**Regression & Its Evaluation — Assignment (DA-AG-010)**

Submitted by: Arun Kumar Rai

Subject: Machine Learning

**Question 1: What is Simple Linear Regression?**

Simple Linear Regression models the relationship between one independent variable and one dependent variable using a straight line.

**Question 2: What are the key assumptions of Simple Linear Regression?**

Linearity, independence, homoscedasticity, normality of residuals, and absence of multicollinearity.

**Question 3: What is heteroscedasticity, and why is it important?**

Heteroscedasticity means non-constant variance of residuals. It makes regression estimates unreliable.

**Question 4: What is Multiple Linear Regression?**

It models the relationship between multiple predictors and one target variable.

**Question 5: What is Polynomial Regression, and how does it differ?**

Polynomial regression fits curves, unlike linear regression which fits a straight line.

**Question 6: Simple Linear Regression with Plot**

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression

X = np.array([1, 2, 3, 4, 5]).reshape(-1, 1)
Y = np.array([2.1, 4.3, 6.1, 7.9, 10.2])

model = LinearRegression()
model.fit(X, Y)

Y_pred = model.predict(X)

print("Slope:", model.coef_[0])
print("Intercept:", model.intercept_)
print("Predicted Y:", Y_pred)

plt.scatter(X, Y)
plt.plot(X, Y_pred)
plt.show()
```

**Question 7: Multiple Regression + VIF**

```
import pandas as pd
import statsmodels.api as sm
from statsmodels.stats.outliers_influence import variance_inflation_factor

df = pd.DataFrame({
    "Area": [1200, 1500, 1800, 2000],
    "Rooms": [2, 3, 3, 4],
    "Price": [250000, 300000, 320000, 370000]
})

X = df[["Area", "Rooms"]]
Y = df["Price"]
```

```
X_const = sm.add_constant(X)
model = sm.OLS(Y, X_const).fit()
print(model.summary())

vif = pd.DataFrame()
vif["Feature"] = X.columns
vif["VIF"] = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]
print(vif)
```

**Question 8: Polynomial Regression (2nd Degree)**

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression

X = np.array([1, 2, 3, 4, 5]).reshape(-1,1)
Y = np.array([2.2, 4.8, 7.5, 11.2, 14.7])

poly = PolynomialFeatures(degree=2)
X_poly = poly.fit_transform(X)

model = LinearRegression()
model.fit(X_poly, Y)

Y_pred = model.predict(X_poly)

print("Coefficients:", model.coef_)
print("Intercept:", model.intercept_)

plt.scatter(X, Y)
plt.plot(X, Y_pred)
plt.show()
```

**Question 9: Residual Plot + Heteroscedasticity Check**

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression

X = np.array([10, 20, 30, 40, 50]).reshape(-1,1)
Y = np.array([15, 35, 40, 50, 65])

model = LinearRegression()
model.fit(X, Y)
Y_pred = model.predict(X)

residuals = Y - Y_pred
print("Residuals:", residuals)

plt.scatter(X, residuals)
plt.axhline(0, color='red')
plt.show()
```

**Question 10: Handling heteroscedasticity & multicollinearity**

Fix heteroscedasticity using log transforms, weighted regression, or robust errors. Fix multicollinearity using VIF, removing correlated features, PCA, or Ridge Regression.