Machine Learning for Object Recognition

Artem Lind & Aleskandr Tkachenko
Outline

• Problem overview
• Classification demo
• Examples of learning algorithms
  – Probabilistic modeling
    • Bayes classifier
  – Maximum margin classification
    • SVM
Object Recognition as a Classification Task

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.

- $\mathcal{X} = \mathbb{R}^2$, $\mathcal{Y} = \{\text{blue, red}\}$.
- $D = \{((1.3, 0.8), \text{red}), ((2.5, 2.3), \text{blue}), \ldots\}$
- $f_D(x_1, x_2) = \text{if } x_1 + x_2 > 3 \text{ then blue else red.}$
Typical Workflow

1. Preprocessing Images
2. Extracting features
3. Learning classifier
4. Assessing results
Object representation

- Vectors of
  - quantitative descriptors (length, weight, area)
  - raw pixel values

- String and trees of structural descriptors
  - Capture spatial relationships between features
Object representation

• Vectors of
  – quantitative descriptors (length, weight, area)
  – raw pixel values

• String and trees of structural descriptors
  – Capture spatial relationships between features
Classification demo: Iris dataset

![Iris Flower](image)

- **Sepal**
- **Petal**
Classification demo: Iris dataset

- Two classes: Iris-versicolor, Iris-virginica
- Two features: sepal length, petal length
Iris dataset: feature vector representation
Classification demo: linear classification
Classification demo: linear classifier

How good is my classifier?
## Classifier evaluation

### Confusion Matrix

<table>
<thead>
<tr>
<th></th>
<th>a = Iris-versicolor</th>
<th>b = Iris-virginica</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>44</td>
<td>8</td>
</tr>
<tr>
<td>b</td>
<td>6</td>
<td>42</td>
</tr>
</tbody>
</table>

- **True positives**: 44
- **False negatives**: 8
- **False positives**: 6
- **True negatives**: 42
Classifier evaluation

Confusion Matrix

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>&lt;- classified as</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>44</td>
<td>6</td>
<td>a = Iris-versicolor</td>
</tr>
<tr>
<td>b</td>
<td>8</td>
<td>42</td>
<td>b = Iris-virginica</td>
</tr>
</tbody>
</table>

True positives
False negatives
False positives
True Negatives

Precision $= \frac{TP}{TP + FP}$
Recall $= \frac{TP}{TP + FN}$
Accuracy $= \frac{TP + TN}{FP + FN}$
F-Measure $= \text{harmonic\_mean}(\text{Precision}, \text{Recall})$
Classifier evaluation

Confusion Matrix

\[
\begin{array}{c|cc}
& a & b \\
\hline
44 & 6 & \text{a = Iris-versicolor} \\
8 & 42 & \text{b = Iris-virginica}
\end{array}
\]

Summary

Correctly Classified Instances 86 86 %
Incorrectly Classified Instances 14 14 %
Total Number of Instances 100

Detailed Accuracy By Class

<table>
<thead>
<tr>
<th>TP Rate</th>
<th>FP Rate</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
<th>ROC</th>
<th>Area</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.88</td>
<td>0.16</td>
<td>0.846</td>
<td>0.88</td>
<td>0.863</td>
<td>0.949</td>
<td></td>
<td>Iris-versicolor</td>
</tr>
<tr>
<td>0.84</td>
<td>0.12</td>
<td>0.875</td>
<td>0.84</td>
<td>0.857</td>
<td>0.949</td>
<td></td>
<td>Iris-virginica</td>
</tr>
</tbody>
</table>
Classifier evaluation

• Most training algorithms optimize \textit{Accuracy / Precision / Recall} for the given data

• However, we want classifier to perform well on “unseen” data
  – This makes algorithms and theory way more complicated.
  – This makes validation somewhat more complicated.
Proper validation

• Split the data:
  - Training set
  - Test set

• Crossvalidation
  - If the data is scarce
Common workflow – Summary:

• Get a decent dataset
• Identify discriminative features
• Train your classifier on the training set
• Validate on the test set
Classifiers

• Probabilistic modeling
  – Bayes classifier

• Margin maximization
  – Support Vector Machines
Bayes classifier
(for binary classification)

Learning:
Estimate from the training data $P(\text{Class} \mid X)$

Classifying:
Bayes Decision Rule:
Predict $C_1$, if $P(\text{C}_1 \mid X) > P(\text{C}_2 \mid X)$; otherwise $C_2$
Bayes classifier

Predict $C_1$, if $P(C_1 | X) > P(C_2 | X)$; otherwise $C_2$

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}.$$ 

$$P(X|C_1) P(C_1) / P(X) > P(X|C_2) P(C_2) / P(X)$$

$$P(X|C_1) P(C_1) > P(X|C_2) P(C_2)$$
Bayes classifier

Predict $C_1$, if $P(X|C_1)P(C_1) > P(X|C_2)P(C_2)$; otherwise $C_2$
Bayes classifier

• If $P(X|Class)$ and $P(Class)$ are known, then the classifier is optimal.

• In practice, distributions $P(X|Class)$ and $P(Class)$ are unknown and need to be estimated from the data:
  – $P(Class)$:
    • Assign equal probability to all classes
    • Use prior knowledge
    • $P(C) = \#\text{examples}_{\text{in}_C}/\#\text{examples}$
  – $P(X|Class)$:
    • Some well-behaved distribution expressed in an analytical form.
    • Parameters are estimated based on data for each class.
    • The closer this assumption is to reality, the closer the Bayes classifier approaches the optimum.
Bayes classifier for Gaussian patterns

\[ p(x \mid c_i) = N(\mu_i, \sigma_i) \]

\[ p(x \mid c_i)P(c_i) = \frac{1}{\sqrt{2\pi\sigma_i}} e^{-\frac{(x-\mu_i)^2}{2\sigma_i^2}} P(c_i), \quad i = 1, 2 \]

\( \mu_i \) and \( \sigma_i^2 \) are sample mean and variance for the class \( i \).
Example from the book

FIGURE 12.13 (a) Multispectral image. (b) Printout of machine classification results using a Bayes classifier. (Courtesy of the Laboratory for Applications of Remote Sensing, Purdue University.)
Summary

• **Advantages:**
  – Creates linear boundaries which are simple to compute.
  – Easy to implement.

• **Disadvantages:**
  – is based on a single prototype per class (class center) which is often insufficient in practice.
  – Usually won’t work well for too many classes.
Support Vector Machines
SVM: Maximum margin classification

Find a linear classifier to separate the two datasets
SVM: Maximum margin classification

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

Linearly separable data
SVM: Maximum margin classification

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

Which one?

The one furthest from both sets
SVM: Maximum margin classification

\[ f(x) = \text{sign}(w \cdot x + b) \]
SVM: Maximum margin classification

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

Maximize the margin ...
SVM: Maximum margin classification

- Hyperplane:

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

- 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:

\[ \text{Margin} = \min \frac{y_i(w \cdot x + b)}{\|w\|} \]
Maximizing the margin

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

\[
\min_w \frac{1}{2} w^T w \\
\text{s.t.} \quad y_i(w^T x_i + b) \geq 1 \\
i = 1 \ldots n
\]

• This is doable using efficient optimization algorithms.
SVM: Non separable case

\[
\min_{w,b} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{N} \xi_i \quad \text{subject to:}
\]

\[
y_i (x_i^T w + b) \geq 1 - \xi_i \quad \forall i, \quad \xi_i > 0 \quad \forall i
\]

\(\xi_i\) are called slack variables

\(C\) constant, penalizes errors
SVM: How to choose $C$ parameter?

- $C$ parameter penalizes training points within the margin
- Large $C$-value can lead to over-fitting.
- Cross-validation + Grid search
- Ad hock: Find $C$-values which give you zero errors on the training set.

$$\min_{w,b} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{N} \xi_i$$
Nonlinear case
Solution: a nonlinear map 😊

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

• 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:
  \[ \widehat{w} = \sum_i \alpha_i x_i \]

• Which means:
  \[ f(x) = \text{sign}(\sum_i \alpha_i x_i \cdot x + b) \]
Kernel methods

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

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

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

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

\[ K(x_i, x) = \phi(x_i) \cdot \phi(x) \]
Kernel methods

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

- 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.
Basic kernels

- linear: $K(x_i, x_j) = x_i^T x_j$.
- polynomial: $K(x_i, x_j) = (\gamma x_i^T x_j + r)^d, \gamma > 0$.
- radial basis function (RBF): $K(x_i, x_j) = \exp(-\gamma ||x_i - x_j||^2), \gamma > 0$.

How to choose the kernel?
SVM: more than 2 classes

Bottom-Up level

A or B or C or D

A or B

A
B

C or D

C
D

Training: \( k \binom{k-1}{2} \)
Classification: \( k-1 \)

1 vs. All

A / B,C,D

B

A,C,D

C

A,B,D

D

A,B,C

Training: \( k \)
Classification: \( k \)
SVM in practice

- Transform data to the format of an SVM package
- Conduct simple scaling on the data
- Consider the RBF kernel $K(x, y) = e^{-\gamma \|x - y\|^2}$
- Use cross-validation to find the best parameter $C$ and $\gamma$
- Use the best parameter $C$ and $\gamma$ to train the whole training set
- Test
SVM Summary

• SVMs are maximum margin classifiers.
• SVMs can handle non-separable data.
• SVMs are known to perform well in high dimensional problems with few examples.
• Depending on the kernel, SVMs can be slow during classification
• Kernelizable
References

• Massachusetts Institute of technology course:
  – 9.913 Pattern Recognition for Machine Vision

• A Practical Guide to Support Vector Classication

• Konstantin Tretyakov slides on datamining
Demo