CS589: Machine Learning - Spring 2017

Homework 1: Classification

Assigned: Thursday, Jan 26. Due: Thursday, Feb 9 at 11:55pm

Getting Started: You should complete the assignment using your own installation of Python 2.7. Download the assignment archive from Moodle and unzip the file. This will create the directory structure as shown below. The data files for each data set are in 'Data' directory respectively. You will write your code under the Submission/Code directory. Make sure to put the deliverables (explained below) into the respective directories.

```
HW01
--- Data
    |--Email_spam
    |--Occupancy_detection
    |--USPS_digits
--- Submission
    |--Code
    |--Figures
    |--Predictions
       |--Email_spam
       |--Occupancy_detection
       |--USPS_digits
```

Data Sets: In this assignment, you will experiment with different classifiers on three different classification problems: classify email from spam, detect occupancy of users in a building, identify hand written USPS digits. The basic properties of these data sets are shown below. Additional details are given in the readme.txt files contained in each data set directory.

Dataset	Training Cases	Test Cases	Dimensionality	Number of Classes
Email/spam	1813	1813	57	2
Occupancy detection	4750	4750	7	2
USPS digits	3540	3540	256	10

Each data set has been split into a training set and a test set and stored in NumPy binary format. The Submission/Code/run_me.py file provides example code for reading in all data sets.

Deliverables: This assignment has three types of deliverables: a report, code files, and Kaggle submissions.

- **Report:** The solution report will give your answers to the homework questions (listed below). The maximum length of the report is 5 pages in 11 point font, including all figures and tables. You can use any software to create your report, but your report must be submitted in PDF format.

- **Code:** The second deliverable is the code that you wrote to answer the questions, which will involve training classifiers and making predictions on held-out test data. Your code must be Python 2.7 (no iPython notebooks, other formats or code from other versions). You may create any additional source files to perform data analysis. However, you should aim to write your code so that it is possible to re-produce all of your experimental results exactly by running *python run_me.py* file from the Submissions/Code directory. 10% of your assignment grade is based on reproducibility and code quality.

- **Kaggle Submissions:** We will use Kaggle, a machine learning competition service, to evaluate the classifiers you create. You will need to register on Kaggle using a umass.edu email address to submit to Kaggle (you can use any user name you like). You will generate test prediction files, save them in Kaggle format (see example.csv file within each data directory to generate Kaggle compliant prediction files), and upload them to Kaggle for scoring. For your convenience we have provided a kaggle.py script that takes your predictions and converts them into Kaggle format. This script is located in the Submission/Code directory. Your scores will be shown on the Kaggle leaderboard, and 10% of your assignment grade will be based on how well you do in these competitions. You have a limit of 10 submission per day per data set. The Kaggle links for each data set are given below:

  - `https://inclass.kaggle.com/c/hw1-email-spam`
  - `https://inclass.kaggle.com/c/hw1-occupancy-detection`
  - `https://inclass.kaggle.com/c/hw1-usps-digits2`

Submitting Solutions: When you complete the assignment, place your final code in Submission/Code and the Kaggle prediction files for your single best-performing submission for each data set in Submission/Predictions/<Data Set>/best.csv. If you used Python to generate plots then place them in Submission/Figures. Finally, create a zip file called 'Submission.zip' from your 'Submission' directory only (do not include 'Data' directory). Only .zip files will be accepted for grading code (not .tar, .rar, .gz, etc.). You will upload your .zip file of code and your pdf report to Gradescope for grading. Further instructions for using gradescope will be provided on Piazza and discussed in class.

Academic Honesty Statement: Copying solutions from external sources (books, web pages, etc.) or other students is considered cheating. Sharing your solutions with other students is considered cheating. Posting your code to public repositories like GitHub is also considered cheating. Any detected cheating will result in a grade of 0 on the assignment for all students involved, and potentially a grade of F in the course.

Questions: Choose a set of three different classifiers to experiment with on these data sets. Using these classifiers, answer the following questions.

1. (*10 points*) **Classifier Selections:** Name and describe the three classifiers you selected. Give a mathematical expression for the form of the classification function for each classifier. Describe the hyperparameters that need to be set for each classifier. Describe why you decided to select each classifier.

2. (*15 points*) **Classifier Relationships:** Write a paragraph describing any relationships between the classifiers you selected. Are they related mathematically? Do they have similar decision boundary geometry?

What are their relative strengths and weaknesses in terms of speed, storage, flexibility, etc.

3. (*15 points*) **Default Classifier Accuracy:** Using the default hyperparameter values in the scikit-learn implementation of the classifiers you selected, train each classifier on each data set and use it to make predictions on the test data. Also record the total time needed to train the classifier and the total time needed to make all predictions. You will need to save your test predictions in the format expected by Kaggle (see the example code), and then upload the prediction file to the Kaggle competition page for scoring.

a. (*10 pts*) Create one bar chart each showing the test accuracy, training time, and prediction time for the three classifiers. Include the charts in your report (see Submission/Code/plotting.py for examples of using the Python plotting library to create bar charts).

b. (*5 pts*) Which classifier works best overall? Which is fastest? Which achieves the best speed-accuracy trade-off in your opinion?

4. (*20 points*) **Hyperparameter Selection Method:** Choose a method for setting the hyper-parameters of your classifiers.

a. (*5 pts*) Write a detailed description of the method including how data is partitioned and what criteria is used to select the best hyperparameter values.

b. (*5 pts*) Explain why you selected this approach and how it compares to other choices.

c. (*10 pts*) Implement the method you selected in Python from scratch and include a code listing in your report. Your grade will depend on the correctness and generality of your implementation. **Do not** use existing hyperparameter optimization code from scikit-learn or elsewhere.

5. (*20 points*) **Optimizing Hyperparameters:** Select a single classifier from among your three classifiers and a single data set. Report your choices.

a. (*10 pts*) Apply your chosen hyperparameter selection method to optimize the classifier's hyperparameters for the selected data set. Using a set of line graphs, report the range of hyper-parameter values you searched over, the score your hyperparameter selection method gave to each hyperparameter value, and the corresponding training error (see Submission/Code/plotting.py for examples of using the Python plotting library to create line plots). Include these figures in your report.

b. (*5 pts*) Use the optimal hyperparameters to make predictions on the test data. Upload the predictions to Kaggle and report the test accuracy obtained. How does the score returned by your approach compare to the score reported by Kaggle?

c. (*5 pts*) How does the accuracy of the classifier using optimal hyperparameters compare to the accuracy obtained using the default parameters? Do you see an improvement in perfromance? If not, explain why.

6. (*10 points*) **Get Creative:** Try anything you like to maximize classification accuracy. You can try additional classifiers, look at feature transformations, feature engineering, feature selection, building en-

semble classifiers, etc. You can focus on a single data set or try all of them. However, you need to provide a complete and accurate description of whatever methods you employ and document the experiments you performed and the accuracies you obtain. Make sure to post your best results for each data set to Kaggle before the assignment is due, and include your best scoring Kaggle prediction file for each data set with your homework submission. Points will be awarded based on how well your methods perform relative to baselines provided by the TAs, as well as how well your methods perform relative to the rest of the class. It is up to you to decide how much effort and time to put into this problem.

7. (*10 points*) **Reproducibility and Code Quality:** You should aim to design your code so that all experimental results shown in your report can be reproduced by running the run_me.py file, including the output of your best performing predictions on each data set. To achieve this, you will need to look at how the algorithms your code uses rely on random numbers, and control the starting states (seeds) of the corresponding random number generators. See the documentation for numpy.random.seed for more information. You are strongly encouraged to use the Python plotting library Matplotlib to create figures from your results programatically. Your code must also be sufficiently documented and commented that someone else (in particular, the TAs and graders) can easily understand what each method is doing. Adherence to a particular Python style guide is not required, but if you need a refresher on what well-structured Pythn should look like, see the Google Python Style Guide: `https://google.github.io/styleguide/pyguide.html`. You will be scored on how reproducible your results are (5 points), and how well documented and structured your code is (5 points).