CS 171: Intro to ML and DM

Christian Shelton

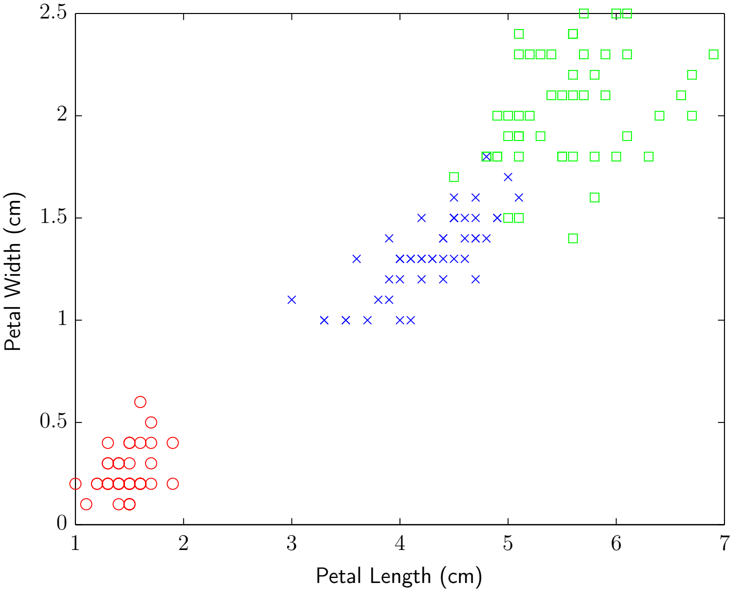
UC Riverside

Slide Set 8: Nearest Neighbor I

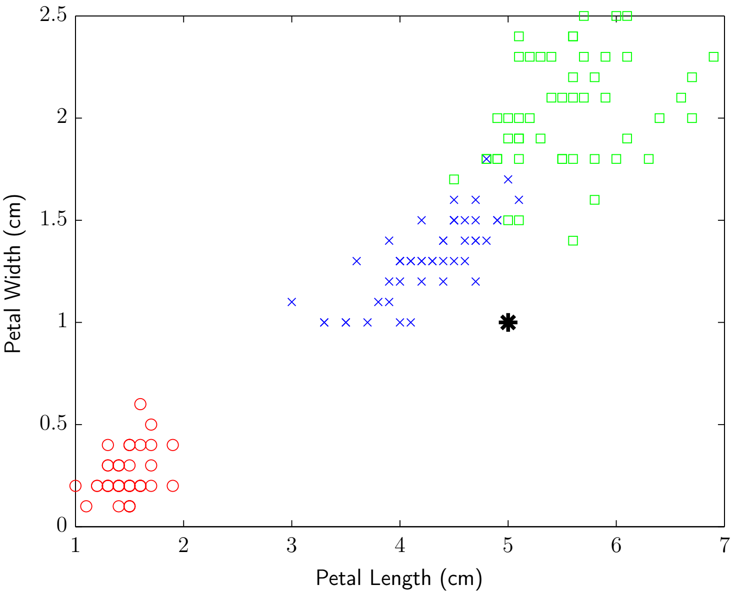
Slides from CS 171

- From UC Riverside
  - CS 171: Introduction to Machine Learning and Data Mining
  - Professor Christian Shelton
- DO NOT REDISTRIBUTE
  - These slides contain copyrighted material (used with permission) from
    - Elements of Statistical Learning (Hastie, et al.)
    - Pattern Recognition and Machine Learning (Bishop)
    - An Introduction to Machine Learning (Kubat)
    - Machine Learning: A Probabilistic Perspective (Murphy)
  - For use only by enrolled students in the course

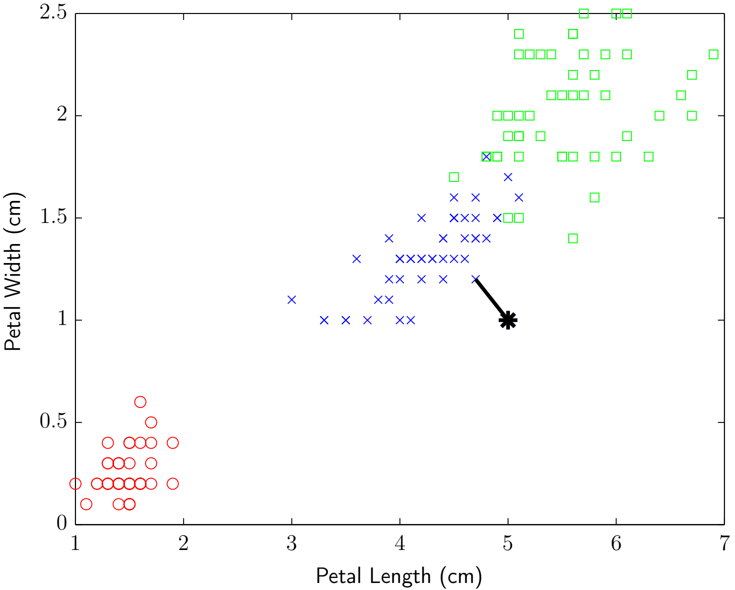
k-Nearest Neighbor



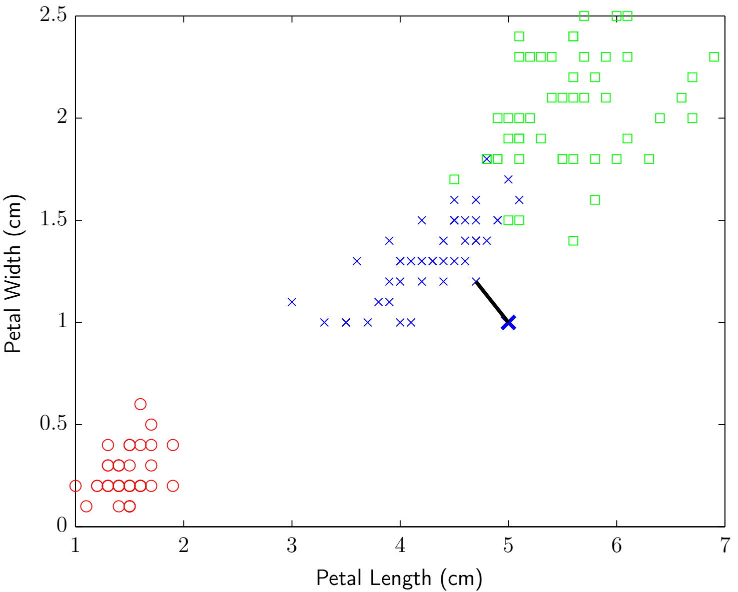
k-Nearest Neighbor



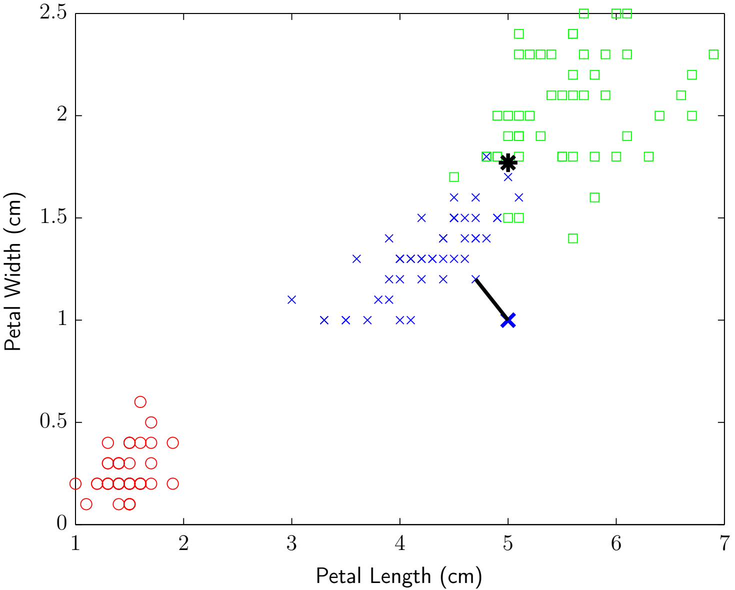
k-Nearest Neighbor



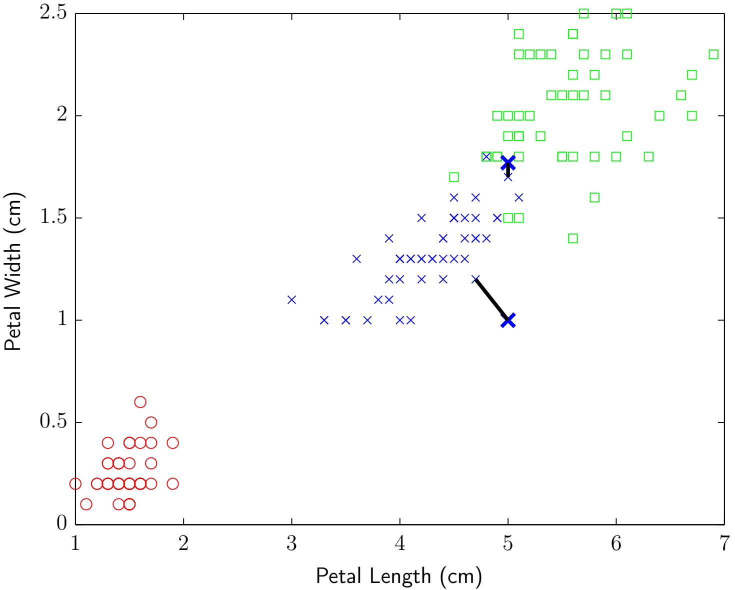
k-Nearest Neighbor



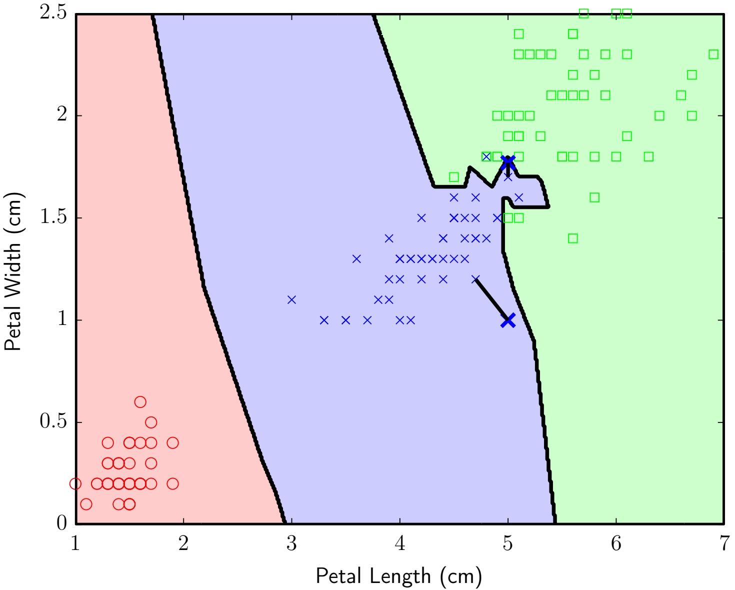
k-Nearest Neighbor



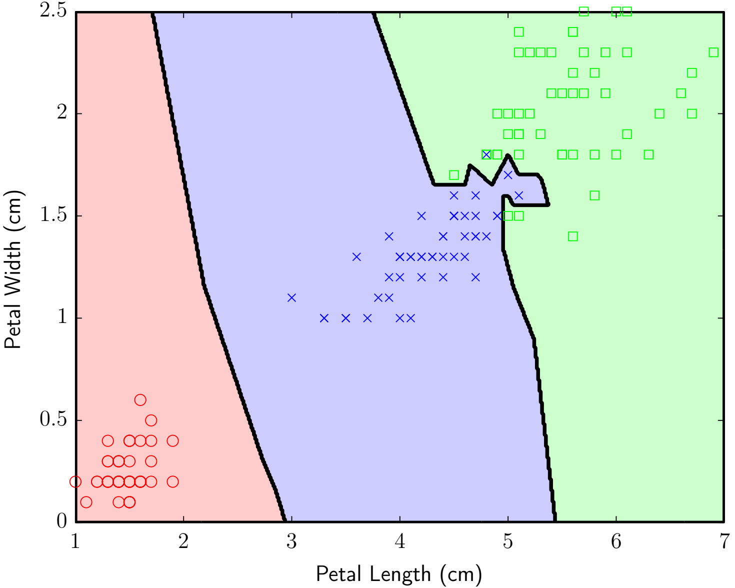
k-Nearest Neighbor



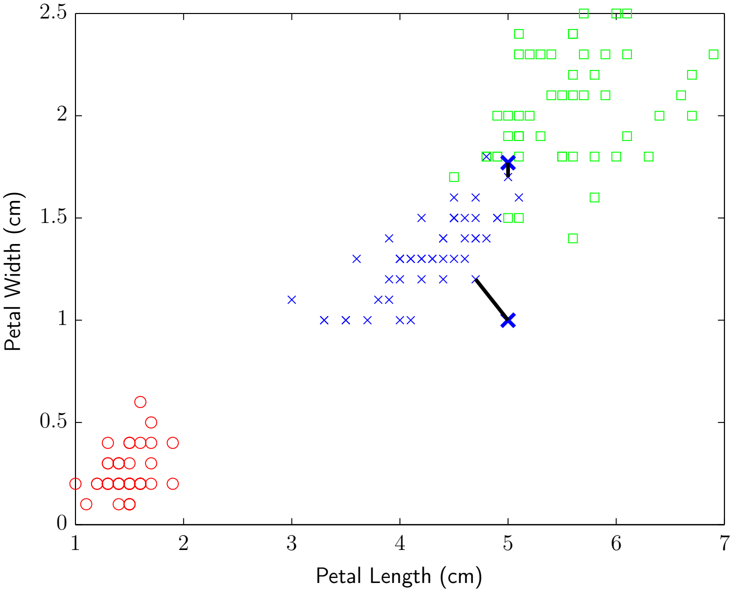
k-Nearest Neighbor



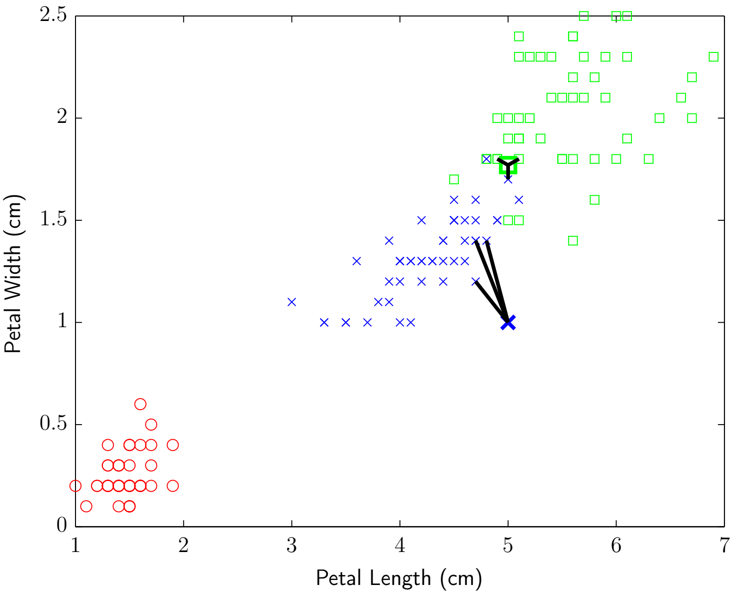
k-Nearest Neighbor



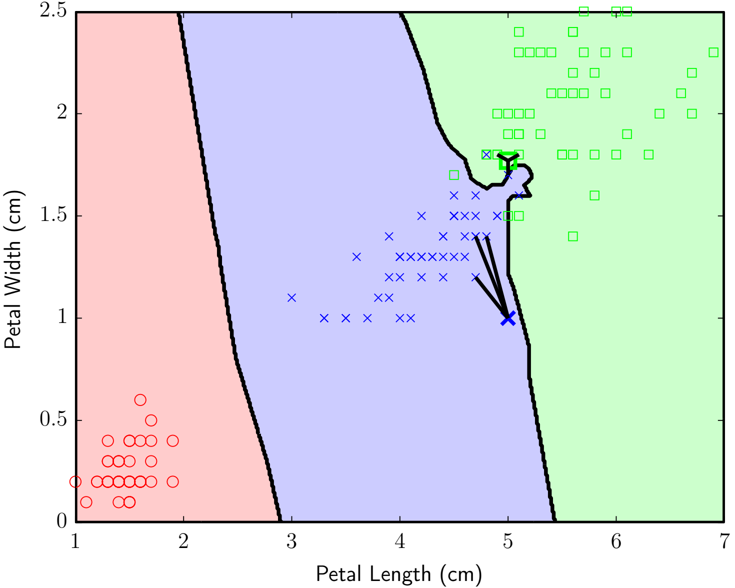
k-Nearest Neighbor



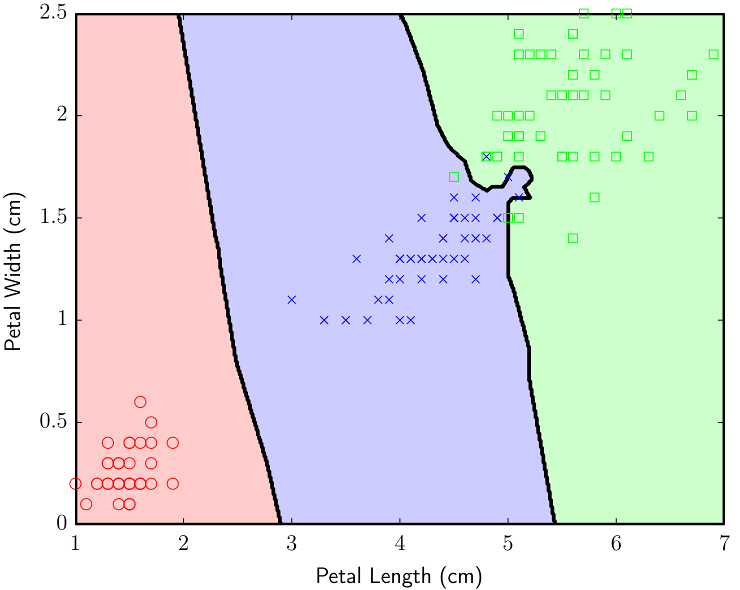
k-Nearest Neighbor



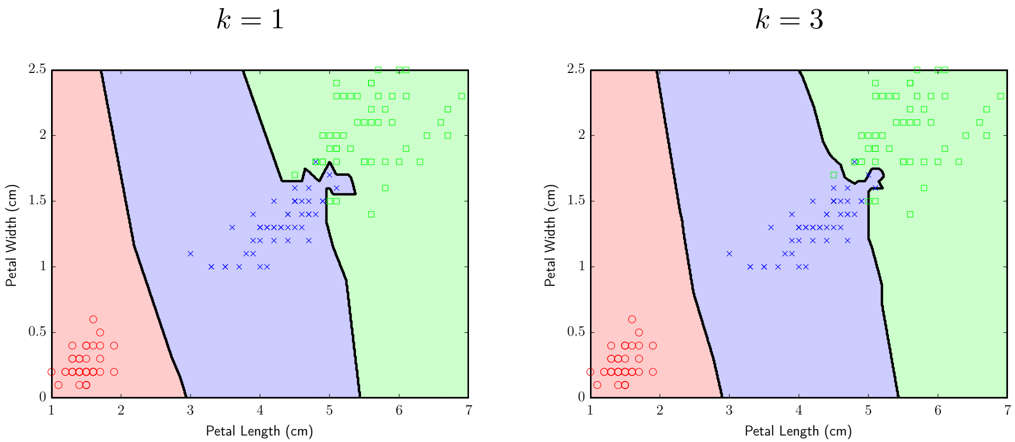
k-Nearest Neighbor



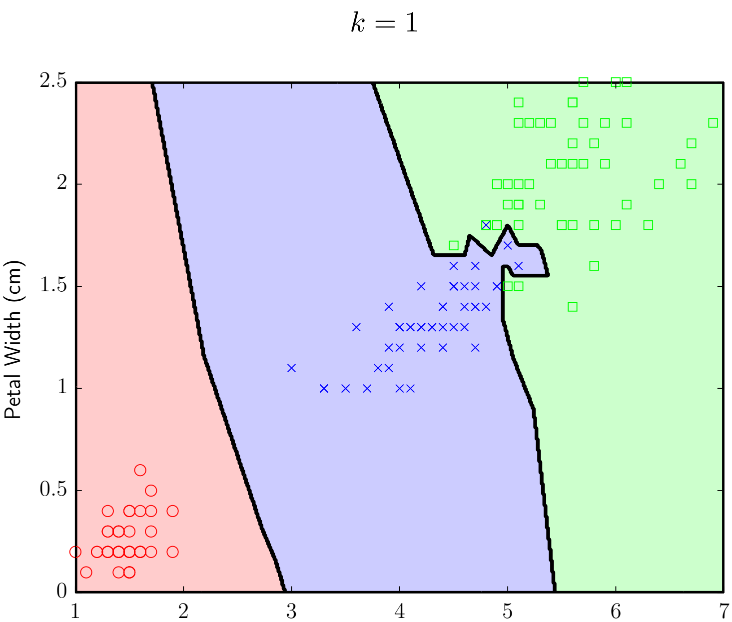
k-Nearest Neighbor



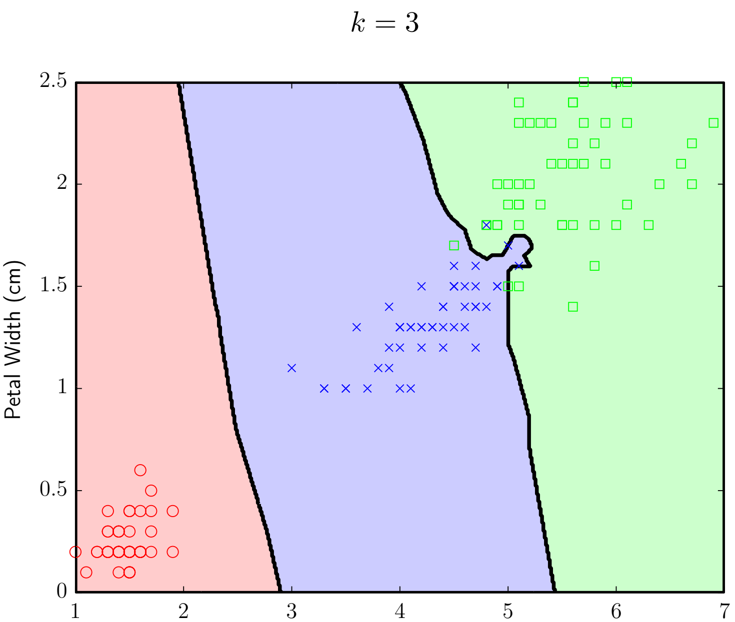
k-Nearest Neighbor



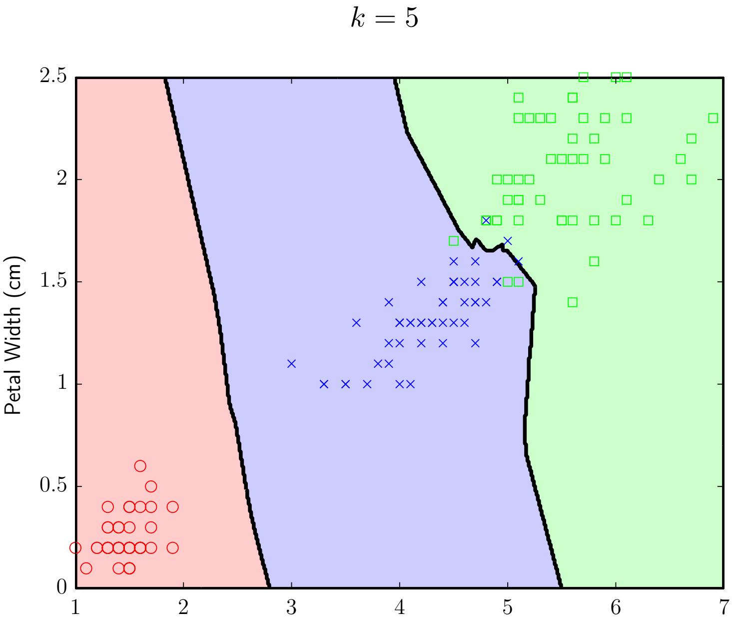
k-Nearest Neighbor



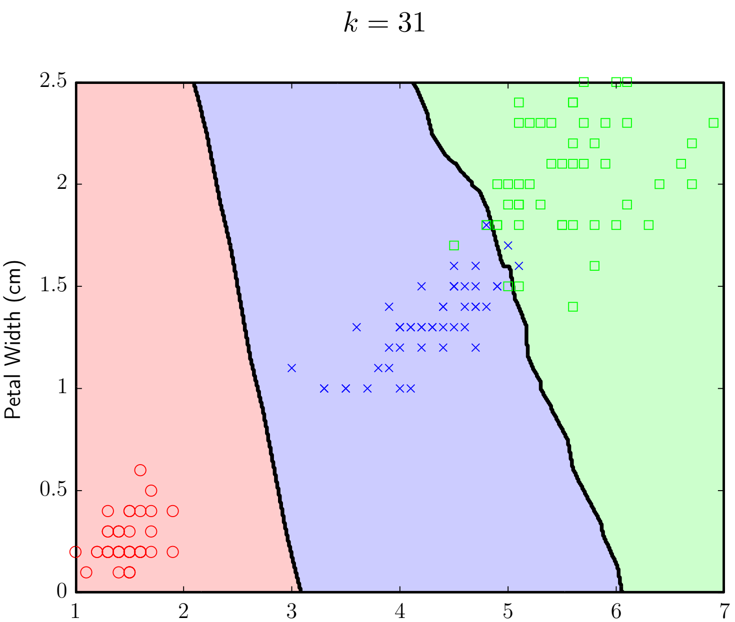
k-Nearest Neighbor



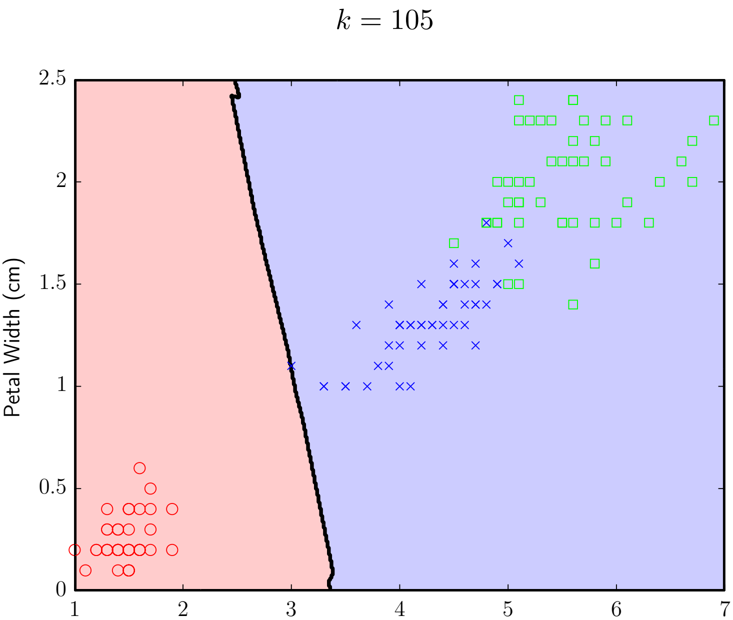
k-Nearest Neighbor



k-Nearest Neighbor

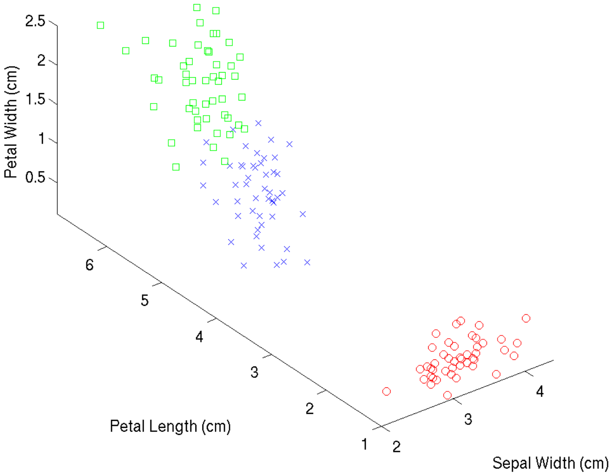


k-Nearest Neighbor



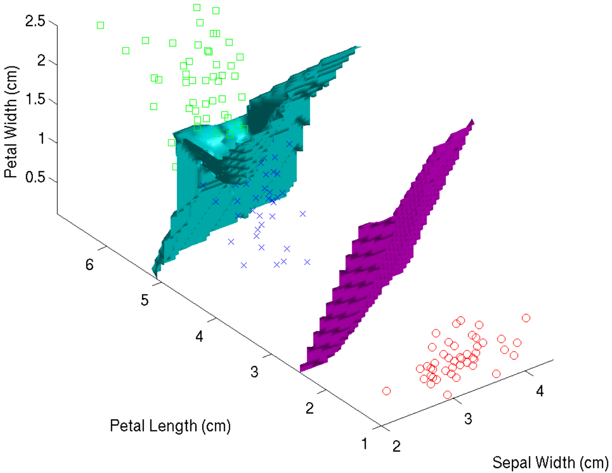
k-Nearest Neighbor, 3D

$$k = 1$$



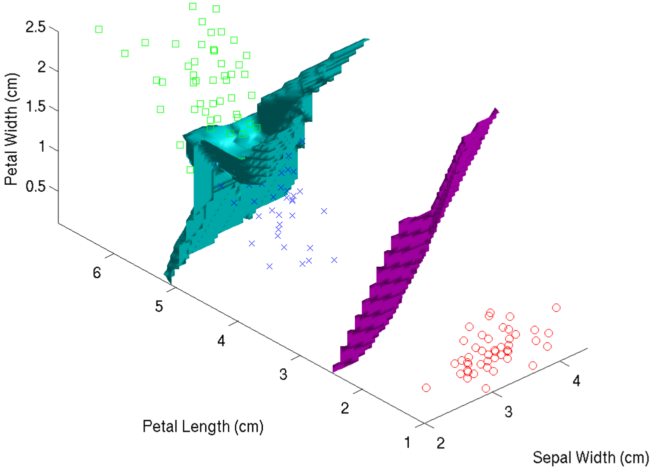
k-Nearest Neighbor, 3D

$$k = 1$$



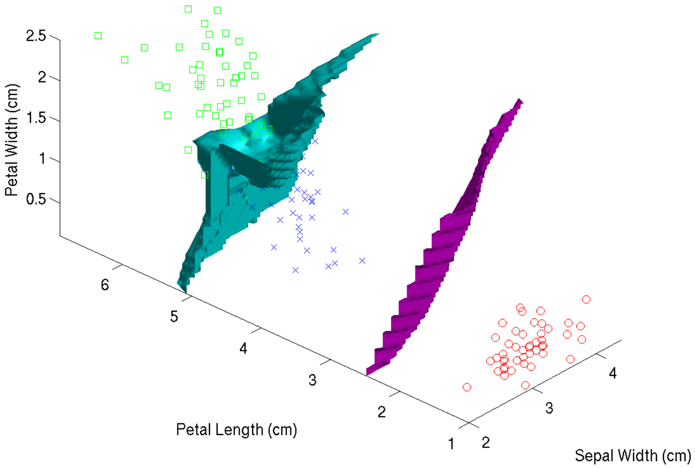
k-Nearest Neighbor, 3D

$k = 1$



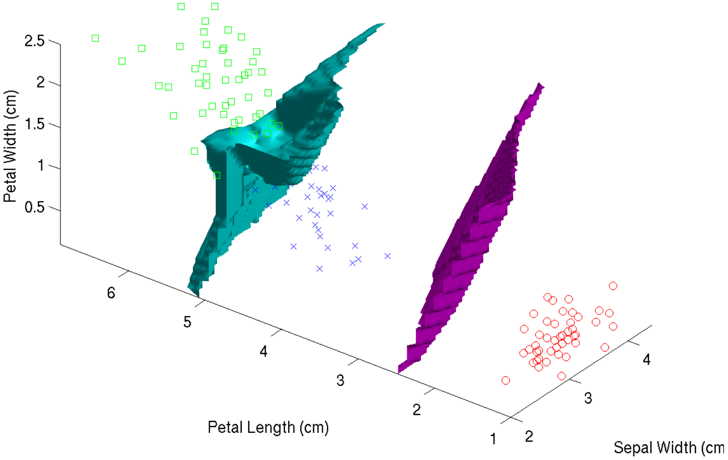
k-Nearest Neighbor, 3D

$$k = 1$$



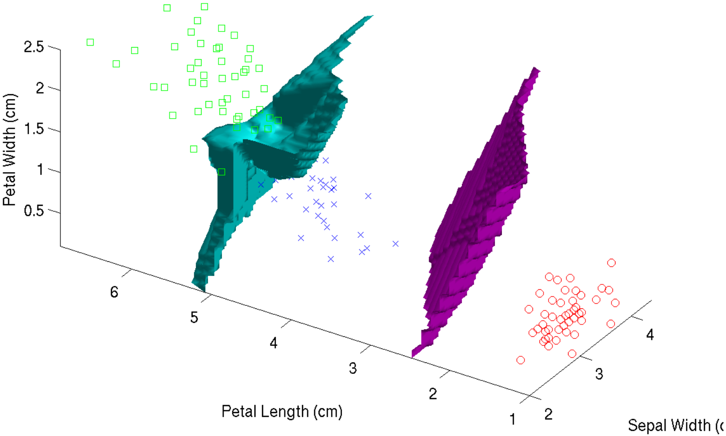
k-Nearest Neighbor, 3D

$$k = 1$$



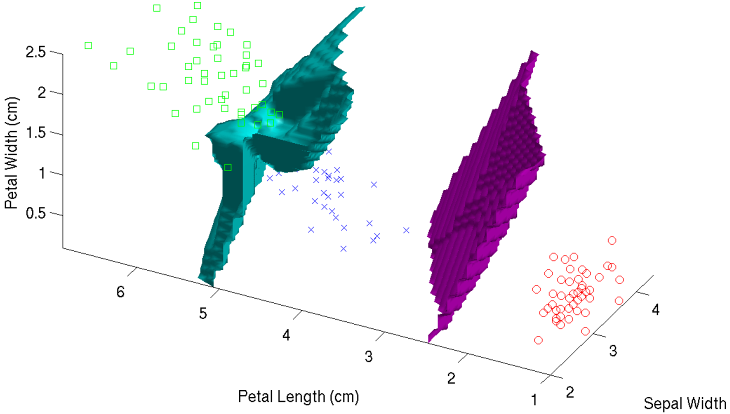
k-Nearest Neighbor, 3D

$$k = 1$$



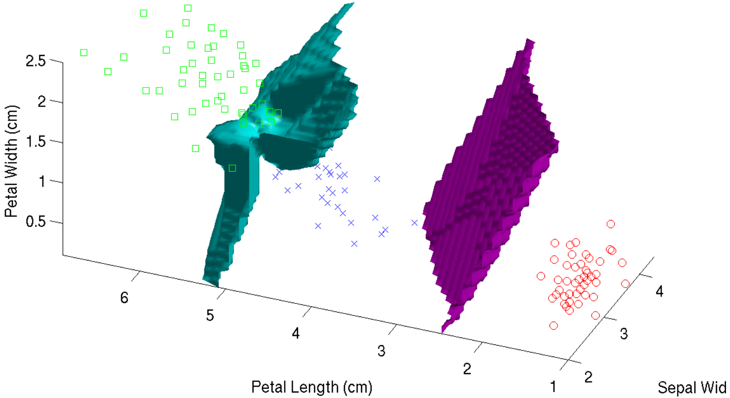
k-Nearest Neighbor, 3D

$$k = 1$$



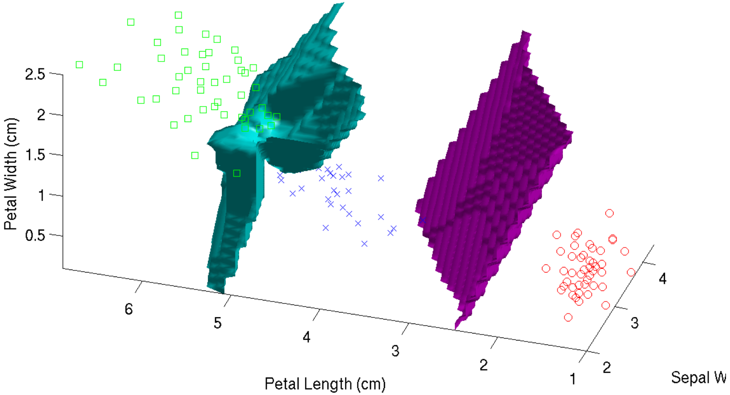
k-Nearest Neighbor, 3D

$$k = 1$$



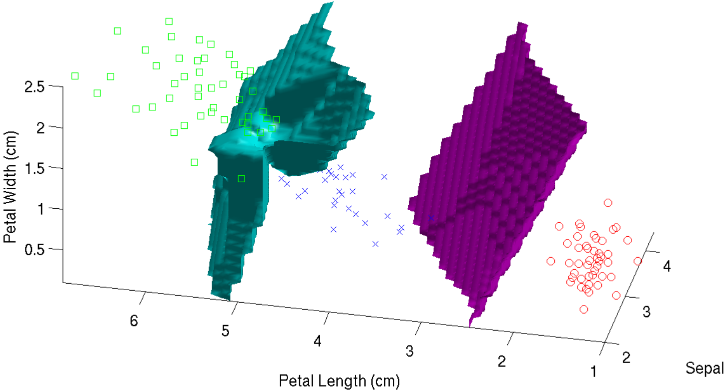
k-Nearest Neighbor, 3D

$$k = 1$$



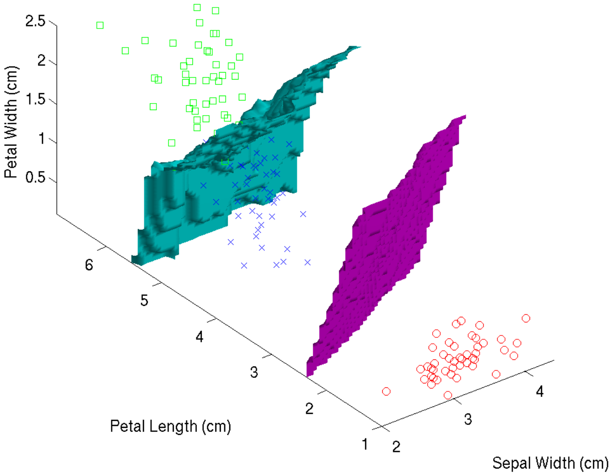
k-Nearest Neighbor, 3D

$$k = 1$$



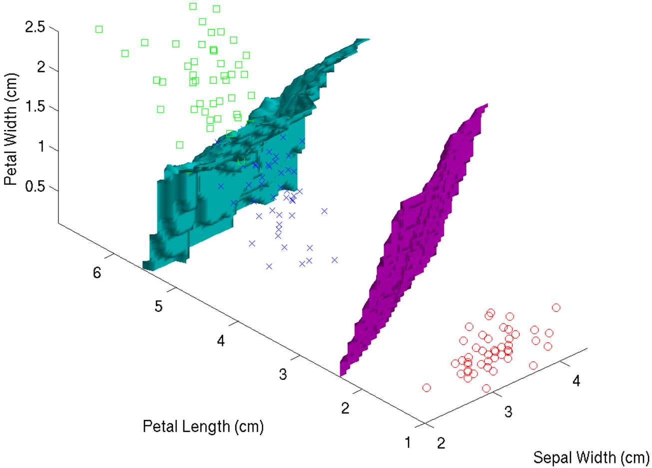
k-Nearest Neighbor, 3D

$$k = 5$$



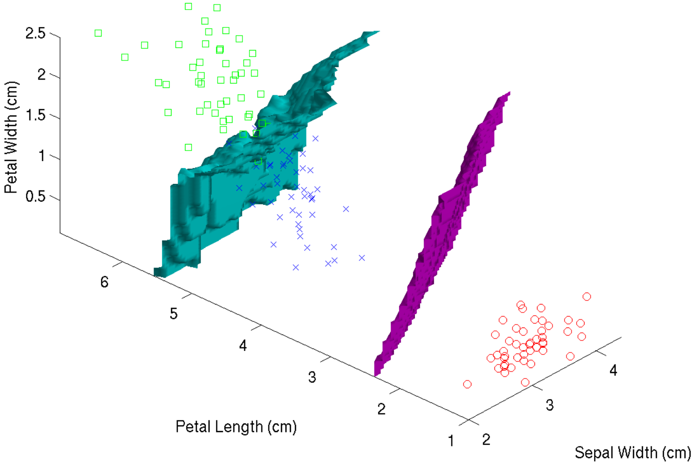
k-Nearest Neighbor, 3D

$k = 5$



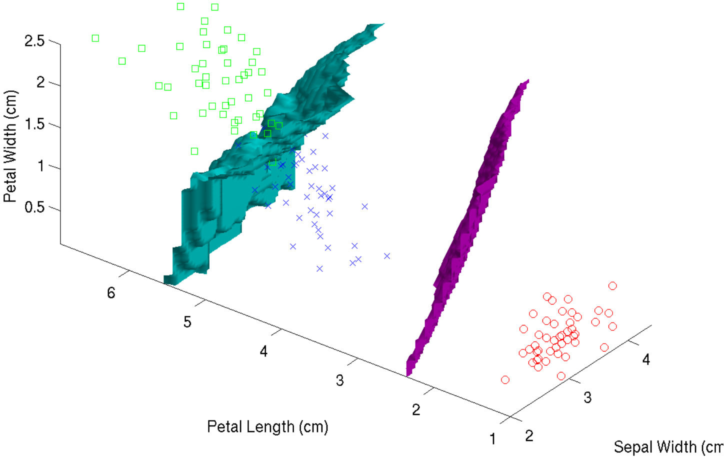
k-Nearest Neighbor, 3D

$$k = 5$$



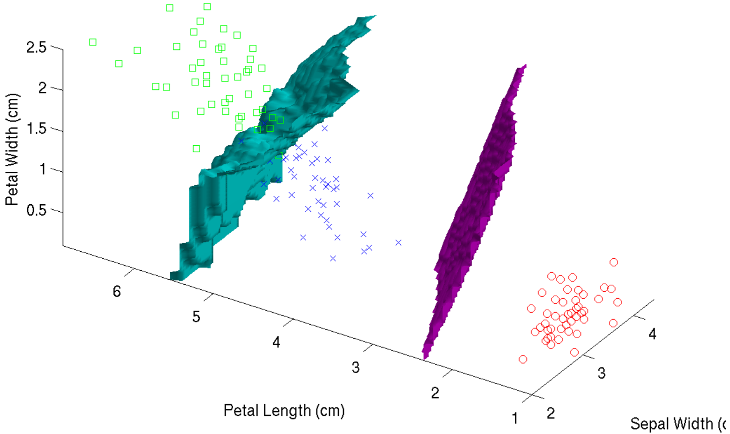
k-Nearest Neighbor, 3D

$k = 5$



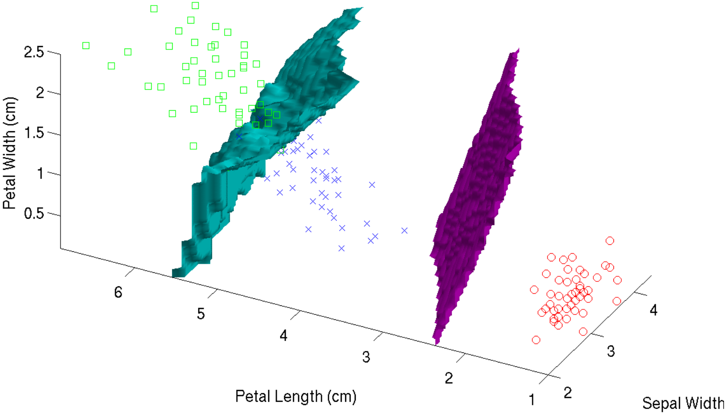
k-Nearest Neighbor, 3D

$$k = 5$$



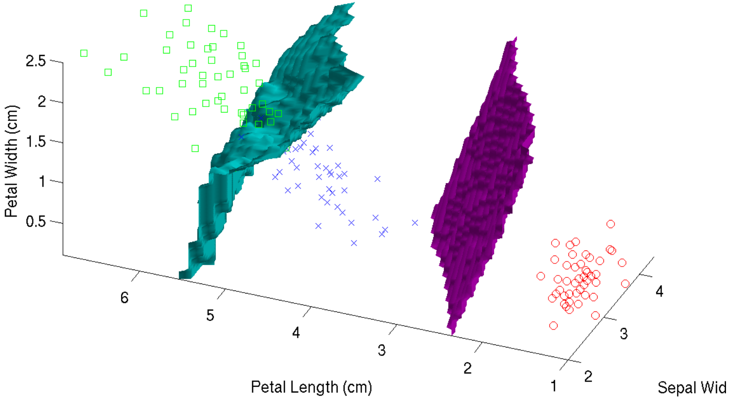
k-Nearest Neighbor, 3D

$k = 5$



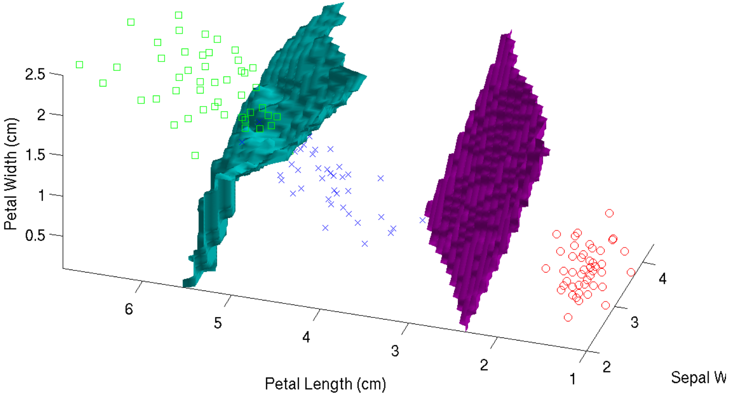
k-Nearest Neighbor, 3D

$$k = 5$$



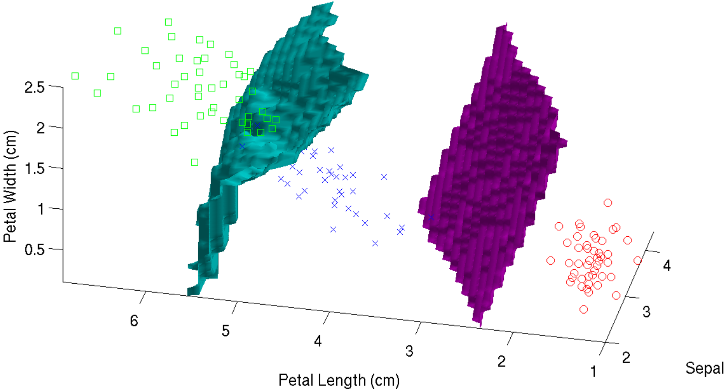
k-Nearest Neighbor, 3D

$$k = 5$$



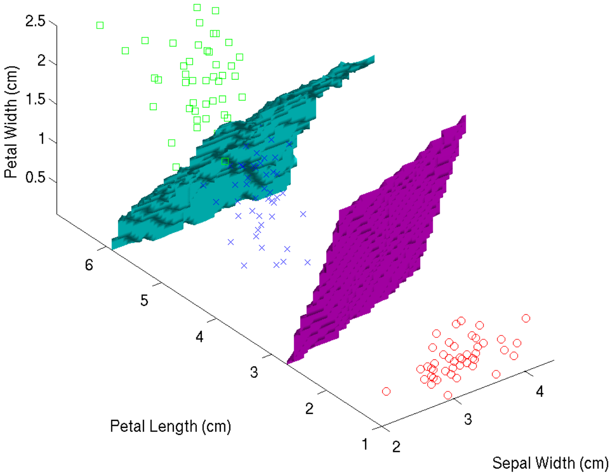
k-Nearest Neighbor, 3D

$$k = 5$$



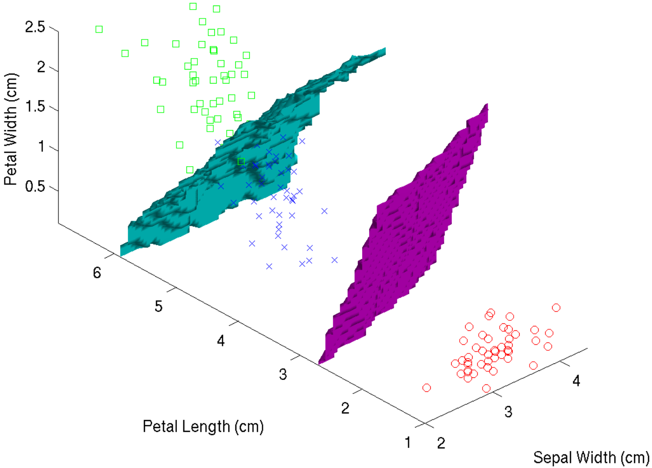
k-Nearest Neighbor, 3D

$$k = 15$$



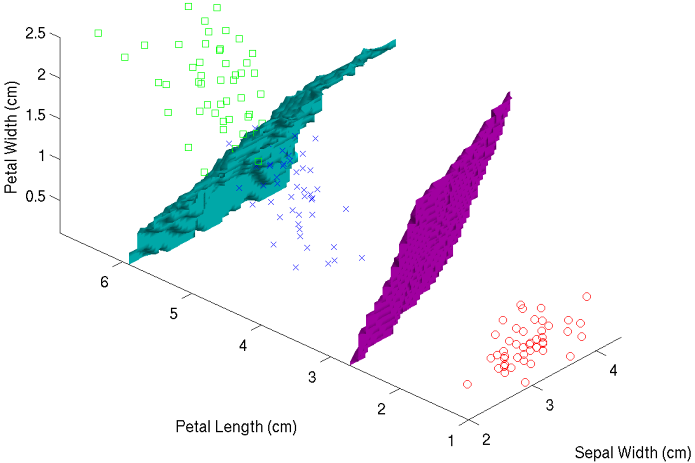
k-Nearest Neighbor, 3D

$$k = 15$$



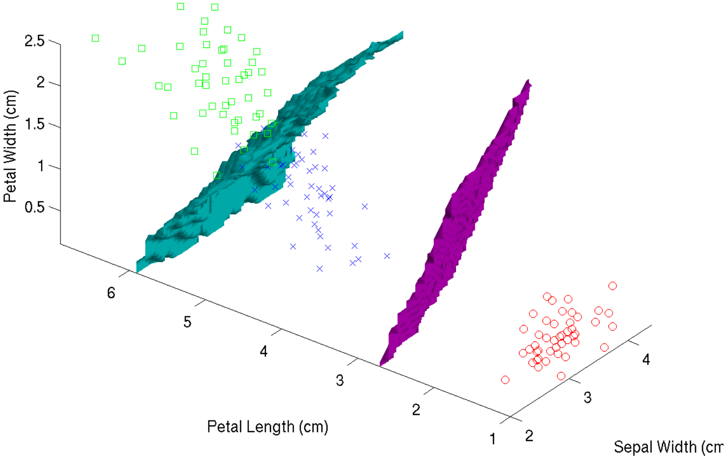
k-Nearest Neighbor, 3D

$k = 15$



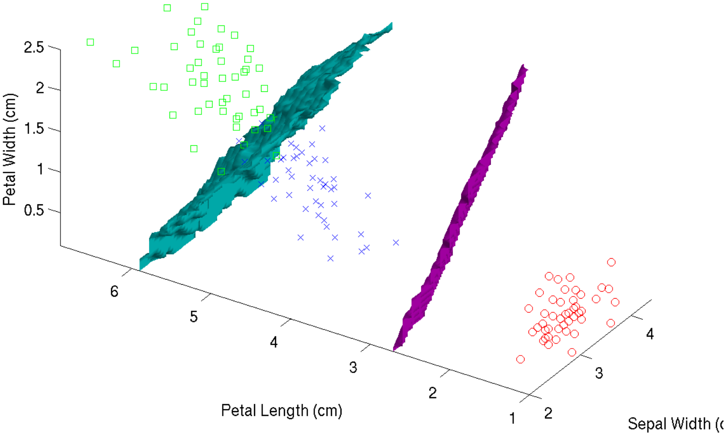
k-Nearest Neighbor, 3D

$k = 15$



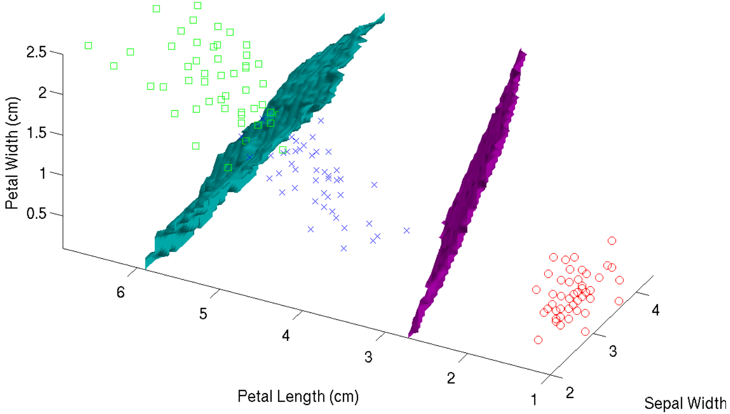
k-Nearest Neighbor, 3D

$$k = 15$$



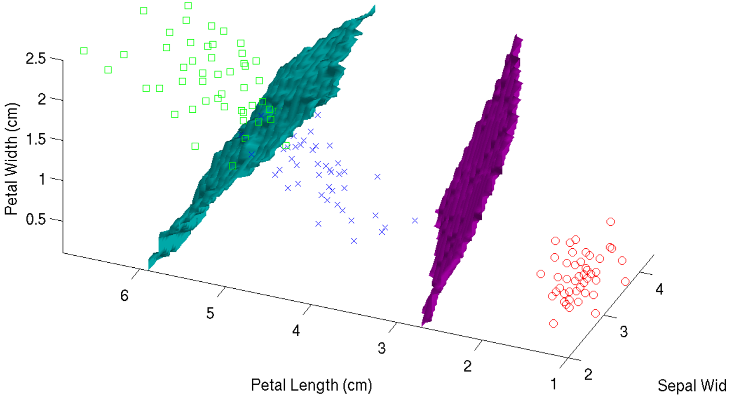
k-Nearest Neighbor, 3D

$$k = 15$$



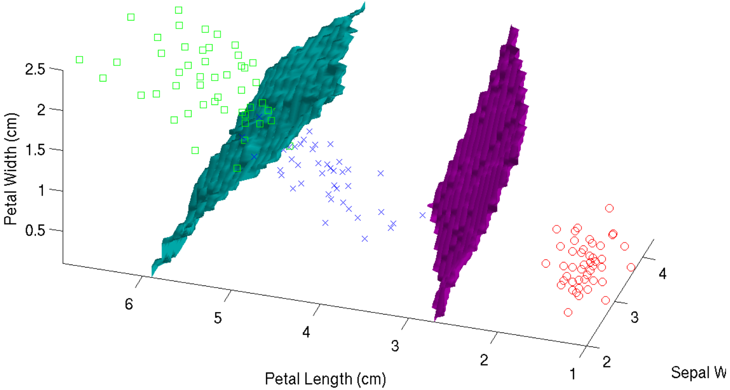
k-Nearest Neighbor, 3D

$$k = 15$$



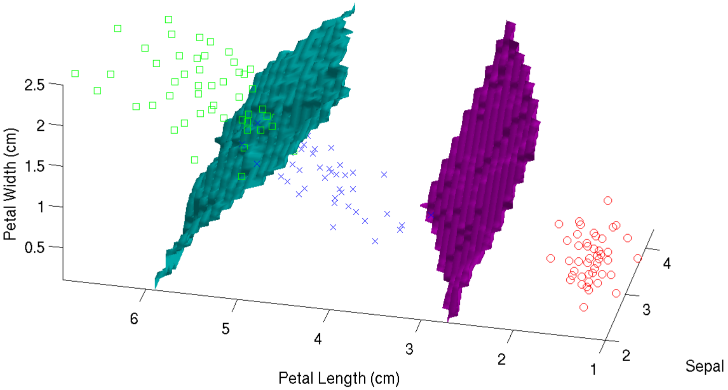
k-Nearest Neighbor, 3D

$$k = 15$$



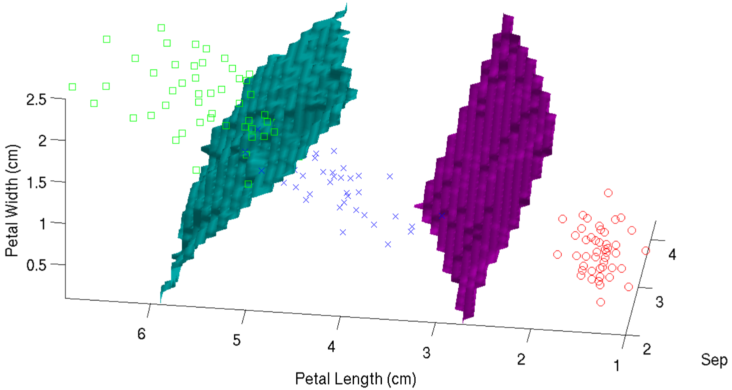
k-Nearest Neighbor, 3D

$$k = 15$$



k-Nearest Neighbor, 3D

$k = 15$



k-Nearest Neighbor Notes

- Point a and point b have (Euclidean) distance

$$d(a,b) = \sqrt{\sum_i (a_i - b_i)^2}$$

- A lazy method: no work done at training time, all work done at testing time

1-NN versus Bayes Optimal

If we knew $P(y \mid x)$ we could produce the Bayes-optimal classifier.
How good would it be?

1-NN versus Bayes Optimal

If we knew $P(y \mid x)$ we could produce the Bayes-optimal classifier.
How good would it be?

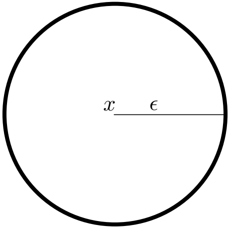
$$\text{error}_{\text{opt}}(x) = \min_y (1 - P(y \mid x))$$

1-NN versus Bayes Optimal

If we knew $P(y \mid x)$ we could produce the Bayes-optimal classifier.
How good would it be?

$$\text{error}_{\text{opt}}(x) = \min_y (1 - P(y \mid x))$$

How well will 1-NN do as $m \to \infty$?

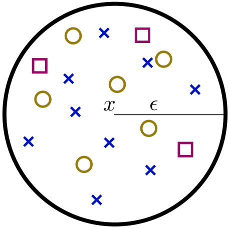


1-NN versus Bayes Optimal

If we knew $P(y \mid x)$ we could produce the Bayes-optimal classifier.
How good would it be?

$$\text{error}_{\text{opt}}(x) = \min_y (1 - P(y \mid x))$$

How well will 1-NN do as $m \to \infty$?

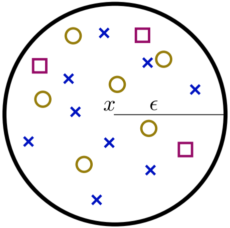


1-NN versus Bayes Optimal

If we knew $P(y \mid x)$ we could produce the Bayes-optimal classifier.
How good would it be?

$$\text{error}_{\text{opt}}(x) = \min_y (1 - P(y \mid x))$$

How well will 1-NN do as $m \to \infty$?

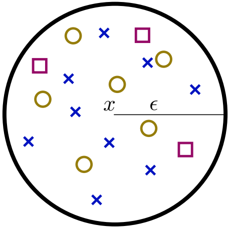


1-NN versus Bayes Optimal

If we knew $P(y \mid x)$ we could produce the Bayes-optimal classifier.
How good would it be?

$$\text{error}_{\text{opt}}(x) = \min_y (1 - P(y \mid x))$$

How well will 1-NN do as $m \to \infty$?



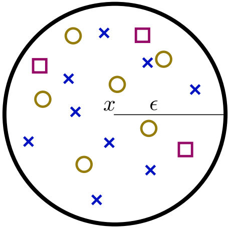
$$\text{error}_{\text{1-NN}}(x) = \sum_y P(y \mid x)\,(1 - P(y \mid x))$$

1-NN versus Bayes Optimal

If we knew $P(y \mid x)$ we could produce the Bayes-optimal classifier.
How good would it be?

$$\text{error}_{\text{opt}}(x) = \min_y (1 - P(y \mid x))$$

How well will 1-NN do as $m \to \infty$?



$$\text{error}_{\text{1-NN}}(x) = \sum_y P(y \mid x)\,(1 - P(y \mid x))$$

$$\text{error}_{\text{opt}} \le \text{error}_{\text{1-NN}}(x) \le 2\text{error}_{\text{opt}}$$

k-NN versus Bayes Optimal

What about k-NN?

k-NN versus Bayes Optimal

What about k-NN?

k must be a function of m (number of examples) to get consistency.

If

- $\lim_{m\to\infty} k(m) = \infty$, and
- $\lim_{m\to\infty} k(m)/m = 0$

then, k-NN converges to the Bayes-optimal error rate

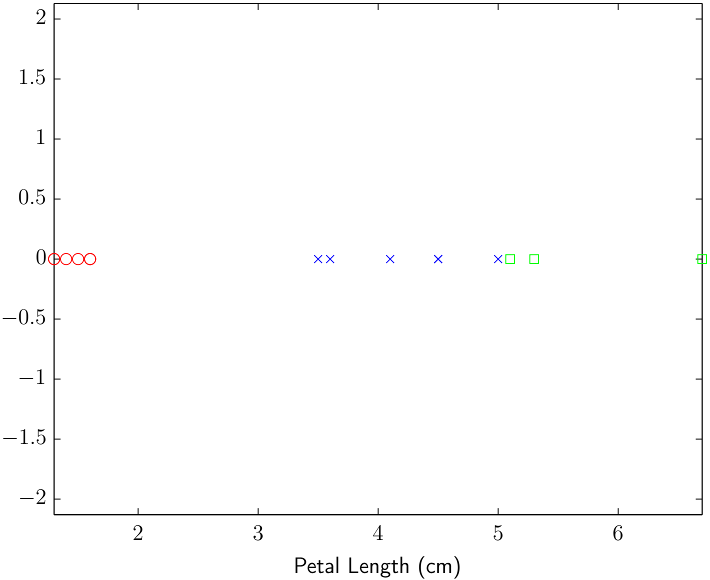
k-NN versus Bayes Optimal

What about k-NN?
k must be a function of m (number of examples) to get consistency.
If

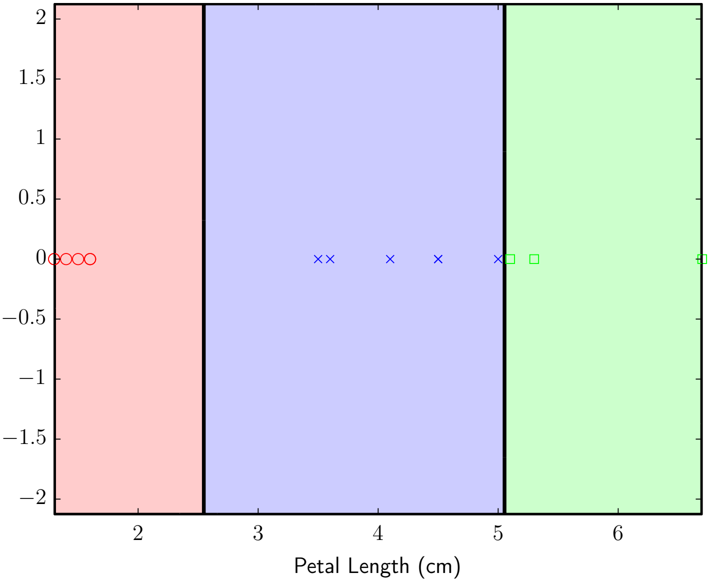
- $\lim_{m\to\infty} k(m) = \infty$, and
- $\lim_{m\to\infty} k(m)/m = 0$

then, k-NN converges to the Bayes-optimal error rate
But if $m < \infty$, almost nothing is known.

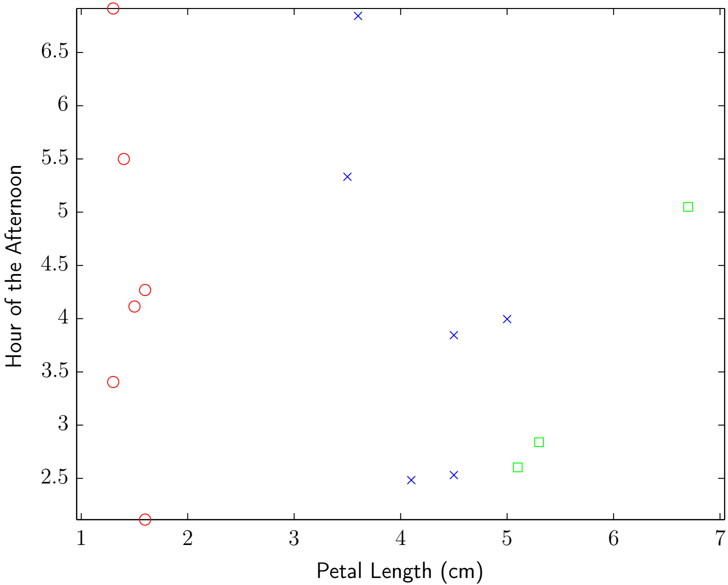
Irrelevant attributes



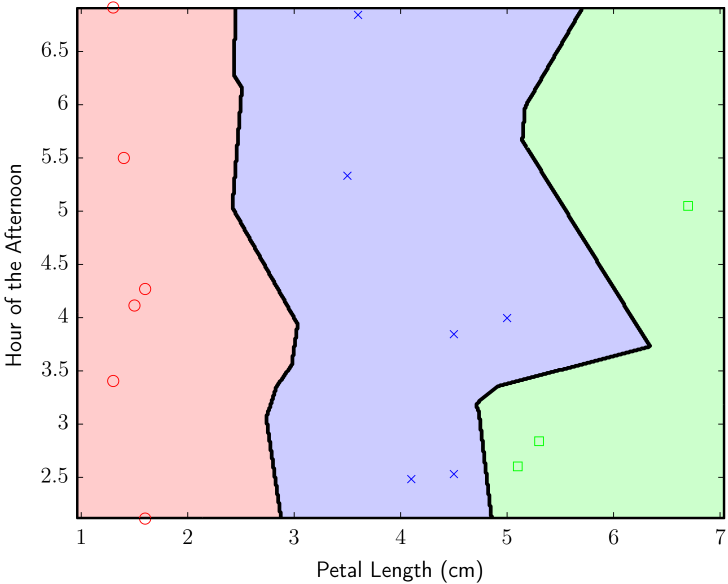
Irrelevant attributes



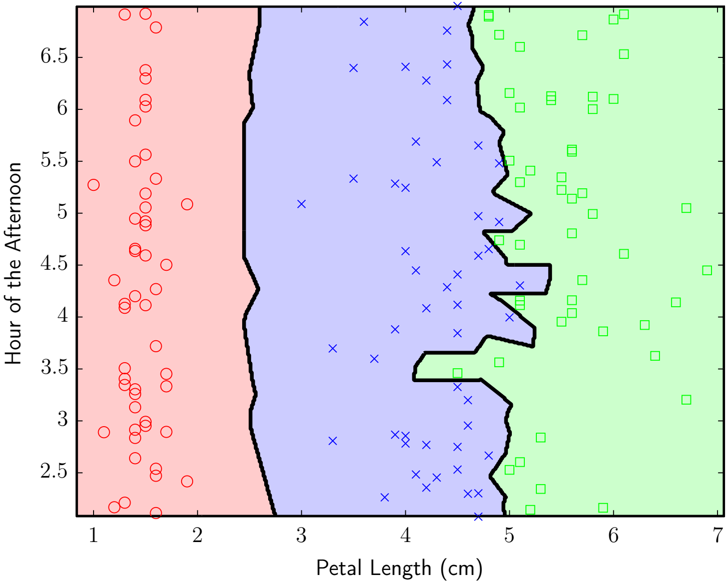
Irrelevant attributes



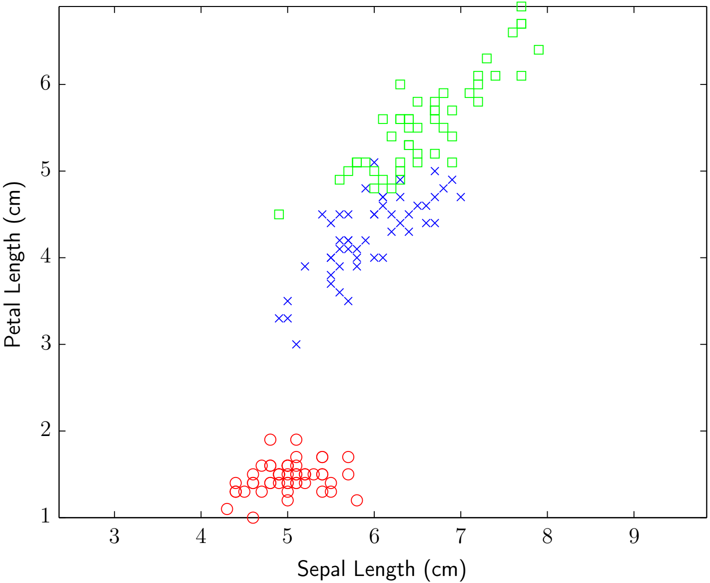
Irrelevant attributes



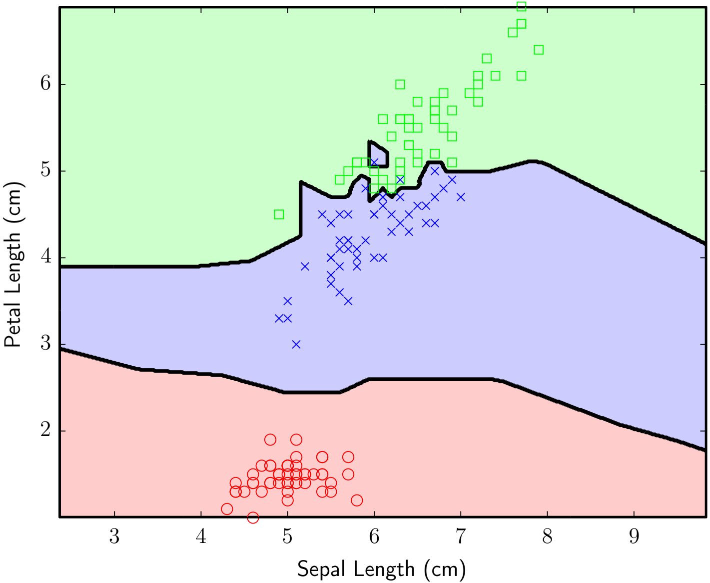
Irrelevant attributes



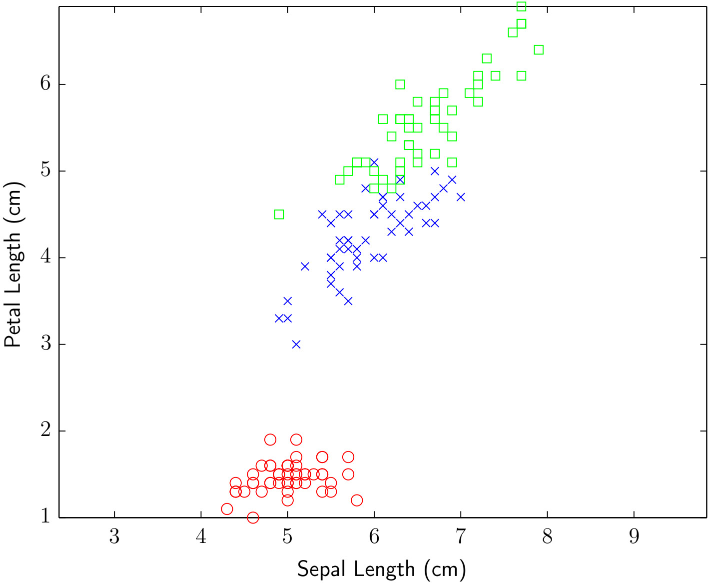
Attribute scaling



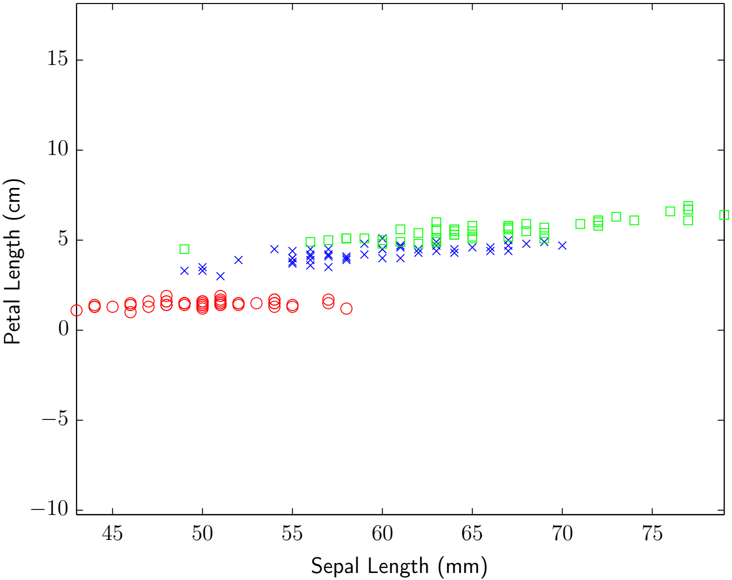
Attribute scaling



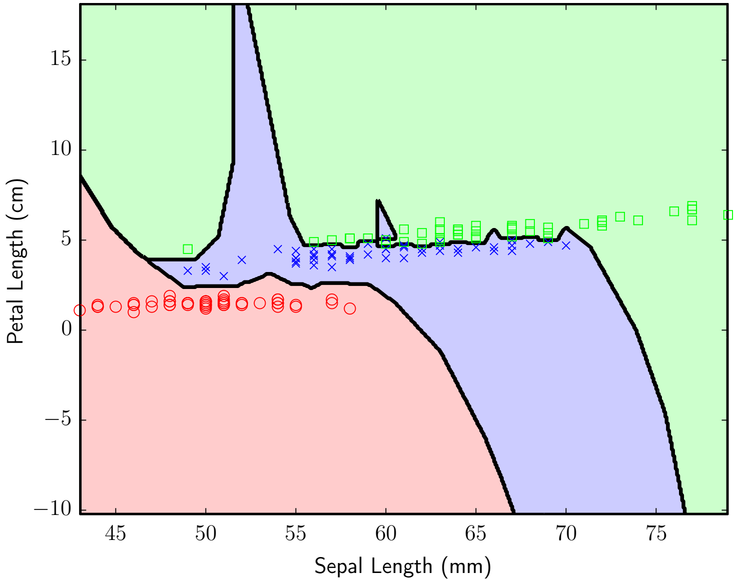
Attribute scaling



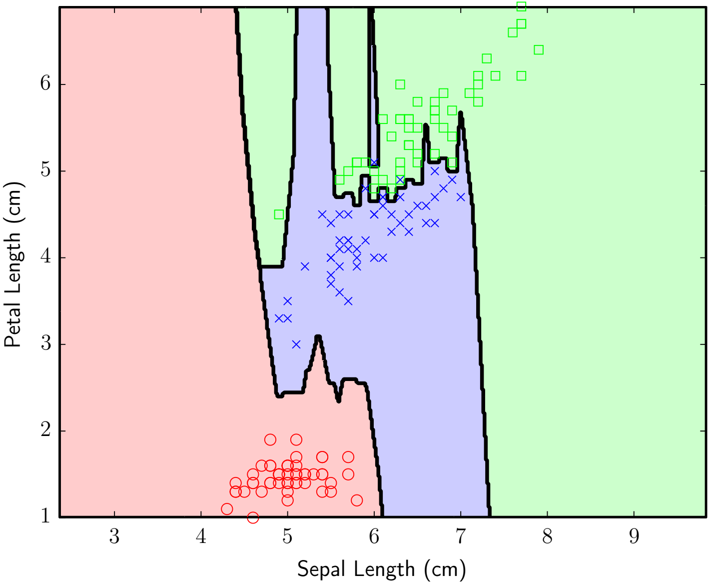
Attribute scaling



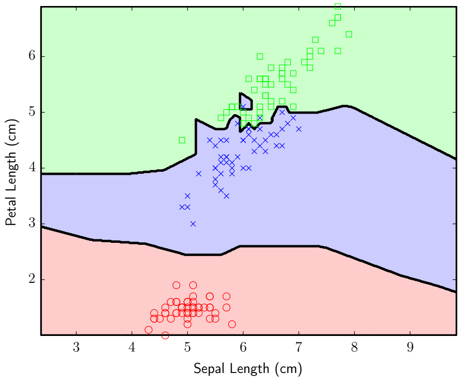
Attribute scaling

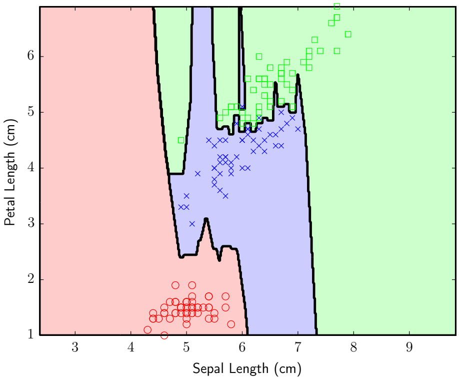


Attribute scaling



Attribute scaling





Attribute scaling

- Not as simple as "pick same units for all attributes"
  - What about temperature and length?
  - It petal length really the same as sepal length?
- What about discrete attributes?
  - Need distance between them
  - Binary can be 0 if same, 1 if different, but then should it be scaled?
  - Non-binary may be ordinal or categorical
- Irrelevant attributes are just an extreme example (scaling should be 0!)

Attribute scaling

Given two points, a training point $x = [x_1, x_2]$ and a testing point $z = [z_1, z_2]$, 2D Euclidean distance:

$$d(x, z) = \sqrt{(x_1 - z_1)^2 + (x_2 - z_2)^2}$$

Attribute scaling

Given two points, a training point $x = [x_1, x_2]$ and a testing point $z = [z_1, z_2]$,
2D Euclidean distance:

$$d(x, z) = \sqrt{(x_1 - z_1)^2 + (x_2 - z_2)^2}$$

If we scale the first attribute by 10 (say measure in mm instead of cm):
2D Euclidean distance:

$$d(x, z) = \sqrt{(10x_1 - 10z_1)^2 + (x_2 - z_2)^2} = \sqrt{100(x_1 - z_1)^2 + (x_2 - z_2)^2}$$

which scales the importance of similarity (or "exaggerates" the dissimilarity) of attribute 1.

Distance Metrics

The Euclidean distance isn't the only method to measure the dissimilarity of two points. Here are some common distance metrics:

- Euclidean (also known as the L_2 metric): [1]

$$d(x,z) = \left(\sum_{i=1}^{n} (x_i - z_i)^2 \right)^{1/2}$$

- Manhattan (also known as the L_1 metric):

$$d(x,z) = \left(\sum_{i=1}^{n} |x_i - z_i| \right)$$

- String edit distance: Given two strings (not vectors), the minimum number of edits (insert symbol, delete symbol, change symbol) necessary to change one string into the other.
- Graph edit distance: similar to strings, but with changes to measure the distance between two graphs

[1] The square root is not necessary if we only need to calculate which is closer; using the squared distance is equivalent for this purpose.

Which metric/scaling

- Even if you don't explicitly pick a scaling or metric, you are implicitly picking one.
- Irrelevant attributes is an extreme case of needing to scale the attribute (by 0)
- Euclidean distance is rotational invariant, but often this is not necessary.
- Selection is a way of injecting your own knowledge into the problem to help the learning.
  - Best metric is one that already "solves" the problem and gives a distance of 0 to members of the same class.
- k-NN (with k properly chosen) will converge to the optimal solution with ∞ data regardless.
- However, with finite data (the common case!) metric matters.