BMI 776

HOMEWORK 3 (3 LATE DAYS - LATEX IDE IS IN A DIFFERENT TIME ZONE)

Noah Cohen Kalafut
Computer Science Doctoral Student
University of Wisconsin-Madison
nkalafut@wisc.edu

April 21, 2021

1 Problem 1

1.1 Part A

File can be found under find_paths.py. All example outputs match exactly.

1.2 Part B

Firstly, we can notice that the shortest path ($2-4-8$, Cost 7) is followed in our flow solution. Similarly, the paths ($3-5-7-9$, Cost 9) and ($1-6-8$, Cost 10) were also utilized. A flow solution can always be decomposed like this. In more complex graphs, however, this may become fairly intensive.

By the nature of the flow problem, we need to provide the additional variables of capacity and demand, which changes the formulation of the problem. Additionally, to solve in the minimal cost flow problem, a directed graph needs to be used while k-shortest paths can use any type of graph. These inherent differences also mean that k-shortest paths will return multiple possibilities while the flow problem will only return one.

In general, a flow solution will give us a more complete solution to our problem while k shortest paths will give us the building blocks and most likely avenues for that solution. As we add more targets and/or sources, this property remains true but k shortest paths results become less interprable due to a higher k being needed to show all paths used in the flow solution. The complexity increases the likelihood of greater-cost paths being utilized due to capacity constraints.

1.3 Part C

1.3.1 Discussion

Given exclusively negative demand $-f$ on our source node and inverted demand on our target node, if the capacity C_e for all edges $e \in E$ satisfies $C_e \geq f$ and our shortest path (i.e. $k = 1$) has a unique solution, then the minimal cost solution to the flow problem will be the same as our result (if $f \neq 1$, our solution will have a cost of f times our shortest path cost, but the path will still be the same).

This property is true due to flow problems minimizing $\sum_{e \in E} F_e W_e$ where F_e is the flow through edge e, our solution, and W_e is the given weight/cost of edge e. Since k shortest paths will minimize $\sum_{e \in E} P_e W_e$, where P_e is 1 iff e is on the path, we can see that these are nearly the same problem.

We can guarantee this property by giving all edges infinite capacity. A formal proof follows in section 1.3.2.

For a non-unique shortest path solution, our solution to the flow problem can be any linear combination of the shortest paths.

1.3.2 Proof

Assume demand $-f$ on our source node and inverted demand on our target node, infinite capacity on all edges, and a unique ($k = 1$) shortest path solution.

Suppose that our nonzero optimal solution $F \neq cP$ for some constant $c \neq 0$. Note that F then must have nonzero entries $F_i \neq F_j$ for some $i \neq j$. For the sake of this problem, we will refer to this quality as heterogeneous. By the constraints of the flow problem, we can decompose our solution F into homogeneous paths G^i such that all nonzero entries of G^i are equal. This necessitates the quality,

$$\sum_{e \in E} F_e W_e = \sum_i \sum_{e \in E} G_e^i W_e$$

Given our original supposition, it must be the case that there exists at least one j such that $G^j \neq cP$ for some constant $c \neq 0$. However, P minimizes $\sum_{e \in E} P_e W_e$ uniquely. Thus, $\sum_{e \in E} G_e^j W_e > \sum_{e \in E} P_e W_e$ and G^j is non-optimal. We can then say that F is non-optimal since all edges involved (nonzero entries in G^j) have capacity greater than or equal to $max(G^j)$.

2 Problem 2

2.1 Part A

2.1.1 PCA

We can utilize PCA to discover intra- and inter- group variability within our dataset. This can be done in a number of ways.

The most differentially expressed genes can be obtained by taking the median expression values within each compound. We now have 54 samples that are more resistant to innate biological differences between replicates. This is especially important to look at with PCA, given that PCA is explicitly not resistant to this type of variance.

PCA can then be performed on compound groupings. These groupings could include the whole dataset or as few as two compounds. In finding the primary directions of variance within the data, through looking at the highest-eigenvalue eigenvectors produced, we can see the most differentially expressed genes within the group.

For broader applications, these vectors can be used for dimensionality reduction – which is fairly important when a singular sample contains $25,000 points$. This can be done by expressing the training data in terms of these vectors and excluding vectors beyond some percentage of cumulative variance captured.

2.1.2 Hierarchical Clustering

Hierarchical Clustering can be used to group our samples by raw similarity. This can reveal biological differences or biases in our testing data. The most straightforward example of this would be performing clustering on our 30×5 controls. We can then see our samples' inherent differences and get an idea of what 'normal' variance is. This same strategy can be used to find intragroup commonalities and differences, potentially leading us to normalization measures.

Keeping this in mind, potentially with a normalizing step as discussed in the previous section (taking the median expression results for each compound), we can examine similarities between compounds. These results can then be compared with our neurotoxicity data to see if any early patterns emerge. This could lead us to a choice about our classification method (Questions to ask could be: Would a linear classifier work well here? Are similarly grouped compounds biologically/otherwise similar?).

As will be seen in Part B, hierarchical clustering can also aid in finding differentially expressed genes.

2.2 Part B

2.2.1 Input

In using an SVM, we have proposed that our data is linearly separable or can be modified such that it is. This is important to keep in mind while utilizing the SVM.

Our first problem is input. The natural approach would be to use the entire range of $25,000$ genes into our classifier. However, this creates an issue. We can end up over-fitting our data on noise since we have 'only' 240 samples. This is an

issue with nearly every classifier and is the backbone of why dimensionality reduction is useful. For our dimensionality reduction, a couple options are picking out differentially expressed genes and PCA.

For PCA, similar to the process discussed in Part A, we could use the eigenvectors from PCA applied to all of our median-normalized samples of each compound. Since these eigenvectors form a basis, we can also apply this to new data to be classified. However, this can unfairly favor outliers and mute the impact of equally important genes with less variance. Additionally, this ignores the class imbalance problem.

Instead, we can compute differentially expressed genes by utilizing our hierarchical clustering on our median-normalized compound samples. First, a cut-off can be chosen for the clustering to produce some number of groups. We can then extract vectors for use from PCA performed between compounds within a group. PCA will be applied between only two compounds at a time and some number of the vectors representing some predetermined percentage of the variance will be chosen for preconditioning for our SVM. In addition to these vectors, we can apply our median normalization *by group* and perform PCA in the same fashion between groups. This will allow us to capture important intra- and inter-group data while also avoiding the bias that comes with class imbalance.

For our final input, our genes will be replaced by these vectors/components. So, for example, if we had gene expression data $[1, 2, 3, 4]'$ and vectors $[0, 1, 1, 0]$, $[1, 0, 1, 0]$, our final feature values would be $[5, 4]'$.

We will withhold 1 sample of every component from our computation throughout this section. This leaves us with 54 samples of testing data and 216 samples for training.

2.2.2 Training and Evaluation

It would be reasonable to think that, when scanning for neurotoxic compounds, we would rather have false-positives than false-negatives. As such, we can evaluate the effectiveness of our method by looking at the area under the precision-recall curve on our testing data.

We can then continue with training a soft-margin SVM, optimizing for auPRC using our C (this can be found iteratively by moving in the direction of a local minima) and kernels (linear, Gaussian, homogeneous and inhomogeneous polynomial).

The confidence interval of our method can be estimated through the use of k-fold cross validation, choosing the interval within which a specified percentage of our validation data falls.

3 Problem 3

3.1 Part A

We have the reads and lengths

Transcript	Reads	Lengths
TS 1	605	45
TS 1 & TS 2 & TS 3	540	
TS 2	1122	620
TS 3	2993	100
TS 3 & TS 5 & TS 6	609	
TS 4	3100	300
TS 4	600	...
TS 5	813	1100
TS 5 & TS 6	108	
TS 6	417	120
TS X	776	200

We can then calculate values proportional to our relative abundances on unique reads by taking $\frac{\text{Unique Reads}}{\text{TS Length}}$.

$$\begin{array}{ll} \text{TS 1} & \frac{605}{45} \approx 13.44 \\ \text{TS 2} & \frac{1122}{620} \approx 1.81 \\ \text{TS 3} & \frac{2993}{100} \approx 29.93 \\ \text{TS 4} & \frac{3100+600}{300} \approx 12.33 \\ \text{TS 5} & \frac{813}{1100} \approx .74 \\ \text{TS 6} & \frac{417}{120} \approx 3.48 \\ \text{TS X} & \frac{776}{200} \approx 3.88 \\ \text{Total} & \approx 65.61 \end{array}$$

Normalizing these by dividing the above by the total, we get our unique relative abundances.[1]

$$\hat{f}^{\text{unique}} = \begin{array}{ll} \text{TS 1} & \approx .20485 \\ \text{TS 2} & \approx .02759 \\ \text{TS 3} & \approx .45618 \\ \text{TS 4} & \approx .18793 \\ \text{TS 5} & \approx .01128 \\ \text{TS 6} & \approx .05304 \\ \text{TS X} & \approx .05914 \end{array}$$

We now create the table G, which computes the portion of shared reads to add to each transcript with elements $G_\alpha = \frac{f_A^{\text{unique}}}{\sum_\alpha f_\alpha^{\text{unique}}}$ where α is a list of transcripts $[A, \dots]$ corresponding to some shared read.

$$\begin{array}{ll} G_{123} & \frac{.20485}{.20485+.02759+.45618} \approx .29748 \\ G_{213} & \frac{.02759}{.20485+.02759+.45618} \approx .04007 \\ G_{312} & \frac{.45618}{.20485+.02759+.45618} \approx .66246 \\ \hline G_{356} & \frac{.45618}{.45618+.01128+.05304} \approx .87643 \\ G_{536} & \frac{.01128}{.45618+.01128+.05304} \approx .02167 \\ G_{635} & \frac{.05304}{.45618+.01128+.05304} \approx .10190 \\ \hline G_{56} & \frac{.01128}{.01128+.05304} \approx .17537 \\ G_{65} & \frac{.05304}{.01128+.05304} \approx .82463 \end{array}$$

We can then add these back into the transcripts A.[2]

$$\begin{array}{ll} \text{TS 1} & 605 + 540 \cdot .29748 \approx 765.63920 \\ \text{TS 2} & 1122 + 540 \cdot .04007 \approx 1143.63780 \\ \text{TS 3} & 2993 + 540 \cdot .66246 + 609 \cdot .87643 \approx 3884.47427 \\ \text{TS 4} & 3100 + 600 = 3700 \\ \text{TS 5} & 813 + 609 \cdot .02167 + 108 \cdot .17537 \approx 845.13699 \\ \text{TS 6} & 417 + 609 \cdot .10190 + 108 \cdot .82463 \approx 568.11714 \\ \text{TS X} & 776 \end{array}$$

We then need to once again adjust for transcript length

$$\begin{array}{ll} \text{TS 1} & \frac{765.63920}{45} \approx 17.01420 \\ \text{TS 2} & \frac{1143.63780}{620} \approx 1.84458 \\ \text{TS 3} & \frac{3884.47427}{100} \approx 38.84474 \\ \text{TS 4} & \frac{3700}{300} \approx 12.33333 \\ \text{TS 5} & \frac{845.13699}{1100} \approx 0.76831 \\ \text{TS 6} & \frac{568.11714}{120} \approx 4.73431 \\ \text{TS X} & \frac{776}{200} \approx 3.88000 \\ \text{Total} & \approx 79.41947 \end{array}$$

[1] Note that these abundances might not add to exactly one due to approximation. Also note that we don't actually need to normalize here, as the next step effectively normalizes our data

[2] Our total is slightly different from our number of reads due to rounding

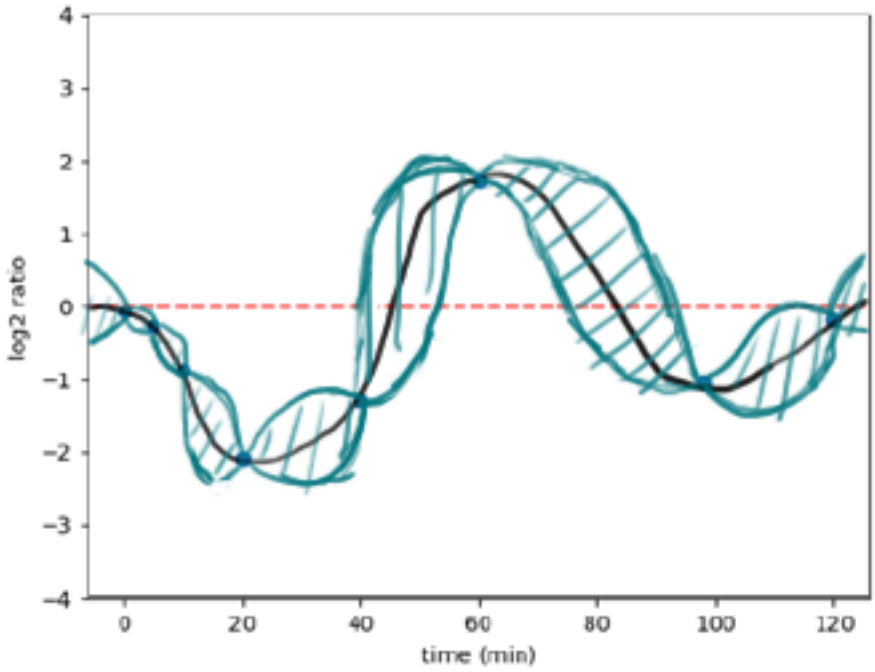

Figure 1: Sketch of a GP posterior and 95% confidence interval given a GP prior with a squared exponential kernel and $\mu = 0$

Lastly, we normalize to obtain our rescued relative abundances.

$$\begin{array}{ll} \text{TS 1} & \frac{17.01420}{79.41947} \approx .21423 \\ \text{TS 2} & \frac{1.84458}{79.41947} \approx .02323 \\ \text{TS 3} & \frac{38.84474}{79.41947} \approx .48911 \\ \text{TS 4} & \frac{12.33333}{79.41947} \approx .15529 \\ \text{TS 5} & \frac{0.76831}{79.41947} \approx .00967 \\ \text{TS 6} & \frac{4.73431}{79.41947} \approx .05961 \\ \text{TS X} & \frac{3.88000}{79.41947} \approx .04885 \end{array} \tag{1}$$

3.2 Part B

Given that TS X has 850 copies and a relative abundance of $.04885$, we can estimate that there are $\frac{850}{.04885} \approx 17400$ total genes divided in the portions from table 1. This provides us with the absolute abundances

$$\begin{array}{ll} \text{TS 1} & .21423 \cdot \frac{850}{.04885} \approx 3728 \\ \text{TS 2} & .02323 \cdot \frac{850}{.04885} \approx 404 \\ \text{TS 3} & .48911 \cdot \frac{850}{.04885} \approx 8511 \\ \text{TS 4} & .15529 \cdot \frac{850}{.04885} \approx 2702 \\ \text{TS 5} & .00967 \cdot \frac{850}{.04885} \approx 168 \\ \text{TS 6} & .05961 \cdot \frac{850}{.04885} \approx 1037 \\ \text{TS X} & .04885 \cdot \frac{850}{.04885} = 850 \end{array}$$

4 Problem 4

4.1 Part A

Given a GP prior with $\mu = 0$ and a squared exponential kernel, a sketch of the GP posterior with optimal kernel coefficients σ^2, ℓ on the given dataset is shown in figure 1. The first thing to notice is that, as time t gets farther

away from any training datapoints, the confidence interval becomes larger. This is a direct result of how our model is conditioned. If we have data X, Y, prior μ, and kernel K, our posterior distribution is given by

$$\mathcal{N}(\quad \mu(x) + K(x,X)K(X,X)^{-1}(Y-\mu(x)) \quad , \quad K(x_1,x_2) - K(x_1,X)K(X,X)^{-1}K(X,x_2) \quad) \tag{2}$$

Notice that, if $x \in X$, our distribution is $\mathcal{N}(Y_x, 0)$. Since we have introduced no noise, we are working on the assumption that our data is entirely correct. Then, we expect the mean to pass through the point. If we add a noise parameter, i.e. replace $K(X,X)$ with $K(X,X) + \alpha I$ in our covariance matrix, we can expect this property to no longer hold – although the confidence interval will still shrink when nearing a known datapoint.

In any other scenario, our variance scales based on distance from known points and our mean interpolates between the points using our mean added to a weighted sum of kernel functions between our desired point x and known points X. In this case, our 95% confidence interval (equal to twice the standard deviation) generally gets larger between points as t increases. This is because the sampling rate was reduced for higher t.

Notice that our standard deviation (and, therefore, the size of our confidence interval) scales based on our kernel function variance σ^2. Since our datapoints are close to our prior mean, we expect the kernel variable σ^2 to be low – perhaps 2.1 or lower as no points are greater than 2.1 units (in $\log_2$ ratios) away from the prior mean.

Similarly, we can anticipate ℓ to be around 15 or greater. The logic behind this is that our maximal datapoint spacing is 30 minutes. As such, $\ell < 15$ could provide kernel functions too steep/thin. This would then lead to valleys between distant sequential datapoints, which is non-optimal.

4.2 Part B

Let's create a statistical test to determine if a gene's temporal expression profile differs from its profile under drug exposure.

For simplicity, let's define

$$\mu_D^*(x) = \mu(x) + K(x,X)K(X,X)^{-1}(Y-\mu(x)) \quad K_D^*(x_1,x_2) = K(x_1,x_2) - K(x_1,X)K(X,X)^{-1}K(X,x_2).$$

as the GP posterior distribution means and variance for dataset D.

Let's start with some GP prior. We assume that an optimal conditioning on our expression data can be completed. Through the process described above, we can formulate two GP posteriors for our drug and control expression data following distribution 2: $N(\mu^*(x), K^*(x_1,x_2))$.

Given the log_2 ratios, the null and alternative hypotheses follow.

H_0: Our distribution is not significantly different than 0.

H_1: Our distribution is significantly different than 0.

We could then potentially discretize on a fine mesh and perform an Armitage test to attempt rejecting the null hypothesis. However, I'd like to suggest something more interesting.

Let's take the area under the absolute value of our GP posterior estimation:

$$\int_{\text{trial}} |\mu^*(x)| dx$$

we now have a measure proportional to the differential expression of our drug and control gene expressions.

Now we can quantify our measure. Say we want a specific p-value. We can use

$$\int_{\text{trial}} |\sigma z^{-1}(p)| dx$$

to calculate its equivalent in this space. Note that z is the z-score mapping and σ is our variance defined by our distribution 2.

Then, we can reject the null hypothesis for a specified p-value if

$$\int_{\text{trial}} |\mu^*(x)| dx > \int_{\text{trial}} |\sigma z^{-1}(p)| dx$$

This can be repeated per-gene to tell which genes have differing temporal expression profiles while responding to an environmental toxin versus their control.