CSCI316 – Big Data Mining Techniques and Implementation
Individual Assignment 1
2022 Session 1 (SIM)

15 Marks
Deadline: Refer to the submission link of this assignment on Moodle

Two (2) tasks are included in this assignment. The specification of each task starts in a separate page.

You must implement and run all your Python code in Jupyter Notebook. ***The deliverables include one Jupyter Notebook source file (with .ipybn extension) and one PDF document for each task.***

Note: To generate a PDF file for a notebook source file, you can either (i) use the Web browser's PDF printing function, or (ii) click "File" on top of the notebook, choose "Download as" and then "PDF via LaTex".

All results of your implementation must be reproducible from your submitted Jupyter notebook source files. In addition, the submission must include all execution outputs as well as clear explanation of your implementation algorithms (e.g., in the Markdown format or as comments in your Python codes).

Submission must be done online by using the submission link associated with assignment 1 for this subject on MOODLE. The size limit for all submitted materials is 20MB. DO NOT submit a zip file.

Submissions made after the due time will be assessed as late submissions. Late submissions are counted in full day increments (i.e. 1 minute late counts as a 1 day late submission). There is a 25% penalty for each day after the due date including weekends. The submission site closes four days after the due date. No submission will be accepted after the submission site has closed.

This is an individual assignment. Plagiarism of any part of the assignment will result in having 0 mark for the assignment and for all students involved.

Marking guidelines

Your Python code will be assessed. The computers in the lab define the standard environment for code development and code execution. Note that the correctness, completeness, efficiency, and results of your executed code will be assessed. Thus, code that produces no useful outputs will receive zero marks. This also means that code that does not run on a computer in the lab would be awarded zero marks or code where none of the core functions produce correct results would be awarded zero marks.

The clearness of explanations for your answers will be assessed.

Task 1

(6 marks)

Dataset: Risk Factor prediction of Chronic Kidney Disease Dataset
Source: https://archive.ics.uci.edu/ml/datasets/Risk+Factor+prediction+of+Chronic+Kidney+Disease

Objective
Use Pandas in Python to clean and pre-process the raw data in this dataset.

Requirements

(1) Load the row data into Pandas with the meta information being removed.
(2) Apply ordinal encoding to all columns contains numerical bins. For instance, in the "age" column, 0 for "<12", 1 for "12-20", 2 for "20-27", etc.
(3) Apply one-hot-encoding to the "stage" column.
(4) Find out and rank the correlations between the "class" column and all other columns

Deliverables

- A Jupiter Notebook source file named `<your_name>_task1.ipybn` which contains your implementation source code in Python
- A PDF document named `<your_name>_task1.pdf` which is generated from your Jupiter Notebook source file, and presents clear and accurate explanation of your implementation and results.

Task 2

(9 marks)

Dataset: The Secondary Mushroom Dataset
Source: https://archive.ics.uci.edu/ml/datasets/Secondary+Mushroom+Dataset

This dataset includes 61,069 hypothetical mushroom. Each mushroom is identified as edible, poisonous, or of unknown edibility and not recommended (the latter class was combined with the poisonous class).

Objective
The objective of this task is to implement *from scratch* Decision Tree classification method to predict whether the mushroom samples are poisonous (p) or editable (e) (thus a binary classification problem). The data are included in the file named "secondary_data.csv" (not "primary_data.csv"!), and the meta information is in the file named "secondary_data_meta.csv". Before training the models, you first need to pre-process the data (e.g.., separate the values). Missing values are included in the data.

Requirements

(1) Implement two DT models by using *any two* split criteria from Information Gain, Gain Ratio, Gini Index and Variance. Note that you can use either binary-split or multiple-split.
(2) Use ~60% samples for training, ~20% for post-pruning, and ~20% for testing.
(3) Report the accuracy of the models.
(4) All DT models must be self-implemented, that is, you *cannot* use any machine learning library in this task.
(5) It is recommended that your implementation includes a "tree induction function", a "classification function" and a "post-pruning function".
(6) You can (but not must) use any suitable pre-processing method. You also can (but not must) use any reasonable early stopping criteria (pre-pruned parameters such as number of splits, minimum data set size, and split threshold) to improve the training speed. If you do so, explain your reasons.
(7) Present clear and accurate explanation of your implementation and results (in the Markdown format).

Deliverables

- A Jupiter Notebook source file named `<your_name>_task2.ipybn` which contains your implementation source code in Python
- A PDF document named `<your_name>_task2.pdf` which is generated from your Jupiter Notebook source file.
- Present clear and accurate explanation of your implementation methods and results (i.e., in the Markdown format).