Problem Set 2

EN 600.438/638

March 2, 2017

Due date: March 14, 2017 by midnight
Submission: Please submit your assignments by email to en600.438@gmail.com, including both code and written work. Problem sets can be submitted in Word, PDF, or plain text. All files should be submitted in a .zip archive labeled as follows: hw2_username.zip, where username is replaced with your JHED ID e.g. hw2_bshapi17.zip.
Reminders: You may discuss problems in small groups but must complete your own write-up and code. You may not copy any of your work or code from others, including but not limited to any resources you may find on the internet. List ALL people with whom you collaborated on your submission. You have 5 total late days for the semester, and can use a maximum of 3 on this assignment. Days will be rounded up; for example, if you submit your assignment 3 hours late, you will use 1 full late day. If you submit your assignment 25 hours after the deadline, you will use 2 full late days.
Programming component: Please follow the below guidelines carefully to ensure that you receive full credit. If you have questions or trouble setting up your code this way, please get in touch with one of the instructors.

- You must submit all source code in addition to answering the questions below. Your code should print all the output you needed to determine solutions to the questions (in a way that is easy to read and interpret).
- If you pre-process the data to enable easier loading or any other changes, please also provide the processed data along with a script and description of the processing applied.
- Instructors should be able to run your code, without exceptions, on gradx or ugradx with the following command: python *problem_number*.py *data_file* (where data_file is whatever data you need to run the code). Please include only one script per problem; if there are multiple parts, there should be one script to execute all parts in succession.
- If your code requires more than one data file, it should take each file as a command-line argument, in the order in which the data files are presented in this problem set.

Exercise 1. Linear Regression With a Multivariate Prior (10 points)
Given a linear regression model:

$$Y|X, \boldsymbol{\beta}, \sigma^2 \sim \mathcal{N}(\boldsymbol{\beta}^T \boldsymbol{X}, \sigma^2 I) \tag{1}$$

Suppose we would like to add a prior on $\boldsymbol{\beta}$ that follows a multivariate Gaussian distribution, then:

$$\boldsymbol{\beta} \sim \mathcal{N}(\boldsymbol{\mu}_p, \boldsymbol{\Sigma}_p) \tag{2}$$

where μ_p is a k-dimensional mean vector and Σ_p is a k by k covariance matrix (where k is the number of features).

a Derive the loss function that would arise from incorporating this prior into the likelihood-based loss function of linear regression.

b Describe its relationship to the ridge regression loss function.

Exercise 2. Support Vector Machines (25 points)

In problem set 1, we used logistic regression to do phenotype classification in gene expression and GWAS data. In this problem set, we explore using supervised machine learning for a very different classification problem. We will be using support vector machines (SVMs) for *splice site prediction* using DNA sequence data.

Background on Splicing:

Eukaryotic gene structure is non-continuous. This means that any given sequence of DNA that can be transcribed and translated to a protein contains interspersed patches of non-coding regions called introns. Introns are spliced from pre-mRNAs after transcription. The coding regions of the genes are called exons. One problem in computational gene finding concerns the recognition of splice sites that mark the boundaries between exons and introns in eukaryotes. The majority of splice sites are characterized by the presence of specific dimers on the intronic regions: **GT** for donor and **AG** for acceptor sites. However, only about 0.1-1% of all GT and AG occurrences in the genome represent true splice sites.

Please refer to pg. 17, figure 6-26 of the Molecular Biology background (from the first week of class) for a visual representation of splicing.

Data:

We provide you with a real splicing dataset. It comes from the UCI ML repository located at http://www.cs.toronto.edu/~delve/data/splice/desc.html. This contains a total of 3175 DNA sequences of which 2175 are training instances and 1000 are test instances. Additionally, each instance has an associated phenotype label, which is either "S" - spliced or "N" - not spliced. Each instance is a sequence of length 60. Our goal is to train a model

which can determine whether a given sequence is likely to undergo splicing or not.

Notes:

- Please standardize features to have mean 0 and standard deviation 1 before training the model. To do this, for each entry c_i of column C of the dataset you will do the following:

$$c_i = \frac{(c_i - \bar{C})}{\sigma_C}$$

  After doing this, if the SVM function you are using gives you an option to scale data, please set the option to False.

- This problem involves doing 5-fold cross-validation. Please use the KFolds.txt file, available in the Exercise2/ directory in the data accompanying this problem set.

- For programming sub-parts 2.1 and 2.2, please submit functions that take 4 inputs, in the following order:

  (a) training data

  (b) training labels

  (c) test data

  (d) test labels

  You may not require all of these inputs to complete each problem, but please write the functions to take them as input regardless.

Problems:

1. One form of feature encoding can be k-mer frequency counts. A k-mer is a string of k nucleotides. You will compute frequencies for 2-mers, 3-mers and 4-mers of the DNA sequenc e with 4 bases A, T, G and C. Since the number of unique k-mers depends on the length of the sequence, this gives us 16 + 64 + 256 = 336 features.
   For example, consider the DNA sequence given below:

   (a) ATGCTGTGC

   The set of possible 2-mers in this sequence is {AT, AC, AG, AA, TT, TC, TG, TA, CT, CC, CG, CA, GT, GC, GG, GA}. If we count the number of times each of these features appear in (a), we see:

   {AT = 1, AC = 0, AG = 0, AA = 0, TT = 0, TC = 0, TG = 3, TA = 0, CT = 1, CC = 0, CG = 0, CA = 0, GT = 1, GC = 2, GG = 0, GA = 0}

   Compute all 2-mers, 3-mers, and 4-mer features for all the given training and test DNA sequences. (10 points)

2. Using this data, train linear, Gaussian and polynomial (with d=3,4, and 6) kernel SVMs with 5 fold cross-validation. Use the default parameters in each case. For each case, report the accuracy on the training set as well as average accuracy across the cross-validation folds. (7 points)

3. From part (2), how does the cross-validation accuracy differ between SVMs with polynomial kernels of degree 3, 4 and 6? What changes do you observe with the increasing degree of the polynomial kernel? (3 points)

4. From part (2), which model gives the best cross-validation accuracy? Use that model on your test data and return test prediction accuracy. (5 points)

Exercise 3. PCA (30 points)

Programming

RNA-seq is widely used to measure trancript expression. However, many factors beyond phenotypes of interest may influence gene expression measurements including experimental variables (batch, RNA integrity, etc.) and biological covariates (age, sex, ethnicity, etc.). These may introduce systematic variability in data. In this exercise, you'll explore how variation in RNA-seq data are correlated with confounding covariates.

We provided a RNA-seq dataset (*expr.txt*) that contains the expression of 17,580 genes 73 human in prefrontal cortex samples. Of the 73 samples, 29 are from subjects with Parkinson's disease, and 44 are normal (control) samples (see *phen.txt*). We also provided data (*cov.txt*) measuring three sample covariates (age, RNA integrity numbers, and post mortem interval) for each subject. You'll compute principal components (PCs) of the gene expression dataset, and then investigate the correlation between the top PCs and these three covariates.

Note: the original data is collected from GEO (dataset id: GSE68719, url: http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE68719). You may follow the given URL and read the corresponding paper if you wish to learn more about the data in question.

1. Preprocessing: (5 points)
   In file *counts.txt*, the rows are genes and columns are the samples.

   (a) Take the log-transform of data. Elements of the matrix

   $$X_{ij}$$

   should be transformed as follows:

   $$L_{ij} = log_2(1 + x_{ij})$$

   where x_{ij} = expression of gene i in sample j.

   (b) Scale the log transformed expression data using the z-score (gene-wise). Transform the elements of the new matrix as follows:

   $$Z_{ij} = (L_{ij} - \mu_i)/\sigma_i$$

   ... where L_{ij} = log transformed expression of gene i in sample j, μ_i = mean expression of gene i and σ_i = standard deviation of gene i.

2. Transpose the scaled data you get from 3.1.b. After transposing, rows will now represent samples and columns will be genes. Compute the principal components of the scaled expression data. Report the percent variance explained by each of the top 10 PCs. (10 points)

3. Visualization of data is also very important in genomics. However, it is difficult to visualize high dimensional data. A common practice is use the top 2 principal components for visualization. Make a scatter plot of all the samples after projecting them onto the first 2 principal components, i.e. compute XP where P is the g x 2 matrix containing the top two principal components (g is the number of genes in the dataset). Use different symbols (e.g. + and x) for normal and Parkinson's samples respectively in the plot. (5 points)

4. Report pairwise Pearson correlation values (along with p-values) between each of the top 10 PCs and the provided covariates (30 correlations in total). How many PCs are strongly correlated with at least one of the covariates? For this exercise, assume a correlation is strong if $|r| > 0.2$ and p-value < 0.05. (3 points)

5. Report Pearson correlation values (along with p-values) between disease status and each of the three covariates. Based on these correlation values, do you think the disease status may be confounded with RNA integrity number or post-mortem interval? In such a scenario, do you think that the principal components fully reflect the disease status of the samples? If not, why not? (3 points)

6. Test for genes with significant differences in expression between normal and Parkinson's samples using a t-test. Report the number of genes that are significant at $p < 0.05$. Then, using Benjamini Hochberg (at FDR = 0.05), report the number of significant genes after correcting for multiple hypothesis testing. Do you think that these results are affected by the covariates you examined in the previous parts? (4 points)

Exercise 4. Clustering (35 points)
In this exercise. you'll separate samples by tissue type using a clustering approach. The provided dataset (*expr.txt*) includes normalized gene expression of 100 genes in 1816 samples. These samples come from 5 different tissues: Muscle-skeletal, Lung, Thyroid, Adipose-subcutaneous, and Whole blood (encoded as 0,1,2,3, and 4, respectively, in *class_labels.txt*). Your job is to cluster the samples in an unsupervised fashion – without using given tissue type labels – and observe how well the resulting clusters reflect the true tissue types.

Note: the data given to you here is a preprocessed subset of the data available from the Genotype-Tissue Expression (GTEx) consortium (v6). If you are interested, you may visit the GTEx website (http://www.gtexportal.org/) and read the relevant papers (http://www.gtexportal.org/home/news?id=176263).

1. **[EN 638 only]** Implement a function *myKmeans()* to train a k-means clustering model from scratch. (15 points)
   The function should take the following parameters:

- *data* - n x d matrix. n = number of samples, d = number of features
- k - int. number of clusters.
- *centers* - k x d-length matrix. initial value of cluster centers.
- *maxit*- int. maximum number of iterations.

The myKmeans() function should return a tuple of two items (in this order):

- *clusters* - n-length vector containing the cluster assignment label of each sample.
- *centers* - k x d matrix containing the final k d-dimensional cluster centroids.

You should also implement a function *predict()* that takes the output of the *myKmeans()* function as well as test samples, and predicts the cluster assignment of each sample using the closest centroid. The *predict()* function should take the following inputs:

- *model* - the tuple returned by the *myKmeans()* function.
- *test_data* - a q x d matrix where q = number of query samples.

The *predict()* function should return the following output:

- *clusters* - A q-length vector containing the cluster assignment of each sample.

2. Run k-means with 5 clusters. Use the first 5 samples in the data as the initial centers, and run a maximum 10 iterations. (EN 638 - use the *myKmeans()* function you wrote in the previous part, EN 438 - you may use any package available in Python.) (5 points)

   (a) Report the number of samples in each cluster.

   (b) Report the final cluster centers.

3. Selection of the number of clusters (k) is an important factor in clustering. There are different approaches for select of k including Bayesian Information Criterion (BIC), the Elbow method, Silhouette, etc. In this exercise, you'll select k using the BIC score defined as:

$$BIC = -2 \log \hat{L} + m \log n \tag{3}$$

   Here, $\hat{L}$ is the estimated maximum likelihood, m is the number of free parameters in the model, and n is number of samples. Generally, the smaller the BIC score, the better the model.

   (a) Free parameters are parameters which are learned by the model. How many free parameters are there in a k-means model with k clusters of d-dimensional points? Write the equation to compute the likelihood (or log-likelihood) for k-means. (5 points)

(b) Train k-means clustering model using values of k from 2 to 10. Use the first k points as initial centers. Plot BIC scores of each model against k. (5 points)

(c) Which k will you select here? (2 points)

4. Make a scatter plot of the projection of the data onto the top 2 principal components. Label the samples by cluster membership from your k-means result (use the k selected in the previous question) using different colors, and also label true tissue types using different symbols (such as + or x). (3 points)

Exercise 5. Covariance Matrix Tricks with PCA [**EN 638 only**] (10 points)

Typically, we find the principal components of an n x d matrix X (n samples, d features) by computing the eigenvectors of the covariance matrix X^TX, i.e. we compute vectors v such that:

$$X^TXv = \lambda v$$

... for some λ. As it turns out, we can compute the principal components without computing X^TX directly, but rather by computing XX^T and its eigenvectors, i.e. vectors u such that:

$$XX^Tu = \lambda u$$

... for some λ. Show a simple transformation that can be applied to u to compute the eigenctors v of X^TX. Justify why we might prefer to compute the principal components this way rather than using the original formulation.