Futurade

LSTM based Pairs Trading strategy for Futures

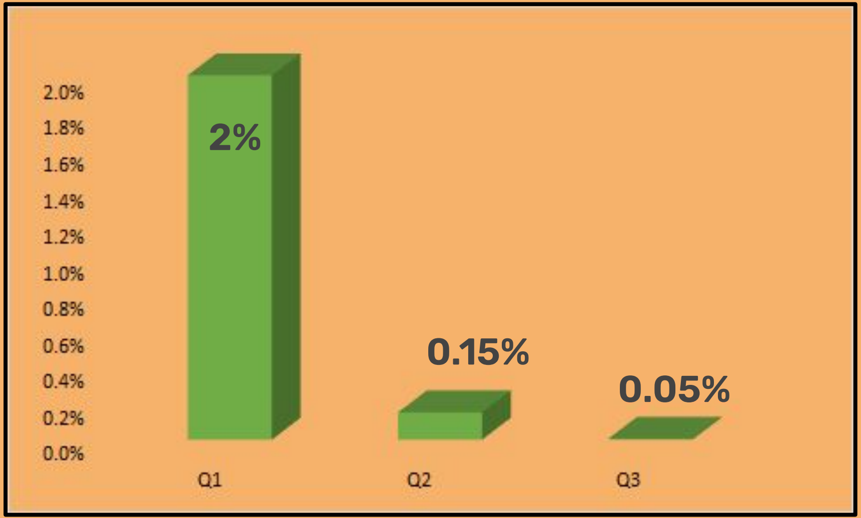
Disclaimer:

Futurade is not a registered investment, legal or tax advisor or a broker/dealer. All investments, financial opinions expressed by Futurade are from personal research and experience of the members of Futurade and are intended as educational material.

Disclaimer:

Futurade is not a registered investment, legal or tax advisor or a broker/dealer. All investments, financial opinions expressed by Futurade are from personal research and experience of the members of Futurade and are intended as educational material.

Typical Indian



Q1.Ever directly traded on Equity markets?

Q2.Ever traded with Futures?

Q3.Do you think you have the domain knowledge to do well here?

We not only help you select the right stocks to be used for Pairs trading but with our model provide you with the right recommendations for Futures Trading *(hence FUTURADES)* and building contracts based on the duration and amount of risk you ready to take…

Want to know more about what we do?? Let’s Go!

Futures

Futures are derivative financial contracts that **obligate the parties to transact an asset at a predetermined future date and price**. Here, the buyer must purchase or the seller must sell the underlying asset at the set price, regardless of the current market price at the expiration date.

Underlying assets include **physical commodities or other financial instruments**. Futures contracts detail the quantity of the underlying asset and are standardized to facilitate trading on a futures exchange. Futures can be used for hedging or trade speculation.

How risky are futures?

- Futures have back-end risks. When buying a futures contract you put relatively little initial money down. The costs and rewards are not established until the contract's expiration date, at which point both parties discover their outcome.
- This means that you have very little control over your risk profile. If an asset's value surges or collapses in value, one can end up owing an enormous (and unforeseeable) amount of money on this contract.
- This makes futures contracts extremely dangerous for the retail investor. Unless you have the assets to cover significant losses a series of bad trades can wipe out not just your portfolio but your personal finances as well.

Pairs Trading

Pairs trading was first introduced in the mid-1980s by a group of technical analyst researchers that were employed by Morgan Stanley. The pairs trade strategy uses statistical and technical analysis to seek out **potential market-neutral profits.**

A pairs trade strategy is based on the **historical correlation** of two securities. The securities in a pairs trade must have a high positive correlation, which is the primary driver behind the strategy’s profits. A pairs trade strategy is best deployed when a trader identifies a correlation discrepancy.

Risks associated with Pairs Trading

- Pairs trading has the potential to achieve profits through simple and relatively low-risk positions. The pairs trade is market-neutral, meaning the direction of the overall market does not affect its win or loss.

- The goal is to match two trading vehicles that are highly correlated, trading one long and the other short when the pair's price ratio diverges "x" number of standard deviations - "x" is optimized using historical data. If the pair reverts to its mean trend, a profit is made on one or both of the positions.

LSTM

Long short-term memory (LSTM) is an **artificial recurrent neural network** (RNN) architecture used in the field of deep learning. It is a network that is capable of learning long term dependencies in data. This is achieved because the recurring module of the model has a combination of four layers interacting with each other.

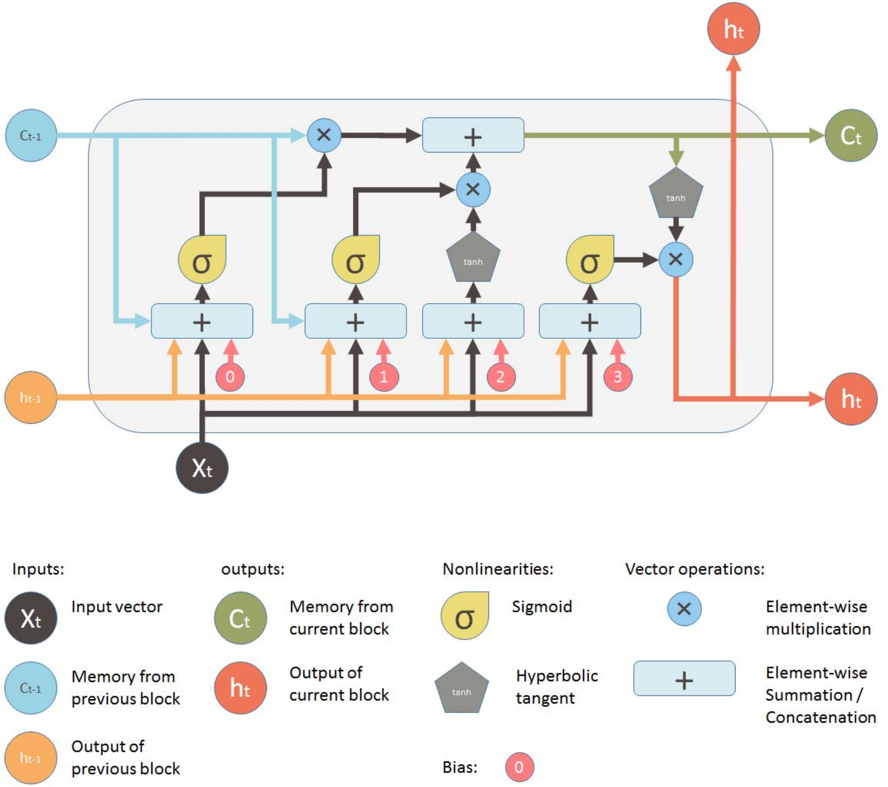
Working of LSTM

- The RNN cannot predict the word stored in the long term. As the gap length increases RNN does not give efficient performance. LSTM can by default retain the information for long period of time.
- LSTM has a chain structure that contains four neural networks and different memory blocks called cells.
- Information is retained by the cells and the memory manipulations are done by the gates. There are three gates :

  Forget Gate: The information that no longer useful in the cell state is removed.

  Input gate: Addition of useful information to the cell state is done by input gate

  Output gate: The task of extracting useful information from the current cell state to be presented as an output is done by output gate



Procedures Pt.1

Downloading data and preprocessing:

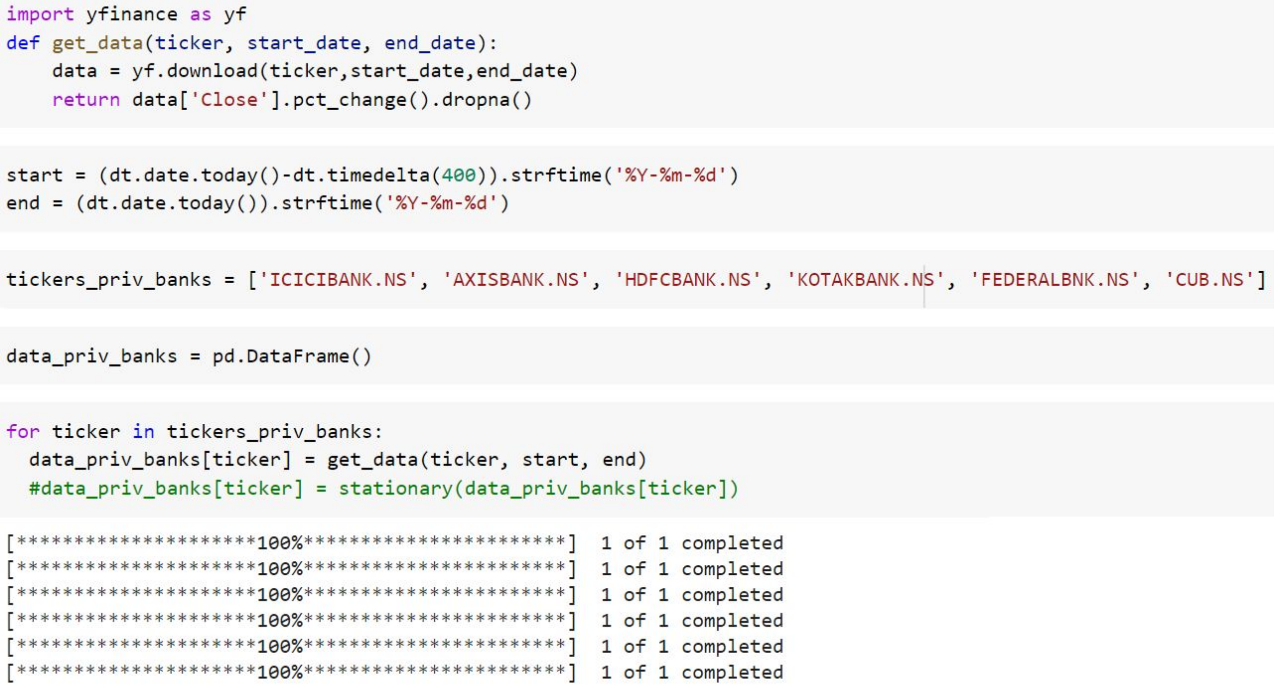
a. **Stock Data - through yfinance** https://pypi.org/project/yfinance/

b. **Futures Data - through jugaad data which provides us the futures data from National Stock Exchange** https://marketsetup.in/documentation/jugaad-data/

c. **Stationary time series - we make the stock data stationary by taking its percentage change and then check whether it is stationary or not by checking the p value of Augmented Dickey Fuller test.**

Note:- we download the data in pandas dataframe format

Importing Datasets and preprocessing



```
import yfinance as yf
def get_data(ticker, start_date, end_date):
    data = yf.download(ticker,start_date,end_date)
    return data['Close'].pct_change().dropna()
```

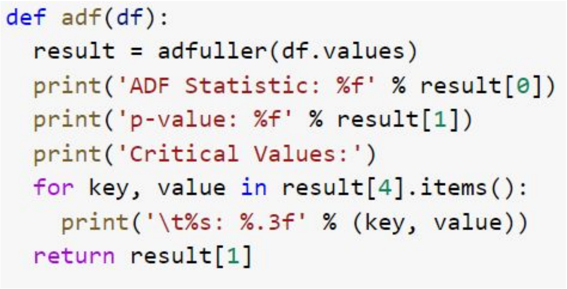
```
start = (dt.date.today()-dt.timedelta(400)).strftime('%Y-%m-%d')
end = (dt.date.today()).strftime('%Y-%m-%d')
```

```
tickers_priv_banks = ['ICICIBANK.NS', 'AXISBANK.NS', 'HDFCBANK.NS', 'KOTAKBANK.NS', 'FEDERALBNK.NS', 'CUB.NS']
```

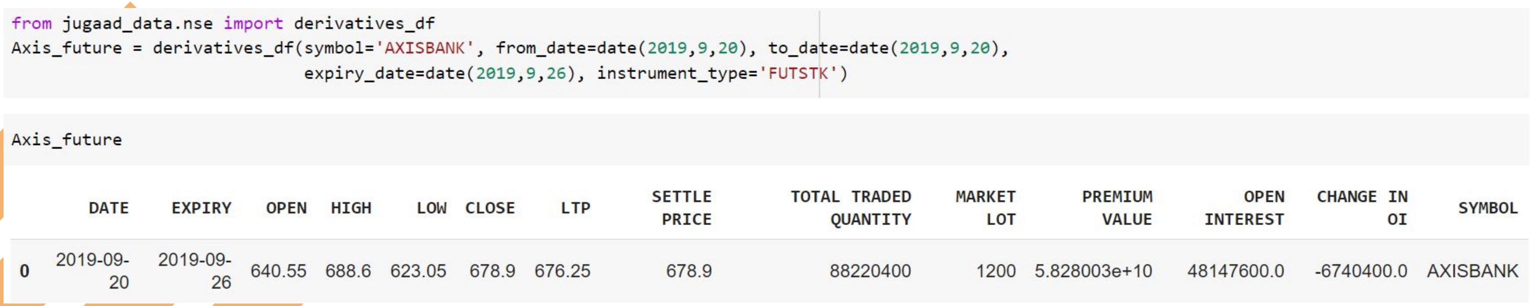
```
data_priv_banks = pd.DataFrame()
```

```
for ticker in tickers_priv_banks:
  data_priv_banks[ticker] = get_data(ticker, start, end)
  #data_priv_banks[ticker] = stationary(data_priv_banks[ticker])
```

```
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
```



```
def adf(df):
  result = adfuller(df.values)
  print('ADF Statistic: %f' % result[0])
  print('p-value: %f' % result[1])
  print('Critical Values:')
  for key, value in result[4].items():
    print('\t%s: %.3f' % (key, value))
  return result[1]
```



```
from jugaad_data.nse import derivatives_df
Axis_future = derivatives_df(symbol='AXISBANK', from_date=date(2019,9,20), to_date=date(2019,9,20),
                             expiry_date=date(2019,9,26), instrument_type='FUTSTK')
```

```
Axis_future
```

	DATE	EXPIRY	OPEN	HIGH	LOW	CLOSE	LTP	SETTLE PRICE	TOTAL TRADED QUANTITY	MARKET LOT	PREMIUM VALUE	OPEN INTEREST	CHANGE IN OI	SYMBOL
0	2019-09-20	2019-09-26	640.55	688.6	623.05	678.9	676.25	678.9	88220400	1200	5.828003e+10	48147600.0	-6740400.0	AXISBANK

Procedures Pt.2

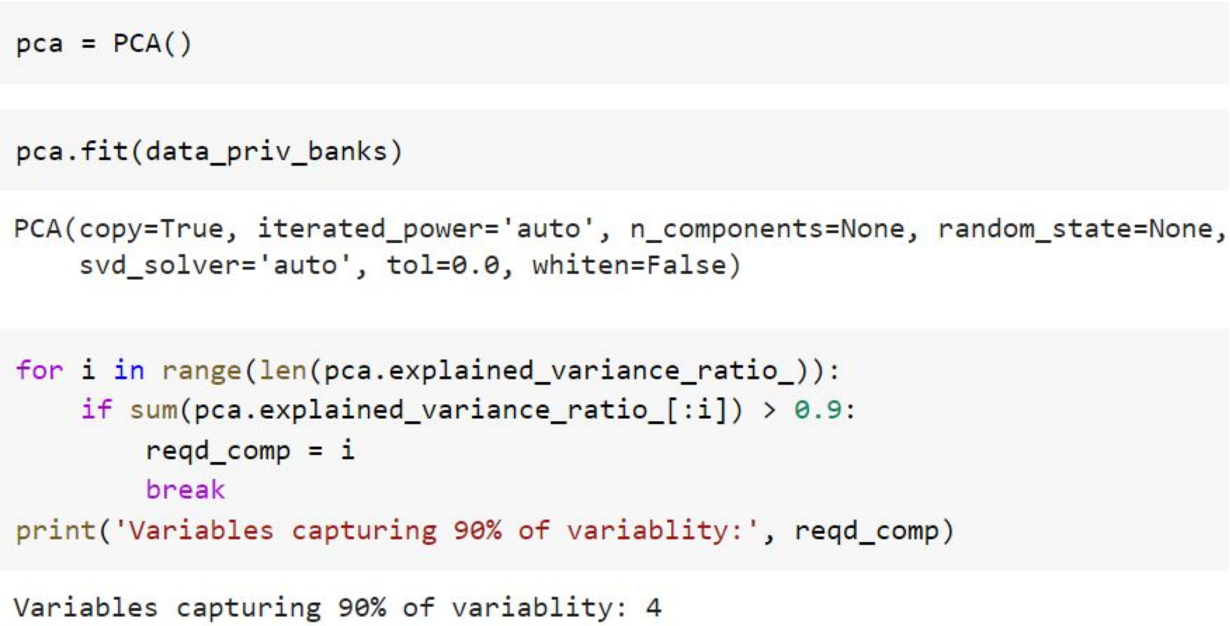
Picking pairs (Usually selected from the same sector):

a. Principal Component Analysis (PCA) - to select the components (here stocks) which capture high variability.

b. Correlation Analysis - to find the positively correlated stocks to be used.

c. Hierarchical Clustering - to find the stocks which are in the same cluster based on their correlation

Principal Component Analysis



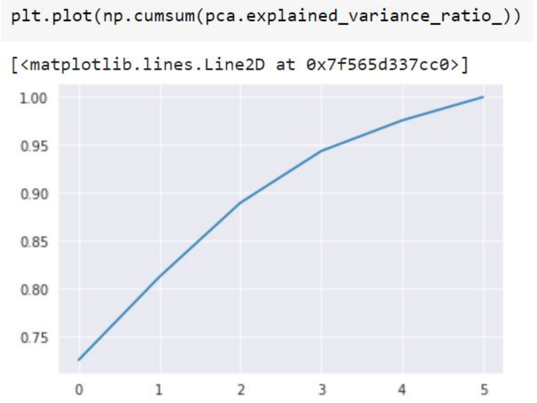
```
pca = PCA()
```

```
pca.fit(data_priv_banks)
```

```
PCA(copy=True, iterated_power='auto', n_components=None, random_state=None,
    svd_solver='auto', tol=0.0, whiten=False)
```

```
for i in range(len(pca.explained_variance_ratio_)):
    if sum(pca.explained_variance_ratio_[:i]) > 0.9:
        reqd_comp = i
        break
print('Variables capturing 90% of variablity:', reqd_comp)
```

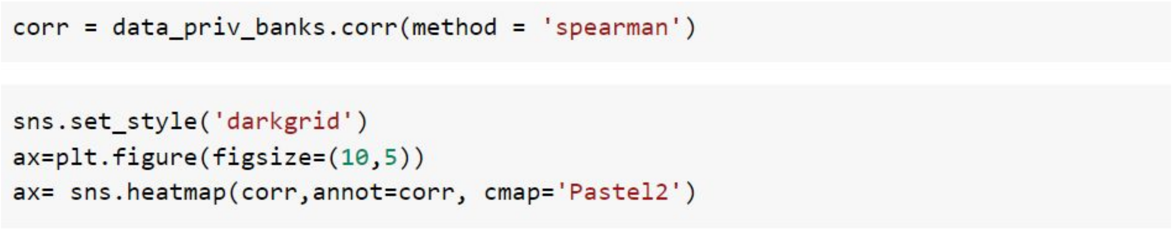
```
Variables capturing 90% of variablity: 4
```



```
plt.plot(np.cumsum(pca.explained_variance_ratio_))
```

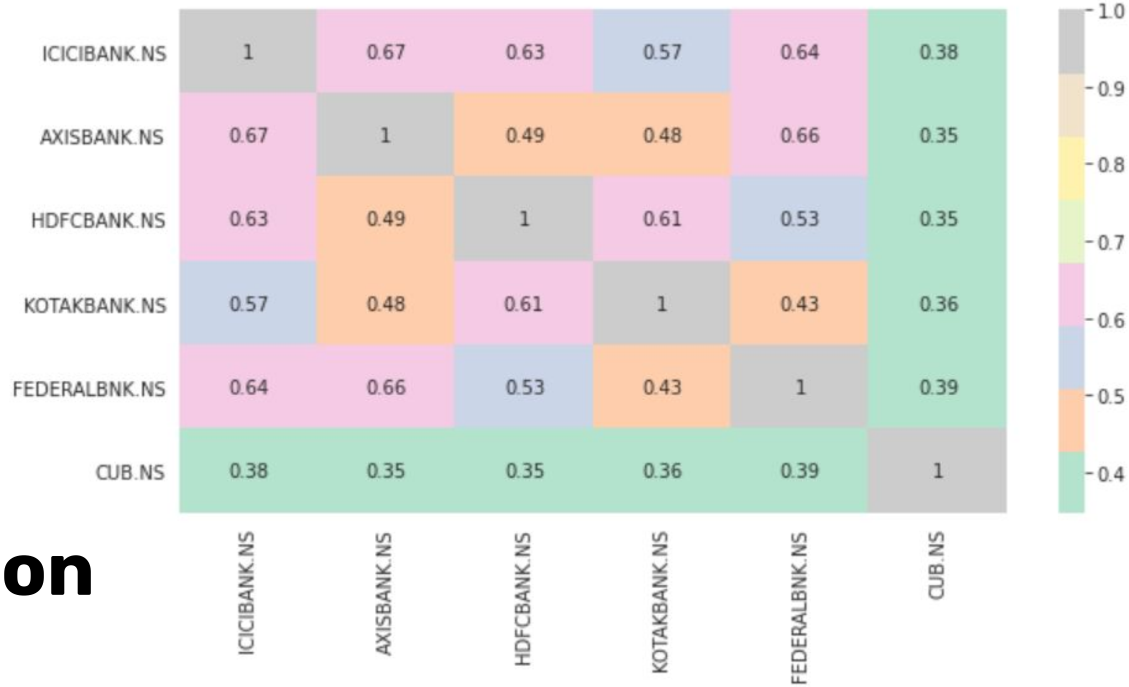
```
[<matplotlib.lines.Line2D at 0x7f565d337cc0>]
```

Correlation Analysis



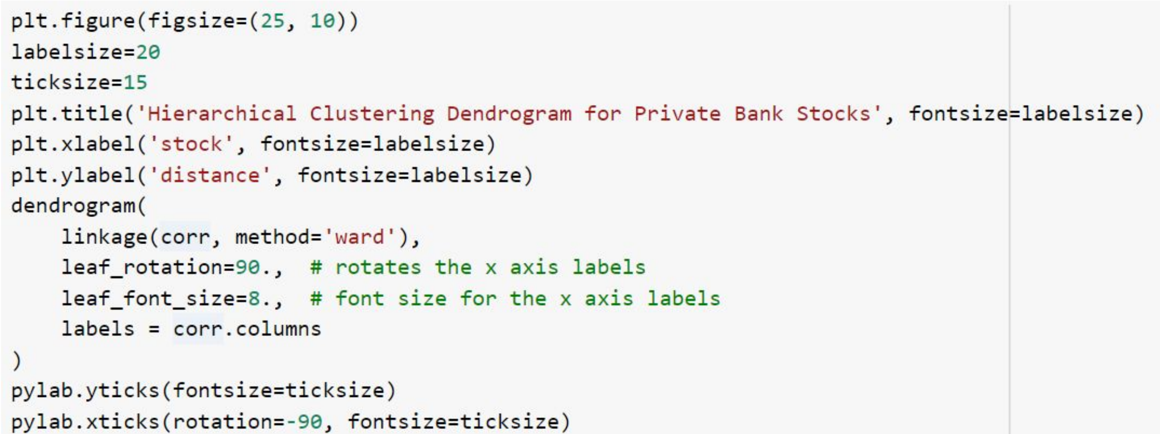
```
corr = data_priv_banks.corr(method = 'spearman')
```

```
sns.set_style('darkgrid')
ax=plt.figure(figsize=(10,5))
ax= sns.heatmap(corr,annot=corr, cmap='Pastel2')
```

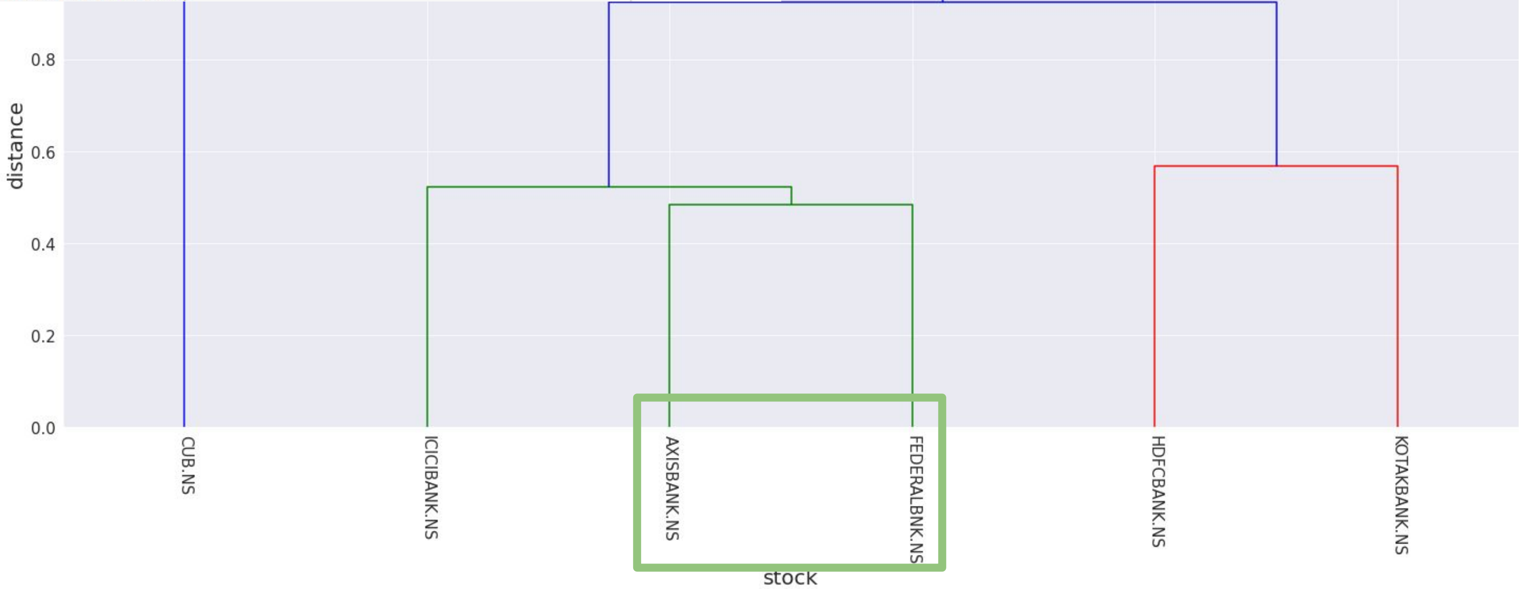


	ICICIBANK.NS	AXISBANK.NS	HDFCBANK.NS	KOTAKBANK.NS	FEDERALBNK.NS	CUB.NS
ICICIBANK.NS	1	0.67	0.63	0.57	0.64	0.38
AXISBANK.NS	0.67	1	0.49	0.48	0.66	0.35
HDFCBANK.NS	0.63	0.49	1	0.61	0.53	0.35
KOTAKBANK.NS	0.57	0.48	0.61	1	0.43	0.36
FEDERALBNK.NS	0.64	0.66	0.53	0.43	1	0.39
CUB.NS	0.38	0.35	0.35	0.36	0.39	1

Hierarchical clustering



```
plt.figure(figsize=(25, 10))
labelsize=20
ticksize=15
plt.title('Hierarchical Clustering Dendrogram for Private Bank Stocks', fontsize=labelsize)
plt.xlabel('stock', fontsize=labelsize)
plt.ylabel('distance', fontsize=labelsize)
dendrogram(
    linkage(corr, method='ward'),
    leaf_rotation=90.,  # rotates the x axis labels
    leaf_font_size=8.,  # font size for the x axis labels
    labels = corr.columns
)
pylab.yticks(fontsize=ticksize)
pylab.xticks(rotation=-90, fontsize=ticksize)
```



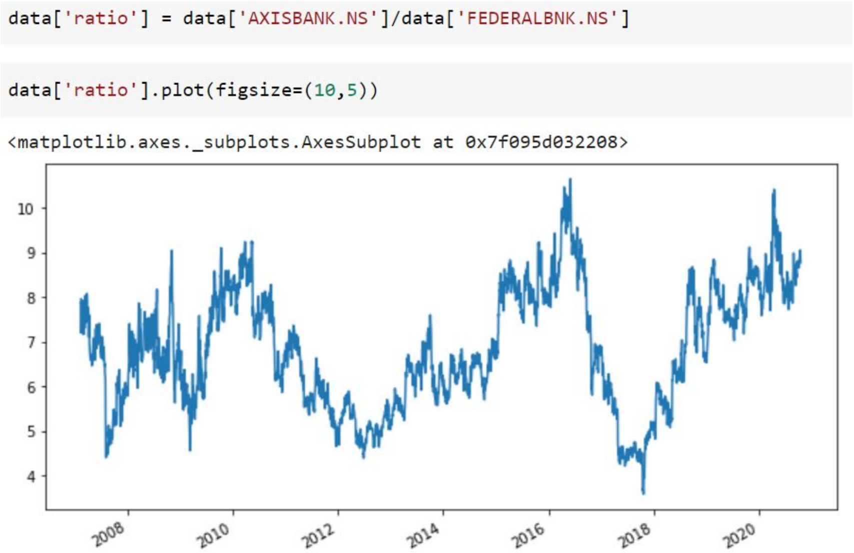
Procedures Pt.3

Forecasting the Spread Ratio:

a. Spread ratio - calculated by dividing the closing price of one stock with another (in our case dividing the closing price of Axis Bank with Federal bank)

b. Forecast the spread with LSTM model - the spread ratio for the next 21 days is calculated by the LSTM model

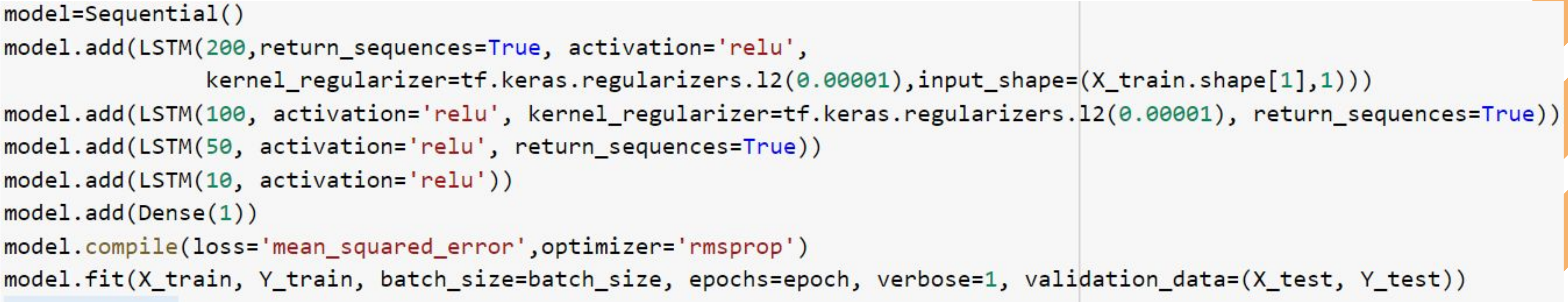
LSTM for Spread Ratio



```
data['ratio'] = data['AXISBANK.NS']/data['FEDERALBNK.NS']
```

```
data['ratio'].plot(figsize=(10,5))
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f095d032208>
```



```
model=Sequential()
model.add(LSTM(200,return_sequences=True, activation='relu',
               kernel_regularizer=tf.keras.regularizers.l2(0.00001),input_shape=(X_train.shape[1],1)))
model.add(LSTM(100, activation='relu', kernel_regularizer=tf.keras.regularizers.l2(0.00001), return_sequences=True))
model.add(LSTM(50, activation='relu', return_sequences=True))
model.add(LSTM(10, activation='relu'))
model.add(Dense(1))
model.compile(loss='mean_squared_error',optimizer='rmsprop')
model.fit(X_train, Y_train, batch_size=batch_size, epochs=epoch, verbose=1, validation_data=(X_test, Y_test))
```

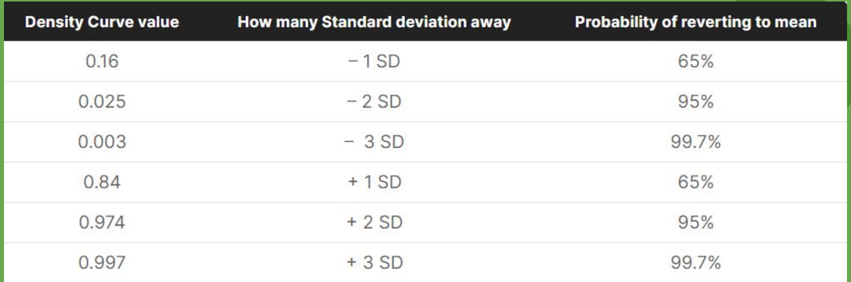
https://colab.research.google.com/drive/16JHExFstFfx0hcKxrJLzo1K9MAVBc6qA#scrollTo=U33rIbZB1Nt-&line=3&uniqifier=1

Procedures Pt.4

Generating Buy/Sell signals through Density Curve:

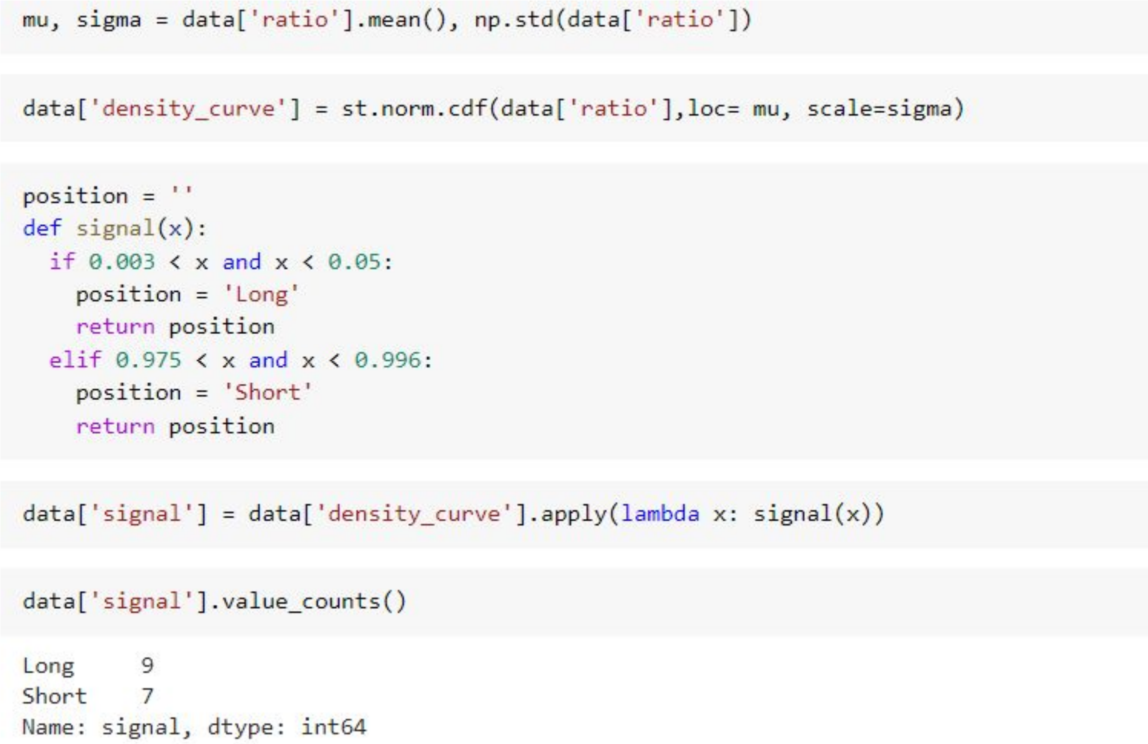
a. Density Curve - density curve of the spread ratio is calculated. The value of the density curve helps us understand the probability of the ratio, falling back to the mean.

b. Buy/Sell signals - a buy signal is generated when the density curve value lies between 0.003 and 0.05. A sell signal is generated when the density curve value lies between 0.975 and 0.996.



Density Curve value	How many Standard deviation away	Probability of reverting to mean
0.16	– 1 SD	65%
0.025	– 2 SD	95%
0.003	– 3 SD	99.7%
0.84	+ 1 SD	65%
0.974	+ 2 SD	95%
0.997	+ 3 SD	99.7%

Density Curve to find Buy/Send Signals



```
mu, sigma = data['ratio'].mean(), np.std(data['ratio'])
```

```
data['density_curve'] = st.norm.cdf(data['ratio'],loc= mu, scale=sigma)
```

```
position = ''
def signal(x):
  if 0.003 < x and x < 0.05:
    position = 'Long'
    return position
  elif 0.975 < x and x < 0.996:
    position = 'Short'
    return position
```

```
data['signal'] = data['density_curve'].apply(lambda x: signal(x))
```

```
data['signal'].value_counts()
```

```
Long     9
Short    7
Name: signal, dtype: int64
```

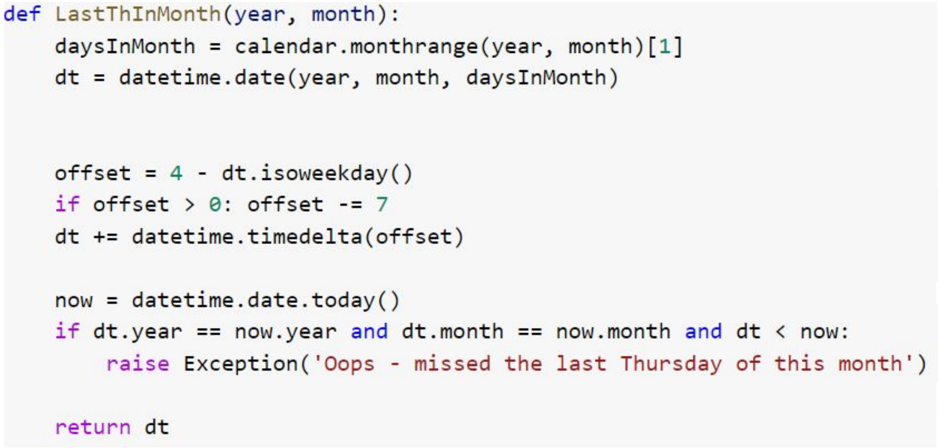
https://colab.research.google.com/drive/16JHExFstFfx0hcKxrJLzo1K9MAVBc6qA#scrollTo=FXmyUBZDTbA2&line=1&uniqifier=1

Procedures Pt.5

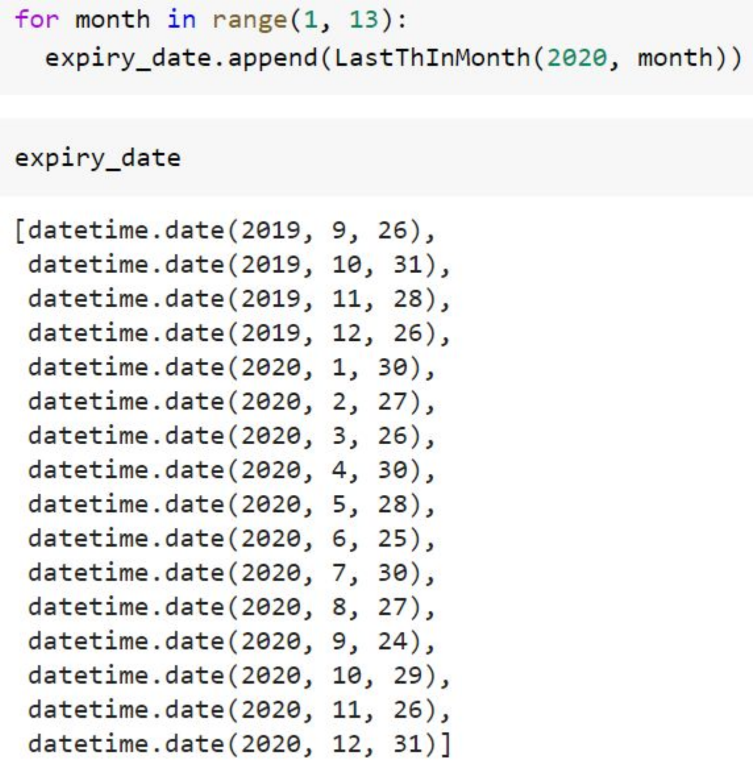
Generating Future contracts:

- Expiry Dates - futures contracts monthly expiry date is the last thursday of the month.
- Relative to the date of buy/sell signal, a function was made to get the near month, next month and far month futures contracts expiry dates.
- Exit Strategy - Using the forecasted Spread ratios, we calculated the Z-scores. When the absolute values of Z-scores are less than 0.5, we exit the trade.

Expiry-Dates



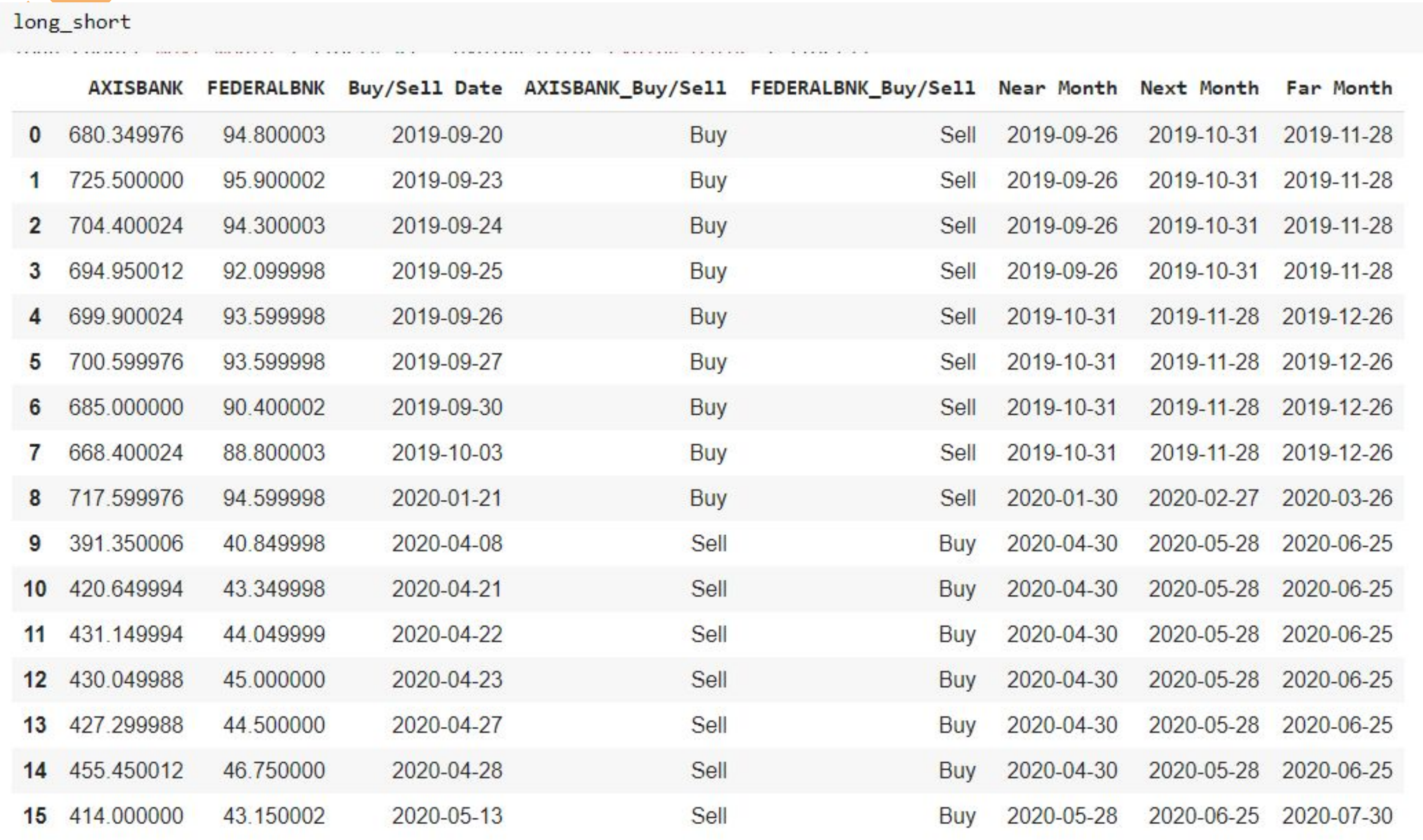
```
def LastThInMonth(year, month):
    daysInMonth = calendar.monthrange(year, month)[1]
    dt = datetime.date(year, month, daysInMonth)


    offset = 4 - dt.isoweekday()
    if offset > 0: offset -= 7
    dt += datetime.timedelta(offset)

    now = datetime.date.today()
    if dt.year == now.year and dt.month == now.month and dt < now:
        raise Exception('Oops - missed the last Thursday of this month')

    return dt
```



```
for month in range(1, 13):
  expiry_date.append(LastThInMonth(2020, month))
```

```
expiry_date
```

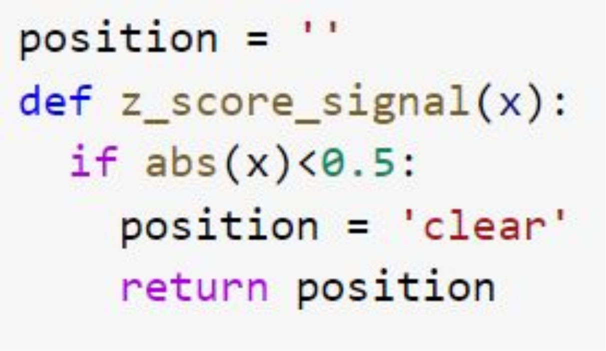
```
[datetime.date(2019, 9, 26),
 datetime.date(2019, 10, 31),
 datetime.date(2019, 11, 28),
 datetime.date(2019, 12, 26),
 datetime.date(2020, 1, 30),
 datetime.date(2020, 2, 27),
 datetime.date(2020, 3, 26),
 datetime.date(2020, 4, 30),
 datetime.date(2020, 5, 28),
 datetime.date(2020, 6, 25),
 datetime.date(2020, 7, 30),
 datetime.date(2020, 8, 27),
 datetime.date(2020, 9, 24),
 datetime.date(2020, 10, 29),
 datetime.date(2020, 11, 26),
 datetime.date(2020, 12, 31)]
```



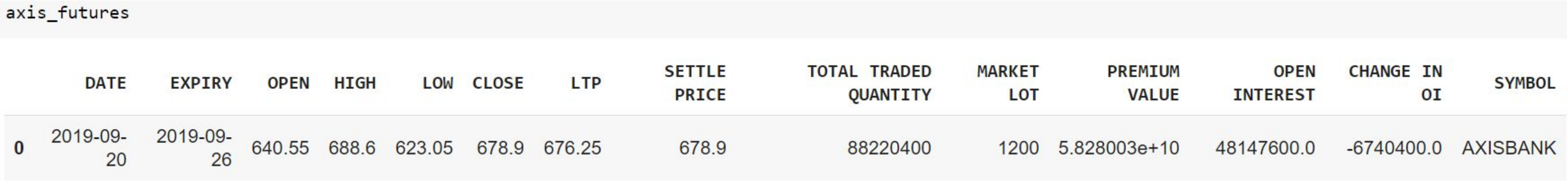
```
long_short
```

	AXISBANK	FEDERALBNK	Buy/Sell Date	AXISBANK_Buy/Sell	FEDERALBNK_Buy/Sell	Near Month	Next Month	Far Month
0	680.349976	94.800003	2019-09-20	Buy	Sell	2019-09-26	2019-10-31	2019-11-28
1	725.500000	95.900002	2019-09-23	Buy	Sell	2019-09-26	2019-10-31	2019-11-28
2	704.400024	94.300003	2019-09-24	Buy	Sell	2019-09-26	2019-10-31	2019-11-28
3	694.950012	92.099998	2019-09-25	Buy	Sell	2019-09-26	2019-10-31	2019-11-28
4	699.900024	93.599998	2019-09-26	Buy	Sell	2019-10-31	2019-11-28	2019-12-26
5	700.599976	93.599998	2019-09-27	Buy	Sell	2019-10-31	2019-11-28	2019-12-26
6	685.000000	90.400002	2019-09-30	Buy	Sell	2019-10-31	2019-11-28	2019-12-26
7	668.400024	88.800003	2019-10-03	Buy	Sell	2019-10-31	2019-11-28	2019-12-26
8	717.599976	94.599998	2020-01-21	Buy	Sell	2020-01-30	2020-02-27	2020-03-26
9	391.350006	40.849998	2020-04-08	Sell	Buy	2020-04-30	2020-05-28	2020-06-25
10	420.649994	43.349998	2020-04-21	Sell	Buy	2020-04-30	2020-05-28	2020-06-25
11	431.149994	44.049999	2020-04-22	Sell	Buy	2020-04-30	2020-05-28	2020-06-25
12	430.049988	45.000000	2020-04-23	Sell	Buy	2020-04-30	2020-05-28	2020-06-25
13	427.299988	44.500000	2020-04-27	Sell	Buy	2020-04-30	2020-05-28	2020-06-25
14	455.450012	46.750000	2020-04-28	Sell	Buy	2020-04-30	2020-05-28	2020-06-25
15	414.000000	43.150002	2020-05-13	Sell	Buy	2020-05-28	2020-06-25	2020-07-30

Exit Strategy and selecting the futures contract

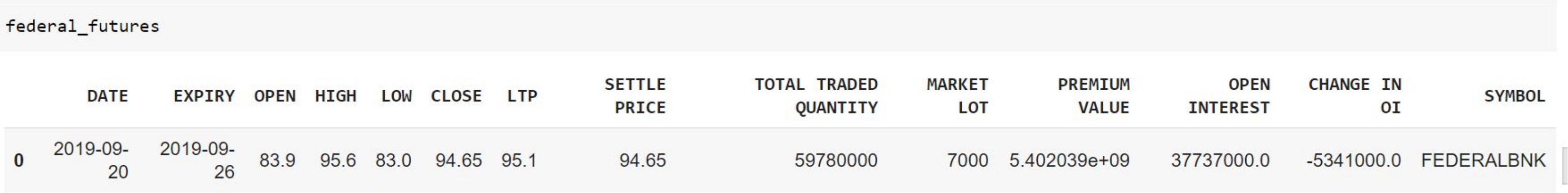


```
position = ''
def z_score_signal(x):
  if abs(x)<0.5:
    position = 'clear'
    return position
```



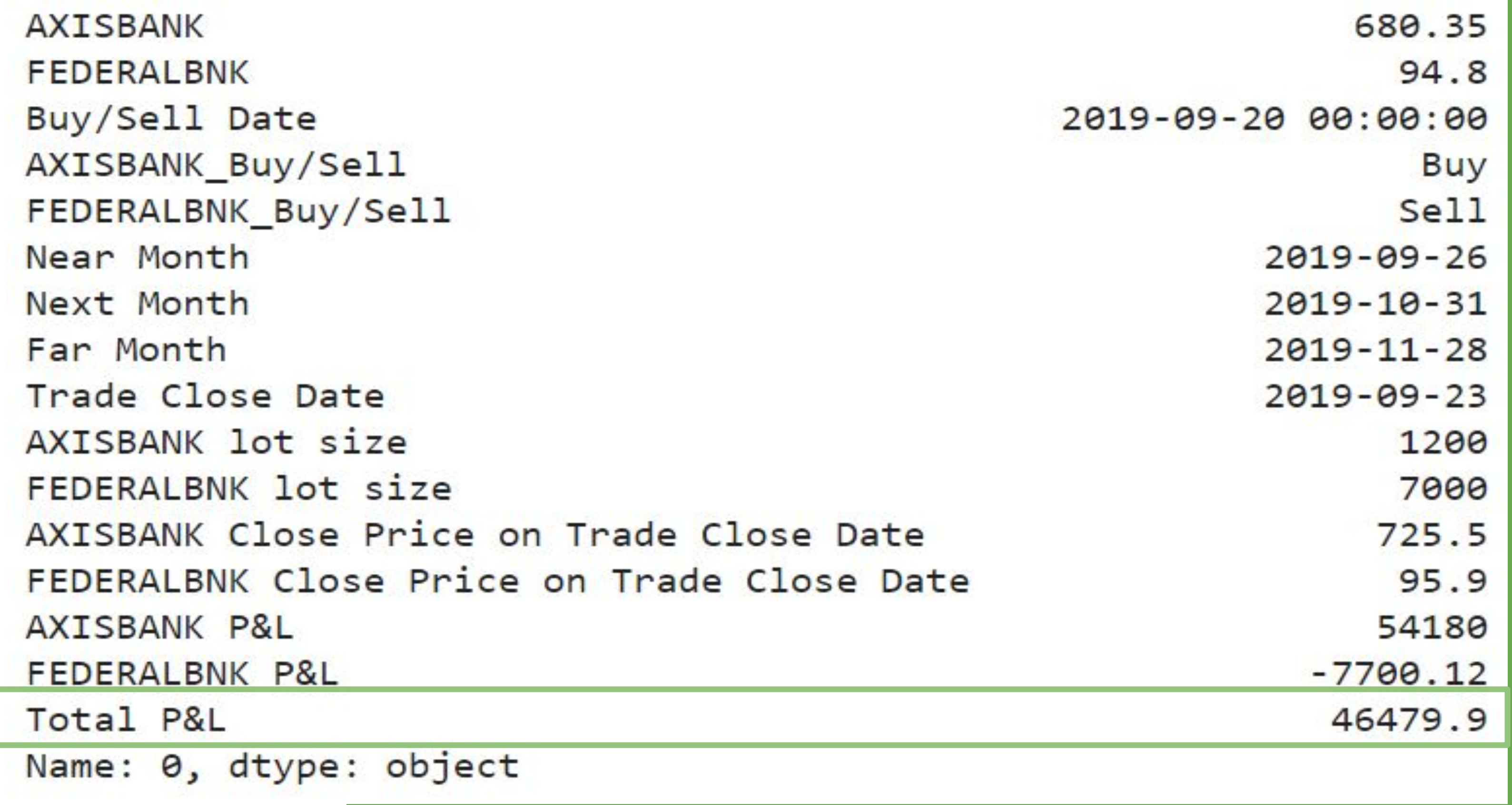
```
axis_futures
```

	DATE	EXPIRY	OPEN	HIGH	LOW	CLOSE	LTP	SETTLE PRICE	TOTAL TRADED QUANTITY	MARKET LOT	PREMIUM VALUE	OPEN INTEREST	CHANGE IN OI	SYMBOL
0	2019-09-20	2019-09-26	640.55	688.6	623.05	678.9	676.25	678.9	88220400	1200	5.828003e+10	48147600.0	-6740400.0	AXISBANK



```
federal_futures
```

	DATE	EXPIRY	OPEN	HIGH	LOW	CLOSE	LTP	SETTLE PRICE	TOTAL TRADED QUANTITY	MARKET LOT	PREMIUM VALUE	OPEN INTEREST	CHANGE IN OI	SYMBOL
0	2019-09-20	2019-09-26	83.9	95.6	83.0	94.65	95.1	94.65	59780000	7000	5.402039e+09	37737000.0	-5341000.0	FEDERALBNK



```
AXISBANK                                              680.35
FEDERALBNK                                              94.8
Buy/Sell Date                            2019-09-20 00:00:00
AXISBANK_Buy/Sell                                        Buy
FEDERALBNK_Buy/Sell                                     Sell
Near Month                                        2019-09-26
Next Month                                        2019-10-31
Far Month                                         2019-11-28
Trade Close Date                                  2019-09-23
AXISBANK lot size                                       1200
FEDERALBNK lot size                                     7000
AXISBANK Close Price on Trade Close Date               725.5
FEDERALBNK Close Price on Trade Close Date              95.9
AXISBANK P&L                                           54180
FEDERALBNK P&L                                      -7700.12
Total P&L                                            46479.9
Name: 0, dtype: object
```

Scope of Improvement

- Selecting entry and exit strategy for the trades
- Improving the LSTM model
- Expanding to more sectors and having multiple trades in multiple sectors
- Expanding into high frequency trading
- Building a reinforcement learning based trading bot to trade the pairs
- Moving into a server and building an app/api for the same

Thank You!