CSC311 Introduction to Machine Learning
Project Instruction (Fall 2025)*

Alice Gao

Last Updated: September 2, 2025

Contents

1 Introduction

Welcome to the CSC311 Machine Learning Project. You will complete this project in **teams of 3–4 students**.

You will receive a training dataset of student responses, and your task is to build a classifier that predicts the category of each response out of three possible categories. Your classifier will be evaluated on a separate test set, compiled from responses by TAs and instructors. This test set will not be shared with you, so your model should generalize well to unseen data.

You are encouraged to use any materials from this course, including individual models or ensembles. Ensure that your submission, including the prediction script, stays within the file size limit. Pay attention to avoid underfitting or overfitting and aim for strong, reliable performance on the test set.

Finally, there will be a prize for the team(s) that achieve the best results on the unseen test set. Details will be announced closer to the deadline.

*These instructions are adapted from the work of Professor Lisa Zhang at the University of Toronto, Mississauga.

2 Components, Grading Scheme, and Due Dates

The Project is worth 15% of your final course grade. It has the following components.

Component	Weight	Due Date
Data Collection	1%	Wed, Sep 17, 2025
Team Formation	1%	Wed, Oct 01, 2025
Project Proposal	2%	Wed, Oct 15, 2025
Prediction Script	3%	Fri, Nov 21, 2025
Final Report	8%	Fri, Nov 21, 2025

Table 1: Project Grading Scheme

2.1 Data Collection

Complete the Quercus quiz titled **Project: Data Collection**. This should take 5-10 minutes to complete.

2.2 Team Formation

In the `project_team` assignment on MarkUs, create your group of 3-4 students. Every member must **accept the group invitation** on MarkUs by the deadline to earn the 1% grade. Your team members can be from any section of the course.

Moreover, complete and submit the **group contract** on MarkUs. A group contract template has been posted on Quercus.

2.3 Project Proposal

Submit a PDF file named `proposal.pdf` in the `project_proposal` assignment on MarkUs. The project proposal will be graded based on Table 2.

Component	Weight
Data Exploration	1%
Planned Methodology	1%

Table 2: Proposal Grading Scheme

Data Exploration

Provide a complete draft of the Data Exploration section. This draft will earn up to 1% of your course grade. In the final report, you will have the chance to revise and expand this section for up to 2% of your course grade.

Planned Methodology

Submit a clear draft plan for the Methodology section. This plan should include:

- The three model families you plan to explore, and why they are appropriate for the dataset.
- The optimization technique(s) you expect to use.
- The validation method you will adopt.
- A list of hyperparameters you plan to tune, including the ranges of values and brief justifications.
- The evaluation metrics you will report. At least one metric should go beyond accuracy, with justification.

Your proposal is graded for completeness and clarity, not for final results. It is an opportunity to receive feedback early and refine your plan for the final report.

2.4 Model Prediction

Submit a Python3 script named `pred.py` to the `project_prediction` assignment on MarkUs.

The script must include a function `predict_all` that takes the name of a CSV file as a parameter and returns predictions for the data in the CSV file.

Requirements

Your submission must meet the following requirements.

- Use Python 3.10 or later.
- `pred.py` may import only: standard Python libraries, `numpy`, and `pandas`. It **cannot** import `sklearn`, PyTorch, or TensorFlow.
- You may submit additional files required by `pred.py`. All submitted files combined must **not** exceed 10MB.
- The script must **not** require networking or download files.
- The script must **run efficiently**: able to produce about 60 predictions within 1 minute using reasonable memory.

How to Make Predictions Without ML Libraries

A common source of confusion is how to make predictions without importing machine learning libraries. You are free to build and train your model using any tools (e.g., `sklearn`, PyTorch, TensorFlow) and any code you have written or reused from labs. However, your final `pred.py` must generate predictions on the test data without relying on these libraries. One possible approach is to "save" your trained model parameters in a separate file and have `pred.py` load these parameters at runtime to produce predictions.

Model Prediction Grading Criteria

The grading rubric for the prediction script is described in Table 3. The basic criteria include: runnable script, file size at most 10MB, only use allowed imports and uses reasonable memory. For the results over the test set, we will set the **reasonable** threshold such that groups who follow good machine learning practices and choose reasonable models should be able to pass the threshold.

Grade	Quality of the results
3/3	Meets basic criteria. Results on the test set are **reasonable**.
2/3	Meets basic criteria. Results on the test set are **reasonable**, but the model is clearly **overfit** to the training data.
1/3	Meets basic criteria. Results on the test set are **better than random**.
0/3	**Does not** meet basic criteria, or results on the test set are **not better than random**.

Table 3: Model Prediction Grading Criteria

2.5 Report

Submit two files to the `project_report` assignment on MarkUs.

- A PDF file named `report.pdf`.

  This file describes your final model and outlines the steps to develop this model. We have provided a report template in LaTeX. Please comment out the instructions and add your content.

- A ZIP file named `code.zip`.

  This file should contain all the `.py` and/or `.ipynb` files used to develop your final model. Please exclude the `/data` folder from the starter code.

  The `code.zip` file will not be graded but serve as evidence of your work. The files do not need to be runnable by the TA and can rely on external imports (e.g. sklearn) and datasets you don't submit.

The detailed grading rubric for the report is in Table 4.

Section	Marks	Descriptions
Data Exploration (2 marks)	0.5	**Summarize the dataset:** Clearly describes feature types (numerical, categorical, text), distributions, and class balance.
	0.5	**Identify and address data issues:** Notes missing values, outliers, or inconsistencies, and explains how they are handled.
	0.5	**Explain preprocessing:** Describes transformations (e.g., normalization, encoding, text representation) with justification.
	0.5	**Prevent data leakage:** Explains how the test set was reserved and not used during exploration.
Methodology (4 marks)	0.5	**Model families:** States the three model families used and why they fit the dataset.
	0.5	**Optimization:** Describes optimizer (e.g., SGD, Adam) and LR schedule and/or regularization/early stopping.
	0.5	**Validation method:** Explains train/validation split or cross-validation.
	0.5	**Hyperparameter list:** Comprehensive list for each model.
	0.5	**Hyperparameter choices:** Ranges/search strategy with evidence they're reasonable (validation results, rationale).
	0.5	**Avoid winner-only tuning:** Tuned all three models before selecting the best.
	1.0	**Evaluation metrics:** Describes which metrics were used. Includes at least one metric beyond accuracy and justifies why the chosen metrics are appropriate.
Results (2 marks)	1.0	**Report results clearly:** Uses evaluation metrics from Methodology; clear tables/plots; compares model families; states final choice.
	0.5	**Analyze errors:** Identifies common misclassifications/weaknesses; includes confusion matrix or examples if helpful.
	0.5	**Test performance:** Single-number estimate for best model with justification (e.g., CV stability, learning curves).
Total	**8.0**	

Table 4: Detailed Report Grading Rubric

3 Advice for Forming Project Groups

Your group project is a major part of the course. Choosing the right teammates can make the difference between a smooth experience and a stressful one. Here are some tips to improve your chance of success:

Balance Skills and Strengths: Form a group with **complementary skills**: coding, analysis, writing, organization, or presentation. Avoid groups where everyone has the same strengths—

diversity helps.

Align Work Habits and Availability: Discuss schedules and preferred work styles before committing. Choose teammates with similar availability and willingness to meet (in-person or online).

Prioritize Commitment and Reliability: **Reliability** is often more important than technical ability. A dependable teammate who meets deadlines will reduce stress for everyone.

Clarify Expectations Early: Talk about how often you will meet, how you will communicate, and what quality of work you expect. Clear expectations now prevent conflict later.

Look Beyond Close Friends: Friends can make great teammates, but sometimes it is harder to hold them accountable. Be open to working with classmates you do not know well but who show responsibility in class.

Think Ahead to Milestones: The project has several deadlines. Pick teammates who are **organized and proactive** so you will not fall behind.

Be Inclusive: Not everyone finds a group right away. Inviting others to join can bring in new strengths and perspectives.

Final Tip: A successful group is not just about skill—it is about communication, reliability, and respect. Choose teammates who will help you enjoy the process as much as the outcome.

4 Recommendations

Start Early: Begin data exploration as soon as the data is available.

Communicate Regularly: Frequent communication is key for task coordination and decision-making.

Take Notes: Use a shared document (e.g., Google Docs or Overleaf) to record your experiments. Clear, reproducible records will simplify writing the Methodology section of your report.

Experiment with Models: Test various models using sklearn before implementing your own. Utilize code from your labs.

Plan for Training Time: Be aware that some machine learning models require significant time to train.