Due Date: 10.11.2024 23:55

CENG 463 INTRODUCTION TO MACHINE LEARNING HOMEWORK 1

- Correlation
- Simple Linear Regression
- Multiple Linear Regression
- Logistic Regression
- Polynomial Regression

!!!Please read the homework instructions, rules and the sections declared as important carefully.

In this assignment, you will explore key concepts in regression analysis using the heart disease dataset from the UCI Machine Learning Repository. The goal is to implement various regression techniques, including simple linear regression, multiple linear regression, logistic regression, and polynomial regression, to predict the likelihood of heart disease based on different features. The steps involve calculating the Pearson correlation between features and the target variable, selecting the most relevant features, and building predictive models. You will also visualize the regression results and evaluate model performance using metrics Mean Squared Error (MSE) and accuracy. Each regression model will be accompanied by relevant visualizations and performance metrics to compare their effectiveness in predicting heart disease.

Task 1: Data Loading and Exploratory Data Analysis

1. Download and Load the Dataset
   - We will use the heart disease dataset from the UCI Machine Learning Repository. The data is loaded into a Pandas DataFrame, and we assign column names for easier access.
   - Task: Load the dataset and display the first few rows using data.head(). Check for any missing values in the dataset.
   - https://archive.ics.uci.edu/dataset/45/heart+disease
2. Identify the Features and Target Variable
   - In this dataset, the features are all the columns except the 'target' column, which indicates the presence or absence of heart disease.
   - Task: Extract the features (independent variables) and the target (dependent variable) from the dataset. Print the names of all the feature columns and the target column.

Task 2: Correlation Analysis

1. Define the Pearson Correlation Function
   - The Pearson correlation coefficient is a measure of the linear relationship between two variables. The formula used to compute Pearson correlation is as follows:

$$r = \frac{n\sum xy - \sum x \sum y}{\sqrt{(n\sum x^2 - (\sum x)^2)(n\sum y^2 - (\sum y)^2)}}$$

   - Task: Implement this formula in a Python function named pearson_correlation. The function should take two arrays, x and y, and return the Pearson correlation coefficient.
2. Compute Correlations Between Each Feature and the Target
   - Now, we want to determine how each feature is correlated with the target variable. For this, we iterate through all features and compute the Pearson correlation coefficient between each feature and the target.
   - Task: For each feature, call the pearson_correlation function and store the results in a dictionary where the key is the feature name and the value is the correlation coefficient.
3. Filter Features Based on a Correlation Threshold
   - We set a threshold to filter out features with weak correlations. We only want to consider features with an absolute correlation coefficient greater than 0.3.
   - Task: Filter the features based on the threshold value and print the features that have a strong correlation with the target variable.

4. Find the Most Correlated Feature
   - Among the features that pass the threshold, we will determine which feature has the highest correlation with the target variable.
   - Task: Identify the feature with the highest absolute correlation coefficient and print its name and correlation value. Why are we examining the correlation between the features and target and what is the advantage of choosing mostly correlated features? What is the advantage for us? Write as a command on your code.

Task 3: Simple Linear Regression
In this task, we will implement simple linear regression using the most correlated feature to predict the target variable. We will compute the linear regression coefficients manually and generate predictions.

1. Create the Simple Linear Regression Function:
   - Write a Python function linear_regression(x, y) to calculate the intercept (b_0) and slope (b_1) of the regression line. We use the formulas:

$$b_1 = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sum(x_i - \bar{x})^2}$$

$$b_0 = \bar{y} - b_1 \cdot \bar{x}$$

2. Calculate Regression Coefficients:
   - Apply the function to the feature and target variables, obtaining the intercept and slope.
3. Generate Predictions:
   - Using the coefficients, calculate the predicted y values for each input x

$$\hat{y} = b_0 + b_1 x$$

4. Visualize the Results:
   - Plot a scatter plot of the real values and overlay the regression line to visually assess the fit.

Task 4: Multiple Linear Regression
Here, we extend the simple regression model to a multiple linear regression model, using several features simultaneously to predict target.
1. Create the Multiple Linear Regression Function:
   - Write a function multiple_linear_regression(X, y) that calculates regression coefficients using the normal equation:

$$\beta = (X^T X)^{-1} X^T y$$

   - where X is the matrix of features (with a column of ones for the intercept), and y is the target vector.
2. Select Features with High Correlation:
   - Use the top five most correlated features with the target (based on previous correlation results).
3. Calculate Regression Coefficients:
   - Apply the function to calculate the regression coefficients for the selected features.
4. Generate Predictions:
   - Use the calculated coefficients to predict the target variable:

$$\hat{y} = Xb \cdot \beta$$

    - where X_b is the feature matrix with an additional column of ones.
5. Visualize the Results:
    - Plot a graph comparing actual vs. predicted values.
6. Calculate Mean Squared Error (MSE):
    - Compute the Mean Squared Error (MSE) to evaluate the performance of both simple and multiple linear regression models.

Task 5: Logistic Regression
In this task, we will implement logistic regression using **Python's scikit-learn library**. The logistic regression model will be trained using the top five features that have a correlation coefficient above a given threshold (based on the correlation results from previous steps).

1. Prepare Data for Logistic Regression:
    - Use the selected features to prepare the feature matrix X and the target variable y. Split the dataset into training and testing sets using train_test_split from scikit-learn.
2. Train Logistic Regression Model:
    - Import LogisticRegression from scikit-learn and train a logistic regression model using the training data.
3. Evaluate the Model:
    - Use the testing set to evaluate the model's performance. Calculate and display the accuracy.
4. Visualize the Results:
    - Generate scatter plots that show both the actual target values and the predicted values for each data point.

Task 6: Polynomial Regression
In this task, we will implement polynomial regression using the feature that has the highest correlation with the target variable. We will use the **PolynomialFeatures function from scikit-learn** to transform the data into polynomial features and then perform regression using LinearRegression.

1. Transform the Feature into Polynomial Features:
    - Use the PolynomialFeatures function from scikit-learn to create polynomial terms (e.g., degree 2 or 3) for the selected feature.
2. Train a Polynomial Regression Model:
    - Use LinearRegression from scikit-learn to train the polynomial regression model using the transformed features.
3. Evaluate the Model:
    - Calculate and display the Mean Squared Error (MSE) for the polynomial regression model
4. Visualize the Results:
    - Generate scatter plots that show both the actual target values and the predicted values for each data point.

!!!!!IMPORTANT
At the end of **each** task, you are expected to discuss the performance of regression model in a text box. There have to be separate text boxes at the end of each task that you discuss the performance of the model and explain whether the model is suitable for the data and provide the reasoning behind your conclusion.

Also, you are expected to compare the performance of all models (with the performance metrics that you find in the task) as a table and answer that which model you think fits the data better and why.

Assignment Rules:
1. In this homework, no cheating is allowed. If any cheating is detected, the homework will be graded as 0, and no further discussion will be entertained.

2. You are expected to submit your homework in groups. Therefore, it will be sufficient if only one member of the group submits the homework.
3. You must upload a .txt file to MS Teams. In this file, include the link to your Google Colab notebook where you have done the project.
4. The .txt file must be named in the following format: group number, course code, and homework number. Example: **G01_CENG463_HW1**.
5. Please be aware that if you do not follow the assignment rules regarding export format and naming conventions, you will lose points.