INTENSHIP REPORT ON UNIFIED MENTOR PRIVATE LIMITED

***A Intership Report Submitted***

***For the partial fulfillment of the requirements for the award of***

**B.TECH IN**

**COMPUTER SCIENCE ENGINEERING**

*Submitted by*

**BASANT KUMAR PASWAN - (001CSL23GT007)**

****

**SESSION (2023-2026)**

**R.K.D.F UNIVERSTY RANCHI**

**DEPARTMENT OF COMPUTER SCIENCE ENGINEERING**



**R.K.D.F UNIVERSITY RANCHI CERTIFICATE**

This is to certify that the project report “**Intership Report” of Unified Mentor Private Limited**” is a bonafide work of **BASANT KUMAR PASWAN (001CSL23GT007)** who carried out authentic project work under supervision and guidance of guide.

This is to further certify to the best of my knowledge that this project has not been carried out earlier in this University.

To the best of my knowledge, the matter embodied in this project has not been submitted to any other University/Institute for the award of any Degree or Diploma.

Date:

**{Signature of HOD} {Signature Of Supervisor}**

**Shubhangni Dey Abhishek kr. Singh**

**Head of the Department Supervisor**

**Department of computer science engineering Department of Computer science Engineering**

**R.K.D.F University R.K.D.F University**

**Ranchi Ranchi**

**Internal Examiner External Examiner**

# ACKNOWLEDGEMENT

The Intenship Training in itself is an acknowledged to the inspiration, drive, technical assistance contributed to it by many individuals. This training work would have never been completed without the guidance and assistance that I received from time to time from Internship during the whole training process.

I express my sincere gratitude and indebtedness **to Dr. AMIT KUMAR PANDEY (Registrar)** and **SHUBHAGNI DEY (Course Coordinator,Department of Computer Science Engineering), R.K.D.F University, Ranchi** for giving me an opportunity to enhance my skill in the field of Computer science Technology.

Last but not the least we also thank all my friends and other people who provided us with an atmosphere conductive to optimum learning during this project

**BASANT KUMAR PASWAN -(001CSL23GT007)**

# CONTENTS

1. Introduction Of TCS Stock Data
2. Objective
3. Explanation Summary
4. Import Required Libraries
5. Load the Dataset
6. Data Preprocessing
7. Exploratory Data Analysis (EDA)
8. Feature Engineering
9. Model Building and Prediction 10. Visualize Model Performance 11. Conclusion

## INTRODUCTION OF TCS Stock Data

Tata Consultancy Services (TCS) is an Indian multinational information technology (IT) services and consulting company headquartered in Mumbai, Maharashtra, India with its largest campus located in Chennai, Tamil Nadu, India. As of February 2021, TCS is the largest IT services company in the world by market capitalisation ($200 billion). It is a subsidiary of the Tata Group and operates in 149 locations across 46 countries.

TCS is the second largest Indian company by market capitalisation and is among the most valuable IT services brands worldwide.In 2015, TCS was ranked 64th overall in the Forbes World's Most Innovative Companies ranking, making it both the highest-ranked IT services company and the top Indian company. As of 2018, it is ranked eleventh on the Fortune India 500 list.In April 2018, TCS became the first Indian IT company to reach

$100 billion in market capitalisation and second Indian company ever (after Reliance Industries achieved it in 2007) after its market capitalisation stood at ₹6.793 trillion (equivalent to ₹7.3 trillion or US$100 billion in 2019) on the Bombay Stock Exchange.

In 2016–2017, parent company Tata Sons owned 72.05% of TCS and more than 70% of Tata Sons' dividends were generated by TCS. In March 2018, Tata Sons decided to sell stocks of TCS worth $1.25 billion in a bulk deal.As of 15 September 2021, TCS has recorded a market capitalisation of US$200 billion, making it the first Indian IT firm to do so.



**Fig:-**Tcs Stock Data

# OBJECTIVE

Analyze the historical data of TCS stock to gain insights into stock behavior, identify trends, and forecast future stock prices**.**

**Dataset Columns Explanation**

* Date- Date of trading data.
* Open- Opening stock price on that day.
* High- Highest stock price of the day.
* Low- Lowest stock price of the day.
* Close- Closing stock price of the day.
* Volume- Number of shares traded.
* Dividends- Dividends paid on the stock.
* Stock Splits- Number of stock splits.

## Explanation Summary

This project covers EDA, visualization, feature engineering, and prediction modeling for TCS stock prices:

* EDA provides insights into the stock's historical patterns.
* Moving Averages help smooth out price trends.
* Linear Regression is used to predict closing prices.
* Evaluation metrics help validate the model’s accuracy, giving insight into its reliability

## Import Required Libraries

# import python libraries import numpy as np import pandas as pd

import matplotlib.pyplot as plt # visualizing data

%matplotlib inline import seaborn as sns

## Load the Dataset

df = pd.read\_csv('TCS\_stock\_info.csv', encoding = 'unicode\_escape')

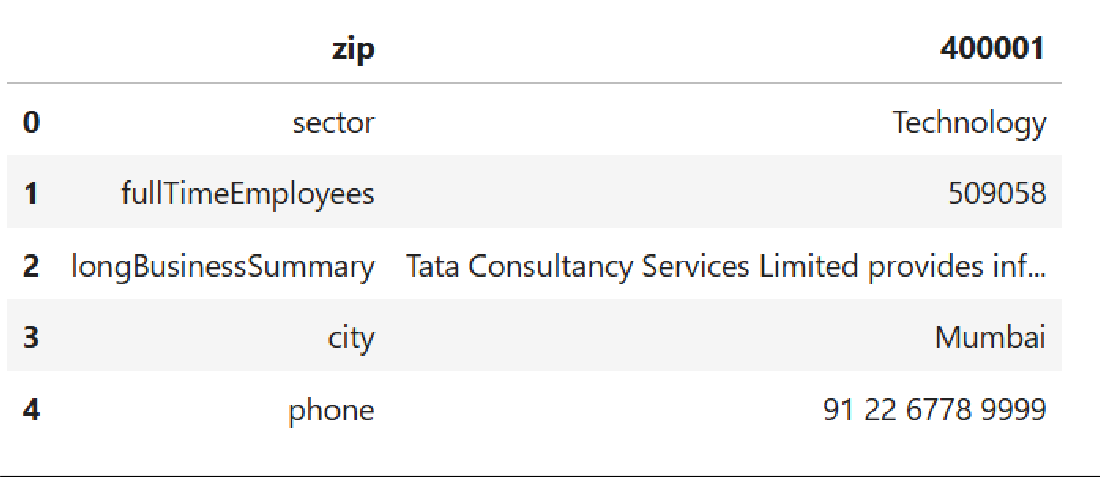
df.shape

OUTPUT

(150, 2)

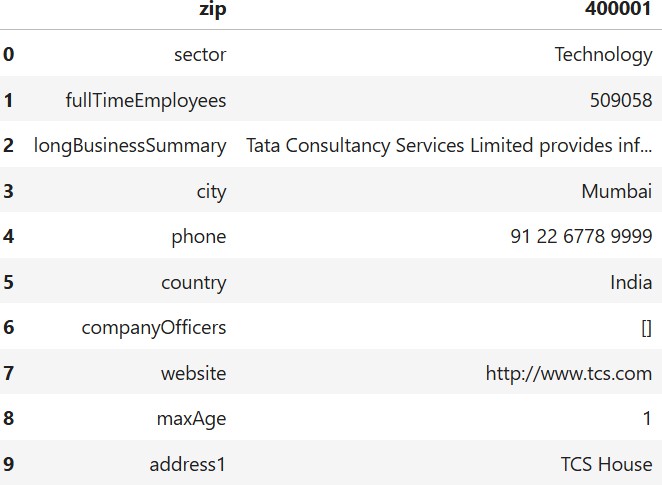
df.head()

OUTPUT



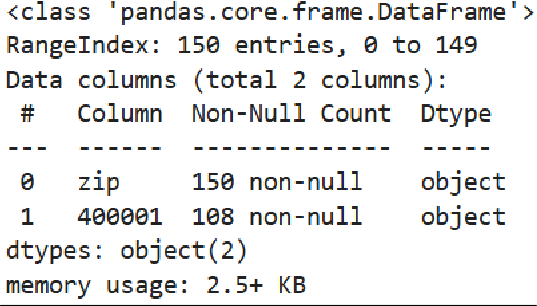
df.head(10)

OUTPUT



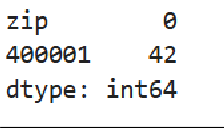
df.info()

OUTPUT



pd.isnull(df).sum()

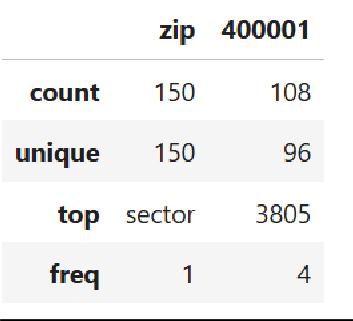
OUTPUT



# describe() method returns description of the data in the DataFrame (i.e. count, mean, std, etc)

df.describe()

OUTPUT



print(df.columns)

OUTPUT



## Data Preprocessing

* Check for null values and handle them.
* Convert necessary columns to numeric if needed.
* Check for any outliers in the data, especially in Volume and Close price.

import pandas as pd # Load the data

# Check actual column names print("Columns:", df.columns.tolist())

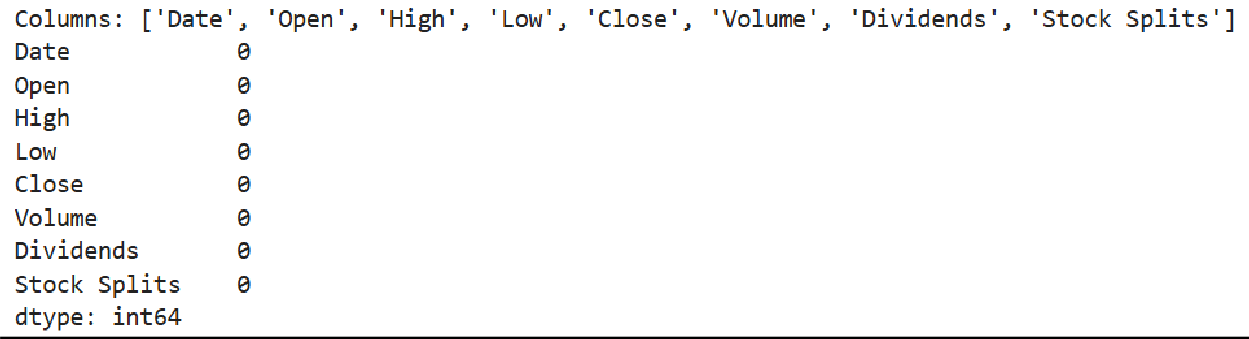
# Check for null values print(df.isnull().sum())

# Convert numeric columns if required

df['Open'] = pd.to\_numeric(df['Open'], errors='coerce') df['High'] = pd.to\_numeric(df['High'], errors='coerce') df['Low'] = pd.to\_numeric(df['Low'], errors='coerce') df['Close'] = pd.to\_numeric(df['Close'], errors='coerce')

# Fill any remaining NaN values df.fillna(method='ffill', inplace=True)

OUTPUT



plt.figure(figsize=(12, 6))

plt.plot(df['Date'], df['Close'], label='Close Price', color='blue') plt.title('TCS Closing Stock Price Over Time') plt.xlabel('Date')

plt.ylabel('Price (INR)') plt.grid(True) plt.legend() plt.tight\_layout() plt.show()

OUTPUT



From Above Graph we can see that Stock gives good return to the investors in a long period of time.

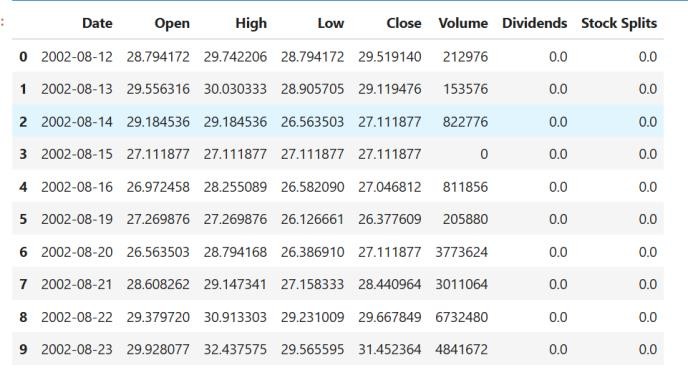
## Exploratory Data Analysis (EDA)

* Price Trends: Visualize the Open, Close, High, and Low prices overtime.
* Volume Analysis: Analyze trading volumes.
* Moving Averages: Calculate moving averages for trend analysis.

df = pd.read\_csv('TCS\_stock\_history.csv', encoding = 'unicode\_escape') df.shape

df.head(10)

OUTPUT



df = pd.read\_csv('TCS\_stock\_history.csv',parse\_dates=['Date'], encoding = 'unicode\_escape')

df.sort\_values('Date', inplace=True)

# Show basic info #print(df.info()) #print(df.head())

import pandas as pd

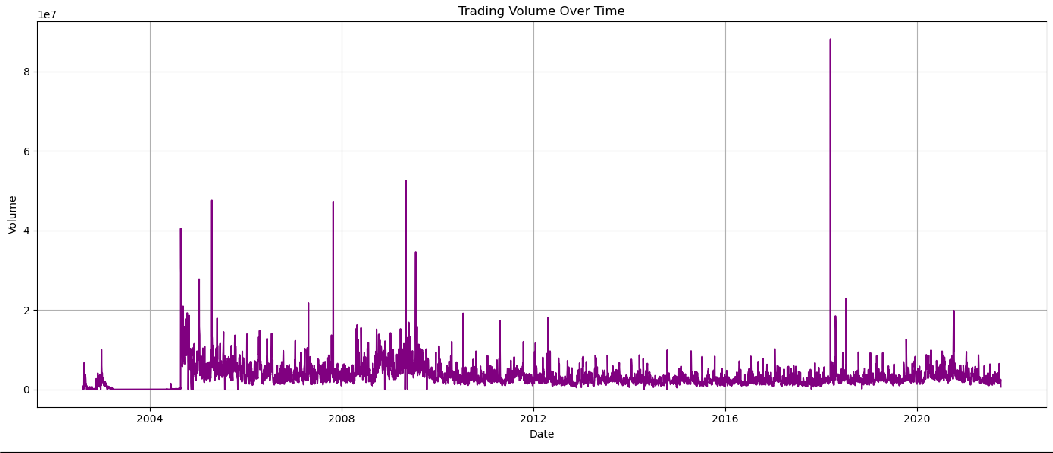
import matplotlib.pyplot as plt import seaborn as sns

# Plot trading volume over time plt.figure(figsize=(14, 6))

sns.lineplot(data=df, x='Date', y='Volume', color='purple') plt.title('Trading Volume Over Time')

plt.xlabel('Date') plt.ylabel('Volume') plt.grid(True) plt.tight\_layout() plt.show()

OUTPUT



From Above graph we can easily understand that Stock has high trading volume, which gives market sentiment and liquidity.

import pandas as pd

import matplotlib.pyplot as plt

# Ensure Date is datetime and sorted df['Date'] = pd.to\_datetime(df['Date']) df = df.sort\_values('Date')

# Calculate moving averages

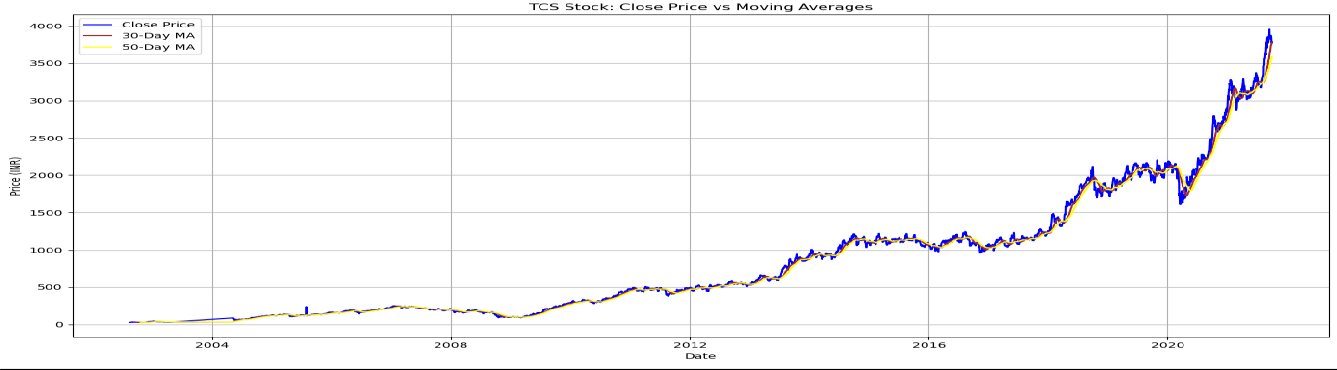
df['30-Day MA'] = df['Close'].rolling(window=30).mean() df['50-Day MA'] = df['Close'].rolling(window=50).mean() # Plotting

plt.figure(figsize=(14, 7))

plt.plot(df['Date'], df['Close'], label='Close Price', color='blue') plt.plot(df['Date'], df['30-Day MA'], label='30-Day MA', color='brown') plt.plot(df['Date'], df['50-Day MA'], label='50-Day MA', color='yellow') plt.title('TCS Stock: Close Price vs Moving Averages') plt.xlabel('Date')

plt.ylabel('Price (INR)') plt.legend() plt.grid(True) plt.tight\_layout() plt.show()

OUTPUT



From Above Graph we can easily understand The consistent alignment of the moving averages with the upward movement of the stock indicates strong price momentum and supports long-term bullish sentiment in TCS stock.

## Feature Engineering

* Extract features like Year, Month, Day, Day of Week from Date.
* Create lag features (e.g., previous day’s close, previous day’s high/low).

import pandas as pd

import matplotlib.pyplot as plt import seaborn as sns

# Load data

df.columns = df.columns.str.strip() # Remove extra spaces print("◻ First 5 rows:")

print(df.head())

# Basic info

print("\n◻ Data Info:") print(df.info())

# Missing values

print("\n◻ Missing Values:") print(df.isnull().sum())

# Descriptive statistics print("\n◻ Summary Statistics:") print(df.describe())

# Convert date column

df['Date'] = pd.to\_datetime(df['Date']) df = df.sort\_values('Date')

# Plot Dividends over time plt.figure(figsize=(12, 5))

plt.plot(df['Date'], df['Dividends'], label='Dividends', color='blue') plt.title('TCS Closing Price Over Time')

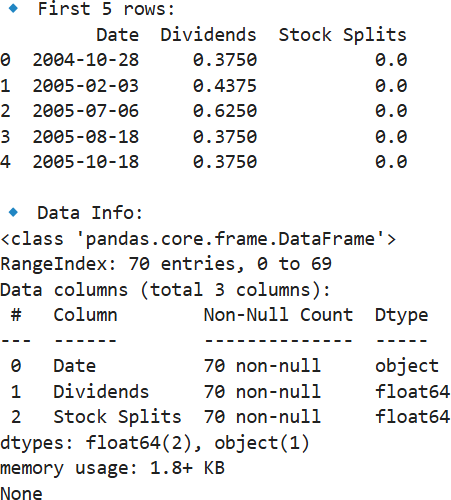
plt.xlabel('Date') plt.ylabel('Dividends') plt.legend() plt.tight\_layout() plt.show()

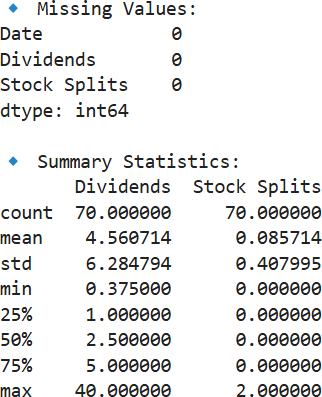
# Plot Stock Splits over time plt.figure(figsize=(12, 4))

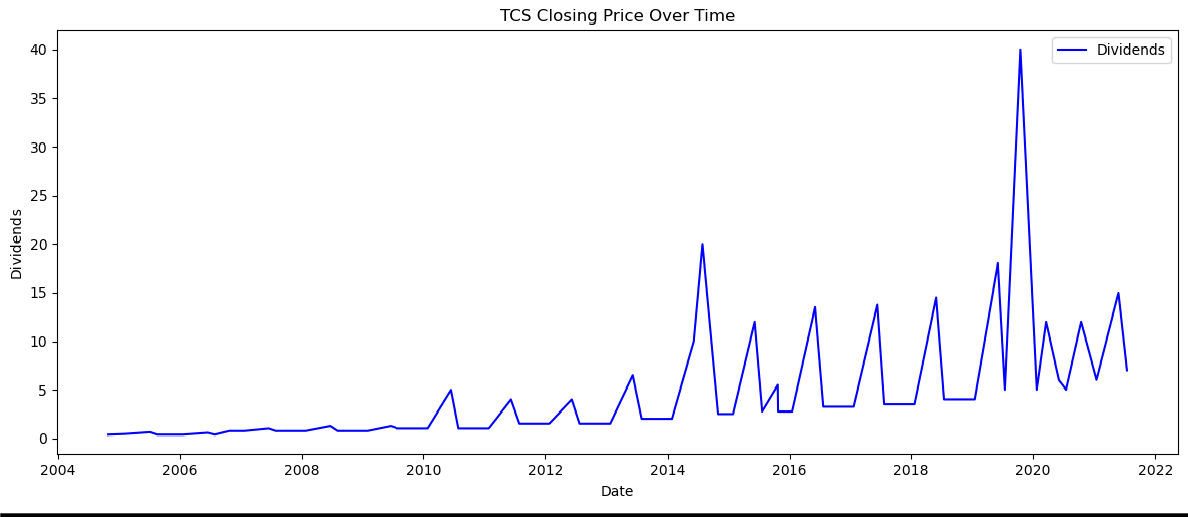
plt.plot(df['Date'], df['Stock Splits'], color='purple') plt.title('TCS Trading Volume Over Time') plt.xlabel('Date')

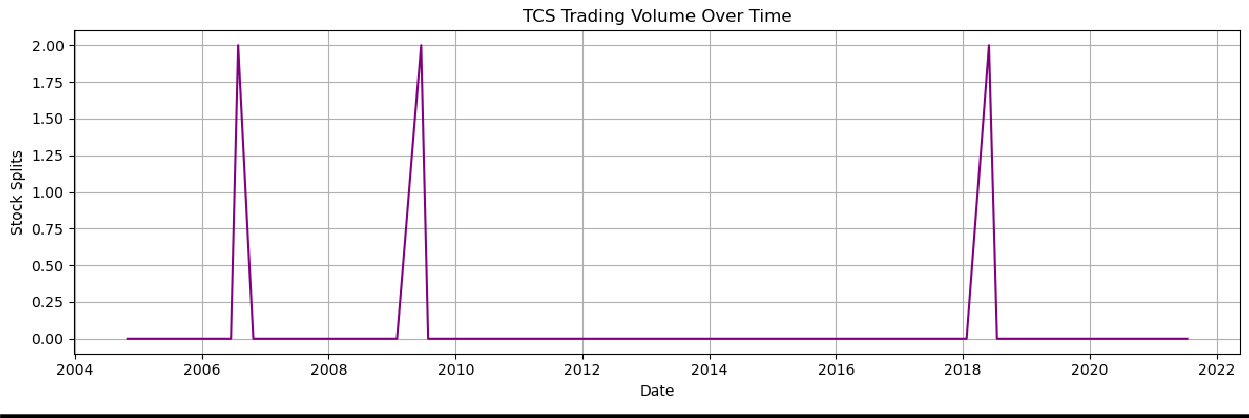
plt.ylabel('Stock Splits') plt.grid(True) plt.tight\_layout() plt.show()

OUTPUT









## Model Building and Prediction

* Use Linear Regression to predict the Close price based on features.
* Train/Test Split for model evaluation.

from sklearn.linear\_model import LinearRegression from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import mean\_squared\_error, r2\_score import numpy as np

import pandas as pd import numpy as np

from sklearn.model\_selection import train\_test\_split import matplotlib.pyplot as plt

# (your existing data loading and feature creation code) # Create lag features

df['Close\_Lag1'] = df['Close'].shift(1) df['Close\_Lag2'] = df['Close'].shift(2) df['Close\_Lag3'] = df['Close'].shift(3)

# Drop rows with NaN values from lagging df\_ml = df.dropna()

# Features (X) and Target (y)

X = df\_ml[['Close\_Lag1', 'Close\_Lag2', 'Close\_Lag3']] y = df\_ml['Close']

from sklearn.model\_selection import train\_test\_split # Use 80% for training, 20% for testing

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, shuffle=False)

import numpy as np

lr\_model = LinearRegression() lr\_model.fit(X\_train, y\_train)

# Predict

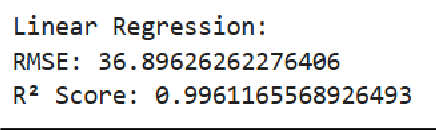
y\_pred\_lr = lr\_model.predict(X\_test)

# Evaluation

print("Linear Regression:")

print("RMSE:", np.sqrt(mean\_squared\_error(y\_test, y\_pred\_lr))) print("R² Score:", r2\_score(y\_test, y\_pred\_lr))

OUTPUT



Linear Regression is performing surprisingly well with just lag-based features. The extremely high R² suggests the stock has a strong short-term autocorrelation

## Visualize Model Performance

* Plot predicted vs. actual values.
* Scatter plot to observe prediction accuracy**.**

import pandas as pd import numpy as np

import matplotlib.pyplot as plt

from sklearn.linear\_model import LinearRegression from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import mean\_squared\_error,r2\_score import xgboost as xgb

# Load data

df.columns = df.columns.str.strip() df['Date'] = pd.to\_datetime(df['Date']) df = df.sort\_values('Date')

# Ensure numeric columns

for col in ['Open', 'High', 'Low', 'Close']:

df[col] = pd.to\_numeric(df[col], errors='coerce') df.fillna(method='ffill', inplace=True)

# Feature engineering df['Close\_Lag1'] = df['Close'].shift(1) df['Close\_Lag2'] = df['Close'].shift(2) df['Close\_Lag3'] = df['Close'].shift(3) df\_ml = df.dropna()

# Prepare features and target

X = df\_ml[['Close\_Lag1', 'Close\_Lag2', 'Close\_Lag3']] y = df\_ml['Close']

# Train-test split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, shuffle=False)

# Linear Regression

lr\_model = LinearRegression() lr\_model.fit(X\_train, y\_train) y\_pred\_lr = lr\_model.predict(X\_test)

# XGBoost

xgb\_model = xgb.XGBRegressor(objective='reg:squarederror', n\_estimators=100) xgb\_model.fit(X\_train, y\_train)

y\_pred\_xgb = xgb\_model.predict(X\_test)

# Evaluation

print("Linear Regression RMSE:", np.sqrt(mean\_squared\_error(y\_test, y\_pred\_lr))) print("XGBoost RMSE:", np.sqrt(mean\_squared\_error(y\_test, y\_pred\_xgb)))

# Plot plt.figure(figsize=(14, 6))

plt.plot(y\_test.index, y\_test.values, label='Actual Close', color='blue') plt.plot(y\_test.index, y\_pred\_lr, label='Linear Regression', linestyle='--') plt.plot(y\_test.index, y\_pred\_xgb, label='XGBoost', linestyle='--', color='green') plt.title('Actual vs Predicted Closing Prices')

plt.xlabel('Index') plt.ylabel('Close Price') plt.legend() plt.grid(True) plt.tight\_layout() plt.show()

OUTPUT



From Above graph we can see that TCS has outperformed in the market

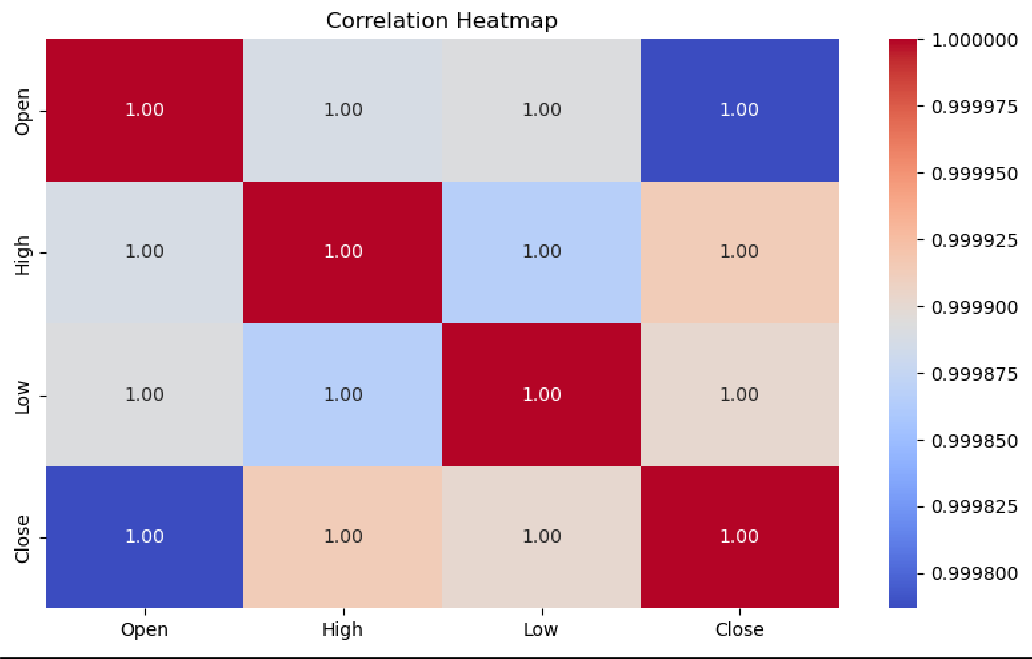
plt.figure(figsize=(8, 5))

sns.heatmap(df[['Open', 'High', 'Low', 'Close']].corr(), annot=True, cmap='coolwarm', fmt=".2f")

plt.title('Correlation Heatmap') plt.tight\_layout()

plt.show()

OUTPUT



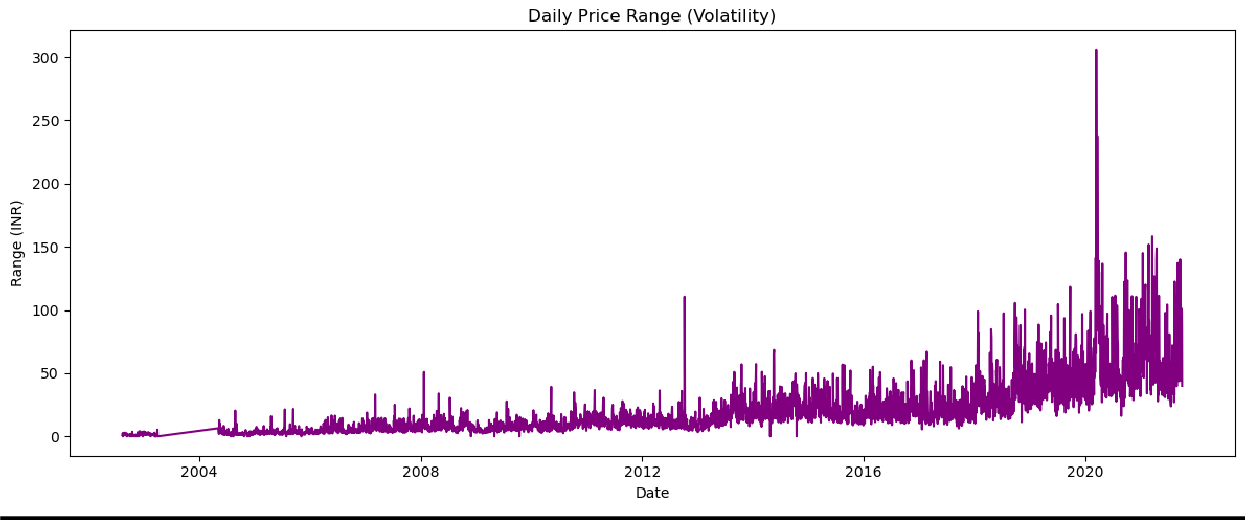
The heatmap confirms that the stock’s daily pricing components are tightly linked, which is typical in time-series stock data.

df['Range'] = df['High'] - df['Low'] plt.figure(figsize=(12, 5))

plt.plot(df['Date'], df['Range'], color='purple') plt.title('Daily Price Range (Volatility)') plt.xlabel('Date')

plt.ylabel('Range (INR)') plt.tight\_layout() plt.show()

OUTPUT



# CONCLUSION

TCS stock has shown steady long-term growth with upward momentum, with closing prices more frequent at lower levels due to stock splits. The distribution is right-skewed, Predictive models like XGBoost perform well, confirming trend stability. Price distribution and volume patterns support consistent investor interest.

