classifier II

Arend Hintze
how good is my classifier?

• confusion matrix:

<table>
<thead>
<tr>
<th>ACTUAL CLASS</th>
<th>PREDICTED CLASS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class = +</td>
<td>a</td>
</tr>
<tr>
<td>Class = -</td>
<td>c</td>
</tr>
</tbody>
</table>

Accuracy = \(rac{\text{#Correct Predictions}}{\text{# Instances Predicted}} = \frac{a + d}{a + b + c + d}\)

Error rate = 1 – Accuracy = \(\frac{b + c}{a + b + c + d}\)
over and under fitting

**Underfitting**: when model is too simple, both training and test errors are large

**Overfitting**: when model is too complex, training error is small but test error is large
what if I only have trainings data?

- Divide training data into two parts:
  - Training
  - Test / Validation set
- Drawback: Less data available for training
example:

Training errors:
Tree A: 10%
Tree B: 20%

Validation errors:
Tree A: 50%
Tree B: 30%

So, tree B is preferred over A
Rule-Based Classifier

- use a set of "if ... then ..." rules (called the rule set)
- Data tuple (X, Y) where X = attribute, Y = class
example:

<table>
<thead>
<tr>
<th>Name</th>
<th>Body Temperature</th>
<th>Skin Cover</th>
<th>Gives Birth</th>
<th>Aquatic Creature</th>
<th>Aerial Creature</th>
<th>Has Legs</th>
<th>Hibernates</th>
<th>Class Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>human</td>
<td>warm-blooded</td>
<td>hair</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>Mammals</td>
</tr>
<tr>
<td>python</td>
<td>cold-blooded</td>
<td>scales</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>Reptiles</td>
</tr>
<tr>
<td>salmon</td>
<td>cold-blooded</td>
<td>scales</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>Fishes</td>
</tr>
<tr>
<td>whale</td>
<td>warm-blooded</td>
<td>hair</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>Mammals</td>
</tr>
<tr>
<td>frog</td>
<td>cold-blooded</td>
<td>none</td>
<td>no</td>
<td>semi</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>Amphibians</td>
</tr>
<tr>
<td>komodo dragon</td>
<td>cold-blooded</td>
<td>scales</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>Reptiles</td>
</tr>
<tr>
<td>bat</td>
<td>warm-blooded</td>
<td>hair</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>Mammals</td>
</tr>
<tr>
<td>pigeon</td>
<td>warm-blooded</td>
<td>feathers</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>Birds</td>
</tr>
<tr>
<td>cat</td>
<td>warm-blooded</td>
<td>fur</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>Mammals</td>
</tr>
<tr>
<td>guppy</td>
<td>cold-blooded</td>
<td>scales</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>Fishes</td>
</tr>
<tr>
<td>alligator</td>
<td>cold-blooded</td>
<td>scales</td>
<td>yes</td>
<td>no</td>
<td>semi</td>
<td>yes</td>
<td>no</td>
<td>Reptiles</td>
</tr>
<tr>
<td>penguin</td>
<td>warm-blooded</td>
<td>feathers</td>
<td>no</td>
<td>semi</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>Birds</td>
</tr>
<tr>
<td>porcupine</td>
<td>warm-blooded</td>
<td>quills</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>Mammals</td>
</tr>
<tr>
<td>eel</td>
<td>cold-blooded</td>
<td>scales</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>Fishes</td>
</tr>
<tr>
<td>salamander</td>
<td>cold-blooded</td>
<td>none</td>
<td>yes</td>
<td>semi</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>Amphibians</td>
</tr>
</tbody>
</table>

Rule set:

$r_1$: (Gives Birth = no) $\land$ (Aerial Creature = yes) $\rightarrow$ Birds
$r_2$: (Gives Birth = no) $\land$ (Aquatic Creature = yes) $\rightarrow$ Fishes
$r_3$: (Gives Birth = yes) $\land$ (Body Temperature = warm-blooded) $\rightarrow$ Mammals
$r_4$: (Gives Birth = no) $\land$ (Aerial Creature = no) $\rightarrow$ Reptiles
$r_5$: (Aquatic Creature = semi) $\rightarrow$ Amphibians
application:

- a test instance is predicted based on the rule it triggers:

<table>
<thead>
<tr>
<th></th>
<th>Body Temperature</th>
<th>Skin Cover</th>
<th>Gives Birth</th>
<th>Aquatic Creature</th>
<th>Aerial Creature</th>
<th>Has Legs</th>
<th>Hibernates</th>
</tr>
</thead>
<tbody>
<tr>
<td>hawk</td>
<td>warm-blooded</td>
<td>feather</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>grizzly bear</td>
<td>warm-blooded</td>
<td>fur</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>

Hawk: triggers rule r1 -> Bird
Grizzly: triggers rule r3 -> mammal
and now?

\[
\begin{align*}
\text{r}_1: & \quad \text{(~Gives Birth = no) \land (Aerial Creature = yes) \rightarrow Birds} \\
\text{r}_2: & \quad \text{(~Gives Birth = no) \land (Aquatic Creature = yes) \rightarrow Fishes} \\
\text{r}_3: & \quad \text{(~Gives Birth = yes) \land (Body Temperature = warm-blooded) \rightarrow Mammals} \\
\text{r}_4: & \quad \text{(~Gives Birth = no) \land (Aerial Creature = no) \rightarrow Reptiles} \\
\text{r}_5: & \quad \text{(Aquatic Creature = semi) \rightarrow Amphibians}
\end{align*}
\]

<table>
<thead>
<tr>
<th>Name</th>
<th>Body Temperature</th>
<th>Skin Cover</th>
<th>Gives Birth</th>
<th>Aquatic Creature</th>
<th>Aerial Creature</th>
<th>Has Legs</th>
<th>Hibernates</th>
</tr>
</thead>
<tbody>
<tr>
<td>lemur</td>
<td>warm-blooded</td>
<td>fur</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>turtle</td>
<td>cold-blooded</td>
<td>scales</td>
<td>no</td>
<td>semi</td>
<td>no</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>dogfish shark</td>
<td>cold-blooded</td>
<td>scales</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
</tbody>
</table>

Lemur: triggers rule r3 -> mammal
Turtle: triggers rule r3 and r4 ???
Dogfish: none ???
Ordered rule set and default!

- Rules are ordered by priority
- A default rule has to be set
Sequential Covering Algorithm

• Many rule-based algorithms use it

• Algorithm:
  - initialize rule set R to be empty
  - repeat until no rule can be added:
    - grow a single rule r
    - eliminate all training instances covered by rule r
    - add rule r to the current rule set R
(i) Original Data

(ii) Step 1
Extract a rule with high accuracy
example

(iii) Step 2
Eliminate instances covered by the rule

(iv) Step 3
Extract the next high accuracy rule

Question: Which class (rule) should we extract first?
Class-based ordering

- learn the rules for the smallest class first, followed by the next highest, and so forth
- default is the largest class remaining
Nearest-Neighbor Classifier

Given a test instance:
- Compute its distance to all the training instances
- Identify its $k$ nearest neighbors
- Use class labels of $k$ nearest neighbors to predict the class label of test instance (e.g., by taking majority vote)
Nearest-Neighbor Classifier

- Requires a distance/similarity measure
  - euclidian, Mahalanobis, cosine similarity …
- specify parameter k (number of nearest neighbors to consider)
Nearest-Neighbor Classifier
choosing value for k

- k too small, sensitive to noise points
- k too large, neighborhood includes points from other classes
Decision Boundary Classifier

• Define a boundary

• boundary divides space into classes
NN meets DB

Decision boundary using $k=15$
classification as a learning problem

• Classification can be viewed as the problem of learning $f(x)$ that defines the decision boundary
Linear Classifier

- Construct a linear decision boundary to separate instances from different classes

\[
\begin{align*}
\text{Linear Model: } f(x) &= w^T x + w_o \\
\text{Predicted class: } \hat{y} &= \begin{cases} +1 & \text{if } f(x) \geq 0 \\ -1 & \text{otherwise} \end{cases}
\end{align*}
\]

- perceptron, linear SVM, logistic regression, …
Nonlinear Classifier

- same idea as linear, but uses a nonlinear model

Nonlinear Model (example):

\[ f(x) = (w_0 + w_1 x_1 + w_2 x_2)^2 \]
\[ - w_3 x_1 x_2 - w_4 (x_1 + x_2) - w_5 \]

Predicted class:
\[ \hat{y} = \begin{cases} 
+1 & \text{if } f(x) \geq 0 \\
-1 & \text{otherwise} 
\end{cases} \]

- nonlinear SVM, artificial neural network, …
Artificial Neural Network

<table>
<thead>
<tr>
<th>$X_1$</th>
<th>$X_2$</th>
<th>$X_3$</th>
<th>$Y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>-1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>-1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-1</td>
</tr>
</tbody>
</table>

Black box

Input

$X_1$

$X_2$

$X_3$

Output

$Y$
inside the black box
typical function

• transfer:
  • sum, product

• activation:
  • sigmoid, tanh, sin, cos, threshold, …
layers in ANN

Input Layer

Hidden Layer

Output Layer

$x_1 \ x_2 \ x_3 \ x_4 \ x_5$

y
where do the weights come from?

- hill climbing, simulated annealing, Manhattan, evolution
- Baum-Welch, Back propagation, forward-backward algorithm
Choosing the Right Classifier

- Sophisticated classifiers like multilayer ANN or nonlinear SVM: great accuracy but non descriptive (black box)

- Simple ones like Decision Tree or Rule based: mediocre accuracy but descriptive model (unless overfit...)
Choosing the Right Classifier

- Sophisticated classifiers like multilayer ANN or nonlinear SVM: assume continuous and numerical data -> transform categories into 0.0 1.0 or -1.0 1.0

- Simple ones like Decision Tree or Rule based: can take everything, finds bins/thresholds automatically (depends on algorithm)
limits and contraints

• imbalanced data (99% in one class 1% in the other)

• noise, uncertainty, discrepancy between learning and testing data

• high dimensionality

• time and storage complexity

• parameter optimization, local maxima, complexity

• sometimes they find simple heuristics (are they true?)