Lecture 9: Bayesian Learning

Cognitive Systems - Machine Learning

Part II: Special Aspects of Concept Learning

Bayes Theorem, MAP / ML hypotheses, Brute-force MAP Learning, Bayes Optimal Classifier, Naive Bayes Classifier, Bayes Belief Networks, Graphical Models

last change December 8, 2015

Motivation

- probabilistic approach to inference
- **basic assumption**:
  - quantities of interest are governed by probability distributions
  - optimal decisions can be made by reasoning about these probabilities together with observed training data
- Bayesian Learning is relevant for two reasons
  - **first reason**: explicit manipulation of probabilities
    - among the most practical approaches to certain types of learning problems
    - e.g. Bayes classifier is competitive with decision tree and neural network learning

Motivation

- Bayesian Learning is relevant for two reasons (cont.)
  - **second reason**: useful perspective for understanding learning methods that do not explicitly manipulate probabilities
    - determine conditions under which algorithms output the most probable hypothesis
    - e.g. justification of the error functions in ANNs
    - e.g. justification of the inductive bias of decision trees
- **features of Bayesian Learning methods:**
  - each observed training example can incrementally decrease or increase the estimated probability that a hypothesis is correct
  - prior knowledge can be combined with observed data to determine the final probability of a hypothesis

Motivation

- **features of Bayesian Learning methods (cont.):**
  - hypotheses make probabilistic predictions
  - new instances can be classified by combining the predictions of multiple hypotheses, weighted by their probabilities
  - standard of optimal decision making against which other practical methods can be measured
- **practical difficulties:**
  - initial knowledge of many probabilities is required
  - significant computational costs

Outline

- Bayes Theorem
- MAP Hypothesis
- Byes Theorem and Concept Learning
- Brute-Force MAP Learning
- Naive Byes Classifier
- Estimating Probabilities
- Bayes Belief Networks / Graphical Models
  - Notation
  - Representation
  - Inference
- Maximum Likelihood and Least-Squared Error
- Minimum Description Length Principle
- Summary

Bayes Theorem

- **machine learning** is interested in the *best hypothesis h* from some space H, given observed training data D
- *best hypothesis* $\approx$ *most probable hypothesis*

$\Rightarrow$ Bayes Theorem provides a **direct method of calculating the probability of such a hypothesis** based on its prior probability, the probabilities of observing various data given the hypothesis, and the observed data itself

Bayes Theorem

- **more formal:**
  - $P(h)$ *prior probability of* h, reflects any background knowledge about the chance that h is correct
  - $P(D)$ *prior probability of* D, probability that D will be observed
  - $P(D|h)$ probability of observing D given a world in which h holds
  - $P(h|D)$ *posterior probability of* h, reflects confidence that h holds after D has been observed
- **Bayes Theorem:**

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

MAP Hypothesis

- in many learning scenarios, the learner considers some set of candidate hypotheses H and is interested in finding the most probable hypothesis $h \in H$ given the observed training data D
- any maximally probable hypothesis is called *maximum a posteriori* (**MAP**) hypotheses

$$
\begin{aligned}
h_{MAP} &= \underset{h \in H}{argmax}\ P(h|D) \\
&= \underset{h \in H}{argmax}\ \frac{P(D|h)P(h)}{P(D)} \\
&= \underset{h \in H}{argmax}\ P(D|h)P(h)
\end{aligned}
$$

note that $P(D)$ can be dropped, because it is a constant independent of h

ML Hypothesis

- sometimes it is assumed that every hypothesis is equally probable a priori
- in this case, the equation before can be simplified
- because $P(D|h)$ is often called the *likelihood of D given h*, any hypothesis that maximizes $P(D|h)$ is called *maximum likelihood (**ML**) hypothesis*

$$h_{ML} = \underset{h \in H}{argmax}\ P(D|h)$$

note that in this case $P(h)$ can be dropped, because it is equal for each $h \in H$

Example

- consider a medical diagnosis problem in which there are two alternative hypotheses
  - the patient has a particular form of cancer (denoted by *cancer*)
  - the patient does not (denoted by $\neg cancer$)
- the available data is from a particular laboratory with two possible outcomes:
  $\oplus$ (positive) and $\ominus$ (negative)

$$P(cancer) = 0.008 \qquad P(\neg cancer) = 0.992$$
$$P(\oplus|cancer) = 0.98 \qquad P(\ominus|cancer) = 0.02$$
$$P(\oplus|\neg cancer) = 0.03 \qquad P(\ominus|\neg cancer) = 0.97$$

- suppose a new patient is observed for whom the lab test returns a positive ($\oplus$) result
- Should we diagnose the patient as having cancer or not?

$$P(\oplus|cancer)P(cancer) = (0.98)0.008 = 0.0078$$
$$P(\oplus|\neg cancer)P(\neg cancer) = (0.03)0.992 = 0.0298$$
$$\Rightarrow h_{MAP} = \neg cancer$$

Example

- the exact posterior probabilities can be determined by normalizing the above properties to 1

$$P(cancer|\oplus) = \frac{0.0078}{0.0078 + 0.0298} = 0.21$$

$$P(\neg cancer|\oplus) = \frac{0.0298}{0.0078 + 0.0298} = 0.79$$

$\Rightarrow$ the result of Bayesian inference depends strongly on the prior probabilities, which must be available in order to apply the method directly

Bayes Theorem and Concept Learning

- What is the relationship between Bayes theorem and the problem of concept learning?
- it can be used for designing a straightforward learning algorithm
- **Brute-Force MAP LEARNING algorithm**
  1. For each hypothesis $h \in H$, calculate the posterior probability

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

  2. Output hypothesis h_{MAP} with the highest posterior probability

$$h_{MAP} = \underset{h \in H}{argmax}\ P(h|D)$$

Brute-Force MAP LEARNING

- in order to specify a learning problem for the algorithm, values for $P(h)$ and $P(D|h)$ must be specified
- **assumptions**
  1. training data D is noise free (i.e., $d_i = c(x_i)$)
  2. target concept c is contained in H (i.e. $(\exists h \in H)[(\forall x \in X)[h(x) = c(x)]]$)
  3. no reason to believe that any hypothesis is more probable than any other

$$\Rightarrow P(h) = \frac{1}{|H|} \text{ for all } h \in H$$

$$\Rightarrow P(D|h) = \begin{cases} 1 & \text{if } d_i = h(x_i) \text{ for all } d_i \in D \\ 0 & \text{otherwise} \end{cases}$$

Brute-Force MAP LEARNING

- now the problem for the learning algorithms is fully-defined
- in a first step, we have to determine the probabilities for $P(h|D)$
  - h is **inconsistent** with training data D

$$P(h|D) = \frac{0 \cdot P(h)}{P(D)} = 0$$

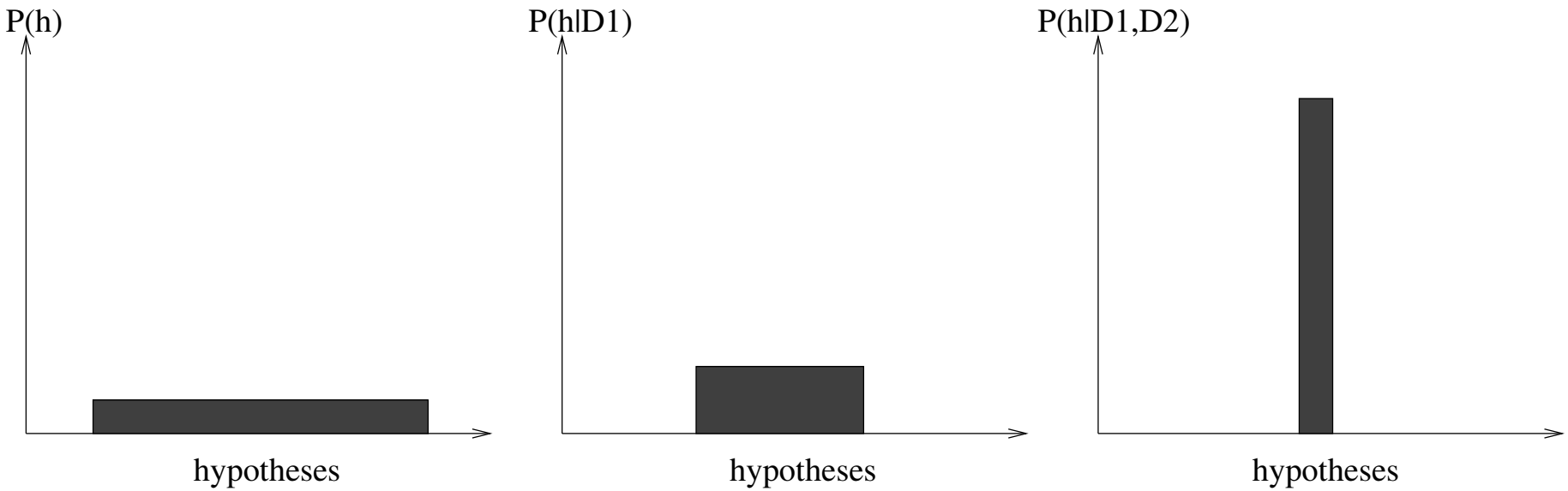
  - h is **consistent** with training data D

$$P(h|D) = \frac{1 \cdot \frac{1}{|H|}}{P(D)} = \frac{1 \cdot \frac{1}{|H|}}{\frac{|VS_{H,D}|}{|H|}} = \frac{1}{|VS_{H,D}|}$$

$\Rightarrow$ The sum over all hypotheses of $P(h|D)$ must be one and the number of hypotheses consistent with D is by definition $|VS_{H,D}|$.

$\Rightarrow$ this analysis implies that, **under these assumptions**, each consistent hypothesis is a MAP hypothesis, because for each **consistent** hypothesis $P(h|D) = \frac{1}{|VS_{H,D}|}$

Brute-Force MAP LEARNING



- **evolution of probabilities**

  (a) all hypotheses have the same probability
  (b) + (c) as training data accumulates, the posterior probability of inconsistent hypotheses becomes zero while the total probability summing to 1 is shared equally among the remaining consistent hypotheses

Consistent Learners

- *every consistent learner outputs a MAP hypothesis, if we assume a uniform prior probability distribution over H and deterministic, noise-free training data*
- FIND-S
  - if c is contained in H, then
  - outputs a consistent hypothesis and therefore a MAP hypothesis under the probability distributions $P(h)$ and $P(D|h)$ defined above
  - i.e. for each $P(h)$ that favors more specific hypotheses, FIND-S outputs a MAP hypothesis

$\Rightarrow$ Bayesian framework is a way to characterize the behaviour of learning algorithms

$\Rightarrow$ by identifying probability distributions $P(h)$ and $P(D|h)$ under which the output is a optimal hypothesis, implicit assumptions of the algorithm can be characterized (**Inductive Bias**)

$\Rightarrow$ inductive inference is modeled by an equivalent *probabilistic reasoning* system based on Bayes theorem

Bayes Optimal Classifier

- **question:** What is the most probable classification of the new instance given the training data?
- simply applying h_{MAP} is not the best solution (as one could wrongly think of)
- **example:**
  - $H = \{h_1, h_2, h_3\}$ where $P(h_1|D) = .4$, $P(h_2|D) = P(h_3|D) = .3$
  - $h_{MAP} = h_1$
  - consider a new instance x encountered, which is classified positive by h_1 and negative by h_2, h_3
  - taking all hypotheses into account,
    - the probability that x is positive is .4 and
    - the probability that x is negative is .6

  $\Rightarrow$ most probable classification $\neq$ classification of h_{MAP}

Bayes Optimal Classifier

- the most probable classification is obtained by combining the predictions of all hypotheses, weighted by their posterior probabilities

$$P(v_j|D) = \sum_{h_i \in H} P(v_j|h_i)P(h_i|D)$$

  where $P(v_j|D)$ is the probability that the correct classification is v_j
- **Bayes optimal classifier:**

$$\underset{v_j \in V}{argmax} \sum_{h_i \in H} P(v_j|h_i)P(h_i|D)$$

Illustrative Example

$V = \{\oplus, \ominus\}$

$$\begin{array}{lll} P(h_1, D) = 0.4 & P(\ominus, h_1) = 0 & P(\oplus, h_1) = 1 \\ P(h_2, D) = 0.3 & P(\ominus, h_2) = 1 & P(\oplus, h_2) = 0 \\ P(h_3, D) = 0.3 & P(\ominus, h_3) = 1 & P(\oplus, h_3) = 0 \end{array}$$

therefore

$$\sum_{h_i \in H} P(\oplus|h_i)P(h_i|D) = 0.4$$

$$\sum_{h_i \in H} P(\ominus|h_i)P(h_i|D) = 0.6$$

and

$$\underset{v_j \in \{\oplus, \ominus\}}{argmax} \sum_{h_i \in H} P(v_j|h_i)P(h_i|D) = \ominus$$

Naive Bayes Classifier

- applies to learning tasks where each instance x is described by a conjunction of attribute values and where the target function $f(x)$ can take on any value from some finite set V
- training examples are described by $\langle a_1, a_2, ..., a_n \rangle$
- Bayesian approach

$$
\begin{aligned}
v_{MAP} &= \underset{v_j \in V}{argmax}\ P(v_j | a_1, a_2, ..., a_n) \\
&= \underset{v_j \in V}{argmax}\ \frac{P(a_1, a_2, ..., a_n | v_j) P(v_j)}{P(a_1, a_2, ..., a_n)} \\
&= \underset{v_j \in V}{argmax}\ P(a_1, a_2, ..., a_n | v_j) P(v_j)
\end{aligned}
$$

Naive Bayes Classifier

- $P(v_j)$ can be estimated by counting the frequency of v_j in D
- $P(a_1, a_2, ..., a_n|v_j)$ cannot by estimated in this fashion
  - number of these terms is $|X| \cdot |V|$
- **simplification of naive Bayes classifier**
  - attribute values are conditionally independent
  - hence, $P(a_1, a_2, ..., a_n|v_j) = \prod_i P(a_i|v_j)$
  - hence, number terms is

$$|\mathit{distinct\ attributes}| \cdot |\mathit{distinct\ target\ values}|$$

  - no explicit search through H, just counting frequencies

$\Rightarrow$ **Naive Bayes Classifier**

$$v_{NB} = \underset{v_j \in V}{\mathit{argmax}}\ P(v_j) \prod_i P(a_i|v_j)$$

Illustrative Example

- example days:

Day	*Sunny*	*Temp.*	*Humidity*	*Wind*	*PlayTennis*
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Illustrative Example

- novel instance:
  $\langle Outlook = Sunny, Temperature = Cool, Humidity = High, Wind = Strong \rangle$
- Instantiation of the Naive Bayes Classifier

$$v_{NB} = \underset{v_j \in \{yes,no\}}{argmax} \; P(v_j) \prod_i P(a_i|v_j)$$

where

$$\prod_i P(a_i|v_j) = P(Outlook = sunny|v_j) \cdot P(Temperature = cool|v_j) \cdot$$
$$P(Humidity = high|v_j) \cdot P(Wind = strong|v_j)$$

- estimation of probabilities

$$P(PlayTennis = yes) = \frac{9}{14} = 0.64$$
$$P(PlayTennis = no) = \frac{5}{14} = 0.36$$

Illustrative Example

- similarly, conditional probabilities can be estimated (e.g. $Wind = Strong$)

$$P(Wind = Strong|PlayTennis = yes) = \frac{3}{9} = 0.33$$
$$P(Wind = Strong|PlayTennis = no) = \frac{3}{5} = 0.60$$

- calculation of v_{NB}

$$P(yes)P(sunny|yes)P(cool|yes)P(high|yes)P(strong|yes) = 0.0053$$
$$P(no)P(sunny|no)P(cool|no)P(high|no)P(strong|no) = 0.0206$$
$$\Rightarrow v_{NB} = no$$

- normalization

$$\frac{0.0206}{0.0206+0.0053} = 0.795$$

Estimating Probabilities

- normally, probabilities are estimated by the fraction of times the event is observed to occur over the total number of opportunities ($\frac{n_c}{n}$)
- in most cases, this method is a good estimate
- but if n_c is very small, it provides poor results
  - biased underestimate of the probability
  - if this estimate equals zero, it will dominate the Bayes classifier
- Bayesian approach: ***m-estimate***

$$\frac{n_c + mp}{n + m}$$

where p is a prior estimate of the probability we wish to determine, and m is a constant called the *equivalent sample size* which determines how heavily to weight p relative to the observed data

Estimating Probabilities

- in the absence of information, it is common to assume a uniform distribution for p
- hence, $p = \frac{1}{k}$ where k is the number of possible attribute values
- if $m = 0$, the ***m-estimate*** is equivalent to $\frac{n_c}{n}$
- m can be interpreted as the number of virtual samples distributed according to p that are added the n actual observed examples

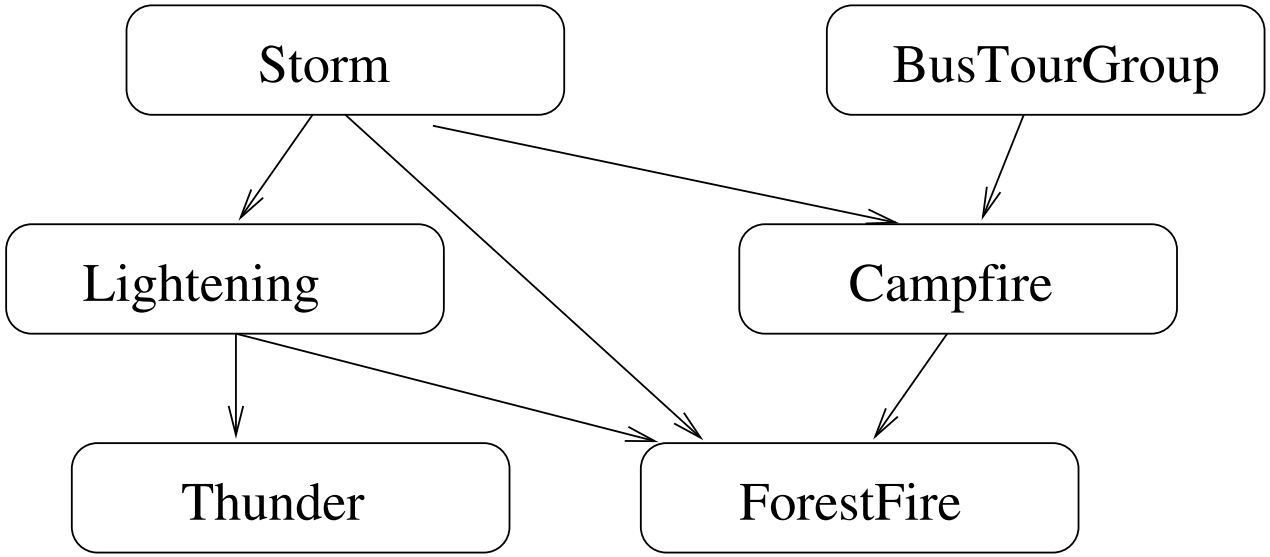
Bayesian Belief Networks

- **motivation**
  - naive Bayes classifier makes significant use of the assumption of conditional independence
  - this assumption dramatically reduces the complexity of the learning task
  - however, in many cases this assumption is overly restrictive
- **Bayesian Belief Network**
  - describes probability distribution governing a set of variables by specifying a **set of conditional independence assumptions** along with a **set of conditional probabilities**
  - conditional independence assumption applies only to subsets of the variables
  - **Graphical models** such as Hidden Markov Models are special cases of Bayesian networks

Notation

- Bayesian Belief Networks describe the probability distribution over a set of variables
- arbitrary set of random variables $Y_1, ..., Y_n$ where $V(Y_i)$ is the set of possible values for Y_i
- *joint space*: $V(Y_1) \times V(Y_2) \times ... \times V(Y_n)$
- *joint probability distribution* specifies the probability for each of the possible variable bindings for the tuple $\langle Y_1, Y_2, ..., Y_n \rangle$

Representation



Conditional Probability Table for “CampFire”

	S,B	S, ¬B	¬S,B	¬S,¬B
C	0.4	0.1	0.8	0.2
¬C	0.6	0.9	0.2	0.8

Representation

- joint probability distribution over the boolean variables *Storm*, *Lighting*, *Thunder*, *ForestFire*, *CampFire*, and *BusTourGroup*
- set of conditional independence assumptions
  - represented by a *directed acyclic graph* (DAG)
  - node $\approx$ variables in the joint space
  - arcs $\approx$ conditional dependence of the originator
- for each node a conditional probability table is given
  - describes probability distribution for the variable given the values of its immediate predecessors
  - the joint probability for any desired assignment of $\langle y_1, y_2, ..., y_n \rangle$ is computed by

$$P(y_1, y_2, ..., y_n) = \prod_{i=1}^{n} P(y_i | Parents(Y_i))$$

where $Parents(Y_i)$ denotes the set of immediate predecessors of Y_i

Inference

- **task:** inference of the probability distribution for a target value, e.g. *ForestFire*
- if values are known for all other variables in the network, the inference is straightforward
- in the more general case, values are only known for a subset of the network variables
- a Bayesian Network can be used to compute the probability distribution for any subset of network variables given the values or distributions for any subset of the remaining variables
- exact inference is NP-hard, even approximate inference can be NP-hard

Categorisation of Algorithms

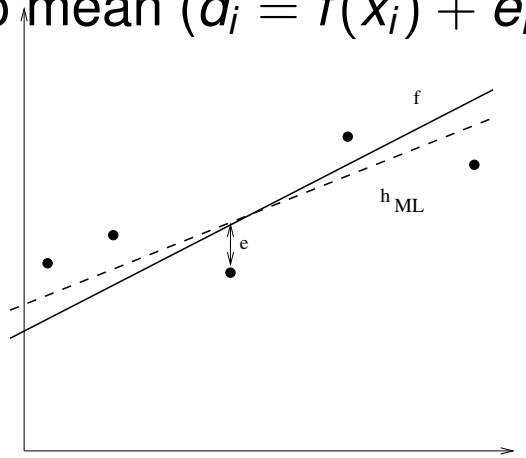
- **network structure**: known or unknown
- **network variables**: observable or partially observable
- in case of known structure and fully observable variables, the conditional probabilities can be estimated as for naive Bayes classifier
- in case of known structure and partially observable variables, the learning problem can be compared to learning weights for an ANN (Russel et al., 1995)
- in case of unknown structure, heuristic algorithms or scoring metric have to be used (Cooper and Herskovits, 1992)

Using the Bayes Approach to Characterize ML-Algorithms

- **Maximum Likelihood and Least-Squared Error**
  - **problem:** learning continuous-valued target functions (e.g. neural networks, linear regression, etc.)
  - *under certain assumptions any learning algorithm that minimizes the squared error between the output hypothesis and the training data, will output a ML hypothesis*
- **Minimum Description Length Principle**
  - closely related to Occam's razor: *choose the shortest explanation for the observed data*
  - consider h_{MAP} by basic concepts of information theory

Maximum Likelihood and Least-Squared Error

- **problem:** learning continuous-valued target functions (e.g. neural networks, linear regression, etc.)
- *under certain assumptions any learning algorithm that minimizes the squared error between the output hypothesis and the training data, will output a ML hypothesis*
- problem setting:
  - $(\forall h \in H)[h : X \rightarrow \Re]$ and training examples of the form $\langle x_i, d_i \rangle$
  - unknown target function $f : X \rightarrow \Re$
  - m training examples, where the target value of each example is corrupted by random noise drawn according to a Normal probability distribution with zero mean ($d_i = f(x_i) + e_i$)



Maximum Likelihood and Least-Squared Error

$$h_{ML} = \underset{h \in H}{argmax}\ p(D|h)$$

The training examples are assumed to be mutually independent given h.

$$h_{ML} = \underset{h \in H}{argmax} \prod_{i=1}^{m} p(d_i|h)$$

Given that the noise e_i obeys a normal distribution with zero mean and unknown variance σ^2, each d_i must also obey a Normal distribution around the true target value $f(x_i)$.

Maximum Likelihood and Least-Squared Error

Density Function of Normal Distribution:

$$\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(x-\mu)^2}$$

Because we are writing the expression for $P(D|h)$, we assume h is the correct description of f. Hence, $\mu = f(x_i) = h(x_i)$

$$h_{ML} = \underset{h \in H}{argmax} \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(d_i - h(x_i))^2}$$

Maximum Likelihood and Least-Squared Error

It is common to maximize the less complicated logarithm, which is justified because of the monotonicity of this function.

$$h_{ML} = \underset{h \in H}{argmax} \sum_{i=1}^{m} \ln \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{2\sigma^2}(d_i - h(x_i))^2$$

The first term in this expression is a constant independent of h and can therefore be discarded.

$$h_{ML} = \underset{h \in H}{argmax} \sum_{i=1}^{m} -\frac{1}{2\sigma^2}(d_i - h(x_i))^2$$

Maximizing this negative term is equivalent to minimizing the corresponding positive term.

$$h_{ML} = \underset{h \in H}{argmin} \sum_{i=1}^{m} \frac{1}{2\sigma^2}(d_i - h(x_i))^2$$

Maximum Likelihood and Least-Squared Error

Finally, all constants independent of h can be discarded.

$$h_{ML} = \underset{h \in H}{argmin} \sum_{i=1}^{m} (d_i - h(x_i))^2$$

$\Rightarrow$ the h_{ML} is one that minimizes the sum of the squared errors

- Why is it reasonable to choose the Normal distribution to characterize noise?
  - good approximation of many types of noise in physical systems
  - Central Limit Theorem shows that the sum of a sufficiently large number of independent, identically distributed random variables itself obeys a Normal distribution
- Only noise in the *target value* is considered, not in the *attributes describing the instances themselves*

Minimum Description Length Principle

- recall **Occam's razor:** *choose the shortest explanation for the observed data*
- here, we consider a Bayesian perspective on this issue and a closely related principle
- **Minimum Description Length (MDL) Principle**
  - motivated by interpreting the definition of h_{MAP} in the light from information theory

$$h_{MAP} = \underset{h \in H}{argmax}\ P(D|h)P(h)$$

$$h_{MAP} = \underset{h \in H}{argmax}\ \log_2 P(D|h) + \log_2 P(h)$$

$$h_{MAP} = \underset{h \in H}{argmin}\ -\log_2 P(D|h) - \log_2 P(h)$$

  - this equation can be interpreted as a statement that short hypotheses are preferred, assuming a particular representation scheme for encoding hypotheses and data

Minimum Description Length Principle

- introduction to a basic result of information theory
  - consider the problem of designing a code C to transmit messages drawn at random
  - probability of encountering message i is p_i
  - interested in the most compact code C
  - Shannon and Weaver (1949) showed that the optimal code assigns $-\log_2 p_i$ bits to encode message i
  - $L_C(i) \approx$ *description length of message i with respect to C*
- $L_{C_H}(h) = -\log_2 P(h)$, where C_H is the optimal code for hypothesis space H
- $L_{C_{D|h}}(D|h) = -\log_2 P(D|h)$, where $C_{D|h}$ is the optimal code for describing data D assuming that both the sender and receiver know hypothesis h

$\Rightarrow$ **Minimum description length principle**

$$h_{MAP} = \underset{h \in H}{\operatorname{argmin}}\, L_{C_H}(h) + L_{C_{D|h}}(D|h)$$

Minimum Description Length Principle

- to apply this principle in practice, specific encodings or representations appropriate for the given learning task must be chosen
- **application to decision tree learning**
  - C_H might be some obvious encoding, in which the description length grows with the number of nodes and with the number of edges
  - choice of $C_{D|h}$?
    - sequence of instances $\langle x_1, ..., x_m \rangle$ is known to the transmitter and the receiver
    - we need only to transmit the classifications $\langle f(x_1), ..., f(x_m) \rangle$
    - if h correctly predicts the classification, no transmission is necessary ($L_{C_{D|h}}(D|h) = 0$)
    - in case of misclassified examples, for each misclassification a message has to be sent that identifies this example (at most $\log_2 m$ bits) as well as its correct classification (at most $\log_2 k$ bits, where k is the number of possible classifications)

Minimum Description Length Principle

- MDL principle provides a way for trading off hypothesis complexity for the number of errors committed by the hypothesis
- **one way of dealing with the issue of overfitting**

Summary

- Bayes classification learning is based on the concept of the probability of hypotheses given observations
- Typically, maximum likelihood (ML) or maximum a posteriori (MAP) hypotheses are learned
- Probabilities are estimated from the training data
- Naive Bayes Classifiers are competitive with decision trees and ANNs
- Bayesian Networks are graphical models where a graph represents the conditional dependence structure between random variables
- The conceptualization of machine learning in the Bayesian framework offers the possibility to formally analyse properties of learning algorithms

Learning Terminology

Naive Bayes

Supervised Learning	unsupervised learning

Approaches:

Concept / **Classification**	Policy Learning
symbolic	**statistical** / neuronal network
inductive	analytical

Learning Strategy:	**learning from examples**
Data:	**categorial/metric features**
Target Values:	**concept**