Project Guidelines

CAPP 30255 Advanced Machine Learning for Public Policy
University of Chicago

Project Type and Scope

The project is an important component of this course. It is an opportunity for you to work on an public policy project of your choice. Also keep in mind that course projects are one of the favorite topics during a job interview.

You should preferably work in groups of two or three for the project. In our experience, larger groups are sometimes not as productive in such short projects, in which case the team members end up with lower grades.

Your project should apply text processing to a task in public policy. It should also significantly advance your skills beyond what is covered in class.

Here are a few possible project ideas, some based on past projects in this course—

- Build a classifier to distinguish opinions from news, or fake news from real news.

- Compare bias (based on gender, race, etc.) in different sources of text, such as transcripts of user comments, online course videos, film scripts, social media posts, blogs, etc.

- Identify toxic comments in online forums.

- Analyze tweets from political leaders and their base to determine differences in sentiment with respect to voter suppression, tax reform, living wage, etc.

- Build a structured dataset from news reports on violent conflicts. In particular, automatically identify the main actors and the type of violence.

- Build a tool to scrape the web for information that allows you to rate the effectiveness of public service providers such as universities or hospitals.

- Predict which tweets or news stories will go viral.

- Analyze the text of legislative bills to predict the number of votes it will receive, or to determine the amount of "pork" in it.
- Analyze news articles or blogs to rank government organizations by level of corruption.
- Determine the level of correlation between politicians' statements and the campaign contributions to them.
- Determine the correlation between a politician's area of expertise—based on biographical accounts and past work experience—and the kinds of bills they author.
- Build a tool that in some manner helps citizens and organizations search or understand data or documents related to policy. For instance, the tool could regularly search a variety of government websites and determine if there are any new developments in, say, women's rights.

Many of the above are topics of active research, and you will likely find research articles containing sophisticated solutions for them. Given the available time, however, you cannot expect to develop a state-of-the-art solution during this course project. But use this opportunity to delve deep into a problem of your choice, apply what you know so far in machine learning, study the literature and learn new techniques relevant to your problem, develop a serious solution, evaluate its performance, and present your experience and results. Along the way look for some insights that future interviewers find interesting—"Horror films are the most gender neutral" or "Restaurant review scores are surprisingly uncorrelated with their level of hygiene."

Time commitment

The project counts for 50% of the grade, and you should plan to work accordingly. Plan to spend a minimum of 50 hours on your project during the quarter.

Most projects will require a combination of, and several iterations of, reading up the existing literature, thinking on the problem, learning how to use relevant software libraries, writing and executing code, and preparing a presentation and report. Depending on the project you will spend more time in one activity than the other. Each team member should, however, expect to write at least 1200 lines of code as their contribution.

Deliverables

There will be three deliverables: (i) Project Proposal (5% of total course grade), (ii) Mid-Quarter Presentation (15%), and (iii) Final Report and Presentation (30%). The proposal and final report should be posted on Piazza, and the two presentations will be made in class. You should offer constructive feedback to other teams on their proposals and presentations.

Project Proposal

Post your proposal by 11:59 p.m., Monday, April 20. It should be 3-4 pages (per team member) long and include the following:

- Title and team members.
- Brief description of what you want to do, including why it is useful, the data and software you will use, and the software you will write, if any.
- A **detailed description of the related work**. You should search for research papers and projects that solve the same or a similar problem. For the closest two or three such papers, you should describe what methods they used for obtaining the data, preprocessing the data, learning models, choosing metrics, and evaluating their results. You should also report their results, and what implications their work has on your project. (Learn as much as you can from such related work as it will give you an idea of what you need to teach yourself—beyond the material covered in class—for your project.)
- Brief plan of action, including any insights you have, the various steps of the project, the software libraries or packages you will be using, and the software you will be developing on your own. Also include what you plan to demonstrate for the mid-quarter and for the final presentation.
- Brief description of how you will evaluate your work. There may be existing techniques or results you could compare to, or you could test how well your solutions performs on test data, or how well it models the available data.
- Brief description of how the work will be divided among team members.

Mid-quarter Presentation

Each team should make a brief presentation in class on Thursday, May 14, describing the project objectives and work done so far, including any demonstrations of software. Each student should speak for 3 minutes, and keep 1 minute for questions. Depending on the number of students, some of the presentations may be held on, Monday, May 19. You may, optionally, pre-record a video of your presentation and play it during your slot.

Final Presentation and Report

Present your project and demonstrate any software in class on June 9 and 11. Each team member should speak for 12 minutes using slides and keep a couple of minutes for questions. Depending on the number of students we'll either have an extended class that week, or make similar arrangements.

Also post a final report on Piazza by 11:59 p.m., Thursday, June 11. The report should resemble a professional research paper and include—

- An abstract-like summary describing the work you have done.
- An introduction describing the problem and its significance.
- A detailed description of the related work.
- A detailed description of your solution and the work you have done.
- Include both what worked and what didn't, particularly if you have insights into the reason why.
- A detailed presentation of the results you have obtained and its analysis.
- Suggestions for future work along the same line.
- Description of your effort, including the relative effort on different activities. What did you have to learn for the project? What skills did you already possess? E.g., did you need to learn how to use particular libraries? Did you spend more time reading research papers, or fine tuning your parameters? If you were a team, what part of the work was done by which team member?
- Bibliography.

Please write your report in clear, concise English, and include clearly captioned, helpful figures. Unclear or sloppy reports will affect your grade.

Also submit a link to a github repository that contains all your source code. Along with the source code include a readme file containing—

- A description of the directory structure, and a brief description of the purpose of each file and the number of lines of code in it.
- A list of files you are be using that contain code not written by a team member, or not written as part of this project (maybe it was counted as a part of another project).
- Any other relevant information about your code.

Your code should be documented such that one can get a reasonable idea on how it works. E.g., each function should have a document string in the recommended style of the programming language.