**TCS (Tata Consultancy Services)**

The dataset contains stock market data for TCS (Tata Consultancy Services) with the following columns:

**Overview**

Tata Consultancy Services (TCS) is a leading global IT services, consulting, and business solutions organization. Part of the Tata Group, India's largest multinational business group, TCS has over 500,000 of the world's best-trained consultants in 46 countries.

#### History

TCS was founded in 1968 by Tata Sons Limited. It is headquartered in Mumbai, India. Over the decades, TCS has grown to become one of the largest IT services firms in the world, known for its innovation and excellence in IT services, digital and business solutions.

**Services**: **IT Services**, **Consulting**, Business Solutions, Digital Transformation,

#### Global Presence

TCS operates in over 46 countries and has a strong presence in North America, Latin America, Europe, Asia-Pacific, and the Middle East & Africa. It has built a robust global delivery network and innovation labs across the globe.

#### Innovation and Research

TCS invests significantly in research and development (R&D). The company focuses on emerging technologies and trends to drive innovation. TCS Research and Innovation (R&I) works on developing next-gen technologies and solutions in areas such as AI, machine learning, *blockchain*, and more.

#### Corporate Social Responsibility (CSR)

TCS is committed to social responsibility and sustainability. Its CSR initiatives focus on education, health, and environmental sustainability. The company supports various community projects, including digital literacy programs, health initiatives, and efforts to promote sustainable practices.

#### Financial Performance

TCS is one of the most valuable IT services brands globally, with a strong financial performance. It is listed on the Bombay Stock Exchange (BSE) and the National Stock Exchange of India (NSE). The company has a robust revenue stream and profitability, consistently delivering value to its shareholders.

The dataset contains stock market data for TCS (Tata Consultancy Services) with the following columns:

1. **Date**: The date of the trading session (timing 9:15 AM to 3:30 PM in India) INTRADAY trading system.
2. Open: The opening price of the stock on that day.
3. High: The highest price of the stock on that day.
4. Low: The lowest price of the stock on that day.
5. Close: The closing price of the stock on that day.
6. **Adj Close**: The adjusted closing price of the stock, which accounts for dividends and stock splits.   
   While the closing price simply refers to the cost of shares at the end of the day, the adjusted closing price takes dividends, **stock splits**, and *new stock offerings into account*. The adjusted closing price is a more accurate indicator of stock value since it starts where the closing price finishes.  
     
   Adjusted close is the closing price after adjustments for all applicable splits and dividend distributions.
7. Volume: The trading volume, or the number of shares traded on that day.

Table 1: TCS given dataset

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Date | Open | High | Low | Close | Adj Close | Volume |
| Start: 12th June 2023 |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
| End: 10th June 2024 |  |  |  |  | 1 Year= 365 days  Market OFF = 365-244 = 144 | 244 rows (or entities)  \* 7 columns |

# Convert the 'Date' column to datetime format

df['Date'] = pd.to\_datetime(df['Date'], errors='coerce')

# Drop any rows where 'Date' conversion failed

df.dropna(subset=['Date'], inplace=True)

# Sort the data by date

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

# Summary statistics

summary\_stats = df.describe()

# Time range

time\_range = df['Date'].min(), df['Date'].max()

summary\_stats, time\_range

**Summary Statistics:**

1. Open:

* Mean: 3664.67
* Standard Deviation: 271.33
* Minimum: 3190.00
* Maximum: 4215.25

1. High:

* Mean: 3696.70
* Standard Deviation: 279.37
* Minimum: 3199.40
* Maximum: 4254.75

Similarly, for rest 5 columns.

Q. Which type of this dataset is it binary or another classifier or univariant, bivariant, multivariant? Please suggest. And how I know any data set nature is your binary class or metaclassifier.

Ans: The dataset you provided is best described as **multivariate time series data**. Here's a breakdown of why this classification is appropriate:

* **Multivariate**: The dataset includes multiple variables or features (e.g., Open, High, Low, Close, Adj Close, Volume).
* **Time Series**: The data points are indexed by time (i.e., Date), and there is an implicit order and temporal dependency between them.

How to Classify the Nature of a Dataset

When determining the nature of a dataset, here are some key considerations:

1. **Number of Variables**:

* **Univariate**: Contains a single variable or feature.
* **Bivariate**: Contains two variables, often analysed to explore relationships between them.
* **Multivariate**: Contains three or more variables.

1. **Classification Types**:

* **Binary Classification**: The target variable has two possible categories or classes (e.g., yes/no, true/false).
* **Multiclass Classification**: The target variable has more than two categories or classes (e.g., types of flowers, different product categories).

1. **Regression**: If the target variable is continuous (e.g., stock prices, temperature).

Note: This section, Ankit’s expertise, he will do the research on the continuous data, which is time series also, along with the concept we may be univariate and multivariate.

The Date column is in string format and will need to be converted to a datetime format for time series analysis.

Steps to Determine Dataset Nature

* **Step1: Identify the Target Variable:**
* For classification tasks, this is the variable you aim to predict, which should be categorical.
* For regression tasks, this is the *variable* you aim to predict, which should be continuous.
* **Step2: Analyse the Target Variable:**
* **Binary Classification**: If the target variable has exactly two unique values.
* **Multiclass Classification**: If the target variable has more than two unique values.
* **Regression**: If the target variable is continuous.
* **Step3: Count the Features**:
* Determine the number of features (***excluding*** the target variable) to classify the dataset as univariate, bivariate, or multivariate.

**18th June’24 (Tuesday)**

Q. How to check model accuracy + precision (performance) using ANN MLP model?

Ans: To evaluate the accuracy and precision of an Artificial Neural Network (ANN) model, such as an MLP (Multi-Layer Perceptron) regressor, you can use several metrics. In the context of regression models, common evaluation metrics include Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared (R²).

Here's how you can compute these metrics:

* **Mean Absolute Error (MAE)**: Measures the average magnitude of the errors in a set of predictions, without considering their direction.
* **Mean Squared Error (MSE)**: Measures the average of the squares of the errors—that is, the average squared difference between the estimated values and what is estimated.
* **Root Mean Squared Error (RMSE)**: The square root of the average of squared differences between prediction and actual observation.
* **R-squared (R²)**: Indicates the proportion of the variance in the dependent variable that is predictable from the independent variables.

ARIMA: Auto Regression Integrated Moving Average

Origin Destination Matrix = ODM, Origin destination Tensor: ODT

# import metric imp library for MAE, MSE, RMSE, R Sq

from sklearn.metrics import mean\_absolute\_error, mean\_squared\_error, r2\_score

# Calculate evaluation metrics

mae\_train = mean\_absolute\_error(y\_train\_actual, train\_predictions)

mse\_train = mean\_squared\_error(y\_train\_actual, train\_predictions)

rmse\_train = np.sqrt(mse\_train)

r2\_train = r2\_score(y\_train\_actual, train\_predictions)

mae\_test = mean\_absolute\_error(y\_test\_actual, test\_predictions)

mse\_test = mean\_squared\_error(y\_test\_actual, test\_predictions)

rmse\_test = np.sqrt(mse\_test)

r2\_test = r2\_score(y\_test\_actual, test\_predictions)

# Print the evaluation metrics

print("Training Metrics:")

print(f"MAE: {mae\_train:.2f}")

print(f"MSE: {mse\_train:.2f}")

print(f"RMSE: {rmse\_train:.2f}")

print(f"R²: {r2\_train:.2f}")

print("\nTesting Metrics:")

print(f"MAE: {mae\_test:.2f}")

print(f"MSE: {mse\_test:.2f}")

print(f"RMSE: {rmse\_test:.2f}")

print(f"R²: {r2\_test:.2f}")

MS Excel knowledge

ctrl + shift + down arrow = select column

use the feature of freeze panel for easy view: select row > view TAB > freeze group (or ribbon) > Freeze top row.

19th June’24 (Wednesday)

Our ANN MLP model run successfully on one year of TCS dataset consider Close price as predication. Today, Hari sir give 1st Sep 2002 – 19th June 2024

My shortcut is Yr\_22 and total years = 22, No. of rows = 5,419

<https://finance.yahoo.com/quote/TCS.NS/history/?period1=1030838400&period2=1718780413>

First, we need to delete manually unwanted things, such keyword (open in MS excel) are null,

Download SPPS software for find out Co-Relation:

https://caeis.etech.fh-augsburg.de/downloads/windows/latest-release/

***21st June’24 (Friday)***

After preprocessing the TCS data yr-22 data, we get release:

Hari sir + Debanjana di who teach me, how to delete null value from MS Excel: answer is very simple that you can make column and then sort by decreasing order, delete from last and then sort again in ascending order.

Before delete null (nine) row from original dataset = 5419 row \* 7 column

After delete row = 5410 rows

MLP Regressor

# Prepare the dataset for MLP

# Use the past 10 days' high prices to predict the next day's high price

X = []

y = []

look\_back = 10

for i in range(len(Close\_prices\_scaled) - look\_back):

    X.append(Close\_prices\_scaled[i:i + look\_back])

    y.append(Close\_prices\_scaled[i + look\_back])

X = np.array(X)

y = np.array(y).flatten()

# Reshape X to be 2D for MLPRegressor

X = X.reshape(X.shape[0], -1)  # Reshape to (samples, time\_steps \* features)

# Split the data into training and test sets

train\_size = int(len(X) \* 0.8)      # 80% taining, 20% testing

X\_train, X\_test = X[:train\_size], X[train\_size:]

y\_train, y\_test = y[:train\_size], y[train\_size:]

# Define and train the MLP REGRESSOR  model

mlp = MLPRegressor(hidden\_layer\_sizes=(100,), max\_iter=200, random\_state=42)

mlp.fit(X\_train, y\_train)

30th June’24 (Sunday) ANN SLP model making

Analysis value

# The value "**0.0002891845069825649**" represents a numerical proportion or fraction. In decimal form, it translates to approximately 0.0289%.

The value "**1.5877230907790363e-05**" in the context of "Train Loss" represents a very small number in scientific notation. Here's how to interpret it:

In standard decimal notation, 1.5877230907790363×10−5

1.5877230907790363×10−5 = 0.000015877230907790363

So, the train loss is approximately 0.00001590.00001590.0000159 or 0.00159%

3rd July’24 (Wednesday) **RNN multi-variant** Script

Prompt:

I need to forecast using only the RNN (recurrent neural network) multi-layer perception i.e. multi-variant (considering Close price) considering y = close (dependent variable) and x = rest all column (independent variable), concept, not by LSTM, the 5 days of CLOSE PRICE based on a given TCS dataset with a graph in simple ways and also suggesting which concept I learned. write code also Google Colab notebook in Python I select for running, please

4th July’24 (Thursday) LSTM and GRU

Graphically visualize

from tensorflow.keras.utils import plot\_model

# Generate the plot and save it to a file

plot\_model(model, to\_file='model\_architecture.png', show\_shapes=True, show\_layer\_names=True)

# Display the plot inline (if using a Jupyter notebook)

from IPython.display import Image

Image(filename='model\_architecture.png')