CS221 Fall 2015 Homework Sentiment]

SUNet ID: prabhjot
Name: Prabhjot Singh Rai

By turning in this assignment, I agree by the Stanford honor code and declare that all of this is my own work.

Problem 1

(a) Mapping reviews into feature vectors as follows,

$$\phi_{x1} = \{pretty : 1, bad : 1\}, y_1 = -1$$
$$\phi_{x2} = \{good : 1, plot : 1\}, y_2 = +1$$
$$\phi_{x3} = \{not : 1, good : 1\}, y_3 = -1$$
$$\phi_{x4} = \{pretty : 1, scenery : 1\}, y_4 = +1$$

Recalling from the graph, gradient of hinge loss, for margin less than one, will be $-\phi_{(x)}y$ and 0 for margin greater than one.

$$\nabla_w Loss_{hinge}(x, y, w) = \begin{cases} -\phi_{(x)}y & \text{when}(w.\phi)y < 1 \\ 0 & \text{when}(w.\phi)y > 1 \end{cases}$$

Stochastic gradient descent is defined as

$$w \leftarrow w - \eta \nabla_w Loss_{hinge}(x, y, w)$$

Initialising $\mathbf{w} = [0, \ldots 0]$, or $\mathbf{w} = \{pretty : 0, bad : 0 \ldots scenery : 0\}$, and iterating over each feature vector to update w
First iteration, $w.\phi_{x1}y = 0, \nabla Loss = -\phi_{(x)}y = \{pretty : 1, bad : 1\}$

$$w = w - \eta(\{pretty : 1, bad : 1\})$$
$$w = \{pretty : 0, bad : 0 \ldots scenery : 0\} - \{pretty : 0.5, bad : 0.5\}$$
$$w = \{pretty : -0.5, bad : -0.5\}$$

Second iteration, $w.\phi_{x2}y = 0, \nabla Loss = -\phi_{(x)}y = \{good : -1, plot : -1\}$

$$\begin{aligned} w &= w - \eta\{good : -1, plot : -1\} \\ &= \{pretty : -0.5, bad : -0.5\} - 0.5\{good : -1, plot : -1\} \\ &= \{pretty : -0.5, bad : -0.5, good : 0.5, plot : 0.5\} \end{aligned}$$

Third iteration, $w.\phi_{x3}y = -0.5, \nabla Loss = -\phi_{(x)}y = \{not : 1, good : 1\}$

$$\begin{aligned} w &= w - 0.5\{not : 1, good : 1\} \\ &= \{not : -0.5, bad : -0.5, plot : 0.5, pretty : -0.5\} \end{aligned}$$

Fourth iteration, $w.\phi_{x4}y = -0.5, \nabla Loss = -\phi_{(x)}y = \{pretty : -1, scenery : -1\}$

$$\begin{aligned} w &= w - \{pretty : -0.5, scenery : -0.5\} \\ &= \{scenery : 0.5, plot : 0.5, bad : -0.5, not : -0.5\} \end{aligned}$$

Therefore, weights of the six words are $\{pretty : 0, good : 0, bad : -0.5, plot : 0.5, not : -0.5, scenery : 0.5\}$

(b) Labelled dataset:
1. "not good" (-1)
2. "good" (+1)
3. "bad" (-1)
4. "not bad" (+1)

Let $\{not : x, good : y, bad : z\}$ be the weights assigned to each feature(our feature extractor considering only single words as per the question), where x, y and z can be both positive and negative. In order to get a total of zero error on all data points, we need to get zero error on each data point. For "good" review, score should be positive, therefore, $y > 0$. For bad, score should be negative, therefore, $z < 0$. For "not good", $x + y$ should be < 0, therefore, since $y > 0$, $x < 0$ and $x < -y$. For "not bad", $x + z$ should be > 0, but through former calculations, $x + z$ will be < 0. Therefore, no linear classifier using word features can get zero error on this dataset.

In order to get a zero error, we can append our feature vector with bi-grams(taking two continuous words into account). This additional feature would add weights to "not bad" in the first data and "not bad" in the last. In such a scenario, weights of each feature would be "not good" < 0, "good" > 0, "bad" < 0 and "not bad" > 0.

Problem 2

(a)

$$\begin{aligned} f_w(x) &= \sigma(w.\phi_x) \\ &= (1 + e^{-w.\phi_x})^{-1} \\ Loss_{squared}(x, y, w) &= (f_w(x) - y)^2 \\ &= ((1 + e^{-w.\phi_x})^{-1} - y)^2 \end{aligned}$$

(b) Let $p = \sigma(w.\phi_x) = (1 + e^{-w.\phi_{(x)}})^{-1}$

$$\begin{aligned}
\nabla_w Loss &= \frac{d(p-y)^2}{dw} \\
&= \frac{d(p-y)^2}{dp}\frac{dp}{dw} \\
&= 2(p-y)\frac{dp}{dw} && \textbf{...(1)} \\
\frac{dp}{dw} &= \frac{d(1+e^{-w.\phi_{(x)}})^{-1}}{dw} \\
&= (-1)(1+e^{-w.\phi_{(x)}})^{-2}e^{-w.\phi(x)}(-\phi_{(x)}) \\
&= (p)^2\frac{(1-p)}{p}\phi_{(x)} && (\text{since } p = (1+e^{-w.\phi_x})^{-1}, \text{therefore } e^{-w.\phi_{(x)}} = \frac{1-p}{p}) \\
&= p(1-p)\phi_{(x)}
\end{aligned}$$

Therefore, substituting the value of $\frac{dp}{dw}$ in (1), we get gradient of the loss is

$$\nabla_w Loss = 2(p-y)p(1-p)\phi_{(x)}$$

(c) Substituting $y = 1$ and arbitary $\phi_{(x)}$ in the above equation,

$$\begin{aligned}
\nabla_w Loss &= 2(p-1)p(1-p)\phi_{(x)} \\
&= -2(p-1)^2 p\phi_{(x)} && \textbf{(...2)}
\end{aligned}$$

In order to make the magnitude of the gradient of the loss arbitrarily small, we need to make the above equation close to zero. That can happen when p approaches 1 and p approaches zero.
When p approaches 1:

$$\begin{aligned}
p &= 1 \\
(1+e^{-w.\phi_{(x)}})^{-1} &= 1 \\
e^{-w.\phi_{(x)}} &= 0
\end{aligned}$$

w approaches ∞.
When p approaches 0:

$$\begin{aligned}
p &= 0 \\
(1+e^{-w.\phi_{(x)}})^{-1} &= 0 \\
e^{-w.\phi_{(x)}} &\text{ approaches } \infty
\end{aligned}$$

w approaches $-\infty$. Therefore, for w approaching ∞ and $-\infty$, magnitude of the gradient of the loss is arbitrarily small(approaching zero).
No, the magnitude of the gradient can never be zero.

(d) For largest magnitude of the gradient, in the equation **2** in **2c** above, we need to maximize $(p-1)^2 p$. Differentiating and equating to zero.

$$\begin{aligned} 2p(p-1) + (p-1)^2 &= 0 \\ 2p^2 - 2p + p^2 + 1 - 2p &= 0 \\ 3p^2 - 4p + 1 &= 0 \qquad \textbf{(...3)} \\ 3p^2 - 3p - p + 1 &= 0 \\ (3p-1)(p-1) &= 0 \\ p &= 1, \frac{1}{3} \end{aligned}$$

To check if it's minima or maxima, we differentiate equation 3 and check signs

$$6p - 4 \text{ is negative only when } p = \frac{1}{3}$$

Therefore, maximum ma of function $p(p-1)^2$ when p ranges from 0 to 1 is

$$p = \frac{1}{3}$$

Substituting value of p in equation **2** in **2c** above

$$\begin{aligned} \nabla_w Loss &= ||2(\frac{2}{3})^2 \frac{1}{3} \phi_{(x)}|| \\ &= ||(\frac{2}{3})^3 \phi_{(x)}|| \\ &= ||\frac{8}{27}\phi(x)|| \end{aligned}$$

(e) Yet to implement

Problem 3

(d)

(1) home alone goes hollywood , a funny premise until the kids start pulling off stunts not even steven spielberg would know how to do . besides , real movie producers aren't this nice .

Truth: -1, Prediction: 1 [WRONG]

This is predicted wrong because "funny" has a count of 1 with a heavy positive weight of 0.4, since with most of the movies, funny sounds positive but in the context, the review starts off as positive but the tone changes as the review proceeds to negative, which isn't given accurate weight to by the model.

(2) 'it's painful to watch witherspoon's talents wasting away inside unnecessary films like legally blonde and sweet home abomination , i mean , alabama . '
Truth: -1, Prediction: 1 [WRONG]

In our weights, sweet has high positive weight of 0.55, whose score overshoots scores of other negative words. but is being used to describe the "home" and not the review in general.

(3) patchy combination of soap opera , low-tech magic realism and , at times , ploddingly sociological commentary
Truth: -1, Prediction: 1 [WRONG]

We see many negative words here, but their weights are zero, for example patchy, low-tech, ploddingly etc. That means our model hasn't seen such words in training, therefore based on other words classifies this as positive.

(4) the best thing i can say about this film is that i can't wait to see what the director does next
Truth: 1, Prediction: -1 [WRONG]

The words such as "does" and "film" have very high positive weights(0.6 and 0.36 to be precise), but in general have equal probability of coming in a positive or negative review. But in our training data, these words occur more frequently in positive than negative reviews, that's why they have high positive values. Ignoring such stop words (words which do not tell about actual sentiment) can make the model better.

(5) even during the climactic hourlong cricket match , boredom never takes hold
Truth: 1, Prediction: -1 [WRONG]

Words like "never", "boredom" etc increase negative sentiment a lot, but in context, the movie review says that negative doesn't happen. Since we aren't considering context through n-grams or some other technique, this is classified only on single word basis hence wrong.

(f) When n = 1, $Testerror = 0.478615644344$
When n = 2, $TestError = 0.419527293191$

When n = 3, $TestError = 0.318232976927$
When n = 4, $TestError = 0.283342712437$
When n = 5, $TestError = 0.270680922904$
When n = 6, $TestError = 0.271525042206$
When n = 7, $TestError = 0.270962296005$
When n = 8, $TestError = 0.292909397862$
When n = 9, $TestError = 0.308384918402$

When n=5, the test error is the minimum. Test error reflects how well our feature vector has defined features which eventually are used in classification. If we look at the weights data from the weights file when our feature extractor is word, maximum number of words which "matter"(having high positive or negative weights which will help increase the score) are: {1: 2.0, 2: 1.0, 3: 4.0, 4: 29.0, 5: 42.0, 6: 27.0, 7: 27.0, 8: 18.0, 9: 13.0, 10: 11.0, 11: 8.0, 12: 6.0, 13: 1.0, 14: 2.0}