CS 886
Applied Machine Learning
K-Nearest Neighbours

Dan Lizotte

University of Waterloo

20 June 2013
Instance-based learning

- Non-parametric learning
- $k$-nearest neighbour
- Efficient implementations
- Variations
Parametric supervised learning

- So far, we have assumed that we have a data set $D$ of labeled examples.
- From this, we learn a parameter vector of a fixed size such that some error measure based on the training data is minimized.
- These methods are called parametric, and their main goal is to summarize the data using the parameters.
- Parametric methods are typically global, i.e. have one set of parameters for the entire data space.
- But what if we just remembered the data?
- When new instances arrive, we will compare them with what we know, and determine the answer.
Non-parametric (memory-based) learning methods

- Key idea: just store all training examples $\langle x_i, y_i \rangle$
- When a query is made, compute the value of the new instance based on the values of the closest (most similar) points
- Requirements:
  - A distance function
  - How many closest points (neighbors) to look at?
  - How do we compute the value of the new point based on the existing values?
Simple idea: Connect the dots!

Wisconsin data set, classification
Simple idea: Connect the dots!

Wisconsin data set, regression
One-nearest neighbor

- Given: Training data \(\{(x_i, y_i)\}_{i=1}^{n}\), distance metric \(d\) on \(\mathcal{X}\).
- Training: Nothing to do! (just store data)
- Prediction: for \(x \in \mathcal{X}\)
  - Find nearest training sample to \(x\).

\[ i^* \in \arg \min_i d(x_i, x) \]

- Predict \(y = y_{i^*}\).
What does the approximator look like?

- Nearest-neighbor does not explicitly compute decision boundaries
- But the effective decision boundaries are a subset of the Voronoi diagram for the training data

Each line segment is equidistant between two points of opposite classes.
What kind of distance metric?

- Euclidian distance
- Maximum/minimum difference along any axis
- Weighted Euclidian distance (with weights based on domain knowledge)
  \[ d(x, x') = \sum_{j=1}^{n} u_j (x_j - x_j')^2 \]
- An arbitrary distance or similarity function \( d \), specific for the application at hand (works best, if you have one)
Distance metric is really important!

features weighted equally

vertical matters more
Distance metric tricks

- You may need to do preprocessing:
  - *Scale* the input dimensions (or normalize them)
  - Determine weights for features based on cross-validation (or information-theoretic methods)
- Distance metric is often domain-specific
  - E.g. string edit distance in bioinformatics
  - E.g. trajectory distance in time series models for walking data
- Distance metric can be learned sometimes (more on this later)
$k$-nearest neighbor

- Given: Training data $\{(x_i, y_i)\}_{i=1}^n$, distance metric $d$ on $\mathcal{X}$.
- Learning: Nothing to do!
- Prediction: for $x \in \mathcal{X}$
  - Find the $k$ nearest training samples to $x$.
    Let their indices be $i_1, i_2, \ldots, i_k$.
  - Predict
    - $y = \text{mean/median of } \{y_{i_1}, y_{i_2}, \ldots, y_{i_k}\}$ for regression
    - $y = \text{majority of } \{y_{i_1}, y_{i_2}, \ldots, y_{i_k}\}$ for classification, or empirical probability of each class
Classification, 2-nearest neighbor, empirical distribution

2-nearest neighbor, mean

non-recurring (0) / recurring (1)

tumor size (mm?)

Dan Lizotte (University of Waterloo)
CS 886 - TUT - KNN
20 June 2013 13 / 32
Classification, 3-nearest neighbor

3-nearest neighbor, mean

non-recurring (0) / recurring (1)

Dan Lizotte (University of Waterloo)
Classification, 5-nearest neighbor

5-nearest neighbor, mean

non-recurring (0) / recurring (1)

Dan Lizotte (University of Waterloo)
CS 886 - TUT - KNN
20 June 2013 15 / 32
Classification, 10-nearest neighbor

10-nearest neighbor, mean

non-recurring (0) / recurring (1) vs tumor size (mm?)

Dan Lizotte (University of Waterloo)
CS 886 - TUT - KNN
20 June 2013 16 / 32
Classification, 15-nearest neighbor

15-nearest neighbor, mean

<table>
<thead>
<tr>
<th>Tumor Size (mm?)</th>
<th>Non-recurring (0) / Recurring (1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td></td>
</tr>
</tbody>
</table>

Dan Lizotte (University of Waterloo)
CS 886 - TUT - KNN
20 June 2013
Classification, 20-nearest neighbor

20–nearest neighbor, mean

non-recurring (0) / recurring (1)

tumor size (mm?)

Dan Lizotte (University of Waterloo)
Regression, 2-nearest neighbor, mean prediction

Dan Lizotte (University of Waterloo)
CS 886 - TUT - KNN
20 June 2013
Regression, 3-nearest neighbor

Dan Lizotte (University of Waterloo)
CS 886 - TUT - KNN
20 June 2013

The graph shows the relationship between nucleus size and time to recurrence.

- The x-axis represents nucleus size, ranging from 10 to 28.
- The y-axis represents time to recurrence, ranging from 0 to 80.
- Data points are marked with crosses, and a line connects them to illustrate the trend.

This visualization helps in understanding how nucleus size correlates with recurrence time, which is crucial for further analysis and prediction in the field of study.
Regression, 5-nearest neighbor

![Graph showing relationship between nucleus size and time to recurrence]
Regression, 10-nearest neighbor

Dan Lizotte (University of Waterloo)
Bias-variance trade-off

- If \( k \) is low, very non-linear functions can be approximated, but we also capture the noise in the data
  
  Bias is low, variance is high

- If \( k \) is high, the output is much smoother, less sensitive to data variation
  
  High bias, low variance

- A validation set can be used to pick the best \( k \)
Distance-weighted (kernel-based) nearest neighbor

- Inputs: Training data \( \{(x_i, y_i)\}_{i=1}^n \), distance metric \( d \) on \( \mathcal{X} \), weighting function \( w : \mathbb{R} \to \mathbb{R} \).
- Learning: Nothing to do!
- Prediction: On input \( x \),
  - For each \( i \) compute \( w_i = w(d(x_i, x)) \).
  - Predict weighted majority or mean. For example,

\[
y = \frac{\sum_i w_i y_i}{\sum_i w_i}
\]

- How to weight distances?
Some weighting functions

\[
\frac{1}{d(x_i, x)} \quad \frac{1}{d(x_i, x)^2} \quad \frac{1}{c + d(x_i, x)^2} \quad e^{-\frac{d(x_i, x)^2}{\sigma^2}}
\]
Example: Gaussian weighting, small $\sigma$

Gaussian-weighted nearest neighbor with $\sigma=0.25$
Gaussian weighting, medium $\sigma$

Gaussian-weighted nearest neighbor with $\sigma=2$

![Graph showing non-recurring (0) / recurring (1) classification based on tumor size.](image)
Gaussian weighting, large $\sigma$

All examples get to vote! Curve is smoother, but perhaps too smooth.
Locally-weighted linear regression

- Weighted linear regression: different weights in the error function for different points (see homework 1)
- Locally weighted linear regression: weights depend on the distance to the query point
- Uses a linear fit rather than just an average

![Diagram showing locally-weighted linear regression with different kernel sizes and effects on the fit.]

Kernel too wide — includes nonlinear region
Kernel just right
Kernel too narrow — excludes some of linear region

$x$
Lazy and eager learning

- **Lazy**: wait for query before generalizing
  E.g. Nearest Neighbor
- **Eager**: generalize before seeing query
  E.g. Backpropagation, Linear regression,

  Does it matter?
Pros and cons of lazy and eager learning

- Eager learners must create global approximation
- Lazy learners can create many local approximations
- An eager learner does the work off-line, summarizes lots of data with few parameters
- A lazy learner has to do lots of work sifting through the data at query time
- Typically lazy learners take longer time to answer queries and require more space
When to consider instance-based learning

- Instances map to points in $\mathbb{R}^p$
- Not too many features per instance (maybe < 20)
- Advantages:
  - Training is very fast
  - Easy to learn complex functions over few variables
  - Can give back confidence intervals in addition to the prediction
  - Variable resolution (depends on the density of data points)
  - Does not lose any information
  - Often wins if you have enough data
- Disadvantages:
  - Slow at query time
  - Query answering complexity depends on the number of instances
  - Easily fooled by irrelevant features (for most distance metrics)