Fatima Khalid
fxk200007

Project 1

Metrics Reporting

Multinomial Naive Bayes on Bag of Words Model

Dataset	Accuracy	Precision	Recall	F1 Score
Enron1	0.9364	0.9286	0.8725	0.8997
Enron2	0.9393	0.9244	0.8462	0.8835
Enron4	0.9724	0.9724	0.9898	0.9810

Discrete Naive Bayes on Bernoulli Model

Dataset	Accuracy	Precision	Recall	F1 Score
Enron1	0.7346	0.9118	0.2081	0.3388
Enron2	0.7762	0.8966	0.2000	0.3270
Enron4	0.9208	0.9009	1.0000	0.9479

Logistic Regression on Bag of Words

Dataset	Accuracy	Precision	Recall	F1 Score	Best lambda
Enron1	0.9583	0.9063	0.9732	0.9385	0.001
Enron2	0.9205	0.9182	0.7769	0.8417	0.001
Enron4	0.9484	0.9353	0.9974	0.9653	0.001

Logistic Regression on Bernoulli

Dataset	Accuracy	Precision	Recall	F1 Score	Best lambda
Enron1	0.9474	0.9433	0.8926	0.9172	0.001
Enron2	0.9017	0.9109	0.7077	0.7965	0.001
Enron4	0.9448	0.9287	1.0000	0.9631	0.001

Hyperparameter Tuning

Lambda: I tuned the L2 regularization parameter lambda by:

1. splitting the training data 70/30 as detailed in the project brief; 70% was a training subset and 30% was for validation
2. Then I tested 5 values for lambda:
    a. .001
    b. .01
    c. .1
    d. 1
    e. 10
3. This gave me a wide range to assess, but I discovered lambda values .001-1.0 often gave me identical F1 scores. Only 10 constantly did poorly.
4. By printing out the f1 value at every step, I was able to see which lambda was best.

This showed me that too much regularization (high lambda) was bad for the overall performance but the model wasn't too sensitive to fluctuation when it came to lower values. This could be because of a sparse feature space or well trained model that's not super prone to overfitting.

Max Iterations: I chose 300 iterations to cap runtime but still allowing sufficient room for convergence.

Learning Rate: 0.1 gave a good balance between stability and convergence speed, based on my research and observation.

Performance Analysis

1. Which combination of algorithm and data representation yielded the best performance? Why?

On average, the Logistic Regression algorithm with Bag of Words representation yielded the best performance because the F1 scores across all 3 datasets are highest (0.9385 + 0.8417 + 0.9653). I am taking F1 as the metric of choice because it's a good balance of recall and precision, which is most important for our task of detecting spam. Catching spam, a result of high recall, is balanced by not mistaking ham for spam (high precision). I also wanted to avoid accuracy as the sole metric because the dataset is not guaranteed to be balanced.

I think reasons for this combinations' impressive performance lie in a few factors. The Bag of Words representation preserves frequency information (repeating words are very key in spam emails) and at the same time, logistic regression is much more flexible than Naive Bayes algorithms. I say flexible because it doesn't make strong independence assumptions unlike Naive Bayes, which does. This way more complex info can be determined from the emails. Also, having a lambda parameter definitely helped prevent overfitting.

2. Did Multinomial Naive Bayes perform better than Logistic Regression on the Bag of Words representation? Explain.

On average, Multinomial Naive Bayes performed slightly better with slightly higher average F1 scores but the difference is not incredibly noticeable. As seen above in the tables, Logistic Regression did fairly better on Enron1. But generally, MNB outperformed. Some reasons could be that MNB is known to be well suited for text classification, especially ones with count based features. The conditional independence assumption in MNB could be pretty reasonable and robust, especially since we have a relatively limited number of training examples.

If we look at metrics apart from F1, it's interesting to notice LR does better on recall but worse on precision. In the real world LR could honestly be helpful since it's usually worse to misclassify a real email than to see a spam one.

3. Did Discrete Naive Bayes perform better than Logistic Regression on the Bernoulli representation? Explain.

No, it did not perform better. It performed significantly worse on Enron1 and Enron2 and slightly worse on Enron4 as shown in the tables above. This can be because DNB has really low recall (.2081 and .2000 for Enron1 and 2). Many spam emails were missed probably because binary representation didn't work well with DNB's independence assumption. DNB was stingy with classifying emails as spam unless it was really confident, indicated by high precision/low recall. I think this could be because the Bernoulli representation loses a lot of information by discarding word frequency.