THE SHAIKH AYAZ UNIVERSITY SHIKARPUR

PAKISTAN STOCK EXCHANGE PREDICTION SYSTEM

FYP Report

SUBMISSION DATE:
10-JAN-2023

THE SHAIKH AYAZ UNIVERSITY SHIKARPUR

DEPARTMENT OF COMPUTER SICENCE

PAKISTAN STOCK EXCHANGE PREDICTION SYSTEM

SUPERVISOR:

Sir Ubedullah alias Kashif Dayo

Introduction to the Project team

(Team Leader)
MUDABIR HUSSAIN (19-BSCS-12)
mudabirnqvi110@gmail.com

(Team Member)
SADAM SHAIKH (19-BSCS-23)
mail.sadamshaikh@gamil.com

(Team Member)
GHULAM YASEEN (19-BSIT-02)
gyhakro@gmail.com

Certificate of Project Completion

This is certified that the following students of BS(CS/IT) of Batch2019

MUDABIR HUSSAIN (19-BSCS-12)

SADAM SHAIKH (19-BSCS-23)

GHULAM YASEEN (19-BSIT-02)

Have successfully completed their final year project titled

"**Pakistan Stock Exchange Prediction System**"

In the partial fulfillment for the Degree of Bachelor of Science in Computer Science/Information Technology

---

Sir Ubedullah (kashif)
The Shaikh Ayaz University

Sir Asadullah Bhatti
The Shaikh Ayaz University

Preferences

Final year projects have importance for any undergraduate students. We are considering developing the PSX prediction system for stock-shares buyers to achieve their actual goal of investing in the company having profited future.

We chose Python (Django) for back-end implementation and JavaScript (React JS) as front-end implementation. The major challenge was learning and researching from scratch in order to implement the abstraction in a correct way.

Abstract

"PSX Prediction System" basic purpose is to facilitate the stock-shares buyers to achieve their actual goal of investing in the company having profited future. The primary goal is tp provide the future rates of the stock-shares and show the growth of the company either in the profited side or the negative way, so the interested individual can choose the company to be invested that is expected brighter predictions.

Acknowledgement

First of all , we thank Almighty Allah who praises us with the ability to think, work and deliver what we are assigned to do. Secondly, we must be grateful to our head of department Sir Assadullah Bhatti, supervisor Sir Ubedullah alias Kashif Dayo for accepting us and our project as Final Year Project, and because of their guidance and motivation we were able to complete the project in the given timeline. We have worked hard enough to present something on the table to our supervisor and university so that it can be helpful for us and for the people we decided to dedicate our project to. We also want to thank the faculty of Computer Science department for all the guidance and motivation.

Table of Contents

Introduction:

As every minute currency has been losing its worth, so saving money without investing is like losing worth of the cash. So this is why investing is the better option, and investing in the stock market is one of the great options.

A Stock Market / Equity Market /Share Market ۔ PSX) is the aggregation of buyers and sellers of stocks, which represents ownership claims on businesses that means percentage in profit of the company, sometimes part of company's lose. As various Countries have their Stock Market, Pakistan also has individual Stock Market named "The Pakistan Stock Exchange PSX" with tending floors in Karachi, Islamabad and Lahore and a proper website https://www.psx.com.pk? Pakistan Stock Market is the platform where the stable, profitable, registered companies are listed for investment and the investors are there to invest in companies.
Now the Actual Problem is the buyer of share/ the inverter don't know which company (listed in Pakistan stock Market) has been in profit and which one is in devolution, because the data is just in numbers, so it's hard to manually calculate and predict that which company had a great past and a bright future.
Our System calculates the previous data, preview the present worth and predict the future of the companies, also suggests to invest in the companies to have a bright profitable future.

Pakistan Stock Exchange

Pakistan Stock Exchange was established on September 18, 1947 and was formally incorporated on March 10, 1949 under the name of ‘Karachi Stock Exchange’, as a Company limited by Guarantee. In October 1970, a second stock exchange was established in Lahore to meet the stock trading needs of the provincial metropolis. In October 1989, Islamabad Stock Exchange was established to cater to the investors of the northern parts of the country. Because the three exchanges had separate management, trading interfaces, indices, and had no mutualized structure, therefore the Stock Exchanges (Corporatization, Demutualization and Integration) Act, 2012 was promulgated by the Government of Pakistan which ultimately resulted in the three exchanges

integrating their operations effective January 11, 2016 under the new name ‘Pakistan Stock Exchange Limited’ (PSX)

The objective of PSX is to provide a safe, reliable, efficient and consistent marketplace where investors can buy and sell common stock of listed companies and other securities. For over 60 years, the Exchange has facilitated capital formation, serving a wide spectrum of participants, including individual and institutional investors, the trading community and listed companies

Inflation.

Inflation occurs due to an imbalance between demand and supply of money, changes in production and distribution cost or increase in taxes on products. When economy experiences inflation, i.e. when the price level of goods and services rises, the value of currency reduces. This means now each unit of currency buys fewer goods and services. It has its worst impact on consumers. High prices of day-to-day goods make it difficult for consumers to afford even the basic commodities in life. This leaves them with no choice but to ask for higher incomes. Hence the government tries to keep inflation under control.

Contrary to its negative effects, a moderate level of inflation characterizes a good economy. An inflation rate of 2 or 3% is beneficial for an economy as it encourages people to buy more and borrow more, because during times of lower inflation, the level of interest rate also remains low. Hence the government as well as the central bank always strive to achieve a limited level of inflation.

Inflation rate of Pakistan 9.50% in 2021

Best investment options in Pakistan to hedge against high inflation

In order to beat inflation and really create wealth, we need to invest in assets that can offer more return than ongoing inflation. Rapid price increases can have a destabilizing effect on an economy and jeopardize your hard-earned savings. The problem is most significant for households with fixed incomes and for retirees living on tight budgets.

Over time, the pressures of inflation can diminish the purchasing power of your income, leaving you scrambling to cover rising housing costs, food prices, energy bills, and medical expenses. The results can devastate to your personal financial situation.

Rental Real Estate Income

First and the best investment option in Pakistan is rental-generating real estate. This asset class has intrinsic value and provides consistent income through monthly it acts as a good inflation hedge since there will always be a demand for homes, regardless of the economic climate, and because as inflation rises, so do property values, and therefore the amount a landlord can charge for rent.

Although in Pakistan our housing sector is not really dependent on mortgage finance. However, in a troubled economy where people are not making enough money to afford a new house, the dependency on rental properties will increase.

Real estate is one of the time-honored inflation hedges. It's a tangible asset, and those tend to hold their value when inflation reigns, unlike paper assets. More specifically, as prices rise, so do property values, and so does the amount a landlord can charge for rent
So that the property earns higher rental income over time.

A good rental property in Pakistan will give you 14 to 20% returns per annum through Capital gains and rental income combined. The rental income can be used to maintain your lifestyle and the capital gains will beat inflation in the longer run. All these elements Make real estate the most valuable protection in inflationary times.

Mutual Funds

Mutual funds are described as pooled investments. When an investor buys the units in a mutual fund, the money is pooled with that of other investors whose goals are similar. A professional fund manager uses this money to buy stocks, bonds, or money market instruments that make up the fund's portfolio of investments.
In Pakistan, various mutual funds are available and handled by professional investment managers. This is especially the best option for investors who do not have the funds to

invest in the rental real estate market. You can easily open a mutual fund account with a bank and start your investment without having to invest a large amount of money upfront. Although mutual funds are subject to market risk, they have shown solid and consistent growth in the past years. Some Mutual funds have shown 12 to 14% gains consistently over the years. This means that if everything goes well, they will successfully beat inflation and keep your wealth intact.

Another benefit of mutual funds is that some of them are sharia compliant and if taking a Fixed interest is something you want to avoid, mutual funds can solve that for you.

Mutual funds could easily be the first step where you make your money multiply before you step into the big game of rental property

Real Estate

Investing in non-rental real estate is still a good investment during inflation times. Real Estate in Pakistan has generally given a capital gain of 10 to 12% per annum, which should beat the high inflation of today.

Real estate works well with inflation. This is because, as inflation rises, so do property values. This helps to keep pace with the rise in inflation. For this reason, real estate Income is one of the best ways to hedge an investment portfolio against inflation

Although the recent Government policies discourage investors to invest in plots and files or other non-rental properties. If you have knowledge of real estate and some holding power, you can effectively hedge against inflation. PKR depreciation is also easily managed by investing in real estate.

You may not see immediate results and you will require a bit of patience when working in Pakistan real estate. However, if you have invested in the right place, you will eventually beat inflation. Though, real wealth creation may not be possible anymore in non-income generating real estate.

Saving Accounts

Pakistani banks are offering one of the highest interest rates in the world right now, where you can put your money and get up to 8 to 9% returns. This would probably not make you

wealthy but it will shield you to an extent against inflation. At least you are growing poor at a slower rate than those who are just keeping cash.

If you are not using the profits and keep compounding, you will probably still be able to afford in ten years what you can today. However, you need to have another active source of income to keep your lifestyle

Commodities

Commodities are a broad category that includes grain, precious metals, electricity, oil, beef, orange juice, natural gas, etc. Commodities and inflation have a unique relationship, where commodities are an indicator of inflation to come. As the price of a commodity rises, so does the price of the products that the commodity is used to produce.

Fortunately, it's possible to broadly invest in commodities via exchange-traded funds (ETFs). The returns on commodity trading are variable and will depend on your experience Before investing in commodities, investors should know they are highly volatile and I advise investor caution in commodity trading. Because commodities depend on demand and supply factors, a slight change in supply because of geopolitical tensions or conflicts can adversely affect the prices of commodities.

In the present geo-political scenario, global inflation, I expect commodities to perform well and rate them above stocks. A good trader will effectively beat inflation and create wealth, however as it is riskier and not for everyone, I rate it at No-5.

Stocks Portfolio

Although if you are a smart investor, you can certainly make some money and beat inflation when investing in stocks.

Stocks offer substantial upside potential in the long term. In general, businesses that gain from inflation are those that require little capital (whereas businesses that are engaged in natural resources are inflation losers). Both technology and communication services are capital-light businesses, so, theoretically, they should be inflation winners

However, stocks are highly volatile in a country like Pakistan suffering political and economic turmoil, and therefore not something I will recommend.

Stocks are a good investment tool, but most people end up losing money instead of making money. It is therefore not something I will recommend in times of political and economic crises going on in Pakistan.

Dollar

If you have been holding cash and are fearful of investing it, considering that the USD has shown an average growth of 6% on a year-to-year basis against PKR, it might be wise to convert your PKR into Dollars. However, while it will safeguard you against PKR depreciation, it may not actually be a good hedge against an 11 to 12% inflation.

So while it is smart to keep your money in USD instead of PKR, it will not win the battle for you. If you are one of those who want to keep extra money at hand, besides your regular investments this is the right place to go. Especially for ex-pats who are being offered good interest rates by banks in Pakistan over their foreign currency bank accounts.

Gold

Gold has often been considered a hedge against inflation. In fact, many people have looked to gold as an "alternative currency" particularly in countries where the native currency is losing value. These countries utilize gold or other strong currencies when their own currency has failed. Gold is a real, physical asset, and holds its value for the most part.

However, gold is not a true perfect hedge against inflation. When inflation rises, central banks increase interest rates as part of monetary policy.1 Holding onto an asset like gold that pays no yields is not as valuable as holding onto an asset that does, particularly when rates are higher, meaning yields are higher.

There are better assets to invest in when aiming to protect yourself against inflation. But like any strong portfolio, diversification is key, and if you are considering to hedge against inflation Gold is worth looking at.

Stocks analysis

Stock analysis refers to the method that an investor or trader uses to evaluate and investigate a particular trading instrument, investment sector, or the stock market as a whole. Stock analysis is

also called equity analysis or market analysis. Investors or traders make buying or selling decisions based on stock analysis information.

Stock analysis helps traders to gain an insight into the economy, stock market, or securities. It involves studying the past and present market data and creating a methodology to choose appropriate stocks for trading. Stock analysis also includes the identification of ways of entry into and exit from the investments.

Types of Stock Analysis

Stock analysis can be grouped into two broad categories:

Fundamental Analysis

The fundamental stock analysis method involves the evaluation of a business at a basic financial level. Investors use fundamental analysis to determine whether the current price of a company's stock reflects the future value of the company.

Fundamental analysis uses different factors such as the current economic environment and finances of the company to estimate its stock value. Different key ratios are also used to determine the financial health and understand the true value of a company's stock.

a. **Earnings per share (EPS)** The EPS is useful when companies operating in the same industry need to be compared. A company's EPS indicates its profitability; hence, traders consider an increasing EPS a good sign. The higher the value of EPS, the more the company shares are worth buying0

b. **Price to Earnings ratio (P/E)** The P/E ratio indicates how much investors are willing to pay for the earnings of a company. A higher P/E value could mean an overvalued stock. Or, it could imply that the market is expecting the company to perform extremely well over time. On the other hand, a low P/E value is seen as unfavorable by the market.

c. **Price to Earnings to Growth ratio (PEG)** – The PEG ratio helps to determine the value of a company's stock while considering the earnings growth of the company. The PEG ratio, along with the P/E ratio, can help obtain a clearer picture of a company's stock than the P/E value alone.

d. **Price to Book ratio (P/B)** – The P/B ratio is used to compare the market value of a company with its book value. It seeks the value that the stock market places on a company's stock relative to the book value of the company. A company with sound financial health will trade for more than its book value since investors will consider the company's future growth while pricing the stocks.
e. **Return on Equity (ROE)** – It measures how effectively a company uses its assets for producing earnings. A high ROE implies that a company squeezes out greater profits with available assets. Hence, with all other things equal, it will be better to invest in high ROE companies in the long run.
f. **Dividend Payout Ratio** – It measures the percentage of the company's earnings paid to shareholders or owners. The earnings of the company, which are not passed on to the shareholders, are used to pay off debts, reinvest in business operations, or are retained for future use

Technical Analysis

The technical analysis method involves examining data generated through market activities, such as volume and prices. Analysts following such a type of stock analysis use technical indicators and tools like charts and oscillators to identify patterns that can indicate future price trends or direction.

Technical analysts examine the historical trading data of a security and estimate the future move of the security. It is frequently used for forex and commodities. The technical analysis is based on the following assumptions:

a. **The market knows it all.** Technical analysis assumes that the market price of a stock reflects all that has or can affect a company. Technical analysts consider that all the factors affecting the company are priced into the security.
b. **Price follows a trend.** It implies that once a trend is established, future prices tend to follow the direction of the trend. Such an assumption is the basis of many strategies for technical trading.
c. **History is likely to be repeated.** History repeats itself mainly concerning price movement. Market psychology causes price movements to repeat. Technical analysis involves using chart patterns to analyze the movements in the market and study trends.

Charts that have been used for over 100 years are still relevant since price movement patterns are often repetitive.

Problem statement

Now the Actual Problem is the buyer of share/ the inverter don't know which company (listed in Pakistan stock Market) has been in profit and which one is in devolution, because the data is just in numbers, so it's hard to manually calculate and predict that which company had a great past and a bright future

Proposed System

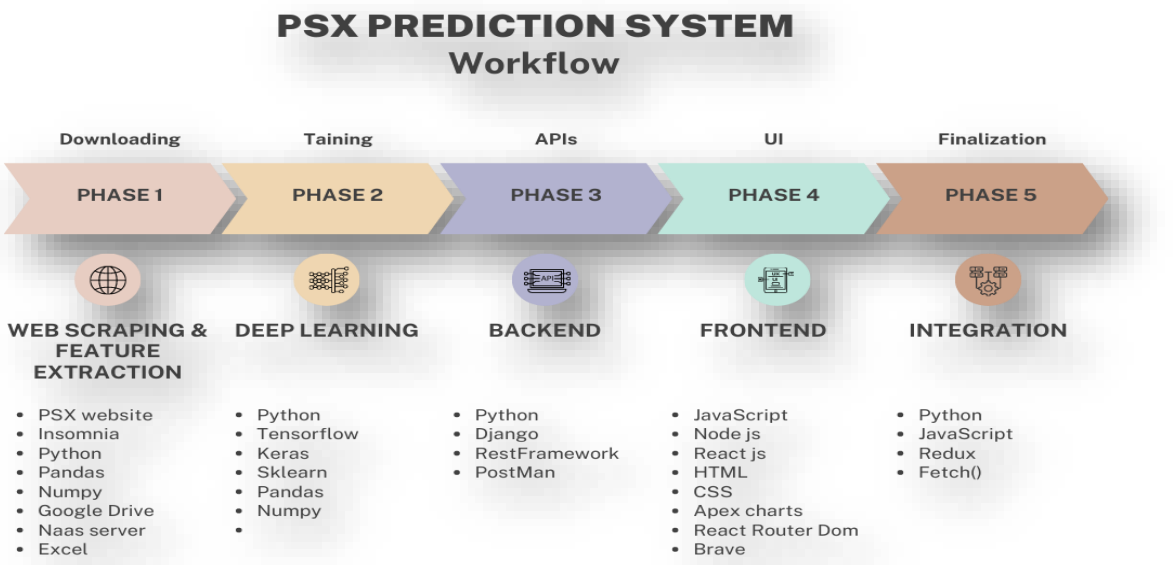
Our System calculates the previous data, preview the present worth and predict the future of the companies, also suggests to invest in the companies to have a bright profitable future.

The core objective of our project is to predict the Companies evaluation or devaluation, using the calculation of the previous data of Companies and their current situation

Methodology:

A web-based cross-platform Mobile Application and also a Website, that simply shows the growth of companies (that are listed in PSX). Simply can say that the system will take PSX real time data & Chosen Trading Style by Trader, Process/Calculates the previous data and current worth, then preview the Prediction Graph as output of the system that will help the trader while choosing the company to invest in.

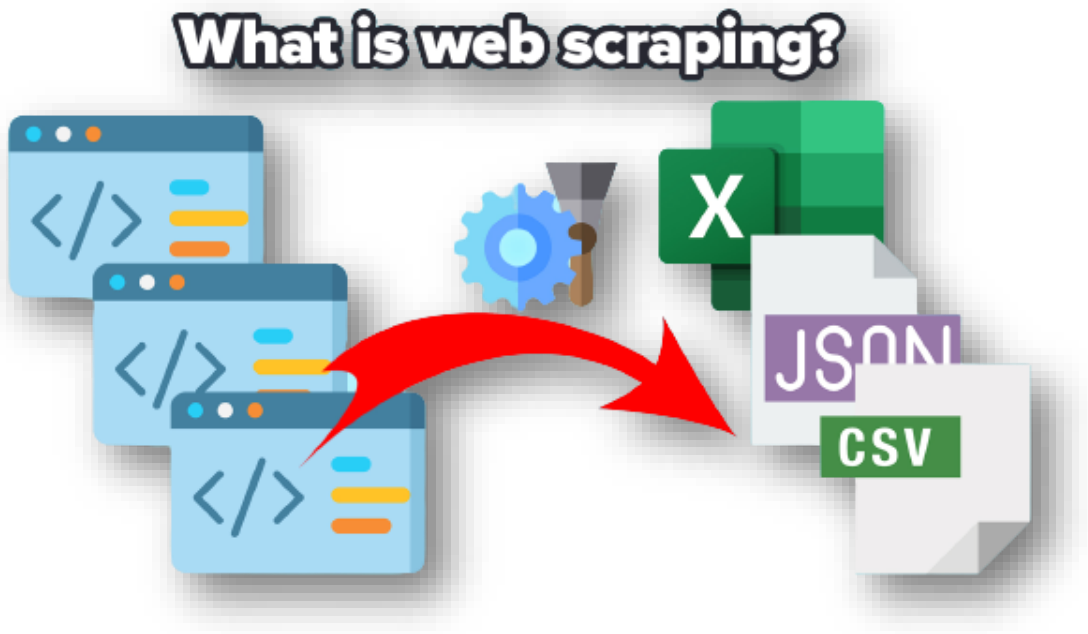
Workflow



Tools and Technologies

WEB scraping and Feature extraction (FHASE 1)

Web scraping refers to the **extraction of data from a website**. This information is collected and then exported into a format that is more useful for the user. Be it a spreadsheet or an API.



Although web scraping can be done manually, in most cases, automated tools are preferred when scraping web data as they can be less costly and work at a faster rate.

But in most cases, web scraping is not a simple task. Websites come in many shapes and forms, as a result, web scrapers vary in functionality and features.

Please note that you may encounter captchas when attempting to scrape some websites, so we suggest reading several guides on how to avoid & bypass captchas before scraping a website

Deep Learning (PHASE 2)

Training a model simply means learning (determining) good values for all the weights and the bias from labeled examples. In supervised learning, a machine learning algorithm builds a model by examining many examples and attempting to find a model that minimizes loss; this process is called empirical risk minimization

Backend (PHASE 3)

Django is an open-source web framework. It has been written in Python and is free. A high-level framework, it allows for faster web development with clean, practical design. With all the common

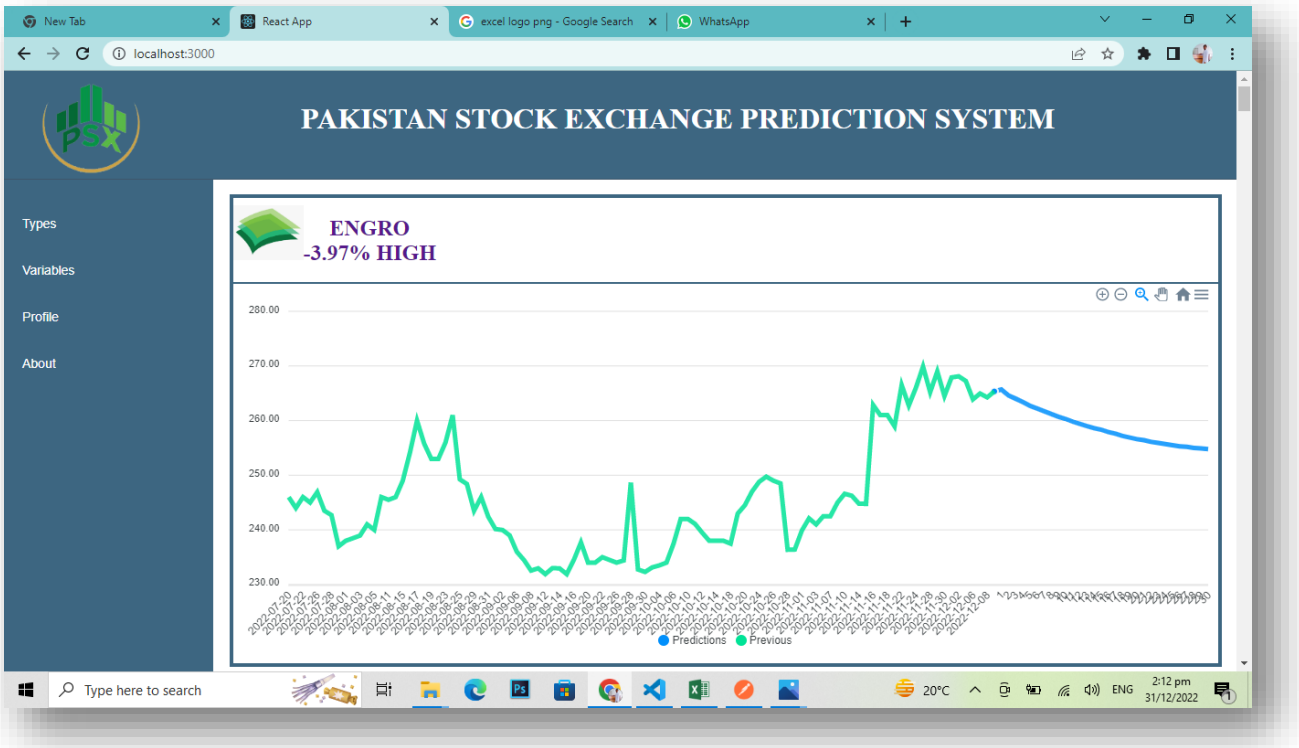
features already built-in, developers can focus on business logic rather than taking care of repetitive functionality. Some interesting features of Django are:

- Template language specifically created to cater to front-end development
- Fast, secure, scalable and versatile
- Takes care of site maps, RSS feeds, content administration, user authentication, etc…
- Eliminates the need to write SQL queries by using Python objects
- Great documentation and support community

Learning Django may take time, but once you do so, you will appreciate the enhancement it brings to your applications.

Frontend (PHASE 4)

This is the front end design



Integration (PHASE 5)

Integration of Django rest framework api, react app and deep learning model with previous data and predicted values which return by lstm model for each company

Code detailed description

final

January 9, 2023

0.1 Historical data web scraping (Downlaoding)

0.1.1 This code is generated by Insomnia application

- Output of this code is raw HTML Code (tags/data)
- Use (payload) variable to assign credencials i.e. month, year and symbol (name of company in short)

[3]:
```
import http.client
conn = http.client.HTTPSConnection("dps.psx.com.pk")
payload = "month=3&year=2022&symbol=OGDC"
headers = {
    'Accept': "text/html, */*; q=0.01",
    'Accept-Language': "en-US,en;q=0.9",
    'Connection': "keep-alive",
    'Content-Type': "application/x-www-form-urlencoded; charset=UTF-8",
    'Origin': "https://dps.psx.com.pk",
    'Referer': "https://dps.psx.com.pk/historical",
    'Sec-Fetch-Dest': "empty",
    'Sec-Fetch-Mode': "cors",
    'Sec-Fetch-Site': "same-origin",
    'Sec-GPC': "1",
    'User-Agent': '''Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/
    537.36(KHTML, like Gecko)Chrome/103.0.5060.114 Safari/537.36,
    'X-Requested-With': "XMLHttpRequest'''
    }
conn.request("POST", "/historical", payload, headers)
res = conn.getresponse()
data = res.read()
print(data.decode("utf-8")[:1000])
```

```
<!-- .note(style="margin-bottom: 1rem;")div * Queued order represents
cummulative volume of all orders placed in the trading system during the day.
div ** Cancelled order represents volume of all the cancelled orders excluding
the orders cancelled by the trading system during dump state.
--><div class="tbl__wrapper tbl__wrapper--scrollable"><table class="tbl"
id="historicalTable" data-page-length="25"><thead class="tbl__head"><!--
trth(colspan="6")
th.center(colspan="2", style="width: 110px;") TOTAL BUY ORDERS
th.center(colspan="2", style="width: 110px;") TOTAL SELL ORDERS--><tr><th data-
```

```
name="time" data-type="string">TIME</th><th class="right" data-name="open" data-
type="number">OPEN</th><th class="right" data-name="high" data-
type="number">HIGH</th><th class="right" data-name="low" data-
type="number">LOW</th><th class="right" data-name="close" data-
type="number">CLOSE</th><th class="right" data-name="volume" data-
type="number">VOLUME</th><!-- th.right(data-name="obq", data-type="numb
```

0.1.2 This is the final altered code for (Web Scraping) downloading Historical Data

- Output of this code is CSV files.
- All thirty (30) companies' names which we have used in our project are listed on the "symbols" list.
- A loop will iterate through all companies's names and create a CSV fiels containing data

[11]:
```
import http.client
import pandas as pd
import time
symbols=['ENGRO','LUCK', 'OGDC','FFC','HBL','HUBC','PPL', 'POL',
         'EFERT','MCB','UBL','DGKC','PSO','SEARL','MLCF','BAHL',
         'MARI','TRG','ATRL','UNITY','SYS','MEBL','GHNI','NML',
         'PIOC','CHCC','PAEL','ISL','KAPCO','DAWH']
months=[1,2,3,4,5,6,7,8,9,10,11,12]
years=[2018,2019,2020,2021,2022,2023]
table_data=""
all_in_one_df=pd.DataFrame()

conn = http.client.HTTPSConnection("dps.psx.com.pk")
headers = {
    'Accept': "text/html, */*; q=0.01",
    'Accept-Language': "en-US,en;q=0.9",
    'Connection': "keep-alive",
    'Content-Type': "application/x-www-form-urlencoded; charset=UTF-8",
    'Origin': "https://dps.psx.com.pk",
    'Referer': "https://dps.psx.com.pk/historical",
    'Sec-Fetch-Dest': "empty",
    'Sec-Fetch-Mode': "cors",
    'Sec-Fetch-Site': "same-origin",
    'Sec-GPC': "1",
    'User-Agent': '''Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.
 ↪36
    (KHTML, like Gecko) Chrome/103.0.5060.114 Safari/537.36''',
    'X-Requested-With': "XMLHttpRequest"
    }
for symbol in symbols:
    time.sleep(1)
    table_data=""
    for year in years:
        time.sleep(1)
```

```
        for month in months:
            time.sleep(2)
            payload = f"month={month}&year={year}&symbol={symbol}"
            conn.request("POST", "/historical", payload, headers)
            res = conn.getresponse()
            data = res.read()
            data=data.decode("utf-8")
            if len(data)>600:
                table_data=table_data+data[1046:-22]
                print(f"{symbol} {month} {year} Added to Table!")
            else:
                print("Error 404!")
            final=data[0:1046]+table_data+data[-22:]
            df=pd.DataFrame(pd.read_html(final)[0])
            df.insert(loc=0, column="Symbol", value=symbol)
            all_in_one_df=pd.concat([all_in_one_df,df],ignore_index=True)
    final=data[0:1046]+table_data+data[-22:]
    df=pd.DataFrame(pd.read_html(final)[0])
    path=f"D:\Documents\FYP\Historical Data (dec)\{symbol}.csv"
    df.to_csv(path)
    path=f"D:\Documents\FYP\Historical Data (dec)\{symbol}.csv"
    data=pd.read_csv(path)
    data['TIME']=pd.to_datetime(data['TIME'])
    data=data.sort_values(by='TIME')
    data.rename(columns={"Unnamed: 0":'index'},inplace=True)
    data.drop("index",axis=1,inplace=True)
    path=f"D:\Documents\FYP\Historical Data (dec)\{symbol}.csv"
    data.to_csv(path)
all_in_one_df.to_csv("D:\Documents\FYP\Historical Data (dec)/All in one.csv")
```

```
ENGRO 1 2018 Added to Table!
ENGRO 2 2018 Added to Table!
ENGRO 3 2018 Added to Table!
ENGRO 4 2018 Added to Table!
ENGRO 5 2018 Added to Table!
ENGRO 6 2018 Added to Table!
ENGRO 7 2018 Added to Table!
```

0.1.3 A format sample of downloaded data.

- Contains data from January 2018 to December 2022
- total of five years
- with the exact interval of one (1 day)

[8]:
```
import pandas as pd
path=f"D:\Documents\FYP\Historical Data 18-22\ENGRO.csv"
pd.read_csv(path).head(),pd.read_csv(path).tail()
```

[8]:
```
(   Unnamed: 0        TIME    OPEN    HIGH     LOW   CLOSE   VOLUME
 0          22  2018-01-01  274.00  283.55  270.15  280.30  1429100
 1          21  2018-01-02  279.00  283.75  277.50  280.39  1577200
 2          20  2018-01-03  279.11  283.73  278.50  281.49  1271800
 3          19  2018-01-04  281.49  287.00  280.30  284.83  1103100
 4          18  2018-01-05  283.10  288.50  283.10  286.70  1146100,
       Unnamed: 0        TIME    OPEN    HIGH     LOW   CLOSE  VOLUME
 1234        1221  2022-12-26  259.00  265.40  259.00  264.69  327545
 1235        1220  2022-12-27  265.45  265.45  260.25  261.26  164502
 1236        1219  2022-12-28  263.00  263.00  253.50  256.18  567995
 1237        1218  2022-12-29  257.01  261.98  256.18  260.79  342240
 1238        1217  2022-12-30  259.01  264.00  259.01  262.01  296703)
```

0.2 Daily data automation Google drive & Naas server

0.2.1 This basic code will scrape the data available at "https://dps.psx.com.pk/timeseries/int/ENGRO" this URL

[16]:
```
import requests
import urllib.request
import pandas as pd
response= urllib.request.urlopen("https://dps.psx.com.pk/timeseries/int/ENGRO").
 ↪read()
data=pd.DataFrame(eval(response))
data.head(),data.tail()
```

[16]:
```
(   status message                      data
 0       1          [1673006683, 284.68, 1]
 1       1          [1673006593, 284.68, 1]
 2       1          [1673006578, 284.68, 0]
 3       1          [1673006473, 284.68, 0]
 4       1          [1673006428, 284.68, 0],
       status message                         data
 1168       1           [1672978825, 279.4, 150]
 1169       1           [1672978815, 278.95, 10]
 1170       1           [1672978759, 278.03, 10]
 1171       1           [1672978637, 276.98, 50]
 1172       1          [1672978501, 279.4, 1000])
```

0.2.2 This is the final altered code for web scarping (downloading) Daily Data

- this code will download data from the given URL and create a CSV
- that CSV will be send and stored in the given Google Drive account (you have to provide Credentials and token of the G.Frive account)
- to automate this code, uncomment the commented code and run this script on the naas.ai server and script will be scheduled for every 5 day of weeks

[21]:
```
import os
import time
import json
# import naas
import os.path
import requests
import pandas as pd
import urllib.request
from google.auth.transport.requests import Request
from google.oauth2.credentials import Credentials
from google_auth_oauthlib.flow import InstalledAppFlow
from googleapiclient.discovery import build
from googleapiclient.errors import HttpError
from googleapiclient.http import MediaFileUpload

scps=["https://www.googleapis.com/auth/drive"]
creds= None
folders={}
list_companies=['ENGRO','LUCK', 'OGDC','FFC','HBL','HUBC',
                'PPL', 'POL','EFERT', 'MCB','UBL','DGKC','PSO',
                'SEARL','MLCF','BAHL','MARI','TRG','ATRL','UNITY',
                'SYS','MEBL','GHNI','NML','PIOC','CHCC','PAEL','ISL',
                'KAPCO','DAWH']

if os.path.exists("token.json"):
    creds=Credentials.from_authorized_user_file("token.json",scps)
if not creds or not creds.valid:
    if creds and creds.expired and creds.refresh_token:
        creds.refresh(Request())
    else:
        flow= InstalledAppFlow.from_client_secrets_file("client_secret.
 ↪json",scps)
        creds= flow.run_local_server(port=0)
    with open("token.json",'w')as token:
        token.write(creds.to_json())
try:
    service= build("drive","v3",credentials=creds)
    response=service.files().list(
        q="name='FYP PSX daily data' and mimeType='application/vnd.google-apps.
 ↪folder'",
        spaces='drive'
    ).execute()
    if not response['files']:
        file_metadata={
            "name":"FYP PSX daily data",
            "mimeType":"application/vnd.google-apps.folder",
        }
```

```
        file = service.files().create(body=file_metadata,fields="id").execute()
        parent_folder_id=file.get('id')
    else:
        parent_folder_id=response['files'][0]['id']
    for item in list_companies:
        time.sleep(2)
        service= build("drive","v3",credentials=creds)
        response=service.files().list(
            q=f"name='{item}' and mimeType='application/vnd.google-apps.
↪folder'",
            spaces='drive'
        ).execute()
        if not response['files']:
            file_metadata={
                "name":item,
                "mimeType":"application/vnd.google-apps.folder",
                "parents":[parent_folder_id]
            }
            file = service.files().create(body=file_metadata,fields="id").
↪execute()
            folder_id=file.get('id')
            folders[item]=folder_id
        else:
            folder_id=response['files'][0]['id']
        cnt= urllib.request.urlopen("https://dps.psx.com.pk/timeseries/int/
↪"+item).read()
        df=pd.DataFrame(eval(cnt))
        loc= f"Daily data/{item}/{item} {time.strftime('%Y-%m-%d')}.csv"
        name= f"{item} {time.strftime('%Y-%m-%d')}.csv"
        mypath ="Daily data/"+item
        if not os.path.isdir(mypath):
            os.makedirs(mypath)
        df.to_csv(loc)
        print(name," Downloaded")
        file_metadata={
            "name": name,
            "parents":[folder_id]
        }
        media= MediaFileUpload(loc)
        upload_file= service.files().create(body=file_metadata,␣
↪media_body=media,fields="id").execute()
        print(name," backed up")
        subject="Data stored Successfully!"
        content='''The daily PSX data has been saved successfully to your␣
↪google drive. Thank you!'''
        print("-"*50)
except HttpError as e:
```

```
    print("Error"+str(e))
    subject="Failled to store data!"
    content='''Data could not be saved to your google drive. please try to␣
 ↪download manually!'''
email_to="email_address"
# naas.notification.send(email_to,subject,content)
# naas.scheduler.add(recurrence="0 17 * * 1,2,3,4,5")
```

```
ENGRO 2022-12-16.csv Downloaded
ENGRO 2022-12-16.csv backed up
--------------------------------------------------
LUCK 2022-12-16.csv Downloaded
LUCK 2022-12-16.csv backed up
--------------------------------------------------
```

0.2.3 A format sample of downloaded data.

- contains a lot of usable data
- 'data' column is the only column of intrest
- data column contains timeseries ,value and volume

[14]:
```
import pandas as pd
path=f"D:\Documents\FYP\FYP PSX daily data New\ENGRO\ENGRO 2022-12-21.csv"
pd.read_csv(path).head(),pd.read_csv(path).head().tail()
```

[14]:
```
(   Unnamed: 0  status  message                       data
 0           0       1      NaN   [1671621199, 260.11, 1]
 1           1       1      NaN  [1671620998, 260.11, 20]
 2           2       1      NaN   [1671620822, 260.11, 2]
 3           3       1      NaN  [1671620462, 260.11, 50]
 4           4       1      NaN  [1671620438, 260.11, 40],
    Unnamed: 0  status  message                       data
 0           0       1      NaN   [1671621199, 260.11, 1]
 1           1       1      NaN  [1671620998, 260.11, 20]
 2           2       1      NaN   [1671620822, 260.11, 2]
 3           3       1      NaN  [1671620462, 260.11, 50]
 4           4       1      NaN  [1671620438, 260.11, 40])
```

0.3 Daily data Cleaning and feature extraction

0.3.1 Data cleanig

- this code will extract the usefull information from the downloaded data and create a CSV file

[ ]:
```
import pandas as pd

symbols = ['ENGRO', 'LUCK', 'OGDC', 'FFC', 'HBL', 'HUBC','PPL',
           'POL', 'EFERT', 'MCB', 'DGKC', 'PSO', 'SEARL', 'MLCF',
           'BAHL', 'MARI', 'ATRL', 'UNITY','SYS', 'MEBL', 'GHNI',
```

```
           'PIOC', 'CHCC', 'PAEL', 'ISL','KAPCO']


for symbol in symbols:
    path = "FYP PSX daily data New"
    data=pd.read_csv(f"FYP PSX daily data New/{symbol}/{symbol} 2022-12-12.csv")
    for item in os.listdir(f"{path}/{symbol}"):
        df=pd.read_csv(f"{path}/{symbol}/{item}")
        data=pd.concat([data,df],ignore_index=True)
    data=data["data"]
    print("Rows before cleanup = ", data.shape)
    data=data.str.split(',',expand=True)
    data.rename(columns={0:'Date',1:'Value',2:"Volume"},inplace=True)
    temp=[]
    for i in data["Date"]:
        temp.append(i[1:])
    data['Date']=temp
    data.drop_duplicates(subset='Date',keep='first',inplace=True)
    data["Date"]=pd.to_datetime(data['Date'],unit='s')
    data.drop("Volume",axis=1,inplace=True)
    path=(f"Daily Data Cleaned New (Hours)/{symbol}.csv")
    data.to_csv(path)
    data=pd.read_csv(path,index_col=False)
    data=data.sort_values(by='Date')
    print("Shape Before ",data.shape)
    datelist=data['Date'].tolist()
    for i,x in enumerate(datelist):
        datelist[i]=datelist[i][:-6]
    data['Date']=datelist
    data.drop_duplicates(subset='Date',keep='first',inplace=True)
    data.rename(columns={"Unnamed: 0":'index'},inplace=True)
    data.drop("index",axis=1,inplace=True)
    print("Shape After ",data.shape)
    path=f"Daily Data Cleaned New (Hours)/{symbol}.csv"
    data.to_csv(path)
    print(symbol+" Cleaned and Stored Successfully.")
```

0.3.2 A format sample of downloaded data.

- final format of useful data
- contains data with the exact interval of an hour
- eight (8) working Hours per day

[22]:
```
import pandas as pd
path=f"D:\Documents\FYP\Daily Data Cleaned New (Hours)\ENGRO.csv"
pd.read_csv(path).head(),pd.read_csv(path).tail()
```

[22]:
```
(   Unnamed: 0           Date  Value
 0         322  2022-12-12 04  263.0
 1         293  2022-12-12 05  262.2
 2         257  2022-12-12 06  261.1
 3         199  2022-12-12 07  260.9
 4         149  2022-12-12 08  260.5,
     Unnamed: 0           Date   Value
 34        3606  2022-12-21 07  264.50
 35        3538  2022-12-21 08  263.35
 36        3483  2022-12-21 09  263.15
 37        3317  2022-12-21 10  261.00
 38        3199  2022-12-21 11  260.11)
```

0.4 Training long term models without training split

- Preparing data from taining i.e. changing sequancial data into 100 time steps (100 columns for each elements/row)
- Defining structure of stacked LSTM nueral network with 3 layers, each containing 50 neurons
- Trainig models on historical/long term data (Jan 2018 to June 2022)

[7]:
```
import pandas as pd
import numpy as np
from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras import layers
import math
from sklearn.metrics import mean_squared_error

list_companies=['ENGRO','LUCK', 'OGDC','FFC','HBL','HUBC','PPL',
                'POL','EFERT', 'MCB','UBL','DGKC','PSO','SEARL','MLCF',
                'BAHL','MARI','TRG','ATRL','UNITY','SYS','MEBL','GHNI',
                'PIOC','CHCC','PAEL','ISL','KAPCO']

for item  in list_companies:
    path=f"Historical Data/{item}.csv"
    data=pd.read_csv(path, index_col='TIME', parse_dates=True)
    data=data.sort_values(by='TIME')
    feat_data=data['HIGH']
    scaler=MinMaxScaler(feature_range=(0,1))

    scaled_feat_data=scaler.fit_transform(np.array(feat_data).reshape(-1,1))
    def create_dataset(dataset, time_step=1):
        dataX, dataY = [], []
        for i in range(len(dataset)-time_step-1):
            a = dataset[i:(i+time_step), 0]
            dataX.append(a)
            dataY.append(dataset[i + time_step, 0])
```

```
        return np.array(dataX), np.array(dataY)
    time_step = 100
    X_train, y_train = create_dataset(scaled_feat_data, time_step)
    X_train =X_train.reshape(X_train.shape[0],X_train.shape[1] , 1)
    model=Sequential()
    model.add(layers.LSTM(50,return_sequences=True,input_shape=(100,1)))
    model.add(layers.LSTM(50,return_sequences=True))
    model.add(layers.LSTM(50))
    model.add(layers.Dense(1))
    model.compile(loss='mean_absolute_error',optimizer='adam',metrics=["mae"])
    trained_LSTM_Model=model.
↪fit(X_train,y_train,epochs=100,batch_size=64,verbose=0)
    path=f"{item}.h5"
    model.save(path)
    print(f"{item} Trained and saved successfully.")
```

```
ENGRO Trained and saved successfully.
ATRL Trained and saved successfully.
```

0.4.1 Evaluation (RMSE)

- Evaluating trained models with testing data
- Checkng Root Mean Squared Error
- Testing data (July 2022 to Dec 2022)

[21]:
```
import pandas as pd
import numpy as np
from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras.models import load_model
from keras import layers
import math
from sklearn.metrics import mean_squared_error

list_companies=['ENGRO','LUCK', 'OGDC','FFC','HBL','HUBC','PPL',
                'POL','EFERT', 'MCB','UBL','DGKC','PSO','SEARL',
                'MLCF','BAHL','MARI','TRG','ATRL','UNITY','SYS',
                'MEBL','GHNI','PIOC','CHCC','PAEL','ISL','KAPCO']

for item  in list_companies:
    path=f"D:\Documents\FYP\Historical Data (dec)\{item}.csv"
    data=pd.read_csv(path, index_col='TIME', parse_dates=True)
    data=data.sort_values(by='TIME')
    feat_data=data['HIGH']
    scaler=MinMaxScaler(feature_range=(0,1))
    scaled_feat_data=scaler.fit_transform(np.array(feat_data).reshape(-1,1))
    def create_dataset(dataset, time_step=1):
```

```
        dataX, dataY = [], []
        for i in range(len(dataset)-time_step-1):
            a = dataset[i:(i+time_step), 0]
            dataX.append(a)
            dataY.append(dataset[i + time_step, 0])
        return np.array(dataX), np.array(dataY)
    time_step = 100
    X_test, ytest = create_dataset(scaled_feat_data ,time_step)
    X_test = X_test.reshape(X_test.shape[0],X_test.shape[1] , 1)
    path=f"D:\Documents\FYP\Trained Models\Historical\Kaggle\HIGH\{item}.h5"
    model=load_model(path)
    test_predict=model.predict(X_test,verbose=0)
    print(f"{item} Testing RMSE= {math.
↪sqrt(mean_squared_error(ytest,test_predict))}")
```

```
ENGRO Testing RMSE= 0.09593436377549973
LUCK Testing RMSE= 0.07090905006224886
OGDC Testing RMSE= 0.0736011275619031
FFC Testing RMSE= 0.07177131337622321
HBL Testing RMSE= 0.023819031200247046
HUBC Testing RMSE= 0.06802470341475683
PPL Testing RMSE= 0.08677431774619589
POL Testing RMSE= 0.056670847258128085
EFERT Testing RMSE= 0.04978023177476003
MCB Testing RMSE= 0.04963986582754888
UBL Testing RMSE= 0.10770490528840722
DGKC Testing RMSE= 0.07689685095234002
PSO Testing RMSE= 0.055381038625196964
SEARL Testing RMSE= 0.03924445432885445
MLCF Testing RMSE= 0.08163775345629895
BAHL Testing RMSE= 0.12103945277319282
MARI Testing RMSE= 0.0739463368330766
TRG Testing RMSE= 0.05642160666862987
ATRL Testing RMSE= 0.07820734921837831
UNITY Testing RMSE= 0.04548630156525249
SYS Testing RMSE= 0.0792659160161357
MEBL Testing RMSE= 0.0423957335338384
GHNI Testing RMSE= 0.0676795066165442
PIOC Testing RMSE= 0.07273333745103971
CHCC Testing RMSE= 0.06039967946633148
PAEL Testing RMSE= 0.054341845814996126
ISL Testing RMSE= 0.05723561803361117
KAPCO Testing RMSE= 0.07291275747815534
```

0.4.2 Predicting long term data

- Preparing data from prediction i.e. changing sequancial data into 100 time steps (100 columns for each elements/row)

- Predicting data for future 10 days

[32]:
```
import numpy as np
import pandas as pd
from keras.models import load_model
from sklearn.preprocessing import MinMaxScaler
def getLongTermPredictions(days,company,variable):
    lst_output=[]
    predicted=[]
    n_steps=100
    path=f"D:\Documents\FYP\Historical Data New (sorted)\{company}.csv"
    data=pd.read_csv(path, index_col='TIME', parse_dates=True)
    data=data.sort_values(by='TIME')
    feat_data=data[variable][:100]
    scaler=MinMaxScaler(feature_range=(0,1))
    test_data=scaler.fit_transform(np.array(feat_data).reshape(-1,1))
    path=f"D:\Documents\FYP\Trained␣
 ↪Models\Historical\Kaggle\{variable}\{company}.h5"
    model=load_model(path, compile=False)
    x_input=test_data[len(test_data)-100:].reshape(1,-1)
    temp_input=list(x_input)
    temp_input=temp_input[0].tolist()
    for i in range(days):
        if(len(temp_input)>100):
            x_input=np.array(temp_input[1:])
            x_input=x_input.reshape(1,-1)
            x_input = x_input.reshape((1, n_steps, 1))
            yhat = model.predict(x_input, verbose=0)
            temp_input.extend(yhat[0].tolist())
            temp_input=temp_input[1:]
            lst_output.extend(yhat.tolist())
        else:
            x_input = x_input.reshape((1, n_steps,1))
            yhat = model.predict(x_input, verbose=0)
            temp_input.extend(yhat[0].tolist())
            lst_output.extend(yhat.tolist())
        predicted.append({i+1:round(float(scaler.
 ↪inverse_transform([lst_output[-1]])),1)})
    return predicted
print(getLongTermPredictions(10,'ENGRO','HIGH'))
```

```
[{1: 268.6}, {2: 267.9}, {3: 267.6}, {4: 267.3}, {5: 267.1}, {6: 266.8}, {7:
266.5}, {8: 266.2}, {9: 265.8}, {10: 265.3}]
```

0.4.3 Original vs Predicted

- comparing original/actual data and predicted data manually

[33]:
```
import pandas as pd
company="ENGRO"
path=f"D:\Documents\FYP\Historical Data New (sorted)/{company}.csv"
data=pd.read_csv(path, index_col='TIME', parse_dates=True)
data=data.sort_values(by='TIME')
feat_data=data['HIGH']
feat_data[100:],getLongTermPredictions(10,company,'HIGH')
```

[33]:
```
(TIME
 2022-11-28    265.20
 2022-11-29    268.99
 2022-11-30    264.49
 2022-12-01    267.90
 2022-12-02    268.10
 2022-12-05    267.25
 2022-12-06    263.89
 2022-12-07    264.95
 2022-12-08    264.25
 2022-12-09    265.34
 Name: HIGH, dtype: float64,
 [{1: 268.6},
  {2: 267.9},
  {3: 267.6},
  {4: 267.3},
  {5: 267.1},
  {6: 266.8},
  {7: 266.5},
  {8: 266.2},
  {9: 265.8},
  {10: 265.3}])
```

0.5 Training short term models without testing split

- Preparing data from taining i.e. changing sequancial data into 25 time steps (25 columns for each elements)
- Defining structure of stacked LSTM nueral network with 3 layers, each containing 50 neurons
- Trainig models on Daily/short term data (07 July 2022 to 09 Dec 2022)

[6]:
```
import pandas as pd
import numpy as np
from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras import layers
import math
from sklearn.metrics import mean_squared_error

list_companies=['ENGRO','LUCK', 'OGDC','FFC','HBL','HUBC','PPL',
                'POL','EFERT', 'MCB','UBL','DGKC','PSO','SEARL',
```

```
                'MLCF','BAHL','MARI','TRG','ATRL','UNITY','SYS',
                'MEBL','GHNI','PIOC','CHCC','PAEL','ISL','KAPCO']

for item  in list_companies:
    path=f"D:\Documents\FYP\Daily Data Cleaned (Hours)\{item}.csv"
    data=pd.read_csv(path, index_col='Date', parse_dates=True)
    data=data.sort_values(by='Date')
    feat_data=data['Value']
    scaler=MinMaxScaler(feature_range=(0,1))
    scaled_feat_data=scaler.fit_transform(np.array(feat_data).reshape(-1,1))
    def create_dataset(dataset, time_step=1):
        dataX, dataY = [], []
        for i in range(len(dataset)-time_step-1):
            a = dataset[i:(i+time_step), 0]
            dataX.append(a)
            dataY.append(dataset[i + time_step, 0])
        return np.array(dataX), np.array(dataY)
    time_step = 25
    X_train, y_train = create_dataset(scaled_feat_data, time_step)
    X_train =X_train.reshape(X_train.shape[0],X_train.shape[1] , 1)
    model=Sequential()
    model.add(layers.LSTM(50,return_sequences=True))
    model.add(layers.LSTM(50,return_sequences=True))
    model.add(layers.LSTM(50))
    model.add(layers.Dense(1))
    model.compile(loss='mean_squared_error',optimizer='adam',metrics=["mse"])
    trained_LSTM_Model=model.
 ↪fit(X_train,y_train,epochs=100,batch_size=64,verbose=0)
    path=f"D:\Documents\FYP\Trained Models\Daily\{item}.h5"
    model.save(path)
    print(f"{item} Trained and saved successfully.")
```

```
ENGRO Trained and saved successfully.
ATRL Trained and saved successfully.
```

0.5.1 Evaluation (RMSE)

- Evaluating trained models with testing data
- Checkng Root Mean Squared Error
- Testing data (12 Dec 2022 to 21 Dec 2022)

[34]:
```
import pandas as pd
import numpy as np
from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras.models import load_model
from keras import layers
```

```
import math
from sklearn.metrics import mean_squared_error

list_companies = ['ENGRO', 'LUCK', 'OGDC', 'FFC', 'HBL', 'HUBC',
                  'PPL', 'POL', 'EFERT', 'MCB', 'DGKC', 'PSO', 'UNITY',
                  'SYS', 'MEBL', 'GHNI', 'PIOC', 'CHCC', 'PAEL']
time_step = 25
for item  in list_companies:
    path=f"D:\Documents\FYP\Daily Data Cleaned New (Hours)\{item}.csv"
    data=pd.read_csv(path)
    data=data.sort_values(by='Date')
    feat_data=data['Value']
    scaler=MinMaxScaler(feature_range=(0,1))
    scaled_feat_data=scaler.fit_transform(np.array(feat_data).reshape(-1,1))
    def create_dataset(dataset, time_step=1):
        dataX, dataY = [], []
        for i in range(len(dataset)-time_step-1):
            a = dataset[i:(i+time_step), 0]
            dataX.append(a)
            dataY.append(dataset[i + time_step, 0])
        return np.array(dataX), np.array(dataY)
    X_test, ytest = create_dataset(scaled_feat_data ,time_step)
    X_test = X_test.reshape(X_test.shape[0],X_test.shape[1] , 1)
    path=f"D:\Documents\FYP\Trained Models\Daily\{item}.h5"
    model=load_model(path)
    test_predict=model.predict(X_test,verbose=0)
    print(f"{item} Testing RMSE= {math.
 ↪sqrt(mean_squared_error(ytest,test_predict))}")
```

```
ENGRO Testing RMSE= 0.10048811280406324
LUCK Testing RMSE= 0.13961047805482424
OGDC Testing RMSE= 0.1663619234805757
FFC Testing RMSE= 0.21430400306733294
HBL Testing RMSE= 0.15295335132192325
HUBC Testing RMSE= 0.06568707546415622
PPL Testing RMSE= 0.16301878154998667
POL Testing RMSE= 0.16097892825045115
EFERT Testing RMSE= 0.15757405078852452
MCB Testing RMSE= 0.1215592059594386
DGKC Testing RMSE= 0.21679293010707215
PSO Testing RMSE= 0.10621913550892488
UNITY Testing RMSE= 0.10966841934886817
SYS Testing RMSE= 0.2912707218925379
MEBL Testing RMSE= 0.2790738318314404
GHNI Testing RMSE= 0.0531345750138595
PIOC Testing RMSE= 0.0899265135616821
CHCC Testing RMSE= 0.2345220253557714
PAEL Testing RMSE= 0.12097795665895905
```

0.5.2 Predicting short term data

- Preparing data from prediction i.e. changing sequancial data into 100 time steps (25 columns for each elements/row)
- Predicting data for future 8 hours

[36]:
```
import numpy as np
import pandas as pd
from keras.models import load_model
from sklearn.preprocessing import MinMaxScaler

def getShortTermPredictions(days,company):
    lst_output=[]
    predicted=[]
    n_steps=25
    path=f"D:\Documents\FYP\Daily Data Cleaned New (Hours)\{company}.csv"
    data=pd.read_csv(path, index_col='Date', parse_dates=True)
    data=data.sort_values(by='Date')
    feat_data=data["Value"][-25:]
    scaler=MinMaxScaler(feature_range=(0,1))
    test_data=scaler.fit_transform(np.array(feat_data).reshape(-1,1))
    path=f"D:\Documents\FYP\Trained Models\Daily\{company}.h5"
    model=load_model(path)
    x_input=test_data[len(test_data)-25:].reshape(1,-1)
    temp_input=list(x_input)
    temp_input=temp_input[0].tolist()
    for i in range(days):
        if(len(temp_input)>25):
            x_input=np.array(temp_input[1:])
            x_input=x_input.reshape(1,-1)
            x_input = x_input.reshape((1, n_steps, 1))
            yhat = model.predict(x_input, verbose=0)
            temp_input.extend(yhat[0].tolist())
            temp_input=temp_input[1:]
            lst_output.extend(yhat.tolist())
        else:
            x_input = x_input.reshape((1, n_steps,1))
            yhat = model.predict(x_input, verbose=0)
            temp_input.extend(yhat[0].tolist())
            lst_output.extend(yhat.tolist())
        predicted.append({i+1:round(float(scaler.
 ↪inverse_transform([lst_output[-1]])),1)})
    return predicted
getShortTermPredictions(8,"ENGRO")
```

[36]:
```
[{1: 261.3},
 {2: 261.0},
 {3: 260.9},
```

```
 {4: 261.0},
 {5: 261.0},
 {6: 261.1},
 {7: 261.1},
 {8: 261.1}]
```

0.5.3 Original vs Predicted

- comparing original/actual data and predicted data manually

[38]:
```
import pandas as pd
company="ENGRO"
path=f"D:\Documents\FYP\Daily Data Cleaned New (Hours)\{company}.csv"
data=pd.read_csv(path, index_col='Date', parse_dates=True)
data=data.sort_values(by='Date')
feat_data=data['Value']
feat_data[-8:],getShortTermPredictions(8,company)
```

[38]:
```
(Date
 2022-12-21 04:00:00    264.90
 2022-12-21 05:00:00    266.01
 2022-12-21 06:00:00    264.97
 2022-12-21 07:00:00    264.50
 2022-12-21 08:00:00    263.35
 2022-12-21 09:00:00    263.15
 2022-12-21 10:00:00    261.00
 2022-12-21 11:00:00    260.11
 Name: Value, dtype: float64,
 [{1: 261.3},
  {2: 261.0},
  {3: 260.9},
  {4: 261.0},
  {5: 261.0},
  {6: 261.1},
  {7: 261.1},
  {8: 261.1}])
```