Exercises week 44

October 27-31, 2025

Date: **Deadline is Friday October 31 at midnight**

Overarching aims of the exercises this week

The exercise set this week has two parts.

1. The first is a version of the exercises from week 39, where you got started with the report and github repository for project 1, only this time for project 2. This part is required, and a short feedback to this exercise will be available before the project deadline. And you can reuse these elements in your final report.

2. The second is a list of questions meant as a summary of many of the central elements we have discussed in connection with projects 1 and 2, with a slight bias towards deep learning methods and their training. The hope is that these exercises can be of use in your discussions about the neural network results in project 2. **You don't need to answer all the questions, but you should be able to answer them by the end of working on project 2.**

Deliverables

First, join a group in canvas with your group partners. Pick an avaliable group for Project 2 in the “People” page. If you don't have a group, you should really consider joining one!

Complete exercise 1 while working in an Overleaf project. Then, in canvas, include

- An exported PDF of the report draft you have been working on.
- A comment linking to the github repository used in exercise **1d)**

Exercise 1:

Following the same directions as in the weekly exercises for week 39:

a) Create a report document in Overleaf, and write a suitable abstract and introduction for project 2.

b) Add a figure in your report of a heatmap showing the test accuracy of a neural network with [0, 1, 2, 3] hidden layers and [5, 10, 25, 50] nodes per hidden layer.

c) Add a figure in your report which meets as few requirements as possible of what we consider a good figure in this course, while still including some results, a title, figure text, and axis labels. Describe in the text of the report the different ways in which the figure is lacking. (This should not be included in the final report for project 2.)

d) Create a github repository or folder in a repository with all the elements described in exercise 4 of the weekly exercises of week 39.

e) If applicable, add references to your report for the source of your data for regression and classification, the source of claims you make about your data, and for the sources of the gradient optimizers you use and your general claims about these.

Exercise 2:

a) Linear and logistic regression methods

1. What is the main difference between ordinary least squares and Ridge regression?
2. Which kind of data set would you use logistic regression for?
3. In linear regression you assume that your output is described by a continuous non-stochastic function $f(x)$. Which is the equivalent function in logistic regression?
4. Can you find an analytic solution to a logistic regression type of problem?
5. What kind of cost function would you use in logistic regression?

Answers exercise 2a)

1) The main difference between OLS and Ridge regression is that Ridge regression adds a penalty (regularization) term that penalizes large coefficients. The aim of this is to reduce the variance, and avoid overfitting. However, it may introduce bias.

2) Logistic regression is used for classification. So we would use a data set where the target values are catagorical.

3) The sigmoid function, $\frac{1}{1+e^{-z}}$, which maps inputs to probabilities between 0 and 1.

4) No, because the cost function is non-linear. So we need to use iterative optimization methods such as gradient descent.

5) In logistic regression we typically use the cross-entropy (log loss) loss function:

$$C = -\frac{1}{n}\sum_i [y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)]$$

b) Deep learning

1. What is an activation function and discuss the use of an activation function? Explain three different types of activation functions?
2. Describe the architecture of a typical feed forward Neural Network (NN).

3. You are using a deep neural network for a prediction task. After training your model, you notice that it is strongly overfitting the training set and that the performance on the test isn't good. What can you do to reduce overfitting?
4. How would you know if your model is suffering from the problem of exploding gradients?
5. Can you name and explain a few hyperparameters used for training a neural network?
6. Describe the architecture of a typical Convolutional Neural Network (CNN)
7. What is the vanishing gradient problem in Neural Networks and how to fix it?
8. When it comes to training an artificial neural network, what could the reason be for why the cost/loss doesn't decrease in a few epochs?
9. How does L1/L2 regularization affect a neural network?
10. What is(are) the advantage(s) of deep learning over traditional methods like linear regression or logistic regression?

Answers exercise 2b)

1) An activation function introduces non-linearity to a neural network, allowing it to "learn" complex relationships. Three types of activation functions:

- Sigmoid function: $\sigma(x) = \frac{1}{1+e^{-x}}$. "Squeezes" values between 0 and 1.
- ReLU function: $f(x) = \max(0, x)$ reduces vanishing gradients.
- Softmax: $Softmax(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}}$. Used in the output layer for multiclass classification. It converts raw output scores into probabilities that sum to one, allowing direct interpretation as class likelihoods.

2) A typical FFNN consists of an input layer, an output layer, and a number of hidden layers. Each layer has a number of nodes, and the layers are connected by synapses. Information flows forward from inputs, through weighted connections and activation functions, to the output.

3) To reduce overfitting, one can add regularization to the cost function, or reduce the model complexity.

4) Training becomes unstable: loss oscillates or becomes NaN, and weights grow extremely large. You can detect it by monitoring large gradients during training.

5)

- Learning rate: the step size in gradient descent
- Number of neurons in each layer, number of hidden layers.

c) Optimization part

1. Which is the basic mathematical root-finding method behind essentially all gradient descent approaches(stochastic and non-stochastic)?
2. And why don't we use it? Or stated differently, why do we introduce the learning rate as a parameter?
3. What might happen if you set the momentum hyperparameter too close to 1 (e.g., 0.9999) when using an optimizer for the learning rate?
4. Why should we use stochastic gradient descent instead of plain gradient descent?
5. Which parameters would you need to tune when use a stochastic gradient descent approach?

d) Analysis of results

1. How do you assess overfitting and underfitting?
2. Why do we divide the data in test and train and/or eventually validation sets?
3. Why would you use resampling methods in the data analysis? Mention some widely popular resampling methods.
4. Why might a model that does not overfit the data (maybe because there is a lot of data) perform worse when we add regularization?