

HEAMNATH

20104028

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:

```
df=pd.read_csv("6_Salesworkload1.csv")
df
```

Out[2]:

	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
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
3	10.2016	1.0	United Kingdom	88253.0	London (I)	4.0	Fish	1623.852	0.0
4	10.2016	1.0	United Kingdom	88253.0	London (I)	5.0	Fruits & Vegetables	1759.173	0.0
...
7653	06.2017	9.0	Sweden	29650.0	Gothenburg	12.0	Checkout	6322.323	0.0
7654	06.2017	9.0	Sweden	29650.0	Gothenburg	16.0	Customer Services	4270.479	0.0
7655	06.2017	9.0	Sweden	29650.0	Gothenburg	11.0	Delivery	0	0.0
7656	06.2017	9.0	Sweden	29650.0	Gothenburg	17.0	others	2224.929	0.0
7657	06.2017	9.0	Sweden	29650.0	Gothenburg	18.0	all	39652.2	0.0

7658 rows × 14 columns

In [3]:

```
df.head()
```

Out[3]:

	MonthYear	Time index	Country	StoreID	City	Dept_ID	Dept. Name	HoursOwn	HoursLease	Sa	ur
0	10.2016	1.0	United Kingdom	88253.0	London (I)	1.0	Dry	3184.764	0.0	39856	

	MonthYear	Time index	Country	StoreID	City	Dept_ID	Dept. Name	HoursOwn	HoursLease	Sa	ur
1	10.2016	1.0	United Kingdom	88253.0	London (I)	2.0	Frozen	1582.941	0.0	8272	
2	10.2016	1.0	United Kingdom	88253.0	London (I)	3.0	other	47.205	0.0	43840	
3	10.2016	1.0	United Kingdom	88253.0	London (I)	4.0	Fish	1623.852	0.0	30942	

DATA CLEANING AND DATA PREPROCESSING

In [4]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7658 entries, 0 to 7657
Data columns (total 14 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   MonthYear        7658 non-null   object 
 1   Time index       7650 non-null   float64
 2   Country          7650 non-null   object 
 3   StoreID          7650 non-null   float64
 4   City              7650 non-null   object 
 5   Dept_ID          7650 non-null   float64
 6   Dept. Name        7650 non-null   object 
 7   HoursOwn         7650 non-null   object 
 8   HoursLease        7650 non-null   float64
 9   Sales units      7650 non-null   float64
 10  Turnover          7650 non-null   float64
 11  Customer          0 non-null     float64
 12  Area (m2)        7650 non-null   object 
 13  Opening hours    7650 non-null   object 
dtypes: float64(7), object(7)
memory usage: 837.7+ KB
```

In [5]:

```
df.describe()
```

Out[5]:

	Time index	StoreID	Dept_ID	HoursLease	Sales units	Turnover	Customer
count	7650.000000	7650.000000	7650.000000	7650.000000	7.650000e+03	7.650000e+03	0.0
mean	5.000000	61995.220000	9.470588	22.036078	1.076471e+06	3.721393e+06	NaN
std	2.582158	29924.581631	5.337429	133.299513	1.728113e+06	6.003380e+06	NaN
min	1.000000	12227.000000	1.000000	0.000000	0.000000e+00	0.000000e+00	NaN
25%	3.000000	29650.000000	5.000000	0.000000	5.457125e+04	2.726798e+05	NaN
50%	5.000000	75400.500000	9.000000	0.000000	2.932300e+05	9.319575e+05	NaN
75%	7.000000	87703.000000	14.000000	0.000000	9.175075e+05	3.264432e+06	NaN
max	9.000000	98422.000000	18.000000	3984.000000	1.124296e+07	4.271739e+07	NaN

In [6]: `df.columns`

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]: `df1=df.fillna(1)`
`df1`

Out[7]:

	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
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
3	10.2016	1.0	United Kingdom	88253.0	London (I)	4.0	Fish	1623.852	0.0
4	10.2016	1.0	United Kingdom	88253.0	London (I)	5.0	Fruits & Vegetables	1759.173	0.0
...
7653	06.2017	9.0	Sweden	29650.0	Gothenburg	12.0	Checkout	6322.323	0.0
7654	06.2017	9.0	Sweden	29650.0	Gothenburg	16.0	Customer Services	4270.479	0.0
7655	06.2017	9.0	Sweden	29650.0	Gothenburg	11.0	Delivery	0	0.0
7656	06.2017	9.0	Sweden	29650.0	Gothenburg	17.0	others	2224.929	0.0
7657	06.2017	9.0	Sweden	29650.0	Gothenburg	18.0	all	39652.2	0.0

7658 rows × 14 columns

In [8]: `df1=df1.replace('#NV',1)`
`df1`

Out[8]:

	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
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

	MonthYear	Time index	Country	StoreID	City	Dept_ID	Dept. Name	HoursOwn	HoursLease
3	10.2016	1.0	United Kingdom	88253.0	London (I)	4.0	Fish	1623.852	0.0
4	10.2016	1.0	United Kingdom	88253.0	London (I)	5.0	Fruits & Vegetables	1759.173	0.0
...
7653	06.2017	9.0	Sweden	29650.0	Gothenburg	12.0	Checkout	6322.323	0.0
7654	06.2017	9.0	Sweden	29650.0	Gothenburg	16.0	Customer Services	4270.479	0.0
7655	06.2017	9.0	Sweden	29650.0	Gothenburg	11.0	Delivery	0	0.0
7656	06.2017	9.0	Sweden	29650.0	Gothenburg	17.0	others	2224.929	0.0
7657	06.2017	9.0	Sweden	29650.0	Gothenburg	18.0	all	39652.2	0.0

In [9]:

`df1.info()`

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7658 entries, 0 to 7657
Data columns (total 14 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   MonthYear        7658 non-null   object 
 1   Time index       7658 non-null   float64
 2   Country          7658 non-null   object 
 3   StoreID          7658 non-null   float64
 4   City              7658 non-null   object 
 5   Dept_ID          7658 non-null   float64
 6   Dept. Name        7658 non-null   object 
 7   HoursOwn         7658 non-null   object 
 8   HoursLease        7658 non-null   float64
 9   Sales units      7658 non-null   float64
 10  Turnover          7658 non-null   float64
 11  Customer          7658 non-null   float64
 12  Area (m2)        7658 non-null   object 
 13  Opening hours    7658 non-null   object 
dtypes: float64(7), object(7)
memory usage: 837.7+ KB
```

In [10]:

`df1.columns`

```
Out[10]: Index(['MonthYear', 'Time index', 'Country', 'StoreID', 'City', 'Dept_ID',
       'Dept. Name', 'HoursOwn', 'HoursLease', 'Sales units', 'Turnover',
       'Customer', 'Area (m2)', 'Opening hours'],
      dtype='object')
```

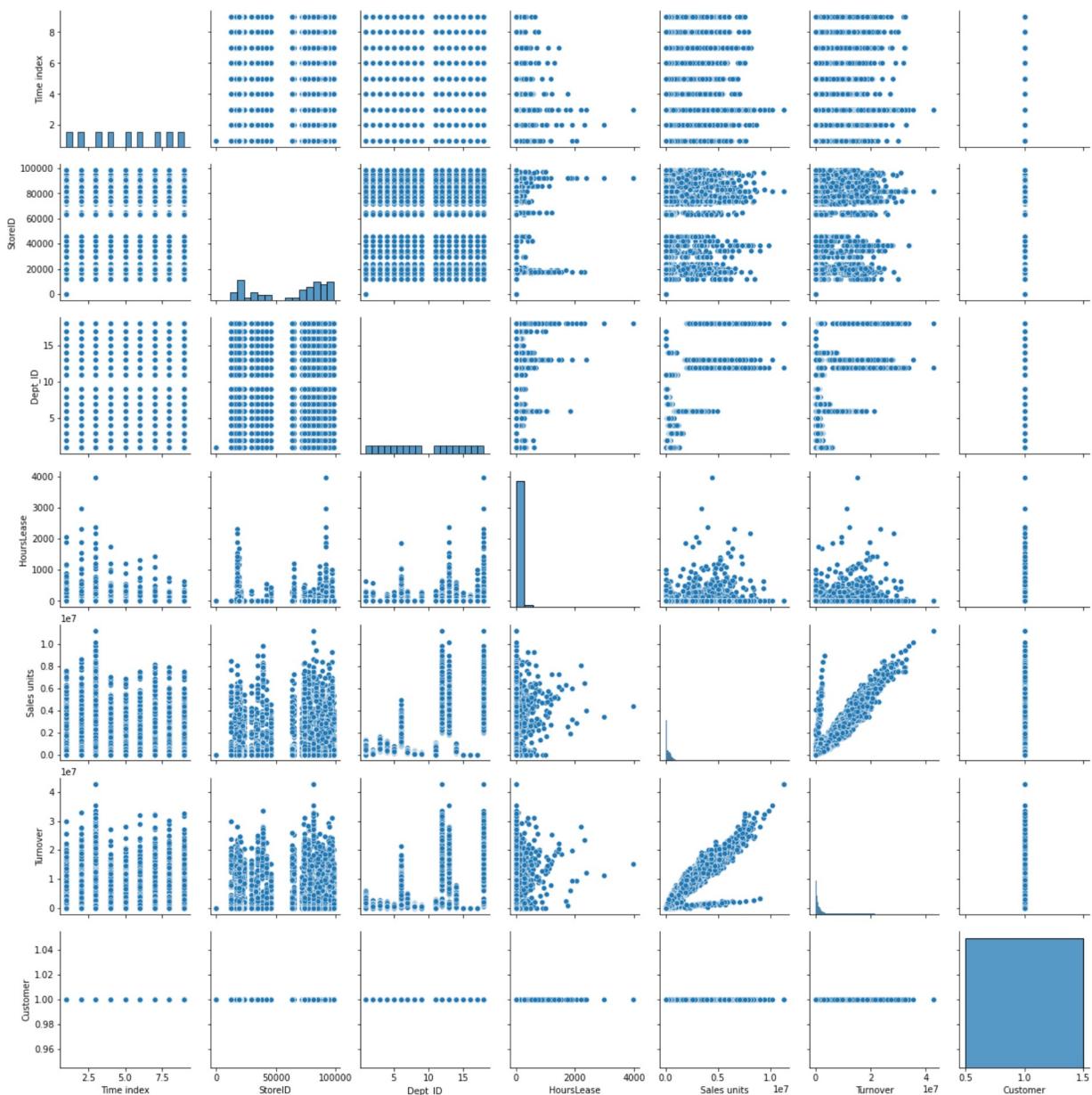
In [11]:

`df1=df1[['Time index', 'StoreID', 'Dept_ID', 'HoursLease', 'Sales units', 'Turnover']]`

EDA AND VISUALIZATION

```
In [12]: sns.pairplot(df1)
```

```
Out[12]: <seaborn.axisgrid.PairGrid at 0x21829364940>
```



```
In [13]: sns.distplot(df1['Turnover'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
```

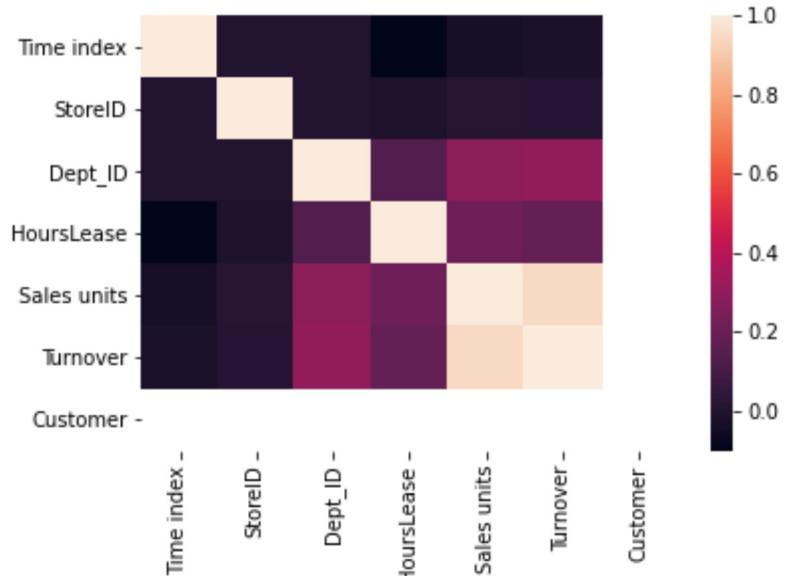
```
warnings.warn(msg, FutureWarning)
```

```
Out[13]: <AxesSubplot:xlabel='Turnover', ylabel='Density'>
```



In [14]: `sns.heatmap(df1.corr())`

Out[14]: <AxesSubplot:>



TO TRAIN THE MODEL AND MODEL BULDING

In [15]: `df1`

Out[15]:

	Time index	StoreID	Dept_ID	HoursLease	Sales units	Turnover	Customer
0	1.0	88253.0	1.0	0.0	398560.0	1226244.0	1.0
1	1.0	88253.0	2.0	0.0	82725.0	387810.0	1.0
2	1.0	88253.0	3.0	0.0	438400.0	654657.0	1.0
3	1.0	88253.0	4.0	0.0	309425.0	499434.0	1.0
4	1.0	88253.0	5.0	0.0	165515.0	329397.0	1.0
...
7653	9.0	29650.0	12.0	0.0	3886530.0	14538825.0	1.0
7654	9.0	29650.0	16.0	0.0	245.0	0.0	1.0

	Time index	StoreID	Dept_ID	HoursLease	Sales units	Turnover	Customer
7655	9.0	29650.0	11.0	0.0	0.0	0.0	1.0
7656	9.0	29650.0	17.0	0.0	245.0	0.0	1.0
7657	9.0	29650.0	18.0	0.0	3886530.0	15056214.0	1.0

In [16]:

```
x=df1[['Time index', 'StoreID', 'Dept_ID', 'HoursLease', 'Sales units','Customer']]  
y=df1['Turnover']
```

In [17]:

```
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 [18]:

```
from sklearn.linear_model import LinearRegression  
lr=LinearRegression()  
lr.fit(x_train,y_train)
```

Out[18]:

```
LinearRegression()
```

In [19]:

```
lr.intercept_
```

Out[19]:

```
-232889.84952322207
```

In [20]:

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

Out[20]:

Co-efficient

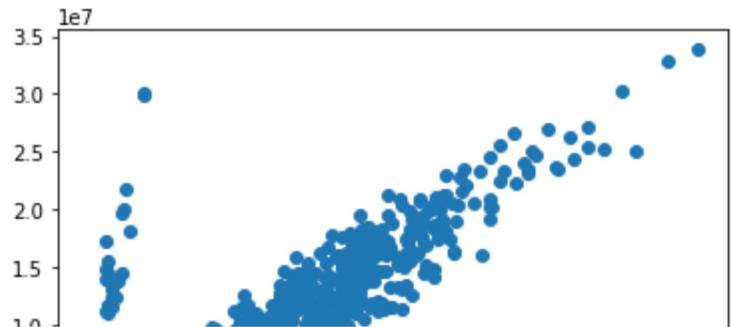
Time index	19894.285250
StoreID	-1.018654
Dept_ID	42638.000834
HoursLease	-1227.926504
Sales units	3.287163
Customer	0.000000

In [21]:

```
prediction =lr.predict(x_test)  
plt.scatter(y_test,prediction)
```

Out[21]:

```
<matplotlib.collections.PathCollection at 0x2182d4c58e0>
```



ACCURACY

```
In [22]: lr.score(x_test,y_test)
```

```
Out[22]: 0.8891247723999043
```