k-NN (k-Nearest Neighbors), Kernel Regression

Pradeep Ravikumar

Co-instructor: Manuela Veloso

Machine Learning 10-701
k-NN classifier
k-NN classifier
k-NN classifier (k=5)

Test document

What should we predict? ... Average? Majority? Why?
k-NN classifier

- Optimal Classifier: \( f^*(x) = \arg \max_y P(y|x) \)
  \[ = \arg \max_y P(x|y)P(y) \]

- k-NN Classifier: \( \hat{f}_{kNN}(x) = \arg \max_y \hat{P}_{kNN}(x|y)\hat{P}(y) \)
  \[ = \arg \max_y k_y \]

\[ \hat{P}_{kNN}(x|y) = \frac{k_y}{n_y} \quad \text{# training pts of class y amongst k NNs of x} \]
\[ \sum_y k_y = k \quad \text{# total training pts of class y} \]

no. of training pts like x with label y / no. of training pts with label y

\[ \hat{P}(y) = \frac{n_y}{n} \]
1-Nearest Neighbor (kNN) classifier
2-Nearest Neighbor (kNN) classifier
3-Nearest Neighbor (kNN) classifier

- Sports
- Science
- Arts
5-Nearest Neighbor (kNN) classifier
What is the best k?

1-NN classifier decision boundary

As k increases, boundary becomes smoother (less jagged).
What is the best k?

Approximation vs. Stability Tradeoff

• Larger $K \Rightarrow$ predicted label is more stable
• Smaller $K \Rightarrow$ predicted label can approximate best classifier well
Parametric methods

• Assume some model (Gaussian, Bernoulli, Multinomial, logistic, network of logistic units, Linear, Quadratic) with fixed number of parameters
  • Gaussian Bayes, Naïve Bayes, Logistic Regression, Perceptron, Neural Networks

• Estimate parameters ($\mu, \sigma^2, \theta, w, \beta$) using MLE/MAP and plug in

• **Pro** – need few data points to learn parameters
• **Con** – Strong distributional assumptions, not satisfied in practice
Non-Parametric methods

• Typically don’t make any distributional assumptions
• As we have more data, we should be able to learn more complex models
• Let number of parameters scale with number of training data

• Some nonparametric methods
  • Decision Trees
  • k-NN (k-Nearest Neighbor) Classifier
Parametric vs Nonparametric approaches

- Nonparametric models place very mild assumptions on the data distribution and provide good models for complex data.
  Parametric models rely on very strong (simplistic) distributional assumptions.

- Nonparametric models requires storing and computing with the entire data set.
  Parametric models, once fitted, are much more efficient in terms of storage and computation.
Local, Kernel Regression
Local Kernel Regression

• What is the temperature in the room?

\[ \hat{T} = \frac{1}{n} \sum_{i=1}^{n} Y_i \]

Average

\[ \hat{T}(x) = \frac{\sum_{i=1}^{n} Y_i 1_{||X_i - x|| \leq h}}{\sum_{i=1}^{n} 1_{||X_i - x|| \leq h}} \]

“Local” Average

at location \( x \)?
Nadaraya-Watson Kernel Regression

\[ \Rightarrow \hat{f}_n(X) = \hat{\beta} = \sum_{i=1}^{n} w_i Y_i \]

with box-car kernel

\[ = \frac{1}{n^h} \sum_{i=1}^{n} Y_i \min(1, \frac{|X-X_i|}{h}) \]

#pts in h ball around X

Sum of Ys in h ball around X

Recall: NN classifier
with majority vote

Here we use Average instead

\[ w_i(X) = \frac{K\left(\frac{X-X_i}{h}\right)}{\sum_{i=1}^{n} K\left(\frac{X-X_i}{h}\right)} \]

boxcar kernel:

\[ K\left(\frac{X-X_i}{h}\right) = 1_{|X-X_i| \leq h} \]
Local Kernel Regression

• Nonparametric estimator akin to kNN
• Nadaraya-Watson Kernel Estimator

\[
\hat{f}_n(X) = \sum_{i=1}^{n} w_i Y_i \quad \text{Where} \quad w_i(X) = \frac{K\left(\frac{X-X_i}{h}\right)}{\sum_{i=1}^{n} K\left(\frac{X-X_i}{h}\right)}
\]

• Weight each training point based on distance to test point
• Boxcar kernel yields local average
Kernels

\[ K(x) \geq 0, \]
\[ \int K(x) \, dx = 1 \]

boxcar kernel:
\[ K(x) = \frac{1}{2} I(x), \]

Gaussian kernel:
\[ K(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2} \]
Choice of kernel bandwidth $h$

- $h=1$: Too small
- $h=10$: Too small
- $h=50$: Just right
- $h=200$: Too large

Choice of kernel is not that important

Image Source: Larry's book – All of Nonparametric Statistics
Kernel Regression as Weighted Least Squares

$$\min_f \frac{1}{n} \sum_{i=1}^{n} w_i (f(X_i) - Y_i)^2$$

$$\frac{1}{n} \sum_{i=1}^{n} w_i = 1$$

Weighted Least Squares

Weigh each training point based on distance to test point

$$w_i(X) = \frac{K \left( \frac{X - X_i}{h} \right)}{\sum_{i=1}^{n} K \left( \frac{X - X_i}{h} \right)}$$

$K$ – Kernel
$h$ – Bandwidth of kernel

**boxcar kernel:**

$$K(x) = \frac{1}{2} I(x).$$

**Gaussian kernel:**

$$K(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}.$$
Kernel Regression as Weighted Least Squares

set \( f(X_i) = \beta \) (a constant)

\[
\min_{\beta} \sum_{i=1}^{n} w_i (\beta - Y_i)^2
\]

\[
\frac{\partial J(\beta)}{\partial \beta} = 2 \sum_{i=1}^{n} w_i (\beta - Y_i) = 0
\]

\[
\Rightarrow \hat{f}_n(X) = \hat{\beta} = \sum_{i=1}^{n} w_i Y_i
\]

\[
\begin{align*}
\text{Notice that } & \sum_{i=1}^{n} w_i = 1 \\
\end{align*}
\]
Choice of Bandwidth

Large Bandwidth – average more data points, reduce noise (Lower variance)

Small Bandwidth – less smoothing, more accurate fit (Lower bias)

Bias – Variance tradeoff

Should depend on $n$, # training data (determines variance)

Should depend on smoothness of function (determines bias)
Spatially adaptive regression

If function smoothness varies spatially, we want to allow bandwidth $h$ to depend on $X$

Local polynomials, splines, wavelets, regression trees ...
Local Linear/Polynomial Regression

\[ \min_{f} \sum_{i=1}^{n} w_i (f(X_i) - Y_i)^2 \]

\[ w_i(X) = \frac{K \left( \frac{X-X_i}{h} \right)}{\sum_{i=1}^{n} K \left( \frac{X-X_i}{h} \right)} \]

Weighted Least Squares

Local Polynomial regression corresponds to locally polynomial estimator obtained from (locally) weighted least squares

i.e. set \( f(X_i) = \beta_0 + \beta_1 (X_i - X) + \frac{\beta_2}{2!} (X_i - X)^2 + \cdots + \frac{\beta_p}{p!} (X_i - X)^p \)

(local polynomial of degree p around X)
Local Regression

\[ f(X) = \sum_{j=0}^{m} \beta_j \phi_j(X) \]

- Basis coefficients
- Nonlinear features/basis functions

Globally supported basis functions (polynomial, Fourier) will not yield a good representation.
Local Regression

\[ f(X) = \sum_{j=0}^{m} \beta_j \phi_j(X) \]

Basis coefficients \( \beta_j \)
Nonlinear features/basis functions \( \phi_j(X) \)

Globally supported basis functions (polynomial, Fourier) will not yield a good representation
Local prediction

Histogram Classifier
Local Adaptive prediction

Let neighborhood size adapt to data – small neighborhoods near decision boundary (small bias), large neighborhoods elsewhere (small variance)

Decision Tree Classifier

Majority vote at each leaf
Regression trees

Binary Decision Tree

$X^{(1)} \ldots X^{(p)} \ Y$

<table>
<thead>
<tr>
<th>Gender</th>
<th>Rich?</th>
<th>Num. Children</th>
<th># travel per yr.</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>No</td>
<td>2</td>
<td>5</td>
<td>38</td>
</tr>
<tr>
<td>M</td>
<td>No</td>
<td>0</td>
<td>2</td>
<td>25</td>
</tr>
<tr>
<td>M</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>72</td>
</tr>
</tbody>
</table>

Average (fit a constant) on the leaves

Num Children?

$\geq 2$

Gender?

Female

Predicted age=39

Male

Predicted age=36