Data Mining MTAT.03.183
Introduction to Machine Learning

Jaak Vilo
2015 Spring
Acknowledgement

• Konstantin Tretyakov – 2009 course materials
• Lecture Notes for E Alpaydın 2010 Introduction to Machine Learning

• Spring – Machine Learning course
  – Sven Laur, Konstantin Tretyakov, Meelis Kull, Raivo Kolde, Anna Leontjeva, Kristjan Korjus...
Coming up next

• **“Machine learning”**
  – Terminology, foundations, general framework.

• **Supervised machine learning**
  – Basic ideas, algorithms & toy examples.

• **Statistical challenges**
  – P-values, significance, consistency, stability

• **State of the art techniques**
  – SVM, kernel methods, graphical models, latent variable models, boosting, bagging, LASSO, on-line learning, deep learning, reinforcement learning, …
Learning Associations

• Basket analysis:
  \[ P(Y \mid X) \] probability that somebody who buys X also buys Y where X and Y are products/services.

Example: \( P(\text{chips} \mid \text{beer}) = 0.7 \)
Classification

- Example: Credit scoring
- Differentiating between low-risk and high-risk customers from their income and savings

**Discriminant:** IF $\text{income} > \theta_1$ AND $\text{savings} > \theta_2$

**THEN** low-risk ELSE high-risk
Classification: Applications

• Aka Pattern recognition
• **Face recognition**: Pose, lighting, occlusion (glasses, beard), make-up, hair style
• **Character recognition**: Different handwriting styles.
• **Speech recognition**: Temporal dependency.
• **Medical diagnosis**: From symptoms to illnesses
• **Biometrics**: Recognition/authentication using physical and/or behavioral characteristics: Face, iris, signature, etc
• ...

Lecture Notes for E Alpaydın 2010
Introduction to Machine Learning 2e ©
The MIT Press (V1.0)
Face Recognition

Training examples of a person

Test images

ORL dataset,
AT&T Laboratories, Cambridge UK
Regression

- Example: Price of a used car
- \( x \): car attributes
  - \( y \): price
  - \( y = g(x | \theta) \)
  - \( g(\cdot) \) model,
  - \( \theta \) parameters

\[ y = wx + w_0 \]
Regression Applications

• Navigating a car: Angle of the steering
• Kinematics of a robot arm

\[ \alpha_1 = g_1(x,y) \]
\[ \alpha_2 = g_2(x,y) \]
Supervised Learning: Uses

• **Prediction of future cases**: Use the rule to predict the output for future inputs

• **Knowledge extraction**: The rule is easy to understand

• **Compression**: The rule is simpler than the data it explains

• **Outlier detection**: Exceptions that are not covered by the rule, e.g., fraud
## Supervised learning

<table>
<thead>
<tr>
<th>Observation</th>
<th>Outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summer of 2003 was <strong>cold</strong></td>
<td>Winter of 2003 was <strong>warm</strong></td>
</tr>
<tr>
<td>Summer of 2004 was <strong>cold</strong></td>
<td>Winter of 2004 was <strong>cold</strong></td>
</tr>
<tr>
<td>Summer of 2005 was <strong>cold</strong></td>
<td>Winter of 2005 was <strong>cold</strong></td>
</tr>
<tr>
<td>Summer of 2006 was <strong>hot</strong></td>
<td>Winter of 2006 was <strong>warm</strong></td>
</tr>
<tr>
<td>Summer of 2007 was <strong>cold</strong></td>
<td>Winter of 2007 was <strong>cold</strong></td>
</tr>
<tr>
<td>Summer of 2008 was <strong>warm</strong></td>
<td>Winter of 2008 was <strong>warm</strong></td>
</tr>
<tr>
<td>Summer of 2009 was <strong>cold</strong></td>
<td>Winter of 2009 was <strong>cold</strong></td>
</tr>
<tr>
<td>Summer of 2010 was <strong>hot</strong></td>
<td>Winter of 2010 was <strong>cold</strong></td>
</tr>
<tr>
<td>Summer of 2011 was <strong>warm</strong></td>
<td>Winter of 2011 will be ?</td>
</tr>
</tbody>
</table>
### Supervised learning

<table>
<thead>
<tr>
<th>Observation</th>
<th>Outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>Study=hard, Professor=:redface:</td>
<td>I get a C</td>
</tr>
<tr>
<td>Study=slack,</td>
<td>I get an A</td>
</tr>
<tr>
<td>Study=hard, Professor=:redface:</td>
<td>I get an A</td>
</tr>
<tr>
<td>Study=slack,</td>
<td>I get a D</td>
</tr>
<tr>
<td>Study=slack,</td>
<td>I get an A</td>
</tr>
<tr>
<td>Study=slack,</td>
<td>I get an A</td>
</tr>
<tr>
<td>Study=hard, Professor=:redface:</td>
<td>I get an A</td>
</tr>
<tr>
<td>Study=slack,</td>
<td>I get an A</td>
</tr>
<tr>
<td>Study=slack,</td>
<td>I get an A</td>
</tr>
<tr>
<td>?</td>
<td>I get an A</td>
</tr>
</tbody>
</table>
## Supervised learning

<table>
<thead>
<tr>
<th>Day</th>
<th>Observation</th>
<th>Outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mon</td>
<td>I was <strong>not</strong> using magnetic bracelet™</td>
<td>In the evening I had a headache</td>
</tr>
<tr>
<td>Tue</td>
<td>I was using magnetic bracelet™</td>
<td>In the evening I had less headache</td>
</tr>
<tr>
<td>Wed</td>
<td>I was using magnetic bracelet™</td>
<td>In the evening no headache!</td>
</tr>
<tr>
<td>Thu</td>
<td>I was using magnetic bracelet™</td>
<td>The headache is gone!!</td>
</tr>
<tr>
<td>Fri</td>
<td>I was <strong>not</strong> using magnetic bracelet™</td>
<td>No headache!!</td>
</tr>
</tbody>
</table>
### Supervised learning

<table>
<thead>
<tr>
<th>Day</th>
<th>Observation</th>
<th>Outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mon</td>
<td>I was <strong>not</strong> using magnetic bracelet™</td>
<td>In the evening I had a headache</td>
</tr>
<tr>
<td>Tue</td>
<td>I was using magnetic bracelet™</td>
<td>In the evening I had less headache</td>
</tr>
<tr>
<td>Wed</td>
<td>I was using magnetic bracelet™</td>
<td>In the evening no headache!</td>
</tr>
<tr>
<td>Thu</td>
<td>I was using magnetic bracelet™</td>
<td>The headache is gone!!</td>
</tr>
<tr>
<td>Fri</td>
<td>I was <strong>not</strong> using magnetic bracelet™</td>
<td>No headache!!</td>
</tr>
</tbody>
</table>

**Magnetic bracelet™ cures headache**
Science

Collect data/observations
Supervised learning

Collect data/observations → Come up with theories
Supervised learning

1. Collect data/observations
2. Come up with theories
3. Better Theory
Supervised learning

1. Collect data/observations
2. Come up with theories
3. Better Theory

Collect data/observations → Come up with theories → Better Theory

Come up with theories → Collect data/observations

Better Theory → Come up with theories
Science 😊
Supervised learning

• Formally,

Let $\mathcal{X}$ and $\mathcal{Y}$ be some sets and let there be a dataset of training samples:

$$D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n) \mid x_i \in \mathcal{X}, y_i \in \mathcal{Y}\}$$

Find a function $f_D : \mathcal{X} \rightarrow \mathcal{Y}$ generalizing the functional relationship present in the data.
Regression

- $\mathcal{X} = \mathbb{R}$, $\mathcal{Y} = \mathbb{R}$.
- $D = \{(0.50, 0.26), (0.43, 0.08), (0.26, 0.00), \ldots \}$
- $f_D(x) = x^2$. 
Classification

- $\mathcal{X} = \mathbb{R}^2$, $\mathcal{Y} = \{\text{blue}, \text{red}\}$.
- $D = \{((1.3, 0.8), \text{red}), ((2.5, 2.3), \text{blue}), \ldots \}$
- $f_D(x_1, x_2) = \text{if } x_1 + x_2 > 3 \text{ then blue else red.}$
Classification as the task of mapping an input attribute set $x$ into its class label $y$. 

**Figure 4.2.** Classification as the task of mapping an input attribute set $x$ into its class label $y$. 

146 Chapter 4 Classification
The “Dumb User” Perspective

Weka, RapidMiner, MS SSAS, Clementine, SPSS, R, scikit-learn,…

\[ f(x) = \sum_i w_i x_i \]
The “Dumb User” Perspective

Validation

\[ f(x) = \sum_i w_i x_i \]
Classification demo: Iris dataset

- 150 measurements, 4 attributes, 3 classes
Classification demo: Iris dataset

Tree View

- petalwidth
  - <= 0.6
    - Iris-setosa (50.0)
  - > 0.6
    - petalwidth
      - <= 1.7
        - petallength
          - <= 4.9
            - Iris-versicolor (48.0/1.0)
          - > 4.9
            - Iris-virginica (46.0/1.0)
      - > 1.7
        - petalwidth
          - <= 1.5
            - Iris-virginica (3.0)
          - > 1.5
            - Iris-versicolor (3.0/1.0)
Validation

a  b  c  <-- classified as
50  0  0 |  a = Iris-setosa
0  49  1 |  b = Iris-versicolor
0  2 48 |  c = Iris-virginica

Correctly Classified Instances 147 98%
Incorrectly Classified Instances 3 2%
Kappa statistic 0.97
Mean absolute error 0.0233
Root mean squared error 0.108
Relative absolute error 5.2482 %
Root relative squared error 22.9089 %
Total Number of Instances 150
Validation

\[
\begin{array}{ccc}
\text{a} & \text{b} & \text{c} \\
50 & 0 & 0 & \Rightarrow \text{classified as} \\
0 & 49 & 1 & \text{a = Iris-setosa} \\
0 & 2 & 48 & \text{b = Iris-versicolor} \\
\end{array}
\]

\[
\begin{array}{cccc}
\text{Class} & \text{setosa} & \text{versic.} & \text{virg.} & \text{Avg} \\
\text{TP Rate} & 1 & 0.98 & 0.96 & 0.98 \\
\text{FP Rate} & 0 & 0.02 & 0.01 & 0.01 \\
\text{Precision} & 1 & 0.961 & 0.98 & 0.98 \\
\text{Recall} & 1 & 0.98 & 0.96 & 0.98 \\
\text{F-Measure} & 1 & 0.97 & 0.97 & 0.98 \\
\text{ROC Area} & 1 & 0.99 & 0.99 & 0.99 \\
\end{array}
\]
Validation

a b c   <-- classified as
50 0 0 | a = Iris-setosa
0 49 1 | b = Iris-versicolor
0 2 48 | c = Iris-virginica

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>TP</td>
<td>0.98</td>
<td></td>
</tr>
<tr>
<td>FP</td>
<td>0.02</td>
<td></td>
</tr>
<tr>
<td>Precision</td>
<td>0.961</td>
<td></td>
</tr>
<tr>
<td>Recall</td>
<td>0.98</td>
<td></td>
</tr>
<tr>
<td>F-Measure</td>
<td>0.97</td>
<td></td>
</tr>
<tr>
<td>ROC Area</td>
<td>0.99</td>
<td></td>
</tr>
</tbody>
</table>

versic.
Validation

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>c</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>50</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>a</td>
<td>=</td>
<td>Iris-setosa</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>49</td>
<td>1</td>
</tr>
<tr>
<td>b</td>
<td>=</td>
<td>Iris-versicolor</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>2</td>
<td>48</td>
</tr>
<tr>
<td>c</td>
<td>=</td>
<td>Iris-virginica</td>
<td></td>
</tr>
</tbody>
</table>

```
TP Rate 0.98 = TP/positive examples
FP Rate 0.02 = FP/negative examples
Precision 0.961 = TP/positives
Recall 0.98 = TP/positive examples
F-Measure 0.97 = 2*P*R/(P + R)
ROC Area 0.99 ~ Pr(s(false)<s(true))
```

"True positives" "False positives"

versic.
## Classification summary

<table>
<thead>
<tr>
<th></th>
<th>Positives</th>
<th>Negatives</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Predicted = Yes</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Actual = Yes</strong></td>
<td>True positives (TP)</td>
<td>False negatives (FN)</td>
</tr>
<tr>
<td></td>
<td>(Type II, β-error)</td>
<td>(Type I, α-error)</td>
</tr>
<tr>
<td><strong>Actual = No</strong></td>
<td>False positives (FP)</td>
<td>True negatives (FN)</td>
</tr>
<tr>
<td></td>
<td>(Type I, α-error)</td>
<td></td>
</tr>
</tbody>
</table>
## Classification summary

<table>
<thead>
<tr>
<th>Positives</th>
<th>Negatives</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Predicted = Yes</td>
<td>Predicted = No</td>
</tr>
<tr>
<td>Actual = Yes</td>
<td>True positives (TP)</td>
</tr>
<tr>
<td>Actual = No</td>
<td>False positives (FP) (Type I, α-error)</td>
</tr>
</tbody>
</table>
ROC curve – Receiver Operator Characteristic, AUC – Area Under Curve
“Training” classifiers on data

• Thus, a “good” classifier is the one which has good Accuracy/Precision/Recall.
  – Or better ROC curve, higher ROC AUC score

• Hence, machine learning boils down to finding a function that optimizes these parameters for given data.
“Training” classifiers on data

• Thus, a “good” classifier is the one which has good Accuracy/Precision/Recall.

• Hence, machine learning boils down to finding a function that optimizes these parameters for given data.

• Yet, there’s a catch
“Training” classifiers on data

• **We want our algorithm to perform well on “unseen” data!**
  
  – This makes algorithms and theory way more complicated.
  
  – This makes validation somewhat more complicated.
Proper validation

• You may not test your algorithm on the same data that you used to train it!
Proper validation

• You may not test your algorithm on the same data that you used to train it!
Proper validation :: Holdout

Training set

Split

Testing set

Validation
Figure 4.3. General approach for building a classification model.
Proper validation

• What are the “sufficient” sizes for the test/training sets and why?

• What if the data is scarce?
  – Cross-validation
  – K-fold cross-validation
  – Leave-one-out cross-validation
  – Bootstrap .632+
The Netflix Prize Data

- Netflix released three datasets
  - 480,189 users (anonymous)
  - 17,770 movies
  - ratings on integer scale 1 to 5

- **Training set**: 99,072,112 <user, movie> pairs with ratings
- **Probe set**: 1,408,395 <user, movie> pairs with ratings
- **Qualifying set** of 2,817,131 <user, movie> pairs with no ratings

---

Jeff Howbert
Why the Netflix Prize Was Hard

- Massive dataset
- Very sparse – matrix only 1.2% occupied
- Extreme variation in number of ratings per user
- Statistical properties of qualifying and probe sets different from training set

<table>
<thead>
<tr>
<th></th>
<th>movie 1</th>
<th>movie 2</th>
<th>movie 3</th>
<th>movie 4</th>
<th>movie 5</th>
<th>movie 6</th>
<th>movie 7</th>
<th>movie 8</th>
<th>movie 9</th>
<th>movie 10</th>
<th>...</th>
<th>movie 17770</th>
</tr>
</thead>
<tbody>
<tr>
<td>user 1</td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>user 2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>user 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5</td>
<td>3</td>
<td></td>
<td></td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>user 4</td>
<td>2</td>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>user 5</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5</td>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>user 6</td>
<td></td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>user 7</td>
<td></td>
<td></td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>user 8</td>
<td>3</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>user 9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>user 10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>user 480189</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Jeff Howbert
Model Building and Submission Process

- **Training Set**: 99,072,112
- **Probe Set**: 1,408,395
- **Quiz Set**: 1,408,342
- **Test Set**: 1,408,789

**Model Tuning**

- **RMSE on Public Leaderboard**: 1,408,342
- **RMSE on Quiz Set**: 1,408,342
- **RMSE on Test Set**: 1,408,789
- **RMSE Kept Secret for Final Scoring**

**Ratings Known**

- **Qualifying Set** (ratings unknown)

**Jeff Howbert**
# Leaderboard

<table>
<thead>
<tr>
<th>Rank</th>
<th>Team Name</th>
<th>Best Score</th>
<th>% Improvement</th>
<th>Last Submit Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>--</td>
<td>No Grand Prize candidates yet</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>1</td>
<td>BellKor in BigChaos</td>
<td>0.8604</td>
<td>9.56</td>
<td>2008-12-03 18:46:15</td>
</tr>
<tr>
<td>2</td>
<td>BigChaos</td>
<td>0.8626</td>
<td>9.33</td>
<td>2008-12-04 19:18:27</td>
</tr>
<tr>
<td>3</td>
<td>BellKor</td>
<td>0.8690</td>
<td>9.29</td>
<td>2008-12-04 19:25:59</td>
</tr>
<tr>
<td>4</td>
<td>PragmaticTheory</td>
<td>0.8638</td>
<td>9.21</td>
<td>2008-11-26 11:46:23</td>
</tr>
<tr>
<td>5</td>
<td>Gravity</td>
<td>0.8654</td>
<td>9.04</td>
<td>2008-11-27 21:18:37</td>
</tr>
<tr>
<td>6</td>
<td>My Brain and His Chain</td>
<td>0.8668</td>
<td>8.89</td>
<td>2008-09-30 02:19:47</td>
</tr>
<tr>
<td>7</td>
<td>Just a guy in a garage</td>
<td>0.8672</td>
<td>8.85</td>
<td>2008-12-07 06:51:12</td>
</tr>
<tr>
<td>8</td>
<td>When Gravity and Dinosaurs Unite</td>
<td>0.8675</td>
<td>8.82</td>
<td>2008-10-05 14:16:53</td>
</tr>
<tr>
<td>9</td>
<td>Opera Solutions</td>
<td>0.8676</td>
<td>8.81</td>
<td>2008-12-02 22:09:45</td>
</tr>
<tr>
<td>10</td>
<td>acmehil</td>
<td>0.8677</td>
<td>8.80</td>
<td>2008-12-05 08:01:00</td>
</tr>
<tr>
<td>11</td>
<td>scientist</td>
<td>0.8677</td>
<td>8.80</td>
<td>2008-12-02 01:10:13</td>
</tr>
<tr>
<td>12</td>
<td>Cas</td>
<td>0.8711</td>
<td>8.44</td>
<td>2008-08-25 05:00:23</td>
</tr>
<tr>
<td>13</td>
<td>Dace</td>
<td>0.8711</td>
<td>8.44</td>
<td>2008-12-07 03:46:04</td>
</tr>
</tbody>
</table>

Progress Prize 2007 - RMSE = 0.8712 - Winning Team: KorBell

<table>
<thead>
<tr>
<th>Rank</th>
<th>Team Name</th>
<th>Best Score</th>
<th>% Improvement</th>
<th>Last Submit Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>KorBell</td>
<td>0.8712</td>
<td>8.43</td>
<td>2007-10-01 23:25:23</td>
</tr>
<tr>
<td>15</td>
<td>basho</td>
<td>0.8714</td>
<td>8.41</td>
<td>2008-05-21 22:06:00</td>
</tr>
<tr>
<td>16</td>
<td>pongpongzhou</td>
<td>0.8714</td>
<td>8.41</td>
<td>2008-11-05 01:11:13</td>
</tr>
<tr>
<td>17</td>
<td>blednotk</td>
<td>0.8717</td>
<td>8.38</td>
<td>2008-11-26 00:12:12</td>
</tr>
</tbody>
</table>
Intermediate summary

• **Supervised learning** = predicting \( f(x) \) **well**.
• For classification, “well” = high accuracy/precision/recall on unseen data.
• To achieve that, most training algorithms will try to optimize their accuracy/precision/recall on training data.
• We can then **validate** how good they are on test data.
Next

• **Three examples of approaches**
  – **Ad-hoc**
    • *Decision tree induction*
  – **Probabilistic modeling**
    • *Naïve Bayes classifier*
  – **Objective function optimization**
    • *Linear least squares regression*
Decision Tree Induction :: ID3

- Iterative Dichotomizer 3
  - Simple yet popular decision tree induction algorithm
  - Builds a decision tree top-down, starting at the root.

Ross Quinlan
ID3

<table>
<thead>
<tr>
<th>Day</th>
<th>Outlook</th>
<th>Temp</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>
ID3 :: First split

Which split is the most informative?
Entropy

• Entropy measures the “informativeness” of a probability distribution.

\[ H(p) = - \sum_i p_i \log_2 p_i \]

• A split is informative if it reduces entropy.
Information gain of a split

• Before split:
  \[ p_{\text{no}} = \frac{5}{14}, \ p_{\text{yes}} = \frac{9}{14}, \ H(p) = 0.94 \]

• After split on outlook:

\[
\begin{align*}
H &= 0.97 \quad \text{H}=0 \quad H=0.97 \\
& \quad \quad 0.97 \frac{5}{14} + 0.4 \frac{4}{14} + 0.97 \frac{5}{14} \\
& = 0.69
\end{align*}
\]

Information gain

\[ = 0.94 - 0.69 = 0.25 \]
Entropy, Gini, Misclassification

Figure 4.13. Comparison among the impurity measures for binary classification problems.
ID3

1. Start with a single node
2. Find the attribute with the largest information gain
3. Split the node according to this attribute
4. Repeat recursively on subnodes
C4.5

• C4.5 is an extension of ID3
  – Supports continuous attributes
  – Supports missing values
  – Supports pruning

• There is also a C5.0
  – A commercial version with additional bells & whistles
Decision trees - discussion

• The goods:
  – Easy & efficient
  – Interpretable and pretty

• The bads
  – Rather ad-hoc – optimal trees NP-hard to train
  – Can overfit unless properly pruned
  – Not the best model for all classification tasks

• Improvements:
  – Random Forest – multiple trees, majority voting
Why Does my Method Overfit?

- **In domains with noise or uncertainty** the system may try to decrease the training error by completely fitting all the training examples.

The learner overfits to correctly classify the noisy instances.

Occam’s razor: Prefer the simplest hypothesis that fits the data with high accuracy.

Noisy instances

Slide from: Pavan J Joshi
Overfitting

- The longer you train, the worse it becomes...

- Always test against data that was not used in training
Pruning

• Delete subtrees if they do not improve predictions
• Top/down
• Bottom/up
• …
• MDL – Minimum Description Length Principle
  – MDL - Minimise length of “Model + Exceptions”
Why do we trust in one tree?

• Tree is different depending on data used

• Which attributes are most important?

• **Randomly** select a subset of data, randomly subselect a subset of attributes ...
Random Forest

Each trained on a possibly random subset of data
Each trained on a possibly random sub-set of attributes

Every tree provides some level of accuracy.

Majority voting!
Data (2 Gaussians), RF, Logistic regression
Figure 1.

Illustration of a case where there is no marginal differential expression of individual genes, however in the multivariate setting the differential expression becomes clear. Projecting the data onto the appropriate direction in this case leads to a clear separation between the classes.

References

Naïve Bayes Classifier

\[ P(A|B) = \frac{P(B|A) P(A)}{P(B)} \]
The Tennis Dataset

<table>
<thead>
<tr>
<th>Day</th>
<th>Outlook</th>
<th>Temp</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>
Shall we play tennis today?

<table>
<thead>
<tr>
<th>PlayTennis</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
</tr>
<tr>
<td>No</td>
</tr>
<tr>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
</tr>
<tr>
<td>No</td>
</tr>
<tr>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
</tr>
<tr>
<td>No</td>
</tr>
</tbody>
</table>

12.11.2009 Machine learning :: Introduction :: Part II
Shall we play tennis today?

- Probabilistic model:

<table>
<thead>
<tr>
<th>PlayTennis</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td></td>
</tr>
<tr>
<td>No</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>No</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>No</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>

\[ P(Yes) = \frac{9}{14} = 0.64 \]
\[ P(No) = \frac{5}{14} = 0.36 \]

⇒ Yes
It’s windy today. Tennis, anyone?

<table>
<thead>
<tr>
<th>Wind</th>
<th>PlayTennis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>Strong</td>
<td>No</td>
</tr>
</tbody>
</table>
It’s windy today. Tennis, anyone?

- Probabilistic model:

<table>
<thead>
<tr>
<th>Wind</th>
<th>PlayTennis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>Strong</td>
<td>No</td>
</tr>
</tbody>
</table>

\[
P(\text{Weak}) = \frac{8}{14} \\
P(\text{Strong}) = \frac{6}{14}
\]

\[
P(\text{Yes} \mid \text{Weak}) = \frac{6}{8} \\
P(\text{No} \mid \text{Weak}) = \frac{2}{8}
\]

\[
P(\text{Yes} \mid \text{Strong}) = \frac{3}{6} \\
P(\text{No} \mid \text{Strong}) = \frac{3}{6}
\]
More attributes

- Probabilistic model:

<table>
<thead>
<tr>
<th>Humidity</th>
<th>Wind</th>
<th>PlayTennis</th>
</tr>
</thead>
<tbody>
<tr>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>Normal</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>High</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>High</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
</tbody>
</table>

\[
P(\text{High,Weak}) = \frac{4}{14}
\]

\[
P(\text{Yes} \mid \text{High,Weak}) = \frac{2}{4}
\]

\[
P(\text{No} \mid \text{High,Weak}) = \frac{2}{4}
\]

\[
P(\text{High,Strong}) = \frac{3}{14}
\]

\[
P(\text{Yes} \mid \text{High,Strong}) = \frac{1}{3}
\]

\[
P(\text{No} \mid \text{High,Strong}) = \frac{2}{3}
\]

...
The Bayes Classifier

In general:

1. **Estimate from data:**
   \[ P(\text{Class} \mid X_1, X_2, X_3, \ldots) \]

2. **For a given instance** \((X_1, X_2, X_3 \ldots)\)
   predict class whose **conditional probability** is **greater**:

   \[ P(C_1 \mid X_1, X_2, X_3, \ldots) > P(C_2 \mid X_1, X_2, X_3, \ldots) \]
   \(\Rightarrow\) predict \(C_1\)
The Bayes Classifier

In general:

1. $P(C_1 \mid X) > P(C_2 \mid X)$

→ predict $C_1$
The Bayes Classifier

In general:

1. $P(C_1 \mid X) > P(C_2 \mid X)$

⇒ predict $C_1$

$$P(A \mid B) = \frac{P(B \mid A) P(A)}{P(B)}$$
The Bayes Classifier

In general:

1. \( P(C_1 | X) > P(C_2 | X) \)
   \[ \Rightarrow \text{predict } C_1 \]

2. \( P(X | C_1)P(C_1)/P(X) > P(X | C_2)P(C_2)/P(X) \)
   \[ \Rightarrow \text{predict } C_1 \]
The Bayes Classifier

In general:

1. \( P(C_1 \mid X) > P(C_2 \mid X) \)
   \( \Rightarrow \) predict \( C_1 \)

2. \( P(X \mid C_1)P(C_1)/P(X) > P(X \mid C_2)P(C_2)/P(X) \)
   \( \Rightarrow \) predict \( C_1 \)

3. \( P(X \mid C_1)P(C_1) > P(X \mid C_2)P(C_2) \)
   \( \Rightarrow \) predict \( C_1 \)
The Bayes Classifier

In general:

1. $P(C_1 | X) > P(C_2 | X)$
   $\Rightarrow$ predict $C_1$

2. $P(X | C_1)P(C_1)/P(X) > P(X | C_2)P(C_2)/P(X)$
   $\Rightarrow$ predict $C_1$

3. $P(X | C_1)P(C_1) > P(X | C_2)P(C_2)$
   $\Rightarrow$ predict $C_1$

4. $P(X | C_1)/P(X | C_2) > P(C_2)/P(C_1)$
   $\Rightarrow$ predict $C_1$
The Bayes Classifier

3. $P(X \mid C_1)P(C_1) > P(X \mid C_2)P(C_2) \implies \text{predict } C_1$
The Bayes Classifier

If the true underlying distribution is known, the Bayes classifier is optimal (i.e. it achieves minimal error probability).
Problem

• We need exponential amount of data

<table>
<thead>
<tr>
<th>Humidity</th>
<th>Wind</th>
<th>PlayTennis</th>
</tr>
</thead>
<tbody>
<tr>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>Normal</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>High</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>High</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
</tbody>
</table>

\[
P(\text{High,Weak}) = \frac{4}{14}
\]
\[
P(\text{Yes} \mid \text{High,Weak}) = \frac{2}{4}
\]
\[
P(\text{No} \mid \text{High,Weak}) = \frac{2}{4}
\]
\[
P(\text{High,Strong}) = \frac{3}{14}
\]
\[
P(\text{Yes} \mid \text{High,Strong}) = \frac{1}{3}
\]
\[
P(\text{No} \mid \text{High,Strong}) = \frac{2}{3}
\]

…
The Naïve Bayes Classifier

To scale beyond 2-3 attributes, use a trick:

Assume that attributes of each class are independent:

\[
P(X_1, X_2, X_3 \mid \text{Class}) = \\
= P(X_1 \mid \text{Class})P(X_2 \mid \text{Class})P(X_3 \mid \text{Class})\ldots
\]
The Naïve Bayes Classifier

\[
P(X | C_1)/P(X | C_2) > P(C_2)/P(C_1)
\]

\[\Rightarrow \text{predict } C_1\]

becomes

\[
\prod_{i} \frac{P(X_i | C_1)}{P(X_i | C_2)} > \frac{P(C_2)}{P(C_1)} \Rightarrow \text{Predict } C_1
\]
Naïve Bayes Classifier

1. Training:
   For each value \( v \) of each attribute \( i \), compute
   \[
   m(i, v) = \frac{P(X_i = v | C_1)}{P(X_i = v | C_2)}
   \]

2. Classification:
   For a given instance \((x_1, x_2, \ldots)\) compute
   \[
   \text{score} = \prod_i m(i, x_i)
   \]
   If \( \text{score} > \frac{P(C_2)}{P(C_1)} \) classify as \( C_1 \), else \( C_2 \)
Naïve Bayes Classifier

1. Training:
   For each value $v$ of each attribute $i$, compute
   \[ l(i, v) = \log \frac{P(X_i = v | C_1)}{P(X_i = v | C_2)} \]

2. Classification:
   For a given instance $(x_1, x_2, \ldots)$ compute
   \[ l\text{score} = \sum_i l(i, x_i) \]
   If $l\text{score} > \log \frac{P(C_2)}{P(C_1)}$ classify as $C_1$, else $C_2$
Naïve Bayes Classifier

• Extendable to continuous attributes via kernel density estimation.

• The goods:
  – Easy to implement, efficient
  – Won’t overfit, interpretable
  – Works better than you would expect (e.g. spam filtering)

• The bads
  – “Naïve”, linear
  – Usually won’t work well for too many classes
Supervised learning

• Four examples of approaches
  – Ad-hoc
    • Decision tree induction
  – Probabilistic modeling
    • Naïve Bayes classifier
  – Objective function optimization
    • Linear least squares regression
  – Instance-based methods
    • K-nearest neighbors
Linear regression

• Consider data points of the type
  \((x_i, y_i) \in \mathbb{R}\)

• Let us search for a function of the form
  \[ f(x) = wx \]

that would have the least possible sum of error squares:

\[ E(w) = \sum_i (f(x_i) - y_i)^2 \]
Linear regression

• Set of functions $f(x) = wx$ for various $w$. 
Linear regression

- The function having the least error
Linear regression

- We need to solve: \( \arg \min_w E(w) \)

where
\[
E(w) = \sum_i (wx_i - y_i)^2
\]

- Setting the derivative to 0:
\[
E'(w) = \sum_i 2(wx_i - y_i)x_i = 0
\]

we get
\[
w = \sum_i y_ix_i / \sum_i x_i^2
\]
Linear regression

• The idea naturally generalizes to more complex cases (e.g. multivariate regression)

• The goods
  – Simple, easy
  – Interpretable, popular, extendable to many variants
  – Won’t overfit

• The bads
  – Linear, i.e. works for a limited set of datasets.
  – Nearly never perfect.
Median Error Regression (quantile)

- Minimize mean error
- More robust: minimize median error
- Requires different optimisation
  - Differential Evolution
  - Genetic Programming
  - ...
Robust Regression

least quantile of squares
(Gilli, Maringer and Schumann, 2011)

\[
\min_{\beta} e^2_\alpha(N) \quad \text{where} \quad e = X\beta - y \\
\frac{e^2_{(j-1)}}{e^2_{(j)}} \leq j = 2..N
\]
Supervised learning

• **Four examples of approaches**
  – Ad-hoc
    • Decision tree induction
  – Probabilistic modeling
    • Naïve Bayes classifier
  – Objective function optimization
    • Linear least squares regression
  – **Instance-based methods**
    • **K-nearest neighbors**
K-Nearest Neighbors
K-Nearest Neighbors

- One-nearest neighbor

![Diagram showing K-Nearest Neighbors with data points and decision boundaries.](image-url)
K-Nearest Neighbors

• **Training:**
  – Store and index all training instances

• **Classification**
  – Given an instance to classify, find $k$ instances from the training sample nearest to it.
  – Predict the majority class among these $k$ instances
K-Nearest Neighbors

• **The goods**
  – Trivial concept
  – Easy to implement
  – Asymptotically optimal
  – Allows various kinds of data

• **The bads**
  – Difficult to implement efficiently
  – Not interpretable (no model)
  – On smaller datasets loses to other methods
### Supervised learning

#### Ad-hoc
- Decision trees, forests
- Rule induction, ILP
- Fuzzy reasoning

#### Objective optimization
- Regression models
- Kernel methods, SVM, RBF
- Neural networks

#### Probabilistic models
- Naïve Bayes
  \[
  \prod_i P(X_i | C) \approx \frac{P(C_2)}{P(C_1)}
  \]
- Graphical models
- Regression models
- Density estimation

#### Instance-based
- K-NN, LOWESS
- Kernel densities
- SVM, RBF

---

[Image of equations and diagrams related to supervised learning concepts]
Supervised learning

Objective optimization
- Regression models
- Kernel methods, SVM, RBF
- Neural networks
  \[ \arg \min_w E(w) \]

Ad-hoc
- Decision trees, forests
- Rule induction, ILP
- Fuzzy reasoning

Instance-based
- K-NN, LOWESS
- Kernel densities
- SVM, RBF

Probabilistic models
- Naïve Bayes
  \[ \prod_{i=1}^{n} \frac{P(X_i|C_1)}{P(X_i|C_2)} \left( \frac{P(C_2)}{P(C_1)} \right)^{N_i} \]
- Graphical models
- Density estimation

Ensemble-learners
- Arcing, Boosting, Bagging, Dagging, Voting, Stacking

Ensemble
- Machines:
  - Ad-hoc
  - Instance-based
  - Probabilistic models
Coming up next

• “Machine learning”
  – Terminology, foundations, general framework.

• Supervised machine learning
  – Basic ideas, algorithms & toy examples.

• **Statistical challenges**
  – Learning theory, consistency, bias-variance, overfitting, ...

• State of the art techniques
  – SVM, kernel methods, graphical models, latent variable models, boosting, bagging, LASSO, on-line learning, deep learning, reinforcement learning, ...
In previous episodes
Approaches to data analysis

The general principle is the same, though:
1. Define a set of patterns of interest
2. Define a measure of goodness for the patterns
3. Find the best pattern in the data
The “No Free Lunch” Theorem
Learning purely from data is, in general, impossible

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>False</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>True</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>True</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>?</td>
</tr>
</tbody>
</table>
The “No Free Lunch” Theorem

Learning purely from data is, in general, impossible

- Is it good or bad?
- What should we do to enable learning?
The “No Free Lunch” Theorem

Learning purely from data is, in general, impossible

• Is it good or bad?
  – Good for cryptographers, bad for data miners

• What should we do to enable learning?
  – Introduce assumptions about data (“inductive bias”):
    1. How does existing data relate to the future data?
    2. What is the system we are learning?
The “No Free Lunch” Theorem

Learning purely from data is, in general, impossible
The “No Free Lunch” Theorem
Learning purely from data is, in general, impossible

Rule 1: Generalization will only come through understanding of similarity!
The “No Free Lunch” Theorem

Learning purely from data is, in general, impossible
The “No Free Lunch” Theorem
Learning purely from data is, in general, impossible
The “No Free Lunch” Theorem

Learning purely from data is, in general, impossible

Rule 2: Data can only partially substitute knowledge about the system!
Statistical Learning Theory

• What is learning and how to analyze it?
  – There are various ways to answer this question. We’ll just consider the most popular one.
Perceptron

\[ f(x) = \begin{cases} 
1 & \text{if } w \cdot x + b > 0 \\
0 & \text{otherwise} 
\end{cases} \]
Neural Network

Tumor Evaluation Neural Network

Input Layer
Area
Perimeter
Texture
Shape

Hidden Layer

Output Layer
Benign
Malignant
Statistical Learning Theory

• What is learning and how to analyze it?
  – There are various ways to answer this question. We’ll just consider the most popular one.

• Let $(x, y) \sim D$ be the distribution of data.

• We observe an i.i.d. sample:
  \[ S = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\} \]

• We produce a classifier:
  \[ g_S(x) \]
Statistical Learning Theory

\[(x, y) \sim D\]

Distribution of data (x – coords, y – color)
Statistical Learning Theory

\[ S = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\} \]

Sample S and trained classifier g
Statistical Learning Theory

\[
\text{training error} = \frac{1}{n} \sum_{i} [g_{S}(x_i) \neq y_i]
\]
Generalization error

\[ \text{error} = \Pr(g_S(x) \neq y) = \int_D [g_S(x) \neq y] f(x, y) \, dx \, dy \]
Statistical Learning Theory

\[ E(\text{error}) = \int_S \text{error}(g_S) \, dS \]

Expected generalization error
Statistical Learning Theory

• Questions:
  – What is the generalization error of our classifier?
    • How to estimate it?
    • How to find a classifier with low generalization error?
  – What is the expected generalization error of our method?
    • How to estimate it?
    • What methods have low expected generalization error?
    • What methods are asymptotically optimal (consistent)?
  – When is learning computationally tractable?
Statistical Learning Theory

• Some answers: for linear classifiers
  – Finding a linear classifier with a small training error is a good idea

\[ E(\text{error}) - \text{optimum} \leq 16 \sqrt{\frac{(d+1) \ln n + 4}{2n}} \]

(however, finding such a classifier is NP-complete, hence an alternative method must be used)
Statistical Learning Theory

• Some answers: in general
  – Small training error $\Rightarrow$ small generalization error.
    • But only if you search a limited space of classifiers.
    • The more data you have, the larger space of classifiers you can afford.
Overfitting

• Why limited space?
  – Suppose your hypothesis space is just one classifier:
    \[ f(x) = \text{if } [x > 3] \text{ then } 1 \text{ else } 0 \]
  – You pick first five training instances:
    \( (1 \rightarrow 0), (2 \rightarrow 0), (4 \rightarrow 1), (6 \rightarrow 1), (-1 \rightarrow 0) \)
  – How surprised are you? How can you interpret it?
Overfitting

• Why limited space?
  – Suppose your hypothesis space is just one classifier:
    \[ f(x) = \text{if } [x > 3] \text{ then } 1 \text{ else } 0 \]
  – You pick first five training instances:
    \[(1 \rightarrow 0), (2 \rightarrow 0), (4 \rightarrow 1), (6 \rightarrow 1), (-1 \rightarrow 0)\]
  – How surprised are you? How can you interpret it?
  – What if you had 100000 classifiers to start with and one of them matched? Would you be surprised?
Overfitting

- Why limited space?
  - Suppose your hypothesis space is just one classifier:
    
    \[ f(x) = \text{if } [x > 3] \text{ then } 1 \text{ else } 0 \]
  
  - You pick first five training instances:
    
    \[(1 \rightarrow 0), (2 \rightarrow 0), (4 \rightarrow 1), (6 \rightarrow 1), (-1 \rightarrow 0)\]
  
  - How surprised are you? How can you interpret it?

- What if you had 100000 classifiers to start with and one of them matched? Would you be surprised?

**Large hypothesis space → Small training error becomes a matter of chance → Overfitting**
Bias-variance dilemma

- So what if the data is scarce?
  - No free lunch
  - Bias-variance tradeoff:

- The only way out is to introduce a strong yet “correct” bias (or, well, to get more data).
$M = 0$

$M = 1$

$M = 3$

$M = 9$
Why Does my Method Overfit?

• In domains with noise or uncertainty, the system may try to decrease the training error by completely fitting all the training examples.

The learner overfits to correctly classify the noisy instances.

Occam’s razor: Prefer the simplest hypothesis that fits the data with high accuracy.
Summary

• Learning can be approached formally
• Learning is feasible in many cases
  – But you pay with data or prior knowledge
Summary

• Learning can be approached formally
• Learning is feasible in many cases
  – But you pay with data or prior knowledge

• You have to be careful with complex models
  – Beware overfitting
  – If data is scarce – use simple models: they are not optimal, but at least you can fit them from data!

Using complex models with scarce data is like throwing data away.
Next

• “Machine learning”
  – Terminology, foundations, general framework.

• Supervised machine learning
  – Basic ideas, algorithms & toy examples.

• Statistical challenges
  – Learning theory, bias-variance, consistency…

• State of the art techniques
  – SVM, kernel methods, graphical models, latent variable models, boosting, bagging, LASSO, on-line learning, deep learning, reinforcement learning, …
Linear classification

\[ f(x_1, x_2) = \text{sign}(x_1 + x_2 - 3) \]
Linear classification

\[ f(x) = \text{sign}(w \cdot x + b) \]

\[ x = (x_1, x_2) \quad w = (1,1) \quad b = -3 \]
Linear classification

\[ f(x) = \text{sign}(w \cdot x + b) \]
H3 (green) doesn't separate the two classes. H1 (blue) does, with a small margin and H2 (red) with the maximum margin.
Linear classification

• Learning a linear classifier from data:
  – Minimizing **training error**
    • NP-complete

  – Minimizing **sum of error squares**
    • Suboptimal, yet can be easy and fun: e.g. the perceptron.

  – Maximizing the **margin**
    • Doable and well-founded by statistical learning theory
The margin
Maximum-margin hyperplane and margins for an SVM trained with samples from two classes. Samples on the margin are called the support vectors.
The margin

• For any point $x$, its distance to the hyperplane:

$$\text{Distance} = \frac{|w \cdot x + b|}{\|w\|}$$

• Assuming all points are classified correctly:

$$\text{Distance} = \frac{y_i(w \cdot x + b)}{\|w\|}$$

• The margin is then:

$$\text{Margin} = \min_i \frac{y_i(w \cdot x_i + b)}{\|w\|}$$
Maximal margin training

- Find \((w, b)\) such that Margin is maximal, can be shown to be equivalent with:

\[
\hat{w} = \arg \min_{w, b} \|w\|^2
\]

subject to

\[
\forall i: y_i (w \cdot x_i + b) \geq 1
\]

- This is doable using efficient optimization algorithms.
Soft margin training

• If data is not linearly separable:

\[
\hat{w} = \arg\min_{w,b} \sum \varepsilon_i + \lambda \|w\|^2
\]

subject to

\[
\forall i: y_i(w \cdot x_i + b) \geq 1 - \varepsilon_i, \quad \varepsilon_i \geq 0
\]

• It is called the **Support Vector Machine (SVM)**.

• It can be generalized to regression tasks.
Soft margin training

• In more general form:

$$\hat{w} = \arg\min_{w,b} \sum \varepsilon_i + \lambda \|w\|^2$$

where

$$\varepsilon_i = (1 - y_i (w \cdot x_i + b))^+$$

is the **hinge loss**.
Regularization

• There are many algorithms which essentially look as follows:

For given data D find a model $M$, which minimizes $\text{Error}(M,D) + \text{Complexity}(M)$

• An SVM is a linear model, which minimizes $\text{Hinge loss} + l_2$-norm penalty
Going nonlinear

• But a linear classifier is so linear!
Solution: a nonlinear map

• Instead of classifying points $x$, we’ll classify points $\phi(x)$ in a higher-dimensional space.
Mapping Example

- map data points into feature space with some function $\phi$
- e.g.:
  - $\phi : \mathbb{R}^2 \rightarrow \mathbb{R}^3$
  - $(x_2, x_2) \rightarrow (z_1, z_2, z_3) := (x_1^2, \sqrt{2}x_1x_2, x_2^2)$

- hyperplane $\langle w \cdot z \rangle = 0$, as a function of $x$:

$$w_1x_1^2 + w_2\sqrt{2}x_1x_2 + w_3x_2^2 = 0$$
Kernel methods (not density!)

• For nearly any linear classifier:
  \[ f(x) = \text{sign}(w \cdot x + b) \]

• Trained on a dataset
  \[ D = \{ (x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n) \} \]

• The resulting vector \( w \) can be represented as:
  \[ \hat{w} = \sum_i \alpha_i x_i \]

• Which means:
  \[ f(x) = \text{sign}(\sum_i \alpha_i x_i \cdot x + b) \]
\[ \mathbf{a} \cdot \mathbf{b} = \sum_{i=1}^{n} a_i b_i = a_1 b_1 + a_2 b_2 + \cdots + a_n b_n \]
Kernel methods (not density!)

\[ f(x) = \text{sign}(w \cdot x + b) \]

\[ f(x) = \text{sign}(\sum \alpha_i x_i \cdot x + b) \]

\[ f(x) = \text{sign}(\sum \alpha_i \phi(x_i) \cdot \phi(x) + b) \]

\[ f(x) = \text{sign}(\sum \alpha_i K(x_i, x) + b) \]

\[ K(x_i, x) = \phi(x_i) \cdot \phi(x) \]
Kernel methods (not density!)

\[ f(x) = \text{sign}(\sum_{i} \alpha_i K(x_i, x) + b) \]

- Function K is called a kernel, it measures similarity between objects.
- The computation of \( \phi(x) \) is unnecessary.
- You can use any type of data.
- Your method is nonlinear.
- Any linear method can be kernelized.
- Kernels can be combined.
Example Kernel Trick

\[ \vec{x} = (x_1, x_2) \]

\[ \vec{z} = (z_1, z_2) \]

\[ K(x, z) = \langle \vec{x} \cdot \vec{z} \rangle^2 \]

\[
K(x, z) = \langle \vec{x} \cdot \vec{z} \rangle^2 \\
= (x_1z_1 + x_2z_2)^2 \\
= (x_1^2z_1^2 + 2x_1z_1x_2z_2 + x_2^2z_2^2) \\
= \langle (x_1^2, \sqrt{2}x_1x_2, x_2^2) \cdot (z_1^2, \sqrt{2}z_1z_2, z_2^2) \rangle \\
= \langle \phi(\vec{x}) \cdot \phi(\vec{z}) \rangle
\]

mapping function \( \phi \) fused in \( K \)

\[ \rightarrow \text{implicit } \phi(\vec{x}) = (x_1^2, \sqrt{2}x_1x_2, x_2^2) \]
Typical Kernels

- **Polynomial Kernel**
  \[ K(x, z) = (\langle x \cdot z \rangle + \theta)^d, \quad \text{for } d \geq 0 \]

- **Radial Basis Function** (Gaussian Kernel)
  \[ K(x, z) = e^{-\frac{||x-z||^2}{2\sigma^2}} \quad \|x\| := \sqrt{\langle x \cdot x \rangle} \]

- **(Sigmoid Kernel)**
  \[ K(x, z) = \tanh(\eta \langle x \cdot z \rangle + \theta) \]

- **Inverse multi-quadric**
  \[ K(x, z) = \frac{1}{\sqrt{\|x - z\|^2 2\sigma^2 + c^2}} \]
Making Kernels from Kernels

- create complex Kernels by combining simpler ones
- Closure Properties:

\[
\begin{align*}
K(x, z) &= c \cdot K_1(x, z) \\
K(x, z) &= c + K_1(x, z) \\
K(x, z) &= K_1(x, z) + K_2(x, z) \\
K(x, z) &= K_1(x, z) \cdot K_2(x, z) \\
K(x, z) &= f(x) \cdot f(z)
\end{align*}
\]

if \( K_1 \) and \( K_2 \) are kernels, \( \forall f : X \rightarrow \mathbb{R} \), and \( c > 0 \)
Separable classification with Radial Basis kernel functions in different space. Left: original space. Right: feature space.
Summary

• SVM:
  – A maximal margin linear classifier.
  – A linear model, which minimizes Hinge loss + $l_2$-norm penalty
  – Kernelizable

• Kernel methods:
  – An easy and elegant way of “plugging-in”
    • nonlinearity
    • different data types
ML Gallery


- A Collection of example data sets and illustration of different ML algorithms performance on them
Scikit learn
scikit-learn

http://scikit-learn.org/

- Simple and efficient tools for data mining and data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license
A Must software

- WEKA
- Mldemos
- R
- ...
Jaak Vilo and other authors

UT: Data Mining 2009

scikit-learn
Machine Learning in Python

- Simple and efficient tools for data mining and data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Classification
Identifying to which set of categories a new observation belong to.
Applications: Spam detection, Image recognition.
Algorithms: SVM, nearest neighbors, random forest, ...
— Examples

Regression
Predicting a continuous value for a new example.
Applications: Drug response, Stock prices.
Algorithms: SVR, ridge regression, Lasso, ...
— Examples

Clustering
Automatic grouping of similar objects into sets.
Applications: Customer segmentation, Grouping experiment outcomes
Algorithms: k-Means, spectral clustering, mean-shift, ...
— Examples

Dimensionality reduction
Reducing the number of random variables to consider.
Applications: Visualization, Increased efficiency
Algorithms: PCA, Isomap, non-negative matrix factorization.
— Examples

Model selection
Comparing, validating and choosing parameters and models.
Goal: Improved accuracy via parameter tuning
Modules: grid search, cross validation, metrics.
— Examples

Preprocessing
Feature extraction and normalization.
Application: Transforming input data such as text for use with machine learning algorithms.
Modules: preprocessing, feature extraction.
— Examples
PyML – machine learning in Python

PyML is an interactive object oriented framework for machine learning written in Python. PyML focuses on SVMs and other kernel methods. It is supported on Linux and Mac OS X.

News

- Added wrapper for liblinear in version 0.7.8. This allows extremely fast training of linear SVMs.

Features

- Classifiers: support vector machines, nearest neighbor, ridge regression
- Multi-class methods (one-against-rest and one-against-one)
- Feature selection (filter methods, RFE)
- Model selection
- Preprocessing and normalization
- Syntax for combining classifiers
- Classifier testing (cross-validation, error rates, ROC curves)
- Various kernels for biological sequences (several variants of the spectrum kernel, and the weighted-degree kernel).
mlpy - Machine Learning Python

**mlpy** is a Python module for Machine Learning built on top of NumPy/SciPy and the GNU Scientific Libraries.

mlpy provides a wide range of state-of-the-art machine learning methods for *supervised* and *unsupervised* problems and it is aimed at finding a reasonable compromise among modularity, maintainability, reproducibility, usability and efficiency. mlpy is *multiplatform*, it works with *Python 2* and *3* and it is *Open Source*, distributed under the GNU General Public License version 3.

**If you use mlpy, please cite:**


mlpy was used in the following applications.

**Features**

**Regression:** Least Squares, Ridge Regression, Last Angle Regression, Elastic Net, Kernel Ridge Regression, Support Vector Machines (SVR), Partial Least Squares (PLS)

**Classification:** Linear Discriminant Analysis (LDA), Basic Perceptron, Elastic Net, Logistic Regression, (Kernel) Support Vector Machines (SVM), Diagonal Linear Discriminant Analysis (DLDA), Golub Classifier, Parzen-based, (kernel) Fisher Discriminant Classifier k-Nearest-Neighbor, Iterative RELIEF, Classification Tree, Maximum Likelihood Classifier

**Clustering:** Hierarchical Clustering, Memory-saving Hierarchical Clustering, k-means
Summary summary

• **“Machine learning”**
  – Terminology, foundations, general framework.

• **Supervised machine learning**
  – Basic ideas, algorithms & toy examples.

• **Statistical challenges**
  – Learning theory, bias-variance, consistency,…

• **State of the art techniques**
  – SVM, kernel methods.
Machine learning is important.
Questions?