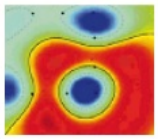
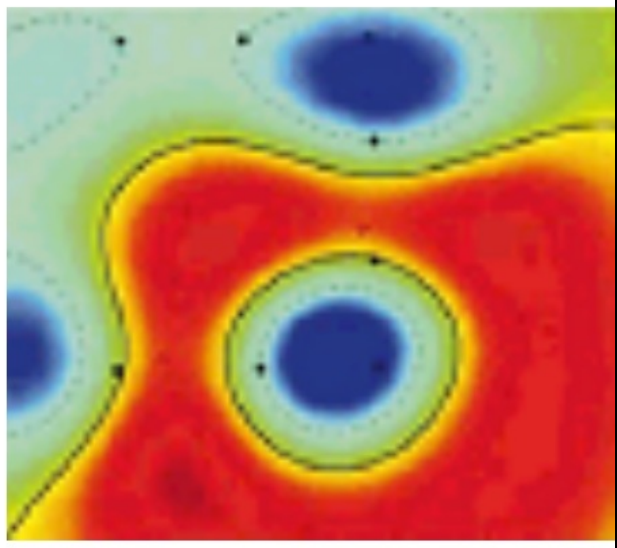


WiSe 2024/25

Machine Learning 1/1-X

Technische Universität Berlin



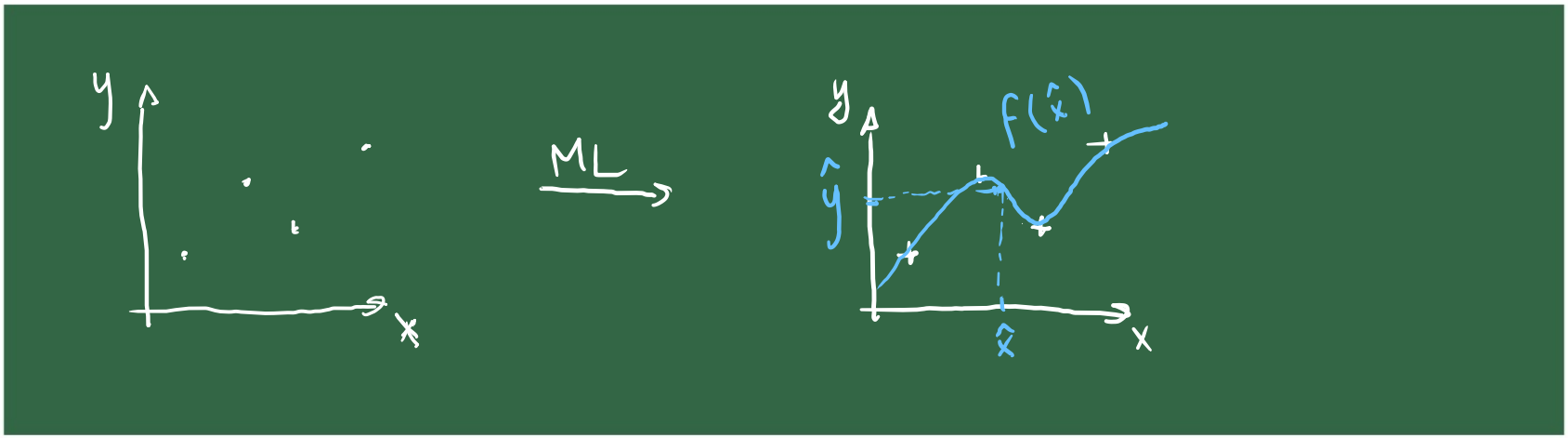
Lecture 1

Bayes Decision Theory

Introduction: Machine Learning

What is Machine Learning?

- Designing algorithms (machines) that efficiently convert finite sets of data (observations) into models with predictive capability.



Why Machine Learning?

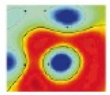
- *Autonomous Decision Making:* Substitute/support human decision making for achieving gains in efficiency in practical applications.
- *Knowledge Discovery:* Finding laws that explain empirical phenomena.

Autonomous Decision Making

Modern economies involve taking good decisions in complex multi-dimensional problems. In many cases, for efficiency reasons, it is necessary that these decisions are taken autonomously, or with the help of predictive models.

Examples:

- Supply Chain Management
  (demand forecasting, planning)
- End-User Services
  (recommendations, personalization,
  translation, monitoring)
- Manufacturing
  (quality control, materials
  optimization)
- Finance / Insurance
  (risk management, forecasting)
- etc.



Autonomous Decision Making

Modern economies involve taking good decisions in complex multi-dimensional problems. In many cases, for efficiency reasons, it is necessary that these decisions are taken autonomously, or with the help of predictive models.

Examples:

- **Supply Chain Management**
  (demand forecasting, planning)
- End-User Services
  (recommendations, personalization, translation, monitoring)
- Manufacturing
  (quality control, materials optimization)
- Finance / Insurance
  (risk management, forecasting)
- etc.

Demand Forecasting

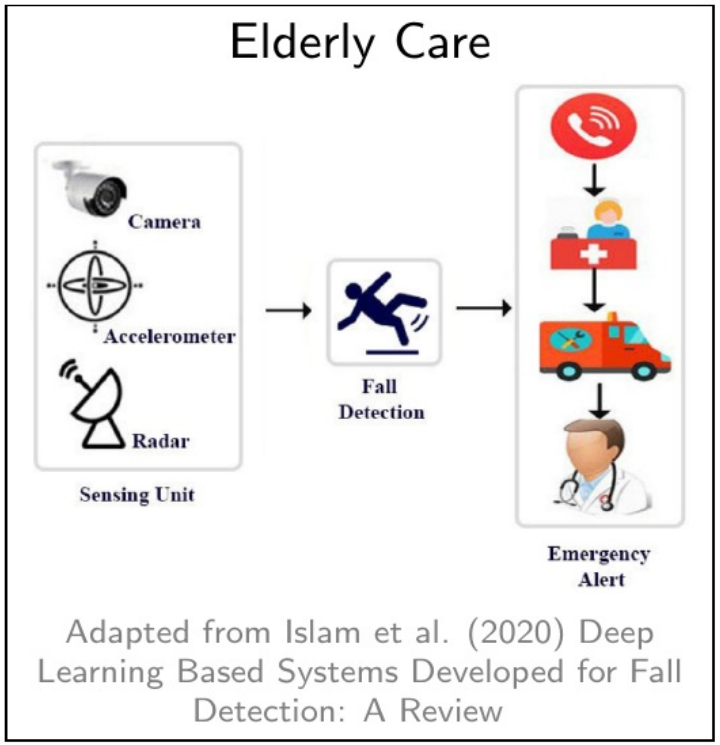
Source: IBM

Autonomous Decision Making

Modern economies involve taking good decisions in complex multi-dimensional problems. In many cases, for efficiency reasons, it is necessary that these decisions are taken autonomously, or with the help of predictive models.

Examples:

- Supply Chain Management
  (demand forecasting, planning)
- **End-User Services**
  (recommendations, personalization, translation, monitoring)
- Manufacturing
  (quality control, materials optimization)
- Finance / Insurance
  (risk management, forecasting)
- etc.



Adapted from Islam et al. (2020) Deep Learning Based Systems Developed for Fall Detection: A Review

Autonomous Decision Making

Modern economies involve taking good decisions in complex multi-dimensional problems. In many cases, for efficiency reasons, it is necessary that these decisions are taken autonomously, or with the help of predictive models.

Examples:

- Supply Chain Management
  (demand forecasting, planning)
- End-User Services
  (recommendations, personalization, translation, monitoring)
- **Manufacturing**
  (quality control, materials optimization)
- Finance / Insurance
  (risk management, forecasting)
- etc.

Surface Inspection

Source: MvTec

Autonomous Decision Making

Modern economies involve taking good decisions in complex multi-dimensional problems. In many cases, for efficiency reasons, it is necessary that these decisions are taken autonomously, or with the help of predictive models.

Examples:

- Supply Chain Management
  (demand forecasting, planning)
- End-User Services
  (recommendations, personalization, translation, monitoring)
- Manufacturing
  (quality control, materials optimization)
- **Finance / Insurance**
  (risk management, forecasting)
- etc.

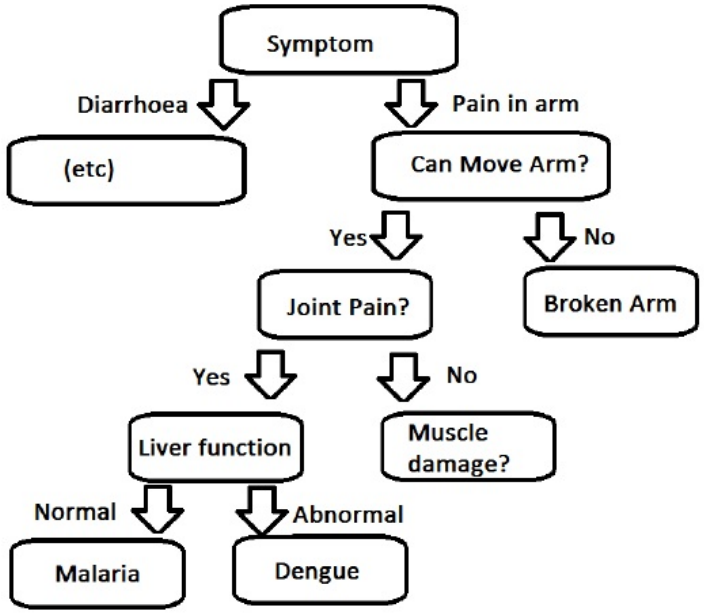
Forecasting (prices/claims)

Image source: https://www.kaggle.com/robikscube/m5-forecasting-starter-data-exploration

1st-Gen Decision Systems

Idea:

- The human programs the decision system by hand (e.g. using if/else controls) in a way that it replicates his own decision strategy.
- If the system performs as expected on the few available test cases, it is then run autonomously on new instances.



Source: https://sites.google.com/site/keremitgsnotes

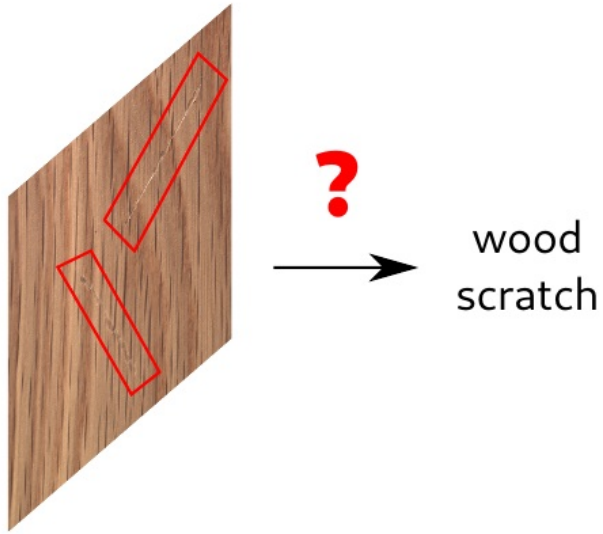
1st-Gen Decision Systems

Idea:

- The human programs the decision system by hand (e.g. using if/else controls) in a way that it replicates his own decision strategy.
- If the system performs as expected on the few available test cases, it is then run autonomously on new instances.

Problem:

- What if the user is not able to translate his own decision behavior into an actual program? (e.g. how does one detect objects in natural images?)



2nd-Gen Decision Systems

Idea: The human collects a **dataset** of examples, and labels them according to his own decision strategy:

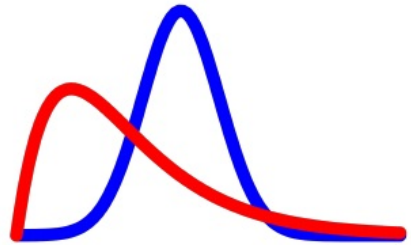
Good examples:

Examples with scratches:

A machine learning model is trained to map each example (i.e. the array of pixels received as input) to the correct class.

Part II

Bayes Decision Theory



Bayesian Decision Theory

A theoretical framework for building decision models that:

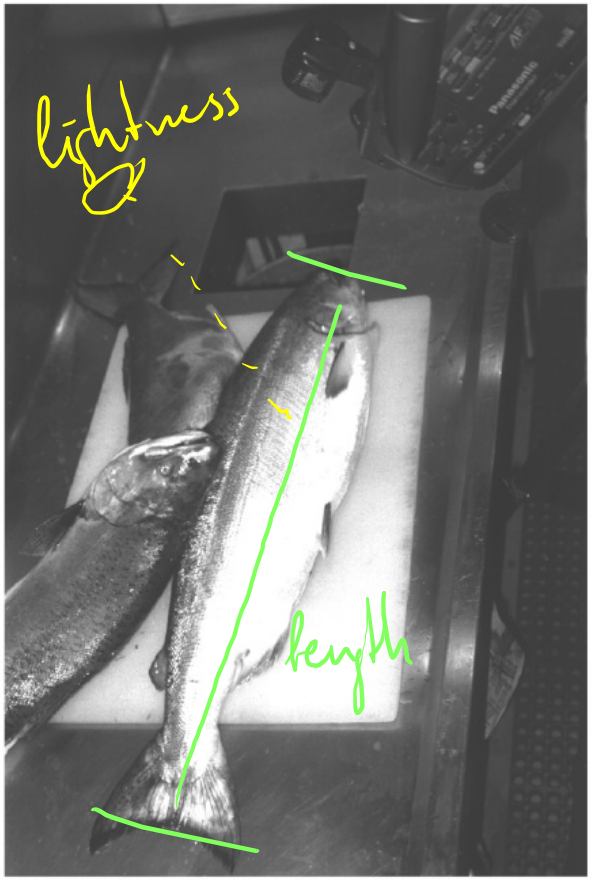
- Yields optimal classifiers (under some favourable conditions).
- Helps us to understand how
  - properties of the data distribution (e.g. Gaussian-distributed),
  - prior probabilities of classes,
  - the criterion to optimize (e.g. maximum classification accuracy),

  affect the classifier qualitatively and quantitatively.

Bayes Decision Theory

Example: (from the book Duda et al. 2000)

- Fishes of various species (*salmon* and *sea bass*) arrive on a conveyor belt.
- Sensors placed on the conveyor belt produce a collection of measurements for each observed fish (e.g. length, lightness).
- We would like to build a decision model that assign each fish to one of the two possible classes (*salmon* and *sea bass*).



Bayesian Decision Theory

Notation:

- $\omega_1, \omega_2, \ldots$ set of classes (e.g. salmon, sea bass, ...),
- $\boldsymbol{x} \in \mathbb{R}^d$ vector of observations (e.g. x_1 is the length and x_2 is the lightness).

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

We are also given the probability laws:

- $P(\omega_j)$: probability of being of class j, "prior"
- $p(\boldsymbol{x} \mid \omega_j)$: density function of measurements for each class, "likelihood"
- $p(\boldsymbol{x})$: density function of measurements (marginalized), "evidence"

but what we are truly interested in is:

- $P(\omega_j \mid \boldsymbol{x})$: probability of being of a certain class after observing $\boldsymbol{x}$.

"posterior"

Bayesian Decision Theory

Bayes Theorem:

$$P(\omega_j \mid \boldsymbol{x}) = \frac{p(\boldsymbol{x} \mid \omega_j) \cdot P(\omega_j)}{p(\boldsymbol{x})}$$

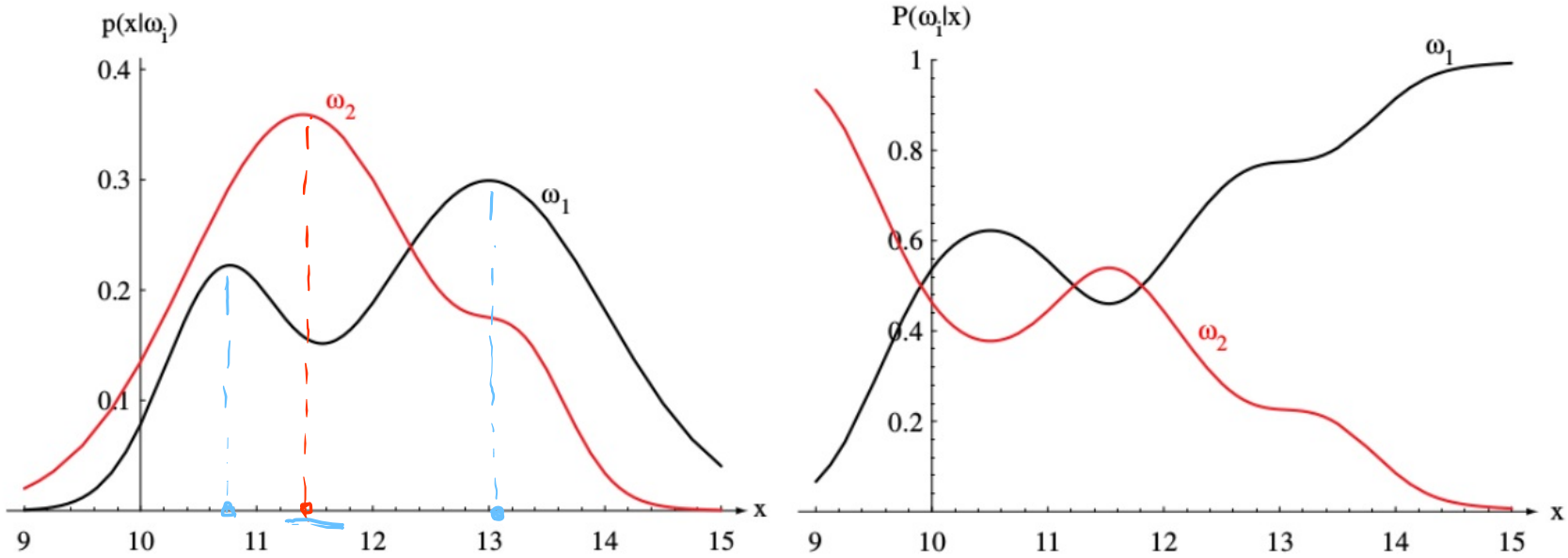


Image source: Duda et al. (2000)

Optimally Accurate Classifier

optimal decision function: $\arg\max_j P(\omega_j \mid \boldsymbol{x})$

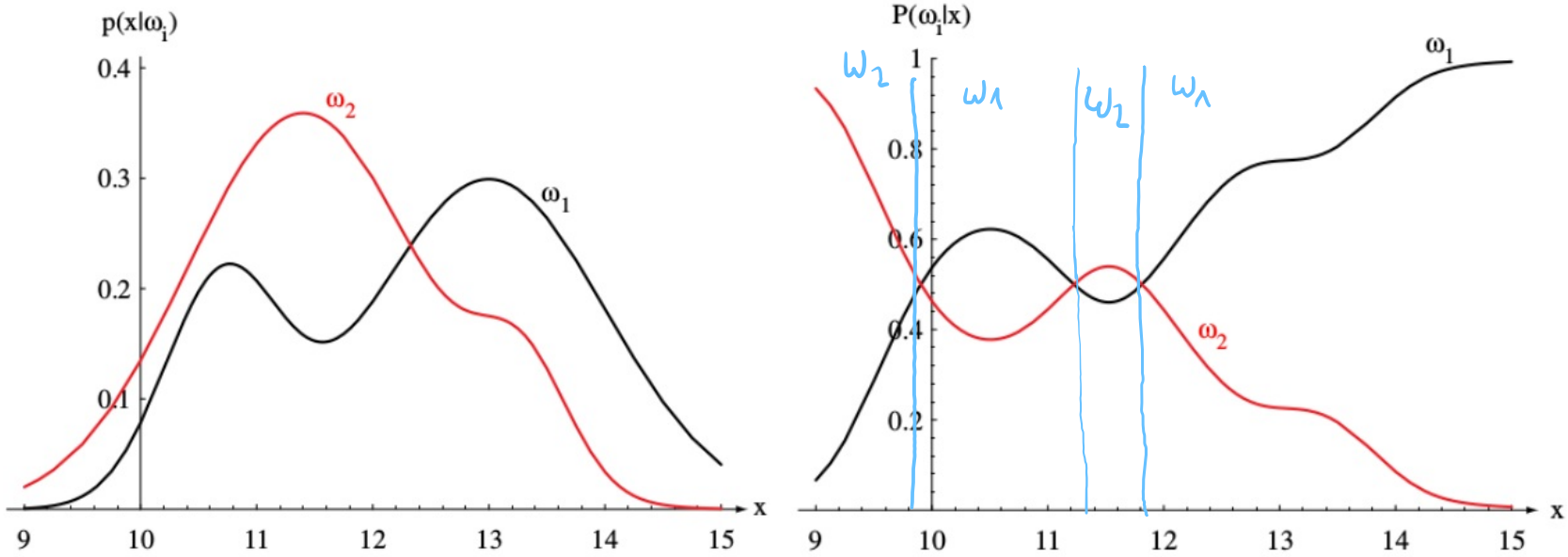


Image source: Duda et al. (2000)

Optimally Accurate Classifier

optimal decision function: $\arg\max_j P(\omega_j \mid \boldsymbol{x})$

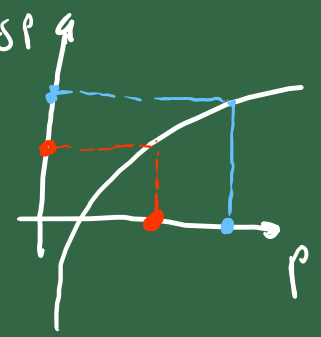
Alternate formulations of the decision:

$$= \arg\max_j \frac{p(x \mid \omega_j)\, p(\omega_j)}{p(x)}$$ Bayes Law

$$= \arg\max_j p(x \mid \omega_j)\, p(\omega_j)$$ invariance under pos. scaling

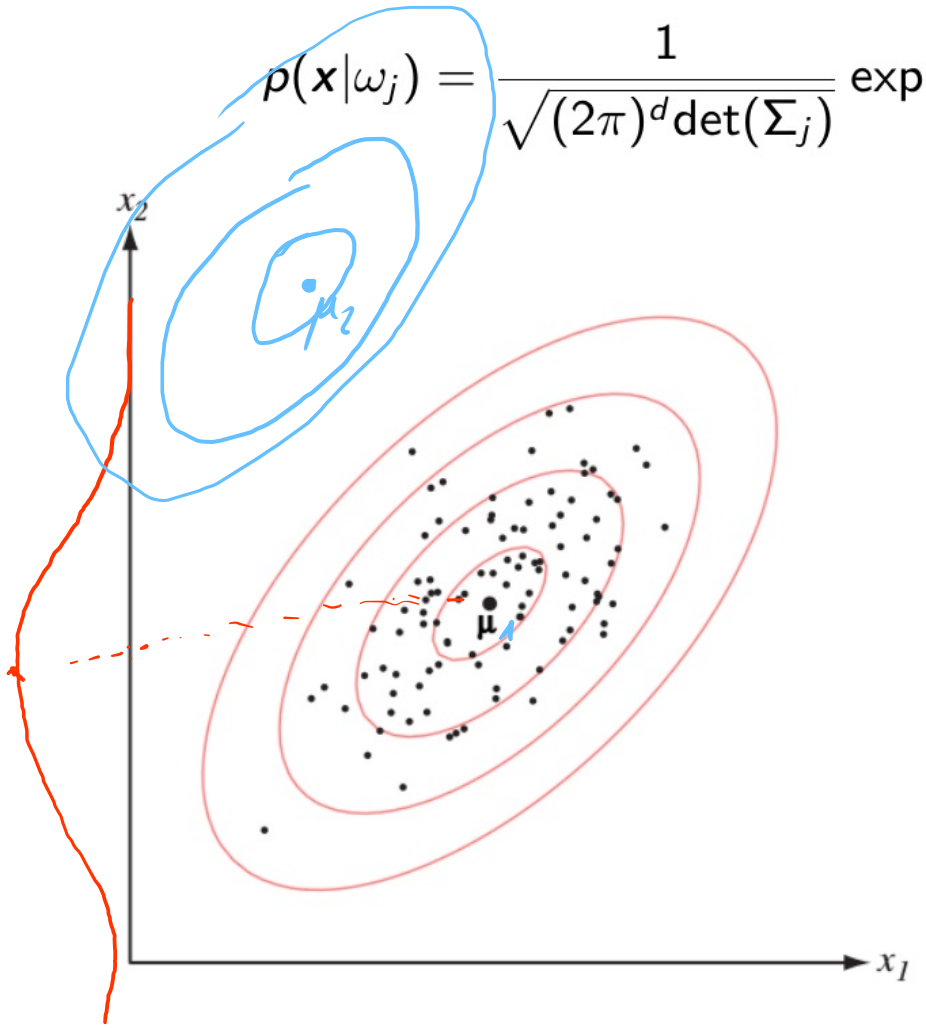
$$= \arg\max_j \log\left[p(x \mid \omega_j)\, p(\omega_j)\right]$$ " " strictly mon. fct.

$$= \arg\max_j \log p(x \mid \omega_j) + \log p(\omega_j)$$



Multivariate Normal Distributions

$$p(\boldsymbol{x}|\omega_j) = \frac{1}{\sqrt{(2\pi)^d \det(\Sigma_j)}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_j)^\top \Sigma_j^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_j)\right)$$



- $\boldsymbol{\mu}_j$ is the mean (center of the data distribution)
- Σ_j is the covariance (elongation of the data distribution and correlation between dimensions).

Image source: Duda et al. (2000)

Classifier for Gaussians ($\Sigma_1 = \Sigma_2$)

Recall: The optimal classifier is $\arg\max_j [\log p(\boldsymbol{x}|\omega_j) + \log P(\omega_j)]$, and we have the data distributions:

$$p(\boldsymbol{x}|\omega_j) = \frac{1}{\sqrt{(2\pi)^d \det(\Sigma_j)}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_j)^\top \Sigma_j^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_j)\right)$$

还有一部分log与x无关,故省略

$$= \arg\max_j \; -\frac{1}{2}(x-\mu_j)^T \Sigma^{-1}(x-\mu_j) + \log P(\omega_j)$$

$$= \arg\max_j \; -\frac{1}{2}x^T\Sigma^{-1}x + x^T\Sigma^{-1}\mu_j - \frac{1}{2}\mu_j^T\Sigma^{-1}\mu_j + \log P(\omega_j)$$

被划掉是因为和j无关

v_j $\quad$ b_j

$$= \arg\max_j \; v_j^T x + b_j$$

$$\arg\max\{100.1, 100.3, 100.5\}$$

Classifier for Gaussians ($\Sigma_1 = \Sigma_2$)

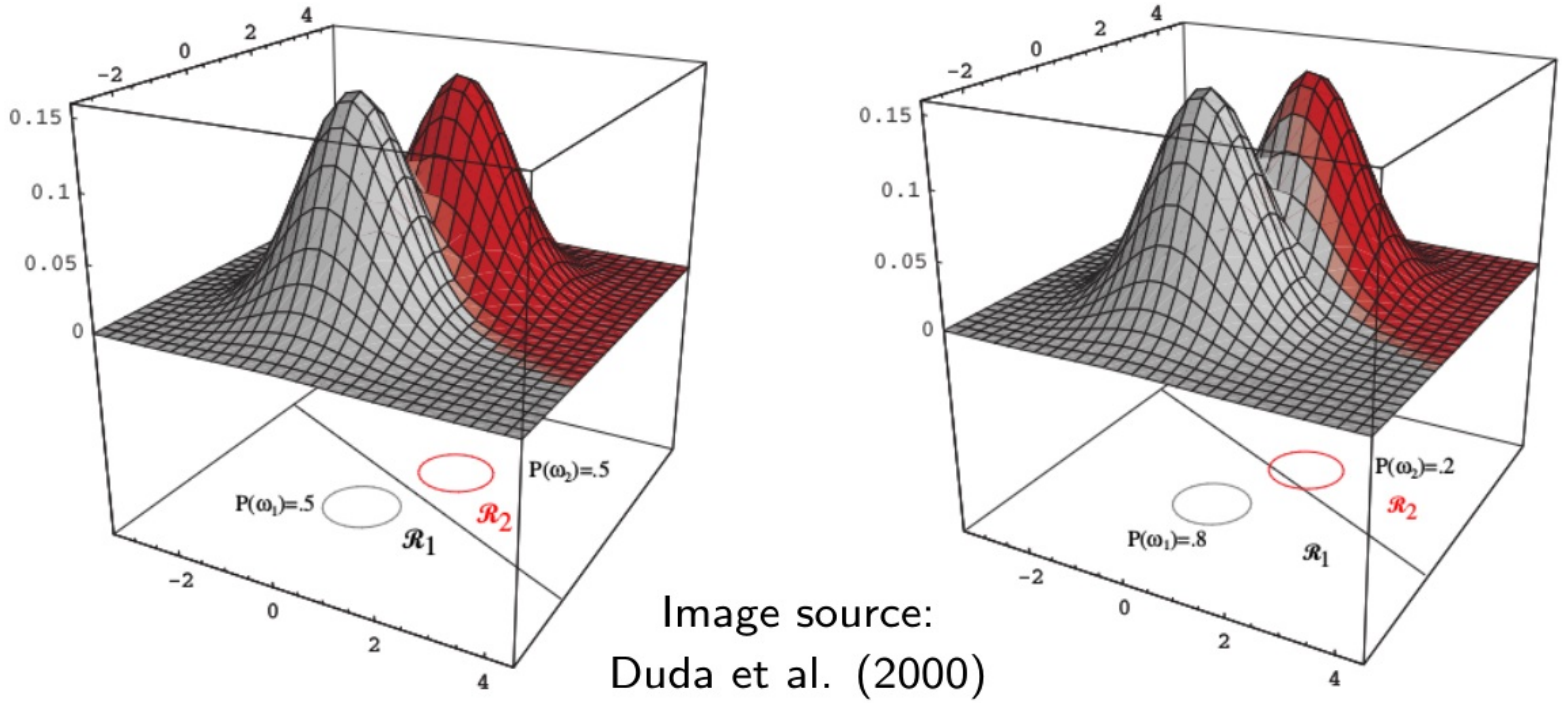
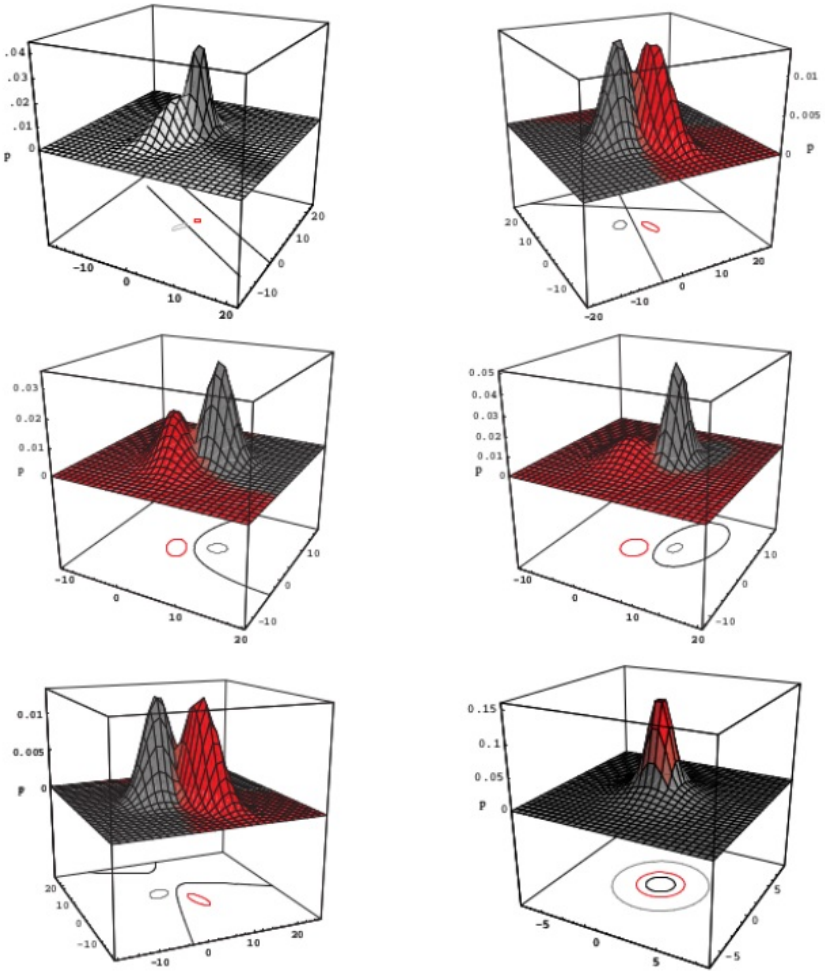


Image source:
Duda et al. (2000)

- Decision boundary is linear and oriented by mean and covariance.
- Offset is controlled by class prior probabilities.

Classifier for Gaussians ($\Sigma_1 \neq \Sigma_2$)



- When covariances Σ_1 and Σ_2 are not the same, the decision boundary is quadric instead of linear. Quadrics include circle, ellipse, parabola, hyperbola, and degenerate forms.

Image source: Duda et al. (2000)

Classifying Non-Numerical Data

Example: Spam Classifier

- *Observation:* Messages $\mathcal{M}$ that need to be handled by the spam classifier come as text and not as numerical vectors.
- *Common approach:* Represent a message $\mathcal{M}$ as a collection of binary predicates testing for typical spam words, and forming a vector $\boldsymbol{x} \in \{0,1\}^d$ containing these predicates:

$$\boldsymbol{x} = \begin{bmatrix} 1_{\{\text{“gift”} \in \mathcal{M}\}} \\ 1_{\{\text{“relief”} \in \mathcal{M}\}} \\ 1_{\{\text{“pain”} \in \mathcal{M}\}} \\ 1_{\{\text{“claim”} \in \mathcal{M}\}} \\ \vdots \end{bmatrix}.$$

Classifying Binary Data

- Assume that our data is binary, i.e. $\boldsymbol{x} \in \{0,1\}^d$, with each dimension generated *independently* according to some Bernoulli distribution:

$$P(x_i = 0 \,|\, \omega_j) = 1 - q_{ij}$$
$$P(x_i = 1 \,|\, \omega_j) = q_{ij}$$

  where q_{ij} are the parameters.

- The probability of the whole multivariate observation can be written as:

$$P(\boldsymbol{x}|\omega_j) = \prod_{i=1}^{d} \left[q_{ij}^{x_i} + (1 - q_{ij})^{(1-x_i)} \right]$$

- **Question:** How to express the optimal decision boundary

$$\arg\max_j P(\omega_j|\boldsymbol{x})$$

Classifying Binary Data

Recall: The optimal classifier is $\arg\max_j [\log p(\boldsymbol{x}|\omega_j) + \log P(\omega_j)]$, and we have the data distributions:

$$P(\boldsymbol{x}|\omega_j) = \prod_{i=1}^{d} \left[q_{ij}^{x_i} + (1 - q_{ij})^{(1-x_i)} \right]$$

$$= \arg\max_j \quad v_j^T x + b_j$$

Minimum Cost Decisions

Example: Buying a Car

- Suppose you would like to purchase a second-hand car. After observing the car (collecting a vector of measurements $\boldsymbol{x}$), you assess that it has a defect with probability

$$P(\text{defect} \mid \boldsymbol{x}) = 0.1 \qquad P(\text{no defect} \mid \boldsymbol{x}) = 0.9$$

- Concretely, the decision you need to take is *not* to *classify* the whether the car has a defect or not, but whether to *buy* the car or not.
- For this, we need to evaluate the cost of each scenario, e.g.

$$\begin{aligned} \text{cost}(\text{ buy} \mid \text{defect}) &= 100.0 \\ \text{cost}(\text{ buy} \mid \text{no defect}) &= -20.0 \\ \text{cost}(\text{ not buy} \mid \text{defect}) &= 0.0 \\ \text{cost}(\text{ not buy} \mid \text{no defect}) &= 0.0 \end{aligned}$$

and take the action with the lowest expected cost.

Minimum Cost Decisions

General problem formulation:

- Let $(\alpha_k)_k$ be the set of actions. The expected cost "λ" of taking a certain action is given by

$$\lambda(\alpha_k|\boldsymbol{x}) = \sum_{j=1}^{C} \lambda(\alpha_k|\omega_j) P(\omega_j|\boldsymbol{x})$$

  where $\lambda(\alpha_k|\omega_j)$ is the cost of taking action α_k given the class ω_j.
- The optimal action to take is therefore: $\arg\min_k \lambda(\alpha_k|\boldsymbol{x})$

Minimum Cost Decisions

General problem formulation:

- Let $(\alpha_k)_k$ be the set of actions. The expected cost "λ" of taking a certain action is given by

$$\lambda(\alpha_k|\boldsymbol{x}) = \sum_{j=1}^{C} \lambda(\alpha_k|\omega_j) P(\omega_j|\boldsymbol{x})$$

  where $\lambda(\alpha_k|\omega_j)$ is the cost of taking action α_k given the class ω_j.
- The optimal action to take is therefore: $\arg\min_k \lambda(\alpha_k|\boldsymbol{x})$

Car example: (cf. previous slide)

$\lambda(\text{buy}|x) = 100 \cdot 0.1 + (-20) \cdot 0.9 = -8$

$\lambda(\text{not buy}|x) = 0 \cdot 0.1 + 0 \cdot 0.9 = 0$

$\text{argmin}\{\text{buy}: -8, \text{not buy}: 0\} = \text{buy}$

Classification Accuracy Special Case

Recall: $\lambda(\alpha_k|\boldsymbol{x}) = \sum_{j=1}^{C} \lambda(\alpha_k|\omega_j) P(\omega_j|\boldsymbol{x})$

Show that the problem of maximum accuracy classification is a special instance of expected cost minimization with a particular set of actions $(\alpha_k)_k$ and a particular cost function $\lambda(\alpha_k|\omega_j)$.

Measuring Classification Error

- So far, we have studied what the decision boundary should be in order to predict optimally.
- However, in certain cases, it is also important to determine what is the *expected error* of the classifier (e.g. to determine whether the classifier is good enough for practical use).
- The expected error is the probability that the data is of a different class than the one predicted, e.g. for a binary classifier:

$$P(\text{Err} \mid \boldsymbol{x}) = \begin{cases} P(\omega_1 \mid \boldsymbol{x}) & \text{if} \quad \text{“decide } \omega_2\text{”} \\ P(\omega_2 \mid \boldsymbol{x}) & \text{if} \quad \text{“decide } \omega_1\text{”} \end{cases}$$

- For the Bayes optimal classifier, this reduces to

$$P(\text{Err} \mid \boldsymbol{x}) = \min\{P(\omega_1 \mid \boldsymbol{x}), P(\omega_2 \mid \boldsymbol{x})\}$$

Measuring Classification Error

- The expected error of this maximally accurate classifier is computed as the integral of its error probability over the distribution $p(\boldsymbol{x})$.

$$
\begin{aligned}
P(\text{Err}) &= \int_{\boldsymbol{x}} P(\text{Err} \mid \boldsymbol{x}) p(\boldsymbol{x}) d\boldsymbol{x} \\
&= \int_{\boldsymbol{x}} \min\{P(\omega_1 \mid \boldsymbol{x}), P(\omega_2 \mid \boldsymbol{x})\} p(\boldsymbol{x}) d\boldsymbol{x}
\end{aligned}
$$

  This is also known as the *Bayes error rate*.

- Generally, this integral cannot be solved analytically, because of the min function. Error must instead be evaluated numerically/empirically, or it can also be **bounded** analytically.

Bounding the Error of the Classifier

Very basic bound

- Observe for binary classification that $P(\omega_2 \mid \boldsymbol{x}) = 1 - P(\omega_1 \mid \boldsymbol{x})$.
- The error of an optimal binary classifier can be bounded as:

$$\begin{aligned} P(\text{Err}) &= \int_{\boldsymbol{x}} \min\{P(\omega_1 \mid \boldsymbol{x}), P(\omega_2 \mid \boldsymbol{x})\} p(\boldsymbol{x}) d\boldsymbol{x} \\ &= \int_{\boldsymbol{x}} \min\{P(\omega_1 \mid \boldsymbol{x}), 1 - P(\omega_1 \mid \boldsymbol{x})\} p(\boldsymbol{x}) d\boldsymbol{x} \\ &\leq \int_{\boldsymbol{x}} 0.5 \, p(\boldsymbol{x}) d\boldsymbol{x} \\ &= 0.5 \end{aligned}$$

  i.e. the classifier predicts the correct class at least 50% of the time.

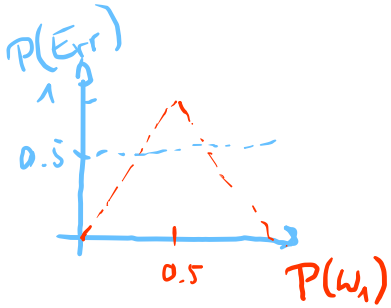
- Note that, unlike an empirical evaluation, this result is general and independent on the data distributions.

Bounding the Error of the Classifier

Another simple bound

$$P(\text{Err}) = \int_x \min\{P(\omega_1 \mid \boldsymbol{x}), P(\omega_2 \mid \boldsymbol{x})\} p(\boldsymbol{x}) d\boldsymbol{x}$$

$$= \int_x \min\{P(\omega_1 \mid \boldsymbol{x}) p(\boldsymbol{x}), P(\omega_2 \mid \boldsymbol{x}) p(\boldsymbol{x})\} d\boldsymbol{x}$$



Recall: $P(\omega_j \mid \boldsymbol{x}) p(\boldsymbol{x}) = p(\boldsymbol{x} \mid \omega_j) P(\omega_j)$

$$= \int_x \min\left\{p(\boldsymbol{x} \mid \omega_1) P(\omega_1), p(\boldsymbol{x} \mid \omega_2) P(\omega_2)\right\} d\boldsymbol{x}$$

$$\leq \int_x \min\left\{\sum_j \{p(\boldsymbol{x} \mid \omega_j)\} P(\omega_1), \sum_j \{p(\boldsymbol{x} \mid \omega_j)\} P(\omega_2)\right\} d\boldsymbol{x}$$

$$= \left(\sum_j \int_x \{p(\boldsymbol{x} \mid \omega_j)\} d\boldsymbol{x}\right) \cdot \min\{P(\omega_1), P(\omega_2)\}$$

$$= 2 \cdot \min\{P(\omega_1), P(\omega_2)\}$$

Additional insight: The optimal classifier improves its accuracy when one class prior probability is strongly dominant over to the other class.

Summary

Machine Learning

- Paradigm that provides a solution to the practically highly relevant problem of autonomous decision making.
- Avoids to the user the task of specifying the decision function at hand, and instead, infers it automatically from the data.

Bayes Decision Theory

- Framework that allows to build optimal machine learning classifiers, assuming we have full knowledge knowledge about the class probabilities and the data distributions.
- Bayesian decision theory highlights the effect of class priors, parameters of the data distribution, and specification of the cost function, on the optimal decision function and the expected cost.