MARUTI STOCK EXCHANGE (2003-2021)

Partial fulfilment for the award of the
Post Graduate Certification
in Data Analytics for Engineers
September 2021

SUBMITTED BY,

RIZAMOL

VISMAYA V

SHEREENA S

ABSTRACT

Maruti Udyog Limited was founded by the Government of India on 24 Feb 1981 with Suzuki Motor Corporation as a minor partner, only to become the formal JV partner and license holder of Suzuki in August 2021. The first manufacturing factory of Maruti was established in Gurugram, Haryana, in the same year.

Maruti Suzuki Limited is the Number 1 automobile company in India, commanding a market share of more than 50% in the 4-wheeler segment. One in every two cars you see on the road today is a Maruti Suzuki. This domination has made Maruti the 14th most-valued company in the Indian stock market.

To analyse Maruti Stock Exchange, we use different tools. We used a data set from Kaggle. In this analysis, we cover the journey of the iconic company from the time it was first listed in 2003 to 2021.

Tools used

- **Python**
- **R**
- **Excel**
- **Tableau**

CONTENTS

1. INTRODUCTION

Maruti Suzuki India Limited (MSIL), formerly known as Maruti Udyog Limited, a subsidiary of Suzuki Motor Corporation of Japan, is India's largest passenger car company, accounting for over 50 per cent of the domestic car market. Maruti Udyog Limited was incorporated in 1981 under the provisions of Indian Companies Act 1956 and the government of India selected Suzuki Motor Corporation as the joint venture partner for the company. In 1982 a JV was signed between Government of India and Suzuki Motor Corporation.

2. ANALYSIS

2. 1. PYTHON

Jupyter notebook is used for analysis of data set. Firstly, import all libraries. Then read the data set and done EDA analysis. Then done Machine learning. We used linear regression for ML approach.

Library

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns


import warnings

warnings.filterwarnings('ignore')

%matplotlib inline
```

Read data set

```
d = pd.read_csv("Maruti.csv")
```

Screen shots

[6]:

	Date	Symbol	Series	Prev Close	Open	High	Low	Last	Close	VWAP	Volume	Turnover	Trades	Deliverable Volume	%Deliverble
0	2003-07-09	MARUTI	EQ	125.00	164.90	170.40	155.00	164.00	164.30	165.95	35164283	5.835528e+14	NaN	8537695.0	0.2428
1	2003-07-10	MARUTI	EQ	164.30	167.00	168.70	164.50	167.00	167.00	166.74	10464179	1.744820e+14	NaN	4363947.0	0.4170
2	2003-07-11	MARUTI	EQ	167.00	167.75	174.85	166.25	173.60	173.35	172.45	11740117	2.024622e+14	NaN	3014852.0	0.2568
3	2003-07-14	MARUTI	EQ	173.35	174.25	179.25	174.25	178.60	177.95	177.91	5982324	1.064313e+14	NaN	1949217.0	0.3258
4	2003-07-15	MARUTI	EQ	177.95	200.00	200.00	173.00	176.30	176.20	176.88	6173689	1.092001e+14	NaN	1307694.0	0.2118
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
4422	2021-04-26	MARUTI	EQ	6676.10	6690.20	6789.00	6600.00	6645.00	6638.90	6678.34	937344	6.259903e+14	74474.0	464999.0	0.4961
4423	2021-04-27	MARUTI	EQ	6638.90	6669.95	6709.00	6542.00	6552.00	6568.75	6620.68	1610651	1.066360e+15	130986.0	588617.0	0.3655
4424	2021-04-28	MARUTI	EQ	6568.75	6568.75	6650.00	6545.00	6581.00	6573.80	6598.62	1406270	9.279437e+14	117843.0	672435.0	0.4782
4425	2021-04-29	MARUTI	EQ	6573.80	6635.00	6647.45	6552.00	6562.00	6565.65	6580.77	757075	4.982135e+14	64393.0	352987.0	0.4663
4426	2021-04-30	MARUTI	EQ	6565.65	6537.10	6559.60	6421.00	6438.35	6455.65	6500.51	849997	5.525418e+14	95248.0	382594.0	0.4501

4427 rows × 15 columns

To print head row

In [4]:
```
1 d.head()
```

Out[4]:

	Date	Symbol	Series	Prev Close	Open	High	Low	Last	Close	VWAP	Volume	Turnover	Trades	Deliverable Volume	%Deliverble
0	2003-07-09	MARUTI	EQ	125.00	164.90	170.40	155.00	164.0	164.30	165.95	35164283	5.835528e+14	NaN	8537695.0	0.2428
1	2003-07-10	MARUTI	EQ	164.30	167.00	168.70	164.50	167.0	167.00	166.74	10464179	1.744820e+14	NaN	4363947.0	0.4170
2	2003-07-11	MARUTI	EQ	167.00	167.75	174.85	166.25	173.6	173.35	172.45	11740117	2.024622e+14	NaN	3014852.0	0.2568
3	2003-07-14	MARUTI	EQ	173.35	174.25	179.25	174.25	178.6	177.95	177.91	5982324	1.064313e+14	NaN	1949217.0	0.3258
4	2003-07-15	MARUTI	EQ	177.95	200.00	200.00	173.00	176.3	176.20	176.88	6173689	1.092001e+14	NaN	1307694.0	0.2118

To print tail row

In [5]:
```
1 d.tail()
```

Out[5]:

	Date	Symbol	Series	Prev Close	Open	High	Low	Last	Close	VWAP	Volume	Turnover	Trades	Deliverable Volume	%Deliverble
4422	2021-04-26	MARUTI	EQ	6676.10	6690.20	6789.00	6600.0	6645.00	6638.90	6678.34	937344	6.259903e+14	74474.0	464999.0	0.4961
4423	2021-04-27	MARUTI	EQ	6638.90	6669.95	6709.00	6542.0	6552.00	6568.75	6620.68	1610651	1.066360e+15	130986.0	588617.0	0.3655
4424	2021-04-28	MARUTI	EQ	6568.75	6568.75	6650.00	6545.0	6581.00	6573.80	6598.62	1406270	9.279437e+14	117843.0	672435.0	0.4782
4425	2021-04-29	MARUTI	EQ	6573.80	6635.00	6647.45	6552.0	6562.00	6565.65	6580.77	757075	4.982135e+14	64393.0	352987.0	0.4663
4426	2021-04-30	MARUTI	EQ	6565.65	6537.10	6559.60	6421.0	6438.35	6455.65	6500.51	849997	5.525418e+14	95248.0	382594.0	0.4501

Check data types

In [6]:
```
1 d.dtypes
```

Out[6]:
```
Date                 object
Symbol               object
Series               object
Prev Close          float64
Open                float64
High                float64
Low                 float64
Last                float64
Close               float64
VWAP                float64
Volume                int64
Turnover            float64
Trades              float64
Deliverable Volume  float64
%Deliverble         float64
dtype: object
```

Size of data set

In [7]:
```
1 d.size
```

Out[7]: 66405

To print number of rows and columns

In [8]:
```
1 d.shape
```

Out[8]: (4427, 15)

To print dimensions of dataset

In [9]:
```
1 d.ndim
```

Out[9]: 2

To check whether any data value is null

In [10]:
```
1 d.isna()
```

Out[10]:

	Date	Symbol	Series	Prev Close	Open	High	Low	Last	Close	VWAP	Volume	Turnover	Trades	Deliverable Volume	%Deliverble
0	False	False	False	False	False	False	False	False	False	False	False	False	True	False	False
1	False	False	False	False	False	False	False	False	False	False	False	False	True	False	False
2	False	False	False	False	False	False	False	False	False	False	False	False	True	False	False
3	False	False	False	False	False	False	False	False	False	False	False	False	True	False	False
4	False	False	False	False	False	False	False	False	False	False	False	False	True	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
4422	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
4423	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
4424	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
4425	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
4426	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False

4427 rows × 15 columns

Sum of null values

In [11]:
```
d.isna().sum()
```

Out[11]:
```
Date                    0
Symbol                  0
Series                  0
Prev Close              0
Open                    0
High                    0
Low                     0
Last                    0
Close                   0
VWAP                    0
Volume                  0
Turnover                0
Trades               1971
Deliverable Volume      1
%Deliverble             1
dtype: int64
```

To check any duplicated value

In [12]:
```
d.duplicated()
```

Out[12]:
```
0       False
1       False
2       False
3       False
4       False
        ...
4422    False
4423    False
4424    False
4425    False
4426    False
Length: 4427, dtype: bool
```

In [13]:
```
d.duplicated().sum()
```

Out[13]:
```
0
```

To check high share value

In [14]:
```
d['High'].max()
```

Out[14]:
```
9996.4
```

Sort basis of high share values

In [15]:
```
d.sort_values(by="High",ascending=False)
```

Out[15]:

	Date	Symbol	Series	Prev Close	Open	High	Low	Last	Close	VWAP	Volume	Turnover	Trades	Deliverable Volume	%Deliverble
3597	2017-12-20	MARUTI	EQ	9801.50	9966.0	9996.40	9700.25	9755.4	9733.90	9848.56	1216280	1.197861e+15	127449.0	325474.0	0.2676
3744	2018-07-24	MARUTI	EQ	9701.00	9778.0	9929.00	9725.00	9842.0	9832.45	9850.96	749626	7.384534e+14	96801.0	307586.0	0.4103
3745	2018-07-25	MARUTI	EQ	9832.45	9853.0	9875.85	9701.00	9769.0	9758.95	9770.80	384750	3.759315e+14	51181.0	130726.0	0.3398
3596	2017-12-19	MARUTI	EQ	9309.10	9345.0	9858.00	9326.00	9846.0	9801.50	9603.52	1275759	1.225177e+15	113969.0	369478.0	0.2896
3746	2018-07-26	MARUTI	EQ	9758.95	9819.8	9831.70	9361.00	9390.0	9396.25	9510.08	1261317	1.199523e+15	151687.0	380582.0	0.3017
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
7	2003-07-18	MARUTI	EQ	172.20	165.0	171.90	160.00	167.4	167.40	167.66	5375212	9.011903e+13	NaN	1427995.0	0.2657
0	2003-07-09	MARUTI	EQ	125.00	164.9	170.40	155.00	164.0	164.30	165.95	35164283	5.835528e+14	NaN	8537695.0	0.2428
8	2003-07-21	MARUTI	EQ	167.40	169.0	169.40	163.05	164.0	164.90	166.67	3315845	5.526646e+13	NaN	980379.0	0.2957
9	2003-07-22	MARUTI	EQ	164.90	164.0	169.15	162.10	168.6	167.25	165.06	4627992	7.639124e+13	NaN	1257167.0	0.2716
1	2003-07-10	MARUTI	EQ	164.30	167.0	168.70	164.50	167.0	167.00	166.74	10464179	1.744820e+14	NaN	4363947.0	0.4170

Sort basis of Turnover

In [16]:
```
d.sort_values(by="Low",ascending=False)
```

Out[16]:

	Date	Symbol	Series	Prev Close	Open	High	Low	Last	Close	VWAP	Volume	Turnover	Trades	Deliverable Volume	%Deliverble
3744	2018-07-24	MARUTI	EQ	9701.00	9778.0	9929.00	9725.00	9842.0	9832.45	9850.96	749626	7.384534e+14	96801.0	307586.0	0.4103
3745	2018-07-25	MARUTI	EQ	9832.45	9853.0	9875.85	9701.00	9769.0	9758.95	9770.80	384750	3.759315e+14	51181.0	130726.0	0.3398
3597	2017-12-20	MARUTI	EQ	9801.50	9966.0	9996.40	9700.25	9755.4	9733.90	9848.56	1216280	1.197861e+15	127449.0	325474.0	0.2676
3603	2017-12-29	MARUTI	EQ	9629.20	9659.0	9769.00	9650.00	9735.3	9729.55	9717.74	471554	4.582438e+14	41973.0	134352.0	0.2849
3604	2018-01-01	MARUTI	EQ	9729.55	9749.0	9789.00	9629.80	9644.0	9651.90	9708.58	426354	4.139291e+14	37440.0	119356.0	0.2799
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1	2003-07-10	MARUTI	EQ	164.30	167.0	168.70	164.50	167.0	167.00	166.74	10464179	1.744820e+14	NaN	4363947.0	0.4170
8	2003-07-21	MARUTI	EQ	167.40	169.0	169.40	163.05	164.0	164.90	166.67	3315845	5.526646e+13	NaN	980379.0	0.2957
9	2003-07-22	MARUTI	EQ	164.90	164.0	169.15	162.10	168.6	167.25	165.06	4627992	7.639124e+13	NaN	1257167.0	0.2716
7	2003-07-18	MARUTI	EQ	172.20	165.0	171.90	160.00	167.4	167.40	167.66	5375212	9.011903e+13	NaN	1427995.0	0.2657
0	2003-07-09	MARUTI	EQ	125.00	164.9	170.40	155.00	164.0	164.30	165.95	35164283	5.835528e+14	NaN	8537695.0	0.2428

4427 rows × 15 columns

Sort value basis of Low Share values

In [17]:
```
d.sort_values(by="Low",ascending=False)
```

Out[17]:

	Date	Symbol	Series	Prev Close	Open	High	Low	Last	Close	VWAP	Volume	Turnover	Trades	Deliverable Volume	%Deliverble
3744	2018-07-24	MARUTI	EQ	9701.00	9778.0	9929.00	9725.00	9842.0	9832.45	9850.96	749626	7.384534e+14	96801.0	307586.0	0.4103
3745	2018-07-25	MARUTI	EQ	9832.45	9853.0	9875.85	9701.00	9769.0	9758.95	9770.80	384750	3.759315e+14	51181.0	130726.0	0.3398
3597	2017-12-20	MARUTI	EQ	9801.50	9966.0	9996.40	9700.25	9755.4	9733.90	9848.56	1216280	1.197861e+15	127449.0	325474.0	0.2676
3603	2017-12-29	MARUTI	EQ	9629.20	9659.0	9769.00	9650.00	9735.3	9729.55	9717.74	471554	4.582438e+14	41973.0	134352.0	0.2849
3604	2018-01-01	MARUTI	EQ	9729.55	9749.0	9789.00	9629.80	9644.0	9651.90	9708.58	426354	4.139291e+14	37440.0	119356.0	0.2799
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1	2003-07-10	MARUTI	EQ	164.30	167.0	168.70	164.50	167.0	167.00	166.74	10464179	1.744820e+14	NaN	4363947.0	0.4170
8	2003-07-21	MARUTI	EQ	167.40	169.0	169.40	163.05	164.0	164.90	166.67	3315845	5.526646e+13	NaN	980379.0	0.2957
9	2003-07-22	MARUTI	EQ	164.90	164.0	169.15	162.10	168.6	167.25	165.06	4627992	7.639124e+13	NaN	1257167.0	0.2716
7	2003-07-18	MARUTI	EQ	172.20	165.0	171.90	160.00	167.4	167.40	167.66	5375212	9.011903e+13	NaN	1427995.0	0.2657
0	2003-07-09	MARUTI	EQ	125.00	164.9	170.40	155.00	164.0	164.30	165.95	35164283	5.835528e+14	NaN	8537695.0	0.2428

4427 rows × 15 columns

To print High share value row set ¶

In [18]:
```
a = d["High"].max()
a
```

Out[18]: 9996.4

In [19]:
```
d_max = d.loc[(d['High']==a)]
d_max
```

Out[19]:

	Date	Symbol	Series	Prev Close	Open	High	Low	Last	Close	VWAP	Volume	Turnover	Trades	Deliverable Volume	%Deliverble
3597	2017-12-20	MARUTI	EQ	9801.5	9966.0	9996.4	9700.25	9755.4	9733.9	9848.56	1216280	1.197861e+15	127449.0	325474.0	0.2676

To print mean, median, mode of data set

In [20]:
```
d.mean()
```

Out[20]:
```
Prev Close            2.923575e+03
Open                  2.927873e+03
High                  2.962918e+03
Low                   2.889128e+03
Last                  2.924652e+03
Close                 2.925005e+03
VWAP                  2.926481e+03
Volume                1.194661e+06
Turnover              2.395307e+14
Trades                5.542851e+04
Deliverable Volume    3.627677e+05
%Deliverble           3.873908e-01
dtype: float64
```

In [21]:
```
d.mode()
```

Out[21]:

	Date	Symbol	Series	Prev Close	Open	High	Low	Last	Close	VWAP	Volume	Turnover	Trades	Deliverable Volume	%Deliverble
0	2003-07-09	MARUTI	EQ	418.95	765.0	954.0	825.0	800.0	418.95	377.51	184121.0	2.131518e+12	10514.0	85105.0	0.3404
1	2003-07-10	NaN	NaN	NaN	NaN	NaN	920.0	NaN	NaN	414.79	261295.0	2.606075e+12	11304.0	93889.0	NaN
2	2003-07-11	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	423.64	328108.0	3.365101e+12	11607.0	96440.0	NaN
3	2003-07-14	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	461.96	420913.0	4.842874e+12	12463.0	188286.0	NaN
4	2003-07-15	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	481.03	679821.0	4.961826e+12	14982.0	192329.0	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
4422	2021-04-26	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2.164002e+15	NaN	NaN	NaN
4423	2021-04-27	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2.264809e+15	NaN	NaN	NaN
4424	2021-04-28	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2.913613e+15	NaN	NaN	NaN
4425	2021-04-29	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	3.187016e+15	NaN	NaN	NaN
4426	2021-04-30	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	4.878992e+15	NaN	NaN	NaN

4427 rows × 15 columns

In [22]:
```
d.median()
```

Out[22]:
```
Prev Close            1.412450e+03
Open                  1.414000e+03
High                  1.432000e+03
Low                   1.390350e+03
Last                  1.412200e+03
Close                 1.412600e+03
VWAP                  1.412210e+03
Volume                6.909590e+05
Turnover              1.121591e+14
Trades                4.403150e+04
```

dtype: float64

To print information of data set

n [23]:
```
1 d.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4427 entries, 0 to 4426
Data columns (total 15 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Date                4427 non-null   object
 1   Symbol              4427 non-null   object
 2   Series              4427 non-null   object
 3   Prev Close          4427 non-null   float64
 4   Open                4427 non-null   float64
 5   High                4427 non-null   float64
 6   Low                 4427 non-null   float64
 7   Last                4427 non-null   float64
 8   Close               4427 non-null   float64
 9   VWAP                4427 non-null   float64
 10  Volume              4427 non-null   int64
 11  Turnover            4427 non-null   float64
 12  Trades              2456 non-null   float64
 13  Deliverable Volume  4426 non-null   float64
 14  %Deliverble         4426 non-null   float64
dtypes: float64(11), int64(1), object(3)
memory usage: 518.9+ KB
```

To print index values

In [24]:
```
1 d.index
```

Out[24]:
```
RangeIndex(start=0, stop=4427, step=1)
```

To print number of unique value in the data set

In [25]:
```
1 d.nunique()
```

Out[25]:
```
Date                4427
Symbol                 1
Series                 1
Prev Close          4296
Open                3404
High                3860
Low                 3897
Last                3788
Close               4296
VWAP                4397
Volume              4421
Turnover            4427
Trades              2418
Deliverable Volume  4406
%Deliverble         3132
dtype: int64
```

In [26]:
```
1 d.nunique().sum()
```

Out[26]:
```
51171
```

Check relationship between data set

In [27]:
```
1 d.corr()
```

Out[27]:

	Prev Close	Open	High	Low	Last	Close	VWAP	Volume	Turnover	Trades	Deliverable Volume	%Deliverble
Prev Close	1.000000	0.999940	0.999819	0.999789	0.999665	0.999677	0.999820	-0.245643	0.649851	0.464892	-0.194373	0.060529
Open	0.999940	1.000000	0.999842	0.999847	0.999696	0.999709	0.999856	-0.245440	0.649903	0.464263	-0.194295	0.060458
High	0.999819	0.999842	1.000000	0.999771	0.999876	0.999884	0.999940	-0.243687	0.655949	0.474093	-0.191977	0.058686
Low	0.999789	0.999847	0.999771	1.000000	0.999837	0.999849	0.999920	-0.246699	0.646019	0.456209	-0.196509	0.060714
Last	0.999665	0.999696	0.999876	0.999837	1.000000	0.999995	0.999943	-0.245025	0.651448	0.465550	-0.194335	0.059028
Close	0.999677	0.999709	0.999884	0.999849	0.999995	1.000000	0.999953	-0.244997	0.651590	0.465683	-0.194240	0.059002
VWAP	0.999820	0.999856	0.999940	0.999920	0.999943	0.999953	1.000000	-0.245102	0.651271	0.465281	-0.194284	0.059298
Volume	-0.245643	-0.245440	-0.243687	-0.246699	-0.245025	-0.244997	-0.245102	1.000000	0.201224	0.778248	0.718711	-0.388508
Turnover	0.649851	0.649903	0.655949	0.646019	0.651448	0.651590	0.651271	0.201224	1.000000	0.914480	0.270873	-0.134744
Trades	0.464892	0.464263	0.474093	0.456209	0.465550	0.465683	0.465281	0.778248	0.914480	1.000000	0.488334	-0.206539
Deliverable Volume	-0.194373	-0.194295	-0.191977	-0.196509	-0.194335	-0.194240	-0.194284	0.718711	0.270873	0.488334	1.000000	0.113439
%Deliverble	0.060529	0.060458	0.058686	0.060714	0.059028	0.059002	0.059298	-0.388508	-0.134744	-0.206539	0.113439	1.000000

To print Columns

In [28]:
```
d.columns
```

Out[28]:
```
Index(['Date', 'Symbol', 'Series', 'Prev Close', 'Open', 'High', 'Low', 'Last',
       'Close', 'VWAP', 'Volume', 'Turnover', 'Trades', 'Deliverable Volume',
       '%Deliverble'],
      dtype='object')
```

Filter date column

In [29]:
```
d.filter(["Date"])
```

Out[29]:

	Date
0	2003-07-09
1	2003-07-10
2	2003-07-11
3	2003-07-14
4	2003-07-15
...	...
4422	2021-04-26
4423	2021-04-27
4424	2021-04-28
4425	2021-04-29
4426	2021-04-30

4427 rows × 1 columns

In [30]:
```
year = d.filter(["Date"])
year
```

Out[30]:

	Date
0	2003-07-09
1	2003-07-10
2	2003-07-11
3	2003-07-14
4	2003-07-15
...	...
4422	2021-04-26
4423	2021-04-27
4424	2021-04-28
4425	2021-04-29
4426	2021-04-30

4427 rows × 1 columns

To show Years from date column

In [31]:
```
pd.DatetimeIndex(year['Date']).year
```

Out[31]:
```
Int64Index([2003, 2003, 2003, 2003, 2003, 2003, 2003, 2003, 2003, 2003,
            ...
            2021, 2021, 2021, 2021, 2021, 2021, 2021, 2021, 2021, 2021],
           dtype='int64', name='Date', length=4427)
```

Add Year Column to data set

In [32]:
```
d["Year"] = pd.DatetimeIndex(year['Date']).year
d
```

Out[32]:

	Date	Symbol	Series	Prev Close	Open	High	Low	Last	Close	VWAP	Volume	Turnover	Trades	Deliverable Volume	%Deliverble	Year
0	2003-07-09	MARUTI	EQ	125.00	164.90	170.40	155.00	164.00	164.30	165.95	35164283	5.835528e+14	NaN	8537695.0	0.2428	2003
1	2003-07-10	MARUTI	EQ	164.30	167.00	168.70	164.50	167.00	167.00	166.74	10464179	1.744820e+14	NaN	4363947.0	0.4170	2003
2	2003-07-11	MARUTI	EQ	167.00	167.75	174.85	166.25	173.60	173.35	172.45	11740117	2.024622e+14	NaN	3014852.0	0.2568	2003
3	2003-07-14	MARUTI	EQ	173.35	174.25	179.25	174.25	178.60	177.95	177.91	5982324	1.064313e+14	NaN	1949217.0	0.3258	2003
4	2003-07-15	MARUTI	EQ	177.95	200.00	200.00	173.00	176.30	176.20	176.88	6173689	1.092001e+14	NaN	1307694.0	0.2118	2003
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
4422	2021-04-26	MARUTI	EQ	6676.10	6690.20	6789.00	6600.00	6645.00	6638.90	6678.34	937344	6.259903e+14	74474.0	464999.0	0.4961	2021

4427 rows × 16 columns

To add a column of Month for printing particular month from date column

In [33]:
```
pd.DatetimeIndex(year['Date']).month
```

Out[33]:
```
Int64Index([7, 7, 7, 7, 7, 7, 7, 7, 7, 7,
            ...
            4, 4, 4, 4, 4, 4, 4, 4, 4, 4],
           dtype='int64', name='Date', length=4427)
```

In [34]:
```
d["Month"]  = pd.DatetimeIndex(year['Date']).month
d
```

Out[34]:

	Date	Symbol	Series	Prev Close	Open	High	Low	Last	Close	VWAP	Volume	Turnover	Trades	Deliverable Volume	%Deliverble	Ye
0	2003-07-09	MARUTI	EQ	125.00	164.90	170.40	155.00	164.00	164.30	165.95	35164283	5.835528e+14	NaN	8537695.0	0.2428	200
1	2003-07-10	MARUTI	EQ	164.30	167.00	168.70	164.50	167.00	167.00	166.74	10464179	1.744820e+14	NaN	4363947.0	0.4170	200
2	2003-07-11	MARUTI	EQ	167.00	167.75	174.85	166.25	173.60	173.35	172.45	11740117	2.024622e+14	NaN	3014852.0	0.2568	200
3	2003-07-14	MARUTI	EQ	173.35	174.25	179.25	174.25	178.60	177.95	177.91	5982324	1.064313e+14	NaN	1949217.0	0.3258	200
4	2003-07-15	MARUTI	EQ	177.95	200.00	200.00	173.00	176.30	176.20	176.88	6173689	1.092001e+14	NaN	1307694.0	0.2118	200
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
4422	2021-04-26	MARUTI	EQ	6676.10	6690.20	6789.00	6600.00	6645.00	6638.90	6678.34	937344	6.259903e+14	74474.0	464999.0	0.4961	202

To print column names

In [35]:
```
d.columns
```

Out[35]:
```
Index(['Date', 'Symbol', 'Series', 'Prev Close', 'Open', 'High', 'Low', 'Last',
       'Close', 'VWAP', 'Volume', 'Turnover', 'Trades', 'Deliverable Volume',
       '%Deliverble', 'Year', 'Month'],
      dtype='object')
```

Extract data basis Year 2021

In [36]:
```
d_Year_2021 = d.loc[(d["Year"]== 2021 )]
d_Year_2021
```

Out[36]:

	Date	Symbol	Series	Prev Close	Open	High	Low	Last	Close	VWAP	Volume	Turnover	Trades	Deliverable Volume	%Deliverble	Year	M
4347	2021-01-01	MARUTI	EQ	7649.60	7654.00	7748.50	7650.0	7687.00	7691.30	7704.92	767574	5.914094e+14	55693.0	93385.0	0.1217	2021	
4348	2021-01-04	MARUTI	EQ	7691.30	7739.00	7755.20	7642.6	7691.05	7702.30	7697.13	598645	4.607847e+14	55195.0	171586.0	0.2866	2021	
4349	2021-01-05	MARUTI	EQ	7702.30	7660.00	7673.00	7586.0	7657.20	7655.45	7635.41	562668	4.296203e+14	57426.0	147432.0	0.2620	2021	
4350	2021-01-06	MARUTI	EQ	7655.45	7654.00	7749.00	7555.5	7625.25	7628.60	7670.88	840678	6.448736e+14	66618.0	204373.0	0.2431	2021	
4351	2021-01-07	MARUTI	EQ	7628.60	7676.00	7704.40	7552.1	7576.85	7566.05	7620.30	642968	4.899611e+14	65629.0	198399.0	0.3086	2021	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
4422	2021-04-26	MARUTI	EQ	6676.10	6690.20	6789.00	6600.0	6645.00	6638.90	6678.34	937344	6.259903e+14	74474.0	464999.0	0.4961	2021	
4423	2021-	MARUTI	EQ	6638.90	6669.95	6709.00	6542.0	6552.00	6568.75	6620.68	1610651	1.066360e+15	130986.0	588617.0	0.3655	2021	

Check and drop a row of Deliverable row as NAN

In [37]:
```
d[d['Deliverable Volume'].isna()]
```

Out[37]:

	Date	Symbol	Series	Prev Close	Open	High	Low	Last	Close	VWAP	Volume	Turnover	Trades	Deliverable Volume	%Deliverble	Year	Month
76	2003-10-25	MARUTI	EQ	285.0	294.9	297.7	291.5	293.6	293.75	294.51	1888755	5.562562e+13	NaN	NaN	NaN	2003	10

In [38]:
```
a = d[d['Date']=='2003-10-25']
```

In [39]:
```
a
```

Out[39]:

	Date	Symbol	Series	Prev Close	Open	High	Low	Last	Close	VWAP	Volume	Turnover	Trades	Deliverable Volume	%Deliverble	Year	Month
76	2003-10-25	MARUTI	EQ	285.0	294.9	297.7	291.5	293.6	293.75	294.51	1888755	5.562562e+13	NaN	NaN	NaN	2003	10

In [40]:
```
d.drop(76,axis=0,inplace=True)
```

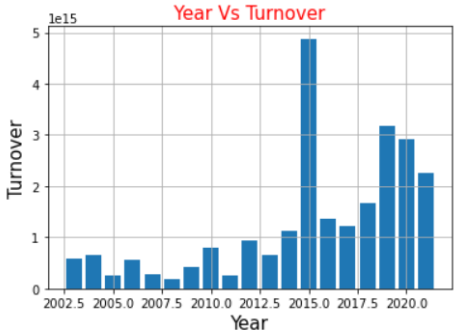
In [41]:
```
d.isna().sum()
```

Out[41]:
```
Date                 0
Symbol               0
Series               0
Prev Close           0
Open                 0
High                 0
Low                  0
Last                 0
Close                0
VWAP                 0
```

Visualization

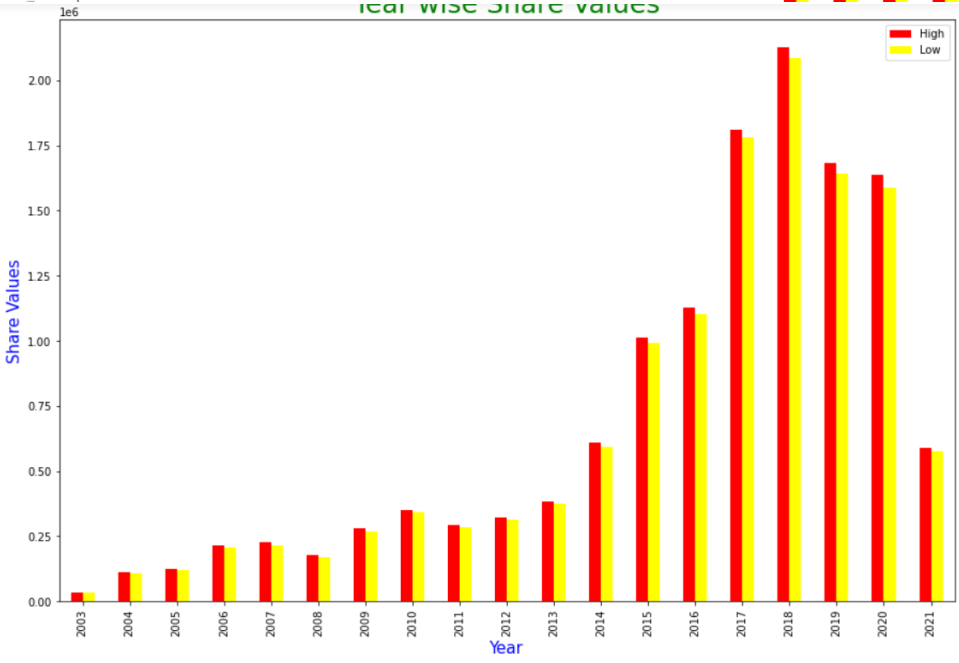
To plot total turnover

In [43]:
```
plt.bar(d['Year'],d['Turnover'])
plt.title('Year Vs Turnover',fontsize=15,color='red')
plt.xlabel('Year',fontsize=15)
plt.ylabel('Turnover',fontsize=15)
plt.grid(True)
plt.show()
```



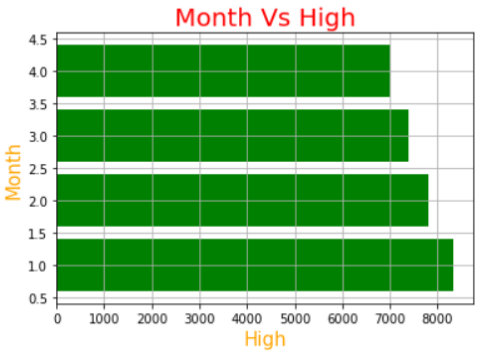
Year wise analysis of high and low shre share analysis

In [110]:
```
d.groupby('Year')[['High','Low']].sum().plot.bar(color=['red','Yellow'],figsize=(15,10))
plt.title("Year wise Share Values",color="Green",fontsize=25)
plt.ylabel('Share Values',fontsize=15,color="blue")
plt.xlabel('Year',fontsize=15,color='blue')
plt.show()
```



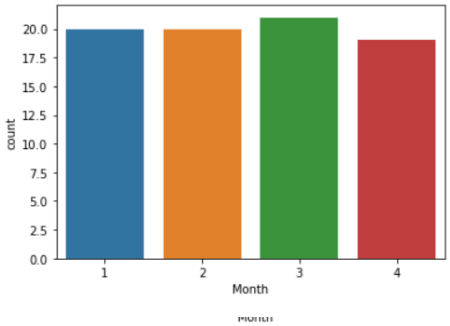
To show High share values in 2021

In [44]:
```
plt.barh(d_Year_2021['Month'],d_Year_2021['High'],color = 'green')
plt.title('Month Vs High',fontsize=20,color='red')
plt.xlabel('High',fontsize=15,color ='orange')
plt.ylabel('Month',fontsize=15, color = 'orange')
plt.grid(True)
plt.show()
```



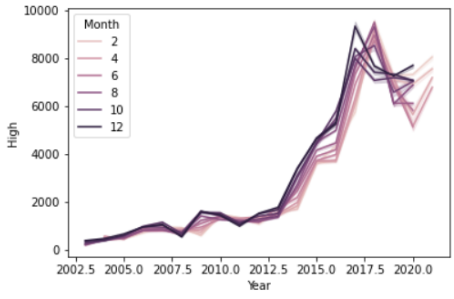
Month basis of Analysis in year 2021

In [39]:
```
sns.countplot(x="Month",data=d_Year_2021,palette='tab10')
plt.show()
```



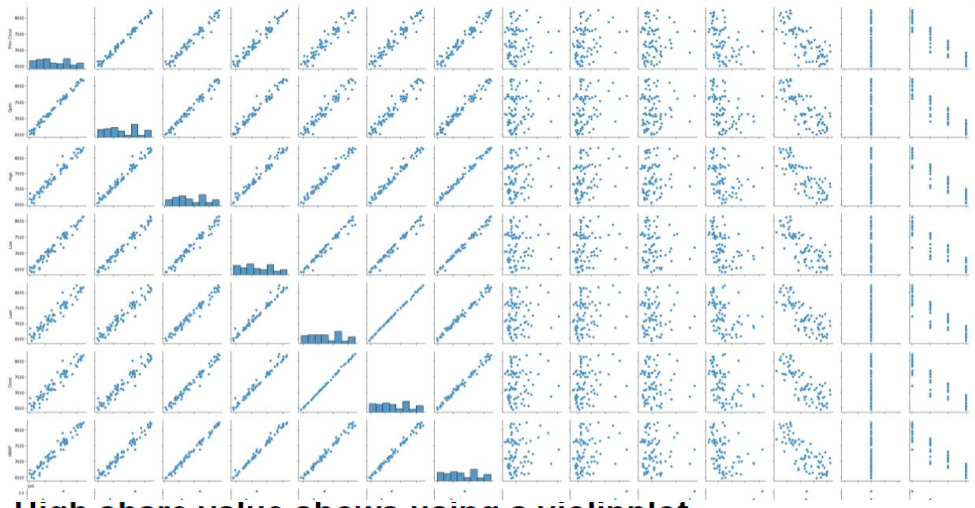
To plot a line plot basis High share values in different year

In [40]:
```
sns.lineplot(x="Year",y='High',hue='Month',data=d)
plt.show()
```



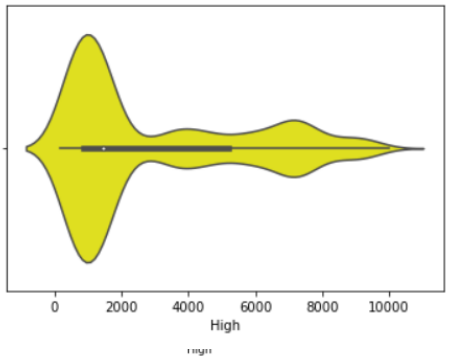
Show a relationship plot in year 2021

In [41]:
```
sns.pairplot(d_Year_2021)
plt.show()
```



High share value shows using a violinplot

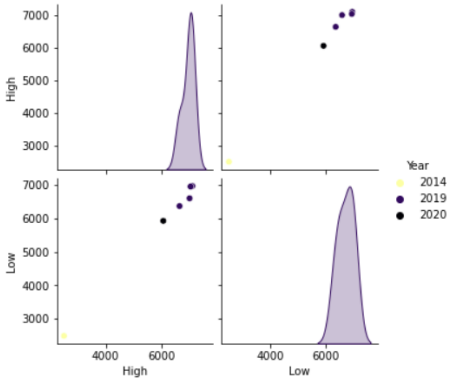
In [54]:
```
sns.violinplot(x=d.High,color='yellow')
plt.show()
```



Check the some share values basis of opening balance

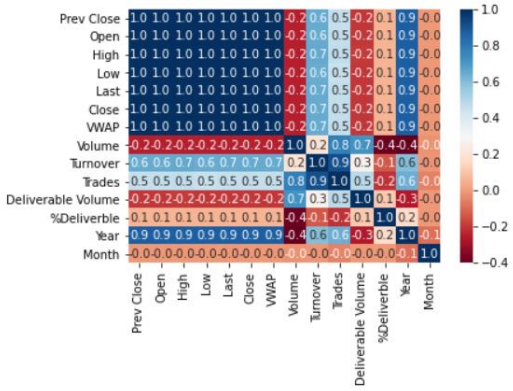
In [45]:
```
sns.pairplot(d[d.Open.isin([7000,6000,2500,7687,7657,6565])],
             hue='Year',
             vars=['High','Low'],
             palette='inferno_r')
```

Out[45]: <seaborn.axisgrid.PairGrid at 0x1e5f5515430>



To plot a heat map

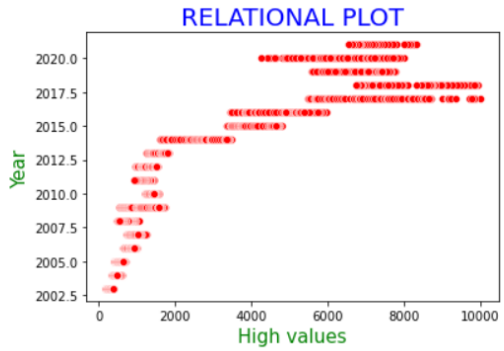
In [47]:
```
sns.heatmap(d.corr(),cmap='RdBu',annot=True,fmt='.1f')
plt.show()
```



Machine learning

To show a relationship of High value basis of Year

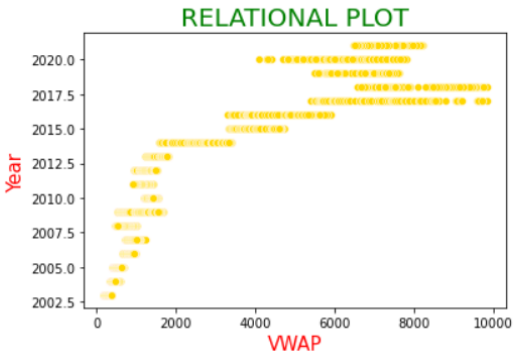
In [58]:
```
sns.scatterplot(y=d['Year'],x=d["High"],color='red')
plt.title("RELATIONAL PLOT",size=20,color='blue')
plt.ylabel("Year",size=15,color='green')
plt.xlabel("High values",size=15,color='green')
plt.show()
```



High values

To plot a relationship of VWAP basis of Year

In [59]:
```
sns.scatterplot(y=d['Year'],x=d["VWAP"],color='gold')  # Volume weighted average price
plt.title("RELATIONAL PLOT",size=20,color='green')
plt.ylabel("Year",size=15,color='red')
plt.xlabel("VWAP",size=15,color='red')
plt.show()
```



4426 rows × 17 columns

To drop Date column

In [42]:
```
d.drop('Date',axis=1,inplace=True)
```

In [43]:
```
d.head(10)
```

Out[43]:

	Symbol	Series	Prev Close	Open	High	Low	Last	Close	VWAP	Volume	Turnover	Trades	Deliverable Volume	%Deliverble	Year	Month
0	MARUTI	EQ	125.00	164.90	170.40	155.00	164.0	164.30	165.95	35164283	5.835528e+14	NaN	8537695.0	0.2428	2003	7
1	MARUTI	EQ	164.30	167.00	168.70	164.50	167.0	167.00	166.74	10464179	1.744820e+14	NaN	4363947.0	0.4170	2003	7
2	MARUTI	EQ	167.00	167.75	174.85	166.25	173.6	173.35	172.45	11740117	2.024622e+14	NaN	3014852.0	0.2568	2003	7
3	MARUTI	EQ	173.35	174.25	179.25	174.25	178.6	177.95	177.91	5982324	1.064313e+14	NaN	1949217.0	0.3258	2003	7
4	MARUTI	EQ	177.95	200.00	200.00	173.00	176.3	176.20	176.88	6173689	1.092001e+14	NaN	1307694.0	0.2118	2003	7
5	MARUTI	EQ	176.20	176.45	179.10	175.35	176.9	177.10	177.59	3759085	6.675695e+13	NaN	1155442.0	0.3074	2003	7
6	MARUTI	EQ	177.10	177.50	178.00	170.65	170.7	172.20	175.48	3814181	6.693030e+13	NaN	1064071.0	0.2790	2003	7
7	MARUTI	EQ	172.20	165.00	171.90	160.00	167.4	167.40	167.66	5375212	9.011903e+13	NaN	1427995.0	0.2657	2003	7
8	MARUTI	EQ	167.40	169.00	169.40	163.05	164.0	164.90	166.67	3315845	5.526646e+13	NaN	980379.0	0.2957	2003	7
9	MARUTI	EQ	164.90	164.00	169.15	162.10	168.6	167.25	165.06	4627992	7.639124e+13	NaN	1257167.0	0.2716	2003	7

To define x Independent variable

In [44]:
```
x = d[[ 'Prev Close', 'Open', 'High', 'Low', 'Last',
       'Close', 'Volume', 'Trades', 'Deliverable Volume',
       '%Deliverble']]
```

In [45]:
```
x
```

Out[45]:

	Prev Close	Open	High	Low	Last	Close	Volume	Trades	Deliverable Volume	%Deliverble
0	125.00	164.90	170.40	155.00	164.00	164.30	35164283	NaN	8537695.0	0.2428
1	164.30	167.00	168.70	164.50	167.00	167.00	10464179	NaN	4363947.0	0.4170
2	167.00	167.75	174.85	166.25	173.60	173.35	11740117	NaN	3014852.0	0.2568
3	173.35	174.25	179.25	174.25	178.60	177.95	5982324	NaN	1949217.0	0.3258
4	177.95	200.00	200.00	173.00	176.30	176.20	6173689	NaN	1307694.0	0.2118
...	...	...	...	...	...	...	...	...	...	...
4422	6676.10	6690.20	6789.00	6600.00	6645.00	6638.90	937344	74474.0	464999.0	0.4961
4423	6638.90	6669.95	6709.00	6542.00	6552.00	6568.75	1610651	130986.0	588617.0	0.3655
4424	6568.75	6568.75	6650.00	6545.00	6581.00	6573.80	1406270	117843.0	672435.0	0.4782
4425	6573.80	6635.00	6647.45	6552.00	6562.00	6565.65	757075	64393.0	352987.0	0.4663
4426	6565.65	6537.10	6559.60	6421.00	6438.35	6455.65	849997	95248.0	382594.0	0.4501

4426 rows × 10 columns

To check null values

In [46]:
```
1 x.isna()
```

Out[46]:

	Prev Close	Open	High	Low	Last	Close	Volume	Trades	Deliverable Volume	%Deliverble
0	False	False	False	False	False	False	False	True	False	False
1	False	False	False	False	False	False	False	True	False	False
2	False	False	False	False	False	False	False	True	False	False
3	False	False	False	False	False	False	False	True	False	False
4	False	False	False	False	False	False	False	True	False	False
...	...	...	...	...	...	...	...	...	...	...
4422	False	False	False	False	False	False	False	False	False	False
4423	False	False	False	False	False	False	False	False	False	False
4424	False	False	False	False	False	False	False	False	False	False
4425	False	False	False	False	False	False	False	False	False	False
4426	False	False	False	False	False	False	False	False	False	False

4426 rows × 10 columns

In [47]:
```
1 x.isna().sum()
```

Out[47]:
```
Prev Close              0
Open                    0
High                    0
Low                     0
Last                    0
Close                   0
Volume                  0
Trades               1970
Deliverable Volume      0
%Deliverble             0
dtype: int64
```

To print trades column data

In [48]:
```
1 x['Trades']
```

Out[48]:
```
0            NaN
1            NaN
2            NaN
3            NaN
4            NaN
           ...
4422     74474.0
4423    130986.0
4424    117843.0
4425     64393.0
4426     95248.0
Name: Trades, Length: 4426, dtype: float64
```

```
In [49]: x['Trades'].tail(10)
```

```
Out[49]: 4417     81274.0
         4418     94720.0
         4419     69676.0
         4420     75599.0
         4421     64854.0
         4422     74474.0
         4423    130986.0
         4424    117843.0
         4425     64393.0
         4426     95248.0
         Name: Trades, dtype: float64
```

```
In [50]: x['Trades'].head(10)
```

```
Out[50]: 0   NaN
         1   NaN
         2   NaN
         3   NaN
         4   NaN
         5   NaN
         6   NaN
         7   NaN
         8   NaN
         9   NaN
         Name: Trades, dtype: float64
```

```
In [51]: x['Trades'].value_counts()
```

```
Out[51]: 24569.0     2
         58345.0     2
         50530.0     2
         81920.0     2
         11304.0     2
                    ..
         50872.0     1
         122984.0    1
         67829.0     1
         11867.0     1
         179002.0    1
         Name: Trades, Length: 2418, dtype: int64
```

```
In [52]: x['Trades'].value_counts().max()
```

```
Out[52]: 2
```

Fill Nill column of trades with a value

In [53]:
```
x['Trades'].fillna(50530,inplace=True)
```

In [54]:
```
x['Trades']
```

Out[54]:
```
0        50530.0
1        50530.0
2        50530.0
3        50530.0
4        50530.0
          ...
4422     74474.0
4423    130986.0
4424    117843.0
4425     64393.0
4426     95248.0
Name: Trades, Length: 4426, dtype: float64
```

To print Y dependent variable

In [55]:
```
y= d["VWAP"]
```

In [56]:
```
y
```

Out[56]:
```
0        165.95
1        166.74
2        172.45
3        177.91
4        176.88
          ...
4422    6678.34
4423    6620.68
4424    6598.62
```

Creating training and testing model

In [57]:
```
from sklearn.model_selection import train_test_split
```

In [58]:
```
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size=0.60,random_state=2)
```

In [59]:
```
x_train.shape
```

Out[59]: (1770, 10)

In [60]:
```
y_train.shape
```

Out[60]: (1770,)

In [61]:
```
x_test.shape
```

Out[61]: (2656, 10)

In [62]:
```
y_test.shape
```

Out[62]: (2656,)

In [63]:
```
from sklearn.linear_model import LinearRegression
```

In [64]:
```
model = LinearRegression()
```

In [65]:
```
model.fit(x_train,y_train)
```

Out[65]: LinearRegression()

In [66]:
```
model.score(x_train,y_train)
```

Out[66]:
```
0.9999871641164005
```

In [67]:
```
y_pred = model.predict(x_test)
```

In [68]:
```
y_pred
```

Out[68]:
```
array([ 754.27672123,  804.90426384, 6638.35853439, ..., 1339.68214756,
        860.15643554,  924.87442132])
```

In [69]:
```
d1 = pd.DataFrame({'Actual':y_test,'Predict':y_pred})
```

In [70]:
```
d1
```

Out[70]:

	Actual	Predict
647	754.52	754.276721
1222	807.26	804.904264
3936	6646.86	6638.358534
1569	1478.07	1487.106710
4367	7315.28	7330.992774
...	...	...
379	467.16	466.216115
3893	6894.86	6899.289934
2173	1338.35	1339.682148
1052	861.20	860.156436
2115	924.07	924.874421

2656 rows × 2 columns

Word Graph

In [1]:
```
pip install wordcloud
```

```
Requirement already satisfied: wordcloud in c:\users\lenovo\anaconda3\lib\site-packages (1.8.1)
Requirement already satisfied: numpy>=1.6.1 in c:\users\lenovo\anaconda3\lib\site-packages (from wordcloud) (1.20.1)
Requirement already satisfied: matplotlib in c:\users\lenovo\anaconda3\lib\site-packages (from wordcloud) (3.3.4)
Requirement already satisfied: pillow in c:\users\lenovo\anaconda3\lib\site-packages (from wordcloud) (8.2.0)
Requirement already satisfied: python-dateutil>=2.1 in c:\users\lenovo\anaconda3\lib\site-packages (from matplotlib->wordcloud)
(2.8.1)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\lenovo\anaconda3\lib\site-packages (from matplotlib->wordcloud)
(1.3.1)
Requirement already satisfied: cycler>=0.10 in c:\users\lenovo\anaconda3\lib\site-packages (from matplotlib->wordcloud) (0.10.
0)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.3 in c:\users\lenovo\anaconda3\lib\site-packages (from ma
tplotlib->wordcloud) (2.4.7)
Requirement already satisfied: six in c:\users\lenovo\anaconda3\lib\site-packages (from cycler>=0.10->matplotlib->wordcloud)
(1.15.0)
Note: you may need to restart the kernel to use updated packages.
```

In [2]:
```
from os import path
from PIL import Image
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator

```

In [9]:
```
text = d.Symbol[1]
```

In [10]:
```
text
```

Out[10]:
```
'MARUTI'
```

Out[10]: MARUTI

In [11]:
```
?WordCloud
```

In [12]:
```
wordcloud = WordCloud().generate(text)
```

In [13]:
```
# Display the generated image:
plt.imshow(wordcloud, interpolation = 'bilinear')
plt.axis("off")
plt.show()
```

2.2 R

Import libraries first

```
library(ggplot2)
library(dplyr)
library(choroplethr)
library(choroplethrMaps)
library(openintro)
library(tidyverse)
library(scales)
library(lubridate)       # For extracting year
library(devtools)        # For plotting bar graphs easy way
library(vioplot)         # Violin Plot
```

Reading data set

```
print(getwd())    # returns absolute file path


  # Read Data file of Maruti


d=read.csv("C:/Users/LENOVO/Desktop/Maruti.csv")
print(d)
# Print head rows and tail rows


print(head(d))
```

```
print(tail(d))
# Print summary of data set

print(summary(d))

 # Print dimension of data set

print(dim(d))

 # Print column names

print(names(d))

 # Print Turnover details

print(d$Turnover)

 # To check length of data set

print(length(d))
# To check statistical values

print(mean(d$High))       # High value mean

print(mean(d$Low))        #Low value mean

print(median(d$High))     #Median-High
```

```
print(median(d$Low))      #Median-Low

print(mode(d$High))       #Mode-High

print(mode(d$Low))        #Mode-Low

print(mean(d$Turnover))   # Mean Turnover

print(range(d$Turnover))  # Print range of Turnover

print(range(d$High))      #Range-high

print(range(d$Low))       #Range-low

print(scale(d$Volume))    #Print scale values of volume

print(dnorm(d$Volume))    # Print normal density of volume

print(pnorm(d$Volume))    #Normal distribution

print(quantile(d$Open))   #Group values

print(sd(d$Open))         #Standard deviation

print(unique(d$Open))     #Unique values
```

```
  # to check high share value and low share value

print(max(d$High))

print(min(d$Low))

print(prod(d$Deliverable.Volume))

  # Sort basis of Turnover

print(sort(d$Turnover))

# print no. of rows and columns

print(nrow(d))

print(ncol(d))

  # To check string values

print(str(d))

print(glimpse(d))

  # Creating a new column for YEAR

d$Year<-year(d$Date)
```

```
print(d$year)

print(head(d))

  # Again check number of columns

print(ncol(d))

  # Extract data of year 2021

d_2021<- data.frame(d[d$Year == 2021, ])
d_2021

  # Extract Month

d_2021$Month<-month(d_2021$Date)
d_2021



  # Adding month in data set

d$Month<-month(d$Date)
d
```

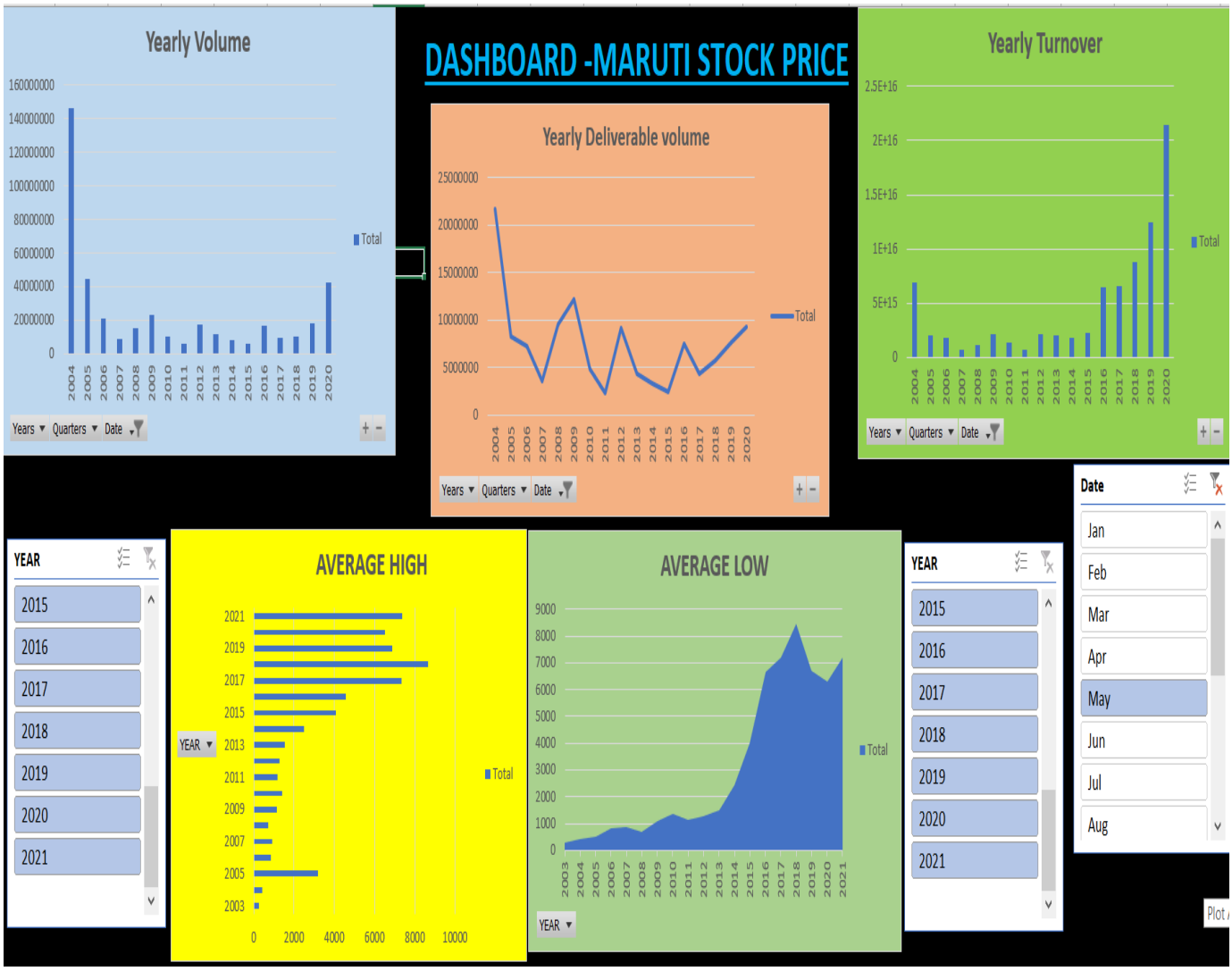
```
  # Visualization



  # Plot high share value range


plot(d$High, col = 'red', xlab = "X-axis", ylab = 'Y-axis',main ='High share
value')


  # Plot low share value range


plot(d$Low, col = 'red', xlab = "X-axis", ylab = 'Y-axis',main ='Low share
value')


  # Turn over plot


plot(d$Turnover, col = 'red', xlab = "X-axis", ylab = 'Y-axis',main ='Turn
Over')


  # High share value in 2021


plot(x=d_2021$Turnover,y=d_2021$High, main="High share values basis
of turnover",
     xlab="Turn Over", ylab="High",col='red')


  # Plot low share values in 2021


ggplot(data=d_2021, aes(fill=Month, y=Low, x=Date)) +
  geom_bar(position="dodge", stat="identity")
```

```
  # Plot high share values in 2021

ggplot(data=d_2021, aes(fill=Month, y=High, x=Date)) +
  geom_bar(position="dodge", stat="identity")

  # Plot for data set basis of high values

ggplot(data=d, aes(fill=Month, y=High, x=Year)) +
        geom_bar(position="dodge", stat="identity")

  # Turn over plot

ggplot(data=d, aes(fill=Month, y=Turnover, x=Year)) +
  geom_bar(position="dodge", stat="identity")

  # sIMPLE LINE PLOT OF closing value

plot(d$Close,type="l")

  #To plot a graph for showing VWAP

ggplot(data = d) +
  stat_count(mapping = aes(x =VWAP),color='green')
```

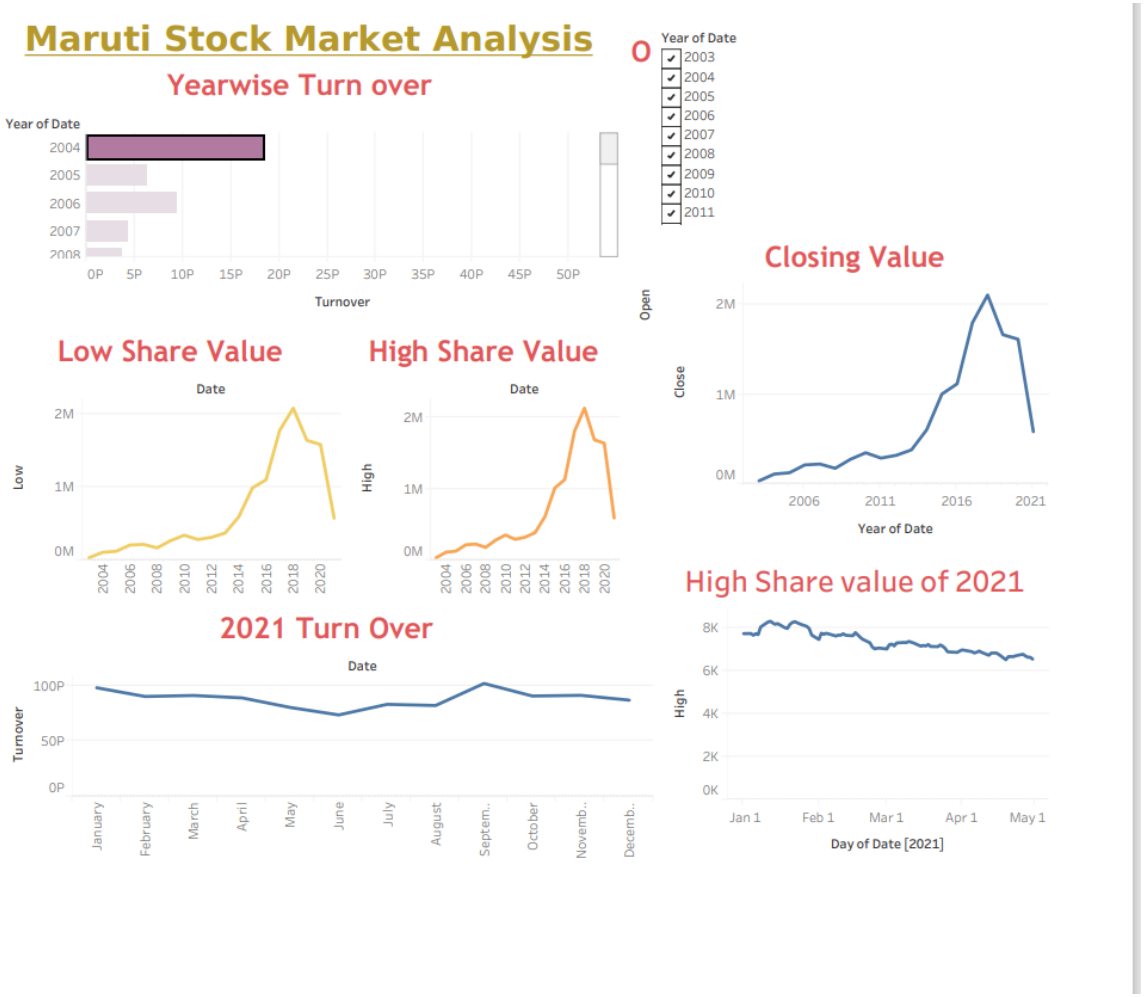
```
  # Plot a relationship of High share value basis Year

ggplot(d, aes(x=Year, y=High)) +
  geom_line(colour="darkgreen", size=1.5)



  # Plot a relationship of Turnover with Year

ggplot(d, aes(x=Year, y=Turnover)) +
  geom_line(colour="darkgreen", size=1.5)

# Graph with a semitransparent shaded area

  # High value with year

ggplot(d, aes(x=Year, y=High)) +
  geom_area(colour="red", alpha=.2)

  # Low value with Year

ggplot(d, aes(x=Year, y=Low)) +
  geom_area(colour="red", alpha=.2)



  # Plot a high share value in Year 2021

ggplot(d_2021, aes(x=Month, y=High)) +
```

```
  geom_area(colour="red", alpha=.2)


  # Plot a low share value in Year 2021

ggplot(d_2021, aes(x=Month, y=Low)) +
  geom_area(colour="red", alpha=.2)




  # Violin Plot

vioplot(d$High, horizontal=TRUE, col="red")          #High value

vioplot(d$Low,horizontal = TRUE,col="pink")          #Low value

vioplot(d$VWAP,horizontal = TRUE,col="blue")         #VWAP

vioplot(d$Deliverable,horizontal = TRUE,col="yellow") #Deliverable

vioplot(d$Volume,horizontal = TRUE,col="green")      #Volume

vioplot(d$VWAP,horizontal = TRUE,col="gold")         #VWAP
```

2.3 Excel


Yearly Volume
Total
Years
Quarters
Date
DASHBOARD -MARUTI STOCK PRICE
Yearly Deliverable volume
Total
Years
Quarters
Date
Yearly Turnover
Total
Years
Quarters
Date
YEAR
2015
2016
2017
2018
2019
2020
2021
AVERAGE HIGH
YEAR
Total
AVERAGE LOW
Total
YEAR
YEAR
2015
2016
2017
2018
2019
2020
2021
Date
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug

2.4 TABLEAU


Maruti Stock Market Analysis
Yearwise Turn over
Year of Date
2004
2005
2006
2007
2008
0P 5P 10P 15P 20P 25P 30P 35P 40P 45P 50P
Turnover
Year of Date
2003
2004
2005
2006
2007
2008
2009
2010
2011
Closing Value
Open
Close
2M
1M
0M
2006 2011 2016 2021
Year of Date
Low Share Value
Date
Low
2M
1M
0M
2004 2006 2008 2010 2012 2014 2016 2018 2020
High Share Value
Date
High
2M
1M
0M
2004 2006 2008 2010 2012 2014 2016 2018 2020
High Share value of 2021
High
8K
6K
4K
2K
0K
Jan 1 Feb 1 Mar 1 Apr 1 May 1
Day of Date [2021]
2021 Turn Over
Date
Turnover
100P
50P
0P
January February March April May June July August Septem.. October Novemb.. Decemb..

CONCLUSION

The Maruti journey at the stock market started in 2003 when the Government of India decided to divest 25% of its stake in Maruti Udyog Limited (old name).

The IPO price was set at ₹ 125 per share. The IPO was over-subscribed 13 times. Those who were lucky to get allotment, were rewarded on the first day itself.

On July 9th 2003, Maruti was first listed on the stock exchanges in India. The share opened at ₹ 165, hit a high of ₹ 170, fell to ₹ 155 and finally ended its day at ₹ 164 per share. A listing gain of 31%.

Great interest in the IPO and fantastic listing. But it did not stop there. Since then, the journey hasn't been smooth, but Maruti has rewarded its shareholders with consistent dividends and excellent returns from growth in value of the share – this has been possible due to Maruti's domination of the automobile sector and steady growth in car sales across the country.

From the end of 2007 to the end of 2008, Maruti crashed by nearly 50%. Nearly all the gains made between 2004 to 2007 was lost in a single year – the great market crash of 2008.

Those who survived 2008 were rewarded in 2009 by 200% gains – but were tested once again in 2010 and 2011 which were extremely painful as the stock crashed by 45% during that period.

The maximum value of share present in the middle of 2018-2019. Share value is decreasing in 2021. High share value is ten thousand. In April 2021, share value is fall down from January 2021.