CSE 5243 Lab 2 Report

Young Suk Cho

Ph.D. student in Computer Science Engineering

cho.648@osu.edu

1. **Group Member**
    a. Young Suk Cho (Ph.D. student in CSE) – 100% contribution

2. **Problem Description**
    a. The first goal of this lab assignment is to implement two different types of classifiers that can predict an appropriate topic for the given Reuter articles that have missing topics.
        i. On the process of modeling, the model is expected to be evaluated based on some of testing techniques. Train/test split method was selected for this lab assignment.
    b. The other goal is to check the effectiveness of the model and the feature selection in terms of accuracy, training time (offline efficiency), and testing time (online efficiency).

3. **Description of Proposed Solution**
    a. To address this problem, I used another feature vector that is from body text terms besides a topic feature vector. To sum up I exploited a 21758 * 13823 feature vector from the body text and a 21758 * 120 feature vector from the topic text.
        i. To reflect the comment I was given from the earlier assignment, I converted all the mat files into txt files. However, I had to choose to use the existing TFIDF_fin.mat to get the relevant information since the file structure was too complex to be converted into a single txt file. If you need to check the content of the file to grade, you can get txt versions of it by executing "converToTxt.m" file.
    b. The project requires to test two classification models on two different feature vectors. Thus, I chose K-nearest neighbor (CL-1) and Naïve bayes classifiers (CL-2) to test on the whole set of body feature vector (FV-1, 13823 features) and a subset of this feature vector (FV-2, 6909 features). Regarding the feature selection I proceeded based on the following selection process.
        i. Selected the top TFIDF 6900 features from the whole dataset.
        ii. Find topic features that do not have their corresponding body feature in the training dataset. This can happen when a topic feature has its corresponding body feature that are beyond the selected 6900 features. There were nine of them. Thus, I found 9 more their corresponding body

features from the not-selected body feature vector set and included them in the finalized half feature set.

c. Once the half sized feature vector FV-2 was determined so I have two different types of feature vectors, I extracted the set of articles that have both of topic features and body features in them. There were 10328 of them and I made two matrices X and T that contains all those information. Each row of the both matrices represents an individual article. Each column of matrix X represents a selected body feature and that of T represents a selected topic feature.
d. From the matrices X and T, I made a 3 types of training and testing matrices by splitting them with the following ratios: 90/10, 80/20, 70/30.
e. Using those training and testing matrices, I trained the classifiers and get the predicted topics for each testing articles then compare the predicted topics with their corresponding ground truth topic in the testing topic matrices.

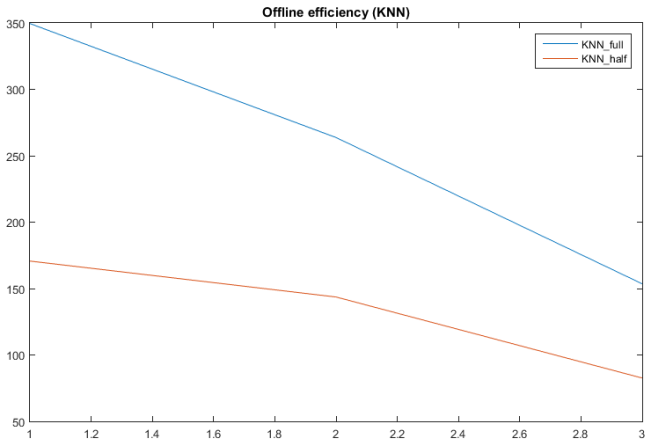
4. Processing Steps

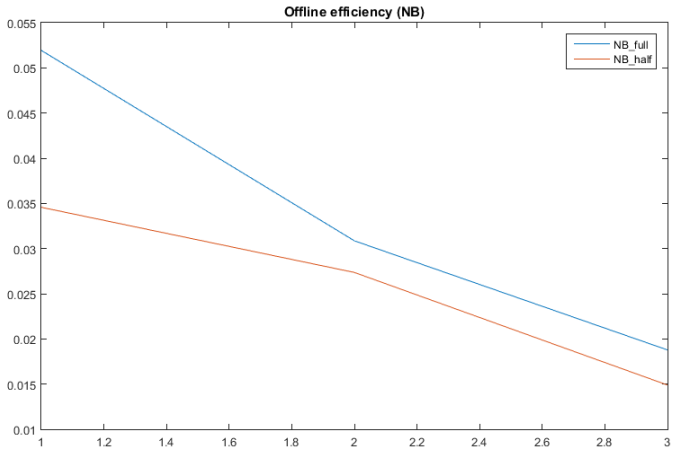
a. Read the two types of feature vectors (Topic, Body) that were made during the earlier lab assignments.
b. Let the whole body feature vector as FV-1 and the half size cut of it as FV-2. I selected the features of FV-2 based on the decreasing order of their TF-IDF values.
c. Train a KNN and a Naïve Bayes model and Time each training process to check offline efficiency.
d. Evaluate the models using the testing set and check the accuracies of all the experiment settings.

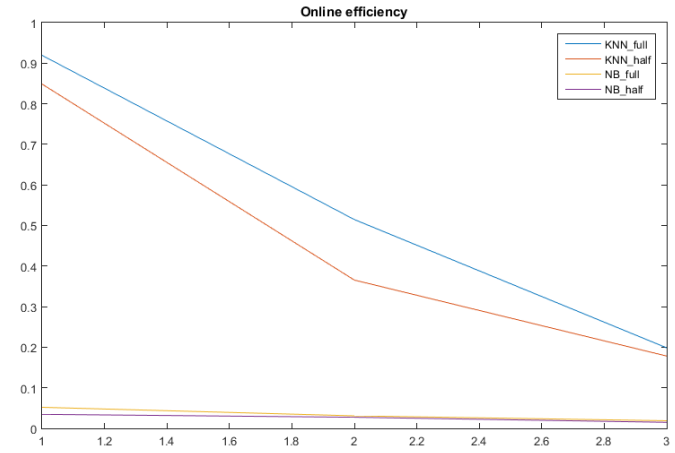
5. Experiments and Results

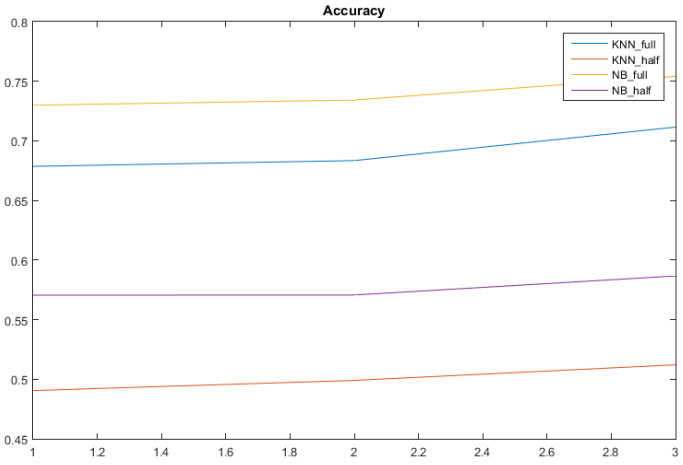
a. Experiments
    i. Accuracy was utilized to measure the performance of the two models on two different types of feature vectors and three different training/testing ratios.
    ii. A predicted topic is considered correct if it matches with the exact ground truth topic or one of the ground truth topics in the corresponding testing data element.
    iii. During the execution of Naïve Bayes, there were some zero probabilities that make all the probability information lost at the conditional probability calculation. To prevent it, I provided a small enough default value (0.00001) for such probabilities.
    iv. To determine the best topic from the KNN, I chose a single topic feature that most frequently appears among those from all of its five nearest neighbors.

		FV-1 (Whole set)			FV-2 (Half set)		
Train/Test ratio		**70/30**	**80/20**	**90/10**	**70/30**	**80/20**	**90/100**
CL-1 (KNN)	Offline Efficiency (Second)	349.55	263.70	153.55	170.70	143.63	82.60
		Average = 255.60			Average = 132.31		
	Online Efficiency (Second, for 2066 tuples)	0.92	0.51	0.20	0.85	0.37	0.18
		Average = 0.54			Average = 0.46		
	Accuracy (%)	67.86	68.34	71.15	49.05	49.90	51.21
		Average = 69.12			Average = 50.05		
CL-2 (Naïve Bayes)	Offline Efficiency (Second)	0.05	0.03	0.02	0.04	0.03	0.02
		Average = 0.03			Average = 0.03		
	Online Efficiency (Second, for 2066 tuples)	0.52	0.31	0.19	0.35	0.27	0.15
		Average = 0.34			Average = 0.26		
	Accuracy (%)	72.99	73.43	75.41	57.05	57.07	58.66
		Average = 73.94			Average = 57.59		




Offline efficiency (NB)
NB_full
NB_half
0.055
0.05
0.045
0.04
0.035
0.03
0.025
0.02
0.015
0.01
1
1.2
1.4
1.6
1.8
2
2.2
2.4
2.6
2.8
3


Online efficiency
KNN_full
KNN_half
NB_full
NB_half
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
1
1.2
1.4
1.6
1.8
2
2.2
2.4
2.6
2.8
3


Accuracy
KNN_full
KNN_half
NB_full
NB_half
0.8
0.75
0.7
0.65
0.6
0.55
0.5
0.45
1
1.2
1.4
1.6
1.8
2
2.2
2.4
2.6
2.8
3

b. Results
    i. Accuracies may vary depending on the random seed value provided to the stdlinux system since the Matlab program on the server does not support any random seed value control function (rng(seed_value)).
    ii. Online and offline efficiency also may vary depending the performance of the server itself.

6. Output Interpretation

a. Since there is no clear cut between training and testing in KNN model, I assumed the offline efficiency as the duration from the beginning of the execution to the end of K-nearest neighbor calculation. For this reason, I assumed the online efficiency as the duration to. The results shows that the KNN model took much more time to be trained than that trained by the Naïve Bayes model.
b. Naïve Bayes not only ran faster than the KNN, but also showed better performance in general. I think the reason why KNN took longer is that it has to calculate five nearest neighbor (topic) for every data element in the training set contrast to the Naïve Bayes which requires only single probability calculation for each feature-topic pair.
c. In terms of feature selection, the larger training set showed the better accuracy achieved in my experiments. I believe the reason is that there was almost no garbage data in the feature vector I used. I ran a script to query every word to online dictionary and get the original form from it so I was able to keep garbage data in my feature vector as minimum as possible. The accuracies from the larger dataset were stable over several repeated experiments. However, the accuracies from the smaller dataset varied considerably depending on the given random seed values.
d. There was no significant difference in online efficiencies between the two models regardless of the size of both datasets.
e. The split ratio of the training and testing datasets did not affect to the accuracy significantly but in general, the more training set generated the higher accuracy in the experiment.

7. Applied External Libraries

a. I implemented KNN algorithm on my own but changed to adopting the following Matlab built-in KNN library for efficiency.
(http://www.mathworks.com/help/stats/classification-using-nearest-neighbors.html)
b. I adopted some utility functions that are built-in functions of the latest version of Matlab but not provided by stdlinux Matlab from open source websites as listed below:
https://github.com/mim/mimlib
The code is already located in the directory that has all the .m files.