CS 4780/5780 Machine Learning Due October 4, 2011

Assignment 3: Perceptron and Linear SVMs

Instructor: Thorsten Joachims *Total Points: 100*

Course Policy: Assignments are due at the beginning of class on the due-date in hard copy.
Write your NetIDs on the first page of the submitted hard-copy.
Late assignments lose 5 points for each late day and can be submitted directly to the TAs.
Assignments submitted after solutions are made available will not be accepted.
Groups of two are strongly encouraged.
All sources of material must be cited. Assignment solutions will be made available along with the graded homework solutions. The University Academic Code of Conduct will be strictly enforced, including running cheating detection software.

Problem 1: Perceptron Implementation and Comparison with Linear SVM [50 points]

Sentiment analysis seeks to identify the viewpoint(s) underlying a text span; an example application is classifying a movie review as "thumbs up" or "thumbs down" [Pang and Lee(2004)]. In this problem, we look at Moview Review Data[1]. There are in total $2,000$ textual movie reviews labeled as positive and negative towards the corresponding movie (1000 postive, 1000 negative). We view it as a binary classification problem trying to classify a movie review into positive or negative.

To prepare data for machine learning tasks, we do the following preprocessing for you:

We split the data into training set and validation set. There are $1,000$ instances in the training set, and 1000 instances in the validation set. They are both balanced, i.e., 500 positive reviews and 500 negative reviews in each. We employ unigram features. That is, we extract all the words from training dataset, and then build feature vectors with the frequency of words in a movie review. We normalize the feature vector to unit length. In total, there are $29,328$ different features.

Our task is to train linear classifiers and see how they perform on the validation set in this problem. On the CMS website you will find the *movie review data* which has been preprocessed to adhere to the SVM-Light format. In the files, 1 corresponds to positive reviews, while -1 refers to negative reviews. The zipped file contain 5 files:

- *polarity.train*. This is the default file to train linear classifiers.

[1]http://www.cs.cornell.edu/people/pabo/movie-review-data/

- *polarity.validation.* This file is used for validation throughout the problem.
- *polarity-reorder-1.train* This file is reordered from *polarity.train.*
- *polarity-reorder-2.train* This file is reordered in another way from *polarity.train.*
- *polarity-reorder-3.train* This file is reordered in a third way from *polarity.train.* This one is reordered so that the first 500 instances are positive, and the rest are negative in training set.

(a) First, implement a biased linear classifier from *polarity.train* using the primal perceptron algorithm. The perceptron algorithm with bias is available in Chapter 2.1, "Introduction to Support Vector Machines".[2]. Table 2.1 gives the primal perceptron algorithm (Set learning rate $\eta = 1$ in the primal perceptron algorithm.). Please answer the following questions. (15 points)

- Get training error rate, validation error rate on *polarity.validation* for each iteration from 1 to 20. Plot the training error rate and validation error rate with the number of iterations. How do error rates change with iterations? Comment on the plot.
- Would you prefer a lower learning rate or a higher one for improved performance? Explain your answer.
- Also, Table 2.2 gives the dual algorithm in the book. Do you think that we should use dual algorithm in this problem? Explain your reason, considering efficiency and accuracy.

(b) Now we train linear SVM classifiers with SVM_{light} `http://svmlight.joachims.org/`. There are instructions on the website about how to use source code [3]. You can also download binaries, but make sure that what you download is SVM_{light} instead of SVM^{struct} or other variations.

Train the SVM classifier with different C from $0.25, 0.5, 1, 2, 4, 8, 16, 32, 64$, plot the training error rate and validation error rate with C in log-scale($log_2(C)$ as x-axis). Make sure the only command-line parameter you use is "$-c$". The file for training is also *polarity.train*, the file for validation is *polarity.validation.*

Where do you achieve the smallest validation error rate? How is it compared to the validation error rate after 20 iterations in (a)? Comment on the plot, explain how the error rates change with C.(10 points)

(c) Effect of data reordering.

Use your perceptron algorithm to train linear classifiers from *polarity.train*, *polarity-reorder-1.train*, *polarity-reorder-2.train* and *polarity-reorder-3.train.* 4 linear classifiers should be trained, one each on polarity*.train. Compute training error rate for each iter-

[2]http://library.books24x7.com.proxy.library.cornell.edu/toc.asp?bookid=3497

[3]`http://download.joachims.org/svm_light/current/svm_light.tar.gz`

ation from 1 to 20. Plot the training error rate with the number of iterations. Compute validation error rate for each iteration from 1 to 20. Plot the validation error rate with the number of iterations in another figure.

In both of your figures. there are four lines for *polarity.train*, *polarity-reorder-1.train*, *polarity-reorder-2.train* and *polarity-reorder-3.train* respectively. Comment on the figures. Could you explain why it works very badly for *polarity-reorder-3.train*?

Also, use the C with the smallest validation error rate in (b) to run SVM_{light} on the four files, and report training error rate and validation error rate.

In all cases, the file used for validation is *polarity.validation*. (15 points)

(d) Now you want to test whether the Linear SVM classifier has a prediction error rate that is significantly different from that of the perceptron. You decide to do the following:

You take the classifier produced by the perceptron algorithm after 20 iterations in (a) (denoted as A_1), as well as the the classifier you got from (b) after selecting the value of C with the smallest validation error rate (denoted as A_2). Using the predictions on the validation set, you now test the hypothesis that A_2 is significantly different from A_1 using thebinomial test with 95% confidence.

Show the steps of computing the binomial sign test and its result. Then step back and think about what you have done. Do you think that the result of the hypothesis test is valid and correct?

Problem 2: Linear Separable and Mistake Bound [20 points]

In a perfect world, out data is "naturally" separable:

$$S := \{(\mathbf{x}_1, y_1) \dots (\mathbf{x}_n, y_n)\}$$

Where $\mathbf{x}_i \in R^N$, $|\mathbf{x}_{ij}| \le M, \|\mathbf{x}_i\| \le R, \forall 1 \le i \le n, 1 \le j \le N$ ($|\cdot|$ takes the absolute value of the parameter, $\|\cdot\|$ denotes 2-norm or Euclidean length.). $y_i \in \{-1, 1\}, \forall 1 \le i \le n$.

It can be separated by an unbiased hyperplane with normal vector $\mathbf{w}$ ($\|\mathbf{w}\| = 1$) with margin $\delta > 0$, i.e., $\forall 1 \le i \le n, y_i(\mathbf{w}^T\mathbf{x_i}) \ge \delta$.

However, in reality, some features are missing for parts of our data. For simplicity, suppose the first m ($m < n$) instances miss the first feature. We decide to set it to 0 (though there are smarter ways to deal with missing features). Formally,

$$\hat{\mathbf{x}}_i = (0, \mathbf{x}_{i2}, \dots, \mathbf{x}_{iN})^T, 1 \le i \le m$$

$$\hat{\mathbf{x}}_i = (\mathbf{x}_{i1}, \mathbf{x}_{i2}, \dots, \mathbf{x}_{iN})^T, m + 1 \le i \le n$$

And we define $\hat{S}$ as

$$\hat{S} := \{(\hat{\mathbf{x}}_1, y_1) \ldots (\hat{\mathbf{x}}_n, y_n)\}$$

There is no guarantee that the new data is separable now. We decide to add m indicator variables to the training examples. Each assumes a non-zero value only for its respective point:

$$\tilde{\mathbf{x}}_i = (\hat{\mathbf{x}}_i^T, t_{i1}, \ldots, t_{im})^T$$

where

$$if\ 1 \le i \le m, t_{ij} = \begin{cases} k > 0 & \text{if } i = j, \\ 0 & \text{otherwise} \end{cases}$$
$$if\ m+1 \le i \le n, t_{ij} = 0, 1 \le j \le m$$

(if t_{ij} is a boolean indicatior, then $k = 1$, it could also be a general indicator.)

Now, we have our new dataset as

$$\tilde{S} := \{(\tilde{\mathbf{x}}_1, y_1) \ldots (\tilde{\mathbf{x}}_n, y_n)\}$$

(a) Show a construction that proves that $\tilde{S}$ is linearly separable after adding indicator variable. (15 points)

(b) In the perfect world, we know that the mistake bound is $\frac{R^2}{\delta^2}$, what is the corresponding upper bound for $\tilde{S}$? Show your derivation. (Hint: just need to provide a valid bound. The tighter, the better. Choosing different k may affect the upper bound) (5 points)

Problem 3: Linear SVM [30 points]

We want to train a linear SVM classifier based on n examples, $S = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)\}$ where each $x_i \in R^N$ and each $y_i \in \{-1, 1\}$. In this data, we have the following property:

- The maximum length of all feature vectors $\mathbf{x}_i$ is 1, i.e., $\mathbf{x}_i^T\mathbf{x}_i = l_i$, $max_{1 \le i \le n} l_i = 1$

(a) First, we train a biased Linear SVM classifier on this data, and get $(\mathbf{w}, b)$. The first three instances is as follows. α_i is the dual variable of example i.

What is the upper bound of leave-one-out error for the three instances? Explain your solution. (10 points)

(b) Suppose we have a different data set, which have the following two properties.

- The maximum length of all feature vectors $\mathbf{x}_i$ is 1, i.e., $\mathbf{x}_i^T\mathbf{x}_i = l_i$, $max_{1 \le i \le n} l_i = 1$

i	y_i	$\mathbf{w}^T x_i + b$	α_i
1	1	1	0
2	-1	-0.4	0.1
3	1	0.3	0.2

- Any two feature vectors in our training set are orthogonal, i.e., $\forall i \neq j, \mathbf{x}_i^T \mathbf{x}_j = 0$.

Suppose we train a hard margin **unbiased** SVM on the full training set. For any training example $(\mathbf{x}_i, y_i)$, what is the corresponding dual variable α_i? (10 points)

(c) Now, let us consider the soft margin case. In the following (primal) form of the (soft-margin) SVM optimization problem:

$$min \frac{1}{2} \|\mathbf{w}^2\| + C \sum_{i=1}^{n} \xi_i$$

$$subject\ to\ y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i, \forall 0 \leq i \leq n$$

Our training data is $S := \{(\mathbf{x}_1, y1) \ldots (\mathbf{x}_n, y_n)\}$, $\mathbf{x}_i \in R^N$, $y_i \in \{-1, 1\}$ the solution to this problem is given by $(\mathbf{w}, \xi)$.

If we scale up x_i, $\mathbf{x}_i^* := k\mathbf{x}_i, \forall 1 \leq i \leq n$ That is, for each dimension of $\mathbf{x}_i \in R^N$, $\mathbf{x}_{ij} = k * \mathbf{x}_{ij}, \forall 1 \leq j \leq N$ how should C be changed so that the new solution $(\mathbf{w}^*, \xi^*)$ defines the same linear classifier? What are the resulting $(\mathbf{w}^*, \xi^*)$? (10 points)

References

[Pang and Lee(2004)] B. Pang and L. Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *ACL*, pages 271–278, 2004.