CSCI 5525 MACHINE LEARNING, Fall 2017,
Prof Schrater
Homework 1

September 27, 2017

1. For data (x, y) with a joint distribution $p(x, y) = p(y|x)p(x)$, the expected loss of a function $f(x)$ to model y using loss function $l(f(x), y)$ is given by

$$E\left[l(f(x), y)\right] = \int_x \left\{ \int_y l(f(x), y)p(y|x) \right\} p(x)dx$$

   (a) (7 points) What is the optimal $f(x)$ when $l(f(x), y) = (f(x) - y)^2$

   (b) (8 points) What is the optimal $f(x)$ when $l(f(x), y) = |f(x) - y)|$, where $|.|$ represents absolute value.

2. (15 Points) For data with joint distribution $p(y, x)$ with y an M-valued discrete target variable. Show that the error rate of the Bayes optimal classifier for class $y = C_j$ is given by:

$$err\left[y = C_j\right] = \sum_{i=1:M} \int_{x \in \mathcal{R}_i} p(C_j|x)p(x)dx - \int_{x \in \mathcal{R}_j} p(C_j|x)p(x)dx$$

   where $\mathcal{R}_k$ is the region of x where $p(y_k|x) > p(y_i|x) \forall i \neq k$

Please show **all** your work! Answers to the first two problems without supporting work will not be given credit.

Programming assignments: The next two problems involve programming. We will be considering two datasets for these assignments:

   (a) *Boston Housing*: The Boston housing dataset comes prepackaged with scikit-learn. The dataset has 506 points, 13 features, and 1 target (response) variable. You can find more information about the dataset here:
   http://www.cs.toronto.edu/ delve/data/boston/bostonDetail.html
   Where the first 13 variables are attributes, and the median home value (variable 14) is the target.

```
from sklearn.datasets import load_boston
boston = load_boston()
```

   While the original dataset is for a regression problem, we will create two classification datasets for the homework. Note that you only need to modify the target t to create these classification datasets, and the attribute data X should not be changed.

      i. *Target = HomeVal50* Create a binary target variable by thresholding at the 50th percentile of the target scores. Note by construction the proportions of the two classes are equal ($\pi = 1/2$).

   (b) *Handwritten Digits* This is a version of the classic handwritten digit dataset. The dataset has 1797 points, 64 features, and 10 classes corresponding to ten numbers 0, 1, . . . , 9.

```
from sklearn.datasets import load_digits
digits = load_digits()
```

Besides loading from scikit, these two datasets are also provided on Moodle course page in csv format. The last column shows target values. A target value is either discrete (e.g., class label) or continuous (e.g., house price) and it is what we are trying to predict or classify. All other columns are data values. Each row is an input point.

3. (30 points) Use Fishers linear discriminant analysis (LDA) for this problem. The problem will require you to explain, and implement the key equations for the discrimant directions $\vec{w}_k$, answer questions about these equations and use the key equations from Gaussian generative modeling to code a classifier. Prediction error of the resulting classifier needs to be assessed by training and evaluating the following classifiers using 10-fold cross-validation:

   (a) (10 points) For the HomeVal50 dataset, apply LDA in the general case, i.e., both SB and SW computed from the data, to project data to R (one dimension). Histogram the projected points. Are the data well-modeled by a linear discriminant? Justify your answer carefully, simple yes or no yields no credit.

   (b) (5 points) Using LDA, can you project the Boston50 data to R2 (two dimensions)? Clearly justify your answer with technical details. (A simple yes/no answer will receive zero credit.)

   (c) (15 points) For the Digits dataset, apply LDA in the general case, i.e., both $\mathbf{S_B}$ and $\mathbf{S_W}$ computed from the data, to project data to $\mathcal{R}^2$ (two dimensions) by using two eigenvectors, followed by bivariate Gaussian generative modeling to do 10-class classification, i.e., by estimating and using class priors π_k and parameters $(\mu_k, \mathbf{C}_k$, $k = 1, \ldots, 10$.
   You will have to submit:

      i. **Summary of Methods and Results** Briefly describe the approaches in (a), (b) and (c) above, along with relevant equations. Also, report the training and test set error rates and standard deviations from 10-fold cross validation for the methods on the dataset in part (c).

      ii. **Code for each algorithm implemented:** For part (a), you will have to submit code for LDA1dProjection(filename,num crossval)(main file). This main file has input: a filename including extension (exactly the same as your downloaded file) and absolute path (e.g., in Unix: /home/mywork/boston.csv). The content of input data must be exactly the same as downloaded file, or load from scikit (Juypter notebook) and generates: a plot of the histograms of projected points, separately for both classes (20 bins).
      For part (b), you will have to submit code for LDA2dGaussGM(filename,numcrossval), with inputs: (1) filename as before (2) the number of folds for cross-validation, and output: (1) the training and test set error rates and standard deviations printed to the terminal (stdout).

4. **(40 points)** The goal is compare simple generative and discriminative classifiers on the learning curves using the HomeVal50 and Digits datasets. For background, see "On Discriminative vs. Generative Classifiers: A comparison of logistic regression and naive Bayes" by A. Ng and M. Jordan, available in moodle. You will train and evaluate two classifiers on both datasets:

   (a) **(20 points)** Logistic regression (LR), and

   (b) **(20 points)** Naive-Bayes with marginal Gaussian distributions (GNB)

   (a) **Evaluation** Evaluation will be done using 10 random class-specific 80-20 train-test splits, i.e., for each class, pick 80% of the data at random for training, train a classifier using training data from all classes, use the remaining 20% of the data from each class as testing, and repeat this process 10 times. We will be doing a learning curve, similar to the Ng-Jordan paper.

   (b) **Code** For logistic regression, you will have to implement the Iterately Reweighted Least Squares algorithm as a function with the form

   ```
   logisticRegression(filename, num splits, train percent)
   ```

   This main file has input: (1) a filename including extension, (2) the number of 80-20 train-test splits for evaluation, (3) and a vector containing percentages of training data to be used for

Cont.

training (use [10 25 50 75 100] for the plots), and output: (1) test set error rates for each training set percent printed to the terminal (stdout). The test set error rates should include both the error rates for each split for each training set percentage as well as the mean of the test set error rates across all splits for each training set percentage (print the mean error rates at the end).
For *naive Bayes*, you will have to submit code for

```
naiveBayesGaussian(filename, num splits, train percent)
```

that implements feature wise generative Gaussian Modeling, with all other guidelines staying the same.

(c) **Plots**: Your plots will be based on 10 random 80-20 train-test splits. For each split, we will always evaluate results on the same test set (20% of the data), while using increasing percentages of the training set (80% of the data) for training. In particular, we will use the following training set percentages: [10 25 50 75 100], so that for each 80-20 split, we use 10%, 25%, all the way up to 100% of the training set for training, and always report results on the same test set. We will repeat the process 10 times, and plot the mean and standard deviation (as error bars) of the test set errors as a function of training set percentage.

(d) **Summary:** You will have to submit a summary of methods and results

i. Briefly describe the approaches for Logistic Regression and Naive Bayes, including key equations modeling the data log likelihood and the (iterative) equations for parameter estimation.

ii. For each dataset and method, create a plot of the test set error rate illustrating the relative performance of the two methods with increasing number of training points. The plots will be similar in spirit to Figure 1 in the Ng-Jordan paper, along with error-bars with standard deviation of the errors.

Additional instructions: Code can only be written in Python or Matlab; no other programming languages will be accepted. One should be able to execute all programs from matlab/python command prompt or the code can be a jupyter notebook. Your code must be runnable on a CSE lab machine.

Please specify instructions on how to run your program in the README file. Information on the size of the datasets, including number of data points and dimensionality of features, as well as number of classes can be readily extracted from the dataset text file. Each function must take the inputs in the order specified in the problem and display the textual output via the terminal. The input data file for your function must be exactly the same as the original downloaded file, which will be used as input for grading. For each part, you can submit additional files/functions (as needed) which will be used by the main file.

For this assignment note whether you are allowed to use scikit code. If indicated, you cannot use machine learning libraries such as those available from scikit-learn for learning the models or for cross-validation. However, you may always use libraries for basic matrix computations. Put comments in your code so that one can follow the key parts and steps in your code.

Cont.

Submission Instructions:
Follow the rules strictly. If we cannot run your code, you will not get any credit.
Things to submit:

1. hw1.pdf: A document which contains the solution to Problems 1, 2, 3, and 4 which including the summary of methods and results.
2. LDA1dProject: Code for Problem 3.
3. logisticRegression and naiveBayesGauss: Code for Problem 4.
4. README.txt: README file that contains your name, student ID, email, instructions on how to compile (if necessary) and run your code, any assumptions you are making, and any other necessary details.
5. Any other files, except the data, which are necessary for your code.

The End.