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

***Simple Linear Regression:***

Simple Linear Regression is a statistical model that predicts a continuous output variable (target variable) based on a single input feature (predictor variable). It assumes a linear relationship between the input feature and the target variable.

**The equation for Simple Linear Regression is:**

`y = β0 + β1 * x + ε`

**where:**
- `y` is the target variable
- `x` is the predictor variable
- `β0` is the intercept (or bias)
- `β1` is the slope coefficient
- `ε` is the error term (residuals)

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

**The key assumptions of Simple Linear Regression are:**

1. Linearity: The relationship between the predictor variable and the target variable is linear.
2. Independence: Observations are independent of each other.
3. Homoscedasticity: The variance of residuals is constant across all levels of the predictor variable.
4. Normality: Residuals are normally distributed.
5. No or little multicollinearity: Not applicable in Simple Linear Regression, but important in Multiple Linear Regression.

**Question 3: What is heteroscedasticity, and why is it important to address in regression models?**
**Answer:**

**Heteroscedasticity:**

Heteroscedasticity refers to a situation in regression analysis where the variance of the residuals (errors) is not constant across all levels of the predictor variable(s). In other words, the spread of the residuals changes as the value of the predictor variable changes.

**Why is it important to address?**

1. Inefficient estimates: Heteroscedasticity can lead to inefficient estimates of model parameters, making it difficult to interpret the results.
2. Biased standard errors: It can cause biased estimates of standard errors, which can lead to incorrect conclusions about the significance of predictor variables.
3. Invalid hypothesis tests: Heteroscedasticity can invalidate hypothesis tests, such as t-tests and F-tests, which rely on constant variance assumptions.
4. Poor predictions: It can result in poor predictions, especially for extreme values of the predictor variable.

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

**Multiple Linear Regression (MLR):**

A statistical technique modeling the relationship between a dependent variable (Y) and multiple independent variables (X1, X2, ..., Xn).

$Y = \beta 0 + \beta 1X1 + \beta 2X2 + \ldots + \beta nXn + \varepsilon$

- Predicts continuous outcomes
- Controls for multiple predictors
- Assumes linearity, independence, homoscedasticity, normality, and no multicollinearity

**Question 5: What is polynomial regression, and how does it differ from linear regression?**
**Answer:**

**Polynomial Regression:**

Polynomial regression is a type of regression analysis where the relationship between the independent variable (X) and dependent variable (Y) is modeled as an nth degree polynomial.

**Equation:**

$Y = \beta 0 + \beta 1X + \beta 2X\text{^}2 + \ldots + \beta nX\text{^}n + \varepsilon$

**Key aspects:**

1. Non-linear relationship: Polynomial regression models non-linear relationships between X and Y.

2. Higher-degree terms: Includes higher-degree terms (e.g., X^2, X^3) to capture curvature in the data.

**Difference from Linear Regression:**

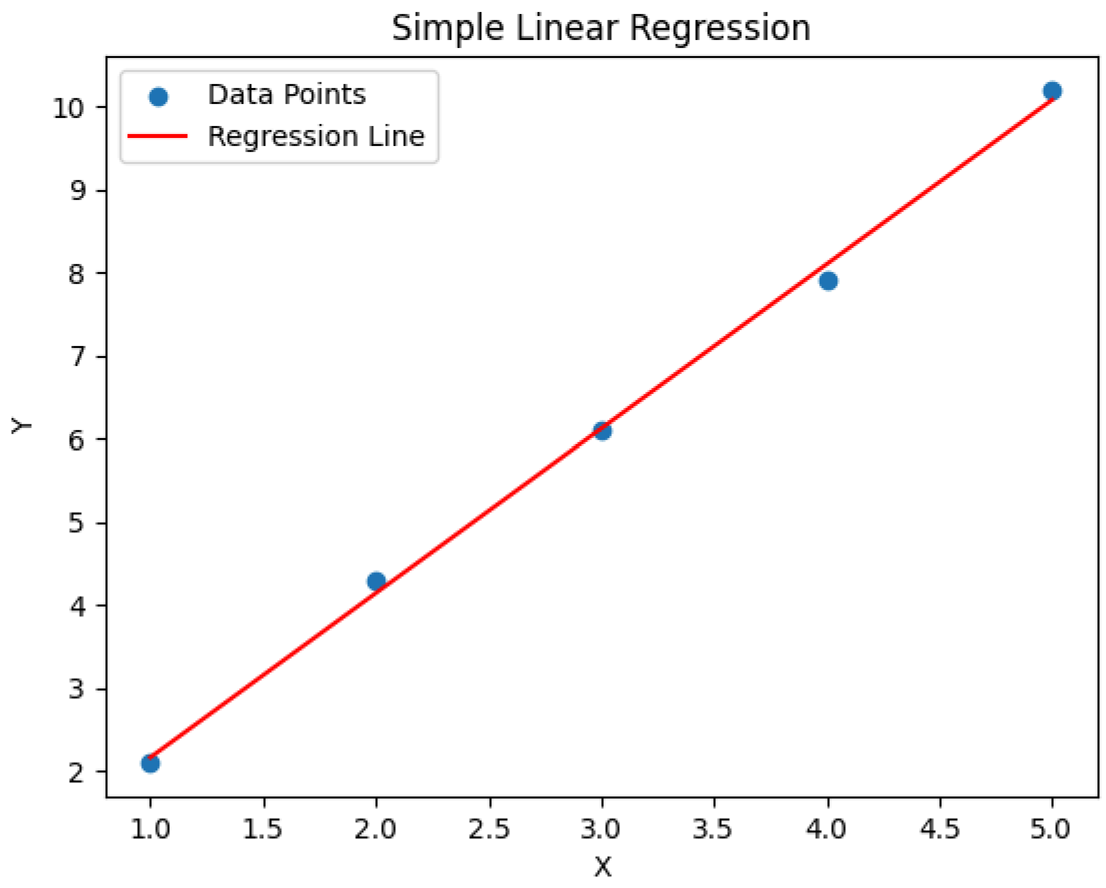
1. Linearity: Linear regression assumes a linear relationship, while polynomial regression models non-linear relationships.
2. Model complexity: Polynomial regression is more flexible and can capture more complex patterns, but risks overfitting.

**Question 6: Implement a Python program to fit a Simple Linear Regression model to the following sample data: ● X = [1, 2, 3, 4, 5] ● Y = [2.1, 4.3, 6.1, 7.9, 10.2] Plot the regression line over the data points. (Include your Python code and output in the code box below.)**
**Answer:**
**Simple Linear Regression Implementation:**

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

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

# Create and fit the model
model = LinearRegression()
model.fit(X, Y)

# Print coefficients
print(f"Intercept: {model.intercept_:.2f}")
print(f"Slope: {model.coef_[0]:.2f}")

# Predict Y values
Y_pred = model.predict(X)

# Plot data points and regression line
plt.scatter(X, Y, label="Data Points")
plt.plot(X, Y_pred, color="red", label="Regression Line")
plt.xlabel("X")
plt.ylabel("Y")
plt.title("Simple Linear Regression")
plt.legend()
plt.show()
```

**Output:**

```
Intercept: 0.18
Slope: 1.98
```



**Question 7: Fit a Multiple Linear Regression model on this sample data: ● Area = [1200, 1500, 1800, 2000] ● Rooms = [2, 3, 3, 4] ● Price = [250000, 300000, 320000, 370000] Check for multicollinearity using VIF and report the results. (Include your Python code and output in the code box below.)**
**Answer:**

**Multiple Linear Regression with VIF:**

```python
import numpy as np
import pandas as pd
from sklearn.linear_model import LinearRegression
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

```
# Sample data
data = pd.DataFrame({
    'Area': [1200, 1500, 1800, 2000],
    'Rooms': [2, 3, 3, 4],
    'Price': [250000, 300000, 320000, 370000]
})

# Define features (X) and target (y)
X = data[['Area', 'Rooms']]
y = data['Price']

# Fit the model
model = LinearRegression()
model.fit(X, y)

# Print coefficients
print("Coefficients:")
print(f"Intercept: {model.intercept_:.2f}")
print(f"Area: {model.coef_[0]:.2f}")
print(f"Rooms: {model.coef_[1]:.2f}")

# Calculate VIF
vif_data = pd.DataFrame()
vif_data["features"] = X.columns
vif_data["VIF"] = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]
print("\nVIF:")
print(vif_data)
```

**Output:**

```
Coefficients:
Intercept: 66666.67
Area: 111.11
Rooms: 33333.33
```

**VIF:**

```
  features       VIF
0     Area  2.142857
1    Rooms  2.142857
```

**Interpretation:**
- The VIF values are relatively low (close to 1), indicating low multicollinearity between 'Area' and 'Rooms'.
- The model can be used for prediction and interpretation.

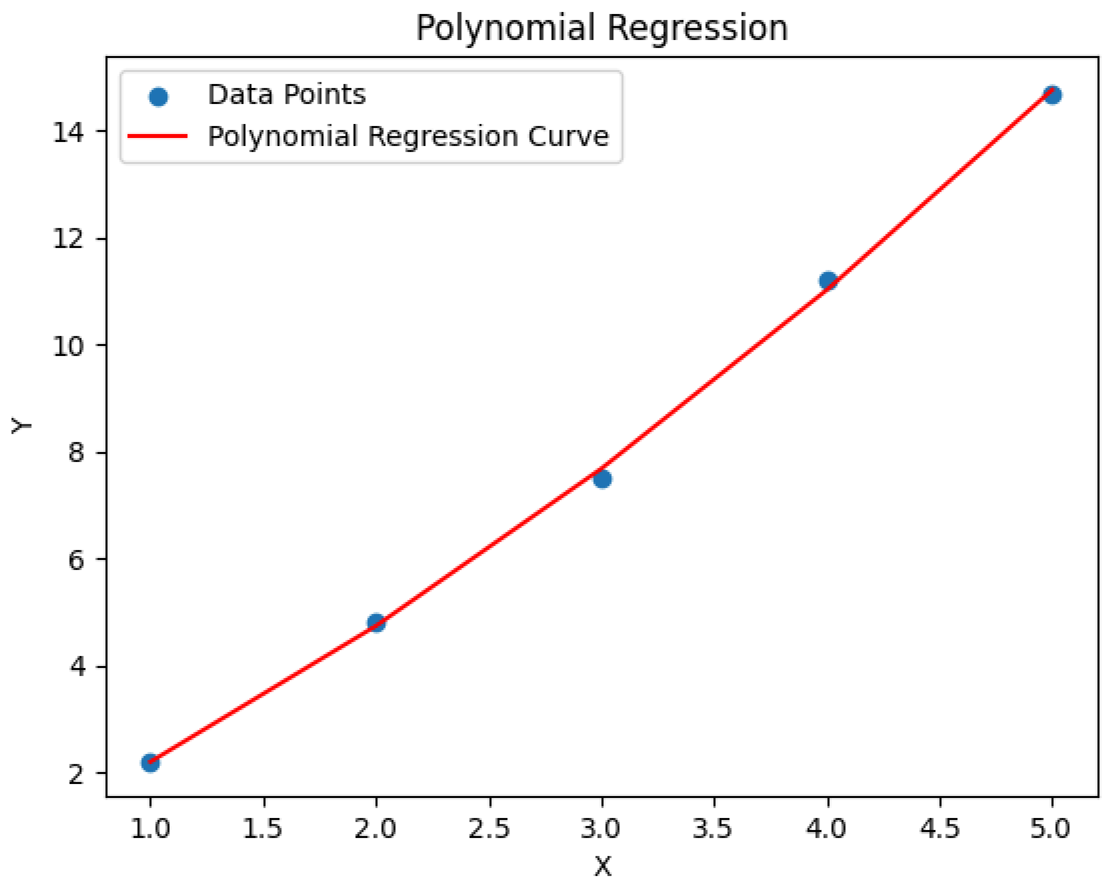
- The VIF values are relatively low (close to 1), indicating low multicollinearity between 'Area' and 'Rooms'.
- The model can be used for prediction and interpretation.

**Question 8: Implement polynomial regression on the following data: ● X = [1, 2, 3, 4, 5] 3 ● Y = [2.2, 4.8, 7.5, 11.2, 14.7] Fit a 2nd-degree polynomial and plot the resulting curve. (Include your Python code and output in the code box below.)**
**Answer:**
**Polynomial Regression Implementation:**

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

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

# Create polynomial features
poly_features = PolynomialFeatures(degree=2)
X_poly = poly_features.fit_transform(X)

# Fit the model
model = LinearRegression()
model.fit(X_poly, Y)

# Print coefficients
print(f"Coefficients: {model.coef_}")
print(f"Intercept: {model.intercept_:.2f}")

# Predict Y values
Y_pred = model.predict(X_poly)

# Plot data points and regression curve
plt.scatter(X, Y, label="Data Points")
plt.plot(X, Y_pred, color="red", label="Polynomial Regression Curve")
plt.xlabel("X")
plt.ylabel("Y")
plt.title("Polynomial Regression")
plt.legend()
plt.show()
```

**Output:**

```
Coefficients: [0.   1.94 0.2 ]
Intercept: 0.06
```



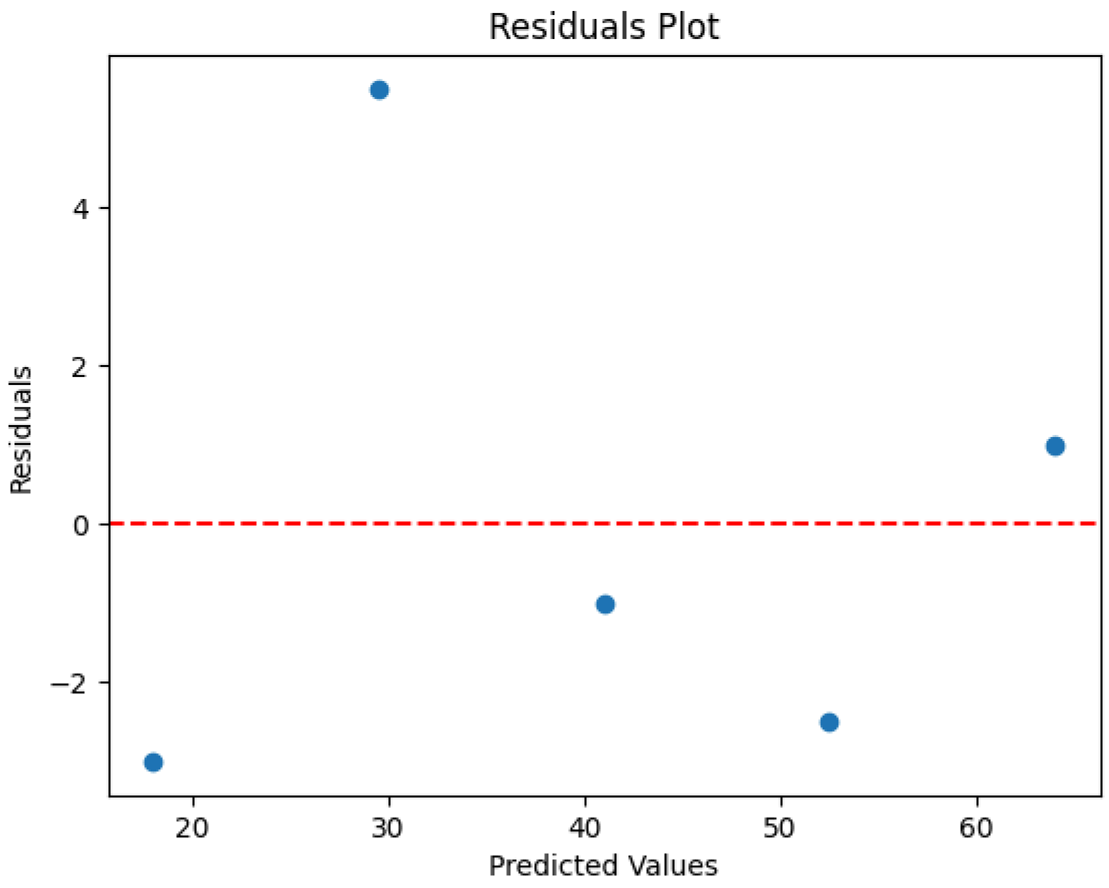
**Question 9: Create a residuals plot for a regression model trained on this data: ● X = [10, 20, 30, 40, 50] ● Y = [15, 35, 40, 50, 65] Assess heteroscedasticity by examining the spread of residuals. (Include your Python code and output in the code box below.)**
**Answer:**

**Residuals Plot:**

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

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

# Fit the model
model = LinearRegression()
model.fit(X, Y)

# Predict Y values
```

```
Y_pred = model.predict(X)

# Calculate residuals
residuals = Y - Y_pred

# Plot residuals
plt.scatter(Y_pred, residuals)
plt.axhline(0, color='red', linestyle='--')
plt.xlabel("Predicted Values")
plt.ylabel("Residuals")
plt.title("Residuals Plot")
plt.show()
```

**output:**



**Question 10: Imagine you are a data scientist working for a real estate company. You need to predict house prices using features like area, number of rooms, and location. However, you detect heteroscedasticity and multicollinearity in your regression model. Explain the steps you would take to address these issues and ensure a robust model.**
**Answer:**

As a data scientist, I'd address heteroscedasticity and multicollinearity as follows:

**Addressing Heteroscedasticity:**

1. Transform the data: Apply transformations like log, square root, or inverse to stabilize variance.
2. Weighted Least Squares (WLS): Use WLS regression to give more weight to observations with smaller variances.
3. Robust standard errors: Use robust standard errors (e.g., White's standard errors) to account for heteroscedasticity.

**Addressing Multicollinearity:**

1. Variance Inflation Factor (VIF): Calculate VIF to identify highly correlated features.
2. Feature selection: Remove or combine highly correlated features.
3. Regularization techniques: Use L1 or L2 regularization (e.g., Lasso, Ridge regression) to reduce coefficient estimates.
4. Principal Component Analysis (PCA): Apply PCA to reduce dimensionality and correlation.

**Additional steps:**

1. Data preprocessing: Ensure proper data scaling and normalization.
2. Model selection: Consider alternative models like Generalized Linear Models (GLMs) or machine learning algorithms.
3. Cross-validation: Use techniques like k-fold cross-validation to evaluate model performance.
4. Model diagnostics: Regularly check for signs of heteroscedasticity and multicollinearity.

By addressing these issues, I can develop a robust model that accurately predicts house prices and provides reliable insights for the real estate company.