# Bayesian learning

## SUMMARY



## 1. Bayes theorem

In a probability space $(\Omega, K, P)$ (a *sample space*, an *event space*, and a *probability function*) let us consider a partition $\{H_1, H_2, \dots, H_n\}$ of Ω, with $H_i \epsilon K, P(H_i) > 0$, and $E \epsilon K, P(E) > 0$. Then, the Bayes formula is

$$P(H_j|E) = \frac{P(E|H_j) \cdot P(H_j)}{P(E)} = \frac{P(E|H_j) \cdot P(H_j)}{P(E|H_1) \cdot P(H_1) + \cdots + P(E|H_n) \cdot P(H_n)}$$

**Applications**

- conditional probabilities are used in
  - *classification problems* (ML)
  - *decision theory*
  - *medical diagnosis*

**Example**

Let us consider the following events (in clinical tests, screening programs):
***H***: a person (randomly chosen from a population) has a certain allergy ***A***
***E***: the clinical test returns a *positive* result regarding ***A***
$\bar{E}$: the clinical test returns a *negative* result regarding ***A***

**Known data** (from previous statistics):

- $p = P(H)$ - the probability that a person (randomly selected from the population) has the allergy ***A*;**

- the *sensitivity* of the test $s_1 = P(E|H)$ - the probability of having a positive test when the allergy is actually present;
- the *specificity* of the test $s_2 = P(\bar{E}|\hat{H})$ - the probability of having a negative test when the allergy is not present;
    - $\hat{H}$ - a person does not suffer from ***A***
- the probability of having a *false positive* answer is $P(E|\hat{H}) = 1 - s_2$
- the probability of having a *false negative* answer is $P(\bar{E}|H) = 1 - s_1$

Given $p, s_1, s_2$ we would like to compute the *predictive value* $P(H|E)$ - the probability that a person having a *positive* test is correctly diagnosed with the allergy ***A***.

$$P(H|E) = \frac{P(H \cap E)}{P(E)} = \frac{P(E|H) \cdot P(H)}{P(E|H) \cdot P(H) + P(E|\hat{H}) \cdot P(\hat{H})} = \frac{s_1 \cdot p}{s_1 \cdot p + (1 - s_2) \cdot (1 - p)}$$

**Statistical data**

| | | ACTUAL | | |
|---|---|---|---|---|
| | | ***H***: suffer from ***A*** **(+)** | $\hat{H}$: does not suffer from ***A*** **(-)** | Total |
| **PREDICTION** | ***E***: positive test for ***A*** **(+)** | 400 (true positives - *TP*) | 210 (false positives - *FP*) | 610 |
| | $\bar{E}$: negative test for ***A*** **(-)** | 310 (false negatives - *FN*) | 1200 (true negatives - *TN*) | 1510 |
| | Total | **710** | **1410** | *n*=**2120** |

Based on the statistical data we compute the following:

$$P(H|E) = \frac{400}{610} \approx 0.65 \qquad prec_+ = \frac{TP}{TP + FP}$$

*precision* for the + class, *positive predictive value* (*PPV*)

$$P(\hat{H}|\bar{E}) = \frac{1200}{1510} \approx 0.79 \qquad prec_- = \frac{TN}{TN + FN}$$

*precision* for the - class, *negative predictive value* (*NPV*)

$$P(E|H) = \frac{400}{710} \approx 0.56 \qquad sens = recall_+ = \frac{TP}{TP + FN}$$

*sensitivity*, *recall*, *probability of detection* (POD)

$$P(\bar{E}|\hat{H}) = \frac{1200}{1410} \approx 0.85 \qquad spec = recall_- = \frac{TN}{TN + FP}$$

*specificity*, *true negative rate*

## 2. Bayesian Learning

- Bayesian reasoning provides a probabilistic approach to inference
- eager inductive learning
- **Naïve Bayes Classifier** (NBC) – competitive with other learning algorithms (DTs, NNs) and in some cases outperforms other methods
    - connection to **generative learning** algorithms
        - seeks to model the distribution of inputs of a given class
    - good performance for text categorization and classification
    - classifying news articles (20 classes) – 89% accuracy
- provides practical learning algorithms
    - combining prior knowledge/probabilities with observed data
    - Naïve Bayes learning
        - Naïve Bayes Classifier (NBC)
    - Bayesian Belief Networks – Bayes Nets, Bayesian Networks
- provides a conceptual framework
    - for evaluating other learning algorithms
    - for obtaining additional insight into Occam's razor
- it is based on the Bayes theorem

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

- $P(h)$ = prior probability of hypothesis $h$
- $P(D)$ = prior probability of training data $D$
- $P(h|D)$ = probability of $h$ given $D$
- $P(D|h)$ = probability of $D$ given $h$

[1]

    - $P(h \mid D)$ – probability that $h$ holds given the observed training data $D$
    - $P(D \mid h)$ – probability of observing data $D$ given some world in which hypothesis $h$ holds

- **Bayesian neural networks** (BNNs)
    - combine NNs and stochastic modeling
    - a stochastic artificial neural network trained using Bayesian inference
- **Bayesian Deep Learning**
    - integrate deep learning and Bayesian models
    - uncertainty in learning
    - Bayesian DL and a probabilistic approach of generalization

## 3. Classes of hypotheses

- in ML, we generally search for the most probable hypothesis given the training data
- **Maximum a Posteriori (MAP)** hypothesis

- the most probable hypothesis given the training data

$$h_{MAP} = \arg\max_{h \in H} P(h|D)$$
$$= \arg\max_{h \in H} \frac{P(D|h)P(h)}{P(D)}$$
$$= \arg\max_{h \in H} P(D|h)P(h)$$

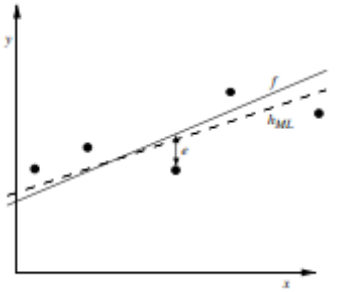
[1]

- **Maximum Likelihood (ML)** hypothesis
  - the hypothesis *h* that maximizes the likelihood of data *D* given *h*

$$h_{ML} = \arg\max_{h_i \in H} P(D|h_i)$$

[1]

  - if $P(h_i) = P(h_j)$ then $h_{MAP} = h_{ML}$
  - the ML hypothesis in learning a real valued function (regression) is the one that minimizes the sum of squared errors [1]




Consider any real-valued target function $f$
Training examples $\langle x_i, d_i \rangle$, where $d_i$ is noisy training value

- $d_i = f(x_i) + e_i$
- $e_i$ is random variable (noise) drawn independently for each $x_i$ according to some Gaussian distribution with mean=0

Then the maximum likelihood hypothesis $h_{ML}$ is the one that minimizes the sum of squared errors:

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

- connection between MAP hypothesis and the MDL principle (*Minimum Description Length*)
  - MDL – a formalization of Occam's razor in which the best hypothesis (a model and its parameters) is the one that leads to the best compression of data
  - the shortest description of the data is the best model

## 3.1 Exemplifying the application of Bayes theorem

Does patient have cancer or not?

A patient takes a lab test and the result comes back positive. The test returns a correct positive result in only 98% of the cases in which the disease is actually present, and a correct negative result in only 97% of the cases in which the disease is not present. Furthermore, .008 of the entire population have this cancer.

$P(\text{cancer}) = 0.008 \qquad P(\neg\text{cancer}) = 0.992$

$h_1 = \text{cancer}, \quad h_2 = \neg\text{cancer}, \quad D = \{+, -\}$

$P(+ \mid \text{cancer}) = 0.98 \qquad P(- \mid \text{cancer}) = 0.02$

$P(+ \mid \neg\text{cancer}) = 0.03 \qquad P(- \mid \neg\text{cancer}) = 0.97$

We have to find $P(h_1 \mid +)$ and $P(h_2 \mid +)$

Applying Bayes theorem (for $D = \{+\}$)

$P(+ \mid \text{cancer}) \cdot P(\text{cancer}) = 0.98 \times 0.008 = 0.0078$

$P(+ \mid \neg\text{cancer}) \cdot P(\neg\text{cancer}) = 0.03 \times 0.992 = 0.0298$

$\Rightarrow h_{MAP} = \neg\text{cancer}$

(We can infer $P(\text{cancer} \mid +) = \frac{0.0078}{0.0078 + 0.0298} = 21\%$

## 3.2. Brute force MAP Learning

- **Training**
  - Choose the hypothesis with the highest posterior probability

$$h_{MAP} = \arg\max_{h \in H} P(h|D)$$
$$= \arg\max_{h \in H} \frac{P(D|h)P(h)}{P(D)}$$
$$= \arg\max_{h \in H} P(D|h)P(h)$$

- **Testing**
  - Given *x*, compute $h_{MAP}(x)$
- **Drawback**
  - Requires to compute all probabilities $P(D|h)$ and $P(h)$

## 3.3. The Bayes Optimal Classifier

- $h_{MAP}$ – the most probable *hypothesis* given the data ***D***
- Given a new instance *x*, what is the most probable classification of *x*?
  - ***$h_{MAP}$(x) is not the most probable classification***!!

$V=\{v_1, v_2 \ldots.\}$ – possible values for the target function

**Answer**

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

**The Bayes optimal classification of *x***

$$\arg\max_{v_j \in V} \sum_{h_i \in H} P(v_j|h_i)P(h_i|D) \qquad \textbf{(*)}$$

- any system that classifies new instances according to (*) is called a **Bayes optimal learner**
  - no other classification method using the same hypothesis space and the same prior knowledge can outperform the method, on average
  - maximizes the probability that the new instance is classified correctly, given the available data

**Example**

Let us consider three hypotheses:

$$P(h_1|D) = .4,$$
$$P(h_2|D) = .3,$$
$$P(h_3|D) = .3,$$

Given new instance $x$ which is classified
+ by $h_1$
- by $h_2$
- by $h_3$

$$P(-|h_1) = 0,\ P(+|h_1) = 1$$
$$P(-|h_2) = 1,\ P(+|h_2) = 0$$
$$P(-|h_3) = 1,\ P(+|h_3) = 0$$

what is the most probable classification of $x$?

**Correct answer**: - is the **Bayes optimal classification** of $x$ (even if $h_{MAP}(x)$=+)

- $h_1$ is the MAP hypothesis
- $h_1(x)$ = +
  $\Rightarrow$ P($f(x)$ = +) = 0.4
  and
  P($f(x)$ = - ) = 0.6

**Bayes optimal classification**

$$\sum_{h_i \in H} P(+|h_i)P(h_i|D) = .4$$
$$\sum_{h_i \in H} P(-|h_i)P(h_i|D) = .6$$

and

$$\arg\max_{v_j \in V} \sum_{h_i \in H} P(v_j|h_i)P(h_i|D) = -$$

## 3.4. Gibbs Optimal Classifier [1]

The Bayes optimal classifier provides the best result but is expensive for many hypotheses.

- it computes the posterior probabilities for every hypothesis in $H$ and combines the predictions of each hypothesis to classify each new instance

### Gibbs algorithm

- an alternative (less optimal) method:

  1. Choose a hypothesis $h$ from ***H*** at random, according to the posterior probability distribution over ***H*** (P($h$ | D) - D: data set; $h$: hypothesis)
  2. Use $h$ to predict the classification of a new instance $x$.

- under certain conditions the expected **misclassification** error for Gibbs algorithm **is at most twice** the expected error of the Bayes optimal classifier.

## 4. Brute force Bayesian Learning

- let us assume that an instance $x$ is described by a vector $<a_1, a_2, \ldots a_n>$ of attribute values
- the learning task is a classification one, there are $m$ possible classes $\{c_1, c_2, \ldots, c_m\}$
- during training we compute the probabilities $P(c_i)$ and $P(x|c_i)$, for all training instances $x$ and for all classes $c_i \in \{c_1, c_2, \ldots, c_m\}$
- for a new query instance $x=<a_1, a_2, \ldots a_n>$, the most probable class $c_{MAP}$ will be computed as the MAP hypothesis

    - $c_{MAP} = \text{argmax } P(c_i \mid a_1, a_2, \ldots a_n) = \text{argmax } P(a_1, a_2, \ldots a_n \mid c_i) \cdot P(c_i)$

- the result of the Bayesian inference depends strongly on the prior probabilities which must be available in order to apply the method
- **Problem with "brute force"**
    - it cannot generalize for unseen examples $x^{new}$, since it does not have estimates $P(c_i \mid x^{new})$
    - brute force does not have any bias
    - in order to make the learning possible, we have to introduce a bias ⇒ Naïve Bayes Classifier (NBC)

## 5. Naïve Bayes Classifier

- when to use NBC
    - the target function $f$ is discrete valued and takes values from a finite set $V=\{v_1, v_2, \ldots, v_m\}$
    - an instance is described by a conjunction of attribute values $x=<a_1, a_2, \ldots a_n>$
    - attribute can take discrete values
    - moderate or large training data is available
    - the attributes $<a_1, a_2, \ldots a_n>$ that describe the instances are conditionally independent with respect to their target classification

$$P(a_1, a_2 \ldots a_n | v_j) = \prod_i P(a_i | v_j)$$ [1]

        - the conditional independence assumption of attributes is often violated in practice, but it works well
- for a new instance $x$ to be classified, the most probable value of $f(x)$ is

$$v_{NB} = \underset{v_j \in V}{\text{argmax}} P(v_j) \prod_i P(a_i | v_j)$$ [1]

- NBC does not search through the hypotheses space
    - the output hypothesis is simply formed by estimating the parameters $P(v_j)$ and $P(a_i \mid v_j)$
    - eager learner
- the NBC classification algorithm

Naive_Bayes_Learn($examples$)

For each target value $v_j$

$\hat{P}(v_j) \leftarrow$ estimate $P(v_j)$

For each attribute value $a_i$ of each attribute $a$

$\hat{P}(a_i|v_j) \leftarrow$ estimate $P(a_i|v_j)$

Classify_New_Instance($x$)

$$v_{NB} = \underset{v_j \in V}{\operatorname{argmax}} \hat{P}(v_j) \prod_{a_i \in x} \hat{P}(a_i|v_j)$$

[1]

- $\hat{P}(v_j)$ is computed as the frequency with which the target value $v_j$ occurs in the training data: $n_{vj}/n$
  - $n_{vj}$ is the number of training examples for which the target value is $v_j$
  - $n$ is the number of training instances
- $\hat{P}(a_i|v_j)$ is computed as $n_{aivj}/n_{vj}$ where
  - $n_{aivj}$ is the number of values for which the value of the i-th attribute is $a_i$ and the target value is $v_j$

**Example – learn the PlayTennis concept**

| Day | Outlook | Temperature | 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 |

- train an NBC on this dataset
- use the trained classifier to classify a new instance

$x$ = (Outlook = sunny, Temperature = cool, Humidity = high, Wind = strong)

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

$$= \underset{v_j \in \{yes,no\}}{\text{argmax}} P(v_j) \quad P(Outlook = sunny|v_j) P(Temperature = cool|v_j)$$

$$P(Humidity = high|v_j) P(Wind = strong|v_j) \quad (6.21)$$

[1]

We have to compute, from the training data, the needed probabilities

$$P(PlayTennis = yes) = 9/14 = .64$$
$$P(PlayTennis = no) = 5/14 = .36$$

$$P(Wind = strong|PlayTennis = yes) = 3/9 = .33$$
$$P(Wind = strong|PlayTennis = no) = 3/5 = .60$$

⇒

$$P(yes)\ P(sunny|yes)\ P(cool|yes)\ P(high|yes)\ P(strong|yes) = .0053$$
$$P(no)\ P(sunny|no)\ P(cool|no)\ P(high|no)\ P(strong|no) = .0206$$

⇒ The classification of $x$ is ***no***.

- **NBC: the problem of unseen data**
  - if none of the training instances with target value $v_j$ have the attribute value $a_i$ ⇒
    $\hat{P}(a_i|v_j) = 0$, and...
    $\hat{P}(v_j) \prod_i \hat{P}(a_i|v_j) = 0$
    [1]
  - typical solution is the so called "**m-estimate of probability**"
    - $\hat{P}(a_i|v_j)$ is computed as $(n_{aivj}+1)/(n_{vj}+m)$ where
      - $m$ is the number of possible values of attribute $a_i$
      - for each class, we consider adding $m$ virtual examples (one for each possible value of attribute $a_i$)
- NBC is successfully applied for text classification (see [1], e.g. spam filtering)

## 6. Bayesian Belief Networks (Bayes Nets, Bayesian networks) [1, 3]

- the NBC assumption of conditional independence of attributes is too restrictive
- Bayesian Belief Networks describe conditional independence among *subsets* of variables

$$P(X|Y,Z) = P(X|Z)$$

- gradient ascent training of Bayesian networks
  - objective function that is maximized is $P(D \mid h)$ => searching for the ML hypothesis
  - following the gradient of $\log P(D \mid h)$
- **Hidden Markov Models** (HMMs)
  - particular kind of Bayesian networks
  - HMMs – tool for representing probability distributions over sequences of observations
    - is a **probabilistic** model of the joint probability of a collection of random variables with both observations and states
    - tools for modelling time series data
    - generative models
    - **applications**
      - NLP
      - text generation
      - bioinformatics, computational biology (biological sequence analysis)
      - speech recognition
  - **Baum-Welch** algorithm for training a HMM
    - finds values for HMM parameters that best fit the observed data
    - a dynamic programming approach and a special case of the EM algorithm
  - **Viterbi algorithm**
    - a dynamic programming algorithm for obtaining the maximum aposteriori probability estimate of the *most likely sequence of hidden states* (called the Viterbi path) that results in a sequence of observed events.
- Dynamic Bayesian Belief Networks
- The **Expectation Maximization** (EM) algorithm
  - learning in the presence of unobserved variables (variables are partially observable - handling missing data)
  - determine MAP estimates for unobserved variables in statistical models
    - alternative formulation of **maximum likelihood** for searching for the appropriate model parameters in the presence of latent (hidden) variables
    - it provides a framework to find the **local maximum likelihood** parameters of a statistical model and infer latent variables in cases where data is missing or incomplete
  - used to train Bayesian Belief Networks, RBFNs
  - foundation for many unsupervised clustering algorithms
  - the basis for the **Baum-Welch** forward-backward algorithm for training Partially Observable Markov Models
  - it can be used to fill in the missing data in a sample
  - **applications**
    - learning useful representations
      - unsupervised learning (Neural EM)
    - Deep EM for image reconstruction

## 7. Applications of Bayesian Learning

- **NBC**
  - text categorization (e.g., spam classification)
  - sentiment analysis and emotion recognition

    - optimizing treatment decisions (medicine)
    - word disambiguation
    - outlier detection
- **Bayesian networks**
    - medical diagnosis
    - biological systems

## 8. Other Bayesian learning related research topics

- Boosted NBC, Fuzzy NBC, Lazy NBC
- NBC for regression
- Multinomial NBC, Bernoulli NBC, Gaussian NBC (for handling continuous features)
- Hybrid models
    - DT+NBC, SVM+NBC
- Bayesian neural networks (BNNs)
    - combine NNs and stochastic modeling
- Bayesian Deep Learning
    - integrate deep learning and Bayesian models
    - Bayesian CNNs
        - probability distribution over the weights
- Bayesian Gaussian Mixture Models
    - Bayesian inference for GMMs
        - GMM
            - a probabilistic model that assumes all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters
            - soft clustering ML method used to determine the probability each data point belongs to a given cluster
    - a form of unsupervised learning
    - useful in fitting multi-modal data for tasks such as clustering, data compression, outlier detection, or generative classifiers.
- Bayesian HMMs
    - financial data

**[SLIDES]**

- Bayesian learning (T. Mitchell) [1]
- Discrete Bayesian classifiers [3]

**[READING]**

- Bayesian learning (T. Mitchell) [1]
- Naïve Bayes Classifier (Zhang et al.) [2]

**Bibliography**

[1] Mitchell, T., *Machine Learning*, McGraw Hill, 1997 (available at www.cs.ubbcluj.ro/~gabis/ml/ml-books)

[2] Aston Zhang, Zachary C. Lipton, Mu Li, and Alexander J. Smola, *Dive into Deep Learning*, 2020 (http://d2l.ai/)