Introduction to Machine Learning
Lecture 4

Mehryar Mohri
Courant Institute and Google Research
mohri@cims.nyu.edu

Nearest-Neighbor Algorithms

Nearest Neighbor Algorithms

- Definition: fix $k \geq 1$, given a labeled sample

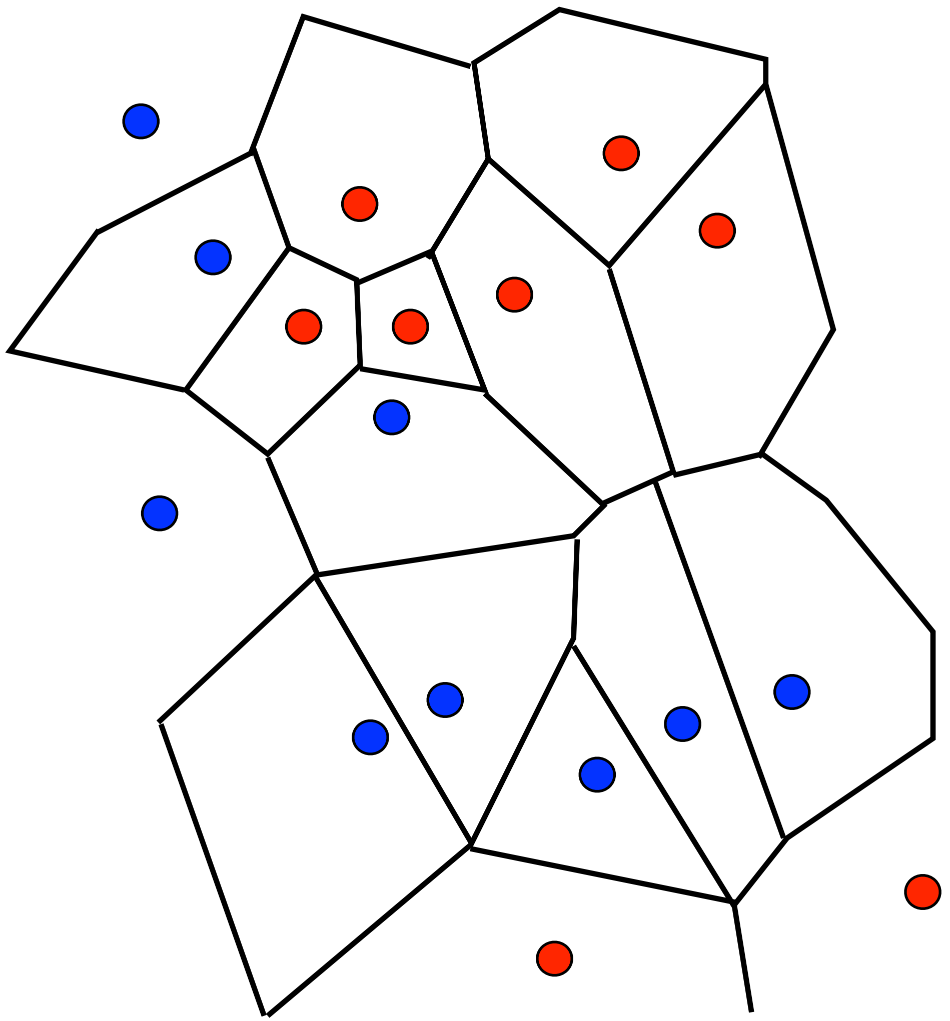
$$S = ((x_1, y_1), \ldots, (x_m, y_m)) \in (\mathcal{X} \times \{0, 1\})^m,$$

the k-NN returns the hypothesis h_S defined by

$$\forall x \in \mathcal{X}, h_S(x) = 1_{\sum_{i:y_i=1} w_i > \sum_{i:y_i=0} w_i},$$

where the weights $w_1, \ldots, w_m$ are chosen such that $w_i = \frac{1}{k}$ if x_i is among the k nearest neighbors of x.

Voronoi Diagram



Questions

- Performance: does it work?
- Choice of the weights: are there better choices than uniform? In particular, can take into account distance to each nearest neighbor.
- Choice of the distance metric: can a useful metric be defined (or even learned) for a particular problem?
- Computation in high dimension: data structures and algorithms to improve upon naive algorithm.

Bayes Classifier

- Definition: the Bayes error is defined by

$$R^\star = \inf_{\substack{h \\ h \text{ measurable}}} \Pr_{(x,y)\sim D}[h(x) \neq y].$$

  - the Bayes classifier is a measurable hypothesis achieving that error.

Set-up

- Sample drawn i.i.d. according to some distribution D

$$S = ((x_1, y_1), \ldots, (x_m, y_m)) \in (\mathcal{X} \times \{0, 1\})^m.$$

- Nearest neighbor of $x \in \mathcal{X}$:

$$\mathrm{NN}(S, x) = \underset{x' \text{ in } S}{\operatorname{argmin}} \; d(x, x').$$

- Error of hypothesis returned on point $x \in \mathcal{X}$:

$$R(h_S, x) = 1_{y(h_S(x)) \neq y(x)},$$

where $y(u)$ is the label of point u (random variable).

Convergence of NN Algorithm

- **Lemma**: for any x in support, $\mathrm{NN}(S, x), x) \to x$ with probability one when $|S| \to +\infty$.
- **Proof**: Let x be in the support of the distribution, then for any $\epsilon > 0$, $\Pr[B(x, \epsilon)] > 0$. Thus,

$$\Pr\Big[d\big(\mathrm{NN}(S, x), x\big) > \epsilon\Big] = \Big(1 - \Pr[B(x, \epsilon)]\Big)^{|S|} \to 0.$$

Since $d\big(\mathrm{NN}(S, x), x\big)$ is decreasing with $|S|$, this also implies convergence with probability one.

NN Algorithm - Limit Guarantee

- **Theorem**: let h_S be the hypothesis returned by the nearest neighbor algorithm. Then,

$$\lim_{|S|\to\infty} \mathop{\mathrm{E}}_{S\sim D^m}[R(h_S)] \leq 2R^*\left(1 - \frac{|\mathcal{Y}|/2}{|\mathcal{Y}|-1}R^*\right).$$

- **Proof**: $\mathop{\mathrm{E}}_{S\sim D^m}[R(h_S, x)]$

$$\begin{aligned}
&= \Pr_{S\sim D^m}[y(\mathrm{NN}(S,x)) \neq y(x)] \\
&= \sum_{x'} \Pr\left[y(x') \neq y(x) \mid \mathrm{NN}(S,x) = x'\right] \Pr_{S\sim D^m}[\mathrm{NN}(S,x) = x'] \\
&= \sum_{x'} \left(1 - \Pr\left[y(x') = y(x) \mid \mathrm{NN}(S,x) = x'\right]\right) \Pr_{S\sim D^m}[\mathrm{NN}(S,x) = x'] \\
&= \sum_{x'} \left(1 - \sum_{y\in\mathcal{Y}} \Pr[y \mid x] \Pr[y \mid x']\right) \Pr_{S\sim D^m}[\mathrm{NN}(S,x) = x'].
\end{aligned}$$

NN Algorithm - Limit Guarantee

- In view of the lemma, $\mathrm{NN}(S, x) \to x$ with probability one when $|S| \to +\infty$. Thus,

$$\lim_{|S| \to +\infty} \mathop{\mathrm{E}}_{S \sim D^m} [R(h_S, x)] = \Big(1 - \sum_{y \in \mathcal{Y}} \Pr[y \mid x]^2\Big).$$

From this it can be concluded that

$$\lim_{|S| \to +\infty} \mathop{\mathrm{E}}_{S \sim D^m} [R(h_S)] = \mathop{\mathrm{E}}_{x \sim D} \Big[1 - \sum_{y \in \mathcal{Y}} \Pr[y \mid x]^2\Big].$$

- Let $y^* = \operatorname*{argmax}_y \Pr[y|x]$, then

$$1 - \sum_{y \in \mathcal{Y}} \Pr[y \mid x]^2 = 1 - \Pr[y^* \mid x]^2 - \sum_{y \neq y^*} \Pr[y \mid x]^2.$$

NN Algorithm - Limit Guarantee

- Now, since the variance is non-negative,

$$\frac{1}{|\mathcal{Y}|-1}\sum_{y\neq y^*}\Pr[y \mid x]^2 - \Big(\frac{1}{|\mathcal{Y}|-1}\sum_{y\neq y^*}\Pr[y \mid x]\Big)^2 \geq 0.$$

Thus, in view of $\sum_{y\neq y^*}\Pr[y \mid x] = (1-\Pr[y^* \mid x])$,

$$\begin{aligned}
\operatorname*{E}_{x\sim D}\Big[1-\sum_{y\in\mathcal{Y}}\Pr[y \mid x]^2\Big] &\leq \operatorname*{E}_{x\sim D}\Big[1-\Pr[y^* \mid x]^2 - \frac{(1-\Pr[y^* \mid x])^2}{|\mathcal{Y}|-1}\Big]\\
&= \operatorname*{E}_{x\sim D}\Big[1-(1-R^*(x))^2 - \frac{R^*(x)^2}{|\mathcal{Y}|-1}\Big]\\
&= \operatorname*{E}_{x\sim D}\Big[2R^*(x) - \frac{|\mathcal{Y}|R^*(x)^2}{|\mathcal{Y}|-1}\Big]\\
&\leq 2R^* - \frac{|\mathcal{Y}|{R^*}^2}{|\mathcal{Y}|-1}. \quad (\text{using } \operatorname{E}[R^*(x)^2] \leq \operatorname{E}[R^*(x)]^2)
\end{aligned}$$

Notes

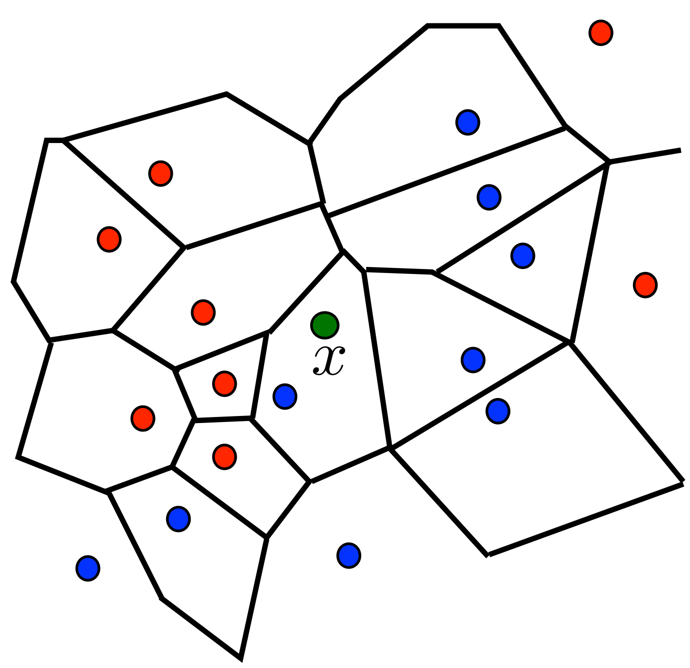
- Similar results for the k-NN algorithm.
  - $m = |S| \to \infty$ or $(k \to \infty) \wedge (\frac{k}{m} \to 0)$.
- Guarantees only for infinite amount of data:
  - machine learning deals with finite samples.
  - arbitrarily slow convergence rate.

NN Problem

- Problem: given sample $S = ((x_1, y_1), \ldots, (x_m, y_m))$, find the nearest neighbor of test point x.
  - general problem extensively studied in computer science.
  - exact vs. approximate algorithms.
  - dimensionality N crucial.
  - better algorithms for small intrinsic dimension (e.g., limited doubling dimension).

NN Problem - Case *N* = 2

- Algorithm:
  - compute Voronoi diagram in $O(m \log m)$.
  - point location data structure to determine NN.
  - complexity: $O(m)$ space, $O(\log m)$ time.



NN Problem - Case *N* > 2

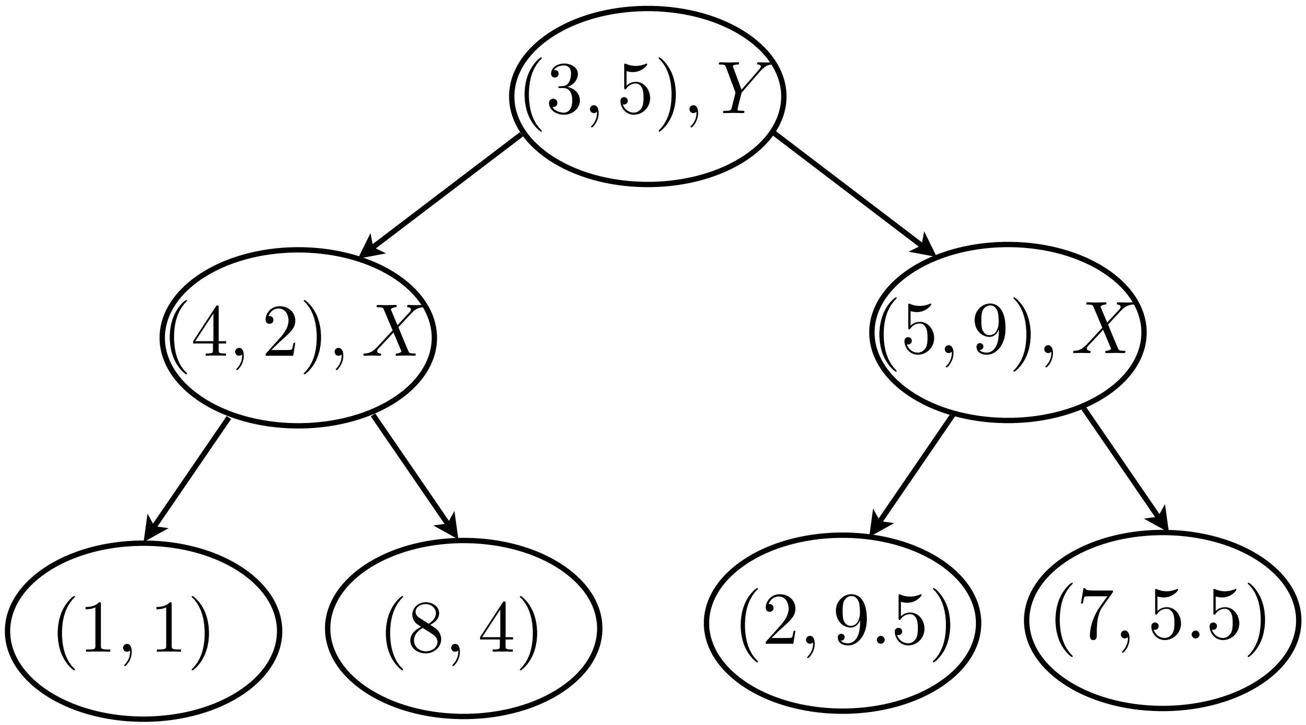
- Voronoi diagram: size in $O(m^{\lceil N/2 \rceil})$.
- Linear algorithm (no pre-processing):
  - compute distance $\|x - x_i\|$ for all $i \in [1, m]$.
  - complexity of distance computation: $\Omega(Nm)$.
  - no additional space needed.
- Tree-based data structures: pre-processing.
  - often used in applications: k-d trees (k-dimensional trees).

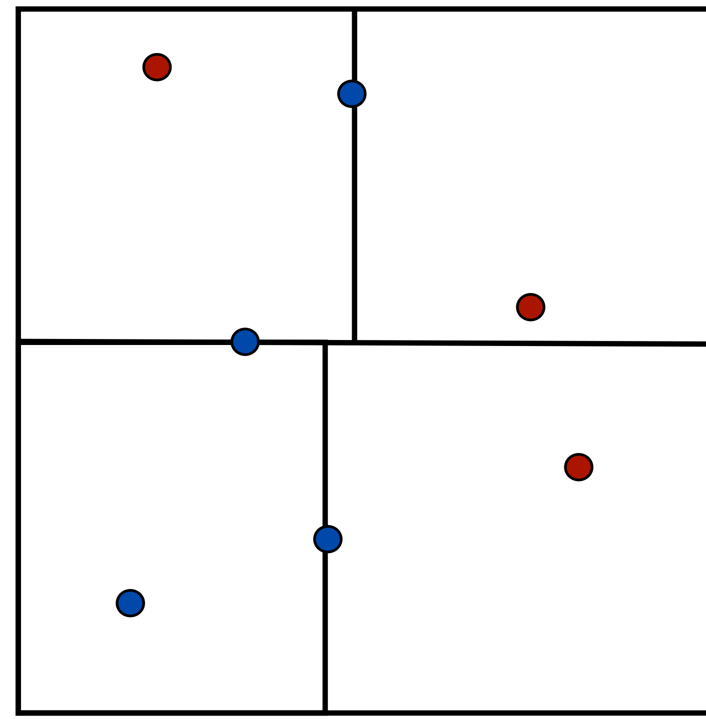
k-d Trees

(Bentley, 1975)

- Binary space partioning trees.
- Prominent tree-based data structure.
- Works for low or medium dimensionality.
- NN search:
  - $O(\log m)$ for randomly distributed points.
  - $O(Nm^{1-\frac{1}{N}})$ in the worst case (Lee and Wong, 1977).
- Can be extended to k-NN search.
- High dimension: typically inefficient.
  → approximate NN methods.

k-d Trees - Illustration



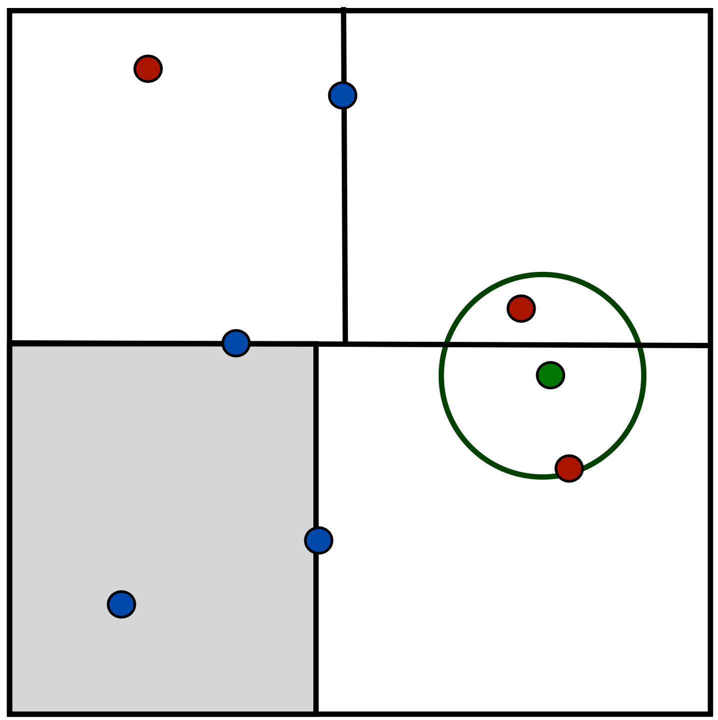


k-d Trees - Construction

- Algorithm: for each non-leaf node,
  - choose dimension (e.g., longest of hyperrectangle).
  - choose pivot (median).
  - split node according to (pivot, dimension).

→ balanced tree, binary space partitioning.

k-d Trees - NN Search



k-d Trees - NN Search

- Algorithm:
  - find region containing x (starting from root node, move to child node based on node test).
  - save region point x_0 as current best.
  - move up tree and recursively search regions intersecting hypersphere $S(x, \|x - x_0\|)$:
    - update current best if current point is closer.
    - restart search with each intersecting sub-tree.
    - move up tree when no more intersecting sub-tree.

References

- Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, Vol. 18, No. 9, 1975.

- Lee, D.T. and Wong, C. K. Worst-case analysis for region and partial region searches in multidimensional binary search trees and balanced quad trees. Acta Informatica Vol. 9, Issue 1. Springer, NY, 1977.