Naive Bayes

* Probabilistic based classification algo.

conditional Probability (wiki- conditional_probability Example section).

$$P(A|B) = P_r(A=a|B=b) = \frac{P(A\cap B)}{P(B)}, \quad P(B) \neq 0$$

* read eq's in english to get idea. $P(\text{dice}=3 \mid \text{dice is odd}) = \frac{1/6}{1/2} = \underline{1/3}$

Independent Events & Mutually Exclusive Events

A & B are said to be independent if

$$\begin{cases} P(A|B) = P(A) \\ P(B|A) = P(B) \end{cases}$$

A:- getting value of 6 in die 1 throw.
B:- getting a value of 3 in die 2 throw.

A & B are said to be mutually exclusive iff

$P(A|B) = P(B|A) = 0$.

when $\underline{P(A\cap B) = P(B\cap A) = 0}$

A: getting value of 6 in dice
B: " " " 3 " ".

Baye's Theorem

likelihood, prior

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

evidence.

posterior

wiki — Baye's Theorem
Problem - A more complicated example.

Proof:- $P(A|B) = \frac{P(A\cap B)}{P(B)} = \frac{P(A,B)}{P(B)}$.

$A\cap B = B\cap A$ — set theory.

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

$P(B|A) = \frac{P(B,A)}{P(A)} \Rightarrow P(B,A) = \underline{P(B|A) * P(A)}$.

So, $\boxed{P(A|B) = \frac{P(B|A)\, P(A)}{P(B)}}$ — Hence, Proved.

Ex. 3 machines. M_1: 20% o/p.
M_2: 30% o/p
M_3: 50% o/p.

Item chosen at random, found to be defective. Probability that it belongs to 3rd m/c ??

fraction of defective items for M_1: 5%
M_2: 3%
M_3: 1%.

$\rightarrow$ $P(M_3|D) = ??$ $\rightarrow$ $\dfrac{P(D/M_3)\,P(M_3)}{P(D)}$.

$P(M_1) = 0.2$, $P(M_2) = 0.3$, $P(M_3) = 0.5$
$P(D/M_1) = 0.05$, $P(D|M_2) = 0.03$
$P(D|M_3) = 0.01$.

$$P(D) = \sum_{i=1}^{3} P(D|M_i) \times P(M_i) = 0.2 \times 0.05 + 0.3 \times 0.03 + 0.5 \times 0.01 = 0.024.$$

$$P(M_3|D) = \frac{(0.01)\times(0.5)}{0.024} = \frac{5}{24} \text{ Ans}$$

Naive Bayes Algorithm (wikis – Naive Bayes classifier).

Consider a data pt with n-features & K-classes. $(C_1, C_2, \ldots C_k)$.

* If k=2, then binary classification

$x = \langle x_1, x_2, \ldots x_n \rangle$

Task:- Given x (all n features), we have to predict its class label.
i.e. $\boxed{P(C_k|x)\ \forall k \in [1, K]}$ – whichever is highest, we'll select that as the class label.

$$P(C_k|x) = \frac{P(x|C_k)\,P(C_k)}{P(x)} \longrightarrow P(x \cap C_k) = P(x, C_k)$$

Denominator constant for all class. So, only numerator matters.

So, $P(C_k|x) \propto P(C_k, x)$. $\longrightarrow$ whichever class has highest val. of $P(C_k, x)$ will be ans.

$$P(C_k, x) = P(C_k, x_1, x_2, x_3, \ldots x_n).$$

$$P(C_k, x_1, x_2 \ldots x_n) = P(\overbrace{x_1}^{A}, \overbrace{x_2, x_3 \ldots x_n, C_k}^{B})$$

$$= P(x_1|x_2, x_3 \ldots x_n, C_k) \cdot P(x_2, x_3, \ldots x_n, C_k).$$

$$= P(x_1|x_2, x_3 \ldots x_n, C_k) \cdot P(x_2|x_3, x_4 \ldots x_n, C_k) \cdot P(x_3, x_4 \ldots x_n, C_k).$$

$$= P(x_1|x_2 \ldots x_n, C_k) \cdot P(x_2|x_3 \ldots x_n, C_k) \ldots P(x_{n-1}|x_n, C_k) \cdot P(x_n|C_k) \cdot P(C_k)$$

Assuming that each feature x_i is conditionally independent of every other feature

$\{ P(A|B, C) = P(A|C)$ {if A & B are independent} $\}$.

$$\boxed{P(C_k, x_1, x_2, \ldots x_n) = P(x_1|C_k)\,P(x_2|C_k) \ldots P(x_n|C_k)\,P(C_k)}$$

$$= P(C_k) \prod_{i=1}^{n} P(x_i|C_k).$$

So, $P(C_k|x_1, x_2 \ldots x_n) \propto P(C_k) \prod_{i=1}^{n} P(x_i|C_k)$.

$$= \frac{1}{Z} P(C_k) \prod_{i=1}^{n} P(x_i|C_k).$$

$Z = P(x)$.

shatterline.com for example.

Naive Bayes on text data
└ very widely used.

SPAM filter:- email → SPAM / not-spam

text classification
↓
NB widely used.

review → (+)ve / (-)ve

Task:- $P(y=1|\text{text}_q)$, $P(y=0|\text{text}_q)$

text → stopwords, stemming, n-gram {pre-processing} → binary bag of words. vector representation

text → $\{w_1, w_2, w_3 \ldots w_d\}$. → BoW features.

$P(y=1|text) = P(y=1) * P(w_1|y=1) * P(w_2|y=1) * \cdots P(w_d|y=1)$.

$$P(y=1|text) = P(y=1) \prod_{i=1}^{d} P(w_i|y=1)$$

Similarly $P(y=0|text) = P(y=0) \prod_{i=1}^{d} P(w_i|y=0)$.

$$P(y=1) = \frac{\text{\# train pts with } y=1}{\text{total train pts.}} \qquad P(y=0) = \frac{\text{\# train pts with } y=0}{\text{total train pts.}}$$

$$P(w_i|y=1) = \frac{\text{\# data pts which contain word } w_i \text{ \&\& } y=1}{\text{\# data pts with } y=1}$$

Similarly for $y=0$.

NB is used as a baseline model (as its very simple) for text classification.

Laplace Smoothing

Q. what if the query text has a word that wasn't present in training set?

→ say $text = \{w_1, w_2, w_3, w'\}$ new word.

$$P(y=1|text) = P(y=1)\, P(w_1|y=1)\, P(w_2|y=1)\, P(w_3|y=1)\, P(w'|y=1).$$

If we completely ignore $P(w'|y=1)$ then it'll be same as assuming $P(w'|y=1)=1$ which is absurd.

If we go by defn, $P(w'|y=1) = \frac{\text{\# data pts where } w' \text{ occurs \&\& } y=1}{\text{\# data pts with } y=1}$

$$= \frac{0}{n_1} = 0.$$

This will make the entire probability 0.

So, we do laplace smoothing (additive smoothing).

$$P(w'|y=1) = \frac{0+\alpha}{n_1+\alpha K}$$

K = no of distinct value the feature w' can take.

Here since we are using binary BoW representation $K=2$.

α – can be any numeric value typically taken as 1.

let $n_1 = 100$.

$$P(w'|y=1) = \frac{0+\alpha}{100+2\alpha}.$$

Case 1: $\alpha = 1$. $P(w'|y=1) = \frac{1}{102}$

Case 2: $\alpha = 10000$ $P(w'|y=1) = \frac{10000}{20100} \approx \frac{1}{2}$.

Laplace smoothing when applied is applied to all the words, not to only the words not present in training.

$$P(w_i|y=1) = \frac{\text{\# data pts with } w_i \text{ \& } y=1 + \alpha}{\text{\# data pts with } y=1 + \alpha K.}$$

* As $\alpha\uparrow$, it moves the likelihood probabilities to uniform distbn.

So, if we have a likelihood prob. with small num / denom, then we have less confidence in the ratio. So, if we have just large enough α, it'll smoothen its probab. value towards uniform distbn.

Log probabilities

Since all the probabilities are b/w 0 & 1, & also text ~~features~~ data are having many features, continued product may cause underflow of data.

So, we take log of all the class priors & likelihood probabilities & add them together.

The class having maxm value is selected.

Since log is monotonically inc↑, its suitable for this purpose.

Bias variance trade-off

high bias — underfitting

high variance — overfitting

Only parameter :- α

* high α — smoothens out likelihood probability towards uniform distⁿ.

case 1 : $\alpha = 0$ with small change on training data, ~~posterior~~ likelihood probab. changes a lot. => high variance. → overfitting

case 2 : α = large for all w_i, $P(w_i|y=1) \approx 1/2$ if α significantly high. So, underfitting.

So, how to find right α?

→ α in naive bayes & k in KNN are hyper-parameters. These are calculated using cross validation.

Feature importance and Interpretability

NB words with high value of $P(w_i|y=1) \longrightarrow$ important words/features in determining that point belongs to (+)ve class.

So, feature importance can be directly obtained fm model.

{ +ve feature :- find words (w_i) with highest value of $P(w_i|y=1)$.

{ -ve feature :- find words (w_i) with highest value of $P(w_i|y=0)$.

Interpretability

$x_q \longrightarrow y_q = 1$

$\{w_1, w_2, \dots w_d\}$

I am concluding $y_q = 1$ because x_q contains words w_3, w_6, w_{10} which have a high value of $P(w_3|y=1)$, $P(w_6|y=1)$ $P(w_{10}|y=1)$.

and also $y_q \neq 0$ because w_8, w_{15}, w_{20}. $P(w_8|y=0)$, $P(w_{15}|y=0)$, $P(w_{20}|y=0)$ are very small.

Imbalanced data

$n \longrightarrow n_1$ — (+)ve class
$\quad \longrightarrow n_2$ — (-)ve class.

$n_1 >> n_2$ Or $n_2 >> n_1$
↓
Imbalance condⁿ.

Say $\frac{n_1}{n} = 0.9$ & $\frac{n_2}{n} = 0.1$.

$$P(y=1|w_1 w_2 \dots w_d) = \underbrace{P(y=1)}_{0.9} \prod_{i=1}^{d} P(w_i|y=1)$$

$$P(y=0|w_1 w_2 \dots w_d) = \underbrace{P(y=0)}_{0.1} \prod_{i=1}^{d} P(w_i|y=0)$$

If the likelihoods are almost the same, then the majority class has an advantage due to high class prior.

Solⁿ :- ① upsampling/downsampling $P(y=1) = P(y=0) = 1/2$

② simply drop the probabilities.

③ modified NB — not often used.

Another problem with imbalanced data.

(+)ve $n_1 = 900$, $\longrightarrow P(w_i|y=1) = \frac{\boxed{}}{900}$ — (0-900)

(-)ve $n_2 = 100$, $\longrightarrow P(w_i|y=0) = \frac{\boxed{}}{100} \longrightarrow$ (0-100).

So, in minority class have small num & denom. when we apply laplace smoothing, the impact of α is more on minority than on majority.

Ex

	minority		majority	
P	$\frac{2}{100}$	2%	$\frac{18}{900}$	2%
	} huge change		} less change.	
$\alpha = 10$	$\frac{2+10}{100+20} \approx 10\%$		$\frac{18+10}{900+10} \approx 3.04$.	

for the same α, ~~we~~ diff behaviour. ~~which~~

① upsampling / downsampling

② hacks :- α_+, α_- diff values

Outliers

Outlier at test time is taken care by laplace smoothing.

During training time:- If occurs very few times, then ignore.
(say less than 10 times)

Missing value

① text data:- no case of missing values.

② categorical data:- say i^{th} data pt has its j^{th} feature missing
then consider the missing as a new category NaN

③ numerical features → use mean / median to figure out.

Numerical features

* Big assumption:- feature f_j is Gaussian distbⁿ.

Say we want to calc. $P(y=1 | x_{i1}, x_{i2}, x \dots x_{id})$ (i^{th} data pt)

$= P(y=1) \prod_{j=1}^{d} P(x_{ij} | y=1)$ (j^{th} feature)

(trivial)

for $P(x_{ij} | y=1) \longrightarrow$ consider all pts having $y=1$.
calc mean & var. of j^{th} feature
say μ & σ.

Then we assume feature f_j to be a Gaussian distbⁿ of $N(\mu, \sigma)$.

Now, we find $P(x_{ij} | y=1)$ by using the pdf of the Gaussian curve & value of x_{ij}.

This is also called Gaussian Naive Bayes.

---

* NB can't use similarity / distance matrix. It needs features.
* NB can easily handle large dimensions.

Best and worst case of NB

① conditional independence:- of features. – assumption.
If true:- NB performs very well.
If false:- NB deteriorates.

② text-classificatⁿ → NB used as the baseline model for spam filters, review polarity.

③ categorical features → extensively used.

④ real-value features → seldom NB used.

⑤ Interpretability & feature importance is great.
We can reason as to why the model gave a certain o/p.
runtime complexity = low.
Train-time complexity = low.
runtime space — low (prior & likelihoods only reqd).

* can easily overfit unless laplace smoothing done with proper α from cross validation.