

Problem statement

Data collection

Importing libraries

```
In [1]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Importing dataset

```
In [2]: data=pd.read_csv(r"C:\Users\user\Downloads\6_Salesworkload1 - 6_Salesworkload1.csv")
data
```

```
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	6.2017	9.0	Sweden	29650.0	Gothenburg	12.0	Checkout	6322.323	0.0
7654	6.2017	9.0	Sweden	29650.0	Gothenburg	16.0	Customer Services	4270.479	0.0
7655	6.2017	9.0	Sweden	29650.0	Gothenburg	11.0	Delivery	0	0.0
7656	6.2017	9.0	Sweden	29650.0	Gothenburg	17.0	others	2224.929	0.0
7657	6.2017	9.0	Sweden	29650.0	Gothenburg	18.0	all	39652.2	0.0

7658 rows × 14 columns



head

In [3]:

```
# to display first 8 dataset values
da=data.head(8)
da
```

Out[3]:

	MonthYear	Time index	Country	StoreID	City	Dept_ID	Dept. Name	HoursOwn	HoursLease	Sale unit
0	10.2016	1.0	United Kingdom	88253.0	London (I)	1.0	Dry	3184.764	0.0	398560.0
1	10.2016	1.0	United Kingdom	88253.0	London (I)	2.0	Frozen	1582.941	0.0	82725.0
2	10.2016	1.0	United Kingdom	88253.0	London (I)	3.0	other	47.205	0.0	438400.0
3	10.2016	1.0	United Kingdom	88253.0	London (I)	4.0	Fish	1623.852	0.0	309425.0
4	10.2016	1.0	United Kingdom	88253.0	London (I)	5.0	Fruits & Vegetables	1759.173	0.0	165515.0
5	10.2016	1.0	United Kingdom	88253.0	London (I)	6.0	Meat	8270.316	0.0	1713310.0
6	10.2016	1.0	United Kingdom	88253.0	London (I)	13.0	Food	16468.251	0.0	3107935.0
7	10.2016	1.0	United Kingdom	88253.0	London (I)	7.0	Clothing	4698.471	0.0	213680.0

info

In [4]:

```
# to identify missing values
data.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

describe

In [5]:

to display summary of the dataset
data.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

columns

In [6]:

to display headings of the dataset
data.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]:

a=da.dropna(axis=1,how='any')
a

Out[7]:

	MonthYear	Time index	Country	StoreID	City	Dept_ID	Dept. Name	HoursOwn	HoursLease	Sale unit
0	10.2016	1.0	United Kingdom	88253.0	London (I)	1.0	Dry	3184.764	0.0	398560.0
1	10.2016	1.0	United Kingdom	88253.0	London (I)	2.0	Frozen	1582.941	0.0	82725.0
2	10.2016	1.0	United Kingdom	88253.0	London (I)	3.0	other	47.205	0.0	438400.0
3	10.2016	1.0	United Kingdom	88253.0	London (I)	4.0	Fish	1623.852	0.0	309425.0
4	10.2016	1.0	United Kingdom	88253.0	London (I)	5.0	Fruits &	1759.173	0.0	165515.0

	MonthYear	Time index	Country	StoreID	City	Dept_ID	Dept. Name	HoursOwn	HoursLease	Sale unit
			Kingdom		(I)		Vegetables			
5	10.2016	1.0	United Kingdom	88253.0	London (I)	6.0	Meat	8270.316	0.0	1713310.1
6	10.2016	1.0	United Kingdom	88253.0	London (I)	13.0	Food	16468.251	0.0	3107935.1
7	10.2016	1.0	United Kingdom	88253.0	London (I)	7.0	Clothing	4698.471	0.0	213680.1

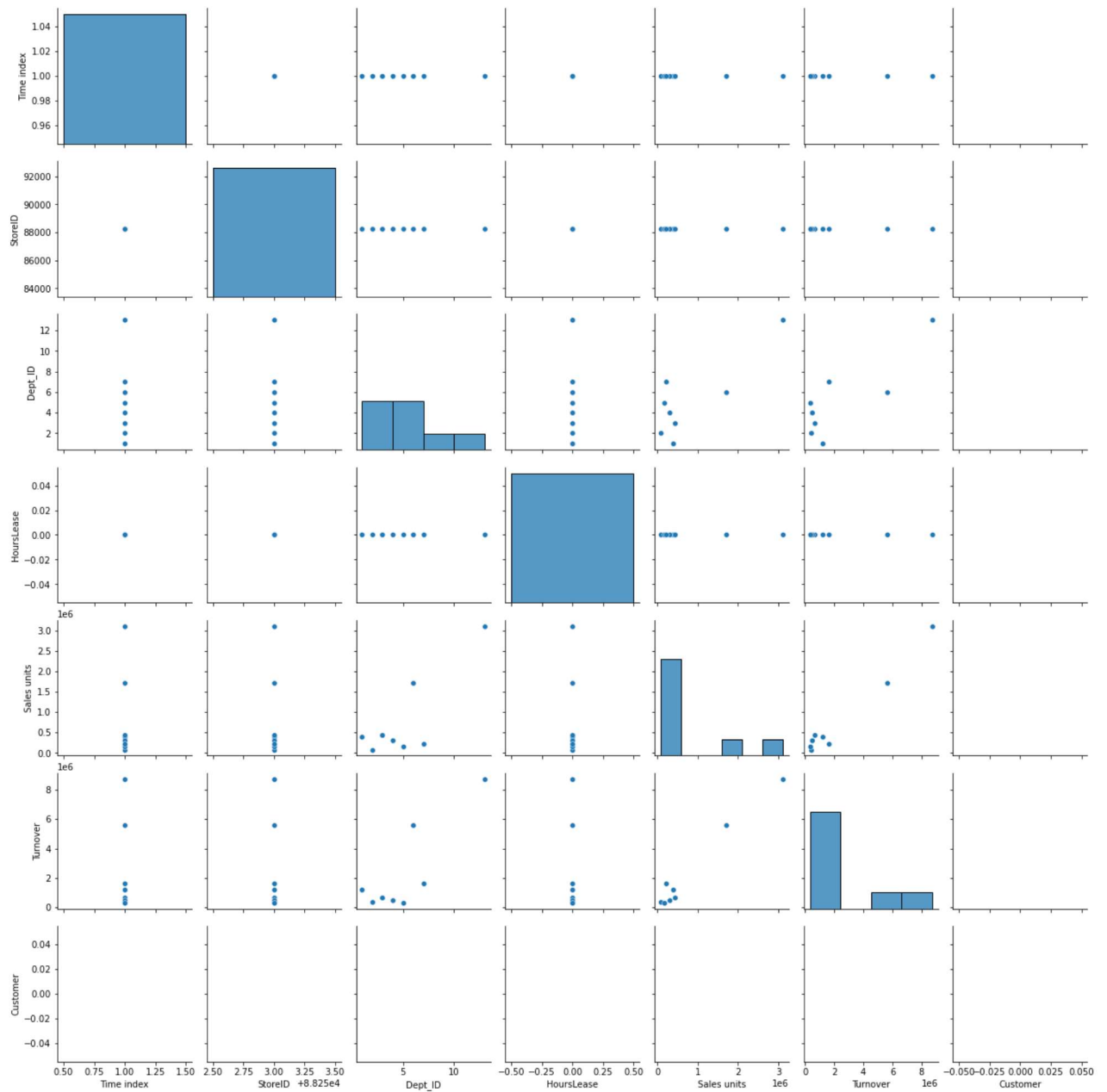
In [8]: `a.columns`

Out[8]: Index(['MonthYear', 'Time index', 'Country', 'StoreID', 'City', 'Dept_ID', 'Dept. Name', 'HoursOwn', 'HoursLease', 'Sales units', 'Turnover', 'Area (m2)', 'Opening hours'], dtype='object')

EDA and Visualization

In [9]: `sns.pairplot(da)`

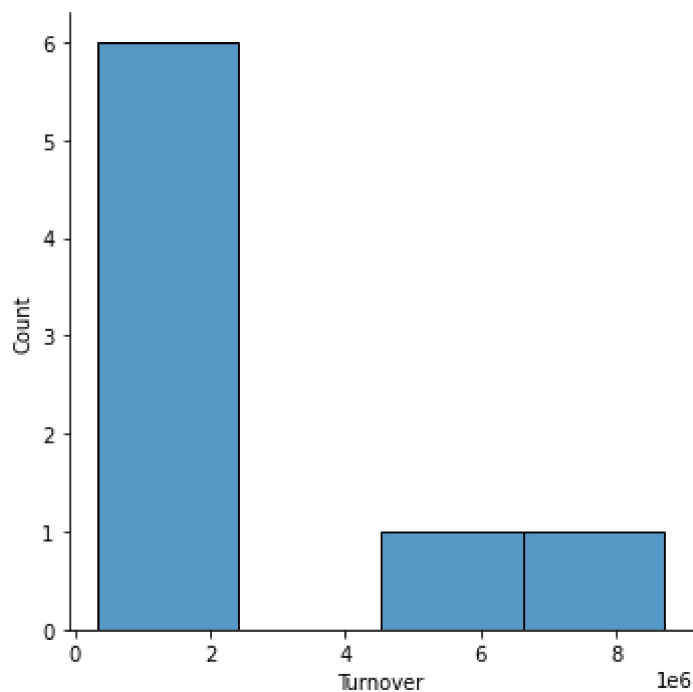
Out[9]: <seaborn.axisgrid.PairGrid at 0x21954640730>



distribution plot

```
In [10]: sns.displot(a['Turnover'])
```

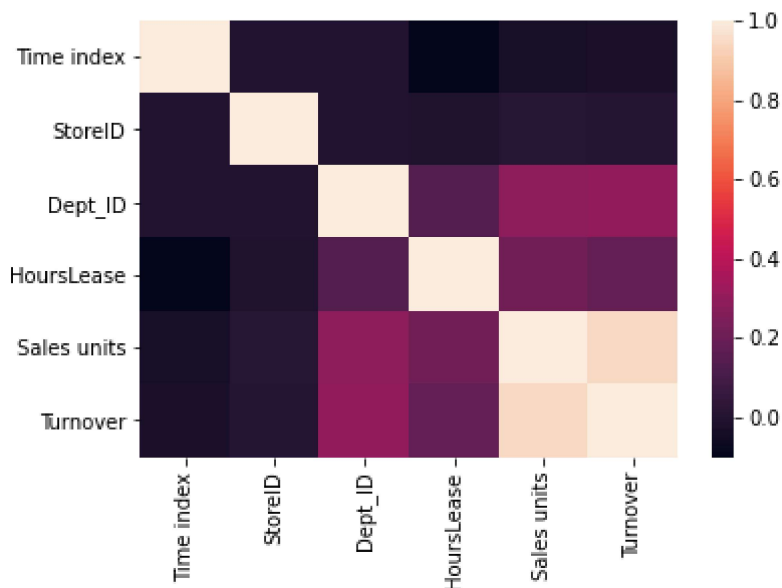
```
Out[10]: <seaborn.axisgrid.FacetGrid at 0x2195b0b6280>
```



correlation

```
In [11]: dat=data[['MonthYear', 'Time index', 'Country', 'StoreID', 'City', 'Dept_ID',
                  'Dept. Name', 'HoursOwn', 'HoursLease', 'Sales units', 'Turnover',
                  'Area (m2)', 'Opening hours']]
          sns.heatmap(dat.corr())
```

Out[11]: <AxesSubplot:>



To train the model-Model Building

```
In [13]: x=a[['StoreID']]
          y=a['Dept_ID']
```

```
In [14]: # 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 [15]: from sklearn.linear_model import LinearRegression
lr= LinearRegression()
lr.fit(x_train,y_train)
```

Out[15]: LinearRegression()

```
In [16]: print(lr.intercept_)
```

6.4

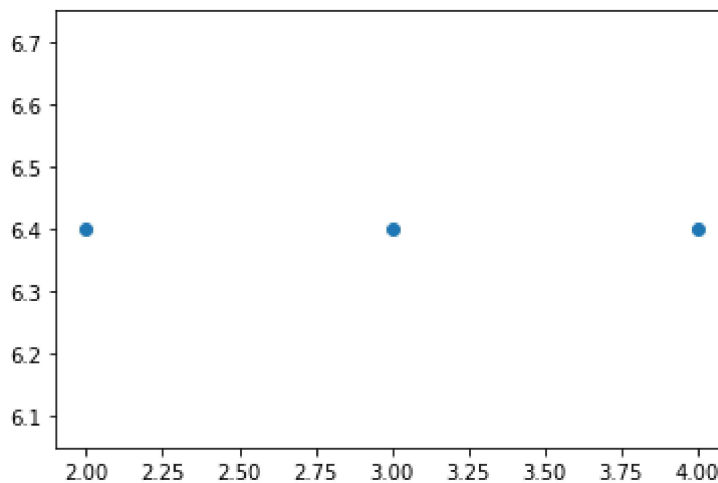
```
In [17]: coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[17]:

	Co-efficient
StoreID	0.0

```
In [18]: prediction=lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[18]: <matplotlib.collections.PathCollection at 0x2195d3804c0>



```
In [19]: print(lr.score(x_test,y_test))
```

-17.340000000000003