Practical 2: Classification of object colour using optical spectroscopy

CS5014 Machine Learning

Due date: Friday 20th April (Week 10) 21:00
60% of the coursework grade

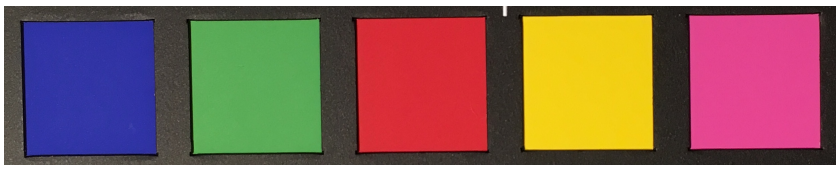

Figure 1: An image of the colours used in the dataset. From left to right: Blue, Green, Red, Yellow and Pink.

Aims

The main aim of this practical is to gain experience in working with real experimental, imperfect, and limited data which has not been analysed before. You will read, perhaps process/clean the data from the dataset. You will then create a classification model to predict output classes based on a set of inputs and evaluate its performance. It is particularly important that the evaluation and discussion is carefully described with the limitations of the data and of the method used also considered.

Task

On studres, you will find two datasets of optical reflectance spectroscopy data acquired with different colours placed underneath a spectroscopy device (Flame-S-Vis-NIR, Ocean Optics). Note a similar type of spectroscopy data is used as a comparator for a classification task in SpeCam.[1]

The file `binary.zip` contains data for a binary classification task (which forms the minimum basic requirement and is to be solved first). The file `multiclass.zip` contains data for a multi-class classification task (to be tackled after the binary classification task is finished – this is a more advanced requirement).

You are asked to predict the class of an object based on the spectrum of the object as shown in the file `XToClassify.csv` using the data provided in files `X.csv` and `y.csv` to create

[1] Hui-Shyong Yeo, Juyoung Lee, Andrea Bianchi, David Harris-Birtill, and Aaron Quigley. 2017. SpeCam: sensing surface color and material with the front-facing camera of a mobile device. In Proceedings of the 19th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI 17). ACM, New York, NY, USA, Article 25, 9 pages, `https://doi.org/10.1145/3098279.3098541`

your models for the binary and multiclass classification tasks. As in the first practical, the solution is expected to consist of several steps:

1. loading the data,
2. (optional) cleaning the data and creating new input features from the given dataset,
3. analysing and visualising the data,
4. preparing the inputs and choosing a suitable subset of features,
5. selecting and training a classification model,
6. selecting and training another classification model,
7. evaluating and comparing the performance of the models, and
8. a critical discussion of the results, your approach, the methods used and the dataset provided.

Each of these steps should be clearly explained in the report. You may find some of the steps more relevant than others, e.g. you may choose to use a subset of features or all of them, as long as you provide a justification for either decision. In all cases, you should show that you understand the consequences of each decision on the performance of your model and provide evidence showing how altering the decisions alters the model performance.

Try to keep the report informative and focussed on the important details and insights – the report also demonstrates an understanding of what is important. If you have large amounts of (relevant!) data, you can move them to an appendix and refer from the main text.

There are no existing results for this dataset, and there are many legitimate ways to approach this task; treat it as an open problem on which you can test everything covered in the module so far. While absolute classification rate is not the primary criterion for marking, we will be looking at how you approach improving the performance of your model.

Datasets

Each zip file contains five files:

- `X.csv`
- `Wavelength.csv`
- `y.csv`
- `key.txt`
- `XToClassify.csv`

The `X.csv` file contains a comma-separated values dataset where the rows are the samples and the columns represent the received optical reflectance intensity of various wavelengths of the electromagnetic spectrum for that sample (i.e. the spectrum).

The `Wavelength.csv` file contains the corresponding wavelength in nm for each column from `X.csv`.

The `y.csv` file contains the corresponding classification class ID (output) for each sample row from `X.csv`.

The `key.txt` file contains the key for converting the class ID into a text description of the class, in JSON format.

The `XToClassify.csv` file contains data in the same format as `X.csv`, however you do not have the corresponding output class ID for this data. Your program should output a file which provides the predicted class IDs for each row in this `XToClassify.csv` file. This output file should be called `PredictedClasses.csv` and have the same format as the ground truth provided in the `y.csv` file.

Deliverables

Hand in via MMS, by the deadline of 9pm on Friday of Week 10 (please leave enough time to upload your submission):

- The source code of your application.
- The predicted output file `PredictedClasses.csv` for each classification task (binary and multi-class). The output files should be copied into separate directories: `binaryTask` and `multiClassTask` ready for inspection. You should also include a list of the predicted classes as two tables as an appendix within your report, one for each task.
- A report in PDF format which contains details of each step of the process, justification for any decisions you take, and an evaluation of the final model. This should also contain evidence of functionality and any notable figures and tables you have produced.

Please create a `.zip` file containing all of these and submit this to MMS.

Marking and Extensions

This practical will be marked according to the guidelines at `https://info.cs.st-andrews.ac.uk/student-handbook/learning-teaching/feedback.html#Mark_Descriptor` Some examples of submissions in various bands are:

- A *basic implementation* **in the 11–13 grade band** is a submission which implements a classification model in a straight-forward way and contains some evaluation, but is lacking in quality and detail, for example only the binary classification task is attempted, or is accompanied by a weaker report which does not evidence good understanding.
- An implementation **in the 14–16 range** should complete all parts of the specification, including both the binary and multi-class classification tasks, consist of clean and understandable code, and be accompanied by a good report which clearly describes the process and reasoning behind each step and contains a good discussion of the achieved results including graphs and evaluation measures.
- To achieve a grade of **17 and higher**, your solution should extend a solid basic solution *in a meaningful way*. Potential extensions include comparison of multiple algorithms with meaningful evaluation and discussion of these, or applying multiple advanced algorithms from course textbooks and research publications. Unrelated extensions will be ignored.

Note that the goal is *solid machine learning methodology and understanding* rather than a collection of extensions – a good scientific approach and analysis are difficult, whereas running many different scikit-learn algorithms on the same data is easy. A basic solution can be based on a

logistic regression model, as long as the methodology and evaluation are sound. Be thorough in your basic solution and see extensions as a means to strengthen your basic argument and methodology. Also note that:

- We will not focus on software engineering practice and advanced Python techniques when marking, but your code should be sensibly organised, commented, and easy to follow.
- Standard lateness penalties apply as outlined in the student handbook at `https://info.cs.st-andrews.ac.uk/student-handbook/learning-teaching/assessment.html`
- Guidelines for good academic practice are outlined in the student handbook at `https://info.cs.st-andrews.ac.uk/student-handbook/academic/gap.html`