**Report on Stock Analysis of ICICI Bank**

**Introduction**

The stock analysis of ICICI Bank form 1-1-2017 to 24-07-2019 with the help of Python codes.

Before we play with stock information, we have to get it in some serviceable arrangement. Stock information can be acquired from Yahoo! Account, Google Finance, or various different sources, and the pandas bundle gives simple access to Yahoo! Money and Google Finance information, alongside different sources. In this address, we will get our information from Yahoo! Money.

The accompanying code shows how to make straightforwardly a DataFrame article containing stock data

Here are the codes, output and their analysis.

1. Importing the important libraries required for the analysis

* import pandas\_datareader as web
* import numpy as np
* import pandas as pd
* import datetime

1. Creating the start and end for Analysis Purpose.

* start=datetime.datetime(2017,1,1)
* end=datetime.datetime(2019,7,24)

1. Now, the stock of ICICI Bank should be loaded in icicibank list from yahoo. The codes are

icicibank = web. DataReader("icicibank.ns", 'yahoo', start, end)

icicibank

| **High** | **Low** | **Open** | **Close** | **Volume** | **Adj Close** |
| --- | --- | --- | --- | --- | --- |
| **Date** |  |  |  |  |  |  |
| **2017-01-02** | 233.000000 | 225.544998 | 233.000000 | 229.044998 | 10404350.0 | 220.773010 |
| **2017-01-03** | 232.182007 | 226.636002 | 229.182007 | 231.272995 | 10236707.0 | 222.920547 |
| **2017-01-04** | 232.500000 | 227.636002 | 231.136002 | 228.272995 | 6901181.0 | 220.028885 |
| **2017-01-05** | 235.000000 | 229.272995 | 230.182007 | 233.863998 | 13830669.0 | 225.417984 |
| **2017-01-06** | 237.636002 | 233.817993 | 234.772995 | 234.682007 | 9055144.0 | 226.206451 |
| **2017-01-09** | 236.227005 | 232.727005 | 234.455002 | 235.182007 | 7875295.0 | 226.688370 |
| **2017-01-10** | 238.091003 | 234.817993 | 236.363998 | 236.544998 | 14784041.0 | 228.002151 |
| **2017-01-11** | 243.727005 | 237.817993 | 238.636002 | 240.636002 | 19565679.0 | 231.945404 |
| **2017-01-12** | 245.000000 | 241.317993 | 242.363998 | 243.955002 | 11711280.0 | 235.144516 |
| **2017-01-13** | 245.455002 | 240.727005 | 245.455002 | 243.682007 | 7810375.0 | 234.881409 |
| **2017-01-16** | 246.227005 | 240.727005 | 243.091003 | 244.817993 | 8863677.0 | 235.976379 |
| **2017-01-17** | 246.817993 | 241.363998 | 245.000000 | 244.000000 | 6968061.0 | 235.187927 |
| **2017-01-18** | 247.363998 | 242.500000 | 243.091003 | 244.044998 | 10085606.0 | 235.231293 |
| **2017-01-19** | 246.091003 | 242.000000 | 244.363998 | 245.182007 | 8605687.0 | 236.327240 |
| **2017-01-20** | 243.591003 | 238.682007 | 242.727005 | 239.500000 | 12207060.0 | 230.850433 |
| **2017-01-23** | 238.000000 | 231.955002 | 238.000000 | 234.136002 | 43793368.0 | 225.680161 |

**icicibank.head()**

| **High** | **Low** | **Open** | **Close** | **Volume** | **Adj Close** |
| --- | --- | --- | --- | --- | --- |
| **Date** |  |  |  |  |  |  |
| **2017-01-02** | 233.000000 | 225.544998 | 233.000000 | 229.044998 | 10404350.0 | 220.773010 |
| **2017-01-03** | 232.182007 | 226.636002 | 229.182007 | 231.272995 | 10236707.0 | 222.920547 |
| **2017-01-04** | 232.500000 | 227.636002 | 231.136002 | 228.272995 | 6901181.0 | 220.028885 |
| **2017-01-05** | 235.000000 | 229.272995 | 230.182007 | 233.863998 | 13830669.0 | 225.417984 |
| **2017-01-06** | 237.636002 | 233.817993 | 234.772995 | 234.682007 | 9055144.0 | 226.206451 |

**icicibank.tail()**

| **High** | **Low** | **Open** | **Close** | **Volume** | **Adj Close** |
| --- | --- | --- | --- | --- | --- |
| **Date** |  |  |  |  |  |  |
| **2019-07-18** | 425.500000 | 417.500000 | 425.000000 | 418.649994 | 10651654.0 | 416.614746 |
| **2019-07-19** | 420.350006 | 408.500000 | 419.899994 | 410.299988 | 12260757.0 | 408.305328 |
| **2019-07-22** | 412.950012 | 404.250000 | 409.750000 | 411.500000 | 17937246.0 | 410.500000 |
| **2019-07-23** | 416.000000 | 409.750000 | 412.799988 | 411.950012 | 14207080.0 | 411.950012 |
| **2019-07-24** | 411.850006 | 401.700012 | 411.850006 | 408.500000 | 16505930.0 | 408.500000 |

How about we quickly talk about this. Open is the cost of the stock toward the start of the exchanging day (it need not be the end cost of the past exchanging day), high is the most astounding cost of the stock on that exchanging day, low the least cost of the stock on that exchanging day, and close the cost of the stock at shutting time. The volume demonstrates what number of stocks were exchanged. Balanced close is the end cost of the stock that alters the cost of the stock for corporate activities. While stock costs are viewed asset generally by brokers, stock parts (when the organization makes each surviving stock worth two and parts the cost) and profits (payout of organization benefits per share) additionally influence the cost of stock and ought to be represented.

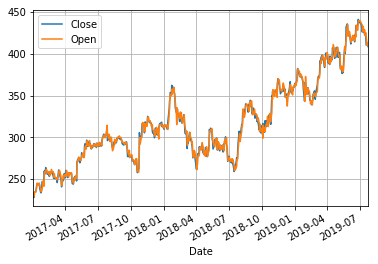
* Open Close plot of icicibank

icicibank['Close'].plot(grid=True)

icicibank['Open'].plot(grid=True)

plt.legend()

plt.show()

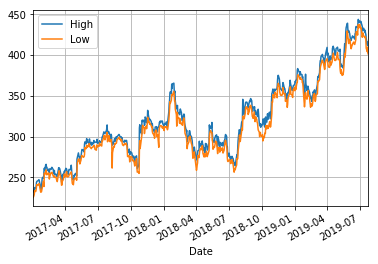


* icicibank['High'].plot(grid=True)

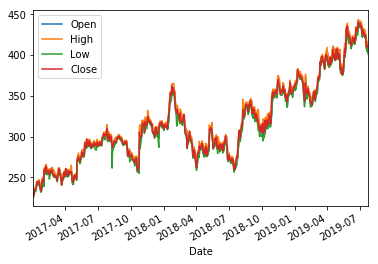
icicibank['Low'].plot(grid=True)

plt.legend()

plt.show()



icicibank[['Open','High','Low','Close']].plot()



In the above graph, if you see in the graph closely then you find that in this month of 2019 the Stock Price of icicibank was very good as it touches Rs. 450. If you see from the beginning of 2019 the Stock rate was going down continuously in fact the Rate was growing . So it’s better to invest in this Stock .You can go with long term investment as now by seeing the prices and in future the stock goes very up and thus help the investor to enjoy good profit.

* Then after Resampling the data into monthly format then after taking there mean to give the monthly mean of the stock price of icici Bank.

# Resample to monthly level

* monthly\_df = icicibank.resample('M').mean()

# Print `monthly\_df`

* print(monthly\_df)
* High Low Open Close Volume \

Date

2017-01-31 241.322478 236.034571 238.762001 238.898286 1.891917e+07

2017-02-28 260.406685 253.533422 257.361212 257.464106 1.994648e+07

2017-03-31 254.468998 249.778910 252.623953 251.518544 1.685488e+07

2017-04-30 255.909166 250.520337 253.118776 253.212055 1.637592e+07

2017-05-31 279.733406 272.812002 275.838724 276.506180 3.121764e+07

2017-06-30 293.280285 287.764997 291.082476 290.338334 1.509382e+07

2017-07-31 300.933334 295.490477 298.519047 298.154762 1.302529e+07

2017-08-31 298.866666 292.271425 296.504762 295.900001 1.095636e+07

2017-09-30 291.690476 286.740476 290.138098 288.750001 1.263553e+07

2017-10-31 281.650000 273.122501 277.055002 277.562497 2.420190e+07

2017-11-30 319.811364 312.436360 315.349999 315.650001 1.540265e+07

2017-12-31 312.547501 305.780000 309.509999 309.550000 1.117851e+07

2018-01-31 336.034094 328.015908 331.390909 333.006818 1.658808e+07

2018-02-28 330.892104 321.776314 326.652630 325.763160 1.534470e+07

2018-03-31 297.734210 289.986842 295.221053 293.092107 1.946142e+07

2018-04-30 285.942858 277.499999 281.057142 282.073807 2.405093e+07

2018-05-31 299.411363 291.299998 294.995456 294.904542 1.988232e+07

2018-06-30 291.576191 284.192857 289.092856 287.440475 2.090191e+07

2018-07-31 277.929547 270.820454 273.888634 274.952272 2.246404e+07

2018-08-31 332.238097 322.992856 325.792857 328.290476 2.308892e+07

2018-09-30 327.672223 317.372221 324.505556 321.916667 1.820762e+07

2018-10-31 326.142857 312.142856 318.169048 320.545240 2.874827e+07

2018-11-30 362.180000 354.129999 358.455000 357.875000 2.085817e+07

2018-12-31 357.720001 350.714998 353.742496 354.697501 1.604661e+07

2019-01-31 372.280433 363.386955 368.017392 367.993479 2.032059e+07

2019-02-28 354.102634 346.189472 350.221053 349.757891 1.462270e+07

2019-03-31 387.244444 379.433331 382.216666 384.583332 1.886126e+07

2019-04-30 401.207894 392.521052 397.494738 396.644737 1.322430e+07

2019-05-31 408.304547 396.761361 401.902274 401.754544 2.307937e+07

2019-06-30 428.068420 419.871054 423.810529 424.373685 1.410430e+07

2019-07-31 429.983332 422.377780 427.480554 425.313889 1.206981e+07

* Now assigning the “Adj Close” to “daily\_close” and then after finding the daily returns.And fill the NaN value with 0 with fillna command

daily\_close = icicibank[['Adj Close']]

daily\_pct\_change = daily\_close.pct\_change()

daily\_pct\_change.fillna(0, inplace=True)

print(daily\_pct\_change)

* Now, add a column in the diff which gave the difference of Opening and Closing price of Stock and then after viewing the data to ensure that the column was added or not.
* icicibank['diff'] = icicibank.Open - icicibank.Close
* icicibank['diff']

Date

2017-01-02 3.955002

2017-01-03 -2.090988

2017-01-04 2.863007

2017-01-05 -3.681992

2017-01-06 0.090988

2017-01-09 -0.727005

2017-01-10 -0.181000

2017-01-11 -2.000000

2017-01-12 -1.591003

2017-01-13 1.772995

2017-01-16 -1.726990

2017-01-17 1.000000

**monthly = icicibank.resample('BM').apply(lambda x: x[-1])**

**monthly.pct\_change()**

| **High** | **Low** | **Open** | **Close** | **Volume** | **Adj Close** |
| --- | --- | --- | --- | --- | --- |
| **Date** |  |  |  |  |  |  |
| **2017-01-31** | NaN | NaN | NaN | NaN | NaN | NaN |
| **2017-02-28** | 0.022261 | 0.034715 | 0.037759 | 0.027513 | -0.114697 | 0.027513 |
| **2017-03-31** | 0.009999 | 0.000000 | 0.005020 | 0.001811 | 0.085923 | 0.001811 |
| **2017-04-28** | -0.012373 | -0.010154 | -0.024437 | 0.005960 | -0.525502 | 0.005960 |
| **2017-05-31** | 0.171795 | 0.167644 | 0.180987 | 0.172173 | 12.710967 | 0.172173 |
| **2017-06-30** | -0.018936 | -0.007532 | -0.005571 | -0.022317 | -0.903179 | -0.006216 |
| **2017-07-31** | 0.049486 | 0.030435 | 0.023459 | 0.042909 | 0.731056 | 0.042909 |
| **2017-08-31** | -0.018111 | 0.002532 | 0.006860 | -0.015036 | -0.309450 | -0.015036 |
| **2017-09-29** | -0.063975 | -0.071549 | -0.077933 | -0.071968 | -0.328579 | -0.071968 |
| **2017-10-31** | 0.076691 | 0.081777 | 0.082177 | 0.084960 | 0.908746 | 0.084960 |
| **2017-11-30** | 0.041385 | 0.026651 | 0.048959 | 0.024825 | 0.251004 | 0.024825 |

* quarter = icicibank.resample("4M").mean()

quarter.pct\_change()

| **High** | **Low** | **Open** | **Close** | **Volume** | **Adj Close** | **diff** |
| --- | --- | --- | --- | --- | --- | --- |
| **Date** |  |  |  |  |  |  |  |
| **2017-01-31** | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| **2017-05-31** | 0.090009 | 0.089164 | 0.089580 | 0.088649 | 0.129787 | 0.088649 | -1.543616 |
| **2017-09-30** | 0.126021 | 0.130257 | 0.130350 | 0.127691 | -0.395183 | 0.144325 | 9.465107 |
| **2018-01-31** | 0.057570 | 0.051639 | 0.050948 | 0.055881 | 0.299718 | 0.057669 | -1.815329 |
| **2018-05-31** | -0.032991 | -0.036026 | -0.032948 | -0.036437 | 0.178399 | -0.036437 | -1.742137 |
| **2018-09-30** | 0.011031 | 0.011074 | 0.011127 | 0.012494 | 0.075238 | 0.020597 | -0.858329 |
| **2019-01-31** | 0.158767 | 0.159590 | 0.157829 | 0.160324 | 0.011659 | 0.169183 | -11.182883 |
| **2019-05-31** | 0.094790 | 0.098612 | 0.096653 | 0.095063 | -0.180727 | 0.095063 | -0.727221 |
| **2019-09-30** | 0.104211 | 0.109854 | 0.109199 | 0.106673 | -0.256773 | 0.107026 | -5.143075 |

# Isolate the `Adj Close` values and transform the DataFrame

daily\_close\_px = all\_data[['Adj Close']].reset\_index().pivot('Date', 'Ticker', 'Adj Close')

# Calculate the daily percentage change for `daily\_close\_px`

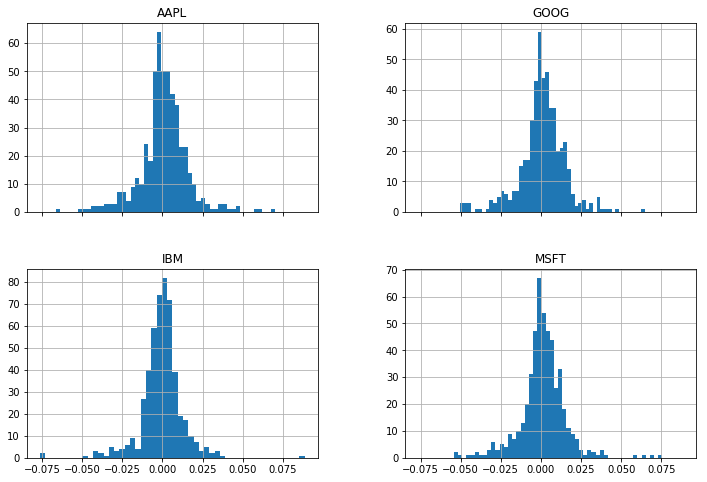
daily\_pct\_change = daily\_close\_px.pct\_change()

# Plot the distributions

daily\_pct\_change.hist(bins=50, sharex=True, figsize=(12,8))

# Show the resulting plot

plt.show()



* stock\_change\_apr = stock\_change \* 252 \* 100 # There are 252 trading days in a year; the 100 converts to percentages

stock\_change\_apr.tail()

| **High** | **Low** | **Open** | **Close** | **Volume** | **Adj Close** | **diff** |
| --- | --- | --- | --- | --- | --- | --- |
| **Date** |  |  |  |  |  |  |  |
| **2019-07-18** | -91.630370 | -300.003543 | 23.728453 | -361.565366 | -721.484570 | -361.564945 | NaN |
| **2019-07-19** | -306.866345 | -549.174276 | -304.229413 | -507.695942 | 3545.349919 | -507.696238 | 10415.360596 |
| **2019-07-22** | -447.581290 | -263.552079 | -616.627759 | 73.595351 | 9587.986607 | 135.089149 | NaN |
| **2019-07-23** | 185.439503 | 340.545723 | 186.882368 | 27.543405 | -5875.100311 | 88.857304 | NaN |
| **2019-07-24** | -252.656209 | -500.009476 | -58.059903 | -211.934472 | 3779.477815 | -211.934472 | 34562.047467 |

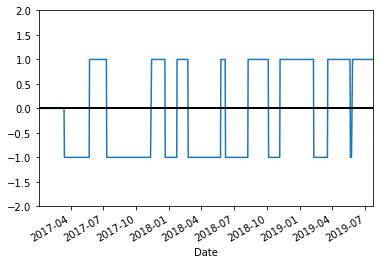
# np.where() is a vectorized if-else function, where a condition is checked for each component of a vector, and the first argument passed is used when the condition holds, and the other passed if it does not

icicibank["Regime"] = np.where(icicibank['20d-50d'] > 0, 1, 0)

# We have 1's for bullish regimes and 0's for everything else. Below I replace bearish regimes's values with -1, and to maintain the rest of the vector, the second argument is apple["Regime"]

icicibank["Regime"] = np.where(icicibank['20d-50d'] < 0, -1, icicibank["Regime"])

icicibank.loc['2016-01-04':'2019-12-31',"Regime"].plot(ylim = (-2,2)).axhline(y = 0, color = "black", lw = 2)



# Create a DataFrame with trades, including the price at the trade and the regime under which the trade is made.

icicibank\_signals = pd.concat([

pd.DataFrame({"Price": icicibank.loc[icicibank["Signal"] == 1, "Adj Close"],

"Regime": icicibank.loc[icicibank["Signal"] == 1, "Regime"],

"Signal": "Buy"}),

pd.DataFrame({"Price": icicibank.loc[icicibank["Signal"] == -1, "Adj Close"],

"Regime": icicibank.loc[icicibank["Signal"] == -1, "Regime"],

"Signal": "Sell"}),

])

icicibank\_signals.sort\_index(inplace = True)

icicibank\_signals

| **Price** | **Regime** | **Signal** |
| --- | --- | --- |
| **Date** |  |  |  |
| **2017-03-15** | 249.864975 | -1 | Sell |
| **2017-05-24** | 268.748474 | 1 | Buy |
| **2017-07-11** | 283.786957 | -1 | Sell |
| **2017-11-13** | 307.889008 | 1 | Buy |
| **2017-12-21** | 309.064697 | -1 | Sell |
| **2018-01-23** | 354.966370 | 1 | Buy |
| **2018-02-23** | 316.216949 | -1 | Sell |
| **2018-05-25** | 289.665497 | 1 | Buy |
| **2018-06-07** | 285.011658 | -1 | Sell |
| **2018-08-09** | 329.907013 | 1 | Buy |
| **2018-10-05** | 305.806061 | -1 | Sell |
| **2018-11-06** | 351.383423 | 1 | Buy |
| **2019-02-08** | 353.373718 | -1 | Sell |
| **2019-03-19** | 396.463165 | 1 | Buy |
| **2019-05-21** | 398.204681 | -1 | Sell |
| **2019-05-27** | 433.382843 | 1 | Buy |
| **2019-07-24** | 408.500000 | 1 | Sell |