Data Science
Unit-VI
Bayes' Classifier

Bayes' Classifier

Naïve Bayes Classifier

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

Thomas Bayes
1702 - 1761

According to *Bayes theorem, we need to calculate the* posterior probability

Likelihood

Class Prior Probability

$$P(c \mid x) = \frac{P(x \mid c)P(c)}{P(x)}$$

Posterior Probability

Predictor Prior Probability

Or simply we calculate in expanded form:

$$P(c \mid \mathrm{X}) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

We need to calculate it for each class and then compare the results to find which gives the higher score.

BAYE'S THOREM

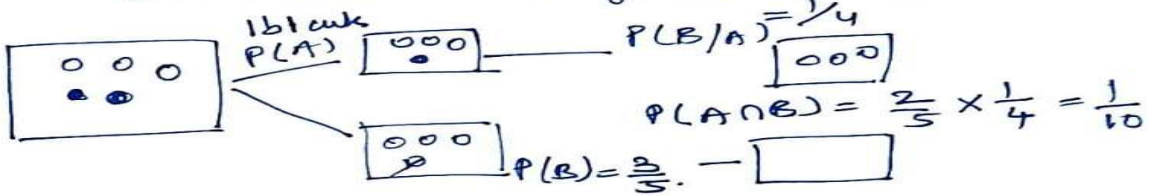
1 Conditional Probability

$$P(A/B) = \frac{P(A \cap B)}{P(B)}$$

where A = Events 1
B = Events 2. (given).

2. Independent Events
Ex: Tossing two coins.
Probability = 0.5, $q = 0.5$

3. Dependent Events
Ex: Marbles in bag (2 Red + 2 black)



$$P(A/B) = \frac{P(A \cap B)}{P(B)} \qquad \therefore A \cap B = B \cap A$$

$$P(B/A) = \frac{P(B \cap A)}{P(A)}$$

$$\therefore P(A/B) \times P(B) = P(B/A) * P(A)$$

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

where $P(A/B)$ = Posterior probability
$P(B/A)$ = Likelyhood.
$P(A)$ = Prior Probability.
$P(B)$ = marginal probability

Naive Bayes' classifier

Apply Baye's theorem on dataset.

Let Dataset

record $X = \{x_1, x_2, x_3, \ldots x_n\}\{y\}$

$$P(y/x_1, x_2 \ldots x_n) = \frac{P(x_1/y)\, P(x_2/y) \ldots P(x_n/y) * P(y)}{P(x_1)\, P(x_2) \ldots P(x_n)}$$

$$P(y/x_1, x_2 \ldots x_n) = \frac{P(y) \prod_{i=1}^{n} P(x_i/y)}{\underbrace{P(x_1)\, P(x_2) \ldots P(x_n)}_{\text{constant.}}}$$

$$P(y/x_1, x_2 \ldots x_n) \propto P(y) \prod_{i=1}^{n} P(x_i/y)$$

$$y = \underbrace{\arg\max_y}_{\text{takes max value.}} P(y) * \prod_{i=1}^{n} P(x_i/y).$$

Example

PlayTennis: training examples

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

PlayTennis: training examples

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

$$P(PlayTennis = yes) = 9/14 = .64$$

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

Outlook	Y	N
sunny	2/9	3/5
overcast	4/9	0
rain	3/9	2/5
Tempreature		
hot	2/9	2/5
mild	4/9	2/5
cool	3/9	1/5

Humidity	Y	N
high	3/9	4/5
normal	6/9	1/5
Windy		
Strong	3/9	3/5
Weak	6/9	2/5

Example

$$\langle Outlook = sunny, Temperature = cool, Humidity = high, Wind = strong \rangle$$

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

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

Test Phase: Predict class for instar

- Given a new instance,

  $\mathbf{x}'$=(Outlook=*Sunny*, Temperature=*Cool*, Humidity=*High*, Wind=*Strong*)

- Look up tables

  P(Outlook=*Sunny* | Play=*Yes*) = 2/9
  P(Temperature=*Cool* | Play=*Yes*) = 3/9
  P(Huminity=*High* | Play=*Yes*) = 3/9
  P(Wind=*Strong* | Play=*Yes*) = 3/9
  P(Play=*Yes*) = 9/14

  P(Outlook=*Sunny* | Play=*No*) = 3/5
  P(Temperature=*Cool* | Play==*No*) = 1/5
  P(Huminity=*High* | Play=*No*) = 4/5
  P(Wind=*Strong* | Play=*No*) = 3/5
  P(Play=*No*) = 5/14

- MAP rule

  P(*Yes* | $\mathbf{x}'$): [P(*Sunny* | Yes)P(*Cool* | *Yes*)P(*High* | Yes)P(*Strong* | *Yes*)]P(Play=*Yes*) = 0.0053
  P(*No* | $\mathbf{x}'$): [P(*Sunny* | No) P(*Cool* | No)P(*High* | *No*)P(*Strong* | *No*)]P(Play=*No*) = 0.0206

  Given the fact P(*Yes* | $\mathbf{x}'$) < P(*No* | $\mathbf{x}'$), we label $\mathbf{x}'$ to be "*No*".

Bayesian Classification: Why?

A statistical classifier: performs *probabilistic prediction, i.e.,* predicts class membership probabilities

Foundation: Based on Bayes' Theorem.

Performance: A simple Bayesian classifier, *naïve Bayesian classifier*, has comparable performance with decision tree and selected neural network classifiers

Incremental: Each training example can incrementally increase/decrease the probability that a hypothesis is correct — prior knowledge can be combined with observed data

Standard: Even when Bayesian methods are computationally intractable, they can provide a standard of optimal decision making against which other methods can be measured

Bayes' Theorem: Basics

Total probability Theorem: $P(B) = \sum_{i=1}^{M} P(B|A_i)P(A_i)$

Bayes' Theorem:

$$P(H|\mathbf{X}) = \frac{P(\mathbf{X}|H)P(H)}{P(\mathbf{X})} = P(\mathbf{X}|H) \times P(H) / P(\mathbf{X})$$

Let **X** be a data sample ("*evidence*"): class label is unknown

Let H be a *hypothesis* that X belongs to class C

Classification is to determine P(H|**X**), (i.e., *posteriori probability):* the probability that the hypothesis holds given the observed data sample **X**

P(H) (*prior probability*): the initial probability

E.g., **X** will buy computer, regardless of age, income, …

P(**X**): probability that sample data is observed

P(**X**|H) (likelihood): the probability of observing the sample **X**, given that the hypothesis holds

E.g., Given that **X** will buy computer, the prob. that X is 31..40, medium income

Prediction Based on Bayes' Theorem

Given training data **X**, *posteriori probability of a hypothesis* H, P(H|**X**), follows the Bayes' theorem

$$P(H \mid \mathbf{X}) = \frac{P(\mathbf{X} \mid H)P(H)}{P(\mathbf{X})} = P(\mathbf{X} \mid H) \times P(H) / P(\mathbf{X})$$

Informally, this can be viewed as

posteriori = likelihood x prior/evidence

Predicts **X** belongs to C_i iff the probability $P(C_i|\mathbf{X})$ is the highest among all the $P(C_k|X)$ for all the *k* classes

Practical difficulty: It requires initial knowledge of many probabilities, involving significant computational cost

Classification Is to Derive the Maximum Posteriori

Let D be a training set of tuples and their associated class labels, and each tuple is represented by an n-D attribute vector **X** = (x_1, x_2, …, x_n)
Suppose there are *m* classes C_1, C_2, …, C_m.
Classification is to derive the maximum posteriori, i.e., the maximal $P(C_i|\mathbf{X})$
This can be derived from Bayes' theorem

Since P(X) is constant for all classes, only

needs to be maximized $$P(C_i|\mathbf{X}) = \frac{P(\mathbf{X}|C_i)P(C_i)}{P(\mathbf{X})}$$

$$P(C_i|\mathbf{X}) = P(\mathbf{X}|C_i)P(C_i)$$

Naïve Bayes Classifier

A simplified assumption: attributes are conditionally independent (i.e., no dependence relation between attributes):

$$P(\mathbf{X}|C_i) = \prod_{k=1}^{n} P(x_k|C_i) = P(x_1|C_i) \times P(x_2|C_i) \times ... \times P(x_n|C_i)$$

This greatly reduces the computation cost: Only counts the class distribution

If A_k is categorical, $P(x_k|C_i)$ is the # of tuples in C_i having value x_k for A_k divided by $|C_{i,D}|$ (# of tuples of C_i in D)

If A_k is continous-valued, $P(x_k|C_i)$ is usually computed based on Gaussian distribution with a mean μ and standard deviation σ

and $P(x_k|C_i)$ is

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

$$P(\mathbf{X}|C_i) = g(x_k, \mu_{C_i}, \sigma_{C_i})$$

Naïve Bayes Classifier: Training Dataset

Given the training data in the table below (*Buy Computer* data), predict the class of the following new example using Naïve Bayes classification: age<=30, income=medium, student=yes, credit-rating=fair

Class:
C1:buys_computer = 'yes'
C2:buys_computer = 'no'

Data to be classified:
X = (age <=30,
Income = medium,
Student = yes
Credit_rating = Fair)

RID	age	income	student	credit_rating	Class: buys_computer
1	<=30	high	no	fair	no
2	<=30	high	no	excellent	no
3	31 . . . 40	high	no	fair	yes
4	>40	medium	no	fair	yes
5	>40	low	yes	fair	yes
6	>40	low	yes	excellent	no
7	31 . . . 40	low	yes	excellent	yes
8	<=30	medium	no	fair	no
9	<=30	low	yes	fair	yes
10	>40	medium	yes	fair	yes
11	<=30	medium	yes	excellent	yes
12	31 . . . 40	medium	no	excellent	yes
13	31 . . . 40	high	yes	fair	yes
14	>40	medium	no	excellent	no

Naïve Bayes Classifier: An Example

age	income	student	credit_rating	com
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

Solution:

E= age<=30, income=medium, student=yes, credit-rating=fair
E_1 is age<=30, E2 is income=medium, student=yes, E4 is credit-rating=fair
We need to compute P(yes|E) and P(no|E) and compare them.

$$P(yes \mid E) = \frac{P(E_1 \mid yes)\,P(E_2 \mid yes)\,P(E_3 \mid yes)\,P(E_4 \mid yes)\,P(yes)}{P(E)}$$

P(yes)=9/14=0.643 P(no)=5/14=0.357

P(E1|yes)=2/9=0.222 P(E1|no)=3/5=0.6
P(E2|yes)=4/9=0.444 P(E2|no)=2/5=0.4
P(E3|yes)=6/9=0.667 P(E3|no)=1/5=0.2
P(E4|yes)=6/9=0.667 P(E4|no)=2/5=0.4

$$P(yes \mid E) = \frac{0.222\ 0.444\ 0.667\ 0.668\ 0.443}{P(E)} = \frac{0.028}{P(E)} \qquad P(no \mid E) = \frac{0.6\ 0.4\ 0.2\ 0.4\ 0.357}{P(E)} = \frac{0.007}{P(E)}$$

Hence, the Naïve Bayes classifier predicts buys_computer=yes for the new example.

Avoiding the Zero-Probability Problem

Naïve Bayesian prediction requires each conditional prob. be **non-zero**. Otherwise, the predicted prob. will be zero

$$P(X \mid C_i) = \prod_{k=1}^{n} P(x_k \mid C_i)$$

Ex. Suppose a dataset with 1000 tuples, income=low (0), income= medium (990), and income = high (10)

Use **Laplacian correction** (or Laplacian estimator)

Adding 1 to each case

Prob(income = low) = 1/1003
Prob(income = medium) = 991/1003
Prob(income = high) = 11/1003

The “corrected” prob. estimates are close to their “uncorrected” counterparts

Advantages & Disadvantages

➢ **Advantages of Naïve Bayes Classifier:**

1. Naïve Bayes is one of the fast and easy ML algorithms to predict a class of datasets.
2. It can be used for Binary as well as Multi-class Classifications.
3. It performs well in Multi-class predictions as compared to the other Algorithms.
4. It is the most popular choice for text classification problems.

➢ **Disadvantages of Naïve Bayes Classifier:**

1. Naive Bayes assumes that all features are independent or unrelated, so it cannot learn the relationship between features.

Baye's Classifier-Example

Confident	Studied	Sick	Result
Yes	No	No	Fail
Yes	No	Yes	Pass
No	Yes	Yes	Fail
No	Yes	No	Pass
Yes	Yes	Yes	Pass

Data set for classification

Find out whether the object with attribute **Confident = Yes, Sick = No** will Fail or Pass using Bayesian classification.

Solution:

The data tuples are described by the attributes *Confident*, *Studied* and *Sick*.

The class label attribute, *Result*, has two distinct values (namely, {*Pass*, *Fail*}).

Let, C1 correspond to the class *Result* = *Pass* and

C2 correspond to *Result* = *Fail*.

The tuple we wish to classify is

X = (Confident = Yes, Sick = No)

Formula:

To predict the class label of X, P(X| Ci)P(Ci) is evaluated for each class Ci .
The classifier predicts that the class label of tuple X is the class Ci if and only if

$$P(\boldsymbol{X}|C_i)P(C_i) > P(\boldsymbol{X}|C_j)P(C_j) \quad \text{for } 1 \leq j \leq m, j \neq i.$$

In other words, the predicted class label is the class Ci for which P(X| Ci)P(Ci) is the maximum.

$$posterior = \frac{prior \times likelihood}{evidence}$$

Step 1: *(Compute prior probability)*

We need to maximize P (X| Ci) P(Ci), for i = 1, 2. P(Ci), the prior probability of each class, can be computed based on the training tuples:

P(Result = Pass) = 3/5 = 0.6
P(Result = Fail) = 2/5 = 0.4

Step 2: *(Compute likelihood probability)*

To compute P(X| Ci), for i = 1, 2, we compute the following conditional probabilities:

P(Confident = Yes | Result = Pass) = 2/3 = 0.6667
P(Confident = Yes | Result = Fail) = 1/2 = 0.5
P(Sick = No | Result = Pass) = 1/3 = 0.3333
P(Sick = No | Result = Fail) = 1/2 = 0.5

Step 3: *(Compute posterior probability)*

P(**X**| Result = Pass)

= P(Confident = Yes | Result = Pass) × P(Sick = No | Result = Pass)

= 0.6667 * 0.3333

= 0.2222

P(X| Result = Fail)

= P(Confident = Yes | Result = Fail) × P(Sick = No | Result = Fail)

= 0.5 * 0.5

= 0.25

Step 4: *(predict the class for X)*

To find the class, Ci , that maximizes P(**X**| Ci)P(Ci), we compute

P(X| Result = Pass)P(Result = Pass) = 0.2222 × 0.6 = 0.1333

P(X| Result = Fail)P(Result = Fail) = 0.25 × 0.4= 0.1

Therefore, the naive Bayesian classifier predicts *Result* = *Pass* for tuple **X**.