

Deep Learning: Lecture 2

Alexander Schönhuth

UU

October 9, 2019

Supervised Learning

SUPERVISED LEARNING

- ▶ There is a functional relationship

$$f^* : \mathbb{R}^d \rightarrow V$$

we would like to understand, or *learn*.

- ▶ *Regression*: $V = \mathbb{R}$
- ▶ *Classification*: $V = \{1, \dots, k\}$
- ▶ To learn it, we are given m *data points*

$$(x_i, f^*(x_i) = y_i)_{i=1, \dots, m}$$

that reflect this functional relationship.

Final goal: Predict $f^*(x)$ well on unknown data points x .

SUPERVISED LEARNING

- ▶ The idea is to set up a *training procedure* (an algorithm) that *learns* f^* from the training data.
- ▶ Learning f^* means to *approximate* it by $f : \mathbb{R}^d \rightarrow V$ sufficiently well, where $f \in \mathcal{M}$ for a certain class of functions \mathcal{M} .
- ▶ In most cases, $f \in \mathcal{M}$ are parameterized by parameters w . This means that we have to pick an appropriate choice of parameters w for learning f^* .

SUPERVISED LEARNING

- ▶ The idea is to set up a *training procedure* (an algorithm) that *learns* f^* from the training data.
- ▶ Learning f^* means to *approximate* it by $f : \mathbb{R}^d \rightarrow V$ sufficiently well, where $f \in \mathcal{M}$ for a certain class of functions \mathcal{M} .
- ▶ In most cases, $f \in \mathcal{M}$ are parameterized by parameters \mathbf{w} . This means that we have to pick an appropriate choice of parameters \mathbf{w} for learning f^* .

SUPERVISED LEARNING

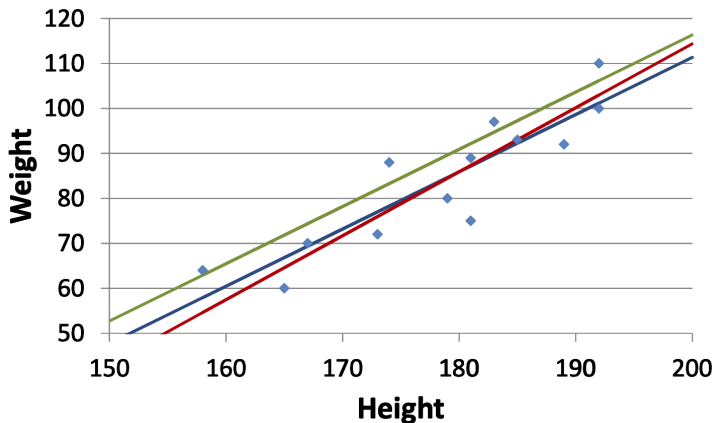
- ▶ We need to determine a *cost (or loss) function* C where $C(f, f^*)$ measures how well $f \in \mathcal{M}$ approximates f^* .
- ▶ *Optimization*: Pick $f \in \mathcal{M}$ (by picking the right set of parameters) that yields small (possibly minimal) cost $C(f, f^*)$
- ▶ *Generalization*: Optimization procedure should address that f is to approximate f^* well on *unknown data points*.

SUPERVISED LEARNING

- ▶ We need to determine a *cost (or loss) function* C where $C(f, f^*)$ measures how well $f \in \mathcal{M}$ approximates f^* .
- ▶ *Optimization*: Pick $f \in \mathcal{M}$ (by picking the right set of parameters) that yields small (possibly minimal) cost $C(f, f^*)$
- ▶ *Generalization*: Optimization procedure should address that f is to approximate f^* well on *unknown data points*.

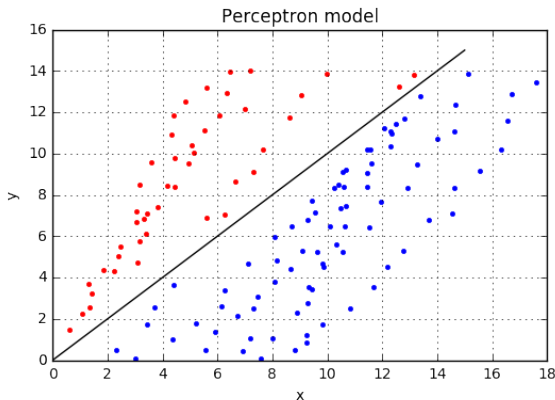
LINEAR REGRESSION

EXAMPLE: $f : \mathbb{R} \rightarrow \mathbb{R}$



PERCEPTRON

EXAMPLE: $f : \mathbb{R}^2 \rightarrow \{0, 1\}$



$$\begin{aligned} f : \mathbb{R}^2 &\longrightarrow \{0 = \text{blue}, 1 = \text{red}\} \\ (x_1, x_2) &\mapsto \begin{cases} 1 & x_2 - x_1 > 0 \\ 0 & x_2 - x_1 \leq 0 \end{cases} \end{aligned} \quad (1)$$

SUPERVISED LEARNING

SUMMARY

We need to specify:

- ▶ How to set up the data being used for training
- ▶ A model class \mathcal{M} , for example linear functions
- ▶ A cost function $C(f, f^*)$ that evaluates the goodness of $f \in \mathcal{M}$
- ▶ An optimization procedure that picks f such that $C(f, f^*)$ is minimal, or very small
- ▶ Keep in mind that f is to perform well on previously unseen data

SUPERVISED LEARNING

SUMMARY

We need to specify:

- ▶ How to set up the data being used for training
- ▶ A model class \mathcal{M} , for example linear functions
- ▶ A cost function $C(f, f^*)$ that evaluates the goodness of $f \in \mathcal{M}$
- ▶ An optimization procedure that picks f such that $C(f, f^*)$ is minimal, or very small
- ▶ Keep in mind that f is to perform well on previously unseen data

SUPERVISED LEARNING

NOTATION

- ▶ The dataset is given by a *design matrix* $\mathbf{X} \in \mathbb{R}^{m \times d}$ where m is the number of data points and d is the number of *features*
- ▶ Each data point x_i (a row in \mathbf{X}) is assigned to a *label* y_i that reflects the true functional relationship $y_i = f^*(x_i)$, where further $\mathbf{y} = (y_1, \dots, y_m) \in V^m$ is the *label vector*.

Generalization

ENABLING GENERALIZATION: DATA

TRAINING, TEST AND VALIDATION

- ▶ Split (\mathbf{X}, \mathbf{y}) into
 - ▶ training data $(\mathbf{X}^{(\text{train})}, \mathbf{y}^{(\text{train})})$
 - ▶ validation data $(\mathbf{X}^{(\text{val})}, \mathbf{y}^{(\text{val})})$
 - ▶ test data $(\mathbf{X}^{(\text{test})}, \mathbf{y}^{(\text{test})})$
- ▶ While *training data* is to pick the optimal set of parameters (which specify elements from \mathcal{M}), using training and *validation data* in combination is for picking *hyperparameters*
- ▶ Hyperparameters can refer to choosing subsets of \mathcal{M} . For example, depth of a neural network, and widths of hidden layers. They may also refer to specifications of cost function or optimization procedure.
- ▶ $(\mathbf{X}^{(\text{test})}, \mathbf{y}^{(\text{test})})$ are never touched during training.
- ▶ The final goal is to minimize the cost on the test data.

ENABLING GENERALIZATION: DATA

TRAINING, TEST AND VALIDATION

- ▶ Split (\mathbf{X}, \mathbf{y}) into
 - ▶ training data $(\mathbf{X}^{(\text{train})}, \mathbf{y}^{(\text{train})})$
 - ▶ validation data $(\mathbf{X}^{(\text{val})}, \mathbf{y}^{(\text{val})})$
 - ▶ test data $(\mathbf{X}^{(\text{test})}, \mathbf{y}^{(\text{test})})$
- ▶ While *training data* is to pick the optimal set of parameters (which specify elements from \mathcal{M}), using training and *validation data* in combination is for picking *hyperparameters*
- ▶ Hyperparameters can refer to choosing subsets of \mathcal{M} . For example, depth of a neural network, and widths of hidden layers. They may also refer to specifications of cost function or optimization procedure.
- ▶ $(\mathbf{X}^{(\text{test})}, \mathbf{y}^{(\text{test})})$ are never touched during training.
- ▶ The final goal is to minimize the cost on the test data.

ENABLING GENERALIZATION: DATA

TRAINING, TEST AND VALIDATION

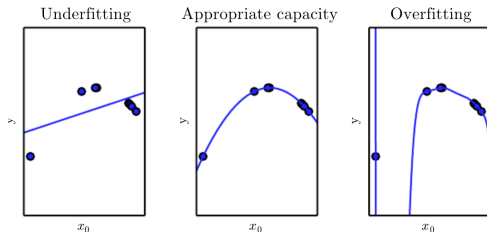
- ▶ Split (\mathbf{X}, \mathbf{y}) into
 - ▶ training data $(\mathbf{X}^{(\text{train})}, \mathbf{y}^{(\text{train})})$
 - ▶ validation data $(\mathbf{X}^{(\text{val})}, \mathbf{y}^{(\text{val})})$
 - ▶ test data $(\mathbf{X}^{(\text{test})}, \mathbf{y}^{(\text{test})})$
- ▶ While *training data* is to pick the optimal set of parameters (which specify elements from \mathcal{M}), using training and *validation data* in combination is for picking *hyperparameters*
- ▶ Hyperparameters can refer to choosing subsets of \mathcal{M} . For example, depth of a neural network, and widths of hidden layers. They may also refer to specifications of cost function or optimization procedure.
- ▶ $(\mathbf{X}^{(\text{test})}, \mathbf{y}^{(\text{test})})$ are never touched during training.
- ▶ The final goal is to minimize the cost on the test data.

ENABLING GENERALIZATION: DATA

TRAINING, TEST AND VALIDATION

- ▶ Split (\mathbf{X}, \mathbf{y}) into
 - ▶ training data $(\mathbf{X}^{(\text{train})}, \mathbf{y}^{(\text{train})})$
 - ▶ validation data $(\mathbf{X}^{(\text{val})}, \mathbf{y}^{(\text{val})})$
 - ▶ test data $(\mathbf{X}^{(\text{test})}, \mathbf{y}^{(\text{test})})$
- ▶ While *training data* is to pick the optimal set of parameters (which specify elements from \mathcal{M}), using training and *validation data* in combination is for picking *hyperparameters*
- ▶ Hyperparameters can refer to choosing subsets of \mathcal{M} . For example, depth of a neural network, and widths of hidden layers. They may also refer to specifications of cost function or optimization procedure.
- ▶ $(\mathbf{X}^{(\text{test})}, \mathbf{y}^{(\text{test})})$ are never touched during training.
- ▶ The final goal is to minimize the cost on the test data.

ENABLING GENERALIZATION: MODEL CAPACITY, UNDER- AND OVERFITTING



Left: Linear functions underfit

Center: Polynomials of degree 2 neither under- nor overfit

Right: Polynomials of degree 9 overfit

- ▶ Choose a class of models that has the right *capacity*
- ▶ Capacity too large: *overfitting*
- ▶ Capacity too small: *underfitting*

ENABLING GENERALIZATION: COST FUNCTION

REGULARIZATION

Let $C(f, f^*)$ be the cost function. Let $\mathbf{w} = (w_1, \dots, w_k)$ be the parameters specifying elements of $f_{\mathbf{w}} \in \mathcal{M}$.

- Usually, C refers to only known data points. That is, C evaluates as

$$C(f, f^*) = \sum_i C(f(x_i), y_i = f^*(x_i)) \quad (2)$$

where x_i runs over all training data points.

- Add a *regularization term* to cost function, and choose $f_{\mathbf{w}}$ that yields minimal

$$C(f_{\mathbf{w}}, f^*) + \lambda \Omega(\mathbf{w}) \quad (3)$$

- λ is a hyperparameter

ENABLING GENERALIZATION: COST FUNCTION

REGULARIZATION

Let $C(f, f^*)$ be the cost function. Let $\mathbf{w} = (w_1, \dots, w_k)$ be the parameters specifying elements of $f_{\mathbf{w}} \in \mathcal{M}$.

- Usually, C refers to only known data points. That is, C evaluates as

$$C(f, f^*) = \sum_i C(f(x_i), y_i = f^*(x_i)) \quad (2)$$

where x_i runs over all training data points.

- Add a *regularization term* to cost function, and choose $f_{\mathbf{w}}$ that yields minimal

$$C(f_{\mathbf{w}}, f^*) + \lambda \Omega(\mathbf{w}) \quad (3)$$

- λ is a hyperparameter

ENABLING GENERALIZATION: COST FUNCTION REGULARIZATION

- ▶ Prominent examples:
 - ▶ L_1 norm: $\Omega(\mathbf{w}) := \sum_i |w_i|$
 - ▶ L_2 norm: $\Omega(\mathbf{w}) := \sum_i w_i^2$
- ▶ Rationale: Penalize too many non-zero weights
- ▶ Virtually less complex model, hence virtually less capacity
- ▶ 🖱 Prevents overfitting, yields better generalization

ENABLING GENERALIZATION: OPTIMIZATION

EARLY STOPPING, DROPOUT

Optimization can be an iterative procedure.

- ▶ *Early stopping*: Stop the optimization procedure before cost function reaches an optimum on the training data.
- ▶ *Dropout*: Neural network specific. Randomly remove neurons and optimize parameters for neurons remaining.

Prominent Model Examples

SUPERVISED LEARNING

EXAMPLE: LINEAR REGRESSION

- ▶ Design matrix $\mathbf{X} \in \mathbb{R}^{m \times d}$, label vector $\mathbf{y} \in \mathbb{R}^m$
- ▶ Model class: Let $\mathbf{w} \in \mathbb{R}^d$

$$\begin{aligned} f_{\mathbf{w}} = f(\mathbf{x}; \mathbf{w}) : \quad \mathbb{R}^d &\longrightarrow \mathbb{R} \\ \mathbf{x} &\mapsto \mathbf{w}^T \mathbf{x} \end{aligned} \quad (4)$$

- ▶ *Remark:* Note that the case $\mathbf{w}^T \mathbf{x} + b$ can be treated as a special case to be included in \mathcal{M} , by augmenting vectors \mathbf{x}_i by an entry 1 (think about this...)
- ▶ Cost function (recall $y_i = f^*(\mathbf{x}_i)$)

$$C(f, f^*) := \frac{1}{m} \|(f(\mathbf{x}_1), \dots, f(\mathbf{x}_m)) - \mathbf{y}\|_2^2 = \frac{1}{m} \sum_{i=1}^m (f(\mathbf{x}_i) - y_i)^2 \quad (5)$$

SUPERVISED LEARNING

EXAMPLE: LINEAR REGRESSION

Optimization

- Solve for

$$\nabla_{\mathbf{w}} C(f_{\mathbf{w}}, f^*) = 0 \quad (6)$$

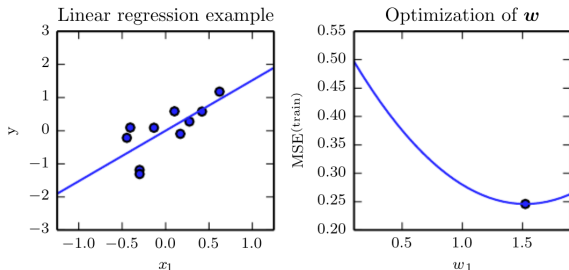
to achieve a minimum. This yields the *normal equations*

$$\mathbf{w} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y} \quad (7)$$

- *Global optimum* if $\mathbf{X}^T \mathbf{X}$ is invertible
- Do this on *training data* (so $\mathbf{X} = \mathbf{X}^{(\text{train})}$, $\mathbf{y} = \mathbf{y}^{(\text{train})}$) only.
Hope that cost on test data is small.

SUPERVISED LEARNING

LINEAR REGRESSION: NORMAL EQUATIONS



- ▶ *Left:* Data points, and the linear function $y = w_1 x$ that approximates them best
- ▶ *Right:* Mean squared error (MSE) depending on w_1
- ▶ *Remark on Perceptrons:* Optimizing is different, but also supported by a very easy optimization scheme (the *perceptron algorithm*)

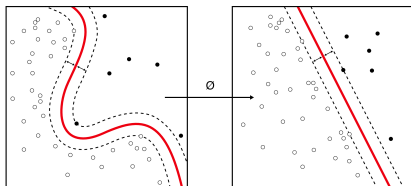
SUPERVISED LEARNING

POPULAR MODELS: SUPPORT VECTOR MACHINES

- *Realization*: From (7), write

$$\mathbf{w}^T \mathbf{x} = \sum_{i=1}^m \alpha_i \mathbf{x}^T \mathbf{x}_i = \sum_{i=1}^m \alpha_i \langle \mathbf{x}, \mathbf{x}_i \rangle \quad (8)$$

- Replace $\langle ., . \rangle$ by different *kernel* (i.e. scalar product) $k(. , .)$, that is by computing $\langle \phi(.), \phi(.) \rangle$ for appropriate ϕ
- ☞ Optimize for choosing good α 's: still easy to optimize both for regression and classification!



SUPERVISED LEARNING

POPULAR MODELS: NEAREST NEIGHBOR CLASSIFICATION

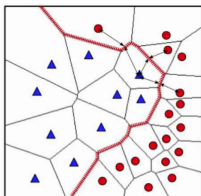
- Consider appropriate distance measure

$$D : \mathbb{R}^d \times \mathbb{R}^d \longrightarrow \mathbb{R}_+ \quad (9)$$

- For unknown data point \mathbf{x} , determine the closest given data point

$$\mathbf{x}_{i^*} := \operatorname{argmin}_i (D(\mathbf{x}, \mathbf{x}_i)) \quad (10)$$

- Predict label of \mathbf{x} as y_{i^*}



LECTURE 2: SUMMARY

- ▶ *Topics:*

- ▶ Supervised Learning
- ▶ Addressing Generalization
- ▶ Prominent Supervised Learning Methods

- ▶ *Reading:*

- ▶ <http://neuralnetworksanddeeplearning.com>:
Chapter 1, up to 'Perceptrons'
- ▶ <https://www.deeplearningbook.org/>: 5.1, 5.2, 5.3,
5.7

LECTURE 3: OUTLOOK

- ▶ *Topics:*

- ▶ Neural Networks
- ▶ Why going deep?
- ▶ Gradient Descent
- ▶ Preventing Slow Training

- ▶ *Reading:*

- ▶ <http://neuralnetworksanddeeplearning.com>:
Chapter 1, Chapter 3 until (and without) Overfitting and
Regularization
- ▶ <https://www.deeplearningbook.org/>: 5.10, 6.1, 6.2,
6.3

Thanks for your attention