**PRACTICAL 9**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Name:** | Harsh Shah | **Semester:** | VII | **Division:** | 6 |
| **Roll No.:** | 21BCP359 | **Date:** | 01-10-24 | **Batch:** | G11 |

**Code**

import pandas as pd

import statsmodels.api as sm

df = pd.read\_csv('CarPrice\_Assignment.csv')

X = df['horsepower']

y = df['citympg']

X = sm.add\_constant(X)

*# Create the regression model*

model\_citympg = sm.OLS(y, X).fit()

print(model\_citympg.summary())

X\_highway = df['horsepower']

y\_highway = df['highwaympg']

X\_highway = sm.add\_constant(X\_highway)

*# Create the regression model*

model\_highway = sm.OLS(y\_highway, X\_highway).fit()

print(model\_highway.summary())

import matplotlib.pyplot as plt

import seaborn as sns

*# Scatterplot for citympg vs. horsepower*

plt.figure(*figsize*=(8, 6))

sns.scatterplot(*x*=df['horsepower'], *y*=df['citympg'], *color*='blue')

plt.plot(df['horsepower'], model\_citympg.predict(sm.add\_constant(df['horsepower'])), *color*='red', *label*='Regression Line')

plt.title('Scatterplot of City MPG vs Horsepower')

plt.xlabel('Horsepower')

plt.ylabel('City MPG')

plt.legend()

plt.show()

*# Scatterplot for highwaympg vs. horsepower*

plt.figure(*figsize*=(8, 6))

sns.scatterplot(*x*=df['horsepower'], *y*=df['highwaympg'], *color*='green')

plt.plot(df['horsepower'], model\_highway.predict(sm.add\_constant(df['horsepower'])), *color*='red', *label*='Regression Line')

plt.title('Scatterplot of Highway MPG vs Horsepower')

plt.xlabel('Horsepower')

plt.ylabel('Highway MPG')

plt.legend()

plt.show()

*# Regression Model 1: price as dependent and citympg as independent variable*

X\_citympg = sm.add\_constant(df['citympg'])

model\_price\_citympg = sm.OLS(df['price'], X\_citympg).fit()

*# Display model statistics for price vs. citympg*

print("Model 1: price vs citympg")

print(model\_price\_citympg.summary())

*# Regression Model 2: price as dependent and highwaympg as independent variable*

X\_highwaympg = sm.add\_constant(df['highwaympg'])  *# Add constant term*

model\_price\_highwaympg = sm.OLS(df['price'], X\_highwaympg).fit()

*# Display model statistics for price vs. highwaympg*

print("\nModel 2: price vs highwaympg")

print(model\_price\_highwaympg.summary())

*# Model 1: Regression of 'price' on 'enginesize'*

X\_enginesize = df[['enginesize']]

y\_price = df['price']

*# Add a constant (intercept)*

X\_enginesize = sm.add\_constant(X\_enginesize)

*# Create and fit the regression model*

model\_enginesize = sm.OLS(y\_price, X\_enginesize).fit()

*# Output the model summary*

print(model\_enginesize.summary())

*# Model 2: Regression of 'price' on 'curbweight'*

X\_curbweight = df[['curbweight']]  *# Independent variable*

y\_price = df['price']  *# Dependent variable*

*# Add a constant (intercept)*

X\_curbweight = sm.add\_constant(X\_curbweight)

*# Create and fit the regression model*

model\_curbweight = sm.OLS(y\_price, X\_curbweight).fit()

*# Output the model summary*

print(model\_curbweight.summary())

*# Scatterplot for Engine Size vs. Price*

plt.figure(*figsize*=(14, 6))

plt.subplot(1, 2, 1)

sns.scatterplot(*x*=df['enginesize'], *y*=df['price'], *color*='blue')

plt.plot(df['enginesize'], model\_enginesize.predict(sm.add\_constant(df['enginesize'])), *color*='red', *label*='Regression Line')

plt.title('Scatterplot of Engine Size vs Price')

plt.xlabel('Engine Size (L)')

plt.ylabel('Price ($)')

plt.legend()

*# Scatterplot for Curb Weight vs. Price*

plt.subplot(1, 2, 2)

sns.scatterplot(*x*=df['curbweight'], *y*=df['price'], *color*='green')

plt.plot(df['curbweight'], model\_curbweight.predict(sm.add\_constant(df['curbweight'])), *color*='orange', *label*='Regression Line')

plt.title('Scatterplot of Curb Weight vs Price')

plt.xlabel('Curb Weight (lbs)')

plt.ylabel('Price ($)')

plt.legend()

*# Show the plots*

plt.tight\_layout()

plt.show()

import pandas as pd

import statsmodels.api as sm

from statsmodels.stats.outliers\_influence import variance\_inflation\_factor

df = pd.read\_csv('CarPrice\_Assignment.csv')

*# Select numeric variables except 'citympg' and 'highwaympg'*

numeric\_df = df.drop(['price', 'citympg', 'highwaympg'], *axis*=1).select\_dtypes(*include*=[*float*, *int*])

independent\_vars = sm.add\_constant(numeric\_df)

*# Variance Inflation Factor (VIF) calculation*

vif\_data = pd.DataFrame()

vif\_data['Feature'] = independent\_vars.columns

vif\_data['VIF'] = [variance\_inflation\_factor(independent\_vars.values, i) for i in range(independent\_vars.shape[1])]

print(vif\_data)

**Output**

 

