Alex Tsun
CSE 312: Foundations of Computing II Due: August 17, 2020

Course Project

It is our happy job to announce that this quarter we are going to have the first ever CSE 312 Course Project! As it is the first time and we need to have fair and consistent grading, we have narrowed down the scope a little bit, but will give you freedom within that scope. We want to give you some space to have fun with the material, to be inspired and to be inspiring.

For this project, we will apply what we've learned about probability to (up to) three core topics in computer science: data structures, algorithms, and machine learning. In Problem Set 2, you implemented your first machine learning algorithm, the **Naive Bayes classifier** for spam filtering! There are many more machine learning models to study in CSE 446 and beyond! In Problem Set 3, you will all implement the **bloom filter** probabilistic data structure and the **distinct elements** randomized algorithm. In fact, the coding sections and associated questions of problem sets 2 and 3 will serve as good examples for what we expect out of you for the project. Just this time, we aren't supplying scaffolding with questions or starter code. For the project, you will go out into the wild and discover one more of a probabilistic data structure, a randomized algorithm, and/or machine learning algorithm!

Categories: There are three categories to choose from for the project. You will implement 1-2 different data structures/algorithms from these categories, depending on group size (see instructions below).

1. Probabilistic Data Structures (PDS)
2. Randomized Algorithms (RA)
3. Machine Learning Algorithm (ML)

We are very excited about this project, and hope you are as well! The course staff will be holding additional Project Office Hours regularly so that you have the support you need. Please come to these with all members of your group if possible, but you can also come individually! We are available to help **anytime** - so feel free to make an Edstem post for any questions that come up, and/or also to schedule a meeting with us if you cannot make it to any Project Office Hours. You can also come to regular office hours too.

Project Guidelines:

1. You will work in groups of 1-3 people, and you may work with previous pset partners. Here are the requirements depending on group size:

Group Size	Number of Topics	Instructions
1	1	Choose either PDS or RA (not ML)
2	1	Choose either PDS or RA (not ML)
3	2	Choose from any two different categories

   We want everyone to get more practice with PDS's and RA's, as there is a specialized course CSE 446 on ML which most people end up taking, but there are no (undergraduate) classes on probabilistic data structures and/or randomized algorithms. Hence, only groups of 3 will be allowed to implement an ML algorithm (as one of their two).

2. See below for some recommendations; you may choose one not on the list if you wish. But please be **extremely careful** when doing so to make sure it's not too advanced. We encourage you to find one you're interested in but we want to make sure it is doable!

3. Deadlines: There are NO late days for any of these three submissions. Details on each of the components are provided below.
   (a) Proposal is due Friday, July 17 at 11:00 PM PDT.
   (b) **(Optional)** Milestone is due Monday, August 3 at 11:00 PM PDT. Submit this if you'd like feedback on what you have so far, but this is not graded.
   (c) Final Report is due Monday, August 17 at 11:00 PM PDT.

4. LaTeX typesetting is required (template provided), and code must be written in Python.

5. The project comprises 30% of your final overall grade, with 3% coming from the proposal and 27% from the final report.

6. Groups can do one additional topic if they wish for **minimal** extra credit (1% out of 30% total). We want to encourage you to explore more only if you're interested and/or have time! Indicate on your final report which one is for extra credit so we know which to grade as your main project and which to grade as extra credit.

7. Your writeup should use notation consistent with that learned in class, and the goal is for your fellow CSE 312 classmates to read your report and be able to learn the topic. Much like the student directory, we will have a place those who opt in can share and view projects. It can be a great way to learn about the topics you didn't do your project on. Write clearly and concisely!

See the next pages for the requirements for each:

Probabilistic Data Structure

1. Describe in 1-2 paragraphs an application or problem which can be solved by the probabilistic data structure.
2. Define the core functionality of the probabilistic data structure, which must support at least two operations. Give the algorithm pseudocode for all of these operations in LaTeX (there is a nice package for this which we will provide).
3. Briefly explain how your probabilistic data structure works.
4. Randomization is usually applied to data structures to do at least one of the three things:
    (a) Reducing the time complexity of at least one of the operations in the data structure (in expectation).
    (b) Reducing the space complexity of at least one of the operations in the data structure (in expectation).
    (c) Reducing the "coding complexity" of the data structure (making it "easier to understand" and/or "easier to code up").

    Find an existing deterministic data structure for your task, provide pseudocode and a brief explanation, and explain in which way(s) the PDS has an advantage over the deterministic one.
5. Write a class which implements the PDS in Python, which can be applied to any problem. Write high quality code with good documentation and style.
6. Use your PDS to "solve" the problem you described in the first part. Provide driver code which sets up the problem and uses your PDS you implemented earlier. Include at least three visual results, across at least two of the following three categories:
    (a) A graph or plot (e.g., with runtime or space vs input size n). Could also plot this against the deterministic data structure's results if you choose to implement that as well.
    (b) A table.
    (c) A figure or diagram (e.g., stepping through some operations on a small sample input.)
7. Provide the "main" probabilistic analysis of the PDS (for at least one of the operations), or how probability is used to make your algorithm work (like in distinct elements). You may consult other sources in determining these, but you must write up your own version of the proof, using notation which is consistent with that from class. Write out all the steps with explanation so that a fellow CSE 312 classmate could understand.

    If you choose a project that is not on the list, it is possible that the probabilistic analysis of the operation of the PDS may be out of the scope of this class. If that is the case, please indicate this in the proposal and you may be approved to do a deterministic implementation instead.

Randomized Algorithm

1. Describe in 1-2 paragraphs an application or problem which can be solved by the randomized algorithm.
2. Define the randomized algorithm formally, specifying its inputs and outputs using proper mathematical notation. Give the algorithm pseudocode in LaTeX (there is a nice package for this which we will provide). Provide the (expected) runtime, (expected) space complexity, and whether or not your algorithm is guaranteed to produce the correct answer.
3. Briefly explain why your algorithm works, and determine whether it is a Monte Carlo or Las Vegas algorithm.
4. Randomization is usually applied to algorithms to do at least one of the three things:
   (a) Reducing the time complexity of the algorithm (in expectation).
   (b) Reducing the space complexity of the algorithm (in expectation).
   (c) Reducing the "coding complexity" of the algorithm (making it "easier to understand" and/or "easier to code up").

   Find an existing deterministic algorithm for your task, provide pseudocode and a brief explanation, and explain in which way(s) the randomized algorithm has an advantage over the deterministic algorithm.
5. Write a function which implements the randomized algorithm in Python, which can be applied to any input. Write high quality code with good documentation and style.
6. Use your algorithm to "solve" the problem you described in the first part. Provide driver code which sets up the problem and calls your general algorithm you implemented earlier. Include at least three visual results, across at least two of the following three categories:
   (a) A graph or plot (e.g., with runtime or space vs input size n). Could also plot this against the deterministic algorithm's results if you choose to implement that as well.
   (b) A table.
   (c) A figure or diagram (e.g., stepping through your algorithm on a small sample input.)
7. Provide the "main" probabilistic analysis of the algorithm (for a Monte Carlo algorithm, the probability of success, and for a Las Vegas algorithm, the expected runtime), or how probability is used to make your algorithm work (like in distinct elements). You may consult other sources in determining these, but you must write up your own version of the proof, using notation which is consistent with that from class. Write out all the steps with explanation so that a fellow CSE 312 classmate could understand.

   If you choose a project that is not on the list, it is possible that the probabilistic analysis of the algorithm may be out of the scope of this class. If that is the case, please indicate this in the proposal and you can be approved to do a deterministic implementation instead.

Machine Learning Algorithm

1. Describe in 1-2 paragraphs an application or problem which can be solved by the ML algorithm.
2. Choose a dataset and provide a link to it (see Kaggle or UCI ML Repository for examples). Describe your data, including the number of examples in the train/test set, and what features are provided. Explain which features are continuous (real numbers) and which are discrete/categorical (e.g., can only take finite values like color, blood type, breed of dog, etc.).
3. Define the ML algorithm formally, specifying its inputs and outputs using proper mathematical notation. Give the algorithm pseudocode for both training and testing in LaTeX (there is a nice package for this which we will provide). Recall your fit and predict function in Naive Bayes for example. Provide the (expected) runtime and space complexity in terms of the number of data points n and number of features per data point d.
4. Briefly explain the motivation or key idea behind the algorithm (in Naive Bayes, it was the use of Bayes Theorem and a "naive" conditional independence assumption).
5. Write a class which implements your ML algorithm in Python, which can be applied to any training data. Write high quality code with good documentation and style. You may not use ML packages such as sklearn, pytorch, and tensorflow, but you may use numpy.
6. Use your ML algorithm to "solve" the problem you described in the first part. Provide driver code which sets up the problem and uses your general ML algorithm you implemented earlier. Report some success metric (e.g., accuracy, F1-score, or AUROC) on the train and test set. Include at least **five** visual results, across at least three of the following four categories:
   (a) A visualization of the data.
   (b) A graph or plot (e.g., with training performance or test performance vs iteration).
   (c) A table.
   (d) A figure or diagram (e.g., stepping through your ML algorithm on a small sample input.)

   Be careful to encode your categorical features using a one-hot encoding (see this Kaggle article).
7. Describe the training and testing phase, and explain how probability was used to model the problem or as the basis for the algorithm. Carefully note the assumptions that were made about the data and the model, and why they were necessary. Write out all the steps with explanation so that a fellow CSE 312 classmate could understand.

Suggested Topics

Here are two topics from each category that we feel are most appropriate for this project. We highly recommend you choose a topic from below, as these are the ones we (the course staff) will be able to help you with most. **If you choose a topic not on the list, we cannot guarantee the same level of support**. We also need to make sure your project is feasible, so there's no guarantee we will approve your project if it's not on the list. With each topic, we list some resources that can help you get started. Some of these links will take you to landing pages, which means the page has links to multiple relevant notes or videos. Let us know if you're having trouble getting started, and we'd love to help! **The topics below are generally presented in relatively increasing difficulty.**

Suggested Probabilistic Data Structures

Name	Resources
Count-Min Sketch (CMS)	UW CSE 312, Stanford CS 168
Skip List	CMU CMSC 420, UW CSE 312, UCSD CSE 100

Suggested Randomized Algorithms

Name	Resources
Karger's Contraction Algorithm for Min-Cut	UW CSE 312, Stanford CS 265, UW CSE 521, Coursera
Locality Sensitive Hashing (LSH) via MinHash	Stanford CS 246, Brainly Article, CMU 15-451, Coursera, CMU 15-451

Suggested Machine Learning Algorithms

Name	Resources
Decision Trees (C4.5 Version with Entropy and Information Gain)	UW CSE 446
Logistic Regression (with Gradient Ascent)	Stanford CS 109

Suggested Resources for Other Ideas

1. UW CSE 490Z1/Stanford CS168: The Modern Algorithmic Toolbox
2. Stanford CS166: Data Structures
3. UW CSE 547/Stanford CS246: Machine Learning for Big Data/Mining Massive Datasets
4. Stanford CS261: Optimization and Algorithmic Paradigms
5. Stanford CS265/UW CSE 525: Randomized Algorithms
6. UW CSE 521: Design & Analysis of Algorithms

Here are the guidelines for the submissions you will make:

Proposal (3% of final grade)

1. Include all group member names on your writeup. Only one person should submit and tag their partners on Gradescope.
2. Do the following for **each** topic (one topic for groups of 1-2, and two topics for groups of 3):
   (a) Determine which topic you will be researching and what category it belongs to.
   (b) Do step 1: describing an application or problem which can be solved.
   (c) Briefly describe your algorithm, and include the pseudocode (can be a screenshot, but for the final report must be typeset with LaTeX).
   (d) Tell us what you will analyze for step (7) the analysis of your algorithm or PDS. You do not need to write out the analysis yet, but you should explain to us what you plan to analyze.
   (e) Provide at least 2 links to additional resources that you've found, at least one of which includes a description of the pseudocode. These must be **different** from the provided resources.
   (f) Feel free to ask any questions in this stage. The purpose of the proposal is to make sure you are on track and to check in with the course staff. Make sure to read the feedback they leave you!
   (g) If you plan on doing an extra credit topic, we will leave it up to you whether you want to include it in the proposal. You can submit it for our review at this stage, or just submit it with your final report.

Milestone (Optional, Not Graded)

1. Include all group member names on your submission. Only one person should submit and tag their partners on Gradescope.
2. The milestone is optional. You can submit your progress thus far and the course staff will look over it and provide feedback. We will not grade for correctness but will let you know how you are doing in terms of making progress and meeting the requirements of the project. You can include code in your submission if you'd like.

Final Report (27% of final grade)

1. Include all group member names on your submission. Only one person should submit and tag their partners on Gradescope.
2. Include all of the components outlined above for your topic(s), using the provided template.
3. You must include your Python code and final report in your submission. You must cite any sources you used in a bibliography section.