CS221 Autumn 2016 Homework [1]

SUNet ID: [06033811]
Name: [Chao Xu]

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

Problem 1: Warm Up

(a) Suppose we run stochastic gradient descent, updating the weights according to

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

once for each of the four examples in order. After the classifier is trained on the given four data points, what are the weights of the six words ('pretty', 'good', 'bad', 'plot', 'not', 'scenery') that appear in the above reviews? Use $\eta = 1$ as the step size and initialize w=[0,...,0]. Assume that $\nabla_{\mathbf{w}} Loss_{hinge}(x, y, \mathbf{w}) = 0$ when the margin is exactly 1.

The gradient of a hinge loss should be as following:

$$\nabla_{\mathbf{w}} Loss_{hinge}(x, y, \mathbf{w}) = 0 \quad if \quad margin \geq 1$$

$$\nabla_{\mathbf{w}} Loss_{hinge}(x, y, \mathbf{w}) = -\phi(x)y \quad if \quad margin < 1$$

For the first set training data: $\phi(x) : \{pretty : 1, good : 1\}$ and $\mathbf{w} = [0, \ldots, 0]$

$\nabla_{\mathbf{w}} Loss_{hinge}(x, y, \mathbf{w}) = 0$ because margin equals to 1

$w[pretty] = 0 + \phi[pretty]y = 1$ and $w[good] = 0 + \phi[good]y = 1$

Apply the similar algorithm to rest three training data:

$w[bad] = 0 + \phi[bad]y = -1$ and $w[plot] = 0 + \phi[plot]y = -1$

$w[not] = 0 + \phi[not]y = -1$ and $w[good] = 1 + w\phi[good]y = 0$

$w[pretty] = 1 + \phi[pretty]y = 1$ and $w[scenery] = 1 + \phi[scenery]y = 1$

Therefore, the final value of w is: [1,0,-1,-1,-1,1]

(b) Create a small labeled dataset of four mini-reviews using the words 'not', 'good', and 'bad', where the labels make intuitive sense. Each review should contain one or two words, and no repeated words. Prove that no linear classifier using word features can get zero error on your dataset. Remember that this is a question about classifiers, not optimization algorithms: your proof should be true for any linear classifier, regardless of how the weights are learned. After providing such a dataset, propose a single additional feature that we could augment the feature vector with that would fix this problem. (Hint: think about the linear effect that each feature has on the classification score.)

The four mini-reviews using the words 'not', 'good', and 'bad' with reasonable meanings, can only be not: -1, good: 1, not bad: 1, not good: -1 with linear classifier. Mapping to sparse features vector as following: not: -1,good : 1,not : -1,good : -1,not : 1,bad : 1. However, as we can see from the previous problem, there is contradiction for y value of not, good, bad in the four mini-review, and with linear classifier, the weight for these features would all be zeros, for example:

With good:1 the w[good] equals to 1 and then with good:-1, the weight changes to 0. Similarity problem applies to both not and bad. The additional feature can be two-word feature like: not bad: 1 and not good: -1, and then there are four features: bad: -1, good: 1, not bad: 1, not good: -1. Then this kinds of single and double words feature eliminates the problem states above.

Problem 2: Predicting Movie Ratings

Suppose that we are now interested in predicting a numeric rating for each movie review. We will use a non-linear predictor that takes a movie review x and returns $\sigma \mathbf{w} \phi(x))$, where $\sigma(z)$ = (1+ez)1 is the logistic function that squashes a real number to the range [0,1]. Suppose that we wish to use the squared loss.

(a) Write out the expression for Loss(x,y,$\mathbf{w}$).

$$\text{Loss(x,y,}\mathbf{w}\text{)} = (\sigma(\mathbf{w}\phi(x)) - y)^2$$

(b) Compute the gradient of the loss.

Predict value $p = \sigma(\mathbf{w}\phi(x))$ and then the gradient of the loss is: Loss(x,y,$\mathbf{w}$) = $2\frac{dp(w)}{du}(p(w) - y)$

(c) Assuming y=0, what is the smallest magnitude that the gradient can take? That is, find a way to set w to make $||\nabla Loss(x, y, \mathbf{w})||$ as small as possible. You are allowed to let the magnitude of w go to infinity. Hint: try to understand intuitively what is going on and the contribution of each part of the expression. If you find doing too much algebra, you're probably doing something suboptimal.
Since y = 0: Loss(x,y,$\mathbf{w}$) = $(\sigma(\mathbf{w}\phi(x)))^2$
then the gradient of loss is: $\nabla Loss(x, y, \mathbf{w}) = 2(\sigma(\mathbf{w}\phi(x)))(\sigma(\mathbf{w}\phi(x)))(1-(\sigma(\mathbf{w}\phi(x)))||\phi(x)||$
and also: $\sigma(\mathbf{w}\phi(x)) = \frac{1}{1+e^{-\mathbf{w}\phi(x)}}$
if $\mathbf{w}$ goes to infinity, then $\sigma(\mathbf{w}\phi(x))$ equals to 1 to 0 based on sign of the corresponding feature value. But anyway, either $\sigma(\mathbf{w}\phi(x))$ equals to 1 or zero when w goes to infinity will lead to the magnitude of the gradient equals to zero, and this is the smallest magnitude that the gradient can take.

(d) Assuming y=0, what is the largest magnitude that the gradient can take? Leave your answer in terms of $||\phi(x)||$
The gradient of loss function can be expressed using p: $p = \sigma(\mathbf{w}\phi(x))$. To maximize the value of $\nabla Loss(x, y, \mathbf{w})$, we need to maximize $p^2(1-p)$. Take derivative of $p^2(1-p)$ and set to zero:

$$\frac{dp^2(1-p)}{dp} = 2p(1-p) - p^2 = 0$$

$$2(1-p) - p = 0 \text{ and } p = \sigma(\mathbf{w}\phi(x)) = \frac{2}{3}$$

Therefore, the maximum value of $\nabla Loss(x, y, \mathbf{w})$ occurs when $\sigma(\mathbf{w}\phi(x)) = \frac{2}{3}$. And the maximum value of $||\nabla Loss(x, y, \mathbf{w})||$ equals to $\frac{8}{27}||\phi(x)||$

(e) The problem with the loss function we have defined so far is that is it is non-convex, which means that gradient descent is not guaranteed to find the global minimum, and in general these types of problems can be difficult to solve. So let us try to reformulate the problem as plain old linear regression. Suppose you have a dataset D consisting of (x,y) pairs, and that there exists a weight vector w that yields zero loss on this dataset. Show that there is an easy transformation to a modified dataset D' of (x,y') pairs such that performing least squares regression (using a linear predictor and the

squared loss) on D' converges to a vector w that yields zero loss on D'. Concretely, write an expression for y' in terms of y and justify this choice. This expression should not be a function of w.
Set $y' = y^2$. Therefore, the squared loss is: Loss(x,y,$\mathbf{w}$) = $(\mathbf{w}\phi(x) - y^2)^2$

Problem 3: Sentiment Classification

In this problem, we will build a binary linear classifier that reads movie reviews and guesses whether they are "positive" or "negative."
(d) When you run the grader.py on test case 3b-2, it should output a weights file and an error-analysis file. Look through 10 example incorrect predictions and for each one, give a one-sentence explanation of why the classification was incorrect. What information would the classifier need to get these correct? In some sense, there's not one correct answer, so don't overthink this problem; the main point is to get you to get intuition about the problem.
Examples 1: 'it's painful to watch witherspoon's talents wasting away inside unnecessary films like legally blonde and sweet home abomination , i mean , alabama . '
The 'painful' is a strong indication of bad review, however, the weights for this word is towards positive.

Example 2: wickedly funny , visually engrossing , never boring , this movie challenges us to think about the ways we consume pop culture .
never boring is a positive comments, but the algorithm treat them separately which leads to a false prediction.

Example 3: i didn't find much fascination in the swinging . what they're doing is a matter of plumbing arrangements and mind games , of no erotic or sensuous charge . but that they are doing it is thought-provoking .
Again, the algorithm treat 'no erotic' which leads to false prediction

Example 4: a heady , biting , be-bop ride through nighttime manhattan , a loquacious videologue of the modern male and the lengths to which he'll go to weave a protective cocoon around his own ego.
the weights for some neutral words are very negative value: male: -0.31 and around -0.35

Example 5: arty gay film .
Not enough information in the comments, it's easy for the logarithm to mis-judge the comments

Example 6: . . . although this idea is " new " the results are tired .
Two quotation marks gives significant high positive weights, 0.27, which lead to wrong positive conclusion.

Example 7: depicts the sorriest and most sordid of human behavior on the screen , then laughs at how clever it's being .
The comments give bad words for the description of the movie plot rather than the review on movies. But the algorithm would would treat those bad words as the comments for the movie.

Example 8: not sweet enough to liven up its predictable story and will leave even fans of hip-hop sorely disappointed .
Algorithm treated 'not sweet' separately. Disappointed does not have strong negative weight as it should be. It only has a weight equals to -0.01

Example 9: these guys seem great to knock back a beer with but they're simply not funny performers .
the algorithm treated 'not funny' separately and gave neutral word 'back' a very positive weight

Example 10: until its final minutes this is a perceptive study of two families in crisis – and of two girls whose friendship is severely tested by bad luck and their own immaturity .
Algorithm treated some bad plot description words as the review and comments for the movie.

Summary: The major issue for the algorithm is the separate treating of words that representing opposite meaning when those words appear alone. The algorithm should expands feature to two words or even three words.

(f) Run your linear predictor with feature extractor extractCharacterFeatures. Experiment with different values of n to see which one produces the smallest test error. You should

observe that this error is nearly as small as that produced by word features. How do you explain this?
The error get smallest when n = 5. The reason for that is the length of most words in the training set is about 5 characters. Therefore, when n=5, the algorithm will have a better understanding and interpretation for the reviews and have better prediction.

Problem 4: K-means clustering

Suppose we have a feature extractor ϕ that produces 2-dimensional feature vectors, and a toy dataset D_{train}=$\{x_1, x_2, x_3, x_4\}$ with
1. $\phi(x_1)$=[0,0]
2. $\phi(x_2)$=[0,1]
3. $\phi(x_3)$=[2,0]
4. $\phi(x_4)$=[2,2]

(a) Run 2-means on this dataset. Please show your work. What are the final cluster assignments z and cluster centers ? Run this algorithm twice, with initial centers:
1. μ_1 = [1,0] and μ_2 = [3,2]
Step one: minimize z.
Distance between $\phi(x_1)$ and μ_1 is 1; Distance between $\phi(x_1)$ and μ_2 is $\sqrt{13}$. Therefore, assign $\phi(x_1)$ to z_1
Distance between $\phi(x_2)$ and μ_1 is $\sqrt{2}$; Distance between $\phi(x_2)$ and μ_2 is $\sqrt{10}$. Therefore, assign $\phi(x_2)$ to z_1
Distance between $\phi(x_3)$ and μ_1 is 1; Distance between $\phi(x_3)$ and μ_2 is $\sqrt{5}$. Therefore, assign $\phi(x_3)$ to z_1
Distance between $\phi(x_4)$ and μ_1 is $\sqrt{5}$; Distance between $\phi(x_4)$ and μ_2 is 1. Therefore, assign $\phi(x_4)$ to z_2
Step two: set mu.
$\mu_1 = [1, \frac{1}{2}]$ and $\mu_2 = [2, 2]$
Step three: minimize z.
Distance between $\phi(x_1)$ and μ_1 is 1.11; Distance between $\phi(x_1)$ and μ_2 is 2.83. Therefore, assign $\phi(x_1)$ to z_1
Distance between $\phi(x_2)$ and μ_1 is 1.11; Distance between $\phi(x_2)$ and μ_2 is 2.24. Therefore, assign $\phi(x_2)$ to z_1
Distance between $\phi(x_3)$ and μ_1 is 1.11; Distance between $\phi(x_3)$ and μ_2 is 2. Therefore,

assign $\phi(x_3)$ to z_1
Distance between $\phi(x_4)$ and μ_1 is 1.80; Distance between $\phi(x_4)$ and μ_2 is 0. Therefore, assign $\phi(x_4)$ to z_2
And the above is the final cluster assignments z and cluster centers

2. μ_1 = [1,1] and μ_2 = [0,2]
Step one: minimize z.
Distance between $\phi(x_1)$ and μ_1 is 1.41; Distance between $\phi(x_1)$ and μ_2 is 2. Therefore, assign $\phi(x_1)$ to z_1
Distance between $\phi(x_2)$ and μ_1 is 1; Distance between $\phi(x_2)$ and μ_2 is 1. Therefore, assign $\phi(x_2)$ to z_2
Distance between $\phi(x_3)$ and μ_1 is 1.41; Distance between $\phi(x_3)$ and μ_2 is 2.83. Therefore, assign $\phi(x_3)$ to z_1
Distance between $\phi(x_4)$ and μ_1 is 1.41; Distance between $\phi(x_4)$ and μ_2 is 2. Therefore, assign $\phi(x_4)$ to z_1
Step two: set mu.
$\mu_1 = [2, \frac{3}{2}]$ and $\mu_2 = [0, 1]$
Step three: minimize z.
Distance between $\phi(x_1)$ and μ_1 is 2.5; Distance between $\phi(x_1)$ and μ_2 is 1. Therefore, assign $\phi(x_1)$ to z_2
Distance between $\phi(x_2)$ and μ_1 is 2.24; Distance between $\phi(x_2)$ and μ_2 is 0. Therefore, assign $\phi(x_2)$ to z_2
Distance between $\phi(x_3)$ and μ_1 is 1.50; Distance between $\phi(x_3)$ and μ_2 is 2.24. Therefore, assign $\phi(x_3)$ to z_1
Distance between $\phi(x_4)$ and μ_1 is 0.5; Distance between $\phi(x_4)$ and μ_2 is 2.24. Therefore, assign $\phi(x_4)$ to z_1
Step four: set mu.
$\mu_1 = [0, \frac{1}{2}]$ and $\mu_2 = [2, 1]$
And the above is the final cluster assignments z and cluster centers

(b)

(c) Sometimes, we have prior knowledge about which points should belong in the same cluster. Suppose we are given a set S of example pairs (i, j) which must be assigned to the same cluster. For example, suppose we have 5 examples; then S=$\{(1, 2), (1, 4), (3, 5)\}$

says that examples 1, 2, 4 must be in the same cluster and that examples 3 and 5 must be in the same cluster. Provide the modified k-means algorithm that performs alternating minimization on the reconstruction loss.