

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

```
In [3]: df=pd.read_csv(r"C:\Users\Admin\Downloads\6_Salesworkload1 - 6_Salesworkload1.csv")
df
```

Out[3]:

	MonthYear	Time index	Country	StoreID	City	Dept_ID	Dept. Name	HoursOwn	HoursLea
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



```
In [4]: df1=df.head(100)
```

```
In [6]: df1.info()
```

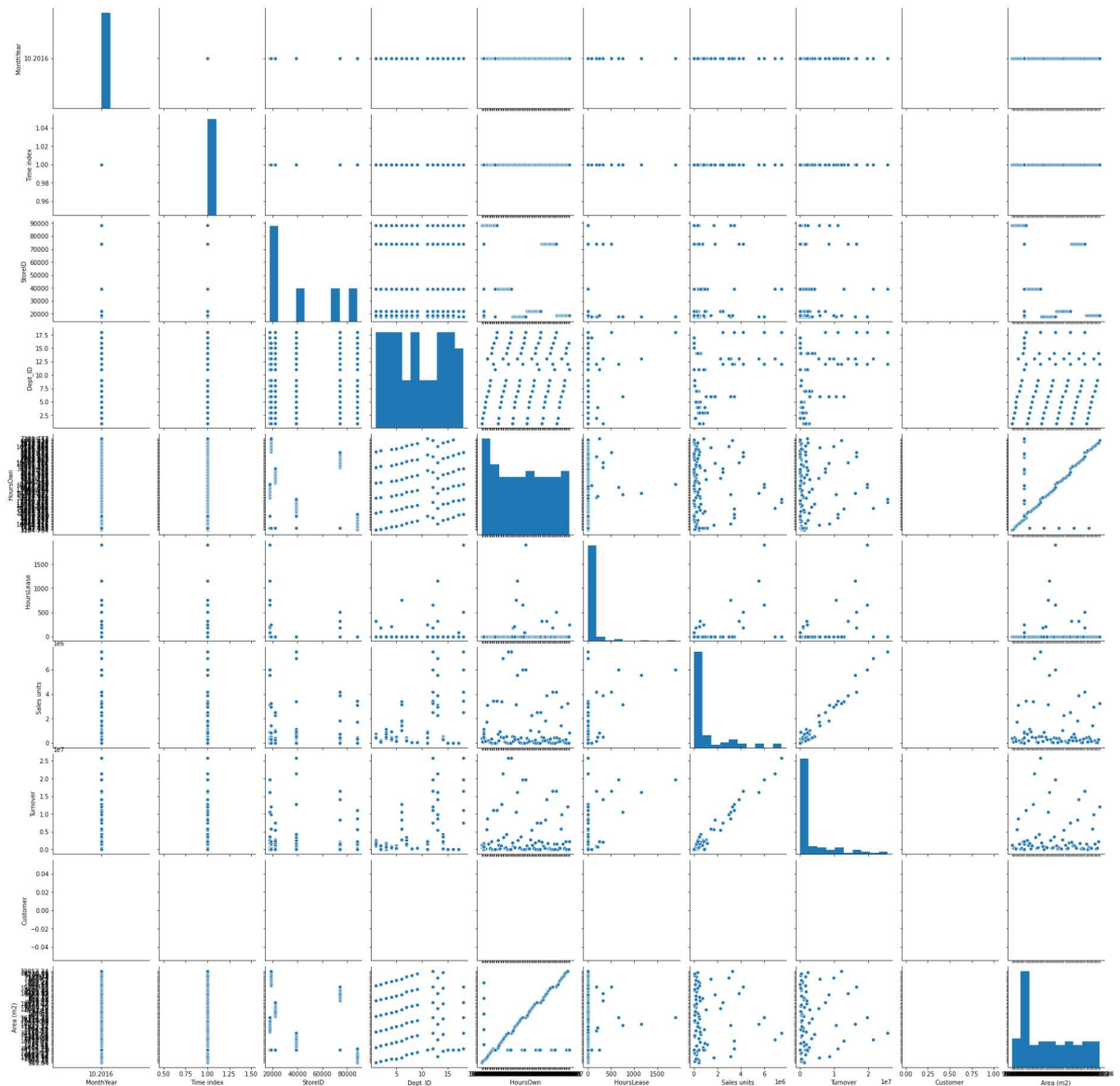
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 14 columns):
 #   Column          Non-Null Count  Dtype
---  -
 0   MonthYear       100 non-null   object
 1   Time index      100 non-null   float64
 2   Country         100 non-null   object
 3   StoreID         100 non-null   float64
 4   City            100 non-null   object
 5   Dept_ID         100 non-null   float64
 6   Dept. Name      100 non-null   object
 7   HoursOwn        100 non-null   object
 8   HoursLease      100 non-null   float64
 9   Sales units     100 non-null   float64
10   Turnover        100 non-null   float64
11   Customer        0 non-null     float64
12   Area (m2)       100 non-null   object
13   Opening hours   100 non-null   object
dtypes: float64(7), object(7)
memory usage: 11.1+ KB
```

```
In [8]: df1.columns
```

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

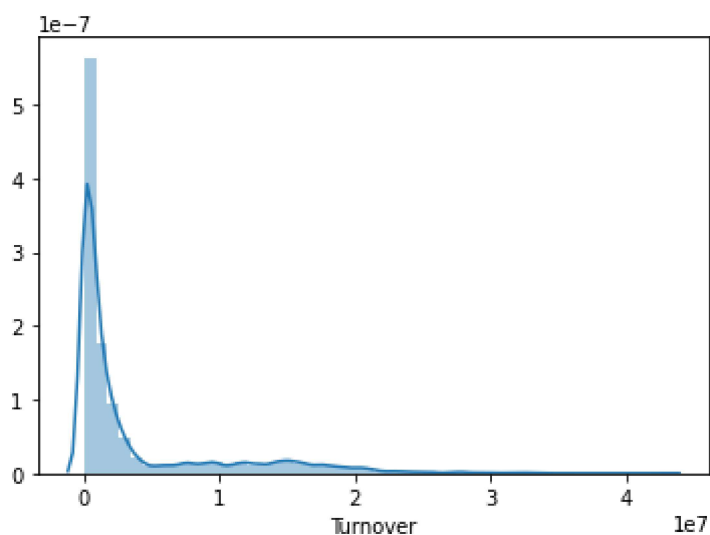
```
In [9]: sns.pairplot(df1)
```

```
Out[9]: <seaborn.axisgrid.PairGrid at 0x229ace70130>
```



```
In [10]: sns.distplot(df['Turnover'])
```

```
Out[10]: <matplotlib.axes._subplots.AxesSubplot at 0x229b3730f70>
```



```
In [12]: df2=df1[[ 'Time index','StoreID','Dept_ID',  
                  'HoursLease', 'Sales units', 'Turnover']]  
df2
```

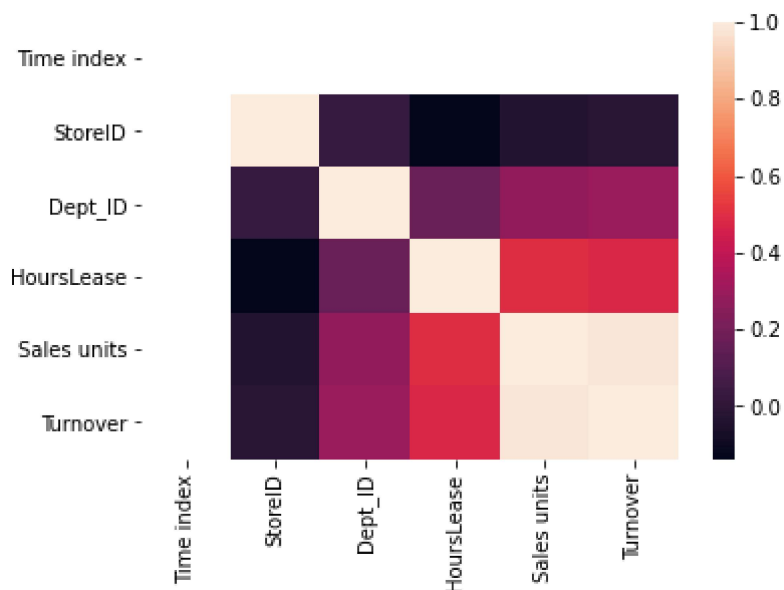
```
Out[12]:
```

	Time index	StoreID	Dept_ID	HoursLease	Sales units	Turnover
0	1.0	88253.0	1.0	0.0	398560.0	1226244.0
1	1.0	88253.0	2.0	0.0	82725.0	387810.0
2	1.0	88253.0	3.0	0.0	438400.0	654657.0
3	1.0	88253.0	4.0	0.0	309425.0	499434.0
4	1.0	88253.0	5.0	0.0	165515.0	329397.0
...
95	1.0	18808.0	14.0	0.0	301500.0	2319717.0
96	1.0	18808.0	15.0	0.0	25.0	0.0
97	1.0	18808.0	12.0	0.0	3262240.0	12161196.0
98	1.0	18808.0	16.0	0.0	25.0	0.0
99	1.0	18808.0	11.0	246.0	843615.0	2204589.0

100 rows × 6 columns

```
In [13]: sns.heatmap(df2.corr())
```

```
Out[13]: <matplotlib.axes._subplots.AxesSubplot at 0x229b551ba00>
```



```
In [14]: x=df2[['Time index','StoreID','Dept_ID',
                'HoursLease', 'Sales units']]
y=df2[['Turnover']]
```

```
In [15]: 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 [16]: from sklearn.linear_model import LinearRegression
lr= LinearRegression()
lr.fit(x_train,y_train)
```

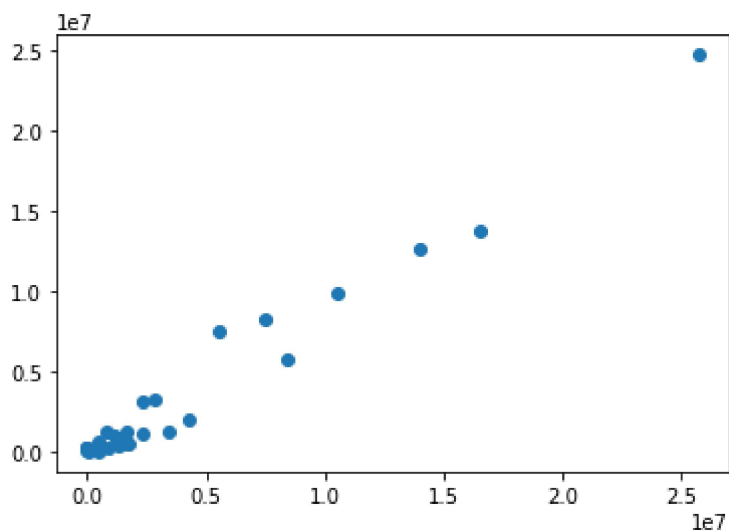
```
Out[16]: LinearRegression()
```

```
In [17]: print(lr.intercept_)

[-257055.70438318]
```

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

Out[18]: <matplotlib.collections.PathCollection at 0x229b5977790>



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

0.9593541937589882

```
In [20]: print(lr.score(x_train,y_train))
```

0.9773549987875161

```
In [21]: from sklearn.linear_model import Ridge,Lasso
```

```
In [22]: rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[22]: Ridge(alpha=10)

```
In [23]: rr.score(x_test,y_test)
```

Out[23]: 0.9593584600514725

```
In [24]: la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

Out[24]: Lasso(alpha=10)

```
In [25]: la.score(x_test,y_test)
```

Out[25]: 0.9593542022002238

```
In [ ]:
```

