Assignment 1

Due: Thursday April 30, 2015

Instructions: Please complete the problems independently and send the softcopy of your answers, together with plots and codes, to `changhong@ict.ac.cn`.

1. (20 points) **Bayes Classifier**

   In this problem you will use Bayes Rule, $p(y|x) = p(x|y)p(y)/p(x)$ to perform classification. Suppose that we observe the following training data, with two binary-valued features x_1 and x_2 and a binary-valued class label y:

   We will classify a small test set:

   (1) What probabilities do you need to create a joint Bayes classifier (i.e., a Bayes classifier that uses the joint distribution over all features), and what are their estimated values given the training data? Use them to classify the test set; report both the class prediction and the estimated probability $p(y = C|x_1, x_2)$ of each class C given the feature values x_1, x_2.

   (2) What probabilities do you need to create a naive Bayes classifier (a Bayes classifier that uses a conditional independence assumption), and what are their estimated values given the training data? Use them to classify the test set and report both the class prediction and estimated probability $p(y = C|x_1, x_2)$ of each class.

x_1	x_2	y
0	0	0
0	0	1
0	0	1
0	0	1
0	1	0
0	1	1
0	1	1
0	1	1
1	0	0
1	0	0
1	0	0
1	1	0
1	1	0
1	1	0
1	1	1
1	1	1

x_1	x_2	y
0	1	1
1	0	1
1	1	0

2. (30 points) **Gaussian Bayes Classifiers**

   Load the iris data[1] into Matlab, and select the first two data features only for the moment. You should first permute the data so that it is not in sorted order.

   In this problem you will explore a simple Gaussian Bayes classifier.

   (0) Plot the iris data X in 2-D space;

[1] You can read about the data at `http://archive.ics.uci.edu/ml/datasets/Iris`.

```
iris = load('iris.txt');
pi = randperm(size(iris,1));
Y = iris(pi,5); X=iris(pi,1:2);
```

(1) For each class, calculate the mean of the data within that class and their covariance matrix. Use the provided gaussPlot(mean,covar,colorString) to plot each of these distributions on top of the data.

(2) Add your calculations to the provided gaussBayesClassify class, and use it along with the class2DPlot function to visualize its predictions.

(3) Try the same thing, but this time use the equalCovariances option to force the model to choose the same covariance for each class. (This will pick the weighted average of the covariances you computed.) Visualize the boundary again. What shape is the decision boundary between each pair of classes?

(4) Even simple classifiers can become powerful given an increasing number of features. You have been provided a class polyClassify that wraps other classifiers and calls them with polynomially computed features (currently only polynomials of each feature individually). You use it like: Test this classifier with increasing polynomial

```
% create a polynomial feature wrapper for a base Gaussian Bayes Classifier
% Gaussian models do not use a constant feature ; set useConstant = 0
learner = polyClassify ( degree , useConstant , gaussBayesClassify());
% Now train and use it as normal
learner = train ( learner , Xtr ,Ytr );
% ...
```

degree $p = 2, 3, 4$. What do you notice about the classification boundaries?

3. (20 points) Consider a (hard margin) support vector machine and the following training

data from two classes:

$$
\begin{aligned}
+1 &: (2,2),(4,4),(4,0) \\
-1 &: (0,0),(2,0),(0,2)
\end{aligned}
$$

(1) Plot these six training points and construct by inspection the weight vector for the optimal hyperplane. In your solution, specify the hyperplane in terms of $\mathbf{w}$ and b such that $w_1x_1 + w_2x_2 + b = 0$. Calculate what the margin is.

(2) What are the support vectors?

4. (30 points) **Run SVMs**

The MNIST dataset is a database of handwritten digits. This problem will apply SVMs to automatically classify digits. [2] We randomly chose a subset of the original MNIST dataset and provide you with two data files, mnist-train.txt, mnist-test.txt. The training set contains 2000 digits, and the test set contains 1000 digits. Each line represents an image of size 28×28 by a vector of length 784, with each feature specifying a grayscale pixel value. The first column contains the labels of the digits, $0 - 9$, the next 28 columns represent the first row of the image, and so on.

(1) Read in mnist-train.txt, mnist-test.txt and transform them into feature vectors. Normalize the feature vectors so that each feature is in the range [-1, 1]. The normalization step can be crucial when you incorporate higher-order features. It also helps prevent numerical difficulties while solving the SVM optimization problem.

(2) Explore the use of linear SVM and SVM with non-linear kernels. You may make use of the widely-used package libSVM Try the default setting of the libSVM, which uses the Gaussian kernel with $\gamma = 1/numoffeatures$ and $C = 1$. Train on the full training set. What is the test error? Then, try different γ and C values to find a model with small cross-validation error. What were the best values that you found? What is the cross-validation error? What is the test error for this setting?

[2] The original dataset can be downloaded at `http://yann.lecun.com/exdb/mnist/`.

Note: your linear classifier will obtain less than 15% test error, and using a Gaussian kernel you will obtain less than 7% test error! Had you used more training data, SVM with Gaussian kernel can get down to 1.4% test error (degree 4 polynomial obtains 1.1% test error). With further fine-tuning (e.g., augmenting the training set by adding deformed versions of the existing training images), a SVM-based approach can obtain 0.56% test error [CMS12]. The state-of-the-art, which uses a convolutional neural network, obtains 0.23% test error [DS01].

References

[CMS12] D. Ciresan, U. Meier, and J. Schmidhuber. Multi-column deep neural networks for image classification. In *CVPR*, 2012.

[DS01] D. Decoste and B. Schölkopf. Training invariant support vector machines. *Machine Learning*, 46(1-3):161–190, 2001.