CS181 Lecture 18: Naïve Bayes and EM

Overview

- The Naïve Bayes model
- Discrete density estimation revisited
- Learning the Naïve Bayes model
- Autoclass: Naïve Bayes for clustering
- EM for Gaussian mixture models

Naïve Bayes model

- Naïve Bayes model: discrete attributes with finite number of values
- Parametric density estimation
- Naïve Bayes classification algorithm
- Autoclass clustering algorithm

Naïve Bayes model

- Want to estimate $P(X_1,\ldots,X_n)$
- Assumption: all attributes independent of each other
  - same assumption as k-means, except this is a discrete model

$$P(X_1,\ldots,X_n) = \prod_{i=1}^{n} P(X_i)$$

- $P(X_i)$ can be any distribution you like
  - e.g. { 0.5 : red, 0.2 : blue, 0.3 : yellow }

Number of parameters

- Assume all attributes Boolean
- How many **independent** parameters in $P(X_1,\ldots,X_n)$ in general (with no independence assumptions)?
  - 2^n states of $X_1,\ldots,X_n$
  - 1 constraint that total must sum to 1
  - 2^n-1 independent parameters
- How many independent parameters in $\prod_{i=1}^{n} P(X_i)$?
  - 2 states of each X_i
  - 1 constraint for each X_i
  - n parameters total

Savings

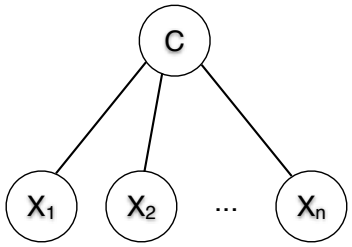
- Incredible savings in number of parameters
- Representing $P(X_1,\ldots,X_n)$ explicitly suffers from curse of dimensionality
- Representing $\prod_{i=1}^{n} P(X_i)$ does not
- This savings results from very strong independence assumptions
- Naïve Bayes model performs very well when assumptions hold
- Performs very badly when variables are dependent

Naïve Bayes classifier

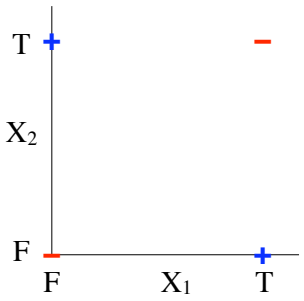
- Learn $P(X_1,...X_n \mid C) = \prod_{i=1}^{n} P(X_i \mid C)$ for each class
  - assumes that X_i and X_j are conditionally independent of each other given C
- Learn $P(C)$
- To classify: given x, choose c that maximizes

$$P(c \mid x) \propto P(c)P(x_i \mid c)$$

Naïve Bayes classifier picture



Naïve Bayes and XOR



Hypothesis space

- **Linear separators**
- Attributes are **independent**
  - they act independently to produce classification
  - they do not interact
  - therefore they cannot capture concepts like XOR
- Just like perceptrons

An important point

- Many real world domains are not linearly separable
- Even for those domains there may be a pretty good linearly separable hypothesis
- You may be better off learning a linearly separable hypothesis than learning a richer hypothesis
  - stronger inductive bias – easier to learn
- It has been found empirically that naïve Bayes can perform reasonably well in practice

Learning naïve Bayes

- In the following discussion we will assume that attributes and class are Boolean
- This is only to keep the notation simple
- Everything generalizes to the case where attributes and class have many possible values

Overview

- The Naïve Bayes model
- Discrete density estimation revisited
- Learning the Naïve Bayes model
- Autoclass: Naïve Bayes for clustering
- EM for Gaussian mixture models

A simple problem

- You are the manager of a soccer team
- You observe a sequence of games that your team plays
- Based on this, you wish to estimate the probability that your team will win a future game

Simplest formulation

- Variable X has states $\{ f, t \}$ (t = win)
- Parameter $\theta = P(X = t)$
- Observations $X^1 = t, X^2 = f, X^3 = f$
- These comprise the data $\mathbf{D}$
- Task: estimate θ
- Use θ to estimate $P(X^4 = t)$

Maximum likelihood (ML)

- **Likelihood**: $L(\theta) = P(\mathbf{D} \mid \theta) = P(X^1, X^2, X^3 \mid \theta)$
- **ML Principle**: Choose θ so as to maximize $L(\theta)$
- $L(\theta) = P(X^1 \mid \theta)P(X^2 \mid \theta)P(X^3 \mid \theta)$
- **Log likelihood**:
  $LL(\theta) = \log P(X^1 \mid \theta) + \log P(X^2 \mid \theta) + \log P(X^3 \mid \theta)$
- ML Principle equivalent: Choose θ so as to maximize $LL(\theta)$

In our example

- $P(X^i = t \mid \theta) = \theta$
- $L(\theta) = P(X^1 = t, X^2 = f, X^3 = f \mid \theta) = \theta(1- \theta)(1- \theta)$
- $LL(\theta) = \log \theta + 2 \log (1- \theta)$
- Set derivative to 0: $\frac{1}{\theta} - \frac{2}{1-\theta} = 0$
- Solve to find $\theta = 1/3$

Property of ML

- $\theta = 1/3$ is exactly the fraction of observed games in which the team won
- This is no coincidence: the ML estimate for the probability of an event is always the fraction of time in which the event happened
- In other words, ML's estimate is exactly the one most suggested by the data

More generally

- Observations $X^1, X^2, \ldots, X^N$
- Let
  - N_t be the number of instances with value t
  - N_f be the number of instances with value f
- Maximum likelihood estimate for θ is:

$$\hat{\theta} = \frac{N_t}{N_t + N_f} = \frac{N_t}{N}$$

Problems with this approach

- Overfits: pays too much attention to noise in the data
  - if your team was particularly unlucky in losing two games, this will be ignored
- Ignores prior experience:
  - if you believe your team is a good team, you shouldn't totally discount that after losing two games
- Events that don't occur in the data are deemed impossible

Incorporating a prior

- **Prior**: $P(\theta)$ before seeing any data
- **Posterior**: $P(\theta \mid \mathbf{D})$
- **Maximum a Posteriori principle (MAP)**: Choose θ so as to maximize $P(\theta \mid \mathbf{D})$
- Note $P(\theta \mid \mathbf{D})$ is proportional to $P(\theta)L(\theta)$

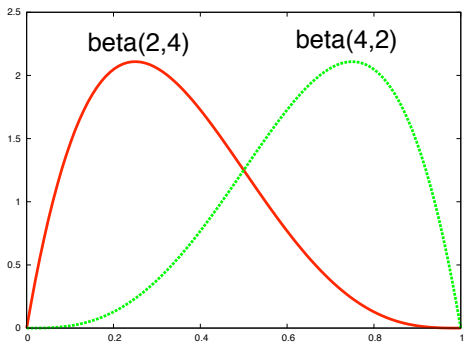
Beta distributions

- For learning the parameter of a Boolean random variable, an appropriate prior over θ is the **beta** distribution
- The beta distribution has two **hyperparameters**: α and β
- The hyperparameters control the shape of the prior
  - what we believe about θ
  - how peaked our beliefs are

Shape of the beta distribution

- α and β control how relatively likely true and false outcomes are
  - if α is large relative to β, θ will be more likely to be large

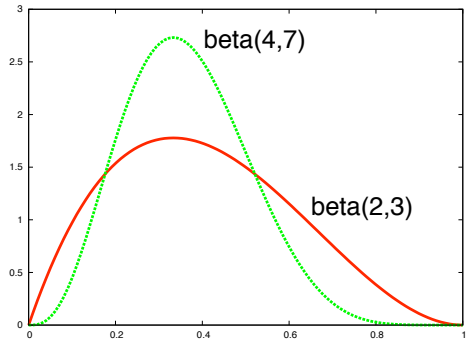
Relative sizes of α and β



Shape of the beta distribution

- α and β control how relatively likely true and false outcomes are
  - if α is large relative to β, θ will be more likely to be large
- The magnitude of α and β control how peaked the beta distribution is
  - if α and β are large, the beta will be sharply peaked

Magnitudes of α and β



Updating the prior

- To get the hyperparameters for the posterior, we take the hyperparameters in the prior, and add to them the actual observations that we get
- E.g.
  - prior is Beta(4,7)
  - we observe 1 positive and 4 negative
  - posterior is Beta(5,11)

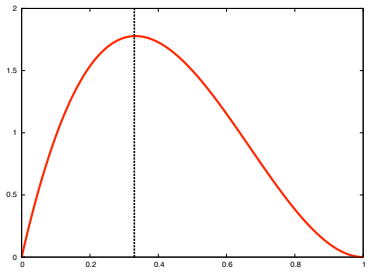
Understanding the hyperparameters

- Hyperparameter α represents the number of previous **positive** observations that we have had, plus 1
- Similarly, β represents the number of previous **negative** observations that we have had, plus 1
- Hyperparameters in the prior as represent imaginary observations in our prior experience
  - the more we trust our prior experience, the larger the hyperparameters in the prior

Mode of the beta distribution

The mode of Beta(α,β) is $\dfrac{\alpha-1}{\alpha+\beta-2}$

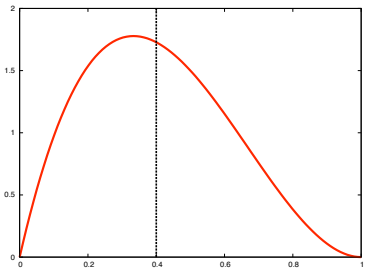
- e.g. mode of Beta(2,3) is 1/3



Mean of the beta distribution

The mean of Beta(α,β) is $\dfrac{\alpha}{\alpha+\beta}$

- e.g. mean of Beta(2,3) is 2/5



MAP estimate

- The MAP estimate is the **mode** of the posterior
  - this is the fraction of the total (real and imaginary) number of observations that are true
  - e.g. for m positive instances out of N total

$$\hat{\theta}_{\text{MAP}} = \frac{m + \alpha - 1}{N + \alpha + \beta - 2}$$

In our example

- Prior: Beta(5,3)
- Observations: $X^1 = t, X^2 = f, X^3 = f$
- Posterior: Beta(6,5)
- MAP estimate: $\hat{\theta}_{\text{MAP}} = \frac{m + \alpha - 1}{N + \alpha + \beta - 2} = \frac{1 + 5 - 1}{3 + 5 + 3 - 2} = \frac{5}{9}$

ML as MAP

- Maximum likelihood estimate: $\hat{\theta}_{\text{ML}} = \frac{m}{N}$
- MAP estimate: $\hat{\theta}_{\text{MAP}} = \frac{m + \alpha - 1}{N + \alpha + \beta - 2}$
- Maximum likelihood is equivalent to MAP with a uniform prior: Beta(1,1)
  - meaning there are no imaginary observations

Drawback of MAP

- Does not fully consider the range of possible values for θ
  - only chooses the maximum value
  - this value may not be representative

Bayesian approach

- Predict the outcome of the next game using the entire distribution over θ

$$\begin{aligned} P(X^4 \mid X^1, X^2, X^3) &= \int_0^1 P(X^4 \mid \theta) P(\theta \mid X^1, X^2, X^3) d\theta \\ &= \int_0^1 \theta P(\theta \mid X^1, X^2, X^3) d\theta \\ &= E\left[\theta \mid X^1, X^2, X^3\right] \\ &= \textbf{mean} \text{ of the posterior} \end{aligned}$$

for beta distribution $E[\theta] = \frac{m + \alpha}{N + \alpha + \beta}$

Important note

- In the Bayesian approach $E[\theta]$ is not the estimate of θ
  - in fact, no estimate of θ is made
  - instead, a posterior distribution is maintained over the value of θ
- $E[\theta]$ is the estimate of the probability that a new instance is true
  - this is obtained by integrating over the posterior distribution

In our example

- Prior: Beta(5,3)
- Observations: $X^1 = t, X^2 = f, X^3 = f$
- Posterior: Beta(6,5)
- $P(X^4 \mid X^1, X^2, X^3) = \frac{m + \alpha}{N + \alpha + \beta} = \frac{6}{11}$
- Compare to 5/9 for the MAP estimate
  - Bayesian estimate is closer to ½
  - more smoothed out

Overview

- The Naïve Bayes model
- Discrete density estimation revisited
- Learning the Naïve Bayes model
- Autoclass: Naïve Bayes for clustering
- EM for Gaussian mixture models

Model parameters

$P(C = T) = \theta_C$

$P(C = F) = 1 - \theta_C$

$P(X_i = T \mid C = T) = \theta_i^T$

$P(X_i = F \mid C = T) = 1 - \theta_i^T$

$P(X_i = T \mid C = F) = \theta_i^F$

$P(X_i = F \mid C = F) = 1 - \theta_i^F$

$\boldsymbol{\theta} = \langle \theta_C, \theta_1^T, \ldots, \theta_n^T, \theta_1^F, \ldots, \theta_n^F \rangle$

$2n + 1$ parameters

Maximum Likelihood

- Goal: Choose $\theta_C, \theta_1^T, \ldots, \theta_n^T, \theta_1^F, \ldots, \theta_n^F$ so as to maximize

$$L(\theta) = p(\mathbf{D} \mid \theta) = \prod_{i=1}^{N} p(x^i \mid \theta)$$

- Approach: take derivatives
- Can also use MAP or Bayesian approaches

Counts

N = total number of instances

N_T = number of instances with class T

N_F = number of instances with class F

$N_{i,T}^T$ = number of instances with class T and X_i = T

$N_{i,F}^T$ = number of instances with class T and X_i = F

$N_{i,T}^F$ = number of instances with class F and X_i = T

$N_{i,F}^F$ = number of instances with class F and X_i = F

$4n+3$ counts

Maximum likelihood for P(C)

$$\theta_C = \frac{N_T}{N}$$

- The probability of C being T is the fraction of times in the data C was T
- This is the completely obvious result!
  - just like Gaussian density estimator

MAP estimate for P(C)

- Prior over θ_c is beta(α_c,β_c)
- MAP estimate for θ_c is $\frac{N_T + \alpha - 1}{N + \alpha + \beta - 2}$

Maximum Likelihood for P(X^i | C)

$$\theta_i^T = \frac{N_{i,T}^T}{N_T}$$

- Probability that X_i is T given that class is T is fraction of times that X_i is T out of all times that class is T
- Similarly, $\theta_i^F = \frac{N_{i,T}^F}{N_F}$

MAP estimate for P(X^i | C)

- Priors for probability that X^i is true given the two possible classes
  - prior given class T: beta(α_i^T,β_i^T)
  - prior given class F: beta(α_i^F,β_i^F)
- MAP parameter estimates

$$\theta_i^T = \frac{N_{i,T}^T + \alpha_i^T - 1}{N_i^T + \alpha_i^T + \beta_i^T - 2} \qquad \theta_i^F = \frac{N_{i,T}^F + \alpha_i^F - 1}{N_i^F + \alpha_i^F + \beta_i^F - 2}$$

Sufficient statistics

- These counts tell us all we need to know to estimate the parameters
- They are called **sufficient statistics**
- We don't need to know **joint statistics** of different variables
  - for example, we don't need to know how often X_1 and X_2 were both T when C was T
- Naïve Bayes' independence assumptions

Multi-valued class and attributes

- |C| = k and |X| = m
- Parameters
  - $P(C=c) = \theta_c$
  - $P(X_i=x \mid C=c) = \theta_{i,x}^c$
  - How many? k*m*n + k -1
- Counts
  - N = total number of instances
  - N_c = total number of instances with class c
  - $N_{i,x}^c$ = total number of instances with class c and $X_i = x$
  - How many? k*m*n + k +1

Pros and cons of naïve Bayes

- Advantages:
  - No curse of dimensionality
  - Simple, easy to implement
  - Fast learning
  - In each dimension, makes no assumption about form of distribution
- Disadvantages:
  - Can perform poorly if independence assumptions do not hold
  - Maximum likelihood can overfit data
    - but Bayesian approach is easy to implement

Naïve Bayes classifier applications

- Text classification
  - email: spam vs. not spam
  - news categories
  - emotion (ditto the donkey)

Overview

- The Naïve Bayes model
- Discrete density estimation revisited
- Learning the Naïve Bayes model
- Autoclass: Naïve Bayes for clustering
- EM for Gaussian mixture models

Autoclass

- Like Naïve Bayes, but for unsupervised learning
- Unlabeled training data $D^1,\ldots,D^N$ where

  $D^i = \langle x_1^i,...x_n^i \rangle$

  - no class labels
- Goal: learn a Naïve Bayes model
  - $P(C)$
  - $P(X_i \mid C)$ for each attribute

Maximum likelihood

- Parameters: θ_C; θ_i^T and θ_i^F for each attribute
- Approach: find θ that maximizes

  $$L(\theta) = p(\mathbf{D} \mid \theta) = \prod_{i=1}^{N} p(x^i \mid \theta)$$

- This is a difficult problem
  - we don't have sufficient statistics because the class labels are missing

Expectation-Maximization (EM)

- A general, powerful algorithm for learning probabilistic models from data that is not fully observed
- Works in many different frameworks
  - Autoclass is just one example

Two observations

1. If we know the sufficient statistics of the data, we can choose parameter values so as to maximize the likelihood.
2. If we know the model parameters, we can compute a probability distribution over the missing attributes. From these, we get the **expected sufficient statistics**.

Expected sufficient statistics

- From observed data and model parameters, we get the probability of every possible **completion** of the data
- Each completion defines sufficient statistics
- The **expected sufficient statistics** is the expectation, taken over all possible completions, of the sufficient statistics for each completion

EM (general form)

Set θ to some initial values.

Repeat:

$\theta_{old} = \theta$

E-step (Expectation): Compute the expected sufficient statistics.

M-step (Maximization): Choose θ so as to maximize the likelihood of the expected sufficient statistics.

Until θ is close to θ_{old}.

Example: E step

$\theta_C = 0.7$
$\theta_1^T = 0.9$
$\theta_1^F = 0.3$
$\theta_2^T = 0.6$
$\theta_2^F = 0.2$

D =

X_1	X_2
F	T
T	T

Completions

	X_1	X_2	C
1	F	T	F
	T	T	F

	X_1	X_2	C
2	F	T	F
	T	T	T

	X_1	X_2	C
3	F	T	T
	T	T	F

	X_1	X_2	C
4	F	T	T
	T	T	T

P(Completion 1) $\propto P(X_1=F, X_2=T, C=F) * P(X_1=T, X_2=T, C=F)$
$= P(C=F)P(X_1=F|C=F)P(X_2=T|C=F)$
$P(C=F)P(X1=T|C=F)P(X2=T|C=F)$
$= 0.3 * 0.7 * 0.2 * 0.3 * 0.3 * 0.2$
$= 0.000756$

Probabilities of completions

$\theta_C = 0.7$
$\theta_1^T = 0.9$
$\theta_1^F = 0.3$
$\theta_2^T = 0.6$
$\theta_2^F = 0.2$

D =

X_1	X_2
F	T
T	T

Completions

	X_1	X_2	C
1	F	T	F
	T	T	F

	X_1	X_2	C
2	F	T	F
	T	T	T

	X_1	X_2	C
3	F	T	T
	T	T	F

	X_1	X_2	C
4	F	T	T
	T	T	T

P(Completion 1) $\propto 0.3 * 0.7 * 0.2 * 0.3 * 0.3 * 0.2 = 0.000756$
P(Completion 2) $\propto 0.3 * 0.7 * 0.2 * 0.7 * 0.9 * 0.6 = 0.015876$
P(Completion 3) $\propto 0.7 * 0.1 * 0.6 * 0.3 * 0.3 * 0.2 = 0.000756$
P(Completion 4) $\propto 0.7 * 0.1 * 0.6 * 0.7 * 0.9 * 0.6 = 0.015876$

Probabilities of completions

$\theta_C = 0.7$
$\theta_1^T = 0.9$
$\theta_1^F = 0.3$
$\theta_2^T = 0.6$
$\theta_2^F = 0.2$

D =

X_1	X_2
F	T
T	T

Completions

	X_1	X_2	C
1	F	T	F
	T	T	F

	X_1	X_2	C
2	F	T	F
	T	T	T

	X_1	X_2	C
3	F	T	T
	T	T	F

	X_1	X_2	C
4	F	T	T
	T	T	T

P(Completion 1) = 0.0227
P(Completion 2) = 0.4773
P(Completion 3) = 0.0227
P(Completion 4) = 0.4773

Expected sufficient statistics

$\theta_C = 0.7$
$\theta_1^T = 0.9$
$\theta_1^F = 0.3$
$\theta_2^T = 0.6$
$\theta_2^F = 0.2$

D =

X_1	X_2
F	T
T	T

Completions

	X_1	X_2	C
1	F	T	F
	T	T	F

	X_1	X_2	C
2	F	T	F
	T	T	T

	X_1	X_2	C
3	F	T	T
	T	T	F

	X_1	X_2	C
4	F	T	T
	T	T	T

$E[N_T] = 0.0227*0 + 0.4773*1 + 0.0227*1 + 0.4773*2 = 1.4546$
$E[N_F] = N - E[N_T] = 2 - 1.4546 = 0.5454$
$E[N^T_{1,T}] = 0.0227*0 + 0.4773*1 + 0.0227*0 + 0.4773*1 = 0.9546$
$E[N^F_{1,T}] = 0.0227*1 + 0.4773*0 + 0.0227*1 + 0.4773*0 = 0.0454$
$E[N^T_{2,T}] = 1.4546$ $E[N^F_{2,T}] = 0.5454$

Maximum likelihood estimates

$E[N_T] = 1.4546$ $E[N_F] = 0.5454$
$E[N^T_{1,T}] = 0.9546$ $E[N^F_{1,T}] = 0.0454$
$E[N^T_{2,T}] = 1.4546$ $E[N^F_{2,T}] = 0.5454$

$\theta_C = E[N_T] / N = 1.4546 / 2 = 0.7273$
$\theta_1^T = E[N^T_{1,T}] / E[N_T] = 0.9546 / 1.4546 = 0.6563$
$\theta_1^F = E[N^F_{1,T}] / E[N_F] = 0.0454 / 0.5454 = 0.0832$
$\theta_2^T = E[N^T_{2,T}] / E[N_T] = 1.4546 / 1.4546 = 1$
$\theta_2^T = E[N^F_{2,T}] / E[N_F] = 0.5454 / 0.5454 = 1$

Problem

- What is the problem with the approach I just showed you?
- Number of completions is exponential in number of instances

Key observation

- We don't care about exact completions, only expected sufficient statistics
- Each instance contributes **separately** to expected sufficient statistics
- So, we can
  - enumerate completions of each instance separately
  - get probability of each completion
  - get expected contribution of that instance to sufficient statistics

E step for naïve Bayes

- $E[N_T]$ is the expected number of instances in which the class is T
- Each instance has a probability of the class being T
- Each instance contributes that probability to $E[N_T]$
- In symbols:

$$E[N_T] = \sum_{j=1}^{N} P(C^j = T \mid x_1^j, \ldots, x_n^j)$$
$$\propto \sum_{j=1}^{N} P(C^j = T) \prod_{i=1}^{n} P(x_i^j \mid C^j = T)$$

E step for naïve Bayes cont'd

- $E\left[N^T_{i,T}\right]$ is the expected number of times the class is T when X_i is T
- If an instance has $X_i \neq T$, it contributes 0 to $E\left[N^T_{i,T}\right]$
- If an instance has $X_i = T$, it contributes the probability that the class is T to $E\left[N^T_{i,T}\right]$
- In symbols:

$$E\left[N^T_{i,T}\right] = \sum_{j:x_i^j=T}^{N} P(C^j = T \mid x_1^j, \ldots, x_n^j)$$
$$\propto \sum_{j:x_i^j=T}^{N} P(C^j = T) \prod_{i=1}^{n} P(x_i^j \mid C^j = T)$$

A notational convenience

- Encode T as 1, F as 0
- Then

$$P(x_i^j \mid C^j = T) = (\theta_i^T)^{x_i^j}(1-\theta_i^T)^{(1-x_i^j)}$$
$$P(x_i^j \mid C^j = F) = (\theta_i^F)^{x_i^j}(1-\theta_i^F)^{(1-x_i^j)}$$

- This has no significance, just makes the algorithm easier to write down

Autoclass algorithm

Set θ_C, θ_i^T and θ_i^F to arbitrary values for all attributes

Repeat until convergence:

Expectation step

Maximization step

Autoclass: expectation step

$E[N_T] = 0$

$E\left[N_{i,T}^T\right] = 0$ for all instances

$E\left[N_{i,T}^F\right] = 0$ for all instances

For each instance D^i:

$$p_T = \theta_C \prod_{i=1}^{n} (\theta_i^T)^{x_i^j} (1-\theta_i^T)^{(1-x_i^j)}$$

$$p_F = (1-\theta_C) \prod_{i=1}^{n} (\theta_i^F)^{x_i^j} (1-\theta_i^F)^{(1-x_i^j)}$$

$$q = \frac{p_T}{p_T + p_F}$$

$E[N_T]$ += q

for each attribute i:

if x_i == T:

$E\left[N_{i,T}^T\right]$ += q

$E\left[N_{i,T}^F\right]$ += (1-q)

Autoclass: maximization step

$$\theta_C = \frac{E[N_T]}{N}$$

For each attribute i:

$$\theta_i^T = \frac{E\left[N_{i,T}^T\right]}{E\left[N_T\right]}$$

$$\theta_i^F = \frac{E\left[N_{i,T}^F\right]}{N - E\left[N_T\right]}$$

Example: E step

$\theta_C = 0.7$

$\theta_1^T = 0.9$

$\theta_1^F = 0.3$

$\theta_2^T = 0.6$

$\theta_2^F = 0.2$

D = $\begin{array}{cc} X_1 & X_2 \\ F & T \\ T & T \end{array}$

$E\left[N_T\right] = 0$

$E\left[N_{1,T}^T\right] = 0$

$E\left[N_{1,T}^F\right] = 0$

$E\left[N_{2,T}^T\right] = 0$

$E\left[N_{2,T}^F\right] = 0$

For instance 1:

$$p_T = 0.7 * (0.9^0 * 0.1^1) * (0.6^1 * 0.4^0) = 0.042$$

$$p_F = 0.3 * (0.3^0 * 0.7^1) * (0.2^1 * 0.8^0) = 0.042$$

$$q = \frac{0.042}{0.042 + 0.042} = 0.5$$

Example: E step

$\theta_C = 0.7$

$\theta_1^T = 0.9$

$\theta_1^F = 0.3$

$\theta_2^T = 0.6$

$\theta_2^F = 0.2$

D = $\begin{array}{cc} X_1 & X_2 \\ F & T \\ T & T \end{array}$

$E\left[N_T\right] = 0.5$

$E\left[N_{1,T}^T\right] = 0.0$

$E\left[N_{1,T}^F\right] = 0.0$

$E\left[N_{2,T}^T\right] = 0.5$

$E\left[N_{2,T}^F\right] = 0.5$

For instance 1:

$$p_T = 0.7 * (0.9^0 * 0.1^1) * (0.6^1 * 0.4^0) = 0.042$$

$$p_F = 0.3 * (0.3^0 * 0.7^1) * (0.2^1 * 0.8^0) = 0.042$$

$$q = \frac{0.042}{0.042 + 0.042} = 0.5$$

Example: E step

$\theta_C = 0.7$

$\theta_1^T = 0.9$

$\theta_1^F = 0.3$

$\theta_2^T = 0.6$

$\theta_2^F = 0.2$

D = $\begin{array}{cc} X_1 & X_2 \\ F & T \\ T & T \end{array}$

$E\left[N_T\right] = 0.5$

$E\left[N_{1,T}^T\right] = 0.0$

$E\left[N_{1,T}^F\right] = 0.0$

$E\left[N_{2,T}^T\right] = 0.5$

$E\left[N_{2,T}^F\right] = 0.5$

For instance 2:

$$p_T = 0.7 * (0.9^1 * 0.1^0) * (0.6^1 * 0.4^0) = 0.378$$

$$p_F = 0.3 * (0.3^1 * 0.7^0) * (0.2^1 * 0.8^0) = 0.018$$

$$q = \frac{0.378}{0.378 + 0.018} = 0.95$$

Example: E step

$\theta_C = 0.7$
$\theta_1^T = 0.9$
$\theta_1^F = 0.3$
$\theta_2^T = 0.6$
$\theta_2^F = 0.2$

D =

X_1	X_2
F	*T*
T	*T*

$E[N_T] = 1.45$
$E[N_{1,T}^T] = 0.95$
$E[N_{1,T}^F] = 0.05$
$E[N_{2,T}^T] = 1.45$
$E[N_{2,T}^F] = 0.55$

For instance 2:

$$p_T = 0.7 * (0.9^1 * 0.1^0) * (0.6^1 * 0.4^0) = 0.378$$

$$p_F = 0.3 * (0.3^1 * 0.7^0) * (0.2^1 * 0.8^0) = 0.018$$

$$q = \frac{0.378}{0.378 + 0.018} = 0.95$$

Example: M step

$\theta_C = 0.7$
$\theta_1^T = 0.9$
$\theta_1^F = 0.3$
$\theta_2^T = 0.6$
$\theta_2^F = 0.2$

D =

X_1	X_2
F	*T*
T	*T*

$E[N_T] = 1.45$
$E[N_{1,T}^T] = 0.95$
$E[N_{1,T}^F] = 0.05$
$E[N_{2,T}^T] = 1.45$
$E[N_{2,T}^F] = 0.55$

$$\theta_C = \frac{1.45}{2.0} = 0.72$$

$$\theta_1^T = \frac{0.95}{1.45} = 0.65 \qquad \theta_2^T = \frac{1.45}{1.45} = 1.0$$

$$\theta_1^F = \frac{0.05}{0.55} = 0.09 \qquad \theta_2^F = \frac{0.55}{0.55} = 1.0$$

Convergence

- EM improves the likelihood on every iteration
- It is guaranteed to converge to a maximum of the likelihood function
- But it may be a local maximum!
  - as with neural networks, this can be a serious problem

A Tip

- Don't start EM with symmetric parameter values
  - in particular, don't start with uniform
- If you do, it won't have any information to tip things one way or the other
- Symmetric parameter values are a saddle point

Overview

- The Naïve Bayes model
- Discrete density estimation revisited
- Learning the Naïve Bayes model
- Autoclass: Naïve Bayes for clustering
- EM for Gaussian mixture models

Continuous attributes

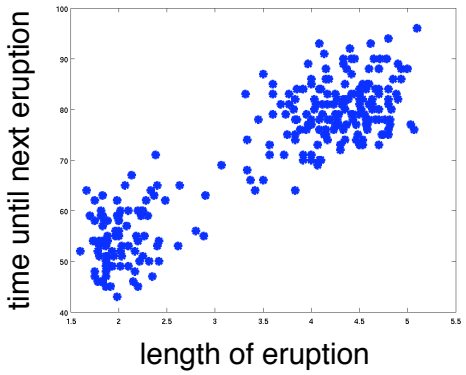
- Autoclass assumes discrete attributes
  - uses counts for each value to estimate parameters
- Many domains are more naturally represented as continuous values
  - x,y(,z) locations
  - time

Continuous attributes so far...

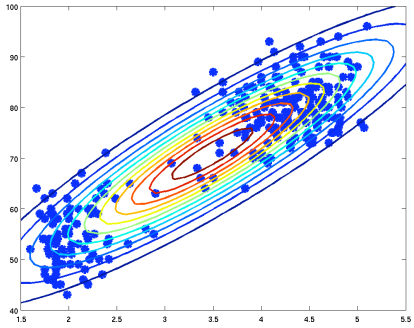
- Density estimation
  - univariate and multivariate Gaussians
- Clustering
  - hierarchical agglomerative clustering
  - k-means

Old faithful data

How can we do density estimation on this data?

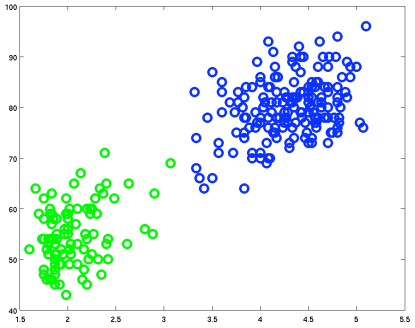


Multivariate Gaussian?



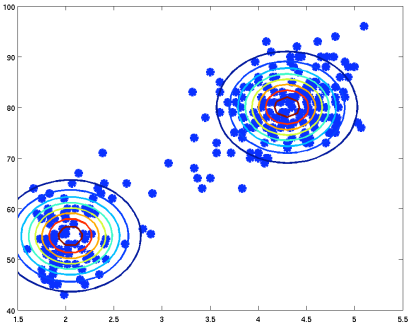
k-means?

Nice clustering. What about density estimate?



k-means

- Covariances shouldn't be spherical
- Clusters shouldn't have equal weight



Mixture of Gaussians

- $p(x) = \sum_{j=1}^{k} \mathcal{N}(x \mid \mu_j, \Sigma_j) P(c_j)$
- k-means is a mixture of Gaussians where
  - covariance is diagonal, equal in all dimension, and equal for all clusters
  - $P(C)$ is uniform
  - $p(x) \propto \sum_{j=1}^{k} \mathcal{N}(x \mid \mu_j, \Sigma)$

$$\Sigma = \begin{bmatrix} \sigma^2 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \sigma^2 \end{bmatrix}$$

Mixture of Gaussians parameters

- Means of Gaussians: μ_j
- Covariances of Gaussians: Σ_j
- Mixing coefficients (P(C)): π_j
  - $\sum_{j=1}^{k} \pi_j = 1$
- Assignment of instances to Gaussians: $\gamma_{i,j}$
  - instances are assigned to multiple clusters: soft assignment
  - $\sum_{j=1}^{k} \gamma_{i,j} = 1$

Mixture of Gaussians algorithm

MixtureOfGaussiansEM($\{x^1,...,x^N\}$, k) =

Assign instances to clusters arbitrarily

Maximization step to initialize parameters

Repeat until convergence:

Expectation step

Maximization step

Mixture of Gaussians E step

For each instance i:

For each cluster j

compute cluster responsibility

$$\gamma_{i,j} = \frac{\pi_k \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{m=1}^{k} \pi_m \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)}$$

Mixture of Gaussians M step

For each cluster j:

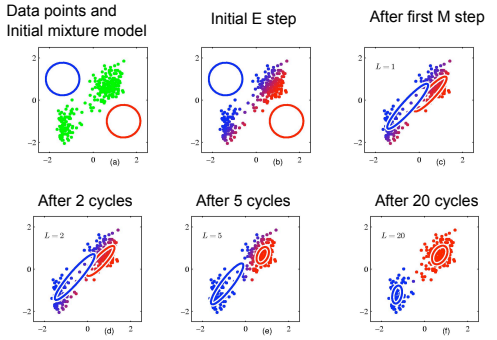
$$N_j = \sum_{i=1}^{N} \gamma_{i,j}$$

$$\boldsymbol{\mu}_j = \frac{1}{N_j} \sum_{i=1}^{N} \gamma_{i,j} \mathbf{x}^i$$

$$\boldsymbol{\Sigma}_j = \frac{1}{N_j} \sum_{i=1}^{N} \gamma_{i,j} (\mathbf{x}^i - \boldsymbol{\mu}_j)(\mathbf{x}^i - \boldsymbol{\mu}_j)^T$$

$$\pi_j = \frac{N_j}{N}$$

Old faithful data



[Bishop, 2006]

Gaussian mixtures and naïve Bayes

- Naïve Bayes assumption was that the attributes were independent
- What does the naïve Bayes assumption mean for a mixture of Gaussians model?

  - diagonal covariance matrix
- Often this assumption is made because it is easier to fit a model with fewer parameters

  - still more flexible than k-means
    - learned variance in each dimension
    - non-uniform P(C)

More EM hints

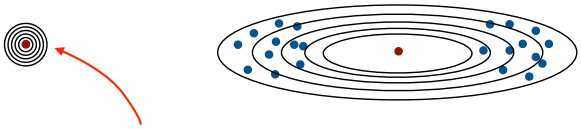
- As with Autoclass, don't initialize with symmetric parameters
- Susceptible to local optima, so do random restarts

Local optima

- With variable covariance, what is global optimum for this data?

Local optima

- With variable covariance, what is global optimum for this data?



infinite likelihood!

More EM hints

- As with Autoclass, don't initialize with symmetric parameters
- Susceptible to local optima, so do random restarts
  - but sometimes we don't actually want the global optimum!

Using logs

- During E step, exponentiation can get ugly

$$\gamma_{i,j} = \frac{\pi_k \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{m=1}^{k} \pi_m \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)}$$

$$\mathcal{N}(\mathbf{x} \mid \mu, \Sigma) = \frac{1}{(2\pi)^{\frac{n}{2}} |\boldsymbol{\Sigma}|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} (\mathbf{x}-\boldsymbol{\mu})\right)$$

- We can take the log and then exponentiate

$$\gamma_{i,j} = \exp\left(\log \pi_k \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) - \log \sum_{m=1}^{k} \pi_m \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)\right)$$

Logsumexp

$$\gamma_{i,j} = \exp\left(\log \pi_k \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) - \log \sum_{m=1}^{k} \pi_m \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)\right)$$

$\log \sum_i \exp(-x_i)$ if some x_i is too big, no good!

Logsumexp

$$\log \sum_i \exp(-x_i)$$

$$a = \max_i x_i$$

$$\log\left[\sum_i \exp(x_i)\right] = \log\left[\sum_i \exp(x_i - a + a)\right]$$

$$= \log\left[\sum_i \exp(-x_i + a)\exp(-a)\right]$$

$$= \log\left[\exp(-a)\sum_i \exp(-x_i + a)\right]$$

$$= -a + \log\left[\sum_i \exp(-x_i + a)\right]$$

Logsumexp

$$\gamma_{i,j} = \exp\left(\log \pi_k \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) - \log \sum_{m=1}^{k} \pi_m \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)\right)$$

$\log \sum_i \exp(-x_i)$ if some x_i is too big, no good!

$$a = \max_i x_i$$

$$\log\left[\sum_i \exp(-x_i)\right] = -a + \log\left[\sum_i \exp(-x_i + a)\right]$$

Generalized EM

- For Gaussians, we can find maximum likelihood parameters analytically
- For some likelihood functions, this is not true
- For EM to work, need only that M step increase the likelihood
- Can use other methods to increase likelihood
  - e.g. gradient descent
- BUT, if so, not always clear the EM is useful
  - could maximize likelihood function directly