COMP 4448: Data Science Tools II Assignment 1

**Directions:** Do this assignment in Jupyter Notebook and provide screenshots of code and output in this word document wherever required. You will upload this word document containing screenshots of code and answers as well as your Jupyter Notebook to Canvas. All assignments will be submitted and graded through canvas and grades will be transferred to the 2U platform.

**Goal:** The goal of this assignment is to give you the opportunity to practice some key aspects of data preprocessing, which is a crucial step of the data science workflow.

**Packages:** Core packages required for this assignment are numpy, pandas and sklearn. You can import additional packages if necessary, but these mentioned packages should be enough.

**Points:** 50

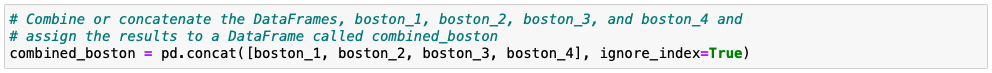
Question 1

1. Read the boston dataset csv files provided for this assignment into Python (you can use pd.read\_csv()). The boston datasets are **boston\_weather\_1, boston\_weather\_2, boston\_weather\_3, boston\_weather\_4, and more\_weather\_variables,** then assign the datasets to a DataFrame variable called **boston\_1, boston\_2, boston\_3, boston\_4, and more\_variables,** respectively. Combine or concatenate the DataFrames, **boston\_1, boston\_2, boston\_3, and boston\_4** and assign the results to a DataFrame called **combined\_boston.** These four datasets should be combined vertically since they have the same variable names, such that **boston\_1** is stacked on top of **boston\_2,** and the result is stacked on top of **boston\_3,** and the result is further stacked on top of **boston\_4.** Horizontally merge, join or concatenate the **combined\_boston** and **more\_variables** DataFrames and assign the results to a DataFrame called **boston\_data.** Print the first five rows of the **boston\_data,** and the last five rows of the **boston\_data**. Also print out the shape of the **boston\_data.**

Paste screenshots of your entire code and printed output here.

Text

Description automatically generated



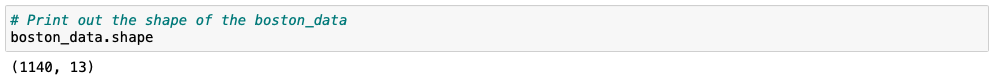


Table

Description automatically generated

Table

Description automatically generated with medium confidence



1. Check the boston\_data to verify how many missing data points exist under each column.

Paste a screenshot of your code and output here

Graphical user interface, text, application, chat or text message

Description automatically generated

1. Drop the rows or instances that contain any missing data. Assign the resulting DataFrame to a variable called **clean\_boston\_data.** Note that this is only one way of dealing with missing data and cases with missing data are usually used if you have sufficient sample size. Check for missing data again to ensure there is no missing data in the **clean\_boston\_data.** Print the shape of the **clean\_boston\_data**

Paste a screenshot of your code and output here

Graphical user interface, text, application

Description automatically generated

1. Format all the column names to lowercase and include underscore between column names that consist of two words. For example, meanTemp should become mean\_temp, and Max24hrPrep can become max\_24hr\_prep, and HighTemp becomes high\_temp, etc. Reassign the DataFrame with the formatted column names to the same variable, **clean\_boston\_data.** Print or output the columns of the **clean\_boston\_data** DataFrame.

Paste a screenshot of your code and output here

Text

Description automatically generated

1. Select or slice all data from the **clean\_boston\_data** DataFrame, except the data where the Year is 1930. You can call this subset data **excluding\_1930.** Using the **excluding\_1930** DataFrame, output the first 20 unique values in the Year column.

Paste a screenshot of your code and output here

A picture containing text

Description automatically generated

1. Select the data from the **clean\_boston\_data** where the Year is 1995 AND the high\_temp is greater than or equal to 90. Output or display the whole selected data. Here, you don’t have to assign it to any variable, but you could if you want to.

Paste a screenshot of your code and output here

A picture containing table

Description automatically generated

1. Select the data from the **clean\_boston\_data** where the Year is 1995 OR the high\_temp is greater than 89. Output or print the first 20 rows of the selected data. Here, you don’t have to assign it to any variable, but you could if you want to.

Paste a screenshot of your code and output here

Table

Description automatically generated with medium confidence

Question 2

1. Read the **student\_data** fileprovided into Python (take note of the file extension to use the appropriate pandas reader to read the data). Drop the first empty column in Python and assign the DataFrame to a variable **student\_data.**

Paste a screenshot of your code and output here

1. The **student\_data** shows the different midterm scores of students in math, reading and science, and their favorite ice cream flavors. Select the data in the **ice\_cream\_flavor** column and convert the flavors to a numpy array, then assign it to a variable called **flavor.** From the **student\_data,** select themath, reading and science scores all at once and convert the selected data to a numpy array and assign it to a variable called **scores**. Print the data in the **flavor** and **scores** arrays.

Paste a screenshot of your code and output here

1. Use the **scores** and **flavor** arrays to slice out the scores where the flavor is chocolate only.

Paste a screenshot of your code and output here

1. Use the **scores** and **flavor** arrays to slice out the scores where the flavor is chocolate OR vanilla.

Paste a screenshot of your code and output here

1. Use the **scores** and **flavor** arrays to slice out the scores where the flavor is not chocolate (you can use the “~ “sign).

Paste a screenshot of your code and output here

1. Slice out all math and reading scores where the flavor is chocolate, then compute the mean of math and reading scores for this subset (you can use the .mean() method on the array and specify the axis parameter value appropriately).

Paste a screenshot of your code and output here

Question 3

Imagine that you wanted to use the **student\_data** in question 2a to make predictions such that the ice\_cream\_flavor, math and reading columns are input variables and science column is the output variable you want to predict.

1. Use the LabelBinarizer() in the sklearn package to transform the ice\_cream\_flavor column in the **student\_data** to dummy variables, then join these dummy variables to the **student\_data** and drop the original ice\_cream\_flavor column. Reassign the resulting DataFrame to a variable called **student\_data\_1.** Printout the entire **student\_data\_1** DataFrame

Paste a screenshot of your code and output here

1. Extract the math, reading and science scores and use the StandardScaler() class in the sklearn.preprocessing module to standardize these scores, then merge the standardized scores to the dummy variables (you need to extract the dummy variables from student\_data\_1), and call the resulting DataFrame **student\_data\_std**. Printout the entire **student\_data\_std** DataFrame.

Paste a screenshot of your code and output here

1. Using a split ratio of 70:30, spit the **student\_data\_std** into training and test set. Reference the input and output of the training set as X\_train and y\_train respectively. Also reference the input and output of the test set as X\_test and y\_test respectively. Print X\_train, y\_train, X\_test and y\_test.

Paste a screenshot of your code and output here

1. Using the **boston\_1** DataFrame in question 1a, select the LowTemp, HighTemp, WarmestMin, ColdestHigh, AveMin, AveMax columns and use the “pipeline” functionality in sklearn to transform the selected data using the MinMaxScalar() and Imputer() classes. With the Imputer() class, the missing data in each column should be imputed using the mean value for the column. Assign the resulting DataFrame to a variable called **pipeline\_data.** Print the **pipeline\_data**

Paste a screenshot of your code and output here

1. Output the descriptive statistics of the **pipeline\_data** including mean, median, variance, minimum value, maximum value, variance, standard deviation and skewness. Your results should be in a single data frame and you can do this in a single line of code using .apply() or .agg() functionality of the pandas DataFrame.

Paste a screenshot of your code and output here