Algothon 2019

Conv-LSTM Multi-factor Trading Signal Processing Model

Quandl: End Of Day US Stock Prices, Global Supply Chain Relationships (GSCR), Ekson Data API, Social Media Analytics (SMA1)

Project Highlights:

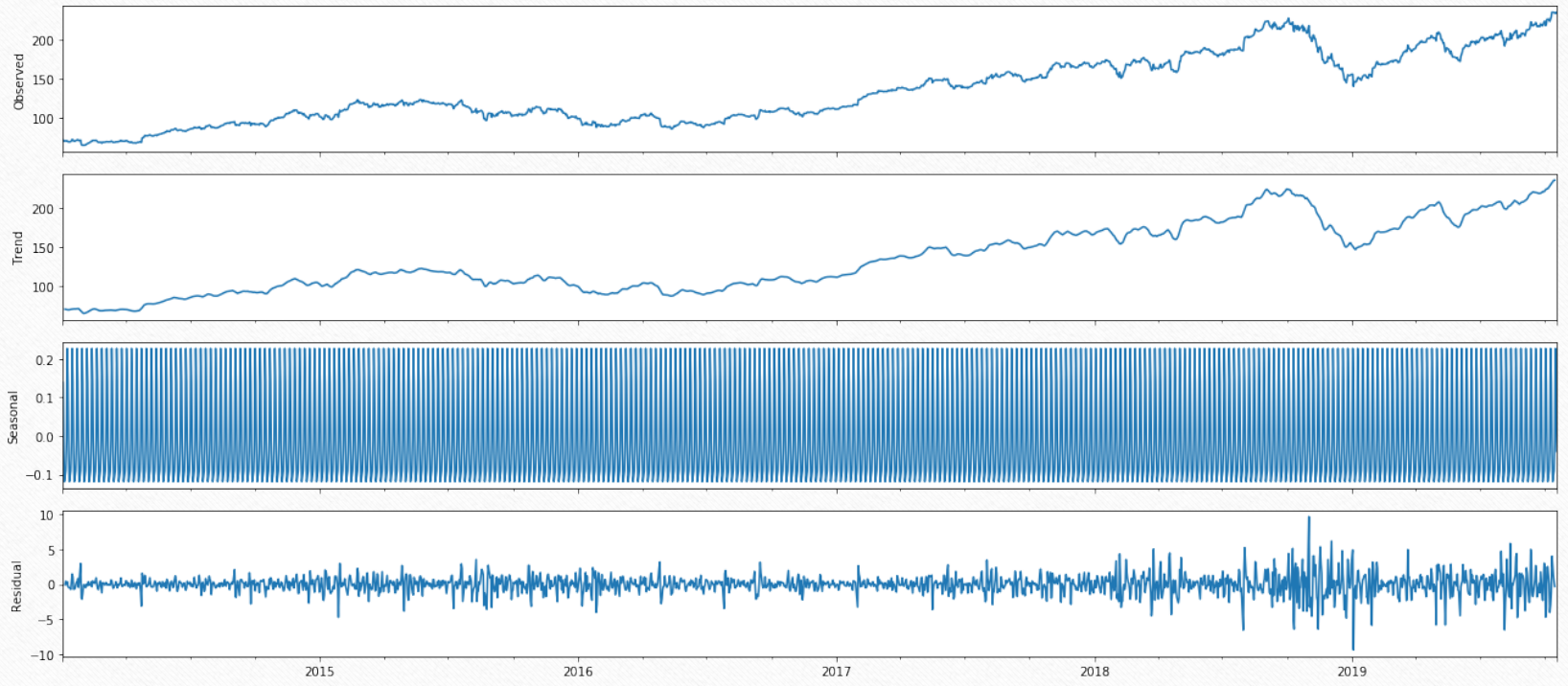
- Statistical modelling on selected US stock, de-trending and find seasonality.
- Forecasting price movement based on SARIMA model with hyperparameter tuning on AWS with optimization in parallel computing.
- Exploratory data analysis on social media sentiment scores and engagement scores to identify feature with highest data quality and relevance.
- Use Principle Component Analysis (PCA) to reduce the dimensions and complexity of features, making model more robust.
- Conduct chain analysis to find one's most related firm.
- Multi-Variate Encoder-decoder Conv-LSTM on sentiment, supply chain and stock prices of relevant firms. Achieved promising result of 76% AUC-ROC result for testing datasets of 3 totally different firms and models, predicting price change after 5 days.
- Extra analysis on how shareholder scores are negatively correlated to stock prices

1. Statistical Analysis On Stock Price.

Before we conduct machine learning models, we need to apply statistical models on time-series to understand the basic properties of the time series. Understand the statistical property will greatly help us to decide which machine learning model we could take and what pitfalls we should be aware of.

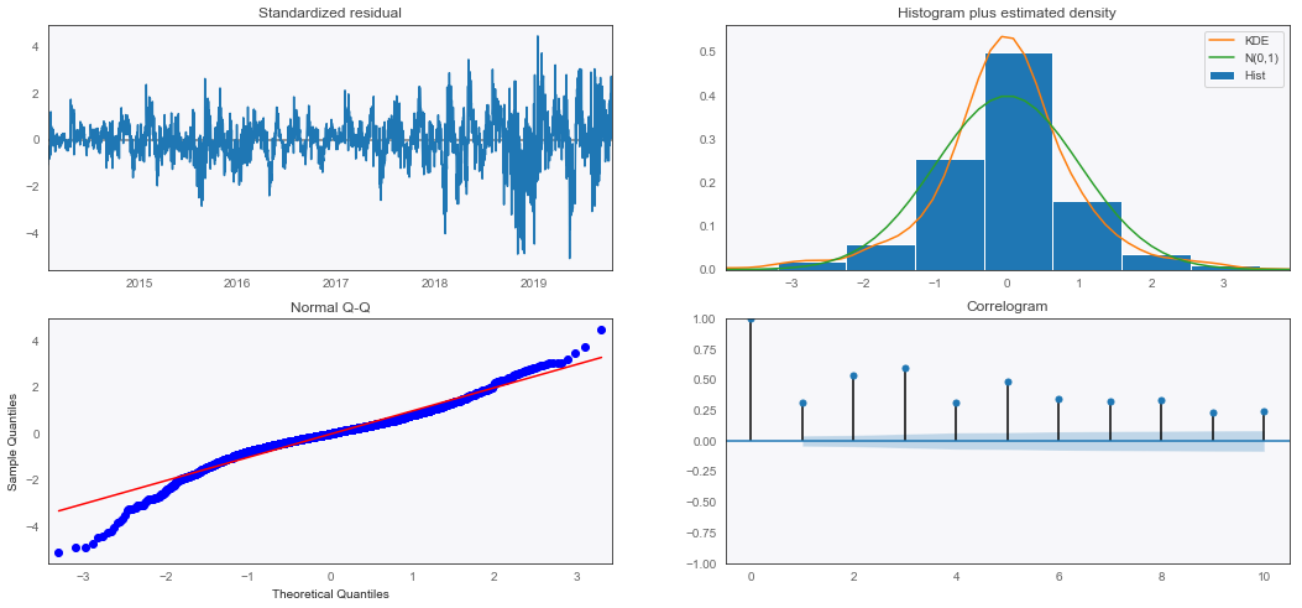
Take an example of ticker 'AAPL'

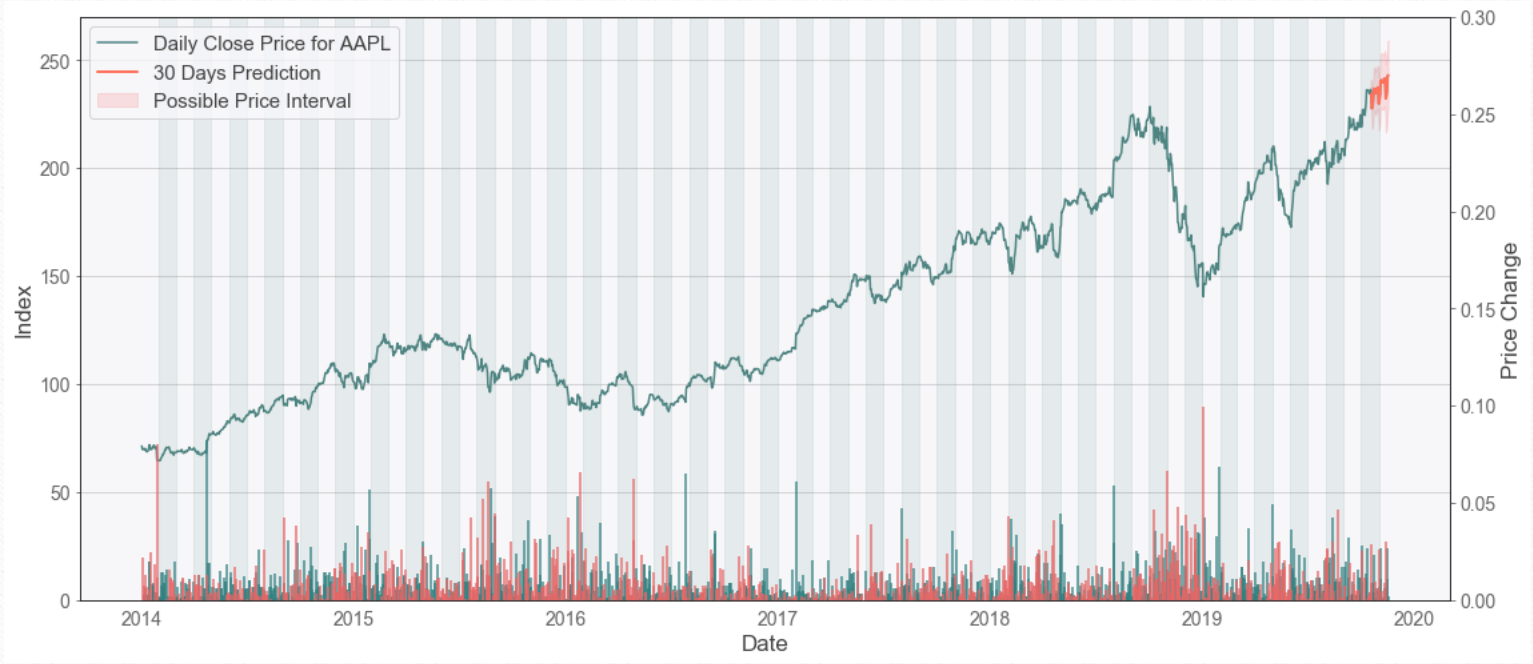
Time series decomposition into trend and seasonality



Multi-CPU grid search for best parameter of SARIMA model

Using AWS to conduct grid search for best hyper-parameter of the time series by fitting SARIMA model. The best parameter given is [(0, 0, 2), (1, 0, 2, 12), 'n'], but it still has huge standard error.

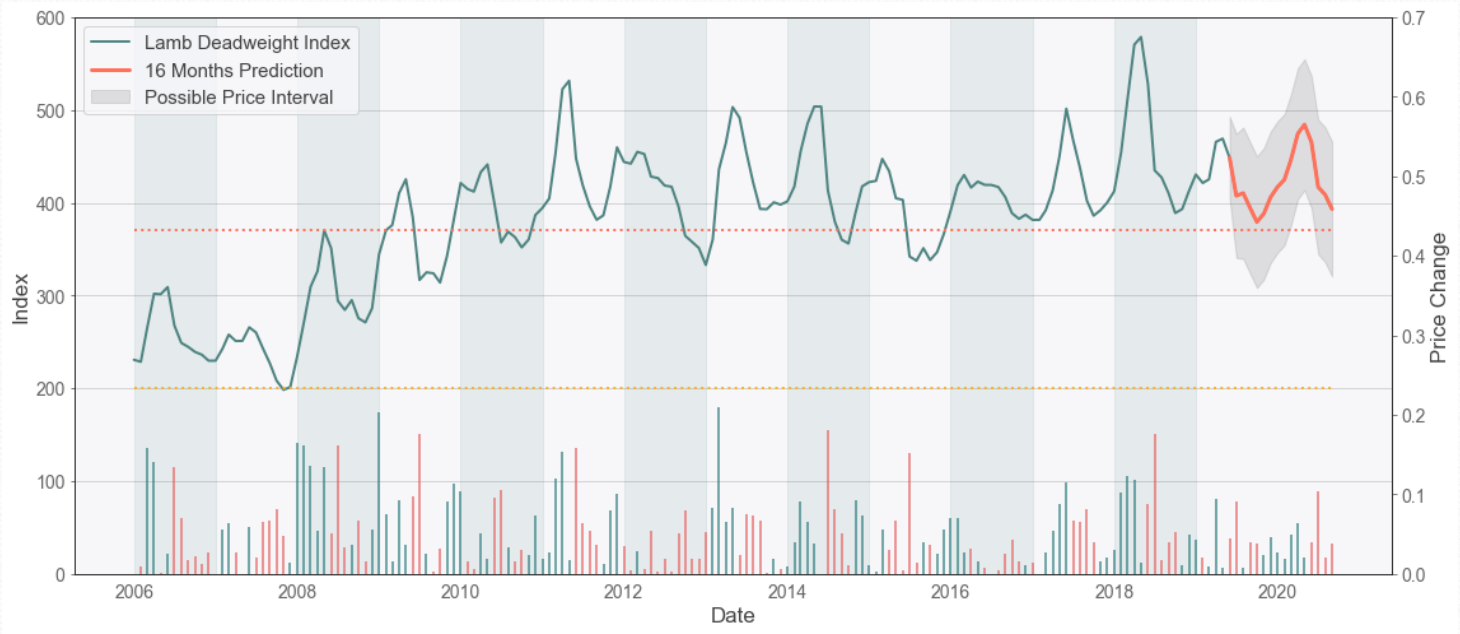




Such SARIMA model gives the prediction as above. It looks reasonable, but since the standard error in the last page is very large, we could see a very broad range of confident interval shaded in pink area. It did not give great result at this case.

Example of good time-series model

It could gave very good result to commodities indexes where there are higher seasonality. Example the lamb price:

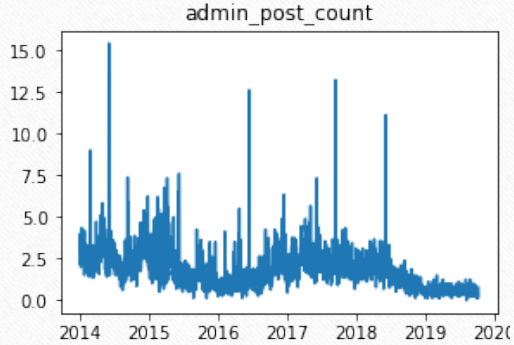


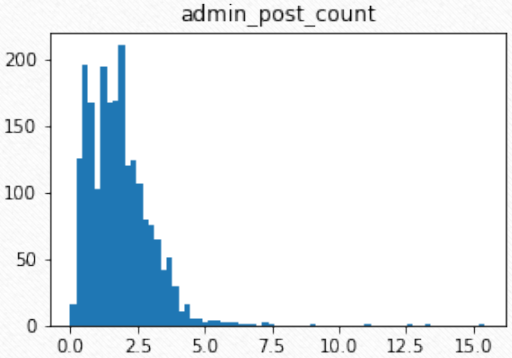
Social Media Sentiment Scoring Model And Data Quality Checking By EDA

The previous analysis means ‘AAPL’ does have much trend from itself and from the past/current prices. This also means directly conduct predictive machining learning model on the univariate stock price itself will not achieve significant result because of lack of statistical evidence and auto_correlation.

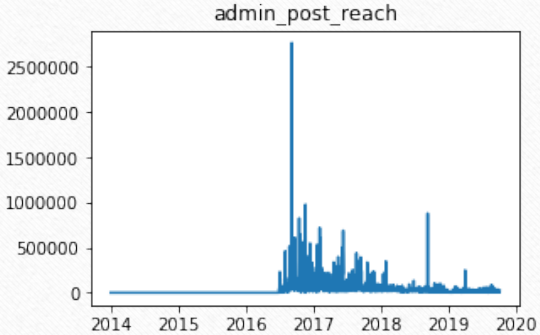
Then we decide to use multi-variate Conv-LSTM in this project. We want to maximize the information the machine learning model we could have before prediction. The first step is to pick the most relevant data, from social network, from the supply chain and from the external factors. Here are the explained descriptions of what's going on in my project.

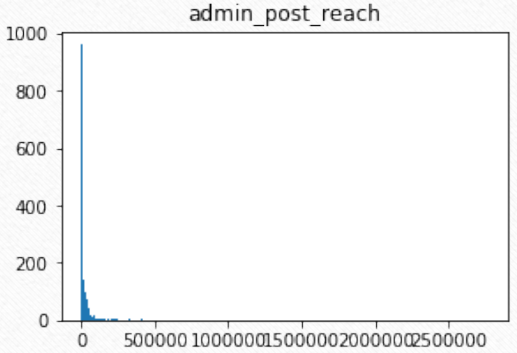
Example of relevant and good quality data



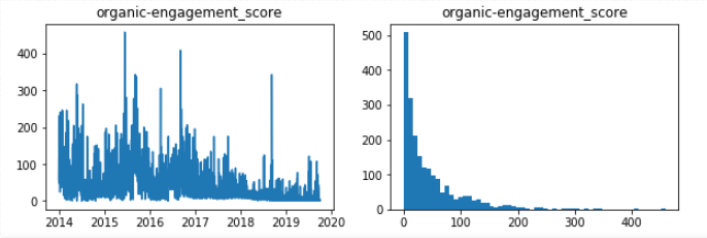


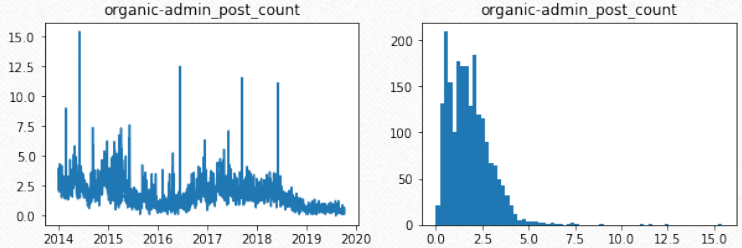
Example of irrelevant and bad quality data





So we keep ['fan_post_count','organic-admin_post_count','organic-admin_post_likes','organic-admin_post_comments','organic-admin_post_shares','organic-engagement_score'] as one of the most important. Same applied to all other social media data feather selection.





The I Use PCA to reduce the dimension while keep the information. PCA can be thought of as fitting a p-dimensional ellipsoid to the data, where each axis of the ellipsoid represents a principal component. If some axis of the ellipsoid is small, then the variance along that axis is also small, and by omitting that axis and its corresponding principal component from our representation of the dataset, we lose only a commensurately small amount of information.

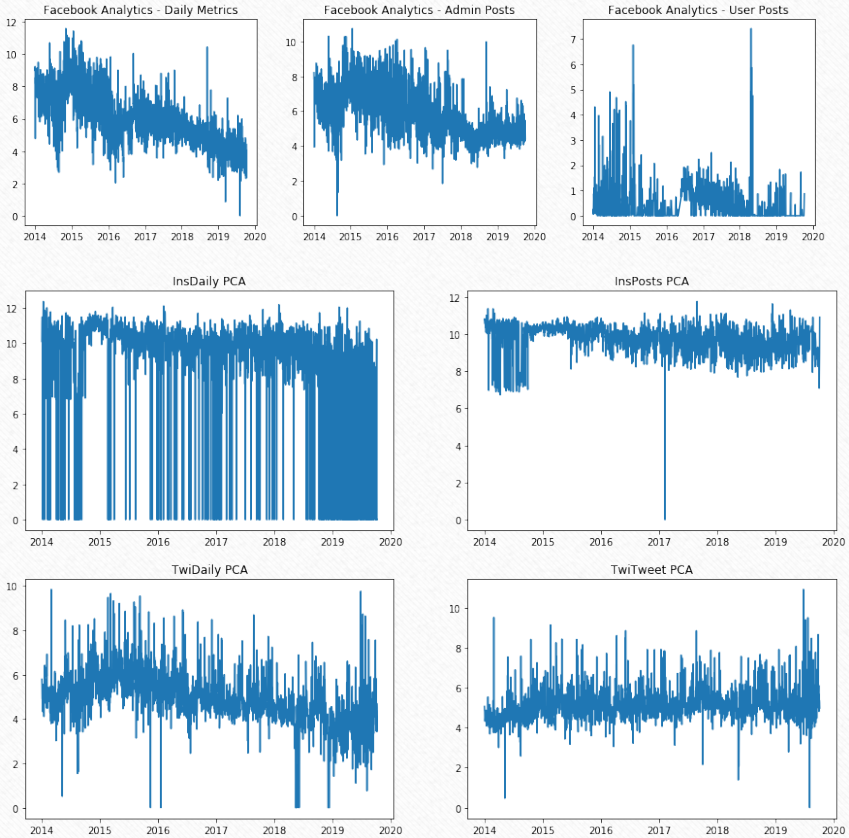
PCA is mostly used as a tool in exploratory data analysis and for making predictive models

The post, share, likes and sentiment combined PCA value.

```
print(np.shape(facebook_daily))
print(np.shape(facebook_admin))
print(np.shape(facebook_user)) # remvoe this one
print(np.shape(InsDaily))
print(np.shape(InsPosts)) # remvoe this one
print(np.shape(TwiDaily))
print(np.shape(TwiTweet))
```

```
(2106, 2)
(2103, 2)
(1351, 2)
(2106, 2)
(1870, 2)
(2106, 2)
(2101, 2)
```

We remove the two PCA value which has the most inconsistent and unexpected shapes. From above two results, remove facebook_user and InsPosts and merge the rest of data



The shape of the data after interpolation of missing value

From above two results, remove facebook_user and InsPosts and merge the rest of data

```
final_df=facebook_daily.join(facebook_admin,how='outer').join(InsDaily,how='outer').join(TwiDaily,how='outer').join(TwiTweet,how
final_df.head()
```

date	Facebook Analytics - Daily Metrics	organic-engagement_score	Facebook Analytics - Admin Posts	facebook admin sentiment	InsDaily PCA	InsDaily engagement_score	TwiDaily PCA	TwiDaily engagement_score	TwiTweet PCA	TwiTweet engagement_score
2014-01-01	8.998135	57.800000	8.208328	0.727273	10.097576	496.5	5.783886	166.666667	5.062437	163.875000
2014-01-02	9.234711	113.333333	7.843942	0.333333	11.235873	498.0	5.510158	156.666667	4.862481	172.555556
2014-01-03	9.063719	71.500000	7.962887	0.277778	11.476927	496.5	4.916147	128.666667	4.817946	188.166667
2014-01-04	4.775993	59.166667	3.942780	0.000000	11.348707	493.5	4.870315	84.000000	4.328116	109.375000
2014-01-05	8.558782	52.200000	7.863399	0.600000	10.972148	490.5	4.747260	90.333333	4.341823	91.000000

Every methods keeps one sentiment score and one PCA values in order to maximize the information and while reduce the complexity to avoid overfitting problem.

Analyze supply chain systems

```
def getMostImportantSuppliers(ticker):
    '''
    Input a ticker name, then output the stock price o 3 most important customer in the supply chain.
    '''
    data_1 = quandl.get_table('GSCR/GSPLY',paginate=True,supplier_ticker=ticker)

    df_selected=pd.concat([data_1.groupby("customer_name").sum()[["revenue_dependency"]],data_1.groupby("customer_name").count()
    customer_selected=df_selected.sort_values(by=["supplier_name","revenue_dependency"],ascending=False).index[:3].values.tolist
    customer_selected_ticker=pd.concat([data_1.groupby("customer_ticker").sum()[["revenue_dependency"]],data_1.groupby("customer_

    data_3 = quandl.get_table('GSCR/GSREF',ticker=customer_selected_ticker,paginate=True)
    data_3=data_3.drop(columns=["publish_date"]).drop_duplicates()
    Final_Result=data_3[data_3['company_name'].isin(customer_selected)]
    comb=[]
    for i in Final_Result.ticker:
        df = quandl.get('EOD/'+i, start_date='2014-01-01', end_date='2019-10-18')
        df = df.loc[:,['Adj_Close']]
        df = df.rename({'Adj_Close':i+' Price'},axis=1)
        comb.append(df)
    comb = pd.concat(comb,axis=1,sort=True)
    return comb

supply_chain_price = getMostImportantSuppliers(ticker)
supply_chain_price.head()
```

Date	BBY Price	S Price	VZ Price
2014-01-02	33.810433	10.40	37.359746
2014-01-03	33.960702	9.94	36.917529
2014-01-06	32.900473	9.77	37.123389
2014-01-07	32.040603	9.87	37.588479
2014-01-08	31.589797	9.98	37.382620

We believe if the customers of the firm are suffering the decreasing revenue, the supplier should also suffer from loss of revenue and hence drop in stock price.

This function goes in a ticker, then search for its customers and gives the most related firms share price.

Combine datasets

Sentiment score is available at weekend but the market price data is not. So i use algorithms to calculate average sentiment scores for the weekend and append it to Monday data

```
]: #download stock price timeseries
price = quandl.get('EOD/'+ticker, start_date='2014-01-01', end_date='2019-10-18')
price['date']=price.index
price=price.loc[:,['Adj_Close','Adj_Volume','date']]

#merge everything
df=price.join(final_df,how='outer').join(supply_chain_price,how='outer')
```

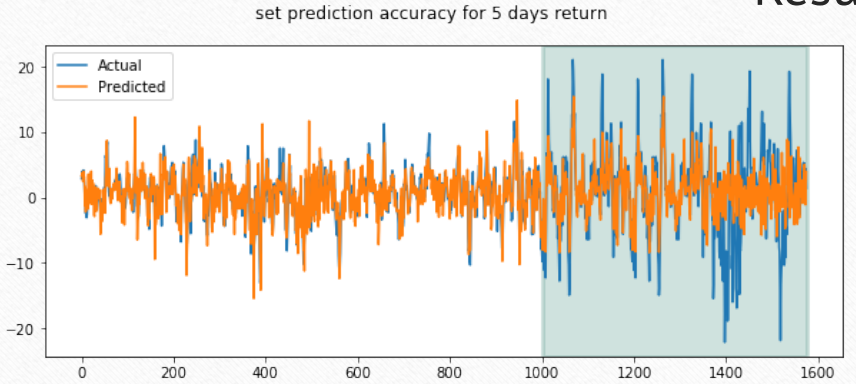
```
]: def interpolate_dataset_style2(dataset):
    for ii,i in enumerate(dataset.index):
        for jj,j in enumerate(dataset):
            if str(dataset.iloc[ii,jj])=='nan':
                try:
                        dataset.iloc[ii,jj]=dataset.iloc[ii-1,jj]
                except:
                        dataset.iloc[ii,jj]=dataset.iloc[ii+1,jj]
    return dataset
df=interpolate_dataset_style2(df)
df=df.groupby('date').mean()
```

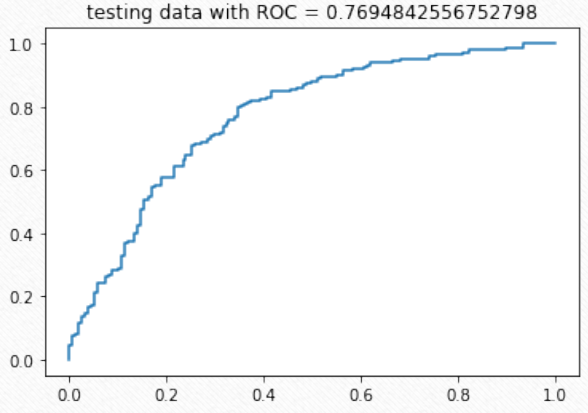
Build Conv-LSTM encoder-decoder Model to predict 5 days stock returns

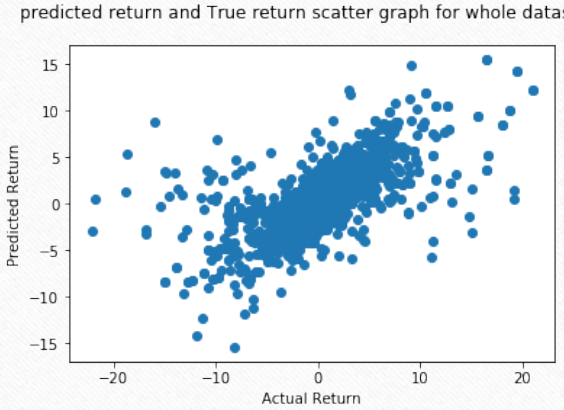
```
]: ######################################### train the model ##########################################

train, test = split_dataset(dataset,n_input)
train_x, train_y = to_supervised(train, n_input,n_out,delay)
# prepare data
verbose, epochs, batch_size = 0, 15, 16
n_timesteps, n_features, n_outputs = train_x.shape[1], train_x.shape[2], 1
# reshape into subsequences [samples, time steps, rows, cols, channels]
train_x = train_x.reshape((train_x.shape[0], n_steps, 1, n_length, n_features))
train_y = train_y.reshape((train_y.shape[0], 1, 1))
model = Sequential()
model.add(ConvLSTM2D(filters=40, kernel_size=(1,5), kernel_initializer='glorot_uniform', return_sequences=True
                     ,unit_forget_bias=False, activation='tanh', recurrent_activation='hard_sigmoid',
                     go_backwards=True, input_shape=(n_steps, 1, n_length, n_features)))
model.add(BatchNormalization())
model.add(Flatten())
model.add(RepeatVector(n_outputs))
model.add(LSTM(50, activation='tanh', return_sequences=True))
model.add(BatchNormalization())
model.add(TimeDistributed(Dense(10, activation='relu')))
model.add(BatchNormalization())
model.add(TimeDistributed(Dense(1)))
model.compile(loss='mse', optimizer='adam')# fit network
model.fit(train_x, train_y, epochs=epochs, batch_size=batch_size, verbose=verbose)
```

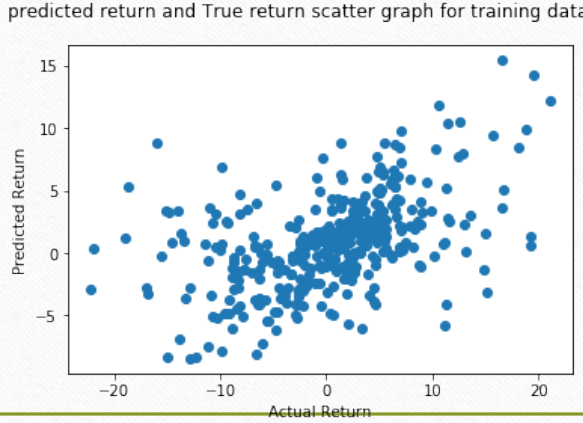
One layer of ConvLSTM2D, then one layer of encoder, then another layer of LSTM, then add two layer of CNN to get the final result. Which aims to reduce difference prediction of 5 days returns and actuals

Result

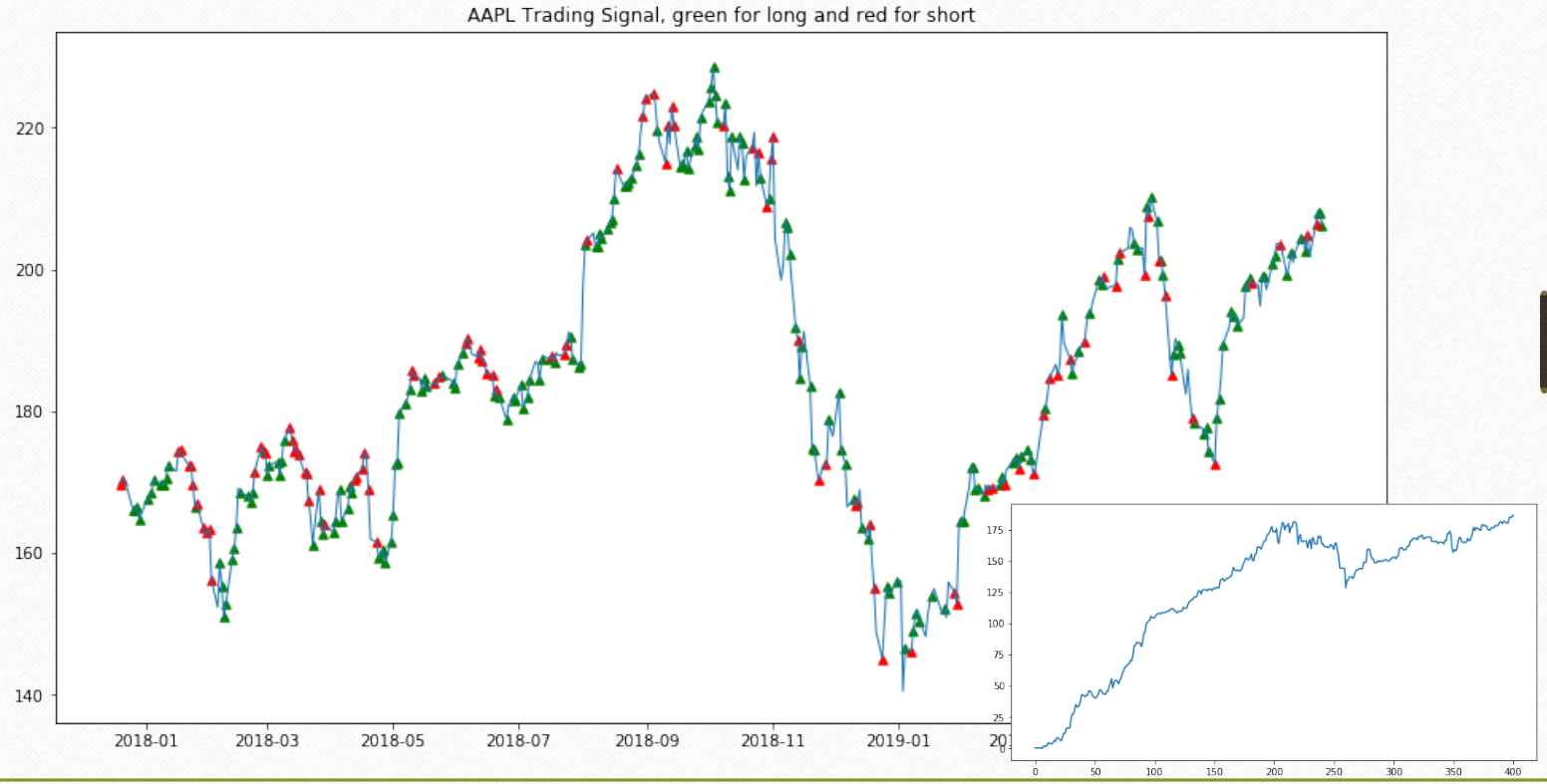

set prediction accuracy for 5 days return
Actual
Predicted
20
10
0
−10
−20
0
200
400
600
800
1000
1200
1400
1600


testing data with ROC = 0.7694842556752798
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0

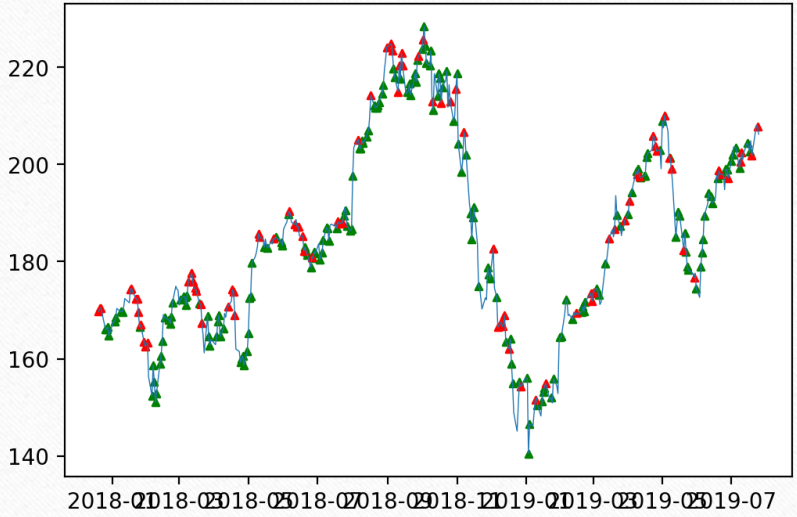

predicted return and True return scatter graph for whole dataset
Predicted Return
Actual Return
15
10
5
0
−5
−10
−15
−20
−10
0
10
20

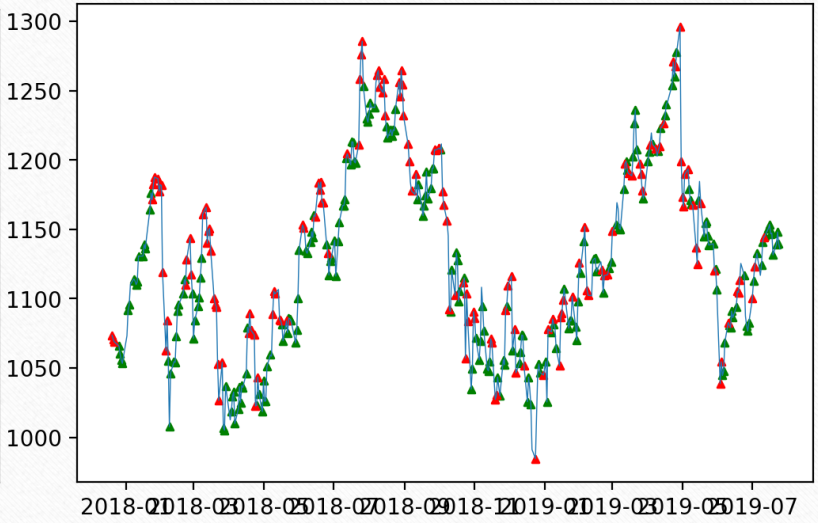

predicted return and True return scatter graph for training dataset
Predicted Return
Actual Return
15
10
5
0
−5
−20
−10
0
10
20

Simulated trading revenue, promising result for AAPL, AMZN and GOOGL



And it also works not only for AAPL, but any other firms. I have tried AMZN and GOOGL, it all shows promising result





One More Thing:

Add-ons for hyper-parameter tuning for Conv_LSTM, and the proof for why shareholder score is negatively correlated to stock value

```
Data access successful
```

[267]:

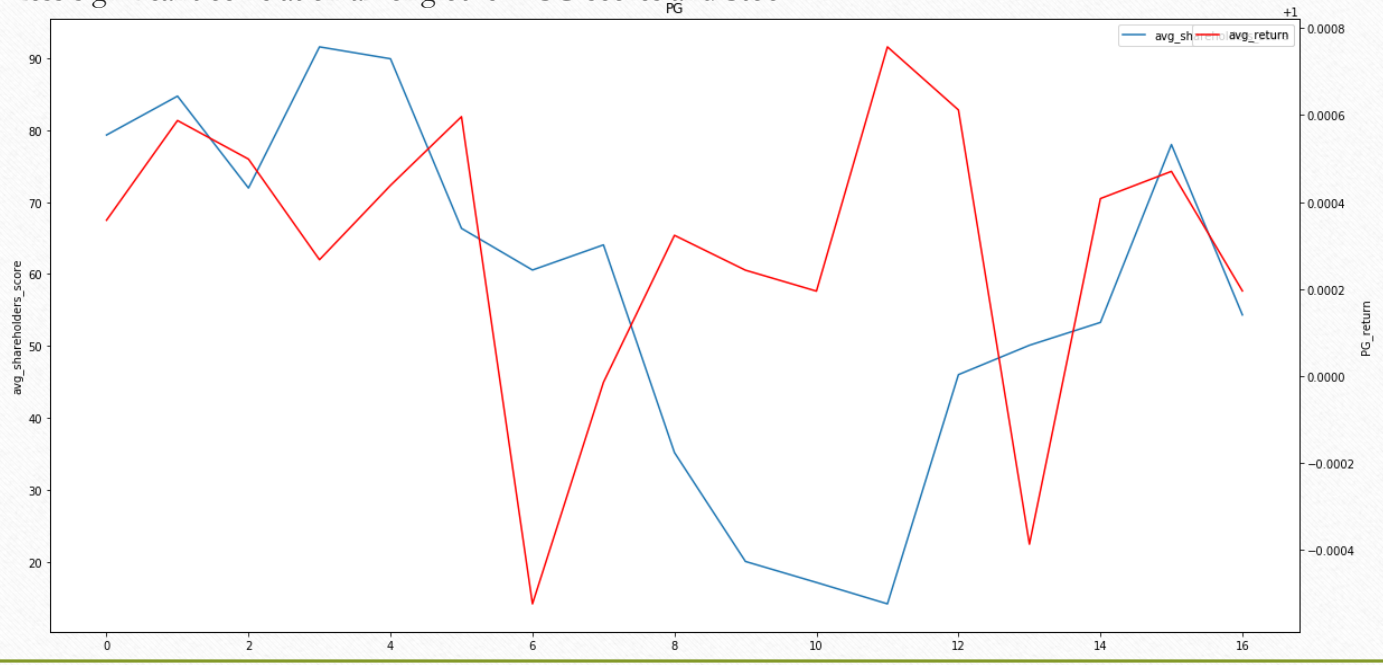
	Instrument	Period End Date	ESG Combined Score	ESG Score	ESG Controversies Score	Environment Pillar Score	Social Pillar Score	Governance Pillar Score	Resource Use Score	Emissions Score	Innovation Score	Workforce Score	Human Rights Score	Co
0	PG.TO	2018-12-31	30.008821	30.008821	63.333333	26.390374	36.081946	26.973780	11.363636	25.606061	42.272727	19.545455	20.303030	6
1	PG.TO	2017-12-31	31.560544	31.560544	59.104478	29.929763	34.610048	29.829040	12.686567	34.179104	42.537313	12.985075	22.985075	7
2	PG.TO	2016-12-31	34.047226	34.047226	61.128049	28.210187	41.789763	31.542284	13.414634	28.506098	42.682927	27.896341	25.609756	6
3	PG.TO	2015-12-31	32.397930	32.397930	57.467532	28.232429	34.374428	34.740925	12.662338	29.545455	42.370130	21.590909	27.597403	4
4	PG.TO	2014-12-31	38.590838	38.590838	63.087248	28.928148	45.228755	41.636261	12.583893	31.208054	42.785235	40.436242	29.362416	5
5	PG.TO	2013-12-31	36.942888	36.942888	62.121212	28.678946	49.238868	31.843437	12.626263	30.639731	42.592593	46.296296	29.629630	5
6	PG.TO	2012-12-31	31.223556	31.223556	60.238908	27.911062	46.332260	17.330630	12.116041	29.180887	42.320819	40.784983	30.546075	5
7	PG.TO	2011-12-31	34.024384	34.024384	59.574468	27.362328	45.337629	28.283064	34.042553	7.801418	42.021277	35.283688	32.092199	6
8	PG.TO	2010-12-31	32.942333	32.942333	59.157509	30.273648	35.892277	32.483719	38.461538	11.904762	42.124542	15.018315	33.333333	5

Project Highlights:

- Statistical modelling on selected US stock, de-trend and find seasonality
- Forecasting price movement based on SARIMA model with hyperparameter tuning on AWS with optimisation in parallel computing
- Exploratory data analysis on social media sentiment scores and engagement scores to identify feature with highest data quality and relevance
- Use Principle Component Analysis (PCA) to reduce the dimensions and complexity of features, making model more robusted
- Supply chain analysis to find one's most related firm and their financials
- Multi-Variate Conv-LSTM on sentiment, supply chain and

Identify correlations between ESG scores & Stock value

- Select a random stock to analyse impact of ESG scores on Stock value
- Result: Stock has some negative correlation with Shareholders Score
- Less significant correlation among other ESG scores and Stock



Identify correlations between ESG scores & Stock value

- Select a Portfolio of stocks to verify impact of ESG scores on Stock value
- Result shows Portfolio has some negative correlation with Shareholders Score