Chapter ML:I

Examples of Learning Tasks

Car Shopping Guide

?

What criteria form the basis of a decision?

Examples of Learning Tasks

Risk Analysis for Credit Approval

Customer 1	
house owner	yes
income (p.a.)	51 000 EUR
repayment (p.m.)	1 000 EUR
credit period	7 years
SCHUFA entry	no
age	37
married	yes
...	

...

Customer n	
house owner	no
income (p.a.)	55 000 EUR
repayment (p.m.)	1 200 EUR
credit period	8 years
SCHUFA entry	no
age	?
married	yes
...	

Examples of Learning Tasks

Risk Analysis for Credit Approval

Customer 1	
house owner	yes
income (p.a.)	51 000 EUR
repayment (p.m.)	1 000 EUR
credit period	7 years
SCHUFA entry	no
age	37
married	yes
...	

...

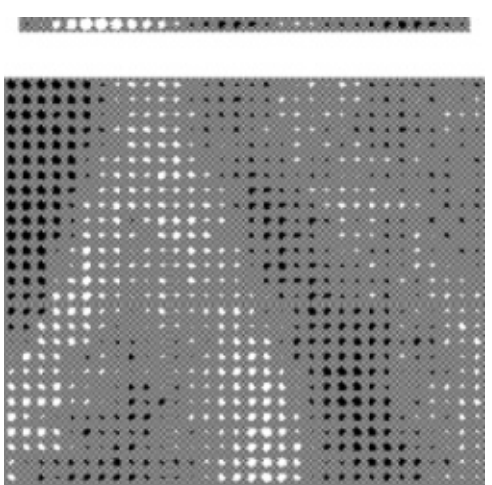
Customer n	
house owner	no
income (p.a.)	55 000 EUR
repayment (p.m.)	1 200 EUR
credit period	8 years
SCHUFA entry	no
age	?
married	yes
...	

Learned rules:

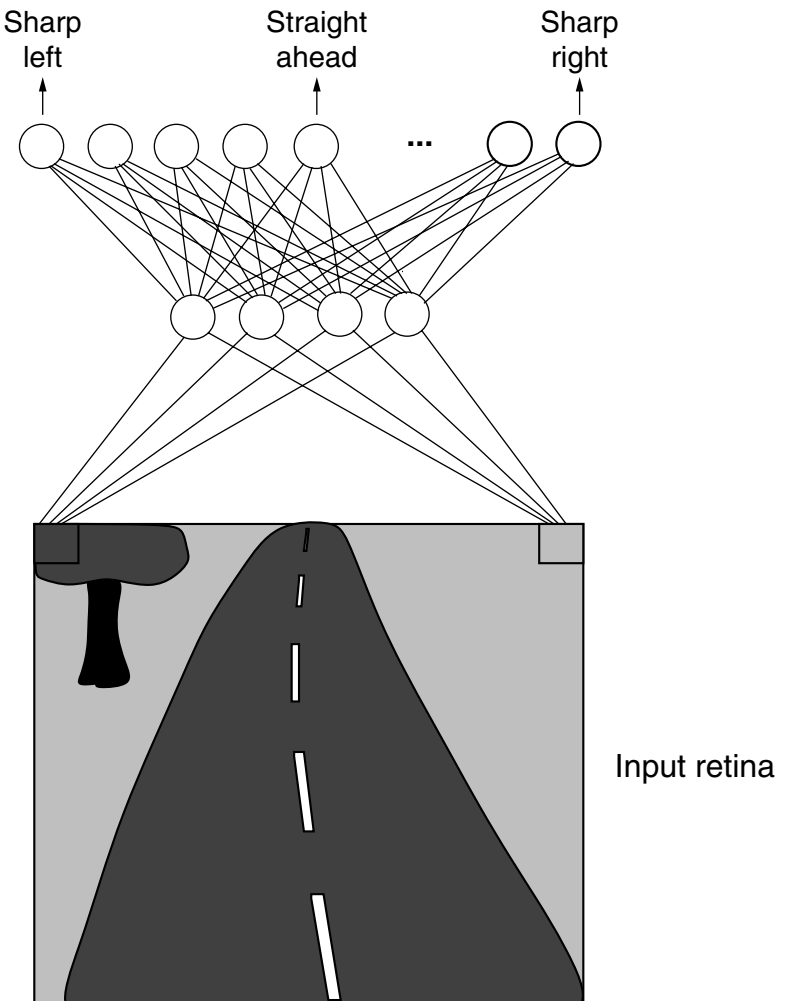
```
IF    ( income>40 000 AND credit_period<3 )  OR  house_owner=yes
THEN  credit_approval=yes

IF    SCHUFA_entry=yes  OR  ( income<20 000 AND repayment>800 )
THEN  credit_approval=no

...
```

Examples of Learning Tasks

Image Analysis [Mitchell 1997]

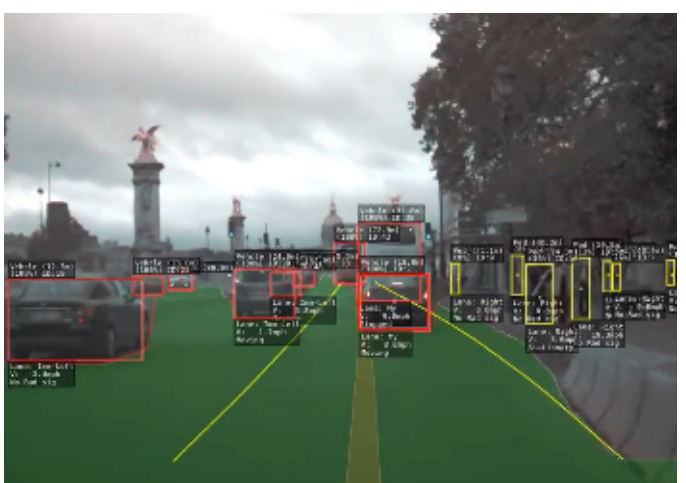


[1992]

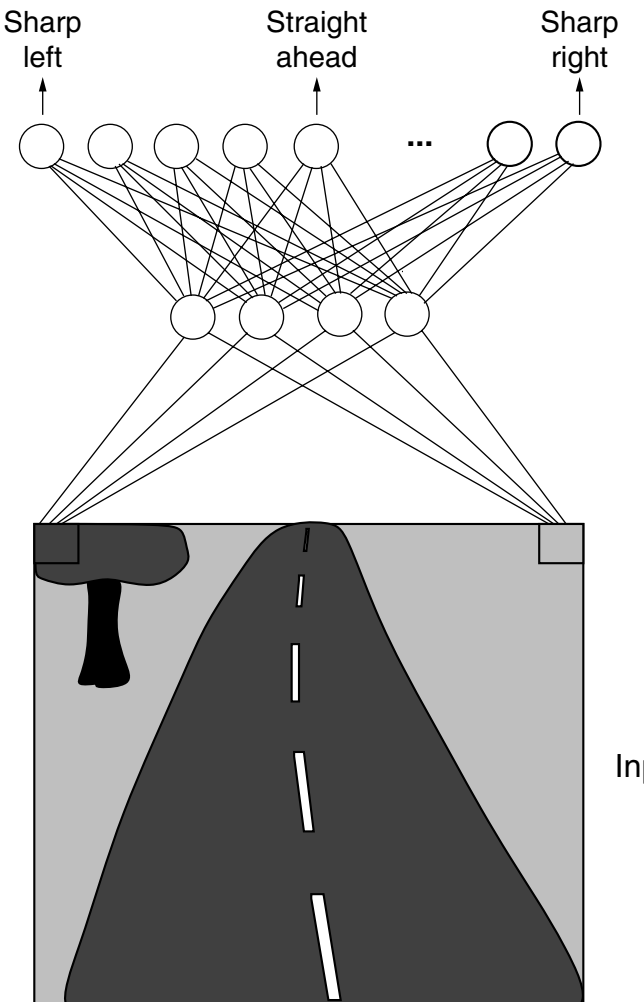


Examples of Learning Tasks

Image Analysis [Mitchell 1997]



[2018]



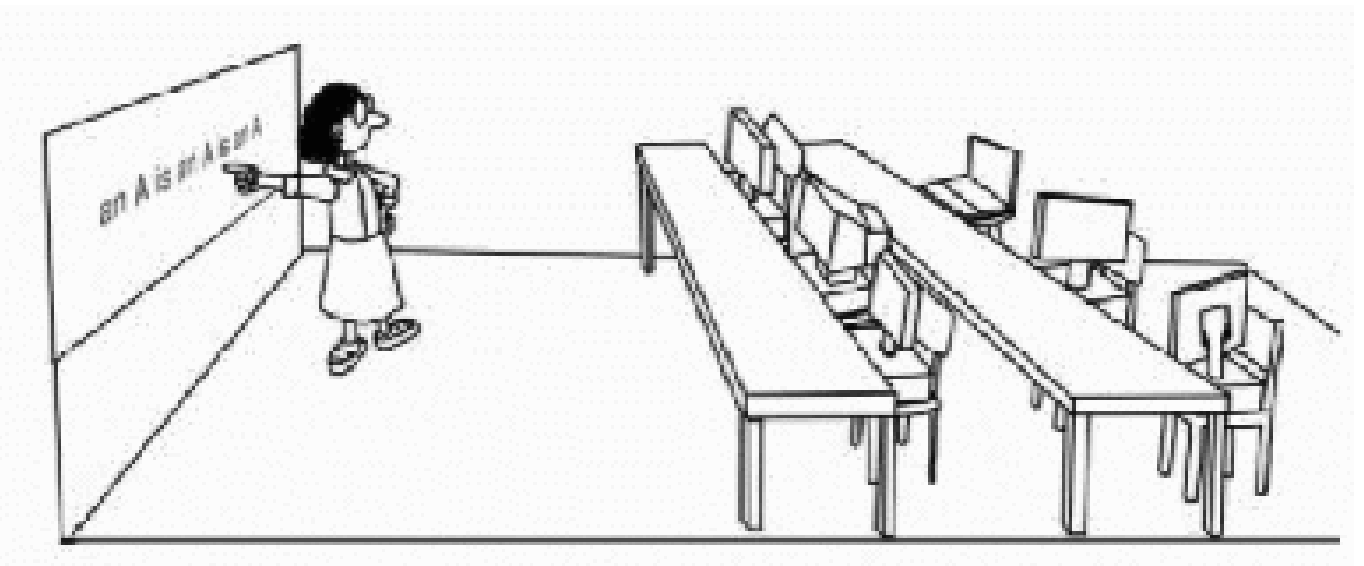
Specification of Learning Tasks

Definition 1 (Machine Learning [Mitchell 1997]**)**

A computer program is said to learn

- from experience
- with respect to some class of tasks and
- a performance measure,

if its performance at the tasks improves with the experience.



Remarks:

- Example: chess
  - task = playing chess
  - performance measure = number of games won during a world championship
  - experience = possibility to play against itself
- Example: optical character recognition
  - task = isolation and classification of handwritten words in bitmaps
  - performance measure = percentage of correctly classified words
  - experience = collection of correctly classified, handwritten words
- A data set (a corpus) with labeled examples forms a kind of “compiled experience”.
- Consider the different data sets that are developed and exploited for different learning tasks in the Webis group. [webis.de/data.html]

Specification of Learning Tasks

Learning Paradigms

1. Supervised learning

   Learn a function from a set of input-output-pairs. An important branch of supervised learning is automated classification.

   Example: optical character recognition

2. Unsupervised learning

   Identify structures in data. Important subareas of unsupervised learning include automated categorization (e.g. via cluster analysis), parameter optimization (e.g. via expectation maximization), and feature extraction (e.g. via factor analysis).

   Example: intrusion detection in a network data stream

3. Reinforcement learning

   Learn, adapt, or optimize a behavior strategy in order to maximize the own benefit by interpreting feedback that is provided by the environment.

   Example: development of behavior strategies in a hostile environment.

Specification of Learning Tasks

Learning Paradigms

1. Supervised learning

   Learn a function from a set of input-output-pairs. An important branch of supervised learning is automated classification.

   Example: optical character recognition

2. Unsupervised learning

   Identify structures in data. Important subareas of unsupervised learning include automated categorization (e.g. via cluster analysis), parameter optimization (e.g. via expectation maximization), and feature extraction (e.g. via factor analysis).

   Example: intrusion detection in a network data stream

3. Reinforcement learning

   Learn, adapt, or optimize a behavior strategy in order to maximize the own benefit by interpreting feedback that is provided by the environment.

   Example: development of behavior strategies in a hostile environment.

Specification of Learning Tasks

Example Chess: Kinds of Experience [Mitchell 1997]

1. Feedback
   - direct: for each board configuration the best move is given.
   - indirect: only the final result is given after a series of moves.

Specification of Learning Tasks

Example Chess: Kinds of Experience [Mitchell 1997]

1. Feedback
   - direct: for each board configuration the best move is given.
   - indirect: only the final result is given after a series of moves.

2. Sequence and distribution of examples
   - A teacher presents important example problems along with a solution.
   - The learner chooses from the examples; e.g., picks a board for which the best move is unknown.

The selection of examples to learn from should follow the (expected) distribution of future problems.

Specification of Learning Tasks

Example Chess: Kinds of Experience [Mitchell 1997]

1. Feedback
   - direct: for each board configuration the best move is given.
   - indirect: only the final result is given after a series of moves.

2. Sequence and distribution of examples
   - A teacher presents important example problems along with a solution.
   - The learner chooses from the examples; e.g., picks a board for which the best move is unknown.

   The selection of examples to learn from should follow the (expected) distribution of future problems.

3. Relevance under a performance measure
   - How far can we get with experience?
   - Can we master situations in the wild?
     (playing against itself may not be enough to become world class)

Specification of Learning Tasks

Example Chess: Ideal Target Function γ [Mitchell 1997]

a) $\gamma : \mathit{Boards} \rightarrow \mathit{Moves}$

b) $\gamma : \mathit{Boards} \rightarrow \mathbf{R}$

Specification of Learning Tasks

Example Chess: Ideal Target Function γ [Mitchell 1997]

a) $\gamma : \mathit{Boards} \rightarrow \mathit{Moves}$

b) $\gamma : \mathit{Boards} \rightarrow \mathbf{R}$

A recursive definition of γ, following a kind of *means-end analysis*:

Let be $o \in \mathit{Boards}$.

1. $\gamma(o) = 100$, if o represents a final board state that is won.
2. $\gamma(o) = -100$, if o represents a final board state that is lost.
3. $\gamma(o) = 0$, if o represents a final board state that is drawn.
4. $\gamma(o) = \gamma(o^*)$ otherwise.

o^* denotes the best final state that can be reached if both sides play optimally.
Related: minimax strategy, α-β pruning. [Course on Search Algorithms, Stein 1998-2020]

Specification of Learning Tasks

Example Chess: Real World $\rightarrow$ Model World

$\gamma(o) \rightarrow y\big(\alpha(o)\big) \equiv y(\mathbf{x})$ "model function"

Specification of Learning Tasks

Example Chess: Real World $\rightarrow$ Model World

$\gamma(o) \rightarrow y\big(\alpha(o)\big) \equiv y(\mathbf{x})$

"model function"

"model formation function"

Specification of Learning Tasks

Example Chess: Real World $\rightarrow$ Model World

$\gamma(o) \rightarrow y\big(\alpha(o)\big) \equiv y(\mathbf{x})$ "model function"

"model formation function"

$$y(\mathbf{x}) = w_0 + w_1 \cdot x_1 + w_2 \cdot x_2 + w_3 \cdot x_3 + w_4 \cdot x_4 + w_5 \cdot x_5 + w_6 \cdot x_6$$

where

$x_{1,\,2}$ = number of black / white pawns on board o

$x_{3,\,4}$ = number of black / white pieces on board o

$x_{5,\,6}$ = number of black / white pieces threatened on board o

D = $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$, a set of board descriptions $\mathbf{x}_i$ with scores $y_i, y_i \in [-100; 100]$.

Specification of Learning Tasks

Example Chess: Real World → Model World

$\gamma(o) \rightarrow y\big(\alpha(o)\big) \equiv y(\mathbf{x})$ "model function"

"model formation function"

$$y(\mathbf{x}) = w_0 + w_1 \cdot x_1 + w_2 \cdot x_2 + w_3 \cdot x_3 + w_4 \cdot x_4 + w_5 \cdot x_5 + w_6 \cdot x_6$$

where

$x_{1,\,2}$ = number of black / white pawns on board o

$x_{3,\,4}$ = number of black / white pieces on board o

$x_{5,\,6}$ = number of black / white pieces threatened on board o

D = $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$, a set of board descriptions $\mathbf{x}_i$ with scores $y_i, y_i \in [-100; 100]$.

Other approaches to specify a model function y :

- case base
- set of rules
- neural network
- higher order polynomial of board features

Remarks:

- The *ideal target function* γ interprets the real world, say, a real-world object o, to "compute" $\gamma(o)$. This "computation" can be operationalized by a human or by some other (even arcane) mechanism of the real world.
- To simulate the interesting aspects of the real world by means of a computer, we consider a model world. This model world is restricted to particular—typically easily measurable—features $\mathbf{x}$ that are derived from o, with $\mathbf{x} = \alpha(o)$.
  In the model world, $y(\mathbf{x})$ is the abstracted and formalized counterpart of $\gamma(o)$.
- y is called *model function* or *model*. The choice and computation of a suited model function is a central aspect in the field of machine learning.
- α is called *model formation function*. The development of a suited model formation function is often not treated as part of machine learning but "outsourced" to respective domain experts. However, tackling most advanced learning tasks such as autonomous driving, automated debating, or playing chess requires a tight cooperation between the developers of α and y.
- The key difference between an ideal target function γ and a model function y lies in the complexity and the representation of their respective domains. Examples:
  - A chess grand master assesses a board o in its entirety, both intuitively and analytically; a chess program is restricted to particular features $\mathbf{x}$, $\mathbf{x} = \alpha(o)$.
  - A human mushroom picker assesses a mushroom o with all her skills (intuitively, analytically, by tickled senses); a classification program is restricted to a few surface features $\mathbf{x}$, $\mathbf{x} = \alpha(o)$.

Remarks (continued):

- For automated chess playing a real-valued assessment function is needed; such kind of tasks form regression problems. If only a small number of values are to be considered (e.g. school grades), we are given a classification problem. A regression problem can be transformed into a classification problem via domain discretization.
- Regression problems and classification problems differ in the way how an achieved accuracy or goodness of fit is assessed. For regression problems the sum of the squared residuals may be a sensible criterion; for classification problems the number of misclassified examples may be more relevant. Keywords: *regression loss* versus *classification loss*
- For classification problems, the ideal target function γ is also called ideal *classifier*; analogously, the model function y is called classifier.
- Decision problems are classification problems with two classes.
- The halting problem for Turing machines is an undecidable classification problem.

Specification of Learning Tasks [model world]

Real World $\rightarrow$ Model World

Characterization of the real world:

- O is a set of objects. (example: emails)
- C is a set of classes. (example: spam versus ham)
- $\gamma : O \rightarrow C$ is the ideal classifier for O. (γ is a human expert)

Classification task:

- Given some $o \in O$, determine its class $\gamma(o) \in C$. (example: is an email spam?)

Specification of Learning Tasks [model world]

Real World $\rightarrow$ Model World

Characterization of the real world:

- O is a set of objects. (example: emails)
- C is a set of classes. (example: spam versus ham)
- $\gamma : O \rightarrow C$ is the ideal classifier for O. (γ is a human expert)

Classification task:

- Given some $o \in O$, determine its class $\gamma(o) \in C$. (example: is an email spam?)

Acquisition of classification knowledge D:

1. Collect real-world examples of the form $(o, \gamma(o)),\ o \in O$.
2. Abstract the objects towards feature vectors $\mathbf{x} \in X$, where $\mathbf{x} = \alpha(o)$.
3. Construct examples as $(\mathbf{x}, c(\mathbf{x}))$, where $\mathbf{x} = \alpha(o)$ and $c(\mathbf{x}) \equiv \gamma(o)$.

Specification of Learning Tasks [real world]

Real World → Model World (continued)

Characterization of the model world:

- X is a set of feature vectors, called feature space. (example: word frequencies)
- C is a set of classes. (as before: spam versus ham)
- $c : X \to C$ is the ideal classifier for X. (c is unknown)
- $D = \{(\mathbf{x}_1, c(\mathbf{x}_1)), \ldots, (\mathbf{x}_n, c(\mathbf{x}_n))\} \subseteq X \times C$ is a set of examples.

Todo:

- Approximate c, which is implicitly given via D, by a model function y.

Specification of Learning Tasks [real world]

Real World $\rightarrow$ Model World (continued)

Characterization of the model world:

- X is a set of feature vectors, called feature space. (example: word frequencies)
- C is a set of classes. (as before: spam versus ham)
- $c : X \rightarrow C$ is the ideal classifier for X. (c is unknown)
- $D = \{(\mathbf{x}_1, c(\mathbf{x}_1)), \ldots, (\mathbf{x}_n, c(\mathbf{x}_n))\} \subseteq X \times C$ is a set of examples.

Todo:

- Approximate c, which is implicitly given via D, by a model function y.

Machine learning:

1. Formulate a model function $y : X \rightarrow C,\ \mathbf{x} \mapsto y(\mathbf{x})$ (y needs to be fitted)
2. Apply statistics, theory, and algorithms from the field of machine learning to maximize the goodness of fit between the functions c and y.

Specification of Learning Tasks

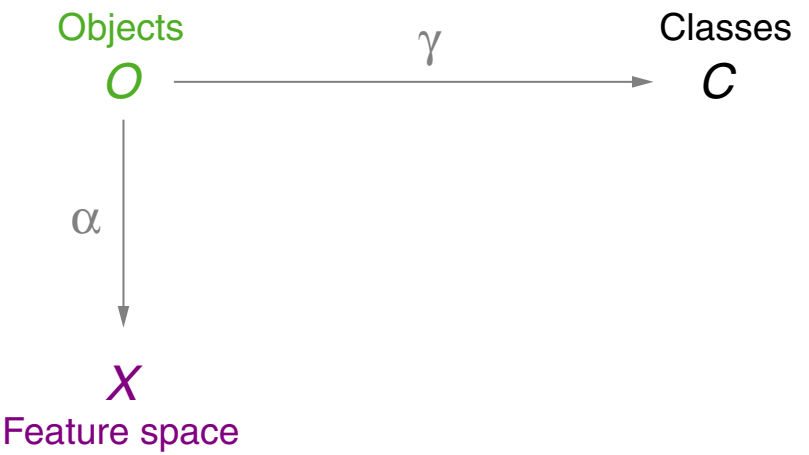
Real World $\rightarrow$ Model World (continued)

Semantics:

- γ Ideal classifier (a human) for real-world objects.

Specification of Learning Tasks

Real World → Model World (continued)

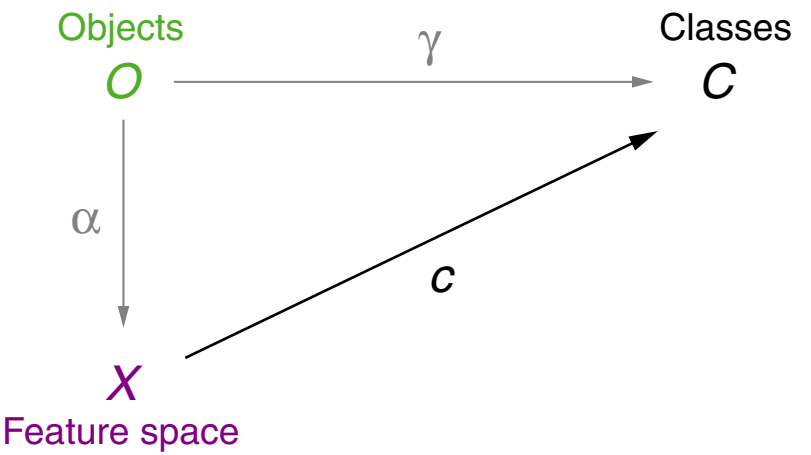


Semantics:

- γ Ideal classifier (a human) for real-world objects.
- α Model formation function.

Specification of Learning Tasks

Real World → Model World (continued)

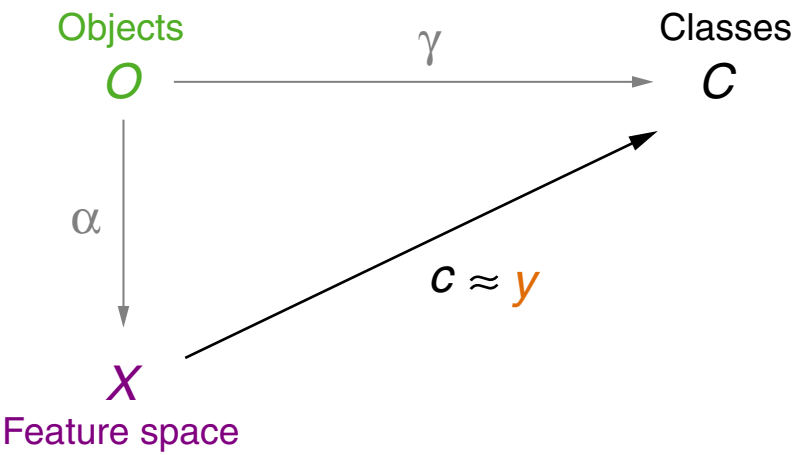


Semantics:

- γ Ideal classifier (a human) for real-world objects.
- α Model formation function.
- c Unknown ideal classifier for vectors from the feature space.

Specification of Learning Tasks

Real World → Model World (continued)



Semantics:

γ	Ideal classifier (a human) for real-world objects.
α	Model formation function.
c	Unknown ideal classifier for vectors from the feature space.
y	Classifier (model function) to be learned.
$c \approx y$	c is approximated by y, based on a set of examples D.

Remarks:

- The feature space X comprises vectors $\mathbf{x}_1, \mathbf{x}_2, \ldots$, which can be considered as abstractions of real-world objects $o_1, o_2, \ldots$, and which have been computed according to our view of the real world.
- The model formation function α determines the level of abstraction between o and $\mathbf{x}$, $\mathbf{x} = \alpha(o)$. I.e., α determines the representation fidelity, exactness, quality, or simplification.
- Though α models an object $o \in O$ only imperfectly as $\mathbf{x} = \alpha(o)$, c must be considered as *ideal* classifier, since $c(\mathbf{x})$ is defined as $\gamma(o)$, the true real-world class. I.e., c and γ have different domains each, but they return the same images.
- The function c is only implicitly given, in the form of the example set D. The representation of c as a closed function is unknown, it is approximated by y.
- $c(\mathbf{x})$ is often termed "ground truth" (for $\mathbf{x}$ and the underlying classification problem). Observe that this term is justified by the fact that $c(\mathbf{x}) \equiv \gamma(o)$.
- Note that in the chess example the scores y_i are not prescribed by $\gamma(o)$, since for $\gamma(o)$ only a recursive definition is given where $\gamma(o)$ *is unknown* for all boards o that fall under Point (4). I.e., for most chess boards o we *cannot provide the ground truth* $\gamma(o)$, say, we can neither give a statement whether o leads to a final board state that is won or lost if both sides play optimally nor provide the next optimum move.

Specification of Learning Tasks

The LMS Algorithm for Fitting y [*IGD* Algorithm]

Algorithm:	*LMS*	Least Mean Squares.
Input:	D	Training examples of the form $(\mathbf{x}, c(\mathbf{x}))$ with target function value $c(\mathbf{x})$ for $\mathbf{x}$.
	η	Learning rate, a small positive constant.
Internal:	$y(D)$	Set of $y(\mathbf{x})$-values computed from the elements $\mathbf{x}$ in D given some $\mathbf{w}$.
Output:	$\mathbf{w}$	Weight vector.

LMS(D, η)

1. *initialize_random_weights*$((w_0, w_1, \ldots, w_p))$
2. **REPEAT**
3. $(\mathbf{x}, c(\mathbf{x})) =$ *random_select*(D)
4. $y(\mathbf{x}) = w_0 + w_1 \cdot x_1 + \ldots + w_p \cdot x_p = \mathbf{w}^T\mathbf{x}$ // $\forall_{\mathbf{x}\in D} : \mathbf{x}|_{x_0} \equiv 1$
5. *error* $= c(\mathbf{x}) - y(\mathbf{x})$
6. $\Delta\mathbf{w} = \eta \cdot$ *error* $\cdot \mathbf{x}$
7. $\mathbf{w} = \mathbf{w} + \Delta\mathbf{w}$
8. **UNTIL**(*convergence*$(D, y(D))$)
9. *return*$((w_0, w_1, \ldots, w_p))$

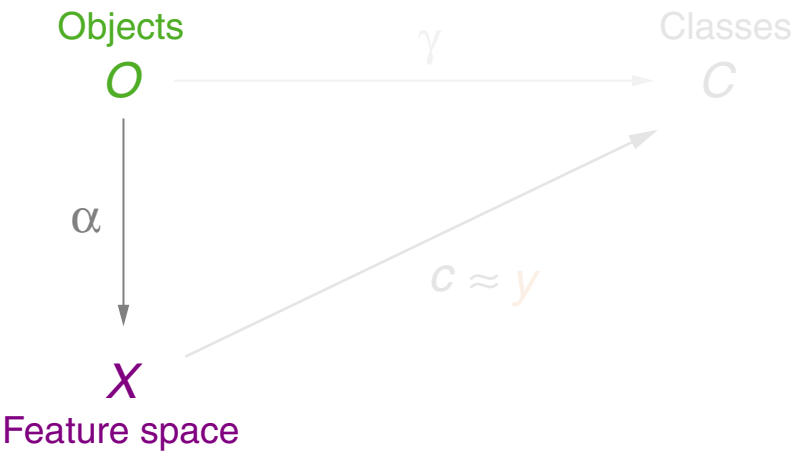
Remarks:

- The LMS weight adaptation corresponds to the incremental gradient descend [*IGD*] algorithm, and approximates the global direction of steepest error descent as used by the batch gradient descent [*BGD*] algorithm, for which more rigorous statements on convergence are possible.
- The *convergence* function may compute the global error quantified as the sum of the squared residuals, $\sum_{(\mathbf{x},c(\mathbf{x}))\in D} \left(c(\mathbf{x}) - y(\mathbf{x})\right)^2$, or employ an upper bound on the number of iterations.

Elements of Machine Learning

Elements of Machine Learning

I. Model Formation: Real World → Model World

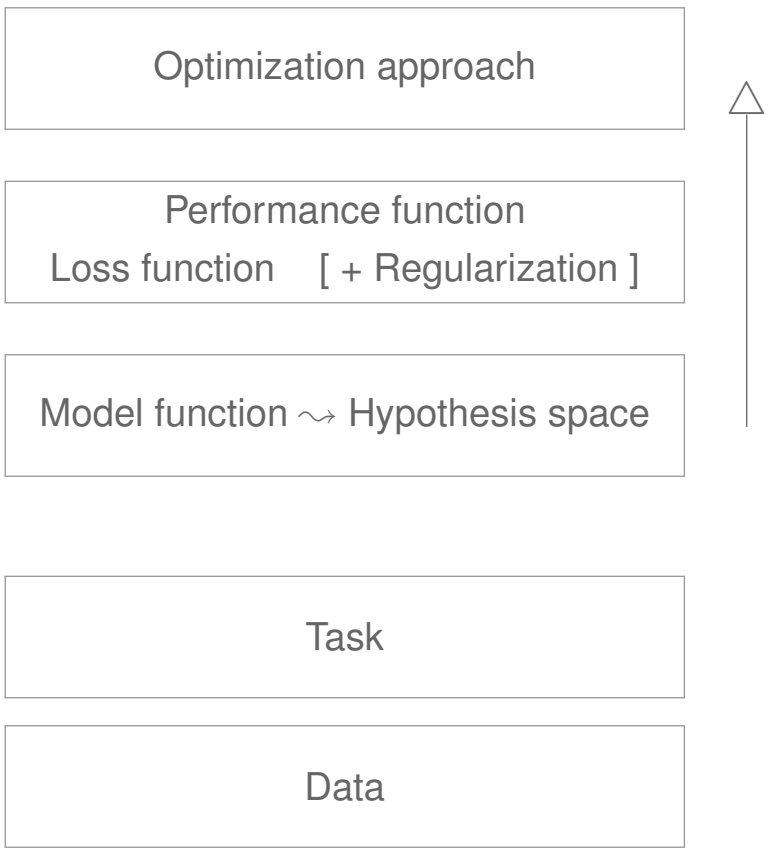


Related questions:

- From what kind of experience should be learned?
- Which level of fidelity is sufficient to solve a certain task?

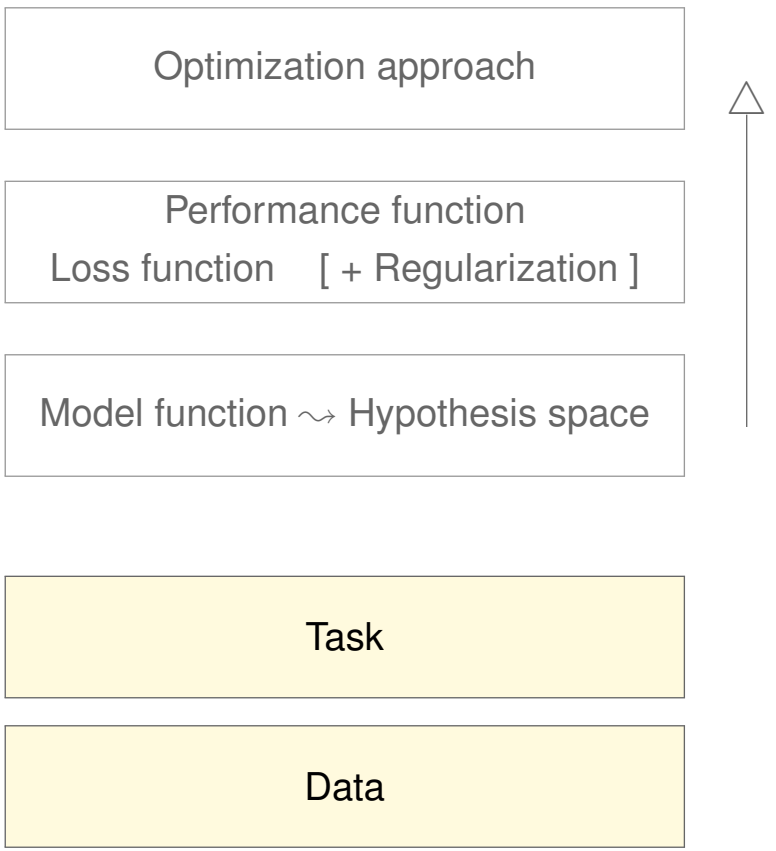
Elements of Machine Learning

II. Design of Supervised Learning Algorithms



Elements of Machine Learning

II. Design of Supervised Learning Algorithms: LMS

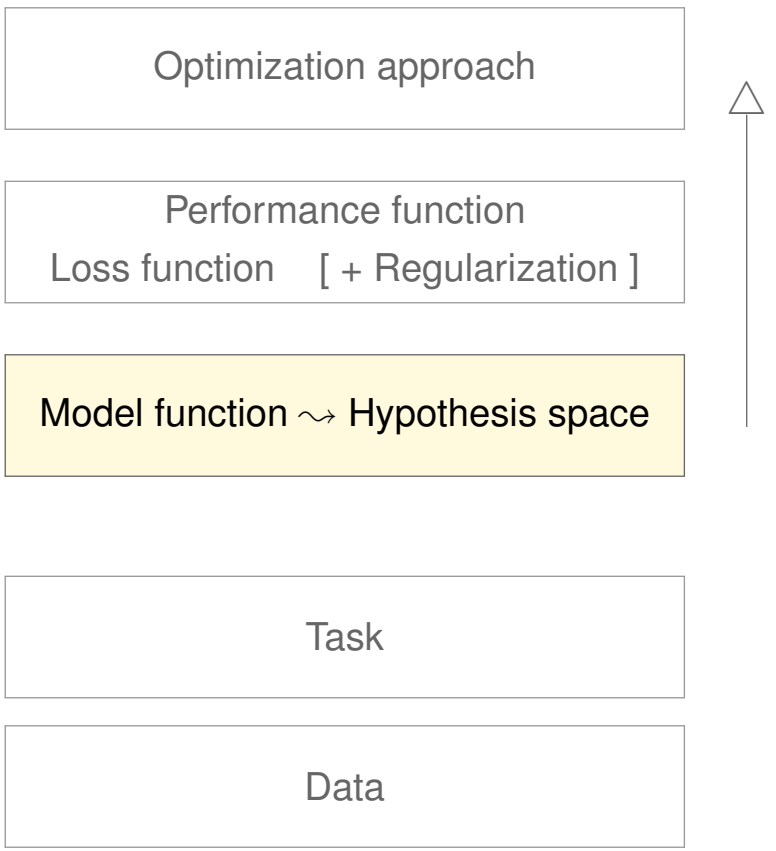


Classification

$$D = \{(\mathbf{x}_1, c(\mathbf{x}_1)), \ldots, (\mathbf{x}_n, c(\mathbf{x}_n))\} \subseteq X \times C$$

Elements of Machine Learning

II. Design of Supervised Learning Algorithms: LMS (continued)



- Hypothesis space: Set of linear functions
- Linear model: $y(\mathbf{x}) = w_0 + \sum_{i=1}^{p} w_i x_i$

Classification

$$D = \{(\mathbf{x}_1, c(\mathbf{x}_1)), \ldots, (\mathbf{x}_n, c(\mathbf{x}_n))\} \subseteq X \times C$$

Elements of Machine Learning

II. Design of Supervised Learning Algorithms: LMS (continued)

Optimization approach

Performance function

Loss function [+ Regularization]

Model function $\rightsquigarrow$ Hypothesis space

Task

Data

- Performance: Minimize MSE
- Regularization: None
- Loss: Sum of squared residuals

- Hypothesis space: Set of linear functions
- Linear model: $y(\mathbf{x}) = w_0 + \sum_{i=1}^{p} w_i x_i$

Classification

$$D = \{(\mathbf{x}_1, c(\mathbf{x}_1)), \ldots, (\mathbf{x}_n, c(\mathbf{x}_n))\} \subseteq X \times C$$

Elements of Machine Learning

II. Design of Supervised Learning Algorithms: LMS (continued)

Optimization approach

Performance function

Loss function [+ Regularization]

Model function $\rightsquigarrow$ Hypothesis space

Task

Data

Incremental/stochastic gradient descend

- Performance: Minimize MSE
- Regularization: None
- Loss: Sum of squared residuals

- Hypothesis space: Set of linear functions
- Linear model: $y(\mathbf{x}) = w_0 + \sum_{i=1}^{p} w_i x_i$

Classification

$$D = \{(\mathbf{x}_1, c(\mathbf{x}_1)), \ldots, (\mathbf{x}_n, c(\mathbf{x}_n))\} \subseteq X \times C$$

Related questions:

- What are useful classes of model functions?
- What are methods to fit (= learn) model functions?
- What are measures to assess the goodness of fit?

- How does (label) noise affect the learning process?
- How does the example number affect the learning process?
- How to deal with extreme class imbalance?

The feature space is an inner product space.

- An inner product space (also called pre-Hilbert space) is a vector space with an additional structure called “inner product”.
- Example: Euclidean vector space equipped with the dot product.
- Enables algorithms such as gradient descent and support vector machines.

Elements of Machine Learning

III. Structure of Feature Spaces (continued)

The feature space is an inner product space.

- An inner product space (also called pre-Hilbert space) is a vector space with an additional structure called “inner product”.
- Example: Euclidean vector space equipped with the dot product.
- Enables algorithms such as gradient descent and support vector machines.

The feature space is a σ-algebra.

- A σ-algebra on a set X is a collection of subsets of X that includes X itself, is closed under complement, and is closed under countable unions.
- Enables probability spaces and statistical learning, such as naive Bayes.

Elements of Machine Learning

III. Structure of Feature Spaces (continued)

The feature space is an inner product space.

- An inner product space (also called pre-Hilbert space) is a vector space with an additional structure called “inner product”.
- Example: Euclidean vector space equipped with the dot product.
- Enables algorithms such as gradient descent and support vector machines.

The feature space is a σ-algebra.

- A σ-algebra on a set X is a collection of subsets of X that includes X itself, is closed under complement, and is closed under countable unions.
- Enables probability spaces and statistical learning, such as naive Bayes.

The feature space is a finite set of vectors with nominal dimensions.

- Requires concept learning via set splitting as done by decision trees.

Remarks:

- The aforementioned examples for feature spaces are not meant to be complete but shall illustrate the range of different structures underlying the example sets we want to learn from.
- The structure of a feature space constrains the applicable learning algorithm. Usually, this structure is inherently determined by the application domain and cannot be chosen.

Elements of Machine Learning

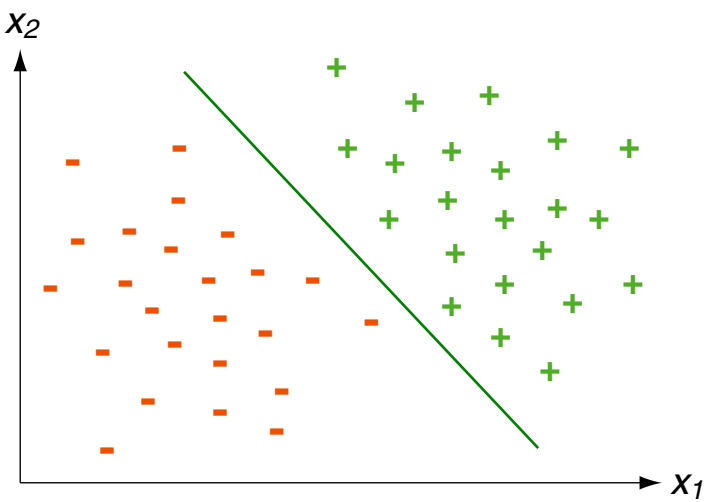
IV. Discriminative versus Generative Approaches

- Discriminative approaches learn a boundary between classes.
- Generative approaches exploit the distributions underlying the classes.

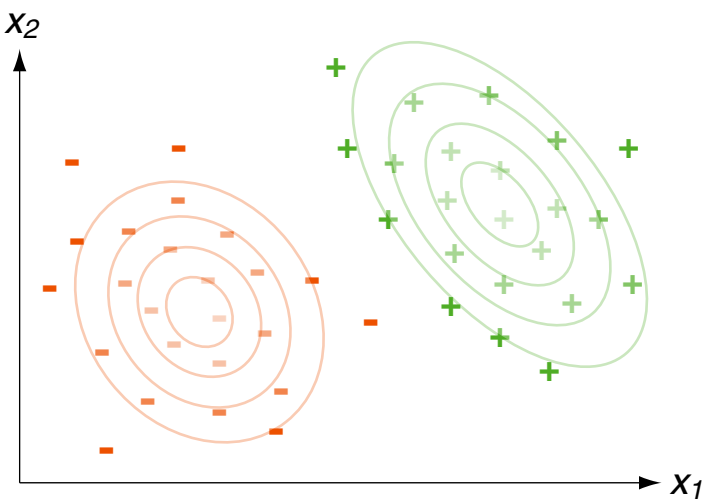
Elements of Machine Learning

IV. Discriminative versus Generative Approaches (continued)

- Discriminative approaches learn a boundary between classes.
- Generative approaches exploit the distributions underlying the classes.



discriminative
⇝ classification rule

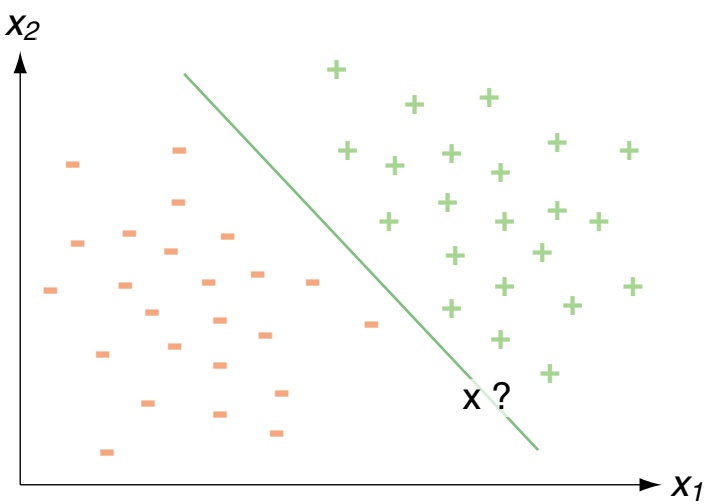


generative
⇝ class membership probability

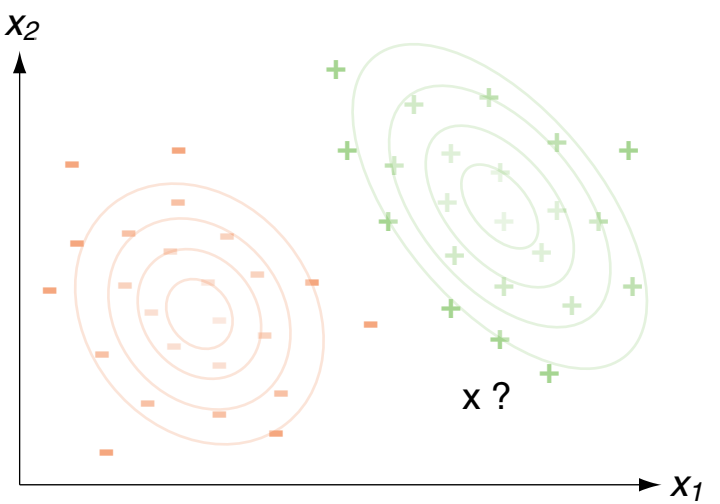
Elements of Machine Learning

IV. Discriminative versus Generative Approaches (continued)

- Discriminative approaches learn a boundary between classes.
- Generative approaches exploit the distributions underlying the classes.



discriminative
⇝ classification rule



generative
⇝ class membership probability

Remarks:

- When classifying a new example (1) discriminative approaches apply a decision rule that was learned via minimizing the misclassification rate given training examples D, while (2) generative approaches maximize the probability of the combined event $P(\mathbf{x}, y)$, or, similarly, the a-posteriori probability $P(y \mid \mathbf{x})$, $y \in \{\ominus, \oplus\}$.
- The LMS algorithm computes "only" a decision boundary, i.e., it constructs a classifier according to the discriminative approach. A Bayes classifier is an example for a classifier constructed according to the generative approach.
- Yoav Freund provides an excellent video illustrating the pros and cons of discriminative and generative approaches respectively. [YouTube]

Elements of Machine Learning

V. Frequentism versus Subjectivism

Frequentism:

- There is a (hidden) mechanism that generates D.
- To model this mechanism you may consider
  - a family of distributions,
  - a model function,
  - a combination of both,

  characterized by θ. The possible values for θ form the hypothesis space H.
- Select a most probable hypothesis $h_{\text{ML}} \in H$ by estimating θ using a sample $D' \subset D$. h_{ML} is called maximum likelihood hypothesis.

Elements of Machine Learning

V. Frequentism versus Subjectivism (continued)

Frequentism:

- There is a (hidden) mechanism that generates D.
- To model this mechanism you may consider
  - a family of distributions,
  - a model function,
  - a combination of both,

  characterized by θ. The possible values for θ form the hypothesis space H.
- Select a most probable hypothesis $h_{\text{ML}} \in H$ by estimating θ using a sample $D' \subset D$. h_{ML} is called maximum likelihood hypothesis.

Elements of Machine Learning

V. Frequentism versus Subjectivism (continued)

Frequentism:

- There is a (hidden) mechanism that generates D.
- To model this mechanism you may consider
  - a family of distributions,
  - a model function,
  - a combination of both,

  characterized by θ. The possible values for θ form the hypothesis space H.
- Select a most probable hypothesis $h_{\text{ML}} \in H$ by estimating θ using a sample $D' \subset D$. h_{ML} is called maximum likelihood hypothesis.

Elements of Machine Learning

V. Frequentism versus Subjectivism (continued)

Frequentism:

- There is a (hidden) mechanism that generates D.
- To model this mechanism you may consider
  - a family of distributions,
  - a model function,
  - a combination of both,

  characterized by θ. The possible values for θ form the hypothesis space H.
- Select a most probable hypothesis $h_{\text{ML}} \in H$ by estimating θ using a sample $D' \subset D$. h_{ML} is called maximum likelihood hypothesis.

$$\theta \rightsquigarrow D', \qquad h_{\text{ML}} = \operatorname*{argmax}_{h \in H} P(D' \mid h)$$

Remarks:

- θ is a parameter or a parameter vector that is considered as fixed (in particular: not as a random variable), but unknown.
- In the experiment of flipping a coin, one may suppose a Laplace experiment and consider the binomial distribution, $B(n, p)$.
- $P(D' \mid h)$ is the probability for observing D' under h. I.e., it is the probability for observing D' if the hidden mechanism that generates D' behaves according to the considered model whose parameter θ is set to h.

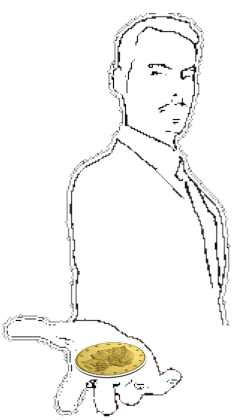
Elements of Machine Learning

V. Frequentism versus Subjectivism (continued)

Subjectivism:

- Consider a model for the mechanism that has generated D.
- There are different beliefs about the parameter (vector) θ that characterizes the model. The possible values for θ form the hypothesis space H.
- Select a most probable hypothesis $h_{\text{MAP}} \in H$ by weighting the ML estimates under D with the priors. h_{MAP} is called maximum a-posteriori hypothesis.

Belief/Prior 1: $\underbrace{P(p = 0.5)}_{\theta_1}$ = 0.95



Belief/Prior 2: $\underbrace{P(p = 0.75)}_{\theta_2}$ = 0.50

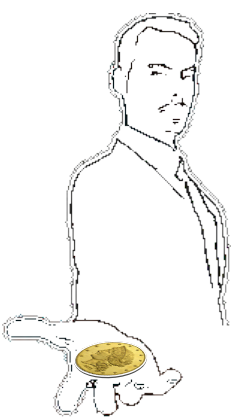
Elements of Machine Learning

V. Frequentism versus Subjectivism (continued)

Subjectivism:

- Consider a model for the mechanism that has generated D.
- There are different beliefs about the parameter (vector) θ that characterizes the model. The possible values for θ form the hypothesis space H.
- Select a most probable hypothesis $h_{\text{MAP}} \in H$ by weighting the ML estimates under D with the priors. h_{MAP} is called maximum a-posteriori hypothesis.

Belief/Prior 1: $\underbrace{P(p = 0.5)}_{\theta_1}$ = 0.95



Belief/Prior 2: $\underbrace{P(p = 0.75)}_{\theta_2}$ = 0.50

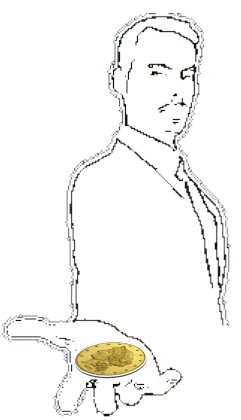
Elements of Machine Learning

V. Frequentism versus Subjectivism (continued)

Subjectivism:

- Consider a model for the mechanism that has generated D.
- There are different beliefs about the parameter (vector) θ that characterizes the model. The possible values for θ form the hypothesis space H.
- Select a most probable hypothesis $h_{\text{MAP}} \in H$ by weighting the ML estimates under D with the priors. h_{MAP} is called maximum a-posteriori hypothesis.

Belief/Prior 1: $\underbrace{P(p = 0.5)}_{\theta_1}$ = 0.95



Belief/Prior 2: $\underbrace{P(p = 0.75)}_{\theta_2}$ = 0.50

Elements of Machine Learning

V. Frequentism versus Subjectivism (continued)

Subjectivism:

- Consider a model for the mechanism that has generated D.
- There are different beliefs about the parameter (vector) θ that characterizes the model. The possible values for θ form the hypothesis space H.
- Select a most probable hypothesis $h_{\text{MAP}} \in H$ by weighting the ML estimates under D with the priors. h_{MAP} is called maximum a-posteriori hypothesis.

Belief/Prior 1: $\underbrace{P(p = 0.5)}_{\theta_1}$ = 0.95

Belief/Prior 2: $\underbrace{P(p = 0.75)}_{\theta_2}$ = 0.50

$$\left.\begin{array}{l} \theta_1 + D \;\rightarrow\; P(D \mid \theta_1) \\ \theta_2 + D \;\rightarrow\; P(D \mid \theta_2) \end{array}\right\}$$

Elements of Machine Learning

V. Frequentism versus Subjectivism (continued)

Subjectivism:

- Consider a model for the mechanism that has generated D.
- There are different beliefs about the parameter (vector) θ that characterizes the model. The possible values for θ form the hypothesis space H.
- Select a most probable hypothesis $h_{\text{MAP}} \in H$ by weighting the ML estimates under D with the priors. h_{MAP} is called maximum a-posteriori hypothesis.

Belief/Prior 1: $\underbrace{P(p = 0.5)}_{\theta_1}$ = 0.95

Belief/Prior 2: $\underbrace{P(p = 0.75)}_{\theta_2}$ = 0.50

$$\left.\begin{array}{lcl} \theta_1 + D & \rightarrow & P(D \mid \theta_1) \\ \theta_2 + D & \rightarrow & P(D \mid \theta_2) \end{array}\right\} \quad h_{\text{MAP}} = \underset{h \in \{\theta_1, \theta_2\}}{\text{argmax}} P(h \mid D) = \underset{h \in \{\theta_1, \theta_2\}}{\text{argmax}} \frac{P(D \mid h) \cdot P(h)}{P(D)}$$

Remarks:

- θ is considered as random variable. There is prior knowledge about the distribution of θ.
- p is a parameter of the binomial distribution and denotes the success probability for each trial.
  - Belief 1: With a probability of 0.95 the coin is fair (both sides are equally likely).
  - Belief 2: With a probability of 0.5 the odds of prefering a particular side is 3:1.
- The subjectivistic approach is also called Bayesian interpretation of probability.
- The Bayesian interpretation of probability enables by design the integration of prior knowledge, background knowledge, and human expertise.
- Food for tought: Discuss the use of frequentist and subjectivist approaches to decision making if you had to develop an AI that plays poker.