CS0011 Introduction to Computing for Scientists
Project 2
Lab 1

Overview

In this project, we will explore some gene expression data from bone marrow samples of leukemia patients and use these data to classify the samples according to the type of leukemia. We will be retracing some of the work published in a 1999 paper by T.R. Golub et al. [GST+99].

We give a brief overview of the setting of the paper here. You can refer to [GST+99] for more details. A major challenge in cancer treatment is precise diagnosis. For different types of cancer, different treatments may be more or less effective. In some cases, two types of cancer may have very similar appearances but may require very different treatments. Examples of two such types of cancer are acute lymphoblastic leukemia (ALL) and acute myeloid leukemia (AML). ALL starts from lymphocytes, white blood cells that play an important role in the immune system. AML, on the other hand, starts from myeloid cells, precursors to many other types of cell in the blood. At the time that [GST+99] was published, treatments for ALL and AML were very different, so they wrote that "distinguishing ALL from AML is critical for successful treatment."

Figure 1: An Affymetrix microarray. ©User:Schutz / Wikimedia Commons / CC-BY-SA-3.0 / GFDL / Cropped from original.

In their paper, [GST+99] demonstrate that gene expression data can be used to accurately distinguish between AML and ALL. Specifically, they consider microarray data. Microarrays measure how much genes are expressed. In particular, they measure how RNA has been transcribed for each gene. A microarray is a grid to which small pieces of DNA are attached. Each piece of DNA corresponds to a particular gene. RNA from a sample is labeled with fluorescent molecules and then put on the microarray. RNA corresponding to the DNA at a particular location on the microarray with bind to it. The amount of RNA at each spot is quantified using the fluorescent labeling: more RNA corresponding to a particular piece of DNA results in a brighter spot at that point on the array. See the figure **??** for a picture of a microarray. (This is not the same model as the one used in the [GST+99] paper.)

Lab 1: Preliminaries

Part 1: Reading files

The data we will be looking at should be stored in *TSV* format (tab-separated values). Recall from Project 1 that in CSV format, values are separated by commas. In TSV, values are separated by tab characters. Tab characters are represented in Python using the escape character `"\t"`.

You can download the data files you will need for this lab here:
http://people.cs.pitt.edu/~nlf4/cs001X/data/data_set_ALL_AML_train.txt
http://people.cs.pitt.edu/~nlf4/cs001X/data/table_ALL_AML_samples.txt

1. Write a function named `read_lines`. This function should:

   - Take a single argument that is a string. This argument should be the filename of the tab-delimited file you would like to read.
   - Print out the contents of each line of the file. Each line in the file should contain a row of the data: the gene description, gene accession number, and 38 alternating integers and strings corresponding to the 38 samples from which the gene expression data was collected. Here is a single example row:

     ```
     Osteomodulin AB000114_at 72 A 21 A -27 A 61 A 16 A 85 A
     -10 A 25 A -38 A 65 A -111 A 12 A 39 A 68 A 42 A 18
     A 97 A 49 A 41 A 83 A 28 A 26 A 174 P -34 A 154 A 45
     A 87 A 36 A 0 A -35 A 36 A -17 A 39 A 1 A 1 A -23 A
     56 A -27 A
     ```

2. Write a function named `main` that takes no arguments. Your `main` function should first call `read_lines` to process the file `data_set_ALL_AML_train.txt`. It should then call `read_lines` again to process the file `table_ALL_AML_samples.txt`.

3. Call `main` at the bottom of your program.

4. Be sure to define all functions in global scope, and make liberal use of comments to clarify the code that you are writing. You will be graded on the readability of your code (including commenting!).

Part 2: Verifying the formatting of the data

As mentioned previously, the data in `data_set_ALL_AML_train.txt` and `table_ALL_AML_samples.txt` *should* be formatted as tab-separated. However, as is unfortunately often the case with data repositories, the data you have to work with is actually not well formatted. To get a clearer view of how the data is formatted, we should look at it character by character.

Modify `read_lines` to, for each line, insert an `"!"` after each character. This will give you a clear picture of how the file is actually formatted.

Note that this will produce **alot** of output. The last few lines of `table_ALL_AML_samples.txt` should be printed as follows (note that some of the lines overflow the PDF here and are truncated):

```
Line75:
7!1! !  !       !A!L!L! !       !P!B!   !B!-!c!e!l!l!   !       !       !       !0!7!/!1!8!/!9
!

Line76:
7!2! !  !       !A!L!L! !       !P!B!   !B!-!c!e!l!l!   !       !       !       !0!7!/!2!8!/!9
!

Line77:

!

Line78:

!

Line79:

!

Line80:

!

Line81:

!

Line82:

!

Line83:

!
```

Scroll through the data printed in your terminal to see if the data is well formatted. Is there any mix of tabs and spaces? Are all of the data items separated by a single tab? How would you plan to try and parse this file?

Once you have completed all parts of the lab, be sure to show your work to the lab instructor.

References

[GST+99] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286(5439):531–537, 1999. (document)