Assignment
FEM21045

September, 2025

Welcome to the assignment of the Machine Learning in Econometrics course! This assignment should be made in **groups of four** students, so please team up with your fellow students. The setup of this assignment is quite non-standard, so please read this document carefully.

In this assignment, you are asked to build a distributional predictive model. That is, instead of modeling $\mathbb{E}[y|x]$, we ask you to model the whole distribution $p(y|x)$. The dataset is derived from the General Society Survey (`https://gss.norc.org/`) and focuses on the development of real income from 1874 to 2018. Your task is to construct probabilistic predictions of the real income conditioned on information about individual's education, occupation and other characteristics.

1 Overarching Task

Your overarching task is to train a machine learning algorithm that estimates the *distribution* of the real income conditional on other observed information. You will train the algorithm on a preprocessed subset of the *General Society Survey* dataset. Each instance of this dataset consists of the target `realrinc` and the following features

- realrinc – individual's base income in 1986 US dollars,
- year – survey year,
- age – age in years,
- occrecode – occupational field,
- prestg10 – occupational prestige score,
- childs – number of children,
- wrkstat – work status,
- gender – self-explanatory,

- educcat – obtained degree level,
- maritalcat – marital status.

The preprocessing was performed in a way such that you can immediately apply your machine learning algorithm of choice without (much) further data manipulation. However, you are free to do additional processing steps such as advanced encodings or feature engineering. Note in particular that none of the features have been standardized.

In designing your predictive algorithm, please adhere to the following requirements:

1. You are free to choose your algorithm, as long as the other requirements below are fulfilled (in particular the part on the likelihood function). Of course, you are very welcome to try a variety of different algorithms, including perhaps algorithms that have not been introduced in this course to see how they compare. But for your final report and predictive performance, please only report on *one* of the algorithms.
2. We evaluate the final predictions on a separate test set based on the **Continuous Ranked Probability Score** (CRPS).
3. If your algorithm relies on a log likelihood objective, we ask you to implement it on your own. That is, do not rely on the likelihood functions provided by packages such as `torch.distributions`.
4. When you are training and tuning your algorithms, we ask you to optimize hyperparameters and evaluate the accuracy of the final prediction algorithm. This requires you to come up with a training, validation, and test strategy. It is up to you to decide what strategy you choose. Note that the strategy does not have to be overly complex to yield a good grade. Which hyperparameters you optimize is also up to you and depends on your chosen algorithm.

Hint: Since this assignment is similar to a Kaggle competition, you do not have to completely reinvent the wheel. We highly recommend you to check out code available online and get some inspiration for how to train and tune your model. Do keep in mind the differences between the competition and code you may find online and do not blindly copy nor plagiarise.

2 Graded Subtasks (20 pt)

You are graded on the basis of a few subtasks: the performance of your algorithm (coined *prediction challenge*), your design choices, and your ability to report on these choices concisely. As part of the report, we ask you to do some specific analyses. These subtasks are elaborated on below.

2.1 Prediction Challenge

The utility of your predictive algorithm is for a large part determined by the accuracy you achieve. We therefore grade your predictions relative to the performance of a benchmark model trained by the TA. In other words: You will receive **full points on the prediction challenge if you beat the benchmark**! While this may seem like a daunting task, in fact, this should be very doable as we won't spend nearly the same amount of time optimizing our algorithms. We will only reveal the TA's model after the final deadline, so it is up to you to decide when your CRPS is good enough. You may want to consider comparing your score with other teams to see how you are doing!

You are provided with three datasets (see assignment page on canvas): (1) `X_trn.csv` contains the features of the training set, (2) `y_trn.csv` contains the corresponding targets and (3) `X_test.csv` contains the features of the test set. Use the first two data sources to train and tune your algorithm. Afterwards, derive predictions for the observations in `X_test.csv`.

Since we will grade your algorithm on its predictive accuracy against predictions made by us, we call this first part of the assignment a *prediction challenge*.

In this prediction challenge, you are asked to predict the target `realrinc` for the instances in the test data `X_test.csv` and provide them to us in a file `predictions.npy` (.npy is the file format used by the numpy Python package) so we can compute the CRPS based on the test set targets. However, we are interested in the whole distribution and CRPS evaluates the distributional fit. Hence, instead of single predictions for each observations in `X_test.csv`, we ask you to provide 1000 predictions, drawn from $p(y|\mathbf{x}^{(i)})$, where $\mathbf{x}^{(i)}$ are the features of observation i you derived from `X_test.csv`. **Important:** The test set targets are not scaled in our side when computing the CRPS. This means that if you choose to scale the data in `y_trn.csv` for training, you need to reverse the scaling in the final predictions before submission!

Below, we give you example code that shows you how you can draw 10 (not 100!) samples from $p(y|\mathbf{x}^{(i)})$, where we assume a Normal distribution with parameters μ and σ.

```
import torch
import numpy as np

# assume we have passed X through the model and obtained mu and sigma
mu = torch.randn(1000)
sigma = torch.randn(1000).exp()

# draw 10 MC samples for each mu, sigma pair
# in a for loop:
y_pred = []
for i in range(mu.shape[0]):
    samples = mu[i] + sigma[i] * torch.randn(10) # 10 samples per observation
    y_pred.append(samples)
y_pred = torch.row_stack(y_pred)  # shape (n_obs, n_samples)

# alternatively: vectorized sampling
# ensure mu and sigma have shape (n_obs, 1)
y_pred = mu.unsqueeze(1) + sigma.unsqueeze(1) * torch.randn(mu.shape[0], 10)

# convert the predictions to numpy array and save them for submission
np.save('predictions.npy', y_pred.numpy())

# we will evaluate the CRPS using the following code
from scoringrules import crps_ensemble
crps_ensemble(y_test, y_pred).mean().item()
```

Your predictions will be graded via pre-constructed Python script after submission, so it is important that you strictly adhere to the above instructions. For example, do not change the file name `predictions.npy` in any form or put any other information into the file. Note also that the dimension of `y_pred` has to be 5578×1000 (number of observations $\times$ number of MC samples), where the first number are the observations in `X_test.csv`.

Some warnings: please triple check that your prediction file is error-free and has the correct dimensions! The score we compute is final, so it is **your responsibility to make sure your file is created correctly and the predictions have the correct scale (i.e., the scale of the data in `y_trn.csv`)**.

Please note that **you are not allowed to use the test data in `X_test.csv` in any way or form during the training of your algorithm**; any attempt of doing so will be considered fraud! We therefore recommend that you *yourselves* partition the data in `X_trn.csv` and `y_trn.csv` into an estimation set (which you use for training) and a validation set (which you use for assessing out-of-sample accuracy). The only time you use `X_test.csv` should be when you make the finaal predictions you intend to submit for the prediction challenge. Moreover, directly copy-pasting code from the discussion forums or from elsewhere is also considered fraud. Using packages is, of course, allowed but remember the rules in Section 1.

You will be asked submit the complete code of your algorithm. We will not grade your code in any form. It merely serves as a check for us that you have not engaged in plagiarism or the forms of fraud described above. After the deadline, we may ask you to replicate your submitted predictions in our presence by using your submitted code.

2.2 Report (15 pt)

Your second subtask is reporting on your design choices/methodology and your main findings. Please refer to the overarching task description and the rubric to determine what information you should include. In general, your report has the following purposes:

First, motivate your choice of algorithm and clearly describe how you have trained it, including any likelihood functions you may have used. Your description of the training process should be concise enough such that it is fully replicable[1] for peers following this course. For example: "we used a grid-search" is not replicable because you did not share which candidate values of the hyperparameters you used exactly. Please note that you are not asked to describe how the algorithm itself works, so there is no need to describe how, say, a neural network works.

Second, you should show a clear motivation for the choices you made. This can either be based on a good rationale or cited relevant scientific sources.

Third, you should arrive to clear conclusions. For example, "our predictive algorithm achieves an out-of-sample CRPS of this value." We again emphasize that you are not allowed to use the data in `X_test.csv` in your training of the algorithm.

Fourth, to derive additional substantive insights, train another model to estimate $p(y|\mathbf{z})$, where $\mathbf{z}$ = (year, gender). Provide results (figures and short answers in the main text are enough) for the following two questions:

1. Which group has a higher real income inequality in 1980, males or females? Briefly explain your reasoning and provide a figure.

2. Using 1000 Monte Carlo samples for each gender, what is the probability that a male earned more than a female in 1980? How does it compare to 2010? Provide a figure for each year and briefly explain your reasoning. The figure should show the histogram of the Monte Carlo samples (consider the correct scale).

Write your report in the provided LaTeX template, called `report.tex`. You are free to add packages of your choosing to create your report, but do not change the font, margins, or any other aspect of the template that determines how much text can be in your document. Your report—returned as the file `report.pdf` by the template—may not exceed one page. A list of references and an appendix may exceed the one page limit. In the appendix, you may **only** list the optimized hyperparameters, grid values and figures but cannot add additional discussions. Moreover, please enter your names and group numbers at the top of the template page `report.tex`. You may ctrl-f or cmd-f "#HERE" in the template to find where you should enter these details.

We will be using the following rubric for grading:

- Fail: Major misunderstandings; incorrect or irrelevant explanation.
- Satisfactory: Basic idea conveyed, but with gaps or vague statements.
- Good: Sound explanation with minor omissions; terminology mostly correct.
- Excellent: Clear, accurate, and insightful explanation; precise use of terminology.

[1] Up to stochastics and programming details, meaning that a stochastic algorithm will not give the same result twice. In the report you do not have to share in detail how you programmed your code.

3 Group Registration

The assignment is supposed to be done in teams of four. You are supposed to join your group in Canvas. The deadline for joining a group on Canvas is **September 18, 2025, 18:00**, which we call the registration deadline.

You are only guaranteed to work with group members of your choice if your group consists of four members. If there are groups of less than four members (including groups consisting of one single member) after the registration deadline, we may break up or rearrange these groups into groups of four. We will notify you of your final group in case you are affected by this rearrangement.

4 Assignment Submission

The assignment submission consists of delivering the following three files; please **keep these file names in your submission**.

1. `predictions.npy` as described in Section 2.1.
2. `report.pdf` is the report described in Section 2.2. This file must be the output of the template `report.tex`.
3. Replication code of the method that generated the predictions for the prediction challenge (submitted in `predictions.npy`). This replication code may consist of several individual files or folders. This code will not be graded, but make sure that it replicates the predictions when run in the intended way.

You will submit these three deliverables via Canvas. There are two assignment pages on Canvas: one for the graded deliverables (`predictions.npy`, `report.pdf`) and one for the ungraded replication code. After you joined a group (see Section 3), it is sufficient if one group member submits all three deliverables.

The deadline for providing these three deliverables is **October 12, 2025, at 11:59 PM**.

If you have questions about the assignment, its intended workflow, or the submission format, please post them in the corresponding discussion forum on Canvas.

Finally, there is a checklist for this document's main takeaways in Appendix A.

Good luck with the assignment!

A Checklist

- ☐ I have read this document thoroughly and registered as part of a group in Canvas.
- ☐ By the assignment deadline (October 12, 11:59PM), I have submitted the following files on Canvas:
  - ☐ `predictions.npy` [in the correct format from Section 2.1]
  - ☐ `report.pdf`
  - ☐ Replication code for the predictions in `predictions.npy`
- ☐ I have double-checked that I can perfectly replicate the predictions in `predictions.npy` with the replication code we intend to submit.
- ☐ I have triple-checked that the predictions in `predictions.npy` have the correct format, as described in Section 2.1.