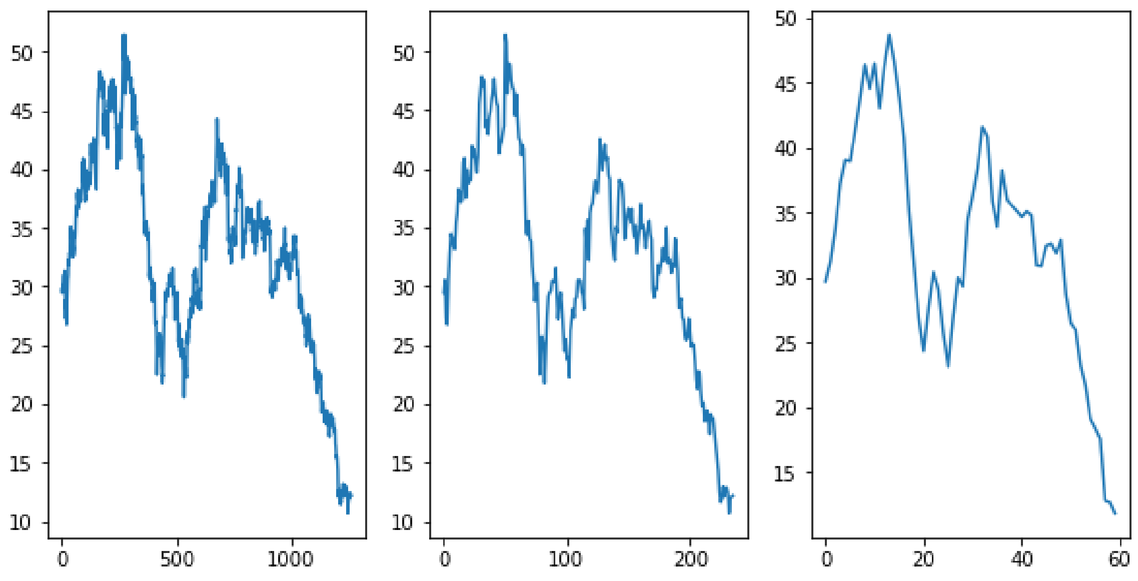
In [2]:

```
#Lab 6 Financial Engineering
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

from numpy import genfromtxt
import scipy.stats as stats
from pandas import to_datetime


def daily_prices(stockname):
    data = pd.read_csv(stockname)
    data.dropna(subset=['Close'], inplace=True)
    X = np.arange(len(data))
    Y = np.array(data['Close'])
    return[X,Y]


def weekly_prices(stockname):
    data = pd.read_csv(stockname)
    data.dropna(subset=['Close'], inplace=True)
    data['Day'] = (to_datetime(data['Date'])).dt.day_name()
    output = data.loc[data['Day'] == 'Monday']
    X = np.arange(len(output))
    Y = np.array(output['Close'])
    return[X,Y]



def monthly_prices(stockname):
    df = pd.read_csv(stockname)
    df.dropna(subset=['Close'], inplace=True)
    df.set_index('Date', inplace=True)
    df.index = pd.to_datetime(df.index)
    data = df.resample('1M').mean()
    X = np.arange(len(data))
    Y = np.array(data['Close'])
    return[X,Y]


def weekly_returns(stockname):
    data = pd.read_csv(stockname)
    data.dropna(subset=['Close'], inplace=True)
    data['Day'] = (to_datetime(data['Date'])).dt.day_name()
    output = data.loc[data['Day'] == 'Monday']
    length = len(output['Close'])
    X_1 = np.array(output['Close'][1:])
    X_2 = np.array(output['Close'][:length-1])
    X = (X_1-X_2)/X_2
    avg = np.average(X)
    var = np.std(X)
    norm_ret = (X - avg)/var
    M, sigma = 0, 1
    X = np.linspace(min(norm_ret), max(norm_ret), len(norm_ret))
    Y = (1/(2*np.pi*(sigma**2))**0.5)*np.exp(-(X-M)**2/(sigma)**2)
    return[norm_ret,X,Y]

def monthly_returns(stockname):
    df = pd.read_csv(stockname)
    df.dropna(subset=['Close'], inplace=True)
    df.set_index('Date', inplace=True)
    df.index = pd.to_datetime(df.index)
    output = df.resample('1M').mean()
    length = len(output['Close'])
    X_1 = np.array(output['Close'][1:])
    X_2 = np.array(output['Close'][:length-1])
    X = (X_1-X_2)/X_2
    avg = np.average(X)
    var = np.std(X)
    norm_ret = (X - avg)/var
    M, sigma = 0, 1
    X = np.linspace(min(norm_ret), max(norm_ret), len(norm_ret))
    Y = (1/(2*np.pi*(sigma**2))**0.5)*np.exp(-(X-M)**2/(sigma)**2)
    return[norm_ret,X,Y]


def daily_returns(stockname):
    output = pd.read_csv(stockname)
    output.dropna(subset=['Close'], inplace=True)
    length = len(output['Close'])
    X_1 = np.array(output['Close'][1:])
    X_2 = np.array(output['Close'][:length-1])
    X = (X_1-X_2)/X_2
    avg = np.average(X)
    var = np.std(X)
    norm_ret = (X - avg)/var
    M, sigma = 0, 1
    X = np.linspace(min(norm_ret), max(norm_ret), len(norm_ret))
    Y = (1/(2*np.pi*(sigma**2))**0.5)*np.exp(-(X-M)**2/(sigma)**2)
    plt.plot(X,Y)
    return[norm_ret,X,Y]


def daily_logreturns(stockname):
    output = pd.read_csv(stockname)
    output.dropna(subset=['Close'], inplace=True)
    length = len(output['Close'])
    X_1 = np.array(output['Close'][1:])
    X_2 = np.array(output['Close'][:length-1])
    X = (X_1-X_2)/X_2
    logX = np.log(1+X)
    avg = np.average(logX)
    var = np.std(logX)
    norm_ret = (logX - avg)/var
    M, sigma = 0, 1
    X = np.linspace(min(norm_ret), max(norm_ret), len(norm_ret))
    Y = (1/(2*np.pi*(sigma**2))**0.5)*np.exp(-(X-M)**2/(sigma)**2)
    return[norm_ret,X,Y]

def monthly_logreturns(stockname):
    df = pd.read_csv(stockname)
    df.dropna(subset=['Close'], inplace=True)
    df.set_index('Date', inplace=True)
    df.index = pd.to_datetime(df.index)
    output = df.resample('1M').mean()
    length = len(output['Close'])
    X_1 = np.array(output['Close'][1:])
    X_2 = np.array(output['Close'][:length-1])
    X = np.log(X_1/X_2)
    avg = np.average(X)
```

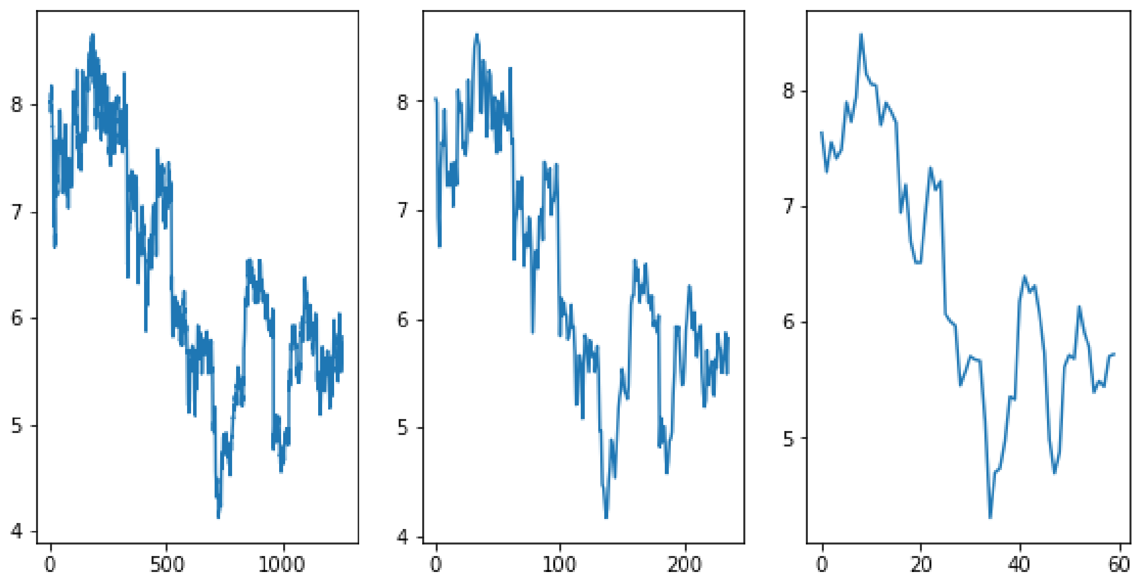
```python
    var = np.std(X)
    norm_ret = (X - avg)/var
    M, sigma = 0, 1
    X = np.linspace(min(norm_ret), max(norm_ret), len(norm_ret))
    Y = (1/(2*np.pi*(sigma**2))**0.5)*np.exp(-(X-M)**2/(sigma)**2)
    return[norm_ret,X,Y]


def weekly_logreturns(stockname):
    data = pd.read_csv(stockname)
    data.dropna(subset=['Close'], inplace=True)
    data['Day'] = (to_datetime(data['Date'])).dt.day_name()
    output = data.loc[data['Day'] == 'Monday']
    length = len(output['Close'])
    X_1 = np.array(output['Close'][1:])
    X_2 = np.array(output['Close'][:length-1])
    X = np.log(X_1/X_2)
    avg = np.average(X)
    var = np.std(X)
    norm_ret = (X - avg)/var
    M, sigma = 0, 1
    X = np.linspace(min(norm_ret), max(norm_ret), len(norm_ret))
    Y = (1/(2*np.pi*(sigma**2))**0.5)*np.exp(-(X-M)**2/(sigma)**2)
    return[norm_ret,X,Y]


def predicted_prices_daily(stockname):
    output = pd.read_csv(stockname)
    output.dropna(subset=['Close'], inplace=True)
    length = len(output['Close'])
    X_1 = np.array(output['Close'][1:])
    X_2 = np.array(output['Close'][:length-1])
    X = (X_1-X_2)/X_2
    X = np.log(1+X)
    f = X[:987]
    M = np.mean(f)
    sigma = np.std(f)
    n = len(X)-987
    phi = np.random.normal(0,1,n)
    W = np.zeros(n)
    W[0] = 0
    for i in range(1, n):
        W[i] = W[i-1]+phi[i]
    S = np.zeros(n)
    S[0] = output['Close'][987]
    for i in range(1, n):
        S[i] = S[0]*np.exp(sigma*W[i]+(M-0.5*(sigma**2))*i/240)
    actual_price = np.array(output['Close'])
    predicted_price = actual_price[987:]
    Y_2 = predicted_price
    return[S,Y_2]



def predicted_prices_monthly(stockname):
    df = pd.read_csv(stockname)
    df.dropna(subset=['Close'], inplace=True)
    df.set_index('Date', inplace=True)
    df.index = pd.to_datetime(df.index)
    output = df.resample('1M').mean()
    length = len(output['Close'])
    X_1 = np.array(output['Close'][1:])
    X_2 = np.array(output['Close'][:length-1])
    X = (X_1-X_2)/X_2
    X = np.log(1+X)
    f = X[:48]
    M = np.mean(f)
    sigma = np.std(f)
    n = len(X)-48
    phi = np.random.normal(0,1,n)
    W = np.zeros(n)
    W[0] = 0
    for i in range(1, n):
        W[i] = W[i-1]+phi[i]
    S = np.zeros(n)
    S[0] = output['Close'][48]
    for i in range(1, n):
        S[i] = S[0]*np.exp(sigma*W[i]+(M-0.5*(sigma**2))*i/240)
    actual_price = np.array(output['Close'])
    predicted_price = actual_price[48:]
    Y_2 = predicted_price
    return[S,Y_2]




STOCKS = ['TTM.csv','NOK.csv','RS.csv'
            ,'AAPL.csv','ASIANPAINT.BO.csv',
            'BERGEPAINT.BO.csv','BLK.csv','HAVELLS.BO.csv',
            'HDB.csv','JINDALSTEL.NS.csv','MARUTI.NS.csv',
       'MUTHOOTFIN.NS.csv','ONGC.NS.csv','RELIANCE.NS.csv','HAVELLS.NS.csv','BOM
DYEING.NS.csv','BAJFINANCE.NS.csv','ADANIENT.NS.csv','VOLTAS.NS.csv','BERGEPAIN
T.NS.csv']



for name in STOCKS:
    print("  DAILY, MONTHLY & WEEKLY STOCK PRICES for: " + name)
    fig, ax = plt.subplots(nrows=1, ncols=3,figsize = (10,5),squeeze=False)
    ax[0,0].plot(daily_prices(name)[0],daily_prices(name)[1])
    ax[0,1].plot(weekly_prices(name)[0],weekly_prices(name)[1])
    ax[0,2].plot(monthly_prices(name)[0],monthly_prices(name)[1])
    plt.show()

for name in STOCKS:
    print(" DAILY, MONTHLY & WEEKLY RETURNS for: " + name)
    fig, ax = plt.subplots(nrows=1, ncols=3,figsize = (10,5),squeeze=False)
    ax[0,0].hist(daily_returns(name)[0],density=True)
    ax[0,0].plot(daily_returns(name)[1],daily_returns(name)[2])
    ax[0,1].hist(weekly_returns(name)[0],density=True)
    ax[0,1].plot(weekly_returns(name)[1],weekly_returns(name)[2])
    ax[0,2].hist(monthly_returns(name)[0],density=True)
    ax[0,2].plot(monthly_returns(name)[1],monthly_returns(name)[2])
    plt.show()


for name in STOCKS:
   print("  DAILY, MONTHLY & WEEKLY LOG RETURNS for: " + name)
   fig, ax = plt.subplots(nrows=1, ncols=3,figsize = (10,5),squeeze=False)
   ax[0,0].hist(daily_logreturns(name)[0],density=True)
   ax[0,0].plot(daily_logreturns(name)[1],daily_logreturns(name)[2])
   ax[0,1].hist(weekly_logreturns(name)[0],density=True)
   ax[0,1].plot(weekly_logreturns(name)[1],weekly_logreturns(name)[2])
   ax[0,2].hist(monthly_logreturns(name)[0],density=True)
   ax[0,2].plot(monthly_logreturns(name)[1],monthly_logreturns(name)[2])
   plt.show()



print("RED IS ACTUAL STOCK PRICE AND BLUE IS PREDICTED STOCK PRICE")
for name in STOCKS:
    print(" DAILY, MONTHLY & WEEKLY PREDICTED PRICES AND ACTUAL PRICES for: " +
name)
    fig, ax = plt.subplots(nrows=1, ncols=3,figsize = (10,5),squeeze=False)
    ax[0,0].plot(np.array(range(1,len(predicted_prices_daily(name)[0])+1)),predi
cted_prices_daily(name)[0])
    ax[0,0].plot(np.array(range(1,len(predicted_prices_daily(name)[1])+1)),predi
cted_prices_daily(name)[1])
    ax[0,2].plot(np.array(range(1,len(predicted_prices_monthly(name)[0])+1)),pre
dicted_prices_monthly(name)[0])
    ax[0,2].plot(np.array(range(1,len(predicted_prices_monthly(name)[1])+1)),pre
dicted_prices_monthly(name)[1])
    plt.show()




def predictedpricesweekly(stockname):
    data = pd.read_csv(stockname)
    data.dropna(subset=['Close'], inplace=True)
    data['Day'] = (to_datetime(data['Date'])).dt.day_name()
    output = data.loc[data['Day'] == 'Monday']
    length = len(output['Close'])
    X_1 = np.array(output['Close'][1:])
    X_2 = np.array(output['Close'][:length-1])
    X = (X_1-X_2)/X_2
    X = np.log(1+X)
    f = X[:208]
    M = np.mean(f)
    sigma = np.std(f)
    n = len(X)-208
    phi = np.random.normal(0,1,n)
    W = np.zeros(n)
    W[0] = 0
    for i in range(1, n):
        W[i] = W[i-1]+phi[i]
    S = np.zeros(n)
    S[0] = output['Close'][208]
    for i in range(1, n):
        S[i] = S[0]*np.exp(sigma*W[i]+(M-0.5*(sigma**2))*i/240)
    actual_price = np.array(output['Close'])
    predicted_price = actual_price[208:]
    Y_2 = predicted_price
    return[S,Y_2]


print("RED IS ACTUAL STOCK PRICE & BLUE IS PREDICTED STOCK PRICE")
for name in STOCKS:
    print("   DAILY, MONTHLY & WEEKLY PREDICTED PRICES AND ACTUAL PRICES: " + na
me)
    fig, ax = plt.subplots(nrows=1, ncols=3,figsize = (10,5),squeeze=False)
    ax[0,0].plot(np.array(range(1,len(predicted_prices_daily(name)[0])+1)),predi
cted_prices_daily(name)[0])
```

```
    ax[0,0].plot(np.array(range(1,len(predicted_prices_daily(name)[1])+1)),predi
cted_prices_daily(name)[1])
    ax[0,1].plot(np.array(range(1,len(predictedpricesweekly(name)[0])+1)),predic
tedpricesweekly(name)[0])
    ax[0,1].plot(np.array(range(1,len(predictedpricesweekly(name)[1])+1)),predic
tedpricesweekly(name)[1])
    ax[0,2].plot(np.array(range(1,len(predicted_prices_monthly(name)[0])+1)),pre
dicted_prices_monthly(name)[0])
    ax[0,2].plot(np.array(range(1,len(predicted_prices_monthly(name)[1])+1)),pre
dicted_prices_monthly(name)[1])
    plt.show()
```

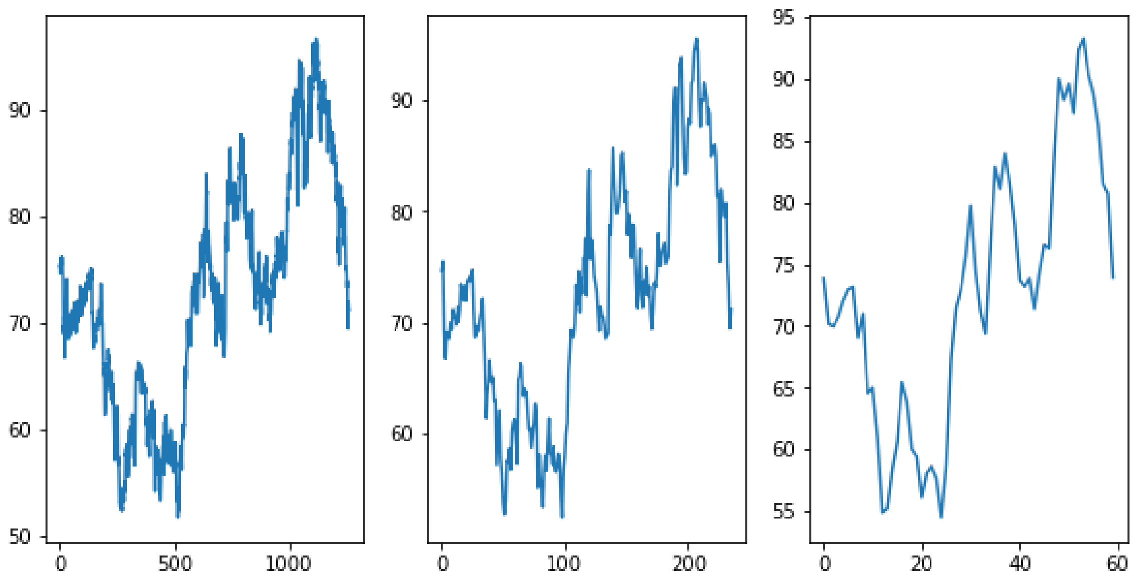
DAILY, MONTHLY & WEEKLY STOCK PRICES for: TTM.csv



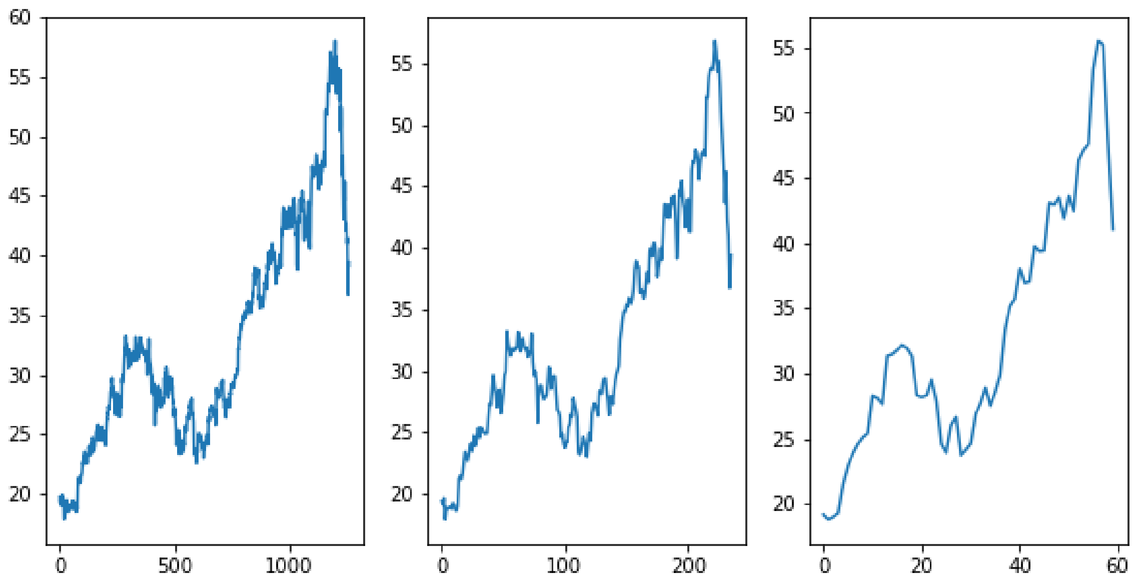
DAILY, MONTHLY & WEEKLY STOCK PRICES for: NOK.csv



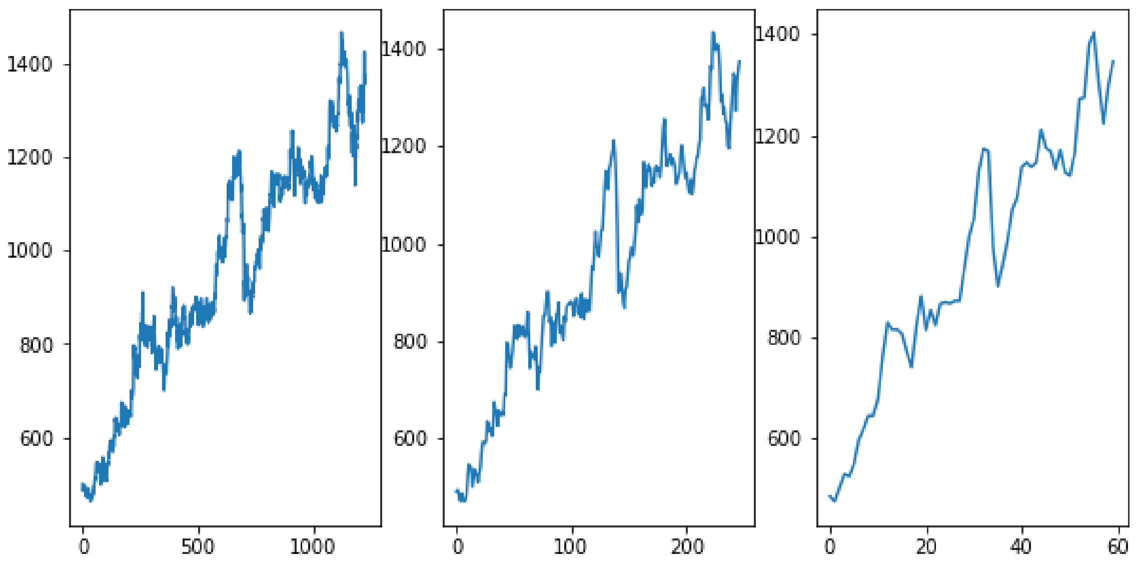
DAILY, MONTHLY & WEEKLY STOCK PRICES for: RS.csv



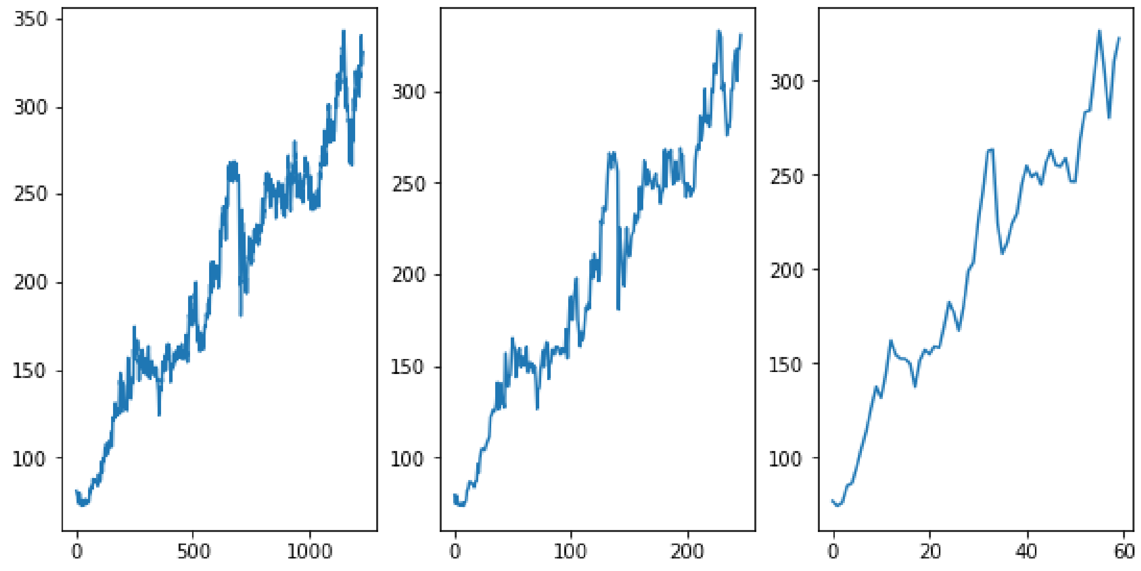
DAILY, MONTHLY & WEEKLY STOCK PRICES for: AAPL.csv



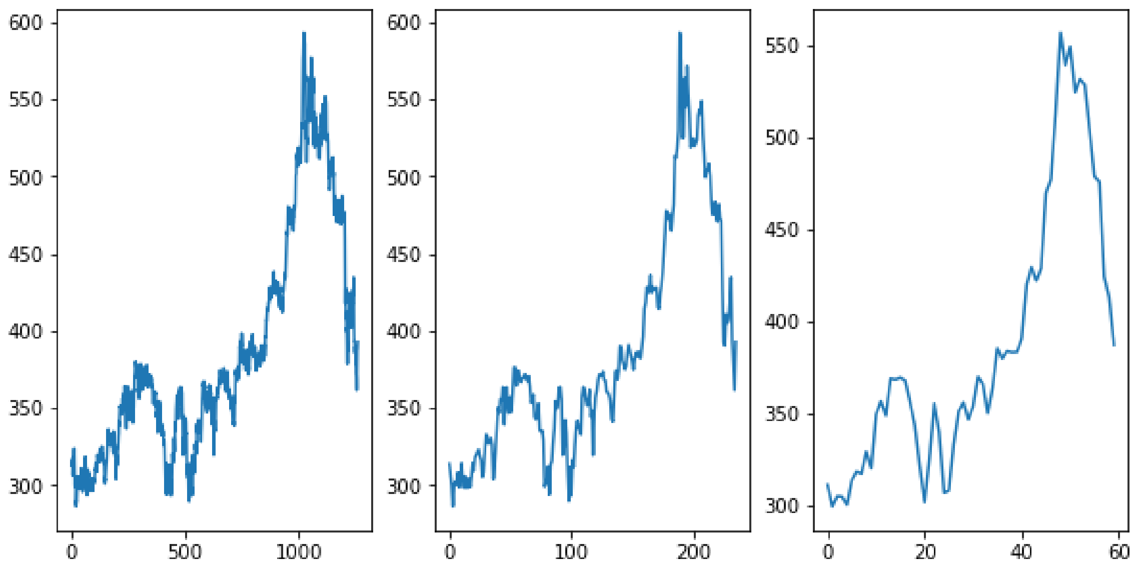
DAILY, MONTHLY & WEEKLY STOCK PRICES for: ASIANPAINT.BO.csv



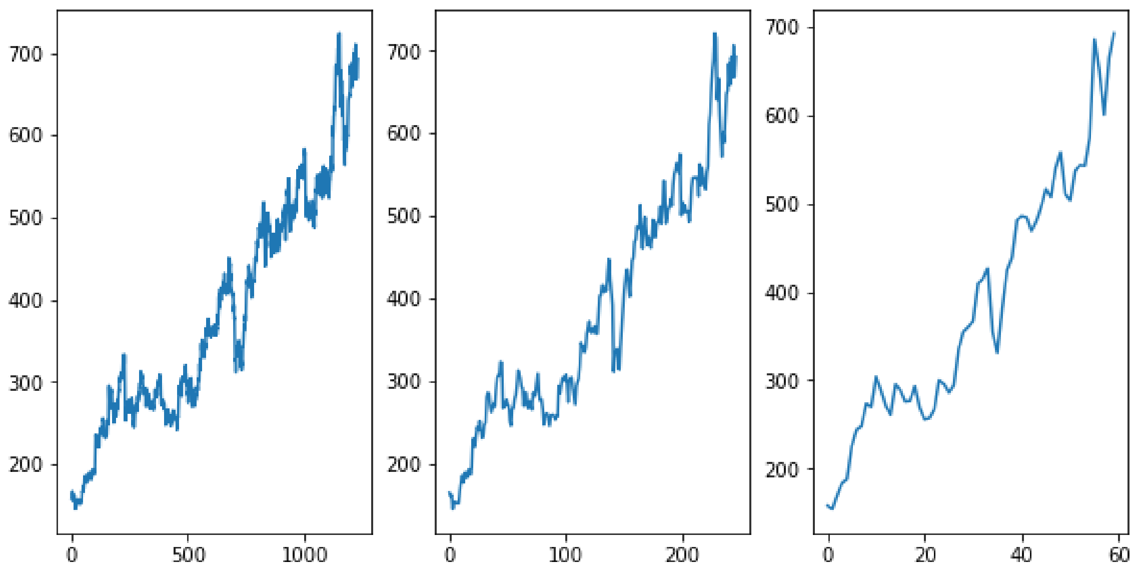
DAILY, MONTHLY & WEEKLY STOCK PRICES for: BERGEPAINT.BO.csv



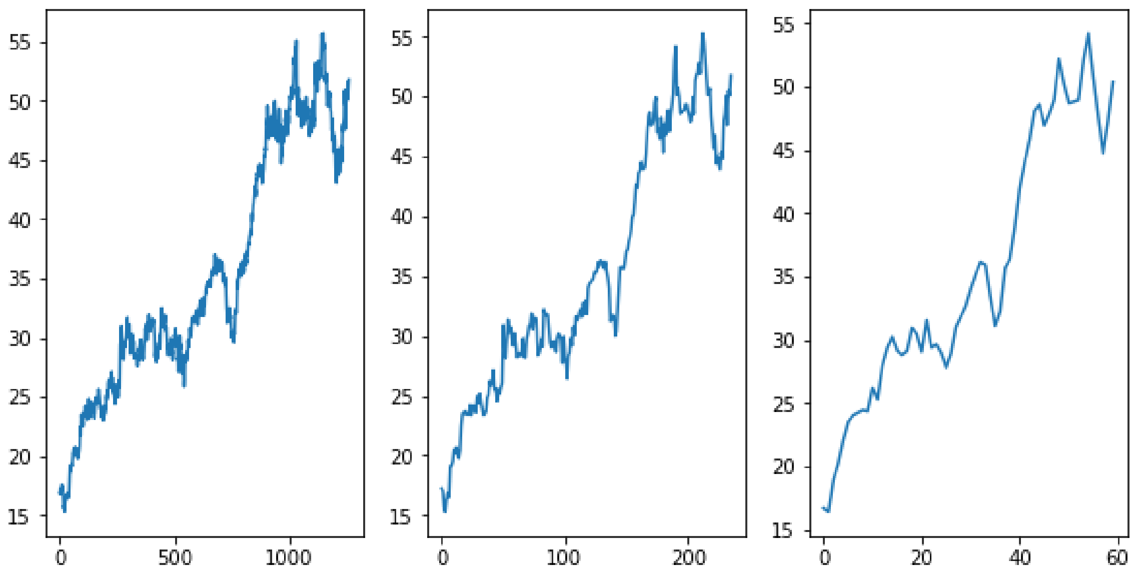
DAILY, MONTHLY & WEEKLY STOCK PRICES for: BLK.csv



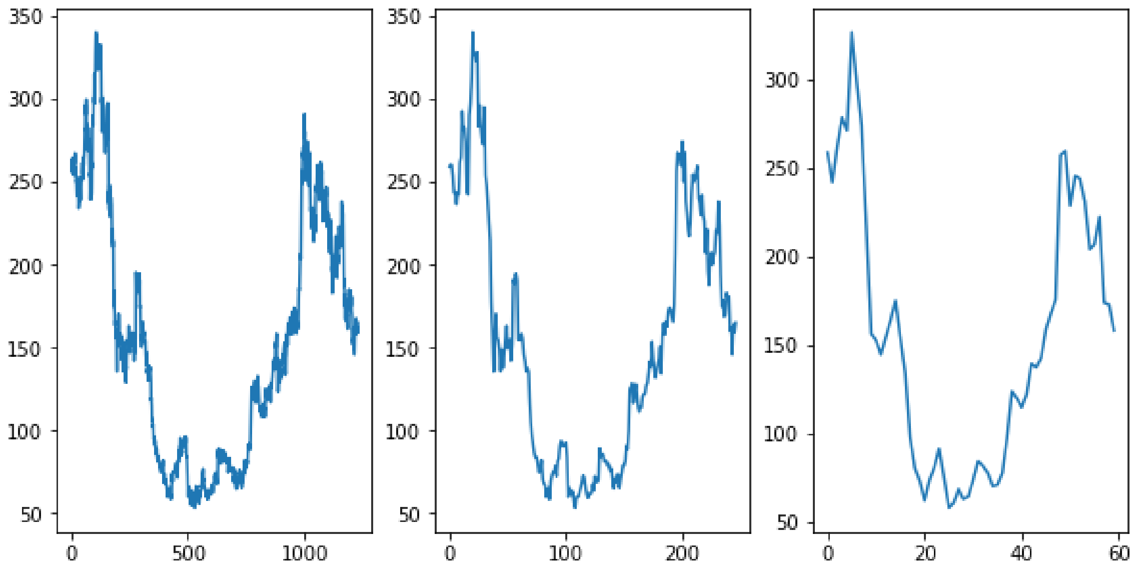
DAILY, MONTHLY & WEEKLY STOCK PRICES for: HAVELLS.BO.csv



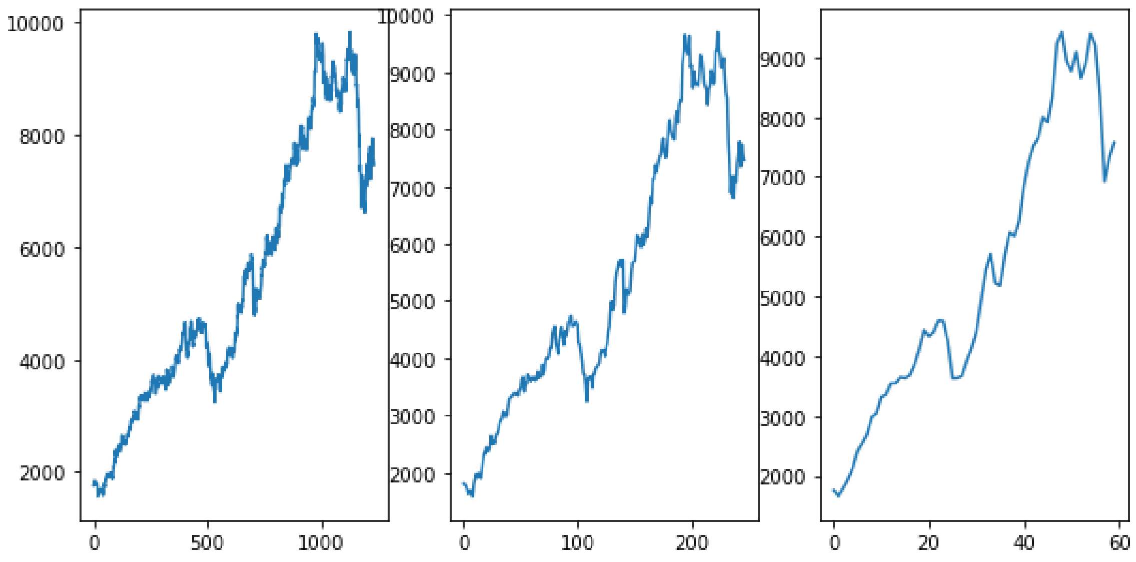
DAILY, MONTHLY & WEEKLY STOCK PRICES for: HDB.csv



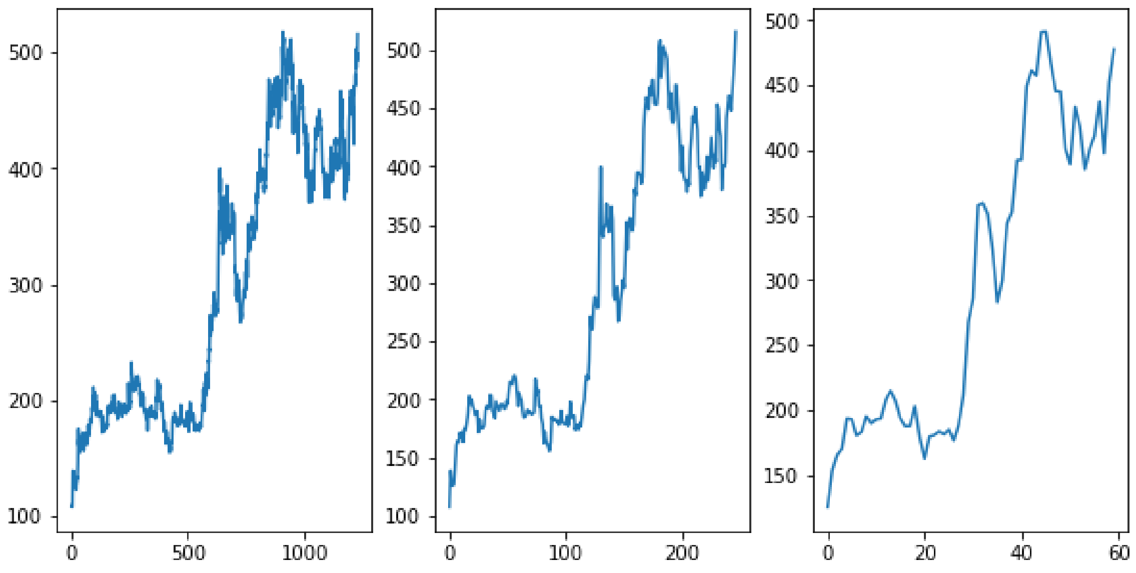
DAILY, MONTHLY & WEEKLY STOCK PRICES for: JINDALSTEL.NS.csv



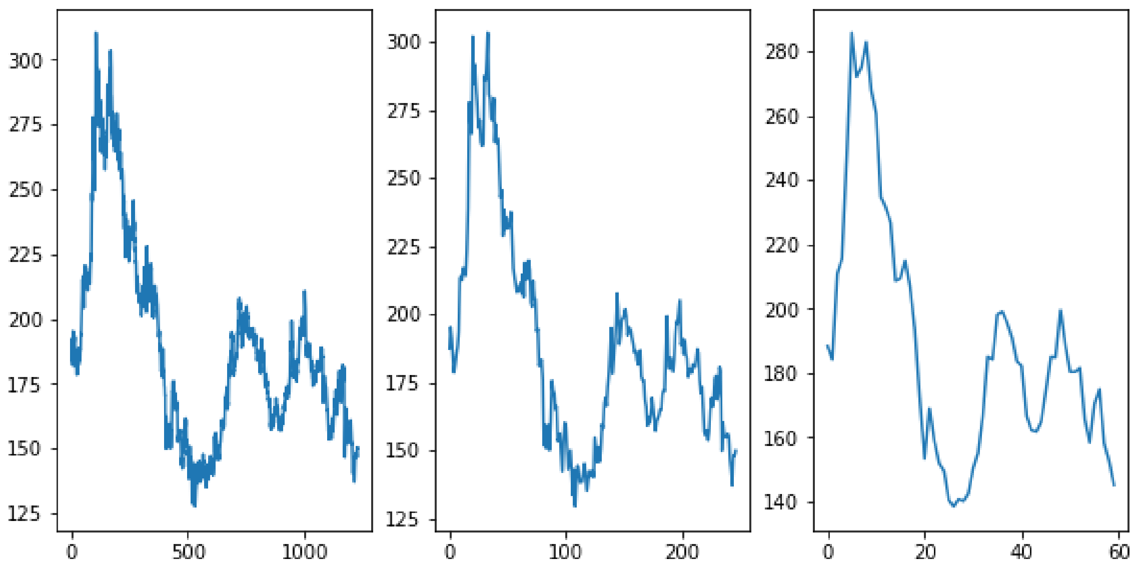
DAILY, MONTHLY & WEEKLY STOCK PRICES for: MARUTI.NS.csv



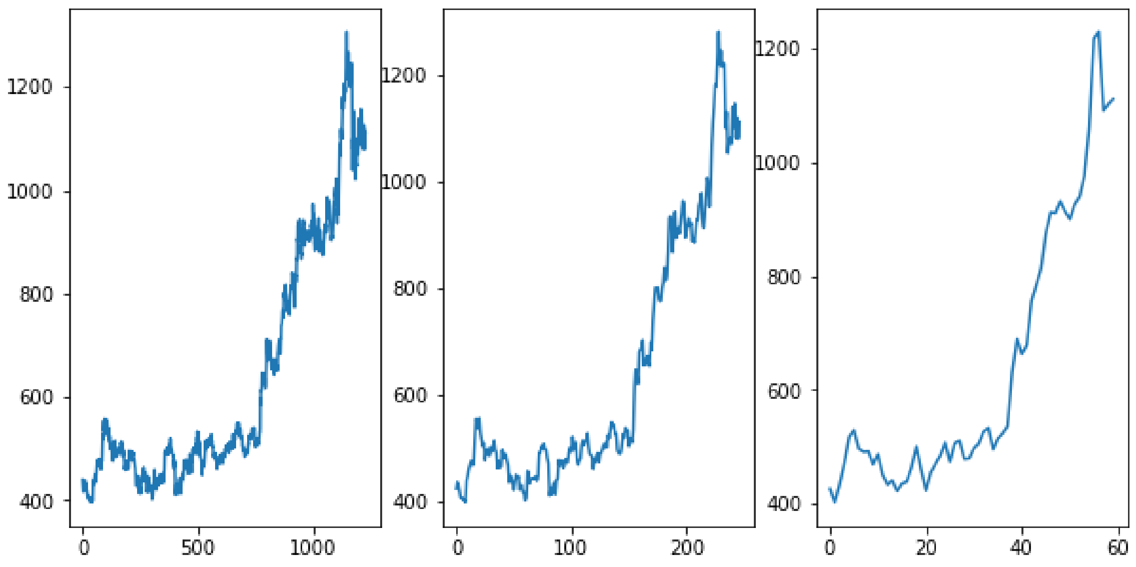
DAILY, MONTHLY & WEEKLY STOCK PRICES for: MUTHOOTFIN.NS.csv



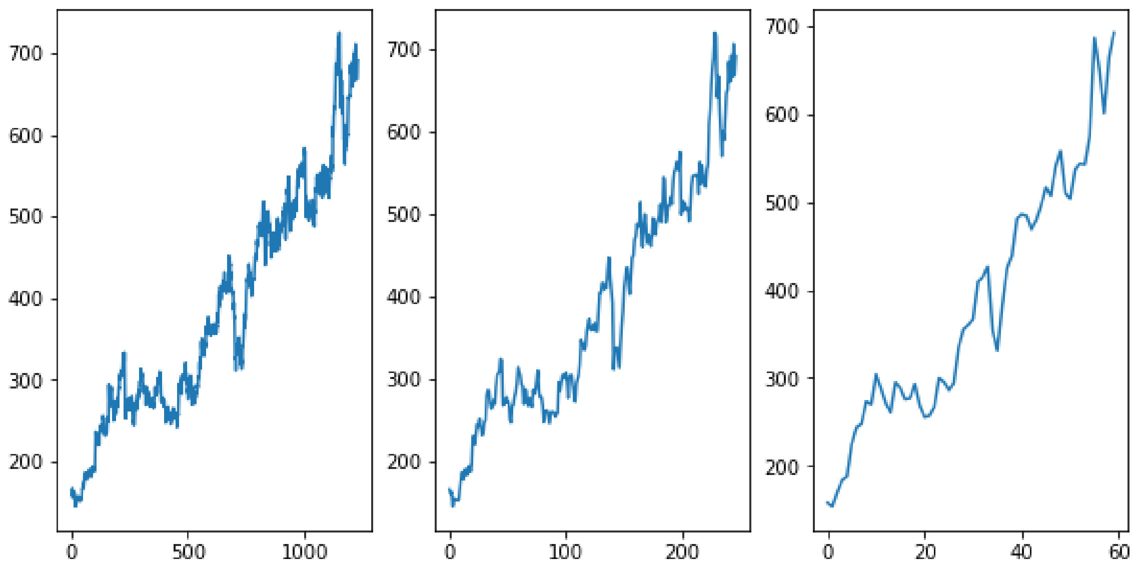
DAILY, MONTHLY & WEEKLY STOCK PRICES for: ONGC.NS.csv



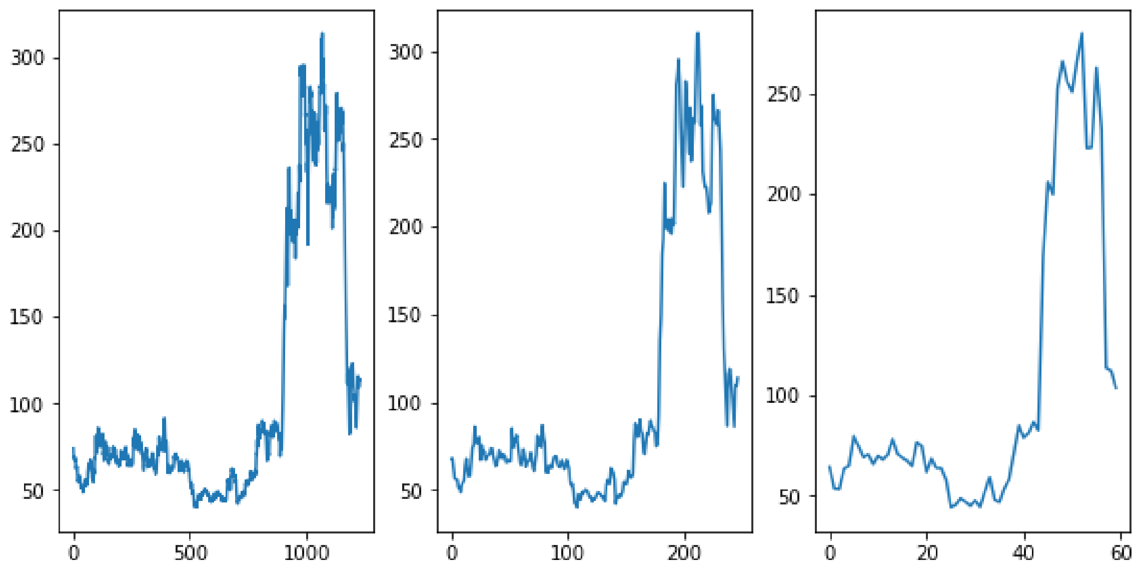
DAILY, MONTHLY & WEEKLY STOCK PRICES for: RELIANCE.NS.csv



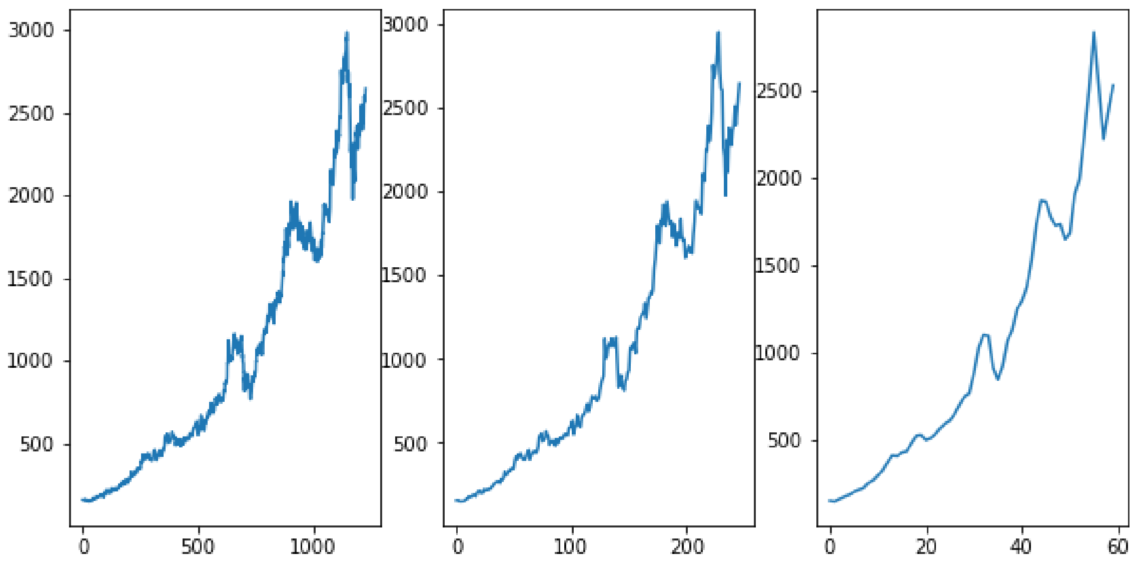
DAILY, MONTHLY & WEEKLY STOCK PRICES for: HAVELLS.NS.csv



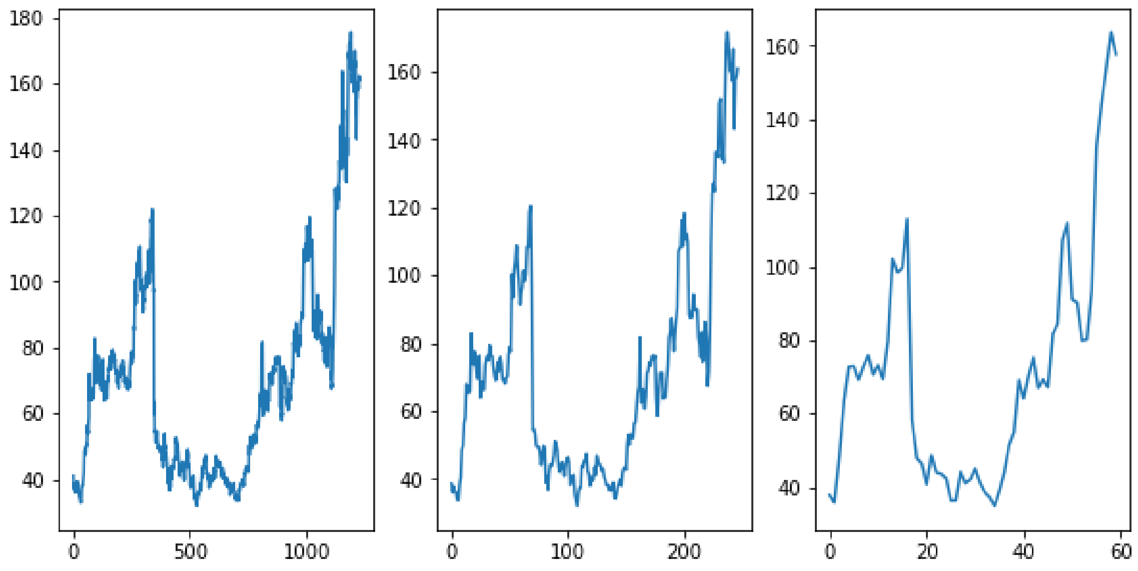
DAILY, MONTHLY & WEEKLY STOCK PRICES for: BOMDYEING.NS.csv



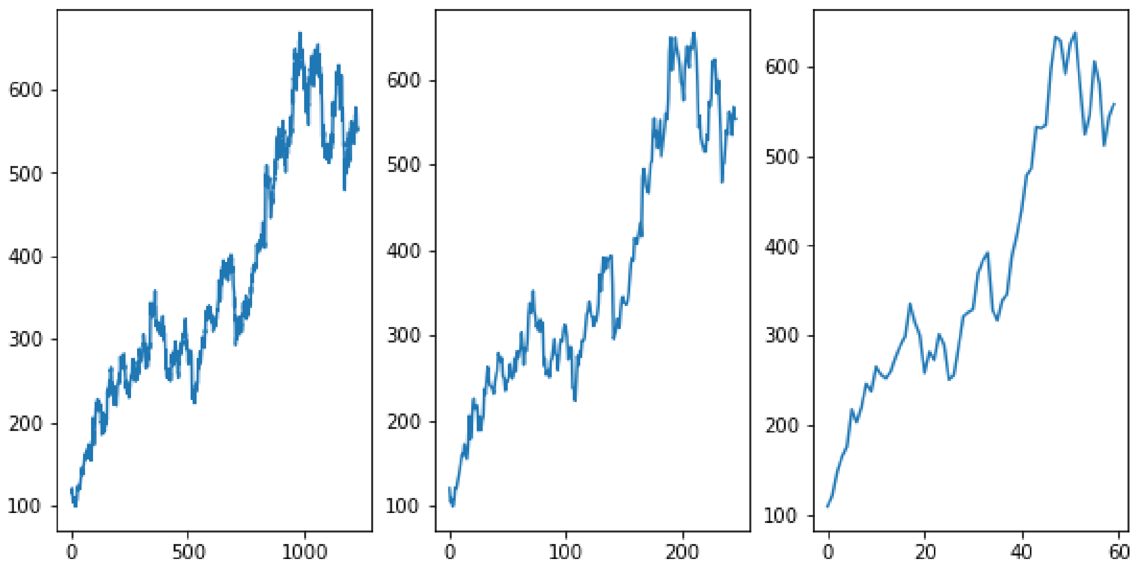
DAILY, MONTHLY & WEEKLY STOCK PRICES for: BAJFINANCE.NS.csv



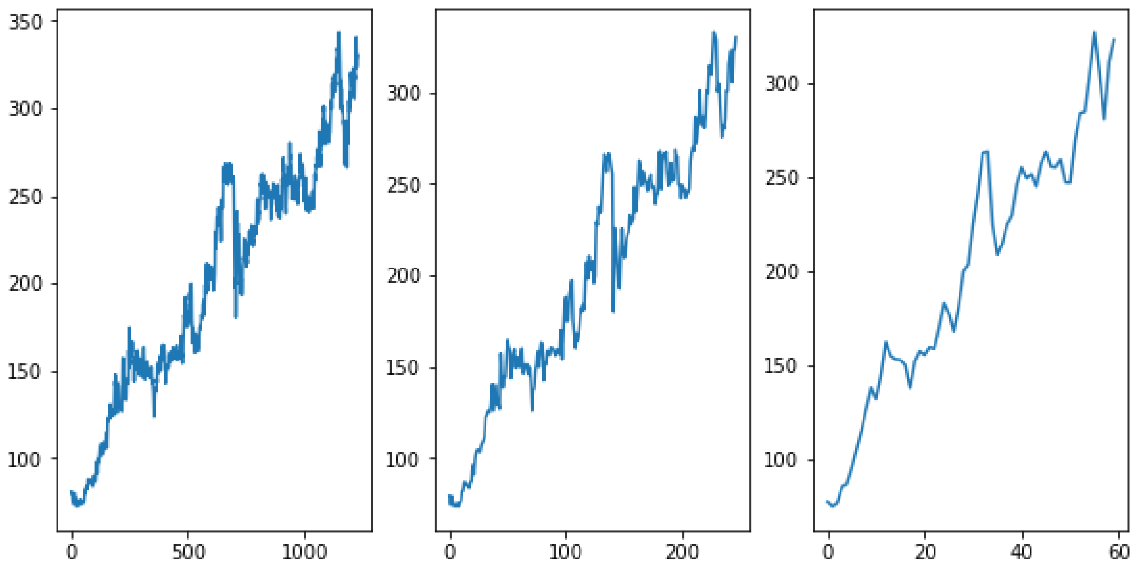
DAILY, MONTHLY & WEEKLY STOCK PRICES for: ADANIENT.NS.csv



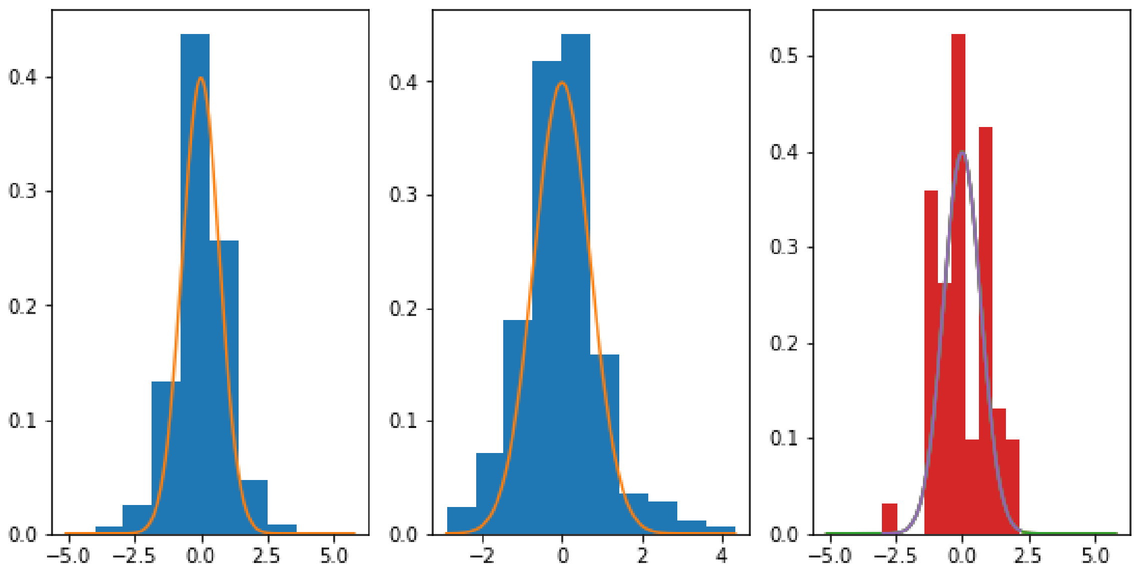
DAILY, MONTHLY & WEEKLY STOCK PRICES for: VOLTAS.NS.csv



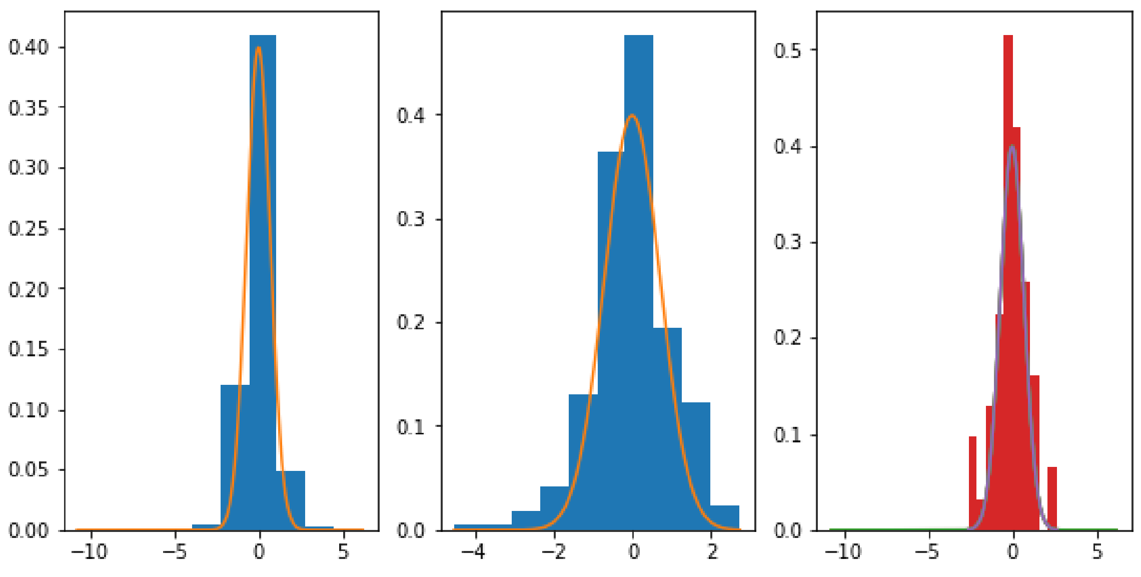
DAILY, MONTHLY & WEEKLY STOCK PRICES for: BERGEPAINT.NS.csv



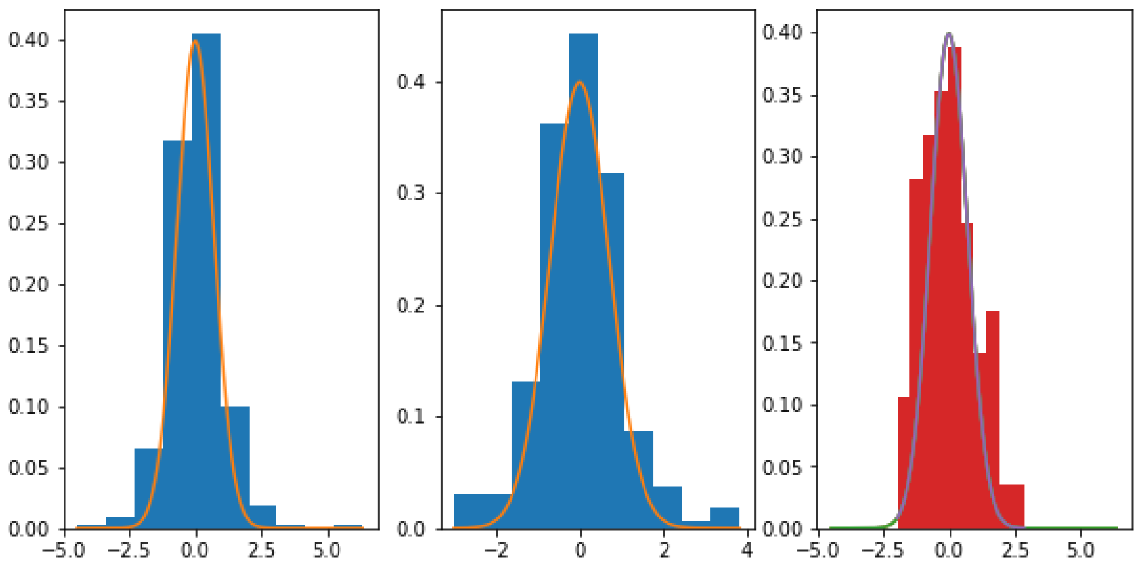
DAILY, MONTHLY & WEEKLY RETURNS for: TTM.csv



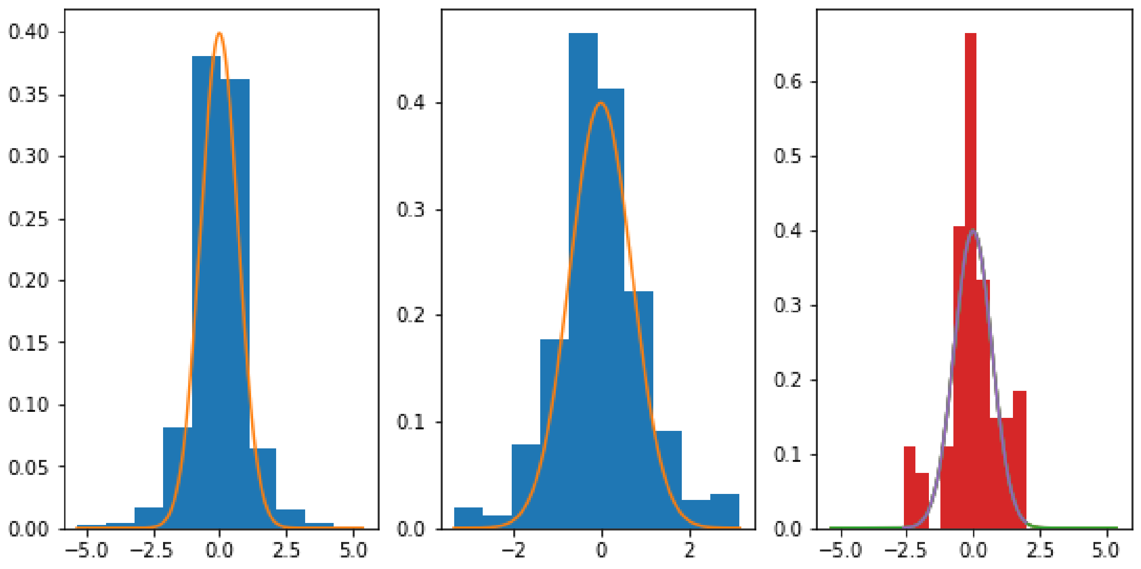
DAILY, MONTHLY & WEEKLY RETURNS for: NOK.csv



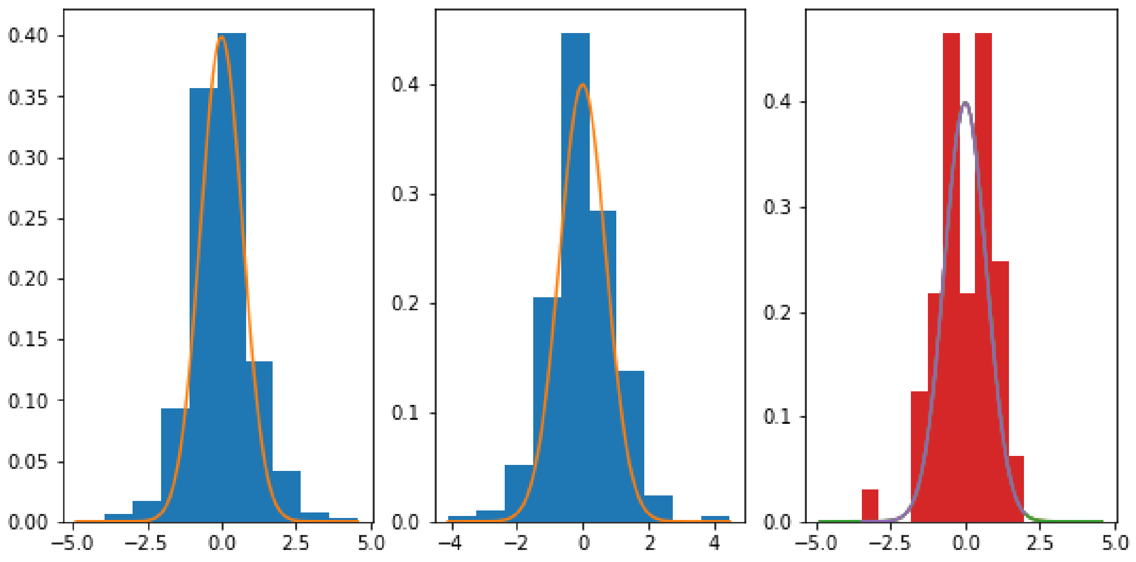
DAILY, MONTHLY & WEEKLY RETURNS for: RS.csv



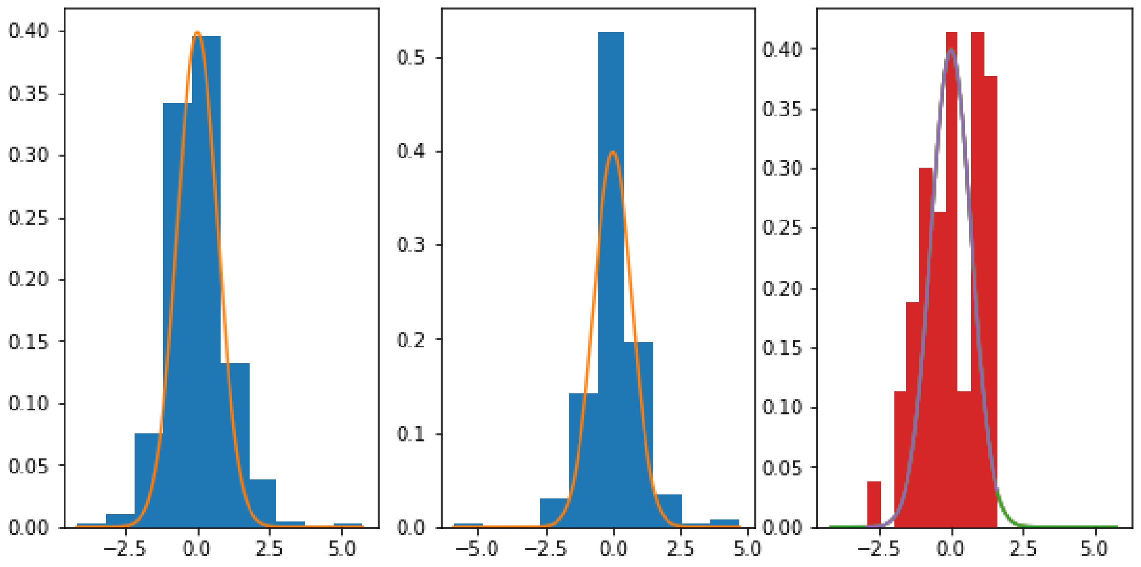
DAILY, MONTHLY & WEEKLY RETURNS for: AAPL.csv



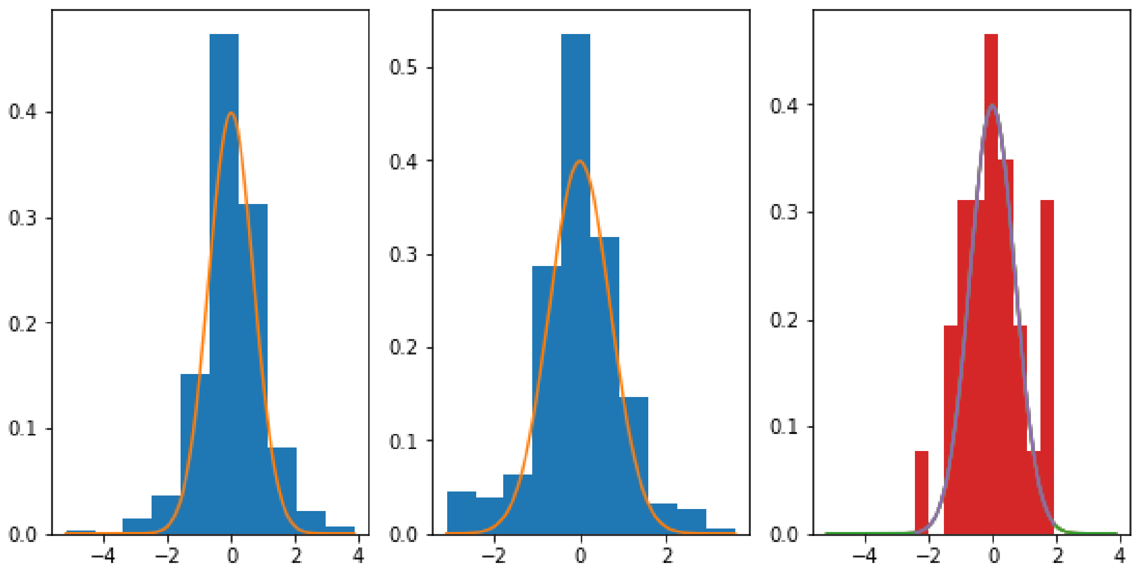
DAILY, MONTHLY & WEEKLY RETURNS for: ASIANPAINT.BO.csv



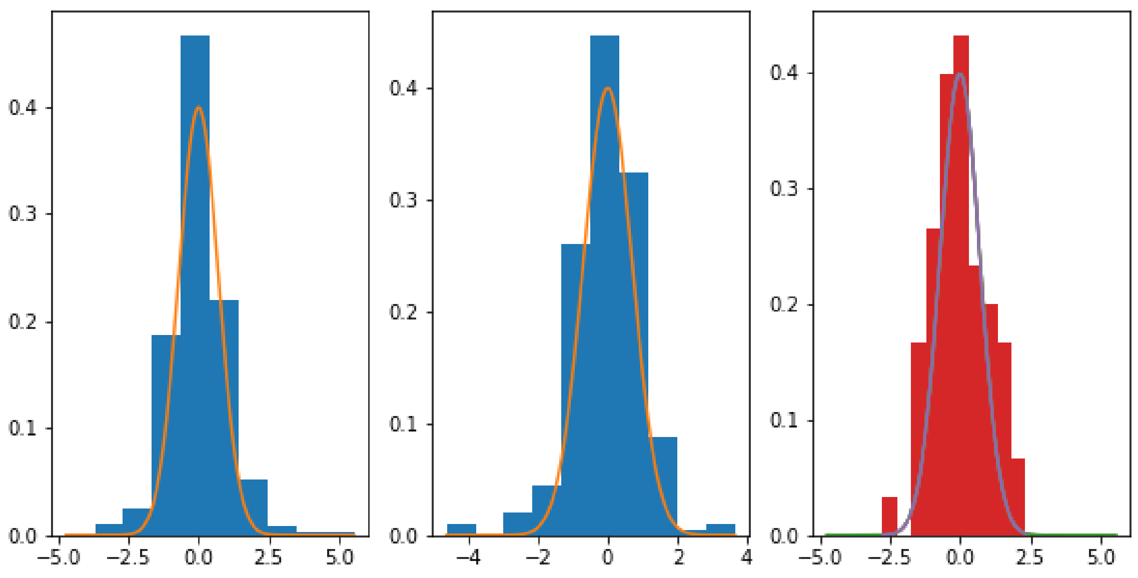
DAILY, MONTHLY & WEEKLY RETURNS for: BERGEPAINT.BO.csv



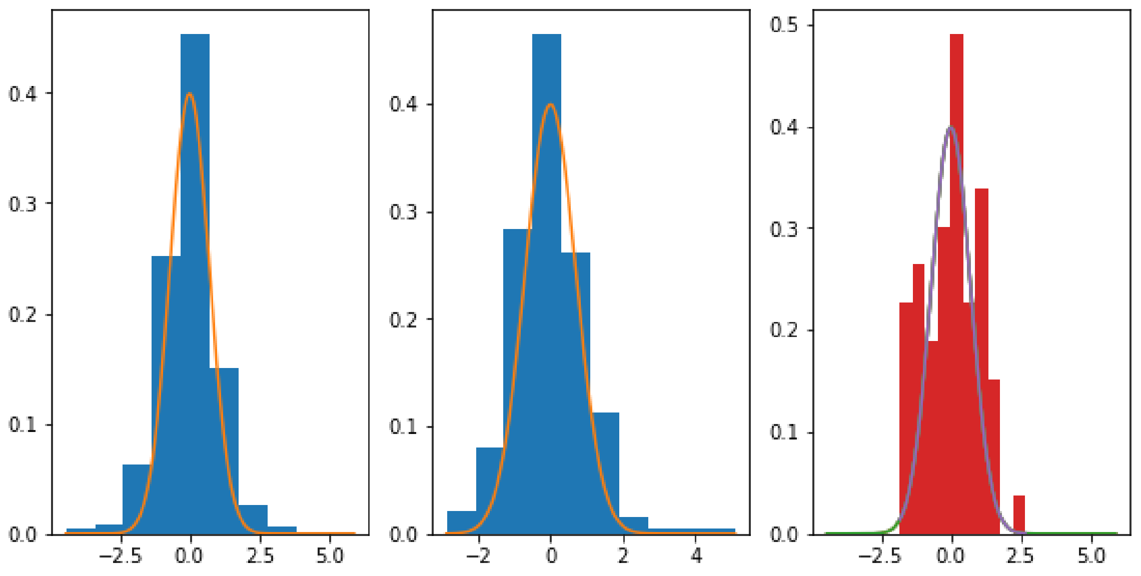
DAILY, MONTHLY & WEEKLY RETURNS for: BLK.csv



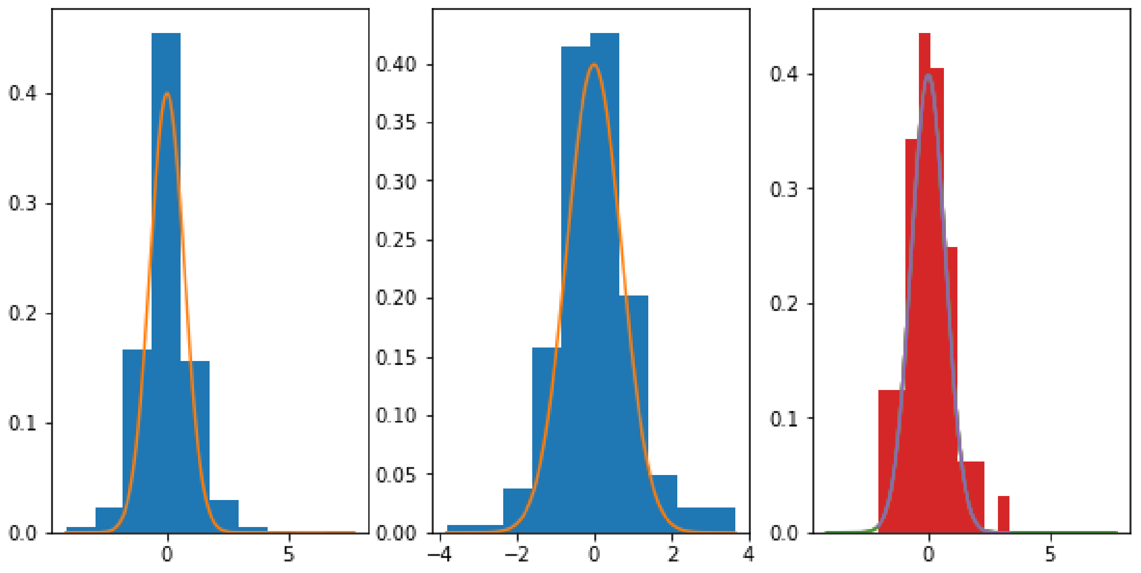
DAILY, MONTHLY & WEEKLY RETURNS for: HAVELLS.BO.csv



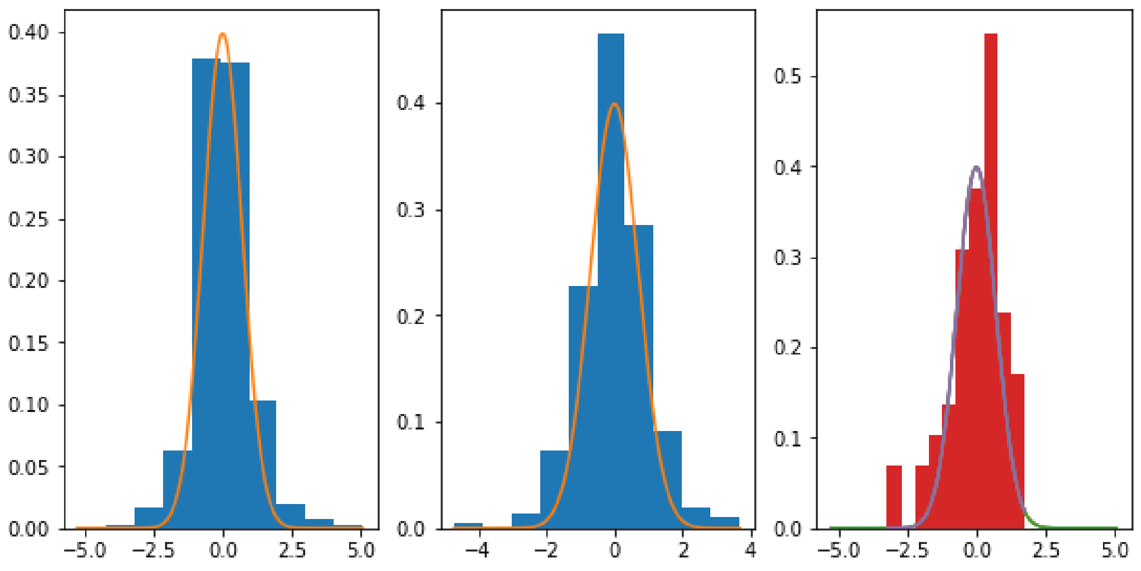
DAILY, MONTHLY & WEEKLY RETURNS for: HDB.csv



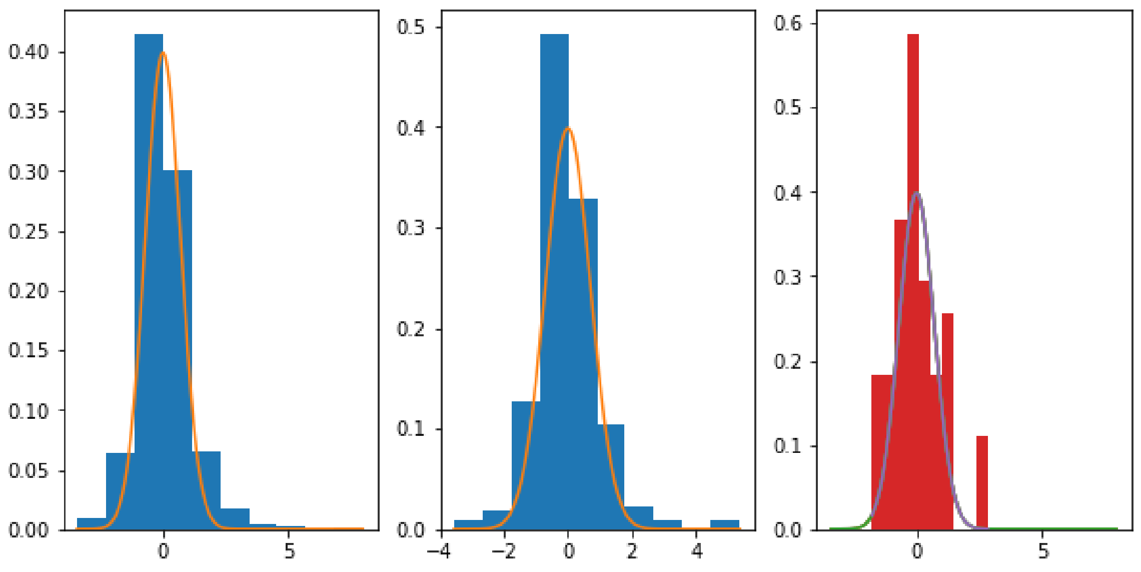
```
DAILY, MONTHLY & WEEKLY RETURNS for: JINDALSTEL.NS.csv
```



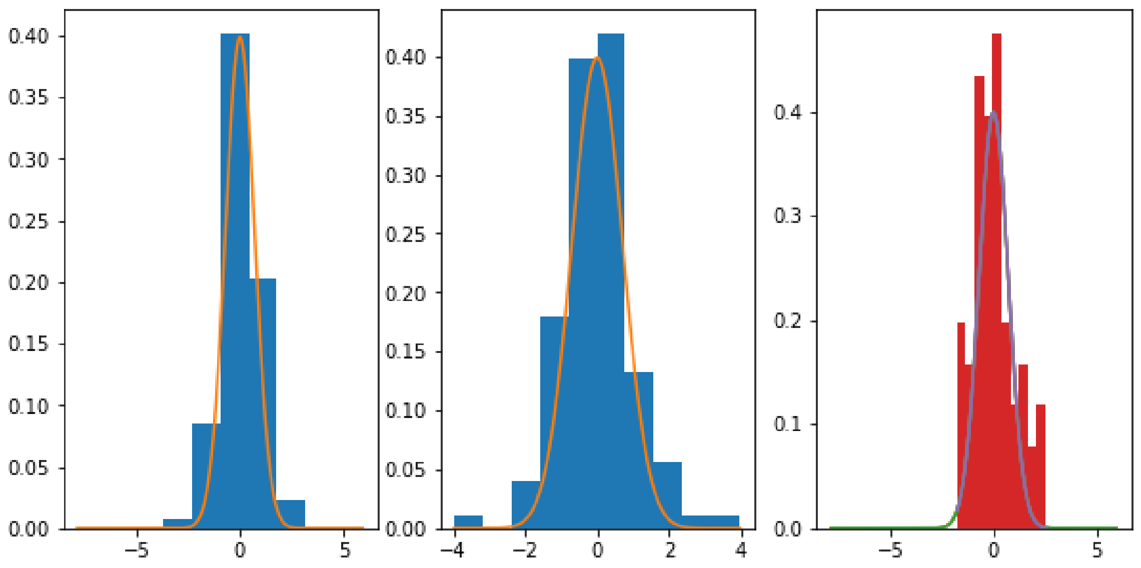
```
DAILY, MONTHLY & WEEKLY RETURNS for: MARUTI.NS.csv
```



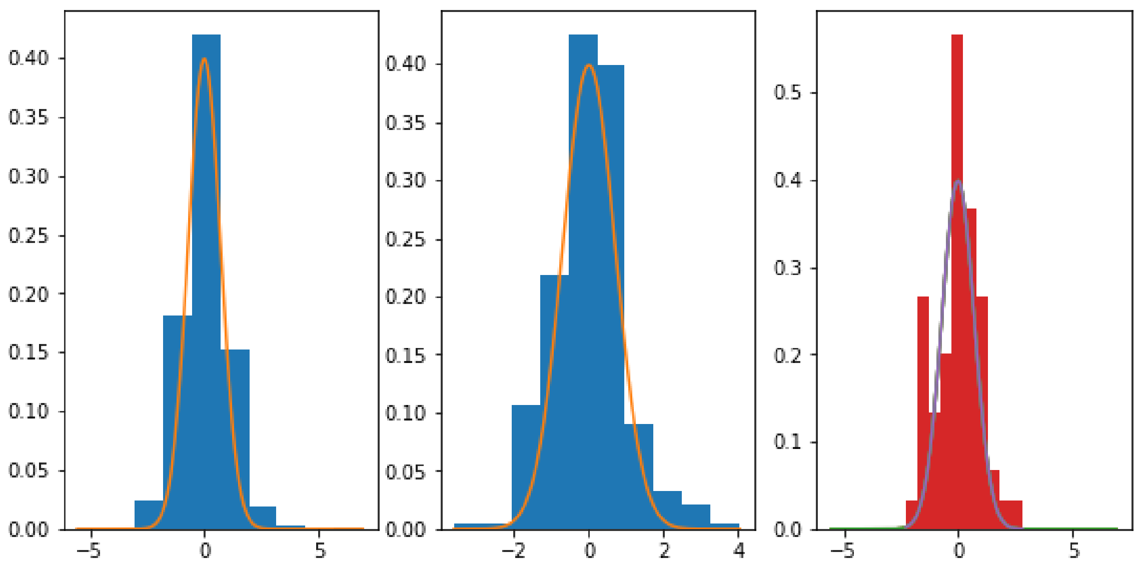
DAILY, MONTHLY & WEEKLY RETURNS for: MUTHOOTFIN.NS.csv



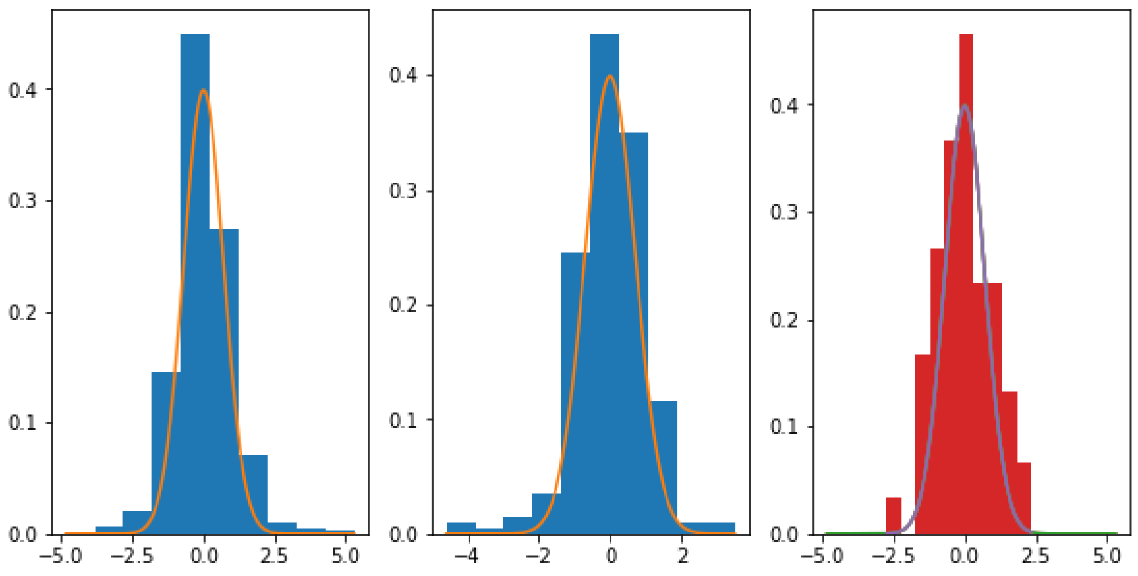
DAILY, MONTHLY & WEEKLY RETURNS for: ONGC.NS.csv



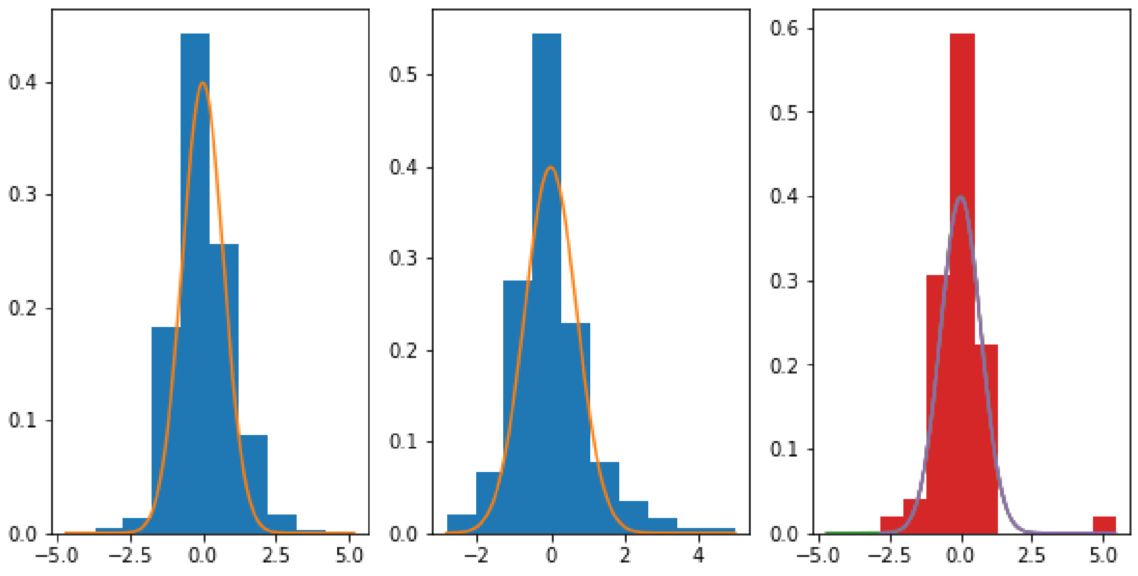
DAILY, MONTHLY & WEEKLY RETURNS for: RELIANCE.NS.csv



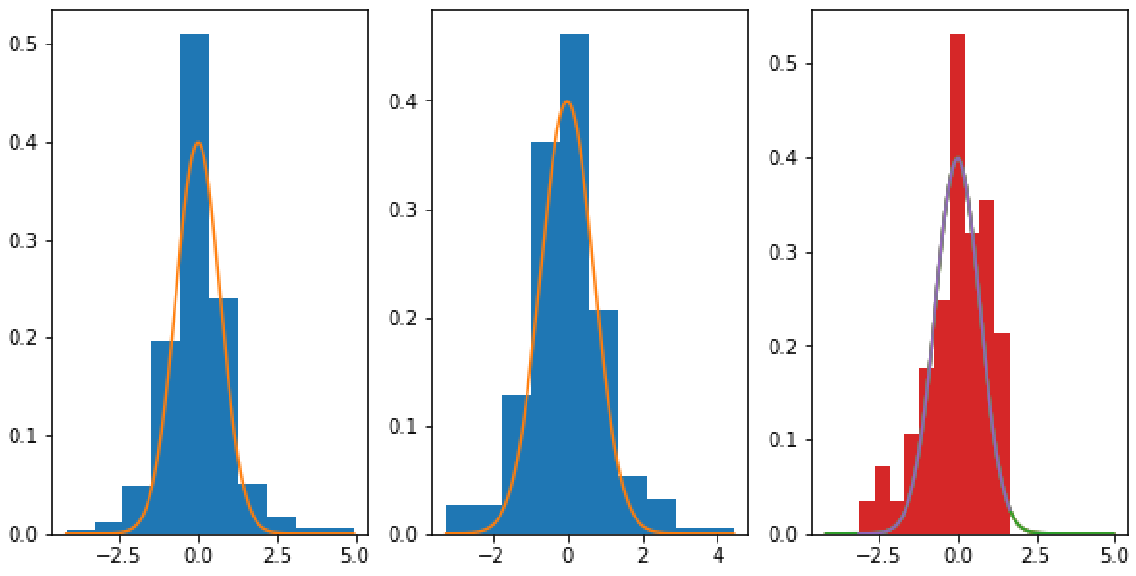
DAILY, MONTHLY & WEEKLY RETURNS for: HAVELLS.NS.csv



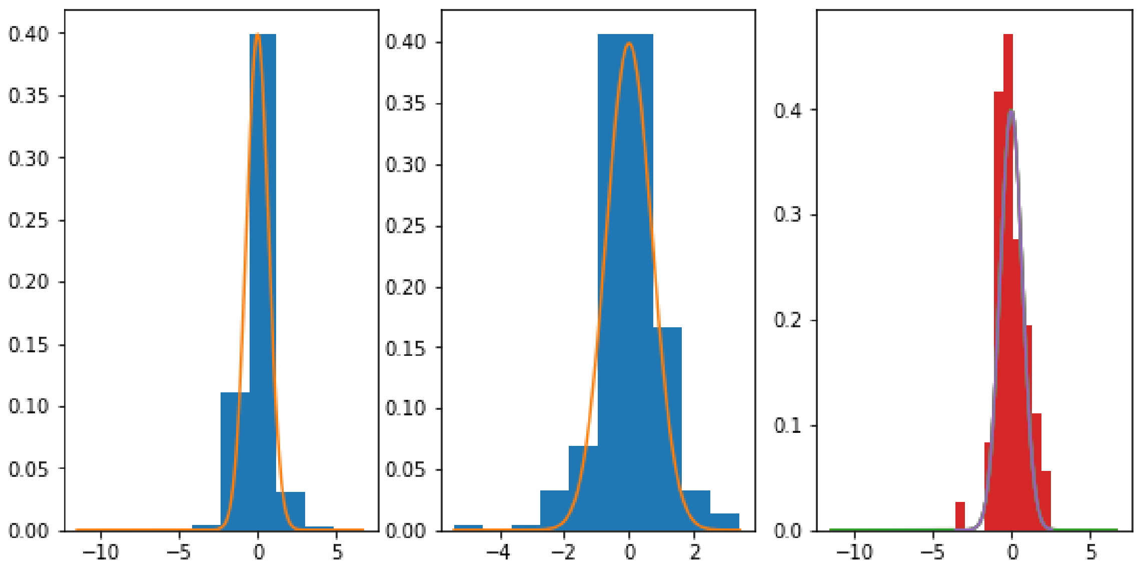
DAILY, MONTHLY & WEEKLY RETURNS for: BOMDYEING.NS.csv



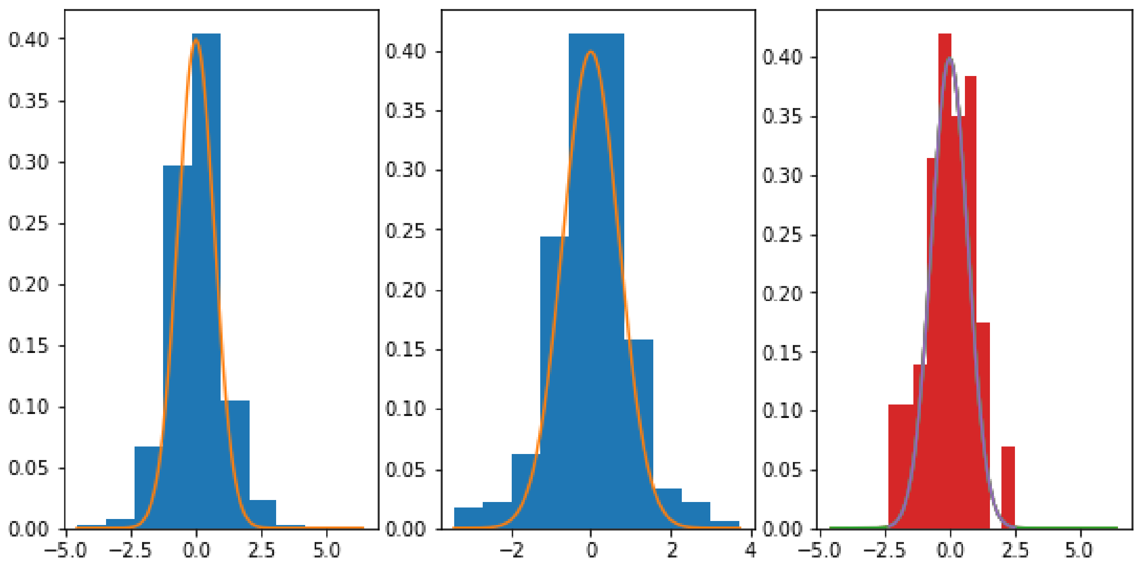
DAILY, MONTHLY & WEEKLY RETURNS for: BAJFINANCE.NS.csv



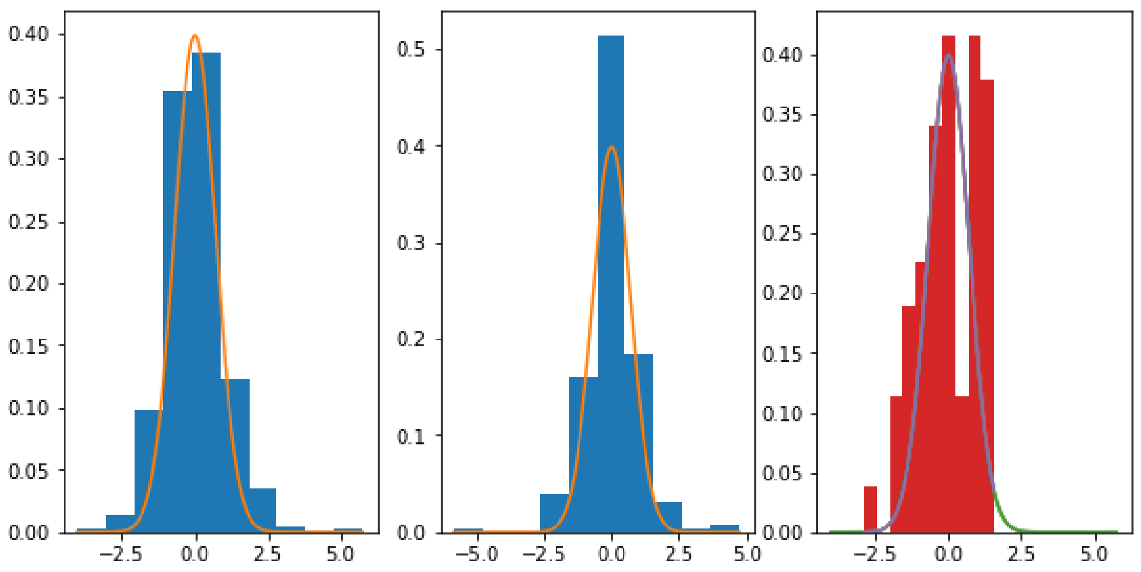
DAILY, MONTHLY & WEEKLY RETURNS for: ADANIENT.NS.csv



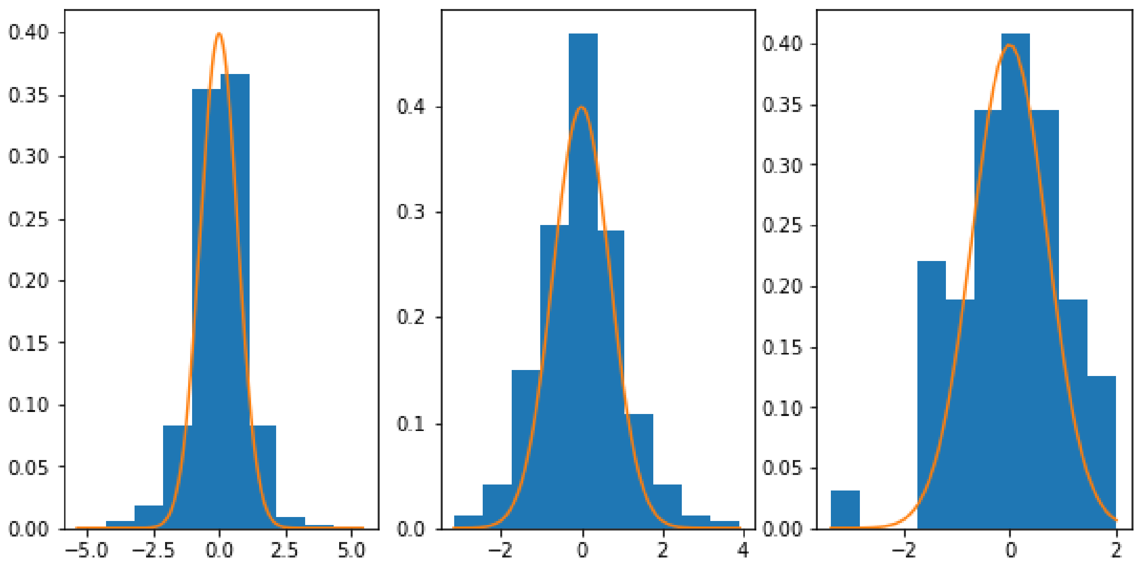
DAILY, MONTHLY & WEEKLY RETURNS for: VOLTAS.NS.csv



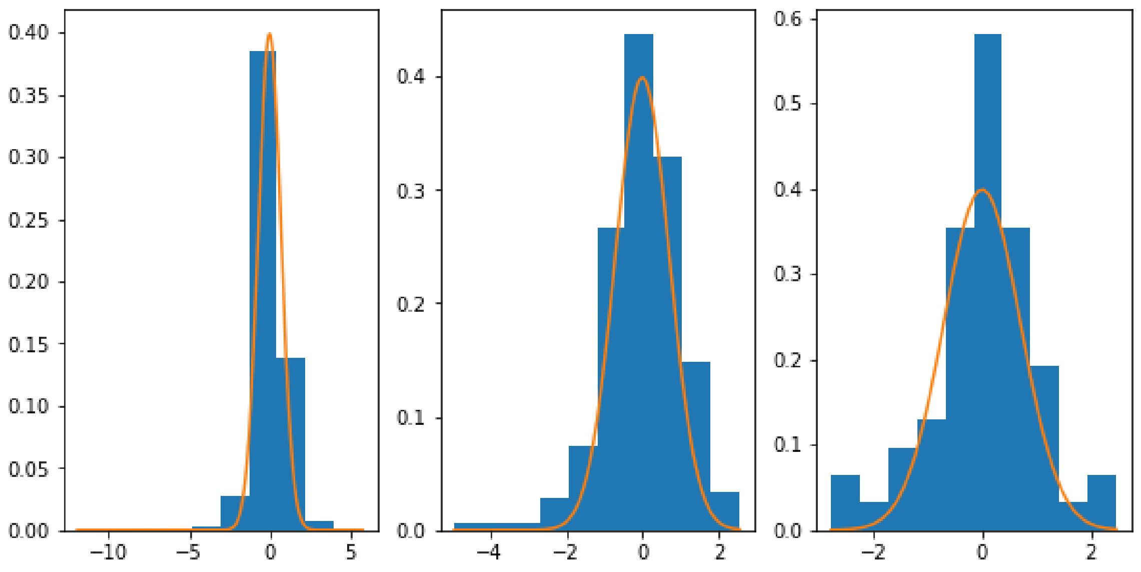
DAILY, MONTHLY & WEEKLY RETURNS for: BERGEPAINT.NS.csv



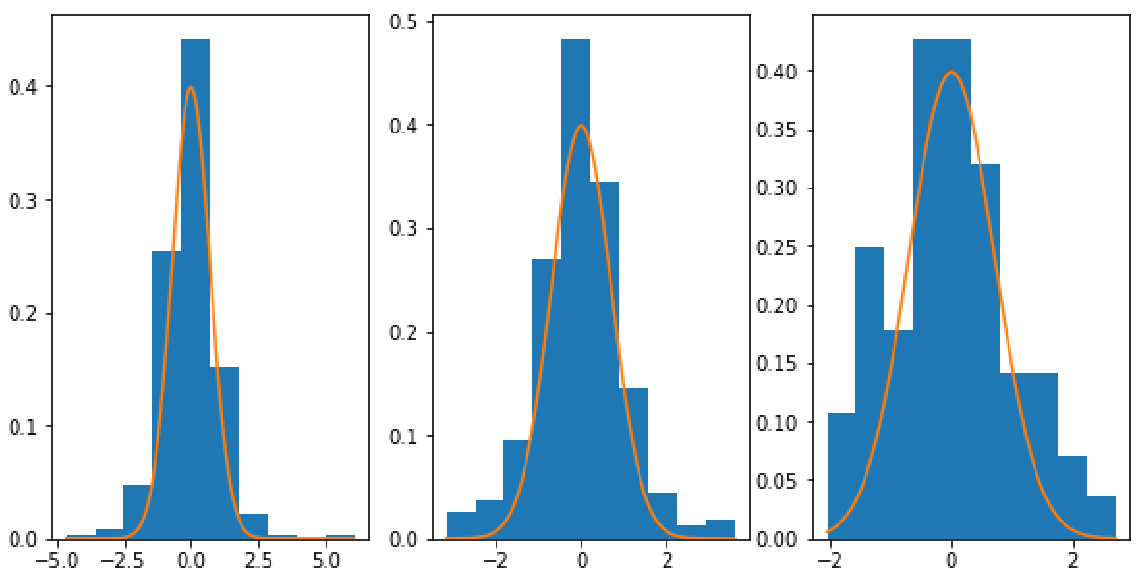
DAILY, MONTHLY & WEEKLY LOG RETURNS for: TTM.csv



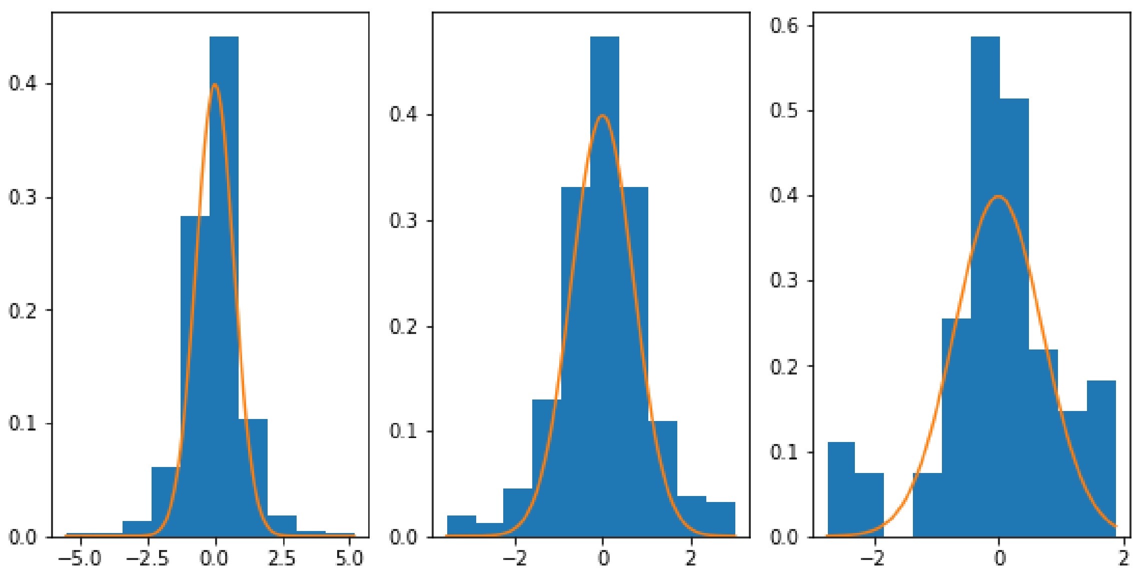
DAILY, MONTHLY & WEEKLY LOG RETURNS for: NOK.csv



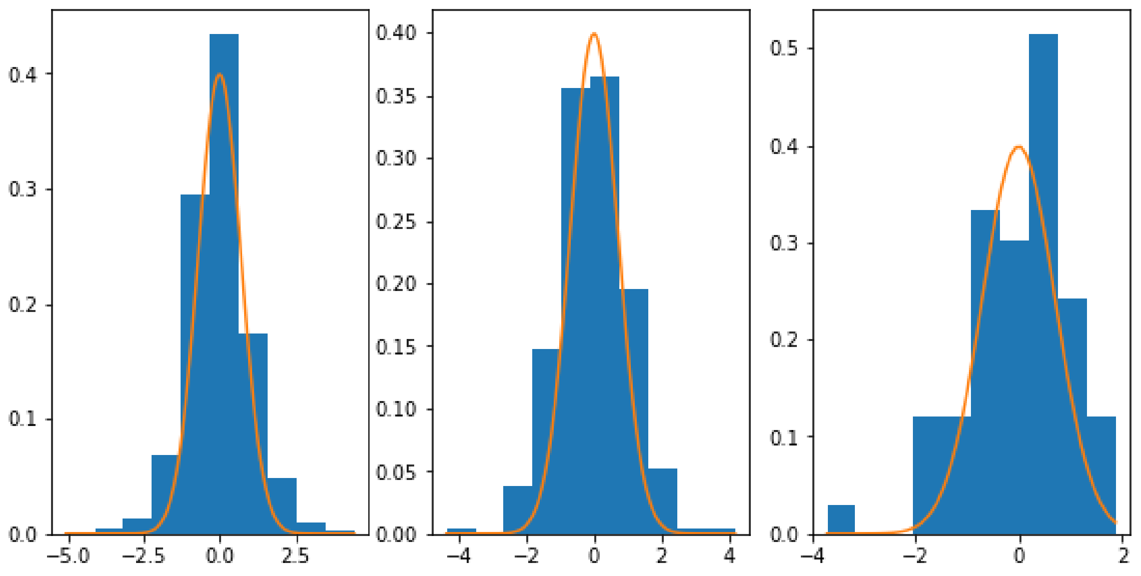
DAILY, MONTHLY & WEEKLY LOG RETURNS for: RS.csv



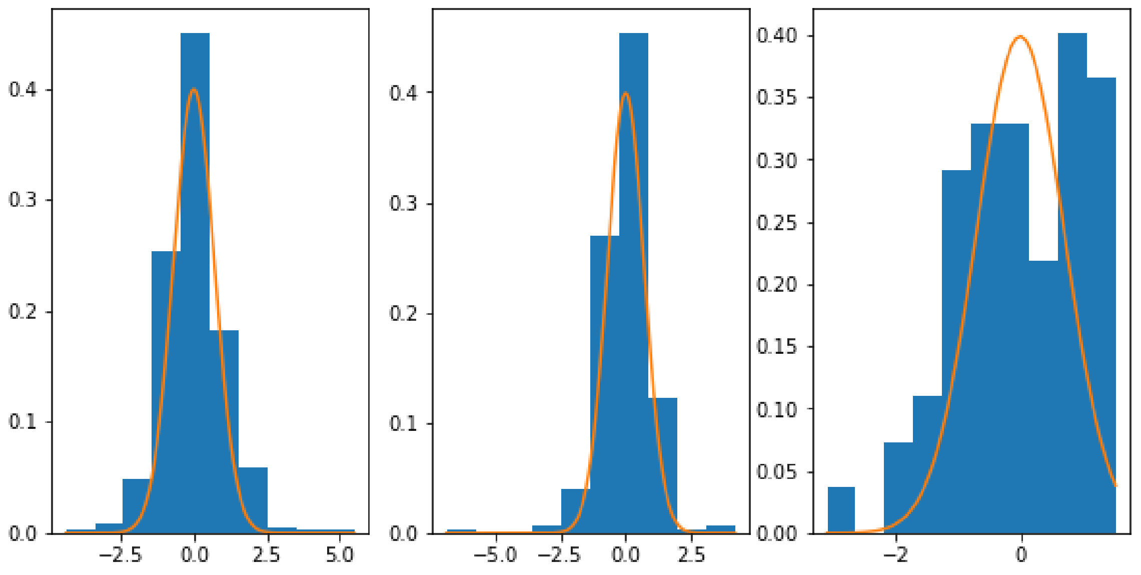
DAILY, MONTHLY & WEEKLY LOG RETURNS for: AAPL.csv



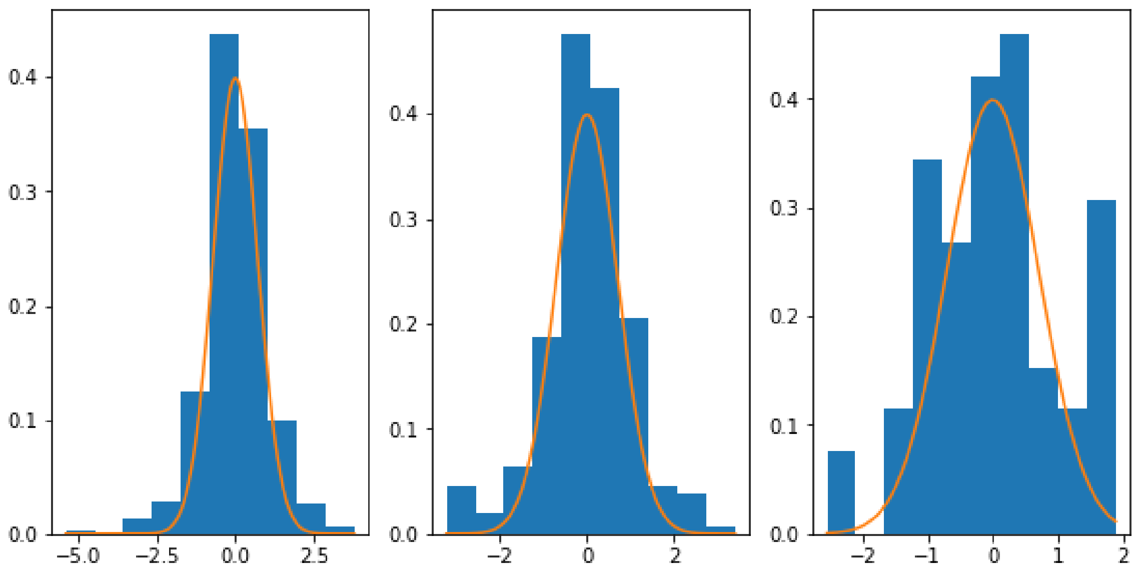
DAILY, MONTHLY & WEEKLY LOG RETURNS for: ASIANPAINT.BO.csv



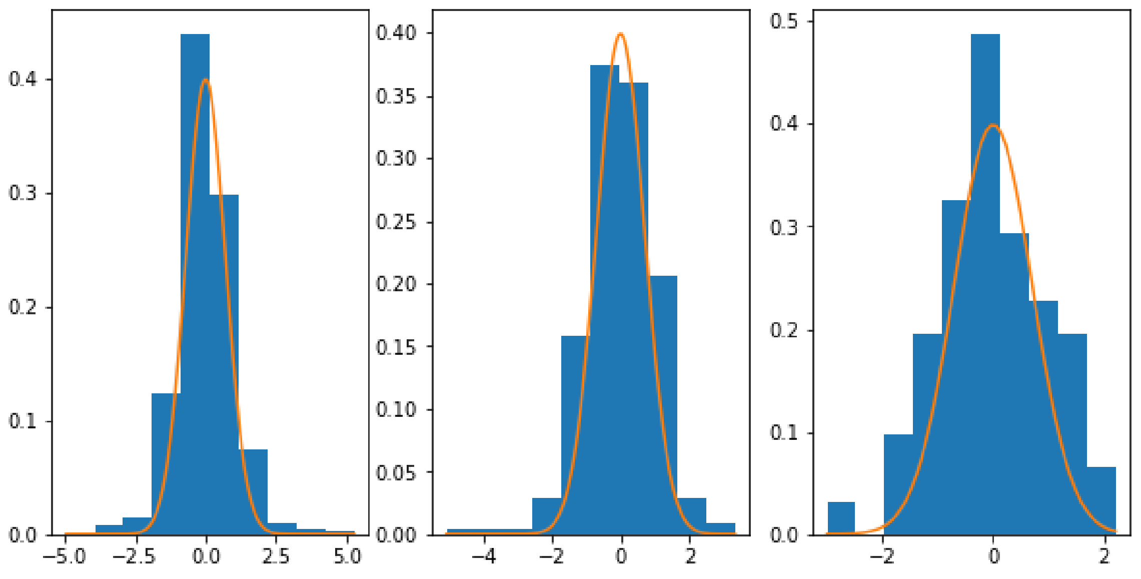
DAILY, MONTHLY & WEEKLY LOG RETURNS for: BERGEPAINT.BO.csv



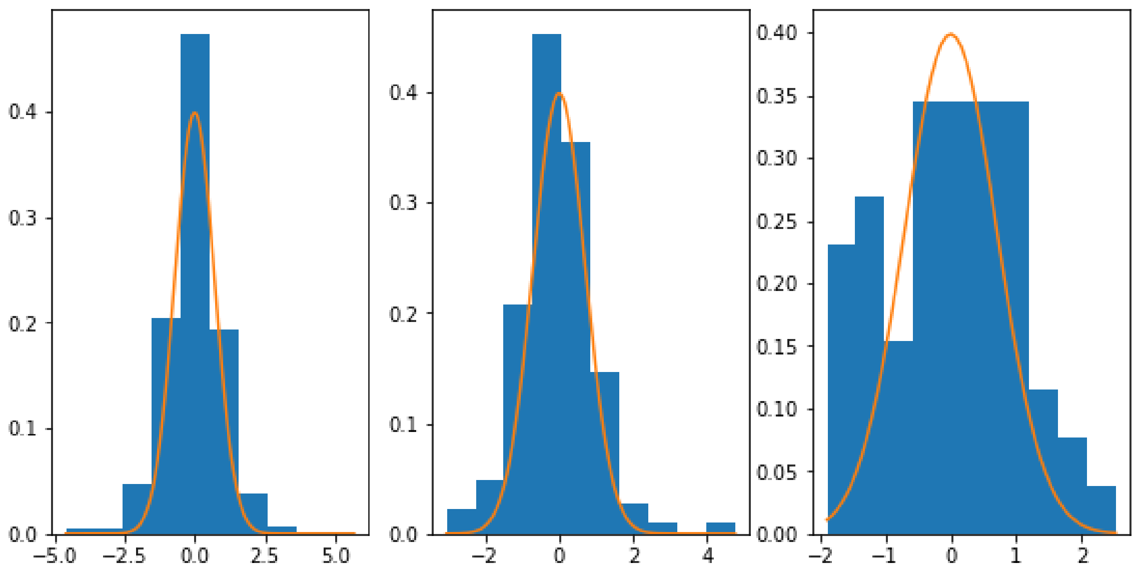
DAILY, MONTHLY & WEEKLY LOG RETURNS for: BLK.csv



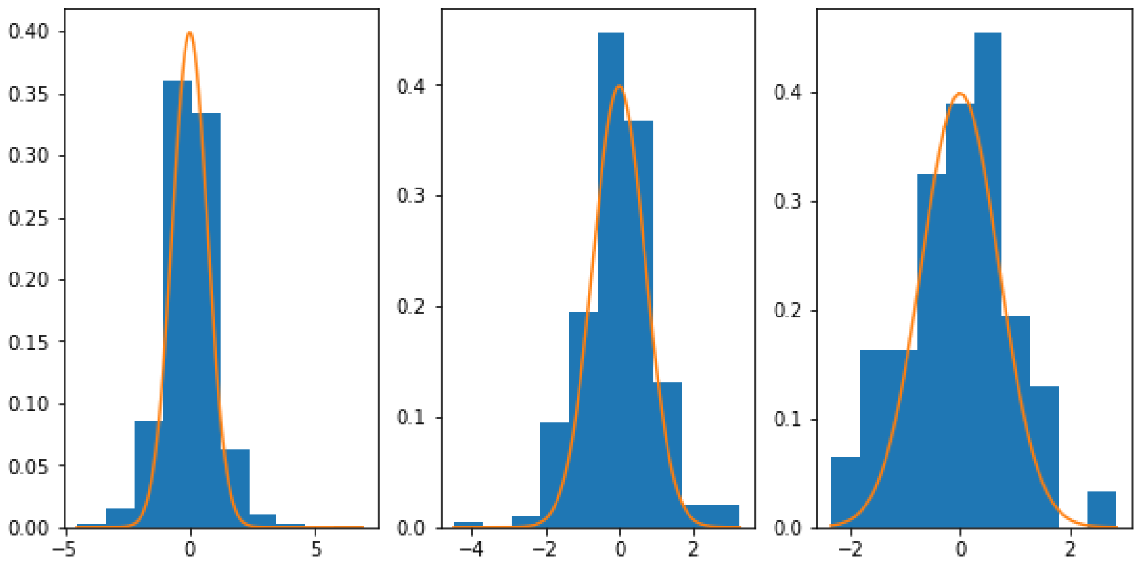
DAILY, MONTHLY & WEEKLY LOG RETURNS for: HAVELLS.BO.csv



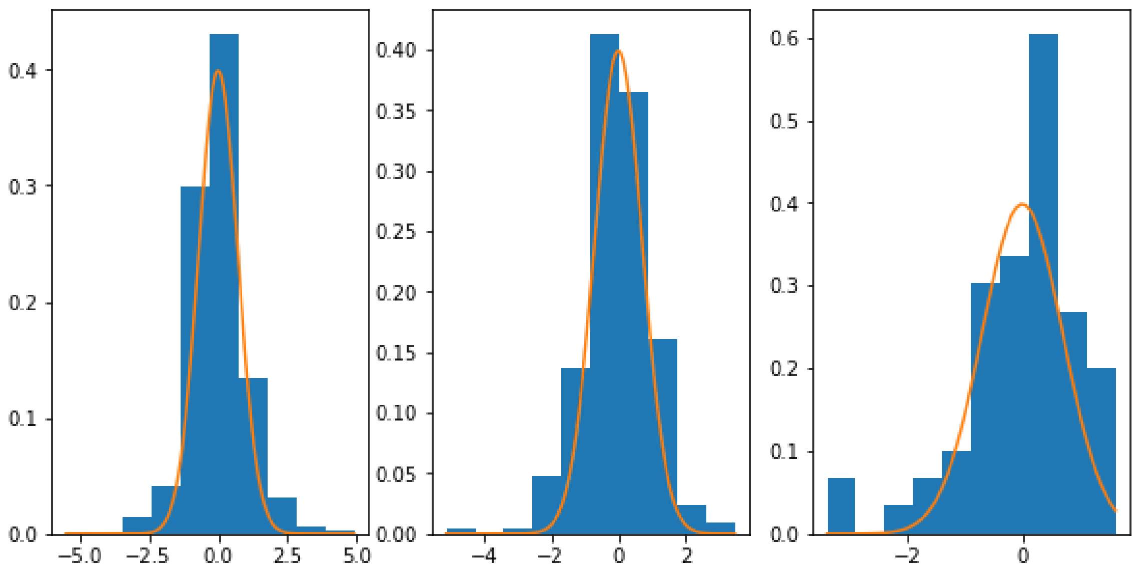
DAILY, MONTHLY & WEEKLY LOG RETURNS for: HDB.csv



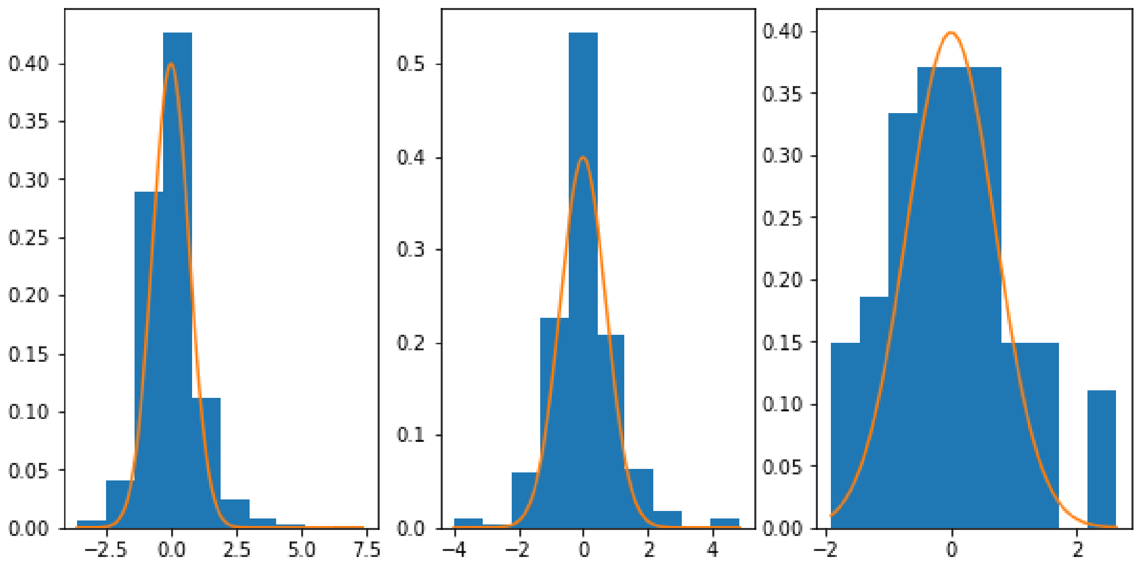
DAILY, MONTHLY & WEEKLY LOG RETURNS for: JINDALSTEL.NS.csv



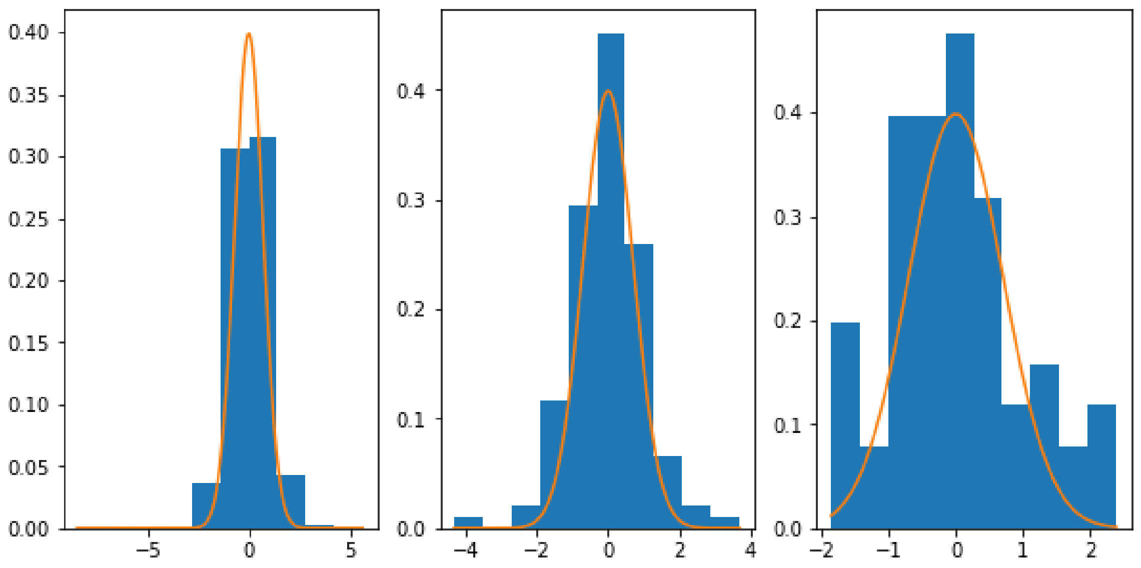
DAILY, MONTHLY & WEEKLY LOG RETURNS for: MARUTI.NS.csv



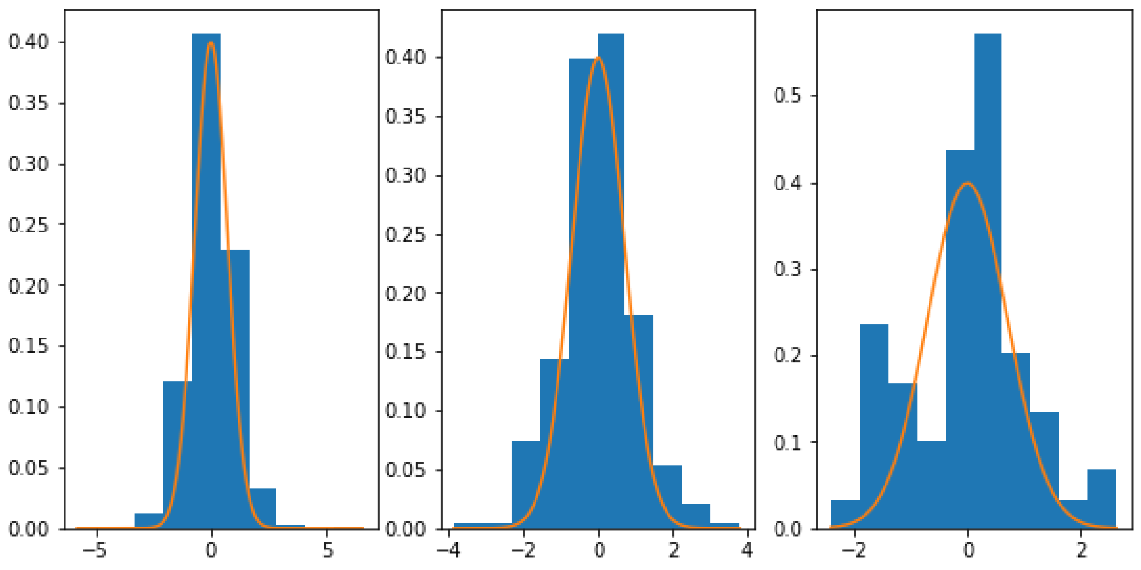
DAILY, MONTHLY & WEEKLY LOG RETURNS for: MUTHOOTFIN.NS.csv



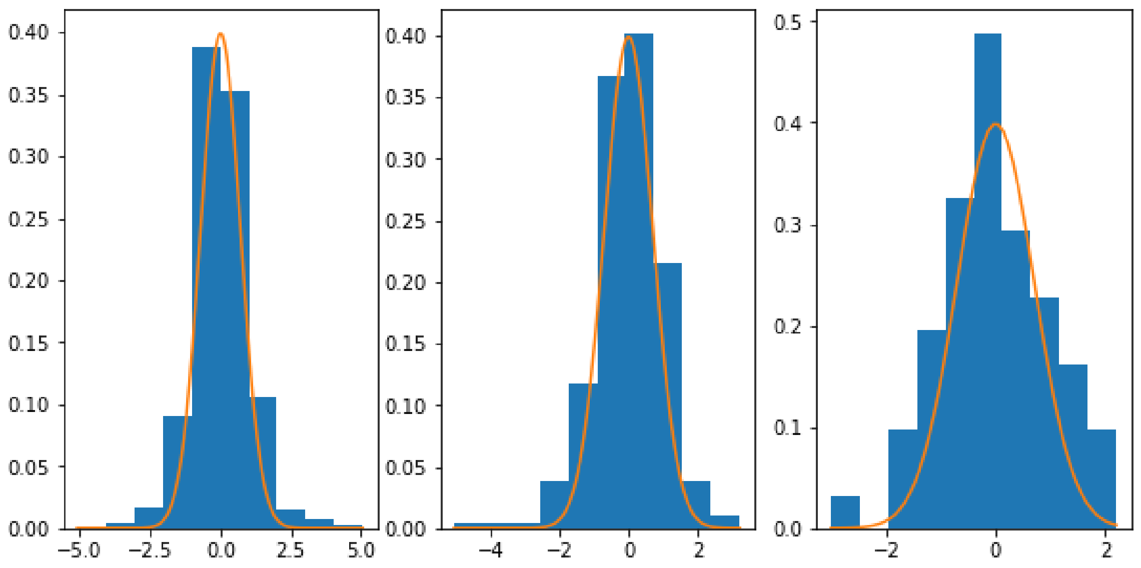
DAILY, MONTHLY & WEEKLY LOG RETURNS for: ONGC.NS.csv



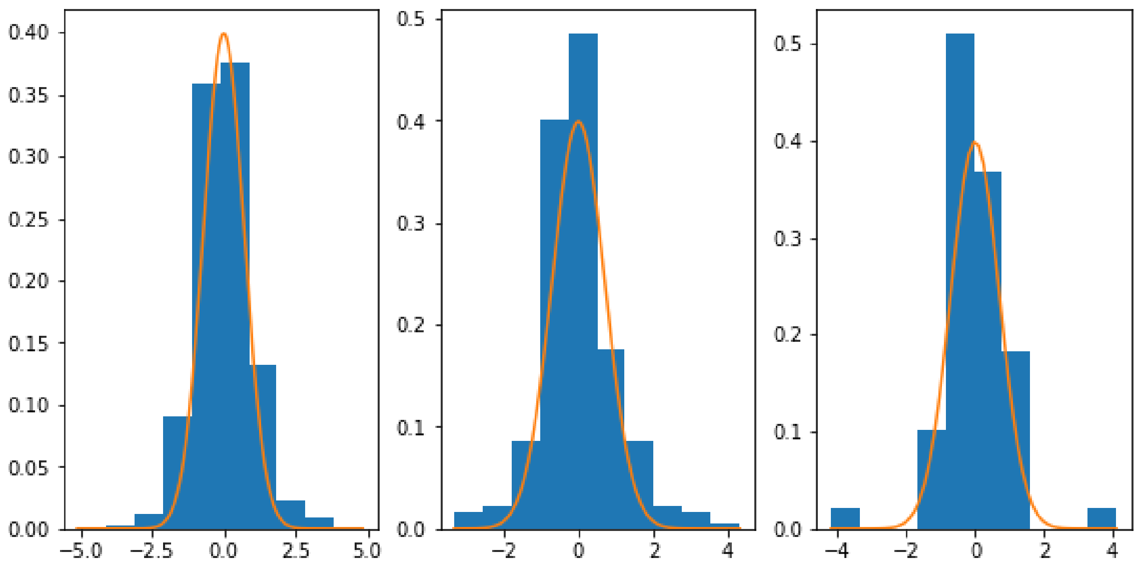
DAILY, MONTHLY & WEEKLY LOG RETURNS for: RELIANCE.NS.csv



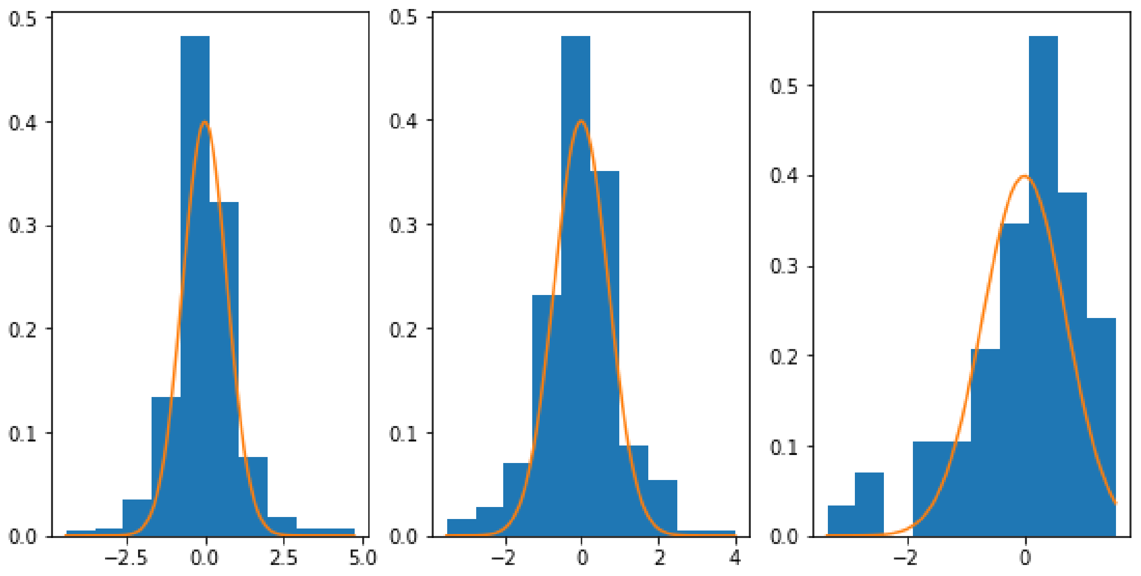
DAILY, MONTHLY & WEEKLY LOG RETURNS for: HAVELLS.NS.csv



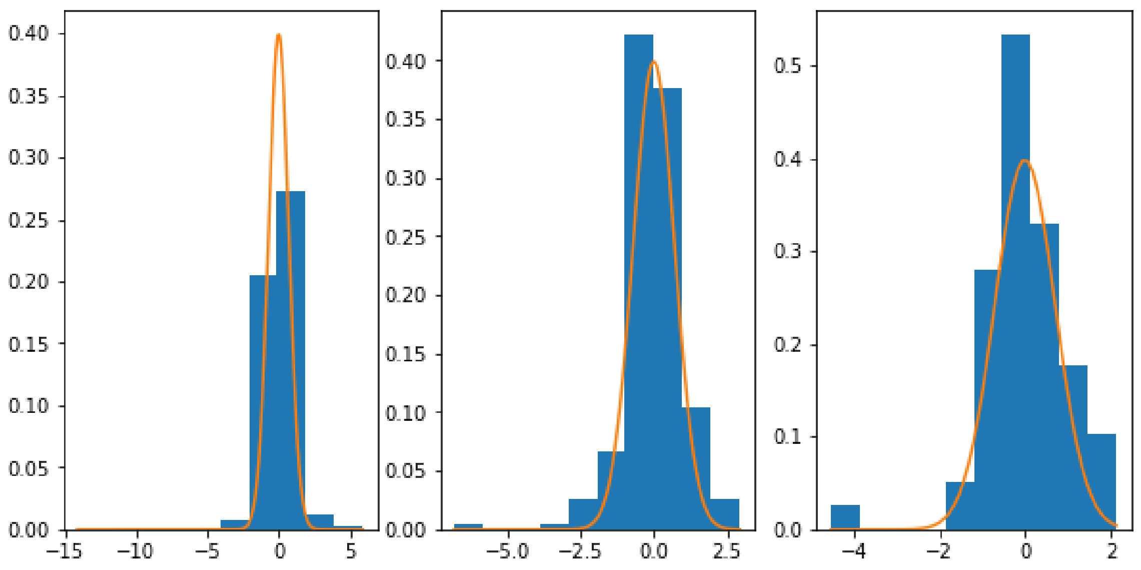
DAILY, MONTHLY & WEEKLY LOG RETURNS for: BOMDYEING.NS.csv



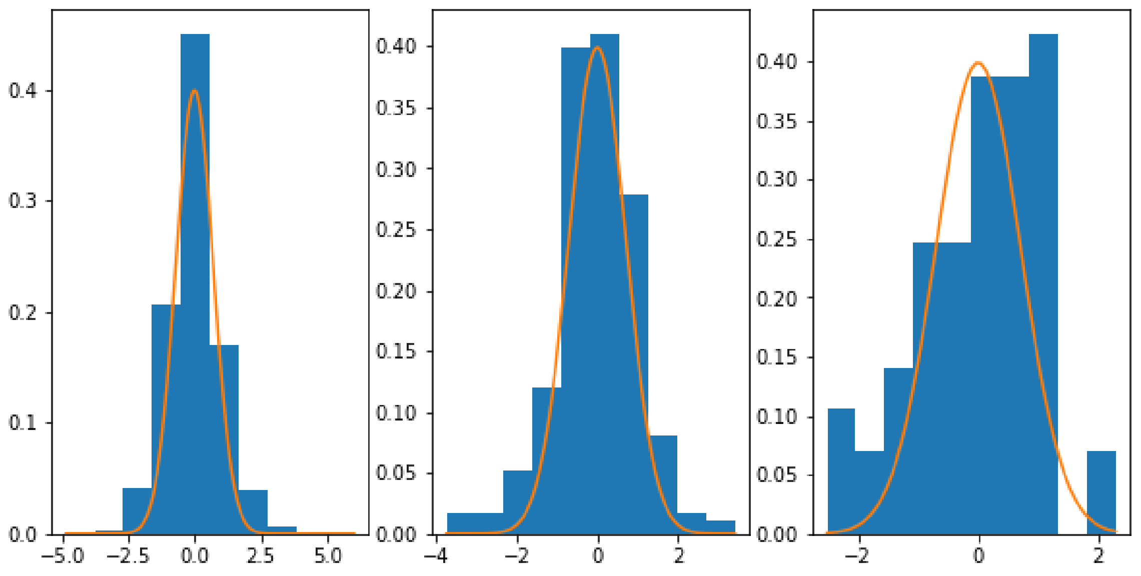
DAILY, MONTHLY & WEEKLY LOG RETURNS for: BAJFINANCE.NS.csv



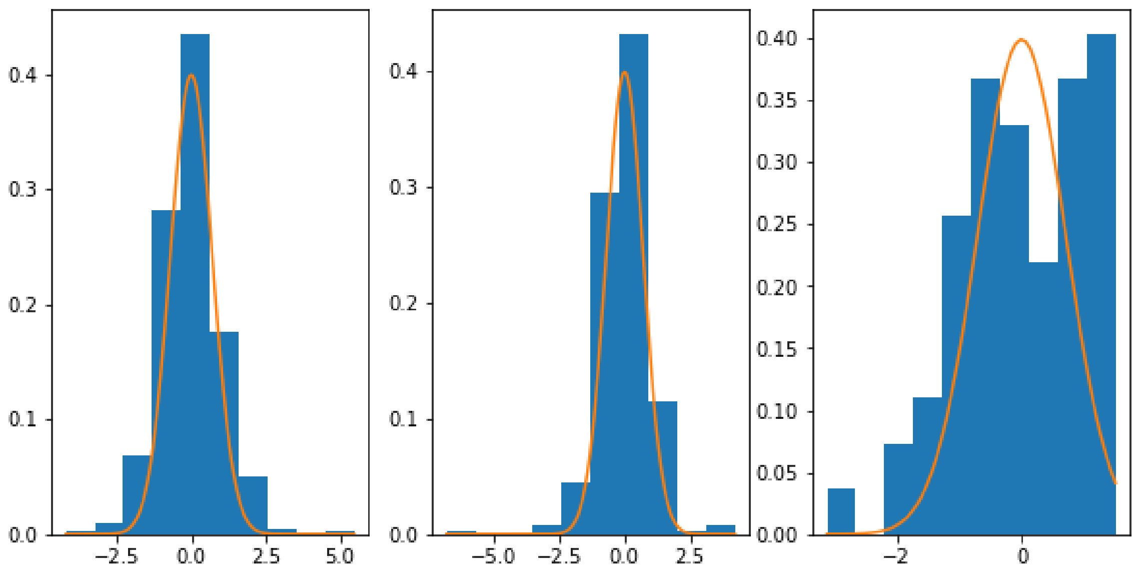
DAILY, MONTHLY & WEEKLY LOG RETURNS for: ADANIENT.NS.csv



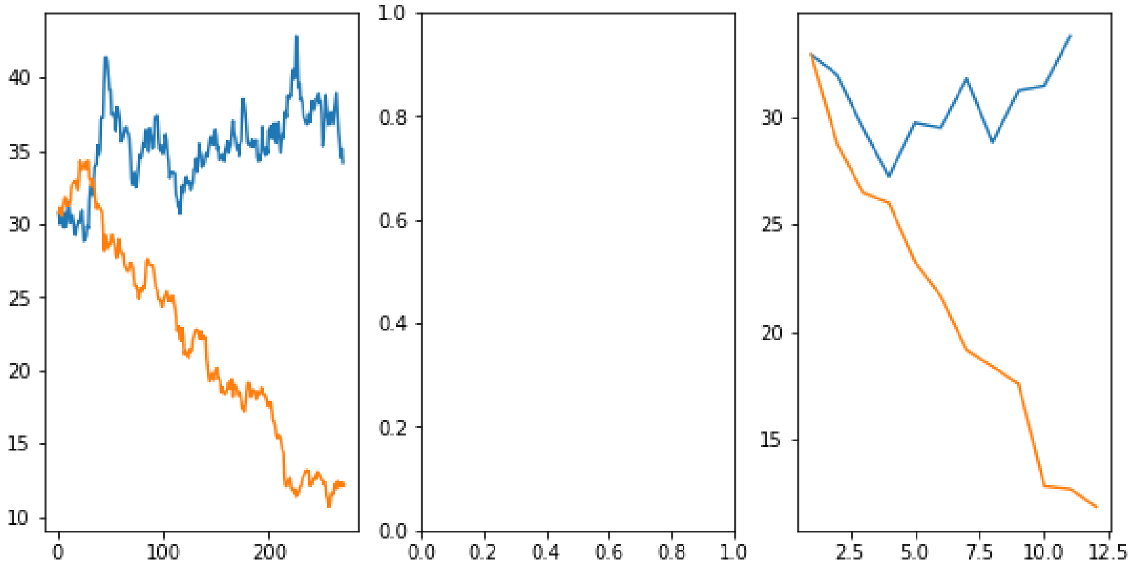
DAILY, MONTHLY & WEEKLY LOG RETURNS for: VOLTAS.NS.csv



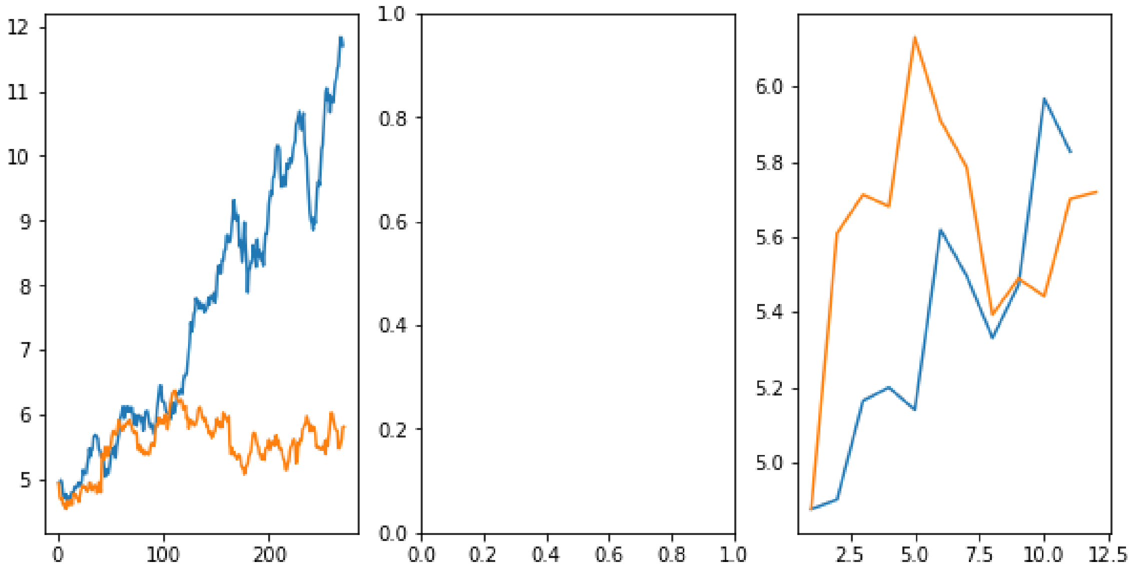
DAILY, MONTHLY & WEEKLY LOG RETURNS for: BERGEPAINT.NS.csv



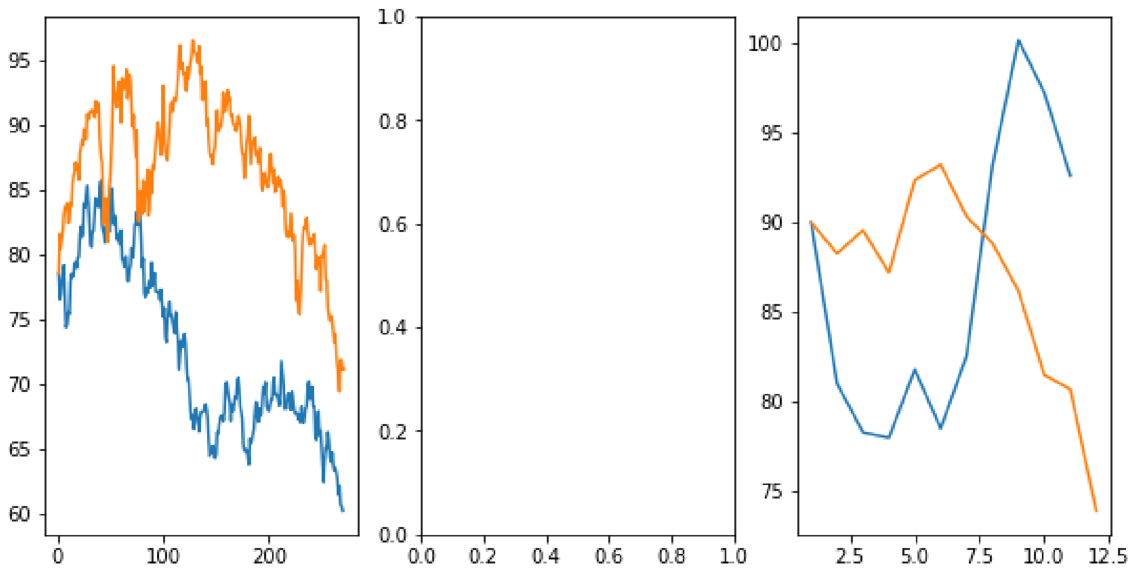
RED IS ACTUAL STOCK PRICE AND BLUE IS PREDICTED STOCK PRICE
 DAILY, MONTHLY & WEEKLY PREDICTED PRICES AND ACTUAL PRICES for: TT
M.csv



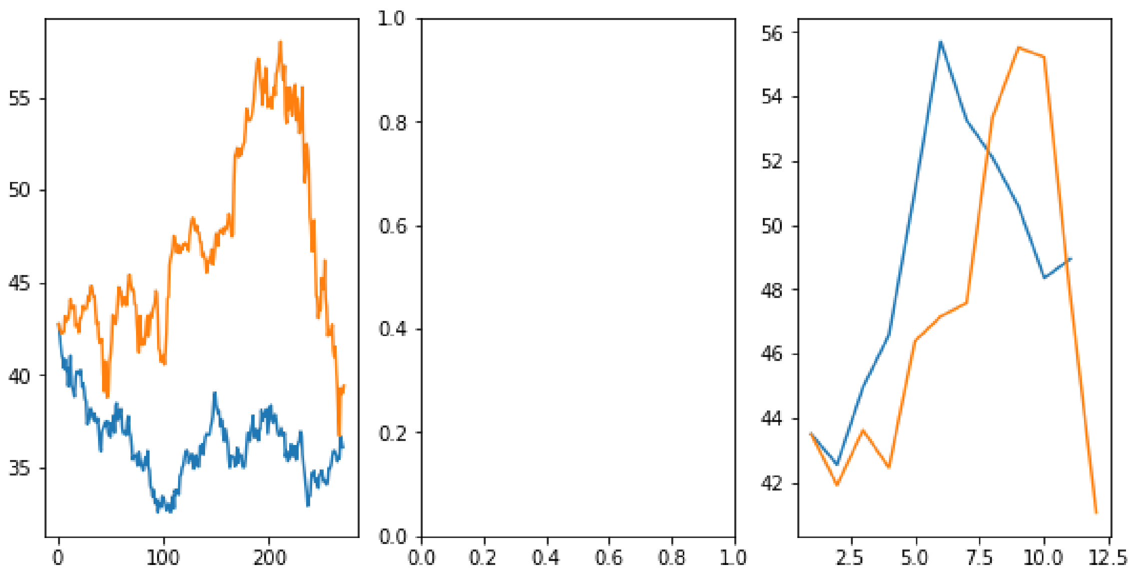
DAILY, MONTHLY & WEEKLY PREDICTED PRICES AND ACTUAL PRICES for: NO
K.csv



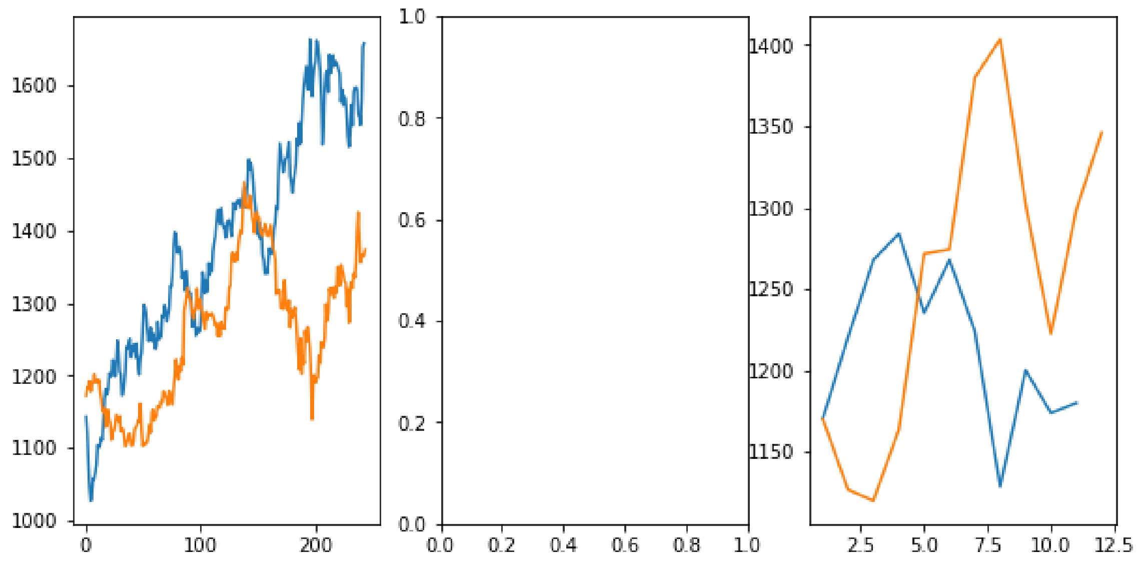
DAILY, MONTHLY & WEEKLY PREDICTED PRICES AND ACTUAL PRICES for: RS.
csv



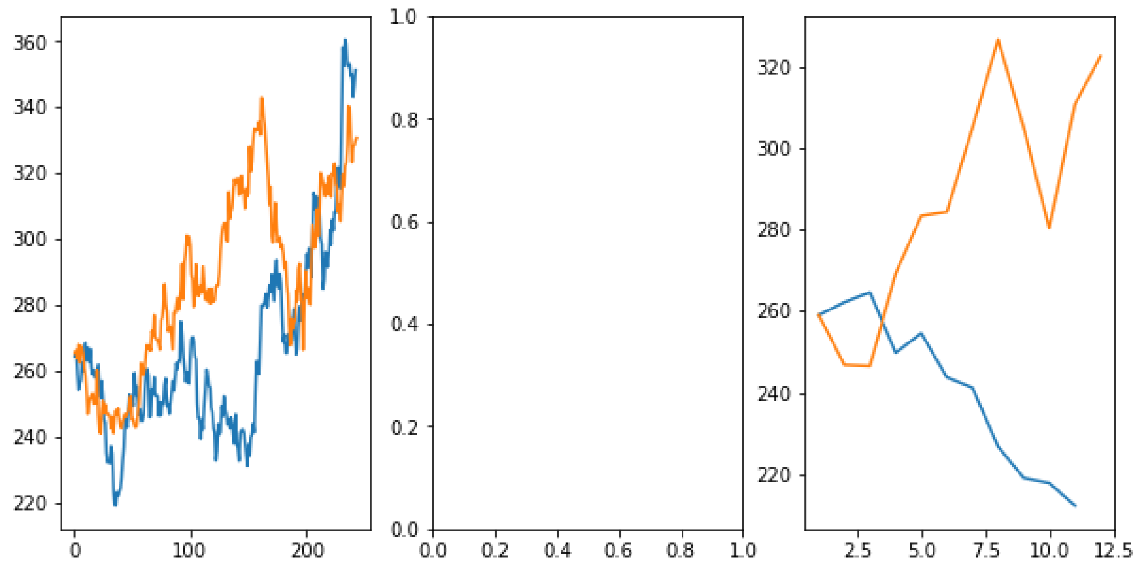
DAILY, MONTHLY & WEEKLY PREDICTED PRICES AND ACTUAL PRICES for: AAP
L.csv



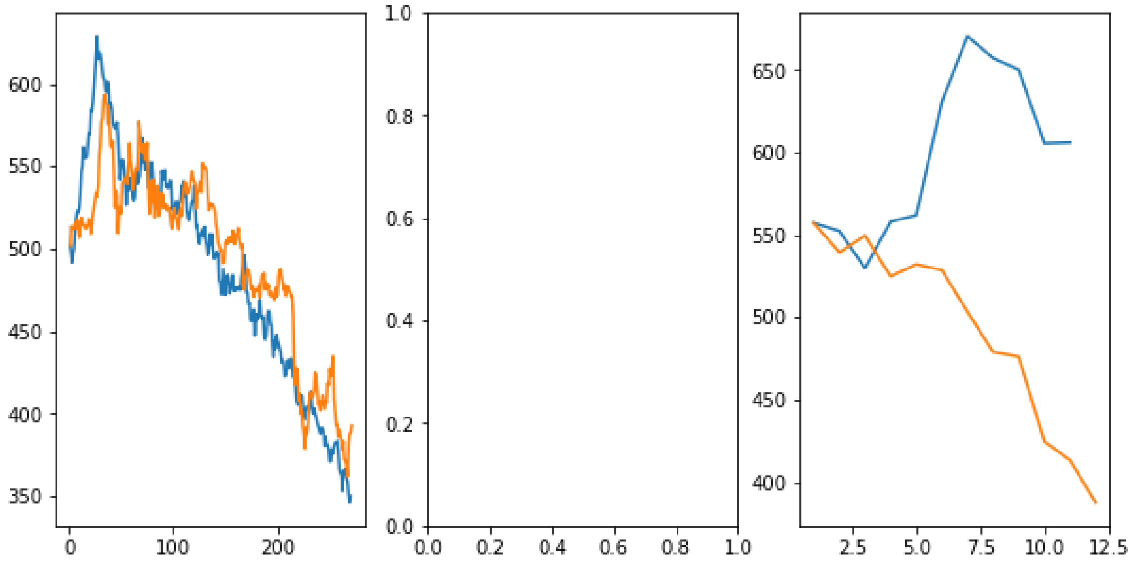
DAILY, MONTHLY & WEEKLY PREDICTED PRICES AND ACTUAL PRICES for: ASI
ANPAINT.BO.csv



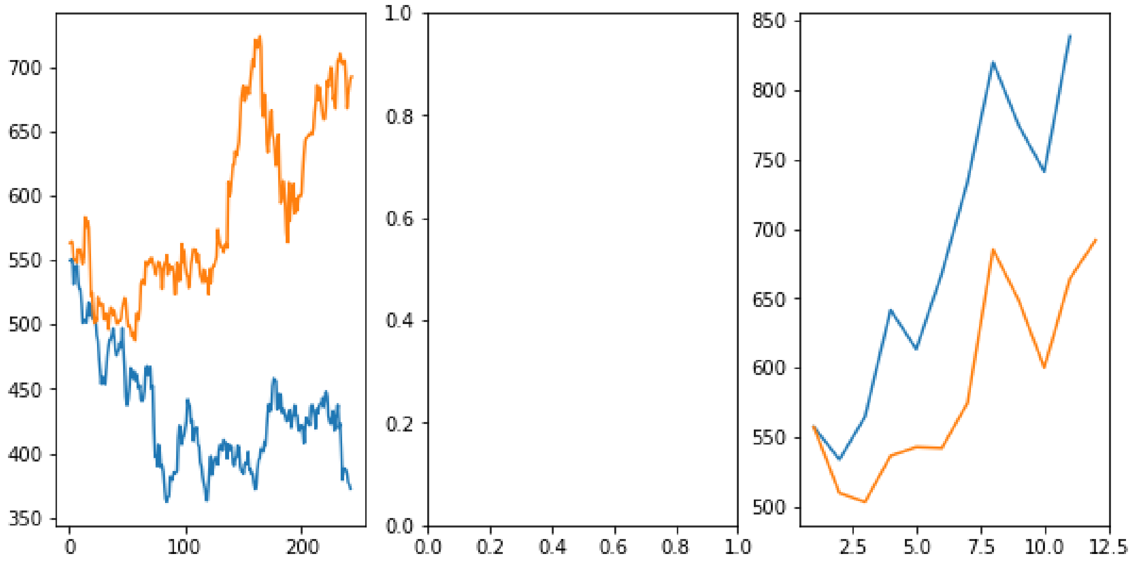
```
 DAILY, MONTHLY & WEEKLY PREDICTED PRICES AND ACTUAL PRICES for: BER
GEPAINT.BO.csv
```



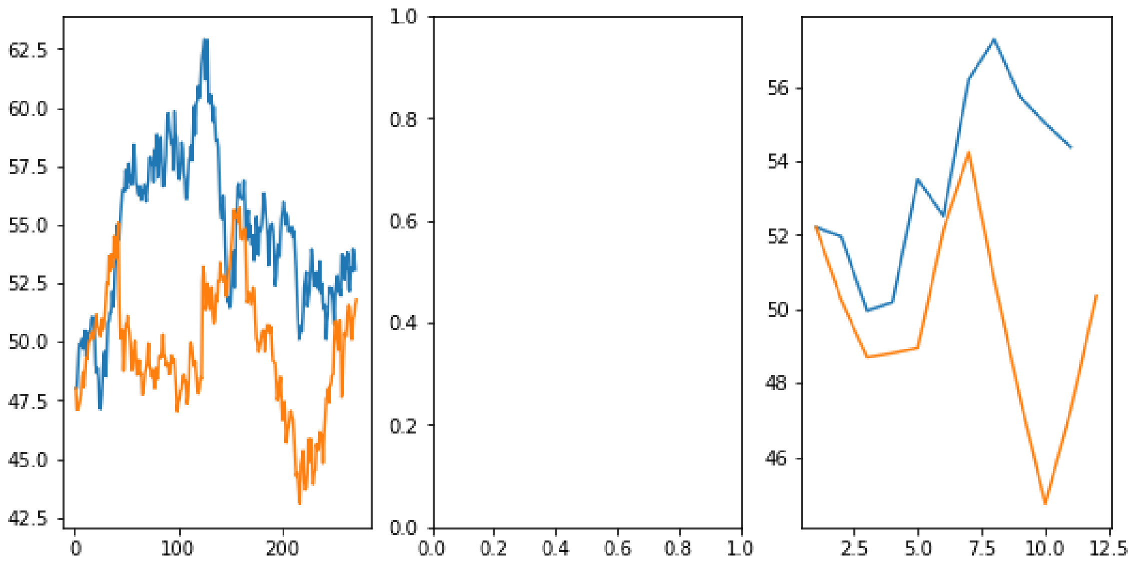
```
 DAILY, MONTHLY & WEEKLY PREDICTED PRICES AND ACTUAL PRICES for: BL
K.csv
```



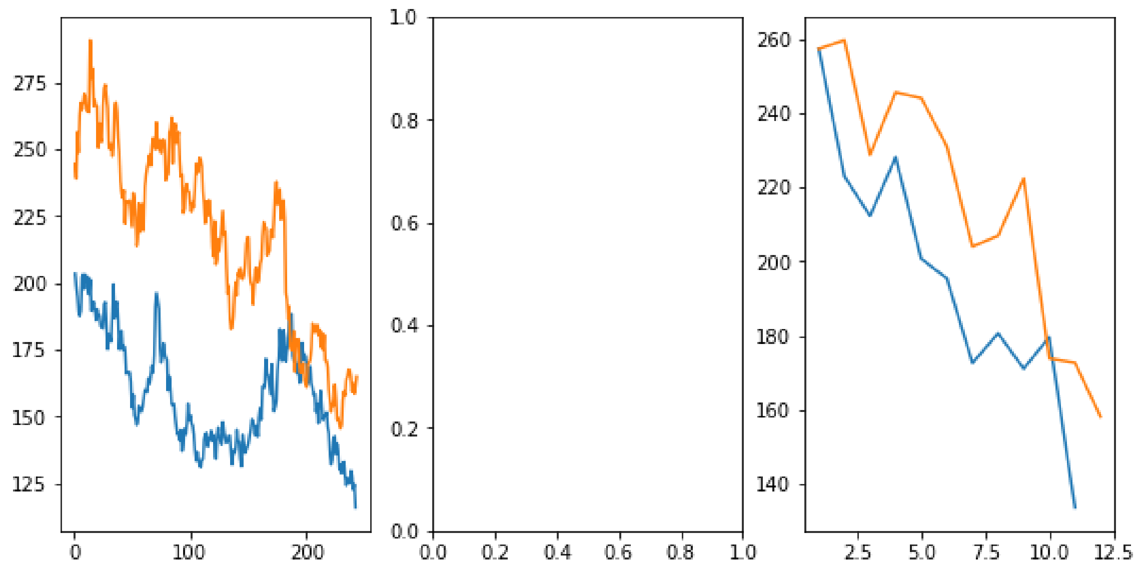
DAILY, MONTHLY & WEEKLY PREDICTED PRICES AND ACTUAL PRICES for: HAV
ELLS.BO.csv



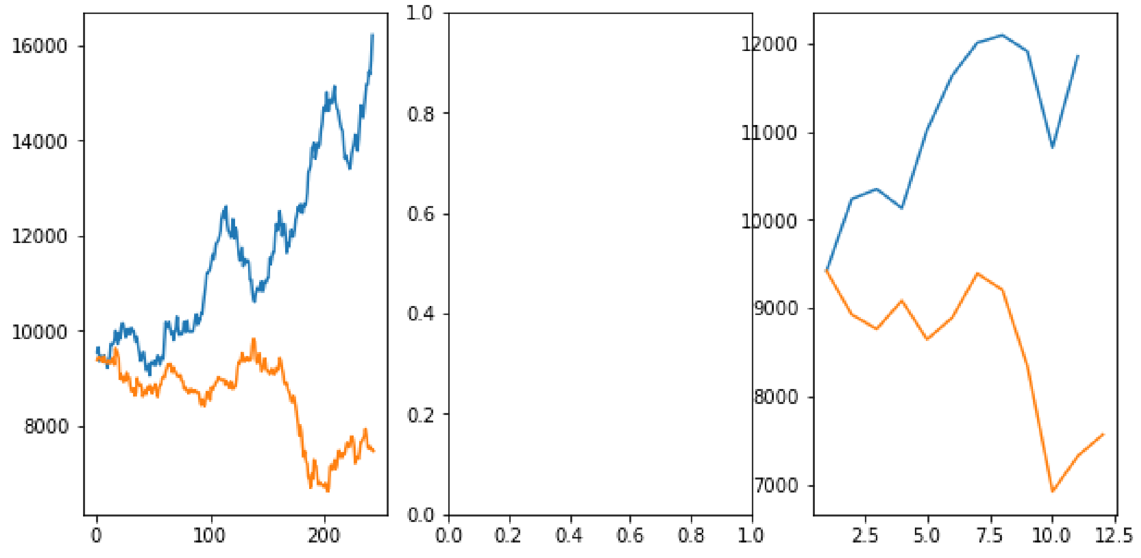
DAILY, MONTHLY & WEEKLY PREDICTED PRICES AND ACTUAL PRICES for: HD
B.csv



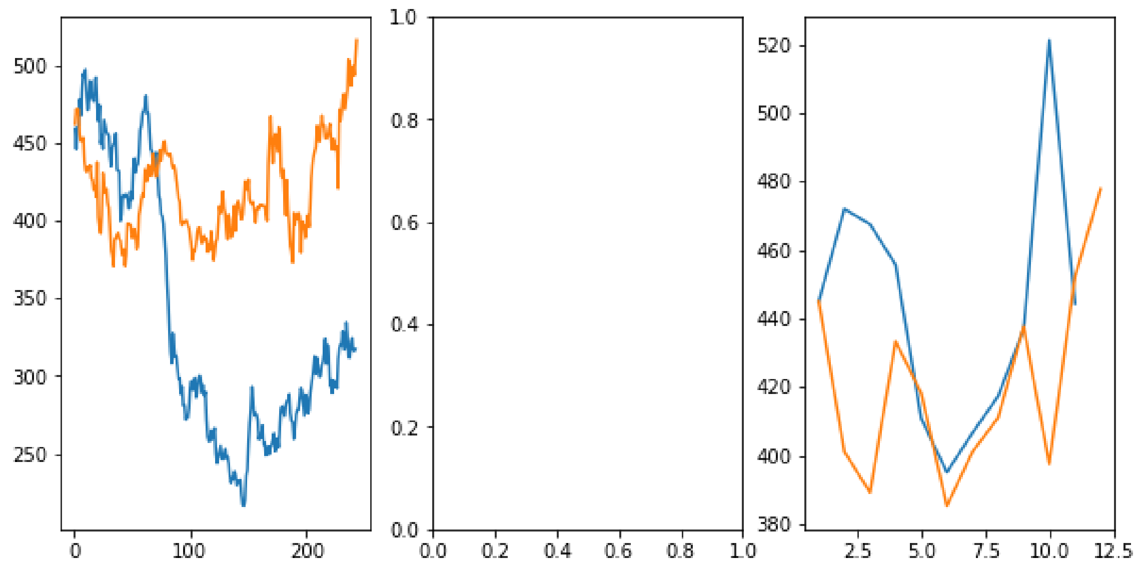
```
 DAILY, MONTHLY & WEEKLY PREDICTED PRICES AND ACTUAL PRICES for: JIN
DALSTEL.NS.csv
```



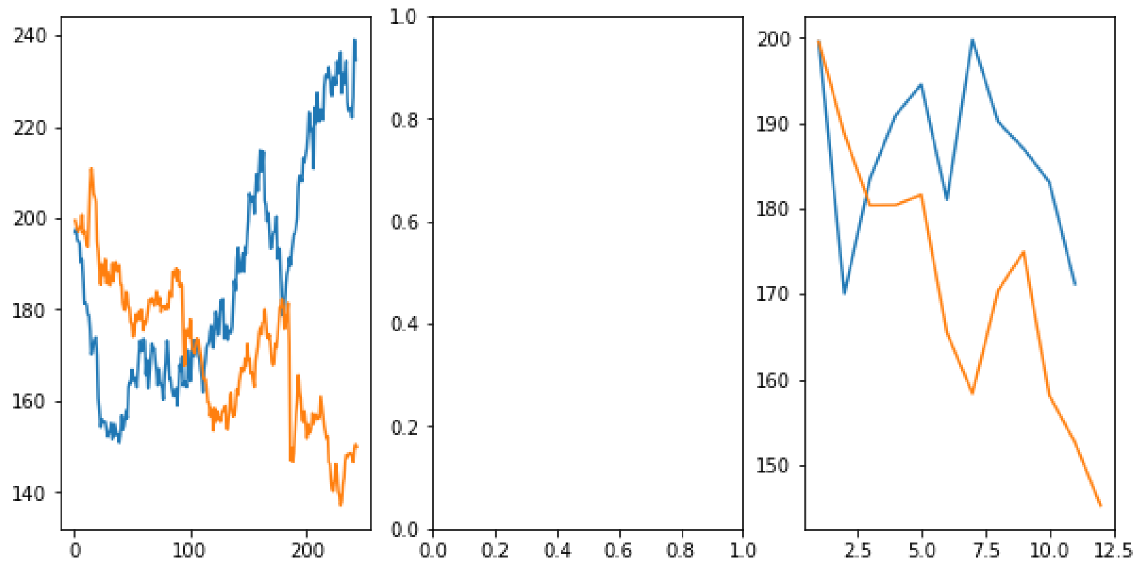
```
 DAILY, MONTHLY & WEEKLY PREDICTED PRICES AND ACTUAL PRICES for: MAR
UTI.NS.csv
```



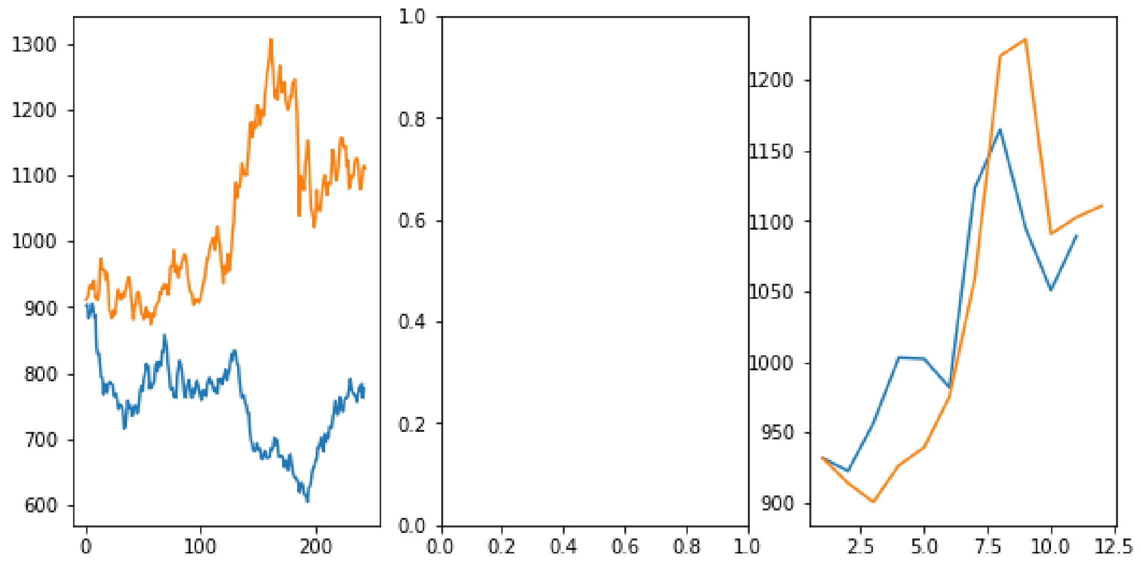
```
 DAILY, MONTHLY & WEEKLY PREDICTED PRICES AND ACTUAL PRICES for: MUT
HOOTFIN.NS.csv
```



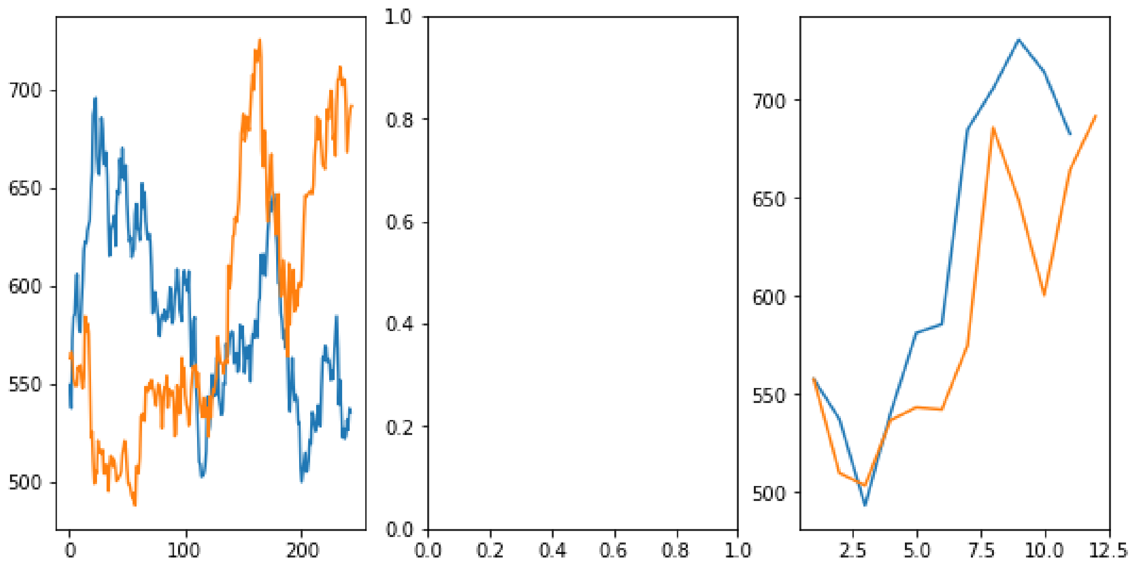
```
 DAILY, MONTHLY & WEEKLY PREDICTED PRICES AND ACTUAL PRICES for: ONG
C.NS.csv
```



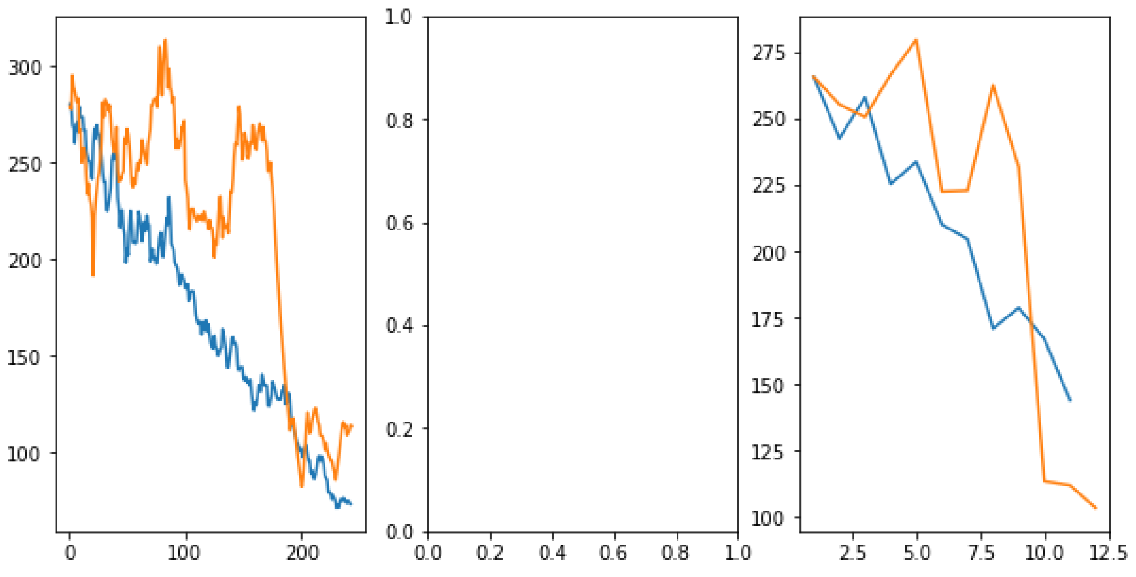
```
 DAILY, MONTHLY & WEEKLY PREDICTED PRICES AND ACTUAL PRICES for: REL
IANCE.NS.csv
```



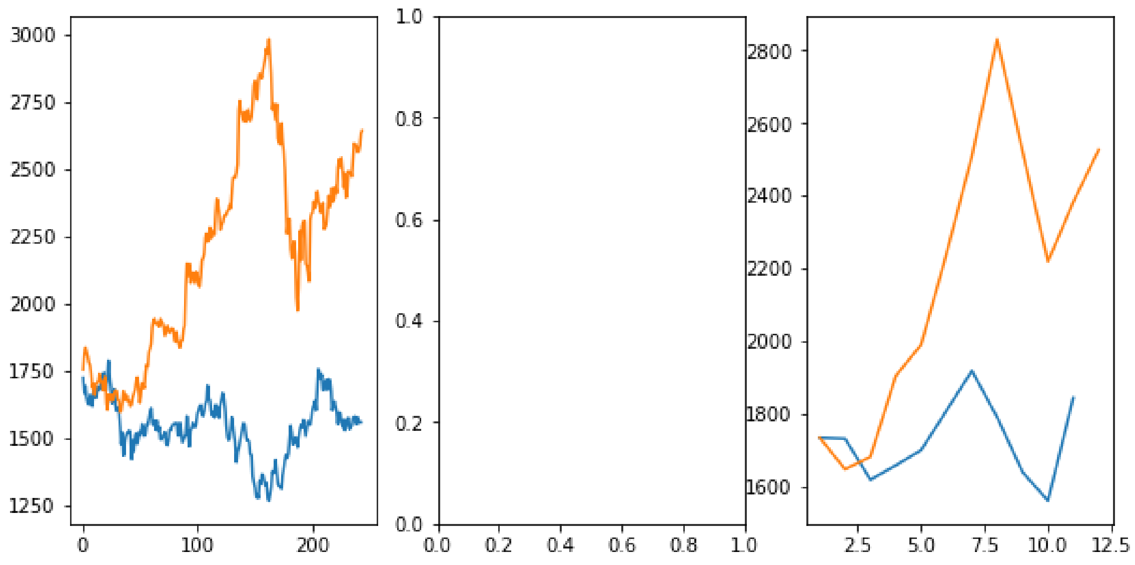
```
 DAILY, MONTHLY & WEEKLY PREDICTED PRICES AND ACTUAL PRICES for: HAV
ELLS.NS.csv
```



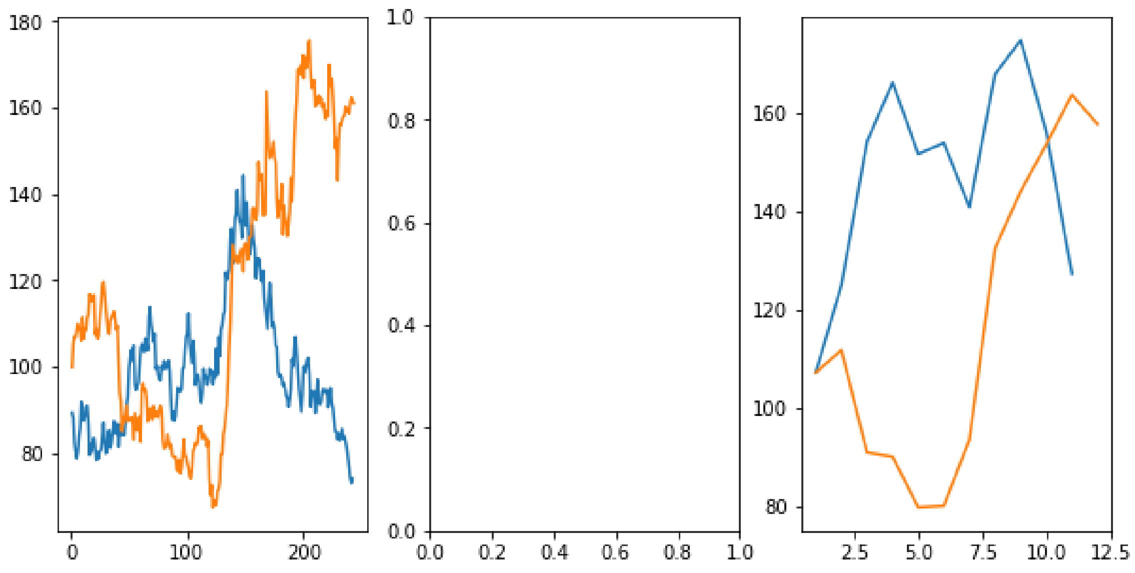
DAILY, MONTHLY & WEEKLY PREDICTED PRICES AND ACTUAL PRICES for: BOM DYEING.NS.csv



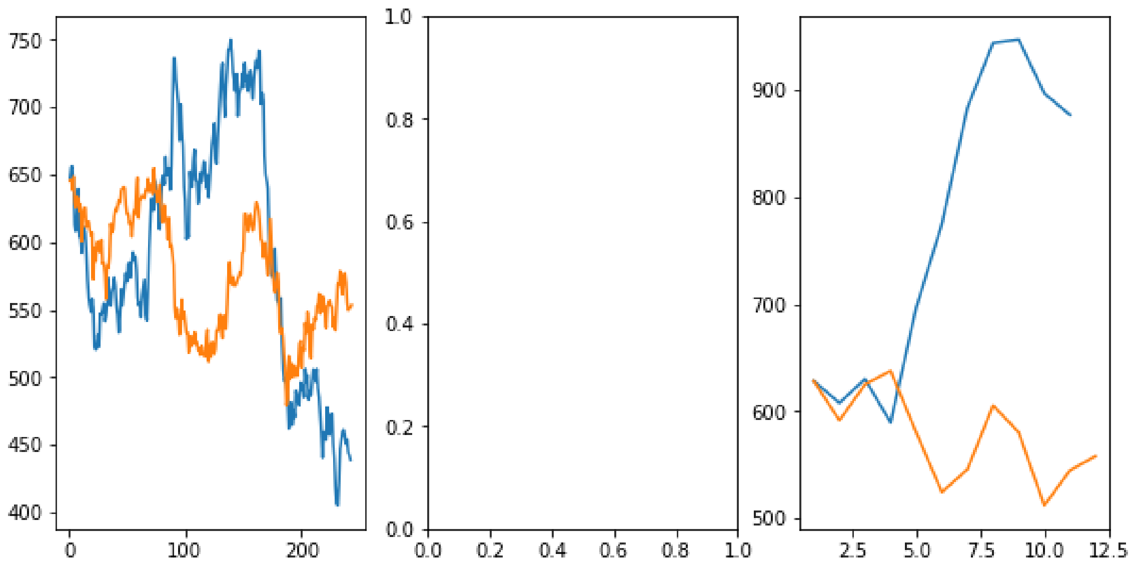
DAILY, MONTHLY & WEEKLY PREDICTED PRICES AND ACTUAL PRICES for: BAJ FINANCE.NS.csv



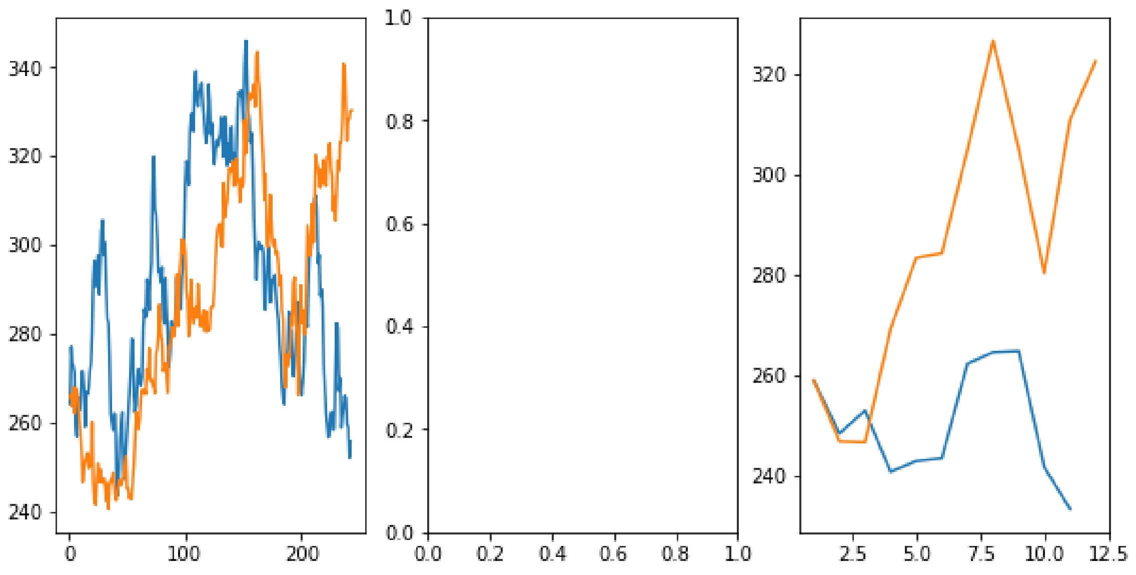
```
 DAILY, MONTHLY & WEEKLY PREDICTED PRICES AND ACTUAL PRICES for: ADA
NIENT.NS.csv
```



```
 DAILY, MONTHLY & WEEKLY PREDICTED PRICES AND ACTUAL PRICES for: VOL
TAS.NS.csv
```



DAILY, MONTHLY & WEEKLY PREDICTED PRICES AND ACTUAL PRICES for: BER
GEPAINT.NS.csv



RED IS ACTUAL STOCK PRICE & BLUE IS PREDICTED STOCK PRICE
   DAILY, MONTHLY & WEEKLY PREDICTED PRICES AND ACTUAL PRICES: TTM.c
sv