In [2]:
```
#Importing Libraries

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)


import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

Importing Datasets and Libraries

In [3]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [35]:
```
df=pd.read_csv('E:/Masters/ARTIFICIAL INTELLIGENCE ENGINEER/AI Capstone/Project 3-Re
df.head()
```

Out[35]:

	Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday
0	1	5	2015-07-31	5263	555	1	1	0	1
1	2	5	2015-07-31	6064	625	1	1	0	1
2	3	5	2015-07-31	8314	821	1	1	0	1
3	4	5	2015-07-31	13995	1498	1	1	0	1
4	5	5	2015-07-31	4822	559	1	1	0	1

Exploratory Data Analysis and Visualization

In [36]:
```
# shape of dataset
print('Shape of dataset')
print('*'*30)
print(f'rows : {df.shape[0]} ')
print(f'columns : {df.shape[1]}')
```

```
Shape of dataset
******************************
rows : 34565
columns : 9
```

In [37]:
```
# info
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 34565 entries, 0 to 34564
```

```
Data columns (total 9 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Store          34565 non-null  int64
 1   DayOfWeek      34565 non-null  int64
 2   Date           34565 non-null  object
 3   Sales          34565 non-null  int64
 4   Customers      34565 non-null  int64
 5   Open           34565 non-null  int64
 6   Promo          34565 non-null  int64
 7   StateHoliday   34565 non-null  int64
 8   SchoolHoliday  34565 non-null  int64
dtypes: int64(8), object(1)
memory usage: 2.4+ MB
```

In [38]:
```
# nan values in dataset
df.isna().sum()
```

Out[38]:
```
Store            0
DayOfWeek        0
Date             0
Sales            0
Customers        0
Open             0
Promo            0
StateHoliday     0
SchoolHoliday    0
dtype: int64
```

In [39]:
```
#No missing values
```

In [40]:
```
#unique values
df['StateHoliday'].unique()
```

Out[40]:
```
array([0], dtype=int64)
```

In [41]:
```
sh={'0':0, 'a':1, 'b':2, 'c':3,0:0}
```

In [42]:
```
df['StateHoliday']=df['StateHoliday'].map(sh)
```

In [43]:
```
df['SchoolHoliday'].unique()
```

Out[43]:
```
array([1, 0], dtype=int64)
```

In [44]:
```
# statistical info
df.describe().T
df.head()
```

Out[44]:

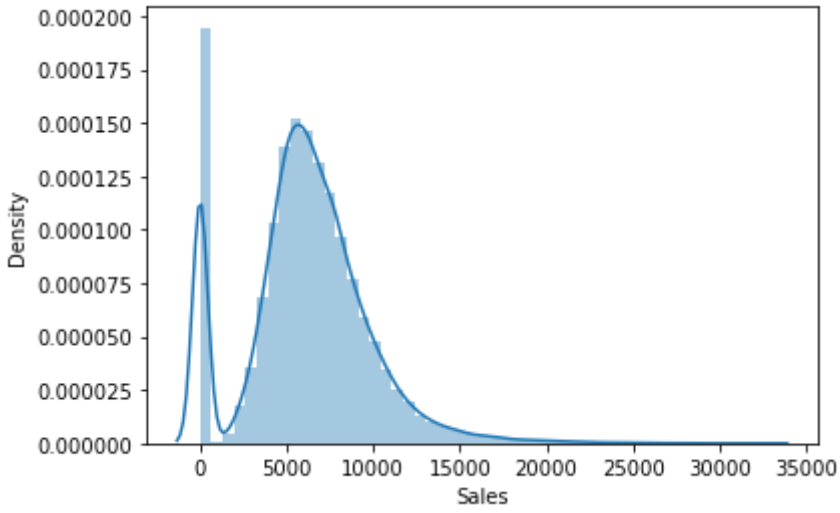
	Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday
0	1	5	2015-07-31	5263	555	1	1	0	1
1	2	5	2015-07-31	6064	625	1	1	0	1
2	3	5	2015-07-	8314	821	1	1	0	1

	Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday
			31						
3	4	5	2015-07-31	13995	1498	1	1	0	1
4	5	5	2015-07-31	4822	559	1	1	0	1

In [49]:
```
#Distribution Plot
sns.distplot(df['Sales'])
```

```
C:\Users\HP\anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarnin
g: `distplot` is a deprecated function and will be removed in a future version. Plea
se adapt your code to use either `displot` (a figure-level function with similar fle
xibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

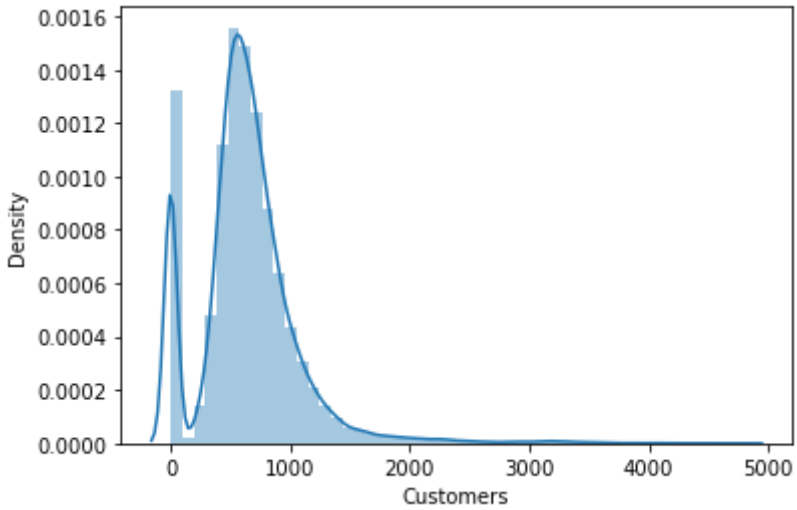
Out[49]: <AxesSubplot:xlabel='Sales', ylabel='Density'>



In [50]:
```
sns.distplot(df['Customers'])
```

```
C:\Users\HP\anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarnin
g: `distplot` is a deprecated function and will be removed in a future version. Plea
se adapt your code to use either `displot` (a figure-level function with similar fle
xibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

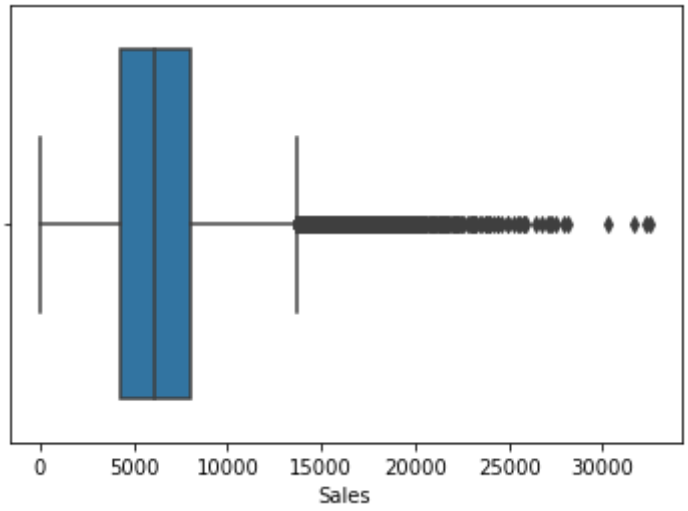
Out[50]: <AxesSubplot:xlabel='Customers', ylabel='Density'>



In [51]:
```
#Boxplot
sns.boxplot(df['Sales'])
```

```
C:\Users\HP\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pa
ss the following variable as a keyword arg: x. From version 0.12, the only valid pos
itional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
```

Out[51]: <AxesSubplot:xlabel='Sales'>



In [52]:
```
#feature engineering
df['day']=pd.to_datetime(df['Date'], format='%Y-%m-%d').dt.day
df['month']=pd.to_datetime(df['Date'], format='%Y-%m-%d').dt.month
df['year']=pd.to_datetime(df['Date'], format='%Y-%m-%d').dt.year
```

In [53]:
```
df.head()
```

Out[53]:

	Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday	day	m
0	1	5	2015-07-31	5263	555	1	1	0	1	31	
1	2	5	2015-07-31	6064	625	1	1	0	1	31	
2	3	5	2015-07-31	8314	821	1	1	0	1	31	
3	4	5	2015-07-31	13995	1498	1	1	0	1	31	
4	5	5	2015-07-31	4822	559	1	1	0	1	31	

In [54]:
```
df.tail()
```

Out[54]:

	Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday	da
34560	1111	3	2015-07-01	3701	351	1	1	0	1	

	Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday	da
34561	1112	3	2015-07-01	10620	716	1	1	0	1	
34562	1113	3	2015-07-01	8222	770	1	1	0	0	
34563	1114	3	2015-07-01	27071	3788	1	1	0	0	
34564	1115	3	2015-07-01	7701	447	1	1	0	0	

In [55]:
```
df.drop('Date', axis=1, inplace=True)
```

In [56]:
```
df.head()
```

Out[56]:

	Store	DayOfWeek	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday	day	month
0	1	5	5263	555	1	1	0	1	31	7
1	2	5	6064	625	1	1	0	1	31	7
2	3	5	8314	821	1	1	0	1	31	7
3	4	5	13995	1498	1	1	0	1	31	7
4	5	5	4822	559	1	1	0	1	31	7

In [57]:
```
df['StateHoliday'].unique()
```

Out[57]:
```
array([0], dtype=int64)
```

In [60]:
```
y=df['Sales']
X=df.drop('Sales', axis=1)
```

Model Building

In [61]:
```
from sklearn.model_selection import cross_val_score, train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2, random_state=42)
```

In [62]:
```
# linear regression
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(X_train,y_train)
pred_lr=lr.predict(X_test)
score_lr=cross_val_score(lr,X,y,cv=5)
print(score_lr)
```

```
[0.8272264  0.74292577 0.79467188 0.8325215  0.82853658]
```

In [63]:
```
score_lr.mean()
```

Out[63]:
```
0.8051764243690339
```

In [64]:
```
from sklearn.metrics import mean_absolute_error, mean_squared_error
mae_lr=mean_absolute_error(y_test,pred_lr)
print(mae_lr)
mse_lr=mean_squared_error(y_test,pred_lr)
print(mae_lr)
```

```
1020.9858020361146
1020.9858020361146
```

In [65]:
```
from sklearn.tree import DecisionTreeRegressor
dt=DecisionTreeRegressor()
dt.fit(X_train,y_train)
pred_dt=dt.predict(X_test)
score_dt=cross_val_score(dt,X,y,cv=5)
print(score_dt)
```

```
[0.78915821 0.72001873 0.78474704 0.80522181 0.80548156]
```

In [66]:
```
score_dt.mean()
```

Out[66]:
```
0.7809254685983696
```

In [75]:
```
mae_dt=mean_absolute_error(y_test,pred_dt)
print(mae_dt)
mse_dt=mean_squared_error(y_test,pred_dt)
print(mae_dt)
```

```
1029.7675394184869
1029.7675394184869
```

In [74]:
```
from sklearn.ensemble import RandomForestRegressor
rf=RandomForestRegressor()
rf.fit(X_train,y_train)
pred_rf=rf.predict(X_test)
score_rf=cross_val_score(rf,X,y,cv=5)
print(score_rf)
```

```
[0.86988681 0.85379213 0.88073415 0.90809576 0.90044387]
```

In [76]:
```
score_rf.mean()
```

Out[76]:
```
0.8825905420561895
```

In [77]:
```
mae_rf=mean_absolute_error(y_test,pred_rf)
print(mae_rf)
mse_rf=mean_squared_error(y_test,pred_rf)
print(mae_rf)
```

```
820.9274482858383
820.9274482858383
```

In [ ]:
```
#The Best Model is Random forest with accuracy Greater than 88% .
```