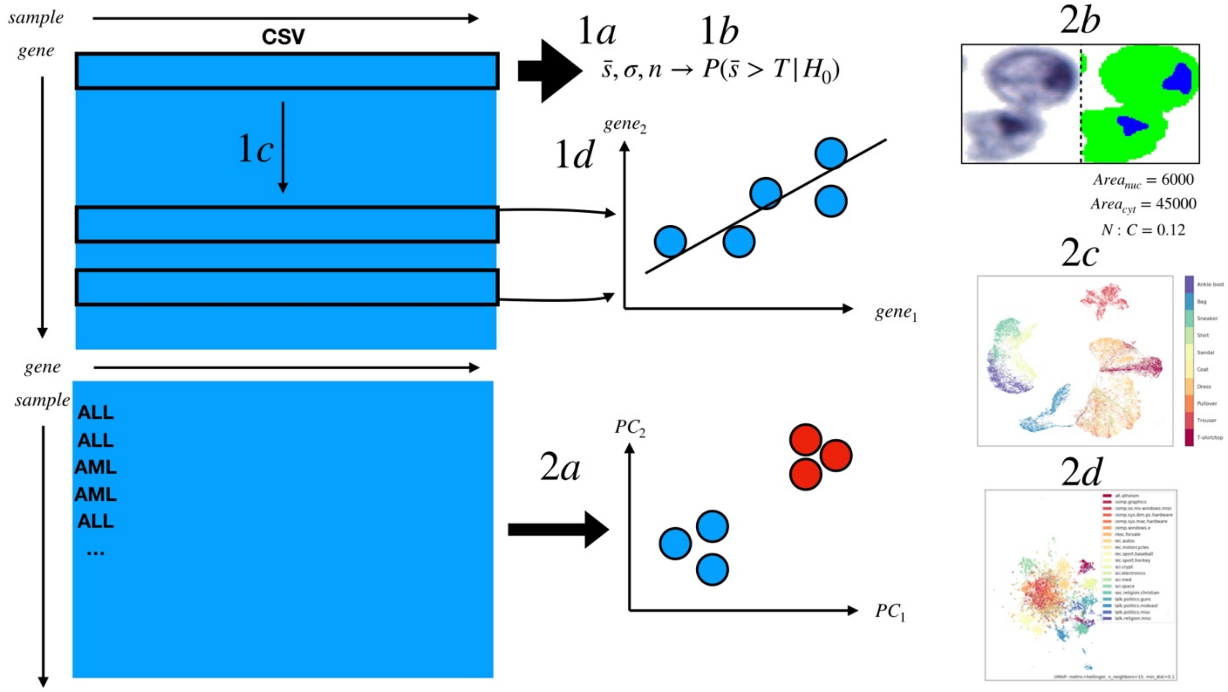
Technical Interview Questions:



Please try to answer some of the following questions to the best of your ability while explaining all of your logic. It is okay if you do not know the answer to these questions, feel free to email Joshua for hints if you are stuck. If you are unable to answer the question, please explain how you would go about solving the question. Popular tools for displaying answers to these questions include jupyter notebook and Rstudio, though any printout is fine as long as you can walk us through it.

We have included test data that matches each question number and letter in the zipped file attached to this email:

1. Programming/Statistics:
    a. Functional programming:
       Write a python/R function that returns the mean, standard deviation and number of elements of a python/R list/vector. As an example, you can utilize this python list: [5.99342831, 4.7234714 , 6.29537708, 8.04605971, 4.53169325, 4.53172609, 8.15842563, 6.53486946, 4.06105123, 6.08512009].
    b. Statistics:
       Given a float array (use the one from the previous problem) and a supplied “threshold” value, using the function from the functional programming problem, devise a one sample z-test that tests to see whether the mean value of the sample exceeds the supplied threshold value (you can use scipy or the package pingouin for help). Assume that the sample level variance matches the population-level variance (feel free to conduct a t-test if you want to violate this assumption). Suppose the threshold here is 4 if using the previous dataset. The analogy here in bioinformatics is that imagine that as a subcomponent of some

experiment, we want to see whether the expression of certain genes surpasses a threshold that carries some previously established diagnostic or prognostic significance. Let’s call a gene with mean expression greater than the threshold a “bad” gene. Be prepared to speak to what the p-value means in the context of your test.

   c. Object Oriented Programming:
      We’re going to generate a python class that is able to perform these tests across many such “genes”. Generate a python/R data class that loads a csv file in its init method. The csv file will be comprised of a header line containing patient names, then each subsequent line will first start with a gene name, followed by “expression” values for each of the patients (this is a gene-by-sample matrix). We want to see if each of the genes is a “good/bad” gene using the previously established threshold (this is an unrealistic situation, but designed this way for simplicity). Define two methods for this class, one that re-implements your hypothesis test function from the previous section, and the other that calls this method on each of the lines of the csv, outputting a list of p-values and some indication of whether the gene was “good” or “bad”. We’ll use the same expression threshold of 4.
   d. Plotting:
      We think that some of the expression of these genes may be correlated, but want to plot a scatter plot of the two genes expression values across the cohort to see if this is the case (each axis is each gene’s expression, each point is an individual). Please write a function or class method that takes as input two lists of floats, or two of the gene lists from the previous problem, and outputs a scatterplot of the two gene’s expression across the cohort. Libraries like matplotlib, seaborn in python and ggplot in R are good for this task.

2. <u>Bonus Questions</u> (Going above and beyond, no need to do these; maybe select one if you are interested):
   a. Machine learning:
      We are studying a population of patients with leukemia and want to understand whether their gene expression profiles are able to determine whether their disease is of the myeloid or lymphoid lineage. We are given a csv of samples-by-genes. First, we want to visualize how the samples cluster into their respective disease subtypes (subtypes are denoted by ALL and AML, first value in each row). Using pandas, scikit-learn and matplotlib, first run a PCA/TSNE on the expression profiles to generate a 2D scatter plot, where each point is a patient, colored by AML/ALL. Then, using scikit-learn, implement a clustering and/or classification algorithm to delineate the disease subtypes and report some measure of concordance/accuracy. For a classification task, make sure the metrics are reported on a manually generated validation set.
   b. Image Analysis:
      First, clone/download this repository:
      https://github.com/jlevy44/PreliminaryGenerativeHistoPath . In this repository, you’ll find a collection of images of urine cells. Each image can be divided half;

the left half contains the actual image of the cell. The right half contains a segmentation mask containing where the nucleus (blue), cytoplasm (green) and background (white) are. When the nucleus is large compared to the cytoplasm, we consider the cell to be malignant / potentially cancerous. Select a subset/all of these images, and calculate the nucleus:cytoplasm (NC) ratio (# pixels nucleus / (# pixels nucleus + # pixels cytoplasm)) for each mask on the right.

   i. How malignant do you think the cell population is? First, generate a hypothesis by stating some “threshold” NC ratio (eg. if the average NC ratio is above 0.3, the population is likely cancerous). Then, test your hypothesis using any of the previously developed statistics functions from above.

   ii. Bonus: Can you devise a method to learn to segment the nucleus and cytoplasm on the left half of the images? If so, see how well it correlates to the NC ratios you calculated on the right.

c. Deep learning / Pytorch for Image Classification:
Using pytorch, keras or tensorflow, try to build an image classification pipeline to classify Fashion MNIST: https://pytorch.org/vision/0.8/datasets.html#fashion-mnist. Here is an example of pipeline that accepts as input data of the ImageFolder class, amongst others: https://github.com/jlevy44/PathPretrain. Please try to visualize the training and validation losses as well as evaluate common classification metrics on the test set, additional visualizations (eg. PCA/UMAP) welcome!

d. Natural Language Processing: Please follow this tutorial and see if you can explain each of the steps used to process the data: https://umap-learn.readthedocs.io/en/latest/document_embedding.html

Few Select Studies for Review (can send more upon request)

1) Reviews
   a) https://www.advancesinmolecularpathology.com/article/S2589-4080(21)00013-2/pdf
   b) https://www.nature.com/articles/s41591-021-01343-4?proof=t%C2%A0
   c) https://www.frontiersin.org/articles/10.3389/fmed.2019.00264/full
   d) https://www.sciencedirect.com/science/article/pii/S2001037017300867
   e) https://www.nature.com/articles/nature14539
   f) https://academic.oup.com/ajcp/advance-article/doi/10.1093/ajcp/aqab085/6327583?login=true
   g) https://arxiv.org/pdf/2107.00272.pdf
2) Select Lab Papers:
   a) https://www.nature.com/articles/s41379-020-00718-1
   b) https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7959046/
   c) http://psb.stanford.edu/psb-online/proceedings/psb22/levy.pdf
   d) https://www.medrxiv.org/content/10.1101/2021.03.13.21253502v1
   e) https://onlinelibrary.wiley.com/doi/10.1002/cncy.22099
   f) https://pubmed.ncbi.nlm.nih.gov/31797614/
   g) https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-020-3443-8
   h) https://www.nature.com/articles/s41540-021-00193-7

i) https://pubmed.ncbi.nlm.nih.gov/31368477/

j) https://www.biorxiv.org/content/10.1101/2021.08.17.456726v3.full.pdf

k) https://www.biorxiv.org/content/10.1101/2021.10.27.466144v2

l) https://www.biorxiv.org/content/10.1101/2021.10.30.466613v1

m) https://bmcmedresmethodol.biomedcentral.com/articles/10.1186/s12874-020-01046-3

n) https://www.frontiersin.org/articles/10.3389/fpubh.2021.766707/full