UiT Norges arktiske universitet

DTE-2501 AI Methods and Applications

Basic introduction to AI

Lecture 2/3 – Machine Learning methodology

Tatiana Kravetc
Førsteamanuensis

Office: D2240

Email: tatiana.kravetc@uit.no

Overview

V Practical example

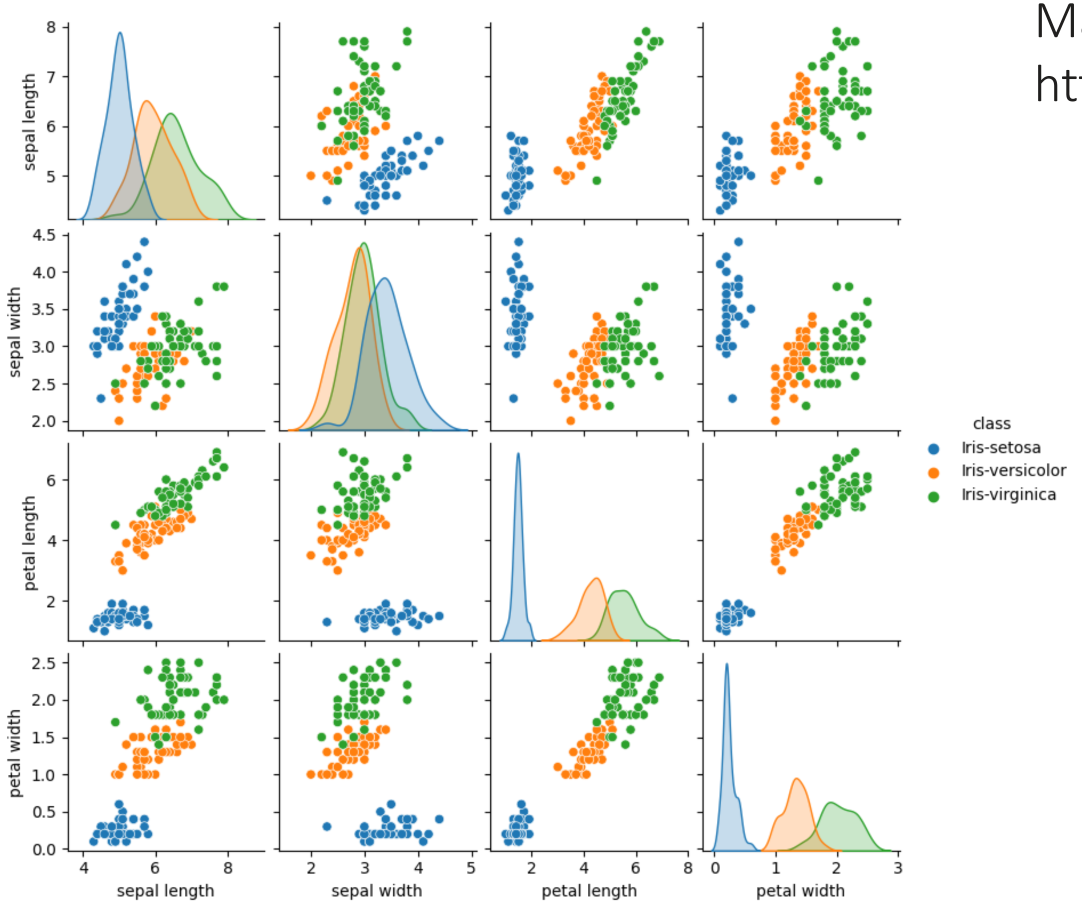
VI Machine learning methodology

a) Data preprocessing
b) Linear model
c) Learning method
d) Loss function
e) Empirical risk minimization
f) Underfitting and overfitting

VII Application examples

V Practical example

Standard practical example. Classification of iris plant (R.A.Fisher)



Machine Learning repository:
https://archive.ics.uci.edu/ml/index.php

VI Machine learning methodology

Data preprocessing

1. Acquire the relevant dataset (https://archive.ics.uci.edu/ml/index.php)
2. Identifying the missing values
3. Splitting the data set into two separate sets: *training set* and *test set*
4. Feature scaling: *standardization* and *normalization*

$$x' = \frac{x - \mathrm{mean}(x)}{\sigma} \qquad x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

Linear model

Linear model $g(x, \theta)$ is a weighted sum of all features (linear combination).

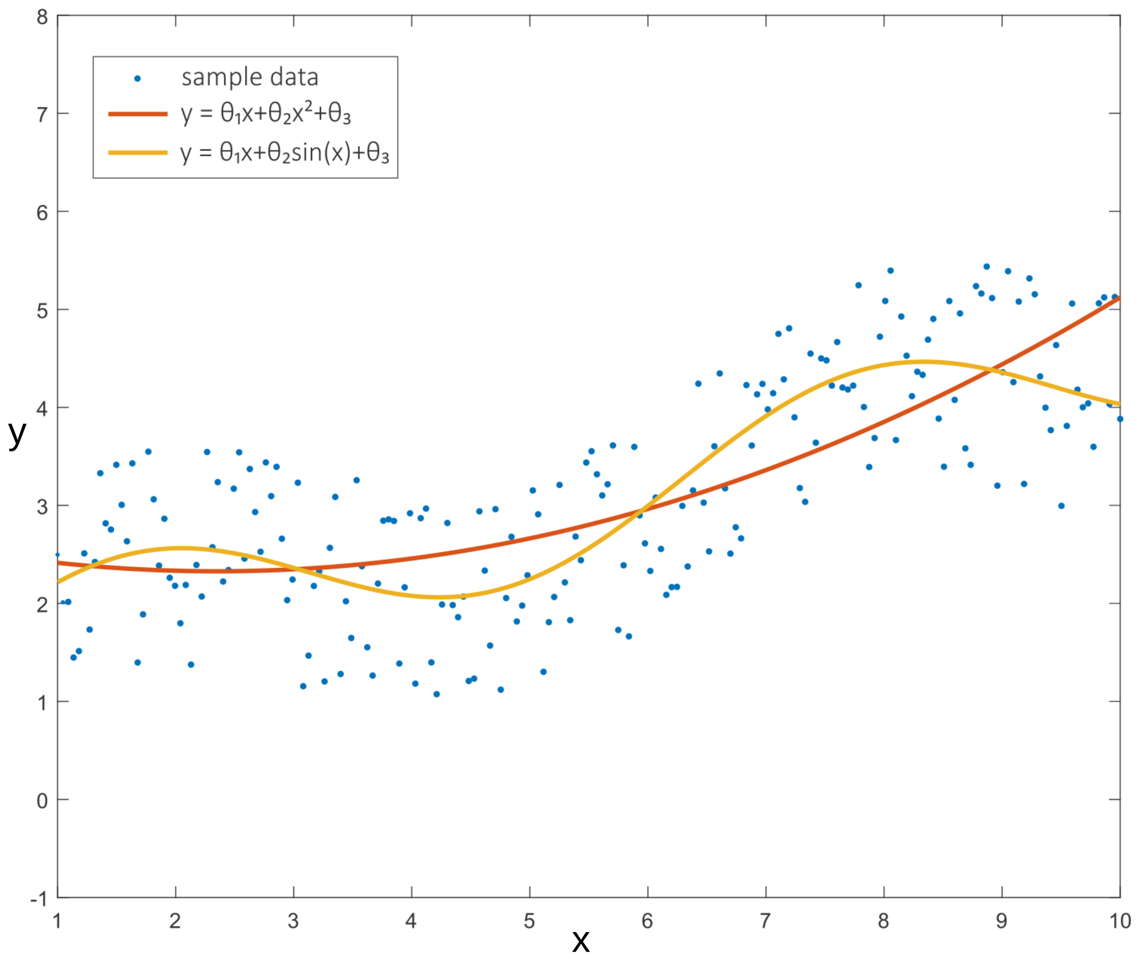
Let $\theta = (\theta_1, \dots, \theta_n)$ be a vector of real coefficients.

$g(x, \theta) = \sum_{j=1}^{n} \theta_j f_j(x)$ is a regression model, $Y = \mathbb{R}$

$g(x, \theta) = \operatorname{sign} \sum_{j=1}^{n} \theta_j f_j(x)$ is a classification model, $Y = \{-1, +1\}$

Example: regression problem, synthetic data

$X = Y = \mathbb{R}, l = 200, n = 3$ features: $\{x, x^2, 1\}$ and $\{x, \sin(x), 1\}$



Learning method

- Training stage

Learning model builds an algorithm a to find coefficients that describe (approximate) the given data

$$\boxed{\begin{pmatrix} f_1(x_1) & \cdots & f_n(x_1) \\ \vdots & \ddots & \vdots \\ f_1(x_l) & \cdots & f_n(x_l) \end{pmatrix} \to \begin{pmatrix} y_1 \\ \cdots \\ y_l \end{pmatrix} \to a}$$

- Testing stage

Applying the trained algorithm to the new data $\widetilde{x_i}$

$$\begin{pmatrix} f_1(\tilde{x}_1) & \cdots & f_n(\tilde{x}_1) \\ \vdots & \ddots & \vdots \\ f_1(\tilde{x}_k) & \cdots & f_n(\tilde{x}_k) \end{pmatrix} \to a \to \begin{pmatrix} a(\tilde{x}_1) \\ \cdots \\ a(\tilde{x}_k) \end{pmatrix}$$

Loss function

Machine learning solves optimization problems. In order to construct an algorithm that is optimal for the given data, we need to introduce algorithm errors, or, in other words, *loss function* $\varepsilon(a, x)$, where a is an algorithm and $x \in X$ is a training sample.

Loss function depends on the problem type. For example,

- Classification: $\varepsilon(a, x) = [a(x) \neq y(x)]$ is an error indicator (boolean variable)
- Regression: $\varepsilon(a, x) = |a(x) - y(x)|$ is an absolute error; $\varepsilon(a, x) = \left(a(x) - y(x)\right)^2$ is a squared error

Thus, we introduce so called *empirical risk* that we will minimize. Empirical risk in an average error functional:

$$Q\left(a, X^l\right) = \frac{1}{l}\sum_{i=1}^{l} \varepsilon(a, x_i)$$

Empirical risk minimization, ERM

Minimization of the empirical risk can be written as

$$\mu(X^l) = \arg\min_a Q(a, X^l)$$

where μ is a learning method and $\mathbf{arg\,min}$ – argument of the minimum – are points x for which the functional attains its smallest value.

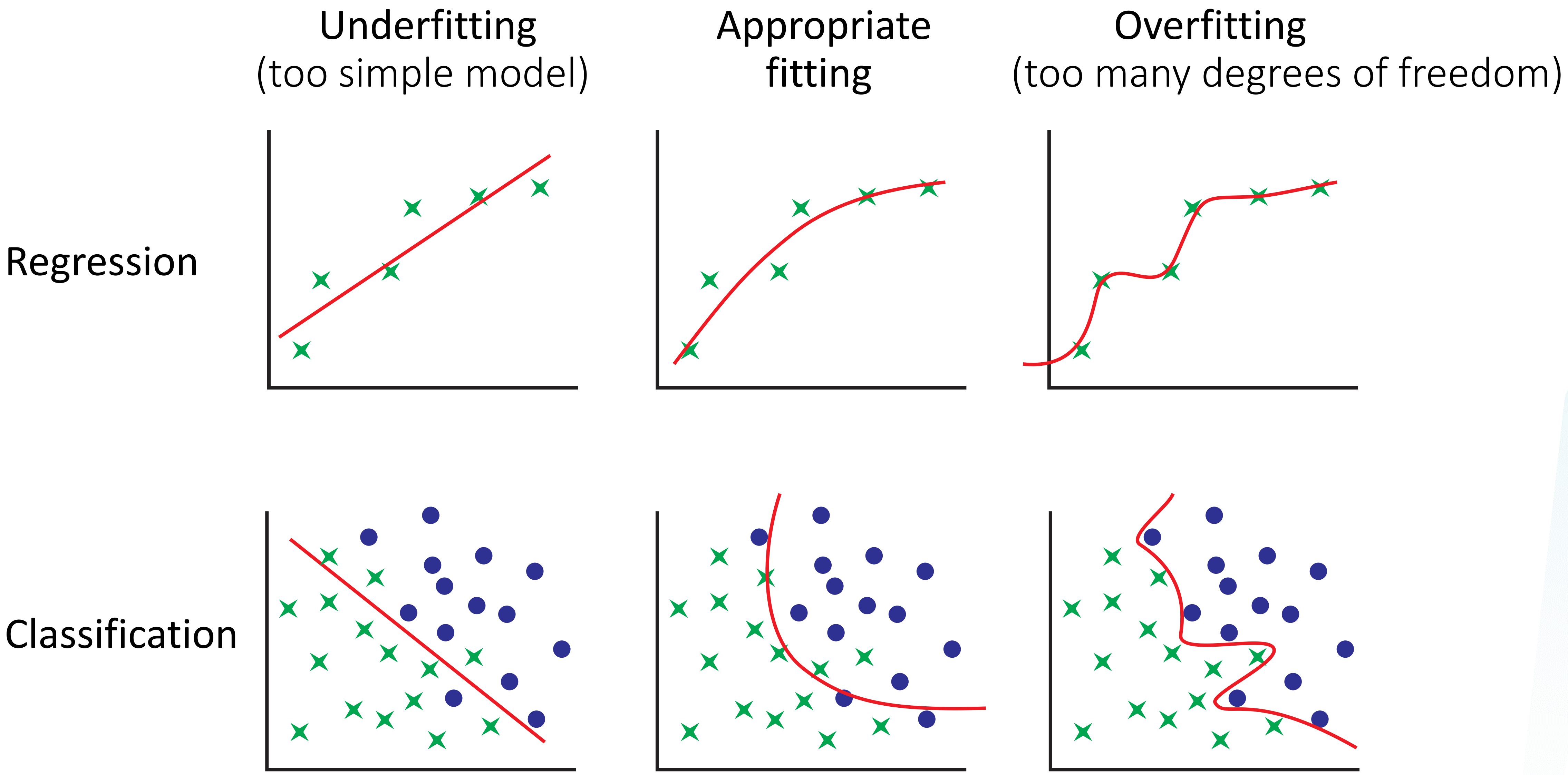
Example: regression problem, $Y = \mathbb{R}$; n features $f_j: X \to \mathbb{R}$, $j = 1, \dots, n$;

Linear regression model: $g(x_i, \theta) = \sum_{j=1}^{n} \theta_j f_j(x)$, $\theta \in \mathbb{R}^n$

Squared error $\varepsilon(a, x) = \big(a(x) - y(x)\big)^2$

A particular ERM case is *a least squares method*:

$$\mu\big(X^l\big) = \arg\min_\theta \sum_{i=1}^{l} (g(x_i, \theta) - y_i)^2$$


Underfitting
(too simple model)
Appropriate
fitting
Overfitting
(too many degrees of freedom)
Regression
Classification

VII Application examples

Classification

a) Medical diagnostics

b) Credit scoring

c) Churn prediction – big data analysis

d) Biometric classification of a person – deep neural network

Regression

e) Forecasting property value

f) Business analytics – sales forecasting