CS57800 Statistical Machine Learning
Homework 2

Ting Zhang
School of Industrial Engineering
zhan1013@purdue.edu

October 6, 2015

1 Foundations

1. (1) Boolean function
   $f(x_1, x_2, x_3, x_4) = \neg[(x_1 \wedge \neg x_2 \wedge \neg x_3 \wedge \neg x_4) \vee (\neg x_1 \wedge x_2 \wedge \neg x_3 \wedge \neg x_4) \vee (\neg x_1 \wedge \neg x_2 \wedge x_3 \wedge \neg x_4) \vee (\neg x_1 \wedge \neg x_2 \wedge \neg x_3 \wedge x_4) \vee (\neg x_1 \wedge \neg x_2 \wedge \neg x_3 \wedge \neg x_4)]$
   (2) Linear function
   $f(x_1, x_2, x_3, x_4) = 1 \quad \text{if} \quad x_1 + x_2 + x_3 + x_4 \geq 2$

2. $size(CON_B) = 2^n$

3. Since $\beta_n = \beta_o + y_i u_i$, then

$$\begin{aligned}
\|\beta_n - \beta^*\|^2 &= \|\beta_o + y_i u_i - \beta^*\|^2 \\
&= (\beta_o - \beta^*)^2 + 2y_i u_i(\beta_o - \beta^*) + (y_i u_i)^2 \\
&= (\beta_o - \beta^*)^2 + 2(\beta_o y_i u_i - \beta^* y_i u_i) + (y_i u_i)^2
\end{aligned}$$

   Since $u_i = x_i{}^*/\|x_i{}^*\|$, then $u_i = \pm 1$. Also, because u_i is a misclassified example, then $y_i \cdot (\beta_o u_i) = -1$. Because β^* is the final separating parameter vector, we will have $y_i \cdot (\beta^* u_i) = 1$. Substitute these values into the above equation, we will have:

$$\begin{aligned}
\|\beta_n - \beta^*\|^2 &= \|\beta_o - \beta^*\|^2 + 2(-1-1) + 1 \\
&= \|\beta_o - \beta^*\|^2 - 3 \\
&\leq \|\beta_o - \beta^*\|^2 - 1
\end{aligned}$$

4. initialize $w = 1$ if $f(x) = 1$, eliminate all x_i that equals 0

5. (1) Both classifiers will converge since the data is linearly separable. However, the one with sorted dataset may converge slower than the one with randomized order. (2) The training error of the first classifier would be 0%. And the training error for the second classifier would be 0% since the data is linearly separable.

6. Proof.

$$
\begin{aligned}
\|\sum_{i\in N} y_i x_i\| &= \|\sum_{i\in N}(w_{i+1}-w_i)\| \\
&= \|(w_{i+1}-w_i)+(w_i-w_{i-1})+(w_{i-1}-w_{i-2})+...+(w_1-w_0)\| \\
&= \|w_{i+1}\| \\
&= \sqrt{(\|w_{i+1}\|^2-\|w_i\|^2)+(\|w_i\|^2-\|w_{i-1}\|^2)+...+(w_1^2-w_0^2)} \\
&= \sqrt{\sum_{i\in N}(\|w_{i+1}\|^2-\|w_i\|^2)} \\
&= \sqrt{\sum_{i\in N}(\|w_i+y_i x_i\|^2-\|w_i\|^2)} \\
&= \sqrt{\sum_{i\in N}(\|w_i\|^2+2y_i w_i x_i+\|y_i x_i\|^2-\|w_i\|^2)} \\
&= \sqrt{\sum_{i\in N}(2y_i w_i x_i+\|x_i\|^2)}
\end{aligned}
$$

Since these are the examples when Peceptron makes a mistake, therefore, $y_i w_i x_i \le 0$. So,

$$\|\sum_{i\in N} y_i x_i\| \le \sqrt{\sum_{i\in N}\|x_i\|^2}$$

2 Programming Report

In this section, Perceptron and Winnow algorithm were implemented to predict positive or negative sentiment from single snippet movie review. For both algorithms, they follow the same procedure. First, features were extracted in three different representations, including unigram, bigram and both unigram and bigram. Then, the weights for each feature in the feature set were trained following each algorithm. At last, the sentiment can be predicted using the trained weights. Detailed explanation is presented in the following sections.

2.1 Feature Extraction

In this task, three representations of features were extracted and experimented with: 1) unigram, 2) bigram and 3) both unigram and bigram.

1) Unigram: Each single word in the movie reviews were extracted as a feature.

2) Bigram: Two consecutive words in a snippet were extracted as one feature.

3) Use Both: The combination of both unigram features and bigram features.

In all these representations, the presense of a feature (one word or group of two words) is denoted as 1, therefore, a single snippet can be represented as a series of 1s and 0s.

2.2 Perceptron and Winnow Algorithm

Both of these onling learning algorithms learns on mistakes. To be specific, the weights for each active feature will be updated if the prediction is wrong in the input snippet.

1) Perceptron: Initialize all weights,w, and the bias term b to 0. And the weights for each active feature and bias term will be updated following this equation:

$$w_d = w_d + yx_d$$
$$b = b + y$$

where d is the index for each feature, and y is the label for this input snippet.

After going through all the input snippets for a certain number of iterations, we should get fewer and fewer errors. This number of iterations is considered as a hyper-parameter that will be tuned using the validation set. The optimal number of iteration will be selected and the weights and bias term updated after this number of iteration will be used in the testing set. To predict the label for each snippet, perceptron simply following this equation:

$$a = \sum_{d=1}^{D} w_d x_d + b$$

The sign of a would be the predicted label.

2) Winnow: Different from perceptron, winnow initialize all weights w to 1, and a fixed value θ to the number of features n. To update the weights for each feature, the equation depends on how the mistake is made. If the mistake is on a postive example, then the weights of active features will be doubled. If the mistake is on a negative example, the weights for active features will be halved, which can be represented in the following equations.

$$\text{if} \quad f(x) = 1, w_i = 2w_i \quad (\text{if} \quad x_i = 1)$$
$$\text{if} \quad f(x) = -1, w_i = w_i/2 \quad (\text{if} \quad x_i = 1)$$

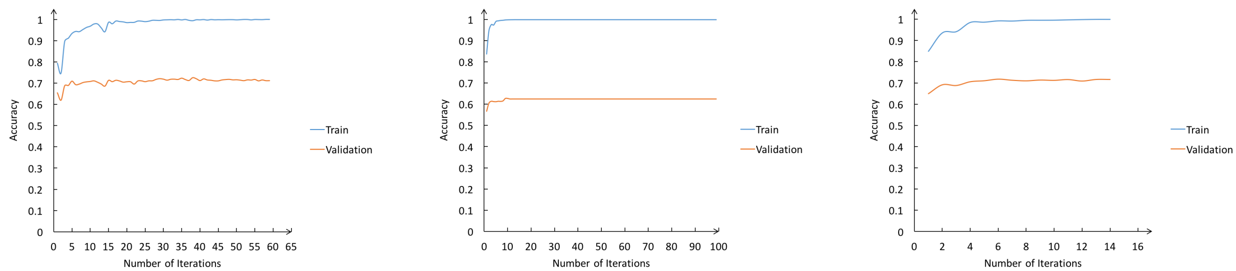
The same as perceptron, an optimal weights will be obtained by tuing the maximum number of iterations on validation set. To predict the label of on snippet, we can apply the following equations:

$$\sum_{d=1}^{D} w_d x_d$$

If the summation is less than theta, then the predicted label would be -1; otherwise, it would be 1.

2.3 Experiment and Results

Here, the results for tuning the hyper-parameter, maximum number of iterations, for all the feature set and onling learning algorithms are presented. Figure 1 shows the results for perceptron with (a)Unigram, (b)Bigram and (c)Both feature set. Figure 2 shows the results for winnow algorithm.



(a) Unigram (b) Bigram (c) Both

Figure 1: Results for Perceptron Algorithm

2.3.1 Perceptron algorithm

For unigram feature set, perceptron converged at iteration 58 for the training set and reached it's highest accuracy, 72.5%, for validation set at iteration 40. For bigram feature set, perceptron didn't converge for the training set until iteration 100 with accuracy of 99.9%. And the accuracy for validation set reached its highest on iteration10 for 62.7%. For the one with both unigram and bigram feature set, perceptron converged quickly at iteration 15 for the training set, with the highest valiation accuracy of 71.8% at iteration 5.

The testing set was then tested with weights after 40 iterations for unigram feature set, 10 iterations for bigram feature set and 5 iterations for both feature set. The accuracy of these feature set are: 71.2%, 63.5% and 73.1%.

2.3.2 Winnow algorithm

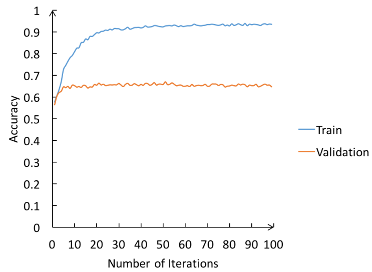
For the winnow algorithm, no feature set actually converged after 100 iterations. Unigram feature set reached its highest accuracy of 93.9% at iteration 66 for the t raining set. For the validation set, It reached its highest accuracy of 66.9% at iteration 51. For the bigram feature, the accrucy of training set goes up to 99.9% and reached the highest for validation set at iteration 25 of 60.1%. For the one using both unigram and bigram feature set, the accuracy of trainning set reached 99.9% at iteration 67. And the accuracy for validation set is 64.1% at iteration 9.

The testing set was then tested with weights after 51 iterations for unigram feature set, 25 iterations for bigram feature set and 9 iterations for both feature set. The accuracy of these feature set are: 65.2%, 62.2% and 66.8%.

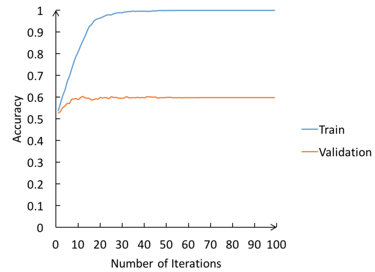
2.3.3 Comparisons

We can first compare among different feature sets. From the results shown above, we know that using both unigram and bigram contribute a lot with less iterations to converge and relatively high accuracy in validation set, for both perceptron and winnow algorithm. And unigram feature set also works well in this task to obtain a weight showing higher accuracy for both algorithms.

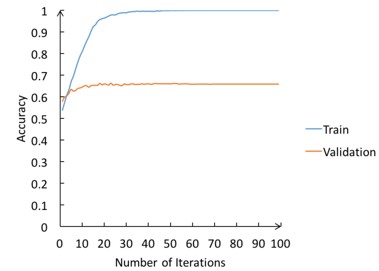
Then compare between perceptron and winnow algorithm, we can observe that perceptron always shows higher accuracy than winnow and it takes less iterations for perceptron to converge



(a) Unigram



(b) Bigram



(c) Both

Figure 2: Results for Winnow Algorithm

or reach the highest accuracy.

2.4 Conclusions and Future Work

In this task, two onling learning algorithm, Perceptron and Winnow, were implemented. Experiments with feature set of unigram, bigram and both of them were conducted. Validation set was applied to tune the hyper-parameter, number of iterations, in this task. The perceptron implemented in this task was of primal representation, in the future, dual representation can be implemented to improve the performance of this algorithm.