5505: Assignment – 1
Linear Regression

Shiva Chakravarthy Gollapudi
Student ID: 11468647

Using the data provided created three Linear Regression models and evaluated the output.

Data: Monet.csv.

Data Description: Price (Target variable), Height, Width, Signed, Picture and House.

Tools and Environment: Using python programming and Google Colab, I have explored the data and created Linear Regression models.

Libraries used:

```
[50] #instal libraries
     import numpy as np
     import pandas as pd
     import matplotlib.pyplot as plt
     import seaborn as sns
     import math
```

Finding Missing Values: Explore data to find the missing values.

```
#Checking for Null values
data.isnull().sum()

PRICE      0
HEIGHT     0
WIDTH      0
SIGNED     0
PICTURE    0
HOUSE      0
dtype: int64
```

So, there is no null values in the input data.

From the above code, we can conclude that there are no missing values in the input data.

Transforming the data: Created a new variable using two independent variable Height and Weight.

SIZE = HEIGHT * WEIGHT

Code:

```
#Transformation of the data
#Size = Width * Height
data["SIZE"] = data["HEIGHT"] * data["WIDTH"]
data.head()
```

	PRICE	HEIGHT	WIDTH	SIGNED	PICTURE	HOUSE	SIZE
0	3.993780	21.3	25.6	1	1	1	545.28
1	8.800000	31.9	25.6	1	2	2	816.64
2	0.131694	6.9	15.9	0	3	3	109.71
3	2.037500	25.7	32.0	1	4	2	822.40
4	1.487500	25.7	32.0	1	4	2	822.40

Created a new variable SIZE and added that column to the data.

Finding Correlation between the data:

Correlation, which assesses the strength of the relationship between two variables. Coefficient correlation range:

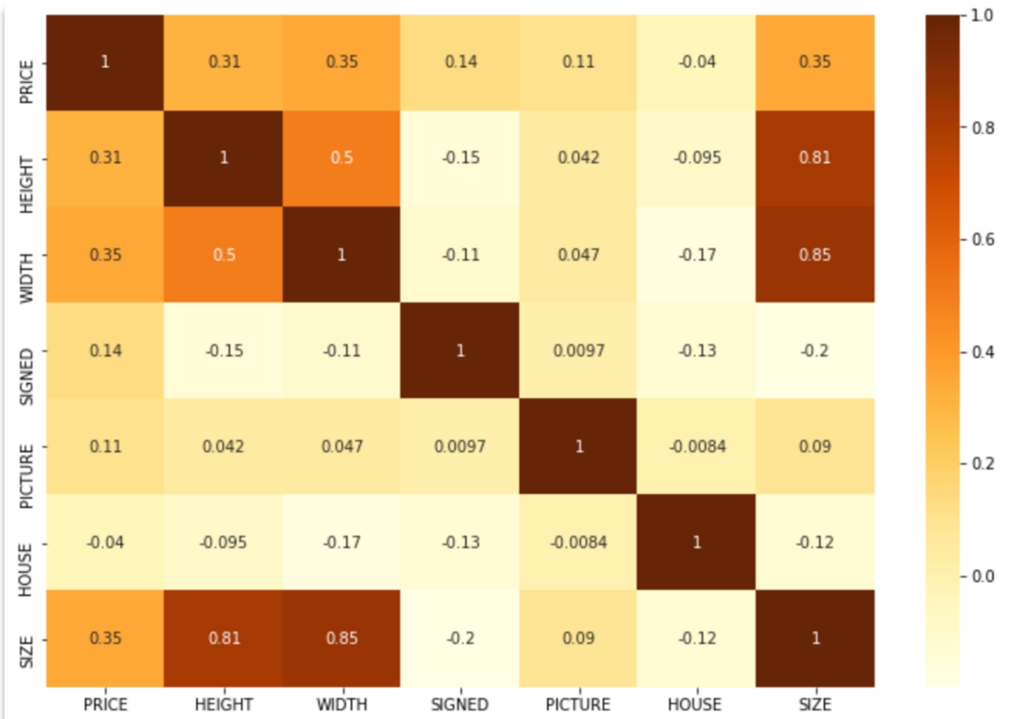
- If the coefficient of the correlation is greater than zero, then there is a positive relation among the variables.
- If the coefficient of the correlation is less than zero, then there is a negative relation among the variables.
- If the coefficient of the correlation equal to zero, then there is no relation among the variables.

Using Heatmap with Seaborn library, we have identified the correlation between the variables.

Code:

```
# finding corelation between the data
plt.subplots(figsize = (12,8))
sns.heatmap(data.corr(), cmap='YlOrBr', annot=True)
plt.show()
```

Output:



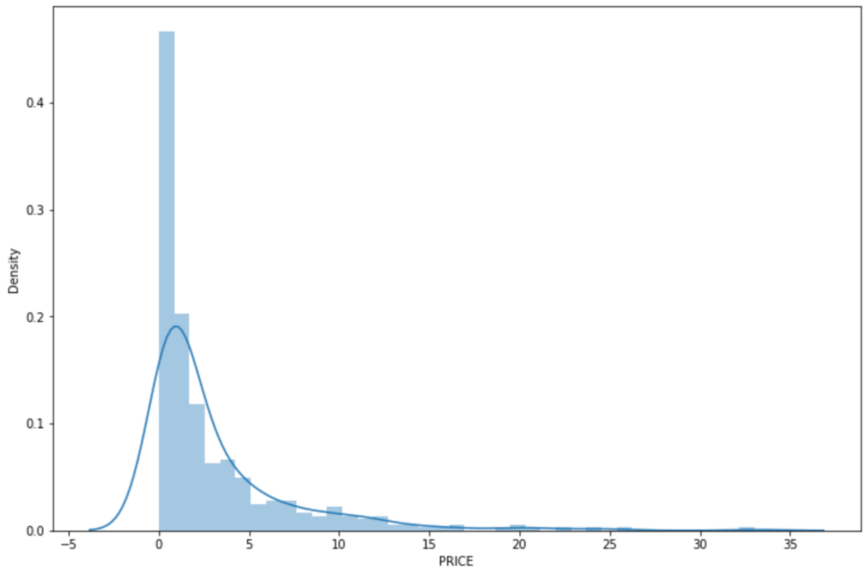
Analysis: From the above map, we conclude that PRICE is correlated to width and size and also, we know that height and width are corelated to size.

Checking the distribution of dependent variable (PRICE):

Code:

```
#checking the distribution of dependent variable(PRICE)
plt.subplots(figsize = (12,8))
sns.distplot(data["PRICE"])
plt.show();
```

Output & Analysis: From the below figure, we can say that PRICE is not normally distributed and there are few outliers (Right skewed).



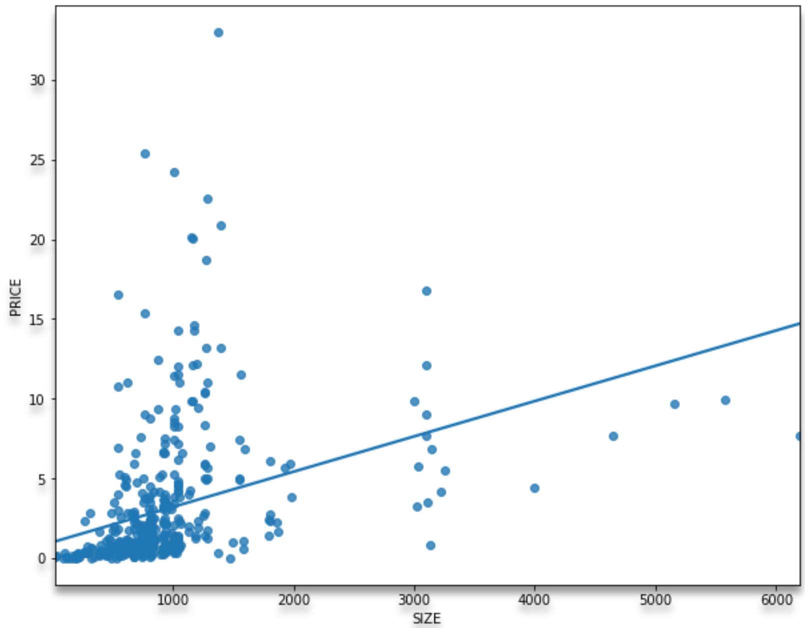
Model 1: Creating a model using Size as the independent variable and Price as dependent variable.

Using Size as the independent variable and Price as dependent variable created a scatter plot for showing the relationship between the independent variable and the dependent variable for this model, and also shown the linear regression line in the same plot.

Code:

```
plt.subplots(figsize = (10,8))
sns.regplot(x = data.SIZE, y = data.PRICE, ci=None);
plt.show();
```

Output:



Creating Linear Regression:

Split the data into training and test partitions. Considering training data size as 80% of the original data and test data size as 20%.

Code:

```python
# Split the data into test and train.
# Considering training/test as 80/20.
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(data[['SIZE']], data['PRICE'], train_size = 0.8)

X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

```
((344, 1), (86, 1), (344,), (86,))
```

Created a Linear regression with SIZE as the independent variable and PRICE as dependent variable.

- Created a linear regression model and fit the model into the training data.
- Predicting the y_value (PRICE) using the x (SIZE).
- Calculated the errors for this model:

    - R^2
    - Mean Square Error (MSE)
    - Root Mean Square Error (RMSE).

Code:

```
[63] # builing Linear Regression

     from sklearn.linear_model import LinearRegression
     lr = LinearRegression()
     model1  = lr.fit (X_train, y_train) # Fit the model into the training data
```

```
[64] y_test_pred = model1.predict (X_test) # Predicting the y value using x for test data.
```

```
# Calculate the error of the prediction with test data.
# Finding mean square error, R2_Score and Root mean square error.

from sklearn.metrics import mean_squared_error, r2_score
mse = mean_squared_error (y_test, y_test_pred)
print('MSE for the test set: {:.2f}'.format(mse))

r2 = r2_score (y_test, y_test_pred)
print('R2_Score for the test set: {:.2f}'.format(r2))

import math
RMSE = math.sqrt(mse)
print('RMSE for the test set: {:.2f}'.format(RMSE))
```

Output:

```
MSE for the test set: 18.45
R2_Score for the test set: 0.04
RMSE for the test set: 4.30
```

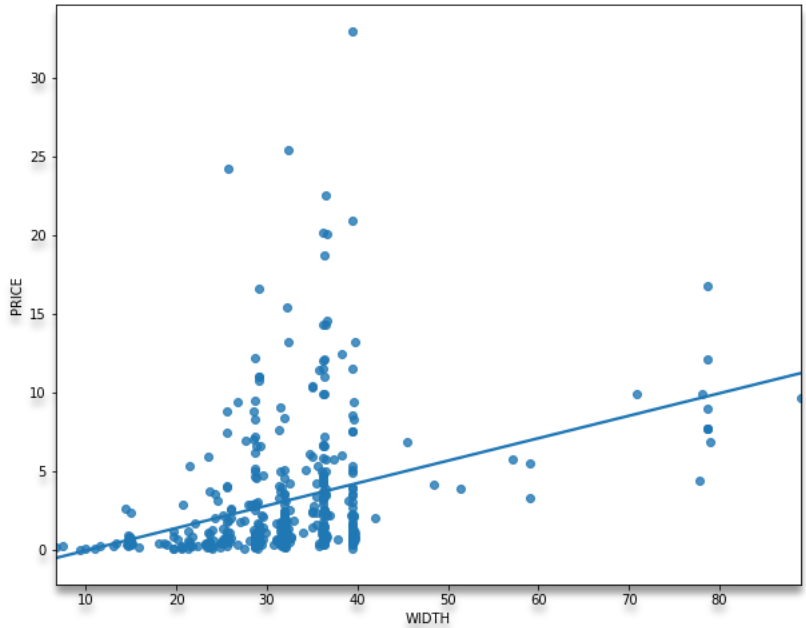
Model 2: Creating a model using Width as the independent variable and Price as dependent variable.

Using Size as the independent variable and Width as dependent variable created a scatter plot for showing the relationship between the independent variable and the dependent variable for this model, and also shown the linear regression line in the same plot.

Code:

```
plt.subplots(figsize = (10,8))
sns.regplot(x = data.WIDTH, y = data.PRICE, ci=None);
plt.show();
```

Output:



Created a Linear regression with WIDTH as the independent variable and PRICE as dependent variable.

- Created a linear regression model and fit the model into the training data.
- Predicting the y_value (PRICE) using the x (WIDTH).
- Calculated the errors for this model:
    - R^2
    - Mean Square Error (MSE)
    - Root Mean Square Error (RMSE).

Code:

```
[69] y_test_pred = model1.predict (X_test) # Predicting the y value using x for test data.
```

```
# Calculate the error of the prediction with test data.
# Finding mean square error, R2_Score and Root mean square error.

from sklearn.metrics import mean_squared_error, r2_score
mse = mean_squared_error (y_test, y_test_pred)
print('MSE for the test set: {:.2f}'.format(mse))

r2 = r2_score (y_test, y_test_pred)
print('R2_Score for the test set: {:.2f}'.format(r2))

import math
RMSE = math.sqrt(mse)
print('RMSE for the test set: {:.2f}'.format(RMSE))
```

Output:

```
MSE for the test set: 15.69
R2_Score for the test set: 0.09
RMSE for the test set: 3.96
```

Model 3: Multivariate Linear Regression model

Multi Linear Regression uses a linear function to predict the value of a dependent variable containing the function more than one independent variables.

- Created a Multi Linear Regression with SIGNED, PICTURE, SIZE, HOUSE_1, HOUSE_2 AND HOUSE_3 as the independent variable and PRICE as dependent variable.
- Selected only SIZE, where HEIGHT and WIDTH are correlated to size.
- Only HOUSE variable has three classes. So, created the dummy columns of HOUSE variable and added them to the original data.
- Created a linear regression model and fit the model into the training data.
- Predicting the y_value (WIDTH) using the x (SIZE).
- Calculated the errors for this model:
    - R^2
    - Mean Square Error (MSE)
    - Root Mean Square Error (RMSE).

Code: To create the dummy columns by HOUSE variable.

```
[ ] # Create dummy columns of the HOUSE variable
    Houses_dummy = pd.get_dummies(data['HOUSE '], prefix='HOUSE')
    data=data.join(Houses_dummy) # Join the all join House colums
```

```
[91] data.head()
```

```
y = data["PRICE"]
x = data[["SIGNED","PICTURE","SIZE","HOUSE_1","HOUSE_2","HOUSE_3"]]
x.head()
```

	SIGNED	PICTURE	SIZE	HOUSE_1	HOUSE_2	HOUSE_3
0	1	1	545.28	1	0	0
1	1	2	816.64	0	1	0
2	0	3	109.71	0	0	1
3	1	4	822.40	0	1	0
4	1	4	822.40	0	1	0

Code:

```
[102] # Builing Multiple Linear Regression

      from sklearn.linear_model import LinearRegression
      mlr = LinearRegression()
      model3  = mlr.fit (X_train, y_train) # Fit the model into the training data

[103] y_test_pred = model1.predict (X_test) # Predicting the y value using x for test data.

      # Calculate the error of the prediction with test data.
      # Finding mean square error, R2_Score and Root mean square error.

      from sklearn.metrics import mean_squared_error, r2_score
      mse = mean_squared_error (y_test, y_test_pred)
      print('MSE for the test set: {:.2f}'.format(mse))

      r2 = r2_score (y_test, y_test_pred)
      print('R2_Score for the test set: {:.2f}'.format(r2))

      import math
      RMSE = math.sqrt(mse)
      print('RMSE for the test set: {:.2f}'.format(RMSE))
```

Output:

```
MSE for the test set: 11.61
R2_Score for the test set: 0.11
RMSE for the test set: 3.41
```

R^2: It estimates the strength of the relationship between the model and the response variable.

MSE: Mean Square Error, the average of the squared difference between the target value predicted by the regression model

RMSE: Root Mean Square Error, is the square root of the average of the squared difference between the target value predicted by the regression model.

Conclusion: After validating the errors of the three models, Model 3 (Multi Linear Regression) is the significant model with low MSE value.