Crypto Carousel

Evan Paliotta,
Daniel Klein Velderman
and Francisco Lopez

Summary

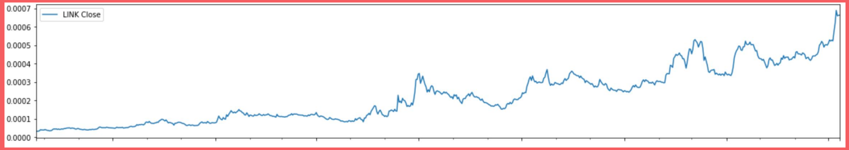
- Create and compare a short-term algorithmic trading strategy with a medium-term buy and hold strategy.
- Utilize the RSI and MACD for short-term strategy
- Implement regression and natural language processing in our buy and hold strategy.
- Apply daily BTC exchange rate data and focuses on nine currencies with three distinct use cases.
- We gathered data from the Binance API for both strategies and the Reddit API for NLP, using VADER
- Finally we compared the returns of the models and ultimately determined which strategy is more profitable to implement.
- TL;DR- Using ML and NLP to filter through altcoins, can we make more money with a short term trading strategy or a buy and hold strategy.

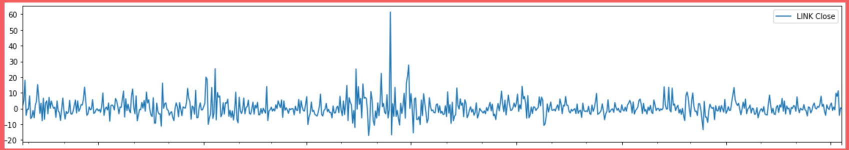
Model Summary

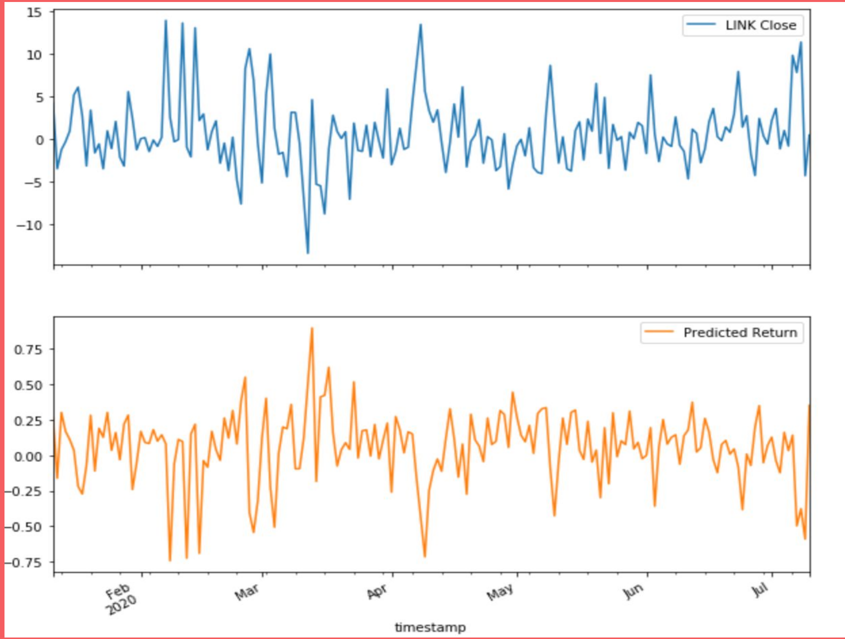
- Time-series price prediction naturally calls for regression
- Logistic: used to find probability of event (binary, success or failure)---classification problems
- Polynomial: used if the power of the independent variable is greater than 1. Risk overfitting
- Stepwise: used if there are multiple independent variables
- Ridge, Lasso, and ElasticNet apply penalties that did not seem to add too much value
- Linear: when the independent and dependent variable have a relationship
- **Regression to understand trends and filter through coins**
- **LSTM to predict price**

```python
# Constants
binsizes = {"1m": 1, "5m": 5, "1h": 60, "1d": 1440}
batch_size = 750

def minutes_of_new_data(symbol, kline_size, data, source):
    if len(data) > 0:  old = parser.parse(data["timestamp"].iloc[-1])
    elif source == "binance": old = datetime.strptime('1 Jan 2015', '%d %b %Y')
    if source == "binance": new = pd.to_datetime(binance_client.get_klines(symbol=symbol, interval=kline_size)[-1][0], unit='ms')
    return old, new

def get_all_binance(symbol, kline_size, save = False):
    filename = '%s-%s-data.csv' % (symbol, kline_size)
    if os.path.isfile(filename): data_df = pd.read_csv(filename)
    else: data_df = pd.DataFrame()
    oldest_point, newest_point = minutes_of_new_data(symbol, kline_size, data_df, source = "binance")
    delta_min = (newest_point - oldest_point).total_seconds()/60
    available_data = math.ceil(delta_min/binsizes[kline_size])
    if oldest_point == datetime.strptime('1 Jan 2015', '%d %b %Y'): print('Downloading all available %s data for %s. Be patient..!' % (kline_size, symbol))
    else: print('Downloading %d minutes of new data available for %s, i.e. %d instances of %s data.' % (delta_min, symbol, available_data, kline_size))
    klines = binance_client.get_historical_klines(symbol, kline_size, oldest_point.strftime("%d %b %Y %H:%M:%S"), newest_point.strftime("%d %b %Y %H:%M:%S"))
    data = pd.DataFrame(klines, columns = ['timestamp', 'open', 'high', 'low', 'close', 'volume', 'close_time', 'quote_av', 'trades', 'tb_base_av', 'tb_quote_av', 'ignore' ])
    data['timestamp'] = pd.to_datetime(data['timestamp'], unit='ms')
    if len(data_df) > 0:
        temp_df = pd.DataFrame(data)
        data_df = data_df.append(temp_df)
    else: data_df = data
    data_df.set_index('timestamp', inplace=True)
    if save: data_df.to_csv(filename)
    print('All caught up..!')
    return data_df
```

	LINK Close	NANO Close	XMR Close	ZIL Close	NEO Close	ADA Close	VET Close	WTC Close	WABI Close
timestamp									
2018-07-25	0.00003003	0.00029750	0.01752800	0.00000942	0.00417300	0.00002113	0.00000258	0.00073210	0.00004028
2018-07-26	0.00003167	0.00027920	0.01720000	0.00000923	0.00416800	0.00002069	0.00000315	0.00071500	0.00003900
2018-07-27	0.00003228	0.00027540	0.01714300	0.00000911	0.00408200	0.00002026	0.00000307	0.00075650	0.00003970
2018-07-28	0.00003396	0.00026960	0.01704800	0.00000890	0.00412800	0.00001994	0.00000312	0.00074460	0.00003992
2018-07-29	0.00004009	0.00026580	0.01650200	0.00000879	0.00406700	0.00001980	0.00000322	0.00071170	0.00003889







- Top Left: function to pull data
- Bottom Left: dataframe
- Top Right: coin/BTC price (trend)
- Middle Right: pct_change plot (volatility)
- Bottom Right: loss function plot

Regression

Mean squared error: average of the square of the errors

Root mean squared error: standard deviation of the residuals

Max error: maximum residual error

XMR, ADA and LINK

```
--------------------Mean Squared Error---------------
LINK Mean Squared Error: 17.91184899916619
NANO Mean Squared Error: 18.11083317070283
XMR Mean Squared Error: 5.34620633007903
ZIL Mean Squared Error: 35.22338259899112
NEO Mean Squared Error: 7.852714657467821
ADA Mean Squared Error: 12.490094315050895
VET Mean Squared Error: 25.98666713208893
WTC Mean Squared Error: 31.524385951094562
WABI Mean Squared Error: 29.081842797203546
------------------Root Mean Squared Error------------
LINK Root Mean Squared Error: 4.232239241721359
NANO Root Mean Squared Error: 4.255682456516561
XMR Root Mean Squared Error: 2.3121864825482894
ZIL Root Mean Squared Error: 5.934929030661708
NEO Root Mean Squared Error: 2.802269554748048
ADA Root Mean Squared Error: 3.534132752890148
VET Root Mean Squared Error: 5.097711950678356
WTC Root Mean Squared Error: 5.614658133056238
WABI Root Mean Squared Error: 5.392758366291183
--------------------Max Error-------------------
LINK Max Error: 13.944501875488347
NANO Max Error: 15.635768112071073
XMR Max Error: 9.595909553943436
ZIL Max Error: 24.08060889798841
NEO Max Error: 16.201799803153207
ADA Max Error: 15.41786594248329
VET Max Error: 22.49025175651177
WTC Max Error: 19.69328509652831 4
WABI Max Error: 24.690874080746305
```

Reddit NLP

Valence Aware Dictionary and sEntiment Reasoner

- Cardano, Monero & Chainlink based on ML results
- Scrape posts and comments from Reddit
- Reddit API & Python PushShift API Wrapper (PSAW)
- Utilized CryptoCurrency subreddit due to less sentiment bias
- Chainlink most positive

Code and visualizations

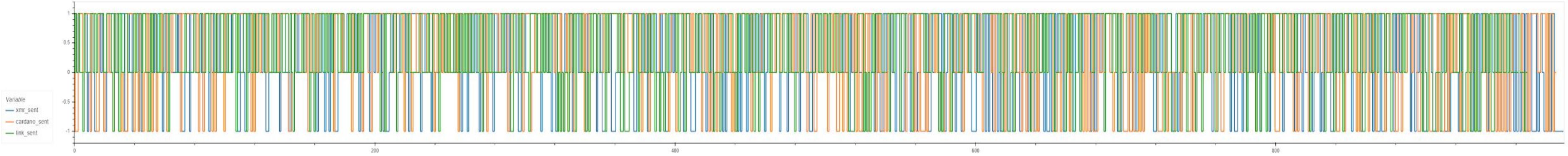
SCRAPING CARDANO POSTS

```
#loop through reddit api to collect post data
after = None
list_posts = []
for _ in range(20):
    posts = requests.get('https://www.reddit.com/r/cardano.json',
                         headers = {'User-agent': 'u/brutprestige'},
                         params = {'after': after, 'limit': None}).json()
    after = posts['data']['after']
    for post in posts['data']['children']:
        list_posts.append(post['data'])
    time.sleep(1)

#create dataframe
cardano_posts = pd.DataFrame(list_posts)

cardano_posts.head()
```

SCRAPE CARDANO COMMENTS IN CRYPTOCURRENCY SUBREDDIT

```
start_epoch=int(dt.datetime(2020, 6, 1).timestamp())

cardano_comm = api.search_comments(after=start_epoch,
                                   subreddit='CryptoCurrency',
                                   q='cardano',
                                   filter=['body'],
                                   limit=None)
```

Results & observations



	xmr_compound	xmr_pos	xmr_neu	xmr_neg	xmr_sent	cardano_compound	cardano_pos	cardano_neu	cardano_neg	cardano_sent	link_compound	link_pos	link_neu	link_neg	link_sent
count	992.000000	992.00000	992.000000	992.000000	992.000000	987.000000	987.000000	987.000000	987.000000	987.000000	968.000000	968.000000	968.000000	968.000000	968.000000
mean	0.254190	0.12093	0.830821	0.048240	0.345766	0.274967	0.130312	0.812032	0.057641	0.398176	0.271449	0.132503	0.834239	0.030161	0.415289
std	0.504302	0.14674	0.159404	0.081266	0.793771	0.521481	0.150938	0.174866	0.114458	0.779112	0.414655	0.181651	0.197611	0.091232	0.673823
min	-0.990300	0.00000	0.000000	0.000000	-1.000000	-0.970800	0.000000	0.000000	0.000000	-1.000000	-0.970900	0.000000	0.000000	0.000000	-1.000000
25%	0.000000	0.00000	0.762000	0.000000	0.000000	0.000000	0.000000	0.734500	0.000000	0.000000	0.000000	0.000000	0.755000	0.000000	0.000000
50%	0.294200	0.08750	0.847000	0.000000	1.000000	0.352700	0.101000	0.837000	0.000000	1.000000	0.202300	0.079000	0.878500	0.000000	1.000000
75%	0.691375	0.17400	0.964250	0.069000	1.000000	0.749650	0.187000	0.945000	0.068000	1.000000	0.612400	0.200000	1.000000	0.000750	1.000000
max	0.997500	1.00000	1.000000	0.608000	1.000000	0.999500	1.000000	1.000000	1.000000	1.000000	0.998400	1.000000	1.000000	1.000000	1.000000

- Next to sentiment, the results also give insight in the attention a certain project gets in the social network
- Analysis return quite a significant amount of ‘neutrals’ - would be interesting to further adjust lexicon
- Reddit API too limited in terms of results. Tried various other APIs and so-called wrappers which eventually gave what we were looking for
- Determined whether coin specific subreddits were to biased (positive)
- Combining results from submissions as well as comments into one dataframe for sentiment testing
- Link most positive based on normalized score ‘sent’

LSTM

ChainLink

- Clearly overfit
- Average runtime of each iteration was 75 seconds
- Wish we had more time to play around with it

```
# 1 month buy and hold strategy pnl
last_price=valid.iloc[-1]['close']
n_days_ago=valid.iloc[-30]['close']
investment=100000

outcome=(1+(last_price-n_days_ago)/n_days_ago)*investment
print(f'A ${investment} investment made 30 days ago would be worth ${outcome:.2f} today')
```

```
A $100000 investment made 30 days ago would be worth $153644.57 today
```

Algo Trading

Sorted original coins based on volatility

LINK, WET, WTC

RSI, SOSC, RSI/MAC

LINK:

- RSI Investment Strategy
- Random forest

Prices / BTC

Chainlink

RSI

Entry Signal = 1 (long)

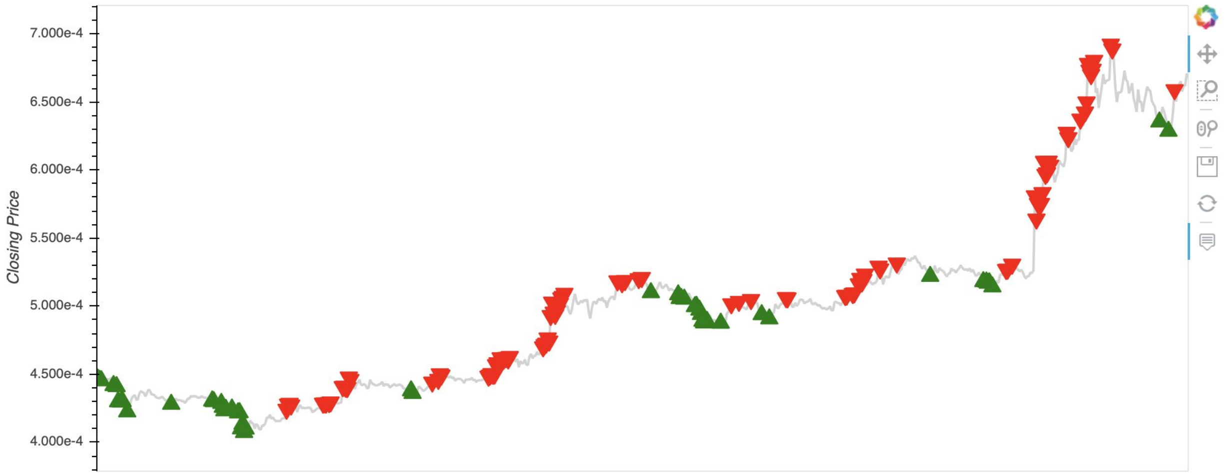
RSI < 30 , Buy

Exit Signal = -1 (short)

RSI < 30, Sell

X Unlimited resources

X Long/Short



[12]:

	Backtest
Annual Return	-0.150829
Cumulative Returns	-0.503687
Annual Volatility	0.434465
Sharpe Ratio	-0.347161
Sortino Ratio	-0.477734

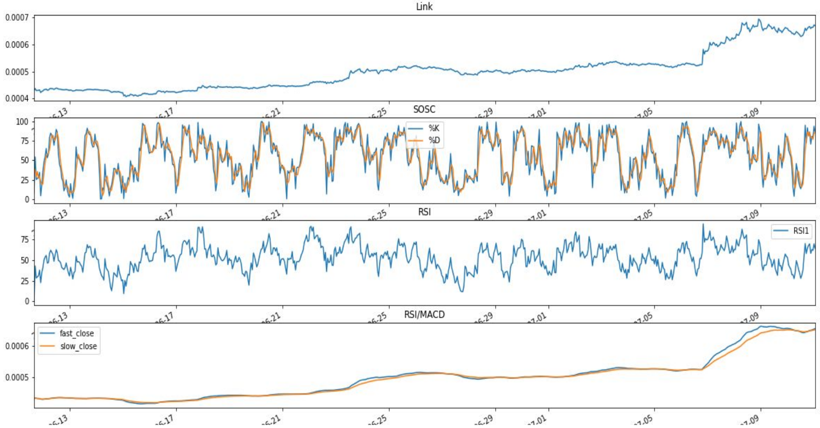
Try other Strategies
+
Other coins

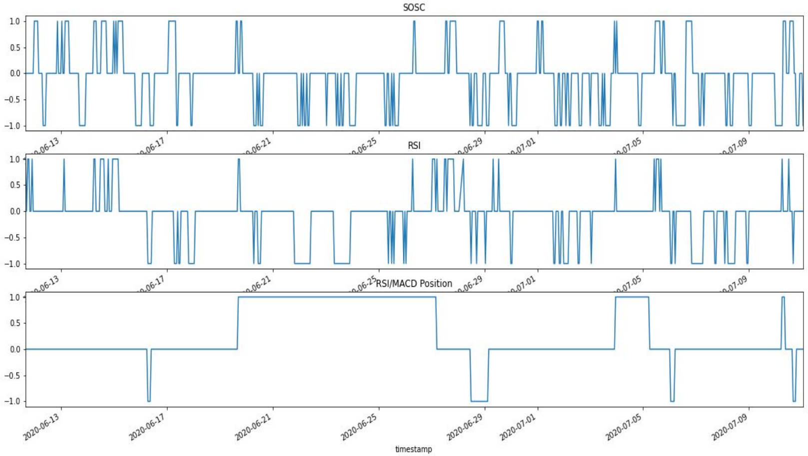
RSI.
Buy Entry | RSI < 30, Buy Exit | RSI > 30
Sell Entry | RSI > 70, Sell Exit | RSI < 70

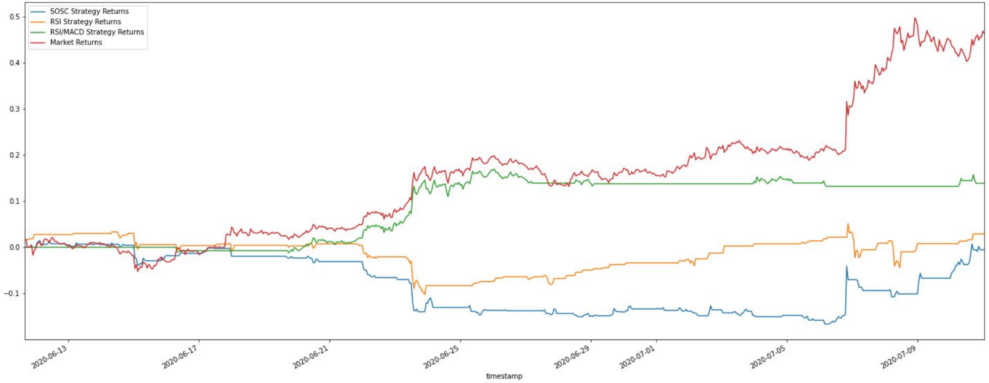
SOSC.
Buy Entry | %K > %D. Buy Exit |%K < %D
Sell Entry | %K < %D, Sell Exit | %K > %D

RSI/MACD.
Buy Entry | fast_close > slow_close + RSI < 30
Buy Exit | fast_close < slow_close
Sell Entry |fast_close < slow_close + RSI > 70
Sell Exit | fast_close > slow_close

Chainlink LINK







Chainlink

Random Forest Training

Separate X and y Training Datasets

[73]:
```
# Construct the X_train and y_train datasets
X_train = trading_signals_df[x_var_list][training_start:training_end]
y_train = trading_signals_df['Positive Return'][training_start:training_end]

X_train.tail()
```

[73]:

timestamp	RSI Position	SOSC Position	RSI/MACD Position
2020-06-30 19:00:00	0.0	0.0	0.0
2020-06-30 20:00:00	0.0	0.0	0.0
2020-06-30 21:00:00	0.0	0.0	0.0
2020-06-30 22:00:00	0.0	0.0	0.0
2020-06-30 23:00:00	0.0	0.0	0.0

[74]:
```
y_train.tail()
```

[74]:
```
timestamp
2020-06-30 19:00:00    1.0
2020-06-30 20:00:00    0.0
2020-06-30 21:00:00    0.0
2020-06-30 22:00:00    1.0
2020-06-30 23:00:00    0.0
Name: Positive Return, dtype: float64
```

Train Random Forest Model

[42]:
```
# Fit a SKLearn linear regression using just the training set (X_train, Y_train):
model = RandomForestClassifier(n_estimators=100, max_depth=3, random_state=0)
model.fit(X_train, y_train)

# Make a prediction of "y" values from the X_test dataset
predictions = model.predict(X_test)

# Assemble actual y data (Y_test) with predicted y data (from just above) into two
Results = y_test.to_frame()
Results["Return"] = link_data['daily_return'].loc['2020-07-01':'2020-07-11']
Results.head
```

[42]:

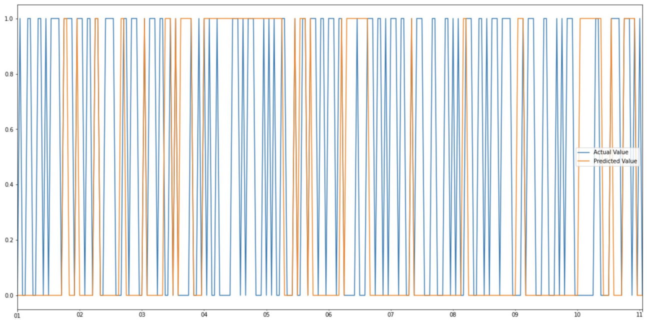
timestamp	Positive Return	Return
2020-07-01 00:00:00	0.0	-0.002383
2020-07-01 01:00:00	1.0	0.001224
2020-07-01 02:00:00	0.0	-0.001223
2020-07-01 03:00:00	0.0	-0.000943
2020-07-01 04:00:00	1.0	0.001607
2020-07-01 05:00:00	1.0	0.002828
2020-07-01 06:00:00	0.0	-0.005861
2020-07-01 07:00:00	0.0	-0.000060
2020-07-01 08:00:00	1.0	0.012557

Random Forest Trading

Plot Predicted Results vs. Actual Results

[21]:
```
# Plot predicted results vs. actual results
results[['Actual Value', 'Predicted Value']].plot(figsize=(20,10))
```

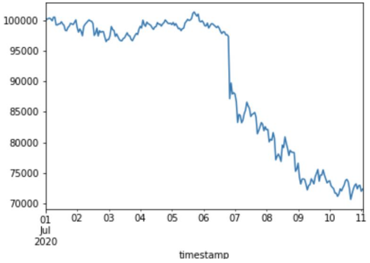
[21]: <matplotlib.axes._subplots.AxesSubplot at 0x1a192200d0>

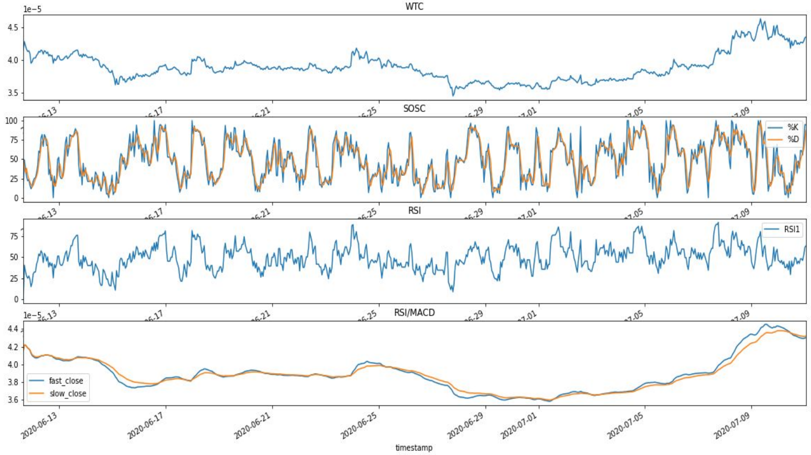


Plot Cumulative Return of Random Forest Model (In Terms of

```
# Set initial capital allocation
initial_capital = 100000

# Plot cumulative return of model in terms of capital
cumulative_return_capital = initial_capital * (1 + (results['Return']
cumulative_return_capital.plot()
```

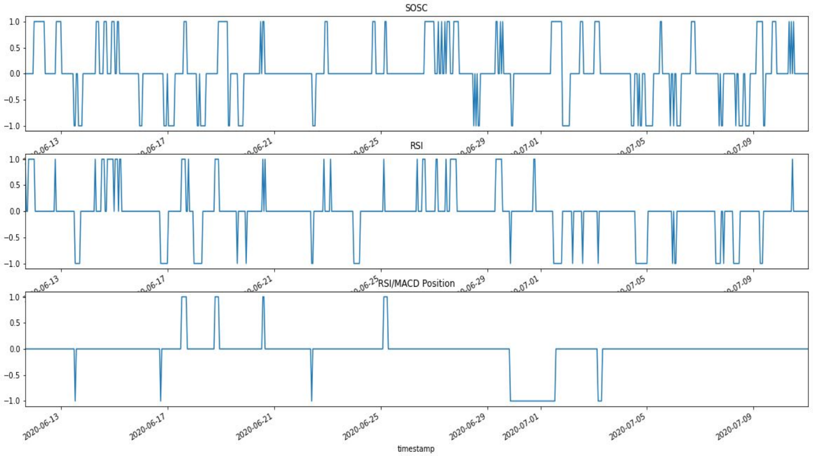
<matplotlib.axes._subplots.AxesSubplot at 0x1a19d33ad0>

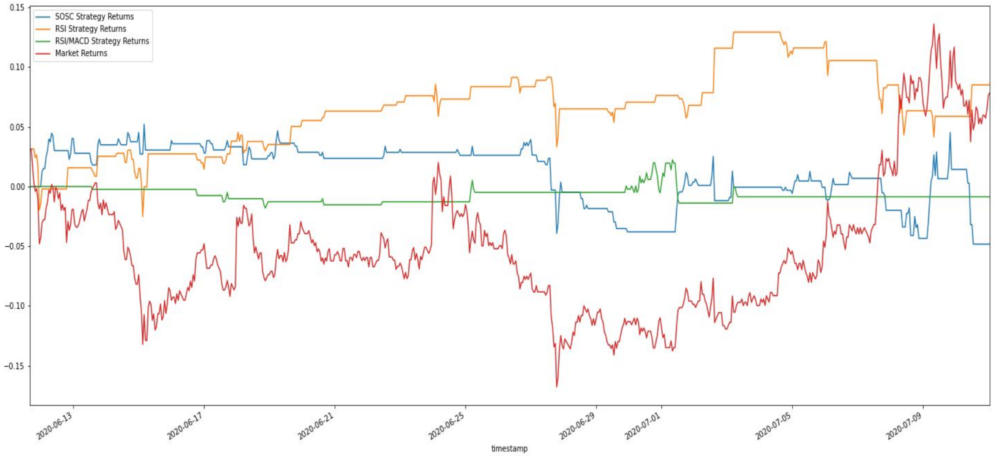




Walton WTC

Try RSI Strategy!





Discussion

Algo Strategies

- Chainlink. Buy and hold strategy outperform
- Random Forest training data was heavily influenced by RSI.
- WTC. Better to execute RSI strategy.
- Binance does not provide coin prices / usd for all the coins.
- Going live!

Regression and LSTM Models

- Regression analysis provided meaningful graphs and metrics to compare coins--**Sufficient**
- LSTM was overfit--**Insufficient**
- 54% return was dumb luck
- Crypto is volatile and a relatively nascent asset class. It's difficult to measure fundamentals.

NLP

- Next to sentiment, the results also give insight in the attention a certain project gets in the social network
- Analysis return quite a significant amount of 'neutrals' - would be interesting to further adjust lexicon
- Reddit API too limited in terms of results. Tried various other APIs and so-called wrappers which eventually gave what we were looking for
- Determined whether coin specific subreddits were to biased (positive)
- Combining results from submissions as well as comments into one dataframe for sentiment testing

Postmortem

ML:

- Collecting the data was the most difficult task
- See how the models performs with new data

NLP:

- Add more social network resources to the sentiment analysis (e.g. Discord, Telegram)
- Further optimize sentiment lexicon to align perfectly with social network language
- Further develop streaming Reddit & sentiment data enabling real-time sentiment analysis, food for thought, does it really bring meaningful investment data?
- Useful to apply Recurrent neural networks? Long-short term memory model to score the sentiment; needs human sentiment grading upfront, what is the benefit?

ALGO:

- Try new strategies with different indicators.
- Combine strategies build more creative entry and exit signals.

Q & A