CSCI 5525: Machine Learning (Fall'15)

Homework 1, Due 10/03/15 11:55pm

1. **(15 points)** The expected loss of a function $f(x)$ in modeling y using loss function $\ell(f(x), y)$ is given by

$$E_{(x,y)}[\ell(f(x), y)] = \int_x \int_y \ell(f(x), y) p(x, y) dy dx = \int_x \left\{ \int_y \ell(f(x), y) p(y|x) dy \right\} p(x) dx \ .$$

   What is the optimal $f(x)$ when $\ell(f(x), y) = (f(x) - y)^2$?

2. **(15 points)** Consider a 2-class classification problem with features $\mathbf{x} \in \mathbb{R}^d$ and labels $y \in \{-1, +1\}$ and $(\mathbf{x}, y) \sim D$, where D is a fixed (but unknown) distribution on $\mathbb{R}^d \times \{-1, +1\}$. Assume $p(y = +1) = p(y = -1) = 1/2$, and recall that the Bayes classifier is given by

$$f^*(\mathbf{x}) = \begin{cases} +1 \ , & \text{if } P(1|\mathbf{x}) > 1/2 \\ -1 \ , & \text{otherwise.} \end{cases}$$

   The error-rate of the Bayes classifier is $L(f^*) = P(f^*(\mathbf{x}) \neq y) = E_{(\mathbf{x},y)\sim D}[\mathbb{1}(f^*(\mathbf{x}) \neq y)]$, where $\mathbb{1}$ is the indicator function (for 0-1 loss), and $L(\cdot)$ is the expected 0-1 loss or true error rate. For any classifier $f : \mathbb{R}^d \mapsto \{-1, +1\}$, show that $L(f^*) \leq L(f)$.

3. **(35 points)** Consider the `MNIST-1378` dataset for 4-class classification of hand written digits: 1,3,7, and 8.[1] Train and evaluate the following classifiers on the dataset using 10-fold cross-validation:

   (i) Fisher's linear discriminant in the general case, i.e., both S_B and S_W computed from the data, followed by multi-variate Gaussian generative modeling of each class in the projected space.

   (ii) Least squares linear discriminant using a bit-vector representation of the classes, e.g., since this is a 4-class problem, $\mathbf{y} = [1\ 0\ 0\ 0]$ for a point belonging to class 1 (say, number '1'), $\mathbf{y} = [0\ 1\ 0\ 0]$ for a point belonging to class 2 (say, number '3'), and so on.

   You will have to submit (a) **summary of methods and results** report and (b) **code** for each algorithm:

   (a) **Summary of methods and results**: Briefly describe the approaches in (i) and (ii) above, along with equations for parameter estimation. Also, report the training and test set error rates and standard deviations from 10-fold cross validation of each method on the dataset.

[1] This is a subset of the widely studied MNIST dataset: `http://yann.lecun.com/exdb/mnist/`.

(b) **Code**: For part (i), you will have to submit code for `Fisher(filename, num_crossval)` (main file). This main file has **input**: (1) a filename containing the dataset and (2) the number of folds for cross-validation as arguments, and **output**: (1) the training and test set error rates and standard deviations printed to the terminal (stdout). The function *must* take the inputs in this order and display the output via the terminal. The filename will correspond to a plain text file for a dataset, with each line corresponding to a data point: the first entry will be the label, i.e., a 1, 3, 7, or 8, corresponding to the hand-written number,and the rest of the entries will be feature values of the data point.

For part (ii), you will have to submit code for `SqClass(filename, num_crossval)` (main file), with all other guidelines staying the same. For each part, you can submit additional files/functions (as needed) which will be used by the main file. Put comments in your code so that one can follow the key parts and steps in your code.

4. **(35 points)** The goal is to evaluate the results reported in the paper "On Discriminative vs. Generative Classifiers: A comparison of logistic regression and naive Bayes" by A. Ng and M. Jordan, using the `spam` dataset.[2] You will train and evaluate two classifiers: (i) Logistic regression, and (ii) Naive-Bayes with marginal univariate Gaussian distributions.[3] Evaluation will be done using 100 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, and use the remaining 20% of the data from each class to test it, and repeat this process 100 times.

   You will have to submit (a) **summary of methods and results** report and (b) **code** for each algorithm:

   (a) **Summary of methods and results**: Briefly describe the approaches in (i) and (ii) above, along with (iterative) equations for parameter estimation. Clearly state which method you are using for logistic regression.[4] 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 (see instructions below). 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.

   Instructions for plots: Your plots will be based on 100 random 80-20 train-test splits. For each split, we will always evaluate results on the same test set (20% of the data), while increasing the number of training examples, as specified by the vector of training set percentages given as input. If the training set percentages are [5 10 15 20 25 30], then for each 80-20 split, we use 5%, 10%, all the way up to 30% of the training set for training, and always report results on the same test set. We will repeat the process 100 times, and plot the mean and standard deviation of the test set errors for different training set percentages.

   (b) **Code**: For logistic regression, you will have to submit code for `logisticRegression(filename, num_splits, train_percent)`. This main file has **input**: (1) a filename containing the

[2] More information about this dataset can be found here: `https://archive.ics.uci.edu/ml/datasets/Spambase`.

[3] As discussed in class, for data point $\mathbf{x}$ where $p(\mathbf{x}|C_k) = \prod_{i=1}^{d} p(x_i|C_k)$, for each feature x_i and class C_k, we model marginal distribution $p(x_i|C_k) \sim \mathcal{N}(\mu_{ik}, \sigma_{ik}^2)$ and should estimate μ_{ik} and σ_{ik}^2 from data.

[4] See notes by Minka, in Moodle.

dataset, (2) number of 80-20 train-test splits for evaluation, (3) and a vector containing percentages of training data to be used for training, and **output**: (1) test set error rates for each training set percent printed to the terminal (stdout). The function *must* take the inputs in this order and display the output via the terminal. The filename will correspond to a text file for a dataset, with each line corresponding to a data point: the first entry will be the label and the rest of the entries will be feature values of the data point.

Put comments in your code so that one can follow the key parts and steps in your code. For training, please use the training set percentage vector [5 10 15 20 25 30].

For naive Bayes, you will have to submit code for `naiveBayesGaussian(filename, num_splits, train_percent)` (main file), with all other guidelines staying the same.

Additional instructions: Code can only be written in Matlab, Python, or Java; no other programming languages will be accepted. All programs must be able to be executed from the terminal command prompt. Please specify instructions on how to run your program in a 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.

Instructions

Follow the rules strictly. If we cannot run your functions, you get 0 points.

- **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. Fisher and SqClass: Code for Problem 3.
  3. logisticRegression and naiveBayesDiscrete: 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.