Overview

- Sample data set with frequencies and probabilities
- Classification based on Bayes rule
- Maximum a posterior and maximum likelihood
- Properties of Bayes classifiers
- Naive Bayes classifiers
- Parameter estimation, properties, example
- Dealing with sparse data
- Application: email classification


A Sample Data Set

Fictional data set that describes the weather conditions for playing some unspecified game.

<table>
<thead>
<tr>
<th>outlook</th>
<th>temp.</th>
<th>humidity</th>
<th>windy</th>
<th>play</th>
</tr>
</thead>
<tbody>
<tr>
<td>sunny</td>
<td>hot</td>
<td>high</td>
<td>false</td>
<td>no</td>
</tr>
<tr>
<td>sunny</td>
<td>hot</td>
<td>high</td>
<td>true</td>
<td>no</td>
</tr>
<tr>
<td>overcast</td>
<td>hot</td>
<td>high</td>
<td>false</td>
<td>yes</td>
</tr>
<tr>
<td>rainy</td>
<td>mild</td>
<td>high</td>
<td>false</td>
<td>yes</td>
</tr>
<tr>
<td>rainy</td>
<td>cool</td>
<td>normal</td>
<td>false</td>
<td>yes</td>
</tr>
<tr>
<td>overcast</td>
<td>cool</td>
<td>normal</td>
<td>true</td>
<td>no</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>outlook</th>
<th>temp.</th>
<th>humidity</th>
<th>windy</th>
<th>play</th>
</tr>
</thead>
<tbody>
<tr>
<td>sunny</td>
<td>mild</td>
<td>high</td>
<td>false</td>
<td>no</td>
</tr>
<tr>
<td>sunny</td>
<td>cool</td>
<td>normal</td>
<td>false</td>
<td>yes</td>
</tr>
<tr>
<td>overcast</td>
<td>mild</td>
<td>high</td>
<td>true</td>
<td>yes</td>
</tr>
<tr>
<td>overcast</td>
<td>cool</td>
<td>normal</td>
<td>false</td>
<td>yes</td>
</tr>
<tr>
<td>rainy</td>
<td>mild</td>
<td>high</td>
<td>true</td>
<td>no</td>
</tr>
</tbody>
</table>

Frequencies and Probabilities

<table>
<thead>
<tr>
<th>outlook</th>
<th>temperature</th>
<th>humidity</th>
<th>windy</th>
<th>play</th>
</tr>
</thead>
<tbody>
<tr>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>sunny</td>
<td>2</td>
<td>3</td>
<td>hot</td>
<td>2</td>
</tr>
<tr>
<td>sunny</td>
<td>2</td>
<td>2</td>
<td>high</td>
<td>3</td>
</tr>
<tr>
<td>overcast</td>
<td>4</td>
<td>0</td>
<td>mild</td>
<td>4</td>
</tr>
<tr>
<td>overcast</td>
<td>4</td>
<td>2</td>
<td>normal</td>
<td>6</td>
</tr>
<tr>
<td>rainy</td>
<td>3</td>
<td>2</td>
<td>cool</td>
<td>3</td>
</tr>
<tr>
<td>rainy</td>
<td>3</td>
<td>2</td>
<td>normal</td>
<td>6</td>
</tr>
<tr>
<td>overcast</td>
<td>4</td>
<td>0</td>
<td>mild</td>
<td>4</td>
</tr>
<tr>
<td>overcast</td>
<td>4</td>
<td>2</td>
<td>normal</td>
<td>6</td>
</tr>
<tr>
<td>sunny</td>
<td>2/9</td>
<td>3/5</td>
<td>hot</td>
<td>2/9</td>
</tr>
<tr>
<td>sunny</td>
<td>2/9</td>
<td>2/5</td>
<td>high</td>
<td>3/9</td>
</tr>
<tr>
<td>overcast</td>
<td>4/9</td>
<td>0/5</td>
<td>mild</td>
<td>4/9</td>
</tr>
<tr>
<td>overcast</td>
<td>4/9</td>
<td>2/5</td>
<td>normal</td>
<td>6/9</td>
</tr>
<tr>
<td>rainy</td>
<td>3/9</td>
<td>2/5</td>
<td>cool</td>
<td>3/9</td>
</tr>
<tr>
<td>rainy</td>
<td>3/9</td>
<td>2/5</td>
<td>normal</td>
<td>6/9</td>
</tr>
</tbody>
</table>

Naive Bayes Classifiers – p.1/22

Naive Bayes Classifiers – p.2/22

Naive Bayes Classifiers – p.3/22

Naive Bayes Classifiers – p.4/22
Classifying an Unseen Example

Now assume that we have to classify the following new instance:

<table>
<thead>
<tr>
<th>outlook</th>
<th>temp.</th>
<th>humidity</th>
<th>windy</th>
<th>play</th>
</tr>
</thead>
<tbody>
<tr>
<td>sunny</td>
<td>cool</td>
<td>high</td>
<td>true</td>
<td>?</td>
</tr>
</tbody>
</table>

Key idea: compute a probability for each class based on the probability distribution in the training data.

First take into account the the probability of each attribute. Treat all attributes equally important, i.e., multiply the probabilities:

\[
P(\text{yes}) = \frac{2}{9} \cdot \frac{3}{9} \cdot \frac{3}{9} \cdot \frac{3}{9} = 0.0082
\]

\[
P(\text{no}) = \frac{3}{5} \cdot \frac{1}{5} \cdot \frac{4}{5} \cdot \frac{3}{5} = 0.0577
\]

Now take into account the overall probability of a given class. Multiply it with the probabilities of the attributes:

\[
P(\text{yes}) = 0.0082 \cdot 9/14 = 0.0053
\]

\[
P(\text{no}) = 0.0577 \cdot 5/14 = 0.0206
\]

Now choose the class so that it maximizes this probability. This means that the new instance will be classified as no.

Bayes Rule

This procedure is based on Bayes Rule, which says: if you have a hypothesis \( h \) and data \( D \) which bears on the hypothesis, then:

\[
P(h|D) = \frac{P(D|h)P(h)}{P(D)}
\]

\( P(h) \): independent probability of \( h \): prior probability

\( P(D) \): independent probability of \( D \)

\( P(D|h) \): conditional probability of \( D \) given \( h \): likelihood

\( P(h|D) \): cond. probability of \( h \) given \( D \): posterior probability

Maximum A Posteriori

Based on Bayes Rule, we can compute the maximum a posteriori hypothesis for the data:

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

\[
= \arg \max_{h \in H} \frac{P(D|h)P(h)}{P(D)}
\]

\[
= \arg \max_{h \in H} P(D|h)P(h)
\]

\( H \): set of all hypotheses

Note that we can drop \( P(D) \) as the probability of the data is constant (and independent of the hypothesis).
Maximum Likelihood

Now assume that all hypotheses are equally probable a priori, i.e., \( P(h_i) = P(h_j) \) for all \( h_i, h_j \in H \).

This is called assuming a \textit{uniform prior}. It simplifies computing the posterior:

\[
    h_{ML} = \arg \max_{h \in H} P(D|h)
\]

This hypothesis is called the \textit{maximum likelihood hypothesis}.

Naive Bayes Classifier

Assumption: training set consists of instances described as \textit{conjunctions of attributes values}, target classification based on finite set of classes \( V \).

The task of the learner is to predict the correct class for a new instance \( \{a_1, a_2, \ldots, a_n\} \).

\textit{Key idea}: assign most probable class \( v_{MAP} \) using Bayes Rule.

\[
    v_{MAP} = \arg \max_{v_j \in V} P(v_j|a_1, a_2, \ldots, a_n)
\]

\[
    = \arg \max_{v_j \in V} P(a_1, a_2, \ldots, a_n|v_j)P(v_j)
\]

\[
    = \arg \max_{v_j \in V} P(a_1, a_2, \ldots, a_n|v_j)P(v_j)
\]

Properties of Bayes Classifiers

- \textbf{Incrementality}: with each training example, the prior and the likelihood can be updated dynamically: flexible and robust to errors.
- \textbf{Combines prior knowledge and observed data}: prior probability of a hypothesis multiplied with probability of the hypothesis given the training data.
- \textbf{Probabilistic hypotheses}: outputs not only a classification, but a probability distribution over all classes.
- \textbf{Meta-classification}: the outputs of several classifiers can be combined, e.g., by multiplying the probabilities that all classifiers predict for a given class.

Naive Bayes: Parameter Estimation

Estimating \( P(v_j) \) is simple: compute the relative frequency of each target class in the training set.

Estimating \( P(a_1, a_2, \ldots, a_n|v_j) \) is difficult: typically not enough instances for each attribute combination in the training set: \textit{sparse data problem}.

Independence assumption: attribute values are conditionally independent given the target value: \textit{naive} Bayes.

\[
    P(a_1, a_2, \ldots, a_n|v_j) = \prod_i P(a_i|v_j)
\]

Hence we get the following classifier:

\[
    v_{NB} = \arg \max_{v_j \in V} P(v_j) \prod_i P(a_i|v_j)
\]
Naive Bayes: Properties

- Estimating $P(a_i|v_j)$ instead of $P(a_1,a_2,\ldots,a_n|v_j)$ greatly reduces the number of parameters (and data sparseness).
- The learning step in Naive Bayes consists of estimating $P(a_i|v_j)$ and $P(v_j)$ based on the frequencies in the training data.
- There is no explicit search during training (as opposed to decision trees).
- An unseen instance is classified by computing the class that maximizes the posterior.
- When conditional independence is satisfied, Naive Bayes corresponds to MAP classification.

Naive Bayes: Example

Apply Naive Bayes to the weather training data. The hypothesis space is $V = \{\text{yes, no}\}$. Classify the following new instance:

<table>
<thead>
<tr>
<th>outlook</th>
<th>temp.</th>
<th>humidity</th>
<th>windy</th>
<th>play</th>
</tr>
</thead>
<tbody>
<tr>
<td>sunny</td>
<td>cool</td>
<td>high</td>
<td>true</td>
<td>?</td>
</tr>
</tbody>
</table>

$v_{NB} = \arg \max_{v_j \in \{\text{yes, no}\}} P(v_j) \prod_i P(a_i|v_j)$

$= \arg \max_{v_j \in \{\text{yes, no}\}} P(v_j)P(\text{outlook = sunny}|v_j)P(\text{temp = cool}|v_j)$

$P(\text{humidity = high}|v_j)P(\text{windy = true}|v_j)$

Compute priors:

$P(\text{play = yes}) = 9/14 \quad P(\text{play = no}) = 5/14$

Naive Bayes: Sparse Data

Conditional probabilities can be estimated directly as relative frequencies:

$P(a_i|v_j) = \frac{n_c}{n}$

where $n$ is the total number of training instances with class $v_j$, and $n_c$ is the number of instances with attribute $a_i$ and class $v_j$.

Problem: this provides a poor estimate if $n_c$ is very small.

Extreme case: if $n_c = 0$, then the whole posterior will be zero.

Compute conditionals (examples):

$P(\text{windy = true}|\text{play = yes}) = 3/9$

$P(\text{windy = true}|\text{play = no}) = 3/5$

Then compute the best class:

$P(\text{yes})P(\text{sunny}|\text{yes})P(\text{cool}|\text{yes})P(\text{high}|\text{yes})P(\text{true}|\text{yes})$

$= 9/14 \cdot 2/9 \cdot 3/9 \cdot 3/9 \cdot 3/9 = 0.0053$

$P(\text{no})P(\text{sunny}|\text{no})P(\text{cool}|\text{no})P(\text{high}|\text{no})P(\text{true}|\text{no})$

$= 5/14 \cdot 3/5 \cdot 1/5 \cdot 4/5 \cdot 3/5 = 0.0206$

Now classify the unseen instance:

$v_{NB} = \arg \max_{v_j \in \{\text{yes, no}\}} P(v_j)P(\text{sunny}|v_j)P(\text{cool}|v_j)P(\text{high}|v_j)P(\text{true}|v_j)$

$= \text{no}$
Naive Bayes: Sparse Data

Solution: use the *m-estimate* of probabilities:

\[ P(a_i|v_j) = \frac{n_c + mp}{n + m} \]

- \( p \): prior estimate of the probability
- \( m \): equivalent sample size (constant)

In the absence of other information, assume a **uniform prior**:

\[ p = \frac{1}{k} \]

where \( k \) is the number of values that the attribute \( a_i \) can take.

Application: Email Classification

**Training data**: a corpus of email messages, each message annotated as spam or no spam.

**Task**: classify new email messages as spam/no spam.

To use a naive Bayes classifier for this task, we have to first find an **attribute representation** of the data.

Treat each text position as an attribute, with as its value the word at this position. Example: email starts: *get rich*.

The naive Bayes classifier is then:

\[ v_{NB} = \arg \max_{v_j \in \{\text{spam}, \text{nospam}\}} P(v_j) \prod_i P(a_i|v_j) \]

\[ = \arg \max_{v_j \in \{\text{spam}, \text{nospam}\}} P(v_j)P(a_1 = \text{get}|v_j)P(a_2 = \text{rich}|v_j) \]

Application: Email Classification

Using naive Bayes means we assume that **words are independent of each other**. Clearly incorrect, but doesn’t hurt a lot for our task.

The classifier uses \( P(a_i = w_k|v_j) \), i.e., the probability that the \( i \)-th word in the email is the \( k \)-word in our vocabulary, given the email has been classified as \( v_j \).

Simplify by assuming that **position is irrelevant**: estimate \( P(w_k|v_j) \), i.e., the probability that word \( w_k \) occurs in the email, given class \( v_j \).

Create a **vocabulary**: make a list of all words in the training corpus, discard words with very high or very low frequency.

Training: estimate priors:

\[ P(v_j) = \frac{n}{N} \]

Estimate likelihoods using the *m-estimate*:

\[ P(w_k|v_j) = \frac{n_k + 1}{n + |\text{Vocabulary}|} \]

\( N \): total number of words in all emails

\( n \): number of words in emails with class \( v_j \)

\( n_k \): number of times word \( w_k \) occurs in emails with class \( v_j \)

\( |\text{Vocabulary}| \): size of the vocabulary

Testing: to classify a new email, assign it the class with the highest posterior probability. Ignore unknown words.
Bayes classifier combines prior knowledge with observed data: assigns a posterior probability to a class based on its prior probability and its likelihood given the training data.

- Computes the maximum a posterior (MAP) hypothesis or the maximum likelihood (ML) hypothesis.

- Naive Bayes classifier assumes conditional independence between attributes and assigns the MAP class to new instances.

- Likelihoods can be estimated based on frequencies. Problem: sparse data. Solution: using the $m$-estimate (adding a constant).

References
