Bayes’ Theorem

Dr. Kuppusamy .P
Associate Professor / SCOPE

Bayes’ Theorem Introduction

- Bayes theorem determines the probability of an event with random knowledge.
- It is used to calculate the probability of occurring one event while another one event already occurred.
- It is a best method to relate the condition probability and marginal probability.

Features of Bayes’ Theorem

- 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.
- Bayesian methods can accommodate hypotheses that make probabilistic predictions.
- New instances can be classified by combining the predictions of multiple hypotheses, weighted by their probabilities.

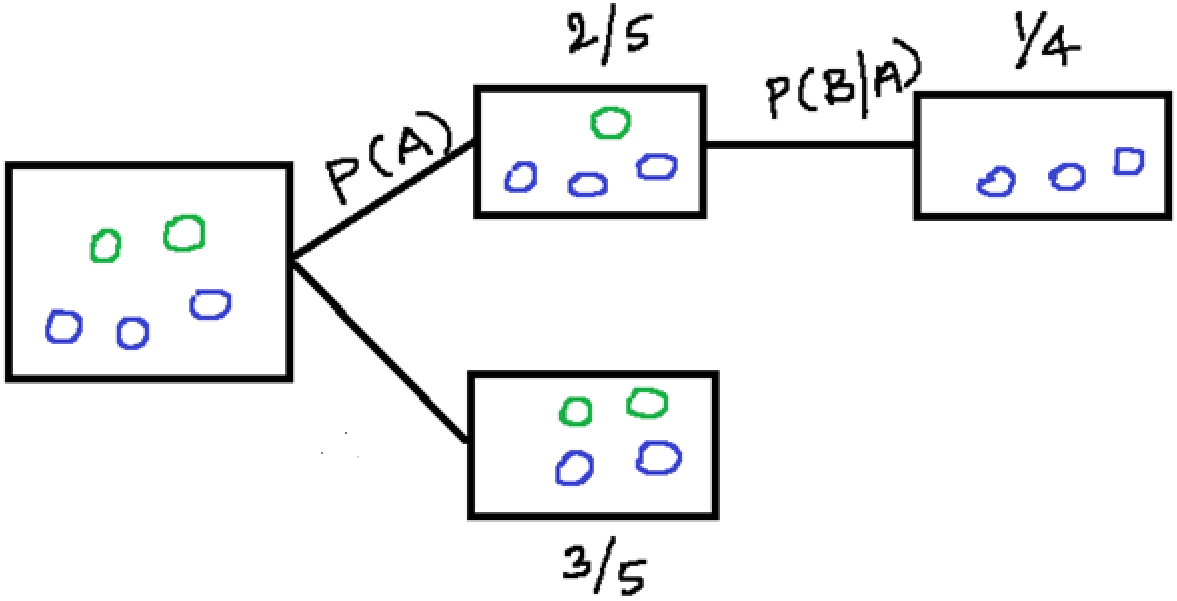
Bayes' Theorem Introduction

Independent Events:

Tossing the Two coins. The outcome of each coin is independent of each other.

Dependent Events:

- Consider 5 Balls in basket.
- P(A) is Green ball taking out from the basket
- P(B|A) is taking one more ball continuously from the basket i.e., this event is dependent on first event.



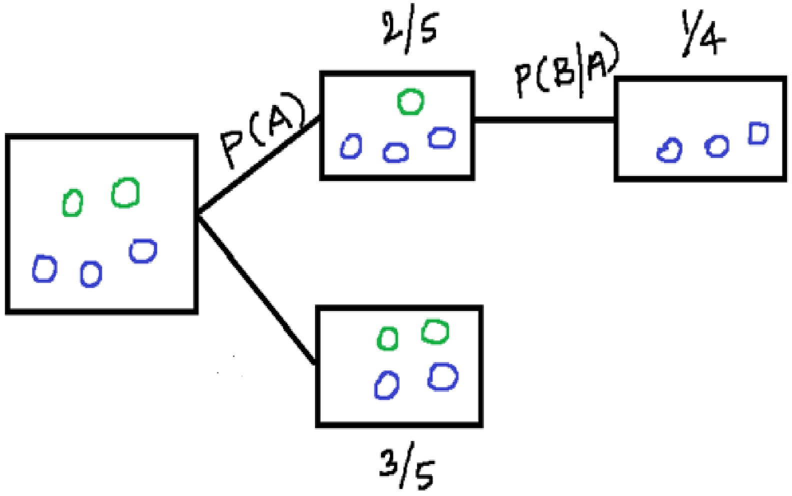
Bayes' Theorem Introduction

- The **joint probability** is the probability of two (or more) simultaneous events.
- A and B are events from two dependent random variables, e.g., X and Y denoted as $P(A \cap B)$ or P(A, B).
- The joint probability can be calculated using the conditional probability as

$$P(A \cap B) = P(A \mid B) * P(B) \quad \text{(product rule)}$$

$$P(B \cap A) = P(B \mid A) * P(A)$$

$$P(B \cap A) = (1/4) * (2/5) = 1/10$$



- The joint probability is symmetrical, $P(A \cap B) = P(B \cap A)$

- The **conditional probability** is the probability of one event given the occurrence of another event.
- A and B are events from two dependent random variables e.g., X and Y denoted as *P(A given B)* or *P(A | B).*
- The conditional probability can be calculated using the joint probability as $P(A \mid B) = P(A \cap B) / P(B)$
- The conditional probability is **not symmetrical** P(A | B) != P(B | A)

Joint distribution

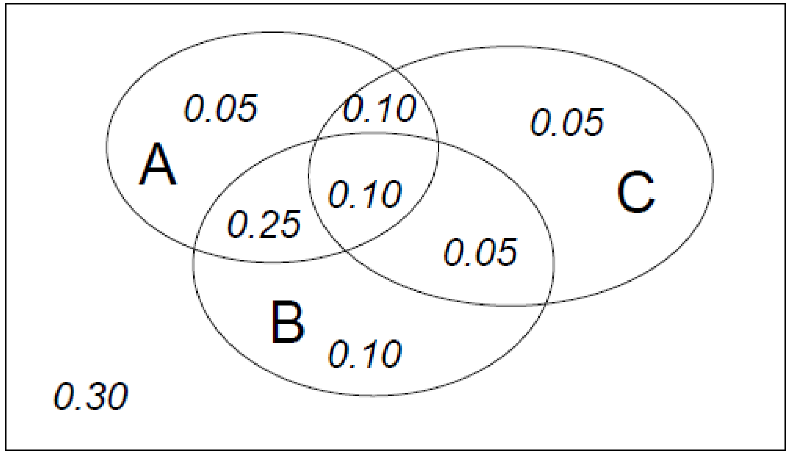
A	B	C	Prob
0	0	0	0.30
0	0	1	0.05
0	1	0	0.10
0	1	1	0.05
1	0	0	0.05
1	0	1	0.10
1	1	0	0.25
1	1	1	0.10

- Making a joint distribution of M variables

1. Make a truth table listing all combinations

2. For each combination of values, how it is probable.

3. Probability must sum to 1



Bayes’ Theorem

- Another approach (**Bayes’ Theorem**) to calculate Conditional Probability:
- One conditional probability can be calculated using the other conditional probability as follows:

$$P(\mathrm{A}|\mathrm{B}) = \frac{\mathrm{P(B|A)} * \mathrm{P(A)}}{\mathrm{P(B)}}$$

- In reverse:

$$P(\mathrm{B}|\mathrm{A}) = \frac{\mathrm{P(A|B)} * \mathrm{P(B)}}{\mathrm{P(A)}}$$

- Bayes Theorem is an approach for calculating a conditional probability **without** the joint probability.
- $P(\mathrm{A}|\mathrm{B}) = \frac{P(A \cap B)}{\mathrm{P(B)}}$ ➔ $P(A \cap B) = P(\mathrm{A}|\mathrm{B}) * P(B)$
- $P(\mathrm{B}|\mathrm{A}) = \frac{P(B \cap A)}{\mathrm{P(A)}}$ ➔ $P(B \cap A) = P(\mathrm{B}|\mathrm{A}) * P(\mathrm{A})$
- $P(B \cap A) = P(A \cap B)$
- P(B | A) * P(A) = P(A | B) * P(B)
- $P(\mathrm{A}|\mathrm{B}) = \frac{\mathrm{P(B|A)} * \mathrm{P(A)}}{\mathrm{P(B)}}$

  P(A | B) – Posterior probability, P(B) - marginal probability, P(B | A) – likelihood, P(A) - prior probability
- P(A) is an initial probability value originally obtained before training sample is obtained
- P(A | B) is a probability value that has been revised by using additional information (training sample) that is later obtained.

Other forms of Bayes rule

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

$$P(A|B,X) = \frac{P(B|A,X)P(A,X)}{P(B,X)}$$

$$P(A|B) = \frac{P(B|A)P(A)}{P(B|A)P(A) + P(B|{\sim}A)P({\sim}A)}$$

Example: Applying Bayes rule

$$P(A|B) = \frac{P(B|A)P(A)}{P(B|A)P(A) + P(B|\sim A)P(\sim A)}$$

- A = Person have the flu
  B = Person just coughed
- Assume:
  - $P(\text{flu}) = P(A) = 0.05$
  - $P(\text{cough}|\text{flu}) = P(B|A) = 0.8$
  - $P(\text{cough}|\sim\text{flu}) = P(B|\sim A) = 0.2$
- Compute P(flu | cough) = P(A|B)?

$$P(A|B) = \frac{0.8 \times 0.05}{0.8 \times 0.05 + 0.2 \times 0.95} \sim 0.17$$

= 0.17

Probability Formulas

Product rule:

- Probability of a conjunction of two events A and B

$$P(A \cap B) = \mathrm{P(A \mid B) * P(B)}$$

$$P(B \cap A) = P(\mathrm{B \mid A}) * \mathrm{P(A)}$$

-

Sum rule:

- Probability of a disjunction of two events A and B
- $P(A \cup B)$ = P(A) + P(B) - P($A \cap$ B)

Bayes theorem:

- The posterior probability of A given B

$$P(\mathrm{A|B}) = \frac{\mathrm{P(B|A) * P(A)}}{\mathrm{P(B)}}$$

Theorem of total probability:

- if events A_1,. . . , A_n, are mutually exclusive with $\sum_{i=1}^{n} P(A_i) = 1,$ then $\mathrm{P(B)} = \sum_{i=1}^{n} \mathrm{P(B|}A_i) * \mathrm{P(}A_i)$

Total Probability

- Let $E_1, E_2, \ldots \ldots E_n$ be *n* mutually exclusive and exhaustive events associated with a random experiment. If 'A' is any event which occurs with E_1 or E_2 $or \ldots \ldots E_n$, then

$$P(A) = P(E_1).P(A|E_1) + P(E_2).P(A|E_2) + \cdots \ldots \ldots \ldots . + P(E_n).P(A|E_n)$$

Example:

The first bag contains 4 red and 3 black balls. A second bag contains 2 red and 4 black balls. One bag is selected randomly. From the selected bag, one ball is drawn. What is the probability that the ball drawn is red?

This problem can be answered using the concept of Total Probability

E_1 =Selecting bag *I* ; E_2 =Selecting bag *II* ; A = Drawing the red ball

Thus, $P(A) = P(E_1).P(A|E_1) + P(E_2).P(A|E_2)$

$P(A|E_1)$ = Probability of drawing red ball when first bag has been chosen and

$P(A|E_2)$ = Probability of drawing red ball when second bag has been chosen

Reverse Probability

Example:

A bag (Bag I) contains 4 red and 3 black balls. A second bag (Bag II) contains 2 red and 4 black balls. You have chosen one ball at random. It is found as red ball. What is the probability that the ball is chosen from Bag I?

Here,

E_1 = Selecting bag I

E_2 = Selecting bag II

A = Drawing the red ball

We have to determine P(E_1|A). Such a problem can be solved using Bayes' theorem of probability.

Reverse Probability

- Let $E_1, E_2, \ldots \ldots E_n$ be n mutually exclusive and exhaustive events associated with a random experiment. If A is any event which occurs with $E_1 \; or \; E_2 \; or \; \ldots \ldots E_n$, then

$$P(E_i|A) = \frac{P(E_i).P(A|E_i)}{\sum_{i=1}^{n} P(E_i).P(A|E_i)}$$

  - P(A) and P(B) are called prior probabilities
  - P(A|B), P(B|A) are called posterior probabilities

Example: Prior versus Posterior Probabilities

- This table shows that the event Y has two outcomes namely A and B, which is dependent on another event X with various outcomes like x_1, x_2 and x_3.
- **Case1:** Suppose, we don't have any information of the event A. Then, from the given sample space, we can calculate $P(Y = A) = \frac{5}{10} = 0.5$
- **Case2:** Now, suppose, we want to calculate $P(X = x_2|Y = A) = \frac{2}{5} = 0.4$.

The later is the conditional or posterior probability, where as the former is the prior probability.

X	Y
x_1	A
x_2	A
x_3	B
x_3	A
x_2	B
x_1	A
x_1	B
x_3	B
x_2	B
x_2	A

Bayes Concept Learning Approach

- **Finds Best hypothesis:**

  Most probable hypothesis in hypothesis space H given training data D i.e., $P(\mathrm{h}|\mathrm{D})$.
- P(h) denotes the initial probability of hypothesis h before observing the training data. P(h) is the prior probability of h and may reflect any background knowledge to the model in deciding h that h is a correct hypothesis.
- If model do not have such prior knowledge, then it might simply assign the same prior probability to each candidate hypothesis.
- P(D) denotes the **prior probability** that training data D will be observed (i.e., the probability of D given no knowledge about the hypothesis).
- P(D|h) to denote the probability of observing data D given some world in which hypothesis h holds.
- The probability P(h|D) denotes h given the observed training data D. P(h|D) is the posterior probability of h, because it reflects our confidence that h holds after processing the training data D.
- Compute the posterior probability P(h|D) reflects the influence of the training data D, in contrast to the prior probability P(h) , which is independent of D.

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

- From the above, $P(\mathrm{h}|\mathrm{D})$ increases when increasing $P(\mathrm{D}|\mathrm{h})$ and $P(\mathrm{h})$. Here P(D) is independent of h.

Maximum A Posteriori Probability

- Mostly the learner considers set of candidate hypotheses H and is interested in finding the most probable hypothesis $h \in H$ given the observed data D (or at least one of the maximally probable if many h exists).
- Any such **Maximally Probable Hypothesis** is a **Maximum A Posteriori** (MAP) hypothesis.
- Determine the MAP hypotheses using Bayes theorem to calculate the posterior probability of each candidate hypothesis.
- $h_{MAP} \equiv \underset{h \in H}{argmax}\ P(h|\mathrm{D})$
- $h_{MAP} \equiv \underset{h \in H}{argmax}\ \frac{\mathrm{P(D|h)} * \mathrm{P(h)}}{\mathrm{P(D)}}$
- $h_{MAP} \equiv \underset{h \in H}{argmax}\ \mathrm{P(D|h)} * \mathrm{P(h)}$

- In above equation, $P(\mathrm{D})$ is omitted due to it is constant and common to all hypothesis i.e., independent of h.
- In some cases, assume that every hypothesis in H is equally probable a priori i.e., (P(h_i) = P(h_j) for all h_i and h_j in H).
- So, above equation is simplified by just considering the term P(D|h) to find the most probable hypothesis. P(D|h) is the likelihood of the data D given h, and any hypothesis that maximizes P(D|h) is a Maximum Likelihood (ML) hypothesis. $h_{ML} \equiv \underset{h \in H}{argmax}\ \mathrm{P(D|h)}$

Brute-Force MAP Learning Algorithm

Deriving P(D) from the theorem of total probability

$$\mathrm{P(D)} = \sum_{h_i \in H} \mathrm{P(D|h)} * \mathrm{P(h)}$$

$$\mathrm{P(D)} = \sum_{h_i \in VS_{H,D}} 1 * \frac{1}{|H|} + \sum_{h_i \notin VS_{H,D}} 0 * \frac{1}{|H|}$$; if consistent 1, inconsistent 0

$$\mathrm{P(D)} = \sum_{h_i \in VS_{H,D}} 1 * \frac{1}{|H|}$$

$$\mathrm{P(D)} = \frac{|VS_{H,D}|}{|H|}$$

- $VS_{H,D}$ is the subset of consistent hypotheses with D from H (Version Space of H with respect to D)

Brute-Force Bayes Concept Learning

- Bayes theorem provides a procedure to calculate the posterior probability of each hypothesis given the training data.
- Based on this, learning algorithm calculates the probability for each possible hypothesis, then outputs the most probable.
- Assume the learner considers some finite hypothesis space H defined over the instance space X, in which the task is to learn some target concept c : X → {0,1}.
- Let's consider training examples $((x_1, d_1) \ldots (x_m, d_m))$; x_i - instance from X, d_i - target of x_i (i.e., $d_i = c(x_i)$).
- Simplify it, Dataset can be written as instances $X = (x_l \ldots x_m)$ and target values $D = (d_l \ldots d_m)$.

Brute-Force MAP Learning Algorithm

- Design a concept learning algorithm to output the maximum a posteriori hypothesis based on Bayes theorem as:
- For each hypothesis h in H, calculate the posterior probability $$P(\mathrm{h}|\mathrm{D}) = \frac{\mathrm{P(D|h)} * \mathrm{P(h)}}{\mathrm{P(D)}}$$
- Output the hypothesis h_{MAP} with the highest posterior probability $$h_{MAP} \equiv \underset{h \in H}{argmax}\, P(h|\mathrm{D})$$
- Calculate P(h|D) for each hypothesis
- But it is Impractical for larger hypothesis spaces due to need of more computational power.
- To overcome it, use the prior knowledge to decide the values of P(D|h) and P(h).

Brute-Force MAP Learning Algorithm

Choose P(h) and P (D|h) to be consistent with following Assumptions:

1. The training data D is noise-free x_i (i.e., $d_i = c(x_i)$)
2. The target concept c is contained in the hypothesis space H.
3. We have no a priori reason to believe that any hypothesis is more probable than another.

Choose P(h) value using given Assumptions

- Given no prior knowledge that one hypothesis (classifier) is more likely than another, it is reasonable to assign the same prior probability is assigned to every hypothesis h in H.
- Since target concept is assumed to be contained in H, we should require that these prior probabilities sum to 1
- So, combine these constraints that we should choose, $P(\mathrm{h}) = \frac{1}{|H|}$ for all h in H

- **Choose P(D|h) value using given Assumptions:**
- **P(D|h)** is the probability of observing the target values D = (d_l . . . d_m) of fixed instances X = (x_l . . . x_m) given a world in which hypothesis h holds (ie, h is the correct description of the target concept c)
- Assuming noise-free training data $\mathbf{P(D|h)} = \begin{Bmatrix} 1 & if\ \mathrm{d_i} = \mathrm{h(x_i)}\ \ for\ all\ \ \mathrm{d}_i \mathrm{in\ D} \\ 0 & Otherwise \end{Bmatrix}$
- i.e., the probability of data D given hypothesis h is 1 if D is consistent with h, and 0 otherwise.

Bayes Concept Learning

Temp	Hypothesis 0	Hypothesis 1	Hypothesis 2	Hypothesis 3
Hot	No	No	Yes	Yes
Cold	No	Yes	No	Yes
P(D|h)	**0**	**0**	**1**	**0**

Consider the temp has binary values. Target **P(D|h)** is given.

$$P(\mathrm{h_0|D}) = \frac{\mathrm{P(D|h_0)} * \mathrm{P(h_0)}}{\mathrm{P(D)}} = \frac{0*\frac{1}{4}}{\frac{1}{4}} = 0;$$

$$P(\mathrm{h_1|D}) = \frac{\mathrm{P(D|h_1)} * \mathrm{P(h_1)}}{\mathrm{P(D)}} = \frac{0*\frac{1}{4}}{\frac{1}{4}} = 0$$

$$P(\mathrm{h_2|D}) = \frac{\mathrm{P(D|h_2)} * \mathrm{P(h_2)}}{\mathrm{P(D)}} = \frac{1*\frac{1}{4}}{\frac{1}{4}} = 1 ;$$

$$P(\mathrm{h_3|D}) = \frac{\mathrm{P(D|h_3)} * \mathrm{P(h_3)}}{\mathrm{P(D)}} = \frac{0*\frac{1}{4}}{\frac{1}{4}} = 0$$

Brute-Force MAP Learning Algorithm

1. Apply Bayes rule to compute posterior probability P(h|D) for each hypothesis h given the training data D.

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

$$\mathrm{P(D)} = \frac{|VS_{H,D}|}{|H|}$$

$VS_{H,D}$ is the subset of consistent hypotheses with D from H (Version Space of H with respect to D)

- Assume, h is inconsistent with the training data D. Then $P(\mathrm{h}|\mathrm{D}) = \frac{0* \mathrm{P(h)}}{\mathrm{P(D)}} = 0$

- Assume, h is consistent with the training data D. Then $P(\mathrm{h}|\mathrm{D}) = \frac{1*\frac{1}{|H|}}{\frac{|VS_{H,D}|}{|H|}} = \frac{1}{|VS_{H,D}|}$

2. So, posterior probability P(h|D) under the assumed P(h) and P(D|h).

$$P(\mathrm{h}|\mathrm{D}) = \left\{\begin{array}{ll} \frac{1}{|VS_{H,D}|} & if\ h\ is\ consistent\ with\ D \\ 0 & Otherwise \end{array}\right\}$$

Brute-Force Bayes Learning

x_0	x_1	x_2	Hypothesis						
			f_0	f_1	f_2	f_3	f_4	…	f_{255}
0	0	0	0	1	0	1	0	…	1
0	0	1	0	0	1	1	0	…	1
0	1	0	0	0	0	0	1	…	1
0	1	1	0	0	0	0	0	…	1
1	0	0	0	0	0	0	0	…	1
1	0	1	0	0	0	0	0	…	1
1	1	0	0	0	0	0	0	…	1

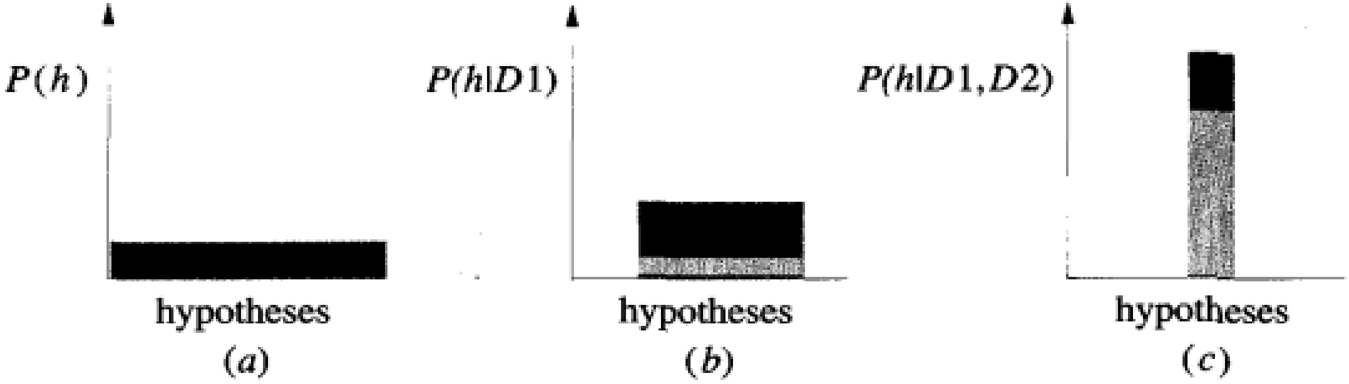
- Consider training data D<(0, 0, 0), 0> and <(0, 0, 1), 0>.
- Assume f_0, f_4, … are consistent with D (Likewise there is 64 consistent hypothesis)i.e., VS.
- Assume f_1, f_2, f_3, …. are consistent with D.

$$P(\mathrm{f}_0|\mathrm{D}) = \frac{\mathrm{P(D|f_0)} * \mathrm{P(f_0)}}{\mathrm{P(D)}} = \frac{1 * \frac{1}{|H|}}{\frac{|VS_{H,D}|}{|H|}} = \frac{1 * \frac{1}{256}}{\frac{64}{256}} = \frac{1}{64}\ ; P(\mathrm{f}_1|\mathrm{D}) = \frac{\mathrm{P(D|f_1)} * \mathrm{P(f_1)}}{\mathrm{P(D)}} = \frac{0 * \frac{1}{256}}{\frac{64}{256}} = 0$$

MAP hypothesis and Consistent Learners

- A learning algorithm is a consistent learner if it outputs a hypothesis that commits zero errors over the training examples.
- Every consistent learner outputs a MAP hypothesis if
  - we assume a uniform prior probability distribution over H (i.e., P(h_i) = P(h_j) for all i, j) and
  - we assume deterministic, noise-free training data (i.e., P(D|h)=1 if D and h are consistent and 0 otherwise)

Evolution of Posterior Probabilities



- **While increasing the training samples,**
  a) uniform priors to each hypothesis
  b) As training data increases first to D1
  c) then to D1^D2 posterior probs for inconsistent hypotheses becomes zero

Example: Two categories, binary-valued attribute

Consider the given example. Apply the bayes theorem to compute the probability of enjoying the game.

Temp	EnjoyGame
Hot	Yes
Hot	Yes
Hot	No
Cold	Yes
Hot	Yes
Cold	No
Cold	No
Cold	No
Cold	Yes

- Prob (Hot/Yes)= (3/5) = 0.6 ; Prob (Cold/No)= (3/4) = 0.75 ;
- Prob (Yes) = (5/9) = 0.56 ; Prob(No) = (4/9) = 0.44
- Prob (Hot) = Prob(Hot/Yes) * P(Yes) + Prob(Hot/No) * Prob(No)

  = 0.6 x 0.56 + 0.25 x 0.44

  =0.336+0.11=0.447

- **Bayes Optimal Decision:**

$$\text{Prob (Yes/Hot)} = \frac{\text{Prob (Hot|Yes)} * \text{P(Yes)}}{\text{P(Hot)}}$$

$$= 0.6 * 0.56 / 0.447 = 0.75$$

Bayes Optimal Rule Example: Medical Diagnosis

Two alternative hypotheses:

- Patient has a particular form of cancer
- Patient not affected with cancer

Available Data: a particular laboratory test

- Lab-test is either +(positive) or negative (-)
- Prior knowledge:
- Over entire population only 0.008 have this disease

Also, the lab test is an imperfect indicator of the disease. The test returns a correct positive result 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.

Known probabilities

- P(cancer)=.008, P(~cancer)=.992
- P(+/cancer)=.98 (True Positive), P(-/cancer)=.02 (False Negative)
- P(+/~cancer)=.03 (False Positive), P(-/~cancer)=.97 (True Negative)

	Predicted Output	
Lab Test (Actual Output)	Positive	Negative
Positive	0.98	0.03
Negative	0.02	0.97

Bayes Optimal Rule Example: Medical Diagnosis

- Observed data: lab test is positive (+)
  - P(+/cancer) * P(cancer) = (.98) * .008 = .0078
  - P(+/~cancer) * P(~cancer) = (.03) * .992 = .0298
- Therefore h_{MAP}= ~cancer
- Exact a posteriori probabilities
  - P(cancer/+) = .0078/(.0078 + .0298) = .21
  - P(~cancer/+) = .79
- The probability of cancer increased from .008 to .21 after the positive lab test, but still it is much more likely that it is not cancer

MAXIMUM LIKELIHOOD (ML)

- Under certain assumptions, Any learning algorithm that minimizes the squared error between the hypothesis predictions and the training data that results a maximum likelihood hypothesis. Bayesian algorithm works based on the above principle.
- **Maximum Likelihood** hypothesis minimizes the sum of the squared errors between the observed training values d_i and the hypothesis predictions $h(x_i)$.

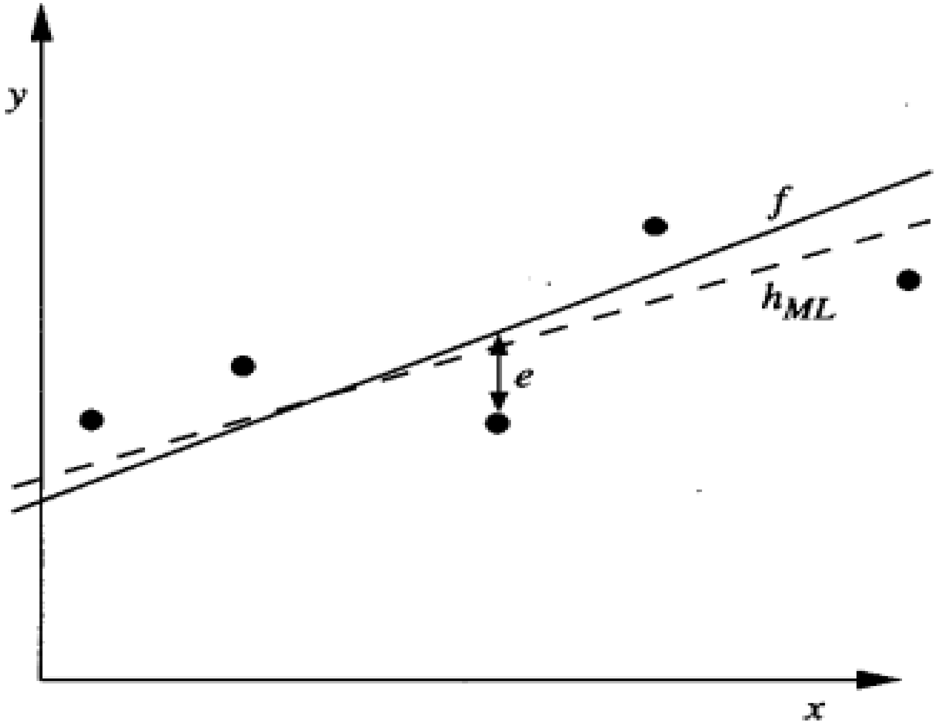
Assumption

- The observed training values d_i are generated by adding random noise to the true target value. Here the random noise is drawn independently for each example from a Normal distribution with zero mean.

Example

- Learner L considers an instance space X and a hypothesis space H consisting **real-valued** functions defined over X (i.e., each h in H is a function of the form $h: X \rightarrow \mathbb{R}$, $\mathbb{R}$ represents the set of real numbers).
- The problem faced by L is to learn an unknown target function $f: X \rightarrow \mathbb{R}$, drawn from H.
- In set of m training examples, the target value of each example is corrupted by random noise drawn according to a Normal probability distribution.
- More precisely, each training example is a pair of the form (x_i, d_i), here $d_i = f(x_i) + e_i$. Here $f(x_i)$ is the noise-free value of the target function and e_i is a random variable representing the noise.
- The task of the learner is to output a maximum likelihood hypothesis, or, equivalently, a MAP hypothesis assuming all hypotheses are equally probable a priori.

Example: Linear algorithm learns Real valued function



- The linear target function *f* corresponds to the solid line.
- The training examples (x_i ,d_i) are assumed to have Normally distributed noise e_i with zero mean added to the true target value f (x_i) . The dashed line corresponds to the linear function that minimizes the sum of squared errors.
- Therefore, it is the maximum likelihood hypothesis h_{ML}, given these five x training examples.

How this hypothesis minimizes the sum of squared errors?

- Total probability over all possible values of the random variable to sum to one.
- In the case of continuous variables, we cannot achieve this by assigning a finite probability to each of the infinite set of possible values for the random variable.
- Instead, use probability density for continuous variables such as *e*, and use the integral of this probability density over all possible values be one.
- In general, lower-case *p* refers the probability density function distinguish it from a finite probability P. The probability density p(x_0) is the limit as ϵ goes to zero, of $\frac{1}{\epsilon}$times the probability that *x* will take on a value in the interval (*xo, xo* + ϵ*)*. Probability Density Function $p(x_0) \equiv \lim_{\epsilon \to 0} \frac{1}{\epsilon} P(x_0 \leq x < x_0 + \epsilon)$

MAXIMUM LIKELIHOOD

- Random noise variable *e* is generated by a Normal probability distribution.
- Maximum likelihood hypothesis using probability is $h_{ML} \equiv \underset{h \in H}{argmax}$ P(D|h)

- Derive the maximum likelihood hypothesis using probability density function (lower case *p)*

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

- Assume fixed set of training samples X = (x_l . . . x_m) and corresponding target values D = (d_l . . . d_m).
- The value $d_i = f(x_i) + e_i$.
- Assume the training examples are mutually independent given h, P(D|h) is the product of the various $p(d_i|h)$.

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

- Noise e_i obeys a Normal distribution $(\text{i.e.}, \boldsymbol{P(x)} = \frac{\boldsymbol{1}}{\sqrt{\boldsymbol{2\pi\sigma}}}\boldsymbol{e}^{-\frac{\boldsymbol{(x-\mu)^2}}{\boldsymbol{2\sigma^2}}}$ with zero mean and unknown variance σ^2.
- Also, each d_i must obey a Normal distribution with variance σ^2 centered around the true target value $f(x_i)$ rather than zero.
- Therefore $p(d_i|h)$ can be written as a Normal distribution with variance σ^2 and mean $\mu = f(x_i)$.

MAXIMUM LIKELIHOOD

- $h_{ML} = \underset{h \in H}{argmax} \prod_{i=1}^{m} p(di|h).$
- Substituting the appropriate μ and σ^2 and writing the expression for the $p(d_i|h)$ is the target function *f*.
- $h_{ML} = \underset{h \in H}{argmax} \prod_{i=1}^{m} \frac{\mathbf{1}}{\sqrt{\mathbf{2}\boldsymbol{\pi}\boldsymbol{\sigma}}} \boldsymbol{e}^{-\frac{(d_i - \boldsymbol{\mu})^{\mathbf{2}}}{\mathbf{2}\boldsymbol{\sigma}^{\mathbf{2}}}}$
- So, substitute $\boldsymbol{\mu} = f\ (x_i) = h\ (x_i)$
- $h_{ML} = \underset{h \in H}{argmax} \prod_{i=1}^{m} \frac{\mathbf{1}}{\sqrt{\mathbf{2}\boldsymbol{\pi}\boldsymbol{\sigma}}} \boldsymbol{e}^{-\frac{(d_i - h\ (xi)^{\mathbf{2}}}{\mathbf{2}\boldsymbol{\sigma}^{\mathbf{2}}}}$
- Transform this into less complicated using logarithm *(i.e., ln p)*
- $h_{ML} = \underset{h \in H}{argmax} \prod_{i=1}^{m} \boldsymbol{ln}\ \frac{\mathbf{1}}{\sqrt{\mathbf{2}\boldsymbol{\pi}\boldsymbol{\sigma}}} - \frac{1}{2\sigma^2}\left(d_i - h\ (xi)\right)^{\mathbf{2}}$
- $\boldsymbol{ln}\ \frac{\mathbf{1}}{\sqrt{\mathbf{2}\boldsymbol{\pi}\boldsymbol{\sigma}}}$ is a constant i.e., independent of *h*. So it can be discarded.
- $h_{ML} = \underset{h \in H}{argmax} \prod_{i=1}^{m} - \frac{1}{2\sigma^2}\left(d_i - h\ (xi)\right)^{\mathbf{2}}$
- Maximizing this negative quantity is equivalent to minimizing the corresponding positive quantity.
- $h_{ML} = \underset{h \in H}{argmin} \sum_{i=1}^{m} \frac{1}{2\sigma^2}\left(d_i - h\ (xi)\right)^{\mathbf{2}}$; Also ignore the constant that is independent of *h*.
- $h_{ML} = \underset{h \in H}{argmin} \sum_{i=1}^{m}\left(d_i - h\ (xi)\right)^{\mathbf{2}}$ It t minimizes the loss.

Maximum Likelihood in Predicting probability

- Assume, Both x_i and d_i as random variables, and assuming that each training example is drawn independently.
- So, P(D|h) $= \prod_{i=1}^{m} P(x_i\,,\, d_i|h)$
- When x is independent of h rewrite the expression using product rule as
- P(D|h) $= \prod_{i=1}^{m} P(x_i\,,\, di|h) = \prod_{i=1}^{m} P(d_i|h,\, x_i) * P(x_i)$
- $p(di|h,\, x_i) = \begin{cases} h\,(xi) & if\, d_i = 1 \\ (1 - h\,(xi)) & if\, d_i = 0 \end{cases}$
- Substitute this above equation $P(di|h,\, x_i) = h\,(xi)^{d_i} * (1 - h\,(xi))^{1-d_i}$
- So, rewrite ML hypothesis $h_{ML} = \underset{h \in H}{argmax} \prod_{i=1}^{m} h\,(xi)^{d_i} * (1 - h\,(xi))^{1-d_i} * P(x_i)$
- $P(x_i)$ is a constant independent of h, so it can be ignored
- $h_{ML} = \underset{h \in H}{argmax} \prod_{i=1}^{m} h\,(xi)^{d_i} * (1 - h\,(xi))^{1-d_i}$
- Apply log of the likelihood to above equation
- $h_{ML} = \underset{h \in H}{argmin} \sum_{i=1}^{m} d_i \ln h(xi) + (\mathbf{1} - \boldsymbol{di}) * \boldsymbol{ln}\, (\mathbf{1} - h\,(xi))$
- It look like a binary cross entropy

References

1. Tom M. Mitchell, Machine Learning, McGraw Hill , 2017.

2. EthemAlpaydin, Introduction to Machine Learning (Adaptive Computation and Machine Learning), The MIT Press, 2017.

3. Wikipedia