Programming Assignment 2: Stream Mining

Advances in Data Mining

Introduction

This assignment focuses on implementing the different approaches outlined in the lecture stream mining.

Task Descriptions

1. Reservoir Sampling

In this task, you are required to implement the reservoir sampling algorithm as presented in the lecture slides. The goal is to return a sample of size k, which is representative of the whole data stream.

2. Bloom Filter

In this task, you are required to implement the bloom filter algorithm as presented in the lecture slides. The goal is to have a false positive rate as low as possible and a true positive rate of 1 while keeping the size for the *bit array* to a minimum. After implementing the Bloom filter, experiment with the appropriate number of hash functions by looping over 2...30 hash functions and seeing the false positive rate change, first down and finally up again. Does the optimal number of hash functions you find correspond to the formula in the slides? In your implementation, check that the produced code, when given the correct number of hash functions, produces an optimal false positive rate.

3. Probabilistic Counting

Your task is to complete the implementation of the Flajolet-Martin algorithm, based on a median R over n number of hash function. In the `main` section, you can experiment with a proper setting for n.

Submission

You have been provided with three python files, `task1.py, task2.py, task3.py` which contain the skeleton code for the tasks.

The code, along with the comments, should clearly instruct you on where to insert your own implementations. Please do **not** change any function names or return statements. Please

don't delete or modify any part of the skeleton code, but add your code in the sections where it is requested. Ensure that all your code is well-commented and follows proper Python standards. You are not allowed to import any external libraries other than those specified in the file. To ensure a correct submission, run the `validate_submission` test. This tests against all common submission errors and ensures that the submitted file can be evaluated. If the test fails, the evaluation will also fail, resulting in a **fail** for the assignment, so make sure the test passes when submitting your work.

Additional Notes

Make sure to test your functions with different test cases to ensure correctness. You can find a `main` section to test your implementations at the end of each file.

Submission

For submission, you are required to rename all the skeleton files to `taskx_<xxxxxxx>.py`, where `<xxxxxxx>` is your student number without the leading 's'. For example, for student number s1005819 and task 1, the submission file should be named `task1_1005819.py`.

You are required to hand in the following **three** files:

- `task1_<xxxxxxx>.py`
- `task2_<xxxxxxx>.py`
- `task3_<xxxxxxx>.py`

Good luck with the assignment!