```
import pandas as pd
import xlsxwriter
import numpy as np

df1 = pd.read_csv("stocksMid.csv")
del df1['Unnamed: 0.5']
del df1['Unnamed: 0.4']
del df1['Unnamed: 0.3']
del df1['Unnamed: 0.2']
del df1['Unnamed: 0.1']
df1['Date'] = pd.to_datetime(df1['Date'])
df = df1.sort_values(by='Date')
print(df)
open = []
close = []
high = []
low = []
# df = pd.read_excel("stocksMid.xlsx")
for i, row in df.iterrows():
    openD = str(row['Open'])
    openD = openD.split('$')[1]
    openD = float(openD)
    closeD = str(row['Close/Last'])
    closeD = closeD.split('$')[1]
    closeD = float(closeD)
    highD = str(row['High'])
    highD = highD.split('$')[1]
    highD = float(highD)
    lowD = str(row['Low'])
    lowD = lowD.split('$')[1]
    lowD = float(lowD)
    open.append(openD)
    close.append(closeD)
    high.append(highD)
    low.append(lowD)
del df['Close/Last']
del df['Open']
del df['Low']
del df['High']

data = {'Close': close, 'Open': open, 'High': high, 'Low': low}

df_new = pd.DataFrame(data)
frames = [df, df_new]
df = pd.concat(frames, axis=1)
print(df)

means = []
median = []
mode = []
cardinality = []
n_median = []
n_mode = []
n_zero = []
max = []
min = []
stddev = []
n_missing = []
labels = []
```

```
for label in df.columns:
    card = 0
    sum = (df[label].isnull()).sum()
    if sum != 0:
        card = 1
    if label not in ['Unnamed: 0', 'Date']:
        labels.append(label)
        means.append(
            df[label].mean(skipna=True))
        median.append(
            df[label].median(skipna=True))
        medVal = df[label].median(skipna=True)
        mode.append(
            df[label].mode(dropna=True))
        modeVal = df[label].mode(dropna=True)
        cardinality.append(
            len(df[label].unique()) - card)
        n_median.append(
            (df[label] == medVal).sum())
        n_mode.append(
            (df[label] == modeVal[0]).sum())
        n_zero.append(
            (df[label] == 0).sum())
        max.append(
            df[label].max())
        min.append(df[label].min())
        stddev.append(
            df[label].std())
        n_missing.append(
            (df[label].isnull()).sum()
            + (df[label] == -9999).sum())

modes = []
for i in mode:
    modes.append(i)

df9 = pd.DataFrame({'Labels':
                        labels,
                    'Mean': means,
                    'Median':
                        median,
                    'n_median':
                        n_median,
                    'cardinality':
                        cardinality,
                    'n_zero':
                        n_zero,
                    'max': max,
                    'min':
                        min,
                    'stddev':
                        stddev,
                    'nmissing':
                        n_missing})

# DQR of raw dataset stored as Excel file
df9.to_excel("StockDQRraw.xlsx")

df1 = pd.read_excel("unemprate.xlsx")
```

```
indices = df[df['Date'] > '2022-03-01'].index
# print(indices)
df.drop(indices, inplace=True)
df1.to_csv("unemprates.csv")
df1 = pd.read_csv("unemprates.csv")
dateList = df['Date']
unemp = []
for date in dateList:
    split = "" + date.split('-')[0] + "-" + date.split('-')[1]
    finalDate = split + "-01"
    urate = df1[df1['DATE'] == finalDate]
    unemp.append(float(urate['UNRATE']))
# print(unemp)
data = {'unemp_rate': unemp}
df_unemp = pd.DataFrame(data)
# print(df_unemp)
frames = [df, df_unemp]
df2 = pd.concat(frames, axis=1)
df2.to_hdf("tillUnemp.hdf", key="ML")
# print(df2)

df = pd.read_hdf("tillUnemp.hdf", key='ML')
dfInf = pd.read_csv("Inflation.csv")
dfInf['DATE'] = pd.to_datetime(dfInf['DATE'])
dfInf = dfInf.sort_values(by='DATE')
indices = dfInf[dfInf['DATE'] > '2022-03-01'].index
# print(indices)
dfInf.drop(indices, inplace=True)
print(dfInf.info())
#print(dateList)
datesList = []
for date in dateList:
    datesList.append(str(date))

dfInf.to_excel("inf.xlsx")
dfInf = pd.read_excel("inf.xlsx")
dfInf["T5YIE"] = np.where(dfInf["T5YIE"] == ".", 0.0, dfInf["T5YIE"])
inf = dfInf['T5YIE']
# print(inf1)
print(inf)

frames = [df, inf]
df2 = pd.concat(frames, axis=1)
print(df2)

df2.to_hdf("tillinf.hdf", key='ML')

df = pd.read_hdf("tillinf.hdf", key='ML')
print(df)
open = []
close = []
high = []
low = []
for i, row in df.iterrows():
    openD = str(row['Open'])
    openD = openD.split('$')[1]
    openD = float(openD)
    closeD = str(row['Close/Last'])
```

```
    closeD = closeD.split('$')[1]
    closeD = float(closeD)
    highD = str(row['High'])
    highD = highD.split('$')[1]
    highD = float(highD)
    lowD = str(row['Low'])
    lowD = lowD.split('$')[1]
    lowD = float(lowD)
    open.append(openD)
    close.append(closeD)
    high.append(highD)
    low.append(lowD)
del df['Close/Last']
del df['Open']
del df['Low']
del df['High']

data = {'Close': close, 'Open': open, 'High': high, 'Low': low}

df_new = pd.DataFrame(data)
frames = [df, df_new]
df = pd.concat(frames, axis=1)
print(df['Open'])
df.to_hdf("moneyCorrected.hdf", key='ML')

df = pd.read_hdf("moneyCorrected.hdf", key='ML')
df_gdp = pd.read_excel("GDP_final.xlsx")
gdp = df_gdp['GDP']
frames = [df, gdp]
df2 = pd.concat(frames, axis=1)
print(df2)
df2.to_hdf("tillGDP.hdf", key='ML')

df = pd.read_hdf("tillGDP.hdf", key='ML')
targetPrice = []
for i, row in df.iterrows():
    if i < 20:
        continue
    targetPrice.append(row['Close'])

for i in range(20):
    targetPrice.append(0)

data = {'targetPrice': targetPrice}
dfT = pd.DataFrame(data)
frames = [df, dfT]
df2 = pd.concat(frames, axis=1)
# print(df2)

df2 = df2[df2['targetPrice'] > 0.00]
# print(df2)
profit = []
for i, rows in df2.iterrows():
    if rows['targetPrice'] > (rows['Close']):
        profit.append(1)
    else:
        profit.append(0)
prev_close = []
prev_open = []
```

```
prev_high = []
prev_low = []

prev_c = 0
prev_o = 0
prev_h = 0
prev_l = 0
for i, rows in df2.iterrows():
    if i == 0:
        prev_c = rows['Close']
        prev_o = rows['Open']
        prev_h = rows['High']
        prev_l = rows['Low']
        prev_close.append(0)
        prev_open.append(0)
        prev_high.append(0)
        prev_low.append(0)
        continue
    prev_close.append(prev_c)
    prev_open.append(prev_o)
    prev_high.append(prev_h)
    prev_low.append(prev_l)
    prev_c = rows['Close']
    prev_o = rows['Open']
    prev_h = rows['High']
    prev_l = rows['Low']

data = {'prev_close': prev_close, 'prev_open': prev_open,
        'prev_high': prev_high, 'prev_low': prev_low,
        'Profit': profit}
dfP = pd.DataFrame(data)

frames = [df2, dfP]
df = pd.concat(frames, axis=1)
df = df[df['prev_close'] > 0]
print(df)
df.to_hdf("theFinaldf.hdf", key='ML')
df.to_excel("finalDataset.xlsx")

means = []
median = []
mode = []
cardinality = []
n_median = []
n_mode = []
n_zero = []
max = []
min = []
stddev = []
n_missing = []
labels = []
for label in df.columns:
    card = 0
    sum = (df[label].isnull()).sum()
    if sum != 0:
        card = 1
    if label not in ['Unnamed: 0', 'Date', 'T5YIE']:
        labels.append(label)
        means.append(
```

```
            df[label].mean(skipna=True))
        median.append(
            df[label].median(skipna=True))
        medVal = df[label].median(skipna=True)
        mode.append(
            df[label].mode(dropna=True))
        modeVal = df[label].mode(dropna=True)
        cardinality.append(
            len(df[label].unique()) - card)
        n_median.append(
            (df[label] == medVal).sum())
        n_mode.append(
            (df[label] == modeVal[0]).sum())
        n_zero.append(
            (df[label] == 0).sum())
        max.append(
            df[label].max())
        min.append(df[label].min())
        stddev.append(
            df[label].std())
        n_missing.append(
            (df[label].isnull()).sum()
            + (df[label] == -9999).sum())

modes = []
for i in mode:
    modes.append(i)
# print(mode)
df9 = pd.DataFrame({'Labels':
                        labels,
                    'Mean': means,
                    'Median':
                        median,
                    'n_median':
                        n_median,
                    'cardinality':
                        cardinality,
                    'n_zero':
                        n_zero,
                    'max': max,
                    'min':
                        min,
                    'stddev':
                        stddev,
                    'nmissing':
                        n_missing})

# DQR of preprocessed dataset stored as Excel file
df9.to_excel("StockDQRFinal.xlsx")

print(df)
```