CS 2750: Machine Learning
Classification: Nearest Neighbors

Prof. Adriana Kovashka
University of Pittsburgh
February 2, 2017
Announcement

• Just for today, my office hours will be slightly shifted, 4:30-6pm
Plan for today

• Basic formulation of the simplest classifier: K-Nearest Neighbors
  – Example use
• Generalizing the distance metric and weighting neighbors differently
• Problems:
  – The curse of dimensionality
  – Picking K
  – Approximation strategies
Key Idea

• A type of supervised learning: We want to learn to predict, for a new data point $x$, its label $y$ (e.g. spam / not spam)
• Don’t learn an explicit function $F: X \rightarrow Y$
• Keep all training data $\{X, Y\}$
• For a test example $x$, find the training example $x_i$ closest to it (e.g. using Euclidean distance)
• Then copy the target label $y_i$ as the label for $x$
Related Methods

• Instance-based methods
• Exemplar methods
• Memory-based methods
• Non-parametric methods
Instance/Memory-based Learning

Four things make a memory based learner:
• A distance metric

• How many nearby neighbors to look at?

• A weighting function (optional)

• How to fit with the local points?
**1-Nearest Neighbor Classifier**

Four things make a memory based learner:

- **A distance metric**
  - Euclidean (and others)

- **How many nearby neighbors to look at?**
  - 1

- **A weighting function (optional)**
  - Not used

- **How to fit with the local points?**
  - Just predict the same output as the nearest neighbor

Slide credit: Carlos Guestrin
1-Nearest Neighbor Classifier

\[ f(x) = \text{label of the training example nearest to } x \]

Adapted from Lana Lazebnik
K-Nearest Neighbor Classifier

Four things make a memory based learner:

• A *distance metric*
  – Euclidean (and others)

• *How many nearby neighbors to look at?*
  – K

• A *weighting function (optional)*
  – Not used

• *How to fit with the local points?*
  – Just predict the average output among the nearest neighbors
K-Nearest Neighbor Classifier

- For a new point, find the $k$ closest points from training data (e.g. $k=5$)
- Labels of the $k$ points “vote” to classify

If query lands here, the 5 NN consist of 3 negatives and 2 positives, so we classify it as negative.

Slide credit: David Lowe
1-nearest neighbor
3-nearest neighbor

Slide credit: Derek Hoiem
5-nearest neighbor

What are the tradeoffs of having a too large $k$? Too small $k$?

Slide credit: Derek Hoiem
Formal Definition

- Let $x$ be our test data point, and $N_k(x)$ be the indices of the $k$ nearest neighbors of $x$

- Classification:
  \[
  y = \arg\max_c \#(y_i = c) \\
  y = \arg\max_c \sum_{i \in N_K(x)} I(y_i = c)
  \]

- Regression:
  \[
  y = \frac{1}{K} \sum_{i \in N_K(x)} y_i
  \]
Example: Predict where this picture was taken

Example: Predict where this picture was taken
Example: Predict where this picture was taken
6+ million geotagged photos by 109,788 photographers

Scene Matches

Scene Matches

Scene Matches

The Importance of Data

k-Nearest Neighbor

Four things make a memory based learner:

- **A distance metric**
  - Euclidean (and others)

- **How many nearby neighbors to look at?**
  - k

- **A weighting function (optional)**
  - Not used

- **How to fit with the local points?**
  - Just predict the average output among the nearest neighbors

Slide credit: Carlos Guestrin
Distances

• Suppose I want to charge my overall distance more for differences in $x_2$ direction as opposed to $x_1$ direction
• Setup A: equal weighing on all directions
• Setup B: more weight on $x_2$ direction
• Will my neighborhoods be longer in the $x_1$ or $x_2$ direction?
Voronoi partitioning

- Nearest neighbor regions
- All points in a region are closer to the seed in that region than to any other seed (black dots = seeds)

Figure from Wikipedia
Multivariate distance metrics

Suppose the input vectors $x_1, x_2, \ldots, x_N$ are two dimensional:

$$x_1 = (x_1^1, x_1^2), \ x_2 = (x_2^1, x_2^2), \ldots, \ x_N = (x_N^1, x_N^2).$$

The relative scalings in the distance metric affect region shapes.
Distance metrics

• Euclidean: \[ d(x, z) = \left[ \sum_{i=1}^{D} (x_i - z_i)^2 \right]^{\frac{1}{2}} \]

• Minkowski: \[ d(x, z) = \left[ \sum_{i=1}^{D} |x_i - z_i|^p \right]^{\frac{1}{p}} \]

• Mahalanobis: \[ d(x, z) = \sqrt{\sum_{i=1}^{D} \frac{(x_i - z_i)^2}{\sigma_i^2}} \]

\[ d(x, z) = (x - z)^T A (x - z) \]

(where \( A \) is a positive semidefinite matrix, i.e. symmetric matrix with all non-negative eigenvalues)
Distance metrics

Voronoi diagrams of 20 points under two different metrics

Euclidean distance

Manhattan distance

Figures from Wikipedia
Another generalization: Weighted K-NNs

• Neighbors weighted differently:
  – Use all samples, i.e. $K = N$
  – Weight on $i$-th sample:
    $$w_i = e^{-\frac{-||x-x_i||^2}{\sigma^2}}$$
  – $\sigma$ = the bandwidth parameter, expresses how quickly our weight function “drops off” as points get further and further from the query $x$

• Classification:
  $$y = \arg\max_c \sum_{i=1}^{N} w_i I(y_i = c)$$

• Regression:
  $$y = \frac{\sum_{i=1}^{N} w_i y_i}{\sum_{i=1}^{N} w_i}$$
Another generalization: Weighted K-NNs

- Extremes
  - Bandwidth = infinity: prediction is dataset average
  - Bandwidth = zero: prediction becomes 1-NN
Kernel Regression/Classification

Four things make a memory based learner:

- A distance metric
  - Euclidean (and others)

- How many nearby neighbors to look at?
  - All of them

- A weighting function (optional)
  - \( w_i = \exp(-d(x_i, \text{query})^2 / \sigma^2) \)
  - Nearby points to the query are weighted strongly, far points weakly. The \( \sigma \) parameter is the **kernel width / bandwidth**.

- How to fit with the local points?
  - Predict the weighted average of the outputs

Adapted from Carlos Guestrin
Problems with Instance-Based Learning

• Too many features?
  – Doesn’t work well if large number of irrelevant features, distances overwhelmed by noisy features
  – Distances become meaningless in high dimensions (the curse of dimensionality)

• What is the impact of the value of $K$?

• Expensive
  – No learning: most real work done during testing
  – For every test sample, must search through all dataset – very slow!
  – **Must use tricks like approximate nearest neighbor search**
  – Need to store all training data

Adapted from Dhruv Batra
Curse of Dimensionality (1)

How many neighborhoods are there?

\begin{align*}
\# \text{bins} &= 10 \times 10 \\
\text{d} &= 2 \\
\# \text{bins} &= 10^d \\
\text{d} &= 1000 \\
\text{Atoms in the universe} &\sim 10^{80}
\end{align*}
Curse of Dimensionality (2)

• Consider: Sphere of radius 1 in d-dims

• Consider: An outer $\epsilon$-shell in this sphere

• What is $\frac{\text{shell volume}}{\text{sphere volume}}$?
Curse of Dimensionality (2)

• The volume of a “sphere” with radius $r$ in $D$ dimensions is $K_D r^D$ (Bishop Sec. 1.4)

• What is $\frac{\text{shell volume}}{\text{sphere volume}}$?

\[
= \frac{K_D (1)^D - K_D (1 - \epsilon)^D}{K_D (1)^D} = 1 - (1 - \epsilon)^D
\]

• As $D$ tends to infinity, this ratio tends to 1
• I.e. most of the volume of the sphere is in that thin outer shell, which is counter-intuitive
Curse of Dimensionality (2)

Figure 1.22 from Bishop
Curse of Dimensionality (2)

- Problem: In very high dimensions, all points are equally close
- This problem applies to all types of classifiers, not just K-NN
Problems with Instance-Based Learning

• Too many features?
  – Doesn’t work well if large number of irrelevant features, distances overwhelmed by noisy features
  – Distances become meaningless in high dimensions (the curse of dimensionality)

• What is the impact of the value of K?

• Expensive
  – No learning: most real work done during testing
  – For every test sample, must search through all dataset – very slow!
  – Must use tricks like approximate nearest neighbor search
  – Need to store all training data

Adapted from Dhruv Batra
kNN Decision Boundary

- Increasing $k$ **complicates** decision boundary
kNN Decision Boundary

- Increasing $k$ “simplifies” decision boundary
  - Majority voting means less emphasis on individual points

$K = 1$

$K = 3$
kNN Decision Boundary

- Increasing $k$ “simplifies” decision boundary
  - Majority voting means less emphasis on individual points

$K = 5$

$K = 7$
kNN Decision Boundary

- Increasing $k$ “simplifies” decision boundary
  - Majority voting means less emphasis on individual points

$K = 25$
Use a validation set to pick K

K=1? Zero error!
Training data have been memorized...

Too complex

Best value of K
Problems with Instance-Based Learning

• Too many features?
  – Doesn’t work well if large number of irrelevant features, distances overwhelmed by noisy features
  – Distances become meaningless in high dimensions (the curse of dimensionality)

• What is the impact of the value of K?

• Expensive
  – No learning: most real work done during testing
  – For every test sample, must search through all dataset – very slow!
  – Must use tricks like approximate nearest neighbor search
  – Need to store all training data

Adapted from Dhruv Batra
An approximate distance method

- Build a balanced tree of data points ($kd$-tree), splitting along different dimensions
- Go down tree (starting from root) to find in which bin a query point lives, declare that leaf “current best” neighbor
- Go up the tree, checking for and exploring branches as needed when a closer neighbor could exist in that branch
- Check for possibility of better neighbor by intersecting regions with hypersphere defined by radius of current best; eliminate parts of the search space if they cannot contain a better current best
- Only search for neighbors up until some budget exhausted
Approximate kNN

- k-d tree: $O(\log N)$ query time

Approximate kNN

- k-d tree: $O(\log N)$ query time

![Fruit data](http://en.wikipedia.org/wiki/K-d_tree)

Subhransu Maji (UMASS)
Approximate kNN

- k-d tree: $O(\log N)$ query time

---


CMPSCI 689
Subhransu Maji (UMASS)
Approximate kNN

- k-d tree: $O(\log N)$ query time

split at the median

Approximate kNN

- k-d tree: $O(\log N)$ query time

split at the median

Approximate kNN

- k-d tree: $O(\log N)$ query time

Summary

• K-Nearest Neighbor is the most basic and simplest to implement classifier
• Cheap at training time, expensive at test time
• Unlike other methods we’ll see later, naturally works for any number of classes
• Pick K through a validation set, use approximate methods for finding neighbors
• Success of classification depends on the amount of data and the meaningfulness of the distance function