CITS5508 Machine Learning
Semester 1, 2021

Lab Sheet 2

Assessed, worth 5%. Due: 11:59pm, Friday 19th March 2021

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. This is the only file that you need to submit. Please do **not** submit any data file. Before the marking process, we will download a single copy of the dataset for everyone.

You can submit your lab02.ipynb file multiple times. Only the latest version will be marked.

3 Dataset

We will use the **Dry Bean dataset** supplied on the UCI Machine Learning webpage:

http://archive.ics.uci.edu/ml/datasets/Dry+Bean+Dataset

Click on the *Data Folder* link there and download the *DryBeanDataset.zip* file. After unzipping the file, you should see 3 files in a subdirectory named *DryBeanDataset*. You only need the *Dry_Bean_Dataset.xlsx* file for your Python code, but you should look at the webpage above or the *Dry_Bean_Dataset.txt* file to make sure you understand what the dataset is about.

4 Tasks

Put your **lab02.ipynb** file in the **same directory** with the the *DryBeanDataset* directory, i.e., your Python program should refer to the spreadsheet file as *DryBeanDataset/Dry_Bean_Dataset.xlsx* or *DryBeanDataset\Dry_Bean_Dataset.xlsx* depending on the operating system you use[1].

As the new version of the pandas library has stopped supporting `xlsx` files, to read in the spreadsheet of the dataset, you will need to install an additional package called *openpyxl*. To do so, activate your Python environment for the unit and type:

```
conda install openpyxl
```

Your tasks for this lab sheet are:

1. Use an appropriate function from the pandas library to read in the spreadsheet file. Inspect what the columns are by displaying the first few lines of the file. Use an appropriate function to display (visualise) the different features (or columns). Select 6 features that you think are interesting and illustrate the scatter matrix of these 6 features. Describe what you see.

[1] **Hint:** Use `os.path.join` to help you. See the sample codes from the author of the textbook.

2. The datasets from many real applications are usually imbalanced. Use an appropriate function to display the number of instances of each class. Which class has the fewest instances? Dealing with class imbalance is an advanced topic and won't be covered in this unit.
3. Use the `train_test_split` function to randomly split the data to form a training set and a testing set. Use a 80/20 split. Your subsequent training process should involve only the training set. How many instances are in your training and testing sets?
4. Perform an appropriate *feature scaling* step (see Chapter 2 of the textbook) on the training set. You can use either the `StandardScaler` or the `MinMaxScaler` class to do this. Make sure that you understand how the `fit`, `fit_transform`, and `transform` functions work and use them correctly in your code. Later on in the test stage on the testing set, you should apply the same transformation (computed from the training set) on the testing set (i.e., you should not compute a new feature scaling transformation for the test set).
5. Use the *Support Vector Classifier* implemented in the `sklearn.svm.SVC` class to perform one-versus-one binary classification on the training set. Choose appropriate hyperparameter values for your Support Vector Classifier and describe them in your markdown cells. You should be able to get reasonably good results if you use the `rbf` kernel. Show the confusion matrix on the testing set (**Hint:** use the `plot_confusion_matrix` function from the Scikit-Learn library).
6. Repeat step #5 above using the *Stochastic Gradient Descent Classifier* from the `SGDClassifier` class.
7. Compare the confusion matrices 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