CS 178: Machine Learning: Winter 2019

Homework 4

Due Date: **Thursday, March 7th, 11:59pm on EEE/Canvas**

The submission for this homework should be `a PDF report and code`. Code submission is accepted in `.py` or `.ipynb` or `.zip` format. Use the .zip format if you are submitting multiple code files, but `do not` zip the pdf report. Your submission must include exactly two files: one (`.py` OR `.ipynb` OR .zip) file AND one `.pdf` file. For many questions you will be completing missing sections in a Python file that we provide. Be sure to include copies of any code you write as answers to the appropriate question, so that it may be graded.

Your solution may be submitted up to 4 days late (by 11:59pm, 96 hours after the deadline); Late homework will be penalized 10% per day late (or part thereof), up to 4 days late (96 hours after the deadline); after this point, solutions will be distributed and handins will no longer be accepted.

Problem 1: Decision Trees for Spam Classification (30 points)

We'll use the same data as in our earlier homework: In order to reduce my email load, I decide to implement a machine learning algorithm to decide whether or not I should read an email, or simply file it away instead. To train my model, I obtain the following data set of binary-valued features about each email, including whether I know the author or not, whether the email is long or short, and whether it has any of several key words, along with my final decision about whether to read it ($y = +1$ for "read", $y = -1$ for "discard").

x_1	x_2	x_3	x_4	x_5	y
know author?	is long?	has 'research'	has 'grade'	has 'lottery'	$\Rightarrow$ read?
0	0	1	1	0	-1
1	1	0	1	0	-1
0	1	1	1	1	-1
1	1	1	1	0	-1
0	1	0	0	0	-1
1	0	1	1	1	1
0	0	1	0	0	1
1	0	0	0	0	1
1	0	1	1	0	1
1	1	1	1	1	-1

In the case of any ties where both classes have equal probability, we will prefer to predict class +1.

1. Calculate the entropy $H(y)$ of the binary class variable y. *Hint:* Your answer should be a number between 0 and 1. *(5 points)*
2. Calculate the information gain for each feature x_i. Which feature should I split on for the root node of the decision tree? *(10 points)*
3. Determine the complete decision tree that will be learned from these data. (The tree should perfectly classify all training data.) Specify the tree by drawing it, or with a set of nested if-then-else statements. *(15 points)*

Problem 2: EigenFaces (65 points)

In class, we discussed how PCA has been applied to faces, and showed some example results. Here, you'll explore this representation yourself. First, load the data and display a few faces to better understand the data format:

```python
X = np.genfromtxt("data/faces.txt", delimiter=None)  # load face dataset
plt.figure()
# pick a data point i for display
img = np.reshape(X[i,:],(24,24))         # convert vectorized data to 24x24 image patches
plt.imshow( img.T , cmap="gray")         # display image patch; you may have to squint
```

1. Subtract the mean of the face images ($X_0 = X - \mu$) to make your data zero-mean. (The mean should be of the same dimension as a face, 576 pixels.) Plot the mean face as an image. *(5 points)*

2. Use `scipy.linalg.svd` to take the SVD of the data, so that

$$X_0 = U \cdot \mathrm{diag}(S) \cdot V_h$$

   Since the number of faces is larger than the dimension of each face, there are at most 576 non-zero singular values; use the `full_matrices=False` argument to avoid using a lot of memory. As in the slides, then compute `W = U.dot( np.diag(S) )` so that $X_0 \approx W \cdot V_h$. Print the shapes of W and V_h. *(10 points)*

3. For $K = 1, \ldots, 10$, compute the approximation to X_0 given by the first K eigenvectors (or eigenfaces): $\hat{X}_0 = W[:,: K] \cdot Vh[: K,:]$. For each K, compute the mean squared error in the SVD's approximation, `np.mean(` $(X_0 - \hat{X}_0)$`**2 )`. Plot these MSE values as a function of K. *(10 points)*

4. Display the first three principal directions of the data, by computing $\mu+\alpha$ V[j,:] and μ-α V[j,:], where α is a scale factor (we suggest setting α to `2*np.median(np.abs(W[:,j]))`, to match the scale of the data). These should be vectors of length $24^2 = 576$, so you can reshape them and view them as "face images" just like the original data. They should be similar to the images in lecture. *(15 points)*

5. Choose any two faces and reconstruct them using the first K principal directions, for $K = 5, 10, 50, 100$. Plot the reconstructed faces as images. *(10 points)*

6. Methods like PCA are often called "latent space" methods, as the coefficients can be interpreted as a new geometric space in which the data are represented. To visualize this, choose 25 of the faces, and display them as images with the coordinates given by their coefficients on the first two principal components:

```python
idx = ...      # pick some data (randomly or otherwise); an array of integer indices

from sklearn.preprocessing import StandardScaler
coord = StandardScaler().fit_transform(W[:, :2]) # normalize scale of "W" locations
plt.figure(figsize = (10, 10))
for i in idx:
    # compute where to place image (scaled W values) & size
    loc = (coord[i,0],coord[i,0]+0.5, coord[i,1],coord[i,1]+0.5)
    img = np.reshape( X[i,:], (24,24) )           # reshape to square
    plt.imshow( img.T , cmap="gray", extent=loc ) # draw each image
    plt.axis( (-2,2,-2,2) )                       # set axis to a reasonable scale
```

   This plot is a good way to gain intuition for what the PCA latent representation captures. *(15 points)*

Problem 4: Statement of Collaboration (5 points)

It is **mandatory** to include a *Statement of Collaboration* in each submission, that follows the guidelines below. Include the names of everyone involved in the discussions (especially in-person ones), and what was discussed.

All students are required to follow the academic honesty guidelines posted on the course website. For programming assignments in particular, I encourage students to organize (perhaps using Piazza) to discuss the task descriptions, requirements, possible bugs in the support code, and the relevant technical content *before* they start working on it. However, you should not discuss the specific solutions, and as a guiding principle, you are not allowed to take anything written or drawn away from these discussions (no photographs of the blackboard, written notes, referring to Piazza, etc.). Especially *after* you have started working on the assignment, try to restrict the discussion to Piazza as much as possible, so that there is no doubt as to the extent of your collaboration.