STAT 6910-001 – Principles of ML – Homework #5

Due: 5:00 PM 11/8/19

1. **Support Vector Regression (25 pts).** Support vector regression (SVR) is a method for regression analogous to the support vector classifier. Let $(\mathbf{x}_i, y_i) \in \mathbb{R}^d \times \mathbb{R}$, $i = 1, \ldots, n$ be training data for a regression problem. In the case of linear regression, SVR solves

$$\begin{aligned} \min_{\mathbf{w}, b, \xi^+, \xi^-} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{n}\sum_{i=1}^{n}\left(\xi_i^+ + \xi_i^-\right) \\ \text{s.t.} \quad & y_i - \mathbf{w}^T\mathbf{x}_i - b \le \epsilon + \xi_i^+ \ \forall i \\ & \mathbf{w}^T\mathbf{x}_i + b - y_i \le \epsilon + \xi_i^- \ \forall i \\ & \xi_i^+ \ge 0 \ \forall i \\ & \xi_i^- \ge 0 \ \forall i \end{aligned}$$

   where $\mathbf{w} \in \mathbb{R}^d$, $b \in \mathbb{R}$, $\xi^+ = \left(\xi_1^+, \ldots, \xi_n^+\right)^T$, and $\xi^- = \left(\xi_1^-, \ldots, \xi_n^-\right)^T$. Here ϵ is fixed.

   (a) (5 pts) Show that for an appropriate choice of λ, SVR solves

$$\min_{\mathbf{w}, b} \frac{1}{n}\sum_{i=1}^{n} \ell_\epsilon\left(y_i, \mathbf{w}^T\mathbf{x}_i + b\right) + \lambda\|\mathbf{w}\|^2$$

   where $\ell_\epsilon(y, t) = \max\{0, |y - t| - \epsilon\}$is the ϵ-insensitive loss, which does not penalize prediction errors below a level of ϵ.

   (b) (13 pts) The optimization problem is convex with affine contraints and therefore strong duality holds. Use the KKT conditions to derive the dual optimization problem in a manner analogous to the support vector classifier (SVC). As in the SVC, you should eliminate the dual variables corresponding to the constraints $\xi_i^+ \ge 0$, $\xi_i^- \ge 0$.

   (c) (4 pts) Explain how to kernelize SVR. Be sure to explain how to recover $\mathbf{w}^*$ and b^* and write the final kernelized regression function $f(\boldsymbol{x})$ in terms of the optimal dual variables and b^*.

   (d) (3 pts) Argue that the final predictor will only depend on a subset of training examples (i.e. support vectors) and characterize those training examples.

2. **PCA (10 pts).** Read over the proof of PCA in the lecture notes. Then solve the following problems. The first problem motivates a method for selecting the number of principal components, which was discussed at the end of the PCA lecture.

   (a) (5 pts) Let $k \in \{0, 1, \ldots, d\}$ be arbitrary. Show that

$$\min_{\boldsymbol{\mu}, A, \{\boldsymbol{\theta}_i\}} \sum_{i=1}^{n} \|\boldsymbol{x}_i - \boldsymbol{\mu} - A\boldsymbol{\theta}_i\|^2 = n\sum_{j=k+1}^{d} \lambda_j,$$

   where A ranges over all $d \times k$ matrices with orthnormal columns.
   Hint: This is easy if you use properties of the trace operator.

   (b) (5 pts) Give a condition involving the spectral decomposition of the sample covariance matrix that is both necessary and sufficient for the subspace $\langle A \rangle$ in PCA to be unique.

3. **Eigenfaces (15 pts).** In this exercise you will apply PCA to a modied version of the Extended Yale Face Database B. The modied database is available in the file `yalefaces.mat` on Canvas. The modication was simply to reduce the resolution of each image by a factor of 4 4 = 16 to hopefully avoid computational and memory bottlenecks.
Plot images of a few of the samples. The data consist of several different subjects (38 total) under a variety of lighting conditions.

   (a) (5 pts) By viewing each image as a vector in a high dimensional space, perform PCA on the full dataset. Do not use a built-in method that performs PCA automatically. Hand in a plot of the sorted eigenvalues (use the `semilogy` command in Matlab; `plt.semilogy` in Python) of the sample covariance matrix. How many principal components are needed to represent 95% of the total variation? 99%? What is the percentage reduction in dimension in each case? Useful commands in Matlab: `reshape, eig, svd, mean, diag`, and Python: `np.reshape, np.linalg.eig, np.linalg.svd, np.mean, np.diag`.

   (b) (5 pts) Hand in a 4×5 array of subplots showing principal eigenvectors ('eigenfaces') 0 through 19 as images, treating the sample mean as the zeroth order principal eigenvector. Comment on what facial or lighting variations some of the different principal components are capturing. Useful commands in Matlab: `subplot, imagesc, colormap(gray)`, in Python: `plt.imshow(x, cmap=plt.get cmap('gray'))`, and for subplots a useful link is
   `http://matplotlib.org/examples/pylab examples/subplots demo.html`

   (c) (5 pts) Turn in your code.

4. **PHATE and Clustering (40 pts).**
Download all of the data for the MNIST dataset from `yann.lecun.com/exdb/mnist/`. This should give you 60,000 images in the training data and 10,000 images in the test data. You will apply PHATE to visualize the data and a couple of clustering algorithms to this dataset. You may use any existing packages or libraries for this problem as long as you cite them. For all parts, you may assume the same number of clusters as classes (10 in this case).

   (a) (4 pts) Run PHATE on just the features of the training data (do not include labels) using the default parameters to obtain a 2D representation of the data. Report the value of t selected using the von Neumann entropy (VNE). Rerun PHATE using two different values of t, one value larger than the value chosen using the VNE and one value smaller. Plot the PHATE visualization for all three values of t (you should end up with 3 different plots) with the data points colored by the labels. Comment on the plots. Which of the three values seems to give better separation between the classes? Does the relative position of the different classes make sense?

   (b) (4 pts) Repeat part (a) for the test data and comment on any similarities and differences between the results of the two datasets.

   (c) (4 pts) Apply k-means clustering to just the features of the training data (do not include labels). Compute the adjusted Rand index (ARI) between your cluster outputs and the true labels of the data points and report the value. Choose one of the PHATE plots from part (a) (a specific value of t) and plot the PHATE visualization colored by the cluster labels. Based on the visualization and the ARI value, does k-means match the true labels well?
   Note: You may need to do subsampling to make this computationally feasible. If you do, repeat the clustering for multiple (say 10-20) random subsamples and report the average ARI. For the PHATE plot, choose one of the subsamples and show the results on that.

   (d) (4 pts) Repeat part (c) for the test data and comment on any similarities and differences between the results of the two datasets.

   (e) (4 pts) Apply spectral clustering to just the features of the training data (do not include labels) using a radial or Gaussian kernel. Compute the ARI between your cluster outputs and the true labels of the data points. Use the ARI to tune the kernel bandwidth parameter. Report the ARI using your selected bandwidth. Choose one of the PHATE plots from part (a) and plot the PHATE visualization colored by the cluster labels. Based on the visualization and the ARI value,

does spectral clustering do better or worse than k-means?
Note: You may need to do subsampling to make this computationally feasible. If you do, repeat the final clustering for multiple (say 10-20) random subsamples and report the average ARI. For the PHATE plot, choose one of the subsamples and show the results on that.

(f) (4 pts) Apply spectral clustering to the features of the test data using the same kernel and bandwidth you selected in part (e). Report the ARI and plot the PHATE visualization colored by the cluster labels. Does spectral clustering do better than k-means here?

(g) (4 pts) Run PHATE on just the features of the training data using the default parameters to obtain a 10-dimensional representation of the data. Report the value of t selected using the VNE. Apply k-means to the 10-dimensional representation. This can be viewed as a variation on spectral clustering. Compute the ARI between your cluster outputs and the true labels of the data points. Choose one of the PHATe plots from part (a) and plot the PHATE visualization colored by the cluster labels. Based on the visualization and the ARI value, which of the three clustering approaches does the best?
Note: You may need to do subsampling to make this computationally feasible. If you do, repeat the clustering for multiple (say 10-20) random subsamples and report the average ARI. For the PHATE plot, choose one of the subsamples and show the results on that.

(h) (4 pts) Repeat part (g) for the test data and comment on any similarities and differences between the results of the two datasets.

(i) (4 pts) Generally when we're clustering data, we don't have access to the true labels which makes it difficult to tune parameters like the kernel bandwidth for spectral clustering. What is another way you could tune the bandwidth without using cluster or class labels?

(j) (4 pts) Turn in your code.

5. **Ncut and Normalized Spectral Clustering (10 pts).** Assuming $K = 2$, show that a relaxation of the Ncut problem discussed in class is solved by normalized spectral clustering, i.e., spectral clustering with the normalized graph Laplacian $\tilde{L} = D^{-1}L$.
Hint: First define $\boldsymbol{f}_A$ in an analogous way to the treatment of RatioCut. Verify analogous formulas for $\boldsymbol{f}_A^T L \boldsymbol{f}_A$, $\mathbf{1}^T D \boldsymbol{f}_A$, and $\boldsymbol{f}_A^T D \boldsymbol{f}_A$ to formulate the relaxation. Then make the substitution $g = D^{1/2}\boldsymbol{f}$ and reformulate the relaxation with g as the variable. Once you solve for g, don't forget to transform back to $\boldsymbol{f}$.