Deena 20104016

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as pp
```

Problem Statement

LINEAR REGRESSION

In [2]:
```
a = pd.read_csv("Salesworkload.csv")
```

Out[2]:

	MonthYear	Time index	Country	StoreID	City	Dept_ID	Dept. Name	HoursOwn	HoursLe
0	10.2016	1.0	United Kingdom	88253.0	London (I)	1.0	Dry	3184.764	
1	10.2016	1.0	United Kingdom	88253.0	London (I)	2.0	Frozen	1582.941	
2	10.2016	1.0	United Kingdom	88253.0	London (I)	3.0	other	47.205	
3	10.2016	1.0	United Kingdom	88253.0	London (I)	4.0	Fish	1623.852	
4	10.2016	1.0	United Kingdom	88253.0	London (I)	5.0	Fruits & Vegetables	1759.173	
...	...	...	...	...	...	...	...	...	
7653	6.2017	9.0	Sweden	29650.0	Gothenburg	12.0	Checkout	6322.323	
7654	6.2017	9.0	Sweden	29650.0	Gothenburg	16.0	Customer Services	4270.479	
7655	6.2017	9.0	Sweden	29650.0	Gothenburg	11.0	Delivery	0	
7656	6.2017	9.0	Sweden	29650.0	Gothenburg	17.0	others	2224.929	
7657	6.2017	9.0	Sweden	29650.0	Gothenburg	18.0	all	39652.2	

7658 rows × 14 columns

HEAD

In [3]:
```python
d=a.head(8)
```

Out[3]:

	MonthYear	Time index	Country	StoreID	City	Dept_ID	Dept. Name	HoursOwn	HoursLease	
0	10.2016	1.0	United Kingdom	88253.0	London (I)	1.0	Dry	3184.764	0.0	3
1	10.2016	1.0	United Kingdom	88253.0	London (I)	2.0	Frozen	1582.941	0.0	
2	10.2016	1.0	United Kingdom	88253.0	London (I)	3.0	other	47.205	0.0	4
3	10.2016	1.0	United Kingdom	88253.0	London (I)	4.0	Fish	1623.852	0.0	3
4	10.2016	1.0	United Kingdom	88253.0	London (I)	5.0	Fruits & Vegetables	1759.173	0.0	1
5	10.2016	1.0	United Kingdom	88253.0	London (I)	6.0	Meat	8270.316	0.0	17
6	10.2016	1.0	United Kingdom	88253.0	London (I)	13.0	Food	16468.251	0.0	31

Data Cleaning and Preprocessing

In [4]:
```python
b=d.dropna(axis=1)
```

Out[4]:

	MonthYear	Time index	Country	StoreID	City	Dept_ID	Dept. Name	HoursOwn	HoursLease	
0	10.2016	1.0	United Kingdom	88253.0	London (I)	1.0	Dry	3184.764	0.0	3
1	10.2016	1.0	United Kingdom	88253.0	London (I)	2.0	Frozen	1582.941	0.0	
2	10.2016	1.0	United Kingdom	88253.0	London (I)	3.0	other	47.205	0.0	4
3	10.2016	1.0	United Kingdom	88253.0	London (I)	4.0	Fish	1623.852	0.0	3
4	10.2016	1.0	United Kingdom	88253.0	London (I)	5.0	Fruits & Vegetables	1759.173	0.0	1
5	10.2016	1.0	United Kingdom	88253.0	London (I)	6.0	Meat	8270.316	0.0	17
6	10.2016	1.0	United Kingdom	88253.0	London (I)	13.0	Food	16468.251	0.0	31

In [5]:

Out[5]:

	Time index	StoreID	Dept_ID	HoursLease	Sales units	Turnover	Custor
count	7650.000000	7650.000000	7650.000000	7650.000000	7.650000e+03	7.650000e+03	
mean	5.000000	61995.220000	9.470588	22.036078	1.076471e+06	3.721393e+06	N
std	2.582158	29924.581631	5.337429	133.299513	1.728113e+06	6.003380e+06	N
min	1.000000	12227.000000	1.000000	0.000000	0.000000e+00	0.000000e+00	N
25%	3.000000	29650.000000	5.000000	0.000000	5.457125e+04	2.726798e+05	N
50%	5.000000	75400.500000	9.000000	0.000000	2.932300e+05	9.319575e+05	N
75%	7.000000	87703.000000	14.000000	0.000000	9.175075e+05	3.264432e+06	N
max	9.000000	98422.000000	18.000000	3984.000000	1.124296e+07	4.271739e+07	N

To display heading

In [6]:

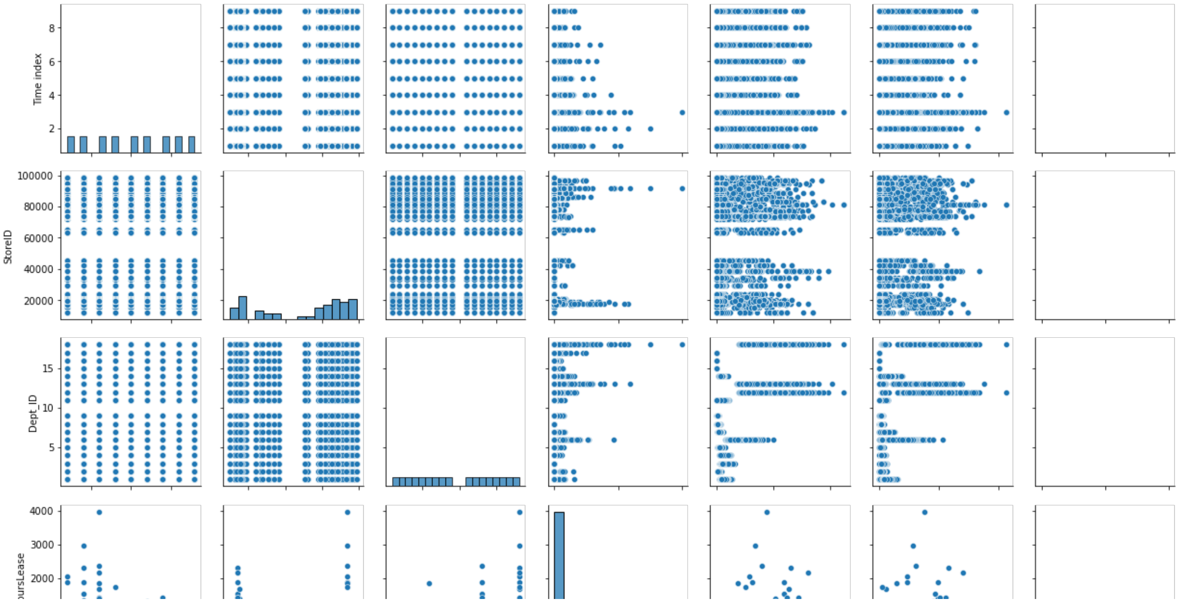
Out[6]:

```
Index(['MonthYear', 'Time index', 'Country', 'StoreID', 'City', 'Dept_ID',
       'Dept. Name', 'HoursOwn', 'HoursLease', 'Sales units', 'Turnover',
       'Customer', 'Area (m2)', 'Opening hours'],
      dtype='object')
```

In [7]:

Out[7]:

```
<seaborn.axisgrid.PairGrid at 0x1dad4826790>
```

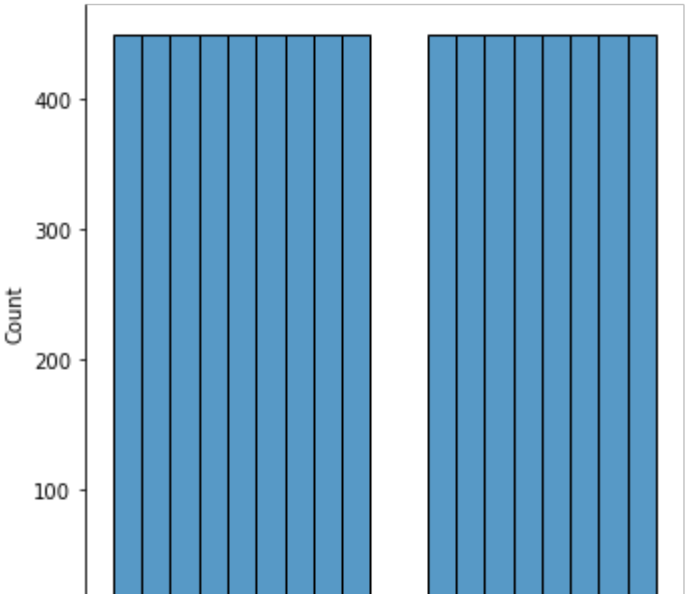


In [8]:

```
[illegible]
```

Out[8]:

```
<seaborn.axisgrid.FacetGrid at 0x1dad49461c0>
```

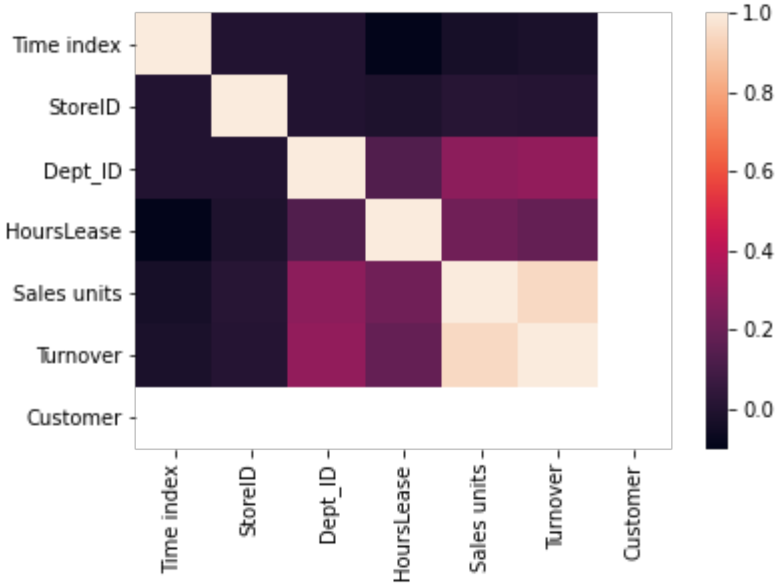


In [9]:

```
[illegible]
```

Out[9]:

```
<AxesSubplot:>
```



TO TRAIN THE MODEL - MODEL BUILDING

In [10]:

```
x=b[['Dept_ID']]
[illegible]
```

In [11]:

```
# to split my dataset into training and test data
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3)
```

In [12]:
```
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
```

Out[12]:
```
LinearRegression()
```

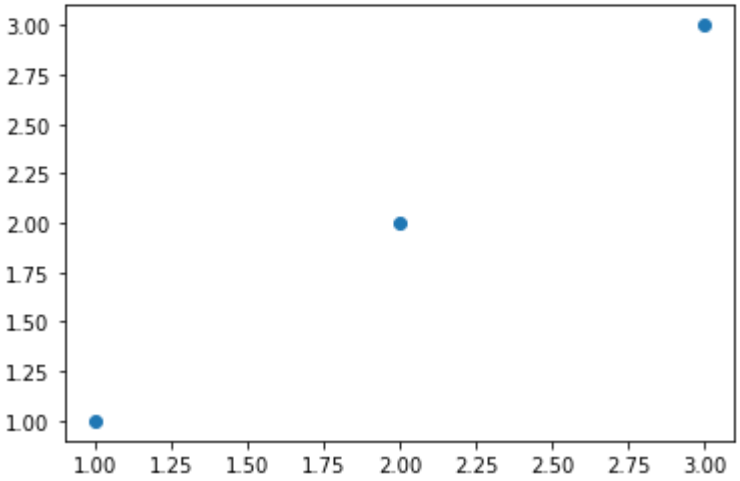
In [13]:
```
coeff = pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
```

Out[13]:

	Co-efficient
Dept_ID	1.0

In [14]:
```
prediction= lr.predict(x_test)
```

Out[14]:
```
<matplotlib.collections.PathCollection at 0x1dad98900d0>
```



In [15]:

Out[15]:
```
1.0
```

RIDGE & LASSO

In [16]:
```
from sklearn.linear_model import Ridge,Lasso
rr=Ridge(alpha=10)
```

Out[16]:
```
Ridge(alpha=10)
```

In [17]:

Out[17]:
```
-0.06944444444444464
```

In [18]:
```
la=Lasso(alpha=10)
```

Out[18]:
```
Lasso(alpha=10)
```

In [19]:

Out[19]:
```
-37.5
```

In [20]:

```
from sklearn.linear_model import ElasticNet
a=ElasticNet()
```

Out[20]:

```
ElasticNet()
```

In [21]:

```
print(a.coef_)
print(a.intercept_)
print(a.score(x_test,y_test))
```

```
[0.9047619]
0.6666666666666667
0.6507936507936505
[2.47619048 1.57142857 3.38095238]
```

In [22]:

```
from sklearn import metrics
print(" Mean Absolute Error :",metrics.mean_absolute_error(y_test,prediction))
print(" Mean Squared Error :",metrics.mean_squared_error(y_test,prediction))
```

```
 Mean Absolute Error : 1.2582527612418441e-15
 Mean Squared Error : 1.6598948214025456e-30
 Root Mean Absolute Error : 3.5471858722681055e-08
```