Machine Learning  
Assignment 07

Name: Jasmine Imtiyaj Sayyad

PRN: 22510033

Batch: T-2

You have been provided data in the CSV file. Use this file to fit a linear regression model . ‘y’ is the target variable and ‘x1,x2,…..x11’ are the input variables.

Steps:

1. **Remove the outliers using IQR method**

Calculates Q1 (25th percentile) and Q3 (75th percentile**)**.

Identifies **outliers** using the IQR formula:

Lower Bound=Q1−1.5×IQR

Upper Bound=Q3+1.5×IQR

Removes data points beyond these bounds.

Outliers are removed to prevent them from distorting the model.

1. **VIF Calculation and Multicollinearity Handling**

Computes **VIF** (Variance Inflation Factor) for all features.

Iteratively removes features with **VIF > 4**.

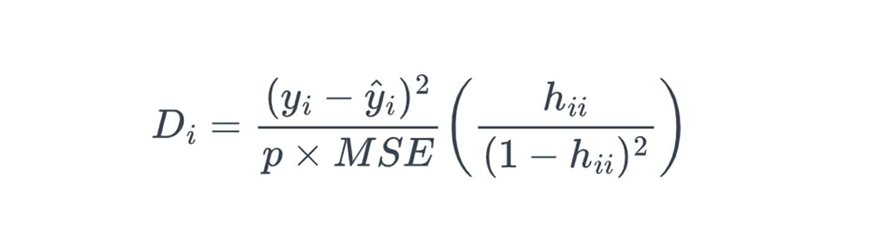
Multicollinearity can be reduced by dropping correlated features.

1. **Influence Detection and Removal**

Calculates Cook’s Distance and DFFITS to detect influential points.

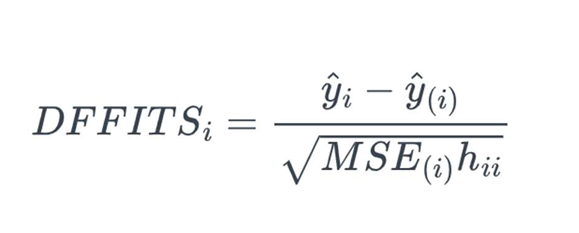
Removes high-influence points using:

**Cook’s Distance**



Threshold Cook’s Distance=2.8/n

**DFFITS**



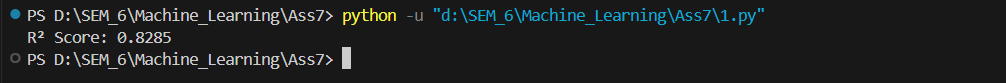
High-influence data points removed to prevent model distortion.

1. Train-Test Split and Model Training:

Splits the data into **training (80%)** and **testing (20%)**.

1. Final Output:

Coefficient of Determination is :



Code:

import pandas as pd

import numpy as np

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import r2\_score

import statsmodels.api as sm

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

file\_path = "linear\_regression\_3.csv"

df = pd.read\_csv(file\_path)

#Remove Outliers using IQR method

Q1 = df.quantile(0.25)

Q3 = df.quantile(0.75)

IQR = Q3 - Q1

lower\_bound = Q1 - 1.5 \* IQR

upper\_bound = Q3 + 1.5 \* IQR

df\_iqr\_clean = df[~((df < lower\_bound) | (df > upper\_bound)).all(axis=1)]

X\_iqr = df\_iqr\_clean.drop(columns=['y'])

y\_iqr = df\_iqr\_clean['y']

#Remove features with high VIF (VIF > 4)

def calculate\_vif(X):

    vif\_data = pd.DataFrame()

    vif\_data["Feature"] = X.columns

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

    return vif\_data

def remove\_high\_vif(X, vif\_threshold=4.0):

    vif\_data = calculate\_vif(X)

    while vif\_data["VIF"].max() > vif\_threshold:

        high\_vif\_feature = vif\_data.sort\_values(by="VIF", ascending=False).iloc[0]["Feature"]

        X = X.drop(columns=[high\_vif\_feature])

        vif\_data = calculate\_vif(X)

    return X

X\_vif\_clean = remove\_high\_vif(X\_iqr, vif\_threshold=4.0)

X\_vif\_const = sm.add\_constant(X\_vif\_clean)

#Detect Influence Points using Cook's Distance and DFFITS

model = sm.OLS(y\_iqr, X\_vif\_const).fit()

influence = model.get\_influence()

cooks\_d, \_ = influence.cooks\_distance

threshold\_cook = 2.8 / len(X\_vif\_clean)

defits\_values, \_ = influence.dffits

threshold\_dffits = 1.1 \* np.sqrt(X\_vif\_clean.shape[1] / len(X\_vif\_clean))

# Detect high influence points

high\_influence\_points = (cooks\_d > threshold\_cook) | (np.abs(defits\_values) > threshold\_dffits)

df\_final\_clean = df\_iqr\_clean[~high\_influence\_points]

# Train-Test Split

X\_final = df\_final\_clean.drop(columns=['y'])

y\_final = df\_final\_clean['y']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_final, y\_final, test\_size=0.2, random\_state=42)

linear\_model = LinearRegression()

linear\_model.fit(X\_train, y\_train)

#Evaluation

y\_pred = linear\_model.predict(X\_test)

r2\_final = r2\_score(y\_test, y\_pred)

print(f"R² Score: {r2\_final:.4f}")

Output:

