```
#Jayesh Mali T084
import pandas as pd
import numpy as np

path='FB_data.csv'
df = pd.read_csv(path)
df.head()
```

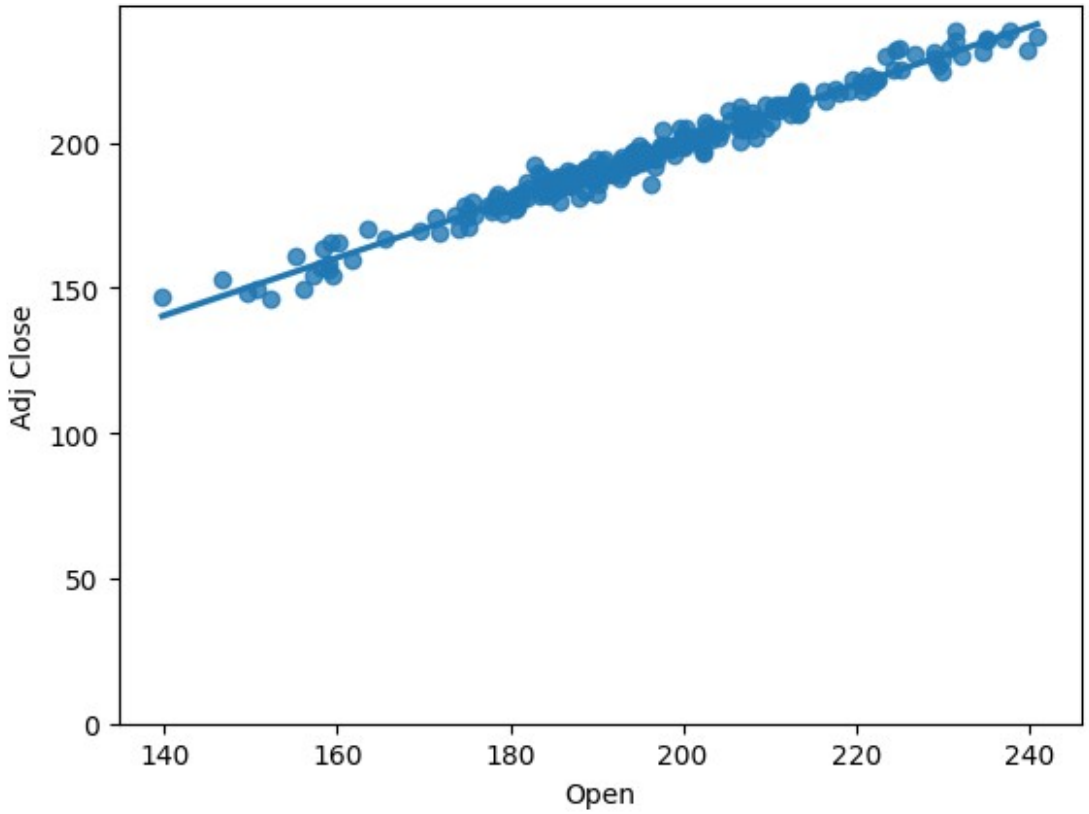
```
        Date        Open        High         Low       Close   Adj
Close
0  6/20/2019  190.949997  191.160004  187.639999  189.529999
189.529999  \
1  6/21/2019  188.750000  192.000000  188.750000  191.139999
191.139999
2  6/24/2019  192.419998  193.979996  191.570007  192.600006
192.600006
3  6/25/2019  192.880005  193.139999  188.130005  188.839996
188.839996
4  6/26/2019  189.539993  190.759995  187.309998  187.660004
187.660004

     Volume
0  14635700
1  22751200
2  15509000
3  16750300
4  12808600
```

```
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

# list the data types for each column
print(df.dtypes)
```

```
Date          object
Open         float64
High         float64
Low          float64
Close        float64
Adj Close    float64
Volume         int64
dtype: object
```

```
# Engine size as potential predictor variable of price
#sns.regplot(x="GenreLabels", y="Active", data=df)
#plt.ylim(0,)
```

```
# Convert data types if needed
df['Open'] = pd.to_numeric(df['Open'], errors='coerce')
df['Adj Close'] = pd.to_numeric(df['Adj Close'], errors='coerce')

# Plot regression plot
sns.regplot(x="Open", y="Adj Close", data=df)
plt.ylim(0,)  # Optional: Set y-axis limit if needed
plt.show()
```

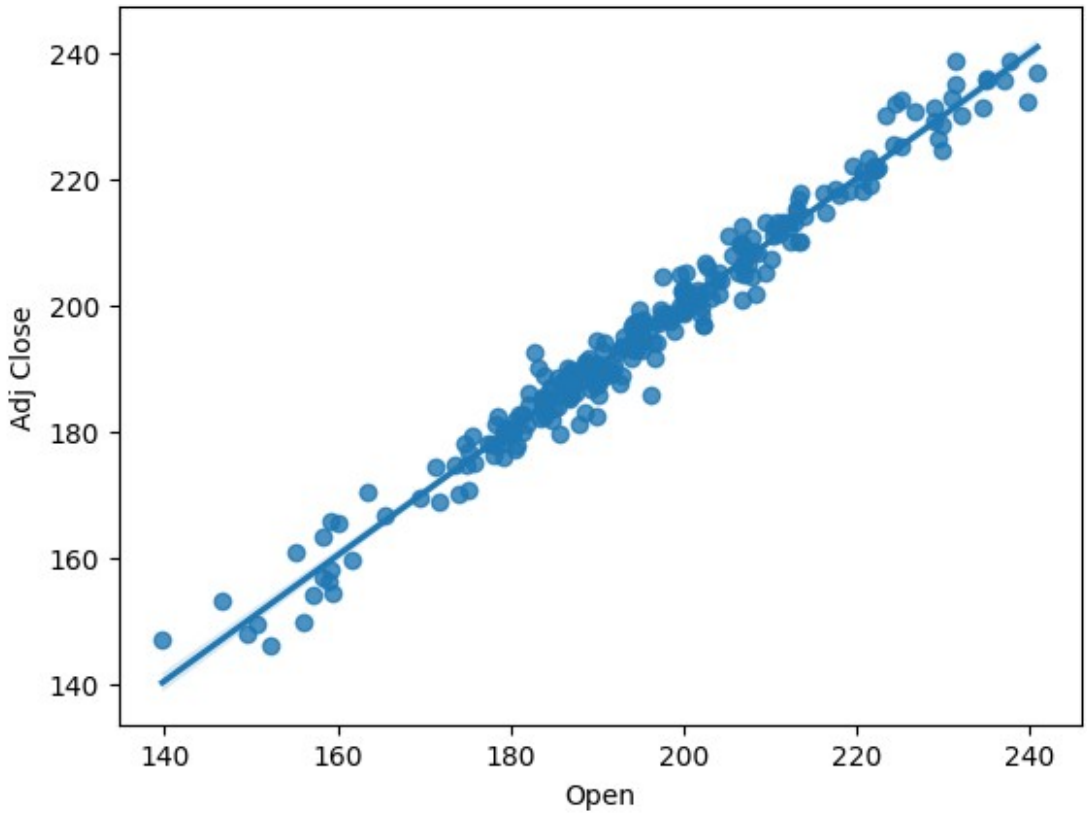



```
df[["Open", "Adj Close"]].corr()
```

```
              Open  Adj Close
Open       1.00000    0.98673
Adj Close  0.98673    1.00000
```

```
sns.regplot(x="Open", y="Adj Close", data=df)
```

```
<Axes: xlabel='Open', ylabel='Adj Close'>
```




```
df[['High', 'Low']].corr()
```

```
          High       Low
High  1.000000  0.990722
Low   0.990722  1.000000
```

```
df[['Volume','Low']].corr()
```

```
          Volume       Low
Volume  1.000000 -0.159472
Low    -0.159472  1.000000
```

```
from scipy import stats

pearson_coef, p_value = stats.pearsonr(df['Volume'], df['Low'])
print("The Pearson Correlation Coefficient is", pearson_coef, " with a
P-value of P =", p_value)
```

```
The Pearson Correlation Coefficient is -0.1594724293825312  with a P-
value of P = 0.01107634800405837
```

```
pearson_coef, p_value = stats.pearsonr(df['High'], df['Low'])
print("The Pearson Correlation Coefficient is", pearson_coef, " with a
P-value of P = ", p_value)
```

```
The Pearson Correlation Coefficient is 0.9907220576545691  with a P-value of P =  1.4018608456441653e-219
```

```
pearson_coef, p_value = stats.pearsonr(df['Open'], df['Low'])
print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P = ", p_value)
```

```
The Pearson Correlation Coefficient is 0.9935652556340527  with a P-value of P =  1.8999934722598045e-239
```

```
pearson_coef, p_value = stats.pearsonr(df['Adj Close'], df['Low'])
print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value )
```

```
The Pearson Correlation Coefficient is 0.9934323971490049  with a P-value of P = 2.4494191059171692e-238
```

```
pearson_coef, p_value = stats.pearsonr(df['Low'], df['Adj Close'])
print( "The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P = ", p_value)
```

```
The Pearson Correlation Coefficient is 0.9934323971490049  with a P-value of P =  2.4494191059171692e-238
```

```
pearson_coef, p_value = stats.pearsonr(df['engine-size'], df['price'])
print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)
```

```
The Pearson Correlation Coefficient is 0.8723351674455182  with a P-value of P = 9.265491622200262e-64
```

```
pearson_coef, p_value = stats.pearsonr(df['bore'], df['price'])
print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =  ", p_value )
```

```
The Pearson Correlation Coefficient is 0.5431553832626603  with a P-value of P =   8.04918948393533e-17
```

```
pearson_coef, p_value = stats.pearsonr(df['Low'], df['Volume'])
print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P = ", p_value)
```

```
The Pearson Correlation Coefficient is -0.1594724293825312  with a P-value of P =  0.01107634800405837
```

```
pearson_coef, p_value = stats.pearsonr(df['Volume'], df['High'])
print( "The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P = ", p_value )
```

```
The Pearson Correlation Coefficient is -0.05651018242975007  with a P-value of P =  0.3707262014073475
```

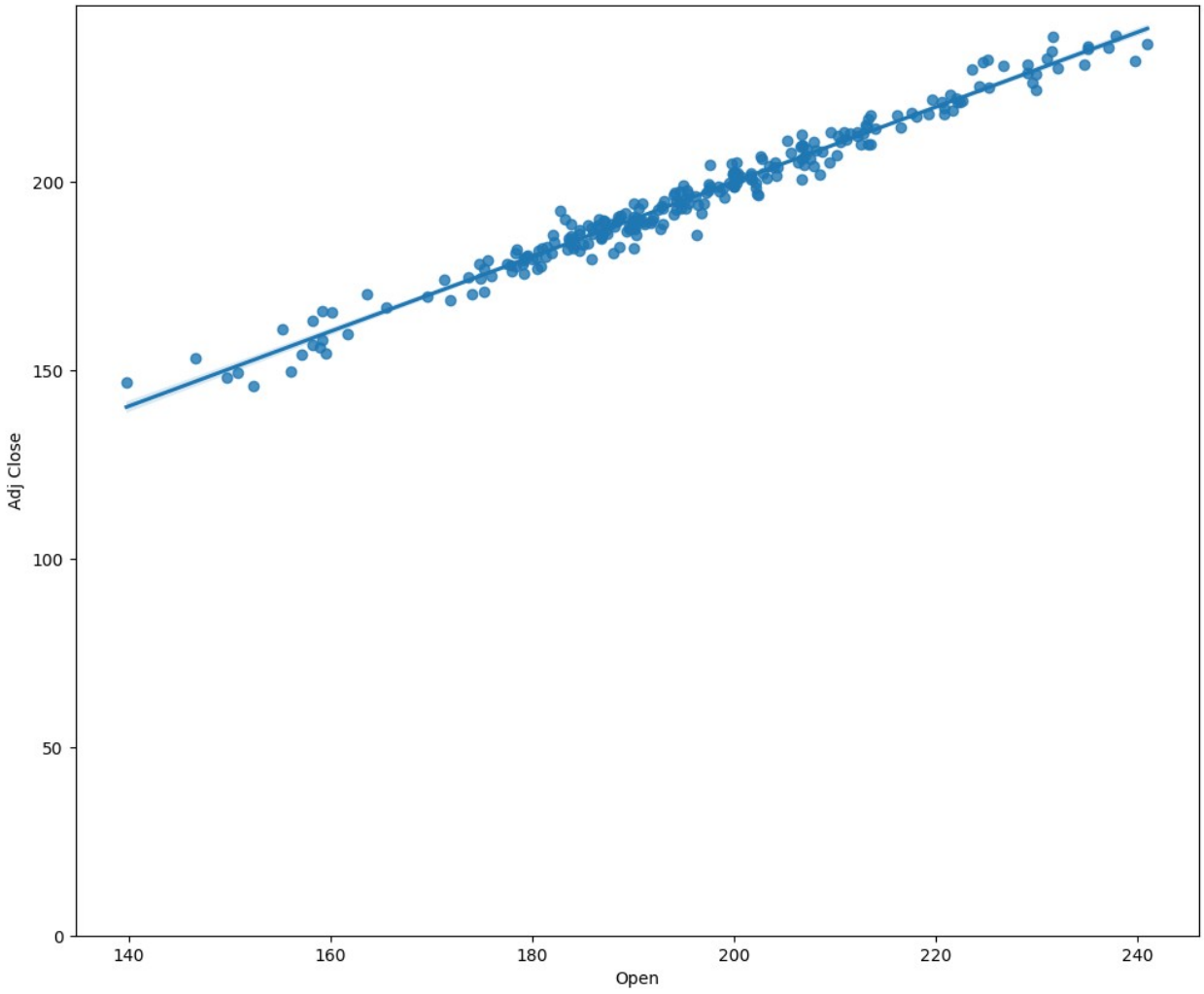
```
from sklearn.linear_model import LinearRegression
```

```
lm = LinearRegression()
lm
```

```
LinearRegression()
```

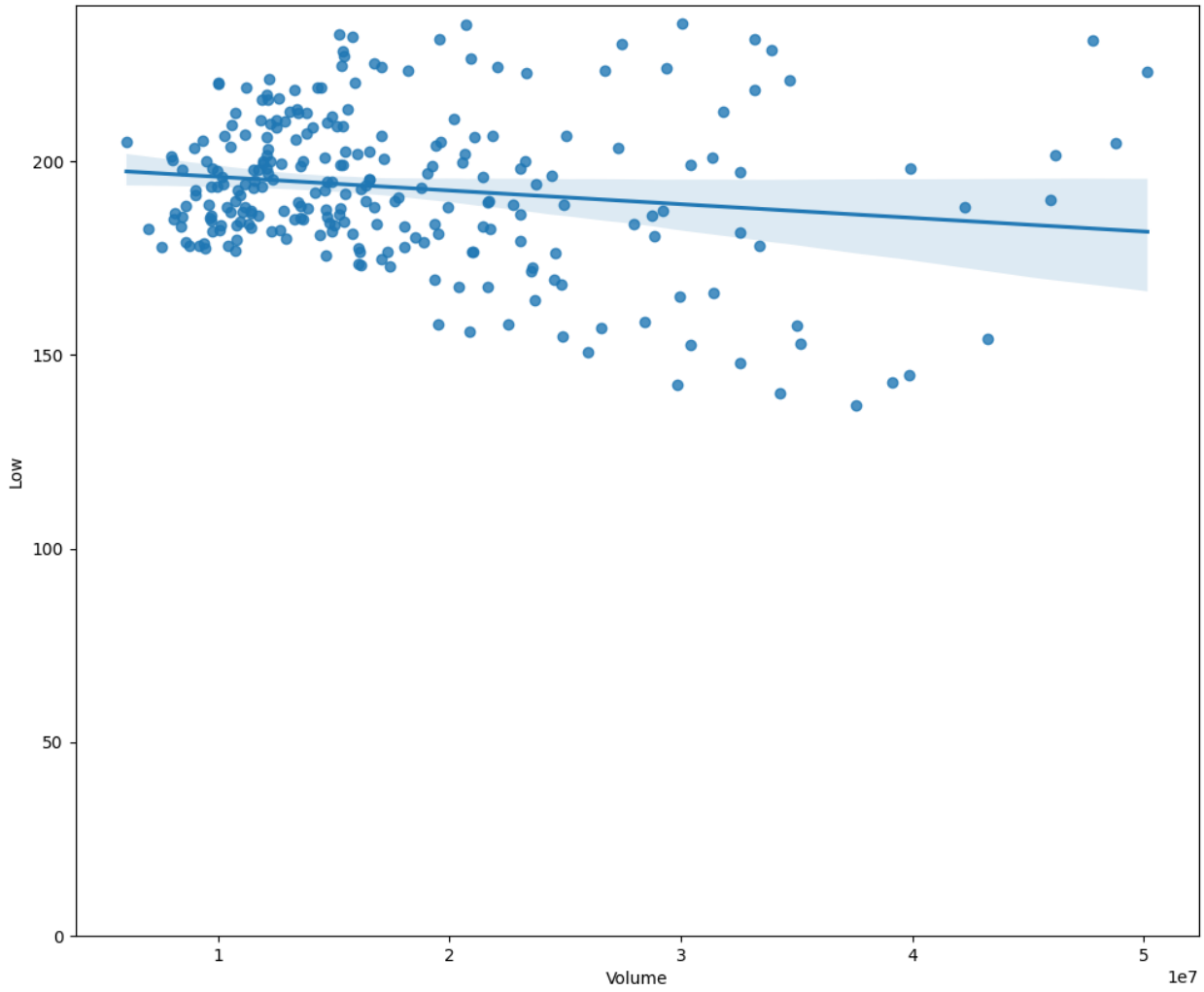
```
X = df[['Open']]
Y = df['Adj Close']

lm.fit(X,Y)

Yhat=lm.predict(X)
Yhat[0:5]

lm.intercept_

lm.coef_

Z = df[['High', 'Low', 'Volume', 'Low']]

# import the visualization package: seaborn
import seaborn as sns
%matplotlib inline

width = 12
height = 10
plt.figure(figsize=(width, height))
sns.regplot(x="Open", y="Adj Close", data=df)
plt.ylim(0,)
```

```
(0.0, 247.05341936415093)
```




```
plt.figure(figsize=(width, height))
sns.regplot(x="Volume", y="Low", data=df)
plt.ylim(0,)
```

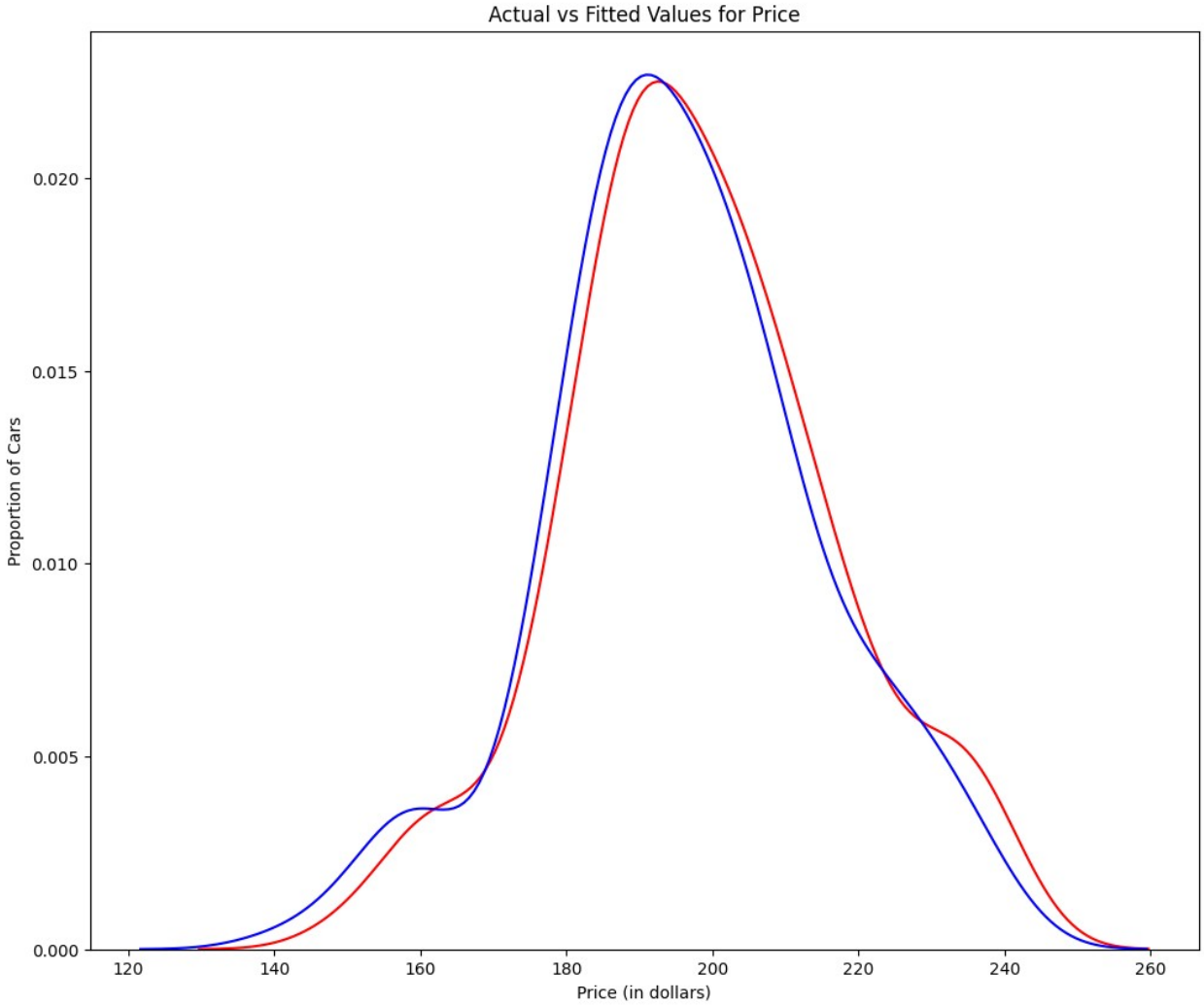
```
(0.0, 240.47250285)
```




```
plt.figure(figsize=(width, height))


ax1 = sns.distplot(df['High'], hist=False, color="r", label="Actual
Value")
sns.distplot(Yhat, hist=False, color="b", label="Fitted Values" ,
ax=ax1)


plt.title('Actual vs Fitted Values for Price')
plt.xlabel('Price (in dollars)')
plt.ylabel('Proportion of Cars')

plt.show()
plt.close()
```

```
C:\Users\Admin\AppData\Local\Temp\ipykernel_16148\2108512286.py:4:
UserWarning:
```

```
`distplot` is a deprecated function and will be removed in seaborn
v0.14.0.

Please adapt your code to use either `displot` (a figure-level
function with
similar flexibility) or `kdeplot` (an axes-level function for kernel
density plots).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  ax1 = sns.distplot(df['High'], hist=False, color="r", label="Actual
Value")
C:\Users\Admin\AppData\Local\Temp\ipykernel_16148\2108512286.py:5:
UserWarning:

`distplot` is a deprecated function and will be removed in seaborn
v0.14.0.

Please adapt your code to use either `displot` (a figure-level
function with
similar flexibility) or `kdeplot` (an axes-level function for kernel
density plots).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(Yhat, hist=False, color="b", label="Fitted Values" ,
ax=ax1)
```


Actual vs Fitted Values for Price
0.020
0.015
0.010
0.005
0.000
Proportion of Cars
120
140
160
180
200
220
240
260
Price (in dollars)