Homework Assignment 3

EU Cities Temperatures, German Credit, Google Playstore Apps

Worth 190 points.

Posted Friday, March 29

Due Monday, April 15 at 11 PM in Canvas

NO LATE SUBMISSIONS ACCEPTED

Everyone will work on the assignment in pairs. You DO NOT have the option of working alone.

For the visualization tasks in this assignment, you will use the `matplotlib` library. Since this is purely hands-on material for this assignment, and since it will not be on the exam, it will not be covered in lecture. However, here is a matplotlib Jupyter notebook (and HTML version) that goes over some of the often used basic matplotlib features. You will need to reference matplotlib documentation for tasks in this homework that require other features not covered in this basic notebook.

Make sure you abide by the DCS Acadmic Integrity Policy for Programming Assignments.

For the problems in this homework, write your code in Jupyter notebooks: **eucities_temps.ipynb**, **german_credit.ipynb**, and **google_apps.ipynb** respectively.

You need to make your code work ONLY on the given input files. We will test your submission on these same files, and no other.
However, you will still need to write general code for all tasks that would work on any set of values in the input within the specified constraints, not just the exact values that appear in the given input files.

In each notebook, write approproate markdowns for each task (just copy the task description from the assignment) so we know which task you are doing in the cells following the markdown.

Zip all of these notebooks into a single file named **hw3.zip** and submit this to Canvas. Do not include any input or output files, we only need your notebooks.
ONLY ONE SUBMISSON PER GROUP, PLEASE!

You are allowed up to 3 submissions, only the last submission will be graded.

How to Structure Your Notebooks

The way to structure a Jupyter notebook for the assignment is to simply write code for the tasks in a sequence of cells. Each (non-markdown) cell should be executable, and may contain any kind of code, whether it is just "global" code or a function, or a combination. (This is the sort of thing we have been doing all along in lectures and recitations.)

You can split a task in any number of sequential cells, but along the way, as we keep executing the cells one after the other, we should be able to monitor the progress for each task. Also, follow the sequence of tasks as assigned in the homework so that we can see how results are progressively updated as the tasks build on top of previous tasks.

You do NOT need to write a main function in a Jupiter notebook cell that calls functions you write in other cells.

For any of the problems in this assignment, you may only import and use modules we have covered in lecture including NumPy, Pandas, and matplotlib. You may NOT use any other module, even for the visualization tasks (e.g. you may not use seaborn, plotly, or any other visualization library instead of matplotlib).

Problem 1: EU Cities Temperatures Dataset (55 points)

Given a CSV data file as represented by the sample file EuCitiesTemperatures.csv (213 records), load it into a Pandas DataFrame and perform the following tasks on it.

Preprocessing/Analysis (28 pts)

1. [9 pts] Fill in the missing latitude and longitude values by calculating the average for that country. Round the average to 2 decimal places.

2. [9 pts] Find out the subset of cities that lie between latitudes 40 to 60 (both inclusive) and longitudes 15 to 30 (both inclusive). Find out which countries have the maximum number of cities in this geographical band. (More than one country could have the maximum number of values.)

3. [10 pts] Fill in the missing temperature values by the average temperature value of the similar region type. A region type would be a combinaton of whether it is in EU (yes/no) and whether it has a coastline (yes/no).

   For example, if we have a missing temperature value for Bergen, Norway, which is not in the EU but lies on the coast, we will fill it with the average temperature of cities with EU='no' and coastline='yes')

Visualization (27 pts)

For all plots, make sure to label the axes, and set appropriate tick labels.

1. [6 pts] Plot a bar chart for the number of cities belonging to each of the regions described in Preprocessing/Analysis #3 above.

2. [7 pts] Plot a scatter plot of latitude (y-axis) v/s longitude (x-axis) values to get a map-like visual of the cities under consideration. All the cities in the same country should have the same color.

3. [6 pts] The population column contains values unique to each country. So two cities of the same country will show the same population value. Plot a histogram of the number of countries belonging to each population group: split the population values into 5 bins (groups).

4. [8 pts] Plot subplots (2, 2), with proper titles, one each for the region types described in Preprocessing/Analysis #3 above.

   Each subplot should be a scatter plot of Latitude (y-axis) vs. City (x-axis), where the color of the plot points should be based on the temperature values: ‘red’ for temperatures above 10, ‘blue’ for temperatures below 6 and ‘orange for temperatures between 6 and 10 (both inclusive). For each subplot, set xticks to an array of numbers from 0 to n-1 (both inclusive), where n is the total number of cities in each region type. This represents each city as a number between 0 and n-1.

Problem 2: German Credit Dataset (72 points)

Given a CSV data file as represented by the sample file GermanCredit.csv (1000 records), load it into a Pandas DataFrame, and perform the following tasks on it.

Preprocessing (31 pts)

1. [8 pts] Drop the 3 columns that contribute the least to the dataset. These would be the columns with the highest number of non-zero 'none' values. **Break ties by going left to right in columns.** (Your code should be generalizable to drop `n` columns, but for the rest of the analysis, you can call your code for `n=3`.)
2. [4 pts] Certain values in some of the columns contain unnecessary apostrophes (‘). Remove the apostrophes.
3. [5 pts] The `checking_status` column has values in 4 categories: 'no checking', '<0', '0<=X<200', and '>=200'. Change these to 'No Checking', 'Low', 'Medium', and 'High' respectively.
4. [5 pts] The `savings_status` column has values in 4 categories: 'no known savings', '<100', '100<=X<500', '500<=X<1000', and '>=1000'. Change these to 'No Savings', 'Low', 'Medium', 'High', and 'High' respectively. (Yes, the last two are both 'High').
5. [4 pts] Change class column values from 'good' to '1' and 'bad' to '0'.
6. [5 pts] Change the employment column value 'unemployed' to 'Unemployed', and for the others, change to 'Amateur', 'Professional', 'Experienced' and 'Expert', depending on year range.

Analysis (17 pts)

For the following tasks, do preprocessing or changing of data types in the data frame as required.

1. [5 pts] Often we need to find correlations between categorical attributes, i.e. attributes that have values that fall in one of several categories, such as "yes"/"no" for attr1, or "low","medium","high" for attr2.

   One such correlation is to find counts in combinations of categorial values across attributes, as in how many instances are "yes" for attr1 and "low" for attr2. A good way to find such counts is to use the Pandas crosstab function. Do this for the following two counts.

   a. [3 pts] Get the count of each category of foreign workers (yes and no) for each class of credit (good and bad).

   b. [2 pts] Similarly, get the count of each category of employment for each category of saving_status.
2. [4 pts] Find the average credit_amount of single males that have 4<=X<7 years of employment. You can leave the raw result as is, no need for rounding.
3. [4 pts] Find the average credit duration for each of the job types. You can leave the raw result as is, no need for rounding.
4. [4 pts] For the purpose 'education', what is the most common checking_status and savings_status? Your code should print:

   ```
   Most common checking status: ...
   Most common savings status: ...
   ```

Visualization (24 pts)

1. [9 pts] Plot subplots of two bar charts: one for savings_status (x-axis) and the other for checking_status (x-axis). In each chart, the y-axis represents number of people. Moreover, for each category of saving_status (checking_status), we need you to display four bars, each

corresponding to one of the "personal_status" categories. Each personal status category bar should be of a different color.

2. [9 pts] For people having credit_amount more than 4000, plot a bar graph which maps property_magnitude (x-axis) to the average customer age for that magnitude (y-axis).

3. [6 pts] For people with a "High" savings_status and age above 40, use subplots to plot the following pie charts:
   a. Personal status
   b. Credit history
   c. Job

Problem 3: Google Playstore Apps Dataset (63 points)

Given an Excel data file as represented by the sample file GooglePlaystore.xlsx (10K records), load it into a Pandas DataFrame (use the Pandas `read_excel` method), and perform the following tasks on it.

Preprocessing (28 pts)

1. [3 pts] Often there are outliers which do not match the overall data type. There is one record in this data where the "Reviews" has value "3.0M" which does not match the rest of the data. Remove that record.

2. [4 pts] Remove rows where **any** of the columns has the value "Varies with device".

3. [5 pts] The values in the Android version column should be floats. Strip the trailing non-numeric characters from all values (ie. the words " and up"), so the result is a number. If there are multiple decimal places (eg. "x.y.z"), keep only the first two parts (eg "x.y"). For example, the value "4.1 and up" should be changed to "4.1". The value "4.5.6 and up" should be changed to "4.5". The value "5.6.7" should be changed to "5.6".

   If there is a range (eg. 5.0 - 8.0), only consider the first number. For example, the value "5.0 - 8.0" should be changed to "5.0". The value "4.0.3 - 7.1.1" should be changed to "4.0".

4. [5 pts] The "Installs" column must have integer values. For values that have commas, remove the commas. For values that have a '+' at the end, remove the '+'. Keep only those rows that have an integer value after these edits.

5. [5 pts] For missing rating values, if the number of reviews is less than 100 and installations is less than 50000, remove the row. Else, fill the missing value with the average value (rounded to 2 decimal places) for the Category of that row.

6. [6 pts] Preprocess the Size column to convert the "M" (millions) and "K" (thousands) values into integers. For instance, 8.7M should be converted to 8700000 and 2.4K should be converted to 2400.

Analysis (19 pts)

For the following tasks, do preprocessing or changing of data types in the data frame as required.

1. [4 pts] Describe (use DataFrame `describe` method) the category wise rating statistics. In other words, for each category, describe the statistics (count, mean, etc.) for ratings in that category.

2. [11 pts] Extract all "Free" apps from the master data frame. Then write a function that, given a numeric column e.g 'Rating'), will create and return a dataframe for the top 3 free applications in each category based on that column. Call the function on each of these columns:

   a. Rating (gives top 3 most highly rated applications in each category)

b. Installs (gives top 3 most installed applications in each category)

c. Reviews (gives top 3 most reviewed applications in each category)

You don't need to do anything explicit to break ties.

Each of the returned dataframes have Category and App for the first two columns, and one of Rating (for a.), Installs (for b.), and Reviews (for c.) as the third column, as for instance:

3. [4 pts] Find the average, maximum and minimum price of the paid applications.

Visualization (16 pts)

1. [9 pts] In the genre column, break the string of genres into a list. For example, ‘Art & Design; Creativity’ should be [‘Art & Design’, Creativity’].

   Count the number of applications per genre and display it using a pie chart.
   Hint: Read about `DataFrame.explode()`

2. [7 pts] Display a box plot of ratings for "Business" and "Education" categories. The boxplots should be in the same plot.