Deep Learning

Andreas Neuhierl

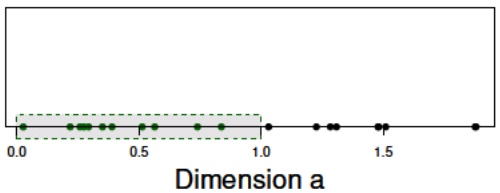
Washington University in St. Louis, Olin School of Business

FinTech Spring 2022

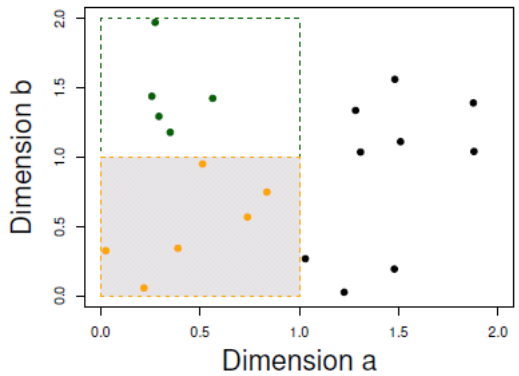
Optional Readings

- Chapter 6 in Goodfellow et al. (2016)
- Michael Nielsen's online book - **Link**

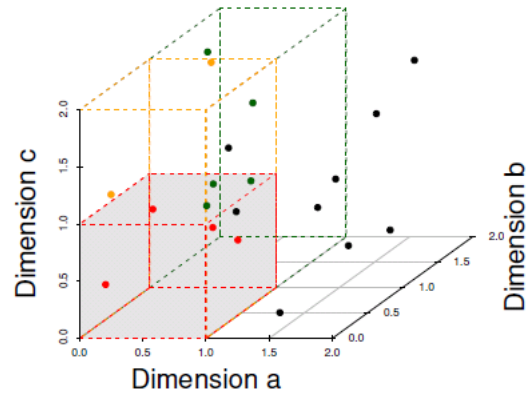
Detour - Curse of Dimensionality



(a) 11 Objects in One Unit Bin



(b) 6 Objects in One Unit Bin



(c) 4 Objects in One Unit Bin

Source: Parsons et. al. (2004)

- Nonparametric - rate of convergence of $N^{-4/(4+p)}$
- Linear model - rate of convergence N^{-1}
- NNs work remarkably well in high dimensions

Why Now?

1952 Stochastic Gradient Descent

1958 Perceptron

1986 Backpropagation

1988 Statistical properties (Hornik et al. (1989), Cybenko (1989))

1995 Convolution neural networks

New tricks, ReLu, Softmax...

- **Not exactly new?!**
- **Big Data**
- **Hardware (GPU, TPU)**
- **Software**

Cookbook Procedure I

1. Training data $\{x_i, y_i\}_{i=1}^N$
2. Choose
   - Decision function $\hat{y} = \hat{f}(x_i)$
   - Loss function $L(\hat{y}_i, y_i) \in \mathbb{R}$
3. Set goal:

$$\hat{f} = \arg\min_{f \in \mathcal{F}} \sum_{i=1}^{N} L(f(x_i), y_i)$$

4. Train the model - stochastic gradient descent

What's new here?

- Neural networks really good at approximating and unknown f
- "Escape" the curse of dimensionality

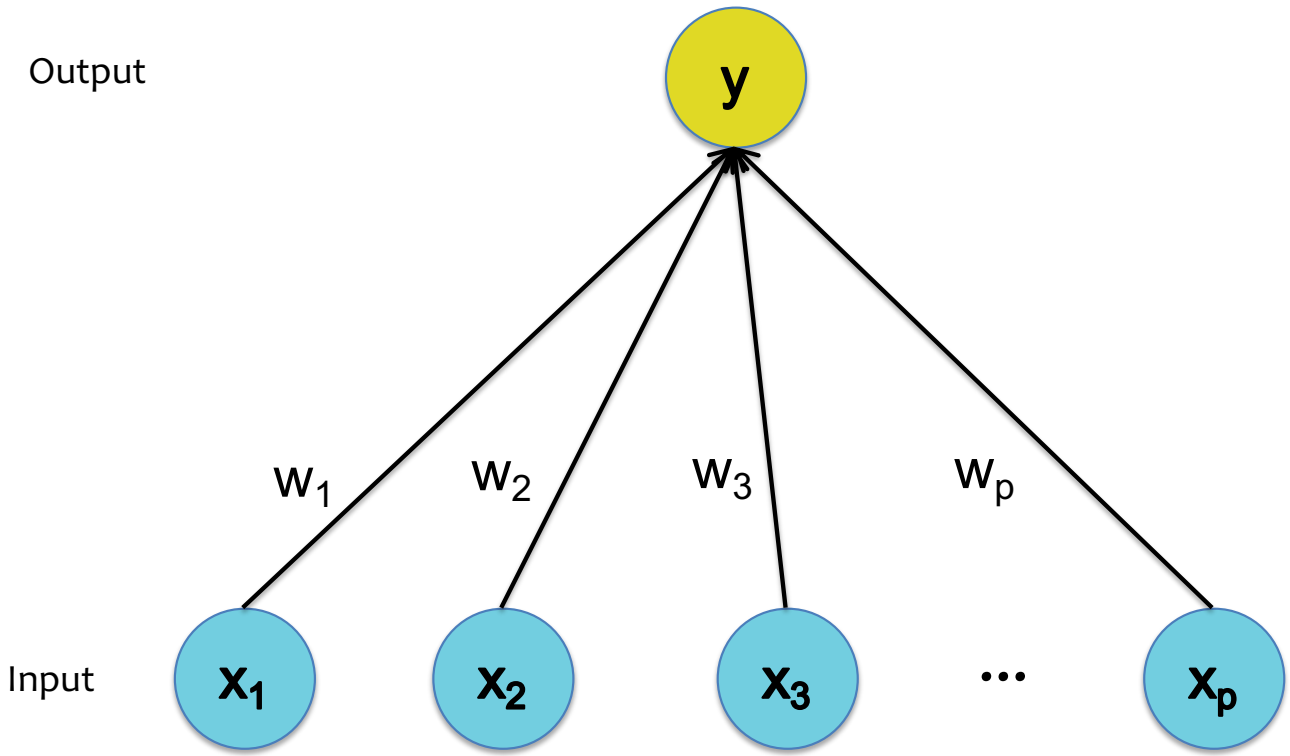
Cookbook Procedure II

1. Loss functions
   - MSE - $L = \frac{1}{N}\sum_i (y_i - \hat{y}_i)^2$ (regression)
   - MAE - $L = \frac{1}{N}\sum_i |y_i - \hat{y}_i|$ (regression)
   - Cross-entropy - $L = \sum_i y_i \log(\hat{y}_i)$ (classification)

2. Some other choices
   - Architecture: Start w/ a simple network and expand
   - Activation functions
   - Tuning parameters

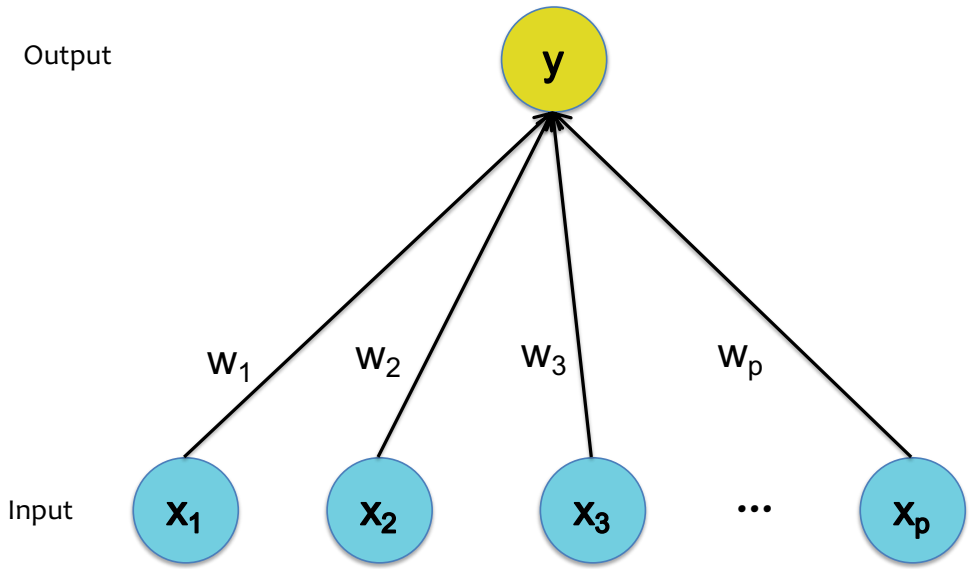
Linear Model as a Neural Network

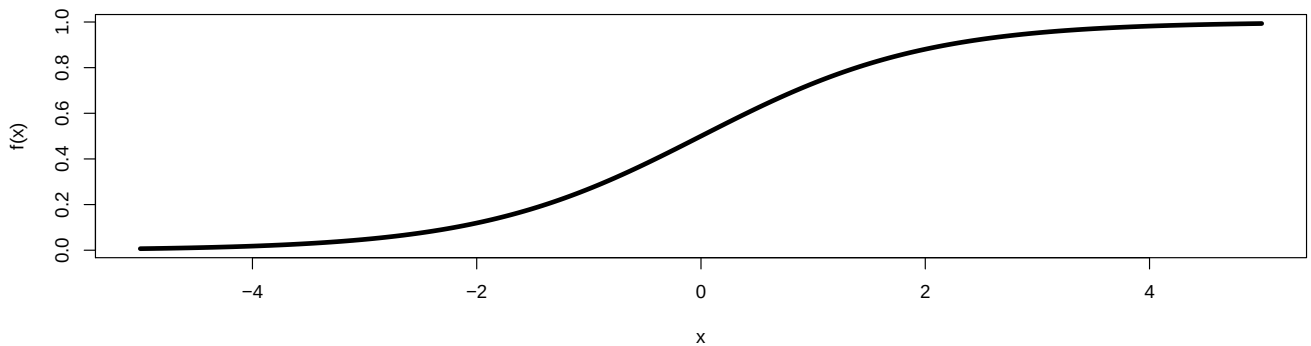
$$y = \sigma(Wx + b), \quad \sigma(a) = a$$



Let's go Nonlinear

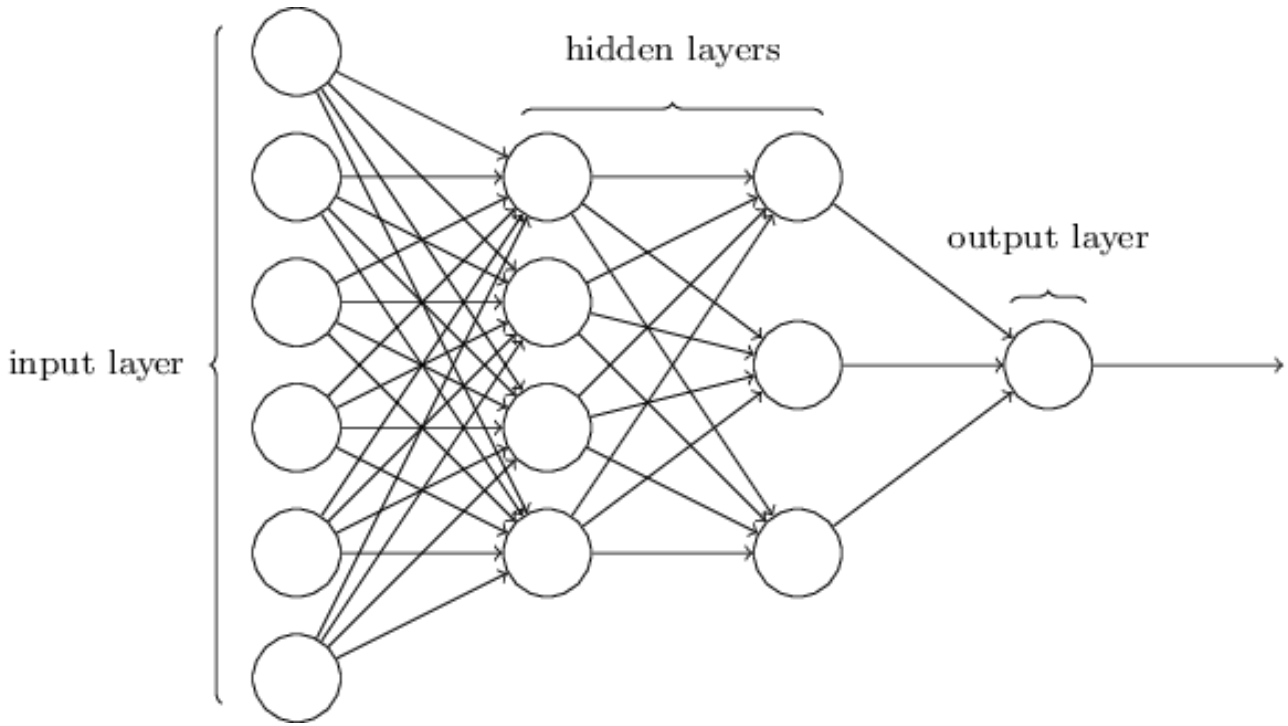
$$y = \sigma(Wx + b), \quad \sigma(a) = \frac{1}{1 + \exp(-a)}$$





Deeper...

- So far our networks had one layer
- Depths denote the number of layers
- Width is the # of nodes in the hidden layers



Source: http://www.sci.utah.edu/

Many, many parameters

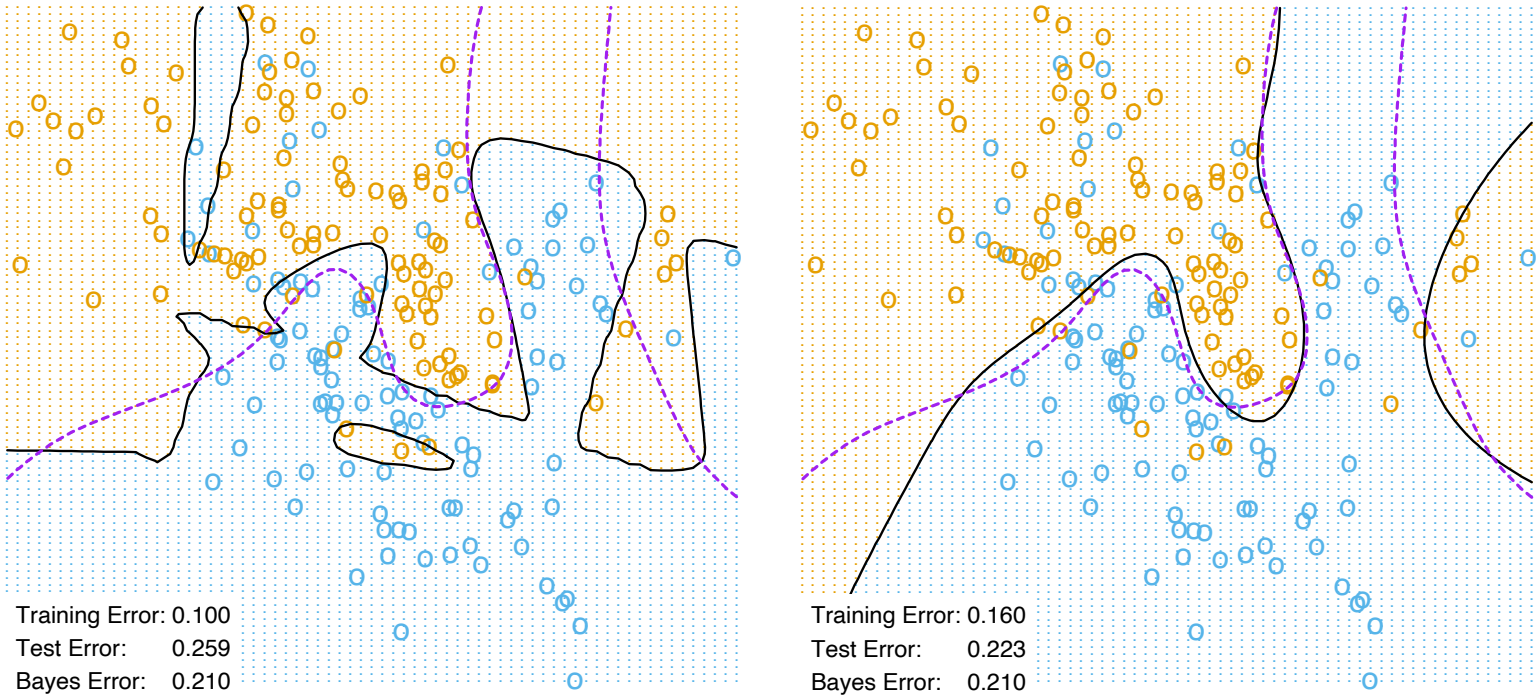
Example from MNIST (digit recognition):

- 784 input layers (28 by 28 pixels)
- 2 hidden layers with 16 nodes each
- 10 output layers
- $784 \times 16 + 16 \times 16 + 16 \times 10$ weights
- $16 + 16 + 10$ biases
- 13002 parameters

Neural networks are high-variance, low bias models

- Intuition is difficult
- Beware of overfit

Overfit



Source: Friedman et al. (2017)

- Overfit on the left
- Regularization on the right

Cross-Validation

Talked about the difference between

- training error rate
- test error rate

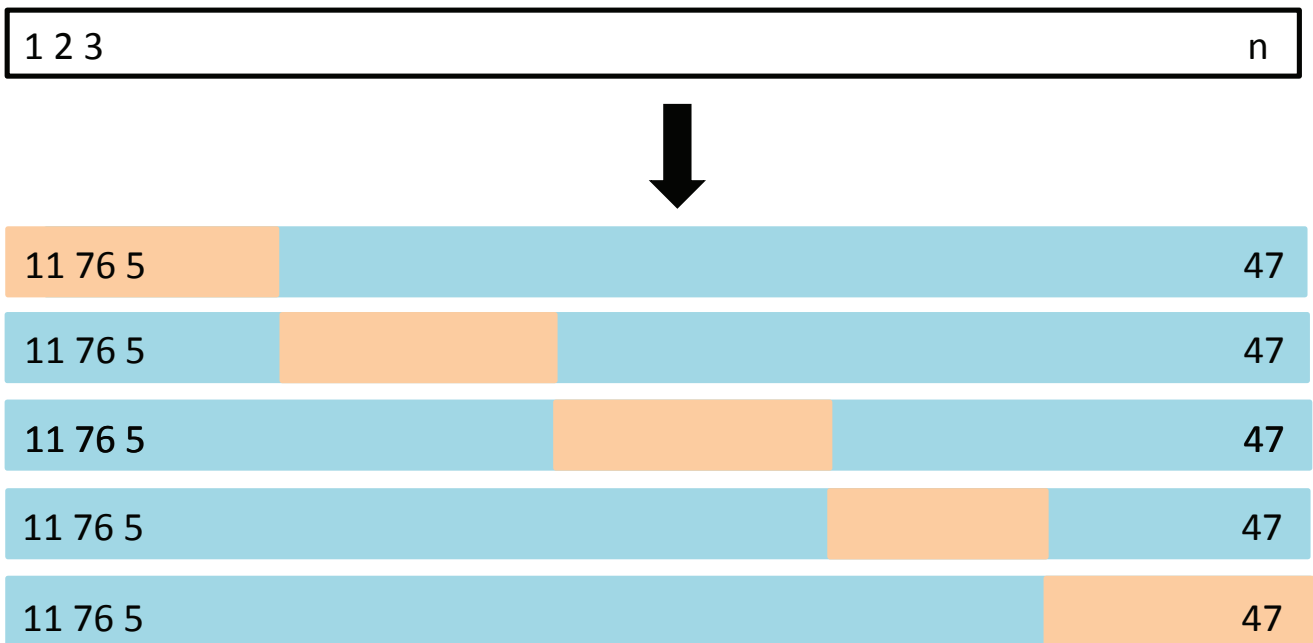
Training error. Easy to compute, but not very informative

Test error. Not always available

Some methods adjust the training error for the complexity of the model (Adjusted R^2)

k-Fold Cross-Validation

- Use part of the sample to estimate the model
- Use the data you did not use to estimate the model to evaluate the performance
- Randomly dividing the set of observations into k groups (folds)



k-Fold Cross-Validation

The k-fold CV estimate is computed by averaging these values

$$CV_{(k)} = \frac{1}{k} \sum_{i=1}^{k} MSE_i$$

In practice, one typically performs k-fold CV using k = 5 or k = 10.

Does it Work?

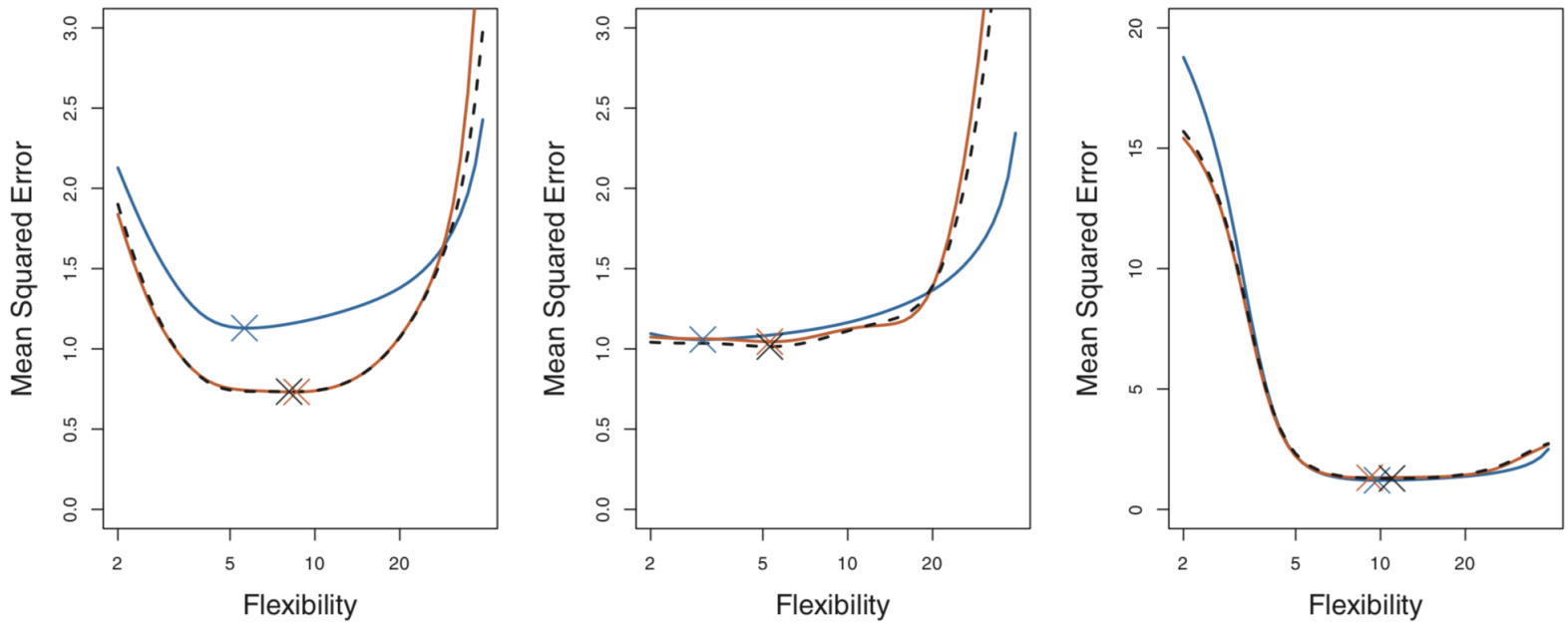


FIGURE 5.6. *True and estimated test MSE for the simulated data sets in Figures 2.9 (*left*), 2.10 (*center*), and 2.11 (*right*). The true test MSE is shown in blue, the LOOCV estimate is shown as a black dashed line, and the* 10*-fold CV estimate is shown in orange. The crosses indicate the minimum of each of the MSE curves.*

Architecture

- Depths denote the # of layers
- Width the # of nodes in the hidden layers
- Activation function for each layer
- Output layer can be
  - One node for regression
  - Multiple nodes for classification
- Choosing a "good" architecture is a bit of an art
- Even "simple" networks require quite some choices from the modeler
- But they can do amazing things

Topics for a more in-depth Course

- Non-convex problem - tough to solve!
- Algorithms
- "Tricks"
  - Shrinkage
  - Early stopping
  - Dropout
  - Momentum
  - Sensitivity
  - ...
  - How do you know the solution is good?
- Same idea from before
  - loss on test sample
  - loss on training sample
- Theory is often quite technical

Conclusions

- We've scratched the surface of a fascinating area.
- There are many newer developments
  - Convolution neural networks
  - Recurrent neural networks (LSTM)
  - General adversarial networks (GAN)
  - ....
- Coding time ☺

References I

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems 2*(4), 303–314.

Friedman, J., T. Hastie, R. Tibshirani, et al. (2017). *The elements of statistical learning* (2 ed.). Springer series in statistics New York.

Goodfellow, I., Y. Bengio, A. Courville, and Y. Bengio (2016). *Deep learning*, Volume 1. MIT press Cambridge.

Hornik, K., M. Stinchcombe, and H. White (1989). Multilayer feedforward networks are universal approximators. *Neural networks 2*(5), 359–366.