TODAY

Mock exam question 1

Please collect a printout from the front bench on your left.
1. A 0/1 signal is being transmitted. The transmitted signal at timeslot $i \in \{1, \ldots, n\}$ is $x_i \in \{0, 1\}$, and we have been told that this signal starts at 0 and then flips to 1, i.e. there is a parameter $\theta \in \{1, \ldots, n - 1\}$ such that $x_i = 1_{i > \theta}$. The value of this parameter is unknown. The channel is noisy, and the received signal in timeslot $i$ is

$$Y_i \sim x_i + \text{Normal}(0, \sigma^2)$$

where $\sigma$ is known.

2. Given received signals $(y_1, \ldots, y_n)$, find an expression for the log likelihood, $\log \Pr(y_1, \ldots, y_n; \theta)$. Explain your working. [5 marks]

3. Give pseudocode for finding the maximum likelihood estimator $\hat{\theta}$. [5 marks]
I have been monitoring average annual river levels for many years, and I have collected a dataset \((z_0, \ldots, z_n)\) where \(z_i\) is the level in year \(i\) since I started monitoring. I believe that for the first few years the level each year was roughly what it was the previous year, plus or minus some random variation; but that some year a drought started, and since then the level has decreased on average each year. I would like to estimate when the drought started. I do not know the other parameters.

(i) Propose a probability model for my dataset. [5 marks]

(ii) Explain how to fit your model. [5 marks]

This “propose a probability model” is open-ended and scary. How should we even begin to think about it?

Part (a) gave us a hammer. Can we see part (b) as a nail?
Deep learning*
Supervised Learning

Data: \( \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\} \)

Labels: \( y_1, y_2, ..., y_n \)

Task: fit the probability model

\[ \text{Pr}_Y(y; f_\theta(x)) \]

Training goal: MLE
Example (regression)

Given a labelled dataset consisting of pairs \((x_i, y_i)\) of real numbers, fit the model \(Y_i \sim \alpha + \beta x_i + \gamma x_i^2 + N(0, \sigma^2)\)

Model for a single observation:

\[
Y \sim \alpha + \beta x + \gamma x^2 + N(0, \sigma^2)
\]

\[
\sim N(\alpha + \beta x + \gamma x^2, \sigma^2)
\]

Likelihood of a single observation:

\[
Pr_{Y}(y; x, \alpha, \beta, \gamma, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(y-(\alpha+\beta x+\gamma x^2))/2\sigma^2}
\]

Log likelihood of the dataset:

\[
\log Pr(y_1, ..., y_n; \alpha, \beta, \gamma, \sigma) = -\frac{n}{2} \log(2\pi\sigma^2) - \frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2
\]

where \(\hat{y}_i = \alpha + \beta x_i + \gamma x_i^2\)

Optimize over the unknown parameters:
Example (regression)
Given a labelled dataset consisting of pairs \((x_i, y_i)\) of real numbers, fit the model \(Y_i \sim \mu_\theta(x_i) + N(0, \sigma^2)\).
(Here \(\mu_\theta(\cdot)\) is some specified function with unknown parameters \(\theta\).)

Log likelihood of the dataset:

\[
\log \Pr(y_1, \ldots, y_n; \theta, \sigma) = -\frac{n}{2} \log(2\pi\sigma^2) - \frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - \mu_\theta(x_i))^2
\]

Optimize over the unknown parameters \(\theta\) and \(\sigma\):

```python
class RWiggle(nn.Module):
    def __init__(self):
        super().__init__()
        self.\mu = ...
        self.\sigma = nn.Parameter(torch.tensor(1.0))

    # compute log \Pr(y;x)
    def forward(self, y, x):
        \(\sigma^2 = self.\sigma ** 2\)
        return -0.5*torch.log(2*\pi*\sigma^2) - torch.pow(y - self.\mu(x), 2) / (2*\sigma^2)

x, y = ...
mymodel = RWiggle()

optimizer = optim.Adam(mymodel.parameters())
for epoch in range(10000):
    optimizer.zero_grad()
    loglik = torch.sum(mymodel(y, x))
    (-loglik).backward()
    optimizer.step()
```

See section 3.3 of printed notes. Or work through the tutorial [to be released tonight].
self.μ = nn.Sequential(
    nn.Linear(1,4), nn.LeakyReLU(),
    nn.Linear(4,20), nn.LeakyReLU(),
    nn.Linear(20,20), nn.LeakyReLU(),
    nn.Linear(20,1)  )
This is your machine learning system?

Yup! You pour the data into this big pile of linear algebra, then collect the answers on the other side.

What if the answers are wrong?

Just stir the pile until they start looking right.

https://xkcd.com/1838
CC BY-NC 2.5
How do linear algebra & nonlinear activation functions let me approximate my dataset?

Let $\sigma(x)$ be the sigmoid function,

$$\sigma(x) = \frac{e^x}{1 + e^x}$$
QUESTIONS FROM WEDNESDAY

How would we specify a probability model that uses several predictor variables?

\[ Y \sim N(\mu_\theta(\tilde{x}), \sigma^2) \]

where \( \tilde{x} \) is a tuple of predictor variables, \( \tilde{x} = (x_1, ..., x_d) \)
and \( \mu_\theta: \mathbb{R}^d \to \mathbb{R} \)
**Supervised Learning**

- **Data:** \( (x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n) \)
- **Labels:** \( y_1, y_2, \ldots, y_n \)
- **Task:** fit the probability model \( \Pr_Y(y : f_\theta(x)) \)
- **Training goal:** MLE

**Generative Modelling**

- **Data:** \( \{x_1, x_2, \ldots, x_n\} \)
- **Labels:** n/a
- **Task:** fit the probability model \( \Pr_X(x ; \theta) \)
- **Training goal:** MLE
I’ve read that all generative modelling is basically about Autoencoders. What does this have to do with probability models?

- To train a generative model, we write out the log likelihood of our dataset \([x_1, \ldots, x_n]\) and find the parameters that maximize it:

\[
\max_{\theta} \sum_i \log \Pr_X(x_i; \theta)
\]

- This requires that we have a formula for \(\Pr_X(x; \theta)\)

CASE 1: There is a nice explicit expression (e.g. Transformers)

CASE 2: There is no closed-form expression (e.g. CNNs for image generation)

Both Autoencoders and GANs can be seen as ways to approximate \(\Pr_X(x; \theta)\)
Example (classification)
The MNIST dataset consists of pairs $(x_i, y_i)$, where each record consists of $x_i \in \mathbb{R}^{28 \times 28}$ an image of a handwritten digit and $y_i \in \{0, 1, \ldots, 9\}$ is its label.

We'd like to predict the digit, given an image. How might we learn to do this?
Example (classification)
The MNIST dataset consists of pairs $(x_i, y_i)$, where each record consists of $x_i \in \mathbb{R}^{28 \times 28}$ an image of a handwritten digit and $y_i \in \{0, 1, \ldots, 9\}$ is its label.

Devise a **probabilistic model** to predict the label of a given input image, and fit it.

What sort of PROBABILITY MODEL might we use for the response $Y$?
Example (classification)
The MNIST dataset consists of pairs \((x_i, y_i)\), where each record consists of \(x_i \in \mathbb{R}^{28 \times 28}\) an image of a handwritten digit and \(y_i \in \{0, 1, \ldots, 9\}\) is its label.

Devise a probabilistic model to predict the label of a given input image, and fit it.

\[
\hat{p}(x) = [p_0, \ldots, p_9] \quad \rightarrow \quad Y \sim \text{Cat}(\hat{p}(x))
\]

How can we make sure that \(\hat{p}\) is a valid probability vector?

(We need \(p_i \in [0,1]\) for each \(i\), and \(\Sigma_i p_i = 1\).)
Example (classification)
The MNIST dataset consists of pairs \((x_i, y_i)\), where each record consists of \(x_i \in \mathbb{R}^{28 \times 28}\) an image of a handwritten digit and \(y_i \in \{0, 1, \ldots, 9\}\) is its label.

Devise a probabilistic model to predict the label of a given input image, and fit it.

![Diagram](image)

Softmax function:
\[
p_k = \frac{e^{s_k}}{\sum_{\ell=0}^{9} e^{s_\ell}}
\]

How should we fit the function parameters \(\theta\)?
Example (classification)
The MNIST dataset consists of pairs \((x_i, y_i)\), where each record consists of \(x_i \in \mathbb{R}^{28 \times 28}\) an image of a handwritten digit and \(y_i \in \{0,1,\ldots,9\}\) is its label.

Devise a probabilistic model to predict the label of a given input image, and fit it.

Model for a single datapoint:

\[
\begin{align*}
\text{input } x & \quad \rightarrow \quad \mathbf{s} \in \mathbb{R}^{10} \\
\text{some arbitrary function,} \\
\text{parameters } \theta \\
\end{align*}
\]

\[
\mathbf{s} \rightarrow \hat{p} = \text{softmax}(\mathbf{s}) \quad Y \sim \text{Cat}(\hat{p})
\]

Likelihood of a single datapoint \(y\):

\[
Pr_Y(y; x, \theta) =
\]

Log likelihood of the dataset:

\[
\log \Pr(y_1, \ldots, y_n) = \quad \text{... we end up with the famous}
\]

"softmax cross-entropy loss function"