Nearest Neighbors

- Non-parametric
- Classification
- Regression
Key Idea: Close-by $\Rightarrow$ Similar

Close in space $\Rightarrow$ close in taste
The 1-nearest-neighbor Algorithm

**Training**

1. Store all \( (x_i, y_i) \) in dataset \( D \).

... that's it.

**Testing / Deployment**

Input: \( x_q \leftarrow \text{"query"} \)

1. Find closest \( x_i \) in training dataset.
2. Set \( \hat{y} = \hat{f}(x_i) \)

Output: \( \hat{y} \leftarrow \text{prediction} \)
What does this look like?

"Blocky"
The K-Nearest Neighbors Algorithm (regression)

Training

1. Store all \((X_i, Y_i)\) in dataset \(D\).

... that's it.

Testing / Deployment

Input: \(X_q \leftarrow \text{"query"}, K\)

1. Find closest \((\tilde{X}_1, \ldots, \tilde{X}_K)\) in training dataset.

2. Set \(\hat{Y} = \frac{1}{K} \sum_{j=1}^{K} \hat{f}(\tilde{X}_j)\)

Output: \(\hat{Y} \leftarrow \text{prediction}\)
The K-Nearest Neighbors Algorithm (classifier)

**Training**

1. Store all \((x_i, y_i)\) in dataset \(D\).
2. ... that's it.

**Testing / Deployment**

Input: \(x_q \leftarrow \text{"query"}, K\)

1. Find closest \((\tilde{x}_1, \ldots, \tilde{x}_k)\) in training dataset.
2. Set \(\hat{y} = \text{majority vote of } f(\tilde{x}_1), \ldots, f(\tilde{x}_k)\)

Output: \(\hat{y} \leftarrow \text{prediction}\)
Real Data

Linear Classifier

1-NN

15-NN

Diagrams from Hastie, Tibshirani & Friedman
What's the Decision Region?

S ← Training set

Voronoi cell of \( x \in S \) = all \( x' \) closer to \( x \) than any other in \( S \).

Region of class \( \bullet \) : All Voronoi cells of \( \bullet \).
What is "closer"?

Any distance metric will work.

E.g., if $x, z$ are $d$-dimensional,

$p$-norm: $\|x - z\|_p = \left(\sum_{j=1}^d |x_j - z_j|^p\right)^{1/p}$

Cosine similarity: $1 - \cos(\theta) = \frac{x^T z}{\|x\|_2 \|z\|_2}$
Behavior in the limit

\[ E^*(x) : \text{average error of Bayes classifier} \]

\[ E_{NN}(x) : \text{error of NN} \]

**Thm 1:**

\[ \lim_{n \to \infty} E_{NN}(x) \leq 2 E^*(x) \]

**Thm 2:**

\[ \lim_{n \to \infty} \lim_{k \to \infty} E_{kNN}(x) = E^*(x) \quad \text{if} \quad \frac{k}{n} \to 0 \]

Nearest Neighbor Pattern Classification

T. M. COVER, MEMBER, IEEE, AND P. E. HART, MEMBER, IEEE
Advantages: + Fast training
+ Learns complex functions easily

Disadvantages: - Slow at test time
- High storage cost
- Can perform poorly when $X$ is high-dimensional

⚠️ Requires a “good” distance metric.

Nowadays, learned!
Curse of Dimensionality

Low-D

High-D

empty!
Conclusion: distances are meaningless?
Good News!

Only the "intrinsic dimension" matters

All intrinsically 1D

What about the "manifold of images?"
Euclidean not good!

Which are closer?
Why is this a problem?
Transformation Invariance

$1 \rightarrow 1$

these should be close

Still far

$6 \rightarrow 9$
Building In Invariances

- Augment the dataset
- Build into features
  - e.g. ConvNets → shift invariance
- Design a good distance
  - e.g. “tangent distance”
Tangent Distance

Details in Simard et al. "Tangent Distance"
Tan distance results on digits

<table>
<thead>
<tr>
<th>Feature</th>
<th>Classifier</th>
<th>Error Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raw Pixels</td>
<td>SVM (linear)</td>
<td>11.3%</td>
</tr>
<tr>
<td>Raw Pixels</td>
<td>SVM (intersection)</td>
<td>8.7%</td>
</tr>
<tr>
<td>Raw Pixels</td>
<td>SVM (poly, $d = 3$) [7]</td>
<td>4.0%</td>
</tr>
<tr>
<td>Raw Pixels</td>
<td>VSV (poly, $d = 3$) [7]</td>
<td>3.2%</td>
</tr>
<tr>
<td>PHOG</td>
<td>SVM (linear)</td>
<td>3.4%</td>
</tr>
<tr>
<td>PHOG</td>
<td>SVM (intersection)</td>
<td>3.4%</td>
</tr>
<tr>
<td>PHOG</td>
<td>SVM (poly, $d = 5$)</td>
<td>3.2%</td>
</tr>
<tr>
<td>PHOG</td>
<td>SVM (rbf, $\gamma = 0.1$)</td>
<td>2.7%</td>
</tr>
<tr>
<td>Raw Pixels</td>
<td>Tangent Distance [23]*</td>
<td>2.6%</td>
</tr>
<tr>
<td>Raw Pixels</td>
<td>Boosted Neural Nets [8]*</td>
<td>2.6%</td>
</tr>
<tr>
<td></td>
<td>Human Error Rate [3]</td>
<td>2.5%</td>
</tr>
</tbody>
</table>
Learned Embeddings

FaceNet: A Unified Embedding for Face Recognition and Clustering

Florian Schroff  
fschroff@google.com  
Google Inc.

Dmitry Kalenichenko  
dkalenichenko@google.com  
Google Inc.

James Philbin  
jphilbin@google.com  
Google Inc.
Contrastive Learning

↑ trained with...
Triplet loss:

\[ L = \sum_{i=1}^{n} \left( \| f(x_i^a) - f(x_i^b) \|_2^2 - \| f(x_i^a) - f(x_i^c) \|_2^2 + \alpha \right)_+ \]
Nowadays...

Masked Autoencoders Are Scalable Vision Learners
Kaiming He*† Xinlei Chen* Saining Xie Yanghao Li Piotr Dollár Ross Girshick
*equal technical contribution †project lead
Facebook AI Research (FAIR)

Siamese Neural Networks for One-shot Image Recognition
Gregory Koch
Richard Zemel
Ruslan Salakhutdinov
Department of Computer Science, University of Toronto, Toronto, Ontario, Canada.