HW 1

Background

The main goal of this assignment is to enable you to be comfortable with the Google colaboratory programming environment (`https://colab.research.google.com/`), which you will be extensively using as part of your assignments. The assignment helps you to set up a basic Python environment on the Google Cloud, access your drive for data/content, load up your datasets using Python libraries and then run simple data analysis exercises. Note that Google colaboratory is free to use (as long as you are using default settings).

The maximum number of points for this assignment is 105 (including bonus of 5 points). You can choose to answer the bonus question – but it is not necessary. You have a week to submit from Thursday (Oct 3, 2019) – which means your assignment is due on Oct 10, 2019 (11.59 PM Central Time).

What to expect?

You will be expected to set up your own Python environment and provide documentation on whether you were successful. Further, you will also run some test codes on existing datasets and test your machine learning models out within the Google colaboratory environment. You will also run your experiment on a graphics processing unit (GPU) assigned by Google and try out a deep learning exercise meant to help you code your homework. Note that as part of your home work, you are not expected to write very long pieces of code – most of them can be done with no more than 100-150 lines of code.

What to hand in?

You are expected to hand in a Google colaborator notebook (usually with the extension: `.ipynb`) as a link from your Google Drive so that we can run your home work. Within your Python notebook, you can easily document your results, visualize plots, and also add comments using Markdown (`https://en.wikipedia.org/wiki/Markdown`). If you are not familiar with Markdown, you can easily learn it from the link above.

Downloading and exploring the dataset [25 points]

We will be playing with a simple dataset, namely P1_data.csv, that consists of a 250×10 matrix, where the rows correspond to individual patients, and the columns correspond to genes. You can directly load the dataset from this location on Google Drive `https://drive.google.com/drive/folders/10LF_xE2qcyA5lW6G-tM7nOBpoqA2PyXQ?usp=sharing_eip&invite=COv2u5kM&ts=5d922b4e` (or if you are working on your personal laptop, you can download it to a local folder). Open it using any text editor, and you should be able to see the contents of the file. The header row in this matrix explains the different patient IDs and the genes that we will be looking at.

Question 1: What are the mean, median and standard deviations for each of the columns in the data? [10 points]

One of the first aspects of working with any dataset is to understand whether the data are distributed 'appropriately'. For example, you want to first explore if there are any outliers (i.e., values that are either too large or too small compared to other values in the data), and understand the basic statistical attributes of your dataset. To go about doing this, we need to extract, for each column, the mean, median and standard deviations from this data.

Using the `pandas` library in Python, create a data frame that allows you to load the downloaded dataset. The `pandas` library allows you to easily create data frames, query and report these results. You can learn more about the basics of the data handling and exploration with `pandas` here: `http://wavedatalab.github.io/datawithpython/index.html`.

Calculate the mean, median and standard deviations and report your results as a table. Do we see any outliers in the data with respect to the genes that we have considered?

Question 2: Visualize the distribution from your data frame [10 points]

One way to explore the data frame is to create and visualize the data as histograms. Since we have only about 250 individual patients in the dataframe, creating histograms with small datasets can be often challenging. You are going to try and visualize the histograms in the dataset per column (gene) so that you can get a sense of how the data is distributed.

First, go through the histogram function in Python `numpy` library (`https://docs.scipy.org/doc/numpy/reference/generated/numpy.histogram.html`). We recommend that you use a bin size of 11. You may also choose to normalize the data – however, that is not needed here since we really want to understand how the raw data is distributed. The histogram function in python returns two arrays: (1) frequency (how often a value occurs in the data) and (2) bins (how we have defined the ranges in the data).

Once you obtain the histogram, you can plot the same with the `matplotlib.pyplot` library. For each of the columns (genes), can you create a function that will automatically output the histogram? From the histogram, can you reason about the distribution of the dataset itself? How many of the genes have outliers?

Question 3: Visualizing the entire dataset as a contour map [5 points]

Now that we know how our data is distributed, we want to visualize the data as a contour map. Use the `matplotlib.pyplot` library to do this. The library has a number of examples for you to understand how to use the contour map function. Use the default colormap so that you can visualize the results. Tutorial link: `https://matplotlib.org/3.1.1/gallery/images_contours_and_fields/`

`contour_demo.html#sphx-glr-gallery-images-contours-and-fields-contour-demo-py`.

Simple clustering of the dataset [25 points]

Now that we have visualized the data, let's try to cluster the data. Clustering is a way of analyzing the data to group the 250 patients based on similar gene expression patterns. There are many ways to cluster a given dataset. Let's assume that we will use k-means clustering as the algorithm of choice.

Question 4: Selecting an initial choice of k [5 points]

Based on the contour map that we visualized above, can you come up with a reasonable choice of k for our clustering algorithm? *Hint:* $1 < k \leq 11$. Note that there is no wrong answer here – this is based on your best initial guess.

Question 5: Setting up and running k-means clustering [20 points]

Now, let's look at an automated approach to see if we can cluster the data – and perhaps, check if your initial guess is right. The key function that you need is the k-means: `https://scikit-learn.org/stable/modules/clustering.html#k-means` and the associated tutorial: `https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_assumptions.html#sphx-glr-auto-exampl`

Use the k-means algorithm to cluster your data. You will vary the `n_clusters` setting from $\{3, \ldots, 10\}$, and check how the clustering looks like. Now you can use the function `fit()` to run it through these settings. Say you set $k = 5$, can you summarize for the 10 genes the mean values in each cluster?

In your response to the question, please include:

1. The number of phenotypes (i.e., k) that you discovered with the k-means algorithm.
2. Description of a rationale for how you decided to use k. Note that as we continue with the class, we will find more quantitative ways to decide on k (which is referred to as model selection).
3. The table that you created above for the k that you think is most appropriate.

Turning to Deep Learning [50 points]

We are not going to implement a deep learning algorithm here. Our goal however, is to be able to use a pre-trained deep learning model and test out a few things that we can do with it. The example that we are going to use is based on brain imaging data.

Question 6: Access the workbook and install the programming environment [10 points]

Access the HW1-DLTK-test.ipynb notebook on the shared Google drive. Your first task is to ensure that you can set up the runtime environment such that you can get a GPU that you can access. The GPU you get can be either a Tesla T4 or a Kepler K80 card. Don't worry about getting a specific GPU type, you can work with either of the two cards and still get your results.

The first five cells of the workbook allow you to set up the programming environment and create the necessary folders that are needed for your work. Make sure you execute them.

The next three cells of the workbook set up the necessary libraries for processing MRI images. We are going to be using the Deep Learning Toolkit (DLTK) based on Tensorflow to build our models. Make sure you execute these cells and end up without any errors.

Question 7: Data pre-processing [15 points]

Your task is to work with the 13 cells in this section. Obviously this section has a lot of code, however, much of it is for you to understand how to write good code that can be reused (perhaps across assignments).

In the subsection, "loading data into memory", we have a function that loads a subset of the MRI data into memory. What we want to experiment here is if we increase the variable `num_datasets`, will the time taken to load the data also increase? Check by varying `num_datasets` = $10, 50, 100$ and rerun the cell for "loading data into memory". Report your results as a table that summarizes what happens when you increase the number of datasets to be loaded.

Bonus question [5 points]: if you are able to change the runtime and run the experiment with a different GPU card, (say you were running using K80, but can check with the T4 card), does the data loading times change? Report your results as a table that summarizes what happens when you increase the number of datasets to be loaded.

Question 8: Playing with a pre-trained model [25 points]

The next part of this assignment is going to deal with a toy problem where in we will use a pre-trained model and see if we can train it. We will be using the Tensorflow Estimator API and setting up a number of parameters. Your task here will be to run each of the cells.

The next subsection on "building a model function" will have 4 cells, of which one cell is particularly useful for training our model. We have set the number of steps for training to be 200. Change the number of steps to $250, 300, 400$. Does it change the overall cross-entropy loss? Report on what the final values of the cross-entropy loss are (after 250, 300 and 400 steps). Note that you need to modify the code to get this done.