CITS5508 Machine Learning
Semester 1, 2020

Lab Sheet 2

Assessed, worth 5%. Due: 11:59pm, Friday 20th March 2020

1 Outline

In this lab sheet, you will learn to develop Python code for a small classification task. You will also learn the *Markdown* syntax to put comments and explanation in your Jupyter Notebook file to improve the presentation of your work.

2 Submission

Name your Jupyter Notebook file as **lab02.ipynb** and submit it to LMS before the due date and time shown above. You can submit your file multiple times. Only the latest version will be marked.

3 Dataset

We will use the **Forest type mapping dataset** supplied on the UCI Machine Learning website:

http://archive.ics.uci.edu/ml/datasets/Forest+type+mapping#

Download the zip file from the webpage above. Unzip to extract the two files *training.csv* and *testing.csv*. A brief description about the dataset is given also on the webpage. Ensure that you save both files to the same directory with your *lab02.ipynb* file.
The training set (*training.csv*) contains 198 instances of multivariate remote sensing data of some forest areas in Japan. There are 4 different forest types labelled in the first column (the column heading is '*class*'), as described in the link above. The test set (file *testing.csv*) contains 325 test instances. This file has the same format as *training.csv*.
As the training set is smaller than the test set, in this dataset, we will swap them for the work in this labsheet. Before carrying out the tasks described below, you should rename *training.csv* to *test.csv* and rename *test.csv* to *training.csv*.

4 Tasks

Your tasks for this lab sheet are:

1. Read in the contents of both csv files[1]. Inspect what the columns are by displaying the first few lines of the file. Use appropriate functions to display (visualize) the different features (or attributes / columns). Display some plots for visualizing the data. Describe what you see.
2. To simplify the classification task, write Python code to remove all the columns whose names begin with `pred_minus_obs`. You should have only 9 features (`b1`, `b2`, $\cdots$, `b9`) left for both the training and test sets. The class labels can be extracted from the first column of both datasets.

[1] Since both files are in the same directory as your Jupyter Notebook file, you should be able to read each one of them without any path name. For example, `pd.read_csv('training.csv')` should work just fine.

3. Write Python code to count the number of instances for each class label. Do you have an imbalanced training set?
4. Perform appropriate data normalization before performing classification. You can use `MinMaxScaler`, `StandardScaler`, or any suitable normalization function in the `sklearn.preprocessing` package. You can also write your own normalization code if you prefer. Either way, ensure that you normalize the training data and the test data consistently.

   You should write a Python function that defines the pipeline for all the cleaning operations described above.
5. Use the *Support Vector Classifier* implemented in the `sklearn.svm.SVC` class to perform one-versus-one binary classification on the 4 class labels. Show the confusion matrix on the test set. You should try experimenting with two or three hyperparameters to see if you can improve the performance of the classifier.
6. Repeat the above step using the *Stochastic Gradient Descent Classifier* from the `SGDClassifier` class.
7. Compare the performance of the two classifiers and give a brief discussion about your experimental results.

5 Presentation

A few tips on the presentation of your `ipynb` files:

- Present your `ipynb` file as a portfolio, with *Markdown* cells inserted appropriately to explain your code. See the following links if you are not familiar with *Markdown*:
  - https://www.markdownguide.org/cheat-sheet/ (basic)
  - https://jupyter-notebook.readthedocs.io/en/stable/examples/Notebook/Typesetting%20Equations.html (more advanced)
- Dividing the portfolio into suitable sections and subsections (with section and subsection numbers and meaningful headings) would make your portfolio easier to follow.
- Avoid having too many small *Markdown* cells that have only one short sentence. In addition to *Markdown* cells, some short comments can be put alongside the Python code.
- Use meaningful variable names.
- When printing out your results, provide some textual description so that the output is meaningful.

6 Penalty on late submissions

See the URL below about late submission of assignments:
https://ipoint.uwa.edu.au/app/answers/detail/a_id/2711/~/consequences-for-late-assignment-submission