Problem Set 1

Code notebook for relevant questions

Problem 3 (20 Points)

You are given a data set that includes wages, education status, and years of experience for 20 individuals. The education status is categorized into two groups: College and Non-College. Your task is to analyze the relationship between education, years of experience, and wage using OLS. The data set is provided in Table 1.

Table 1: Sample Data for Problem 3

Wage ($)	Education	Years of Experience
55	College	5
67	College	11
60	College	7
63	College	9
58	College	6
65	College	10
62	College	8
61	College	8
64	College	9
66	College	10
45	Non-College	4
49	Non-College	5
44	Non-College	2
48	Non-College	4
50	Non-College	5
46	Non-College	3
46	Non-College	3
47	Non-College	3
42	Non-College	1
43	Non-College	2

Task

1. Using Python, estimate the regression model with wage as the dependent variable and both education and years of experience as the independent variables (as well as an intercept!). **Hint:** Education serves as a dummy variable.

2. Interpretation:

   - What does the coefficient for the dummy variable (education) tell you about the difference in wages between college-educated and non-college-educated individuals, holding experience constant?
   - What does the coefficient for experience tell you about how wages increase with experience, holding education constant?

3. Predict the expected wage for:

   - A college-educated individual with 6 years of experience.
   - A non-college-educated individual with 4 years of experience.

Task 1

In [11]:
```
import pandas as pd
import statsmodels.api as sm

# data from problem 3
data = {
    'Wage': [55, 67, 60, 63, 58, 65, 62, 61, 64, 66, 45, 49, 44, 48, 50, 46, 46,
    'Education': ['College']*10 + ['Non-College']*10,
    'Years of Experience': [5, 11, 7, 9, 6, 10, 8, 8, 9, 10, 4, 5, 2, 4, 5, 3, 3
}

# Create DataFrame
df = pd.DataFrame(data)

# Convert 'Education' to dummy variable using one-hot encoding
# 1 - College, 0 - Non-College
df['Education'] = df['Education'].apply(lambda x: 1 if x == 'College' else 0)

# Define dependent and independent variables
X = df[['Education', 'Years of Experience']]
X = sm.add_constant(X)  # Adds a constant term to the predictor
y = df['Wage']

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

# Print the summary of the regression
print(model.summary())
```

```
                            OLS Regression Results                            
==============================================================================
Dep. Variable:                   Wage   R-squared:                       0.992
Model:                            OLS   Adj. R-squared:                  0.991
Method:                 Least Squares   F-statistic:                     1004.
Date:                Thu, 12 Sep 2024   Prob (F-statistic):           2.27e-18
Time:                        10:09:19   Log-Likelihood:                -23.628
No. Observations:                  20   AIC:                             53.26
Df Residuals:                      17   BIC:                             56.24
Df Model:                           2                                         
Covariance Type:            nonrobust                                         
==================================================================================
======
                          coef    std err          t      P>|t|      [0.025      
0.975]
----------------------------------------------------------------------------------
------
const                  39.9153      0.480     83.191      0.000      38.903      
40.928
Education               6.4025      0.738      8.671      0.000       4.845       
7.960
Years of Experience     1.9015      0.124     15.354      0.000       1.640       
2.163
==============================================================================
Omnibus:                       13.961   Durbin-Watson:                   1.888
Prob(Omnibus):                  0.001   Jarque-Bera (JB):               13.330
Skew:                          -1.385   Prob(JB):                      0.00128
Kurtosis:                       5.885   Cond. No.                         27.4
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly 
specified.
```

Task 2

In [9]:
```python
# Interpretation
edu_coeff = model.params['Education']
exp_coeff = model.params['Years of Experience']

print(f"Education Coefficient: {edu_coeff}")
print(f"Experience Coefficient: {exp_coeff}")
```

```
Education Coefficient: 6.402515723270483
Experience Coefficient: 1.9014675052410865
```

2.1 The coefficient for the dummy variable (education) is `6.4025`. This tells us that, holding years of experience constant, college-educated individuals earn, on average, `$6.40` more per hour (per unit) than individuals without college education *(assuming the data provided measures wage in terms of hourly wages)*. This coefficient represents the average wage premium associated with having a college education.

2.2 The coefficient for Experience is `1.9015`. This tells us that, holding Education constant, experience (in terms of years) increases wage by approximately `$1.90` per hour *(again, assuming the data for wages is an hourly wage)*. This means that for every additional year of an experience that a worker has, the average increase in wage is

`$1.90` per hour. This coefficient indicates that wages is increases linearly (positive correlation) with years of experience of worker.

To further interpret this result, we go one step further to check the statistical significance of the coefficient.

Additionally, we also consider the R-squared value of the model to understand how well the model explains the variability in wages. Typically, higher R-squared value indicates a better fit of the model to the data.

Code to check significance and R-sq values

In [3]:
```
# Extract p-values for the coefficients
edu_pval = model.pvalues['Education']
exp_pval = model.pvalues['Years of Experience']
print(f"P-value for Education Coefficient: {edu_pval}")
print(f"P-value for Experience Coefficient: {exp_pval}")

# Extract R-squared and Adjusted R-squared values
r_squared = model.rsquared
adj_r_squared = model.rsquared_adj
print(f"R-squared: {r_squared}")
print(f"Adjusted R-squared: {adj_r_squared}")
```

```
P-value for Education Coefficient: 1.1991195013600496e-07
P-value for Experience Coefficient: 2.1375038613226402e-11
R-squared: 0.9916020815863917
Adjusted R-squared: 0.9906140911847907
```

We see that both the coefficients for the education and experience variable is indeed significant since the p-value associated with each respective coefficient is less than `0.05`.

P-value for Education Coefficient: `1.1991195013600496e-07` < `0.05` This tells us that there is a statistically significant difference in wages between `college-educated` and `non-college-educated` individuals, holding `Experience` constant.

P-value for Experience Coefficient: `2.1375038613226402e-11` < `0.05` This tells us that there is a statistically significant difference in wages for each additional `year of experience` of an individual, holding `Education` constant.

Furthermore, the high R-squared value of `0.991` and the adjusted R-squared value of `0.992` indicate that the model is a good fit for the data. The values suggests that the model explains approximately 99% of the variability in terms of the Wages, indicating a strong relationship between the independent variables (Education & Experience) and the dependent variable (wage).

Task 3

In [10]:
```
# 1
college_6_years = model.predict([1, 1, 6])[0]  # Intercept, College, 6 years
# 2
non_college_4_years = model.predict([1, 0, 4])[0]  # Intercept, Non-College, 4 y
```

```
print(f"Predicted wage for a college-educated individual with 6 years of experie
print(f"Predicted wage for a non-college-educated individual with 4 years of exp
```

```
Predicted wage for a college-educated individual with 6 years of experience: $57.
73
Predicted wage for a non-college-educated individual with 4 years of experience:
$47.52
```

Some Personal Notes and Learning

Adding a constant term to the predictor in a regression model is important for the following reasons:

1. Intercept Estimation: The constant term (intercept) allows the model to estimate the baseline value of the dependent variable when all independent variables are zero. Without it, the model is forced to pass through the origin (0,0), which may not be appropriate for the data.

2. Model Flexibility: Including a constant term increases the flexibility of the model, allowing it to fit the data better by adjusting the baseline level.

3. Bias Reduction: It helps in reducing bias in the estimation of the coefficients of the independent variables.

4. Statistical Properties: Many statistical properties and tests assume that the model includes an intercept.

reference line of code: `X = sm.add_constant(X)  # Adds a constant term to the predictor`

TLDR:

- This line adds a column of ones to the predictor matrix X, which represents the intercept term in the regression model. This allows the model to estimate the baseline wage when both Education and Years of Experience are zero.