Corey McFarland
05/03/2021

HW 2

1 - Feature Map

1. In HW1, we binarized all categorical fields and kept the numerical fields, which makes sense for distance metrics. But for perceptron (and most other classifiers), it's actually better to binarize the numerical fields as well (so that “age" becomes many binary features such as “age=40"). Why?

   This allows you to weight various numbers and potential groups, rather than letting the numerical field itself be a weight or a weight multiplier. This allows for more consistent adjustments to the weight of the numerical fields.

2. How many binary features do you get in total after binarizing all fields? (Hint: around 230; Recall that HW1 had about 90 features).

   I had 230 features from the training data, plus an additional feature for bias for a total of 231 features.

2 - Perceptron and Averaged Perceptron

1. Implement the basic perceptron algorithm. Run it on the training data for 5 epochs (each epoch is a cycle over the whole training data, also known as an “iteration"). After each epoch, print the number of updates (i.e., mistakes) in this epoch (and update %, i.e., # of updates =|D|), and evaluate on the dev set and report the error rate and predicted positive rate on dev set, e.g., something like:

   epoch 1 updates 1257 (25.1%) dev_err 23.8% (+:27.5%)
   ...
   epoch 5 updates 1172 (23.4%) dev_err 20.5% (+:17.7%)

   Q: what's your best error rate on dev, and at which epoch did you get it? (Hint: should be around 19%).

   Epoch 3 was where my lowest dev error rate of 17.5% occurred:

   epoch: 3 updates: 1177 (23.5%) dev_err 17.5% (+:21.5%)

2. Implement the averaged perceptron. You can use either the naive version or the smart version (see slides).

   Note: the difference in speed between the two versions is not that big, as we only have around 230 features, but in HW4 the difference will be huge, where we'll have sparse features.

   Run the averaged perceptron on the training data for 5 epochs, and again report the error rate and predicted positive rate on dev set (same as above).

Q: This time, what's your best error rate on dev, and at which epoch did you get it? (Hint: around 15%).

Epoch 4 was where my lowest dev error rate of 14.7% occurred:

epoch: 4 updates: 1170 (23.4%) dev_err 14.7% (+:19.3%)

3. Q: What observations can you draw by comparing the per-epoch results of standard and averaged perceptrons? What about their best results?

   In general, the average perceptron had lower error rates than standard and it appeared that standard had a higher range of error rates. This showcases the under and over adjustments standard perceptron can make.

4. Q: For the averaged perceptron, what are the five most positive/negative features? Do they make sense?

   The features below made sense, representing those of older age (and likely more work experience) or those of the highest levels of education were more likely to be positive. Those on the opposite end of these fields were found in the most negative features.

   The five most positive features were:

   Ages: 58
   Country: Iran
   Education: Prof-School, Doctorate
   Marital Status: Married-civ-spouse

   The five most negative features (outside of the bias feature, which was the most negative) were:

   Ages: 24, 26, 28
   Education: 7^{th}-8^{th}
   Occupation: Farming-Fishing

5. Q: We know that males are more likely to earn >50K on this dataset. But the weights for Sex=Male and Sex=Female are both negative. Why?

   I believe this represents age, education, occupation, and other fields contributing more to the >50k income than gender.

6. Q: What is the feature weight of the bias dimension? Does it make sense?

   The feature weight of my bias dimension was -146,333 for average and -7 for standard perceptron, which makes sense as the majority of the examples are <50k income so the weight represents the likelihood of the person being in that category overall.

7. Q: Is the update % above equivalent to “training error"?

   Yes, as the updates occur as incorrect predictions over the training data are found. More updates show the model did not train as accurately resulting in an increased training error.

3 - Comparing Perceptron with k-NN

1. What are the major advantages of perceptron over k-NN? (e.g., efficiency, accuracy, etc.)

   Perceptron appeared to be significantly more efficient, as it achieved near same or lower errors rates than k-NN (especially when k was larger) in significantly less time. Accuracy appeared similar except for average perceptron, which had lower error rates than either standard perceptron or k-NN.

2. Design and execute a small experiment to demonstrate your point(s).

   Using the time it takes the k-NN algorithm to run k values of [50, 60, 70, 80, 90, 100], we will run the average perceptron algorithm to compare the number of epochs and the lowest, average, and range of error rate.

   The k-NN algorithm took 69.72 seconds to run and resulted in the following:

   Lowest error: 17.84% - k = 70, 90
   Highest error: 18.00% - k = 60
   Average error: 17.91%

   The average perceptron algorithm ran a total of 70 epochs and resulted in the following:

   Lowest error: 14.70% - epoch 4
   Highest error: 15.90% - epoch 8
   Average error: 15.51%

   The highest error rate for average perceptron was lower than the lowest error rate for k-NN. Additionally, perceptron ran 70 epochs compared to the 6 points utilized in finding an optimal k in k-NN.

4 - Experimentations

1. Try this experiment: reorder the training data so that all positive examples come first, followed by all negative ones. Did your (vanilla and averaged) perceptron(s) degrade in terms of dev error rate(s)?

   Yes, updates decreased significantly and error rate degraded to greater than or equal to 23.6%, with average perceptron performing worse than simple.

2. Try the following feature engineering:

   (a) adding the original numerical features (age, hours) as two extra, real-valued, features.

(b) centering of each numerical dimension (or all dimensions) to be zero mean;

(c) based on the above, and make each numerical dimension (or all dimensions) unit variance.

(d) adding some binary combination features (e.g., edu=X-and-sector=Y)

Q: which ones (or combinations) helped? did you get a new best error rate?

Combinations appeared to contribute to reducing the error rate with Industry + Gender, Industry + Citizenship, and Martial Status + Gender appearing to contribute the most individually. When combining these metrics, only Industry + Gender and Martial Status + Gender reduced the error rate further and resulted in a new best error rate of 14.2%:

epoch: 3 updates: 1161 (23.2%) dev_err 14.2% (+:20.6%)

3. Collect your best model and predict on income.test.blind to income.test.predicted. The latter should be similar to the former except that the target (>=50K, <50K) field is added. Do not change the order of examples in these files.

   Q: what's your best error rate on dev? which setting achieved it? (should be able to get 14% or less)

   My best error rate using the same model as above (average perceptron with additional features for Industry + Gender and Martial Status + Gender) resulting in an error rate of 14.2%.

   Q: what's your predicted positive % on dev? how does it compare with the ground truth positive %?

   The predictive positive % using this mapping was 20.6%, which is 3.0% less than the actual predictive positive % of the dev set of 23.6%

   Q: what's your predicted positive % on test?

   The predicted positive % is 21.5%.

Debriefing

1. Approximately how many hours did you spend on this assignment?

   10-12 hours

2. Would you rate it as easy, moderate, or difficult?

   It felt easy overall as it tied very well with the first homework.

3. Did you work on it mostly alone, or mostly with other people?

   Mostly alone

4. How deeply do you feel you understand the material it covers (0%-100%)?

   I feel I understand almost all of the covered material (95%), I would say that areas I could improve my understanding would be in possible feature manipulations.

5. Any other comments?

   None, other than the homework continues to be interesting.