Machine learning foundations II

Mitko Veta

Eindhoven University of Technology
Department of Biomedical Engineering

2020

Learning goals

At the end of this lecture you will:

- Have a good understanding of the basic principles of machine learning (ML) and be able to apply them in the analysis of ML methods.
- Be able to design good experimental setups for developing ML models.
- Have a good understanding of the different evaluation measures for ML models.

Overview

Topics covered in this lecture:

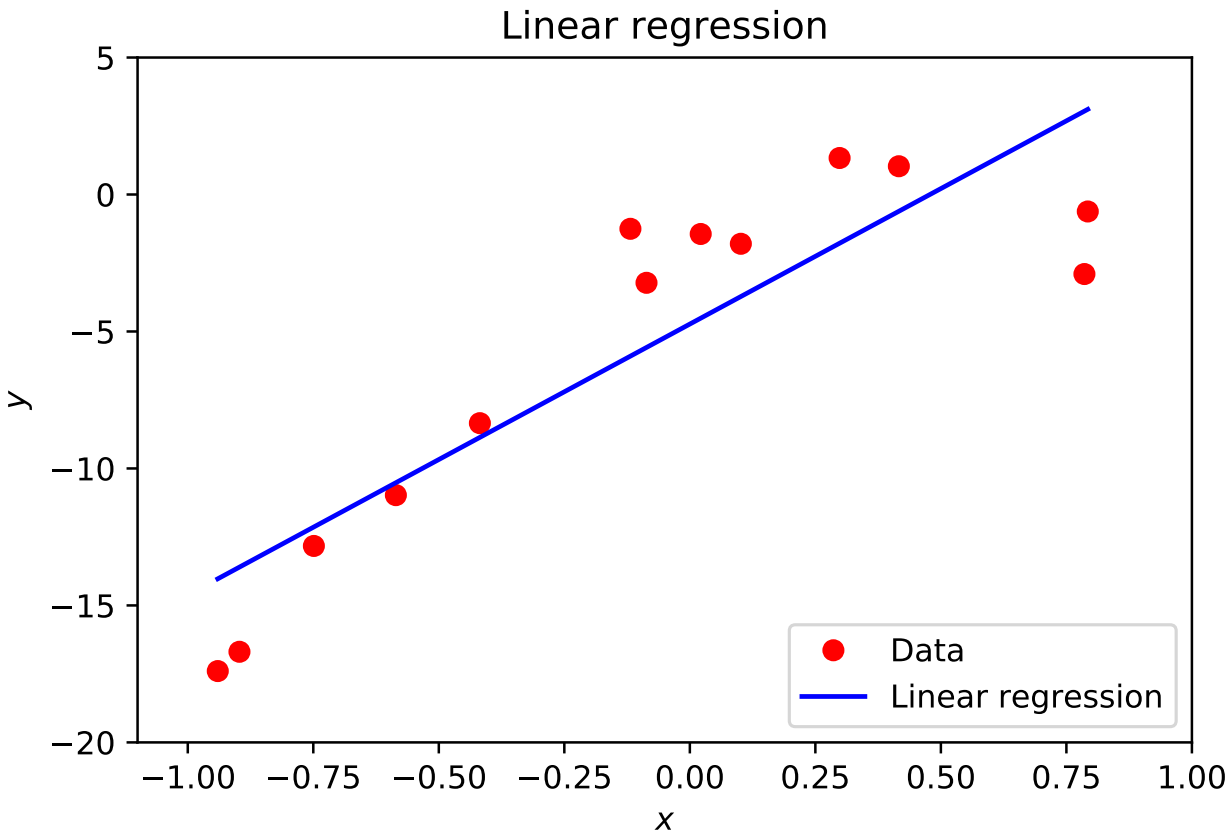
1. Model capacity, underfitting and overfitting
2. Model selection
3. Bias and variance trade-off
4. Maximum likelihood estimation
5. Model evaluation
6. Supervised and unsupervised learning algorithms
7. Ensambling

Model capacity, underfitting and overfitting

Materials:

- Chapter I.5.2 from Goodfellow et al., *Deep Learning*

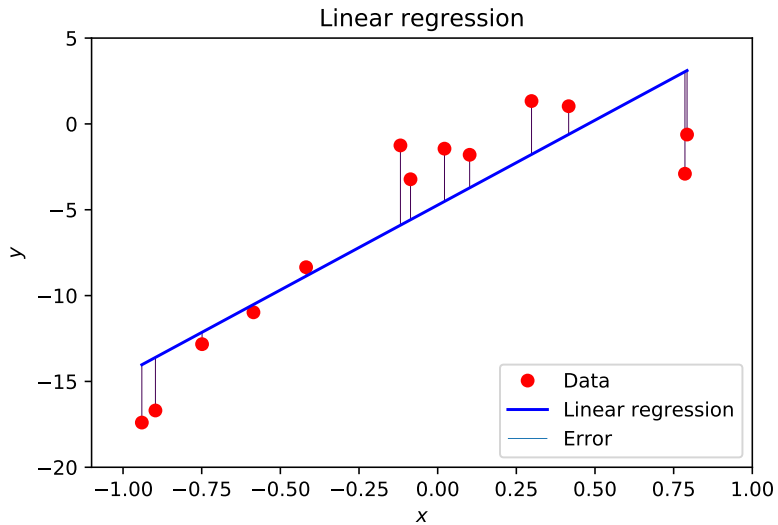
Linear regression

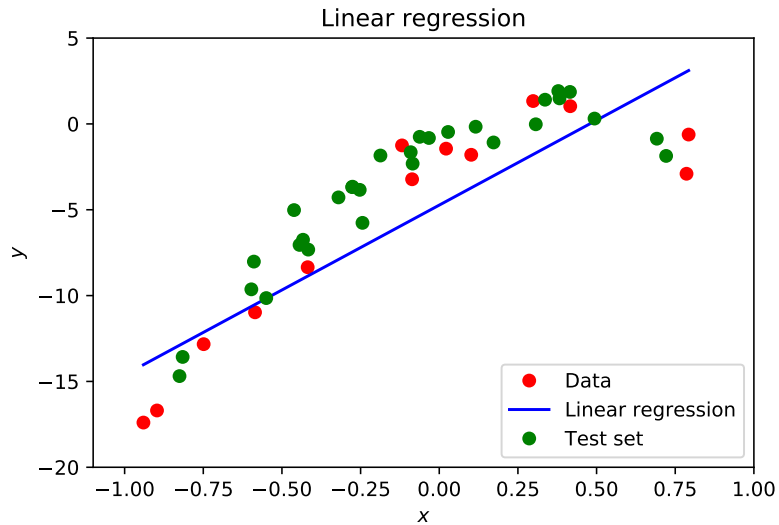


$$\hat{y} = \hat{w}_0 + \sum_{i=1}^{n} x_i \hat{w}_i$$
$$\hat{y} = \boldsymbol{x}^T \hat{\boldsymbol{w}}$$

Generalization

- The central challenge in machine learning is to design an algorithm which will perform well on new data (different from the training set data).
- This ability is called **generalization**.
- **Training error** is the error computed on the training set.





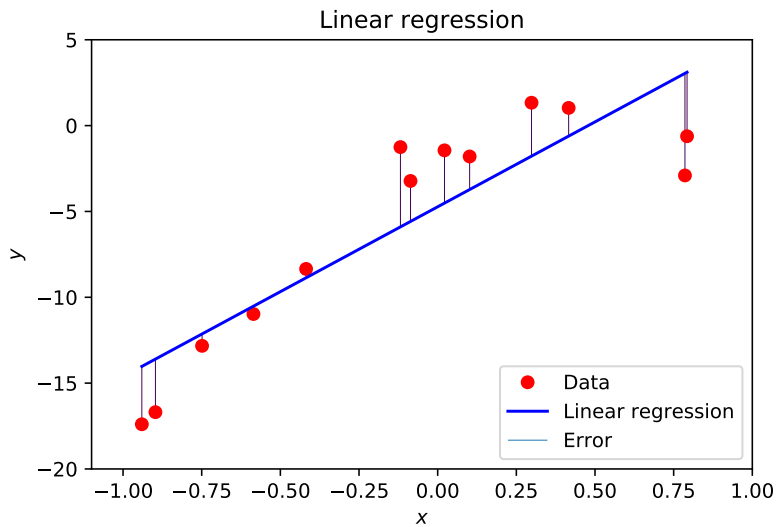
Example: Linear regression

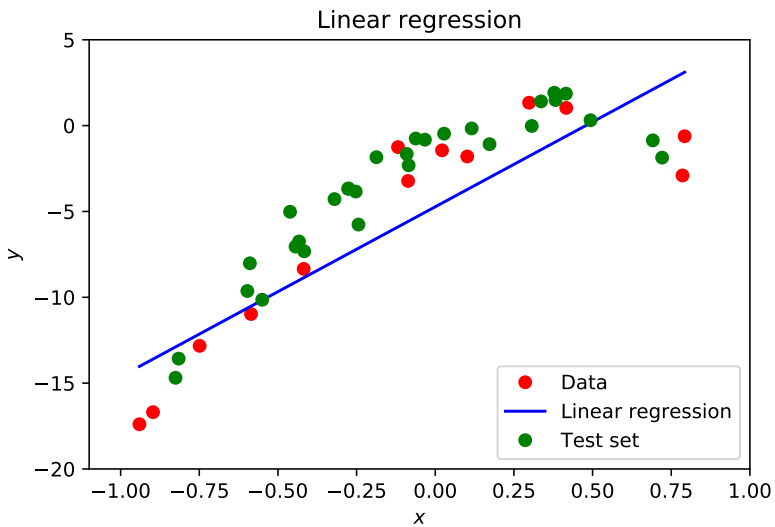
- Previously, we trained the model by minimizing the training error

$$\frac{1}{m^{(\text{train})}} \left\| \boldsymbol{X}^{(\text{train})} \hat{\boldsymbol{w}} - \boldsymbol{y}^{(\text{train})} \right\|_2^2$$

- We would like actually to minimize the test error

$$\frac{1}{m^{(\text{test})}} \left\| \boldsymbol{X}^{(\text{test})} \hat{\boldsymbol{w}} - \boldsymbol{y}^{(\text{test})} \right\|_2^2$$





Statistical learning theory

- **Statistical learning theory** provides methods to mathematically reason about the performance on the test set although we can observe only the training set.
- This is possible under some assumptions about the data sets
  - The training and test data are generated by drawing from a probability distribution over data sets. We refer to that as **data-generating process**.
  - **i.i.d. assumptions**
    - Examples in each data sets are **independent** from each other.
    - The training data set and the test data set are **identically distributed**, i.e., drawn from the same probability distribution.

Discussion point

Can you name a scenario in medical image analysis practice where the i.i.d. assumptions are bound to be broken?

Underfitting and overfitting

- The factor that determines how well a machine algorithm will perform is its ability to
  1. Make the training error small.
  2. Make the difference between the training and test error small.
- These two factors correspond to the two central challenges in machine learning: **underfitting** and **overfitting**.

Model capacity

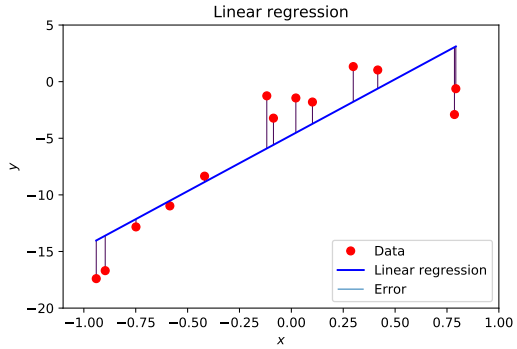
- A **capacity of the model** is its ability to fit a wide variety of functions.
- The capacity can be controlled by choosing its **hypothesis space**, i.e. the set of functions from which the learning algorithm is allowed to select the solution.
- Example: The linear regression algorithm has the set of all linear functions as its hypothesis space.

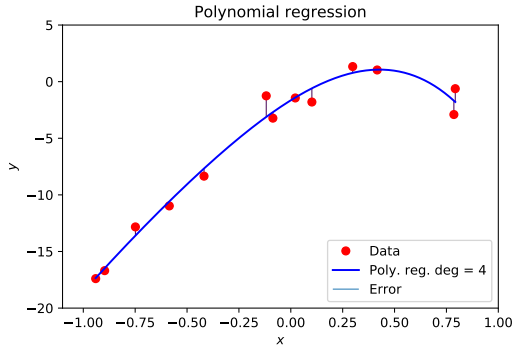
Polynomial regression

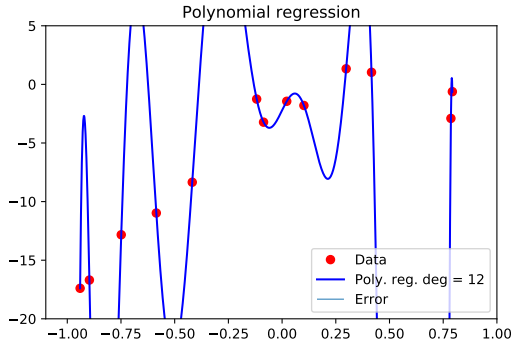
- The linear regression algorithm can be generalized to include all polynomial functions instead of just the linear ones.
- Moving to degree two to we obtain: $\hat{y} = b + w_1 x + w_2 x^2$.
  - This can be seen as adding a new feature x^2.
  - In fact, we can generalize this approach to create all sorts of hypothesis spaces, e.g.: $\hat{y} = b + w_1 x + w_2 \sin(x) + w_3 \sqrt{x}$.
- The **outuput** is still a **linear** function of the parameters, so in principle it can be trained in the same way as the linear regression.

Polynomial regression

A comparison of a linear, degree-4, and degree-12 polynomials as predictors

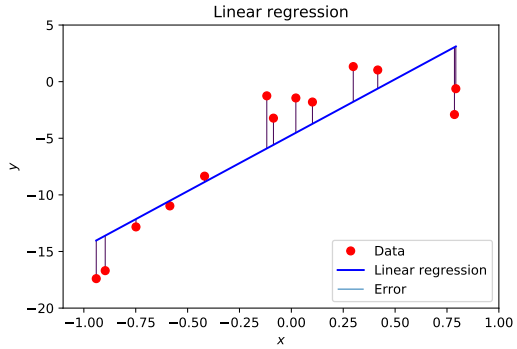


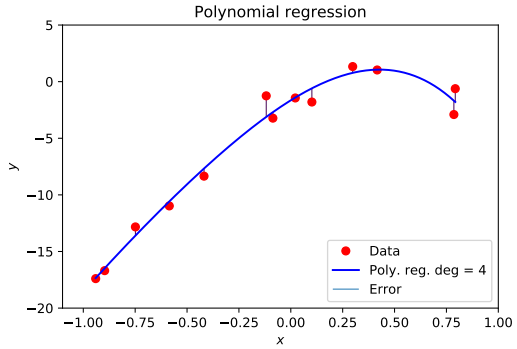


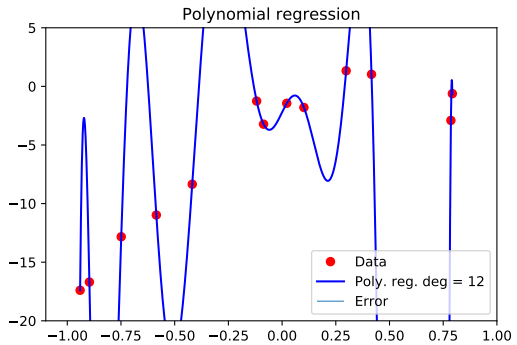


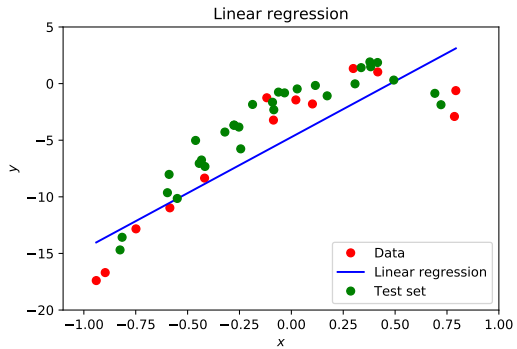
Polynomial regression

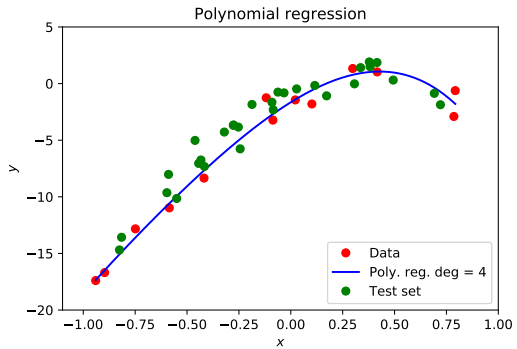
A comparison of a linear, degree-4, and degree-12 polynomials as predictors

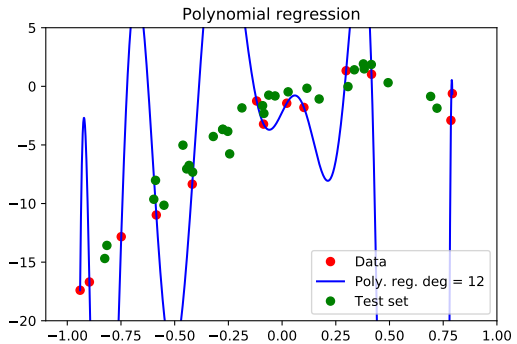








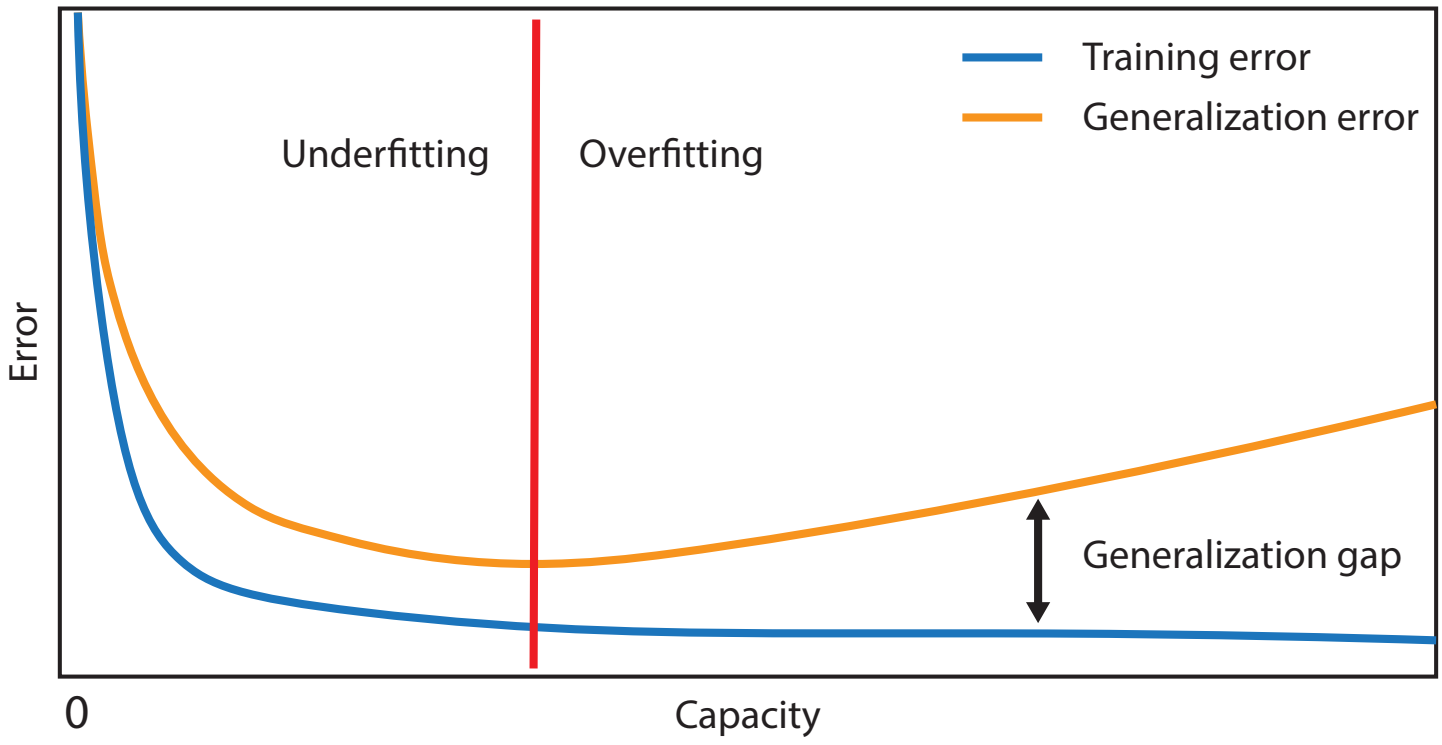




Generalization and capacity

- Simpler functions generalize more easily, but we still need to choose a sufficiently complex hypothesis (function) to obtain small training error.
- Typically training error decreases with the increase of the model capacity until an (asymptotic) value is reached.
- The generalization error is U-shaped with the capacity range split in an underfitting and an overfitting zone.

Generalization and capacity



Regularization

- In addition to increasing and decreasing of the hypothesis space, i.e., the capacity, we can influence the learning algorithm by **giving preference to one solution over another in the hypothesis space**.
- In case both functions are eligible we can define a condition to express preference about one of the functions.
- The unpreferred solution is chosen only if it gives significantly better performance with the training data.
- *More on regularization in the next lecture.*

Model selection

Materials:

- Chapter I.5.3 from Goodfellow et al., *Deep Learning*

Hyperparameters and validation sets

- **Hyperparameters** are settings that can be used to control the behaviour of the algorithm.
- In general, the hyperparameters are not modified by the learning algorithm itself.
- **Example**: In **polynomial regression** the degree of the polynomial is a **capacity** hyperparameter.
- A setting can be chosen to be hyperparameter when it is **difficult to optimize** or - more often - when its derivation from the training set **can lead to overfitting**.
  - Example: in polynomial regression we can always fit the data better with a higher degree polynomial.

Choice of training, validation, and test sets

- The **validation set** is used during training to predict the behaviour (generalization error) of the algorithm on new data, i.e., on the test set and to chose the hyperparameters.
- Ideally these two sets are disjoint.
- The validation set is chosen from the training data.
- The training data is split in two disjoint subsets.

Choice of training, validation, and test sets

Training	Validation	Test
Used to find the optimal **parameters** of the model.	Used to find the optimal **model** (hyper-parameters).	Used to estimate the **performance** of the optimal model.
w	$f(\cdot)$	$\|\|\hat{y} - y\|\|$

Discussion point

How large should the training, validation and testing datasets be as a percentage (%) of the total available data?

Cross-validation

- Dividing the data set into disjoint training and test sets can result in a result in a too small validation and/or test set.
- In such cases all data is used to estimate the generalization error.
- We use procedures that repeat the training and testing on different randomly chosen subsets or splits of the original data set.
- The most common such procedure is the **k-fold cross-validation**.

Expectation (recap)

- The **expectation** or **expected** value of a function $f(x)$ with respect to a probability distribution $P(x)$ is the average value of f over all values x assuming they are drawn from P

$$\mathbb{E}_{x\sim P}[f(x)] = \sum_x P(x)f(x)$$

$$\mathbb{E}_{x\sim P}[f(x)] = \int p(x)f(x)dx$$

Variance (recap)

- The **variance** gives a measure of variation of the values of a random variable x

$$\mathrm{Var}(f(x)) = \mathbb{E}[(f(x) - E[f(x)])^2]$$

Bias and variance trade-off

Materials:

- Chapter I.5.4 from Goodfellow et al., *Deep Learning*

Point estimation

- For efficient design of learning algorithms it is useful to have formal characterizations of notions like generalization, overfitting and underfitting.
- To this end we introduce some definitions.
- **Point estimation** is the attempt to provide the single "best" prediction of some quantity of interest.
- The quantity of interest can be a single parameter, parameter vector of some model, e.g., the weights $\boldsymbol{w}$ in the linear regression model.

Point estimation

- Given a parameter $\boldsymbol{\theta}$ we denote its point estimate with $\hat{\boldsymbol{\theta}}$.
- As usual, let $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ be m independent and identically distributed (i.i.d.) data points.
- A **point estimator** or **statistic** is any function of the data

$$\hat{\boldsymbol{\theta}}_m = g(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}))$$

- This definition is very general. For instance, that the value returned by g need not be close to the true value $\boldsymbol{\theta}$. Also g might return a value which is outside the values that $\boldsymbol{\theta}$ is allowed to have.

Point estimation

- Of course, a good estimator is still a function that returns values close to θ.
- Since the data is drawn from a random process, point estimate $\hat{\theta}$ is considered to be a random variable and θ is fixed, but unknown parameter.

Bias

- A bias of an estimator $\hat{\boldsymbol{\theta}}_m$ is defined as

$$\text{bias}(\hat{\boldsymbol{\theta}}_m) = \mathbb{E}(\hat{\boldsymbol{\theta}}_m) - \boldsymbol{\theta}$$

  where the expectation is over the data and $\boldsymbol{\theta}$ is the true underlying value.
- An estimator $\hat{\boldsymbol{\theta}}_m$ is **unbiased** if $\text{bias}(\hat{\boldsymbol{\theta}}_m) = 0$. Note that this implies $\mathbb{E}(\hat{\boldsymbol{\theta}}_m) = \boldsymbol{\theta}$.
- $\hat{\boldsymbol{\theta}}_m$ is **asymptotically unbiased** if $\lim_{m\to\infty} \text{bias}(\hat{\boldsymbol{\theta}}_m) = 0$ (implying $\lim_{m\to\infty} \mathbb{E}(\hat{\boldsymbol{\theta}}_m) = \boldsymbol{\theta}$).

Bias: example

- **Example**: Consider samples $\{x^{(1)}, \ldots, x^{(m)}\}$ i.i.d distributed according to the Gaussian distribution

$$p(x^{(i)}; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2}\frac{x^{(i)} - \mu}{\sigma^2}\right)$$

- The **sample mean** is a common estimator of the Gaussian mean parameter

$$\hat{\mu}_m = \frac{1}{m}\sum_{i=1}^{m} x(i)$$

Bias: example

We compute the bias as expectation by substituting the Gaussian distribution in the formula

$$\begin{aligned}
\text{bias}(\mu_m) &= \mathbb{E}[\mu_m] - \mu \\
&= \mathbb{E}\left[\frac{1}{m}\sum_{i=1}^{m} x^{(i)}\right] - \mu \\
&= \left(\frac{1}{m}\sum_{i=1}^{m} \mathbb{E}[x^{(i)}]\right) - \mu \\
&= \left(\frac{1}{m}\sum_{i=1}^{m} \mu\right) - \mu \\
&= \mu - \mu = 0
\end{aligned}$$

The sample mean is an unbiased estimator of Gaussian mean parameter.

Bias: example

- **Example**: Estimators of the variance of a Gaussian distribution
- We compare two different estimators of the variance σ^2 parameter
- **Sample variance**

$$\hat{\sigma}^2 = \frac{1}{m} \sum_{1}^{m} \left(x^{(i)} - \hat{\mu}_m \right)^2$$

where $\hat{\mu}$ is the sample mean.

- We are interested in computing

$$\text{bias}(\hat{\sigma}_m^2) = \mathbb{E}[\hat{\sigma}_m^2] - \sigma^2$$

Bias: example

- First we evaluate $\mathbb{E}[\hat{\sigma}_m^2]$:

$$\mathbb{E}[\hat{\sigma}_m^2] = \mathbb{E}\left[\frac{1}{m}\sum_{1}^{m}\left(x^{(i)} - \hat{\mu}_m\right)^2\right] = \frac{m-1}{m}\sigma^2$$

- Back to the bias

$$\text{bias}(\hat{\sigma}_m^2) = \mathbb{E}[\hat{\sigma}_m^2] - \sigma^2 = \frac{m-1}{m}\sigma^2 - \sigma^2 = -\frac{\sigma^2}{m}$$

- Therefore the sample variance is a **biased** estimator.

Bias: example

- The **unbiased variance estimator** is defined as

$$\tilde{\sigma}^2 = \frac{1}{m-1} \sum_{1}^{m} \left(x^{(i)} - \hat{\mu}_m \right)^2$$

- Indeed

$$\mathbb{E}[\tilde{\sigma}_m^2] = \mathbb{E}\left[\frac{1}{m-1} \sum_{1}^{m} \left(x^{(i)} - \hat{\mu}_m \right)^2 \right] = \frac{m-1}{m-1} \sigma^2 = \sigma^2$$

and the bias is 0.

Variance and standard error

- Another important feature of an estimator is its variance.
- The **variance** of an estimator is simple its statistical variance $\text{Var}(\hat{\theta})$ over the training set as a random variable.
- Alternatively we can compute the **standard error** (the square root of the variance) $\text{SE}(\hat{\theta})$.
- The variance or the standard error provide a measure how much the estimate would vary as we resample the data independently from the underlying data generating process.
- We would prefer a relatively low variance of the estimator.

Variance and standard error

- The standard error of the mean estimator is given as

$$\mathrm{SE}(\hat{\mu}) = \sqrt{\mathrm{Var}\left[\frac{1}{m}\sum_{i=1}^{m} x^{(i)}\right]} = \frac{\sigma}{\sqrt{m}}$$

  where σ is the true variance of the distribution, i.e., the samples $x^{(i)}$.
- Neither the square root of the sample variance nor the square root of the unbiased estimator of the variance give an unbiased estimate of the standard deviation.
- Both approaches underestimate the true standard deviation.
- However, for large m the approximation works quite well.

Variance and standard error

- Often the generalization error is estimated based on the sample mean of the error on the test set.
- The accuracy of the estimate depends on the number of the examples.
- From the statistical theory (central limit theorem) we know that the mean is distributed with normal distribution for which we can establish confidence intervals.
- For instance, the 95% confidence interval is given by

$$[\hat{\mu_m} - 1.96\text{SE}(\hat{\mu}_m), \hat{\mu_m} + 1.96\text{SE}(\hat{\mu}_m]$$

- Then we can say that algorithm A is better than algorithm B of the confidence upper bound for the error of A is less than the corresponding lower bound of B.

Trading off bias and variance to minimize mean squared error

- Bias and variance measure two different sources of error in an estimator.
- Bias measures the expected deviation with the true value of the estimator.
- Variance provides a measure of the deviation from the expected value of the estimator depending on the particular data sampling.

Trading off bias and variance to minimize mean squared error

- Often we need to make a trade-off between these two.
- The most common way to do this is via cross-validation.
- An alternative is to compare the **mean squared error** (MSE) of the estimates.

$$\text{MSE} = \mathbb{E}[(\hat{\theta}_m - \theta)^2] = \text{bias}(\hat{\theta}_m)^2 + \text{Var}(\hat{\theta}_m)$$

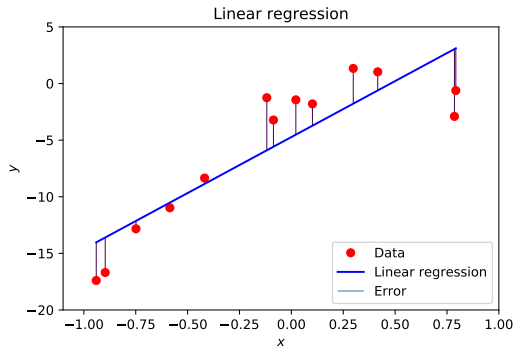
- The smaller MSE the better - so minimizing both the bias and variance is always preferable.

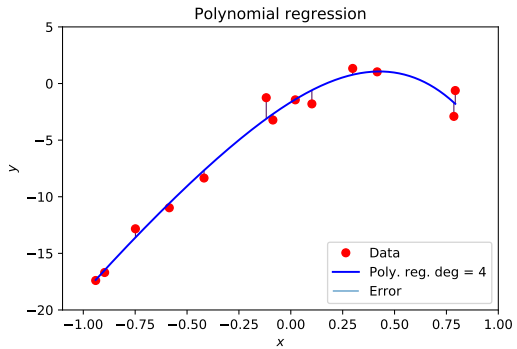
Bias and variance

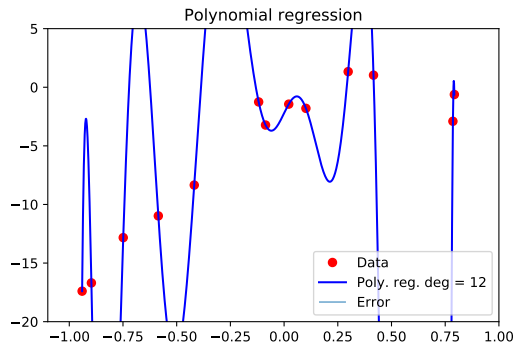
- Our original goal was to provide a mathematical support for the notions of capacity, underfitting, and overfitting.
- Indeed there is a close relationship between these three concepts and bias and variance.
- When generalization error is measured by MSE (and hence indirectly via bias and variance) increasing capacity tends to increase variance and decrease bias.
- Again the generalization as a function of capacity is given by an U-shaped curve.

Discussion point

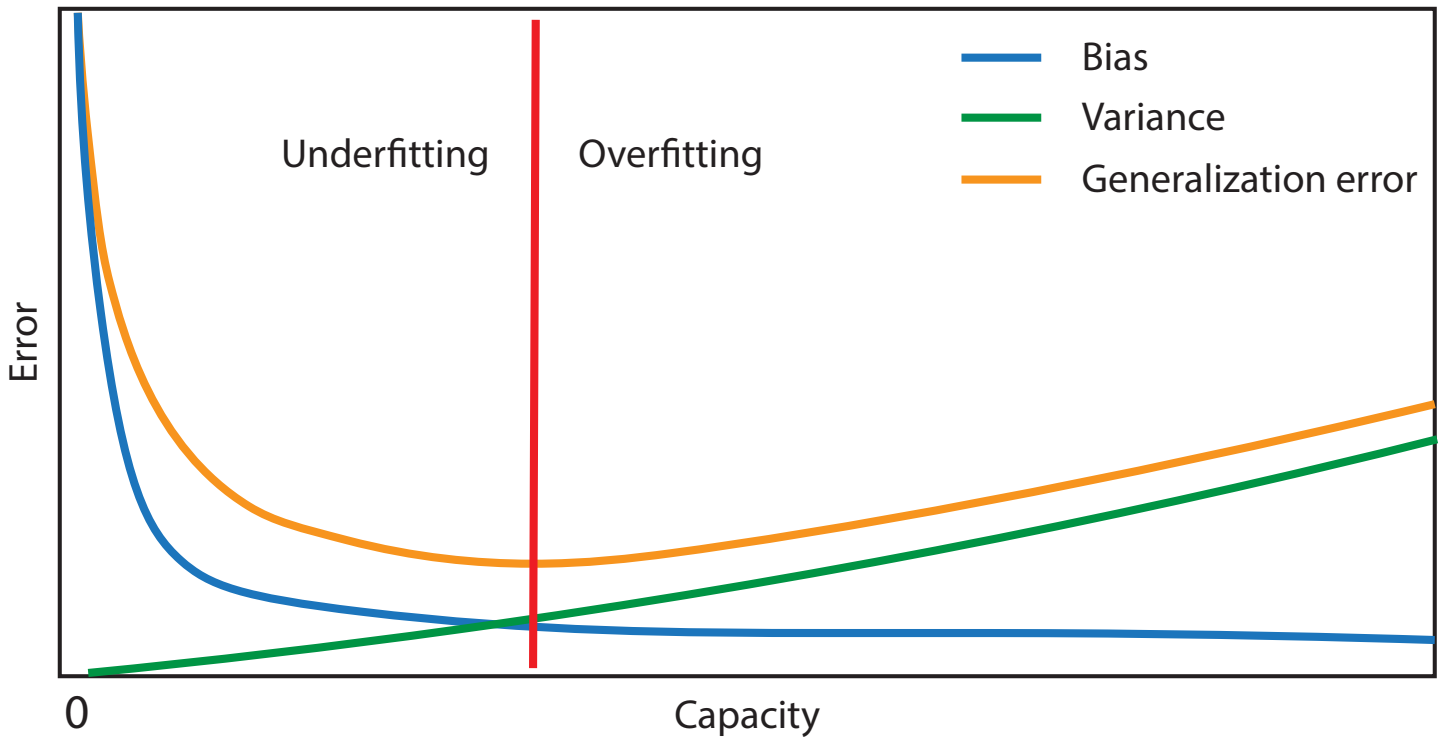
How will the estimated regression model change when one training data point is replaced with another one?







Bias and variance



Maximum likelihood estimation

Materials:

- Chapter I.5.5 from Goodfellow et al., *Deep Learning*

Maximum likelihood estimation

- We would like to have some principle from which we can derive good estimator functions for a large scale of models.
- The **maximum likelihood estimation** is the most common such principle.
- Given observation data and a corresponding (statistical) model our goal is to find the parameter vector which imply the highest probability to obtain the data.

Maximum likelihood estimation

- Consider a set of m examples $\mathbb{X} = \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(m)}\}$ drawn independently from the true but unknown distribution $p_{\text{data}}(\mathbf{x})$.
- Let $p_{\text{model}}(\mathbf{x}; \boldsymbol{\theta})$ be a parametric family of probability distributions, i.e., for each $\boldsymbol{\theta}$ we get a different distribution p_{model}.
- $p_{\text{model}}(\boldsymbol{x}; \boldsymbol{\theta})$ maps any configuration $\boldsymbol{x}$ to a real number estimating the (true) probability $p_{\text{data}}(\boldsymbol{x})$

Maximum likelihood estimation

- The maximum likelihood estimator for $\boldsymbol{\theta}$ is then defined as

$$\boldsymbol{\theta}_{\text{ML}} = \text{argmax}_{\boldsymbol{\theta}} p_{\text{model}}(\mathbb{X}; \boldsymbol{\theta}) = \text{argmax}_{\boldsymbol{\theta}} \prod_{i=1}^{m} p_{\text{model}}(\boldsymbol{x}^{(i)}; \boldsymbol{\theta})$$

  Note that also the empirical distribution $\hat{p}_{\text{data}}$ is implicitly present in the formula through $\boldsymbol{x}^{(i)}$.

- A more convenient equivalent optimization problem is obtained by taking logarithm of the product

$$\boldsymbol{\theta}_{\text{ML}} = \text{argmax}_{\boldsymbol{\theta}} p_{\text{model}}(\mathbb{X}; \boldsymbol{\theta}) = \sum_{i=1}^{m} \log p_{\text{model}} \boldsymbol{x}^{(i)}; \boldsymbol{\theta})$$

Maximum likelihood estimation

- We can further rescale by dividing the expression by m

$$\boldsymbol{\theta}_{\text{ML}} = \text{argmax}_{\boldsymbol{\theta}} p_{\text{model}}(\mathbb{X}; \boldsymbol{\theta})$$

$$\boldsymbol{\theta}_{\text{ML}} = \text{argmax}_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{x} \sim \hat{p}_{\text{data}}} \log p_{\text{model}}(\boldsymbol{x}; \boldsymbol{\theta})$$

- In this way the problem is expressed as an equivalent expectation problem (now the empirical distribution $\hat{p}_{\text{data}}$ becomes explicit).

Maximum likelihood estimation

- Perhaps more straightforwardly, the maximum likelihood estimation can be seen as minimizing the dissimilarity between $\hat{p}_{\text{data}}$ and p_{model}.
- The degree of dissimilarity is given by the KL-divergence

$$D_{\text{KL}}(\hat{p}_{\text{data}} \| p_{\text{model}}) = \mathbb{E}_{\mathbf{x} \sim \hat{p}_{\text{data}}} [\log \hat{p}_{\text{data}}(\boldsymbol{x}) - \log p_{\text{model}}(\boldsymbol{x})]$$

- Only the term of the right is function of the model, so it is the only one which needs to be minimized

$$-\mathbb{E}_{\mathbf{x} \sim \hat{p}_{\text{data}}} [\log p_{\text{model}}(\boldsymbol{x})]$$

  which is equivalent with the maximization problem from the previous slide.
- It boils down to minimizing the **cross-entropy** between the two distributions.

Maximum likelihood estimation

- **The maximum likelihood estimation can be seen as an attempt to make the model distribution p_{model} to match the empirical distribution $\hat{p}_{\text{data}}$.**
- Ideally we would like to match the data generating distribution p_{data}, but we do not have access to it.

Conditional log likelihood and mean square error

- The maximal likelihood estimator can be generalized to estimate a conditional probability $P(\mathbf{y} \mid \mathbf{x}; \boldsymbol{\theta})$.
- Let all inputs be given by $\boldsymbol{X}$ and all observed outputs by $\boldsymbol{Y}$. Then the conditional maximum likelihood estimator is

$$\boldsymbol{\theta}_{\mathrm{ML}} = \arg\max_{\theta} P(\boldsymbol{Y} \mid \boldsymbol{X}; \boldsymbol{\theta})$$

- If the examples are assumed to be i.i.d., then this can be decomposed into

$$\boldsymbol{\theta}_{\mathrm{ML}} = \arg\max_{\theta} \sum_{i=1}^{m} \log P(\boldsymbol{y}^{(i)} \mid \boldsymbol{x}^{(i)}; \boldsymbol{\theta})$$

Example: linear regression as maximum likelihood

- With an infinitely large training set we might see several examples whit the same input $\boldsymbol{x}$ but different y.
- The learning algorithm needs to fit the distribution to all these y corresponding to the same $\boldsymbol{x}$.
- To derive the linear regression algorithm we assume $p(y \mid \boldsymbol{x}) = \mathcal{N}(y; \hat{y}(\boldsymbol{x}; \boldsymbol{w}), \sigma^2)$, where $\hat{y}(\boldsymbol{x}; \boldsymbol{w})$ gives the (prediction of the) mean of the normal distribution and σ is fixed to some chosen constant.
- The parameter vector θ corresponds in this case to $\boldsymbol{w}$.

Example: linear regression as maximum likelihood

- By substituting (the full Gaussian function version of) $p(y \mid \boldsymbol{x})$ in the conditional log-likelihood formula we obtain

$$\sum_{i=1}^{m} \log p(\boldsymbol{x}^{(i)} \mid y^{(i)}; \boldsymbol{\theta}) = -m \log \sigma - \frac{m}{2} \log(2\pi) - \sum_{i=1}^{m} \frac{\left\| \hat{y}^{(i)} - y^{(i)} \right\|^2}{2\sigma^2}$$

  where $\hat{y}^{(i)}$ is the linear regression on the i-th input $\boldsymbol{x}^{(i)}$.

- By comparing with the mean squared error

$$\mathrm{MSE}_{\mathrm{train}} = \frac{1}{m} \sum_{i=1}^{m} \left\| \hat{y}^{(i)} - y^{(i)} \right\|^2$$

  one can see that maximizing the log-likelihood with respect to $\boldsymbol{w}$ results with the same estimate of $\boldsymbol{x}$ as minimizing MSE.

Model evaluation

Materials:

- Fawcett, “An introduction to ROC analysis”

Model evaluation

- To quantitatively evaluate a machine learning algorithm we need to define a **performance measure**.
- Usually the performance measure is specific to the task carried out by the algorithm.
- For classification tasks a natural measure is the model **accuracy**.
- The **accuracy** is defined as the proportion of examples for which the model produces the correct output.
- An equivalent (complementary) measure is the **error rate** defined as the proportion of incorrect outputs.

Model evaluation

- The best way to evaluate a machine learning algorithm is by applying it to a **test set** data which has not been seen before.
- Ideally there should be **no overlap** between the **test set** and the **training set** used to obtain the model.

Binary classification

- We consider **binary classification** problems, i.e., problems using only two classes/
- Formally each input example $x^{(i)}$ needs to be mapped into one element of the set $\{\boldsymbol{p}, \boldsymbol{n}\}$ of **true** classes.
- A **classification model** (**classifier**) is a function from the input examples to the set $\{\boldsymbol{Y}, \boldsymbol{N}\}$ of **predicted classes** or **hypothesized classes**.
- $\boldsymbol{p}, \boldsymbol{n}$ correspond to $\boldsymbol{Y}, \boldsymbol{N}$, respectively.

Binary classification

- For a given classifier there are four possible outcomes.
- If the true class of $x^{(i)}$ is $\boldsymbol{p}$ and the predicted class is $\boldsymbol{Y}$ then we have a **true positive** (TP); if it was classified $\boldsymbol{N}$, then we have a **false negative** (FN).
- Symmetrically, a $x^{(i)}$ with true class $\boldsymbol{n}$ which is assigned a predicted class $\boldsymbol{N}$ is a **true negative** (TN); if the predicted class is $\boldsymbol{Y}$, then it is a false positive (FP).

Confusion matrix

These four combinations can be put together in a **confusion matrix**, also called **contingency table**.

		True class	
		p	***n***
Predicted class	***Y***	True positives (TP)	False positives (FP)
	N	False negatives (FN)	True negatives (TN)

Binary classifications metrics

- Using the four basic categories of prediction outcomes (TP, FP, TN, FN) we can derive various **measures** of performance of classification models.
- For instance the accuracy can be defined as

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN}$$

- Also quite frequently used measures are

$$\textit{Sensitivity} = \frac{TP}{TP + FN} \quad \textit{Specificity} = \frac{TN}{TN + FP} \quad \textit{Precision} = \frac{TP}{TP + FP}$$

Binary classification metrics

- Sensitivity is also called **recall**, **true positive rate** or **hit rate**.
- In medical contexts the sensitivity can be interpreted as a measure of the extent to which diseased individuals are correctly diagnosed.
- In general: measures the proportion of the target group the method is able to detect, i.e. how sensitive is to this group.
- Specificity is also called **true negative rate** or **selectivity**.
- In medical contexts the specificity can be interpreted as a measure of the extent to which healthy individuals are correctly diagnosed.
- The precision tells us which proportion of the positive predictions is correct.

Discussion point

You have developed a method for some image analysis diagnostic task that has very high sensitivity (e.g. 0.99) but relatively low specificity (e.g. 0.25).
Can this be still a useful tool for clinicians and if so in what context?

Binary classification metrics

- Sometimes the above mentioned measures are not sufficient.
- For example, in a population in which the percentage of healthy individuals is much larger than the diseased individuals, it is easy to achieve high specificity by trivially classifying each patient as healthy.
- We can obtain more objective evaluation by combing metrics.
- The metrics F_1 is the harmonic mean (average) of the precision and recall (sensitivity)

$$\frac{2}{F_1} = \frac{1}{Precision} + \frac{1}{Recall} \text{ or } F_1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

Areas under the curve measures

- (Binary) classifications often depend on some parameter (e.g., threshold).
- Hence on way to combine two metrics is by assigning them to the axes of a coordinate system and varying this parameter to construct a graphical plot.
- We obtain a curve (actually, most of the time series of points) such that each point corresponds to a particular parameter value.
- The area under the curve is a measure of how good is the classification.

Receiver Operating Characteristic (ROC) curve

- The ROC curve plots the true positive rate (sensitivity) versus the false positive rate (1 - specificity).

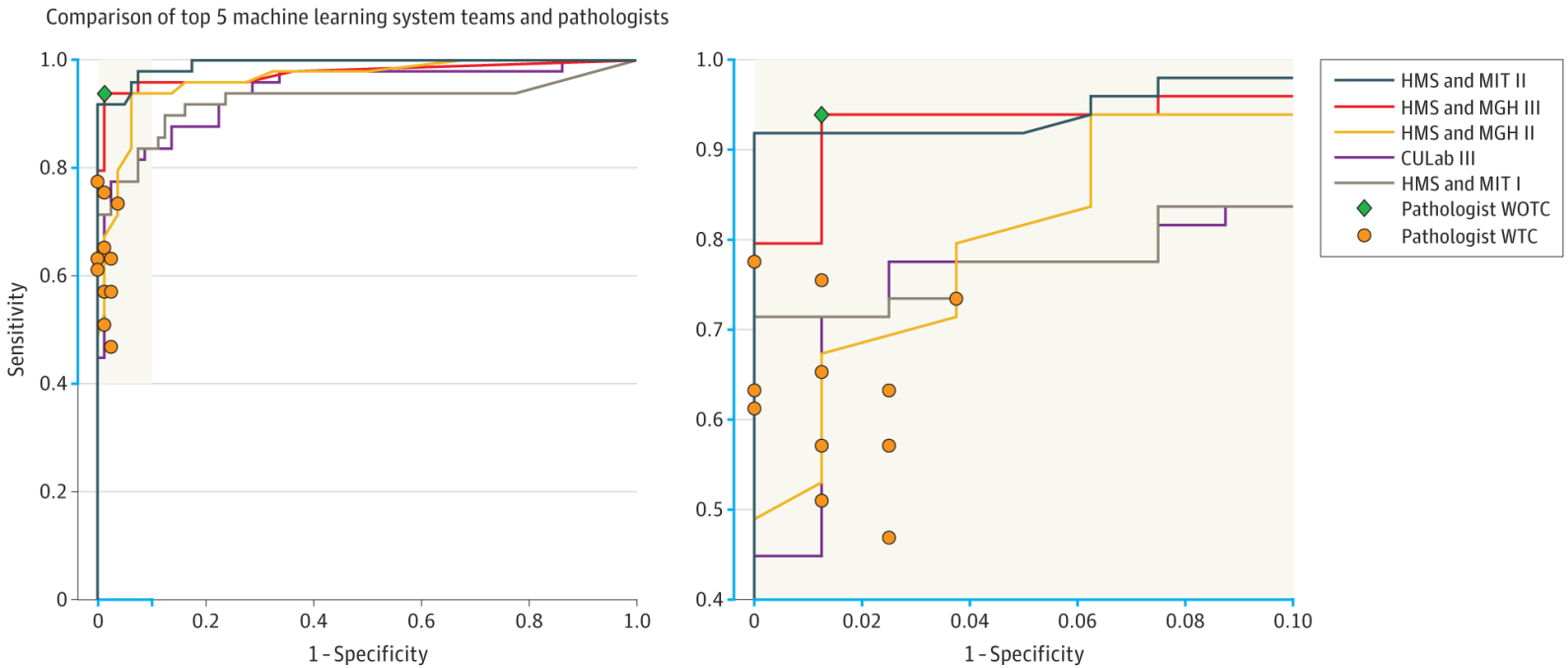


Figure from Bejnordi et al., "Diagnostic assessment of deep learning algorithms for detection of lymph node metastases in women with breast cancer"

Receiver Operating Characteristic (ROC) curve

- There are several characteristic points points in the ROC space:
  - (0,0) corresponds to the strategy of never making a positive classification.
  - (1,1) is the opposite: unconditionally issuing a positive classification.
  - (0,1) represents perfect classification.
  - Obviously we strive to achieve this ideal point as as a result have as much as possible area under the curve covered (ideally it should cover the whole square corresponding to the ROC space)
- A less common example of a measure combination into a graphical plot is the precision-recall plot (recall on the x-axis, precision on the y-axis).

Supervised and unsupervised learning algorithms

Materials:

- Chapters I.5.6 and I.5.7 from Goodfellow et al., *Deep Learning*

Supervised learning algorithms

- Learning algorithms that learn based on a given training examples $\boldsymbol{x}$ and their corresponding outputs $\boldsymbol{y}$.
  - Linear and logistic regressions
  - Support vector machines
  - k-nearest neighbours
  - Decision trees

Unsupervised learning algorithms

- Unsupervised algorithms experience only "features", but not supervision feedback.
- The distinction with the supervised algorithms is not always clear since there is no good test to distinguish if something is a feature or a target provided by the supervisor.
- Rule of thumb: in unsupervised algorithms no human annotation is needed for the training examples.
  - Principal component analysis
  - k-means clustering
  - t-Distributed Stochastic Neighbor Embedding
  - Generative adversarial networks

Ensambling

Materials:

- Chapter II.7.11 from Goodfellow et al., *Deep Learning*

Acknowledgements

The slides for this lecture were prepared by Mitko Veta and Dragan Bošnacki.
Some of the slides are based on the accompanying lectures of Goodfellow et al., *Deep Learning*.

References

- Bejnordi, B. E. et al. "Diagnostic assessment of deep learning algorithms for detection of lymph node metastases in women with breast cancer". In: *Jama* 318.22 (2017), pp. 2199–2210.
- Fawcett, T. "An introduction to ROC analysis". In: *Pattern recognition letters* 27.8 (2006), pp. 861–874.
- Goodfellow, I., Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.