Computational Data Analysis

ISYE 6740

Final Exam– Due Dec. 13, 2019

Total Score: 100

If you think a question is unclear or multiple answers are reasonable, please write a brief explanation of your answer, to be safe. Also, show your work if you want wrong answers to have a chance at some credit: it lets us see how much you understood.

(Please sign the honor code below.) I have neither given nor received any unauthorized aid on this exam. I understand that the work contained herein is wholly my own without the aim from a 3rd person. I understand that violation of these rules, including using an authorized aid or copying from another person, may result in my receiving a 0 on this exam .

Name:

GT ID:

GT Account:

Question 1 [15 points]	
Question 2 [15 points]	
Question 3 [15 points]	
Question 4 [20 points]	
Question 5 [10 points]	
Question 6 [25 points]	

1 Clustering [15 pts]

1. (5 points) Explain what is the difference between K-means and spectral clustering?

   Both K-means and spectral clustering are able to programmatically assign data points into groups given a similarity/dissimilarity function. The main difference between the two algorithms is that K-means, the much simpler algorithm, only uses a distance metric as it's similarity/dissimilarity measure. Spectral clustering, on the other hand, is able to use the geometry/connectivity of the data to assign data into groups.

   K-means steps:

   (i) Initialize cluster centers There are multiple ways to do this, either pick randomly within the feature space or pick a random K actual points from the dataset.

   (ii) Assign each point to the closest cluster center

   (iii) Adjust the cluster centers based on the average center of the above assignment.

   (iv) Iterate until cluster centers converge

   Spectral Clustering steps:

   (i) Build an adjacency matrix A using nearest neighbors

   (ii) Represent graph as adjacency matrix $A \epsilon R^{mxm}$

   (iii) Form a special matrix $L = D - A$ the graph Laplacian

   (iv) Compute k eigenvectors of L corresponding to the k smallest eigenvalues

   (v) Run k-means algorithm on the above eigenbectors treating each row as a new datapoint

   The two methods above both use K-means as the end method to assign points into groups. Spectral clustering has an initial data transformation that allows the final K-means operation to consider the connectivity of the data.

2. (5 points) What is in common, and what is the main difference between spectral clustering and PCA?

   Spectral clustering and PCA are two methods that use eigendecomposition to find patterns within a dataset. The difference between the two are how eigendecomposition is used within the algoritms.

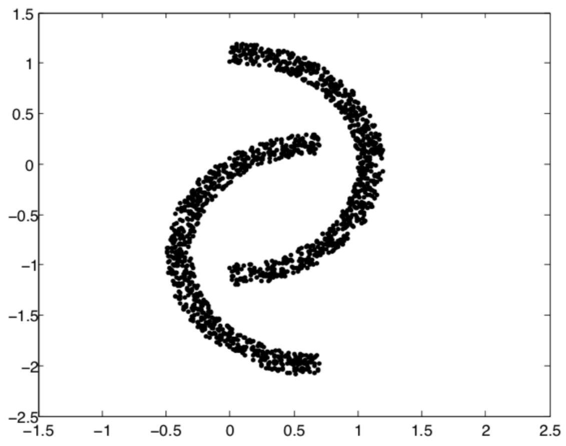
   PCA uses eigendecomposition for dimensionality reduction. The purpose is to preserve as much information of the underlying data as possible but reducing the dimensions to be able to visualize the data, apply simpler models to the data, or speed up a subsequent learning task. After eigendecomposition on the covariance matrix, the eigenvectors that correspond to the K largest eigenvalues are chosen to represent the data. The eigenvalues represent the variation in the data and variation can be seen as a proxy for information within the data. It is therefore possible to drastically reduce the dimensions within while preserving the majority of the underlying information.

Spectral clustering uses eigendecomposition to determine the connectedness of a graph. It does this by performing eigendecomposition on the graph Laplacian of the data or any other similarity matrix (not strictly a covariance matrix as in PCA) and then performing eigendecomposition on the transformed data.

Both spectral clustering and PCA use eigendecomposition. Data that has been transformed through PCA can be used for clustering, although the main goal is dimensionality reduction where spectral clustering is obviously used for clustering.

3. (5 points) For the following data (two moons), give one method that will successfully separate the two moons? Explain your rationale. Spectral clustering will be able

to easily separate the below data. K-means is not a viable option because it will only consider the distance between points not the connectedness of the data points. The obvious separation of the points is the two moons, therefore spectral clustering's ability to consider the geometry of the data is the obvious choice to cluster the below data. There will be a considerable eigengap between the 2 k smallest eigenvalues that will be able to determine the connectedness between the datapoints that make up each half moon.



2 Classification [15 points]

1. (5 points) List all methods below which can be used for classification:

   (a) AdaBoost (b) Decision Trees (c) EM and Gaussian Mixture (d) Histogram (e) K-nearest neighbors (f) K-means (g) Kernel density estimation (h) Linear Regression (i) Logistic Regression (j) Naive Bayes.

   Methods that can be used for classification are:

   - Adaboost
   - Decision Trees
   - EM and Gaussian Mixture Model
   - K-nearest neighbors
   - K-means (classification through clustering)
   - Logistic Regression
   - Naive Bayes

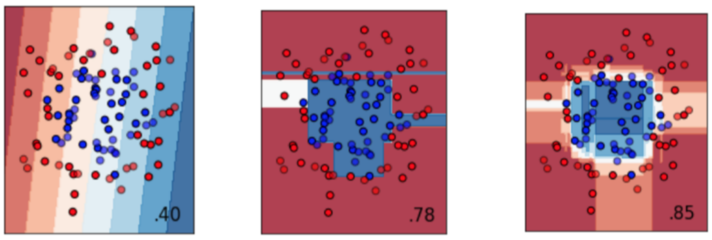
   Kernel density estimation and histograms are not necessarily classification. They can be used to determine the likelihood of a particular data point and are often used as a building block for more sophisticated classification algorithms.

   K-means is not a clustering algorithm and should not be used as a classification algorithm if possible. One approach I read about is using k-means to cluster the data into various classes then use a more robust classification algorithm such as SVM to classify new data based on these results.

2. (5 points) Which of the decision boundaries below correspond to (a) Random Forest, (b) Decision Tree, (c) SVM. Explain your reasons to fully justify your answers.

   Going from left to right the decision boundaries correspond to SVM, decision tree, and random forest.

   The first image is obviously SVM because the decision boundaries are linear and SVM is a linear classifier. The next two decision boundaries are similar, but the last image on the very right performed better (.85 > .78) and looks like it has a lot more noise assoicated with it. That extra noise and layers is the result of the randomness in the random forest, which generally performs better than the single decision tree, which is the middle image..

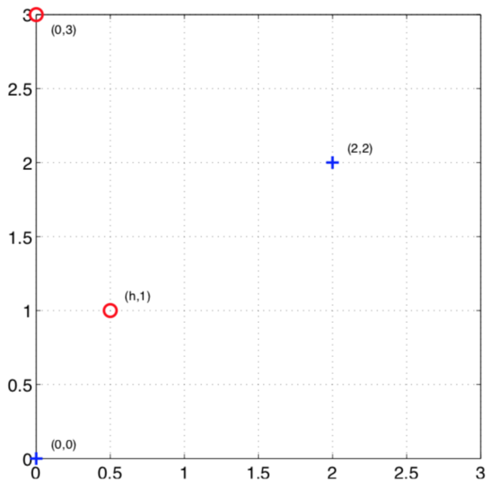


3. (5 points) Is the following statement true / false: "In the AdaBoost algorithm, the weights on all the misclassified points will go up by the same multiplicative factor." Explain your reason.

   True, this comes from the equation used in updating, D_T. All misclassified points are scaled equally in this equation to determine the new weights.

3 SVM [15 points]

Suppose we only have four training examples in two dimensions as shown in Fig. The positive samples at $x_1 = (0, 0)$, $x_2 = (2, 2)$ and negative samples at $x_3 = (h, 1)$ and $x_4 = (0, 3)$.



1. (5 points) For what h s.t. $h > 0$ to be so that the training points are still linearly separable?

   The blue cross class are on the line $y = x$ and the point in question has a $y = 1$. Therefore any $0 < h < 1$ should allow the points to be linearly sperable.

2. (5 points) Does the orientation of the maximum margin decision boundary change as a function of h when the points are separable?

   No because the other points remain and would be the support vecotrs.

3. (5 points) Explain why only the data points on the "margin" will contribute to the decision boundary?

The maximum margin decision boundary is the perpendicular distance between the margins of the decision boundary, so by definition they define the decision boundary.

4 Variable section [20 points]

Suppose we have data $\{x_i, y_i\}$, $i = 1, \dots, m$, where $x_i \in \mathbb{R}^p$ corresponds to p features.

1. (5 points) Write down the optimization problem we solve with Ridge Regression and Lasso. Make sure you explain your notations: which are the decision variables, and which are data.

   Ridge Regression:

   Given m data points, find θ that minimizes the regularized mean square error. Where x are the features and y are the target variables, θ is the decision variable to be optimized and λ a positive regularization paramter that can be tuned.

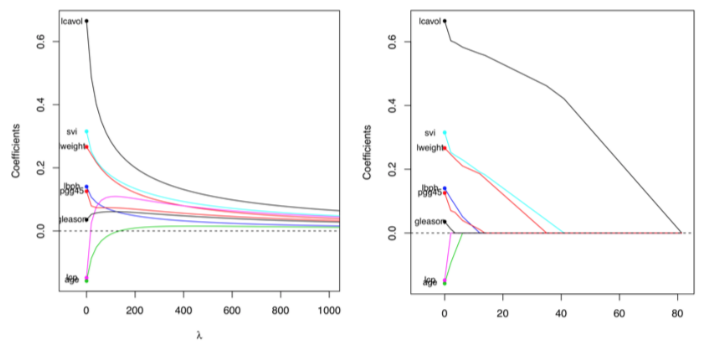
   $$\theta^r = argmin_\theta L(\theta) = \frac{1}{m}\sum_{i=1}^{m}(y^i - \theta^T x^i)^2 + \lambda||\theta||^2$$

   Lasso Regression:

   Given m data points, find θ that minimizes the regularized mean square error. Where x are the features and y are the target variables, θ is the decision variable to be optimized and λ a positive regularization paramter that can be tuned.

   $$\hat{\theta} = argmin_\theta L(\theta) = \frac{1}{m}\sum_{i}^{m}(y^i - \theta^T x^i)^2 + \lambda||\theta||$$

2. (5 points) Which of the solution paths below corresponds to Ridge regression and which corresponds to Lasso?



   Ridge regression force coefficients to be lower, but it does not force them to 0. Lasso can actually return 0 coefficients. Therefore the left hand image is Ridge and the right is Lasso.

3. (5 points) Explain what's the difference between Lasso and Ridge regression. We need Lasso for what setting?

   The difference between Ridge and Lasso are the regularization terms. Lasso uses L1 regularization and Ridge uses L2 regularization. L1 regularization, as discussed above is able to set coefficients to 0. This has the added benefit of a built in feature selection tool.

4. (5 points) Explain how to tune the regularization parameters for Lasso and Ridge regression.

   Cross validation is used to tune the regularization parameter λ. K-fold cross-validation is performed by the steps below.

   For each fold i:

   1) set aside $\alpha \cdot m$ samples of D (where $m = |D|$) as the held-out data. They will be used to evaluate the error.
   2) Fit a model to the remaining $(1 - \alpha) \cdot m$ samples in D
   3) Calculate the error of the model with the held-out data.

   Repeat the above K times, choosing a different held-out data set each time, and the errors are averaged over the folds.

   K-fold cross-validation is performed on different values of λ and the λ that results in the lowest error is chosen.

5 Neural networks [10 points].

1. (5 points) Consider a neural networks for a binary classification using sigmoid function for each unit. If the network has no hidden layer, explain why the model is equivalent to logistic regression.

   A neural network is basically a combination of layered and stacked generalized linear models, which logistic regression is a member of. If you have just a single perceptron, without any hidden layers, and choose the logit as your activation function, this is equivalent to logistic regression.

2. (5 points) Consider a simple two-layer network in the lecture slides. Given m training data (x^i, y^i), $i = 1, \ldots, m$, the cost function used to training the neural networks

   $$\ell(w, \alpha, \beta) = \sum_{i=1}^{m} (y^i - \sigma(w^T z^i))^2$$

   where $\sigma(x) = 1/(1 + e^{-x})$ is the sigmoid function, z^i is a two-dimensional vector such that $z^i_1 = \sigma(\alpha^T x^i)$, and $z^i_2 = \sigma(\beta^T x^i)$. Show the that the gradient is given by

   $$\frac{\partial \ell(w, \alpha, \beta)}{\partial w} = \sum_{i=1}^{m} 2(y^i - \sigma(u^i))\sigma(u^i)(1 - \sigma(u^i))z^i,$$

   where $u^i = w^T z^i$. Also find the gradient of $\ell(w, \alpha, \beta)$ with respect to α and β and write down their expression.

6 Programming: Bayes and KNN classifier [25 points]

In this programming assignment, you are going to apply the Bayes Classifier to handwritten digits classification problem. Here, we use the binary 0/1 loss for binary classification, i.e., you will calculate the miss-classification rate as a performance metric.

To ease your implementation, we selected two categories from USPS dataset in usps-2cls.mat (or usps-2cls.dat, usps-2cls.csv).

1. (15 points) Your first task is implementing the classifier by assuming the covariance matrices for two classes are a diagonal matrix Σ_1, Σ_2.

   Using slides from "Classification I", assuming $P(y = 1) = P(y = -1)$ (i.e., the prior distribution for two classes are the same), using Bayes decision rule to write down the decision boundary. (Hint, it should be a quadratic decision boundary.) The

   multivariate normal distribution is as follows:

$$P(x|y) = N(x|\mu_y, \Sigma_y) = \frac{1}{(2\pi)^d|\Sigma|^{1/2}} \exp[-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)]$$

   The posterior probability of a given distribution i using Bayes rule is:

$$q(x) = P(y|x) = \frac{P(x|y)P(y)}{P(x)} = \frac{N(x|\mu_y, \Sigma_y)P(y)}{\sum_y P(y)N(x|\mu_y, \Sigma_y)}$$

   If you have a binary classification with 2 multivariate Probability Distributions i and j then the Bayes decision occurs when $i = j$.

   A quadratic form of this boundary in the canonical form $x^TAx + b^Tx + c = 0$ can be obtained by solving for $q(x)_i - q(x)_j$

   Note: $\sum_y P(y)N(x|\mu_y, \Sigma_y)$ is a normalizer and will be ignored in the below derivation.

$$\frac{1}{(2\pi)^d|\Sigma_i|^{1/2}} \exp[-\frac{1}{2}(x-\mu_i)^T\Sigma_i^{-1}(x-\mu_i)]\pi_i = \frac{1}{(2\pi)^d|\Sigma_j|^{1/2}} \exp[-\frac{1}{2}(x-\mu_j)^T\Sigma_j^{-1}(x-\mu_j)]\pi_j$$

   Where $\pi = P(y)$, the prior.

   Taking the natural log of both sides, multiplying by 2 and bringing the right side over to the left we have:

$$x^T(\Sigma_i - \Sigma_j)^{-1}x + 2(\Sigma_j^{-1}\mu_j - \Sigma_i^{-1}\mu_i)^Tx + (\mu_i^T\Sigma_i^{-1}\mu_i - \mu_j^T\Sigma_j^{-1}\mu_j) + \ln\frac{|\Sigma_i|}{|\Sigma_j|} + 2\ln\frac{\pi_j}{\pi_i} = 0$$

   We can further reduce the above because we are assuming Σ_i, Σ_j are diagonal matrices and thus the features are assumed to be conditionally independent. Therefore we can use the univariate Gaussian distribution of each individual feature.

$$q(x_j) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{(x_j - \mu)^2}{2\mu^2})$$

Now we will estimate the mean vector and the sample covariance matrices for two classes using the training data (hint: you can use sample mean and sample covariance vector). Report the misclassification rate (error rate) over the training set and over the testing set averaged over the 100 random train/test splits by using different value of splitting ratio p. Explain and compare the performance of each classifier.

After implementing these methods, you should evaluate your algorithm on the given set. Repeat 100 times: split the dataset into two parts randomly, use p portion for training and the other $1-p$ portion for testing. Let p change from 0.1, 0.2, 0.5, 0.8, 0.9.

Please implement the algorithm **from scratch** yourself. Make sure to provide code, results (required above) together with necessary explanations to your results.

The below is a summary of my train and test misclassification rates.

p	train	test
0.1	0.50	0.50
0.2	0.50	0.50
0.5	0.47	0.47
0.8	0.29	0.29
0.9	0.19	0.19

The classifier was able to improve it's error rate as it used more of the data for training. The 50% for p=.01, 02 is intersting. It turns out that it was predicting class 0 for all instances. It is also interesting that training and testing error rate is equal for each classifier. I assumed this to be an error and that the training error rate woulb be less than the testing error rate, but after inspection it appears this is correct.

2. (10 points) Now repeat the classification again using K-nearest neighbors, for $K = 5, 10, 15, 30$. Repeat 100 times: split the dataset into two parts randomly, use p portion for training and the other $1-p$ portion for testing. Let p change from 0.1, 0.2, 0.5, 0.8, 0.9. Report the training error and testing error for each case.

	k=5		k=10		k=15		k=30	
	train	test	train	test	train	test	train	test
p								
0.1	0.05	0.09	0.09	0.13	0.09	0.11	0.14	0.16
0.2	0.03	0.06	0.06	0.09	0.06	0.08	0.10	0.11
0.5	0.02	0.04	0.04	0.05	0.04	0.05	0.06	0.07
0.8	0.02	0.03	0.03	0.04	0.03	0.04	0.05	0.05
0.9	0.02	0.03	0.03	0.03	0.03	0.03	0.05	0.05

Wow, Knn performs much better than Bayes. I was not expecting that. It follows a similar pattern as Bayes in that it performs better the larger amount of data it has to train with, but it outperforms the best Bayes classifier when it uses only 10% of the data to train, that is amazing. Unfortunately, it is a very slow algorithm. Performing the iterations took significantly longer for Knn and using this in a real time situation with many more data points with many more models does not seem to be realistic no matter how well it performs.

Focusing only on Knn it is also noteworthy that 5 neighbors performed better than the models using more neighbors.

Thank you for all your hard work. I enjoyed the class! ☺