Homework 3

Machine Learning

Due on April 5 at 12:00PM (noon) on Canvas

Ham or Spam

Email has been one of the most important forms of communication. Spam is one of the major threats posed to email users. Links in spam emails may lead to users websites with malware or phishing schemes. Therefore, an effective spam filtering technology is a significant contribution to the sustainability of the cyberspace and to our society. Our client company provides a collection of spam emails as well as regular work and personal emails that came from its employee. The client company asked us to design an effective spam-filtering system. In order to increase the effectiveness of spam detection, we plan to analyze the content of the email to determine if the email is spam. After pre-processing the content of the emails, we have extracted 57 features from those emails. Your task now is to build a spam filter to predict whether an email is a spam or not.

Data

In the dataset (see spambase.csv) provided by the client company, the first 48 columns indicate the feature word_freq_WORD, the percentage of words in the email that match WORD (number of times the WORD appears in the email/total number of words in email * 100). The following 6 columns indicates the feature char_freq_CHAR, the percentage of characters in the email that match CHAR. The next column indicates the feature capital_run_length_longest, the length of longest uninterrupted sequence of capital letters. The last feature is capital_run_length_total, the total number of capital letters in the email. The last three features have been standardized for improving the efficiency of training process of the models. The last column indicates whether the email was considered as spam (1) or not (0). There are in total 4600 instances in this dataset.

Programming assignment

In this assignment, you will implement Support Vector Machine on the spambase dataset to predict whether the email is spam or not using Python. You can either use the functions from libraries such as scikit-learn or write the code from scratch to implement the algorithm. After you run the sample code (see, "svm_demo.py") successfully (this step is not required, but recommended), you should apply the Support Vector Machine following the instructions as below. In this assignment, you should use linear kernels and set the penalty coefficient C to 1 if those parameters are not specified.

1. Learn and apply the algorithm (20pts)

    a. For each W in [10, 20, 30, 40, 50, 57]

        i. Use the first W features and randomly sample 70% of the data for training and use the remaining 30% of the data for testing

        iii. Learn a model from the training data and apply it to test data

        iv. Measure the performance on both the training and test data using accuracy

        v. Repeat 100 times with different random samples (using the same W)

        vi. Record and report the average and the standard deviation of accuracy on both training and test sets across the 100 trials

    b. Plot curves for the results (the number of features vs avg., the number of features vs std. dev.) for both training and test data. Discuss the results.

2. Explore the effect of the penalty coefficient C (25pts)

    a. Randomly sample 80% of the data for training and use the remaining 20% of data for testing.

    b. For each C in [0.001, 0.01, 0.1, 1, 10]

        i. Use 10-fold cross validation on the training set to evaluate the accuracy of the SVM with the penalty coefficient C.

        ii. Record and report the average and the standard deviation of accuracy on 10 validation sets.

iii. Record and report the average margin of the SVM that you trained.

c. Plot curves for the results (the penalty coefficient vs the avg. accuracy, the penalty coefficient vs the std. dev. of accuracy, the penalty coefficient vs the avg. margin). Discuss the results.

d. Choose a penalty coefficient C and give the reason why you choose this penalty coefficient.

e. Learn the SVM model on the **whole** training set with the chosen penalty coefficient C and report the accuracy on the testing set.

3. Comparison with Logistic Regression (15pts)

   a. Evaluate the accuracy of SVM and Logisitc Regression using 10-fold cross validation on the whole dataset.

   b. Record and report the average and the standard deviation of the accuracy of SVM and Logistic Regression on the validation sets.

   c. Record and report the average precision and average recall of SVM and Logistic Regression on validation sets.

   d. Is one classifier better than the other based on the performance metric accuracy? Use the paired t-test to justify your conclusion.

Hint

1. For part 1, you may use `sklearn.svm.SVC()` to train the model.

2. For part 2, you may use `sklearn.model_selection.KFold` to implementation k-fold cross validation.

3. For part 3, you may use `sklearn.metrics.recall_score` and `sklearn.metrics.precision_score` to evaluate your models.

4. HMK3 requires more running time that HMK2. Please start working on it as soon as you can to avoid late submission.

Submission

1. The source code in python.

   Name your file as "svm.py". You should include the code that you write for the above assignments. In addition, you should prepare the code such that TA can run without additional modifications and the output should report the results for the question (b) in the part 2 like this:

   ```
   $python svm.py

   The average accuracy with different penalty coefficients:

   [0.111 0.333 0.777 0.212 0.676 ]

   The std.  dev.  of accuracy with different penalty coefficients:

   [0.013 0.013 0.012 0.011 0.012]

   The average margin with different penalty coefficients:

   [1.432 2.212 0.266 3.154 1.089]
   ```

2. Your evaluation & analysis in .pdf format.

   Note that your analysis should include the graphs as well as a discussion of results.