Data Mining

Lesson 7

Bayesian Learning

MSc in Computer Science

University of New York Tirana

Assoc. Prof. Dr. Marenglen Biba

Data Mining: Content

- Introduction to data mining and machine learning
- Inductive learning
- Decision trees
- Rule induction
- Instance-based learning
- Bayesian learning
- Neural networks
- Support vector machines
- Other machine learning models
- Engineering data mining tasks

Lesson Outline

- Bayesian Learning
- Bayes Optimal Classifier
- Naïve Bayes Classifier

Introduction

- Bayesian reasoning provides a probabilistic approach to learning and inference.

- It is based on the assumption that the quantities of interest are governed by probability distributions and that optimal decisions can be made by reasoning about these probabilities together with observed data.

Usefulness of BL

- Bayesian learning methods are relevant to our study of machine learning for two different reasons:

- First, Bayesian learning algorithms that calculate explicit probabilities for hypotheses, such as the naive Bayes classifier, are among the most practical approaches to certain types of learning problems.

- In this lesson we describe the naïve Bayes classifier.
  - Other classifiers: Bayesian Networks

Usefulness of BL

- The second reason that Bayesian methods are important to our study of machine learning is that they provide a useful perspective for understanding many learning algorithms that do not explicitly manipulate probabilities.

- For example, in this lesson we analyze algorithms such as the Find-S and Candidate-Elimination algorithms to determine conditions under which they output the most probable hypothesis given the training data.

Features of Bayesian learning

Features of Bayesian learning methods include:

1. Each observed training example can incrementally decrease or increase the estimated probability that a hypothesis is correct.
   - This provides a more flexible approach to learning than algorithms that completely eliminate a hypothesis if it is found to be inconsistent with any single example.

Features of Bayesian learning

2. Prior knowledge can be combined with observed data to determine the final probability of a hypothesis.

In Bayesian learning, prior knowledge is provided by asserting

(1) a prior probability for each candidate hypothesis, and

(2) a probability distribution over observed data for each possible hypothesis.

Features of Bayesian learning

3. Bayesian methods can accommodate hypotheses that make probabilistic predictions (e.g., hypotheses such as "this pneumonia patient has a 93% chance of complete recovery").
4. New instances can be classified by combining the predictions of multiple hypotheses, weighted by their probabilities.
5. Even in cases where Bayesian methods prove computationally intractable, they can provide a standard of optimal decision making against which other practical methods can be measured.

Difficulties in Bayesian Learning

- One practical difficulty in applying Bayesian methods is that they typically require initial knowledge of many probabilities.

- When these probabilities are not known in advance they are often estimated based on background knowledge, previously available data, and assumptions about the form of the underlying distributions.

Difficulties in Bayesian Learning

- A second practical difficulty is the significant computational cost required to determine the Bayes optimal hypothesis in the general case (linear in the number of candidate hypotheses).

- In certain specialized situations, this computational cost can be significantly reduced.

Bayes Theorem in Machine Learning

- In machine learning we are often interested in determining the best hypothesis from some space H, given the observed training data D.

- One way to specify what we mean by the best hypothesis is to say that we demand the *most probable hypothesis*, given the data D plus any initial knowledge about the prior probabilities of the various hypotheses in H.

- Bayes theorem provides a direct method for calculating such probabilities.

Prior probability

- We shall write *P(h)* to denote the initial probability that hypothesis h holds, before we have observed the training data.
- *P(h)* is often called the prior probability of h and may reflect any background knowledge we have about the chance that h is a correct hypothesis.
- If we have no such prior knowledge, then we might simply assign the same prior probability to each candidate hypothesis.
- Similarly, we will write *P(D)* to denote the prior probability that training data D will be observed (i.e., the probability of D given no knowledge about which hypothesis holds).

Prior probability

- P(D|h) denotes the probability of observing data D given some world in which hypothesis h holds.
- In machine learning problems, we are interested in the probability P(h|D) that h holds given the observed training data D.
- P(h|D) is called the *posterior probability* of *h*, because it reflects our confidence that h holds after we have seen the training data D.
- Notice the posterior probability P(h|D) reflects the influence of the training data D, in contrast to the prior probability P(h), which is independent of D.

Bayes' Theorem

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

- $P(h)$ = prior probability of hypothesis h
- $P(D)$ = prior probability of training data D
- $P(h|D)$ = probability of h given D
- $P(D|h)$ = probability of D given h

Maximum a posteriori (MAP) hypothesis

- In many learning scenarios, the learner considers some set of candidate hypotheses H and is interested in finding the most probable hypothesis $h \varepsilon H$ given the observed data D (or at least one of the maximally probable if there are several).

- Any such maximally probable hypothesis is called a *maximum a posteriori (MAP)* hypothesis.

- We can determine the MAP hypotheses by using Bayes theorem to calculate the posterior probability of each candidate hypothesis.

Choosing Hypotheses

Find most probable hypothesis given training data

Maximum a posteriori hypothesis h_{MAP}:

$$
\begin{aligned}
h_{MAP} &= \arg\max_{h \in H} P(h|D) \\
&= \arg\max_{h \in H} \frac{P(D|h)P(h)}{P(D)} \\
&= \arg\max_{h \in H} P(D|h)P(h)
\end{aligned}
$$

P(D) is a constant independent of h.

Assuming $P(h_i) = P(h_j)$ we can further simplify, and choose the *Maximum likelihood* (ML) hypothesis

$$h_{ML} = \arg\max_{h_i \in H} P(D|h_i)$$

Before the example

- Basic Probability Formulas

- *Product rule*: probability $P(A \wedge B)$ of a conjunction of two events A and B

$$P(A \wedge B) = P(A|B)P(B) = P(B|A)P(A)$$

- *Sum rule*: probability of a disjunction of two events A and B

$$P(A \vee B) = P(A) + P(B) - P(A \wedge B)$$

- *Bayes theorem*: the posterior probability $P(h|D)$ of h given D

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

- *Theorem of total probability*: if events $A_1, \ldots, A_n$ are mutually exclusive with $\sum_{i=1}^{n} P(A_i) = 1$, then

$$P(B) = \sum_{i=1}^{n} P(B|A_i)P(A_i)$$

TABLE 6.1
Summary of basic probability formulas.

Example of the Bayes rule

- Consider a medical diagnosis problem in which there are two alternative hypotheses:
  (1) that the patient has a particular form of cancer, and
  (2) that the patient does not.
- The available data is from a particular laboratory test with two possible outcomes: + (positive) and - (negative).
- We have prior knowledge that over the entire population of people only .008 have this disease.
- Furthermore, the lab test is only an imperfect indicator of the disease.
- The test returns a correct positive result in only 98% of the cases in which the disease is actually present and a correct negative result in only 97% of the cases in which the disease is not present. In other cases, the test returns the opposite result.

Computing probabilities

- The above situation can be summarized by the following probabilities:
  - P(cancer) = .008, P(¬cancer) = .992
  - P(+|cancer) = .98, P(-|cancer) = .02
  - P(+|¬cancer) = .03, P(-|¬cancer) = .97
- Suppose we now observe a new patient for whom the lab test returns a positive result.
- Should we diagnose the patient as having cancer or not?
- The maximum a posteriori hypothesis can be found using Equation (6.2):
  - P(+|cancer)P(cancer) = (.98) .008 = .0078
  - P(+|¬cancer)P(¬cancer) = (.03).992 = .0298
- Thus, h_{map} = ¬*cancer*.

Remarks

- As the example illustrates, the result of Bayesian inference depends strongly on the prior probabilities, which must be available in order to apply the method directly.

- Note also that in this example the hypotheses are not completely accepted or rejected, but rather become more or less probable as more data is observed.

Bayes Theorem And Concept Learning

Bayes Theorem And Concept Learning

- What is the relationship between Bayes theorem and the problem of concept learning?

- Since Bayes theorem provides a principled way to calculate the posterior probability of each hypothesis given the training data, we can use it as the basis for a straightforward learning algorithm that calculates the probability for each possible hypothesis, then outputs the most probable.

Bayes Theorem And Concept Learning

- This section considers a brute-force Bayesian concept learning algorithm, then compares it to concept learning algorithms we considered in Lesson 2.

- As we shall see, one interesting result of this comparison is that under certain conditions several algorithms discussed earlier output the same hypotheses as this brute-force Bayesian algorithm, despite the fact that they do not explicitly manipulate probabilities and are considerably more efficient.

Brute-Force Bayes Concept Learning

- Consider the concept learning problem first introduced in Lesson 2.
- In particular, assume the learner considers some finite hypothesis space H defined over the instance space X, in which the task is to learn some target concept c: X -> {0,1}.
- As usual, we assume that the learner is given some sequence of training examples $((x_1, d_1)... (x_m, d_m))$ where x_i is some instance from X and where d_i is the target value of x_i (i.e., $d_i = c(x_i)$).
- To simplify, we assume the sequence of instances $(x_1,\ldots,x_m)$ is held fixed, so that the training data D can be written simply as the sequence of target values $D = (d_1,\ldots,d_m)$.

Brute-Force MAP Hypothesis Learner

1. For each hypothesis h in H, calculate the posterior probability

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

2. Output the hypothesis h_{MAP} with the highest posterior probability

$$h_{MAP} = \underset{h \in H}{\operatorname{argmax}} P(h|D)$$

Algorithm's complexity

- This algorithm may require significant computation, because it applies Bayes theorem to each hypothesis in H to calculate P(h|D).

- While this may prove impractical for large hypothesis spaces, the algorithm is still of interest because it provides a standard against which we may judge the performance of other concept learning algorithms.

Learning problem for Brute Force MAP Learning

- In order to specify a learning problem for the Brute-Force MAP Learning algorithm we must specify what values are to be used for P(h) and for P(D|h) (as we shall see, P(D) will be determined once we choose the other two).
- We may choose the probability distributions P(h) and P(D|h) in any way we wish, to describe our prior knowledge about the learning task.
- Here let us choose them to be consistent with the following assumptions:
  1. The training data D is noise free (i.e., $d_i = c(x_i)$).
  2. The target concept c is contained in the hypothesis space H.
  3. We have no a priori reason to believe that any hypothesis is more probable than any other.

Relation to concept learning

- Given these assumptions, what values should we specify for P(h)?
- Given no prior knowledge that one hypothesis is more likely than another, it is reasonable to assign the same prior probability to every hypothesis h in H.
- Furthermore, because we assume the target concept is contained in H we should require that these prior probabilities sum to 1.
- Together these constraints imply that we should choose

$$P(h) = \frac{1}{|H|} \quad \text{for all } h \text{ in } H$$

Relation to concept learning

- What choice shall we make for P(D|h)?
- P(D | h) is the probability of observing the target values D = ($d_1 \ldots d_m$) for the fixed set of instances ($x_1 \ldots x_m$) given a world in which hypothesis h holds (i.e., given a world in which h is the correct description of the target concept c).
- Since we assume noise-free training data, the probability of observing classification d_i given h is just 1 if $d_i = h(x_i)$ and 0 if $di \neq h(xi)$. Therefore,

$$P(D|h) = \begin{cases} 1 \text{ if } d_i = h(x_i) \text{ for all } d_i \text{ in } D \\ 0 \text{ otherwise} \end{cases}$$

- In other words, the probability of data D given hypothesis h is 1 if D is consistent with h, and 0 otherwise.

Relation to concept learning

- Given these choices for P(h) and for P(D|h) we now have a fully-defined problem for the Brute-Force MAP Learning algorithm.
- Let us consider the first step of this algorithm, which uses Bayes theorem to compute the posterior probability P(h|D) of each hypothesis h given the observed training data D.
- Recalling Bayes theorem, we have:

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

- First consider the case where h is inconsistent with the training data D. Since the equation defines P(D|h) to be 0 when h is inconsistent with D, we have:

$$P(h|D) = \frac{0 \cdot P(h)}{P(D)} = 0 \text{ if } h \text{ is inconsistent with } D$$

- The posterior probability of a hypothesis inconsistent with D is zero.

Relation to concept learning

- Now consider the case where h is consistent with D.
- Since the equation defines P(D|h) to be 1 when h is consistent with D, we have:

$$P(h|D) = \frac{1 \cdot \frac{1}{|H|}}{P(D)}$$

$$= \frac{1 \cdot \frac{1}{|H|}}{\frac{|VS_{H,D}|}{|H|}}$$

$$= \frac{1}{|VS_{H,D}|} \text{ if } h \text{ is consistent with } D$$

- where $VS_{H,D}$ is the subset of hypotheses from H that are consistent with D (i.e., $VS_{H,D}$ is the version space of H with respect to D as defined in Lesson 2).
- Note that $P(D) = |\ VS_{H,D}\ | / |H|$.

Relation to concept learning

- To summarize, Bayes theorem implies that the posterior probability P(h|D) under our assumed P(h) and P(Dlh) is:

$$P(h|D) = \begin{cases} \frac{1}{|VS_{H,D}|} & \text{if } h \text{ is consistent with } D \\ 0 & \text{otherwise} \end{cases}$$

- where $|VS_{H,D}|$ is the number of hypotheses from H consistent with D.

Summary of the analysis

- The above analysis implies that under our choice for P(h) and P(D|h), every consistent hypothesis has posterior probability $(1 / |VS_{H,D}|)$ and every inconsistent hypothesis has posterior probability 0.

- Every consistent hypothesis is, therefore, a MAP hypothesis.

MAP Hypotheses and Consistent Learners

- The above analysis shows that in the given setting, every hypothesis consistent with D is a MAP hypothesis.

- This statement translates directly into an interesting statement about a general class of learners that we might call consistent learners.

- *We will say that a learning algorithm is a consistent learner provided it outputs a hypothesis that commits zero errors over the training examples.*

MAP Hypotheses and Consistent Learners

- Given the above analysis, we can conclude that every consistent learner outputs a MAP hypothesis, if we assume:
  - a uniform prior probability distribution over H (i.e., $P(h_i) = P(h_j)$) for all i, j)
  - deterministic, noise-free training data (i.e., P(D|h) = 1 if D and h are consistent, and 0 otherwise).

Find-S outputs a MAP hypothesis

- The concept learning algorithm Find-S searches the hypothesis space H from specific to general hypotheses, outputting a maximally specific consistent hypothesis (i.e., a maximally specific member of the version space).
- Because Find-S outputs a consistent hypothesis, we know that it will output a MAP hypothesis under the probability distributions P(h) and P(D|h) defined above.
- Of course Find-S does not explicitly manipulate probabilities at all — it simply outputs a maximally specific member of the version space.
- However, by identifying distributions for P(h) and P(D|h) under which its output hypotheses will be MAP hypotheses, we have a useful way of characterizing the behavior of Find-S.

Other remarks

- Are there other probability distributions for P(h) and P(D|h) under which Find-S outputs MAP hypotheses?
  - The answer is Yes.
- Because Find-S outputs a maximally specific hypothesis from the version space, its output hypothesis will be a MAP hypothesis relative to any prior probability distribution that favors more specific hypotheses.
- More precisely, suppose H_p is any probability distribution P(h) over H that assigns $P(h_1) > P(h_2)$ if h_1 is more specific than h_2.
- Then it can be shown that Find-S outputs a MAP hypothesis assuming the prior distribution H_p and the same distribution P(D|h) discussed above.

Characterizing the behavior of learning algorithms

- To summarize, the Bayesian framework allows one way to characterize the behavior of learning algorithms (e.g., Find-S), even when the learning algorithm does not explicitly manipulate probabilities.

- By identifying probability distributions P(h) and P(D|h) under which the algorithm outputs optimal (i.e., MAP) hypotheses, we can characterize the implicit assumptions under which this algorithm behaves optimally.

Summary of Bayesian perspective

- Using the Bayesian perspective to characterize learning algorithms in this way is similar in spirit to characterizing the inductive bias of the learner.
- Recall that in Lesson 2 we defined the inductive bias of a learning algorithm to be the set of assumptions B sufficient to *deductively* justify the inductive inference performed by the learner.
  - For example, we described the inductive bias of the Candidate-Elimination algorithm as the assumption that the target concept c is included in the hypothesis space H.
  - Furthermore, we showed there that the output of this learning algorithm follows deductively from its inputs plus this implicit inductive bias assumption.
- The above Bayesian interpretation provides an alternative way to characterize the assumptions implicit in learning algorithms.

Summary of Bayesian perspective

- Instead of modeling the inductive inference method by an equivalent deductive system, we model it by an equivalent probabilistic reasoning system based on Bayes theorem.

- And here the implicit assumptions that we attribute to the learner are assumptions of the form: "the prior probabilities over H are given by the distribution P(h), and the strength of data in rejecting or accepting a hypothesis is given by P(D|h).

- *The definitions of P(h) and P(D|h) given in this section characterize the implicit assumptions of the Candidate-Elimination and Find-S algorithms.*

How about noisy training data?

- The discussion throughout this section corresponds to a special case of Bayesian reasoning, because we considered the case where P(D|h) takes on values of only 0 and 1, reflecting the deterministic predictions of hypotheses and the assumption of noise-free training data.

- We can also model learning from noisy training data, by allowing P(D|h) to take on values other than 0 and 1, and by introducing into P(D|h) *additional assumptions* about the probability distributions that govern the noise.

Learning continuous-valued target functions

Learning continuous-valued target functions

- A problem faced by many learning approaches such as neural network learning, linear regression, and polynomial curve fitting is that of learning a continuous-valued target function .

- *A straightforward Bayesian analysis shows that under certain assumptions any learning algorithm that minimizes the squared error between the output hypothesis predictions and the training data will output a maximum likelihood hypothesis.*

- The significance of this result is that it provides a Bayesian justification (under certain assumptions) for many neural network and other curve fitting methods that attempt to minimize the sum of squared errors over the training data.

Learning settings with least-squared error

- Minimizing the sum of squared errors is a common approach in many neural network, curve fitting, and other approaches to approximating real-valued functions.

- Gradient descent methods seek the least-squared error hypothesis in neural network learning.

Maximum Likelihood Hypothesis For Predicting Probabilities

Maximum Likelihood Hypothesis For Predicting Probabilities

- Until now we have determined that the maximum likelihood hypothesis is the one that minimizes the sum of squared errors over the training examples.

- Now visit an analogous criterion for a second setting that is common in neural network learning: learning to predict probabilities.

Choosing the criterion

- What criterion should we optimize in order to find a maximum likelihood hypothesis for f' in this setting?

- The quantity that must be optimized in order to obtain the maximum likelihood hypothesis in our current problem setting, is analogous to our earlier result showing that:
  - *minimizing the sum of squared errors produces the maximum likelihood hypothesis in the earlier problem setting.*

Bayes Optimal Classifier

Bayes Optimal Classifier

- So far we have considered the question "what is the most probable hypothesis given the training data?"

- In fact, the question that is often of most significance is the closely related question:
  - *what is the most probable classification of the new instance given the training data?*

- Although it may seem that this second question can be answered by simply applying the MAP hypothesis to the new instance, in fact it is possible to do better.

Bayes Optimal Classifier

- To develop some intuitions consider a hypothesis space containing three hypotheses, h_1, h_2, and h_3.
- Suppose that the posterior probabilities of these hypotheses given the training data are 0.4, 0.3, and 0.3 respectively.
- Thus, h_1 is the MAP hypothesis.
- Suppose a new instance x is encountered, which is classified positive by h_1, but negative by h_2 and h_3.
- Taking all hypotheses into account, the probability that x is positive is 0.4 (the probability associated with h_1), and the probability that it is negative is therefore 0.6.
- The most probable classification (negative) in this case is different from the classification generated by the MAP hypothesis.

Bayes Optimal Classifier

- In general, the most probable classification of the new instance is obtained by combining the predictions of all hypotheses, weighted by their posterior probabilities.
- If the possible classification of the new example can take on any value v_j from some set V, then the probability $P(v_j|D)$ that the correct classification for the new instance is v_j, is just:

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

- The optimal classification of the new instance is the value V_j for which $P(v_j|D)$ is maximum.

Bayes optimal classification:

$$\underset{v_j \in V}{\operatorname{argmax}} \sum_{h_i \in H} P(v_j|h_i)P(h_i|D) \qquad (6.18)$$

Optimal learner

- Any system that classifies new instances according to Equation (6.18) is called a Bayes optimal classifier, or Bayes optimal learner.

- No other classification method using the same hypothesis space and same prior knowledge can outperform this method on average.

- This method maximizes the probability that the new instance is classified correctly, given the available data, hypothesis space, and prior probabilities over the hypotheses.

Example

To illustrate in terms of the above example, the set of possible classifications of the new instance is $V = \{\oplus, \ominus\}$, and

$$P(h_1|D) = .4, \; P(\ominus|h_1) = 0, \; P(\oplus|h_1) = 1$$
$$P(h_2|D) = .3, \; P(\ominus|h_2) = 1, \; P(\oplus|h_2) = 0$$
$$P(h_3|D) = .3, \; P(\ominus|h_3) = 1, \; P(\oplus|h_3) = 0$$

therefore

$$\sum_{h_i \in H} P(\oplus|h_i) P(h_i|D) = .4$$
$$\sum_{h_i \in H} P(\ominus|h_i) P(h_i|D) = .6$$

and

$$\underset{v_j \in \{\oplus, \ominus\}}{\text{argmax}} \sum_{h_i \in H} P(v_j|h_i) P(h_i|D) = \ominus$$

Concept learning and BOC

- BOC has an interesting implication for the concept learning problem described earlier.
- In particular, it implies that if the learner assumes a uniform prior over *H*, and if target concepts are in fact drawn from such a distribution when presented to the learner, then classifying the next instance according to a hypothesis drawn at random from the current version space (according to a uniform distribution), will have expected error at most twice that of the Bayes optimal classifier.
- Again, we have an example where a Bayesian analysis of a non-Bayesian algorithm yields insight into the performance of that algorithm.

Naive Bayes Classifier

Naive Bayes Classifier

- One highly practical Bayesian learning method is the naive Bayes learner, often called the naive Bayes classifier.

- In some domains its performance has been shown to be comparable to that of neural network and decision tree learning.

Naive Bayes Classifier

- The naive Bayes classifier applies to learning tasks where each instance x is described by a conjunction of attribute values and where the target function f(x) can take on any value from some finite set V.

- A set of training examples of the target function is provided, and a new instance is presented, described by the tuple of attribute values (a_1,a_2…a_n).

- The learner is asked to predict the target value, or classification, for this new instance.

Naive Bayes Classifier

The Bayesian approach to classifying the new instance is to assign the most probable target value, v_{MAP}, given the attribute values $\langle a_1, a_2 \ldots a_n \rangle$ that describe the instance.

$$v_{MAP} = \underset{v_j \in V}{\operatorname{argmax}} P(v_j | a_1, a_2 \ldots a_n)$$

We can use Bayes theorem to rewrite this expression as

$$v_{MAP} = \underset{v_j \in V}{\operatorname{argmax}} \frac{P(a_1, a_2 \ldots a_n | v_j) P(v_j)}{P(a_1, a_2 \ldots a_n)}$$

$$= \underset{v_j \in V}{\operatorname{argmax}} P(a_1, a_2 \ldots a_n | v_j) P(v_j) \qquad (6.19)$$

Naive Bayes Classifier

- Now we could attempt to estimate the two terms in Equation (6.19) based on the training data.

- It is easy to estimate each of the $P(v_j)$ simply by counting the frequency with which each target value v_j occurs in the training data.

- However, estimating the different $P(a_1,a_2...a_n|v_j)$ terms in this fashion is not feasible unless we have a very, very large set of training data.

Naive Bayes Classifier

- The naive Bayes classifier is based on the simplifying assumption that the attribute values are conditionally independent given the target value.
- In other words, the assumption is that given the target value of the instance, the probability of observing the conjunction $a_1,a_2...a_n$ is just the product of the probabilities for the individual attributes: $P(a_1,a_2...a_n|v_j) = \prod_i P(a_i|v_j)$.
- Substituting this into Equation (6.19), we have the approach used by the naive Bayes classifier.

Naive Bayes classifier:

$$v_{NB} = \underset{v_j \in V}{\operatorname{argmax}} P(v_j) \prod_i P(a_i|v_j) \qquad (6.20)$$

- where v_{NB} denotes the target value output by the naive Bayes classifier.

Summary of Naive Bayes Classifier

- To summarize, the naive Bayes learning method involves a learning step in which the various $P(v_j)$ and $P(a_i|v_j)$ terms are estimated, based on their frequencies over the training data.
- The set of these estimates corresponds to the learned hypothesis.
- This hypothesis is then used to classify each new instance by applying the rule in Equation (6.20).
- Whenever the naive Bayes assumption of conditional independence is satisfied, this naive Bayes classification v_{nb} is identical to the MAP classification.

Difference of NBC with other methods

- One interesting difference between the naive Bayes learning method and other learning methods we have considered is that there is no explicit search through the space of possible hypotheses (in this case, the space of possible hypotheses is the space of possible values that can be assigned to the various $P(v_j)$ and $P(c_i|v_j)$ terms).

- Instead, the hypothesis is formed without searching, simply by counting the frequency of various data combinations within the training examples.

An Illustrative Example of NBC

- Let us apply the naive Bayes classifier to a concept learning problem we considered during our discussion of decision tree learning: classifying days according to whether someone will play tennis.
- In Lesson 3 we provided a set of 14 training examples of the target concept PlayTennis, where each day is described by the attributes Outlook, Temperature, Humidity, and Wind.
- Here we use the naive Bayes classifier and the training data from this table to classify the following novel instance:
- (Outlook = sunny, Temperature = cool, Humidity = high, Wind = strong)

Training examples

Day	*Outlook*	*Temperature*	*Humidity*	*Wind*	*PlayTennis*
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

TABLE 3.2
Training examples for the target concept *PlayTennis*.

An Illustrative Example of NBC

- Our task is to predict the target value (yes or no) of the target concept PlayTennis for this new instance.
- Instantiating Equation (6.20) to fit the current task, the target value v_{NB} is given by:

$$
\begin{aligned}
v_{NB} &= \underset{v_j \in \{yes,no\}}{\operatorname{argmax}} P(v_j) \prod_i P(a_i|v_j) \\
&= \underset{v_j \in \{yes,no\}}{\operatorname{argmax}} P(v_j) \quad P(Outlook = sunny|v_j) P(Temperature = cool|v_j) \\
&\qquad \cdot P(Humidity = high|v_j) P(Wind = strong|v_j) \qquad (6.21)
\end{aligned}
$$

- Notice in the final expression that a_i has been instantiated using the particular attribute values of the new instance.

An Illustrative Example of NBC

- To calculate v_{NB} we now require 10 probabilities that can be estimated from the training data.
- First, the probabilities of the different target values can easily be estimated based on their frequencies over the 14 training examples:
  - P(PlayTennis = yes) = 9/14 = .64
  - P(PlayTennis = no) = 5/14 = .36
- Similarly, we can estimate the conditional probabilities. For example, those for Wind = strong are:
  - P(Wind = strong|PlayTennis = yes) = 3/9 = .33
  - P(Wind = strong|PlayTennis = no) = 3/5 = .60

An Illustrative Example of NBC

- Using these probability estimates and similar estimates for the remaining attribute values, we calculate v_{NB} according to Equation (6.21) as follows (now omitting attribute names for brevity)
  - 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
- Thus, the naive Bayes classifier assigns the target value PlayTennis = no to this new instance, based on the probability estimates learned from the training data.
- Furthermore, by normalizing the above quantities to sum to one we can calculate the conditional probability that the target value is no, given the observed attribute values.
- For the current example, this probability is .0206/.0206+.0053 = .795.

Most cited papers about naïve Bayes classifier

- http://www.cs.washington.edu/ai/naive.html
  - Domingos and Pazzani. On the Optimality of the Simple Bayesian Classifier under Zero-One Loss. In *Machine Learning*. 1997.

Bayesian learning in Weka

- Iris
- Colic
- Diabetes
- Glass
- Sonar
- Soybean
- Vote
- Zoo

Text Classification with NB

- Classifying text with Naïve Bayes

Readings for this part

- Machine Learning. T. Mitchell
  - Chapter 6