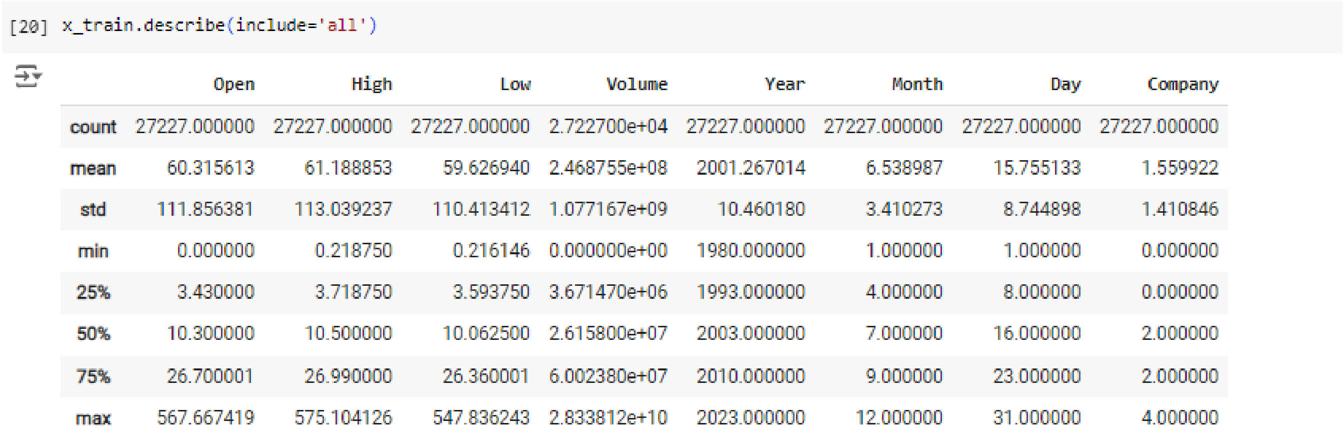
Data Collection and Preprocessing Phase

Date	24 April 2024
Team ID	739877
Project Title	Crystal Ball Analysis: Projecting Share Prices Of The Leading Gpu Titans
Maximum Marks	6 Marks

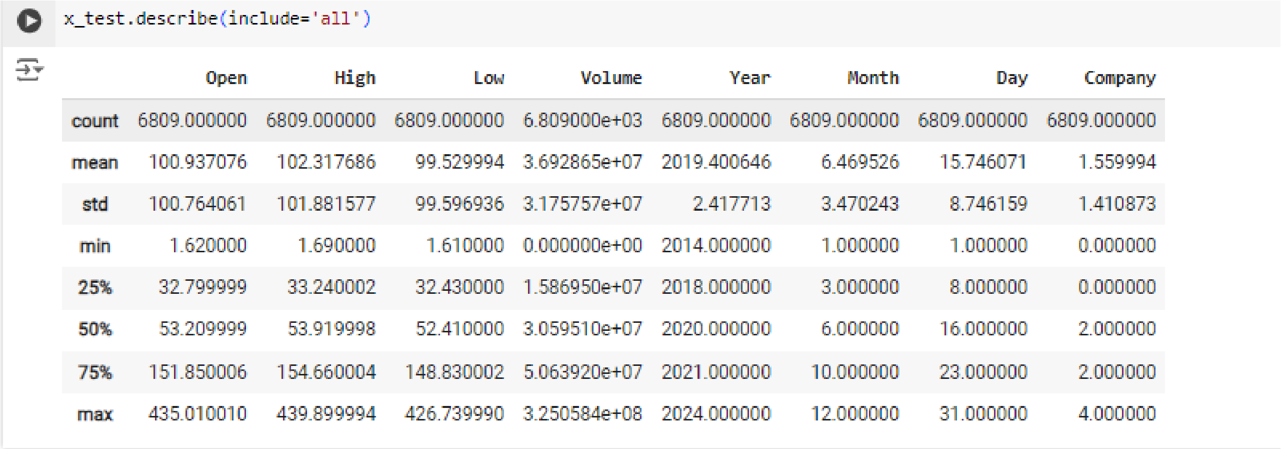
Data Exploration and Preprocessing Report

Dataset variables will be statistically analysed to identify patterns and outliers, with Python employed for preprocessing tasks like normalization and feature engineering. Data cleaning will address missing values and outliers, ensuring quality for subsequent analysis and modeling, and forming a strong foundation for insights and predictions.

Section	**Description**
Data Overview	Descriptive Analysis:- 

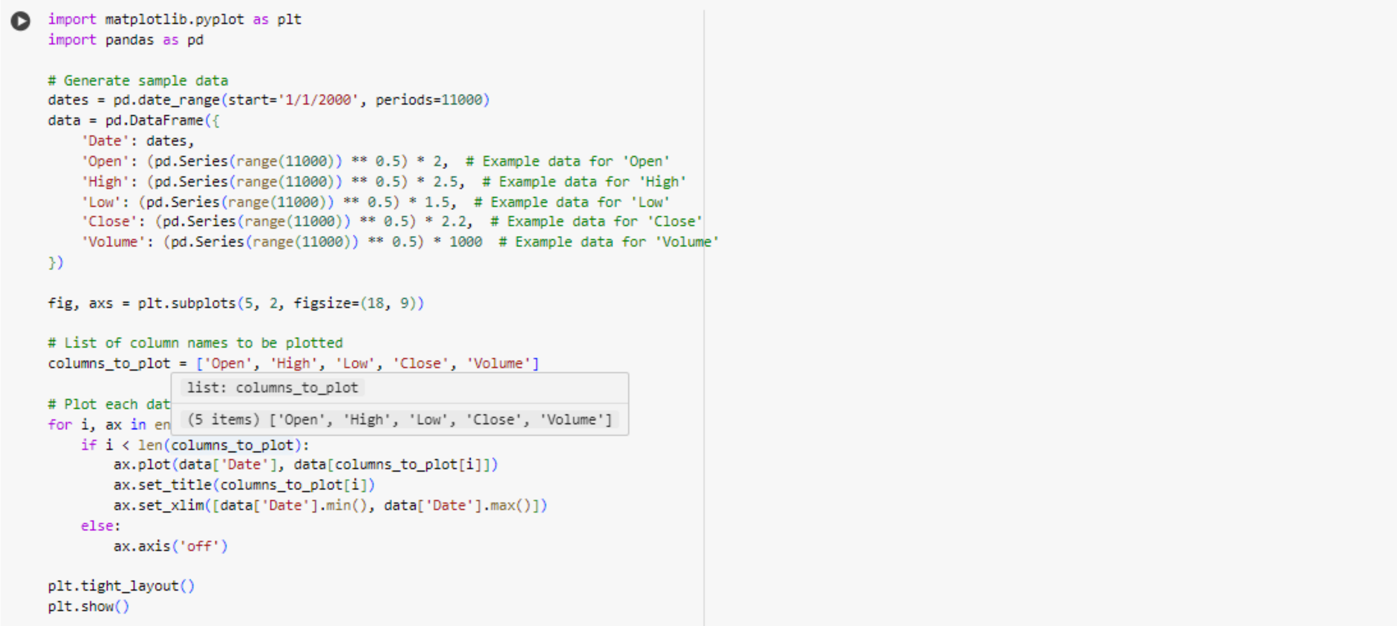
```
[20] x_train.describe(include='all')
```

```
               Open         High          Low        Volume         Year        Month          Day      Company
count  27227.000000 27227.000000 27227.000000  2.722700e+04 27227.000000 27227.000000 27227.000000 27227.000000
mean      60.315613    61.188853    59.626940  2.468755e+08  2001.267014     6.538987    15.755133     1.559922
std      111.856381   113.039237   110.413412  1.077167e+09    10.460180     3.410273     8.744898     1.410846
min        0.000000     0.218750     0.216146  0.000000e+00  1980.000000     1.000000     1.000000     0.000000
25%        3.430000     3.718750     3.593750  3.671470e+06  1993.000000     4.000000     8.000000     0.000000
50%       10.300000    10.500000    10.062500  2.615800e+07  2003.000000     7.000000    16.000000     2.000000
75%       26.700001    26.990000    26.360001  6.002380e+07  2010.000000     9.000000    23.000000     2.000000
max      567.667419   575.104126   547.836243  2.833812e+10  2023.000000    12.000000    31.000000     4.000000
```

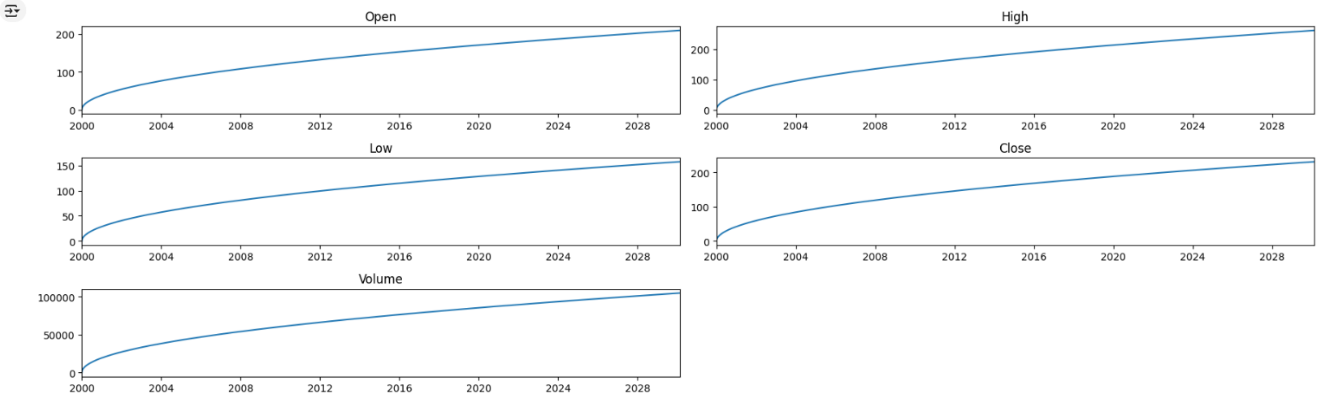


```
x_test.describe(include='all')
```

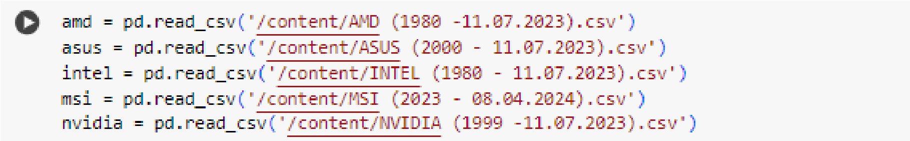
	Open	High	Low	Volume	Year	Month	Day	Company
count	6809.000000	6809.000000	6809.000000	6.809000e+03	6809.000000	6809.000000	6809.000000	6809.000000
mean	100.937076	102.317686	99.529994	3.692865e+07	2019.400646	6.469526	15.746071	1.559994
std	100.764061	101.881577	99.596936	3.175757e+07	2.417713	3.470243	8.746159	1.410873
min	1.620000	1.690000	1.610000	0.000000e+00	2014.000000	1.000000	1.000000	0.000000
25%	32.799999	33.240002	32.430000	1.586950e+07	2018.000000	3.000000	8.000000	0.000000
50%	53.209999	53.919998	52.410000	3.059510e+07	2020.000000	6.000000	16.000000	2.000000
75%	151.850006	154.660004	148.830002	5.063920e+07	2021.000000	10.000000	23.000000	2.000000
max	435.010010	439.899994	426.739990	3.250584e+08	2024.000000	12.000000	31.000000	4.000000



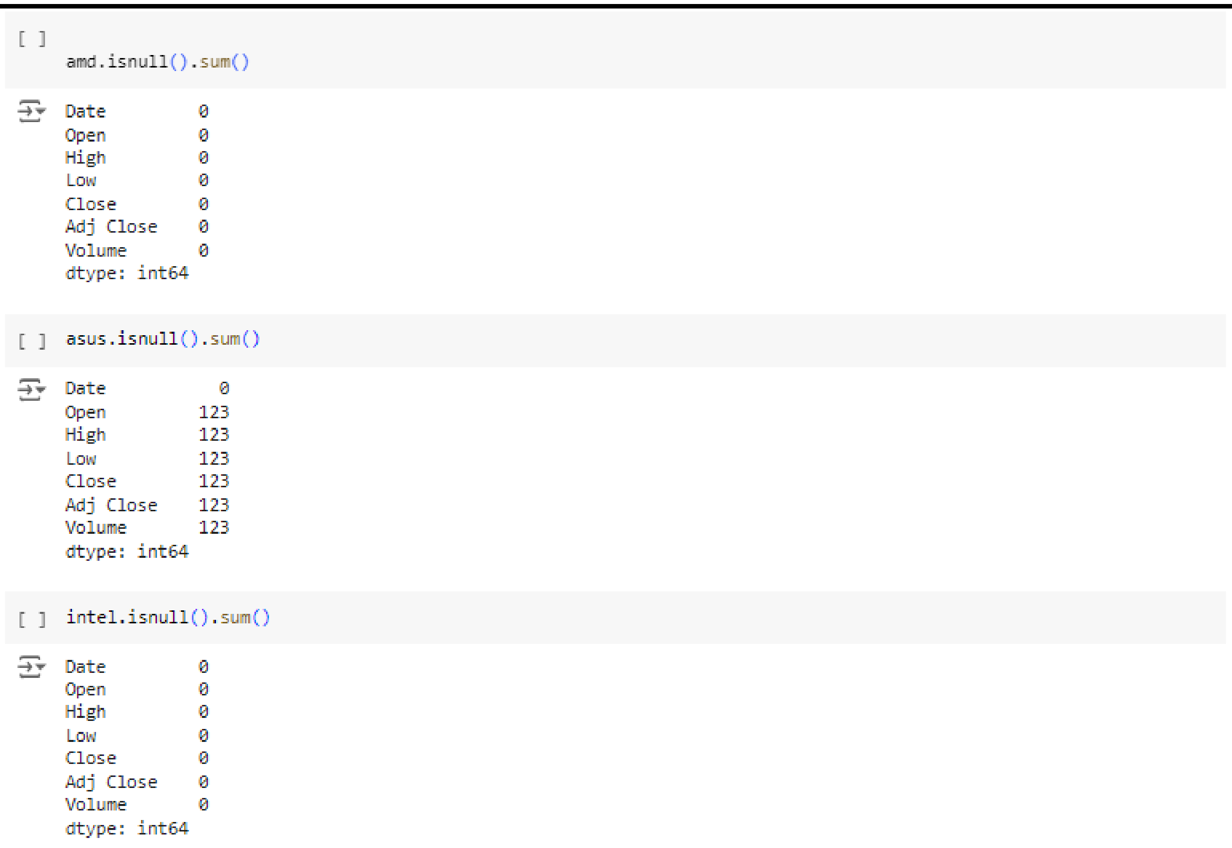
```
import matplotlib.pyplot as plt
import pandas as pd

# Generate sample data
dates = pd.date_range(start='1/1/2000', periods=11000)
data = pd.DataFrame({
    'Date': dates,
    'Open': (pd.Series(range(11000)) ** 0.5) * 2,  # Example data for 'Open'
    'High': (pd.Series(range(11000)) ** 0.5) * 2.5,  # Example data for 'High'
    'Low': (pd.Series(range(11000)) ** 0.5) * 1.5,  # Example data for 'Low'
    'Close': (pd.Series(range(11000)) ** 0.5) * 2.2,  # Example data for 'Close'
    'Volume': (pd.Series(range(11000)) ** 0.5) * 1000  # Example data for 'Volume'
})

fig, axs = plt.subplots(5, 2, figsize=(18, 9))

# List of column names to be plotted
columns_to_plot = ['Open', 'High', 'Low', 'Close', 'Volume']
                    list: columns_to_plot
# Plot each dat
for i, ax in en     (5 items) ['Open', 'High', 'Low', 'Close', 'Volume']
    if i < len(columns_to_plot):
        ax.plot(data['Date'], data[columns_to_plot[i]])
        ax.set_title(columns_to_plot[i])
        ax.set_xlim([data['Date'].min(), data['Date'].max()])
    else:
        ax.axis('off')

plt.tight_layout()
plt.show()
```



Loading Data



```
amd = pd.read_csv('/content/AMD (1980 -11.07.2023).csv')
asus = pd.read_csv('/content/ASUS (2000 - 11.07.2023).csv')
intel = pd.read_csv('/content/INTEL (1980 - 11.07.2023).csv')
msi = pd.read_csv('/content/MSI (2023 - 08.04.2024).csv')
nvidia = pd.read_csv('/content/NVIDIA (1999 -11.07.2023).csv')
```

<table>
<tr>
<td></td>
<td>Handling Missing Data</td>
<td>


```
[ ]
amd.isnull().sum()
```

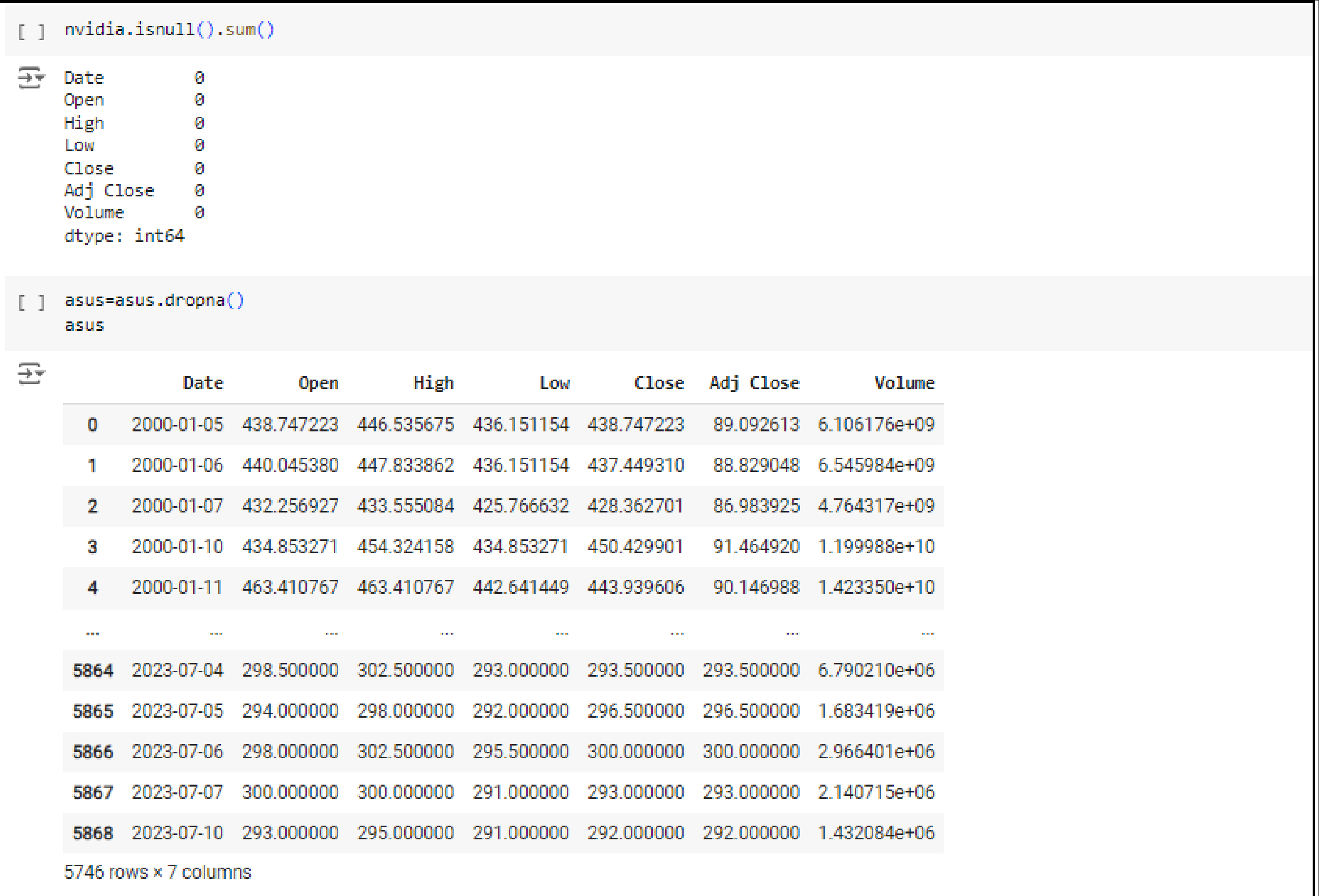
```
Date         0
Open         0
High         0
Low          0
Close        0
Adj Close    0
Volume       0
dtype: int64
```

```
[ ] asus.isnull().sum()
```

```
Date           0
Open         123
High         123
Low          123
Close        123
Adj Close    123
Volume       123
dtype: int64
```

```
[ ] intel.isnull().sum()
```

```
Date         0
Open         0
High         0
Low          0
Close        0
Adj Close    0
Volume       0
dtype: int64
```
</td>
</tr>
</table>



```
[ ] nvidia.isnull().sum()
```

```
Date         0
Open         0
High         0
Low          0
Close        0
Adj Close    0
Volume       0
dtype: int64
```

```
[ ] asus=asus.dropna()
    asus
```

	Date	Open	High	Low	Close	Adj Close	Volume
0	2000-01-05	438.747223	446.535675	436.151154	438.747223	89.092613	6.106176e+09
1	2000-01-06	440.045380	447.833862	436.151154	437.449310	88.829048	6.545984e+09
2	2000-01-07	432.256927	433.555084	425.766632	428.362701	86.983925	4.764317e+09
3	2000-01-10	434.853271	454.324158	434.853271	450.429901	91.464920	1.199988e+10
4	2000-01-11	463.410767	463.410767	442.641449	443.939606	90.146988	1.423350e+10
...	...	...	...	...	...	...	...
5864	2023-07-04	298.500000	302.500000	293.000000	293.500000	293.500000	6.790210e+06
5865	2023-07-05	294.000000	298.000000	292.000000	296.500000	296.500000	1.683419e+06
5866	2023-07-06	298.000000	302.500000	295.500000	300.000000	300.000000	2.966401e+06
5867	2023-07-07	300.000000	300.000000	291.000000	293.000000	293.000000	2.140715e+06
5868	2023-07-10	293.000000	295.000000	291.000000	292.000000	292.000000	1.432084e+06

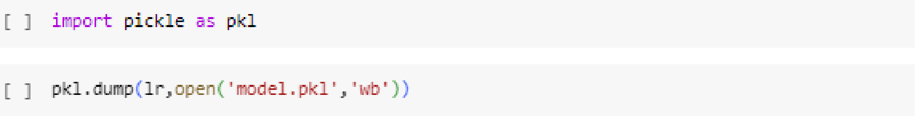
5746 rows × 7 columns

Feature Engineering

Attached the codes in final submission.

Save Processed Data

-



```
[ ] import pickle as pkl
```

```
[ ] pkl.dump(lr,open('model.pkl','wb'))
```