SIT220/731 2024.T3: Task 8HD

Evaluating Algorithms for Imbalanced Data

3rd January 2025

Contents

1 Introduction 1

2 Task 1

3 Additional Tasks for Postgraduate (SIT731) Students (*) 2

4 Artefacts 2

5 Intended Learning Outcomes 3

1 Introduction

> Tasks 5–8 are not obligatory, you can decide not to tackle them at all. C/D/HD is merely a subjective estimate of their difficulty level.

This task is due in **Week 11 (Sunday)**. Start tackling it as early as possible. If we find your first solution incomplete or otherwise incorrect, you will still be able to amend it based on the generous feedback we will give you. In case of any problems/questions, do hot hesitate to attend our on-campus/online classes or use the Discussion Board on the unit site.

Submitting after the aforementioned due date will incur a late penalty.

All submissions will be checked for plagiarism. You are expected to work independently on your task solutions. Never share/show parts of solutions with/to anyone.

2 Task

Imbalanced data is a common challenge in machine learning, where one class significantly outnumbers others. This task focuses on evaluating algorithms designed to address class imbalance in static datasets. You will replicate and adapt an evaluation framework inspired by the published paper (https://arxiv.org/pdf/2204.03719) on learning from imbalanced data streams. However, instead of handling data streams, this task will focus on static datasets.

You will:

1. Generate synthetic datasets with varying imbalance ratios: Create datasets with different class imbalance ratios based on the size of the majority class relative to the minority class, using the ratios $\{5, 10, 20, 50, 100\}$ (see Section 7.1.1 of the paper for details).

2. Test algorithms designed for imbalanced data: Select appropriate algorithms for testing, ensuring they are specifically designed to handle imbalanced data. You are not required to implement these algorithms yourself if suitable libraries or existing implementations are available.
3. Evaluate performance using relevant metrics: Assess the performance of the chosen algorithms using metrics such as Kappa, G-mean, and AUC (refer to Section 6.3 of the paper for a detailed explanation of these metrics).
4. Report findings and insights: Present your findings and insights in a formal manner, following the structure and style used by the authors in the paper (see Section 7 for an example of the report format).

3 Additional Tasks for Postgraduate (SIT731) Students (*)

There are no specific additional tasks, because the whole exercise has an open-ended formulation.

4 Artefacts

Make sure that your notebook has a **readable structure**; in particular, that it is divided into sections. Use rich Markdown formatting (text in dedicated Markdown chunks – not just Python comments).

Do not include the questions/tasks from the task specification. Your notebook should read nicely and smoothly – like a report from data analysis that you designed yourself. Make the flow read natural (e.g., *First, let us load the data on... Then, let us determine... etc.*). Imagine it is a piece of work that you would like to show to your manager or clients — you certainly want to make a good impression. Check your spelling and grammar. Also, use formal language.

At the start of the notebook, you need to provide: the **title** of the report (e.g., *Task 42: How Much I Love This Unit*), your **name**, **student number**, **email address**, and whether you are an **undergraduate (SIT220) or postgraduate (SIT731)** student.

Then, add 1–2 introductory paragraphs (an introduction/abstract – what the task is about).

Before each nontrivial code chunk, briefly **explain** what its purpose is. After each code chunk, **summarise and discuss the obtained results** (in a few sentences).

Conclude the report with 1–2 paragraphs (summary/discussion/possible extensions of the analysis etc.).

Limitations of the OnTrack ipynb-to-pdf renderer:

Ensure that your report as seen in OnTrack is aesthetic (see *Download submision PDF* after uploading the .ipynb file). The OnTrack ipynb-to-pdf renderer is imperfect. We work with what we have. Here are the most common Markdown-related errors.

- Do not include any externally loaded images (via the `![label](href)` Markdown command), for they lead to upload errors.
- Do not input HTML code in Markdown.
- Make sure you leave one blank line before and after each paragraph and bullet list. Do not use backslashes at the end of the line.
- Currently, also $LaTeX formulae$ and Markdown tables are not recognised. However, they do not lead to any errors.

Checklist:

1. Header, introduction, conclusion (Markdown chunks).
2. Text divided into sections, all major code chunks commented and discussed in your own words (Markdown chunks).
3. Every subtask addressed/solved. In particular, all reference results that are part of the task specification have been reproduced (plots, computed aggregates, etc.).
4. The report is readable and neat. In particular:
   - all code lines are visible in their entirety (they are not too long),
   - code chunks use consecutive numbering (select *Kernel - Restart and Run All* from the Jupyter menu),
   - rich Markdown formatting is used (`# Section Title`, `* bullet list`, `1. enumerated list`, `| table |`, `*italic*`, etc.),
   - the printing of unnecessary/intermediate objects is minimised (focus on reporting the results specifically requested in the task specification).

Submissions which do not *fully* (100%) conform to the task specification *on* the cut-off date will be marked as FAIL.

Good luck!

5 Intended Learning Outcomes

ULO	Is Related?
ULO1 (Data Processing/Wrangling)	YES
ULO2 (Data Discovery/Extraction)	YES
ULO3 (Requirement Analysis/Data Sources)	YES
ULO4 (Exploratory Data Analysis)	YES
ULO5 (Data Privacy and Ethics)	NO