Project 1

Machine Learning Course

Naive Bayes and Logistic Regression for Text Classification

In this project you will implement and evaluate Naive Bayes and Logistic Regression for text classification.

Note from the TA: You must use Python 3 to implement your algorithms.

0 Points Download the spam/ham (ham is not spam) datasets (see the zip file). The datasets were used in the Metsis et al. paper [1]. There are three datasets. You have to perform the experiments described below on all the three datasets. Each data set is divided into two (sub)sets: training set and test set. Each of them has two directories: spam and ham. All files in the spam and ham folders are spam and ham messages respectively.

20 points Convert the text data into a matrix of features × examples (namely our canonical data representation), using the following approaches.

- **Bag of Words model:** Recall that we use a vocabulary having w words—the set of all unique words in the training set—and represent each email using a vector of word frequencies (the number of times each word in the vocabulary appears in the email).
- **Bernoulli model:** As before we use a vocabulary having w words and represent each email (training example) using a 0/1 vector of length w where 0 indicates that the word does not appear in the email and 1 indicates that the word appears in the email.

Thus you will convert each of the three "text" datasets into two datasets, one using the Bag of words model and the other using the Bernoulli model. You can use any text processing tool or library (e.g., NLP toolkit) to accomplish this task. However, please clearly cite them in your report.

20 points Implement the multinomial Naive Bayes algorithm for text classification described here: `http://nlp.stanford.edu/IR-book/pdf/13bayes.pdf` (see Figure 13.2 in the document). Note that the algorithm uses add-one laplace smoothing. Make sure that you do all the calculations in log-scale to avoid underflow. Use your algorithm to learn from the training set and report accuracy on the test set. **Important:** Use the datasets generated using the Bag of words model and not the Bernoulli model for this part.

20 points Implement the discrete Naive Bayes algorithm we discussed in class. To prevent zeros, use add-one laplace smoothing. Make sure that you do all the calculations in log-scale to avoid underflow. Use your algorithm to learn from the training set and report accuracy on the test set. **Important:** Use the datasets generated using the Bernoulli model and not the Bag of words model for this part.

30 points Implement the MCAP Logistic Regression algorithm with L2 regularization that we discussed in class (see Mitchell's new book chapter). Try different values of λ. Divide the given training set into two sets using a 70/30 split (namely the first split has 70% of the examples and the second split has the remaining 30%). Learn parameters using the 70% split, treat the 30% data as validation data and use it to select a value for λ. Then, use the chosen value of λ to learn the parameters using the full training set and report accuracy on the test set. Use gradient ascent for learning the weights (you have to set the learning rate appropriately. Otherwise, your algorithm may diverge or take a long time to converge). Do not run gradient ascent until convergence; you should put a suitable hard limit on the number of iterations. **Important:** Use the datasets generated using both the Bernoulli model and the Bag of words model for this part.

10 points Run the SGDClassifier from scikit-learn on the datasets. Tune the parameters (e.g., loss function, penalty, etc.) of the SGDClassifier using GridSearchCV in scikit-learn. Compare the results you obtain for SGDClassifier with your implementation of Logistic Regression. **Important:** Use the datasets generated using both the Bernoulli model and the Bag of words model for this part.

What to Turn in

- Your code and a Readme file for compiling and executing your code.
- A detailed write up that reports the accuracy, precision, recall and F1 score obtained on the three test sets for the following algorithms and feature types:
  - Multinomial Naive Bayes on the Bag of words model
  - Discrete Naive Bayes on the Bernoulli model
  - Logistic Regression on both Bag of words and Bernoulli models
  - SGDClassifier on both Bag of words and Bernoulli models

  Your report should also describe how you tuned the hyper-parameters for logisitic regression and SGDClassifier for each dataset (specifically values of λ used, hard limit on the number of iterations, etc.). The write up should be self contained in that we should be able to replicate your results.
- Answer the following questions:
  1. Which data representation and algorithm combination yields the best performance (measured in terms of the accuracy, precision, recall and F1 score) and why?
  2. Does Multinomial Naive Bayes perform better (again performance is measured in terms of the accuracy, precision, recall and F1 score) than LR and SGDClassifier on the Bag of words representation? Explain your yes/no answer.
  3. Does Discrete Naive Bayes perform better (again performance is measured in terms of the accuracy, precision, recall and F1 score) than LR and SGDClassifier on the Bernoulli representation? Explain your yes/no answer.
  4. Does your LR implementation outperform the SGDClassifier (again performance is measured in terms of the accuracy, precision, recall and F1 score) or is the difference in performance minor? Explain your yes/no answer.

References

[1] V. Metsis, I. Androutsopoulos and G. Paliouras, “Spam Filtering with Naive Bayes - Which Naive Bayes?”. Proceedings of the 3rd Conference on Email and Anti-Spam (CEAS 2006), Mountain View, CA, USA, 2006.