10-601A/SV: Introduction to Machine Learning (F15)
HOMEWORK 1
MLE/MAP, Naive Bayes and Perceptron

OUT: Sept 18, 2015

QNA DUE: Oct 2, 2015 11:59 PM EDT

Other Parts DUE: Oct 5, 2015, 11:59 PM EDT

START HERE: Instructions

- **Late homework policy**: QNA part is due Oct 2, 2015 at 11:59 PM EDT. Other parts including Problem 2 (Gaussian Bayes Classifier) and Problem 3 (Programming) are due Oct 5, 2015 at 11:59 PM EDT. You have five late days to use throughout the semester. Once the allowed late days are used up, each additional day (or part of a day) will subtract 1 from the normalized score for the assignment. View the full homework policy on Piazza. You may also find additional clarification on this homework somewhere on Piazza, if any.
- **Collaboration policy**: You may discuss with other students to tackle any of the assignments; however, you must write the solutions completely on your own. Please include a section that elaborates on how you have collaborated with other students, and what kind of external resources you have made use of substantially (e.g., "Jane explained to me what is asked in Question 3.4" or "I found an explanation of conditional independence on page 17 of Mitchell's textbook"). Any violation of academic integrity will result in severe consequences to the maximum extent permitted by university policies.
- **Programming**: All programming assignments will be graded automatically on Autolab using **Octave 3.6.4**. You may develop your code in your favorite IDE, but please make sure that it runs as expected on Octave before submitting. The code which you write will be executed remotely against a suite of tests, and the results are used to automatically assign you a grade. To make sure your code executes correctly on our servers, you should avoid using libraries which are not present in the basic Octave install.
- **Submitting your work**: For QNA part, please use the provided link and directly submit your answers on that webpage. For other parts, please submit electronically through Autolab. You should be able to sign in with your Andrew ID at `https://autolab.cs.cmu.edu/`; if not, contact the teaching assistants for access privileges.
  - Start by downloading the submission template. The template consists of directory with placeholders for your writeup (`problem2.pdf`, `problem3.pdf`), and other files for the programming parts of the assignment. *Do not modify the structure of these directories or rename these files.*
  - **IMPORTANT**: When you download the template, you should confirm that the autograder is functioning correctly by compressing and submitting the directory provided. This should result in a grade of zero for all programming questions, and an unassigned grade (- or 0) for the written questions.

- **Writeup**: Replace the placeholders with your actual writeup. Make sure to keep the expected file names (e.g. `problem2.pdf`), and to submit one PDF per problem. We recommend you prepare your solutions using LaTeX but a scanned handwritten solution is also OK. Please make sure that your handwriting is clear and legible, otherwise it won't be graded.
- **Code**: For each programming sub-question you will be given a single function signature. You will be asked to write a single Octave function which satisfies the signature. Submission template that we provide contains stubs for each of the functions you need to complete.
- **Putting it all together**: Once you have provided your writeup and completed each of the function stubs, compress the top level directory *as a tar file* and submit to Autolab online (URL above). You may submit your answers as many times as you like. You will receive instant feedback on your autograded problems, and your writeups will be graded by the instructors once the submission deadline has passed.

* * *

1 QNA Questions [Ru, Shi - 36 pts]

Reminder: Please use the following link for questions and directly submit your answers on that webpage. Note that for each question, although you can submit several times, you will get full credit only if your answer is correct *for the first time*.

https://qna-app.appspot.com/view.html?aglzfnFuYS1hcHByGQsSDFF1ZXN0aW9uTGlzdBiAgIDA2Y2MCgw

2 Gaussian Bayes Classifier [Shi - 20 pts]

Reminder: Please prepare your answers to this problem as a PDF file and replace with `problem2.pdf` in the submission template.

This problem will help you understand the posterior distribution and Bayes classifier in some detail.

2.1 Posterior Distribution: Conjugate Prior (10 pts)

In class we have learnt about conjugate prior, which means the prior $p(\theta)$ and the posterior $p(\theta|\mathcal{D})$ have same forms. Conjugate prior is quite useful as it can greatly simplify our calculations when finding the posterior. Also, conjugate prior may give us some intuition about how a prior distribution is updated by a likelihood function. In this problem, we will use univariate Gaussian distribution as an example of a likelihood function and see how the conjugate priors work.

Consider a dataset $\mathcal{D}$ containing N i.i.d samples $x_i \sim \mathcal{N}(\mu, \sigma^2)$. For simplicity we will denote σ^2 as $\frac{1}{\tau}$ (which means $\tau = \sigma^{-2}$). Here τ is called "precision". Then the univariate Gaussian distribution is given by equation 1,

$$p(x_i|\mu,\tau) = \sqrt{\frac{\tau}{2\pi}} \exp\left(-\frac{\tau}{2}(x_i-\mu)^2\right) \sim \mathcal{N}(\mu,\tau^{-1}) \tag{1}$$

Suppose the mean μ and the precision τ are both unknown to us. As we will see later, the conjugate prior for this case is **Normal-Gamma** distribution, which is given by equation 2,

$$\begin{aligned} NG(\mu,\tau|\mu_0,\nu_0,\alpha_0,\beta_0) &\stackrel{\text{def}}{=} \mathcal{N}(\mu|\mu_0,(\nu_0\tau)^{-1})Ga(\tau|\alpha_0,\beta_0) \\ &= \frac{\beta_0^{\alpha_0}}{\Gamma(\alpha_0)}\left(\frac{\nu_0}{2\pi}\right)^{\frac{1}{2}} \tau^{\alpha_0-\frac{1}{2}} \exp\left(-\frac{\tau}{2}\left[\nu_0(\mu-\mu_0)^2+2\beta_0\right]\right) \end{aligned} \tag{2}$$

Here $\Gamma(\cdot)$ is the Gamma function. In Normal-Gamma distribution, the mean μ has a Gaussian prior with mean μ_0 and precision $\nu_0\tau$ and the precision τ has a Gamma prior with parameters α_0 and β_0.

TASK 1 (10 pts)

Derive the posterior distribution $p(\mu,\tau|\mathcal{D})$ for the univariate Gaussian case (both the mean μ and the precision τ are unknown) using the conjugate prior $NG(\mu,\tau|\mu_0,\nu_0,\alpha_0,\beta_0)$.
[Hint 1: You may find the following equation useful,

$$\sum_{i=1}^{N}(x_i-\mu)^2 = N(\mu-\overline{x})^2 + \sum_{i=1}^{N}(x_i-\overline{x})^2 \tag{3}$$

Hint 2: As the posterior and conjugate prior have same forms, the posterior will still follow the Normal-Gamma distribution $p(\mu,\tau|\mathcal{D}) \sim NG(\mu,\tau|\mu_N,\nu_N,\alpha_N,\beta_N)$. Your task is to find out how these four parameters $\mu_N,\nu_N,\alpha_N,\beta_N$ are updated.]

2.2 Decision Boundary: Linear or Non-Linear? (10 pts)

In class we have seen that the decision boundary for Gaussian Bayes classifier may vary in different cases (as shown in Figure 1). Here for simplicity we consider 2 classes, we will prove that the decision boundary could be linear if covariance matrix is shared by these two classes.

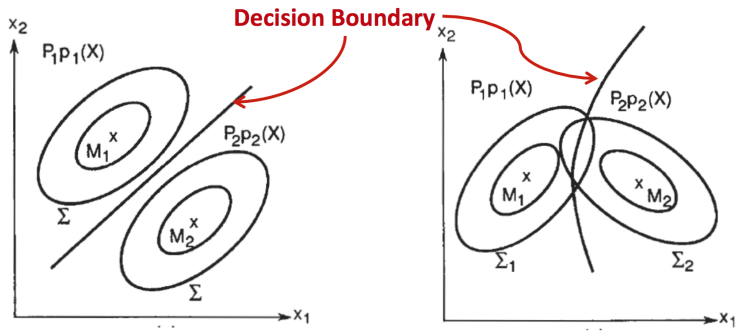

Figure 1: Decision Boundary for Gaussian Bayes Classifier

Consider a dataset $\mathcal{D}$ containing N i.i.d samples which are d dimensional ($\mathbf{x}_i \in \mathbb{R}^d$). There are 2 classes (denote as c and c') and the class prior follows the categorical distribution $p(y) \sim \mathsf{Categorical}(\boldsymbol{\pi})$. Note that in Gaussian Bayes classifier we assume the conditional distribution for each class is a multivariate Gaussian, e.g. $p(\mathbf{x}|y=c) \sim \mathcal{N}(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$, given by equation 4,

$$p(\mathbf{x}|y=c) = \frac{1}{\sqrt{(2\pi)^d|\boldsymbol{\Sigma}_c|}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_c)^T\boldsymbol{\Sigma}_c^{-1}(\mathbf{x}-\boldsymbol{\mu}_c)\right) \tag{4}$$

By Bayes rule, we can easily write out the following posterior distribution,

$$p(y=c|\mathbf{x}) \propto \pi_c|\boldsymbol{\Sigma}_c|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_c)^T\boldsymbol{\Sigma}_c^{-1}(\mathbf{x}-\boldsymbol{\mu}_c)\right) \tag{5}$$

TASK 2 (10 pts)

1. Consider 2 classes, show that if $\boldsymbol{\Sigma}_c = \boldsymbol{\Sigma}_{c'}$, then the decision boundary for Gaussian Bayes classifier is linear.
2. Give out one condition that results non-linear decision boundary and briefly explain why.

3 Programming: Implementing Gaussian Naive Bayes and Perceptron [Di, Dawei - 44 pts + 10 pts]

Professor Gordon has recently revealed to us that he was actually a botanist. He claimed that throughout the past few years he has been secretly collecting flower samples around his office, and at the time of this writing his assortment consists of the following species: [1]

(a) Setosa (b) Versicolor (c) Virginica

Apparently most of the teaching assistants were shocked to find out about Professor Gordon's versatility. As our mental status gradually stablized, he explained the true intention behind his revelation:

"I noticed that there were three species of Iris near my office: Iris Setosa, Iris Versicolor, and Iris Virginica. For each sample I have carefully measured the width and the length of the sepal,

[1] Actually, this dataset was collected by R.A. Fisher with his 1936 paper *The use of multiple measurements in taxonomic problems*, but this in no way invalidates Professor Gordon's passion on botany.

and labeled it with an numerical identifier that denotes its species. However, some of the flowers looked extremely similar, and I simply could not tell them apart. **It would be great if we have some classifiers that could learn from the dataset that I am confident with, and make some educated guess on what the species of the rest of the samples might be.** I have plotted the dataset on a Cartesian coordinate system, with the x and y axes corresponding to the length and the width of the sepals, and colors designating different species. ”

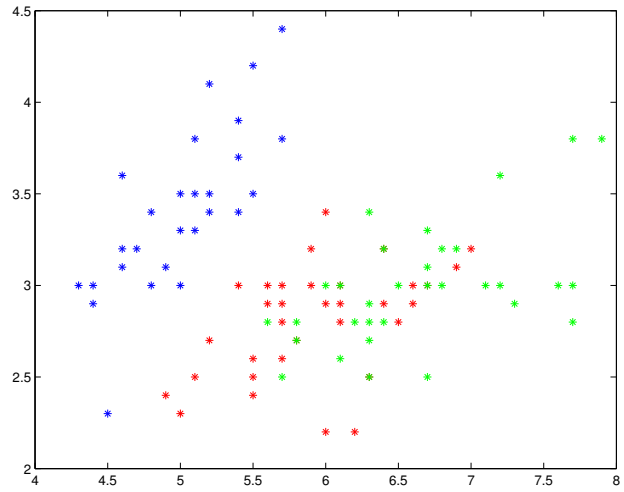

Figure 2: Petals length and sepals width projection. Three colors are used for discernibility. Blue: Iris-setosa; Red: Iris-versicolor; Green: Iris-virginica

Submission template that we provide consists of three Octave function files, namely `nb.m`, `lp.m`, and `kp.m`, which you will have to fill-in in the subsequent tasks; and the datasets you are going to use. The following section depicts the details of the programming problem.

3.1 Gaussian Naive Bayes Classifier (15 pts)

Gaussian Naive Bayes Classifier is a classifier based on applying Bayes' theorem with strong (naive) independence assumptions between the features. As explained in the lecture, it is a popular (baseline) method for text categorization, and it can also be applied in this scenario. At the initial step, you decided to start with something simple, with the assumption that in either dimension the samples are Gaussian distributed. That is, for a class c we have:

$$X_c \sim \mathcal{N}(\mu_{X,c}, \sigma^2_{X,c})$$
$$Y_c \sim \mathcal{N}(\mu_{Y,c}, \sigma^2_{Y,c})$$

where $\mu_{X/Y,c}, \sigma^2_{X/Y,c}$ denotes the mean and the variance in either dimension for samples in class c, respectively. E.g., the mean and the variances of x coordinates of the data points in class c can

be calculated as following:

$$\mu_{X,c} = \frac{1}{n}\sum_{i=1}^{n} x_{c,i}$$
$$\sigma^2_{X,c} = \frac{1}{n}\sum_{i=1}^{n}(x_{c,i} - \mu_{X,c})^2 \tag{6}$$

With these assumptions, you are able to train your Naive Bayes for classification on the training dataset with format $\langle x_i, y_i, l_i \rangle$. Suppose your training dataset has n points, m of them are with the same label c. The prior can be calculated as following:

$$p(c) = \frac{m}{n} \tag{7}$$

The likelihood for a testing data point e obeys the following distribution:

$$p(x_e|c) \sim \mathcal{N}(\mu_{X,c}, \sigma^2_{X,c})$$
$$p(y_e|c) \sim \mathcal{N}(\mu_{Y,c}, \sigma^2_{Y,c})$$

Based on the conditional independence assumption, we have the following:

$$p(x_e, y_e|c) = p(x_e|c)p(y_e|c) \tag{8}$$

Finally, according to the Bayes' theorem, we have:

$$p(c|x_e, y_e) = \frac{p(x_e, y_e|c)p(c)}{p(x_e, y_e|c)p(c) + p(x_e, y_e|\bar{c})p(\bar{c})} \tag{9}$$

where $\bar{c}$ denotes the class opposite to class c.

In this assignment, you will be implementing a *Gaussian Naive Bayes Classifier* to discriminate the class Iris-setosa (class 1: Blue) and Iris-versicolor (class 2: Red). **Note that the class label $l_i \in \{1, 2, 3\}$, which is slightly different from the content in lecture.**

TASK 1 (15 pts)

Write a function with the signature

```
function [v] = nb(d, t)
```

that finds the predicted labels v of the testing dataset.

DATASETS:

- `iris_training.csv`: $n \times 3$ matrix.

  Given a data point i, the 1st column value x_i denotes the sepal length, the 2nd column value y_i denotes the sepal width, the 3rd column value l_i denotes the labels, $l_i \in \{1, 2, 3\}$, for $i = 1, 2, \ldots, n$;
- `iris_testing_nb.csv`: an $m \times 2$ matrix, the 1st column value x_i denotes the sepal length, the 2nd column value y_i denotes the sepal width.

INPUT:

- `d`: reference to the $n \times 3$ training dataset;
- `t`: reference to the $m \times 2$ testing dataset.

OUTPUT:

- `v`: $m \times 1$ column vector for prediction. $v_i \in \{1, 2\}$ for $i = 1, 2, \ldots, m$;

FILE: `nb.m`

3.2 Perceptron Classifier (39 pts)

The perceptron is an algorithm for supervised learning of binary classifiers. Combined with kernel trick, it can be extended to solve non-linear classification problems. In this section, you will use the same datasets as above to implement two perceptron algorithms to distinguish (class 2: Red) and (class 3: Green).

3.2.1 Linear Perceptron (15 pts)

The linear perceptron algorithm is shown in Algo. 1. In this assignment, your task is to implement this algorithm.

Algorithm 1 Perceptron algorithm

Argument:
$E = \{e_1, e_2, ..., e_n\}$ where $e_i = \langle x_i, y_i, 1\rangle$ where $x_i, y_i \in$ iris_training
$L = \{l_1, l_2, ..., l_n\}$ where $l_i \in$ labels in iris_training.
α step size. $\alpha = 0.005$.

Initialize $w = [0, 1, 0]$.
for each data point $\{e_i, l_i\}$ **do**
    predict sign$(w^T e_i)$
    if sign$(w^T e_i)$ $!= l_i$ **then**
        $w = w + l_i e_i \alpha$
    end if
end for

TASK 2 (15 pts)

Write a function with the signature

```
function [v] = lp(d, t)
```

that finds the predicted labels v of the testing dataset through the implementation of Algo.1.

DATASETS:

- `iris_training.csv`: $n \times 3$ matrix.

  Given a data point i, the 1st column value x_i denotes the sepal length, the 2nd column value y_i denotes the sepal width, the 3rd column value l_i denotes the labels, $l_i \in \{1, 2, 3\}$, for $i = 1, 2, \ldots, n$;
- `iris_testing_perceptron.csv`: an $m \times 2$ matrix, the 1st column value x_i denotes the sepal length, the 2nd column value y_i denotes the sepal width.

INPUT:

- `d`: reference to the $n \times 3$ training dataset;
- `t`: reference to the $m \times 2$ testing dataset.

OUTPUT:

- `v`: $m \times 1$ column vector for prediction. $v_i \in \{2, 3\}$ for $i = 1, 2, \ldots, m$;

FILE: `lp.m`

3.2.2 Featurized / Kernel Perceptron (OPTIONAL: 10 pts)

You may find that the linear perceptron algorithm does not fit this dataset very well. In fact, with proper modification on perceptron algorithm, you will obtain a non-linear classification algorithm. You can choose to do nonlinear transformation of the data, or perform kernel tricks to the algorithm. Presumably you would see precision improvement in this task.

TASK 3 (10 pts)

Write a function with the signature

```
function [v] = kp(d, t)
```

that finds the predicted labels v of the testing dataset through the implementation of the algorithm you choose.

DATASETS:

- `iris_training.csv`: $n \times 3$ matrix.

  Given a data point i, the 1st column value x_i denotes the sepal length, the 2nd column value y_i denotes the sepal width, the 3rd column value l_i denotes the labels, $l_i \in \{1, 2, 3\}$, for $i = 1, 2, \ldots, n$;
- `iris_testing_perceptron.csv`: an $m \times 2$ matrix, the 1st column value x_i denotes the sepal length, the 2nd column value y_i denotes the sepal width.

INPUT:

- `d`: reference to the $n \times 3$ training dataset;
- `t`: reference to the $m \times 2$ testing dataset.

OUTPUT:

- `v`: $m \times 1$ column vector for prediction. $v_i \in \{2, 3\}$ for $i = 1, 2, \ldots, m$;

FILE: `kp.m`

3.2.3 Sense Making (14 pts)

Now you are going to analyze the differences between the models you have created, and perform a brief summary on what you found. Create a `problem3.pdf` file, and put the following in it:

1. 1 plot that combines the scatter plot of testing dataset and the boundaries you obtained in the perceptron algorithm(s).
2. Comparison on precision of the two (or three, if you choose to do the optional perceptron) classifiers, and summarization of what you find.

Reminder: Put all three files (`nb.m`, `lp.m`, `kp.m`) in the submission template and replace `problem3.pdf` with your answers to Section 3.2.3; then, proceed to Autolab and submit your *tar file* under the appropriate assessment. It will take several seconds for the autograder to run your code. If your code exits without any problems, grades will be assigned for each individual task; if you are getting zero points for some of the tasks, a runtime error might have occurred. Check the job status for details. In this assignment, we will use a "hidden" dataset to test your algorithms, so please avoid explicitly putting a vector of labels to your solution.

You might submit as many times as you would like. Your final grade for this programming assignment will be *the grade for your last submission before the due date.*