CAI 4104/6108 – Machine Learning Engineering:

Midterm Review

Prof. Vincent Bindschaedler

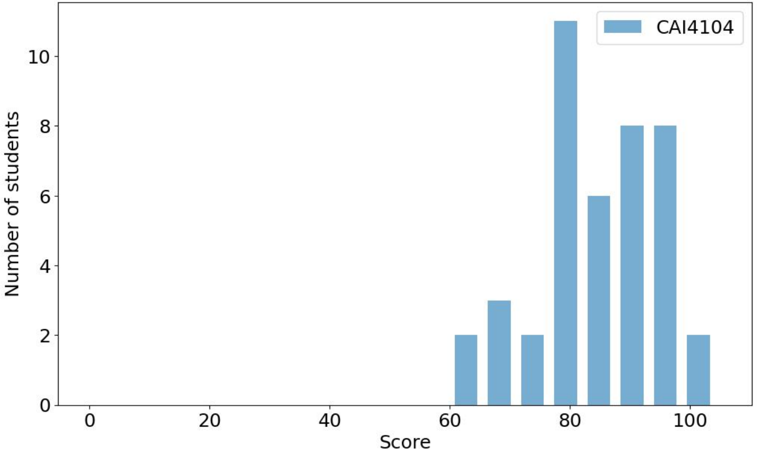
Spring 2024

Administrivia

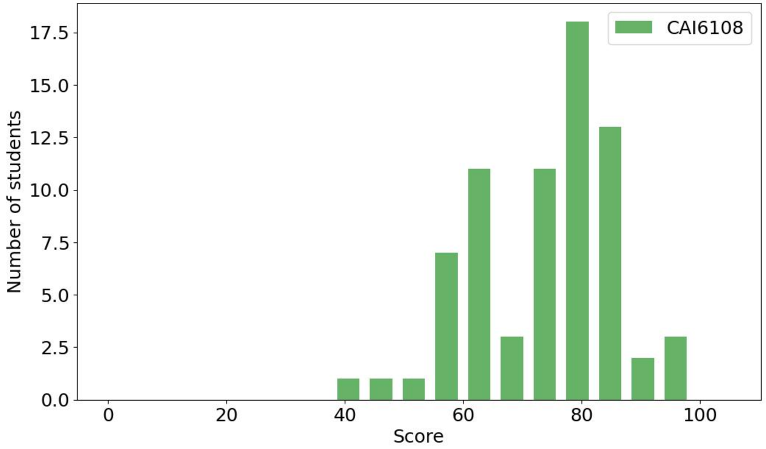
- **Homework 3** is out
  - Topic: Gradient Descent & Dimensionality Reduction
  - Due **3/20**
  - Do **not** submit the data file, only submit the notebook (.ipynb)
  - Advice: **start early**

Administrivia

- Midterm grades are posted
  - We will be adjusting grades based on **max score**
    - **Tentative** new max (=97 for 4104; 93 for 6108)
  - If you have questions about your exam, please contact me and (or) the TA team



- Mean (± std): 85.0 (± 9.4)
- Median: 85.5
- Min: 62, Max: 100



- Mean (± std): 74.5 (± 11.7)
- Median: 78.
- Min: 42.5, Max: 96

Short Answers Questions

- Given a dataset and a ML algorithm, briefly describe a procedure to determine if you have enough data to solve the problem.
- Consider k-Nearest Neighbors. What is the effect of increasing k in terms of bias and variance?
- Briefly describe is a way to reduce the number of values for a numerical feature/attribute.

Reminder: The Kernel Trick

- Wait! It is (computationally) expensive to transform our data to higher dimensional space
  - Can we do this transformation implicitly?
  - In other words: can we only reflect the transformation only in our cost function for optimization?
    - Yes! This is called the kernel trick!
  - Suppose we have a mapping Φ such that $\Phi(\boldsymbol{x})$ is in the higher dimensional space
    - For example: if $\boldsymbol{x}=(x_1,x_2)$ we can take: $\Phi(\boldsymbol{x})=(x_1^2, x_1 x_2, x_2^2)$
  - In the formulation of the *dual problem*, the only term involving feature vectors is their dot-product $\boldsymbol{x}_i \boldsymbol{x}_j$
    - So we define kernels in terms of $\boldsymbol{x}_i, \boldsymbol{x}_j$. That is: $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \Phi(\boldsymbol{x}_i)\, \Phi(\boldsymbol{x}_j)$ (dot-product)
  - Popular kernels
    - $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i \boldsymbol{x}_j)^2$ ("quadratic kernel")
    - $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i \boldsymbol{x}_j)^k$ ("polynomial kernel" of degree exactly k)
    - $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(- ||\boldsymbol{x}_i - \boldsymbol{x}_j||^2 /(2\sigma^2))$ ("RBF kernel") [feature space has infinite dimensions]
    - $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \tanh(\gamma\, \boldsymbol{x}_i \boldsymbol{x}_j + r)$ ("sigmoid kernel")
    - Note: σ, γ, and r are hyperparameters. For RBF we can set $\gamma = 1/(2\sigma^2)$ to be consistent with Scikit-learn!

Reminder: Kernel Trick: Why Does it Work?

- Mercer's Theorem:
  - If a function $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ satisfies *some conditions* then there exists some mapping Φ to possibly much higher dimension such that $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \Phi(\boldsymbol{x}_i)^{\mathrm{T}} \Phi(\boldsymbol{x}_j)$

- Why does this matter?
  - The dual formulation depends only (for the data) on the dot-product: $\boldsymbol{x}_i^T \boldsymbol{x}_j$

$$\min_{\boldsymbol{\alpha}} \tfrac{1}{2} \sum_i \sum_j \boldsymbol{\alpha}_i \boldsymbol{\alpha}_j y_i y_j \boxed{\boldsymbol{x}_i^T \boldsymbol{x}_j} - \sum_i \boldsymbol{\alpha}_i$$

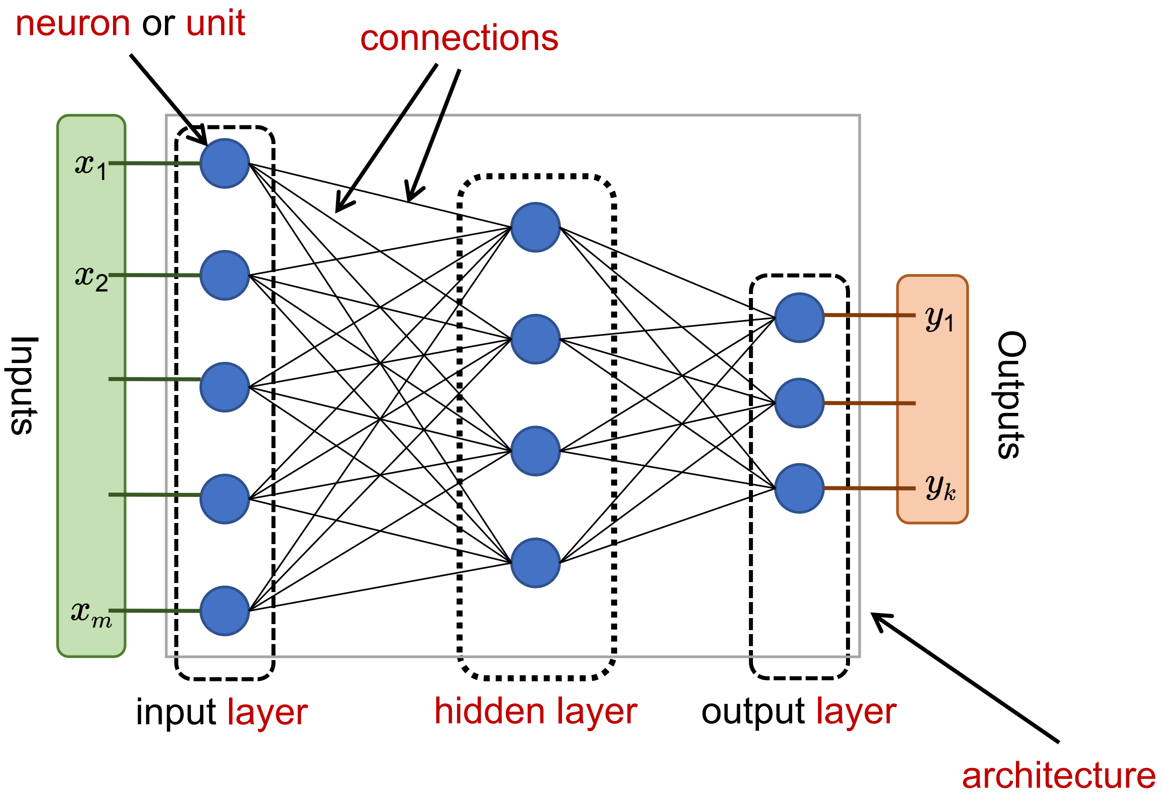
$$\text{such that: } \boldsymbol{\alpha}_i \geq 0 \text{ for } i=1,2,\ldots,n \text{ and } \sum_i \boldsymbol{\alpha}_i \boldsymbol{\alpha}_j y_i = 0$$

  - So we can replace that term by $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ since $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \Phi(\boldsymbol{x}_i)^{\mathrm{T}} \Phi(\boldsymbol{x}_j)$
  - Observe: we do not need to know how to compute Φ
    - In some cases we cannot even compute it
    - For example: RBF kernel Φ maps points to **infinite-dimensional** space

Problems

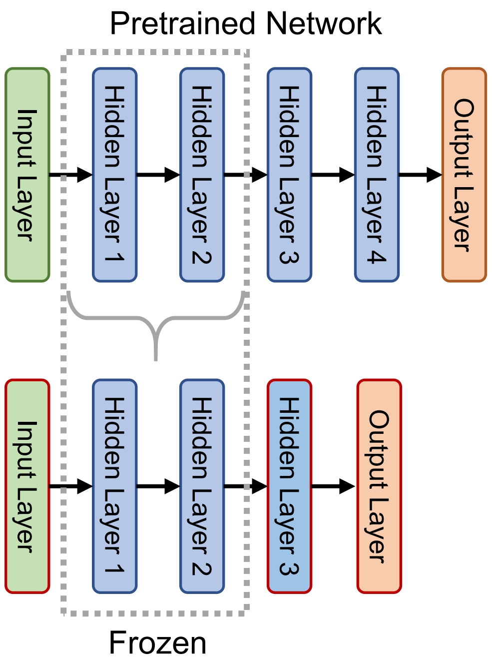
- <u>Kernels:</u> Your friend Carol is starting to do ML research. For this she needs to learn about kernels. Recall that a kernel is a function $K(x_1, x_2)$ that maps a pair of input feature vectors x_1, x_2 to a different space.
  - (a) (3pts) Briefly explain for what reason we may want to use a kernel?
  - (b) (4pts) In machine learning, we use kernels that satisfy $K(x_1, x_2) = \Phi(x_1) \cdot \Phi(x_2)$, where $\cdot$ denotes the dot- product and Φ is some mapping. What is the "kernel trick" and what does it have to do with this condition?
  - (c) (8pts) The following are kernels that Carol is considering using for her ML pipeline. For each one state whether it is a valid kernel.
    1. $K(x_1, x_2) = -1$
    2. $K(x_1, x_2) = x_1 \cdot x_2 + c$, where $c>0$ is a constant
    3. $K(x_1, x_2) = \exp(x_1) \cdot \exp(x_2)$
    4. $K(x_1, x_2) = x_1 - x_2$
  - (d) (3pts) Carol decided to use the RBF kernel $K(x_1, x_2) = \exp(-\alpha||x_1-x_2||^2)$. Explain how Carol should optimize the value of α.

Reminder: Neural Network Terminology



Transfer Learning

- Should you train a deep neural network from scratch?
  - Not always. When possible you should use transfer learning:
    - Pick a pre-trained deep neural network in the same or related domain
    - Then fine-tune on the task you care about
- Reusing an existing deep neural network
  1. Pick some layers to reuse (typically the earlier layers)
  2. Freeze these layers
     - This will set the corresponding parameters as non-trainable
       - Optimization: you can actually ***cache*** the outputs of frozen layers for every input
  3. Add your own layers hidden layer(s)
  4. Replace or discard upper layers
     - You should always discard the existing output layer and use your own



Next Time

- Friday (3/8): Exercise

- Upcoming:
  - Homework 3 is due 3/20