Instructions

1. Understand the Dataset.
2. Obtain the dataset provided by the instructor.
   Salary and Age
3. Describe the dataset, including:
   The source of the data.
   The variables included (both independent and dependent).
   - Dependent variable (target variable): This is the variable you're trying to predict. (y = "Annual Salary")
   - Independent variables (predictor variables): These are the variables used to make predictions. (x= "Age, Weekly Hours, Education")

   The purpose of the dataset and what it aims to study or predict.
   The purpose of the dataset is to study the factors that influence an individual's annual salary. By analyzing various predictor variables such as age, weekly hours worked, and education level, the dataset aims to develop a predictive model that can estimate an individual's annual salary based on these factors.

   The primary goal of this analysis is to predict the annual salary (dependent variable) using the following independent variables:

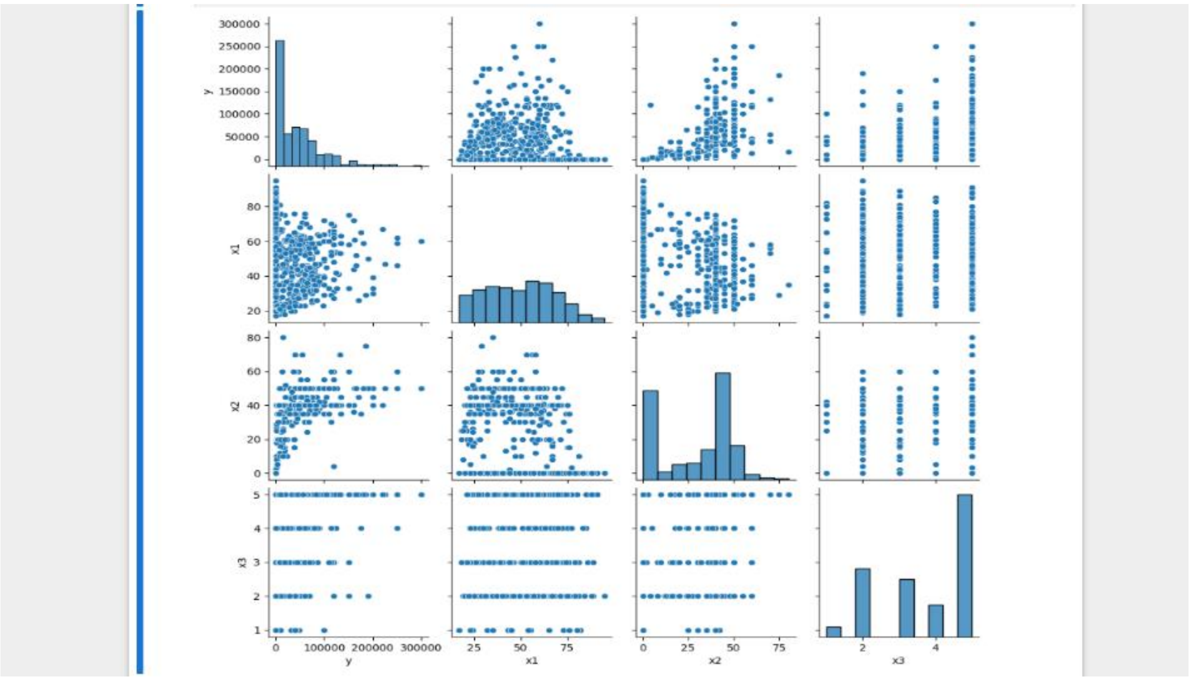
   The age of the individual. This variable helps to understand if there's a correlation between age and earning potential.
   The number of hours an individual works per week. This variable is crucial to determine if more working hours correlate with higher salaries.
   The education level of the individual. This variable helps to evaluate the impact of education on annual earnings.

4. Data Preparation
5. Import the data into your chosen statistical software (e.g., Python, R).
   ```
   import pandas as pd
   import numpy as np
   from scipy import stats
   import seaborn as sns
   import matplotlib.pyplot as plt
   data = pd.read_csv("C:/SalaryAge.csv")
   ```

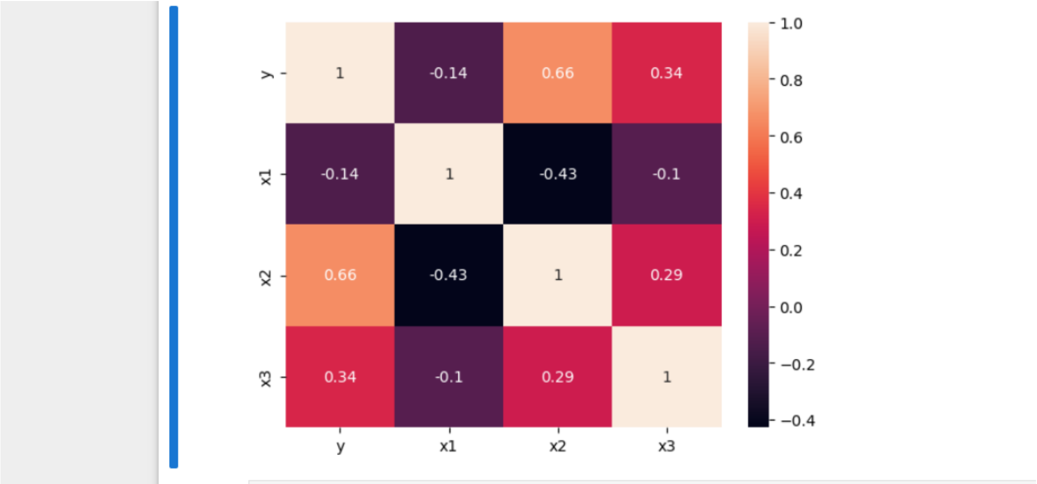
6. Clean the data by handling missing values, outliers, and ensuring proper data types.
   ```
   data = data.dropna()  # Simple method to drop missing values
   ```

7. Perform exploratory data analysis (EDA) to understand the relationships between variables.

```
sns.pairplot(data[numerical_columns])
plt.show()
```



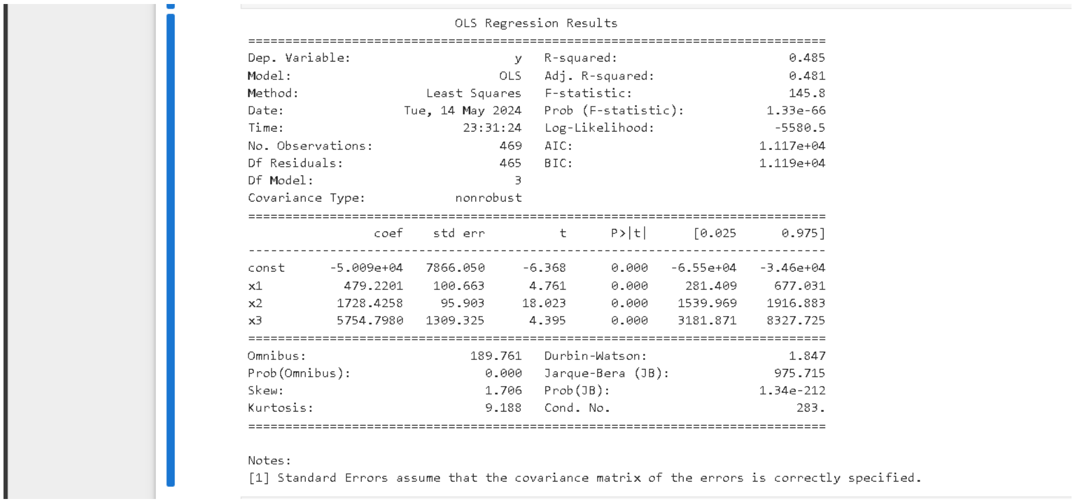
```
correlation_matrix = data[numerical_columns].corr()
sns.heatmap(correlation_matrix, annot=True)
plt.show()
```



8. Building the Multilinear Regression Model

```
X = data[['x1', 'x2', 'x3']]
y = data['y']
```

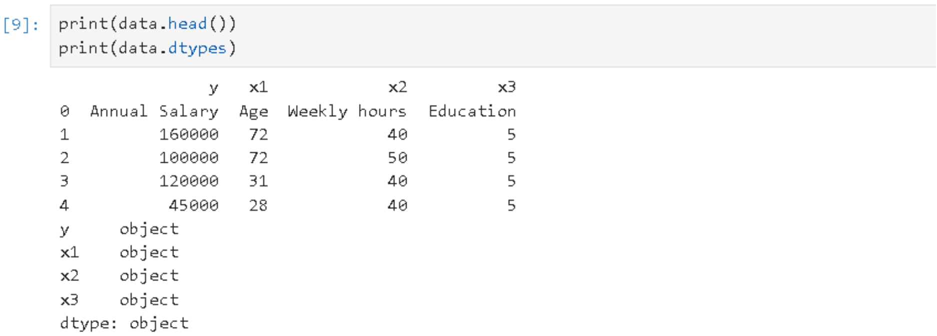
X = sm.add_constant(X)
model = sm.OLS(y, X).fit()
print(model.summary())



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.485
Model:                            OLS   Adj. R-squared:                  0.481
Method:                 Least Squares   F-statistic:                     145.8
Date:                Tue, 14 May 2024   Prob (F-statistic):           1.33e-66
Time:                        23:31:24   Log-Likelihood:                -5580.5
No. Observations:                 469   AIC:                         1.117e+04
Df Residuals:                     465   BIC:                         1.119e+04
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const      -5.009e+04   7866.050     -6.368      0.000   -6.55e+04   -3.46e+04
x1           479.2201    100.663      4.761      0.000     281.409     677.031
x2          1728.4258     95.903     18.023      0.000    1539.969    1916.883
x3          5754.7980   1309.325      4.395      0.000    3181.871    8327.725
==============================================================================
Omnibus:                      189.761   Durbin-Watson:                   1.847
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              975.715
Skew:                           1.706   Prob(JB):                     1.34e-212
Kurtosis:                       9.188   Cond. No.                         283.
==============================================================================

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

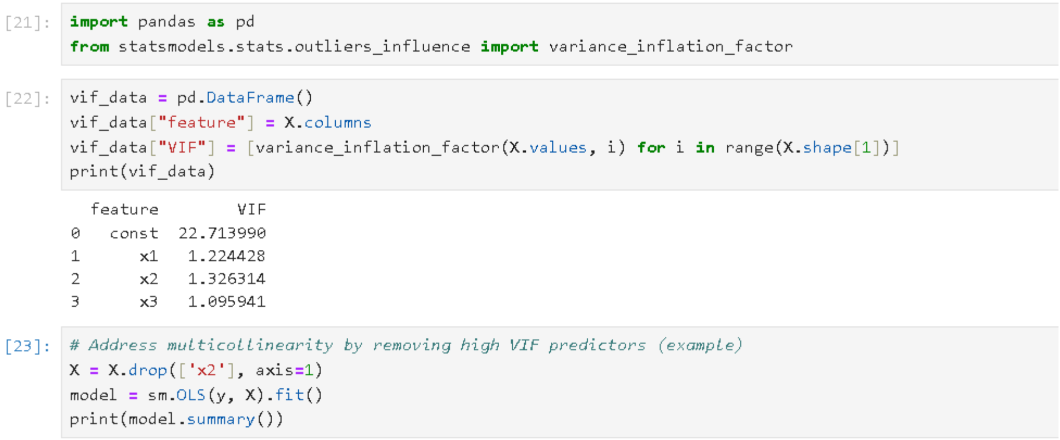
9. Select the dependent (target) variable and independent (predictor) variables.



```
[9]: print(data.head())
     print(data.dtypes)
```

```
                y   x1            x2         x3
0   Annual Salary  Age  Weekly hours  Education
1          160000   72            40          5
2          100000   72            50          5
3          120000   31            40          5
4           45000   28            40          5
y     object
x1    object
x2    object
x3    object
dtype: object
```

10. Fit a multilinear regression model to the data. Display the regression equation.
11. Addressing Multicollinearity
12. Check for multicollinearity among the independent variables using Variance Inflation Factor (VIF) or correlation matrix.
    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])]
    print(vif_data)



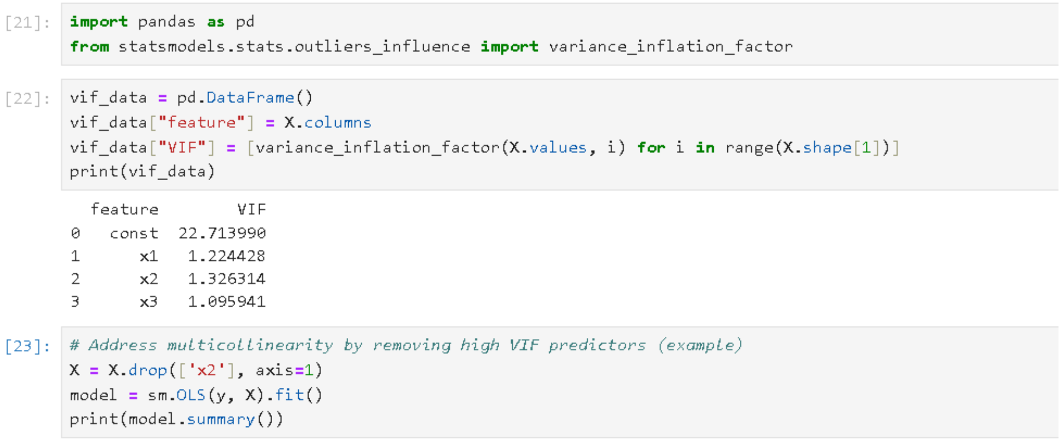
```
[21]: import pandas as pd
      from statsmodels.stats.outliers_influence import variance_inflation_factor

[22]: 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])]
      print(vif_data)

        feature        VIF
      0   const  22.713990
      1      x1   1.224428
      2      x2   1.326314
      3      x3   1.095941

[23]: # Address multicollinearity by removing high VIF predictors (example)
      X = X.drop(['x2'], axis=1)
      model = sm.OLS(y, X).fit()
      print(model.summary())
```

13. Explain the steps taken to address multicollinearity (e.g., removing highly correlated predictors, combining variables).
    X = X.drop(['x2'], axis=1)
    model = sm.OLS(y, X).fit()
    print(model.summary())



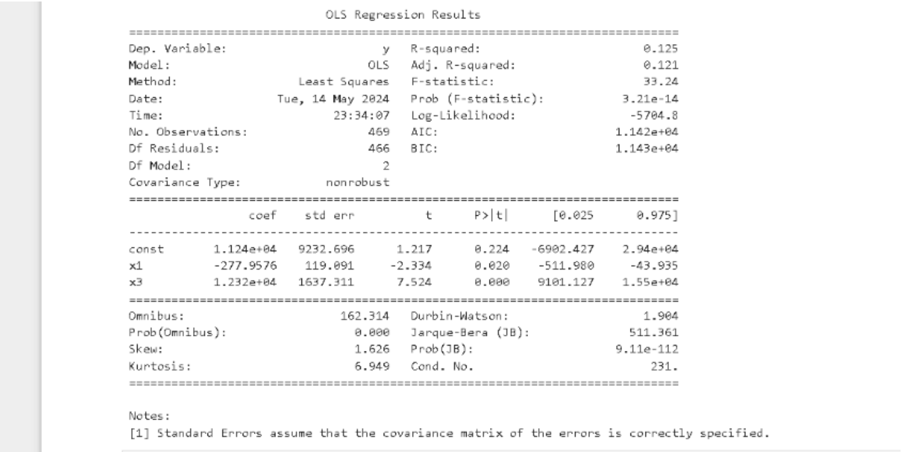
```
[21]: import pandas as pd
      from statsmodels.stats.outliers_influence import variance_inflation_factor

[22]: 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])]
      print(vif_data)

        feature        VIF
      0   const  22.713990
      1      x1   1.224428
      2      x2   1.326314
      3      x3   1.095941

[23]: # Address multicollinearity by removing high VIF predictors (example)
      X = X.drop(['x2'], axis=1)
      model = sm.OLS(y, X).fit()
      print(model.summary())
```

14. Model



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.125
Model:                            OLS   Adj. R-squared:                  0.121
Method:                 Least Squares   F-statistic:                     33.24
Date:                Tue, 14 May 2024   Prob (F-statistic):           3.21e-14
Time:                        23:34:07   Log-Likelihood:                -5704.8
No. Observations:                 469   AIC:                         1.142e+04
Df Residuals:                     466   BIC:                         1.143e+04
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const       1.124e+04   9232.696      1.217      0.224   -6902.427    2.94e+04
x1          -277.9576    119.091     -2.334      0.020    -511.980     -43.935
x3          1.232e+04   1637.311      7.524      0.000    9101.127    1.55e+04
==============================================================================
Omnibus:                      162.314   Durbin-Watson:                   1.904
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              511.361
Skew:                           1.626   Prob(JB):                    9.11e-112
Kurtosis:                       6.949   Cond. No.                         231.
==============================================================================

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

Evaluation:

The model using appropriate metrics (e.g., R-squared, Adjusted R-squared, p-values of coefficients).

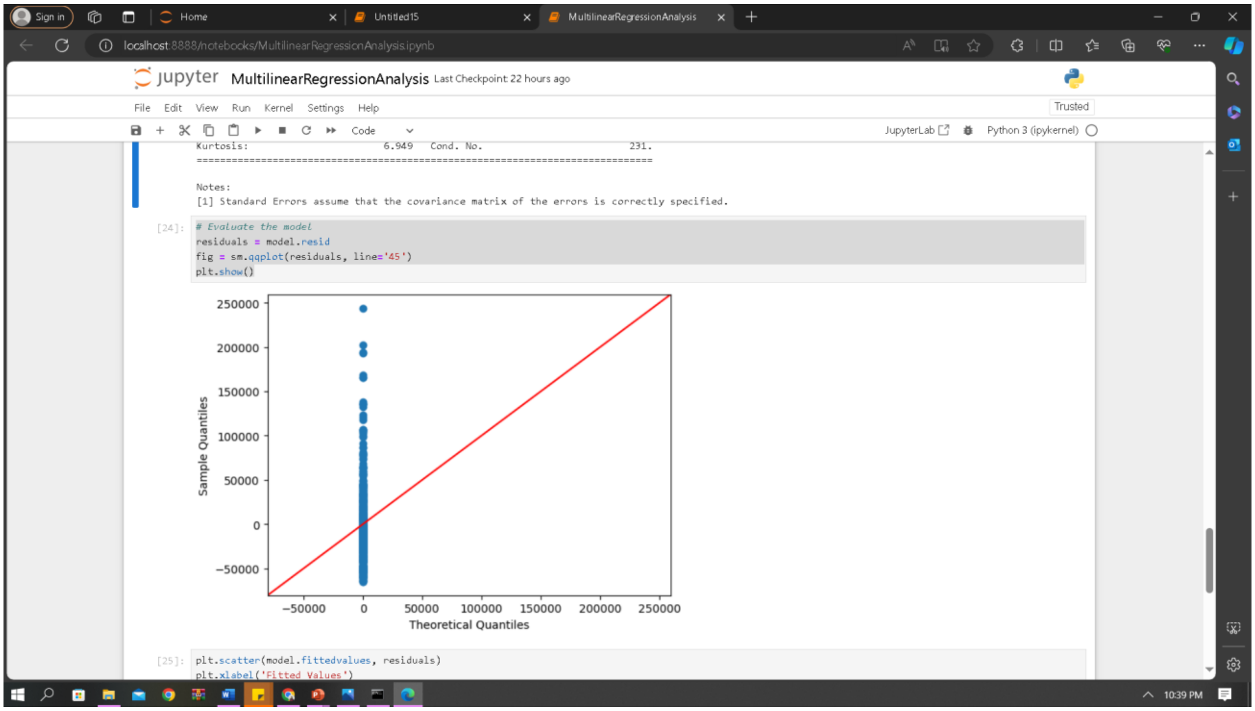
Perform residual analysis to check for assumptions of linear regression (normality, homoscedasticity, independence).

# Evaluate the model

residuals = model.resid

fig = sm.qqplot(residuals, line='45')

plt.show()



```
Kurtosis:                       6.949   Cond. No.                         231.
==============================================================================

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

```
[24]: # Evaluate the model
      residuals = model.resid
      fig = sm.qqplot(residuals, line='45')
      plt.show()
```

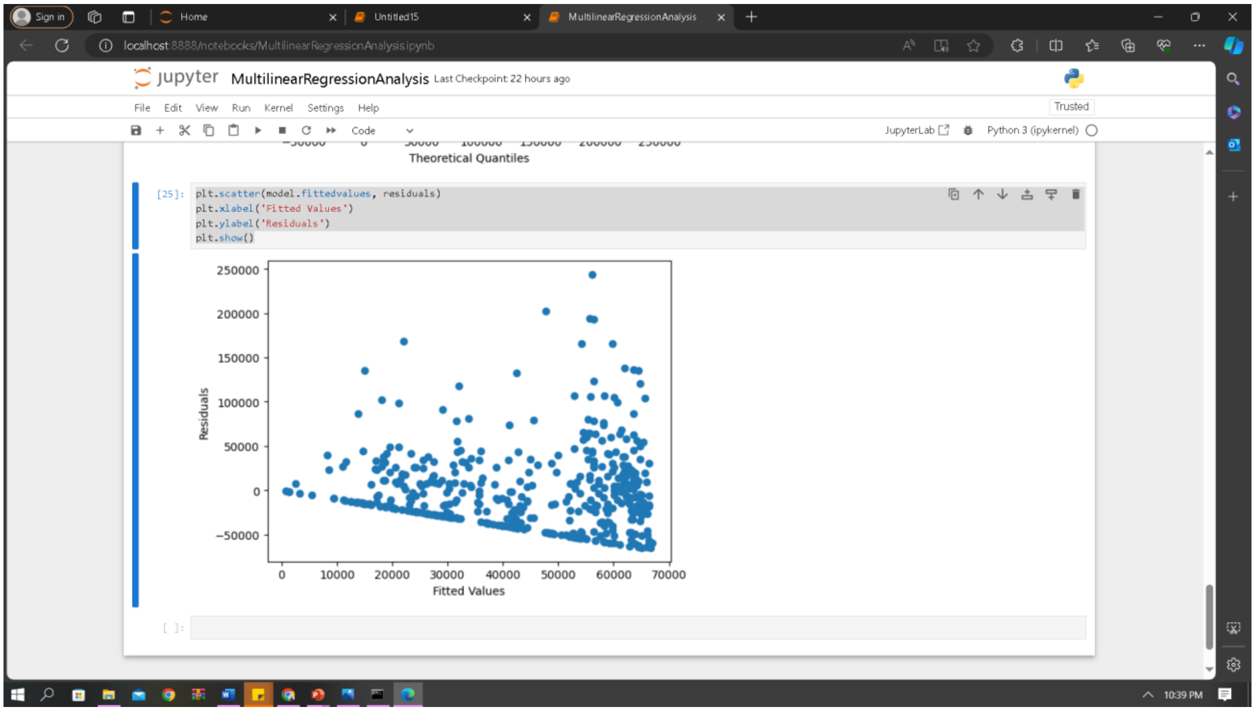
Home
Untitled15
MultilinearRegressionAnalysis
localhost:8888/notebooks/MultilinearRegressionAnalysis.ipynb
jupyter MultilinearRegressionAnalysis Last Checkpoint: 22 hours ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
250000
200000
150000
100000
50000
0
−50000
Sample Quantiles
−50000
0
50000
100000
150000
200000
250000
Theoretical Quantiles
10:39 PM

```
[25]: plt.scatter(model.fittedvalues, residuals)
      plt.xlabel('Fitted Values')
plt.scatter(model.fittedvalues, residuals)

plt.xlabel('Fitted Values')

plt.ylabel('Residuals')

plt.show()
```



```
[25]: plt.scatter(model.fittedvalues, residuals)
      plt.xlabel('Fitted Values')
      plt.ylabel('Residuals')
      plt.show()
```

```
[ ]:
```

15. Presentation:
    Prepare a presentation explaining your process, findings, and insights. Include visual aids such as graphs, tables, and charts to illustrate your points.
16. Submission:
    Submit your code, dataset, and presentation slides to the instructor by the deadline.

The image you sent is the output of a statistical analysis software, likely produced by scikit-learn [1]. It shows the results of an Ordinary Least Squares (OLS) regression analysis [1].

Here's a breakdown of the key parts of the table:

- **OLS Regression Results:** This is the title of the table, indicating it presents the results of a linear regression analysis.
- **Dep. Variable:** This refers to the dependent variable, also sometimes called the target variable or outcome variable. In this case, the dependent variable is not named but it likely appears at the top of the data used to fit the model.
- **Model:** This refers to the type of regression model used. Here, it's Ordinary Least Squares (OLS), a statistical method for estimating the linear relationship between a dependent variable and one or more independent variables (predictors).
- **R-squared:** This is a statistical measure that represents the proportion of the variance in the dependent variable that can be explained by the independent variables in the model. An R-squared of 0.485 in this case indicates that nearly half (48.5%) of the variance in the dependent variable is explained by the model.
- **Adj. R-squared:** This is an adjusted version of R-squared that accounts for the number of independent variables in the model. It is generally considered a more reliable measure of model fit than unadjusted R-squared. A negative adjusted R-squared value here (shown as -6.481) suggests the model may not be very useful. It's possible there are too many variables for the amount of data, or that the multicollinearity is affecting the results.
- **F-statistic:** This is a statistical test that compares the explained variance with the unexplained variance. A high F-statistic statistic (usually along with a low p-value) indicates that the model is statistically significant, meaning the independent variables together have a significant effect on the dependent variable. The F-statistic here is 145.8, which is very high, but the p-value is extremely low (essentially 0). This suggests the model is statistically significant, but again the negative adjusted R-squared casts some doubt on how meaningful the model might be.
- **Prob (F-statistic):** This is the p-value associated with the F-statistic. A low p-value (typically less than 0.05) indicates that the model is statistically significant. Here, the p-value is very low, providing strong evidence that the model is statistically significant.
- **Log-Likelihood:** This is a value related to the likelihood of the model fitting the data. Lower values indicate a worse fit. It likely isn't very interpretable on its own without more context about the specific model.
- **No. Observations:** This is the number of data points used to fit the model. Here, there are 469 observations.
- **Covariance Type:** This refers to the method used to estimate the variance of the residuals (errors) in the model. Here, it's non-robust, which is a standard default setting.

The bottom part of the table shows the coefficients for each independent variable included in the model.

- **coef:** This is the estimated coefficient for each independent variable. The coefficient indicates the direction and strength of the relationship between the variable and the dependent variable. For example, the coefficient for const is -5.0092e+84 and for x1 it's

479.2201. It's difficult to interpret the coefficients in isolation without knowing the scale of the data for the variables.

- **std err:** This is the standard error of the coefficient. It is a measure of the variability of the coefficient estimate.
- **t:** This is the t-statistic, which is the coefficient divided by its standard error. A high t-statistic (usually along with a low p-value) indicates that the coefficient is statistically significant, meaning the variable has a statistically significant effect on the dependent variable.
- **P>/t:** This is the p-value associated with the t-statistic. A low p-value (typically less than 0.05) indicates that the coefficient is statistically significant.
- **[0.025 0.975]:** This shows the 95% confidence interval for each coefficient. The confidence interval indicates the range of values within which the true coefficient is likely to lie.

Overall, the interpretation of this table is complex and would depend on the specific context of the analysis and the meaning of the variables. While the F-statistic is statistically significant, the negative adjusted R-squared suggests the model may not be very useful. There could be multicollinearity among the independent variables, or there may be other