Week 11 Tutorial: Neural Networks II

POP77032 Quantitative Text Analysis for Social Scientists

Tom Paskhalis

Exercise 1: Logistic Regression in PyTorch

- Let’s start by re-visiting the example from last week.
- Now, instead of implementing the logistic regression with SGD from scratch, we can use PyTorch similar to how we implemented it for linear regression in lecture.
- First, re-visit the solution for logistic regression with SGD from last week and modify it to use PyTorch tensors instead of NumPy arrays.
- Next, build a logistic regression model specifying its architecture in PyTorch.
- Apply this model to the simulated dataset.

- We start by creating a simulated dataset so that we know the true parameters of the data generating process and can evaluate how well our model recovers them:

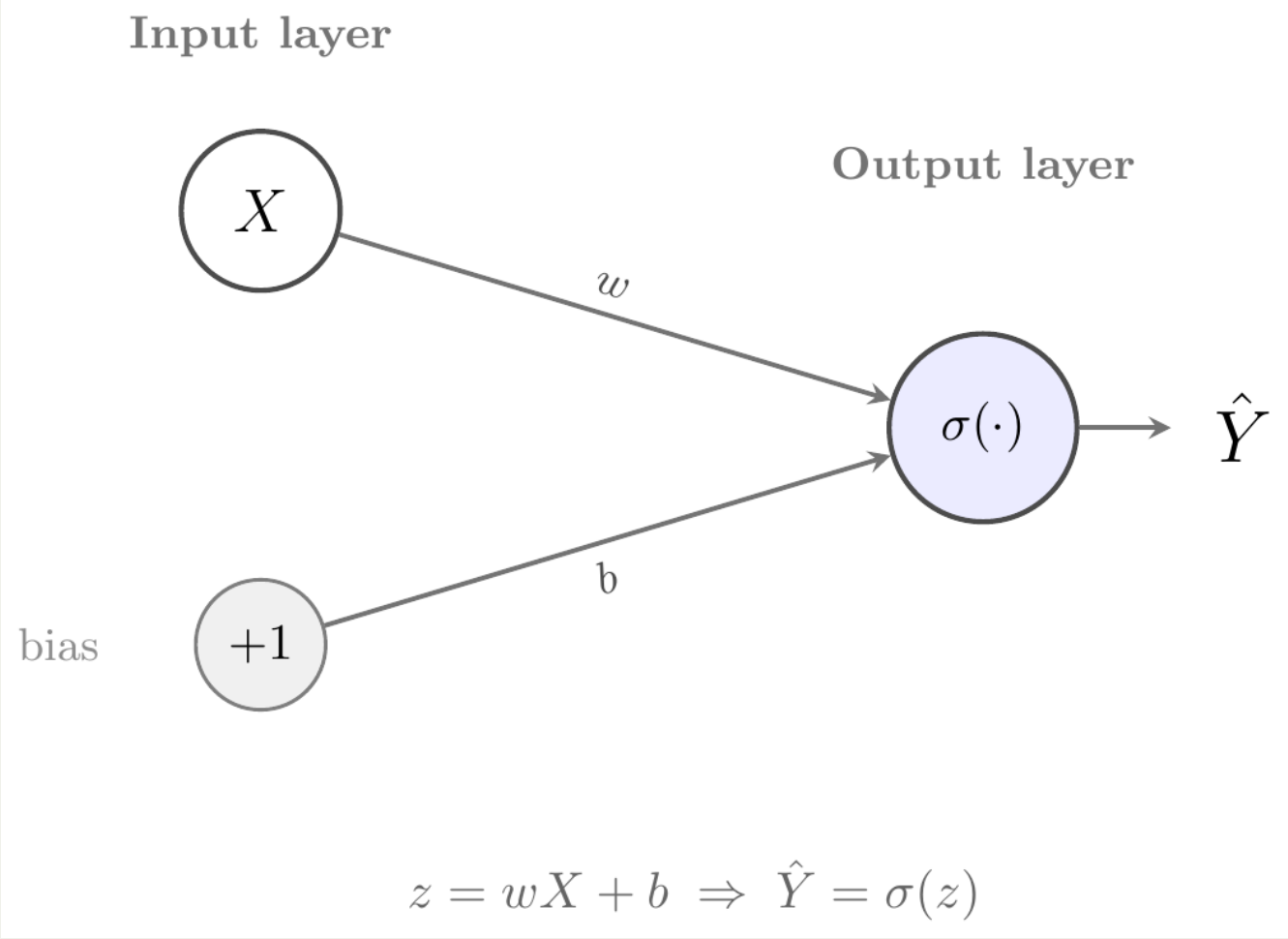
```
import torch

rng = torch.Generator().manual_seed(123)
X = torch.randn(1000, 1, generator=rng)
bias = torch.tensor([-1.5])
weights = torch.tensor([0.8])
pi = 1/(1 + torch.exp(-(X @ weights + bias))) # pi = sigmoid(X @ weights + bias)

y = torch.bernoulli(pi).squeeze() # binary labels sampled from Bernoulli distribution with probabilities give
```

Neural Network Architecture

- If we were to draw a digram of a neural network architecture for logistic regression, it would look something like this:



Exercise 2: Text Classification

- Building upon the logistic regression in PyTorch and our previous work on supervised learning, let's build a simple feedforward neural network for text classification.
- We'll use the Dáil speeches dataset for this exercise and try to predict whether a given speech was made by a member of the government or the opposition based on its text.
- First, load the dataset and create a binary target variable indicating whether the speaker is a member of the government or the opposition
  (Fianna Fáil, Fine Gael, and the Green Party were in government).
- Next, preprocess the text data by converting it into a numerical format suitable for input into a neural network (e.g., using TF-IDF vectorization).
- Then, set up a simple feedforward neural network architecture in PyTorch, e.g., with one hidden layer and a sigmoid activation function for the output layer.
- Train the model on the training data and evaluate its performance on a held-out test set using usual classification metrics (e.g., accuracy, precision, recall).