Course Project Guidelines

Statistical Methods for Machine Learning II

1 Project Overview

The goal of this course is to learn how to derive, extend, and understand custom probabilistic models. Unlike a standard applied ML project where the focus is often solely on predictive performance, the focus of this project is on modeling rigor, inference correctness, and deep understanding of capabilities and limitations.

There is no better way to learn about probabilistic modelling than by applying it to a research question of your own. The purpose of this open-ended project format is to give you some experience working on a piece of original research and writing up your results in a paper style format.

You have a lot of freedom in choosing a topic for your final project. The only criterion is that it deeply involves the concepts (or extensions of them) that we are discussing in class. We expect you to choose your topic. We hope that this project will be useful to you, so you are encouraged to design a project that is related to your research interests (or the research interests of your teammate(s)). However, please be honorable and do not suggest a project that you have already fully completed as part of your research.

Choosing a Topic

Your first task is to select a project topic. We encourage you to pick an application area that excites you or a subfield of machine learning you wish to explore deeply. If you are looking for ideas, please come to office hours to brainstorm.

Most projects fall into one of three categories:

- **Application (Most Common):** Choose a dataset or a real-world problem (potentially from your own research) and address it using probabilistic modeling. The focus is on how best to derive and apply models to solve the specific problem.
- **Algorithmic:** Develop a novel variant of an existing model or a new inference algorithm (e.g., a custom MCMC sampler or Variational approximation) and study its performance empirically.
- **Theoretical:** Prove interesting properties of a new or existing learning algorithm. (Note: Purely theoretical projects are difficult and less common).

Projects combining elements of application, algorithms, and theory are also welcome.

Once you have identified a topic of interest, it can be useful to look up existing research on relevant topics by searching related keywords on an academic search engine such as `http://scholar.google.com`. Another important aspect of designing your project is to identify one or several datasets suitable for your topic of interest. If that data needs considerable pre-processing to suit your task, or if you intend to collect the needed data yourself, keep in mind that this is only one part of the expected project work, but can often take considerable time. We still expect a solid methodology and discussion of results, so pace your project accordingly.

Notes on a few specific types of projects:

- **Deep learning projects:** Since STA2104 discusses many other concepts besides deep learning, we ask that if you decide to work on a deep learning project, please make sure that you use other material you learned in the class as well. For example, you might set up logistic regression and SVM baselines, or do some data analysis using the unsupervised methods covered in class. We prioritize methodological and experimental rigor and standardize our grading, so note that pursuing (or not pursuing) a deep learning project will not itself impact your grade. Finally, training deep learning models can be very time consuming, so make sure you have the necessary computing resources.
- **Preprocessed datasets:** While we don't want you to have to spend much time collecting raw data, the process of inspecting and visualizing the data, trying out different types of preprocessing, and doing error analysis is often an important part of machine learning. Hence if you choose to use preprepared datasets (e.g. from Kaggle, the UCI machine learning repository, etc.) we encourage you to do some data exploration and analysis to get familiar with the problem.
- **Replicating results:** Replicating the results in a paper can be a good way to learn. However, we ask that instead of just replicating a paper, also try using the technique on another application, or do some analysis of how each component of the model contributes to final performance. In other words, your project does not need to be completely novel, but should not just duplicate previous work done by others.

Logistics

Students can work on projects individually, or in **groups of up to four**. The grade will depend on the ideas, how well you present them in the report, how clearly you position your work relative to existing literature, how illuminating your experiments are, and well-supported your conclusions are. Full marks will require a novel contribution.

Each group of students will write a short (2-4 pages) research project proposal, which ideally will be structured similarly to a standard paper. It should include a description of a minimum viable project, some nice-to-haves if time allows, and a short review of related work. You don't have to do what your project proposal says - the point of the proposal is mainly to have a plan and to make it easy for me to give you feedback.

At the end of the class you'll hand in a project report (around 4 to 8 pages), ideally in the format of a machine learning conference paper such as NIPS or ICML. All submissions must be in PDF format. You may include algorithm blocks, tables, and figures. The write-ups should be prepared in the NeurIPS paper format: https://neurips.cc/Conferences/2019/PaperInformation/StyleFiles. You may find online editors such as Overleaf helpful for writing the reports: https://www.overleaf.com/latex/tem 2019/tprktwxmqmgk

Total Weight: 35% of Final Grade

The project consists of two submissions:

1. **Project Proposal:** 10%
2. **Final Report:** 25%

The guidelines for submission of each parts are TBD.

2 Part 1: Project Proposal (10%)

Length: 2 to 4 pages (excluding references and appendices). Don't be afraid to keep the text short and to the point, and to include large illustrative figures.

Submission: Only one proposal needs to be handed in per group.
Goal: To ensure you have a feasible idea and a concrete plan to "fail fast."

A good proposal focuses on the specific probabilistic structure you intend to use and how you will verify it works. You are strongly encouraged to propose a *"Toy Data" experiment* as a sanity check where you generate synthetic data from your proposed model and verify that your inference algorithm can recover the true parameters. This allows you to quickly confirm the feasibility of your project.

The goal of this proposal is to:

1. Give you feedback on the feasibility of your probabilistic model and inference strategy.
2. Force you to design a "Fail Fast" work plan, so you can quickly assess the ultimate feasibility of your project.
3. Help you start writing the final report (much of this text can be reused).

Note: You are not strictly bound to this plan. Research changes. However, you must start with a concrete, scientifically sound plan.

Rubric and Structure

The proposal will be graded on the following five components (20% each).

2.1 Abstract (20%)

Summarize the main idea of the project.

- What is the problem?
- What is the proposed probabilistic approach?
- What is the specific contribution (e.g., "We will apply a Hidden Markov Model to...", "We will use a variational autoencoder for...")?

This should be understandable to anyone in the course. You don't need to say everything, just what the main idea is and one or two takeaways.

2.2 Introduction (20%)

State the problem being addressed and the motivation.

- Why is this problem interesting?

2.3 Work Plan (20%)

This is the most critical section for your success. Write a list of activities that will get you to the finished product, including time estimates.

The "Fail Fast" Requirement: You must order your work plan to check your assumptions and the basic feasibility of your idea before polishing it.

- **Recommended:** You should include a **"Toy Data Sanity Check"** in your plan.
- Plan to generate synthetic data from your model (where you know the true parameters) and verify that your inference algorithm can recover them.

Example Work Plan:

1. Extended literature search and writing related work section (3 hours)
2. Downloading baseline and reproducing previous results (4 hours)
3. Buid a synthetic dataset and check the method gives satisfactory results in this controlled environment (3 hours)
4. Finding appropriate datasets and converting them to same format (3 hours)
5. Running baselines on datasets and tweaking hyperparameters (4 hours)
6. Instrumenting model to track quantities of interest (2 hours)
7. Plotting quantities of interest (1 hour)
8. Making main figure (1 hour)
9. Writing project report (4 hours)

If the project doesn't go as planned, that's OK too, but you should try to allow as much flexibility as possible.

2.4 Description of Proposed Results (20%)

Write a list of proposed experiments, figures, results, tables, or summaries. For each one, say what the reader could learn from it.

- **Sanity Checks:** "Figure 1 will show the true parameters vs. learned parameters on toy data."
- **Baselines:** "Table 1 will compare test log-likelihood against a standard GLM."
- **Qualitative Analysis:** "Figure 2 will visualize the learned latent space."
- **Diagnostics:** "We will report Effective Sample Size (ESS) or R-hat for our sampler."

2.5 Related Work (20%)

Distinguish your work from previous approaches.

- Include a 1–2 sentence summary of at least 3 closely related papers. I realize you might not know about all related papers (or have time to carefully read all related papers), and that's OK.
- Clearly state what previous works did, how your model differs (e.g., "Unlike Paper X, we use Y to handle Z...") and what is the novelty in your approach.
- *Tip: Use BibTeX (available from Google Scholar) now to save time later.*

3 Part 2: Final Report (25%)

Length: 4–8 pages (excluding appendices and bibliography, NeurIPS/ICML format recommended). Don't be afraid to keep the text short and to the point, and to include large illustrative figures. You can include as many proofs, extra details, experiments, etc. as you want in the appendices.
Goal: To demonstrate your ability to derive a probabilistic model, implement inference, and critically analyze the results. While we encourage creativity, the primary focus is on technical rigor and deep understanding of the methods used.

The idea is that this project report should be a manageable amount of work, but that if you want to turn your project into a paper, everything in the project report will need to be done anyways.

Code Submission: You must submit your code, details will be shared later.

3.1 Abstract (5 points)

Summarize the main idea, the probabilistic framework used, and the key results.

- Should be understandable to anyone in the course.
- Clearly state the contribution (e.g., "We derive a Variational Autoencoder for...").
- You don't need to say everything you did, just what the main idea was and one or two takeaways.

3.2 Introduction (5 points)

State the problem being addressed and the motivation.

- Why is this problem interesting?

3.3 Model Diagram or Figure (10 points)

"A good figure is worth 1000 words."

You want to show the overall model or idea. The idea is to make your paper more accessible, especially to readers who are starting by skimming your paper.

- For instance, you can include a **Probabilistic Graphical Model (PGM)** diagram (e.g., in plate notation).
- Try to be clear as to how to interpret nodes, edges, arrows...
- Differentiate between observed variables (x), latent variables (z), and parameters (θ).
- For the project, taking a picture of a hand-drawn diagram is fine, as long as it's legible. Otherwise, we recommend using Tikz, a LaTeX package.
- *Note: You must create a new figure, not just copy one from a paper.*

3.4 Formal description (20 points)

Explicitly describe the generative model / inference strategy / loss function / conjecture / problem domain. Include at least one of:

- An algorithm box.

- Equations describing your model.
- A theorem or formally stated conjecture.
- A formal description of a problem domain.

Highlight how your model is different from other approaches, or what the main relevant considerations are for the domain. This can be done by comparing it to an existing model, perhaps by using another diagram or in words. E.g. if you are proposing a new algorithm that only changes one line in an existing algorithm, highlight that one line, or do a side-by-side comparison.

Note: If the derivation is standard, put the bulk in the Appendix, but summarize the key steps in the main text.

3.5 Related Work (10 points)

As in the project proposal, distinguish your work from previous approaches. If your project builds on previous work, clearly distinguish what they did from what your new contribution is.

- Include a 1-2 sentence summary of other closely related papers (you might not know about all related papers or have time to carefully read them all, and that's OK for this project).
- Clearly explain how your model differs from baselines (e.g., "Standard VAE assumes Gaussian prior, we use...").

3.6 Comparison or demonstration (16 points).

Include at least one of:

- A demonstration of a theorem or conjecture. For example, a counter-example.
- A comparison of data generated by your model to a baseline model. Qualititative evaluation is OK.
- An experiment demonstrating a property that your model has that a baseline model does not. Experiments should also include a description of how you prepared your datasets, how you trained your model, and any tricks you used to get it to work.
- If doing a review, include a table comparing the properties of the different approaches.

Toy data is OK (and always recommended in a first step, can your model recover true parameters from synthetic data?)! The point is to help the reader understand why or when we would want to use one approach over another, or to understand something better. Try to summarize the main takeaways.

Note: Include details on dataset preparation and training hyperparameters.

3.7 Technical Depth & Difficulty (20 points)

(Replaces "Novelty" from the standard rubric)

- **High (15-20 pts):** Successfully derived / implemented a non-trivial custom model or inference algorithm (e.g., custom MCMC sampler, non-standard variational family). You should motivate your method, not just implement a random tweak for no reason. For full marks, you need to check if the reason you said it should work better is actually the reason why it worked better. Alternatively, do a theoretical analysis with a new result.

- **Medium (5-15 pts):** Applied off-the-shelf probabilistic methods to a complex, novel dataset with thoughtful model selection. You should study a novel combination of method and type of data, you would get novelty points for making tweaks to a method to take advantage of the structure of a new dataset or domain. Alternatively, motivate and formally pose a new problem, such as characterizing the difficulty of some task
- **Low (0-5 pts):** Either applied standard black-box methods (e.g., 'sklearn.LogisticRegression') without probabilistic interpretation or derivation, or produced a literature review with no new content. This is completely OK as a project, but it won't earn an A+.

3.8 Limitations (10 points)

Honest discussion of where the model fails. Describe some settings in which we'd expect your approach to perform poorly, or where all existing models fail. Try to guess or explain why these limitations are the way they are.

- When does the inference fail to converge?
- What assumptions are likely violated?
- Suggest concrete future improvements, or give some examples of possible extensions, ways to address these limitations, or open problems.

3.9 Conclusions (4 points)

State the results achieved in relation to the problem described in the introduction. Repeat the main takeaways from your paper.

Submission Checklist

PDF Report (4–8 pages).

Appendix (optional, for long proofs).

Code.

Note on "Novelty" vs. "Depth"

Unlike a research conference, this course does not strictly require *novelty* (inventing a new algorithm). Instead, we value **Technical Depth**.

- **Excellent:** Rigorously deriving and implementing a known but complex method (e.g., implementing HMC or a VAE) or extending a standard model with a custom prior/likelihood.
- **Poor:** Applying an off-the-shelf 'sklearn' method or a pre-trained net without understanding the underlying probabilistic mechanics.

Additional advice

- **Be honest!** You are not being marked on how good the results are. It doesn't matter if your method is better or worse than the ones you compare to. What matters is that you clearly describe the problem, your method, what you did, and what the results were. Just be scientific.
- **Be careful!** Don't do things like test on your training data, set hyperparameters using test accuracy, compare unfairly against other methods, include plots with unlabeled axes, use undefined symbols in equations, etc. Do sensible crosschecks like running your algorithms several times to understand the between-run variability, performing gradient checking, etc.