Name: Seowoo Han
Andrew ID: seowooh
Nickname: khan5555

Machine Learning for Text Mining

Homework 5 – Template

1. Statement of Assurance

1. Did you receive any help whatsoever from anyone in solving this assignment? Yes / No.

Insoo Kim and Euiyoung Oh explained to me what is asked in the whole question and gave an institution.

2. Did you give any help whatsoever to anyone in solving this assignment? Yes / No.

3. Did you find or come across code that implements any part of this assignment? Yes / No.

https://www.csie.ntu.edu.tw/~cjlin/papers/liblinear.pdf

https://www.codeproject.com/Articles/821347/MultiClass-Logistic-Classifier-in-Python

2. Data Preprocessing

(1) **[5 pts]** After your finishing the data preprocessing, report the top 9 frequent tokens and corresponding counts in the report.

Rank	Token	Count	Rank	Token	Count	Rank	Token	Count
No. 1	good	742807	No. 2	place	716146	No. 3	food	691499
No. 4	great	585143	No. 5	like	546150	No. 6	just	521348
No. 7	time	442451	No. 8	service	431304	No. 9	really	387324

(2) **[5 pts]** Before continuing to the next step, another interesting problem is to check the star distribution of training samples. Report the count of training samples for each star (i.e., 1 to 5).

Star	1	2	3	4	5
# of training data	128038	112547	178215	373469	463084
Percentage	10.1994	8.9654	14.1964	29.7501	36.8887

Do you find something unexpected from the distribution (e.g., whether the dataset is balanced)?

The ratio of Star 4 to Star 5 is over half. I thought it would be an average of three, but I was surprised to find that between 4 and 5 was the average. In this situation, we can say that the distribution is skewed to one side.

Will this be a problem in training the model?

Train sets may not represent the whole. If so, it's a problem, but if you represent it fully, it doesn't matter.

If so, could you give some ideas about how to address it and explain why your idea should work?

When learning with a train set, the k-fold method can reduce the skew of data. Another solution is to scale the current star's score from 1 to 5, rescale it to 0 and 1. If the scale is rescaled, the skewed

distribution between 4 and 5 points will be less than the other scores, which will reduce the skew to one side.

3. Model Design

(1) **[5 pts]** Show that the gradient of regularized conditional log-likelihood function with respect to the weight vector of class c (i.e., $\frac{\partial l(W)}{\partial w_c}$) is equal to

$$\sum_{i=1}^{n}\left(y_{ic}-\frac{e^{\mathbf{w}_c^T\mathbf{x}_i}}{\sum_{c'=1}^{C}e^{\mathbf{w}_{c'}^T\mathbf{x}_i}}\right)\cdot\mathbf{x}_i-\lambda\mathbf{w_c}$$

Notice that the gradient of log-likelihood function with respect to a vector w_c is itself a vector, whose i-th element is defined as $\frac{\partial l(W)}{\partial w_{ci}}$, where w_{ci} is the i-th element of vector w_c.

This function $l(\mathrm{W})$ is the conditional log-likelihood function for the multi-class logistic regression model.

$$l(\mathrm{W})=\log\left(\prod_{i=1}^{n}\prod_{c=1}^{C}y_{ic}log\left(\mathrm{P}(y_i=\mathrm{c}|\boldsymbol{x}_i,\mathrm{W})\right)\right)-\frac{\lambda}{2}\sum_{i=0}^{C}\|\boldsymbol{w}_i\|^2,$$

$$=\sum_{i=0}^{n}(\sum_{c=1}^{C}y_{ic}{w_c}^T\boldsymbol{x}_i-log\sum_{c'=1}^{C}e^{{w_{c'}}^T\boldsymbol{x}_i})-\frac{\lambda}{2}\sum_{i=0}^{C}\|\boldsymbol{w}_i\|^2,$$

Let denotes the gradient of regularized conditional log-likelihood function as $\boldsymbol{g}_c(\mathrm{W})$.

$$\boldsymbol{g}_c(\mathrm{W})=\nabla_{\boldsymbol{w}_c}l(\mathrm{W}),$$

$$=\frac{\partial}{\partial\boldsymbol{w}_c}\sum_{\boldsymbol{i=1}}^{\boldsymbol{n}}(\sum_{c=1}^{C}y_{ic}{w_c}^T\boldsymbol{x}_i-log\sum_{c'=1}^{C}e^{{w_{c'}}^T\boldsymbol{x}_i})-\frac{\partial}{\partial\boldsymbol{w}_c}\frac{\lambda}{2}\sum_{i=0}^{C}\|\boldsymbol{w}_i\|^2,$$

$$=\sum_{i=1}^{n}\left(\frac{\partial}{\partial\boldsymbol{w}_c}\sum_{c=1}^{C}y_{ic}{w_c}^T\boldsymbol{x}_i-\frac{\partial}{\partial\boldsymbol{w}_c}log\sum_{c'=1}^{C}e^{{w_{c'}}^T\boldsymbol{x}_i}\right)-\lambda\boldsymbol{w}_c,$$

$$=\sum_{i=1}^{n}\left(y_{ic}\boldsymbol{x}_i-\frac{\frac{\partial}{\partial\boldsymbol{w}_c}\sum_{c'=1}^{C}e^{{w_{c'}}^T\boldsymbol{x}_i}}{\sum_{c'=1}^{C}e^{{w_{c'}}^T\boldsymbol{x}_i}}\right)-\lambda\boldsymbol{w}_c,$$

$$=\sum_{i=1}^{n}\left(y_{ic}\boldsymbol{x}_i-\frac{e^{{w_c}^T\boldsymbol{x}_i}\boldsymbol{x}_i}{\sum_{c'=1}^{C}e^{{w_{c'}}^T\boldsymbol{x}_i}}\right)-\lambda\boldsymbol{w}_c,$$

$$=\sum_{i=1}^{n}\left(y_{ic}-\frac{e^{{w_c}^T\boldsymbol{x}_i}}{\sum_{c'=1}^{C}e^{{w_{c'}}^T\boldsymbol{x}_i}}\right)\cdot\boldsymbol{x}_i-\lambda\boldsymbol{w}_c$$

(2) **[5 pts]** Let the learning rate be α, outline the algorithm (Batched-SGD) for implementation. You should cover how would you like to update the weights in each iteration, how to check the convergence and stop the algorithm and so on.

how would you like to update the weights in each iteration:

I upgraded weight using mini-batch SGD method. Multiplying W's gradient by the learning rate adds to W and finds the optimal concave.

how to check the convergence and stop the algorithm:

I repeated 5, 10, 50 and 100 times and compared the results. Just five or ten times, the values converged. But I thought it might be overfitting, so I designed the program to reduce the learning rate and increase the iteration.

(3) **[10 pts]** After implementing your model, please use these two types of prediction to calculate and report the Accuracy and RMSE (See definition in Evaluation part) on the entire training set with the two features designed in Task 2.

Feature	CTF		DF	
Dataset	Training	Development	Training	Development
Accuracy	0.6417	0.5880	0.6425	0.5886
RMSE	0.7679	0.7955	0.7670	0.7947
Parameters Setting	Learning Rate alpha=0.001 Regularization Parameter lambda=0.001 How many iterations used:50 Batch size=1000			

[10 pts] Multi-class Support Vector Machine

After you figure them out, report only the accuracy on the training and development set using the two features designed in Task 2.

Feature	CTF		DF	
Dataset	Training	Development	Training	Development
Accuracy	0.5931	0.5888	0.5936	0.5898
Parameters Setting	#for training train.exe -s 0 SVM_train.txt #for predicting predict.exe SVM_train.txt SVM_train.txt.model predict_train.txt predict.exe SVM_dev.txt SVM_train.txt.model predict_dev.txt predict.exe SVM_test.txt SVM_train.txt.model predict_test.txt			

4. Feature Engineering

[10 pts] Describe in details your most satisfying design and the corresponding considerations, use formula to illustrate your idea if necessary. Besides, report the evaluation results on training and development set here (The reported result here should match the record on the leaderboard).

My implementation showed that DF performed better than CTF. The reason DF performs better is that when people star, the words they use affect the star rating. But it is not effective to count the same

person's repeated use of the same word. So how many documentations appear in a document that counts how many times a CTF counts up to a single word used by a person is better.

In addition, both CTF and DF showed better results than training. This is natural because we learned through training data set.

Comparing the results with the multi-class SVM shows that the RMLR is better. svm learns and predicts to minimize the margin of each class, but RMLR uses softmax to predict each class using probabilities. Minimizing margins is likely to have errors. However, finding the probability that belongs to each class and selecting the largest probability has a small probability of error.

5. One sentence of your feeling for this homework

This assignment can be said to represent the course. Up to homework 4, we had a partial implementation. But this assignment preprocessed big data, learned how to handle sparse data, and understood both SVM and logistic regression.

It took a long time to process the data. I didn't think the data was big, and I just wanted to store the entire matrix. Because of this, I faced the problem and solved it, and there was a lot of sparse data in the world.

Also, on other assignments, no matter how bad the code was. But the challenge is exponentially increasing execution time if the code is not written efficiently. Therefore, much effort has been put into writing code efficiently.

It was a task to have various experiences, and it was fun.