upper and lower bands while the blue and the black lines represent stock price and its 20 day simple moving average respectively.

TRADING STRATEGY #2: BITCOIN PRICE Z-SCORE



The chart represents the z-score of the bitcoin price. It is obvious that prices are mostly within 2 standard deviations of the average price. Some interesting points to note on the chart are the spikes that occurred around late 2017 which indicates an abnormal movement of the market. The points where the z-score dip below zero indicates that it was a great opportunity to buy bitcoin. This is a metric that should be used in conjunction with other metrics to boost confidence in investment strategy.

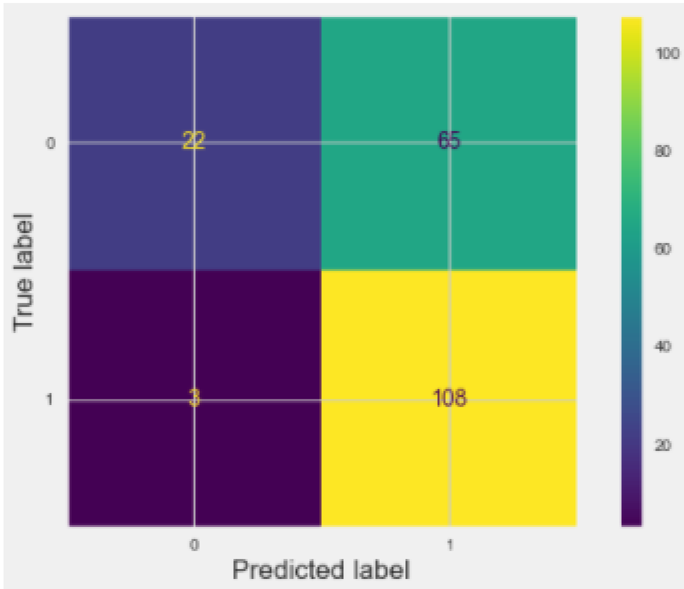
TRADING STRATEGY #3: LOGISTIC REGRESSION

A logistics regression was built to see if the price of bitcoin can be better predicted using metrics for accuracy as well as f1 score to measure the recall rate. The output variable was generated (buy/sell) based on the % change feature, that is, If the new percent change is greater than the old, then it's buy but if the new percent change is less than the old, it is time to sell.
The effectiveness of the model was evaluated based on how it correctly predicted classes as well as the recall value which allowed us to identify that the logistic regression model outperformed the linear regression model with accuracy rate of 66 percent for the model with alert feature.

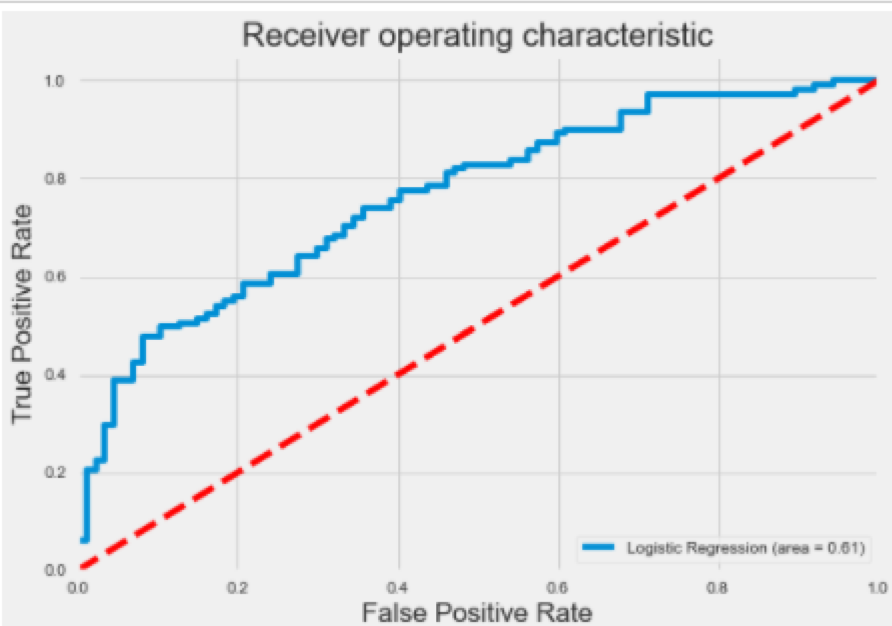
Logistic Model + Model with Alerts Feature

- **Added features**
  - high_daily_%_change
  - volume_daily_%_change
  - low_daily_%_change
  - alert_daily_%_change

- **Dependent Variable**
  - Change (Class 0 or 1)



	precision	recall	f1-score	support
0	0.88	0.25	0.39	87
1	0.62	0.97	0.76	111
accuracy			0.66	198
macro avg	0.75	0.61	0.58	198
weighted avg	0.74	0.66	0.60	198



```
Optimization terminated successfully.
         Current function value: 0.539455
         Iterations 7
                              Results: Logit
===========================================================================
Model:                   Logit               Pseudo R-squared:    0.212
Dependent Variable:      y                   AIC:                 717.9227
Date:                    2021-09-28 23:02    BIC:                 735.8795
No. Observations:        658                 Log-Likelihood:      -354.96
Df Model:                3                   LL-Null:             -450.45
Df Residuals:            654                 LLR p-value:         3.7365e-41
Converged:               1.0000              Scale:               1.0000
No. Iterations:          7.0000
---------------------------------------------------------------------------
                              Coef.  Std.Err.    z     P>|z|   [0.025  0.975]
---------------------------------------------------------------------------
high_daily_%_change          9.3883   4.0181  2.3365 0.0195  1.5129 17.2638
low_daily_%_change          27.7147   3.9251  7.0609 0.0000 20.0217 35.4078
volume_daily_%_change        1.2109   0.4404  2.7493 0.0060  0.3477  2.0741
alert__daily_count_%_change -0.6446   0.3365 -1.9154 0.0554 -1.3042  0.0150
===========================================================================
```

Logistic Model: Model using market data

- **Added features**
- ❖ high_daily_%_change
- ❖ volume_daily_%_change
- ❖ low_daily_%_change

- **Dependent Variable**
- ❖ Change (Class 0 or 1)

Classification table without alert data

The accuracy is 64 percent in this case which implies that the model with alert data is slightly better.

	precision	recall	f1-score	support
0	0.83	0.23	0.36	87
1	0.61	0.96	0.75	111
accuracy			0.64	198
macro avg	0.72	0.60	0.56	198
weighted avg	0.71	0.64	0.58	198

RESULTS: QUANTIFIED AUTOMATED TRADING PERFORMANCE


```
print('Portfolio Total Value as of 2018-11-29 ')
print(portfolio['total'].tail(1))

print('Absolute return as of 2018-11-29 ')

print((((portfolio['total'].tail(1)/float(1000)) - float(1))*100))
```

```
Portfolio Total Value as of 2018-11-29
date
2018-11-29    9502.38
Name: total, dtype: float64
Absolute return as of 2018-11-29
date
2018-11-29    850.238
Name: total, dtype: float64
```

The result above shows that a trader that started with $1000 in portfolio as of 2017 will have a portfolio total value of $9502.38 as of 2018-11-29.

FUTURE WORK

1. Investigates how much user activity predicts crypto market cap.
2. Use number of alerts to predict volume for the stock

CREDIT

Thanks to my mentor Vaughn DiMarco for his stellar advice and support throughout this project. You are such an amazing mentor.