Text Mining

Assignment: 2

(15 points)

Q.1 (4 Points)

Build a rule based classifier to predict if a document addresses the topic "EARN" or not. Use the 'cls' and 'tm' package provided with this assignment to do this task.First create training and testing dataset for the classifier; use 'create_training_data' and 'create_testing_data' modules in 'cls' package to do this task. Use decision-tree as a rule based classifier on this data; utilize 'train_decision_tree' module for this task. Print the rules obtained from the learnt tree using 'print_decision_tree' module. Obtain predictions over the test dataset using 'evaluate_classifier' module. Finally report precision-recall values of predictions using 'compute_evaluation_metrics' module.

Or if you wish to use TMSK

Try riktext on the train/test vectors created in week-1 for the category "EARN". Note that the program name is "riktext" with an "x". Also note that the category name specified on the command-line should be some string that does NOT occur in the dictionary. So you can't use "earn" (all lowercase). But if you don't like EARN, you can use E1 or CLASS or whatever. Note that you will need the dictionary used to create the vectors. Use the example in section 3.2 of the riktext documentation as a guide. Use a default properties file and submit the ruleset file you obtain using 2-fold cross-validation on the training set. See above for instructions on how to upload your answer.

Q.2 (2 Points)

Try some variants of decision-tree module to induce different rule learning strategies. To do this, change the 'train_decision_tree' module in 'cls' package. Look closely at tree.DecisionTreeClassifier() to see how you can pass different parameters to this function to induce different learning strategies. HINT: This function uses scikit-learn decision tree module 'sklearn.tree.DecisionTreeClassifier' which takes two forms of initialization such as 'criterion' and 'max_depth' to learn a different tree.

Or if you wish to use TMSK

Use the program to examine other induced solutions. See if you can modify the rule file to manually improve performance on the test set. What is the problem with this manual approach to tuning rules?

Q.3 (1 Point)

Verify the probabilities in Figure 3.17 (see text) and compute the class for a document that has w1 and w2.

Q.4 (0 Point)
NO NEED FOR A RESPONSE

Try out the naïve bayes classifier as an alternative to decision-tree in above question. Use 'train_naive_bayes' module from the package for this task. Obtain predictions using this method and observe new precision-recall values of the predictions.

Or if you wish to use TMSK

Try out the nbayes and testnbayes programs on the same train/test sets you used for riktext. Remember that nbayes and testnbayes are part of TMSK. Besides the vector files, note that you must use the same dictionary you used for riktext. Be sure you have the infile and vectorfile parameters set correctly in tmsk.properties. Look at the examples in sections 4.3 and 4.4 of the TMSK documentation to see what the command line should look like.

Q.5 (0 Point)
NO NEED FOR A RESPONSE

Try out a linear classifier as an alternative to decision-tree in above question. Use 'train_linear_model' module from the package for this task. Obtain predictions using this method and observe new precision-recall values of the predictions.

Or if you wish to use TMSK

Now try the linear classifier on the train/test vectors. Use the programs linear and testline. These too are part of TMSK. Note that the files specified in tmsk.properties have to be in sync. For example, the index file (if specified) has to correspond to the vector file. Otherwise the program reads inconsistent data from different files and things blow up!! Note that the index file is an optional parameter. It can be safely commented out in the properties file. The main benefit of specifying the index file is somewhat faster execution. It is probably not noticeable in these assignments.

Q.6 (1 Point)
Try the linear classifier with tf*idf features. Use 'extract_tfidf_feature' from the 'cls' package to do this task. How do the results differ with the solution in (5)?

Or if you wish to use TMSK

Try the linear classifier with tf*idf features. How do the results differ with the solution in (5)?

Q.7 (2 Points)
Report precision-recall values for linear and decision tree based classifier. Use 'compute_evaluation_metrics' module for this task.

Or if you wish to use TMSK

Study precision-recall tradeoff in both linear and riktext. Discuss why precision goes down when recall goes up.

Q.8 (1 Point)
Generate a shorter dictionary from the current dictionary by taking the top 50 words. Generate vectors with this shorter dictionary using 'extract_frequent_k_bow_feature' module. Compare the precision-recall performance of decision tree and linear classifier on the same train/test sets. Discuss the differences with the results from the larger dictionary.

Or if you wish to use TMSK

Generate a shorter dictionary from the current dictionary by taking the top 50 words. Generate vectors with this shorter dictionary and compare the performance of riktext and linear on the same train/test sets. Discuss the differences with the results from the larger dictionary.

Q.9 (2 Points)
Lets say you have a document collection that changes rapidly in a non-cumulative fashion and you wish to use it for classifying new documents. Which kind of classifier would be most appropriate for this task?

Q.10 (2 Points)
Can you think of what advantages the robust loss function has over the hinge loss function?

<u>Optional</u> part for students with knowledge of Python

Questions 1-2, 4-8 can be done in Python instead of using TMSK/Riktext. Since they build on the Python solution of assignment-1, be sure to try the model solution and make sure you run it successfully and obtained the data needed for this assignment.

We have provided a sample python module 'cls' along with the python module 'tm' from previous assignment which contain some basic building blocks useful for this assignment. You can freely import 'cls' and 'tm' modules to write your answers.