Classification Outline

- Classification Problem Overview
- Algorithm performance measures
- Classification Techniques
  - Decision Trees
    - ID3, MDL Pruning, CART, Gini
  - Naïve Bayes
- Ensemble techniques (Bagging, Boosting, Stacking)
Classification Problem

- Given a database $D = \{t_1,t_2,\ldots,t_n\}$ and a set of classes $C = \{C_1,\ldots,C_m\}$, the *Classification Problem* is to define a mapping $f: D \rightarrow C$ where each $t_i$ is assigned to one class.
- Actually divides $D$ into *equivalence classes*.
- Our aim is to find this function that minimizes generalization error.
Classification Examples

• Teachers classify students’ by grades.
• Is a session/transaction malignant or benign
• Identify individuals with credit risks.
• Allocate a web session as being a buyer or browser
• Is a group of transactions fraudulent
Confusion Matrix Example

Using height data example with Output1 correct and Output2 actual assignment

<table>
<thead>
<tr>
<th>Actual Membership</th>
<th>Assignment Membership</th>
</tr>
</thead>
<tbody>
<tr>
<td>Short</td>
<td>Short</td>
</tr>
<tr>
<td>Short</td>
<td>0</td>
</tr>
<tr>
<td>Medium</td>
<td>0</td>
</tr>
<tr>
<td>Tall</td>
<td>0</td>
</tr>
</tbody>
</table>
Classification
Functional Overview – Learning Stage

Examples

- 
+ + - 
+ - + 
- 

Age, … income, {Default | NoDefault}
Case A. 30, …, $110K, Default
Case B. 50, …, $110K, NoDefault
Case C. 45, …, $90K, NoDefault
Case A. 32, …, $105K, Default
Case B. 49, …, $82K, NoDefault
Case C. 29, …, $50K, NoDefault
Classification
Functional Overview – Application Stage

Unlabeled Examples

Age, … income, \{Default | NoDefault\}
Case zx. 29, …, $113K, ?
Case zy. 42, …, $81K, ?
Case zz. 41, …, $92K, ?

Predictions

zx, Default
zy. NoDefault
zz. NoDefault
Classification Terminology

- Attribute, instances
- Training set, test set
- Training set accuracy
- Test set accuracy
- X-fold validation
- Confusion Matrix
- False positive
- False negative
- Cost sensitive classification
- Independent and dependent attributes
Decision Tree For Playing Tennis

**Root Node**

**Branch**

**Internal Node**

**Leaf Node**

Disjunction of conjunctions
How and When To Use Decision Trees

• How
  – Classification
    • Gain insights into why customers default on loans
  – Prediction
    • Screen customer loan applications
  – Feature/column/attribute selection
  – To explain other learning techniques

• When
  – Instances are attribute value pairs
  – Discrete target function
  – Noisy training data or missing values
Classification Using Decision Trees

• *Partitioning based:* Divide search space into rectangular regions.
• Tuple placed into class based on the region within which it falls.
• DT approaches differ in how the tree is built: *DT Induction*
  • Internal nodes associated with attribute and arcs with values for that attribute.
• Algorithms: ID3, C4.5, CART
Top-Down Tree Induction

Main loop:
1. \( A \leftarrow \) the “best” decision attribute for next \( node \)
2. Assign \( A \) as decision attribute for \( node \)
3. For each value of \( A \), create new descendant of \( node \)
4. Sort training examples to leaf nodes
5. If training examples perfectly classified, Then STOP, Else iterate over new leaf nodes

Which attribute is best?

```
[29+, 35-] A1=?
  t               t
  [21+, 5-]  [8+, 30-]
```

```
[29+, 35-] A2=?
  t               t
  [18+, 33-]  [11+, 2-]
```
Decision Tree

Given:

- \( D = \{t_1, \ldots, t_n\} \) where \( t_i = \langle t_{i1}, \ldots, t_{ih} \rangle \)
- Database schema contains \( \{A_1, A_2, \ldots, A_h\} \)
- Classes \( C = \{C_1, \ldots, C_m\} \)

**Decision or Classification Tree** is a tree associated with \( D \) such that

- Each internal node is labeled with attribute, \( A_i \)
- Each arc is labeled with predicate which can be applied to attribute at parent
- Each leaf node is labeled with a class, \( C_j \)
Which Column and Split Point?

• Multitude of techniques:
  • Entropy/Information gain
  • Chi square test (CHAID)
    • Test of independence
  • GINI index
Information Gain

\[ \text{Gain}(S, A) = \text{expected reduction in entropy due to sorting on } A \]

\[ \text{Gain}(S, A) \equiv \text{Entropy}(S) - \sum_{v \in \text{Values}(A)} \frac{|S_v|}{|S|} \text{Entropy}(S_v) \]
Entropy

$Entropy(S) = \text{expected number of bits needed to encode class (⊕ or ⊙) of randomly drawn member of } S \text{ (under the optimal, shortest-length code)}$

Why?

Information theory: optimal length code assigns $-\log_2 p$ bits to message having probability $p$.

So, expected number of bits to encode ⊕ or ⊙ of random member of $S$:

$p_\oplus(-\log_2 p_\oplus) + p_\ominus(-\log_2 p_\ominus)$

$Entropy(S) \equiv -p_\oplus \log_2 p_\oplus - p_\ominus \log_2 p_\ominus$
Information/Entropy

- Given probabilities $p_1, p_2, \ldots, p_s$ whose sum is 1, *Entropy* is defined as:

  $$H(p_1, p_2, \ldots, p_s) = \sum_{i=1}^{s} (p_i \log(1/p_i))$$

- Entropy measures the amount of randomness or surprise or uncertainty.
- Goal in classification
  - no surprise
  - entropy $= 0$
Entropy

\[ \log \left( \frac{1}{p} \right) \]

\[ H(p, 1-p) \]
## Data Set

<table>
<thead>
<tr>
<th>Day</th>
<th>Outlook</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>PlayTennis</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>D2</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>D3</td>
<td>Overcast</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D4</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D5</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D6</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>D7</td>
<td>Overcast</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D8</td>
<td>Sunny</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>D9</td>
<td>Sunny</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D10</td>
<td>Rain</td>
<td>Mild</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D11</td>
<td>Sunny</td>
<td>Mild</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D12</td>
<td>Overcast</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D13</td>
<td>Overcast</td>
<td>Hot</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D14</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
</tbody>
</table>
Choosing the Next Attribute - 1

Which attribute is the best classifier?

\[ \text{Gain}(S, \text{ Humidity}) = 0.940 - (7/14) \times 0.985 - (7/14) \times 0.592 = 0.151 \]

\[ \text{Gain}(S, \text{ Wind}) = 0.940 - (8/14) \times 0.811 - (6/14) \times 1.0 = 0.048 \]
Choosing the Next Attribute - 2

Which attribute should be tested here?

\[ S_{\text{Sunny}} = \{D1,D2,D8,D9,D11\} \]

Gain (\(S_{\text{Sunny}}, \text{Humidity}\)) = \(.970 - (3/5)0.0 - (2/5)0.0 = .970\)

Gain (\(S_{\text{Sunny}}, \text{Temperature}\)) = \(.970 - (2/5)0.0 - (2/5)1.0 - (1/5)0.0 = .570\)

Gain (\(S_{\text{Sunny}}, \text{Wind}\)) = \(.970 - (2/5)1.0 - (3/5) .918 = .019\)
Occam’s Razor

• 14th Century Franciscan friar; William of Occam.
• The principle states that "Entities should not be multiplied unnecessarily."
• People often reinvented Occam's Razor
  – Newton - "We are to admit no more causes of natural things than such as are both true and sufficient to explain their appearances."
• To most scientist the razor is:
  – "when you have two competing theories which make exactly the same predictions, the one that is simpler is the better."
Comparing DTs

Balanced

Gender
  \(=F\)
  \(=M\)

Height
  \(<1.3m\)
  \(>1.8m\)
  \(<1.5m\)
  \(>2m\)

Short
  Medium
  Tall

M

F

1.3 1.5 1.8 2.0

Deep

Height
  \(<1.3m\)
  \(>2m\)

Short
Gender
Tall

\(=F\)
\(=M\)

Height
  \(\leq1.8m\)
  \(>1.8m\)
  \(<1.5m\)
  \(\geq1.5m\)

Medium
Tall
Short
Medium

M

F

1.3 1.5 1.8 2.0
DT Issues

• Choosing Splitting Attributes
• Ordering of Splitting Attributes
• Splits
• Tree Structure
• Stopping Criteria
• Pruning
## Data Set

<table>
<thead>
<tr>
<th>Day</th>
<th>Outlook</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>PlayTennis</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>D2</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>D3</td>
<td>Overcast</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D4</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D5</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D6</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>D7</td>
<td>Overcast</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D8</td>
<td>Sunny</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>D9</td>
<td>Sunny</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D10</td>
<td>Rain</td>
<td>Mild</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D11</td>
<td>Sunny</td>
<td>Mild</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D12</td>
<td>Overcast</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D13</td>
<td>Overcast</td>
<td>Hot</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D14</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
</tbody>
</table>
Choosing the Next Attribute - 1

Which attribute is the best classifier?

\[
\text{Gain}(S, \text{Humidity}) = 0.940 - (7/14) \times 0.985 - (7/14) \times 0.592 = 0.151
\]

\[
\text{Gain}(S, \text{Wind}) = 0.940 - (8/14) \times 0.811 - (6/14) \times 1.0 = 0.048
\]
Confidences

outlook = sunny
| humidity = high: no (3.0)
| humidity = normal: yes (2.0)
outlook = overcast: yes (4.0)
outlook = rainy
| windy = TRUE: no (2.0)
| windy = FALSE: yes (3.0)

Number of Leaves : 5
Size of the tree: 8

How many possible combos
Test on ...
D15, Sunny, Mild, Low, Weak, ?
D16, Sunny, Mild, Low, Strong, ?
D17, Overcast, Hot, Normal, Strong, ?
Operating Characteristic Curve

Lift, gains and response curves,
Lift Curve
Stopping Criteria

- What type of tree will perfectly classify the training data (i.e., 100% training set accuracy)?
- Is this a bad thing? Why?
- What does this tell you about the relationship between the dependent and independent attributes?
- Apriori stopping criteria, when:
  - A certain tree depth is reached
  - Number of records at a node goes below some threshold.
  - All potential splits are insignificant
- Want the tree to be simple, but not too simple
How Do We Know When We’ve Overfitted The Training Data?

Consider error of hypothesis $h$ over

- training data: $\text{error}_{\text{train}}(h)$
- entire distribution $\mathcal{D}$ of data: $\text{error}_{\mathcal{D}}(h)$

Hypothesis $h \in H$ overfits training data if there is an alternative hypothesis $h' \in H$ such that

$$\text{error}_{\text{train}}(h) < \text{error}_{\text{train}}(h')$$

and

$$\text{error}_{\mathcal{D}}(h) > \text{error}_{\mathcal{D}}(h')$$

Is there any other way?
Training Set Error Should Approximately Equal Test Set Error

Optimum model complexity?
Trimming/Pruning Trees

• Stopping criterion can be somewhat arbitrary.

• Automatic pruning of trees
  – Ask the data, “How far should we split the data”.
  – Two general approaches:
    • Use part of the training set as a validation set
    • Use entire training set (usually an MDL approach).
Using Pruning To Prevent Overfitting

How can we avoid overfitting?

- stop growing when data split not statistically significant
- grow full tree, then post-prune

How to select “best” tree:

- Measure performance over training data
- Measure performance over separate validation data set
- MDL: minimize
  \[
  \text{size(tree)} + \text{size(misclassifications(tree))}
  \]
Reduced Error Pruning

Split data into *training* and *validation* set

Do until further pruning is harmful:

1. Evaluate impact on *validation* set of pruning each possible node (plus those below it)

2. Greedily remove the one that most improves *validation* set accuracy

- produces smallest version of most accurate subtree
- What if data is limited?
Reduced Error Pruning
Results of Reduced Error Pruning

Consider the use of learning a tree is to make prediction. What is the fundamental assumption that this learning algorithm is making?
Rule Post-Pruning

1. Convert tree to equivalent set of rules
2. Prune each rule independently of others
3. Sort final rules into desired sequence for use

Perhaps most frequently used method (e.g., C4.5)
## Data Set

<table>
<thead>
<tr>
<th>Day</th>
<th>Outlook</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>PlayTennis</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>D2</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>D3</td>
<td>Overcast</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D4</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D5</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D6</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>D7</td>
<td>Overcast</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D8</td>
<td>Sunny</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>D9</td>
<td>Sunny</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D10</td>
<td>Rain</td>
<td>Mild</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D11</td>
<td>Sunny</td>
<td>Mild</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D12</td>
<td>Overcast</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D13</td>
<td>Overcast</td>
<td>Hot</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D14</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
</tbody>
</table>
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 \( h \)
Interpretation of a Probability

• Frequentist
  – Relative frequency of an event occurring
  – What about rare events?

• Degree of belief
  – Our belief that the event will occur
About the Hypothesis Space $P(h)$

• Priors
  – Each $h_i$ should be *Mutually exclusive*
  – Together the hypotheses must be *Totally exhaustive*
  – $\Sigma P(h_i)=1$
  – Priors encode knowledge before we see the data
About the Data $P(D)$ and $P(D|H)$

- **Data, $P(D)$**
  - Data is considered to be a sample of all available data.
  - $P(D)$, probability the data will be observed given no knowledge of the hypothesis.
  - Constant for fixed data and if comparing hypotheses, can be ignored

- **Likelihood, $P(D|h)$**
  - Probability a hypothesis generated the observed data or probability of observing data given the hypothesis is true.
  - If the $n$ instances are independent then
    - $P(D|h) = P(D_1|h) \cdot P(D_2|h) \cdots P(D_n|h)$
  - Often use the Loglikelihood ($P(D|h)$).
Bayesian Posterior

• $P(h|D)$ is the posterior probability of the hypothesis (given the data).
• Usual aim of Bayesian learning is to find the MAP estimate
  – Most probable model in the model space
  – May be many highly probable models
A Simple Example

Does patient have cancer or not?

A patient takes a lab test and the result comes back positive. 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. Furthermore, .008 of the entire population have this cancer.

\[
P(\text{cancer}) = \quad P(\neg \text{cancer}) = \\
P(\text{+} | \text{cancer}) = \quad P(\text{+} | \neg \text{cancer}) = \\
P(\text{-} | \text{cancer}) = \quad P(\text{-} | \neg \text{cancer}) =
\]
Choosing the Hypothesis

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

Generally want the most probable hypothesis given the training data. The **maximum a posteriori** hypothesis \( h_{MAP} \):

\[
\begin{align*}
    h_{MAP} &= \underset{h \in H}{\arg \max} P(h|D) \\
    &= \underset{h \in H}{\arg \max} \frac{P(D|h)P(h)}{P(D)} \\
    &= \underset{h \in H}{\arg \max} P(D|h)P(h)
\end{align*}
\]

Assume all hypothesis have equal probability

\[
    h_{ML} = \underset{h \in H}{\arg \max} P(D|h_i)
\]
Basic Rules of Probability

- **Product Rule**: probability $P(A \land B)$ of a conjunction of two events $A$ and $B$:
  \[ P(A \land B) = P(A|B)P(B) = P(B|A)P(A) \]

- **Sum Rule**: probability of a disjunction of two events $A$ and $B$:
  \[ P(A \lor B) = P(A) + P(B) - P(A \land B) \]

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

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

Along with decision trees, neural networks, nearest nbr, one of the most practical learning methods.

When to use

- Moderate or large training set available
- Attributes that describe instances are conditionally independent given classification

Successful applications:

- Diagnosis
- Classifying text documents

Read example on text classification in book.
Definition of NBC

Assume target function $f : X \to V$, where each instance $x$ described by attributes $\langle a_1, a_2 \ldots a_n \rangle$. Most probable value of $f(x)$ is:

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

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

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

Naive Bayes assumption:

$$P(a_1, a_2 \ldots a_n|v_j) = \prod_{i} P(a_i|v_j)$$

which gives

**Naive Bayes classifier**: $v_{NB} = \arg\max_{v_j \in V} P(v_j) \prod_{i} P(a_i|v_j)$
Coding the Algorithm

Naive_Bayes_Learn(\textit{examples})

For each target value \( v_j \)

\[ \hat{P}(v_j) \leftarrow \text{estimate } P(v_j) \]

For each attribute value \( a_i \) of each attribute \( a \)

\[ \hat{P}(a_i|v_j) \leftarrow \text{estimate } P(a_i|v_j) \]

Classify_New_Instance(\( x \))

\[ v_{NB} = \arg\max_{v_j \in V} \hat{P}(v_j) \prod_{a_i \in x} \hat{P}(a_i|v_j) \]
## Data Set

<table>
<thead>
<tr>
<th>Day</th>
<th>Outlook</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>PlayTennis</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>D2</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>D3</td>
<td>Overcast</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D4</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D5</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D6</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>D7</td>
<td>Overcast</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D8</td>
<td>Sunny</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>D9</td>
<td>Sunny</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D10</td>
<td>Rain</td>
<td>Mild</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D11</td>
<td>Sunny</td>
<td>Mild</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D12</td>
<td>Overcast</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D13</td>
<td>Overcast</td>
<td>Hot</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D14</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
</tbody>
</table>
**NB: Simple Example**

Consider *PlayTennis* again, and new instance

\[
\{\text{Outlk} = \text{sun}, \text{Temp} = \text{cool}, \text{Humid} = \text{high}, \text{Wind} = \text{strong}\}
\]

Want to compute:

\[
v_{NB} = \arg\max_{v_j \in V} P(v_j) \prod_{i} P(a_i | v_j)
\]

\[
P(y) P(\text{sun} | y) P(\text{cool} | y) P(\text{high} | y) P(\text{strong} | y) = .005
\]

\[
P(n) P(\text{sun} | n) P(\text{cool} | n) P(\text{high} | n) P(\text{strong} | n) = .021
\]

\[\to v_{NB} = n\]
NB: Subtleties

Conditional independence assumption is often violated

\[ P(a_1, a_2 \ldots a_n|v_j) = \prod_{i} P(a_i|v_j) \]

- ...but it works surprisingly well anyway. Note don’t need estimated posteriors \( \hat{P}(v_j|x) \) correct; need only that

\[
\arg\max_{v_j \in V} \hat{P}(v_j) \prod_{i} \hat{P}(a_i|v_j) = \arg\max_{v_j \in V} P(v_j)P(a_1 \ldots , a_n|v_j)
\]

- see [Domingos & Pazzani, 1996] for analysis
- Naive Bayes posteriors often unrealistically close to 1 or 0
2. what if none of the training instances with target value \( v_j \) have attribute value \( a_i \)? Then

\[
\hat{P}(a_i|v_j) = 0, \text{ and...}
\]

\[
\hat{P}(v_j) \prod_i \hat{P}(a_i|v_j) = 0
\]

Typical solution is Bayesian estimate for \( \hat{P}(a_i|v_j) \)

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

where

- \( n \) is number of training examples for which \( v = v_j \),
- \( n_c \) number of examples for which \( v = v_j \) and \( a = a_i \)
- \( p \) is prior estimate for \( \hat{P}(a_i|v_j) \)
- \( m \) is weight given to prior (i.e. number of “virtual” examples)
Performance Issues

• Very fast to compute (just need to count)
• What type of probabilities are used to make predictions, what are two differences to decision trees?
• Assignment clarifications

5 Conclusions

Despite its unrealistic independence assumption, the naive Bayes classifier is surprisingly effective in practice since its classification decision may often be correct even if its probability estimates are inaccurate. Although some optimality conditions of naive Bayes have been already identified in the past [2], a deeper understanding of data characteristics that affect the performance of naive Bayes is still required.
Adding Data Removes Uncertainty

Figure 1. Posterior Distribution for a Univariate Mixture Gaussian Model with 8 Labeled Examples.

Figure 2. Posterior Distribution for a Univariate Mixture Gaussian Model with 4 Labeled and 4 Unlabeled Examples.
Decision Regions for NB
Text Document Classification

Why?

- Learn which news articles are of interest
- Learn to classify web pages by topic

Naive Bayes is among most effective algorithms

What attributes shall we use to represent text documents??
Target concept *Interesting?:* Document $\rightarrow \{+, -, \}$

1. Represent each document by vector of words
   - one attribute per word position in document

2. Learning: Use training examples to estimate
   - $P(+)$
   - $P(-)$
   - $P(doc | +)$
   - $P(doc | -)$

Naive Bayes conditional independence assumption

$$P(doc | v_j) = \prod_{i=1}^{\text{length}(doc)} P(a_i = w_k | v_j)$$

where $P(a_i = w_k | v_j)$ is probability that word in position $i$ is $w_k$, given $v_j$

one more assumption: $P(a_i = w_k | v_j) = P(a_m = w_k | v_j), \forall i, m$
LEARN NAIVE BAYES TEXT(Examples, V)

1. collect all words and other tokens that occur in Examples
   - Vocabulary ← all distinct words and other tokens in 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
     - \( docs_j \) ← subset of Examples for which the target value is \( v_j \)
     - \( P(v_j) \) ← \( \frac{|docs_j|}{|Examples|} \)
     - \( Text_j \) ← a single document created by concatenating all members of \( docs_j \)
     - \( n \) ← total number of words in \( Text_j \) (counting duplicate words multiple times)
     - for each word \( w_k \) in Vocabulary
       * \( n_k \) ← number of times word \( w_k \) occurs in \( Text_j \)
       * \( P(w_k|v_j) \) ← \( \frac{n_{k}+1}{n+|Vocabulary|} \)
\textsc{Classify} \_\textsc{naive} \_\textsc{Bayes} \_\textsc{text}(Doc)

- \( \textit{positions} \leftarrow \text{all word positions in } Doc \text{ that contain tokens found in } Vocab\text{ulary} \)
- Return \( v_{NB} \), where

\[
v_{NB} = \arg \max_{v_j \in V} P(v_j) \prod_{i \in \text{positions}} P(a_i|v_j)
\]

Given 1000 training documents from each group
Learn to classify new documents according to which newsgroup it came from

\begin{align*}
\text{comp.graphics} & \quad \text{misc.forsale} \\
\text{comp.os.windows.misc} & \quad \text{rec.autos} \\
\text{comp.sys.ibm.pc.hardware} & \quad \text{rec.motorcycles} \\
\text{comp.sys.mac.hardware} & \quad \text{rec.sport.baseball} \\
\text{comp.windows.x} & \quad \text{rec.sport.hockey}
\end{align*}
Accuracy

Article from rec.sport.hockey

Path: cantaloupe.srv.cs.cmu.edu!das-news.harvard
From: xxx@yyy.zzz.edu (John Doe)
Subject: Re: This year’s biggest and worst (opin
Date: 5 Apr 93 09:53:39 GMT

I can only comment on the Kings, but the most obvious candidate for pleasant surprise is Alex Zhitnik. He came highly touted as a defensive defenseman, but he’s clearly much more than that. Great skater and hard shot (though wish he were more accurate). In fact, he pretty much allowed the Kings to trade away that huge defensive liability Paul Coffey. Kelly Hrudey is only the biggest disappointment if you thought he was any good to begin with. But, at best, he’s only a mediocre goaltender. A better choice would be Tomas Sandstrom, though not through any fault of his own, but because some thugs in Toronto decid