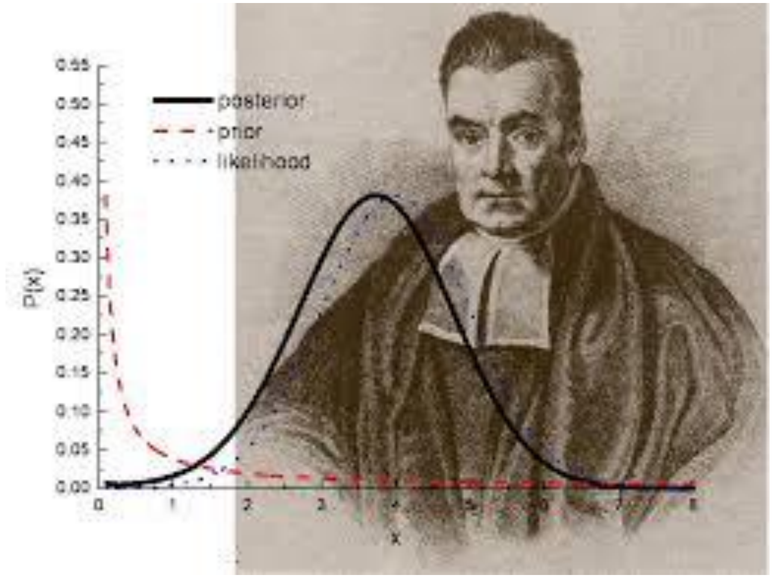
# Bayesian approach to ML

- A simple and effective framework for machine learning
- Based on sound, mathematical foundations
- Theoretically provides an optimal classifier
- Scales to Big Data (esp. text)




Thomas Bayes
1702 - 1761

## Bayesian learning - general properties

- Observed data and [inductive] hypothesis
- Combines probability of (1) observed data probability for each candidate hypothesis and (2) a probability distribution over observed data for each possible hypothesis, to obtain (3) final probability of chosen hypothesis (posterior)
- (1) and (2) are called **priors**, (3) is a **posterior**
- Bayesian methods can accommodate hypotheses that make probabilistic predictions (e.g., hypotheses such as "this pneumonia patient has a 93% chance of complete recovery").

# Bayes' law of conditional probability:

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

results in a simple "learning rule": choose
the most likely (Maximum APosteriori)hypothesis

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

# Example

- Two hypo:

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

- (1) the patient has cancer
- (2) the patient is healthy

*P(cancer) = .008* *P(~cancer) = .992*
*P(+|cancer) = .98* *P(-|cancer) = .02*
*P(+|~cancer) = .03* *P(-|~cancer) = .97*

New patient with +
How should we diagnose this patient?

*P(cancer|+)=*
*P(~cancer|+)=*

*Why **such** result?*

Determine exact posteriori probabilities, knowing that *P(cancer|+) + P(~cancer|+) = 1*

# Minimum Description Length

revisiting the def. of h*MAP*:

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

we can rewrite it as:

or

$$h_{MAP} = \arg\max_{h \in H} \log_2 P(D|h) + \log_2 P(h)$$

But the first log is the cost of coding the data *given* the theory, and the second - the cost of coding the theory

$$h_{MAP} = \arg\min_{h \in H} -\log_2 P(D|h) - \log_2 P(h)$$

Observe that:

for data, we only need to code the exceptions; the others are correctly predicted by the theory

MAP principles tells us to choose the theory which encodes the data in the shortest manner

the MDL states the trade-off between the complexity of the hypo. and the number of errors

# Bayes optimal classifier

- so far, we were looking at the “most probable hypothesis, given a priori probabilities”. But we really want the most probable classification
- this we can get by combining the predictions of all hypotheses, weighted by their posterior probabilities:
- this is the bayes optimal classifier BOC:

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

$$\underset{v_j \in V}{\arg\max} \sum_{h_i \in H} P(v_j|h_i)P(h_i|D)$$

Example of hypotheses
h1, h2, h3 with posterior probabilities
.4, .3. .3
A new instance D is classif. pos. by h1 and
neg. by h2, h3 : *P(+|h1) = 1, P(-|h1)=0,* etc.
What will be the classification according to the BOC?

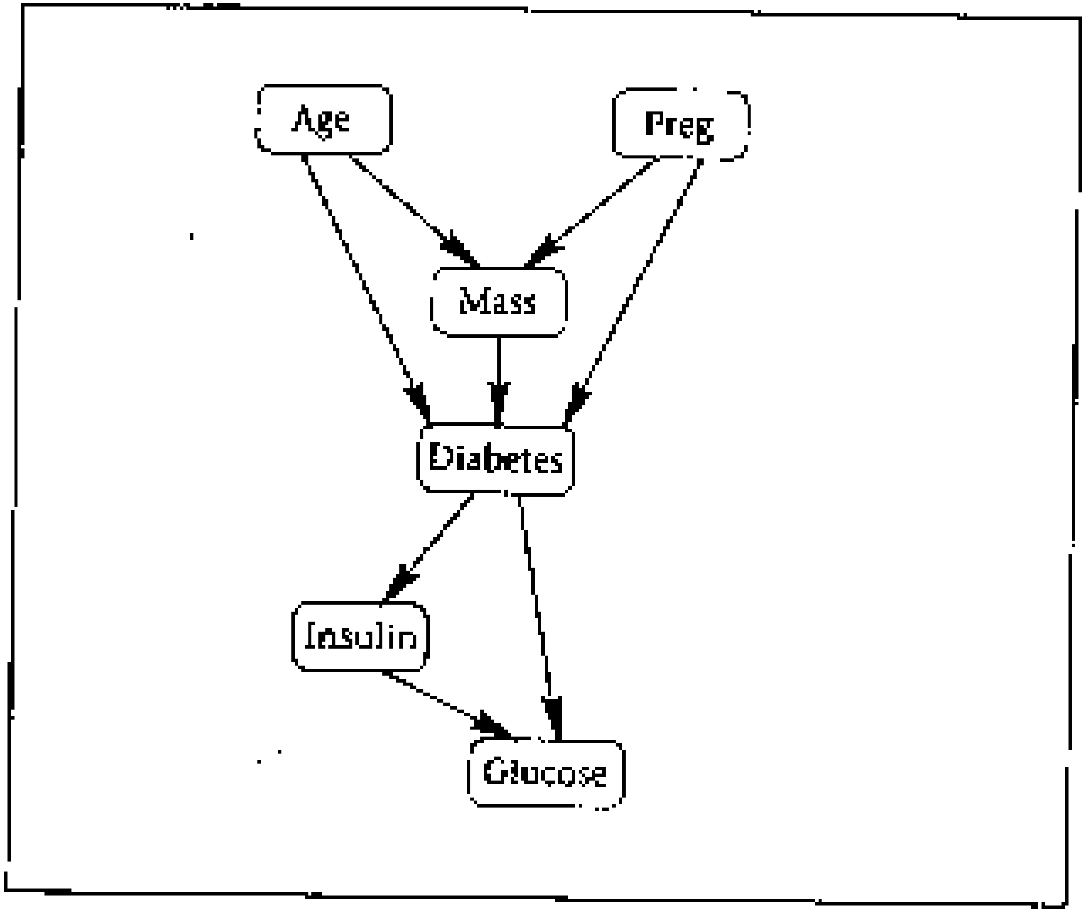



*Figure 19. A Probabilistic Network for Diabetes Diagnosis.*

- Captures probability dependencies
- ea node has probability distribution: the task is to determine the join probability on the data
- In an appl. a model is designed manually and forms of probability distr. Are given
- Training set is used to fit the model to the data
- Then probabil. Inference can be carried out, eg for prediction

First five variables are observed, and the model is Used to predict diabetes

$P(A, N, M, I, G, D)=P(A)*P(n)*P(M|A, n)*P(D|M, A, N)*P(I|D)*P(G|I,D)$

| Age | P(A) |
|---|---|
| 0–25 | |
| 26–50 | |
| 51–75 | |
| > 75 | |

| Preg | P(N) |
|---|---|
| 0 | |
| 1 | |
| >1 | |

| | | P(M\|A, N) | | |
|---|---|---|---|---|
| Age | Preg | 0–50 | 51–100 | >100 |
| 0–25 | 0 | | | |
| 0–25 | 1 | | | |
| 0–25 | > 1 | | | |
| 26–50 | 0 | | | |
| 26–50 | 1 | | | |
| 26–50 | >1 | | | |
| 51–75 | 0 | | | |
| 51–75 | 1 | | | |
| 51–75 | >1 | | | |
| >75 | 0 | | | |
| >75 | 1 | | | |
| >75 | > 1 | | | |

*Table 3. Probability Tables for the Age, Preg, and Mass Nodes from Figure 19.*
A learning algorithm must fill in the actual probability values based on the observed training data.

- how do we specify prob. distributions?
- discretize variables and represent probability distributions as a table
- Can be approximated from frequencies, eg table P(M|A, N) requires 24parameters
- For prediction, we want (D|A, n, M, I, G): we need a large table to do that

- in NB, the conditional probabilities are *estimated* from training data simply as normalized frequencies: how many times a given attribute value is associated with a given class wrt to all classes: $\frac{n_c}{n}$
- no search!
- There is no pre-computed classifier for a given training set (unlike for linear models)

- no other classifier using the same hypo. space e and prior K can outperform BOC
- the BOC has mostly a theoretical interest; practically, we will not have the required probabilities
- another approach, Naive Bayes Classifier (NBC)

$$v_{MAP} = \arg\max_{v_j \in V} P(v_j \mid a_1, \ldots a_n) = \arg\max_{v_j \in V} \frac{P(a_1, \ldots a_n \mid v_j) P(v_j)}{P(a_1, \ldots a_n)} =$$

$$\arg\max_{v_j \in V} P(a_1, \ldots a_n \mid v_j) P(v_j)$$

To estimate this, we need (#of all possible values of all attributes)*(#of possible classes) examples

under a simplifying assumption of independence of the attribute values given the class value:

$$v_{NB} = \arg\max_{v_j \in V} P(v_j) \prod_i P(a_i \mid v_j)$$

# Geometric decision boundary

- Assume a binary NB classifier *f* with instances $[x_1,\ldots,x_n,y]$, $y=0$ or $y=1$. Denote by $v_0$ $(v_1)$ the vector of probabilities of all instances belonging to class *0 (1),* respectively.

$$f(x) = \log \frac{P(y=1|x)}{P(y=0|x)} = \log P(y=1|x) - \log P(y=0|x) =$$

$$(\log v_1 - \log v_0)x + \log p(y=1) - \log p(y=0)$$

- This expression is linear in x. Therefore the decision boundary of the NB classifier is linear in the feature space *X,* and is defined by *f(x) = 0.*

# Discriminative *vs* generative models

- discriminative models model the distribution *P(Y|X) given X, return a probability of Y*
- generative models model the joint probability *P(Y,X).* They can be described by the likelihood function *P(X|Y),* because *P(Y,X) = P(X|Y)P(Y). [P(Y),* the prior, is easily estimated from data]
- Knowing a generative model we can sample new data points with their labels, or knowing the priors we can sample a class from *P(Y)* and new data points from *P(X|Y).* Not so for a linear classifier that models *P(Y|X)* but not *P(X).*
- Generative model = black box

***Likelihood function= conditional probability distribution***

# Background on

- Bernoulli
- Binomial
- Categorical (Bernoulli multivariate)
- Bernoulli multinomial
- Bernoulli multinomial is perfect for generative text modeling:

- Bernoulli: a coin throw: $P(X)=\theta$, $P(X=0)=1-\theta$, mean= $\theta$, variance= $\theta(1-\theta)$.
- Binomial: number of successes in *n* independent Bernoulli trials
- Categorical (Bernoulli multivariate – a dice)
- Bernoulli multinomial: *n* independent categorical trials
- Generative text modeling: Bernoulli multinomial, each trial is for a word with *k* possible outcomes (size of vocabulary)

- Back to estimating probabilities from word counts , we estimate $P(w_k|v_j)$ as m-estimate with equal priors, eg as (smoothing!)

$$\frac{n_k + 1}{n + |vocabulary|}$$

- incorrectness of NB for text classification (e.g. if ‘Matwin’ occurs, the previous word is more likely to be ‘Stan’ than any other word; violates independence of features)
- but amazingly, in practice it does not make a big difference

# Multinomial Naïve Bayes (MNB)

- designed for text categorization - requires BOW input data

- attempts to improve the performance of text classification by the incorporation the words frequency information

- models the distribution of words (features) in a document as a multinomial distribution

# Multinomial model and classifying documents

- We assume the *generative* model: a “source” generates an *n*-word long document, from a vocabulary of *k* words *(|V| = k*)
- Here we usually find the hypothesis (model) *most likely to have generated the data* (whereas in MAP we are looking for a model most likely *given* the observed data
- Word occurrences are *independent*
- A new document can then be modeled by a multinomial distribution

# MNB (Multinomial naïve Bayes classifier)

[check papers below for details]

- MNB model: $$P(d \mid c\ ) = \frac{(\sum_i f_i)!}{\prod_i f_i!} \prod_{i=1} P(w_i \mid c)^{f_i}$$
- where $f_i$ = # of occurrences of word $w_i$ in $d$
- Three independence assumptions:
  - occurrence of $w_i$ is independent of occurrences of all the other words
  - occurrence of $w_i$ is independent of itself
  - $|d|$ is independent of class of $d$
- MNB classifier:

$$P(c \mid d) = \frac{P(c) \prod_{i=1}^{n} P(w_i \mid c)^{f_i}}{P(d)} \tag{1}$$

# Frequency Estimate

- How do we get $P(w_i|c)$ ?
- We estimate it by Frequency Estimate (FE): this is the essence of the generative approach:

$$\hat{P}(w_i \mid c) = \frac{f_{ic}}{f_c}$$

- where $f_{ic}$ = # of occurrences of $w_i$ in docs of class $c$
- $f_c$ = total # of word occurrences in documents of class $c$
- FE is efficient: a single scan thru all the instances

# MNB is efficient

- Using  the conditional probability (from the multinomial framework of MNB), we easily get the aposteriori probability:

$$P(c \mid d) = \alpha P(c) \prod P(w_i \mid c)^{f_i}$$ and

$$C(d) = \arg\max_{c} P(c) \prod P(w_i \mid c)^{f_i}$$

- This means that we can ignore all the words from the corpus missing in a given document! (why?). In practice, this saves a lot of time!

# References


- McCallum, A., & Nigam, K. (1998). A comparison of event models for naive Bayes text classification. Proceedings of AAAI '98.
- J. D. M. Rennie, L. Shih, J. Teevan, and D. R. Karger (2003). Tackling the poor assumptions of Naive Bayes text classifiers. In T. Fawcett and N. Mishra (eds.), International Conference on Machine Learning Washington D.C.: Morgan Kaufmann

# Problems with MNB

- FE is not meant to optimize accuracy! It is meant to optimize likelihood
- If the independence assumptions are true, then FE also maximizes accuracy. But they are not true.
- See Su, Matwin "Large Scale Text Classification using Semisupervised Multinomial Naive Bayes", ICML 2011