Naive Bayes

Type	Lecture Notes
Reviewed	☐
Available Summary?	Done
Week	5

YOU SHOULD KNOW:

- training and using classifier based on Bayes rule
- conditional independence:
    - what it is
    - why is it important
- Naive Bayes:
    - what it is
    - how to estimate the parameters
    - how to make predictions

Bayes Optimal Classifier - *Part 1*

- Goal is to learn $f : X -> Y$
    - X → features
    - Y → target class

$Y^* = argmax_{y_k} P(Y = y_k|X)$

how many parameters must we estimate?

boolean and 2 features

Suppose $X = (X_1, X_2)$ where X_i and Y are boolean RV's, to estimate $P(Y|X_1, X_2)$:

X_1	X_2	$P(Y = 1\|X_1, X_2)$	$P(Y = 0\|X_1, X_2)$
0	0	0.1	0.9
1	0	0.24	0.76
0	1	0.54	0.46
1	1	0.23	0.77

4 parameters!

$P(Y = 0|X) = 1 - P(Y = 1|X)$

boolean and n features

Suppose $X = (X_1, X_2, ..., X_n)$ where X_i and Y are boolean RV's, to estimate $P(Y|X_1, X_2, ..., X_n)$:

we need 2^n rows to create the table

can we reduce parameters using Bayes Rule?

Suppose $X = [X_1, ..., X_n]$ and all X_is and Y are boolean variables.

To estimate $P(X_1, \ldots, X_n|Y)$:

2^n rows to create the table and -1 cause all the probabilities are sum up to 1 (so we already know the final probability we do not need to calculate)

$2 * (2^n - 1)$ parameters we need

To define $P(Y)$:

we need 1 parameter

Finally:

we need $2 * (2^n - 1) + 1$ parameters

Naive Bayes

- **Naive bayes assumes:**

  $P(X_1, ..., X_n|Y) = \prod_i P(X_i|Y)$

  that X_i and X_j are conditionally independent given Y, for all $i \neq j$

- **Naive bayes uses assumption that the X_i are conditionally independent given Y**

  $P(X_1, X_2|Y) = P(X_1|X_2, Y).P(X_2|Y)$

  more generally;

  $\underbrace{P(X_1, ..., X_d|Y)}_{\text{likelihood}} = \prod_{i=1}^{d} P(X_i|Y)$

> 💡 ***Recall conditionally independence:***
>
> X is conditionally independent of Y given Z, if the probability distribution governing X is independent of the value Y, given the value of Z
>
> $P(X = i|Y = j, Z = k) = P(X = i|Z = k)$
>
> $P(X, Y|Z) = P(X|Z).P(Y|Z)$

▼ How many parameters to describe $P(X_1, \ldots, X_n|Y)$ and $P(Y)$?

Without conditional indep assumption → $2 * (2^n - 1)$ and 1

With conditional indep assumption → $2n$ and 1

▼ Naive Bayes classification

Bayes Rule:

$$P(Y = y_k|X_1, ..., X_n) = \frac{P(Y=y_k).P(X_1,...,X_n|Y=y_k)}{\sum_j P(Y=y_j).P(X_1,...,X_n|Y=y_j)}$$

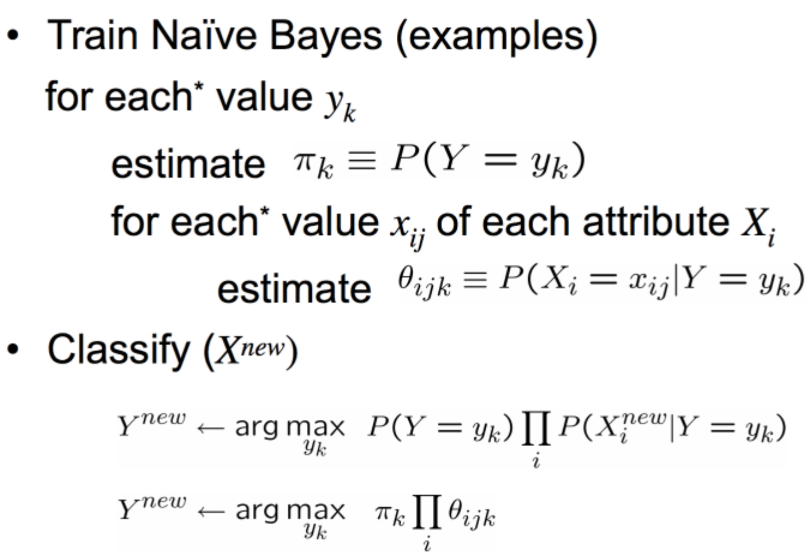
Assuming conditional independence among X_i's:

$$P(Y = y_k|X_1, ..., X_n) = \frac{P(Y=y_k).\prod P(X_i|Y=y_k)}{\sum_j P(Y=y_j).\prod P(X_i|Y=y_j)}$$ estimate in training

So, to pick most probable Y for $X^{new} = (X_1, ..., X_n)$:

$$Y^{new} \leftarrow argmax_{y_k} P(Y = y_k) \prod_i P(X_i^{new}|Y = y_k)$$ testing

▼ *EXAMPLE*



> 💡 *Posterior = Likelihood * Prior*

In naive bayes, we are multiplying lots of small numbers. It may cause underflow.

> 💡 Underflow occurs when you perform an operation that's smaller than the smallest magnitude non-zero number.

Solution:

Perform all computations by summing logs of probabilities rather than multiplying probabilities.

$p_1 * p_2 = e^{log(p_1) + log(p_2)}$

▼ Document Classification using Naive Bayes:

💡 To generate a document for giving class; randomly pick a number to predict for, if the *word_i* should be in the document or not.

▼ Bernoulli document model:

a document is represented by a binary feature vector, whose elements indicate absence or presence of corresponding word in the document

▼ Classification with Bernoulli Model

Training data:

Matrix: X

document: i

feature vector: X_i

presence of word t in the document i: X_{it}

Parameter Estimation (MLE):

Priors: $P(Y=y_k)=\frac{N_K}{N}$

Likelihoods: $P(w_t|Y=y_k)=\frac{n_k(w_t)}{N_k}$

(use a likelihood vector)

Classify new document D with feature vector X:

$$P(Y=y_k|X)=P(Y=y_k).P(X|Y=y_k)$$

$$P(X|Y=y_k)=\prod_{t=1}^{|V|}\left[X_t.P(w_t|Y=y_k)+(1-X_t)(1-P(w_t|y=y_k))\right]$$

▼ Multinomial document model:

a document is represented by an integer feature vector, whose elements indicate frequency of corresponding word in the document

▼ Classification with Multinomial Model

Training data:

Matrix: X

document: i

feature vector: X_i

the count of number of times w_t occurs in document i: X_{it}

1 if document i is of class y_k, otherwise 0: $Z_{i,k}$

Parameter Estimation (MLE):

Priors: $P(Y=y_k)=\frac{N_K}{N}$

Likelihoods: $P(w_t|Y=y_k)=\frac{\sum_{i=1}^{N}X_{i,t}.Z_{i,k}}{\sum_{t'=1}^{|V|}\sum_{i=1}^{N}X_{i,t'}.Z_{i,k}}$

The relative frequency of w_t in documents of class $Y=y_k$ with respect to the total number of words in documents of that class

Classify new document D with feature vector X:

$$P(Y=y_k|X)=P(Y=y_k).P(X|Y=y_k)$$

$$P(X|Y=y_k)=\prod_{t=1}^{|V|}P(w_t|Y=y_k)^{X_t}$$

Parameter estimation (MAP, with $\alpha=1$):

Likelihoods: $P(w_t|Y=y_k)=\frac{\alpha+\sum_{i=1}^{N}X_{i,t}.Z_{i,k}}{|V|\alpha+\sum_{t'=1}^{|V|}\sum_{i=1}^{N}X_{i,t'}.Z_{i,k}}$

A trick to avoid zero counts which is equivalent to MAP estimation with Dirichlet prior with $\alpha=1$.

▼ Bayes Optimal Classifier - *Part 2*

- Very fast, low storage requirements
- robust to irrevelant features

- optimal if the conditional independence assumptions hold
- a good dependable baseline for text classification

▼ Violating the NB classifier:

- usually features are not conditionally independent
- in practice it often works well
- it does not produce accuracte probability estimates when its independence assumptions are violated, it may still (and often) pick the correct maximum-probability class in many cases
- typically handles noise well since it does not even focus on completely fitting the training data