Bayesian Learning

Computer Science 760

Patricia J Riddle

Introduction

- Bayesian learning algorithms calculate explicit probabilities for hypotheses
- Naïve Bayes classifier is among the most effective in classifying test documents
- Bayesian methods can also be used to analyze other algorithms
- Training example incrementally increases or decreases the estimated probability that a hypothesis is correct
- Prior knowledge can be combined with observed data to determine the final probability of a hypothesis

Prior Knowledge

- Prior knowledge is:
  1. Prior probability for each candidate hypothesis and
  2. A probability distribution over observed data for each possible hypothesis

Bayesian Methods in Practice

- Bayesian methods accommodate hypotheses that make probabilistic predictions “this pneumonia patient has a 98% chance of complete recovery”
- New instances can be classified by combining the predictions of multiple hypotheses, weighted by their probabilities
- Even when computationally intractable, they can provide a standard of optimal decision making against which other practical measures can be measured

Practical Difficulties

1. Require initial knowledge of many probabilities - estimated based on background knowledge, previously available data, assumptions about the form of the underlying distributions
2. Significant computational cost to determine the Bayes optimal hypothesis in the general case - linear in the number of candidate hypotheses - in certain specialized situations the cost can be significantly reduced

Bayes Theorem Intuition

- Learning - we want the best hypothesis from some space H, given the observed training data D. Best can be defined as most probable given the data D plus any initial knowledge about prior probabilities of the various hypotheses in H.
- This is a direct method!!! (No Search)

Bayes Terminology

- P(h) - the initial probability that hypothesis h holds before we observe the training data - prior probability - if we have no prior knowledge we assign the same initial probability to them all (it is trickier than this!!)
- P(D) - prior probability training data D will be observed given no knowledge about which hypothesis holds
- P(D|h) - the probability of observing data D given that hypothesis h holds
- P(h|D) - the probability that h holds given the training data D - ***posterior probability***

Bayes Theorem

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

- Probability increases with P(h) and P(Dlh) and decreases with P(D) - this last is not true with a lost of other scoring functions!

MAP & ML Hypothesis

- So we want a **maximum a posteriori** hypothesis (MAP) - P(D) same for every hypothesis

$$h_{MAP} \equiv \arg\max_{h \in H} P(D \mid h) P(h)$$

- If we assume every hypothesis is equally likely a priori then we want the **maximum likelihood** hypothesis

$$h_{ML} \equiv \arg\max_{h \in H} P(D \mid h)$$

- Bayes theorem is more general than Machine Learning!!

A general Example

- Two hypothesis: the patient has cancer, $\oplus$, the patient doesn't have cancer, $\ominus$
- Prior knowledge: over the entire population of people .008 have cancer
- The lab test returns a correct positive result in 98% of the cases in which cancer is actually present and a correct negative in 97% of the cases in which cancer is actually not present
- P(cancer) = .008, P(¬cancer) = .992
- P($\oplus$|cancer) = .98, P($\ominus$|cancer) = .02
- P($\oplus$|¬cancer)=.03, P($\ominus$|¬cancer)=.97
- So given a new patient with a positive lab test, should we diagnose the patient as having cancer or not??
- Which is the MAP hypothesis?

Example Answer

- Has cancer - P(⊕|cancer)P(cancer) = (.98).008 = .0078
- Doesn't have cancer -
- P(⊕|¬cancer)P(¬cancer)=(.03).992=.0298
- h_{MAP}=¬cancer
- Exact posterior probabilities -

$$P(cancer \mid \oplus) = \frac{P(\oplus \mid cancer)P(cancer)}{P(\oplus)} = \frac{.0078}{P(\oplus)}$$

- Posterior as a real probability

$$\frac{.0078}{P(\oplus \mid cancer) + P(\oplus \mid \neg cancer)} = \frac{.0078}{.0078 + .0298} = .21$$

Minimum Description Length

- Let us look at h_{MAP} in the light of basic concepts of information theory

- $h_{MAP} \equiv \text{argmax}_{h \in H} P(D|h)\, P(h)$

  $= \text{argmin}_{h \in H} - \log_2 P(D|h) - \log_2 P(h)$

- This can be interpreted as a statement that short hypotheses are preferred.

Information Theory

- Consider the problem of designing a code to transmit messages drawn at random, where the probability of encountering message i is p_i.
- We want the code that minimizes the expected number of bits we must transmit in order to encode a message drawn at random.
- To minimize the expected code length we should assign shorter codes to more probable messages.

Optimal Code

- Shannon and Weaver (1949) showed the optimal code assigns $-\log_2 p_i$ bits to encode message i. Where p_i is the probability of i appearing.
- $L_C(i)$ is the description length of message i with respect to code C.
- L_{CH} is the size of the description of the hypothesis h using the optimal representation for encoding the hypothesis space H.

$$L_{C_H}(h) = -\log_2 P(h)$$

- $L_{CD|h}$ is the size of the description of the training data D given the hypothesis h using the optimal representation for encoding the data D assuming that both the sender and receiver know the hypothesis h.

$$L_{C_{D|h}}(D \mid h) = -\log_2 P(D \mid h)$$

Applying MDL

- To apply this principle we must choose specific representations C1 and C2 appropriate for the given learning task!
- **Minimum Description Length Principle**:

$$h_{MDL} \equiv \arg\min_{h \in H} L_{C_1}(h) + L_{C_2}(D \mid h)$$

- ***If*** C1 and C2 are chosen to be optimal encodings for their respective tasks, then $h_{MDL}=h_{MAP}$

MDL Example

- Apply MDL principle to the problem of learning decision trees.
- C_1 is an encoding of trees where the description length grows with the number of nodes in the tree and the number of edges.
- C_2 transmits misclassified examples by identifying
  - which example is misclassified ($\log_2 m$ bits, where m is the number of training instances) and
  - its correct classification ($\log_2 k$ bits, where k is the number of classes).

MDL Intuition

- MDL principle provides a way of trading off hypothesis complexity for the number of errors committed by the hypothesis
- So the MDL principle produces a MAP hypothesis if the encodings C_1 and C_2 are optimal. But to show that we would need all the prior probabilities P(h) as well as P(D|h).
- **No reason** to believe the MDL hypothesis relative to arbitrary encodings should be preferred!!!!

What I hate about MDL

- “But you didn’t find the optimal encodings C1 and C2.”
- “Well it doesn’t matter if you see enough data it doesn’t matter which one you use.”
- So why are we using a Bayesian approach????

Bayes Optimal Classifier

- What is the most probable classification of the new instance given the training data?

- Could just apply the MAP hypothesis, but can do better!!!

Bayes Optimal Intuitions

- Assume three hypothesis h1, h2, h3 whose posterior probabilities are .4, .3 and .3 respectively.
- Thus h1 is the MAP hypothesis.
- Suppose we have a new instance x which is classified positive by h1 and negative by h2 and h3.
- Taking all hypothesis into account, the probability that x is positive is .4 and the probability it is negative is .6.
- The most probable classification (negative) is different than the classification given by the MAP hypothesis!!!

Bayes Optimal Classifier II

- We want to combine the predictions of all hypotheses weighted by their posterior probabilities.

$$P(v_j \mid D) = \sum_{h_i \in H} P(v_j \mid h_i) P(h_i \mid D)$$

- where v_j is from the set of classifications V.

- **Bayes Optimal Classification**:

$$\arg\max_{v_j \in V} \sum_{h_i \in H} P(v_j \mid h_i) P(h_i \mid D)$$

- No other learner using the same hypothesis space and same prior knowledge can outperform this method on average. It maximizes the probability that the new instance is classified correctly.

Gibbs Algorithm

- Bayes Optimal is quite costly to apply. It computes the posterior probabilities for every hypothesis in H and combines the predictions of each hypothesis to classify each new instance.
- An alternative (less optimal) method:
  1. Choose a hypothesis h from H at random, according to the posterior probability distribution over H.
  2. Use h to predict the classification of the next instance x.
- Under certain conditions the expected misclassification error for Gibbs algorithm is at most twice the expected error of the Bayes optimal classifier.

What is Naïve Bayes?

- Results comparable to ANN and decision trees in some domains
- Each instance x is described by a conjunction of attribute values and the target value f(x) can take any value from a set V. A set of training instances are provided and a new instance is presented and the learner is asked to predict the target value.

$$V_{MAP} = \arg\max_{v_j \in V} P(v_j \mid a_1, a_2 ... a_n)$$

$$= \arg\max_{v_j \in V} P(a_1, a_2, ... a_n \mid v_j) P(v_j)$$

- $P(v_j)$ is estimated by the frequency of each target value in the training data.
- Cannot use frequency for $P(a_1, a_2, \ldots a_n \mid v_j)$ unless we have a very, very large set of training data to get a reliable estimate.

Conditional Independence

- Naïve Bayes assumes attribute values are conditionally independently given the target value - $P(a_1, a_2, \ldots a_n \mid v_j) = \prod_i P(a_i \mid v_j)$

- **Naïve Bayes Classifier**:

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

- where v_{NB} denotes the target values
- $P(a_i|v_j)$ can be estimated by frequency

When is Naïve Bayes a MAP?

- When conditional independence assumption is satisfied the naïve Bayes classification is a MAP classification

- Naïve Bayes entails no search!!

An Example

- Target concept PlayTennis
- Classify the following instance: <Outlook=sunny, Temperature = cool, Humidity = high, Wind = strong>

$$v_{NB} = \arg\max_{v_j \in \{yes,no\}} P(v_j)P(Outlook = sunny \mid v_j)P(Temperature = cool \mid v_j)$$
$$P(Humidity = high \mid v_j)P(Wind = strong \mid v_j)$$

- P(PlayTennis=yes)=9/14=.64
- P(PlayTennis=no)=5/14=.36
- P(Wind=strong|PlayTennis=yes)=3/9=.33
- P(Wind=strong|PlayTennis=no)=3/5=.60
- …..

An Example II

- P(yes) P(sunny|yes) P(cool|yes) P(high|yes) P(strong|yes) = .0053
- P(no) P(sunny|no) P(cool|no) P(high|no) P(strong|no) = .0206
- Naïve Bayes returns "Play Tennis = no" with probability

$$\frac{.0206}{.0206 + .0053} = 0.7954 = 79.5\%$$

Learning to Classify Text

- Learn “electronic news articles I find interesting” or “pages on www that discuss machine learning topics”
- Two design issues: attribute representation and probability estimates
- Define an attribute for each word position in the document and define the value to be the word found in that position
- Notice that short documents will have fewer attributes than longer ones

Sample Document

- "Our approach to representing….us any trouble."

$$v_{NB} = \arg\max_{v_j \in \{like, dislike\}} P(v_j)$$

$$P(a_1 = "our" \mid v_j)$$

$$P(a_2 = "approach" \mid v_j)$$

$$\ldots$$

$$P(a_{111} = "trouble" \mid v_j)$$

Independence Assumption

- The independence assumption states that the word probabilities for one test position are independent of words that occur in other positions.
- This is clearly incorrect, but in practice naïve Bayes performs remarkably well in many text classification problems.
- Requires estimates of $P(v_j)$ and $P(a_i=w_k|v_j)$ where w_k is the k^{th} word in the vocabulary.
- The first is easy but the second is too computationally complex.

Additional Assumptions

- For 111 text positions and 2 possible targets and 50,000 vocabulary words, the number of probability estimates is 2 * 111 * 50,000 or about 10 million.
- An Additional assumption is added that the probability of encountering a specific word is independent of the specific word position.
- The complexity is now 2* 50,000.
- Even more importantly many less training examples are needed!!!

M-estimate

- The m-estimate is used for estimating probabilities

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

- Where n_k is the number of times word w_k is found in the document.
- And n is the number of distinct words in the text.

Terms for Text Algorithm

- Examples is a set of text documents along with their target values.
- V is the set of all possible target values.

- This function learns the probability terms $P(w_k|v_j)$, describing the probability that a randomly drawn word from a document in class v_j will be the English work w_k.
- It also learns the class prior probabilities $P(v_j)$.

Learn Naïve Bayes Text Algorithm

Learn_Naïve_Bayes_Text(Examples,V)

1. Collect all words, punctuation, and other tokens, that occur in Examples
   - Vocabulary <-- the set of all distinct words and other tokens occurring in any text document from Examples
2. Calculate the required $P(v_j)$ and $P(w_k|v_j)$ probability terms
   - For each target value v_j in V do
     - $docs_j$ <-- the subset of documents from Examples for which the target value is v_j
     - $P(v_j)$ <-- $|docs_j| / |Examples|$
     - $Text_j$ <-- a single document created by concatenating all members of $docs_j$
     - n <-- total number of times word wk occurs in $Text_j$
     - For each word w_k in Vocabulary
       - n_k <-- number of times word w_k occurs in $Text_j$
       - $P(w_k|v_j)$ <-- $n_k+1 / n+|Vocabulary|$

Test Classify algorithm

Classify_Naïve_Bayes_Text(Doc)

Return the estimated target value for the document Doc. a_i denotes the word found in the i^{th} position within Doc.

- positions <-- all word positions in Doc that contain tokens found in Vocabulary
- Return v_{NB}, where

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

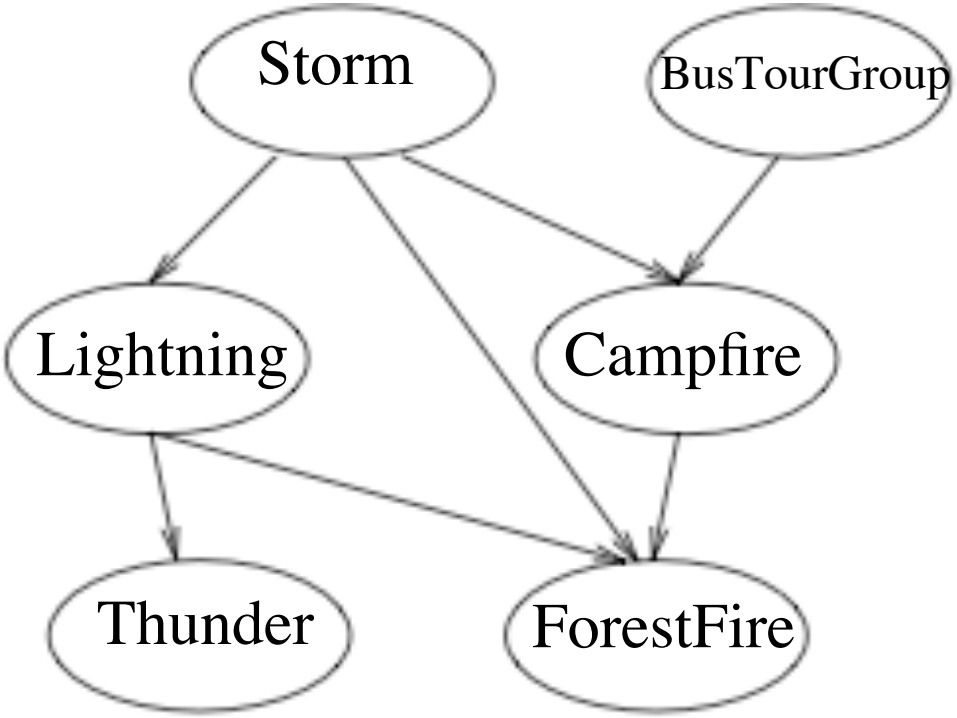
Experimental Results

- Newsgroup posting service - learns to assign documents to the appropriate newsgroup
- 20 newsgroups, 1,000 articles from each newsgroup, learn over 2/3. Random accuracy would be 5%. The program achieved 89%. Used algorithm above but only used words which occurred at least 3 times and were not the 100 most frequent (leaving 38,500 words).
- Usnet articles I find interesting - Newsweeder system - for some users increases rate of interesting articles from 16% to 59%.

Bayesian Belief Networks

- Naïve Bayes assumes all the attributes are conditionally independent
- Bayesian Belief Networks (BBNs) describe a joint probability distribution over a set of variables by specifying a set of conditional independence assumptions and a set of conditional probabilities
- X is conditionally independent of Y means $P(X|Y,Z) = P(X|Z)$

A Bayesian Belief Network



	S,B	S,¬B	¬S,B	¬S,¬B
C	0.4	0.1	0.8	0.2
¬C	0.6	0.9	0.2	0.8

Campfire

Representation

- Each variable is represented by a node and has two types of information specified.
  1. Arcs representing the assertions that the variable is conditionally independent of its nondescendents given its immediate predecessors (I.e., Parents). X is a descendent of Y if there is a directed path from Y to X.
  2. A conditional probability table describing the probability distribution for that variable given the values of its immediate predecessors. This joint probability is computed by

$$P(y_1,\ldots,y_n) = \prod_{i=1}^{n} P(y_i \mid Parents(y_i))$$

Representation II

- Campfire is conditionally independent of its nondescendents Lightning and Thunder given its parents Storm and BusTourGroup

$$P(Campfire = True \mid Storm = True, BusTourGroup = True) = 0.4$$

- Also notice that ForestFire is conditionally independent of BusTourGroup and Thunder given Campfire and Storm and Lightning.
- Similarly, Thunder is conditionally independent of Storm, BusTourGroup, Campfire, and ForestFire given Lightning.
- BBNs are a convenient way to represent causal knowledge. The fact that Lightning causes Thunder is represented in the BBN by the fact that Thunder is conditionally independent of other variables in the network given the value of Lightning.

Inference

- Can we use the BBN to infer the value of a target variable ForestFire given the observed values of the other variables.
- Infer not a single value but the probability distribution for the target variable which specifies the probability it will take on each possible value given the observed values of the other variables
- Generally, we may wish to infer the probability distribution of a variable (e.g., ForestFire) given observed values for only a subset of the other variables (e.g., Thunder and BusTourGroup are the only observed values available).
- Exact inference of probabilities (and even some approximate methods) for an arbitrary BBN is known to be NP-hard.
- Monte Carlo methods provide approximate solutions by randomly sampling the distributions of the unobserved variables

Learning BBNs

- If the network structure was given in advance and the variables are fully observable, then just use the Naïve Bayes formula modulo only some of the variables are conditionally independent.
- If the network structure is given but only some of the variables are observable, the problem is analogous to learning weights for the hidden units in an ANN.
- Similarly, use a gradient ascent procedure to search through the space of hypotheses that corresponds to all possible entries in the conditional probability tables. The objective function that is maximized is P(D|h).
- By definition this corresponds to searching for the maximum likelihood hypothesis for the table entries.

Gradient Ascent Training of BBN

- Let w_{ijk} denote a single entry in one of the conditional probability tables. Specifically that variable Y_i will take on value y_{ij} given that its parents U_i take on the values u_{ik}.
- If w_{ijk} is the top right entry, then Y_i is the variable Campfire, U_i is the tuple of parents <Storm, BusTourGroup>, y_{ij}=True and u_{ik}=<False,False>.
- The derivative for each w_{ijk} is

$$\frac{\partial \ln P(D \mid h)}{\partial w_{ij}} = \sum_{d \in D} \frac{P(Y_i = y_{ij}, U_i = u_{ik} \mid d)}{w_{ijk}}$$

Weight Updates

- So back to our example we must calculate P(Campfire = True, Storm = False, BusTourGroup = False | d) for each training example d in D. If the required probability is unobservable then we can calculate it from other variables using standard BBN inference.
- As weights w_{ijk} are updated they must remain in the interval [0,1] and the sum $\Sigma_j w_{ijk}$ remains 1 for all i,k. So must have a two step process.

1. $$w_{ijk} \leftarrow w_{ijk} + \eta \sum_{d \in D} \frac{P_h(y_{ij}, u_{ik} \mid d)}{w_{ijk}}$$
2. Renormalize the weights w_{ijk}

- Will converge to a locally maximum likelihood hypothesis for the conditional probabilities in the BBn.

Learning BBN Structure

- Bayesian scoring metric for choosing among alternative networks.

- The K2 algorithm performs greedy search that trades off network complexity for accuracy over the training data when the data is fully observable.

K2 example

- In experiments K2 was given 3,000 training examples generated at random from a manually constructed BBN containing 27 nodes and 46 arcs representing anesthesia problems in hospital operating rooms.
- In addition it was given an initial ordering over the 37 variables that was consistent with the partial ordering in the actual BBN.
- K2 reconstructed the BBN structure with only the loss of one arc and the addition of another.

Other approaches

- Other approaches infer dependence and independence relationships from the data and then use these relationships to construct BBN.

- This is a very active line of research.

EM Algorithm

- Widely used approach for learning in the presence of unobserved variables.
- Can be used for variables which are never directly observed, provided the general form of the probability distribution is known (unlike Gradient descent).
- EM algorithm used widely in BBN and clustering algorithms, and Partially Observable Markov Models.
- Easiest to describe EM from an example.

Estimating Means of k Gaussians

- The data D is generated by a probability distribution that is a mixture of k distinct normal distributions.
- Each instance is generated by
  1. One of the k distributions is chosen at random.
  2. A single random instance x_i is generated according to the selected distribution.
- To simplify our discussion, we will assume the Normal distributions are chosen at each step based on uniform probability and each of the k Normal distributions has the same variance σ^2 and σ^2 is known.

EM Learning Task

- The learning task is to output h=<μ_1,…,μ_k> which describes the means of the k distributions.
- We would like to find the maximum likelihood hypothesis (I.e., the h that maximizes p(D|h)).
- Finding the mean for a single normal distribution is a special case of the sum of squared errors formula:

$$\mu_{ML} = \arg\min_{\mu} \sum_{i=1}^{m} (x_i - \mu)^2 = \frac{1}{m} \sum_{i=1}^{m} x_i$$

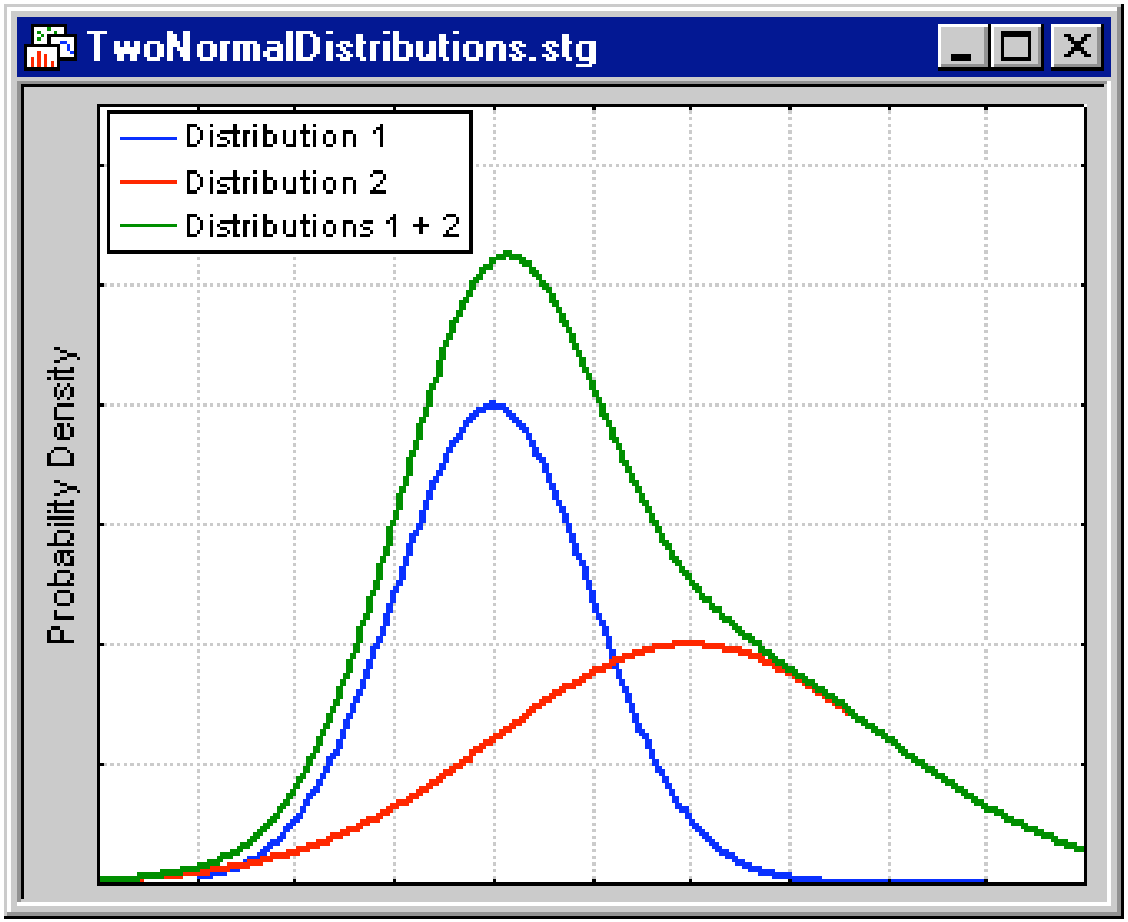
K Normal distributions

- But we have k different Normal distributions, and we cannot observe which instances were generated by which distributions. This is a prototypical example of a problem involving hidden variables!!!
- So each instance can be seen as $<z_i, z_{i1}, z_{i2}>$ where x_i is the observed value of the i^{th} instance and z_{ij} has the value 1 if x_i was created by the j^{th} Normal distribution and 0 otherwise.
- Note if z_{i1} and z_{i2} were observed we could use the sum of squared errors formula above instead of EM.

EM Algorithm

- In a nut shell, EM repeatedly re-estimates the expected values of z_{ij} given its current hypothesis $<\mu_1,\ldots,\mu_k>$ then recalculate the maximum likelihood hypothesis using the expected values for the hidden variables.
- This instance of the EM algorithm is
  1. Calculate the expected value $E[z_{ij}]$ of each hidden variable z_{ij} assuming the current hypothesis $h=<\mu_1,\mu_2>$ holds.
  2. Calculate a new maximum likelihood hypothesis $h'=<\mu'_1,\mu'_2>$ assuming the value taken on by each hidden variable z_{ij} is its expected value $E[z_{ij}]$ calculated in Step 1. Then replace the hypothesis $H=<\mu_1,\mu_2>$ by the new hypothesis $h'=<\mu'_1,\mu'_2>$ and iterate.

K-means Problem Visualization



Practical Implementation for k-means EM

- $E[z_{ij}]$ is just the probability that instance x_i was generated by the j^{th} Normal distribution. $E[z_{ij}] = \frac{p(x = x_i \mid \mu = \mu_j)}{\sum_{n=1}^{2} p(x = x_i \mid \mu = \mu_n)} = \frac{e^{-\frac{1}{2\sigma^2}(x_i - \mu_j)^2}}{\sum_{n=1}^{2} e^{-\frac{1}{2\sigma^2}(x_i - \mu_n)^2}}$
- In the first step,
- In the second step, $\mu_j \leftarrow \frac{1}{m} \sum_{i=1}^{m} E[z_{ij}] x_i$
- Similar to the formula for a single Normal distribution

EM Intuition

- The new formula just has each instance weighted by the expectation $E[z_{ij}]$ that it was generated by the j^{th} Normal distribution.
- The algorithm will converge to a local maximum likelihood hypothesis for $<\mu_1,\mu_2>$.
- The General EM algorithm without simplifications is in the book.

Summary

- Bayesian methods provide a basis for probabilistic learning methods that accommodate knowledge about prior distributions of alternative hypothesis and about the probability of observing the data given various hypothesis. They assign a posterior probability to each candidate hypothesis, based on these assumed priors and the observed data,
- Bayesian methods return the most probable hypothesis (e.g., a MAP hypothesis).
- Bayes Optimal classfier combines the predictions of all alternative hypotheses weighted by their posterior probabilities, to calculate the most probable classification of a new instance.

Naïve Bayes Summary

- Naïve Bayes has been found to be useful in many killer apps.
- It is naïve because it has no street sense….no no no…it incorporates the simplifying assumption that attribute values are conditionally independent given the classification of the instance.
- When this is true naïve Bayes produces a MAP hypothesis.
- Even when the assumption is violated Naïve Bayes tends to perform well.
- BBNs provide a more expressive representation for sets of conditional independence assumptions.

Minimum Description Length Summary

- The Minimum Description Length principle recommends choosing the hypothesis that minimizes the description length of the hypothesis plus the description length of the data given the hypothesis.
- Bayes theorem and basic results from information theory can be used to provide a rationale for this principle.

EM Summary

- In many practical learning tasks, some of the relevant instance variables may be unobservable The EM algorithm provides quite a general approach to learning in the presence of unobservable variables.
- It begins with an arbitrary initial hypothesis. It then repeatedly calculates the expected values of the hidden variables (assuming the current hypothesis is correct) and then recalculates the ML hypothesis (assuming the hidden variables have the expected values calculated by the first step).
- This procedure converges to a local ML hypothesis (i.e., maximum likelihood hypothesis) along with the estimated values for the hidden variables.