IC 272 - Data Science III

Assignment 1: Data Preprocessing and Visualization

Deadline: September 10, 2023: 23.59 Hr.

Dataset Description:

A. You are given a CSV file "landslide_data_original.csv" containing the landslide-related readings from various sensors installed at 10 locations around the Mandi district. These sensors give details about the factors like temperature, humidity, pressure etc of a location. Following are the description and utility of the columns of the file for this assignment:

   i. "`dates`": date of collecting the data (not concerning any other information in its corresponding row for this task).

   Independent variables/ Attributes/ Features:

   i. "`temperature`": Atmospheric temperature around the sensor in Celsius.

   ii. " `humidity`": The concentration of water vapour present in the air (in g.m-3).

   iii. "`pressure`": Atmospheric pressure in millibars (mb).

   iv. "`rain`": Measure of rainfall in millilitres (ml).

   v. "`lightavg`": The average light throughout the daytime (in lux units).

   vi. "`lightmax`": The maximum lux count by the sensor.

   vii. "`moisture`": the amount of water stored in the soil (measured on a scale of 0 to 100).

   Dependent variable/ Target Attribute/ Class:

   i. "`stationid`": the location code of the sensors collecting respective row information.

B. You are given another file, "landslide_data_miss.csv". It is a version of the above file containing some missing values. The meaning and utility of the columns of this file are the same as above.

Problem Statements:

I. Write a Python program to read the given data file (using Pandas) and display the following:

1. Compute the mean, minimum, maximum, median, and standard deviation of the first attribute ("`temperature`") without using any built-in *statistical* function.

- Print these statistics up to 2 decimal places in self-explanatory format, e.g. "`The statistical measures of Temperature attribute are: mean=_, maximum=_, minimum=_, STD=_`".

2. Compute the Pearson correlation between all the attributes (independent variables). Print the correlation matrix in a tabular format with column names as the header and column names on the side of the table (you do not need to create anything fancy).
   - Given that the correlation value above or equal to 0.6 (-0.6) is considered a high correlation between two attributes, print the name of the redundant attribute with respect to "`lightavg`".

   Implement the following formula of Pearson correlation yourself; do **not** use any in-built correlation function. You may use a built-in *mean, sum* etc.

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

   r = correlation coefficient

   x_i = values of the x-variable in a sample

   $\bar{x}$ = mean of the values of the x-variable

   y_i = values of the y-variable in a sample

   $\bar{y}$ = mean of the values of the y-variable

3. Compute the histogram of `humidity` for "`stationid = t12`" without using any in-built **histogram** function; use **bin_size = 5**. Plot your computed histogram using a suitable type from **matplotlib.pyplot**. Your plot **must** have a title and descriptions for the x and y axes.

II. Write a Python program (with pandas) to do the following on the data file "landslide_data_miss.csv":

1. Drop the tuples (rows) having missing values in the target attribute ("`stationid`"). Delete (drop) the tuples (rows) having equal to or more than one-third of attributes with missing values (use in-built functions of Pandas).

2. Write a code to fill up the missing values in the attributes (independent variables) using the linear interpolation technique. Do Not use any in-built function, implement the code to compute linear interpolation using the previous and next values of a missing cell.
   a) Compute the mean, median, and standard deviation for each attribute and compare the same with that of the original file. Do Not use any in-built function to compute the statistics.
   b) Calculate the (RMSE) between the original and replaced values for each attribute (independent variables). Get original values from the original file provided. Implement the following RMSE equation. Plot these RMSEs with respect to the attributes using **matplotlib.pyplot**.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} \|y(i) - \hat{y}(i)\|^2}{N}},$$

N = Number of data points or samples in your dataset
yi = Actual observed values for the ith data point.
y^i = Predicted values for the ith data point

III. Outlier Detection: Outliers are the values that do not satisfy the following condition:

$$(Q_1 - (1.5 * IQR)) < x < (Q_3 + (1.5 * IQR)),$$

where x is the value of an attribute,

IQR is the interquartile range,

Q_1 and Q_3 are the first and third quartiles.

1. Detect the outliers from the data obtained after using the linear interpolation technique done in PartII - Q2. Using Boxplot (from **matplotlib.pyplot**), conclude if there are outliers in the data.

2. Once the outliers are detected, Replace them with the median of the attribute without using any in-built statistical function. Generate boxplots again and observe the difference with that of the boxplots in Part III - Q1. Do you still get outliers? Why?

IV. Standardization and Normalization:

1. Perform the Min-Max normalization of the outlier corrected data to scale the attribute values in the range 5 to 12. Find the minimum and maximum values before and after performing the Min-Max normalization of the attributes. Implement the following formula yourself (you can use in-built min, max functions):

$$\bar{x_i} = \frac{x_i - \min(x_i)}{max(x_i) - \min(x_i)} \cdot (upper - lower) + lower \;;$$

$where:$

x_i: i^{th} attribute

$min(x_i)$: minimum value of the i^{th} attribute

$max(x_i)$: maximum value of the i^{th} attribute

$lower$: lower limit of the range

$upper$: upper limit of the range

2. Find the mean and standard deviation of the attributes of the outlier corrected data. Standardize each attribute (independent variable) by implementing the formula given below. Compare the mean and standard deviations before and after the standardization (you can use in-built mean, std functions in **numpy**).

$\hat{x}_i = \frac{x_i - \mu}{\sigma}$;

$where:$

x_i: i^{th} attribute

μ: mean of the i^{th} attribute

σ: is the standard deviation of the i^{th} attribute

$\hat{x}_n$ is the standardized value of the i^{th} attribute