Practical Exercise 1: Linear Regression and Statistical Testing using Python

Tutorial Objectives:

- Data manipulation with Python.
- Data exploration.
- Use Python to calculate the linear model estimators.
- Conduct a statistical test.

Dataset:

We will use the "House Prices - Advanced Regression Techniques" dataset from Kaggle. Specifically, we'll use the `train.csv` file.

Exercise Steps:

1. Load the `train.csv` file into a pandas DataFrame.

- Import the necessary library (`pandas`) and load the dataset.
- Ensure the dataset is correctly loaded by displaying the first few rows.

2. Display basic information and statistical summaries of the dataset using functions like `info()` and `describe()`.

- Use `info()` to display information about the DataFrame.
- Use `describe()` to display statistical summaries of the numerical columns.

3. Assign the feature `GrLivArea` (Above grade (ground) living area square feet) to a variable `x` and the target variable `SalePrice` to a variable `y`.

- Extract the `GrLivArea` column and assign it to `x`.
- Extract the `SalePrice` column and assign it to `y`.

4. Plot histograms to show the distribution of `x` and `y`.

- Use matplotlib to plot the histogram of `x`.
- Plot the histogram of `y`.
- Label the axes and add appropriate titles.

5. Create a scatter plot of `SalePrice` versus `GrLivArea`.

- Plot `y` against `x` using a scatter plot.
- Label the axes and add a title to the plot.

6. Calculate the estimators for simple linear regression (intercept and slope) manually using the formulas for least squares estimation. Store these values in a vector called `simple_ml_estim`.

- Calculate the mean of `x` and `y`.
- Compute the slope (`beta1`) and intercept (`beta0`) manually.
- Store the results in `simple_ml_estim`.

7. Using the normal equation, calculate the estimator values again and store them in a vector called `normal_ml_estim`.

- Prepare the design matrix `X` including a column of ones for the intercept.
- Use the normal equation: β=(XTX)−1XTy\beta = (X^TX)^{-1}X^Tyβ=(XTX)−1XTy.
- Store the results in `normal_ml_estim`.

8. Use the `LinearRegression` function from scikit-learn to fit a simple regression model and assign the coefficients to a new vector `r_ml_estim`.

- Import the `LinearRegression` class from scikit-learn.
- Fit the model using `x` and `y`.
- Extract the intercept and coefficient.
- Store the results in `r_ml_estim`.

9. Test the equality between `simple_ml_estim`, `normal_ml_estim`, and `r_ml_estim`.

- Compare the three sets of estimators to check if they are equal or approximately equal.
- Document any discrepancies.

10. Based on the predicted/fitted values and residuals from the model, conduct a residuals analysis to check the Gauss-Markov assumptions.

You can plot residuals versus fitted values, a histogram of residuals, and a Q-Q plot.

- Calculate the predicted values and residuals.
- Plot residuals versus fitted values.
- Plot a histogram of residuals.
- Create a Q-Q plot of residuals.
- Interpret the plots to assess linearity, normality, and homoscedasticity.

11. Based on your analysis, what do you suggest in terms of transforming the target variable `y`?

- Consider if a transformation (e.g., logarithmic) of `y` is necessary.
- Justify your suggestion based on the residuals analysis.

12. Apply the suggested transformation to `y` and rerun the linear regression using `LinearRegression`.

- Transform `y` accordingly.
- Fit the linear regression model with the transformed `y`.
- Perform residuals analysis on the new model.

13. Calculate the unbiased estimate of the variance of the error terms.

- Compute the sum of squared residuals (SSR).
- Calculate the variance using the formula: σ2=SSRn−p−1\sigma^2 = \frac{SSR}{n - p - 1}σ2=n−p−1SSR, where nnn is the number of observations and ppp is the number of predictors.

14. Calculate the standard errors of the estimators (intercept and slope).

- Compute the variance-covariance matrix.
- Extract the variances of the intercept and slope.
- Calculate the standard errors by taking the square roots of the variances.

15. Compute the t-values for each coefficient.

- Divide each estimator by its standard error to obtain the t-values.

16. At a 1% significance level, perform a statistical test to determine if each coefficient is significantly different from zero.

- Determine the critical t-value for a 99% confidence level.
- Compare the calculated t-values with the critical t-value.
- State whether to reject or fail to reject the null hypothesis for each coefficient.

17. Calculate the p-values corresponding to the t-values computed above.

- Use the appropriate statistical function to compute the p-values.
- Interpret the p-values in the context of the hypothesis test.

18. Compare your results with the output of the `summary()` function from the `statsmodels` library.

- Use `statsmodels` to fit the regression model.
- Display the summary output.
- Compare coefficients, standard errors, t-values, and p-values with your calculations.

19. Find the 95% confidence intervals for the two coefficients.

- Calculate the confidence intervals using the standard errors and critical t-value.
- Alternatively, extract them from the `statsmodels` summary.

20. Find the prediction intervals for the first 10 observations in your dataset.

- Use the model to predict the values for the first 10 observations.
- Calculate the prediction intervals for these predictions.
- Explain the difference between prediction intervals and confidence intervals.

21. What is the R-squared value of your model? Based on this, is the model a good fit? Justify your answer.

- Extract the R-squared value from your model.
- Discuss what the R-squared value indicates about the model's performance.
- Provide reasoning on whether the model is appropriate to keep.