Classification
16-385 Computer Vision (Kris Kitani)
Carnegie Mellon University
typical perception pipeline

representation

- Nearest Neighbor classifier
- Naive Bayes classifier
- Support Vector Machine

classifier

output
Nearest Neighbor

Naive Bayes

Support Vector Machine
Distribution of data from two classes
Which class does $q$ belong too?
Distribution of data from two classes

Look at the neighbors
K-nearest neighbor
K-Nearest Neighbor (KNN) Classifier

Non-parametric pattern classification approach

Consider a two class problem where each sample consists of two measurements \((x,y)\).

For a given query point \(q\), assign the class of the nearest neighbor

Compute the \(k\) nearest neighbors and assign the class by majority vote.

\[ k = 1 \]

\[ k = 3 \]
Nearest Neighbor is competitive

MNIST Digit Recognition

- Handwritten digits
- 28x28 pixel images: \( d = 784 \)
- 60,000 training samples
- 10,000 test samples

Test Error Rate (%)

<table>
<thead>
<tr>
<th>Method</th>
<th>Error Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear classifier (1-layer NN)</td>
<td>12.0</td>
</tr>
<tr>
<td>K-nearest-neighbors, Euclidean</td>
<td>5.0</td>
</tr>
<tr>
<td>K-nearest-neighbors, Euclidean, deskewed</td>
<td>2.4</td>
</tr>
<tr>
<td>K-NN, Tangent Distance, 16x16</td>
<td>1.1</td>
</tr>
<tr>
<td>K-NN, shape context matching</td>
<td>0.67</td>
</tr>
<tr>
<td>1000 RBF + linear classifier</td>
<td>3.6</td>
</tr>
<tr>
<td>SVM deg 4 polynomial</td>
<td>1.1</td>
</tr>
<tr>
<td>2-layer NN, 300 hidden units</td>
<td>4.7</td>
</tr>
<tr>
<td>2-layer NN, 300 HU, [deskewing]</td>
<td>1.6</td>
</tr>
<tr>
<td>LeNet-5, [distortions]</td>
<td>0.8</td>
</tr>
<tr>
<td>Boosted LeNet-4, [distortions]</td>
<td>0.7</td>
</tr>
</tbody>
</table>

Yann LeCunn
Pros

• simple yet effective

Cons

• search is expensive (can be sped-up)
• storage requirements
• difficulties with high-dimensional data
What is the best distance metric between data points?

- Typically Euclidean distance
- Locality sensitive distance metrics
- Important to normalize. Dimensions have different scales

How many K?

- Typically k=1 is good
- Cross-validation
Distance metrics

\[ D(x, y) = \sqrt{(x_1 - y_1)^2 + \cdots + (x_N - y_N)^2} \quad \text{Euclidean} \]

\[ D(x, y) = \frac{x \cdot y}{\|x\| \|y\|} = \frac{x_1 y_1 + \cdots + x_N y_N}{\sqrt{\sum_n x_n^2} \sqrt{\sum_n y_n^2}} \quad \text{Cosine} \]

\[ D(x, y) = \frac{1}{2} \sum_n \frac{(x_n - y_n)^2}{(x_n + y_n)} \quad \text{Chi-squared} \]
Distribution of data from two classes
Which class does $q$ belong too?
Distribution of data from two classes

- Learn parametric model for each class
- Compute probability of query
Naive Bayes
This is called the posterior: the probability of a class $z$ given the observed features $X$.

\[
p(z | X)
\]

For classification, $z$ is a discrete random variable (e.g., car, person, building).

$X$ is a set of observed feature (e.g., features from a single image).

(it’s a function that returns a single probability value)
This is called the posterior: the probability of a class $z$ given the observed features $X$

$$p(z|x_1, \ldots, x_N)$$

For classification, $z$ is a discrete random variable (e.g., car, person, building)

Each $x$ is an observed feature (e.g., visual words)

(it’s a function that returns a single probability value)
Recall:

The posterior can be decomposed according to Bayes’ Rule

\[ p(A|B) = \frac{p(B|A)p(A)}{p(B)} \]

In our context...

\[ p(z|x_1, \ldots, x_N) = \frac{p(x_1, \ldots, x_N|z)p(z)}{p(x_1, \ldots, x_N)} \]
The naive Bayes’ classifier is solving this optimization

\[
\hat{z} = \arg \max_{z \in \mathcal{Z}} p(z | X)
\]

MAP (maximum a posteriori) estimate

\[
\hat{z} = \arg \max_{z \in \mathcal{Z}} \frac{p(X | z)p(z)}{p(X)}
\]

Bayes’ Rule

\[
\hat{z} = \arg \max_{z \in \mathcal{Z}} p(X | z)p(z)
\]

Remove constants
To optimize this...

\[ \hat{z} = \arg \max_{z \in \mathcal{Z}} p(z|X) \]

We need to compute this

\[ p(z|x_1, \ldots, x_N) = \frac{p(x_1, \ldots, x_N|z)p(z)}{p(x_1, \ldots, x_N)} \]

Compute the likelihood...
A naive Bayes’ classifier assumes all features are \textit{conditionally independent}.

\[
p(x_1, \ldots, x_N | z) = p(x_1 | z)p(x_2, \ldots, x_N | z) \\
= p(x_1 | z)p(x_2 | z)p(x_3, \ldots, x_N | z) \\
= p(x_1 | z)p(x_2 | z) \cdots p(x_N | z)
\]

\[\text{Recall:}\]

\[
p(x, y) = p(x | y)p(y) \quad p(x, y) = p(x)p(y)
\]
Graphical model visualization

$p(z)$

class (random variable)

$z$

edge (dependence relation)

$p(x_1 | z)$  
$p(x_2 | z)$  
$p(x_3 | z)$  
$p(x_4 | z)$

features (random variable)

$x_1$  
$x_2$  
$x_3$  
$x_4$
To compute the MAP estimate

Given (1) a set of known parameters

\[ p(z) \quad p(x|z) \]

(2) observations

\[ \{x_1, x_2, \ldots, x_N\} \]

Compute which \( z \) has the largest probability

\[ \hat{z} = \arg \max_{z \in Z} p(z) \prod_{n} p(x_n|z) \]
\[ p(X | z) = \prod_{v} p(x_v | z)^{c(w_v)} \]

\[ = (0.09)^1 (0.55)^6 \cdots (0.09)^1 \]

Numbers get really small so use log probabilities

\[ \log p(X | z = \text{‘grandchallenge’}) = -2.42 - 3.68 - 3.43 - 2.42 - 0.07 - 0.07 - 0.07 - 2.42 = -14.58 \]

\[ \log p(X | z = \text{‘softrobot’}) = -7.63 - 9.37 - 15.18 - 2.97 - 0.02 - 0.01 - 0.02 - 2.27 = -37.48 \]

* typically add pseudo-counts (0.001)
** this is an example for computing the likelihood, need to multiply times prior to get posterior
log $p(X|z=\text{grand challenge}) = -14.58$

log $p(X|z=\text{bio inspired}) = -37.48$

* typically add pseudo-counts (0.001)

** this is an example for computing the likelihood, need to multiply times prior to get posterior
Nearest Neighbor

Naive Bayes

Support Vector Machine
Distribution of data from two classes
Distribution of data from two classes

Which class does q belong to?
Distribution of data from two classes

Learn the decision boundary
Support Vector Machine
First we need to understand hyperplanes...
Hyperplanes (lines) in 2D

\[ w_1 x_1 + w_2 x_2 + b = 0 \]

A line can be written as dot product plus a bias

\[ \mathbf{w} \cdot \mathbf{x} + b = 0 \]

Another version, add a weight 1 and push the bias inside

\[ \mathbf{w} \cdot \mathbf{x} = 0 \]

\[ \mathbf{w} \in \mathcal{R}^2 \]

\[ \mathbf{w} \in \mathcal{R}^3 \]
Hyperplanes (lines) in 2D

\[ \mathbf{w} \cdot \mathbf{x} + b = 0 \quad \text{(offset/bias outside)} \quad \mathbf{w} \cdot \mathbf{x} = 0 \quad \text{(offset/bias inside)} \]

\[ w_1 x_1 + w_2 x_2 + b = 0 \]
Hyperplanes (lines) in 2D

\[ \mathbf{w} \cdot \mathbf{x} + b = 0 \quad \text{(offset/bias outside)} \quad \mathbf{w} \cdot \mathbf{x} = 0 \quad \text{(offset/bias inside)} \]

\[ w_1 x_1 + w_2 x_2 + b = 0 \]

**Important property:**
Free to choose any normalization of \( \mathbf{w} \)

The line
\[ w_1 x_1 + w_2 x_2 + b = 0 \]
and the line
\[ \lambda(w_1 x_1 + w_2 x_2 + b) = 0 \]
define the same line.
What is the distance to origin?
(hint: use normal form)
You get the normal form distance to origin by scaling \( \mathbf{w} \cdot \mathbf{x} + b = 0 \) by \( \frac{1}{\|\mathbf{w}\|} \):

\[
x \cos \theta + y \sin \theta = \rho
\]
What is the distance between two parallel lines? (hint: use distance to origin)
distance between two parallel lines \( \frac{1}{\|w\|} \)

\[ w \cdot x + b = -1 \]

\[ w \cdot x + b = 0 \]

Difference of distance to origin

\[ \frac{b + 1}{\|w\|} - \frac{b}{\|w\|} = \frac{1}{\|w\|} \]
Now we can go to 3D ...

Hyperplanes (planes) in 3D

What happens if you change $b$?

Hyperplane equation: $w \cdot x + b = 0$

Dimensions of the vector $w$?

What are the dimensions of this vector?
Hyperplanes (planes) in 3D

\[ \mathbf{w} \cdot \mathbf{x} + b = -1 \]
What's the distance between these parallel planes?
Hyperplanes (planes) in 3D

\[ \frac{2}{\|w\|} \]

\[ w \cdot x + b = 0 \]

\[ w \cdot x + b = -1 \]

\[ w \cdot x + b = 1 \]
Support Vector Machine
What’s the best $w$?
What’s the best $w$?
What’s the best $w$?
What’s the best $w$?
What’s the best $w$?
What’s the best $w$?

Intuitively, the line that is the farthest from all interior points.
What’s the best $w$?

Maximum Margin solution:
most stable to perturbations of data
What’s the best $w$?

Want a hyperplane that is far away from ‘inner points’
Find hyperplane $\mathbf{w}$ such that ...

the gap between parallel hyperplanes $\frac{2}{\|\mathbf{w}\|}$ is maximized
Can be formulated as a maximization problem

\[
\max \frac{2}{\|w\|}
\]

subject to \( w \cdot x_i + b \geq +1 \) if \( y_i = +1 \) for \( i = 1, \ldots, N \)

\(\leq -1 \) if \( y_i = -1 \)

**What does this constraint mean?**

**Why is it +1 and -1?**

_label of the data point_
Can be formulated as a maximization problem

\[
\max_{\mathbf{w}} \frac{2}{\|\mathbf{w}\|}
\]

subject to \(\mathbf{w} \cdot \mathbf{x}_i + b \geq +1\) if \(y_i = +1\), \(\mathbf{w} \cdot \mathbf{x}_i + b \leq -1\) if \(y_i = -1\) for \(i = 1, \ldots, N\)

Equivalently,

Where did the 2 go?

\[
\min_{\mathbf{w}} \|\mathbf{w}\|
\]

subject to \(y_i (\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1\) for \(i = 1, \ldots, N\)

What happened to the labels?
‘Primal formulation’ of a linear SVM

\[
\min_w \|w\|
\]

Objective Function

subject to \( y_i(w \cdot x_i + b) \geq 1 \) for \( i = 1, \ldots, N \)

Constraints

This is a convex quadratic programming (QP) problem
(a unique solution exists)
‘soft’ margin
What’s the best $w$?
What’s the best $w$?

Very narrow margin
Separating cats and dogs

Very narrow margin
What’s the best $w$?

Intuitively, we should allow for some misclassification if we can get more robust classification.
What’s the best $w$?

Trade-off between the MARGIN and the MISTAKES (might be a better solution)
Adding slack variables \( \xi_i \geq 0 \)

\[
\frac{\xi_i}{\|w\|} > \frac{2}{\|w\|}
\]
‘soft’ margin

objective

\[
\min_{\mathbf{w}, \xi} \|\mathbf{w}\|^2 + C \sum_i \xi_i
\]

subject to

\[
y_i (\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i \\
\text{for } i = 1, \ldots, N
\]
The slack variable allows for mistakes, as long as the inverse margin is minimized.
‘soft’ margin

**objective**

\[
\min_{\mathbf{w},\xi} \|\mathbf{w}\|^2 + C \sum_{i} \xi_i
\]

**subject to**

\[
y_i (\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i
\]

for \( i = 1, \ldots, N \)

- Every constraint can be satisfied if slack is large
- \( C \) is a regularization parameter
  - Small \( C \): ignore constraints (larger margin)
  - Big \( C \): constraints (small margin)
- Still QP problem (unique solution)
C = Infinity  hard margin

SVM (L1) by Sequential Minimal Optimizer
Kernel: linear (-), C: Inf
Kernel evaluations: 971
Number of Support Vectors: 3
Margin: 0.0966
Training error: 0.00%
C = 10   soft margin

SVM (L1) by Sequential Minimal Optimizer
Kernel: linear (-), C: 10.0000
Kernel evaluations: 2645
Number of Support Vectors: 4
Margin: 0.2265
Training error: 3.70%