E0 270 Machine Learning

Assignment 1 - Solutions

Questions - Answers

1. **Probability Review**. Suppose that X and Y are discrete random variables. Their joint *pmf* function P(X,Y) is given by the table below. Show how to compute P(X), P(Y), P(X/Y) and P(Y/X) (use conditional probabilities). **[5 Mark]**

		X		
		x1	x2	x3
Y	y1	0.1	0.2	0.1
	y2	0.1	0.2	0.3

Solution:
P(X) $= \sum_{i=1}^{n} P(X, y_i)$
P(x_1) = P(x_1, y_1)+ P(x_1, y_2) = 0.1+0.1 = 0.2
P(x_2) = P(x_2, y_1)+ P(x_2, y_2) = 0.4
P(x_3) = P(x_3, y_1)+ P(x_3, y_2) = 0.4

P(Y) $= \sum_{i=1}^{n} P(x_i, Y)$
P(y_1) = P(x_1, y_1)+ P(x_2, y_1) +(x_3, y_1) = 0.1+0.2+0.1= 0.4
P(y_2) = P(x_1, y_2)+ P(x_2, y_2) +(x_3, y_2) = 0.6

Conditional probabilities:
$P(x|Y) = \frac{P(x,Y)}{P(Y)}$
e.g.
$P(x_1|y_1) = \frac{P(x_1,y_1)}{P(y_1)} = \frac{0.1}{0.4} = 0.25$

Table 1: Conditional Probabilities

x/y	x1	x2	x3
y1	0.25	0.5	0.25
y2	0.166	0.33	0.5

$P(y|X) = \frac{P(X,y)}{P(X)}$
e.g.
$P(y_1|x_1) = \frac{P(x_1,y_1)}{P(x_1)} = \frac{0.1}{0.2} = 0.5$

Table 2: Conditional Probabilities

y/x	x1	x2	x3
y1	0.5	0.5	0.25
y2	0.5	0.5	0.75

2. Given a 2-class classification problem in 2 dimensional space. What is the Bayes decision boundary for this problem. **[5 Mark]**

   $p(x|w_1) \sim N(0, I)$
   $p(x|w_2) \sim N([1,1]^T, I)$
   and $P(w_1) = P(w_2) = 0.5$ **Solution:**

   Bayes decision boundary :

   $P(w_1|x) > P(w_2|x)$

   $\frac{P(x|w_1)P(w_1)}{P(x)} > \frac{P(x|w_2)P(w_2)}{p(x)}$

   as, $P(w_1) = P(w_2)$, bayes decision boundary is given by
   $P(x|w_1) > P(x|w_2)$
   Plug in the gaussian probabilities, therefore $P(x|w1) = \frac{1}{2*\pi} * exp^{-0.5*({x_1}^2+{x_2}^2)}$
   and, $P(x|w2) = \frac{1}{2*\pi} * exp^{-0.5*((x_1-1)^2+(x_2-1)^2)}$
   take log on both sides and simplify the equation
   $x_1 + x_2 < 1$

   therefore, the bayes decision boundary is $x_1 + x_2 = 1$
   where x is in class 1 is $x_1 + x_2 < 1$ else in class 2

3. **[10 Mark]** Consider a cost-sensitive binary classification task in which misclassifying a positive instance as negative incurs a cost of $c \in (0, 1)$, and misclassifying a negative instance as positive incurs a cost of $1 - c$. This can be formulated as a learning task with instance space $\mathcal{X}$, label and prediction spaces $\mathcal{Y}, \widehat{\mathcal{Y}} \in \{\pm 1\}$, and cost-sensitive loss $\ell_c : \{\pm 1\} \times \{\pm 1\} \to \{0, c, 1 - c\}$ defined as

$$\ell_c(y, \widehat{y}) = \begin{cases} c & \text{if } y = -1 \text{ and } \widehat{y} = 1 \\ 1 - c & \text{if } y = 1 \text{ and } \widehat{y} = -1 \\ 0 & \text{if } \widehat{y} = y. \end{cases}$$

   Let D be a joint probability distribution on $\mathcal{X} \times \{\pm 1\}$, with marginal distribution μ on $\mathcal{X}$ and conditional label probabilities given by $\eta(x) = \mathbf{P}(y = 1|x)$, and for any classifier $h : \mathcal{X} \to \{\pm 1\}$, define

$$\mathrm{er}_D^c[h] = \mathbf{E}_{(x,y)\sim D}\left[\ell_c(y, h(x))\right].$$

   Derive a Bayes optimal classifier in this setting, i.e. a classifier $h^* : \mathcal{X} \to \{\pm 1\}$ with

$$\mathrm{er}_D^c[h^*] = \inf_{h:\mathcal{X}\to\{\pm 1\}} \mathrm{er}_D^c[h].$$

   How does your derivation change if the loss incurred on misclassifying a positive instance as negative is a and that for misclassifying a negative instance as positive is b for some arbitrary $a, b > 0$?

   solution

   Optimal classifier: $h^*(x) = sign(\eta(x) - c)$

   Follow the derivation as given in section 4 of Introduction. Binary Classification and Bayes Error By Prof. Shivani Agarwal `http://drona.csa.iisc.ernet.in/~e0270/Jan-2013/Lectures/1.pdf`

4. **Programming Exercise: Logistic Regression. [30 Mark]** Write a piece of MATLAB code to implement logistic regression on a given binary classification data set $(\mathbf{X}, \mathbf{y})$, where $\mathbf{X}$ is an $m \times d$ matrix (m instances, each of dimension d) and y is an m-dimensional vector (with $y_i \in \{0,1\}$ being a binary label associated with the i-th instance in $\mathbf{X}$): your program should take the training data $(\mathbf{X}, \mathbf{y})$ as input and output a d-dimensional weight vector $\mathbf{w}$ and bias (threshold) term b representing the learnt classifier. For this problem, you are provided a spam classification data set[1], where each

[1] UCI Machine Learning Repository: `http://archive.ics.uci.edu/ml/datasets/Spambase`.

instance is an email message represented by 57 features and is associated with a binary label indicating whether the email is spam or non-spam. The last column of 'spambase.data' denotes whether the e-mail was considered spam (1) or not (0). The data set is divided into training and test sets. The goal is to learn from the training set a classifier that can classify new email messages in the test set.

(a) Use your implementation of logistic regression to learn a classifier from 10% of the training data, then 20% of the training data `Problem-4\data\train.txt`), then 30% and so on upto 100%. In each case, measure the classification error on the training examples used, as well as the error on the given test set [test.txt] (you can use the provided code `classification_error.m` to help you compute the errors). Plot a curve showing both the training error and the test error (on the y-axis) as a function of the number of training examples used (on the x-axis). Such a plot is often called a **learning curve**. (Note: the training error should be calculated only on the subset of examples used for training, not on all the training examples available in the given data set.)

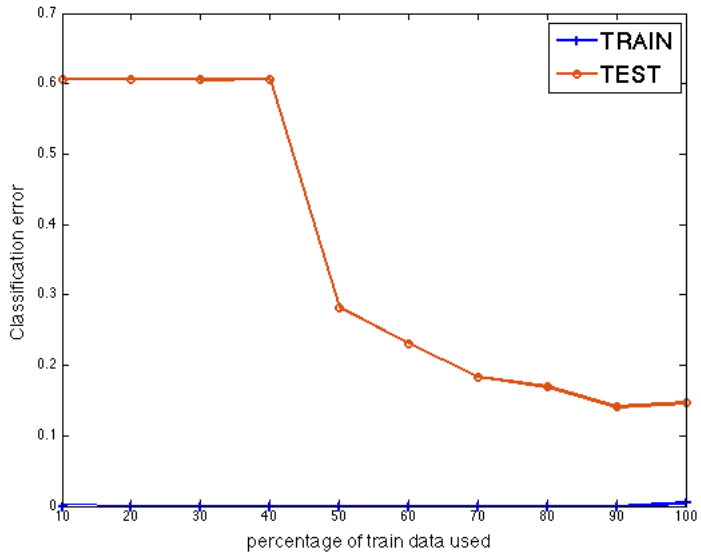
(b) Incorporate a L_2-regularizer in your implementation of logistic regression (taking the regularization parameter λ as an additional input). Run the regularized version of logistic regression on the entire training set for different values of λ in the range $\{10^{-7}, 10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1\}$. Plot the training and test error achieved by each value of λ (λ on the x-axis and the classification error on the y-axis) and identify the value of λ that achieves the lowest test error (resolving ties in favor of the smallest value of λ). Now, select λ from the same range through 5-fold cross-validation on the training set (the train and test data for each fold are provided in a separate folder `Problem-4\data\spambase-cross-validation`); report the average cross-validation error (across the five folds) for each value of λ and identify the value of λ that achieves the lowest average cross-validation error (again resolving ties in favor of the smallest value of λ). Does the cross-validation procedure select the right value of λ?

Hints: The `glmfit` function in MATLAB, which implements (unregularized) logistic regression, can be used for sanity check in this problem; note that the input labels to this function must be in $\{0, 1\}$ and not in $\{\pm 1\}$. For implementing your own version of logistic regression, the `fminunc` MATLAB function for unconstrained optimization will be useful. Also, do not forget to include the bias (threshold) term in the logistic regression model.

Solutions

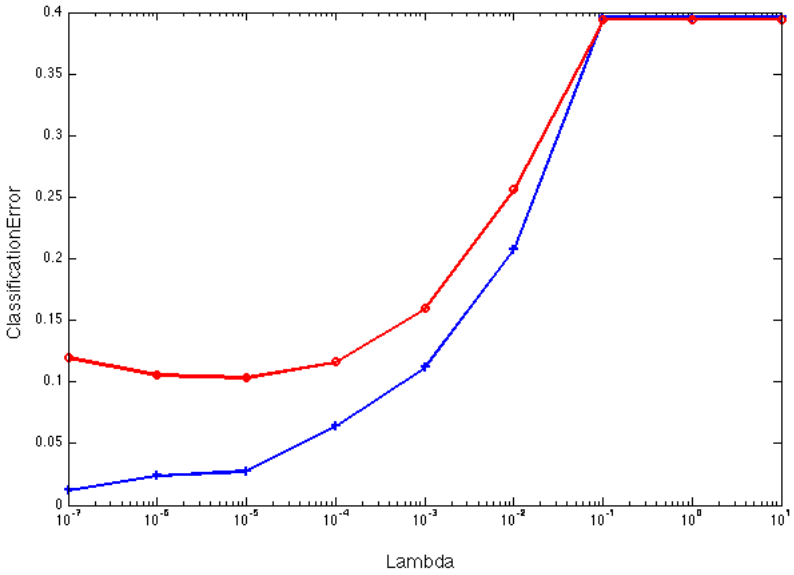
4a

Figure 1: Learning curve



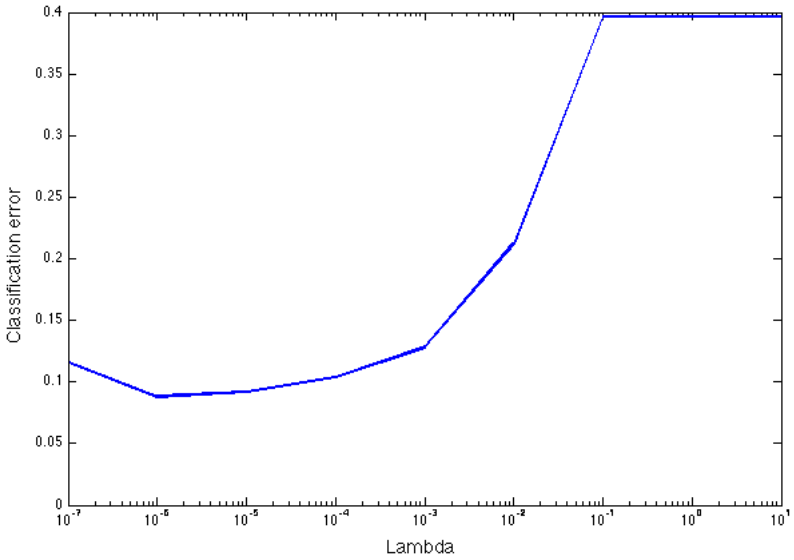
4b

Figure 2: Learning curve with regularisation [red-circle-line represents test error and blue-plus-line represents train error]



Best value of λ is 10^{-5} with test error of 0.1037.

Figure 3: Learning curve with regularisation on cross-validation data



Selection of best lambda using the cross-validation splits

Table 3: Average classification error [cross validation] for value of lambda

λ	Avg. Error#1
10^-7	0.1160
10^-6	0.0880
10^-5	0.0920
10^-4	0.1040
10^-3	0.1280
10^-2	0.2120
10^-1	0.3960
1	0.3960
10	0.3960

Best value for Lambda is observed to be 10^{-6} for cross validation method. Therefore, the lambda value does not match exactly between real and cross-validation method (but is quiet close).

5. **Programming Exercise: Naive Bayes, Perceptron and Winnow Algorithm. [50 Mark]** A dataset for sentiments of movie review is available at `http://ai.stanford.edu/~amaas/data/sentiment/`. The dataset contains movie reviews in natural language and their sentiment whether they are positive[1] or negative[-1]. We have processed this large dataset into three train sets [small, medium and large] and one test set. Use train-small dataset for training in sub-question a-f. Use all three train sets for question part g. There is a single test set, which should be used for all sub-questions. The dataset contains processed documents, the last column in each line denotes the label [1,-1], all other columns of vector represent encoded data [14666 columns]. These are large files and may take about 5 mins to load in matlab running on a desktop/laptop. Each index on the vector corresponds to the count of a vocabulary word [one-hot-vector encoding], imdb_vocab.csv file contains the word for index i at line i (use this information for sub-question d).

   Code for accuracy prediction is included in the code folder (Classification_Accuracy.m). You can use 'confusionmat' function for computing confusion matrix in matlab (sklearn.metrics.confusion_matrix in python).

   (a) Implement a naive bayes classifier for document classification for sentiment analysis dataset. Please report train and test accuracies. Also provide the confusion matrix for test results.

   (b) Implement Perceptron algorithm for the same problem of document sentiment classification. Run your code for the train_small.csv dataset for 25 rounds(A round is an iteration over the training data). After each round report the training and test accuracies in a plot, with round number on x-axis, train & test accuracies on y-axis.

   (c) Implement Winnow algorithm for the same problem of document sentiment classification. Run your code for the train_small.csv dataset for 25 rounds(A round is an iteration over the training data). After each round report the training and test accuracies in a plot, with round number on x-axis, train & test accuracies on y-axis.

   (d) After complete training in the perceptron algorithm, list top 10 words, ordered based on the magnitude of w_i is the final weight vector w for perceptron. For example, if index i=60 is one of the top 10 w_i, then word at line 60 is one of the influential features for this task prediction. *Non-credit: Why do you think these words have high coefficient value in the final weight vector? Hint: relate to the problem of sentiment classification*

   (e) Shuffle your training dataset using given shuffle script (shuffle.m), train a perceptron using the shuffled data and note the number of updates that you are making to the weight vector. Repeat shuffle and train cycle 20 times. Give a histogram plot of the number of updates you made for the 20 shuffles of the training dataset. What can you say about γ from these values?*

   (f) In the perceptron training algorithm you were making an update to the weight vector when making a mistake as $w + (y * x)$. Now change this update rule as $w + (\eta * y * x)$, where η is a small value, typically known as learning rate. Train your perceptron using different values of η ranging from 0.05 to 0.6 with increments of 0.05. For each value report the number of updates made, train and test accuracies in a table. Plot the number of updates (y-axis) made against η on x-axis.

(g) Run-time comparison: Train perceptron algorithm using train-small, train-medium, train-large datasets and test on given test dataset. State the training time for 3 the train sets in a plot, with number of examples in train set on x-axis and runtime on y-axis. Report test accuracy on models trained on these 3 train datasets in a table. State your conclusion about the run-time complexity of perceptron.

Solutions

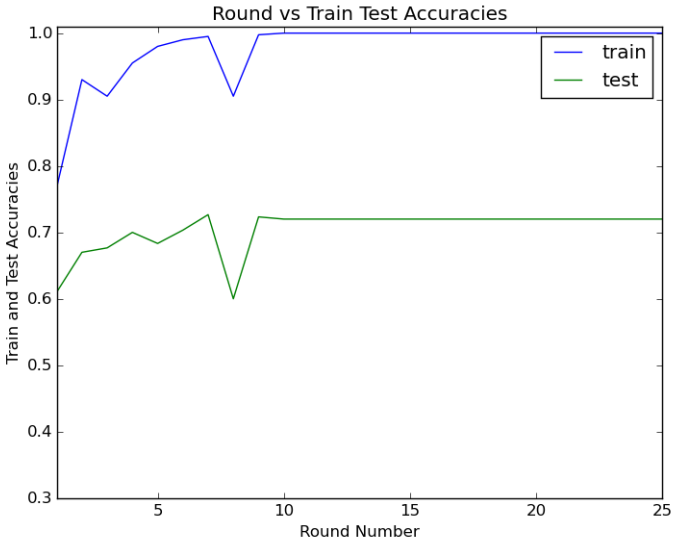
5a

(a) Train Accuracy 1.0

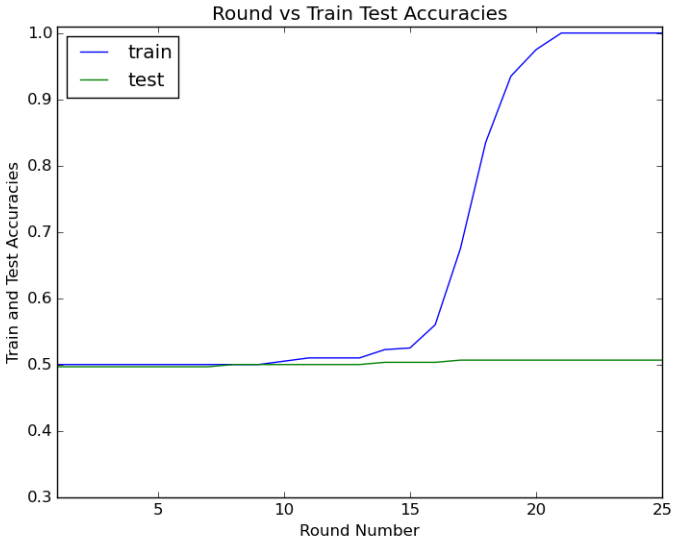
(b) Test Accuracy 0.59

(c) Confusion Matrices. Train and Test [[200 0][0 200]], [[93 58] [65 84]]

5b



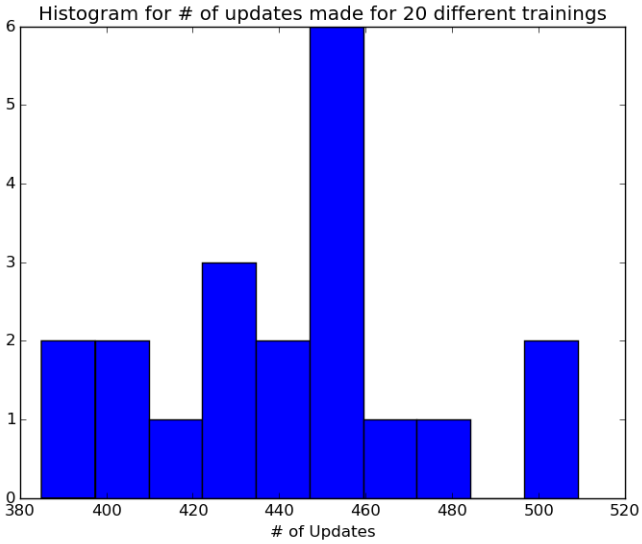
5c



5d

Top 10 words: ['titanic', 'bad', 'they', 'seagal', 'no', 'nothing', 'would', 'story', 'love', 'out'] Most of these words are sentiment(positive or negative) bearing words.

5e

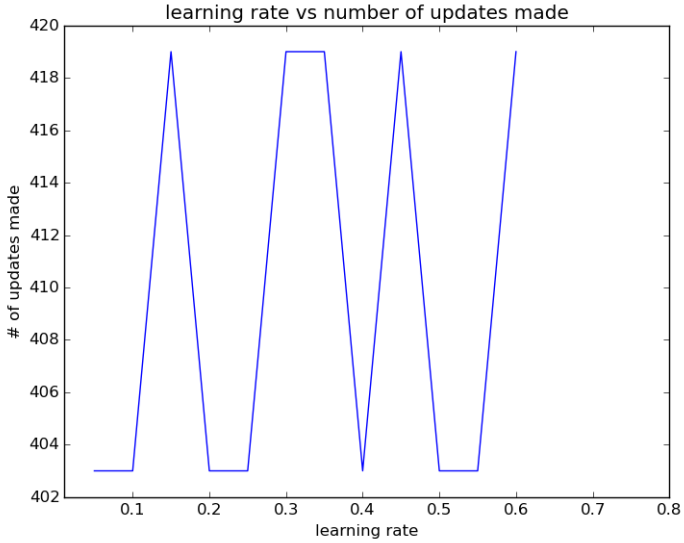


As the shuffles are random, the number of updates(and histogram) can vary. Upper bound for gamma can be calculated from the maximum value of updates.

5f

Table 4: Updates,Train and Test Accuarcies for various values of η

η	Updates	Train Accuracy	Test Accuarcy
0.05	403	1.0	0.72
0.1	403	1.0	0.72
0.15	419	1.0	0.73
0.2	403	1.0	0.72
0.25	403	1.0	0.72
0.3	419	1.0	0.73
0.35	419	1.0	0.726666666667
0.4	403	1.0	0.72
0.45	419	1.0	0.73
0.5	403	1.0	0.72
0.55	403	1.0	0.72
0.6	419	1.0	0.73



5g

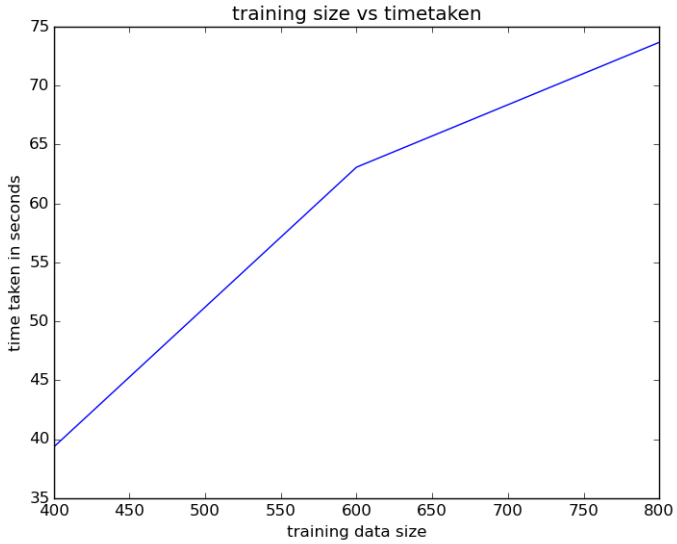


Table 5: Training Data Size and Test Accuracy

Train Data Size	Train Accuracy	Test Accuracy
small	1.0	0.72
medium	1.0	0.75
large	1.0	0.756666666667