Data Mining MTAT.03.183
Introduction to Machine Learning

Jaak Vilo
2014 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 income > $\theta_1$ AND 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
• ...
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
  \[ y = g ( x \mid \theta ) \]
  \( g ( ) \) 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</td>
<td>Winter of 2003 was warm</td>
</tr>
<tr>
<td>Summer of 2004 was</td>
<td>Winter of 2004 was cold</td>
</tr>
<tr>
<td>Summer of 2005 was</td>
<td>Winter of 2005 was cold</td>
</tr>
<tr>
<td>Summer of 2006 was</td>
<td>Winter of 2006 was warm</td>
</tr>
<tr>
<td>Summer of 2007 was</td>
<td>Winter of 2007 was cold</td>
</tr>
<tr>
<td>Summer of 2008 was</td>
<td>Winter of 2008 was warm</td>
</tr>
<tr>
<td>Summer of 2009 was</td>
<td>Winter of 2009 was cold</td>
</tr>
<tr>
<td>Summer of 2010 was</td>
<td>Winter of 2010 was cold</td>
</tr>
<tr>
<td>Summer of 2011 was</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,</td>
<td>I get a C</td>
</tr>
<tr>
<td>Professor=😆</td>
<td></td>
</tr>
<tr>
<td>Study=slack,</td>
<td>I get an A</td>
</tr>
<tr>
<td>Professor=😆</td>
<td></td>
</tr>
<tr>
<td>Study=hard,</td>
<td>I get an A</td>
</tr>
<tr>
<td>Professor=😆</td>
<td></td>
</tr>
<tr>
<td>Study=slack,</td>
<td>I get a D</td>
</tr>
<tr>
<td>Professor=😆</td>
<td></td>
</tr>
<tr>
<td>Study=slack,</td>
<td>I get an A</td>
</tr>
<tr>
<td>Professor=😆</td>
<td></td>
</tr>
<tr>
<td>Study=slack,</td>
<td>I get an A</td>
</tr>
<tr>
<td>Professor=😆</td>
<td></td>
</tr>
<tr>
<td>Study=hard,</td>
<td>I get an A</td>
</tr>
<tr>
<td>Professor=😆</td>
<td></td>
</tr>
<tr>
<td>Study=slack,</td>
<td>I get a B</td>
</tr>
<tr>
<td>Professor=😆</td>
<td></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**
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

Collect data/observations

Come up with theories

Better Theory
Science 😊
Supervised learning

• Formally,

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

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

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

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

5.11.2009
Classification

- $\mathcal{X} = \mathbb{R}^2$, $\mathcal{Y} = \{\text{blue, red}\}$.
- $D = \{((1.3, 0.8), \text{red}), ((2.5, 2.3), \text{blue}), \ldots \}$
- $f_D(x_1, x_2) =$ if $x_1 + x_2 > 3$ then blue else red.
Figure 4.2. Classification as the task of mapping an input attribute set $x$ into its class label $y$. 
The “Dumb User” Perspective

Weka, RapidMiner, MSSSAS, Clementine, SPSS, R, ...

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

\[ f(x) = \sum_{i} w_i x_i \]

Validation
Classification demo: Iris dataset

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

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

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

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

<table>
<thead>
<tr>
<th>Class</th>
<th>setosa</th>
<th>versic.</th>
<th>virg.</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>TP Rate</td>
<td>1</td>
<td>0.98</td>
<td>0.96</td>
<td>0.98</td>
</tr>
<tr>
<td>FP Rate</td>
<td>0</td>
<td>0.02</td>
<td>0.01</td>
<td>0.01</td>
</tr>
<tr>
<td>Precision</td>
<td>1</td>
<td>0.961</td>
<td>0.98</td>
<td>0.98</td>
</tr>
<tr>
<td>Recall</td>
<td>1</td>
<td>0.98</td>
<td>0.96</td>
<td>0.98</td>
</tr>
<tr>
<td>F-Measure</td>
<td>1</td>
<td>0.97</td>
<td>0.97</td>
<td>0.98</td>
</tr>
<tr>
<td>ROC Area</td>
<td>1</td>
<td>0.99</td>
<td>0.99</td>
<td>0.99</td>
</tr>
</tbody>
</table>

Machine learning :: Introduction
Validation

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

<table>
<thead>
<tr>
<th></th>
<th>1.00</th>
<th>0.98</th>
<th>0.96</th>
</tr>
</thead>
<tbody>
<tr>
<td>TP Rate</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FP Rate</td>
<td></td>
<td>0.02</td>
<td></td>
</tr>
<tr>
<td>Precision</td>
<td>1.00</td>
<td>0.961</td>
<td></td>
</tr>
<tr>
<td>Recall</td>
<td>1.00</td>
<td>0.98</td>
<td></td>
</tr>
<tr>
<td>F-Measure</td>
<td>1.00</td>
<td>0.97</td>
<td></td>
</tr>
<tr>
<td>ROC Area</td>
<td>1.00</td>
<td>0.99</td>
<td></td>
</tr>
</tbody>
</table>

5.11.2009 Machine learning :: Introduction
Validation

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

“True positives” “False positives”

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))
## Classification summary

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

<table>
<thead>
<tr>
<th></th>
<th>Positives</th>
<th>Negatives</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Predicted = Yes</td>
<td>Predicted = No</td>
</tr>
<tr>
<td>Actual = Yes</td>
<td>True positives (TP)</td>
<td>False negatives (FN)</td>
</tr>
<tr>
<td></td>
<td>(Type II, (\beta)-error)</td>
<td>(Type I, (\alpha)-error)</td>
</tr>
<tr>
<td>Actual = No</td>
<td>False positives (FP)</td>
<td>True negatives (FN)</td>
</tr>
<tr>
<td></td>
<td>(Type I, (\alpha)-error)</td>
<td></td>
</tr>
</tbody>
</table>

- **Precision**: \(\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}\)
- **Recall**: \(\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}\)
- **Accuracy**: \(\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}}\)
- **F-measure**: \(\text{F-measure} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}\)
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

Split

Training set

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

• After split on outlook:

\[
\begin{align*}
H &= 0.97 & H &= 0 & H &= 0.97 \\
\text{outlook} & \quad \text{sunny} \quad \text{overcast} \quad \text{rainy} \\
\text{yes} & \quad \text{yes} & \quad \text{yes} & \quad \text{yes} & \quad \text{yes} & \quad \text{yes} & \quad \text{no} & \quad \text{no} \\
\text{yes} & \quad \text{yes} & \quad \text{yes} & \quad \text{yes} & \quad \text{yes} & \quad \text{yes} & \quad \text{no} & \quad \text{no}
\end{align*}
\]

\[
0.97 \left( \frac{5}{14} \right) + 0 \left( \frac{4}{14} \right) + 0.97 \left( \frac{5}{14} \right) = 0.69
\]

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

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

• The bads
  – Rather ad-hoc
  – Can overfit unless properly pruned
  – Not the best model for all classification tasks

• Improvements:
  – Random Forest – multiple trees, majority voting
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
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>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>Yes</td>
</tr>
<tr>
<td>Yes</td>
</tr>
<tr>
<td>No</td>
</tr>
</tbody>
</table>
Shall we play tennis today?

- Probabilistic model:

  \[
  \begin{array}{c}
  \text{PlayTennis} \\
  \hline
  \text{No} & \text{No} & \text{Yes} & \text{Yes} & \text{Yes} & \text{No} & \text{Yes} & \text{Yes} & \text{Yes} & \text{Yes} & \text{Yes} & \text{No} \\
  \hline
  \end{array}
  \]

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

  \[\Rightarrow \text{Yes}\]
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>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:**

   \[ P(C_1 \mid X_1, X_2, X_3, \ldots) > P(C_2 \mid X_1, X_2, X_3, \ldots) \]

   \[ \Rightarrow \text{predict } C_1 \]
The Bayes Classifier

In general:

1. $P(C_1 | X) > P(C_2 | X)$

$\Rightarrow$ predict $C_1$
The Bayes Classifier

In general:

1. \( P(C_1 | X) > P(C_2 | X) \)  
   \( \rightarrow \) 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 \( C_1 \)
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 \( C_1 \)
The Bayes Classifier

In general:

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

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

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

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

3. $P(X \mid C_1)P(C_1) > P(X \mid C_2)P(C_2) \implies$ 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>No</td>
</tr>
<tr>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
</tbody>
</table>

\[
P(\text{High,Weak}) = \frac{4}{14}
\]
\[
P(\text{Yes | High,Weak}) = \frac{2}{4}
\]
\[
P(\text{No | High,Weak}) = \frac{2}{4}
\]

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

...
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

\[ \frac{P(X | C_1)}{P(X | C_2)} > \frac{P(C_2)}{P(C_1)} \]

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

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
   
   $$\text{lscore} = \sum_i l(i, x_i)$$

   If $\text{lscore} > \log \frac{P(C_2)}{P(C_1)}$ 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

$$w = \frac{\sum_{i} y_i x_i}{\sum_{i} x_i^2}$$
Linear regression

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

• The goods
  – Simple, easy
  – Interprettable, 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
  - ...

Jaak Vilo and other authors

UT: Data Mining 2009
Robust Regression

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

$$\min_{\beta} e_{(\alpha N)}^2 \quad \text{where} \quad e = X\beta - y$$

$$e_{(j-1)}^2 \leq e_{(j)}^2 \quad j = 2..N$$

least mean squares
least median of squares
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
K-Nearest Neighbors

- One-nearest neighbor
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 looses 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

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

Probabilistic models
- Naïve Bayes
- \[ \prod P(X_i | C_1) \leq P(C_2) \leq P(C_1) \]
- Graphical models
- Regression models
- Density estimation

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

\[ \arg \min_w E(w) \]
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
...

Probabilistic models
Naïve Bayes
Graphical models
Regression models
Density estimation
...

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

Kernel densities
SVM, RBF
...

12.11.2009
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

$training\ error = \frac{1}{n} \sum_{i}[g_S(x_i) \neq y_i]$
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).
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

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

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

$x_1$ vs $x_2$
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.
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:
  \[ f(x) = \text{sign}(\sum_i \alpha_i x_i \cdot x + b) \]
Kernel methods (not density!)

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

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

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

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

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

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

- 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.
\[ a \cdot b = \sum_{i=1}^{n} a_i b_i = a_1 b_1 + a_2 b_2 + \cdots + a_n b_n \]
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
A Must software

- WEKA
- Mldemos
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?