



# Coffee Shop Revenue Prediction

## Prediction using Scikit-learn

This presentation explores how to utilize Scikit-learn for predicting coffee shop revenue based on various factors.

**A** by Arindam Adhikari

# Data Import and Library Setup

Import necessary libraries:

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

Load data from a CSV file:

```
df =
pd.read_csv(r'C:\Downloads\coffee_shop_revenue.
csv')
```

	Number_of_Customers_Per_Day	Average_Order_Value	Operating_Hours_Per_Day	Number_of_Employees	Marketing_Spend_Per_Day	Location_Foot_Traffic	Daily_Revenue
0	152	6.74	14	4	106.62	97	1547.81
1	485	4.50	12	8	57.83	744	2084.68
2	398	9.09	6	6	91.76	636	3118.39
3	320	8.48	17	4	462.63	770	2912.20
4	156	7.44	17	2	412.52	232	1663.42

# Handling Missing and Duplicate Data

Identify missing values:

```
data.isnull().sum()
```

Impute missing values:

```
data['column'].fillna(data['column'].mean(), inplace=True)
```

Identify duplicates:

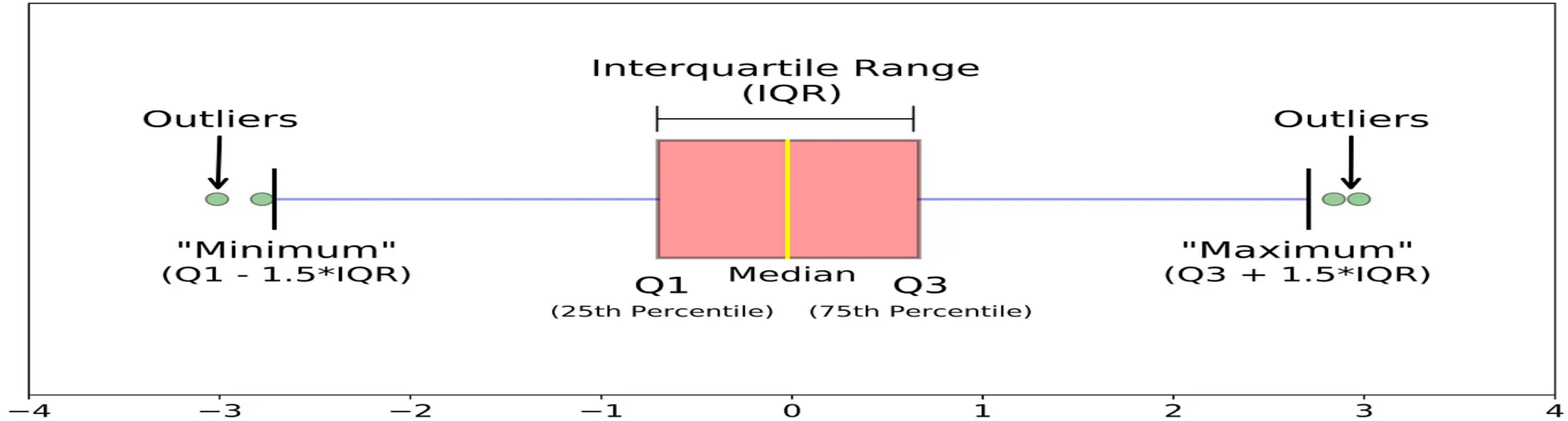
```
data.duplicated().sum()
```

Remove duplicates:

```
data.drop_duplicates(inplace=True)
```

	Msite	Value
1	1123	1449
2	1899	1200
3	1689	1555
4	1690	1508
4	1895	1198
5	1890	1100
6	1599	1600
0	0	4200





## Outlier Detection and Treatment Strategies



Box plots can visually identify outliers.



Remove outliers using IQR method or other suitable techniques.

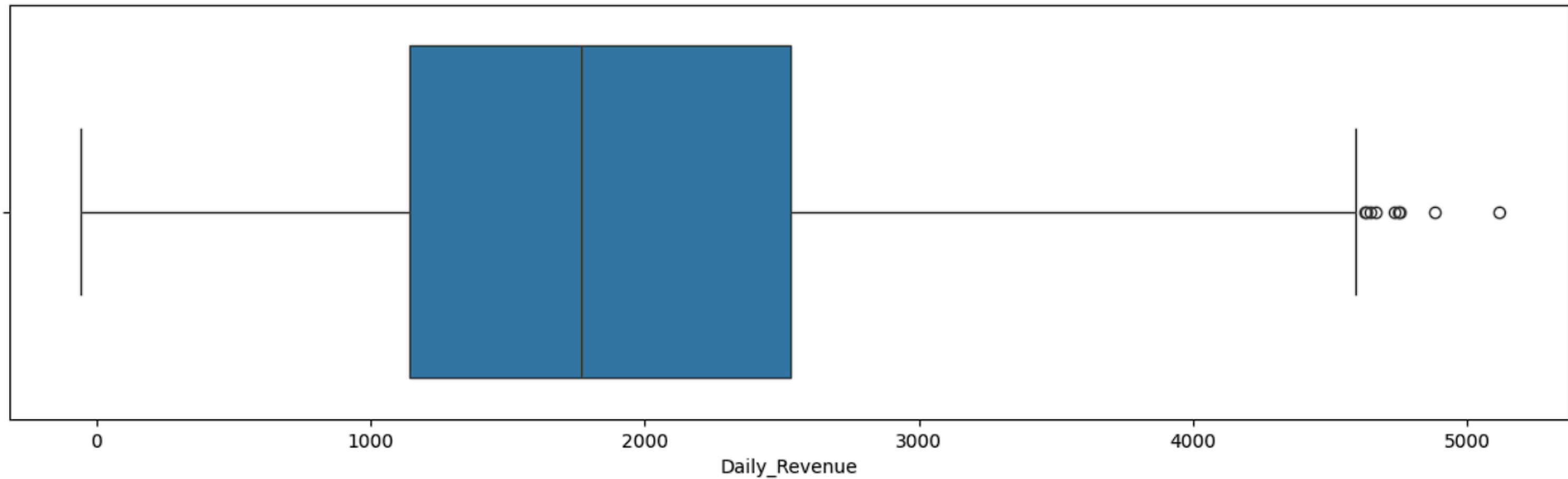
# HOW TO FIND OUTLIER PRESENT OR NOT

```
df.describe()
```

	Number_of_Customers_Per_Day	Average_Order_Value	Operating_Hours_Per_Day	Marketing_Spend_Per_Day	Location_Foot_Traffic	Daily_Revenue
count	2000.000000	2000.000000	2000.000000	2000.000000	2000.000000	2000.000000
mean	274.296000	6.261215	11.667000	252.614160	534.893500	1917.325940
std	129.441933	2.175832	3.438608	141.136004	271.662295	976.202746
min	50.000000	2.500000	6.000000	10.120000	50.000000	-58.950000
25%	164.000000	4.410000	9.000000	130.125000	302.000000	1140.085000
50%	275.000000	6.300000	12.000000	250.995000	540.000000	1770.775000
75%	386.000000	8.120000	15.000000	375.352500	767.000000	2530.455000
max	499.000000	10.000000	17.000000	499.740000	999.000000	5114.600000

In the dataset the large difference between the minimum (-58.95), mean (1917.33), and maximum (5114.60) values in the **Daily\_Revenue** column suggests potential outliers.

```
: # checking for outlier  
plt.figure(figsize=(15,4))  
sns.boxplot(x="Daily_Revenue",data=df)  
plt.show()  
plt.savefig("outlier.jpg")
```



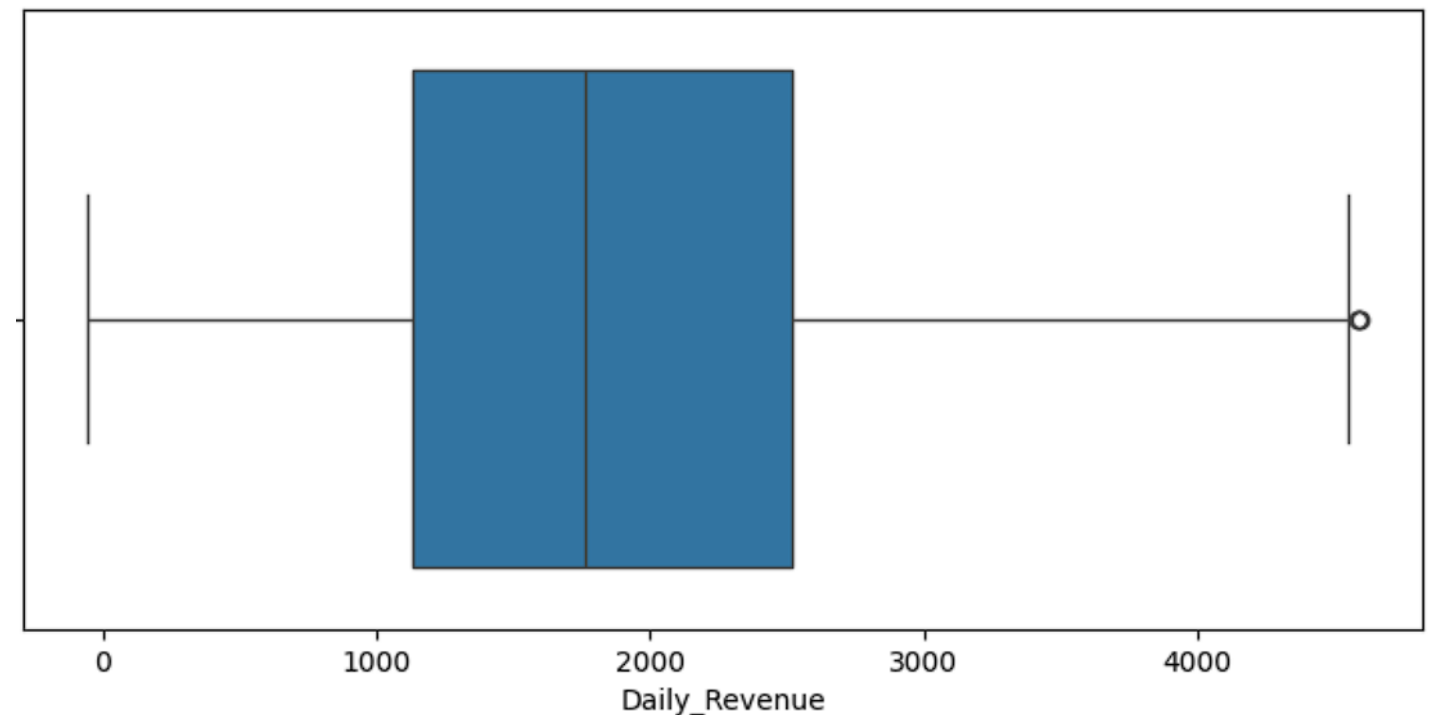
## *Remove the outlier by the INTER QUARTILE RANGE (IQR) method*

```
# remove the outlier
q1 = df["Daily_Revenue"].quantile(0.25)
q3 = df ["Daily_Revenue"].quantile(0.75)
IQR= (q3 - q1)
```

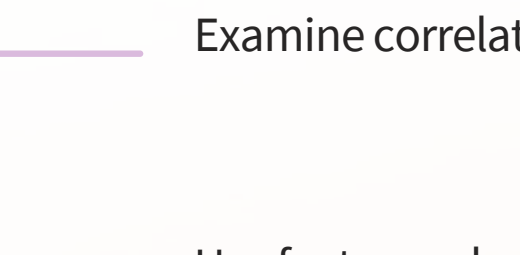
```
min_range= q1 -(1.5*IQR)
max_range=q3 +(1.5*IQR)
df1=df[df["Daily_Revenue"]<= max_range]
```

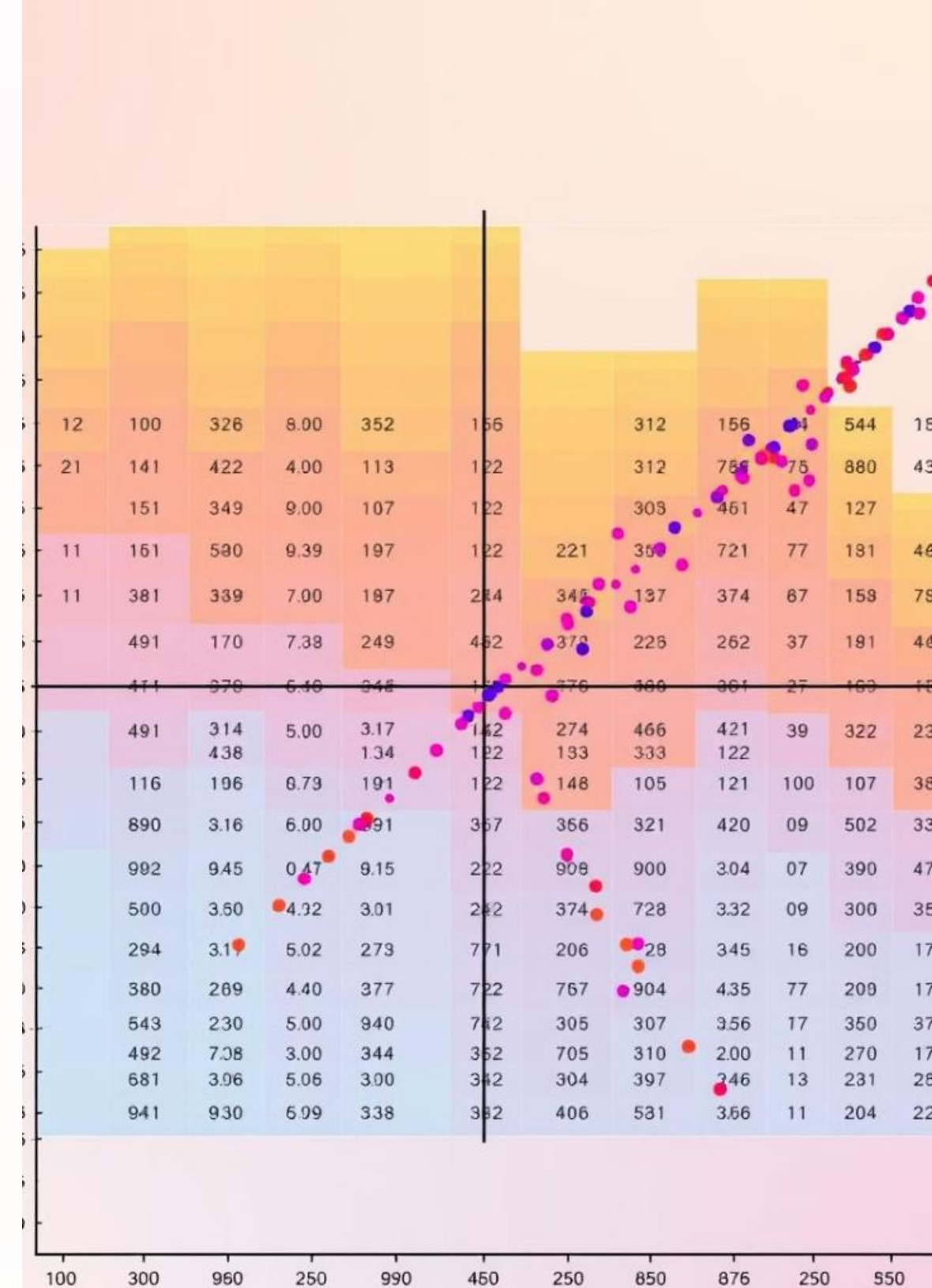
## *Checking the outlier present or not*

```
# checking for outlier
plt.figure(figsize=(9,4))
sns.boxplot(x="Daily_Revenue",data=df1)
plt.show()
```



# Feature Selection: Analyzing Input-Output Relationships

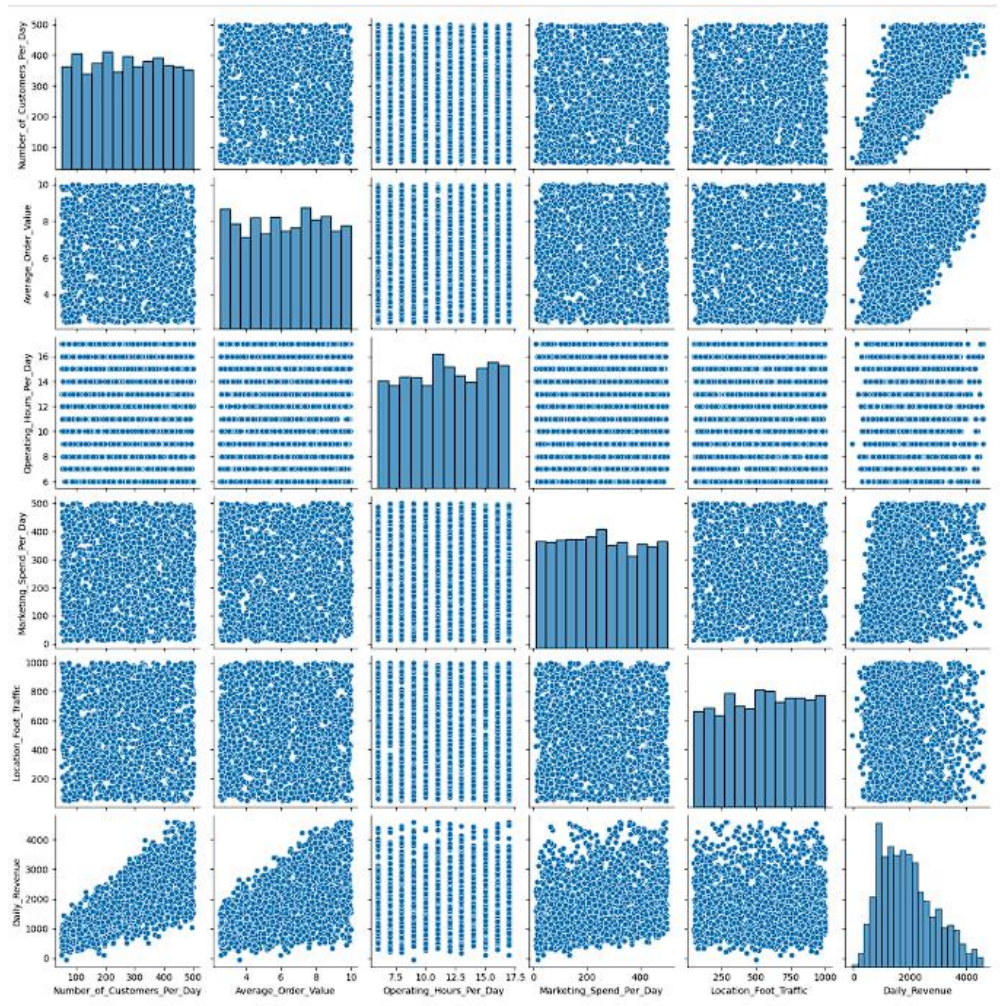
- 
- 1 Examine correlation between features and target variable.
  - 2 Use feature selection methods such as SelectKBest or recursive feature elimination.
  - 3 Remove irrelevant features to improve model accuracy.





## Find relationship between output and input features

```
sns.pairplot(data=df1)
plt.show()
```



```
sns.heatmap(data=df1.corr(),annot=True)
plt.show()
plt.savefig("heatmap.jpg")
```



, from the pair plot, we can see that:

- **Number\_of\_Customers\_Per\_Day vs. Daily\_Revenue** shows a strong positive linear relationship.
- **Average\_Order\_Value vs. Daily\_Revenue** also exhibits a linear trend.
- **Marketing\_Spend\_Per\_Day vs. Daily\_Revenue** has a somewhat positive linear correlation.

# Marchnshe Lewinling

## Mataligtloy terder

- 1 training
- 2 pranges
- 2 mocnatling
- 3 pretgages
- 4 closs
- 5 most leaining
- 6 pretection
- 9 culeses
- 8 aorges
- 10 stopling learning
- 1 mat the masage
- 3 ldocalptions
- 7 motellactons
- 8 matleagen

sear1

- 184199
- 108191
- 1293.97
- 1361628
- 149199

Incranirang Learne learniessclalliq)

```
( Traniriling;terrann.laclo:
  erectfaling;:aveunr, compDStairnastonce:
  nustfuide.tomqDliesLlerizstle:
  nushifzids.unwbolretestes
  rechivang;terrann.tcompSttestlagere:
  machiriang.comornbit.onnolasporeveolstarion):
  machiriang:romalesstiealsatres
  nustfuide;:raoblzeretes
  nushirang;torome conplaget/Statastorlo;
)
PORPAN.CADT
```

sear1

- 1138413
- 1375195
- 106697
- 2126123
- 0096509
- 107137

Incranirang Learne learniessctanic)

```
( Traniriling;crastoms
  entcfraide;:reger.compollestiesIS.atoy)
  eutcfurde:ressat.coordlstimn.ahigninstretl8
  muscfuugg(/ resechwiq8glastiow:lo)
  mashfaide;:reser.conrpoolset.achistSAle()
  eutcfurde:rcestfleafnes:
  =F dettimd;:egpilic
) Ty Aopler
undaffand;:(torannaltacchasting-fcar19.
rascfurde;:rreste.vojiesstaling:
unesffurde;:(trannvlocq/lnclastlagolisstwtter).
naseffurde;:restfilesatess
=F asttine;:comnplierage
```

Progice Rectiom

Trning: Pecluting

01.1	10000rliys/lays
41.2	10000rliys/langes
72.2	10000rtivs/lages
33.4	10000rliys/banges
33.5	10000rlave/langes

Training Model Trating:

1ST 109422	02,008	Im005.1185/PayS
1ST 109723	02,099	Inet1064173/Payla)
1ST 121016	2019	In600S116s/4L1)
1ST 105722	00,008	Inn0024415/Pappls
SST 1741	33,067	Im005Iaste/Payle)

# Machine Learning Model Deployment and Prediction

1

Split data into training and testing sets.

2

Train a linear regression model on the training data.

3

Make predictions on the testing data.

## Separate the all-input features and output

```
x= df1.iloc[:, :3]
y=df1["Daily_Revenue"]
```

x

	Number_of_Customers_Per_Day	Average_Order_Value	Marketing_Spend_Per_Day
0	152	6.74	106.62
1	485	4.50	57.83
2	398	9.09	91.76
3	320	8.48	462.63
4	156	7.44	412.52
...	...	...	...
1995	372	6.41	466.11
1996	105	3.01	12.62



## *Split the the data into two parts and Deployment Machine Learning Model*

```
# split the data into 2 parts
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.25,random_state=35)
```

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

▼ LinearRegression



LinearRegression()

```
# testing data
y_pred=lr.predict(x_test)
|
```



```
lr.score(x_test,y_test)*100
```

90.14407476562933

```
lr.coef_
```

```
array([ 5.46355433, 238.07113104,  1.53244795])
```

```
lr.intercept_
```

```
-1466.7042318707547
```

## Model Evaluation: Accuracy, Coefficients, and Intercept

90.144

Accuracy

Evaluate model performance using metrics like R-squared or mean squared error.

5.463, 238.071, 1.532

Coefficients

Interpret coefficients to understand the impact of each feature on revenue.

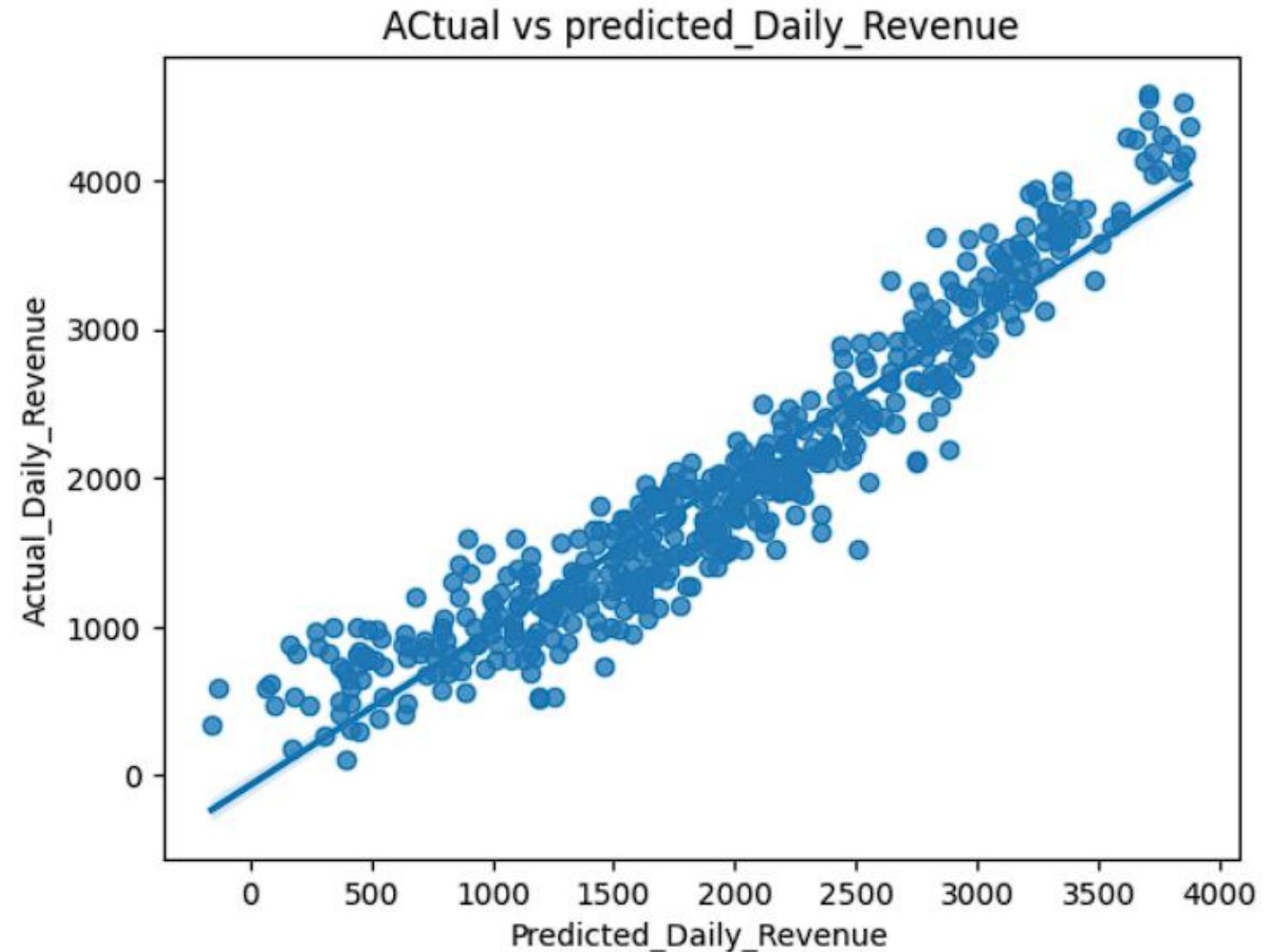
-1466.704

Intercept

The intercept represents the baseline revenue when all features are zero.

# Actual vs predicted\_Daily\_Revenue

```
sns.regplot(x=y_pred, y=y_test)
plt.xlabel("Predicted_Daily_Revenue")
plt.ylabel('Actual_Daily_Revenue')
plt.title("Actual vs predicted_Daily_Revenue")
plt.show()
plt.savefig("Daily_Revenue_prediction.jpg")
```





## Conclusion: Key Findings and Future Improvements

The model provides a valuable tool for predicting coffee shop revenue, helping to inform decision-making and optimize operations. Future improvements include exploring more advanced models, incorporating seasonal trends, and expanding the dataset for better accuracy.

GITHUB LINK - [CLICK HERE](#)