CS0011 Introduction to Computing for Scientists
Project 2
Lab 3

Background: classification

For this final lab of Project 2, you will be building a very basic *classifier*. The goal is that, given a sample of gene expression data, your classifier will be able to determine if that sample is of an AML leukemia, or ALL.

Generally, using a classifier will proceed in 3 phases:

Feature Selection First, *feature selection* is performed. The goal of feature selection is to determine what features of the data will prove useful in classification. In our case, this would be determining which genes would be most useful in trying to differentiate ALL and AML samples. While [GST+99] describes how automated feature selection can be done, to keep things (relatively) simple for this lab, you will not need to program a feature selection step. We have already determined that the following genes (listed by GAN) should be considered useful features in differentiating ALL and AML samples:

- M55150_at
- U50136_rna1_at
- X95735_at
- U22376_cds2_s_at
- M16038_at
- M23197_at
- M84526_at
- U66497_at
- U82759_at
- D49950_at
- X59417_at
- M27891_at
- X17042_at
- U05259_rna1_at
- U12471_cds1_at

- U46751_at
- M92287_at
- Y00787_s_at
- L08246_at
- X74262_at

Training For the second phase, you will need to *train* the classifier using a dataset and the features that you identified as useful during feature selection. For this lab, we will be building a relatively simple classifier that looks at the average expression values for each relevant gene across all ALL samples and all AML samples. We will be using `ALL_AML_training.tsv` as our training dataset.

Testing The final phase will use the trained classifier to classify another dataset. We will be using `ALL_AML_testing.tsv` as our testing dataset. For each sample in `ALL_AML_testing.tsv`, your program should use the trained classifier to guess if it is an ALL or AML sample. Since we know the type of leukemia each sample was drawn from, you will be able to verify the results of your classifier.

Lab 3

Part 1: Training

You should write a function named `train` that takes in two arguments:

- The first argument should be named `labels`, and should be a list of lists containing the diagnosis of each sample.
- The second argument should be named `data`, and should be a list of lists containing the gene expression measurements of a set of samples.

Your `train` function should create a return a dictionary of dictionaries. This dictionary should map each of the 20 relevant genes listed above to a nested dictionary containing its average expression value for ALL samples and its average expression value for all AML samples. Each nested dictionary should have two keys: `"ALL"` and `"AML"`. Printing the dictionary of dictionaries generated for `ALL_AML_training.tsv` to the terminal should produce the following output:

```
{'D49950_at': {'ALL': 75.37037037037037, 'AML': 243.9090909090909},
'L08246_at': {'ALL': 1066.5555555555557, 'AML': 3767.4545454545455},
'M16038_at': {'ALL': 375.3703703703704, 'AML': 1811.6363636363637},
'M23197_at': {'ALL': 175.03703703703704, 'AML': 767.2727272727273},
'M27891_at': {'ALL': 144.44444444444446, 'AML': 7423.545454545455},
'M55150_at': {'ALL': 810.2962962962963, 'AML': 1836.2727272727273},
'M84526_at': {'ALL': -177.40740740740742, 'AML': 3483.0}, 'M92287_at':
{'ALL': 4029.4814814814813, 'AML': 1073.6363636363637}, 'U05259_rna1_at':
{'ALL': 3562.925925925926, 'AML': 288.1818181818182}, 'U12471_cds1_at':
{'ALL': 152.22222222222223, 'AML': 287.1818181818182}, 'U46751_at': {'ALL':
```

```
1512.148148148148, 'AML': 5291.272727272727}, 'U50136_rna1_at': {'ALL':
977.7777777777778, 'AML': 2562.181818181818}, 'U66497_at': {'ALL':
58.81481481481482, 'AML': 57.45454545454545}, 'U82759_at': {'ALL':
241.37037037037038, 'AML': 850.7272727272727}, 'X17042_at': {'ALL':
1642.8148148148148, 'AML': 7108.727272727273}, 'X59417_at': {'ALL':
4361.37037037037, 'AML': 1138.1818181818182}, 'X74262_at': {'ALL':
1416.962962962963, 'AML': 255.72727272727272}, 'X95735_at': {'ALL':
349.8888888888889, 'AML': 3023.6363636363635}, 'U22376_cds2_s_at': {'ALL':
3863.296296296296, 'AML': 702.3636363636364}, 'Y00787_s_at': {'ALL':
892.2222222222222, 'AML': 10496.90909090909}}
```

Note the large difference between the expression values for AML and ALL samples for each gene. This is the reason that these genes were selected as our features.

Part 2: Testing

You should write a function named `classify` that takes in two arguments:

- The first argument should be named `averages` and should be a dictionary of dictionaries that maps each of the 20 relevant gene GANs to a nested dictionary containing its average expression values for ALL and AML samples (i.e., the data structure returned by `train`).
- The second argument should be named `data`, and should be a list of lists containing the gene expression measurements of a set of samples.

Your `classify` function should return a dictionary that maps each sample ID in `data` to a string value indicating its classification (i.e., the string values `"ALL"`, `"AML"`, or `"Undetermined"` if the classifier determines it to be ALL, AML, or is unable to make a determination).

`classify` should determine if a sample is ALL or AML by, for each of the 20 relevant genes, determining whether the value from that sample is closer to ALL average or closer to the AML average. If a sample has more reading that are closer to the ALL average, it should be classified as ALL. If a sample has more readings that are closer to the AML average, it should be classified as AML. If a sample has an equal number of readings that are closer to AML and closer to ALL, the result should be Undetermined. For example, consider a sample with the following gene expression values:

```
D49950_at: 80
L08246_at: 900
M16038_at: 400
M23197_at: 300
M27891_at: 200
M55150_at: 1000
M84526_at: 0
M92287_at: 4000
U05259_rna1_at: 3000
U12471_cds1_at: 150
U46751_at: 2000
```

```
U50136_rna1_at: 1000
U66497_at: 56
U82759_at: 900
X17042_at: 5000
X59417_at: 1400
X74262_at: 300
X95735_at: 2500
U22376_cds2_s_at: 800
Y00787_s_at: 8000
```

This sample should be classified as ALL using the training data from `ALL_AML_training.tsv` because 12 of these gene expression values are closer to the ALL averages for those genes, and 8 are closer to the AML averages.

Part 3: Verification

Finally, you should write a function named `verify` that takes in two arguments:

- The first argument should be named `results` and should be a dictionary that maps sample IDs to a string values indicating their classifications (i.e., the data structure returned by `classify`)
- The first argument should be named `labels`, and should be a list of lists containing the diagnosis of each sample.

`verify` should, for each sample in `results`, print out the classifier's determination, as well as the actual diagnosis for that sample. Further, it should, after printing out the results for each sample, print out the overall accuracy of the classifier.

At this point, you should ensure that your main function does the following:

1. Assign the variable name `labels` to the result of calling `read_label_tsv` for `ALL_AML_labels.tsv`
2. Assign the variable name `training_data` to the result of calling `read_data_tsv` for `ALL_AML_training.tsv`
3. Assign the variable name `testing_data` to the result of calling `read_data_tsv` for `ALL_AML_testing.tsv`.
4. Assign the variable name `training_results` to the result of calling `train` for `labels` and `testing_data`.
5. Assign the variable name `testing_results` to the result of calling `classify` for `training_results` and `training_data`.
6. Call `verify` with `testing_results` and `labels` as arguments.

Here is the output from a run of the desired program:

```
Sample #39:
    Classification: ALL
    Actually: ALL

Sample #40:
    Classification: ALL
    Actually: ALL

Sample #42:
    Classification: ALL
    Actually: ALL

Sample #47:
    Classification: ALL
    Actually: ALL

Sample #48:
    Classification: ALL
    Actually: ALL

Sample #49:
    Classification: ALL
    Actually: ALL

Sample #41:
    Classification: ALL
    Actually: ALL

Sample #43:
    Classification: ALL
    Actually: ALL

Sample #44:
    Classification: ALL
    Actually: ALL

Sample #45:
    Classification: ALL
    Actually: ALL

Sample #46:
    Classification: ALL
    Actually: ALL

Sample #70:
```

```
    Classification: ALL
    Actually: ALL

Sample #71:
    Classification: ALL
    Actually: ALL

Sample #72:
    Classification: ALL
    Actually: ALL

Sample #68:
    Classification: ALL
    Actually: ALL

Sample #69:
    Classification: ALL
    Actually: ALL

Sample #67:
    Classification: AML
    Actually: ALL

Sample #55:
    Classification: ALL
    Actually: ALL

Sample #56:
    Classification: ALL
    Actually: ALL

Sample #59:
    Classification: ALL
    Actually: ALL

Sample #52:
    Classification: AML
    Actually: AML

Sample #53:
    Classification: AML
    Actually: AML

Sample #51:
    Classification: AML
```

```
    Actually: AML

Sample #50:
    Classification: AML
    Actually: AML

Sample #54:
    Classification: AML
    Actually: AML

Sample #57:
    Classification: ALL
    Actually: AML

Sample #58:
    Classification: AML
    Actually: AML

Sample #60:
    Classification: ALL
    Actually: AML

Sample #61:
    Classification: AML
    Actually: AML

Sample #65:
    Classification: AML
    Actually: AML

Sample #66:
    Classification: ALL
    Actually: AML

Sample #63:
    Classification: AML
    Actually: AML

Sample #64:
    Classification: AML
    Actually: AML

Sample #62:
    Classification: AML
    Actually: AML
```

```
88.24% correct (30/34)
```

Rubric

Your program will be evaluated according to the following rubric:

`read_data_tsv` works as specified	10
`read_label_tsv` works as specified	10
`train` works as specified	15
`classify` works as specified	30
`verify` works as specified	15
All functions are defined in global scope	10
All other code appears inside of a function (aside from a call to `main` and any magic value definitions)	10

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)