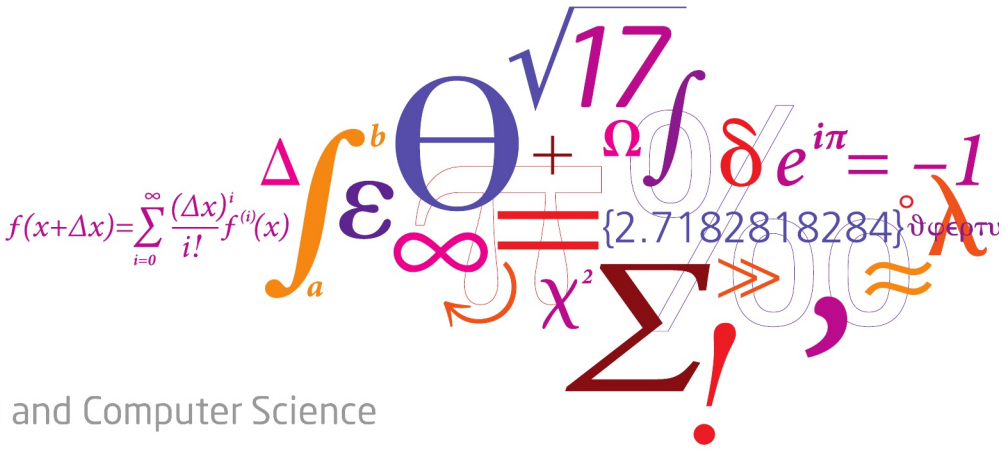
02450: Introduction to Machine Learning and Data Mining

Artificial Neural Networks and Bias/Variance

Tue Herlau

DTU Compute, Technical University of Denmark (DTU)



DTU Compute
Department of Applied Mathematics and Computer Science

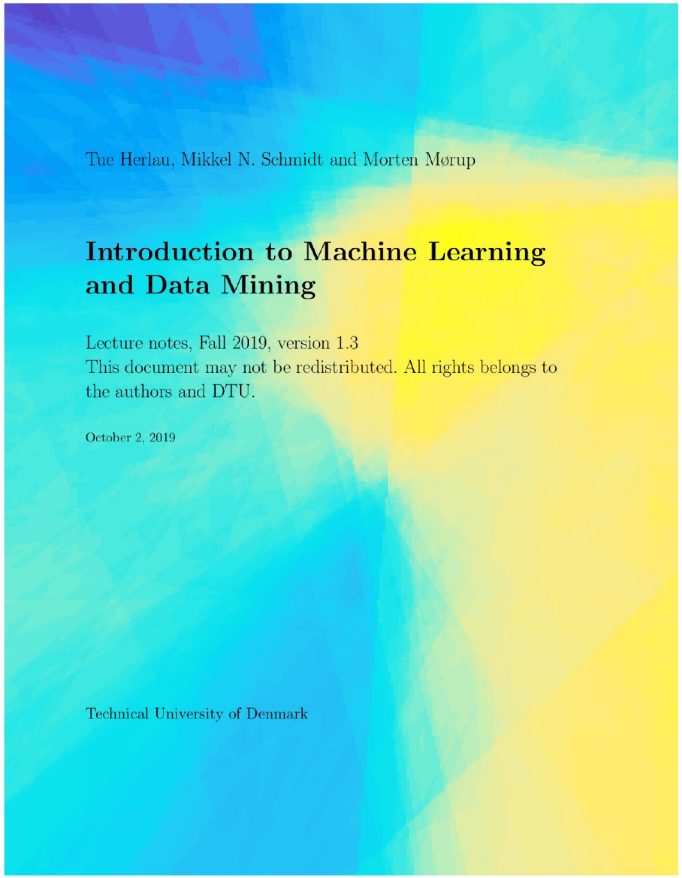
Today

Feedback Groups of the day:

Julie Liu Olin, Nicklas Leander Lund, Jakob Andreas Lodberg, Asbjørn Kjær Olling, David Collin Jørgensen, Loreto Andrea Maldonado Mamut, Jens Dieter Kjær Modvig, Victoria Zillmer Payne, Anne Agathe Pedersen, Laura López Acedo, Agne Suminaite, Mahammed Zeeshan Siddique, Renjue Sun, Søren Winkel Holm, Peter Øllgaard Vilhelmsen, Benjamin Søndberg, Christian Noes-Rasmussen, Kabir Khanna, Samia Siddique Sama, Yiming Sun, ZhengZhong Sun, Miklós Kristóf Jásdi, Laia Poqui i Sallés, Nilas Tim Schüsler, Olivier Stephane Bonde Petersen, Jacob Kæstel-Hansen, Andreas Alsing Hauschild, Anders Thuelund Jensen, Simon Busch Iversen, Martin Nemes, Marius Spagl, Isaac Irani, Miguel Temboury Gutierrez, Kristian Sofus Knudsen, Carina Thusgaard Refsgaard, Martin Enggaard Kristensen, Anton Juhl, Zhaofeng Cai, François Gruwé, Michael Koch, Nicolaj Hostrup Langkjær, Lars Lohmann, Oldouz Majidi, Miriam Mazzeo

Reading material:

Chapter 14, Chapter 15



Lecture Schedule

1. Introduction
   3 September: C1

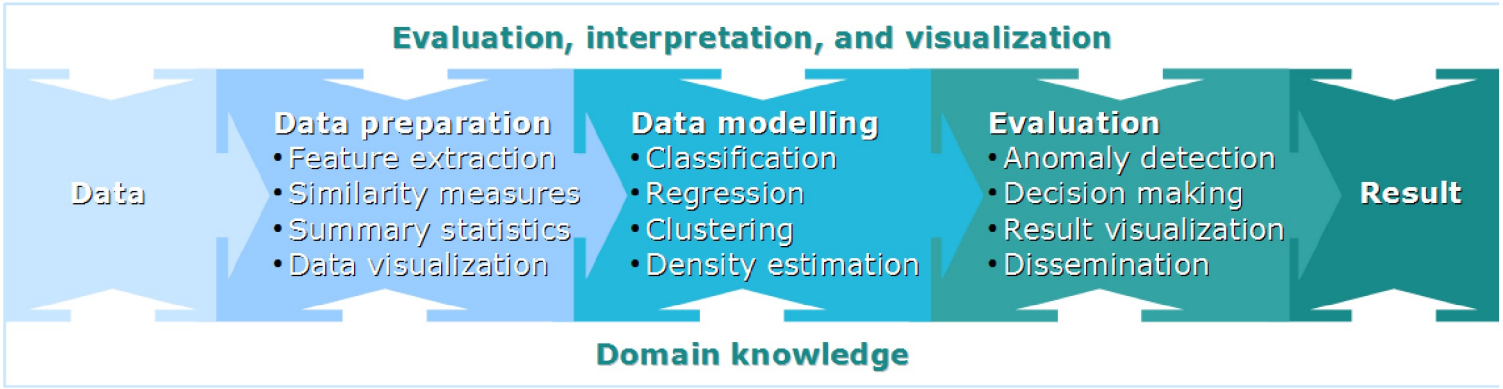
   Data: Feature extraction, and visualization
2. Data, feature extraction and PCA
   10 September: C2, C3
3. Measures of similarity, summary statistics and probabilities
   17 September: C4, C5
4. Probability densities and data Visualization
   24 September: C6, C7

   Supervised learning: Classification and regression
5. Decision trees and linear regression
   1 October: C8, C9 **(Project 1 due before 13:00)**
6. Overfitting, cross-validation and Nearest Neighbor
   8 October: C10, C12
7. Performance evaluation, Bayes, and Naive Bayes
   22 October: C11, C13
8. **Artificial Neural Networks and Bias/Variance**
   29 October: C14, C15
9. AUC and ensemble methods
   5 November: C16, C17

   Unsupervised learning: Clustering and density estimation
10. K-means and hierarchical clustering
    12 November: C18 **(Project 2 due before 13:00)**
11. Mixture models and density estimation
    19 November: C19, C20
12. Association mining
    26 November: C21

    Recap
13. Recap and discussion of the exam
    3 December: C1-C21 **(Project 3 due before 13:00)**

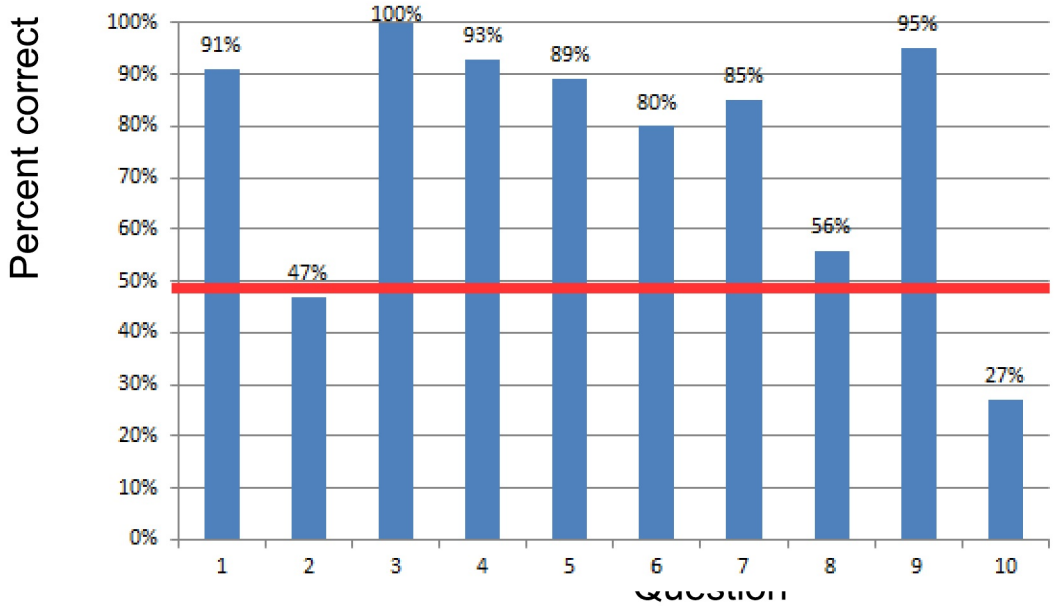
Piazza online help: `https://piazza.com/dtu.dk/fall2019/02450`
Videos/streaming of lectures: `https://video.dtu.dk`



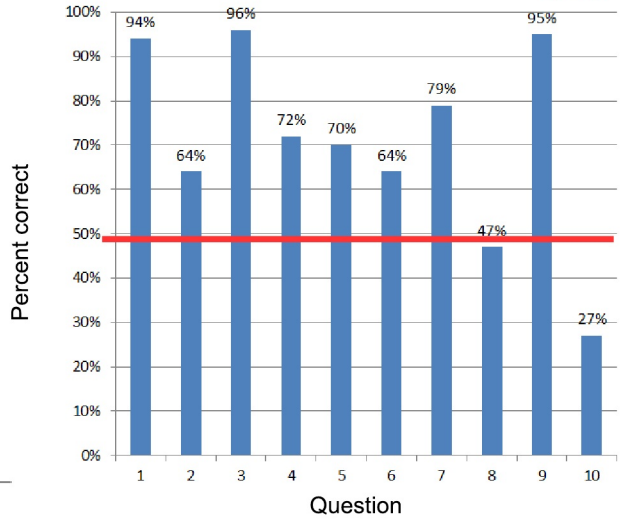
Learning Objectives

- Understand the Bias-Variance decomposition
- Understand and apply regularized least squares regression (i.e. ridge regression)
- Understand the principles behind artificial neural networks (ANNs) and how ANNs can be used for classification and regression
- Understand how logistic regression and ANNs can be extended to multi-class classification

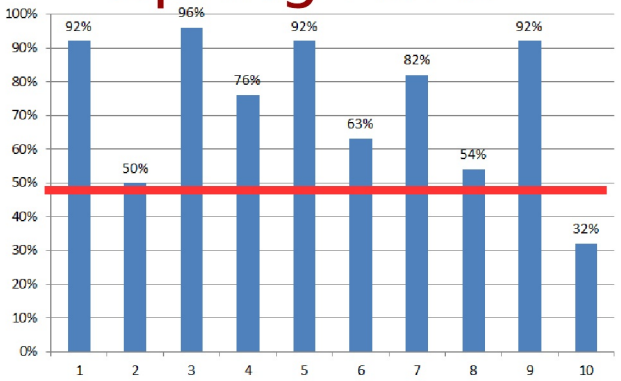
Midterm practice test results



Fall 2018



Spring 2019



Question 2:
Consider the classification problem given in figure 1 and the Decision Tree in figure 2 with two decisions denoted A and B. We will let $\boldsymbol{x}_n$ define the x_1 and x_2 coordinates of a given observation whereas $\boldsymbol{x}_n$ -0.5·**1** denotes the subtraction of 0.5 from x_1 and x_2.
Which one of the following classification rules would lead to a correct classification of the data?

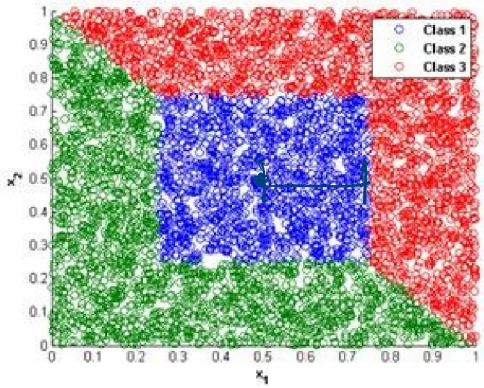


Figure 1

A: A: $\|x_n - 0.5 \cdot 1\|_1 \leq 0.25$, B: $\|x_n\|_\infty \leq 1$

B: A: $\|x_n\|_1 \leq 1$, B: $\|x_n - 0.5 \cdot 1\|_2 \leq \infty$

C: A: $\|x_n - 0.5 \cdot 1\|_2 \leq 0.25$, B: $\|x_n\|_\infty \leq 1$

D: A: $\|x_n - 0.5 \cdot 1\|_\infty \leq 0.25$, B: $\|x_n\|_1 \leq 1$

E: Don't know

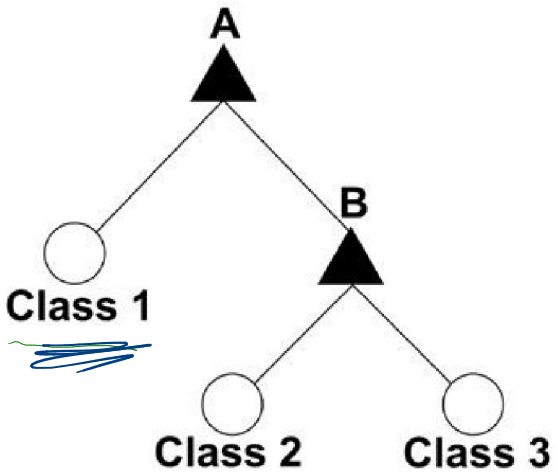


Figure 2

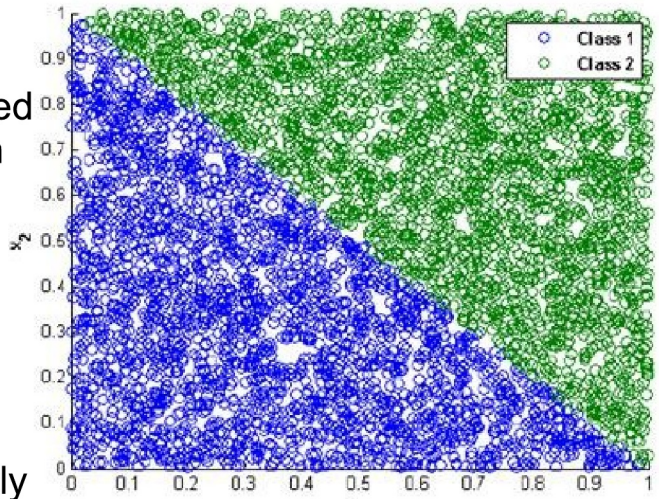
$$\tilde{X} = U \Sigma V^T \qquad \Sigma = \begin{bmatrix} s_1 & & & \\ & s_2 & & \\ & & s_3 & \\ & & & s_4 \end{bmatrix}$$

Question 8:
When carrying out a principal component analysis of a dataset with four attributes we obtain the following singular values $s_1=4$, $s_2=2$, $s_3=1$, and $s_4=0$. Which one of the following statements is wrong?

A: The first principal component accounts for more than 60 % of the variation in the data.
B: The third principal component accounts for less than 5 % of the variation in the data.
C: The second principal component accounts for more than 20 % of the variation in the data.
D: The data can be perfectly represented in a three dimensional sub-space.
E: Don't know.

$$\frac{s_2^2}{s_1^2 + \cdots + s_4^2}$$

Question 10:
Consider the classification problem given in Figure 5 where x_1 and x_2 are used as features for a logistic regression classifier and a decision tree. The considered logistic regression models all include the constant term w_0. Which one of the following statements is wrong?



A: The two classes can be perfectly separated by a logistic regression model using x_1 and x_2 as features.

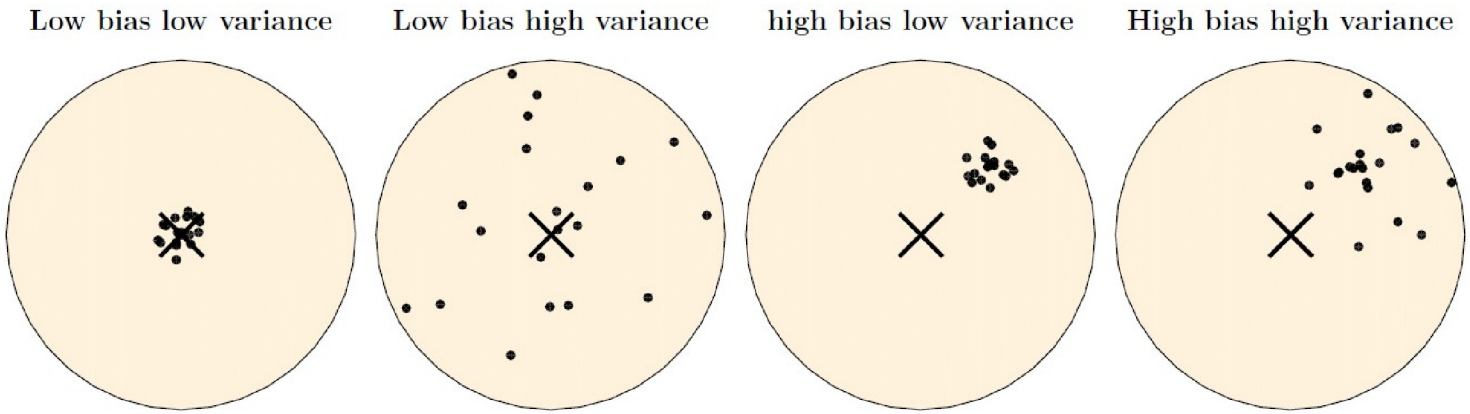
B: A decision tree with less than five nodes can perfectly separate the classes using only x_1 and x_2 as features.

C: A logistic regression model can perfectly separate the two classes using only the feature t given by $t = x_1 + x_2$.

D: In logistic regression the probability that each observation belong to the two classes can be derived from the logistic function.

E: Don't know.

What is bias and what is variance?



Regularized least squares

- Recall cost function from linear regression

$$E(\boldsymbol{w}) = \left\|\boldsymbol{y} - \tilde{\boldsymbol{X}}\boldsymbol{w}\right\|^2$$

- A parsimonious model can be obtained by **forcing** parameters towards zero.
- Problem: Columns of $\boldsymbol{X}$ have very different scale (i.e. require large/small values of $\boldsymbol{w}$)
- Therefore, standardize $\boldsymbol{X}$:

$$\hat{X}_{ij} = \frac{X_{ij} - \mu_j}{\hat{s}_j}, \quad \mu_j = \frac{1}{N}\sum_{i=1}^{N} X_{kj}, \quad \hat{s}_j = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(X_{ij} - \mu_j)^2}$$

- Note $\hat{\boldsymbol{X}}$ contains no constant term.

- Introduce regularization term $\lambda\|\boldsymbol{w}\|^2$ to penalize large weights:

$$E_\lambda(\boldsymbol{w}, w_0) = \sum_{i=1}^{N}(y_i - w_0 - \hat{\boldsymbol{x}}^\top \boldsymbol{w})^2 + \lambda\|\boldsymbol{w}\|^2 = \left\|\boldsymbol{y} - w_0\mathbf{1} - \hat{\boldsymbol{X}}\boldsymbol{w}\right\|^2 + \lambda\|\boldsymbol{w}\|^2$$

- We can solve for w_0 and $\boldsymbol{w}$:

$$\frac{dE_\lambda}{dw_0} = \sum_{i=1}^{N} -2(y_i - w_0 - \hat{\boldsymbol{x}_i}^\top \boldsymbol{w}) = -2N\mathbb{E}[y] - 2Nw_0 - N\left(\frac{1}{N}\sum_{i=1}^{N}\hat{\boldsymbol{x}_i}^\top\right)\boldsymbol{w} = 0$$

$$\Rightarrow w_0 = \mathbb{E}[y]$$

- With $\hat{y}_i = y_i - \mathbb{E}[y]$

$$E_\lambda = \left\|\hat{\boldsymbol{y}} - \hat{\boldsymbol{X}}\boldsymbol{w}\right\|^2 + \lambda\|\boldsymbol{w}\|^2$$

- Setting the derivative wrt. $\boldsymbol{w}$ equal to zero and solving for $\boldsymbol{w}$ yields

$$\boldsymbol{w}^* = (\hat{\boldsymbol{X}}^\top \hat{\boldsymbol{X}} + \lambda \boldsymbol{I})\backslash(\hat{\boldsymbol{X}}^\top \hat{\boldsymbol{y}})$$

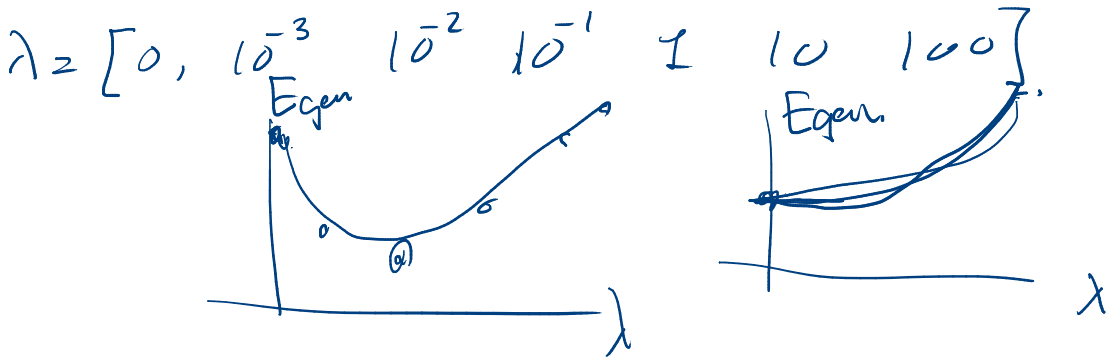
$$\frac{\partial E_\lambda}{\partial w} = -\hat{X}^T 2(\hat{y} - \hat{X}w) + 2\lambda w = 0$$
$$= (2\hat{X}^T X + 2\lambda I)w = 2\hat{X}^T y$$
$$w = (\hat{X}^T X + \lambda I)^{-1}\hat{X}^T y.$$

Selecting λ

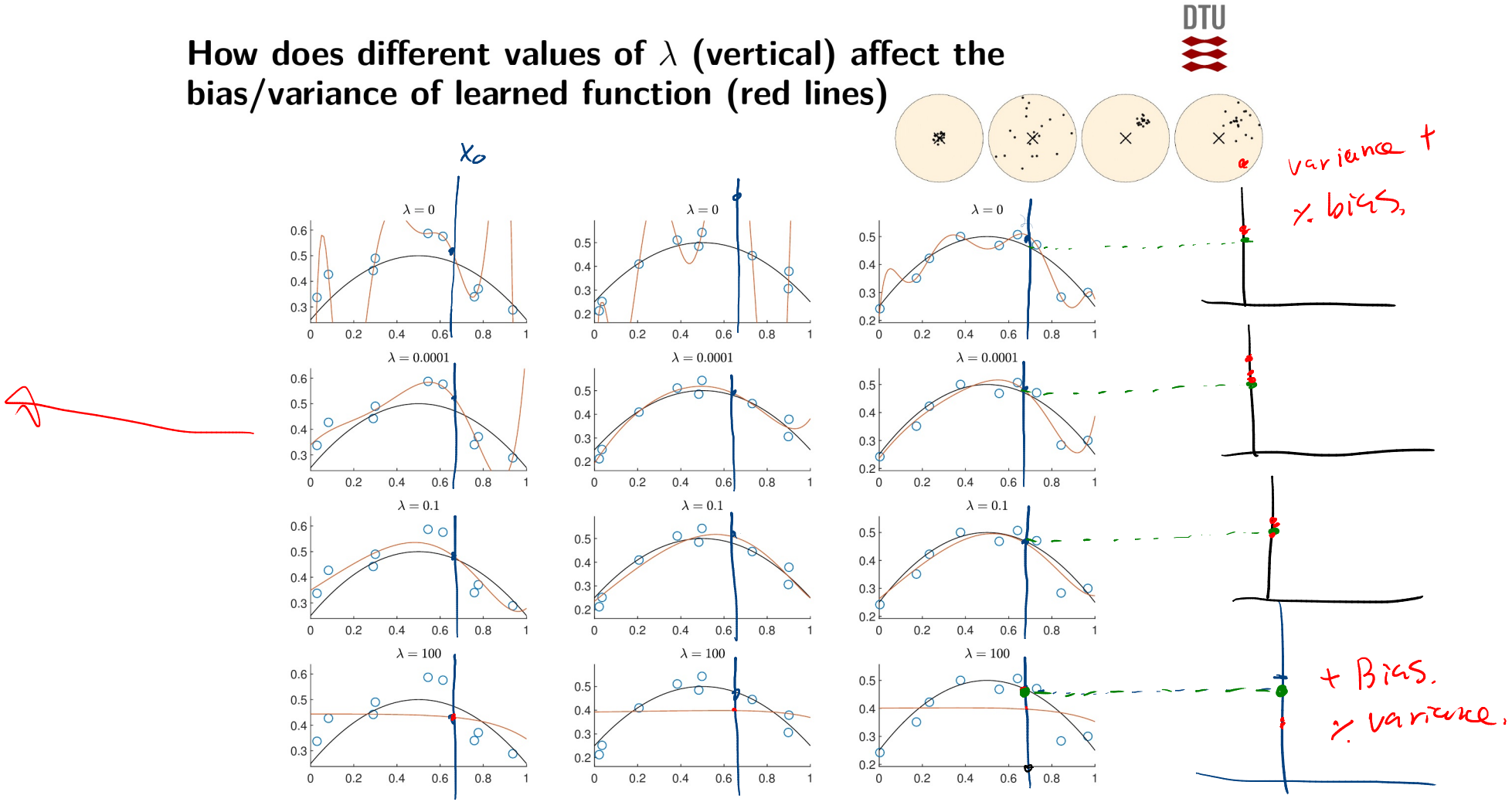
- Suppose

$$\boldsymbol{w}^* = (\hat{\boldsymbol{X}}^\top \hat{\boldsymbol{X}} + \lambda \boldsymbol{I}) \backslash (\hat{\boldsymbol{X}}^\top \hat{\boldsymbol{y}}) \propto \frac{Xy}{X^2 + \lambda}$$

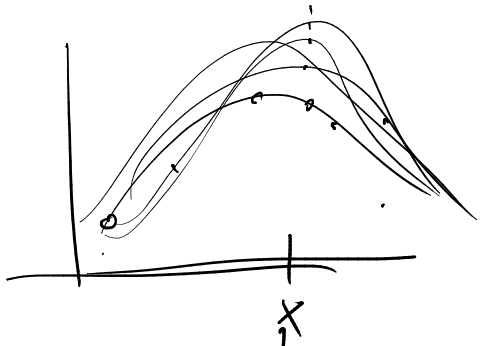
- So if $\lambda = 0$ then no effect, else if $\lambda \to \infty$ then $\boldsymbol{w}^* \to 0$
- λ controls complexity of model. Select λ using cross-validation



How does different values of λ (vertical) affect the bias/variance of learned function (red lines)



The Bias-Variance decomposition



We first consider $\boldsymbol{x}$ fixed

$$\mathbb{E}_{\mathcal{D},y|\boldsymbol{x}}\left[(y - f_{\mathcal{D}}(\boldsymbol{x}))^2\right]$$

$$= \mathbb{E}_{\mathcal{D},y|\boldsymbol{x}}\left[(y - \bar{y}(\boldsymbol{x}) + \bar{y}(\boldsymbol{x}) - f_{\mathcal{D}}(\boldsymbol{x}))^2\right]$$

$$= \mathbb{E}_{y|\boldsymbol{x}}\left[(y - \bar{y}(\boldsymbol{x}))^2\right] + \mathbb{E}_{\mathcal{D}}\left[(\bar{y}(\boldsymbol{x}) - f_{\mathcal{D}}(\boldsymbol{x}))^2\right] + 2\mathbb{E}_{\mathcal{D},y|\boldsymbol{x}}\left[(y - \bar{y}(\boldsymbol{x}))\,(\bar{y}(\boldsymbol{x}) - f_{\mathcal{D}}(\boldsymbol{x}))\right]$$

$$\mathbb{E}_{\mathcal{D}}\left[E^{\text{gen}}\right] = \mathbb{E}_{\mathcal{D},(\boldsymbol{x},y)}\left[(y - f_{\mathcal{D}}(\boldsymbol{x}))^2\right]$$

$$= \int p(\mathcal{D}) \int p(x,y)\,(y - f_{\mathcal{D}}(x))^2\,dx\,dy\,d\mathcal{D}$$

$$= \frac{1}{T}\sum_{t=1}^{T}\left(E^{\text{test}}_{\mathcal{D}_t^{\text{test}}}, f_{\mathcal{D}_t}\right)$$

$$= \frac{1}{T}\sum_{t=1}^{T}\left(\frac{1}{N}\sum_{i=1}^{N}\left(y_i^{t,\text{test}} - f_{\mathcal{D}_t}(x_i^{t,\text{test}})\right)^2\right)$$

T training sets, $\mathcal{D}_t$.
N test observations $\mathcal{D}_t^{\text{test}}$.
train model $f_{\mathcal{D}_t}$ on $\mathcal{D}_t$

$$\bar{y}(\boldsymbol{x}) = \mathbb{E}_{y|\boldsymbol{x}}[y] = \int p(y|x)\,y\,dy.$$

$$\mathbb{E}\left[(y - \bar{y}(x))^2 + (\bar{y}(x) - f_{\mathcal{D}}(x))^2 + (y - \bar{y}(x))(\bar{y} - f_{\mathcal{D}}(x))\right]$$

$\mathbb{E}[y - \bar{y}(x)] = \mathbb{E}[y] - \bar{y}(x) = \bar{y}(x) - \bar{y}(x) = 0$

$\mathbb{E}_x[f(x) + g(x)] = \mathbb{E}[f] + \mathbb{E}[g]$.

$\mathbb{E}_{z,x}[g(x)f(z)] = \mathbb{E}_z[\mathbb{E}_x[g(x)f(z)]] = \mathbb{E}_z[f(z)\mathbb{E}_x[g(x)]]$

$= \mathbb{E}_z[f_z]\,\mathbb{E}_x[g(x)]$

https://commons.wikimedia.org/wiki/File:Frustrated_man_at_a_desk_(cropped).jpg

The Bias-Variance decomposition

$$\mathbb{E}_{\mathcal{D}}\left[E^{\text{gen}}\right] = \mathbb{E}_{\mathcal{D},(\boldsymbol{x},y)}\left[\left(y - f_{\mathcal{D}}(\boldsymbol{x})\right)^2\right]$$

We first consider $\boldsymbol{x}$ fixed

$$\bar{y}(\boldsymbol{x}) = \mathbb{E}_{y|\boldsymbol{x}}\left[y\right]$$

$$\begin{aligned}
&\mathbb{E}_{\mathcal{D},y|\boldsymbol{x}}\left[\left(y - f_{\mathcal{D}}(\boldsymbol{x})\right)^2\right] \\
&= \mathbb{E}_{\mathcal{D},y|\boldsymbol{x}}\left[\left(y - \bar{y}(\boldsymbol{x}) + \bar{y}(\boldsymbol{x}) - f_{\mathcal{D}}(\boldsymbol{x})\right)^2\right] \\
&= \mathbb{E}_{y|\boldsymbol{x}}\left[\left(y - \bar{y}(\boldsymbol{x})\right)^2\right] + \mathbb{E}_{\mathcal{D}}\left[\left(\bar{y}(\boldsymbol{x}) - f_{\mathcal{D}}(\boldsymbol{x})\right)^2\right] + 2\mathbb{E}_{\mathcal{D},y|\boldsymbol{x}}\left[\left(y - \bar{y}(\boldsymbol{x})\right)\left(\bar{y}(\boldsymbol{x}) - f_{\mathcal{D}}(\boldsymbol{x})\right)\right]
\end{aligned}$$

https://commons.wikimedia.org/wiki/File:Frustrated_man_at_a_desk_(cropped).jpg

The Bias-Variance decomposition

$$\mathbb{E}_{\mathcal{D},y|\boldsymbol{x}}\left[\left(y - f_{\mathcal{D}}(\boldsymbol{x})\right)^2\right] = \mathbb{E}_{y|\boldsymbol{x}}\left[\left(y - \bar{y}(\boldsymbol{x})\right)^2\right] + \mathbb{E}_{\mathcal{D}}\left[\left(\bar{y}(\boldsymbol{x}) - f_{\mathcal{D}}(\boldsymbol{x})\right)^2\right]$$

$$\bar{f}(\boldsymbol{x}) = \mathbb{E}_{\mathcal{D}}\left[f_{\mathcal{D}}(\boldsymbol{x})\right]$$

$$\begin{aligned}
&\mathbb{E}_{\mathcal{D}}\left[\left(\bar{y}(\boldsymbol{x}) - f_{\mathcal{D}}(\boldsymbol{x})\right)^2\right] \\
&= \mathbb{E}_{\mathcal{D}}\left[\left(\bar{y}(\boldsymbol{x}) - \bar{f}(\boldsymbol{x}) + \bar{f}(\boldsymbol{x}) - f_{\mathcal{D}}(\boldsymbol{x})\right)^2\right] \\
&= \mathbb{E}_{\mathcal{D}}\left[\left(\bar{y}(\boldsymbol{x}) - \bar{f}(\boldsymbol{x})\right)^2\right] + \mathbb{E}_{\mathcal{D}}\left[\left(\bar{f}(\boldsymbol{x}) - f_{\mathcal{D}}(\boldsymbol{x})\right)^2\right] + 2\mathbb{E}_{\mathcal{D}}\left[\left(\bar{y}(\boldsymbol{x}) - \bar{f}(\boldsymbol{x})\right)\left(\bar{f}(\boldsymbol{x}) - f_{\mathcal{D}}(\boldsymbol{x})\right)\right]
\end{aligned}$$

$\bar{f}(x) - \bar{f}(x) = 0$

https://commons.wikimedia.org/wiki/File:Frustrated_man_at_a_desk_(cropped).jpg

The Bias-Variance decomposition

$$\mathbb{E}_{\mathcal{D},y|\boldsymbol{x}}\left[(y - f_{\mathcal{D}}(\boldsymbol{x}))^2\right] = \mathbb{E}_{y|\boldsymbol{x}}\left[(y - \bar{y}(\boldsymbol{x}))^2\right] + \mathbb{E}_{\mathcal{D}}\left[(\bar{y}(\boldsymbol{x}) - f_{\mathcal{D}}(\boldsymbol{x}))^2\right]$$

$$\bar{f}(\boldsymbol{x}) = \mathbb{E}_{\mathcal{D}}\left[f_{\mathcal{D}}(\boldsymbol{x})\right]$$

$$\begin{aligned}
&\mathbb{E}_{\mathcal{D}}\left[(\bar{y}(\boldsymbol{x}) - f_{\mathcal{D}}(\boldsymbol{x}))^2\right] \\
&= \mathbb{E}_{\mathcal{D}}\left[(\bar{y}(\boldsymbol{x}) - \bar{f}(\boldsymbol{x}) + \bar{f}(\boldsymbol{x}) - f_{\mathcal{D}}(\boldsymbol{x}))^2\right] \\
&= \mathbb{E}_{\mathcal{D}}\left[(\bar{y}(\boldsymbol{x}) - \bar{f}(\boldsymbol{x}))^2\right] + \mathbb{E}_{\mathcal{D}}\left[(\bar{f}(\boldsymbol{x}) - f_{\mathcal{D}}(\boldsymbol{x}))^2\right] + 2\mathbb{E}_{\mathcal{D}}\left[(\bar{y}(\boldsymbol{x}) - \bar{f}(\boldsymbol{x}))(\bar{f}(\boldsymbol{x}) - f_{\mathcal{D}}(\boldsymbol{x}))\right]
\end{aligned}$$

$$\begin{aligned}
&\mathbb{E}_{\mathcal{D},y|\boldsymbol{x}}\left[(y - f_{\mathcal{D}}(\boldsymbol{x}))^2\right] \\
&= \mathbb{E}_{y|\boldsymbol{x}}\left[(y - \bar{y}(\boldsymbol{x}))^2\right] + (\bar{y}(\boldsymbol{x}) - \bar{f}(\boldsymbol{x}))^2 + \mathbb{E}_{\mathcal{D}}\left[(\bar{f}(\boldsymbol{x}) - f_{\mathcal{D}}(\boldsymbol{x}))^2\right] \\
&= \mathrm{Var}_{y|\boldsymbol{x}}\left[y\right] + (\bar{y}(\boldsymbol{x}) - \bar{f}(\boldsymbol{x}))^2 + \mathrm{Var}_{\mathcal{D}}\left[f_{\mathcal{D}}(\boldsymbol{x})\right]
\end{aligned}$$

The Bias-Variance decomposition

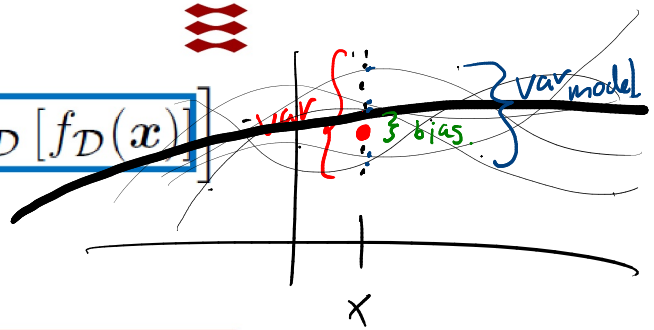
$$\mathbb{E}_{\mathcal{D}}\left[E^{\text{gen}}\right] = \mathbb{E}_{\boldsymbol{x}}\left[\mathbb{E}_{\mathcal{D},y|\boldsymbol{x}}\left[\left(y - f_{\mathcal{D}}(\boldsymbol{x})\right)^2\right]\right]$$

$$\mathbb{E}_{\mathcal{D}}\left[E^{\text{gen}}\right] = \mathbb{E}_{\boldsymbol{x}}\left[\text{Var}_{y|\boldsymbol{x}}\left[y\right] + \left(\bar{y}(\boldsymbol{x}) - \bar{f}(\boldsymbol{x})\right)^2 + \text{Var}_{\mathcal{D}}\left[f_{\mathcal{D}}(\boldsymbol{x})\right]\right]$$

https://commons.wikimedia.org/wiki/File:Frustrated_man_at_a_desk_(cropped).jpg

The Bias-Variance decomposition

$$\mathbb{E}_{\mathcal{D}}\left[E^{\text{gen}}\right] = \mathbb{E}_{\boldsymbol{x}}\left[\text{Var}_{y|\boldsymbol{x}}\left[y\right] + \left(\bar{y}(\boldsymbol{x}) - \bar{f}(\boldsymbol{x})\right)^2 + \text{Var}_{\mathcal{D}}\left[f_{\mathcal{D}}(\boldsymbol{x})\right]\right]$$

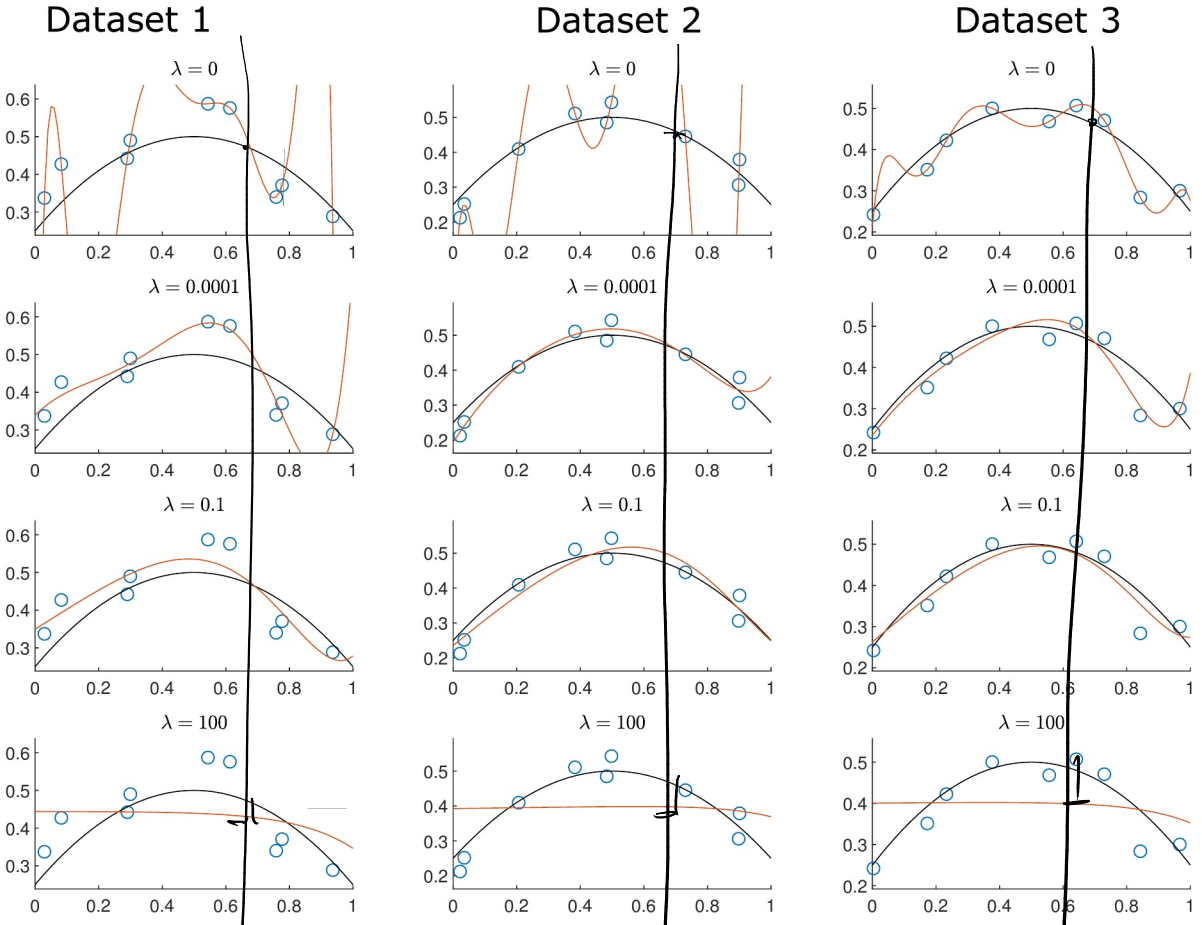


The first term does not depend at all upon our choice of model but simply represents the intrinsic difficulty of the problem. We cannot make this term any larger or smaller by selecting one model over another.

The second term is the **bias** term. It tells us how much the average values of models trained on different training datasets differ compared to the true mean of the data.

The third term is the **variance** term. It tells us how much the model wiggles when trained on different sets of training data. That is, when you train the models on N different (random) sets of training data and the models (the prediction curves) are nearly the same this term is small.

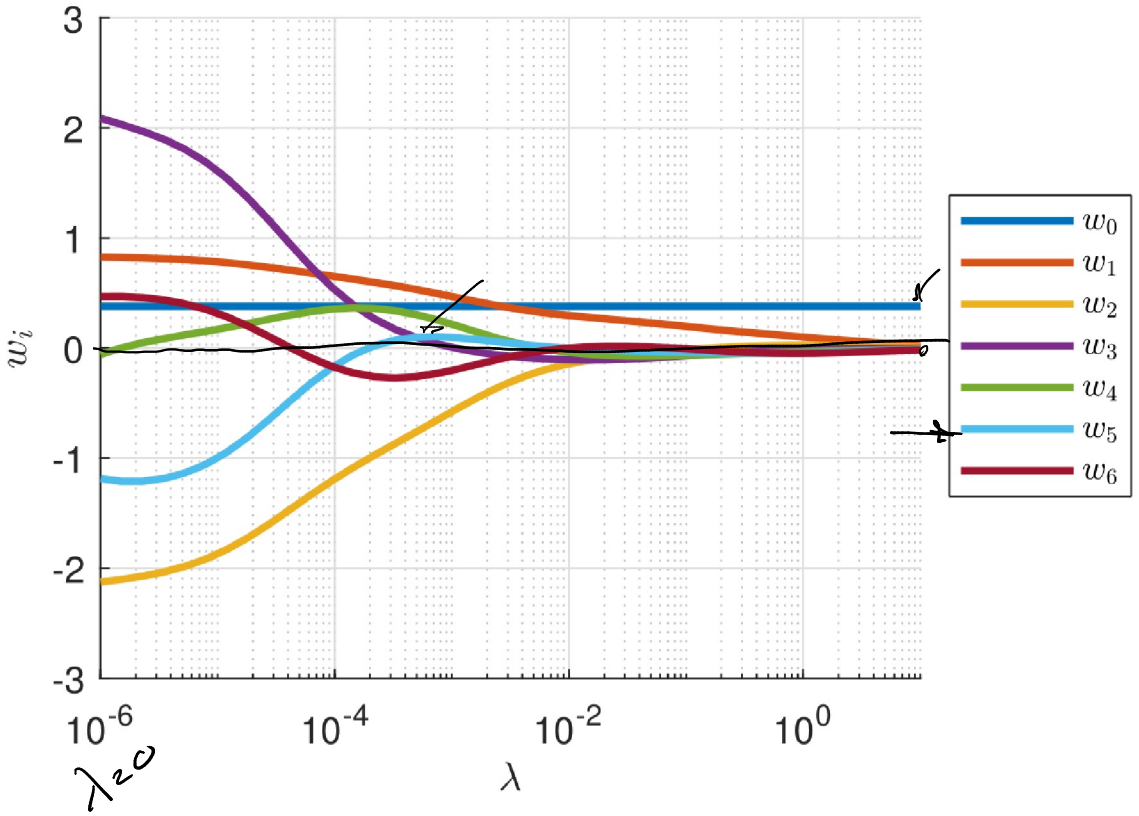
The bias variance decomposition



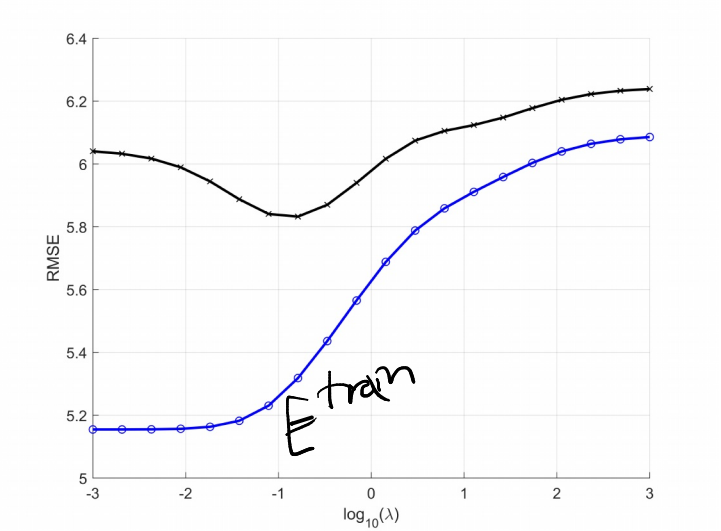
By regularization we can tradeoff bias and variance, in particular, we can hope to substantially reduce variance without introducing too much bias!

Parameters w^* as function of λ

$$E_\lambda(\boldsymbol{w}) = \sum_{i=1}^{N} (\hat{y}_i - w_0 - \hat{\boldsymbol{x}}_i{}^\top \boldsymbol{w})^2 + \lambda \|\boldsymbol{w}\|^2$$



Quiz 1, Bias-variance (Fall 2017)



Using 54 observations of a dataset about Basketball, we would like to predict the average points scored per game (y) based on the four features. For this purpose we consider regularized least squares regression which minimizes with respect to $\boldsymbol{w}$ the following cost function:

$$E(\boldsymbol{w}) = \sum_n (y_n - [1 \; x_{n1} \; x_{n2} \; x_{n3} \; x_{n4}]\boldsymbol{w})^2 + \lambda \boldsymbol{w}^\top \boldsymbol{w},$$

We consider 20 different values of λ and use leave-one-out cross-validation to estimate the performance (measured by mean-squared error) of each of these different values of λ and plot the result in the figure. For the value of $\lambda = 0.6952$ the following model is identified:

$$f(\boldsymbol{x}) = 2.76 - 0.37x_1 + 0.01x_2 + 7.67x_3 + 7.67x_4.$$

Which one of the following statements is correct?

A. In the figure the blue curve with circles corresponds to the training error whereas the black curve with crosses corresponds to the test error.

B. According to the model defined for $\lambda = 0.6952$ increasing a players height x_1 will increase his average points scored per game.

C. There is no optimal way of choosing λ since increasing λ reduces the variance but increases the bias.

D. As we increase λ the 2-norm of the weight vector $\boldsymbol{w}$ will also increase.

E. Don't know.

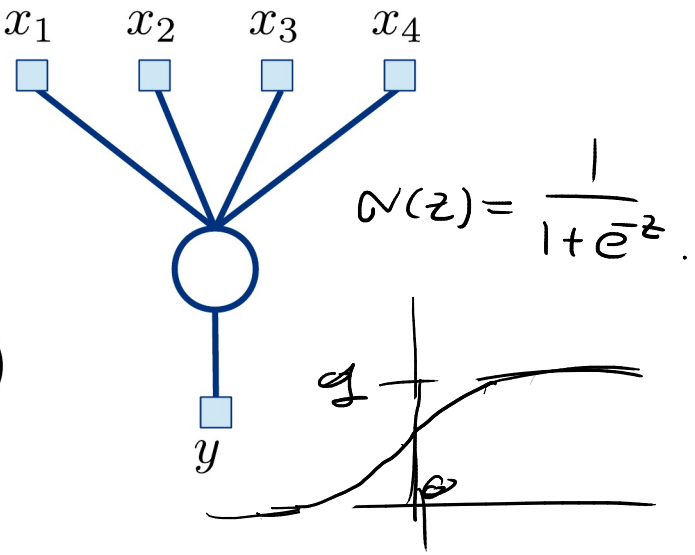
The correct answer is A: The blue curve monotonically increases with λ reflecting a worse fit to the training set as we increase λ using regularization we can reduce the variance by introducing bias and the black curve indicates that an optimal tradeoff at around $10^{-0.8}$ as reflected by the test error indicated in the black curve being minimal. As we increase λ we will penalize the weights according to the squared 2-norm more and more and thus the 2-norm will be reduced. Finally, according to the fitted model we observe that the coefficient in front of x_1 (Height) is negative thus indicating that an increase in height will reduce the models prediction of average points scored per game.

General linear model

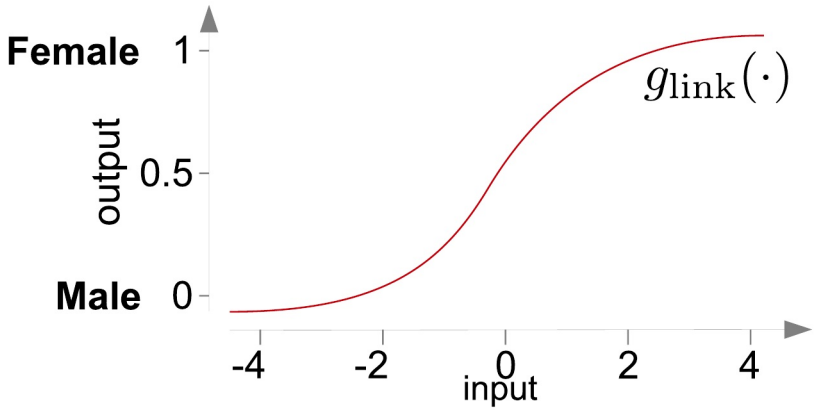
$g_{link}(z) = z$, $d(y, \hat{y}) = \|y - \hat{y}\|^2$

$g_{link}(z) = \sigma(z)$, $d(y, \hat{y}) = -\left(y \log \hat{y} + (1-y)\log(1-\hat{y})\right)$

- **Remember the generalized linear model?**
  - Data $\{\boldsymbol{x}_n, y_n\}_{n=1}^{N}$
  - Model $f(\boldsymbol{x}) = g_{\text{link}}(\boldsymbol{x}^\top \boldsymbol{w})$
  - Cost function $d\big(y, f(\boldsymbol{x})\big)$
  - Parameters $\boldsymbol{w} = \arg\min_{\boldsymbol{w}} \sum_{n=1}^{N} d\big(y_n, f(\boldsymbol{x}_n)\big)$



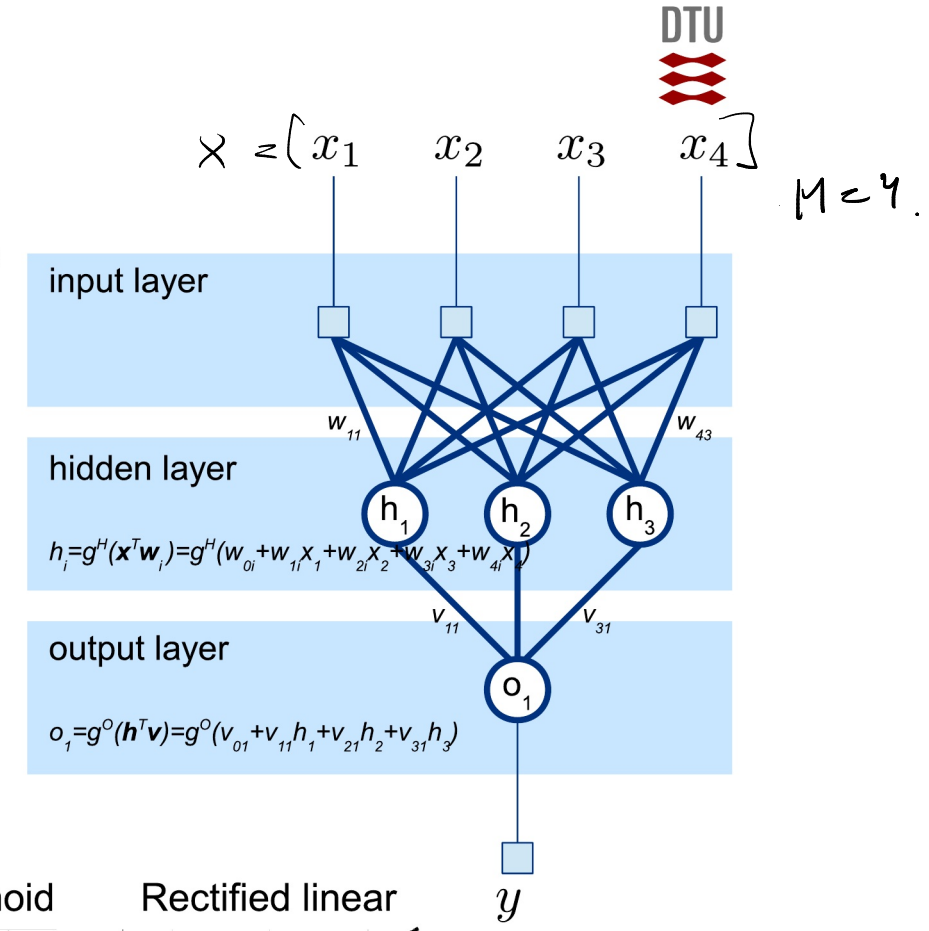
$\sigma(z) = \dfrac{1}{1+e^{-z}}$.



Artificial neural networks

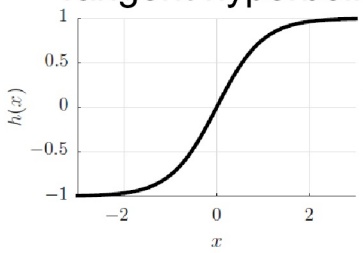
Feed forward network

- Each "neuron"
  - Computes a non-linear function of the sum of its inputs
  - Is just like a generalized linear model
  - Has its own set of parameters
- Modeling choices
  - Cost function
  - Non-linearities
  - Number of neurons and hidden layers
  - Selection of inputs
- Parameter estimation using numerical optimization methods
- Very flexible model: Can easily overfit

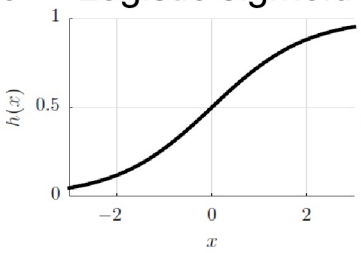


Example of non-linearities:

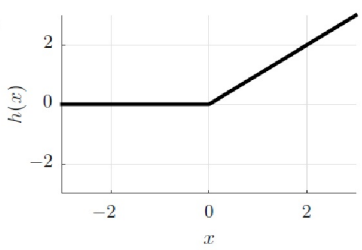
Tangent hyperbolic



Logistic sigmoid



Rectified linear

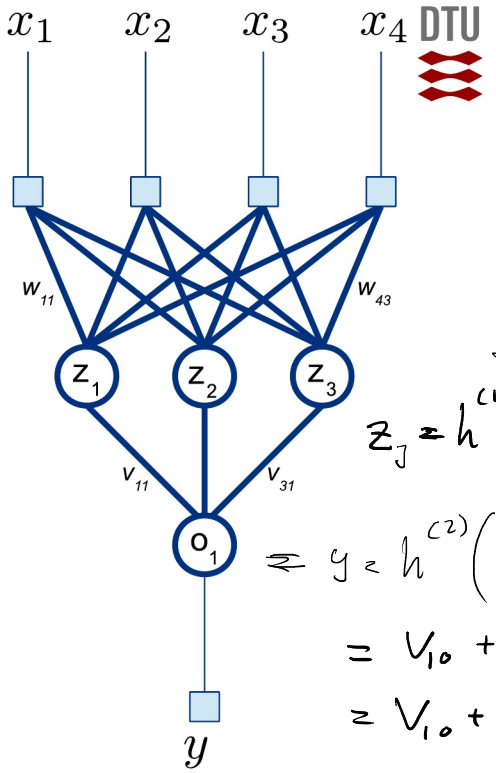


Data: $\{\boldsymbol{x}_i, y_i\}$

Model: $f(\boldsymbol{x}) = h^{(2)}\left(v_{10} + \sum_{j=1}^{H} v_{1j} h^{(1)}\left(\tilde{\boldsymbol{x}}^\top \boldsymbol{w}_j\right)\right)$

Distance: $d(y, f(\boldsymbol{x}))$

Cost: $E = \sum_{i=1}^{N} d(y_i, f(\boldsymbol{x}_i))$



$z_j = h^{(1)}(\tilde{x}^T w_j)$

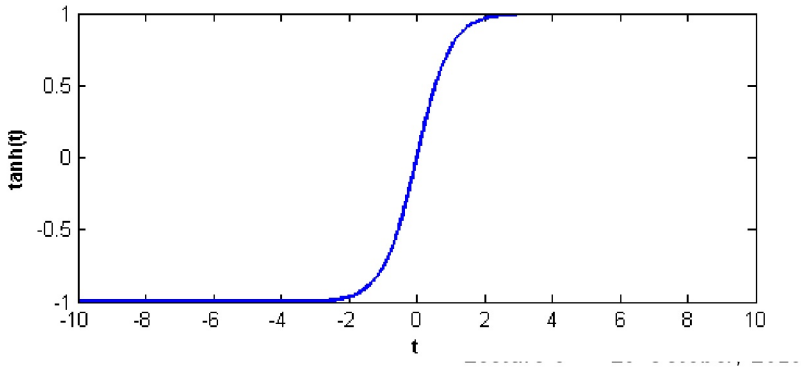
$y = h^{(2)}(\tilde{z}^T v_1)$

$= v_{10} + z^T v_{1[1:3]}$

$= v_{10} + \sum_{j=1}^{H=3} v_{1j} h^{(2)}(\tilde{x}^T w_j)$

Common choices

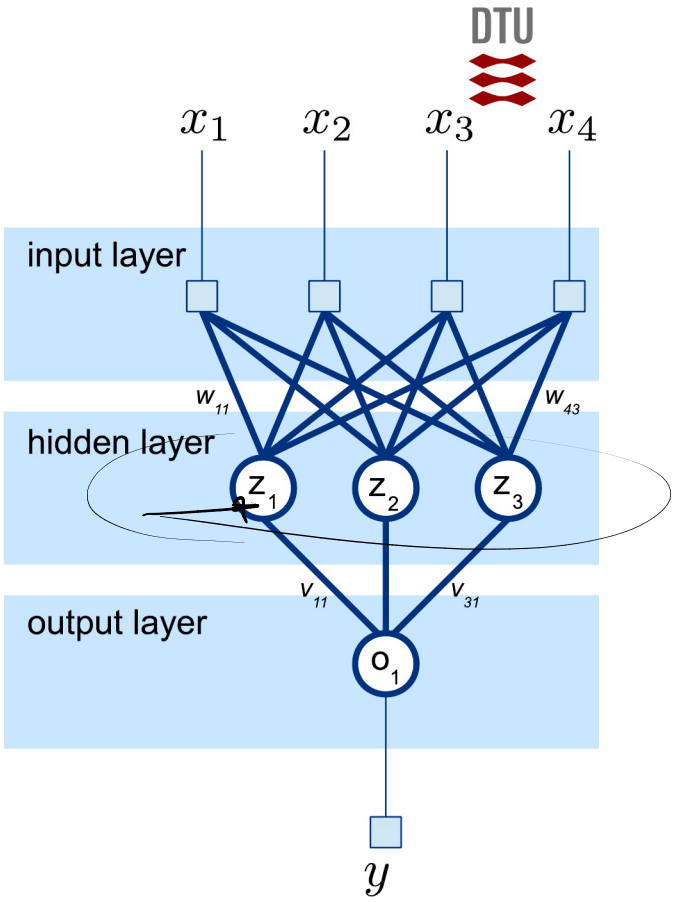
$$h^{(1)}(x) = \tanh(x)$$
$$h^{(2)}(x) = x$$
$$d(y, f(\boldsymbol{x})) = (y - f(\boldsymbol{x}))^2$$



Neurons and layers

Recall:

$$f(\boldsymbol{x}) = h^{(2)}\left(v_{10} + \sum_{j=1}^{H} v_{1j} h^{(1)}\left(\tilde{\boldsymbol{x}}^\top \boldsymbol{w}_j\right)\right)$$



- Let $z_j^{(1)}$ be output of j'th hidden unit

$$z_j^{(1)} = h^{(1)}\left(\boldsymbol{w}_j^{(1)\top} \tilde{x}\right)$$

Abbreviated $\boldsymbol{z}^{(1)} = h^{(1)}\left(\boldsymbol{W}^{(1)} \tilde{x}\right)$

- Output

$$\begin{bmatrix} z_1^{(1)} \\ \vdots \\ z_H^{(1)} \end{bmatrix} = h^{(1)}\left(\begin{bmatrix} (w_1^{(1)})^\top \\ \vdots \\ (w_H^{(1)})^\top \end{bmatrix} \tilde{x}\right)$$

$$f(\boldsymbol{x}) = h^{(2)}\left(v_{10} + \sum_{j=1}^{H} v_{1j} z_j^{(1)}\right) = h^{(2)}\left(\boldsymbol{W}^{(2)} \tilde{\boldsymbol{z}}^{(1)}\right)$$

$$W^{(2)} = \begin{bmatrix} v_{10} & v_{11} & v_{12} & \cdots & v_{1H} \end{bmatrix}$$

We consider each $z_j^{(1)}$ a neuron and $\boldsymbol{z}^{(1)}$ a (hidden) layer

Quiz 2, Artificial Neural Network (Fall 2017)

We will consider an artificial neural network (ANN) trained to predict the average score of a player (i.e., y). The ANN is based on the model:

$$f(\boldsymbol{x}, \boldsymbol{w}) = w_0^{(2)} + \sum_{j=1}^{2} w_j^{(2)} h^{(1)}([1\ \boldsymbol{x}]\boldsymbol{w}_j^{(1)}).$$

where $h^{(1)}(x) = \max(x, 0)$ is the rectified linear function used as activation function in the hidden layer (i.e., positive values are returned and negative values are set to zero). We will consider an ANN with two hidden units in the hidden layer defined by:

$$\boldsymbol{w}_1^{(1)} = \begin{bmatrix} 21.78 \\ -1.65 \\ 0 \\ -13.26 \\ -8.46 \end{bmatrix}, \boldsymbol{w}_2^{(1)} = \begin{bmatrix} -9.60 \\ -0.44 \\ 0.01 \\ 14.54 \\ 9.50 \end{bmatrix},$$

and $w_0^{(2)} = 2.84$, $w_1^{(2)} = 3.25$, and $w_2^{(2)} = 3.46$.
What is the predicted average score of a basketball player with observation vector $\boldsymbol{x}^* = [6.8\ 225\ 0.44\ 0.68]$?

A. 1.00

B. 3.74

C. 8.21

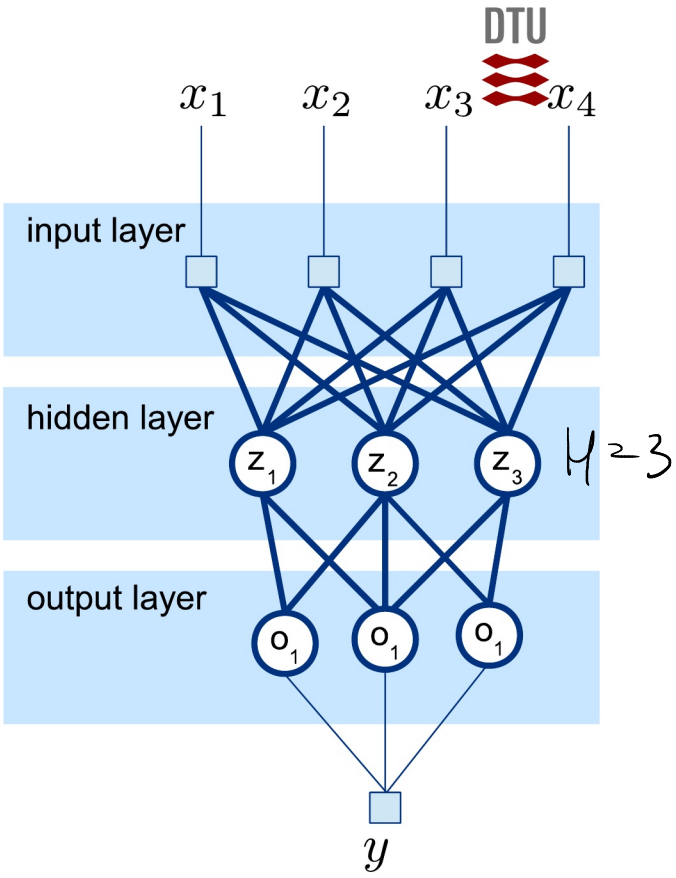
D. 11.54

E. Don't know.

The output is given by:

$$
\begin{aligned}
f(\boldsymbol{x}, \boldsymbol{w}) &= 2.84 \\
&\quad + 3.25 \cdot \max([1\ 6.8\ 225\ 0.44\ 0.68] \cdot \begin{bmatrix} 21.78 \\ -1.65 \\ 0 \\ -13.26 \\ -8.46 \end{bmatrix}, 0) \\
&\quad + 3.46 \max([1\ 6.8\ 225\ 0.44\ 0.68] \cdot \begin{bmatrix} -9.60 \\ -0.44 \\ 0.01 \\ 14.54 \\ 9.50 \end{bmatrix}, 0) \\
&= 2.84 + 3.25 \cdot \max(-1.027, 0) + 3.46 \max(2.516, 0) \\
&= 11.54
\end{aligned}
$$

Generalization 1: Multiple outputs



- As before define: $\boldsymbol{z}^{(1)} = h^{(1)}\left(\boldsymbol{W}^{(1)}\tilde{x}\right)$
- Now let $\boldsymbol{W}^{(2)}$ be a $C \times H$ matrix then:

  $W^{(2)}: 1 \times H$ $\quad \boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x}) = h^{(2)}\left(\boldsymbol{W}^{(2)}\tilde{\boldsymbol{z}}^{(1)}\right)$ $\quad z^{(2)}$

  $y: \begin{bmatrix} \\ \\ \end{bmatrix} C.$

  will be C-dimensional
- Re-define error function

$$E = \sum_{i=1}^{N} \|\boldsymbol{y}_i - \boldsymbol{f}(\boldsymbol{x}_i)\|_2^2$$

$(x, y) \sim \left(\begin{bmatrix} x_1 \\ \vdots \\ x_M \end{bmatrix}, \begin{bmatrix} y_1 \\ \vdots \\ y_C \end{bmatrix}\right)$

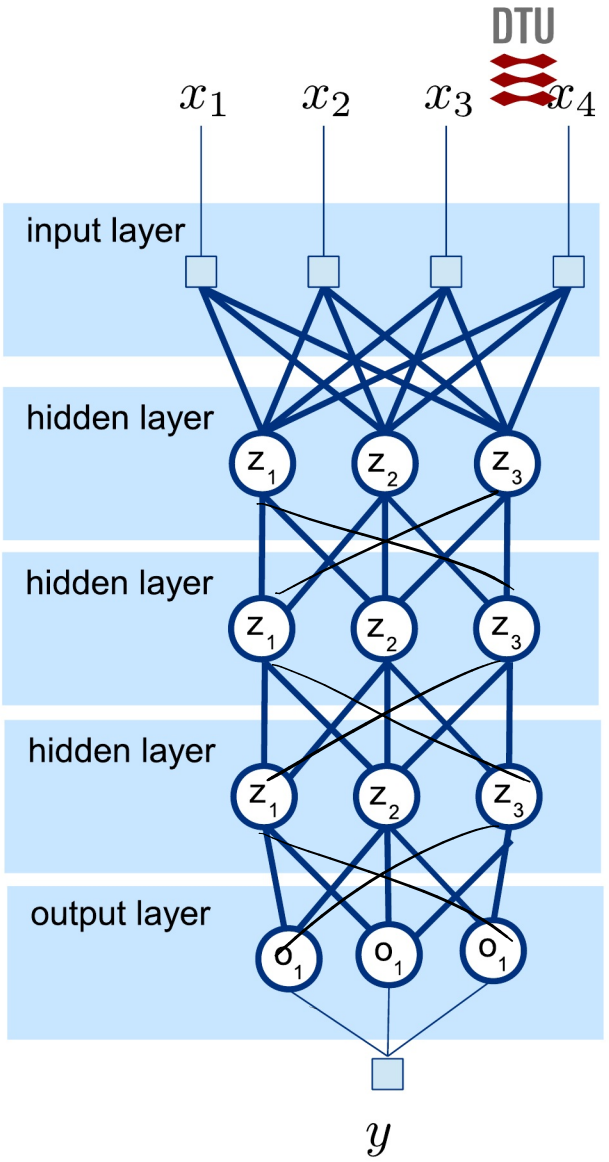
Generalization 2: Multiple layers

- Define $\boldsymbol{z}^{(0)} = \boldsymbol{x}$
- For each layer $l = 1, \ldots, L$ compute

$$z_j^{(l)} = h^{(l)}\left(\boldsymbol{W}^{(l)}\tilde{\boldsymbol{z}}^{(l-1)}\right)$$

- Output is simply

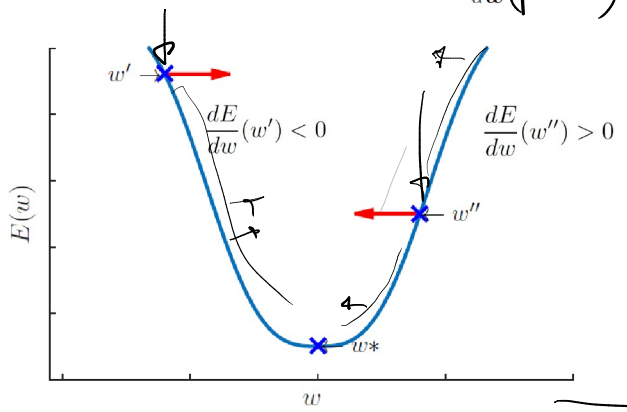
$$\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{z}^{(L)}$$

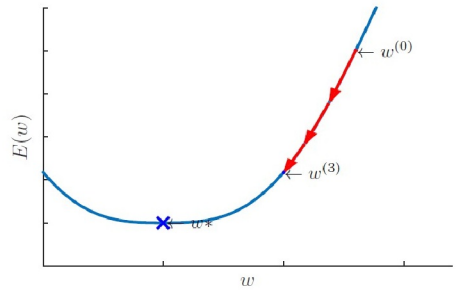


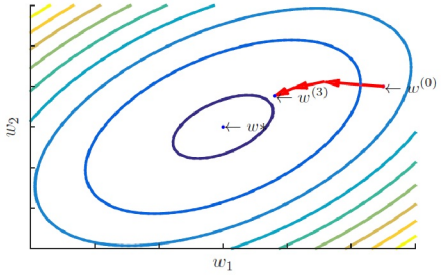
Gradient descent

- Start from an initial guess at $\boldsymbol{w}^*$, $\boldsymbol{w}^{(0)}$
- At step t of the algorithm, modify $\boldsymbol{w}^{(t-1)}$ to produce a better guess $\boldsymbol{w}^{(t)}$:

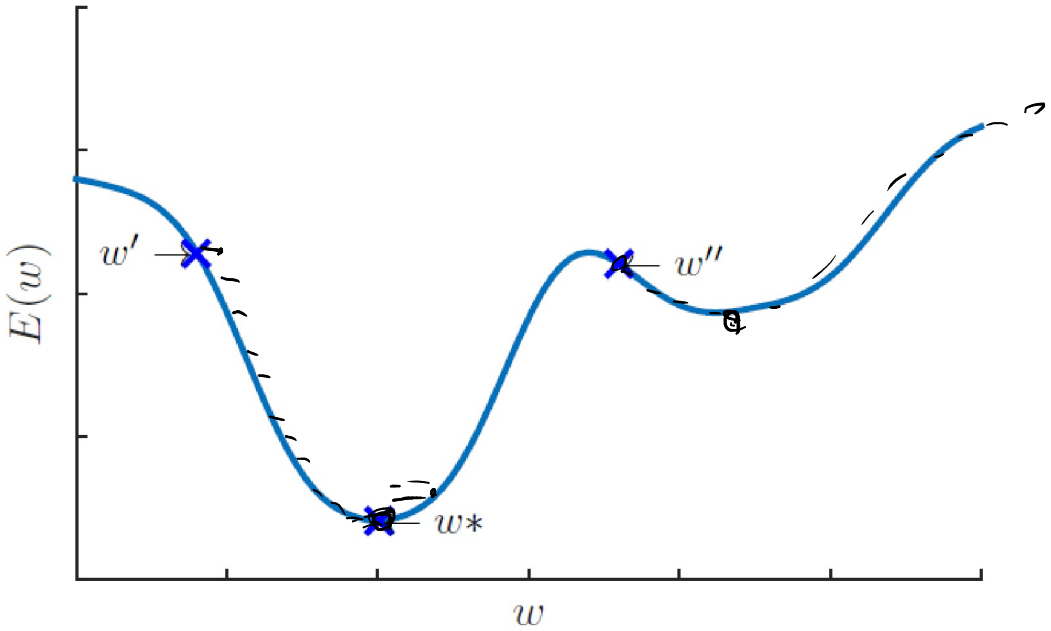
$$\boldsymbol{w}^{(t)} = \boldsymbol{w}^{(t-1)} - \epsilon \frac{dE}{d\boldsymbol{w}}\left(\boldsymbol{w}^{(t-1)}\right)$$





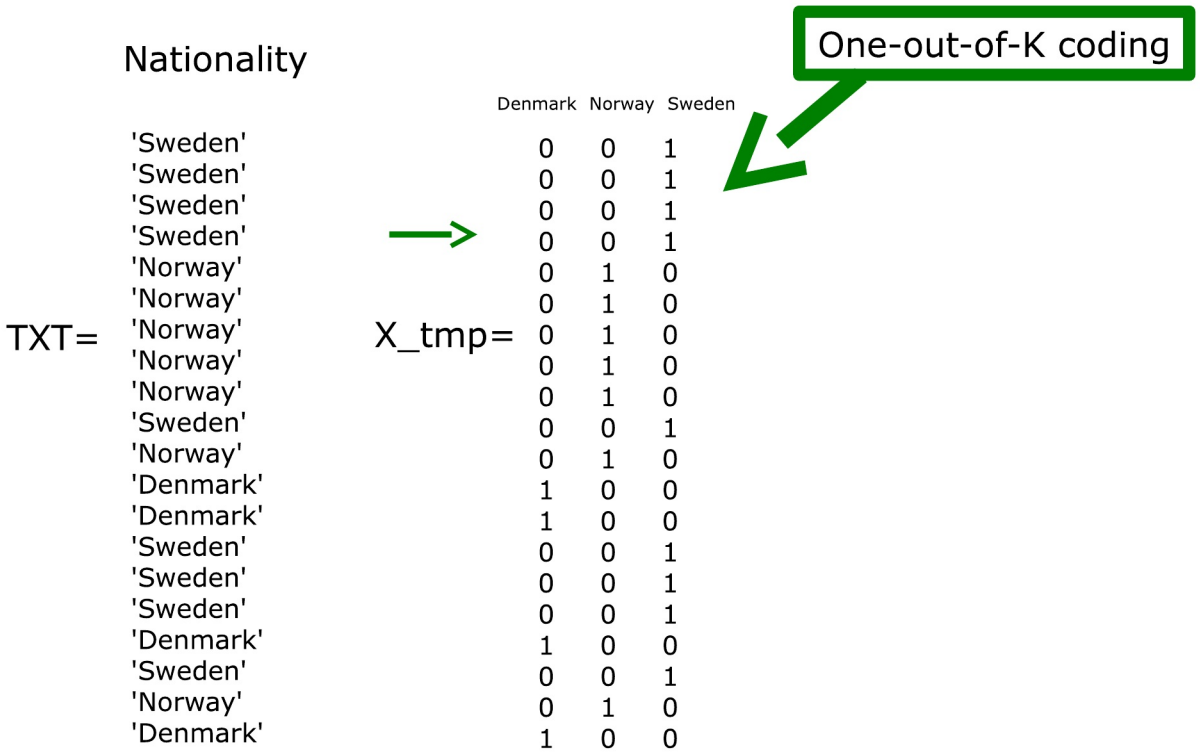


Contrary to least-squares linear regression and logistic regression ANNs have issues of local minima



Single and multi-class: One out of K coding

$y = 1, \ldots, K$



Multi-class classification

- Logistic regression, $y = 0, 1$:

$$p(y|\theta) = \theta^y (1-\theta)^{1-y}$$

$$\theta = \sigma(\boldsymbol{x}^\top \boldsymbol{w})$$

- Multinomial regression, $y = 1, 2, \ldots, K$

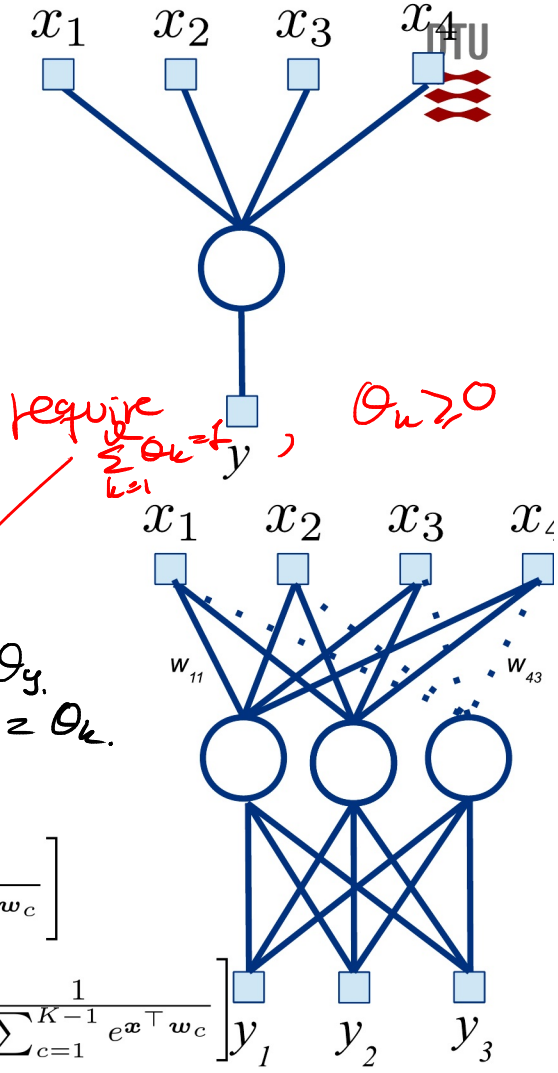
z_k : one-of-K encoding of y,

$$p(y|\boldsymbol{\theta}) = \prod_{i=1}^{K} \theta_k^{z_k}$$

$$\boldsymbol{\theta} = \text{softmax}\left(\begin{bmatrix}\boldsymbol{x}^\top \boldsymbol{w}_1 & \cdots & \boldsymbol{x}^\top \boldsymbol{w}_K\end{bmatrix}\right)$$

$$= \begin{bmatrix} \frac{e^{\boldsymbol{x}^\top \boldsymbol{w}_1}}{\sum_{c=1}^{K} e^{\boldsymbol{x}^\top \boldsymbol{w}_c}} & \cdots & \frac{e^{\boldsymbol{x}^\top \boldsymbol{w}_{K-1}}}{\sum_{c=1}^{K} e^{\boldsymbol{x}^\top \boldsymbol{w}_c}} & \frac{e^{\boldsymbol{x}^\top \boldsymbol{w}_K}}{\sum_{c=1}^{K} e^{\boldsymbol{x}^\top \boldsymbol{w}_c}} \end{bmatrix}$$

or: $$\boldsymbol{\theta} = \begin{bmatrix} \frac{e^{\boldsymbol{x}^\top \boldsymbol{w}_1}}{1+\sum_{c=1}^{K-1} e^{\boldsymbol{x}^\top \boldsymbol{w}_c}} & \cdots & \frac{e^{\boldsymbol{x}^\top \boldsymbol{w}_{K-1}}}{1+\sum_{c=1}^{K-1} e^{\boldsymbol{x}^\top \boldsymbol{w}_c}} & \frac{1}{1+\sum_{c=1}^{K-1} e^{\boldsymbol{x}^\top \boldsymbol{w}_c}} \end{bmatrix}$$



Connection to neural networks

Multinomial regression:

- Define:

$$\boldsymbol{\theta} = \begin{bmatrix} \frac{e^{\boldsymbol{x}^\top \boldsymbol{w}_1}}{1+\sum_{c=1}^{K-1} e^{\boldsymbol{x}^\top \boldsymbol{w}_c}} & \cdots & \frac{1}{1+\sum_{c=1}^{K-1} e^{\boldsymbol{x}^\top \boldsymbol{w}_c}} \end{bmatrix}$$

- Cost function is (z_i. is one-of-K encoding of y_i)

$$E = -\sum_{i=1}^{N} \log p(y_i|\boldsymbol{x}_i) = -\sum_{i=1}^{N}\sum_{c=1}^{K} z_{ic} \log \theta_{ic}$$

Multi-class neural network:

- Suppose $\tilde{y}_1, \ldots, \tilde{y}_K$ are outputs of a neural network
- Define

$$\boldsymbol{\theta} = \begin{bmatrix} \frac{e^{\tilde{\boldsymbol{y}}^\top \boldsymbol{w}_1}}{\sum_{c=1}^{K} e^{\tilde{\boldsymbol{y}}^\top \boldsymbol{w}_c}} & \cdots & \frac{e^{\tilde{\boldsymbol{y}}^\top \boldsymbol{w}_K}}{\sum_{c=1}^{K} e^{\tilde{\boldsymbol{y}}^\top \boldsymbol{w}_c}} \end{bmatrix}$$

- Cost function is:

$$E = -\sum_{i=1}^{N} \log p(y_i|\tilde{\boldsymbol{y}}_i) = -\sum_{i=1}^{N}\sum_{c=1}^{K} z_{ic} \log \theta_{ic}$$

Quiz 3, Multinomial Regression (Spring 2016)

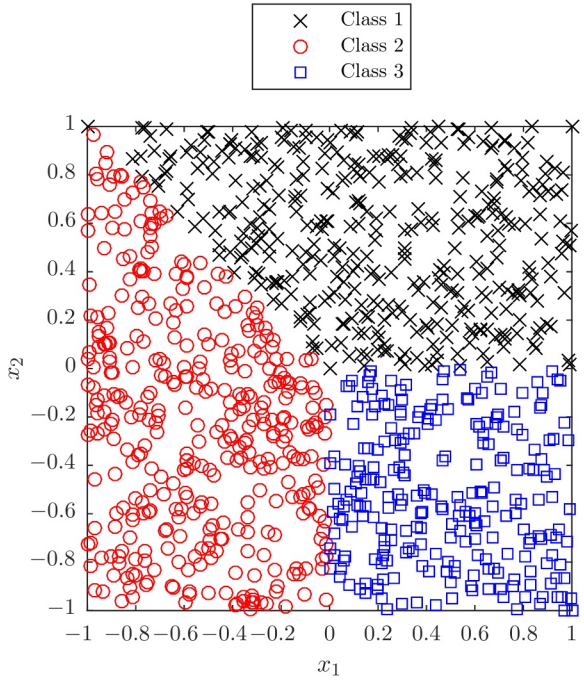


Figure 1: Observations labelled with the most probable class

Consider a multinomial regression classifier for a three-class problem where for each point $\boldsymbol{x} = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^\top$ we compute the class-probability using the softmax function

$$P(\hat{y} = k) = \frac{e^{\boldsymbol{w}_k^\top \boldsymbol{x}}}{e^{\boldsymbol{w}_1^\top \boldsymbol{x}} + e^{\boldsymbol{w}_2^\top \boldsymbol{x}} + e^{\boldsymbol{w}_3^\top \boldsymbol{x}}}.$$

A dataset of $N = 1000$ points where each point is labeled according to the maximum class-probability is shown in Figure 1. Which setting of the weights was used?

A. $w_1 = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$, $w_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$, $w_3 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$

B. $w_1 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$, $w_2 = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$, $w_3 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$

C. $w_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$, $w_2 = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$, $w_3 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$

D. $w_1 = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$, $w_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$, $w_3 = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$

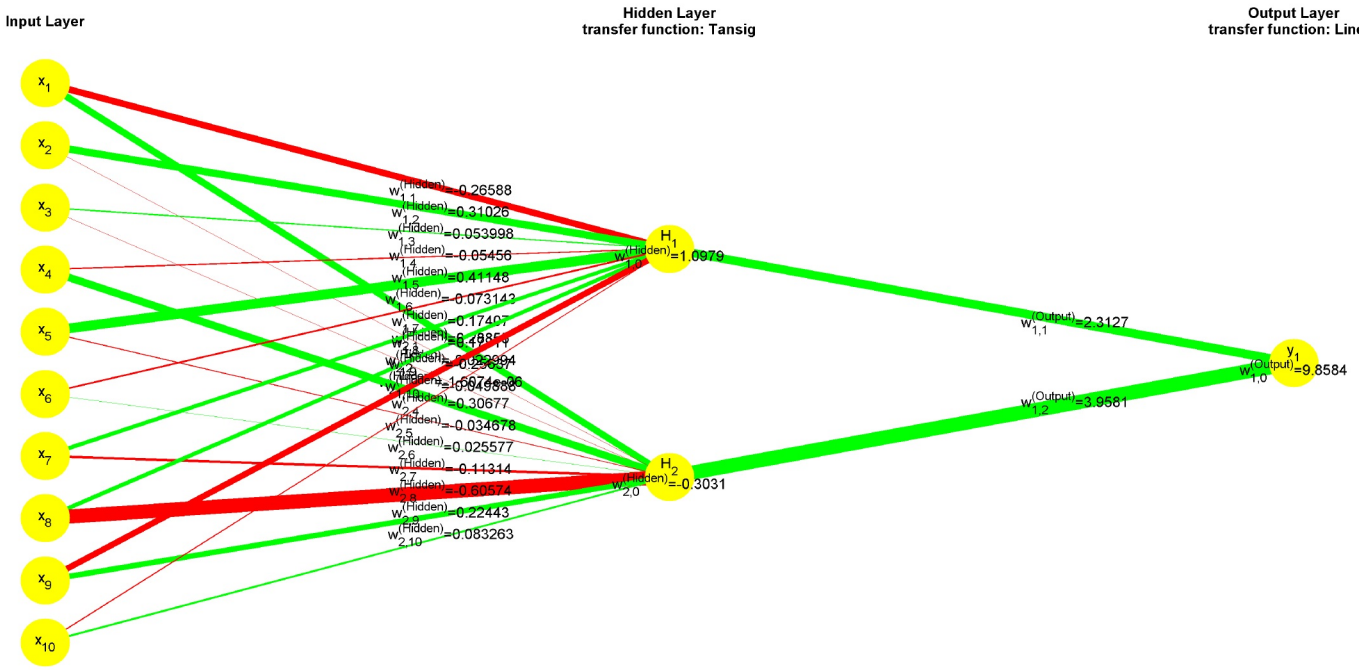
E. Don't know.

Consider for instance the point $\boldsymbol{x}$ where $x_1 = 0$ and $x_2 = 1$. Then, letting $y_k = \boldsymbol{w}_k^T \boldsymbol{x}$, we obtain:

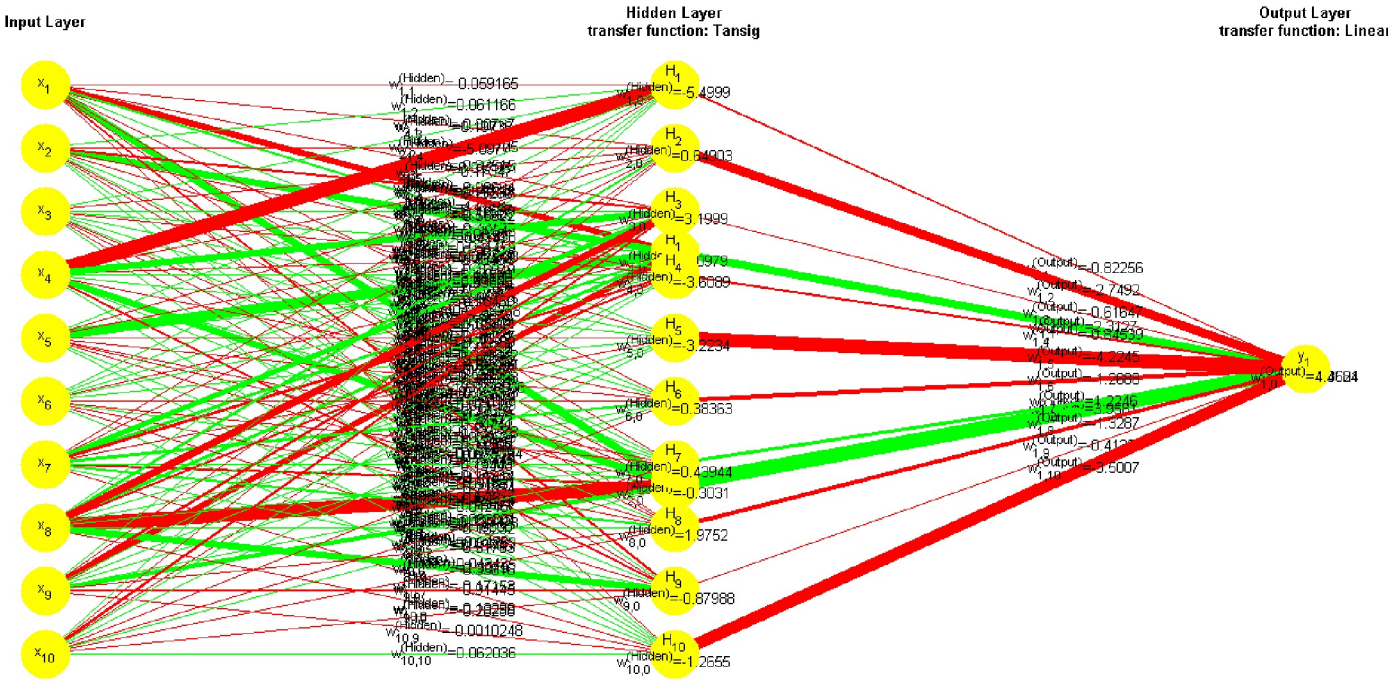
$$
\begin{aligned}
A : \begin{bmatrix} y_1 & y_2 & y_3 \end{bmatrix} &= \begin{bmatrix} -1 & 1 & -1 \end{bmatrix} \\
B : \begin{bmatrix} y_1 & y_2 & y_3 \end{bmatrix} &= \begin{bmatrix} -1 & -1 & 1 \end{bmatrix} \\
C : \begin{bmatrix} y_1 & y_2 & y_3 \end{bmatrix} &= \begin{bmatrix} 1 & -1 & -1 \end{bmatrix} \\
D : \begin{bmatrix} y_1 & y_2 & y_3 \end{bmatrix} &= \begin{bmatrix} -1 & 1 & 1 \end{bmatrix}
\end{aligned}
$$

Next, since the multinomial regression function preserves order we need only consider the maximal value. Accordingly the point $\boldsymbol{x}$ is only classified to the correct class 1 for option C.

Interpreting neural networks can be difficult



Interpreting neural networks can be difficult



Resources

https://www.youtube.com Exellent video resource explaining the concepts behind neural networks (https://www.youtube.com/watch?v=aircAruvnKk&list=PLZHQObOWTQDNU6R1_67000Dx_ZCJB-3pi)

http://playground.tensorflow.org Sleek interactive neural network example where you can examine the effect of different number of hidden neurons, activation functions, and many other things on training (http://playground.tensorflow.org/)

https://www.tensorflow.org Most popular and well-documented deep learning framework. While well documented, notice it requires some python knowledge (https://www.tensorflow.org/)

https://pytorch.org Upcoming (and in some ways slightly simpler) framework for deep learning; alternative to tensorflow (https://pytorch.org/)