Practical 2: Classification of objects using a radar signal and machine learning

CS5014 Machine Learning

Due date: Thursday 18th April (Week 10) 21:00
60% of the coursework grade

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 and process and clean the data from the dataset in a suitable way. 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 is carefully described along with a discussion of the limitations of the data and of the method used.

Task

On studres, you will find two datasets of radar signals acquired by placing various objects on top of a radar chip (BGT60 - Infineon Technologies). The data were acquired in a similar (but not identical) way and for a similar purpose to that used in the paper RadarCat.[1]

The file `binary.zip` contains data for a binary classification task (to be solved first as the minimum basic requirement). 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 classification of an object based on the signal of the radar as shown in the XToClassify.csv using the X.csv and y.csv data provided to create 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,

[1]Yeo, H.-S., Flamich, G., Schrempf, P., Harris-Birtill, D., and Quigley, A. (2016) RadarCat: Radar Categorization for Input & Interaction. In Proceedings of the 29th Annual ACM Symposium on User Interface Software and Technology New York, NY, USA: ACM UIST 16, `https://doi.org/10.1145/2984511.2984515` & `https://sachi.cs.st-andrews.ac.uk/research/interaction/radarcat-exploits-googles-soli-radar-sensor-for-object-and-material-recognition/`
Please do not distribute the PDF to people outside the University for copyright reasons.

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.

Unlike the first practical, there is no baseline to compete against - these are new data. Some of you may wish to compare your results against those of your peers and discuss strategies and insights. 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.

Dataset(s)

Each zip file contains four files:

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

The X.csv file contains a dataset of comma-separated values where the rows represents individual samples and the columns are the components of the radar signal for that sample. Note that the data is structured as follows: there are 64 components to each radar receiver channel. There are 4 of these receiver channels stacked back to back, generating 256 components containing these 4 channels. The information from from many radar pulses (100) is used to calculate the mean signal, the minimum signal, and the maximum signal for each component. The data in the columns direction is therefore a concatenation of the mean, min and max signals each comprising 4 channels where each channel comprises 64 components. This gives a total of 768 potential inputs. Note that this is different from the data used in the RadarCat paper.

The y.csv file contains the corresponding classification class ID (output) for each sample row from X.csv. This can be used as ground truth for the supervised learning task.

The key.txt file contains the key for converting the class ID into a text description of the class. This file is in the 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 and will therefore not be useful for training. This forms the test dataset for the practical. You will write a program which outputs a file containing class ID predictions each row in this XToClassify file. This output file should be called PredictedClasses.csv. You should submit this file along with your submission.

Deliverables

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

- The source code of your application which works in the Python3 virtual environment set up in the school labs.
- 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 an appendix containing a table of predicted classes in 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 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 solved, 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, should 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, which can includ more advanced algorithms from course textbooks and research publications. Any applied algorithm must be accompanied by a critical comparison to any other algorithms you used, with a discussion of any differences. 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`