CS 221 – Autumn 2018
Anand Natu
anatu - 06264867

Homework #2: Sentiment

Problem 1

Problem 1a

We have a weight vector of dimension 6 for the 6 different unique words appearing in the review. For the first pass we have

$$Loss_{hinge} = \max\{0, 1 - [0,0] \cdot [1,1](-1)\} = 1, \therefore \nabla\big(Loss_{hinge}\big) = 0$$

$$\therefore w \leftarrow 0 - (0.5)(0) = 0$$

From the above, we see that with our initialized weights being zero, the margin for the first pass is 1, but since the gradient $\nabla\big(Loss_{hinge}(x, y, w)\ @\ margin = 1\big) = 0$, we see that the weights are unable to update via gradient descent, which in turn reduces the weight vector to all zeros, the same as the value to which they were initialized:

$$\boldsymbol{w} = [\mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{0}]$$

Problem 1b

Suppose that we have the following reviews as our dataset:

- (-1) not good
- (-1) bad
- (+1) good
- (+1) not bad

The fundamental property of the linear classifier is that its learning score is a linear combination of weights and features:

$$y = f(Score), \qquad Score \propto w \cdot \phi(x) = \sum_{i=1}^{n} w_i \phi_i(x)$$

This means that in order for the classifier to have zero error, the features must be linearly separable, i.e. for two categories of points $X_1, X_2\ \epsilon\ \mathbb{R}^n$ there must be some hyperplane with component coefficients $w_1, w_2 \ldots w_n$ such that every point in X_1 satisfies $\sum_{i=1}^{n} w_i \phi_i(x) > 0$ and each point in X_2 satisfies $\sum_{i=1}^{n} w_i \phi_i(x) < 0$ (assuming the hyperplane passes through the origin).

However, since the feature-vector is a mapping of word frequencies, **linear separation becomes geometrically impossible, because there is no linear / constant separator which can divide the word frequencies on their own.** Plotting the sample feature vectors from our dataset, which exist in $\mathbb{R}^3$, illustrates this point:

CS 221 – Autumn 2018
Anand Natu
anatu - 06264867

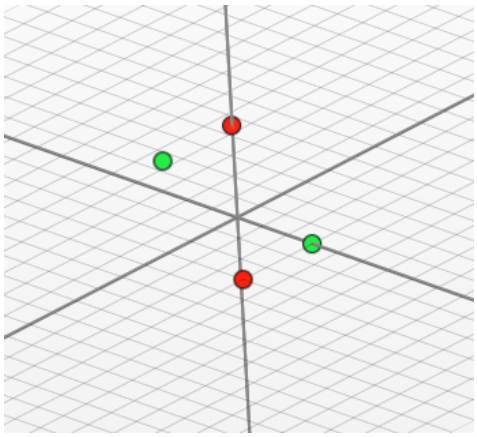

Figure 1: Plot of the sample feature points in $\mathbb{R}^3$. Red points denote bad reviews (y=-1) while green denote good reviews (y=1). Note that no 3D plane exists which can perfectly the two classes of data, indicating linear inseparability of the classification in its existing dimensionality.

In order to fix this problem we have to augment the feature vector somehow. Since part of the problem is that the existing features are used in both classification outcomes, one such augmentation that would fix it would be **to include bigrams of the words as features, e.g. counting the number of instances of "not good" in succession as opposed to just the words "not" and "good" individually within the feature vector $\boldsymbol{\phi(x)}$.** This will augment the feature vector into a higher dimension in which linear separability is possible.

Problem 2

Problem 2a

Since we are using the squared loss, its general expression is given by

$$Loss_{sq} = (f_w(x) - y)^2$$

In this case, we use a nonlinear predictor in the form of a sigmoid which is of the form

$$f_w(x) = \sigma(w \cdot \phi(x)), where\ \sigma(z) = (1 + e^{-z})^{-1}$$

By substitution we then have

$$Loss_{sq} = \left[\left(1 + e^{-(w \cdot \phi(x))}\right)^{-1} - y\right]^2$$

Problem 2b

Computing the gradient of the loss, we apply the chain rule, letting $p = \sigma(w \cdot \phi(x))$ such that:

$$\frac{d(Loss)}{dw} = 2 * (p - y) * \frac{d}{dw}(p - y)$$

Applying the derivative identity for the sigmoid function, $\frac{d}{dz}\sigma(z) = \sigma(z)(1 - \sigma(z))$, we have

$$\frac{d(Loss)}{dw} = 2 * (p - y) * \big(p(1 - p)\big) \cdot \frac{dp}{dw}$$

And, since $p = \sigma(w \cdot \phi(x))$, we have

$$\frac{d(Loss)}{dw} = 2\phi * (p - y) * \big(p(1-p)\big)$$

Problem 2c

Substituting y=1 into the above for the SGD run on the data point provided in the problem statement, we have

$$\frac{d(Loss)}{dw} = \nabla_w Loss = 2\phi(p-1)\big(p(1-p)\big) = 2\phi(p-1)(p-p^2) = 2(p^2 - p^3 - p + p^2)$$
$$= 2\phi(-p^3 + 2p^2 - p)$$

Finding the points which make the gradient small involve finding the roots of the polynomial expression on the left-hand side, which has a non-trivial root at p=1. Therefore, it is **theoretically** possible to obtain a zero-gradient result given the above analytical solution. However, recall that

$$p = \sigma\big(w \cdot \phi(x)\big) = \big(1 + e^{-(w \cdot \phi(x))}\big)^{-1}$$

Therefore the underlying weights which obtain this zero gradient are given by

$$p|_{\nabla Loss = 0} = \sigma\big(w \cdot \phi(x)\big) = \big(1 + e^{-(w \cdot \phi(x))}\big)^{-1} = 1$$

Which simplifies to

$$e^{-(w \cdot \phi(x))} = 0$$

Therefore, for some given value of $\phi(x)$, it follows that

$$\boldsymbol{\lim_{w \to \infty} e^{-(w \cdot \phi(x))} = 0}$$

Based on the above, **it is not actually possible to get the magnitude of the gradient to zero, since it only approaches zero as the magnitude of the weights approaches an infinitely large magnitude.** This is easily verified by visual inspection of the sigmoid curve, i.e. the curve "flattens" for very large values of the independent variables, which mathematically translates to a smaller gradient magnitude.

Problem 2d

Writing out the gradient in its full form using the same data point as above, we have

$$\nabla Loss = 2\phi(-p^3 + 2p^2 - p)$$

We know that the largest possible magnitude will be in the case when the weights are zero, so that we can substitute w=0 which yields the expression for the maximum magnitude of the gradient of the loss function as:

CS 221 – Autumn 2018
Anand Natu
anatu - 06264867

$$|\nabla Loss|_{w=0} = |\nabla Loss|_{max} = 2\phi\left(-\left(\frac{1}{1+e^{-w\phi}}\right)^3 + 2\left(\frac{1}{1+e^{-w\phi}}\right)^2 - \left(\frac{1}{1+e^{-w\phi}}\right)\right)$$
$$= 2\phi\left(-\left(\frac{1}{1+1}\right)^3 + 2\left(\frac{1}{1+1}\right)^2 - \left(\frac{1}{1+1}\right)\right) = 2\phi\left(-\frac{1}{8} + 2\left(\frac{1}{4}\right) - \frac{1}{2}\right) = 2\phi\left(-\frac{1}{8}\right)$$

Therefore,

$$\boldsymbol{|\nabla Loss|_{max} = -\frac{1}{4}||\phi||}$$

Problem 2e

For some dataset $D \in (x, y)$, there is a value of the weights w such that $Loss = 0$. For $D \in (x, y)$, the loss function is potentially non-convex. We seek to transform into a modified dataset $D' \in (x, y')$ where

$$Score = f = \phi(x) \cdot w, \qquad Loss_{sq} = (f_w(x) - y')^2 = (\phi(x) \cdot w - y')^2$$

The original definition of the prediction output is given by the sigmoid of the score:

$$y = \sigma\big(w \cdot \phi(x)\big) = \frac{1}{1 - e^{-(w \cdot \phi(x))}}$$

We can rearrange this expression:

$$\frac{1}{y} = 1 - e^{-(w \cdot \phi(x))}$$

$$1 - \frac{1}{y} = e^{-(w \cdot \phi(x))}$$

$$\ln\left(\frac{y-1}{y}\right) = -\big(w \cdot \phi(x)\big)$$

$$\ln\left(\frac{y}{y-1}\right) = w \cdot \phi(x)$$

The RHS of the above expression now reflects the predictor of the linear case that we seek to obtain. Therefore, it follows that in order to transform from the non-linear sigmoid classification into an ordinary linear regression, we must set

$$\boldsymbol{y' = \ln\left(\frac{y}{y-1}\right)}$$

Problem 3

Problem 3d

Review 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 a negative review, but the classifier fails because it cannot predict on any words that, by themselves, are obviously negative (e.g. “goes” has the most negative weight at -0.28, but it’s not enough to pull the classification down to a negative review against words that are obviously positive, like “funny” with +0.42 weight). This could be addressed by augmenting the feature set to have more syntactical understanding (e.g. n-grams of words)

Review 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 this example, the classifier mis-predicts because it has learned on confusing training examples. First of all, the reference to Sweet Home Alabama, and the word “painful” has a positive weight likely due to positive reviews of sad movies. Therefore, a more diverse training set across genres would help to alleviate the issue.

Review 3

=== wickedly funny , visually engrossing , never boring , this movie challenges us to think about the ways we consume pop culture .

Truth: 1, Prediction: -1 [WRONG]

Here, the classifier has trouble resolving the positivity of this review into its scores mainly because of the use of the negative in “never boring”. Boring scores as the most aggressive weight in the set (-1.15), which significantly drags the score into negative territory. Use of 2-grams in the feature set would help to solve this problem.

Review 4

=== rain is a small treasure , enveloping the viewer in a literal and spiritual torpor that is anything but cathartic .

Truth: 1, Prediction: -1 [WRONG]

The complexity of the vocabulary used here is what confuses the classifier. The rare word “torpor” has the most negative weight, and the positive words “spiritual”, “enveloping”, etc. are not common enough for them to have strong positive weights. Augmenting the input data to contain more verbose reviews / normalize for word frequency would alleviate this issue.

Review 5

=== 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]

This is an example of a complex grammatical structure which confuses the classifier. Some words clearly stand out as positive or negative (best, does, can’t), but many of the weights scatter closely around zero, and there is no real basis for the classification. Once again, using improved feature extraction (e.g. n-grams) can help here, or controlling for writing style of the reviewers in the training data.

CS 221 – Autumn 2018
Anand Natu
anatu - 06264867

Problem 3f

The reason why *n*-grams can give us a smaller test error is that they provide more flexibility in how we represent features, which means we can better account for cases in which the test set contains data that does not appear in the training set. For example, consider the review:

"That was poorer than the prequel, and maybe the poorest of them all".

The above will likely be better with n-grams than with words. This is because if our training set, say, only has the word "poor", our words-based feature set will not give the classifier any way of understanding that "poorer" and "poorest" are conjugated from it. However, e.g. 4-grams will be able to capture the common gram and root word "poor", and incorporate this information into the learning.

Problem 4

Problem 4a

To cluster using the K-means algorithm, we first assign each data point to its closest centroid using

$$z_i \leftarrow \arg\min \left|\left|\phi(x_i) - \mu_k\right|\right|^2$$

Then, we will update the location of the centroids based on the points that fall into their clusters as

$$\mu_k \leftarrow \frac{1}{|\{i: z_i = k\}|} \sum_{i:z_i=k} \phi(x_i)$$

With data

$$\phi(x) = \{x_1 = [1,0], x_2 = [1,2], x_3 = [3,0], x_4 = [2,2]\}$$

First, we have $\mu_1 = [2,3], \mu_2 = [2,-1]$. This leads to assignments of

$$z_1 = \{x_2, x_4\}, z_2 = \{x_1, x_3\}$$

Which then results in updated centroids of

$$\mu_1 = \left[\frac{3}{2}, 2\right], \mu_2 = [2,0]$$

Which then gives new assignments of

$$z_1 = \{x_2, x_4\}, z_2 = \{x_1, x_3\}$$

Since these assignments are the same as before, the update step is identical to before, which means the above is the final converged result $\boldsymbol{z_1 = \{x_2, x_4\}, z_2 = \{x_1, x_3\}, \mu_1 = \left[\frac{3}{2}, 2\right], \mu_2 = [2,0]}$

Second, we have $\mu_1 = [0,1], \mu_2 = [3,2]$. The first assignment step gives

$$z_1 = \{x_1, x_2\}, z_2 = \{x_3, x_4\}$$

Which then updates the centroids to

$$\mu_1 = [1, 1], \mu_2 = \left[\frac{5}{2}, 1\right]$$

Which in turn gives new assignments of

$$z_1 = \{x_1, x_2\}, z_2 = \{x_3, x_4\}$$

These assignments are the same as the previous iteration, indicating convergence at $\boldsymbol{z_1 = \{x_1, x_2\}, z_2 = \{x_3, x_4\}, \mu_1 = [1, 1], \mu_2 = \left[\frac{5}{2}, 1\right]}$

Problem 4c

In order to modify the K-means algorithm to minimize the reconstruction loss subject to the constraint function $S(i, j)$, we would proceed according to the same fundamental steps, with modifications made along the way:

1. **Initialize Centroids:** This is done in the same way as vanilla K-means (i.e. randomly). There is no modification to be made here.
2. **Calculate Distances:** Compute the distances of all data points to all of the K initialized clusters as $d_{x_i,k} = \left||\phi(x_i) - \mu_k|\right|^2$. There is no modification to be made here.
3. **Assignment Subject to Constraint Function:** Here, we modify by splitting assignment into two steps to account for the constraint.
    - **Assignment of unconstrained points:** Unconstrained points are assigned in the typical manner, i.e. using $z_i \leftarrow \arg\min\left||\phi(x_i) - \mu_k|\right|^2$
    - **Assignment of constrained points:** For the constrained points, we have a pre-defined set of "arbitrary" assignment sets $z_S, S \leq K$ which we must specify to our actual clusters μ_K. To do so, we calculate the WCSS (within-cluster sum of squares) for each z_s against each real cluster μ_K as $WCSS_{s,k} = \sum_{x_i \in z_s}(x_i - \mu_k)^2$, and associate the constrained assignment set z_s to the cluster that is closest (i.e. lowest loss) to where the points already are, i.e. $z_s \leftarrow z_k, where\ z_k = \arg\min\left||WCSS_k|\right|$. This minimizes the cost of assignment, and the number of iterations required for convergence.
4. **Update Centroids:** Update the centroids according to their constituent data points using $\mu_k \leftarrow \frac{1}{|\{i:z_i=k\}|}\sum_{i:z_i=k}\phi(x_i)$. There is no modification made here.
5. **Iteration**: Repeat steps 3-4 above until maximum # of iterations or convergence.

Another option to account for the constraint is at the initialization stage (i.e. initialize a cluster to immediately lie at the centroid of each constraint set), but this approach has less generality since it does not treat the real clusters and the constraint sets independently.

Problem 4d

The advantage of running K-means multiple times on the same dataset with different initializations is robustness – the nature of the centroid calculation and the K-means algorithm itself means that the converged outcome is deterministic from the initial values of the centroids. Therefore by testing many different initial values of the centroids, one can arrive at a final clustering answer which is most robust

(e.g. represents the statistical majority of results for different initial values), which provides confidence that the result obtained is not a contrived / edge case stemming from a certain set of initial values.

Problem 4e

If we scale all dimensions by the same factor, the clustering result will not change. This is because if all dimensions are scaled the same way, their relative position to one another will not change, which means their spatial proximity to centroids (and thus the eventual K-means clustering result) will be unaffected. (NOTE: This does not hold for any non-linear transformation of the original problem space, e.g. squaring all values in the original domain may result in different results)

However, if only some dimensions are scaled, then the relative positions of the data points and centroids change, which means that the final K-means result can change as a result of this type of scaling.