Problem Set 4

CS 6375

Due: 11/12/2017 by 11:59pm

Note: all answers should be accompanied by explanations for full credit. Late homeworks will not be accepted.

Problem 1: PCA and Feature Selection (50pts)

In this problem, we will explore ways that we can use PCA for the problem of generating or selecting "good" features.

SVMs and PCA (25pts)

Consider the UCI Spambase data set (attached to this problem set).

- Perform PCA on the training data to reduce the dimensionality of the data set (ignoring the class labels for the moment). What are the top six eigenvalues of the data covariance matrix?
- For each $k \in \{1, 2, 3, 4, 5, 6\}$, project the training data into the best k dimensional subspace (with respect to the Frobenius norm) and use the SVM with slack formulation to learn a classifier for each $c \in \{1, 10, 100, 1000\}$. Report the error of the learned classifier on the validation set for each k and c pair.
- What is the error of the best k/c pair on the test data? How does it compare to the best classifier (with the same possible c choices) without feature selection?
- If you had to pick a value of k before evaluating the performance on the validation set (e.g., if this was not a supervised learning problem), how might you pick it?

PCA for Feature Selection (25pts)

Consider the UCI Spambase data set. To start, train a naïve Bayes classifier on the training data. Note that this data set contains continuous variables. As a result, you should use Gaussian distributions for the conditional probability distributions of each continuous feature conditioned on the value of the label (i.e., $p(x_i|y)$ is a normal distribution in x_i with a possibly different mean and variance for each fixed value of y). What is its accuracy on the test set?

If we performed PCA directly on the training data as we did in the first part of this question, we would generate new features that are linear combinations of our original features. If instead, we wanted to find a subset of our current features that were good for classification, we could still use PCA, but we would need to be more clever about it. The primary idea in this approach is to select

features from the data that are good at explaining as much of the variance as possible. To do this, we can use the results of PCA as a guide. Implement the following algorithm for a given k and s:

1. Compute the top k eigenvalues and eigenvectors of the covariance matrix of the transpose of the data matrix (again omitting the class labels). The transpose of the data matrix is the matrix whose columns are features and whose rows are data points. Denote the top k eigenvectors as $v^{(1)}, \ldots, v^{(k)}$.
2. Define $\pi_j = \frac{1}{k} \sum_{i=1}^{k} {v_j^{(i)}}^2$.
3. Sample s columns independently from the probability distribution defined by π.

- Why does π define a probability distribution?
- For each $k \in \{1, \ldots, 10\}$ and each $s \in \{1, \ldots, 20\}$, report the average test error over 100 iterations of naïve Bayes using only the s selected features (note that there may be some duplicates, so only include each feature once).
- Does this provide a reasonable alternative to naïve Bayes without feature selection on this data set? What are the pros and cons of this approach?

Problem 2: Mixtures of Naive Bayes (50pts)

Consider the problem of fitting a mixture of k naive Bayes classifiers on the UCI Molecular Biology data set, attached here as `bio.data`. Note that this data set contains 60 dimensional features and one label.

That is, fitting a probability model of the form

$$p(x, y) = \sum_{z=1}^{k} \lambda_z p(x, y|\theta_z),$$

where $\theta_1, \ldots, \theta_k$ are the parameters of k naive Bayes models, $p(x, y|\theta_z)$ is the probability of data point x and label y under the z^{th} naive Bayes model, and $\lambda_1, \ldots, \lambda_k \in [0, 1]$ such that $\lambda_1 + \cdots + \lambda_k = 1$ are the mixture probabilities.

1. What is the optimization problem solved by EM for this mixture problem?
2. Derive closed form equations for the E and M step updates of the EM algorithm.
3. Implement the EM algorithm for this problem using the supplied training data with $k = 5$. Using the first 2,126 instances for training and the remaining for testing, run EM to convergence starting from 10 random initializations. Report the average accuracy on the training and test sets.
4. Repeat the previous experiment using MAP estimation instead of MLE by introducing a Dirichlet prior for each set of θ parameters that are required to be probability distributions. You should set the parameters of each Dirichlet distribution uniformly to 2. How does this affect the generalization performance of the method?