Customer Spending Score Prediction

In [1]:
```
import pandas as pd
import numpy as np
```

In [2]:
```
df=pd.read_csv(r'C:\Users\USER\Downloads\Mall_Customers.csv')
```

In [3]:
```
df.head()
```

Out[3]:

	CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40

In [4]:
```
df.tail(3)
```

Out[4]:

	CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
197	198	Male	32	126	74
198	199	Male	32	137	18
199	200	Male	30	137	83

In [5]:
```
df.sample(4)
```

Out[5]:

	CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
127	128	Male	40	71	95
189	190	Female	36	103	85
47	48	Female	27	40	47
139	140	Female	35	74	72

NEED TO RENAME SOME COLUMN NAME

In [6]:
```
df.rename(columns={'CustomerID':'ID'},inplace=True)
```

In [7]:
```
df.head()
```

Out[7]:

	ID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40

In [8]:
```
df.rename(columns={"Annual Income (k$)": 'Annual Income'},
inplace=True)
df.rename(columns = {'Spending Score (1-100)':'Spending_Score'},
inplace=True)
```

In [9]:
```
df.head()
```

Out[9]:

	ID	Genre	Age	Annual Income	Spending_Score
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40

LETS CHECK DATA

In [10]:
```
df.info()
```

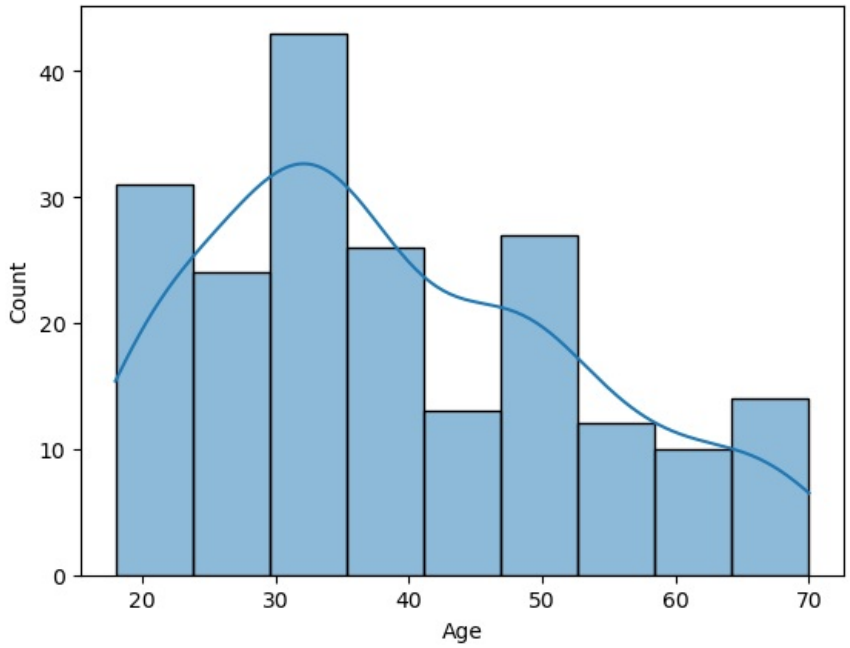
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 5 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   ID              200 non-null    int64
 1   Genre           200 non-null    object
 2   Age             200 non-null    int64
 3   Annual Income   200 non-null    int64
 4   Spending_Score  200 non-null    int64
dtypes: int64(4), object(1)
memory usage: 7.9+ KB
```

GLANCE OF DATA AND ITS RELATION USING BASIC VISUALIZATION

In [11]:
```
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

In [12]:
```
sns.histplot(df,x="Age",kde=True)
```

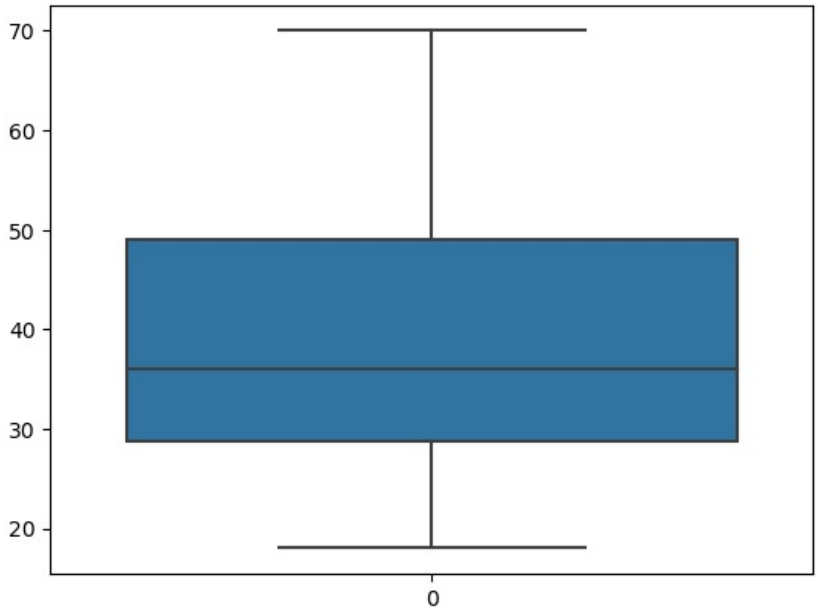
Out[12]:
```
<Axes: xlabel='Age', ylabel='Count'>
```



From above Graph, we can say that age group data is rightly sekewed means there may be some outlier, it can be more easily seen via box-plot.

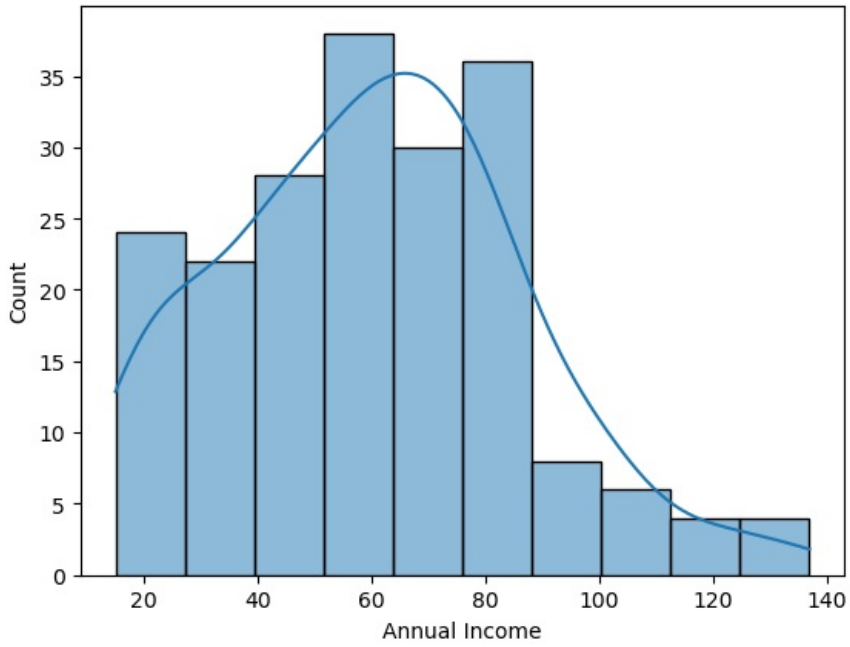
In [13]:
```
sns.boxplot(df["Age"])
```

Out[13]:
```
<Axes: >
```



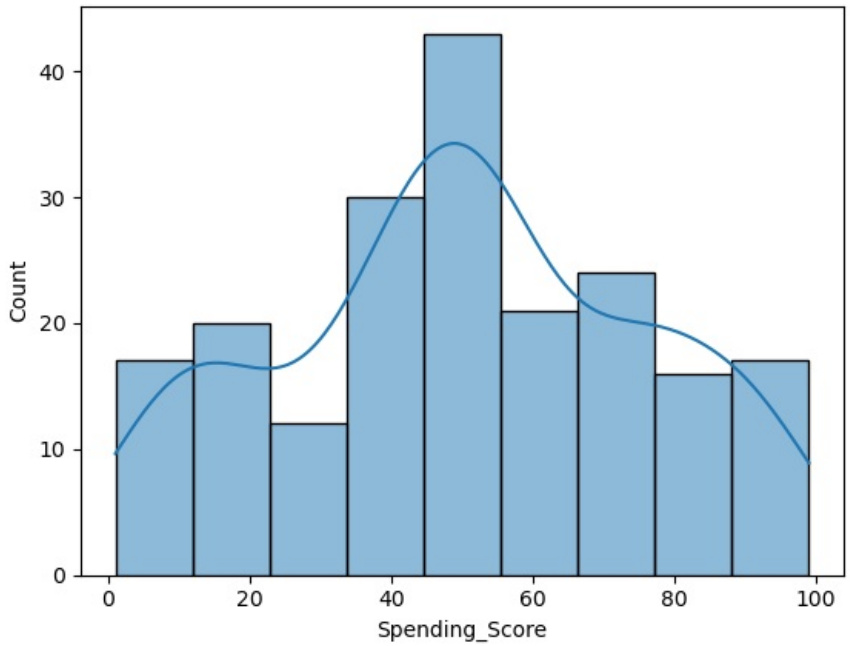
In [14]:
```
sns.histplot(df,x="Annual Income",kde=True)
```

Out[14]: <Axes: xlabel='Annual Income', ylabel='Count'>



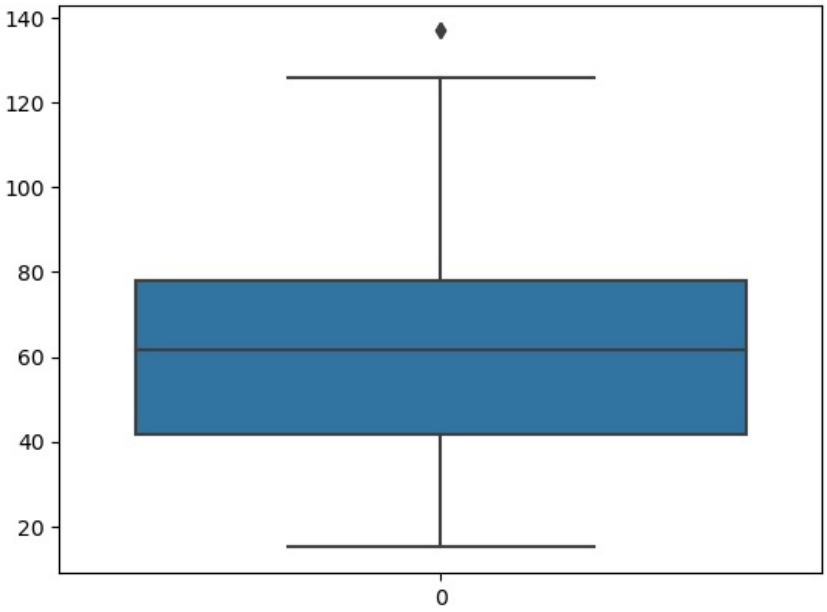
In [15]:
```
sns.histplot(df,x="Spending_Score",kde=True)
```

Out[15]: <Axes: xlabel='Spending_Score', ylabel='Count'>



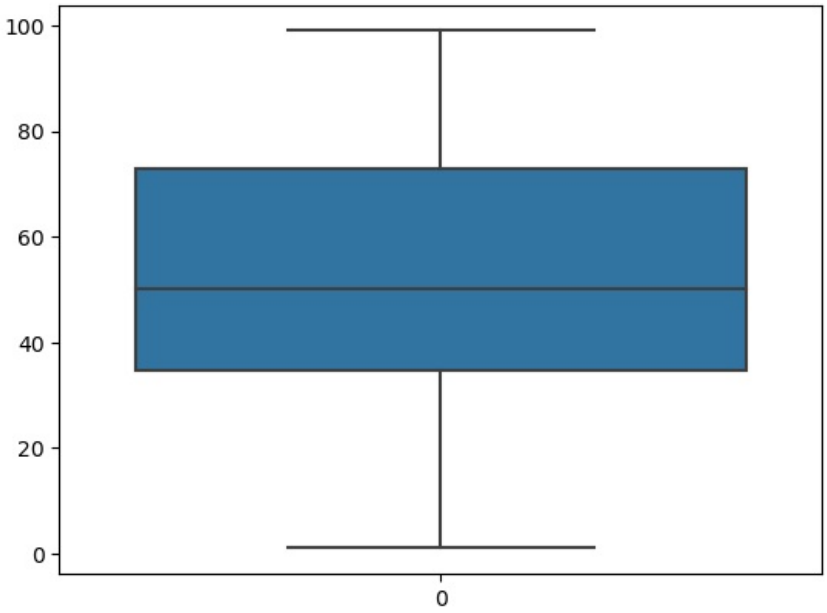
In [16]:
```
sns.boxplot(df["Annual Income"])
```

Out[16]: <Axes: >



In [17]:
```
sns.boxplot(df["Spending_Score"])
```

Out[17]:
```
<Axes: >
```

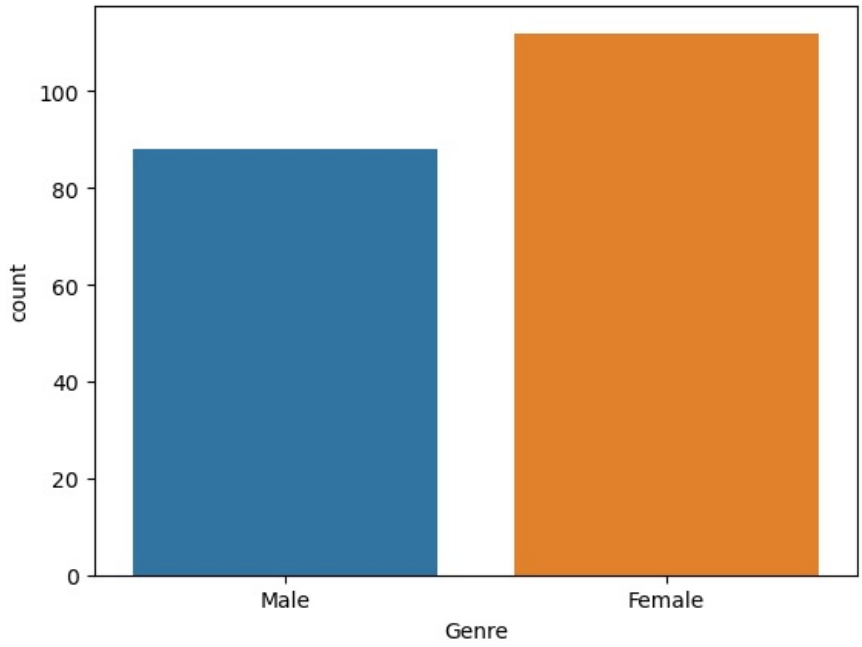


From above graphs, we only finds outlier in Annual Income.

Similarly, for objective data, we can just look for countplot for now

In [18]:
```
sns.countplot(data=df,x="Genre")
```

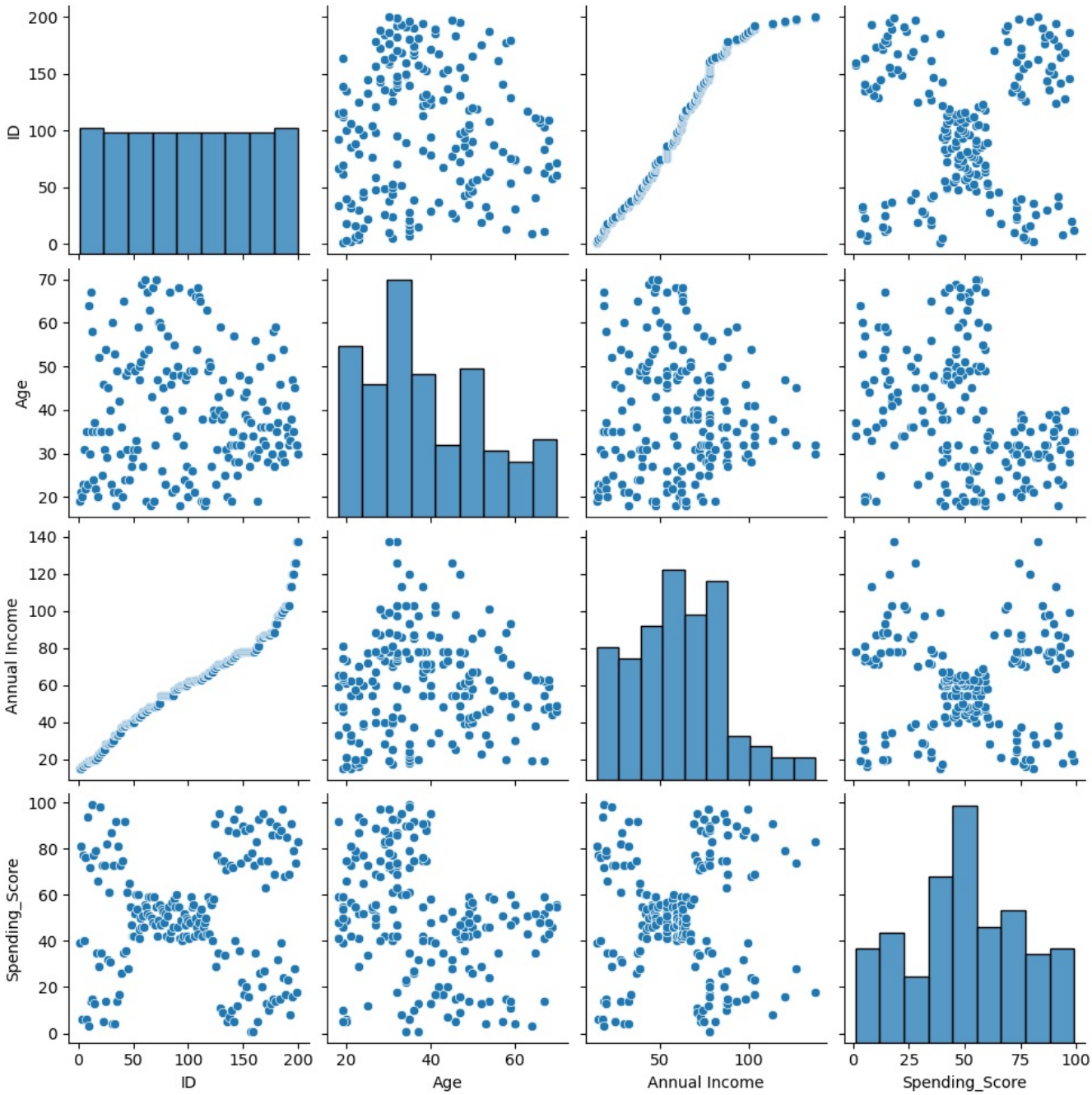
Out[18]: <Axes: xlabel='Genre', ylabel='count'>



AT A GLANCE, WE CAN SEE RELATION BETWEEN ALL COLUMNS AT A ONCE

In [19]:
```
sns.pairplot(df)
```

Out[19]: <seaborn.axisgrid.PairGrid at 0x1e550c71a10>



We see few visualization, now lets see our data once and lets drop S.N

In [20]:
```
df.head()
```

Out[20]:

	ID	Genre	Age	Annual Income	Spending_Score
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40

Checking for Null values before separating it in features(X) and labels(y)

In [21]:
```
df.isnull().sum()
```

Out[21]:
```
ID                0
Genre             0
Age               0
Annual Income     0
Spending_Score    0
dtype: int64
```

In [22]:
```
y = df['Spending_Score']
X = df.drop(columns='Spending_Score')
```

In [23]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
```

```
random_state=45)
```

Lets check shape whether it got splited or not

In [24]:
```
X_train.shape, y_train.shape
```

Out[24]:
```
((160, 4), (160,))
```

In [25]:
```
X_test.shape, y_test.shape
```

Out[25]:
```
((40, 4), (40,))
```

Before feeding our data to a machine-learning model we need to convert the categorical column to a numerical column, and we need to do column transfer too so,

In [30]:
```
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import StandardScaler, OneHotEncoder
```

In [31]:
```
# Define the numerical and categorical columns
numerical_cols = ['Age', 'Annual Income']
categorical_cols = ['Genre']
```

In [32]:
```
# Define the preprocessing steps
numeric_transformer = StandardScaler()
categorical_transformer = OneHotEncoder(handle_unknown='ignore')
```

In [33]:
```
# Create the ColumnTransformer
preprocessor = ColumnTransformer(
    transformers=[
        ('num', numeric_transformer, numerical_cols),
        ('cat', categorical_transformer, categorical_cols)
    ])
```

In [34]:
```
# Fit and transform the training data
X_train_transformed = preprocessor.fit_transform(X_train)

# Transform the test data (using the same transformations as training data)
X_test_transformed = preprocessor.transform(X_test)
```

Lets see now how data looks like

In [35]:
```
X_train_transformed
```

Out[35]:
```
array([[-8.62675496e-01, -3.84078599e-02,  1.00000000e+00,
         0.00000000e+00],
       [ 9.60199298e-01,  1.04377587e+00,  1.00000000e+00,
         0.00000000e+00],
       [-1.44599543e+00,  5.02684003e-01,  0.00000000e+00,
         1.00000000e+00],
       [ 2.12683917e+00, -5.02200885e-01,  1.00000000e+00,
         0.00000000e+00],
       [-6.43930521e-01,  1.04377587e+00,  1.00000000e+00,
         0.00000000e+00],
       [ 1.76226421e+00,  1.54839234e-01,  1.00000000e+00,
         0.00000000e+00],
       [-7.16845513e-01,  4.64034584e-01,  1.00000000e+00,
         0.00000000e+00],
       [-1.15433546e+00, -1.23653984e+00,  1.00000000e+00,
         0.00000000e+00],
       [-1.44599543e+00, -5.79499723e-01,  0.00000000e+00,
         1.00000000e+00],
       [ 5.22709347e-01, -2.70304372e-01,  1.00000000e+00,
         0.00000000e+00],
       [-7.16845513e-01,  1.43027005e+00,  1.00000000e+00,
         0.00000000e+00],
       [ 1.58134388e-01,  1.46891947e+00,  1.00000000e+00,
         0.00000000e+00],
       [ 1.39768925e+00,  1.04377587e+00,  0.00000000e+00,
         1.00000000e+00],
       [-4.25185546e-01,  9.66477028e-01,  1.00000000e+00,
         0.00000000e+00],
       [-2.06440570e-01, -9.27344491e-01,  1.00000000e+00,
         0.00000000e+00],
       [ 7.41454323e-01, -2.70304372e-01,  1.00000000e+00,
         0.00000000e+00],
       [ 1.10602928e+00,  1.54621831e+00,  1.00000000e+00,
         0.00000000e+00],
       [ 1.58134388e-01,  1.62351715e+00,  1.00000000e+00,
         0.00000000e+00],
       [-1.30016545e+00, -2.70304372e-01,  1.00000000e+00,
         0.00000000e+00],
       [-2.06440570e-01,  1.00512645e+00,  1.00000000e+00,
         0.00000000e+00],
       [ 1.47060424e+00, -2.70304372e-01,  0.00000000e+00,
```

```
  1.00000000e+00],
[-1.44599543e+00,  1.16189815e-01,  0.00000000e+00,
  1.00000000e+00],
[ 4.49794355e-01, -2.70304372e-01,  1.00000000e+00,
  0.00000000e+00],
[-1.44599543e+00,  1.54839234e-01,  1.00000000e+00,
  0.00000000e+00],
[ 2.12683917e+00, -7.70572786e-02,  1.00000000e+00,
  0.00000000e+00],
[-2.79355562e-01,  5.02684003e-01,  1.00000000e+00,
  0.00000000e+00],
[-7.16845513e-01,  6.95931097e-01,  1.00000000e+00,
  0.00000000e+00],
[ 7.41454323e-01, -1.08194217e+00,  1.00000000e+00,
  0.00000000e+00],
[-1.51891042e+00,  1.54839234e-01,  1.00000000e+00,
  0.00000000e+00],
[-4.98100538e-01, -3.84078599e-02,  1.00000000e+00,
  0.00000000e+00],
[-6.43930521e-01,  6.57281678e-01,  1.00000000e+00,
  0.00000000e+00],
[ 3.03964372e-01, -5.02200885e-01,  1.00000000e+00,
  0.00000000e+00],
[ 8.14369314e-01, -1.15706697e-01,  1.00000000e+00,
  0.00000000e+00],
[-2.79355562e-01, -1.54573519e+00,  1.00000000e+00,
  0.00000000e+00],
[-1.15433546e+00,  3.88909776e-02,  1.00000000e+00,
  0.00000000e+00],
[-1.51891042e+00, -7.70572786e-02,  0.00000000e+00,
  1.00000000e+00],
[ 1.47060424e+00,  3.86735746e-01,  0.00000000e+00,
  1.00000000e+00],
[ 1.47060424e+00,  1.23702296e+00,  0.00000000e+00,
  1.00000000e+00],
[-5.71015529e-01, -8.50045654e-01,  1.00000000e+00,
  0.00000000e+00],
[ 1.54351923e+00, -1.19789042e+00,  0.00000000e+00,
  1.00000000e+00],
[ 2.05392417e+00,  3.88909776e-02,  0.00000000e+00,
  1.00000000e+00],
[ 5.95624339e-01, -3.84078599e-02,  1.00000000e+00,
  0.00000000e+00],
[-1.30016545e+00, -1.77763170e+00,  0.00000000e+00,
  1.00000000e+00],
[ 3.03964372e-01,  6.57281678e-01,  0.00000000e+00,
  1.00000000e+00],
[-1.37308044e+00,  2.41558867e-04,  0.00000000e+00,
  1.00000000e+00],
[ 6.68539331e-01, -2.70304372e-01,  0.00000000e+00,
  1.00000000e+00],
[-1.00850548e+00, -1.42978694e+00,  0.00000000e+00,
  1.00000000e+00],
[-3.52270554e-01,  6.57281678e-01,  1.00000000e+00,
  0.00000000e+00],
[ 7.41454323e-01,  1.54839234e-01,  1.00000000e+00,
  0.00000000e+00],
[-8.62675496e-01,  6.57281678e-01,  1.00000000e+00,
  0.00000000e+00],
[-6.43930521e-01, -1.04329275e+00,  1.00000000e+00,
  0.00000000e+00],
[-4.25185546e-01, -7.34097398e-01,  0.00000000e+00,
  1.00000000e+00],
[-5.71015529e-01, -6.95447979e-01,  1.00000000e+00,
  0.00000000e+00],
[-4.98100538e-01,  4.64034584e-01,  0.00000000e+00,
  1.00000000e+00],
[-4.98100538e-01,  2.93759738e+00,  0.00000000e+00,
  1.00000000e+00],
[ 8.52193966e-02,  1.00512645e+00,  0.00000000e+00,
  1.00000000e+00],
[-7.16845513e-01, -8.11396235e-01,  1.00000000e+00,
  0.00000000e+00],
[ 2.31049380e-01,  9.66477028e-01,  0.00000000e+00,
  1.00000000e+00],
[-2.79355562e-01, -1.46843635e+00,  1.00000000e+00,
  0.00000000e+00],
[ 8.52193966e-02, -3.84078599e-02,  1.00000000e+00,
  0.00000000e+00],
[-4.98100538e-01,  5.41333421e-01,  0.00000000e+00,
  1.00000000e+00],
[-8.62675496e-01, -8.11396235e-01,  1.00000000e+00,
  0.00000000e+00],
[ 8.52193966e-02,  3.09436909e-01,  1.00000000e+00,
  0.00000000e+00],
[ 1.10602928e+00,  7.75403964e-02,  0.00000000e+00,
  1.00000000e+00],
[ 5.95624339e-01, -4.63551466e-01,  1.00000000e+00,
  0.00000000e+00],
```

```
[ 6.68539331e-01,  6.18632259e-01,  0.00000000e+00,
  1.00000000e+00],
[-1.37308044e+00, -1.73898229e+00,  1.00000000e+00,
  0.00000000e+00],
[ 2.27266915e+00, -4.63551466e-01,  0.00000000e+00,
  1.00000000e+00],
[-1.37308044e+00, -9.27344491e-01,  1.00000000e+00,
  0.00000000e+00],
[-1.08142047e+00, -8.88695073e-01,  0.00000000e+00,
  1.00000000e+00],
[-6.06105869e-02,  2.01001133e+00,  1.00000000e+00,
  0.00000000e+00],
[-9.35590488e-01, -2.70304372e-01,  0.00000000e+00,
  1.00000000e+00],
[-5.71015529e-01, -1.39113752e+00,  0.00000000e+00,
  1.00000000e+00],
[ 7.41454323e-01,  3.88909776e-02,  1.00000000e+00,
  0.00000000e+00],
[-4.25185546e-01,  2.01001133e+00,  0.00000000e+00,
  1.00000000e+00],
[-1.15433546e+00, -2.70304372e-01,  1.00000000e+00,
  0.00000000e+00],
[-1.33525579e-01,  1.39162063e+00,  1.00000000e+00,
  0.00000000e+00],
[-1.15433546e+00, -1.66168345e+00,  1.00000000e+00,
  0.00000000e+00],
[ 7.41454323e-01, -7.34097398e-01,  1.00000000e+00,
  0.00000000e+00],
[ 1.98100918e+00,  7.75403964e-02,  1.00000000e+00,
  0.00000000e+00],
[ 8.14369314e-01,  2.32138071e-01,  1.00000000e+00,
  0.00000000e+00],
[-6.43930521e-01,  1.46891947e+00,  0.00000000e+00,
  1.00000000e+00],
[ 5.95624339e-01,  3.86735746e-01,  0.00000000e+00,
  1.00000000e+00],
[ 1.54351923e+00, -4.24902047e-01,  1.00000000e+00,
  0.00000000e+00],
[-5.71015529e-01,  3.48086328e-01,  1.00000000e+00,
  0.00000000e+00],
[-6.43930521e-01, -1.62303403e+00,  1.00000000e+00,
  0.00000000e+00],
[-2.79355562e-01, -1.42978694e+00,  0.00000000e+00,
  1.00000000e+00],
[-1.00850548e+00,  4.25385165e-01,  1.00000000e+00,
  0.00000000e+00],
[-1.22725046e+00, -1.54356116e-01,  1.00000000e+00,
  0.00000000e+00],
[ 9.60199298e-01, -1.46843635e+00,  0.00000000e+00,
  1.00000000e+00],
[-2.06440570e-01,  9.27827609e-01,  1.00000000e+00,
  0.00000000e+00],
[-4.98100538e-01,  5.79982840e-01,  1.00000000e+00,
  0.00000000e+00],
[ 1.25185926e+00,  6.95931097e-01,  1.00000000e+00,
  0.00000000e+00],
[-6.06105869e-02, -2.70304372e-01,  0.00000000e+00,
  1.00000000e+00],
[-1.22725046e+00, -1.70033287e+00,  1.00000000e+00,
  0.00000000e+00],
[-3.52270554e-01, -1.15706697e-01,  1.00000000e+00,
  0.00000000e+00],
[ 5.95624339e-01, -6.95447979e-01,  0.00000000e+00,
  1.00000000e+00],
[ 8.52193966e-02,  3.86735746e-01,  0.00000000e+00,
  1.00000000e+00],
[ 6.68539331e-01, -3.84078599e-02,  0.00000000e+00,
  1.00000000e+00],
[ 1.03311429e+00, -5.79499723e-01,  0.00000000e+00,
  1.00000000e+00],
[-4.98100538e-01,  1.62351715e+00,  1.00000000e+00,
  0.00000000e+00],
[ 5.95624339e-01,  6.57281678e-01,  1.00000000e+00,
  0.00000000e+00],
[-1.30016545e+00,  3.88909776e-02,  1.00000000e+00,
  0.00000000e+00],
[ 2.27266915e+00, -5.79499723e-01,  0.00000000e+00,
  1.00000000e+00],
[ 4.49794355e-01,  2.51245378e+00,  1.00000000e+00,
  0.00000000e+00],
[-2.79355562e-01, -1.27518926e+00,  0.00000000e+00,
  1.00000000e+00],
[-9.35590488e-01,  3.88909776e-02,  0.00000000e+00,
  1.00000000e+00],
[ 5.22709347e-01, -1.39113752e+00,  1.00000000e+00,
  0.00000000e+00],
[ 6.68539331e-01,  2.41558867e-04,  0.00000000e+00,
  1.00000000e+00],
[-1.33525579e-01,  6.57281678e-01,  0.00000000e+00,
```

```
  1.00000000e+00],
[-2.79355562e-01,  2.28055727e+00,  1.00000000e+00,
  0.00000000e+00],
[-1.44599543e+00,  7.75403964e-02,  1.00000000e+00,
  0.00000000e+00],
[ 3.76879364e-01,  6.57281678e-01,  1.00000000e+00,
  0.00000000e+00],
[-1.00850548e+00,  6.18632259e-01,  0.00000000e+00,
  1.00000000e+00],
[ 8.14369314e-01, -6.95447979e-01,  1.00000000e+00,
  0.00000000e+00],
[-2.79355562e-01,  1.23702296e+00,  0.00000000e+00,
  1.00000000e+00],
[ 1.10602928e+00, -5.40850304e-01,  1.00000000e+00,
  0.00000000e+00],
[ 1.23044049e-02,  3.09436909e-01,  0.00000000e+00,
  1.00000000e+00],
[ 8.52193966e-02, -2.70304372e-01,  0.00000000e+00,
  1.00000000e+00],
[-5.71015529e-01, -8.11396235e-01,  1.00000000e+00,
  0.00000000e+00],
[ 3.76879364e-01,  4.64034584e-01,  1.00000000e+00,
  0.00000000e+00],
[-2.79355562e-01, -1.62303403e+00,  1.00000000e+00,
  0.00000000e+00],
[ 6.68539331e-01, -8.50045654e-01,  0.00000000e+00,
  1.00000000e+00],
[ 2.12683917e+00,  7.75403964e-02,  0.00000000e+00,
  1.00000000e+00],
[-2.79355562e-01, -1.66168345e+00,  1.00000000e+00,
  0.00000000e+00],
[-7.89760505e-01,  6.18632259e-01,  0.00000000e+00,
  1.00000000e+00],
[ 8.14369314e-01,  9.27827609e-01,  0.00000000e+00,
  1.00000000e+00],
[ 1.17894427e+00, -1.54356116e-01,  1.00000000e+00,
  0.00000000e+00],
[-1.30016545e+00, -1.19789042e+00,  1.00000000e+00,
  0.00000000e+00],
[ 2.31049380e-01, -1.04329275e+00,  1.00000000e+00,
  0.00000000e+00],
[-1.33525579e-01, -1.58438461e+00,  0.00000000e+00,
  1.00000000e+00],
[ 8.87284306e-01, -6.56798560e-01,  1.00000000e+00,
  0.00000000e+00],
[-6.43930521e-01,  2.93759738e+00,  0.00000000e+00,
  1.00000000e+00],
[ 3.03964372e-01,  3.86735746e-01,  0.00000000e+00,
  1.00000000e+00],
[ 1.90809419e+00,  7.75403964e-02,  0.00000000e+00,
  1.00000000e+00],
[-1.37308044e+00, -1.54573519e+00,  0.00000000e+00,
  1.00000000e+00],
[ 1.39768925e+00, -1.58438461e+00,  1.00000000e+00,
  0.00000000e+00],
[ 1.10602928e+00, -1.27518926e+00,  1.00000000e+00,
  0.00000000e+00],
[-4.98100538e-01,  2.51245378e+00,  0.00000000e+00,
  1.00000000e+00],
[-1.44599543e+00,  7.73229934e-01,  0.00000000e+00,
  1.00000000e+00],
[ 1.32477426e+00,  5.41333421e-01,  1.00000000e+00,
  0.00000000e+00],
[-4.98100538e-01,  1.39162063e+00,  1.00000000e+00,
  0.00000000e+00],
[ 1.83517920e+00, -1.62303403e+00,  0.00000000e+00,
  1.00000000e+00],
[ 8.14369314e-01, -8.11396235e-01,  1.00000000e+00,
  0.00000000e+00],
[ 8.87284306e-01,  2.32138071e-01,  1.00000000e+00,
  0.00000000e+00],
[ 2.19975416e+00, -6.56798560e-01,  0.00000000e+00,
  1.00000000e+00],
[ 1.47060424e+00, -6.95447979e-01,  0.00000000e+00,
  1.00000000e+00],
[-7.89760505e-01,  5.79982840e-01,  1.00000000e+00,
  0.00000000e+00],
[-1.51891042e+00, -5.02200885e-01,  0.00000000e+00,
  1.00000000e+00],
[-1.22725046e+00, -1.58438461e+00,  0.00000000e+00,
  1.00000000e+00],
[-8.62675496e-01,  2.32138071e-01,  0.00000000e+00,
  1.00000000e+00],
[-6.06105869e-02,  2.32138071e-01,  1.00000000e+00,
  0.00000000e+00],
[-1.44599543e+00, -5.02200885e-01,  0.00000000e+00,
  1.00000000e+00],
[-5.71015529e-01,  7.73229934e-01,  1.00000000e+00,
  0.00000000e+00],
```

```
       [-2.06440570e-01,  1.62351715e+00,  1.00000000e+00,
         0.00000000e+00],
       [-1.08142047e+00, -3.84078599e-02,  0.00000000e+00,
         1.00000000e+00],
       [ 1.03311429e+00, -1.08194217e+00,  0.00000000e+00,
         1.00000000e+00],
       [-1.15433546e+00,  3.48086328e-01,  1.00000000e+00,
         0.00000000e+00],
       [ 1.23044049e-02,  3.86735746e-01,  0.00000000e+00,
         1.00000000e+00],
       [-3.52270554e-01,  6.57281678e-01,  0.00000000e+00,
         1.00000000e+00]])
```

Lets create model

Linear Regression

In [37]:
```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
```

In [38]:
```
model1=LinearRegression()
```

In [40]:
```
model1.fit(X_train_transformed, y_train)
y_pred = model1.predict(X_test_transformed)
mse = mean_squared_error(y_test, y_pred)
rmse = mean_squared_error(y_test, y_pred, squared=False)
r2 = r2_score(y_test, y_pred)
```

In [41]:
```
print(mse)
print(rmse)
print(r2)
```

```
555.6660712716779
23.572570315340624
0.07214919444219348
```

Decision Tree Regression

In [43]:
```
from sklearn.tree import DecisionTreeRegressor
model2=DecisionTreeRegressor(random_state=42)
```

In [44]:
```
model2.fit(X_train_transformed, y_train)
y_pred = model2.predict(X_test_transformed)
mse = mean_squared_error(y_test, y_pred)
rmse = mean_squared_error(y_test, y_pred, squared=False)
r2 = r2_score(y_test, y_pred)
```

In [45]:
```
print(mse)
print(rmse)
print(r2)
```

```
922.05
30.36527622136838
-0.5396384258384741
```

Random Forest Regressor

In [46]:
```
from sklearn.ensemble import RandomForestRegressor
model3=RandomForestRegressor(random_state=42)
```

In [47]:
```
model3.fit(X_train_transformed, y_train)
y_pred = model3.predict(X_test_transformed)
mse = mean_squared_error(y_test, y_pred)
rmse = mean_squared_error(y_test, y_pred, squared=False)
r2 = r2_score(y_test, y_pred)
```

In [48]:
```
print(mse)
print(rmse)
print(r2)
```

```
489.0080875
22.11352725143594
0.1834546477297514
```

From here we can judge that Linear regression model looks good among other 2 model with highest rmse and lowest r2.