Learning and Bayesian Decision Task

Radek Mařík & Petr Pošík

FEE CTU, K13132, K13133

February 28, 2017

A modified version of Petr Pošík's presentation.
Based on the book by M. I. Schlesinger and V. Hlaváč [SH99, SH02]

Outline

What is AI?

The science of making machines

- **think like people.** Not AI anymore, mix of cognitive science and computational neuroscience.
- **act like people.** No matter how they think, actions and behavior must be human-like. Dates back to Turing. But should we mimic even human errors?
- **think rationally.** Requires correct thought process. Builds on philosophy and logic: how shall you think in order not to make a mistake? Our limited ability to express the logical deduction.
- **act rationally.** Care only about what they do and if they achieve their goals optimally. Goals are described in terms of the utility of the outcomes. *Maximize the expected utility of the outcomes of their decisions.*

Good decisions:

- Take into account similar situations that happened in the past. **Machine learning.**
- Simulations using a model of the world. Be aware of the consequences of your actions and plan ahead. **Inference, planning.**

Artificial Intelligence (AI)

Studies of *intelligence in general*:

- How do we *perceive* the world?
- How do we *understand* the world?
- How do we *reason* about the world?
- How do we *predict* the consequences of our actions?
- How do we act to *influence* the world?

AI not only wants to understand the "intelligence", but also wants to

- create an intelligent entity (agent, robot)
- imitating or improving
- the human behavior and effects in the outer world, and/or
- the inner human mind processes and reasoning.

Robot vs. agent:

- very often interchangeable terms describing systems with varying degrees of autonomy able to predict the state of the world and effects of their own actions. Sometimes, however:
- **agent:** the software responsible for the "intelligence"
- **robot:** the hardware, often used as substitute for humans in dangerous situations, in poorly accessible places, or for routine repeating actions

Requirements for an Ideal Agent

Knowledge representation:

- how to store the model of the world, the relations between the entities in the world, the rules that are valid in the world, . . .

Automated reasoning:

- how to infer some conclusions from what is known or answer some questions

Planning:

- how to find an action sequence that puts the world in the desired state

Pattern recognition:

- how to decide about the state of the world based on observations

Machine learning:

- how to adapt the model of the world using new observations

Requirements for an Ideal Agent (cont.)

Multiagent systems:

- how to coordinate and cooperate in a group of agents to reach the desired goal

Natural language processing:

- how to understand what people say and how to say something to them

Computer vision:

- how to understand the observed scene, what is going on in a sequence of pictures

Robotics:

- how to move, how to manipulate with objects, how to localize and navigate

. . .

Course outline

1. The relation of artificial intelligence, pattern recognition, learning and robotics. Decision tasks, Empirical learning.
2. Linear methods for classification and regression.
3. Non-linear models. Feature space straightening. Overfitting.
4. Nearest neighbors. Kernel functions, SVM. Decision trees.
5. Bagging. Adaboost. Random forests.
6. Graphical models. Bayesian networks.
7. Markov statistical models. Markov chains.
8. Expectation-Maximization algorithm.
9. Planning. Planning problem representations. Planning methods.
10. Scheduling. Local search.
11. Constraint satisfaction problems.
12. Neural networks. Basic models and methods, error backpropagation.
13. Special neural networks. Deep learning.
14. Evolutionary algorithms (if time permits).

Joint and Conditional Probabilities

The joint probability $P(x, y)$ of two random variables x and y can be expressed as (**the product rule**)

$$P(x \wedge y) = P(x, y) = P(x|y) \cdot P(y) = P(y|x) \cdot P(x)$$

Bayes' rule (Bayes's law, Bayes' theorem)

$$P(y|x) = \frac{P(x|y)P(y)}{P(x)}$$

$$\text{posterior} = \frac{\text{likelihood} \cdot \text{prior}}{\text{evidence}}$$

Observations and States

2^{nd} World War

- Radar for detection aircraft, code-breaking, decryption.
- The task is to estimate the state but we only have a noisy, or corrupted, observation.

An **object (situation)** is described by two parameters:

- $x \in X$ which is observable; called **observation**, measurement, or *feature vector*.
- $k \in K$ which is unobservable; called **hidden state**, parameter, state-of-nature or *class*.

Decision Strategy Design

Decision Strategy

Using

- an observation $x \in X$ of an object of interest
- with a hidden state $k \in K$,
- we should design a decision strategy $q : X \to D$
- which would be optimal with respect to certain criterion.

Bayesian decision theory requires

- complete statistical information $p_{XK}(x,k)$ of the object of interest to be known,
- and a suitable penalty function $W : K \times D \to \mathbb{R}$ must be provided.

Non-Bayesian decision theory studies tasks for which some of the above information is not available.

Decision Strategy Notation Puzzle

- $X \times K \times D \times W$ by Schlesinger and Hlaváč [SH02]
  - observations X,
  - hidden states K,
  - decisions D,
  - penalty function W.
- $X \times \Omega \times A \times W$ by Duda, Hart, and Stork [DHS01],
  - observations X,
  - hidden states/classes Ω (Y),
  - decisions/actions A,
  - penalty function W.
- $E \times S \times A \times U$ by Russell and Norvig [RN10]
  - evidence E,
  - hidden states S,
  - decisions/actions A,
  - penalty function/utility U.

Definitions of Concepts - Data

An **object** of interest is characterized by the following parameters:

- **observation** $x \in X$
  - vector of numbers, graph, picture, sound, ECG, . . . , and
- **hidden state** $k \in K$.
  - k is often viewed as the object **class**, but it may be something different, e.g. when we seek for the location k of an object based on the picture x taken by a camera.

Joint probability distribution $p_{XK} : X \times K \rightarrow \langle 0, 1 \rangle$

- $p_{XK}(x, k)$ is the joint probability that the object is in the state k and we observe x.
- $p_{XK}(x, k) = p_{X|K}(x|k) \cdot p_K(k)$

Definitions of Concepts - Decision

Decision strategy (or function or rule) $q : X \to D$

- D is a set of possible decisions. (Very often $D = K$.)
- q is a function that assigns a decision $d = q(x), d \in D$, to each $x \in X$.

Penalty function (or loss function) $W : K \times D \to \mathbb{R}$ (real numbers)

- $W(k, d)$ is a penalty for decision d if the object is in state k.

Risk $R : Q \to \mathbb{R}$

- the mathematical expectation of the penalty which must be paid when using the strategy q.

Notes to decision tasks

In the following, we consider *decision tasks* where

- the decisions do not influence the state of nature
  - unlike *game theory* or *control theory*.
- a single decision is made, issues of time are ignored in the model
  - unlike *control theory*, where decisions are typically taken continuously in real time.
- the costs of obtaining the observations are not modelled
  - unlike *sequential decision theory*.

The hidden parameter k (state, class) is considered not observable. Common situations are:

- k can be observed, but at a high cost.
- k is a future state (e.g. price of gold) and will be observed later.

Classification is a special case of the decision-making problem where the set of decisions D and hidden states K coincide.

Pattern recognition task examples

The description of the concepts is very general, so far we did not specify

- what the items of the X, K, and D sets actually are,
- how they are represented.

Application	Observation (measurement)	Decisions
Coin value in a slot machine	$x \in \mathbb{R}^n$	Value
Cancerous tissue detection	Gene-expression profile, $x \in \mathbb{R}^n$	{yes, no}
Medical diagnostics	Results of medical tests, $x \in \mathbb{R}^n$	Diagnosis
Optical character recognition	2D bitmap, intensity image	Words, numbers
License plate recognition	2D bitmap, grey-level image	Characters, numbers
Fingerprint recognition	2D bitmap, grey-level image	Personal identity
Face detection	2D bitmap	{yes, no}
Speech recognition	$x(t)$	Words
Speaker identification	$x(t)$	Personal identity
Speaker verification	$x(t)$	{yes, no}
EEG, ECG analysis	$\mathbf{x}(t)$	Diagnosis
Forfeit detection	Various	{yes, no}

Two types of pattern recognition

1. **Statistical pattern recognition**
   - Objects are represented as points in a vector space.
   - The point (vector) x contains the individual observations (in a numerical form) as its coordinates.
2. **Structural pattern recognition**
   - The object observations contain a structure which is represented and used for recognition.
   - A typical example of the representation of a structure is *a grammar*.

Bayesian Decision Task

Given the sets X, K, and D, and functions $p_{XK} : X \times K \to \langle 0, 1 \rangle$ and $W : K \times D \to \mathbb{R}$, find a strategy $q : X \to D$ which minimizes the **Bayesian risk** of the strategy q

$$R(q) = \sum_{x \in X} \sum_{k \in K} p_{XK}(x,k) \cdot W(k, q(x)).$$

The optimal strategy q, denoted as q^*, is then called **the Bayesian strategy.** The Bayesian risk can be expressed as

$$\begin{aligned} R(q) &= \sum_{x \in X} \sum_{k \in K} p_{XK}(x,k) \cdot W(k, q(x)) = \\ &= \sum_{x \in X} \sum_{k \in K} p_{K|X}(k|x) \cdot p_X(x) \cdot W(k, q(x)) = \\ &= \sum_{x \in X} p_X(x) \sum_{k \in K} p_{K|X}(k|x) \cdot W(k, q(x)) = \\ &= \sum_{x \in X} p_X(x) \cdot R(q(x), x) \end{aligned}$$

Bayesian Decision Task - Partial Risk

$$R(q) = \sum_{x \in X} p_X(x) \sum_{k \in K} p_{K|X}(k|x) \cdot W(k, q(x)) =$$
$$= \sum_{x \in X} p_X(x) \cdot R(q(x), x), \text{ where}$$
$$R(d, x) = \sum_{k \in K} p_{K|X}(k|x) \cdot W(k, d)$$

is the **partial risk**, i.e. the expected penalty for decision d given the observation x. The minimization of the Bayesian risk can be formulated as

$$R(q^*) = \min_{q \in Q} R(q) = \sum_{x \in X} p_X(x) \cdot \min_{d \in D} R(d, x),$$

i.e. the Bayesian strategy can be constructed by choosing the decision d^* that minimizes the partial risk for each observation x.

Bayesian Strategy Characteristics - Determinism

Bayesian strategy can be derived for infinite X, K and/or D by replacing summation with integration and probability mass function with probability density function in the formulation of Bayesian decision task.

Bayesian strategy is deterministic.

- q provides the same decision $d = q(x)$ for the same x, despite k may be different.
- What if we used a randomized strategy q of the form $q(d|x)$, i.e. if the decision d would be chosen randomly using the probability distribution $q(d|x)$?
- The risk of the randomized strategy $q(d|x)$ is equal or greater than the risk of the well chosen deterministic Bayesian strategy $q^*(x)$.

Bayesian Strategy Characteristics - Convex Cones

Bayesian strategy divides the probability space to $|D|$ convex cones $C(d)$.

- *Probability space?* Any observation x is mapped to a point in a $|K|$-dimensional linear space with the coordinates

$$(p_{X|1}(x|1), p_{X|2}(x|2), \ldots, p_{X|k}(x|k)),$$

  i.e. the space delimited by the positive coordinates.
- *Cone?* Let S be a linear space. Any subspace $C \subset S$ is a *cone* if for each $x \in C$ also $\alpha x \in C$ for any real number $\alpha > 0$.
- *Convex cone?* For any 2 points $x_1 \in C$ and $x_2 \in C$, and for any point x lying on the line between x_1 and x_2, also $x \in C$.

$$x = \alpha_1 \cdot x_1 + \alpha_2 \cdot x_2, \quad \alpha_1 + \alpha_2 = 1, \alpha_i \geq 0$$

- The decisions $C(d)$ are linearly separable!!!

Two Special Cases of the Bayesian Decision Task I

Minimization of the probability of the incorrect estimation of the actual hidden state k^* (i.e. minimization of classification error)

- The task is to decide the object state $d = q(x) = k$, i.e. $D = K$.
- The goal is to minimize $Pr(q(x) \neq k^*)$.
- $Pr(q(x) \neq k^*) = R(q)$ if a unit penalty

$$W(k^*, q(x)) = \begin{cases} 0 & \text{if } q(x) = k^*, \\ 1 & \text{otherwise.} \end{cases}$$

- In this case:

$$\begin{aligned} q(x) &= \arg\min_{k \in K} \sum_{k^* \in K} p_{XK}(x, k^*) W(k^*, k) = \\ &= \arg\max_{k \in K} p_{K|X}(k|x), \end{aligned} \qquad (1)$$

  i.e. compute *posterior probabilities* of all states k given the observation x, and decide for the most probable state.
- **Maximum a posterior (MAP) estimation.**

Two Special Cases of the Bayesian Decision Task II

Bayesian strategy with the `dontknow` decision

- Using the partial risk $R(d,x) = \sum_{k \in K} p_{K|X}(k|x) \cdot W(k,d)$, for each observation x, we shall provide the decision d minimizing $R(d,x)$.
- But even this optimal $R(d,x)$ may not be sufficiently low, i.e. x does not convey sufficient information for a low-risk decision.
- Let's use $D = K \cup \{\texttt{dontknow}\}$ and define

$$W(k,d) = \begin{cases} 0 & \text{if } d = k, \\ 1 & \text{if } d \neq k \text{ and } d \neq \texttt{dontnow} \\ \epsilon & \text{if } d = \texttt{dontknow}. \end{cases}$$

- In this case:

$$q(x) = \begin{cases} \arg\max_{k \in K} p_{K|X}(k|x) \\ \qquad \text{if } \max_{k \in K} p_{K|X}(k|x) > 1 - \epsilon, \\ \texttt{dontknow} \\ \qquad \text{if } \max_{k \in K} p_{K|X}(k|x) \leq 1 - \epsilon. \end{cases}$$

Limitations of the Bayesian Approach

To use the Bayesian approach we need to know:

1. The penalty function W.
2. The *a priori* probabilities of states $p_K(k)$.
3. The conditional probabilities of observations $p_{X|K}(x|k)$.

Penalty function:

- Important: $W(k, d) \in R$
- We cannot use the Bayesian formulation for tasks where identifying the penalties with R substantially deforms the task, i.e. *when the penalties cannot be measured in (or easily transformed to) the same units.*
- How do you compare the following penalties:
    - games, fairy tales:
      loose your horse vs. loose your sword vs. loose your fiancee
    - system diagnostics, health diagnosis:
      false alarm (costs you some money) vs. overlooked danger (may cost you a human life)
    - judicial error:
      to convict an innocent (huge harm for 1 innocent person) vs. to free a killer (potential harm to many innocent persons)

Limitations of the Bayesian Approach (cont.)

Prior probabilities of states:

- Probabilities $p_K(k)$
  - may be unknown (then we can determine them by further study), or
  - may not exist at all (if the state k is not random).
- E.g. we observe a plane x and we want to decide if it is an enemy aircraft or not.
  - $p_{X|K}(x|k)$ may be quite complex, but known (it at least exists).
  - $p_K(k)$, however, do not exist
    - the frequency of enemy plane observation does not converge to any number.

Conditional probabilities of observations:

- Again, probabilities $p_{X|K}(x|k)$ may not be known or may not exist.
- E.g. if we want to decide what characters are on paper cards written by several persons, the observation x of the state k is influenced by an unobservable non-random intervention—by the writer z.
  - We can only talk about $p_{X|K,Z}(x|k,z)$, not about $p_{X|K}(x|k)$.
  - If Z was random and if we knew $p_Z(z)$, than we could compute also $p_{X|K}(x|k)$.

Decision Strategy in Practical Applications

Typically, none of the probabilities are known! The designer is only provided with the **training (multi)set** T of examples.

$T = \{(x_1, k_1), (x_2, k_2), \ldots, (x_\ell, k_\ell)\} = \bigcup_{k_j \in K} T_{k_j}, \quad x_i \in X, k_i \in K$

- It is simpler to provide good examples than to gain complete or partial statistical model, build general theories, or create explicit descriptions of concepts (hidden states).
- The aim is to find definitions of concepts (classes, hidden states) which are
  - complete (all positive examples are satisfied), and
  - consistent (no negative examples are satisfied).
- The training (multi)set is *finite*, the found concept description is only a *hypothesis*.

When do we need to use learning?

- When knowledge about the recognized object is insufficient to solve the pattern recognition task.
- Most often, we have insufficient knowledge about $p_{X|K}(x|k)$.

Types of Feedback in Learning I

Supervised learning: (with a teacher)

- A training multi-set of examples is available. Correct answers (hidden state, class, the quantity we want to predict) are **known** for all observations.
  - **Classification:** the answers (the output variable of the model) are **nominal**, i.e. the value specifies a class ID.
    - predict spam/ham based on email contents,
    - predict 0/1/. . . /9 based on the image of the number, etc.
  - **Regression:** the answers (the output variable of the model) are **quantitative**, often **continuous**
    - predict temperature in Prague based on date and time,
    - predict height of a person based on weight and gender, etc.

Types of Feedback in Learning II

Unsupervised learning: (clustering)

- A training multi-set of examples is available. Correct answers are **not known**, they must be sought in data itself $\Rightarrow$ data analysis.

Semisupervised learning:

- A training multi-set of examples is available. Correct answers are **known only for a subset** of the training set.

Reinforcement learning:

- A training multi-set of examples is **not available**. Correct answers, or rather **rewards** for good decisions in the past, *are given occasionally after decisions are taken*.

Learning as Parameter Estimation

1. **Assume** $p_{XK}(x,k) = p_{XK|\Theta}(x,k|\boldsymbol{\theta})$ has a particular form with a small number of parameters Θ.
   - e.g. Gaussian, mixture of Gaussians, piece-wise constant
2. **Estimate** the values of parameters $\boldsymbol{\Theta}$ using the training set T.
3. **Solve** the classifier design problem as if the estimated $\hat{p}_{XK}(x,k) = p_{XK|\Theta}(x,k|\hat{\boldsymbol{\theta}})$ was the true (and unknown) $p_{XK}(x,k)$.

Pros and cons:

- If the true $p_{XK}(x,k)$ does not have the assumed form, the resulting strategy $q'(x)$ can be **arbitrarily bad**, even if the training set size $|T|$ approaches infinity. (GIGO effect)
- Implementation is often straightforward, especially if the parameters Θ_k are assumed to be **independent** for each class (**naive bayes classifier**).

Optimal Strategy Selection

- Choose a class Q of strategies $q_\Theta : X \to D$. The class Q is usually given as a set of parametrized strategies of the same kind.
- The problem can be formulated as a non-Bayesian task with non-random interventions:
    - The unknown parameters θ_k are the non-random interventions.
    - The probabilities $p_{X|K,\Theta}(x|k,\theta_k)$ must be known.
    - The solution may be e.g. such a strategy that minimizes the maximal probability of incorrect decision over Θ, i.e. strategy that minimizes the probability of incorrect decision in case of the worst possible parameter settings.
    - But even this minimal probability may not be low enough
        - this happens especially in cases when the class Q of strategies is too broad.
    - It is necessary to narrow the set of possible strategies using additional information
        - **the training (multi)set** T.

Learning as Optimal Strategy Selection

Learning amounts to **selecting a particular strategy q_{Θ^*} from the *a priori* known set** Q using information provided as training set T.

- A **natural criterion** for the selection of one particular strategy is the risk $R(q_\Theta)$, but it cannot be computed because $p_{XK}(x,k)$ is unknown.

$$R(q_\Theta) = \sum_{k \in K} \sum_{x \in X} p_{XK}(x,k) W(k, q_\Theta(x)), \quad q_\Theta \in Q$$

- The strategy $q_{\Theta^*} \in Q$ is chosen by minimizing some other **surrogate criterion** on the training set which approximates $R(q_\Theta)$.
- The choice of the surrogate criterion determines the *learning paradigm*.

All the following surrogate criteria can be computed using the training data T.

Surrogate Criteria - Maximum Likelihood

Learning as parameter estimation according to the **maximum likelihood**

- The likelihood of an instance of the parameters $\boldsymbol{\theta} = (\theta_k : k \in K)$ is the **probability of T given $\boldsymbol{\theta}$** (T as a multiset):

$$L(\boldsymbol{\theta}) = p(T|\boldsymbol{\theta}) = \prod_{(x_i,k_i)\in T} p_{XK|\Theta}(x_i, k_i|\boldsymbol{\theta}) = \prod_{(x_i,k_i)\in T} p_K(k_i) p_{X|K,\Theta}(x_i|k_i, \theta_{k_i})$$

- Learning then means to find $\boldsymbol{\theta}^*$ that maximizes the probability of T:

$$\boldsymbol{\theta}^* = (\theta_k^* : k \in K) = \arg\max_{\boldsymbol{\theta}} L(\boldsymbol{\theta})$$

  which can be decomposed to

$$\theta_k^* = \arg\max_{\theta_k} \sum_{x\in X(T_k)} \alpha(x,k) \log p_{X|K}(x|k,\theta_k),$$

  where $\alpha(x,k)$ is the frequency of the pair (x,k) in T
  - T, T_k are multisets, $X(T_k) \subseteq X$ is a set. Note: $x \in X(T_k) \approx x \in T_k$
- The recognition is then performed according to $q_{\theta^*}(x) = q_\Theta(x, \theta^*)$.

Surrogate Criteria - Non-random Training Set

Learning as parameter estimation according to a **non-random training set**.

- When random samples are not easy to obtain, e.g. in recognition of images.
- T is carefully crafted by the designer:
  - it should cover the whole recognized domain
  - the examples should be typical ("quite probable") prototypes
- Let $T_k, k \in K$, be a subset of the training set T with examples for state k.
- A strategy treating each $(x_i, k_i) \in T$ as **a quite probable representative** of the k-class (i.e. a maximization of the worst case probability). Then for all $k \in K$

$$\theta_k^* = \arg\max_{\theta_k} \min_{x \in T_k} p_{X|K,\Theta}(x|k, \theta_k)$$

- Note that the $\boldsymbol{\theta}^*$ does not depend on the frequencies of (x, k) in T
  - T is a set.

An Example

Multi-dimensional Gaussian Distributions

$$p(x|k,\mu_k) = \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}} \exp(\frac{-(x_i - \mu_{ik})^2}{2})$$

- **the maximal likelihood**

$$\hat{\mu}_k^* := (1/\ell) \sum_{i=1}^{\ell} x_i$$

- **the non-random training set** - the μ_k^* is estimated as the centre of the smallest circle containing all vectors which were selected by the teacher as rather good representatives of objects in the k-th state.
  - complexity? ($\Theta(N)$)

Surrogate Criteria - Minimization of the Empirical Risk

Learning as optimal strategy selection by **minimization of the empirical risk**.

- The set Q of parametrized strategies $q_\Theta \colon X \to D$, penalty function $W \colon K \times D \to \mathbb{R}$.
- The quality of each strategy $q_\theta \in Q$ could be described by the risk (i.e. the quality of each parameter set $\boldsymbol{\theta}$)

$$R(\boldsymbol{\theta}) = R(q_{\boldsymbol{\theta}}) = \sum_{k \in K} \sum_{x \in X} p_{XK}(x,k) W(k, q_\Theta(x, \boldsymbol{\theta})),$$

  but p_{XK} is unknown.
- We thus use the **empirical risk** R_{emp} (training set error):

$$R_{\text{emp}}(\boldsymbol{\theta}) = R_{\text{emp}}(q_{\boldsymbol{\theta}}) = \frac{1}{|T|} \sum_{(x_i, k_i) \in T} W(k_i, q_\Theta(x_i, \boldsymbol{\theta})).$$

- Strategy $q_{\boldsymbol{\theta}^*}(x) = q_\Theta(x, \boldsymbol{\theta}^*)$ is used where $\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} R_{\text{emp}}(\boldsymbol{\theta})$.
- Examples: Perceptron, neural networks (backprop.), classification trees, …

Surrogate Criteria - Minimization of the Structural Risk

Learning as optimal strategy selection by **minimization of the structural risk**.

- Based on Vapnik-Chervonenkis theory
- Examples: Optimal separating hyperplane, support vector machine (SVM)

Learning Revisited

Do we need learning? When?

- If we are about to solve one particular task which is **sufficiently known** to us, we should try to develop a recognition method *without learning*.
- If we are about to solve a task belonging to a well defined class, develop a recognition method *with learning*.
    - we only do not know which particular task from the class we shall solve

The designer

- should understand all the varieties of the task class, i.e.
- should find a solution to the whole class of problems.

The solution

- is a parametrized strategy and
- its parameters are learned from the training (multi)set.

The *supervised learning* is a topic for several upcoming lectures:

- Decision trees and decision rules.
- Linear classifiers.
- Adaboost.

Summary

Learning:

- Needed when we do not have sufficient statistical info for recognition.
- There are several types of learning differing in the types of information the learning process can use.

Approaches to learning:

- Assume p_{XK} has a certain form and use T to estimate its parameters.
- Assume the right strategy is in a particular set and use T to choose it.
- There are several learning paradigms depending on the choice of criterion used instead of Bayesian risk.

Non-Bayesian Decision Tasks

When?

- Tasks where W, p_K, or $p_{X|K}$ are not known.
- Even if all the events are random and all probabilities are known, it is sometimes helpful to approach the problem as a non-Bayesian task.
- In practical tasks, it can be more intuitive for the customer to express the desired strategy properties as allowed rates of false positives (false alarm) and false negatives (overlooked danger).

Special cases of practically useful non-Bayesian formulations

There are several ones for which the solution is known:

- The strategies that solve these non-Bayesian tasks are of the same form as Bayesian strategies
    - **they divide the probability space to a set of convex cones**.
- These non-Bayesian tasks can be formulated as linear programs and **solved by linear programming methods**.

Many other non-Bayesian tasks for which the solution is not known yet.

Neyman-Pearson Task

Situation:

- Observation $x \in X$, states $k = 1$ (normal), $k = 2$ (dangerous), $K = \{1, 2\}$.
- The probability distribution $p_{X|K}(x|k)$ exists and is known.
- Given the observation x, the task is to decide k, i.e. if the object is in normal or dangerous state.
- The set X is to be divided to 2 subsets X_1 and X_2, $X = X_1 \cup X_2$.
- In this formulation, $p_K(k)$ and $W(k, d)$ is not needed.

Each strategy q is characterized by 2 numbers:

- The conditional probability of false positive (false alarm):

$$\omega(1) = \sum_{x \in X_2} p_{X|K}(x|1)$$

- The conditional probability of false negative (overlooked danger):

$$\omega(2) = \sum_{x \in X_1} p_{X|K}(x|2)$$

Neyman-Pearson Task (cont.)

Neyman-Pearson task formulation:
Find a strategy q, i.e. a decomposition of X into X_1 and X_2, such that

- the probability of overlooked danger (FN) is not larger than a predefined value ϵ, i.e.

$$\omega(2) = \sum_{x \in X_1} p_{X|K}(x|2) \le \epsilon,$$

- and the probability of false alarm (FP) is minimal, i.e.

$$\text{minimize } \omega(1) = \sum_{x \in X_2} p_{X|K}(x|1),$$

- under the additional conditions

$$X_1 \cap X_2 = \emptyset, X_1 \cup X_2 = X.$$

Solution: The optimal strategy q^* separates X_1 and X_2 according to the *likelihood ratio*:

$$q^*(x) = \begin{cases} 1 & \text{iff } \frac{p_{X|K}(x|1)}{p_{X|K}(x|2)} > \theta, \\ 2 & \text{iff } \frac{p_{X|K}(x|1)}{p_{X|K}(x|2)} < \theta. \end{cases}$$

Minimax Task

Situation:

- Observation $x \in X$, states $k \in K$.
- $q : X \to K$ — given the observation x, the strategy decides the object state k.
- The set X is to be divided to $|K|$ subsets $X_1, \dots, X_{|K|}$, $X = X_1 \cup \dots \cup X_{|K|}$.
- Again, $p_K(k)$ and $W(k, d)$ are not required.

Each strategy is described by $|K|$ numbers

$$\omega(k) = \sum_{x \notin X_k} p_{X|K}(x|k),$$

i.e. by the conditional probabilities of a wrong decision under the condition that the true hidden state is k.

Minimax Task (cont.)

Minimax task formulation: Find a strategy q^* which minimizes

$$\max_{k \in K} \omega(k)$$

Solution:

- The solution is of the same form as the Bayesian strategies.
- The solution for the $|K| = 2$ case is similar to the Neyman-Pearson task, with the exception that in minimax task the probability of FN cannot be controlled explicitly.

Wald Task

Motivation:

- The Neyman-Pearson task is asymmetric: the prob. of FN is controlled explicitly, while the probability of FP is minimized (but can be quite high).
- Can we find a strategy for which *both* the error probabilities would not exceed a predefined ϵ? No, the demands often cannot be accomplished in the same time.

Wald Task (cont.I) - Wald's Relaxation

- Decompose X into X_1, X_2, and X_0 corresponding to $D = K \cup \{\texttt{dontknow}\}$.
- Strategy of this form is characterized by 4 numbers:
  - the conditional prob. of a wrong decision about the state k,

$$\omega(1) = \sum_{x \in X_2} p_{X|K}(x|1) \quad \text{and} \quad \omega(2) = \sum_{x \in X_1} p_{X|K}(x|2),$$

  - the conditional prob. of the $\texttt{dontknow}$ decision when the object state is k,

$$\chi(1) = \sum_{x \in X_0} p_{X|K}(x|1) \quad \text{and} \quad \chi(2) = \sum_{x \in X_0} p_{X|K}(x|2).$$

- The requirements $\omega(1) \le \epsilon$ and $\omega(2) \le \epsilon$ are no longer contradictory for an arbitrarily small $\epsilon > 0$, since the strategy $X_0 = X, X_1 = \emptyset, X_2 = \emptyset$ is plausible.
- Each strategy fulfilling $\omega(1) \le \epsilon$ and $\omega(2) \le \epsilon$ is then characterized by how often the strategy refuses to decide, i.e. by the number $\max(\chi(1), \chi(2))$.

Wald Task (cont.II)

Wald task formulation:
Find a strategy q^* which minimizes

$$\max(\chi(1), \chi(2))$$

subject to conditions $\omega(1) \leq \epsilon$ and $\omega(2) \leq \epsilon$.

Solution: The optimal decision is based on the likelihood ratio and 2 thresholds $\theta_1 > \theta_2$:

$$q^*(x) = \begin{cases} 1 & \text{iff } \frac{p_{X|K}(x|1)}{p_{X|K}(x|2)} > \theta_1, \\ 2 & \text{iff } \frac{p_{X|K}(x|1)}{p_{X|K}(x|2)} < \theta_2, \\ \texttt{dontknow} & \text{otherwise.} \end{cases}$$

In [SH02], the generalization for $|K| > 2$ is also given.

Linnik Tasks

a.k.a. statistical decisions with non-random interventions
a.k.a. evaluations of complex hypotheses.

Previous non-Bayesian tasks did not require

- the a priori probabilities of the states $p_K(k)$, and
- the penalty function $W(k, d)$ to be known.

In Linnik tasks,

- the conditional probabilities $p_{X|K}(x|k)$ do not exist,
- the a priori probabilities $p_K(k)$ may exist (it depends on the fact if the state k is a random variable or not),
- but the conditional probabilities $p_{X|K,Z}(x|k, z)$ do exist, i.e. the random observation x depends not only on the (random or non-random) object state k, but also on a non-random intervention z.

Linnik Tasks (cont.)

Goal:

- find a strategy that minimizes the probability of incorrect decision in case of the worst intervention z.

See examples in [SH02].

Summary of PR

- The aim of PR is to design decision strategies (classifiers) which
  - given an observation x of an object with a hidden state k
  - provide a decision d such that this decision making process is optimal with respect to a certain criterion.
- If the statistical properties of (x, k) are completely known, and if we are able to design a suitable penalty function $W(k, d)$, we should solve the task in the *Bayesian framework* and search for the *Bayesian strategy* which optimizes the *Bayesian risk* of the strategy.
  - The minimization of the probability of an error is a special case, the resulting Bayesian strategy decides for the state with the *maximum a posteriori probability*.
- If the statistical properties are known only partially, or are not known at all, or if a reasonable penalty function cannot be constructed, we face a *non-Bayesian task*.
  - Several practically important special cases of non-Bayesian tasks are well-analyzed and solved (Neyman-Pearson, minimax, Wald, . . .).
  - There are plenty of non-Bayesian tasks we can say nothing about.

References I

Richard O. Duda, Peter E. Hart, and David G. Stork.
Pattern Classification.
Wiley, New York, 2 edition, 2001.

Stuart J. Russell and Peter Norvig.
Artificial Intelligence, A Modern Approach.
Pre, third edition, 2010.

Michail I. Schlesinger and Václav Hlaváč.
Deset pednáek z teorie statistického a strukturálního rozpoznávání.
VUT, 1999.

Michail I. Schlesinger and Václav Hlaváč.
Ten Lectures on Statistical and Structural Pattern Recognition.
Springer Science, 2002.