CMP-6002B - Machine Learning

Dr Gavin Cawley

Lecture 3 : Nearest Neighbour Classifiers

Introduction

- In the last lecture:
  - Basic principles.
  - Univariate classifiers.
- In this lecture:
  - Nearest Neighbour (NN) classifiers.
- Objectives:
  - Introduce the principle of k-NN classifiers.
  - Understand its basic operation and application issues.
- Desired Outcomes:
  - Ability to describe informally the motivation for k-NN.
  - Ability to perform a basic k-NN by hand.
  - Understanding of the strengths and weaknesses of k-NN.
  - Understanding of the issues in a real application of k-NN.

Nearest Neighbour Classifiers

- One of the oldest machine learning algorithms.

  E. Fix and J. L Hodges, *Discriminatory analysis, non-parametric discrimination: consistency properties*, Tech Report for USAF School of aviation and medicine, Randolph Field, 1951.

- Fast, simple and effective in many applications.
- Non-probabilistic (but can estimate posterior probabilities).
- Very widely used as a baseline for evaluation.
- Lecture topics:
  - Overview of k-NN.
  - Strengths and weaknesses.
  - Implementation issues for k-NN.
  - Application issues.
  - Extensions to the basic algorithm.

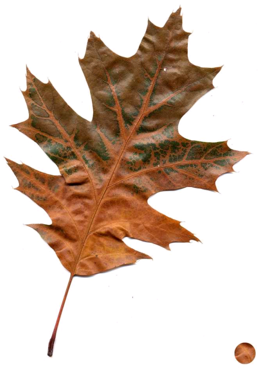
Joseph Lawson Hodges Jr.

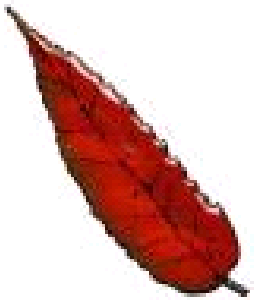
Example Image Recognition Task

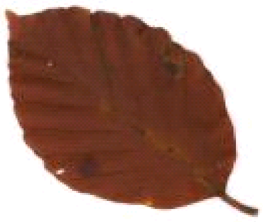
Does this image contain an Oak, Willow or Beech tree?

Feature Selection

- Look for distinguishing features that you have previously noted belong to their type
- In this case leaf shapes might work nicely:


Oak

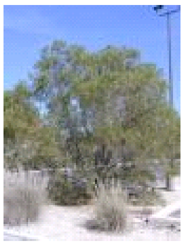

Willow


Beech

- Basic idea: Find the most similar tree you have seen before, assume this tree belongs to the same species.

Collect Training Data

(oak)


(willow)

(beech)

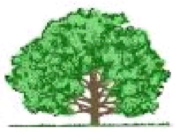
(beech)

(willow)

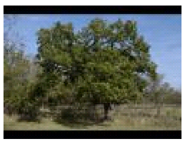
(willow)

(willow)

(oak)


(oak)

(beech)


(oak)

(beech)

Nearest Neighbour in Operation

test case

OAK

closest training example

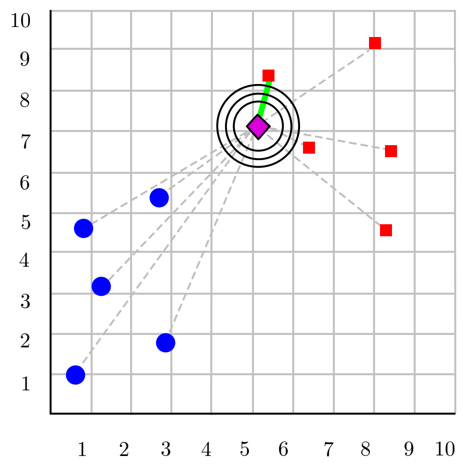
Find closest (somehow) and label new one the same.

Verdict: Oak

Assessing Similarity

- Nearest neighbour methods are based on a similarity metric.
- The Euclidean distance is the most common

$$D_{\text{Euclid}}(Q, C) = \sqrt{\sum_{i=1}^{d} (q_i - c_i)^2}$$



- Q is the query pattern.
- C is a pattern from the training corpus.
- No need for the square root.
  - Sum of squares distance.
- Many other distance metrics are feasible.

Some Computational Learning Theory

- CoLT aims to establish bounds on the generalisation performance of classifier systems.
- The Bayes error, R^*, is the lowest error achievable.
- For the NN classifier, Cover and Hart[1] proved that as the number of training patterns becomes large

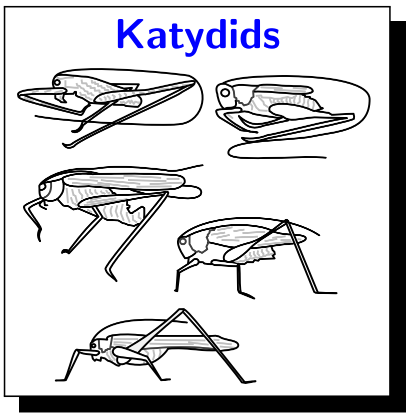
$$R^* < R \leq R^*(2 - MR^*/(M-1))$$

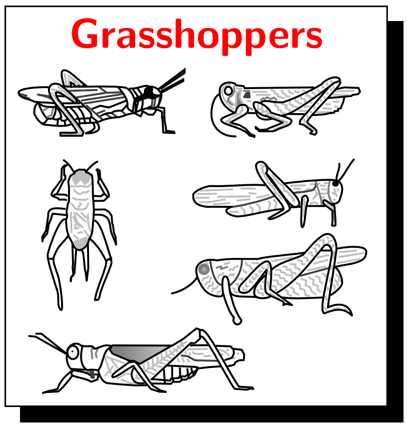
  where R is the error rate and M is the number of classes.
- This is an *asymptotic bound*.
- For Binary classification $R^* < R \leq 2R^*$.
- Nearest neighbour contains half of the information!

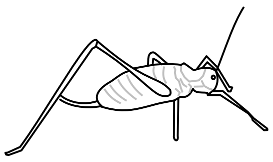
[1] T. M. Cover and P. E. Hart, *Nearest Neighbor Pattern Classification*, IEEE Transactions on Information Theory, vol. IT-13, pages 21-27, January 1967.

Example: Insect Classification

Given a collection instances **Katydids** and of **Grasshoppers**, decide what type of insect the unlabelled example is



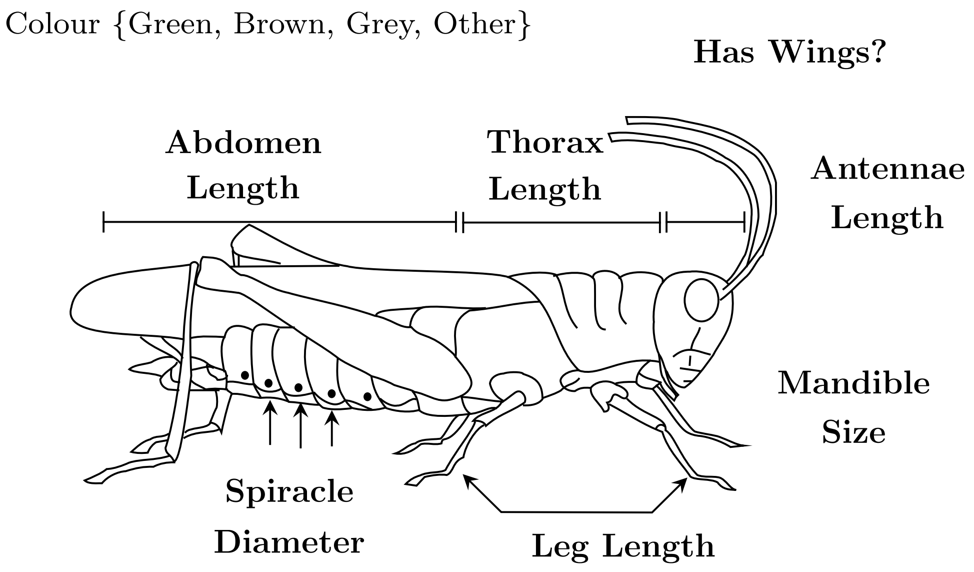




Katydid or **Grasshopper**?

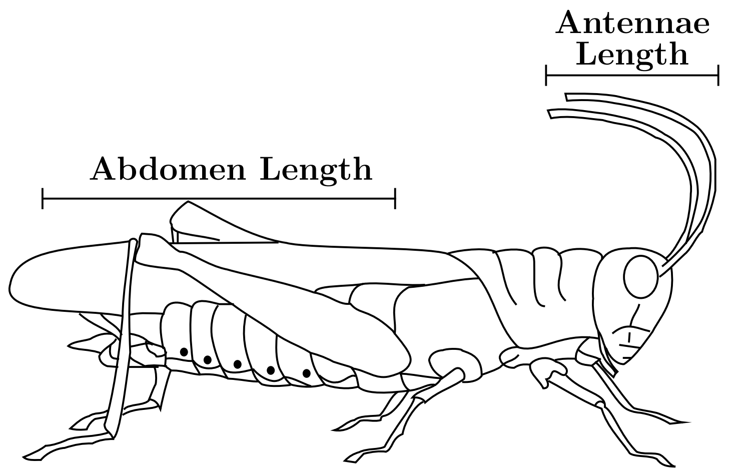
Decide Relevant Features

For any domain of interest, we can measure a variety of *features*



Get Expert Advice

- The entomologist says two features are relevant:
  - Abdomen length.
  - Antennae length.



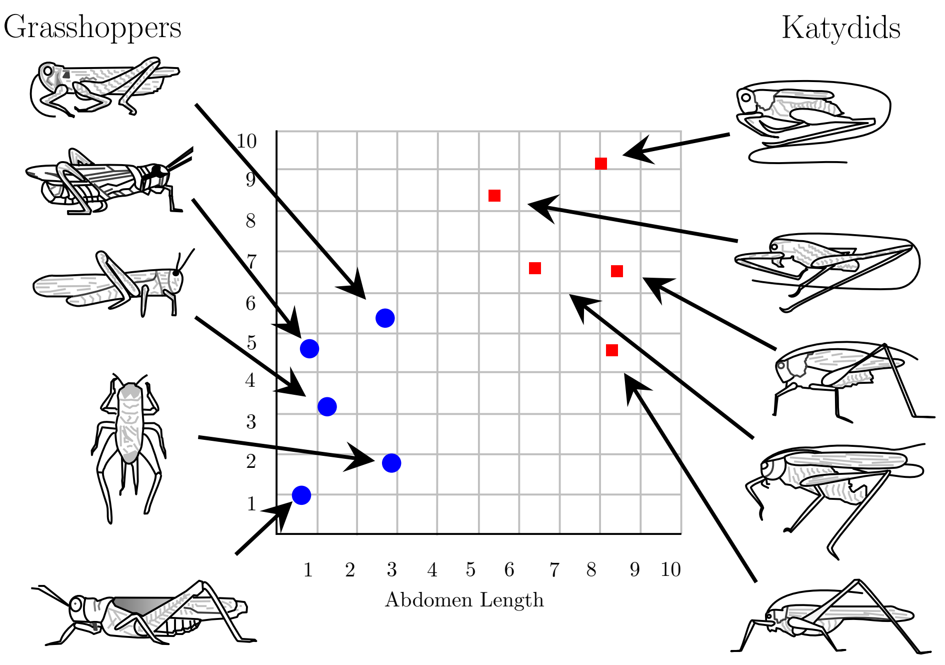
Gather Measurements on Training and Test Examples

Insect ID	Abdomen Length	Antennae Length	Insect Class
1	2.7	5.5	Grasshopper
2	8.0	9.1	Katydid
3	0.9	4.7	Grasshopper
4	1.1	3.1	Grasshopper
5	5.4	8.5	Katydid
6	2.9	1.9	Grasshopper
7	6.1	6.6	Katydid
8	0.5	1.0	Grasshopper
9	8.3	6.6	Katydid
10	8.1	4.7	Katydids

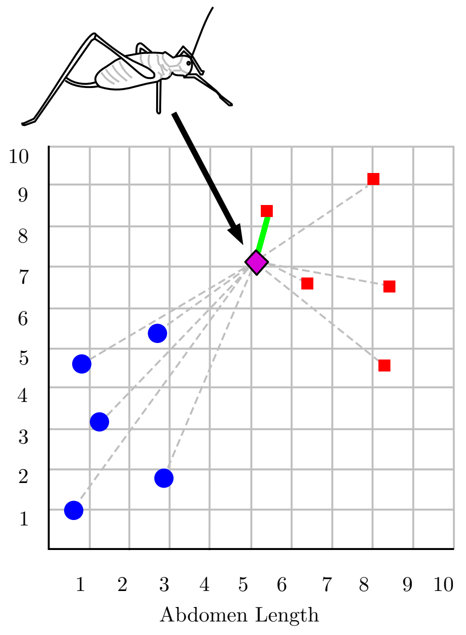
Previously unseen case

	Abdomen Length	Antennae Length	Insect Class
	5.1	7.0	??

Graphical Representation



Nearest Neighbour Classification



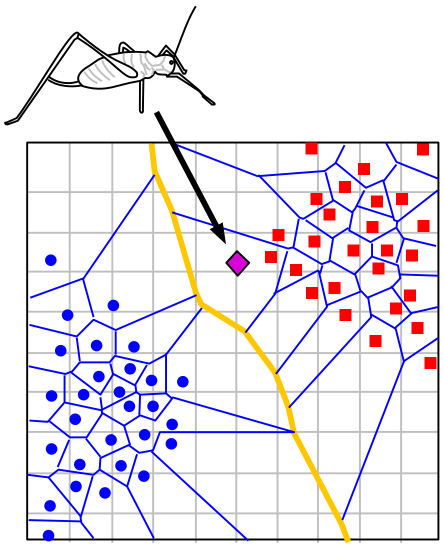
Verdict: Katydid.

Computing the Euclidean Distance

	New Case			5.1	7	
	Data	A	B			Distance
Grasshopper	1	2.7	5.5	5.76	2.25	8.01
Grasshopper	3	0.9	4.7	17.64	5.29	22.93
Grasshopper	4	1.1	3.1	16	15.21	31.21
Grasshopper	6	2.9	1.9	4.84	26.01	30.85
Grasshopper	8	0.5	1	21.16	36	57.16
Katydid	2	8	9.1	8.41	4.41	12.82
Katydid	5	5.4	8.5	0.09	2.25	2.34
Katydid	7	6.1	6.6	1	0.16	**1.16**
Katydid	9	8.3	6.6	10.24	0.16	10.4
Katydid	10	8.1	4.7	9	5.29	14.29

Decision Surface for 1-NN

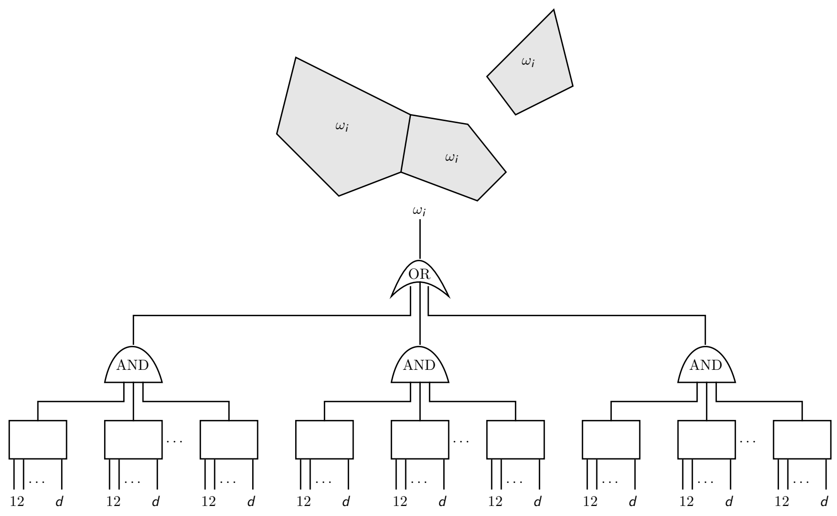
- We can visualize the nearest neighbour algorithm in terms of a decision surface.
- Note that we don't actually have to construct these surfaces.



- Implicit boundaries delimit regions within which every point is closer to one training point than all of the others.
- Known as:
  - Dirichlet Tessallation.
  - Voronoi diagram.
  - Theissen regions.

Computational Complexity

- Obvious implementation $\mathcal{O}(n)$ operations where n is the number of training patterns.
- However, Voronoi diagram suggest $\mathcal{O}(1)$ parallel implementation
  - Constant time, but $\mathcal{O}(n)$ hardware resources!



The k-Nearest Neighbour Classifier

- Rather than look at only the nearest neighbour, look at the labels of the k most similar patterns.
- The simplest voting system is majority vote
  - Choose the class label as the one most represented.
  - Good idea to make k odd!
- For example, if $k = 3$, the closest cases are 1, 5 and 7.

Grasshopper	1	2.7	5.5	5.76	2.25	8.01
Katydid	5	5.4	8.5	0.09	2.25	2.34
Katydid	7	6.1	6.6	1	0.16	**1.16**

- Since two neighbours are Katydid and only one is Grasshopper, we would still classify the case as Katydid.
- Why is this a good idea?

More Computational Learning Theory

- Duda *et al.*[2] show that as the number of training patterns and k become large, the error rate of the k-NN classifier approaches the Bayes error, i.e.

$$\lim_{k,n\to\infty} R = R^*.$$

- This is intuitively reasonable:
  - Consider an infinitesimally small region.
  - Conditional probabilities constant across region.
  - As $n \to \infty$, the region becomes well-populated.
  - As $k \to \infty$ the nearest neighbours accurately represent the relative frequencies within the region.
  - If we use the 1-NN algorithm, this will not be the case!

[2] R.O. Duda, P.E. Hart & D.G. Stork, *Pattern Classification*, Wiley, 2001.

Estimating A-Posteriori Probabilities

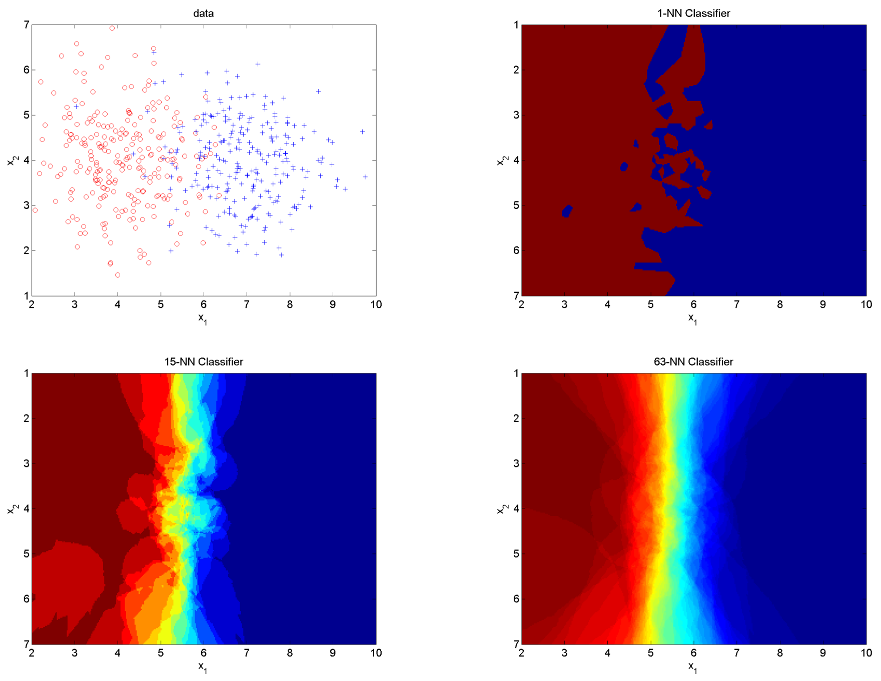
- The nearest neighbour (1-NN) classifier is purely discrete.
- Estimates of posterior probability can be obtained for k-NN classifier.

  $$p(\mathcal{C}_i|\boldsymbol{x}) = \frac{k_i}{k}$$

  where k_i is the number of votes for class $\mathcal{C}_i$.
- Can then take advantage of decision theory:
  - Option to reject sample.
  - Dealing with disparate training and operational priors.
  - Dealing with misclassification costs.
- However, estimates are heavily quantised for small k.
- m-estimate may be beneficial for small k.

Generalisation and Overfitting

- k controls the complexity of the classifier.



- Tune k to achieve best generalisation.

Summary

- Assign pattern to the class best represented in the k most similar labelled examples seen so far.
- Requires definition of a suitable distance metric.
- Non-paramatric - decision boundary decided by the data.
- Generalisation bounds from computational learning theory.
- Estimates of posterior probability available for $k > 1$.
- Lazy or Instance Based learning.
  - No real training phase, just store the data.
  - Simple and fast.
- Linear scan through data for recall (Naïve implementation).
  - Slow for large training dataset.
- k controls complexity and so generalisation.

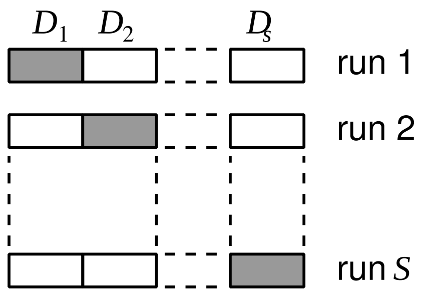
Choosing a Good Value for k

- Choose odd value to simplify voting.
  - Devijver[3] shows that with random tie breaking $2k$-NN gives the same results as $(2k - 1)$-NN.
- The value of k controls generalisation
  - Too small $\rightarrow$ over-fitting.
  - Too large $\rightarrow$ under-fitting.
- Simple solution - partition available data into three sets:
  - Training set - forms the corpus for the k-NN rule.
  - Validation set - used to choose the best value of k.
  - Test set - used to measure generalisation performance.
- Better solution - cross-validation

[3]P.A. Devijver, *A Note on Ties in Voting with the k-NN Rule*, Pattern Recognition, vol. 10, no. 1, pp. 297-298, 1978.

Crossvalidation

- Useful if only a limited amount of data is available.
- Partition the available data into S subsets of approximately equal size.
- Train S classifiers, the i^{th} classifier is tested on D_i and trained on the remainder of the data.
- Average test error over the S models.
- Chose the value of k minimising the average error.

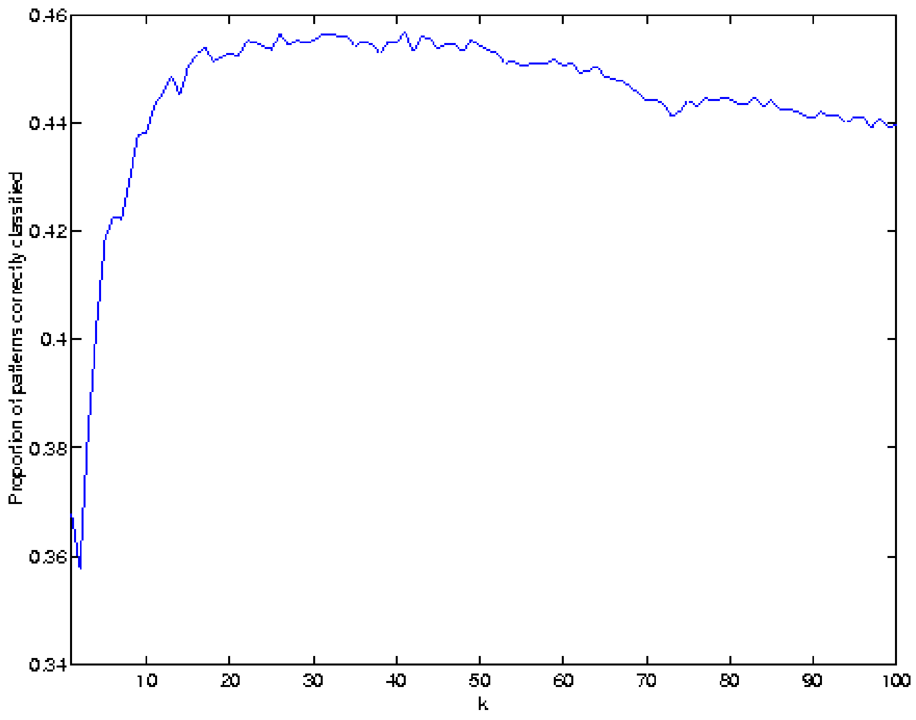


- Generally reliable.
- All data is used as training data and as test data.
- Use an ensemble of S classifiers in operation.

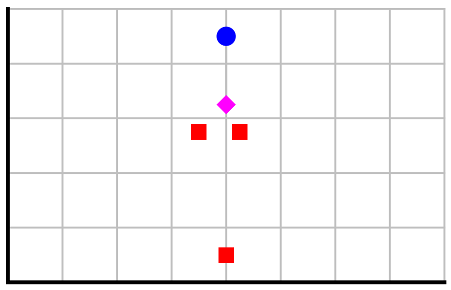
Leave-One-Out Cross-Validation

- Most extreme form of cross-validation
  - Each subset contains one training pattern.
  - Known as LOOCV.
- Can evaluate LOOCV for free as a by product of evaluating training set error.
  - Compute the distance between one training pattern and the entire training set.
  - Sort to give a ranked list.
  - Classify using the 2^{nd} to the $(k+1)^{th}$ in the list.
  - Repeat for all patterns in the training set and average errors.
- Very common to tune k by minimising LOOCV error.
  - Very efficient!
  - Effective (but slightly less reliable than e.g. 10-fold CV).

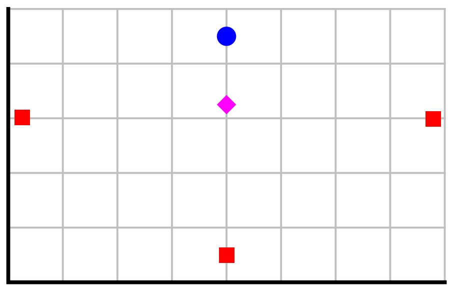
Example: Classifying Circulation Patterns



k-NN is Sensitive to Attribute Scaling



x axis is measured in **cm**, y axis measured in pounds. NN of the pink unknown instance is square (red).



x axis is measured in **mm**, y axis measured in pounds. NN of the pink unknown instance is circle (blue).

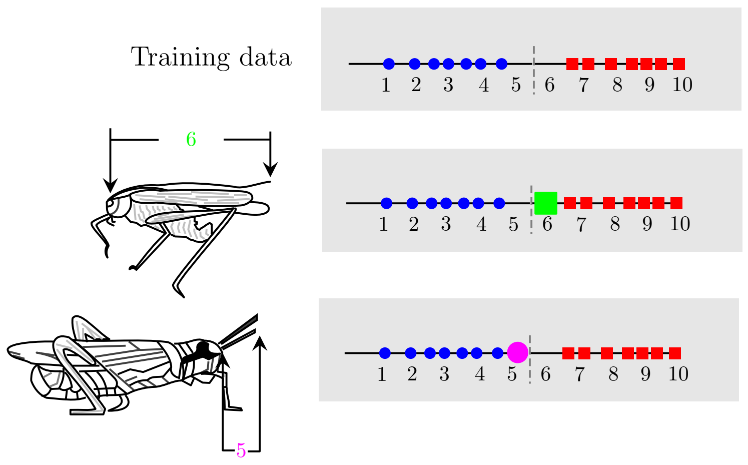
- One solution is to standardise the units,

$$X = (X - \text{mean}(X)) / \text{std}(X)$$

gives zero mean and unit variance attribute values.

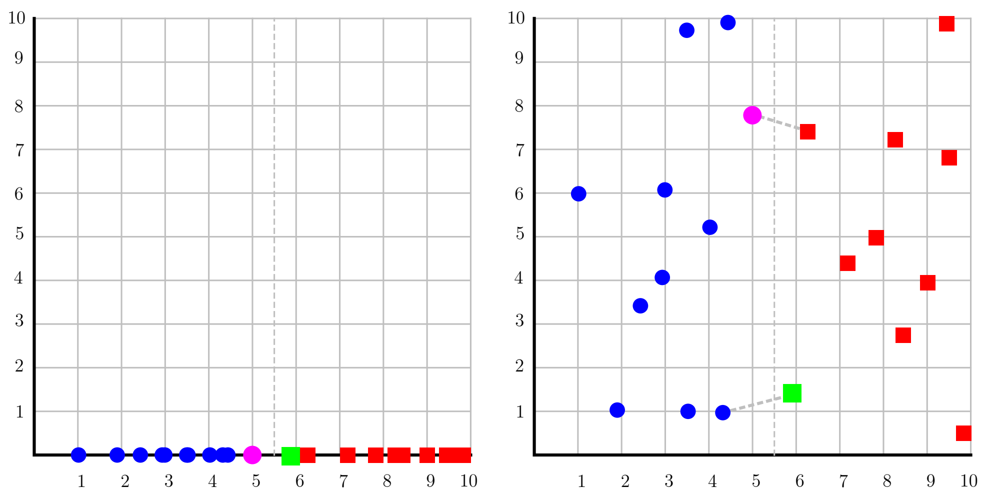
k-NN is Sensitive to Irrelevant Features

- Suppose that if an insects antenna is longer than 5.5 it is a **Katydid**, otherwise it is a **Grasshopper**.
- Using just the antenna length we get perfect classification



Sensitive to Irrelevant Features

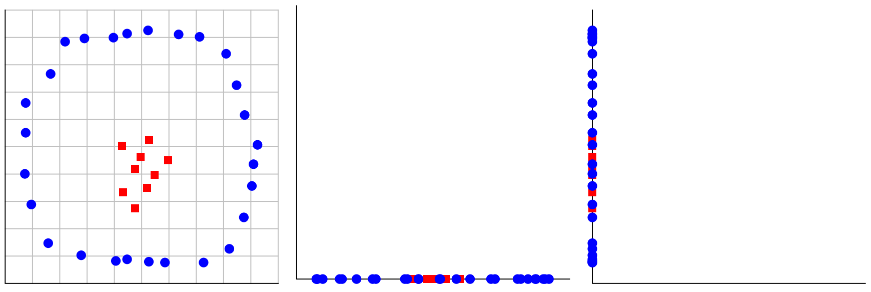
- Suppose however, we add in an **irrelevant** feature, for example the insect's mass.
- Using both the antenna length and the insect's mass with the 1-NN algorithm we get the wrong classification.



Attribute selection algorithms

k-NN is sensitive to irrelevant and/or redundant attributes. How do we overcome this?

- Ask an expert what features are relevant.
- Use more training instances
- Use statistical test to try to determine which features are useful.
- Search over feature subsets (this is hard)



Attribute Scaling Methods

- Use a *weighted* sum of squares distance metric,

$$d_{\mathrm{WSS}}(Q, C) = \sum_{i=1}^{d} \omega_i [q_i - c_i]^2.$$

- Choose weights to minimised e.g. the LOOCV error
  - Informative attributes gain a high weight.
  - Redundant attributes assigned a low weight.
  - Irrelevant attributes (hopefully) assigned a weight close to zero.
- With judicious approximations can use gradient descent.
- Deals with irrelevant and redundant features.
- Deals with attribute scaling problem.
- Prone to over-fitting if there are too many attributes.

Alternative voting schemes

- Most implementations use a simple majority vote.
- Weighted voting is also common, proposed by Dudani[4]
  - Nearer neighbors more informative than more distant ones.
  - Weight patterns according to distance, e.g.

$$w_i = \frac{1}{d(\boldsymbol{x}_q, \boldsymbol{x}_i)^2} \quad \text{or} \quad w_i = \frac{d_{max} - d(\boldsymbol{x}_q, \boldsymbol{x}_i)}{d_{max} - d_{min}}$$

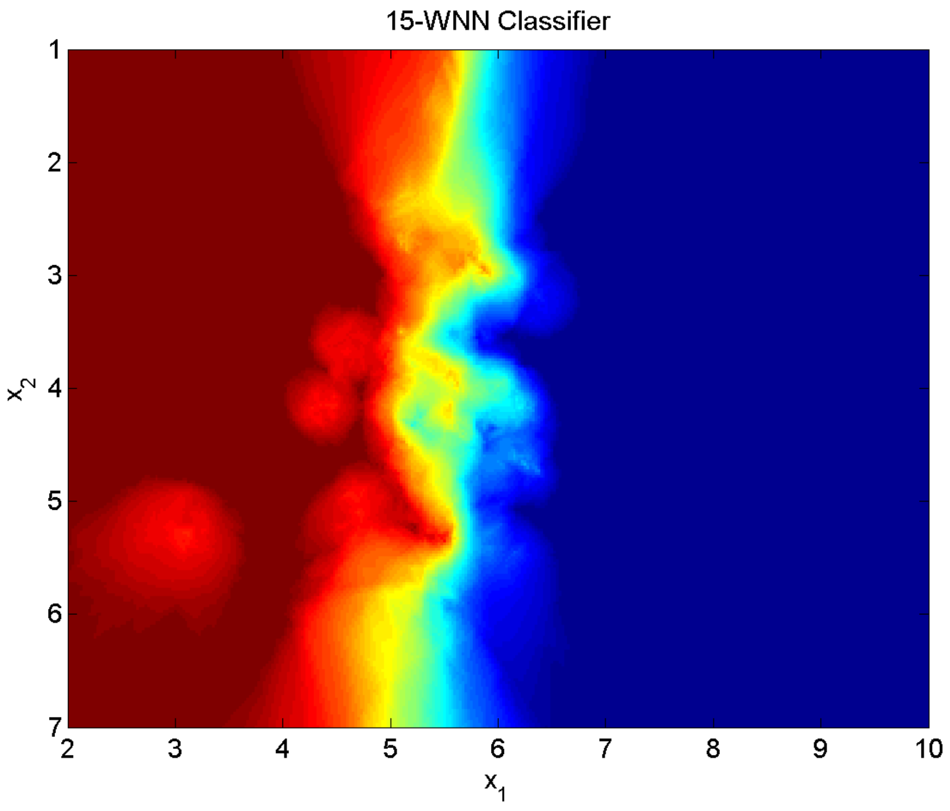
  - Probability estimate becomes

$$p(\mathcal{C}_i|\boldsymbol{x}) = \frac{\sum_{j=1}^{k} w_j t_j}{\sum_{j=1}^{k} w_j}$$

- Useful for disparate training set and operational priors.

[4]S.A. Dudani, *The Distance-Weighted k-Nearest Neighbor Rule*, IEEE Transactions on Systems, Man and Cybernetics, vol. SMC-6, no. 4, pp. 325-327, April 1976.

Example



k-NN Regression

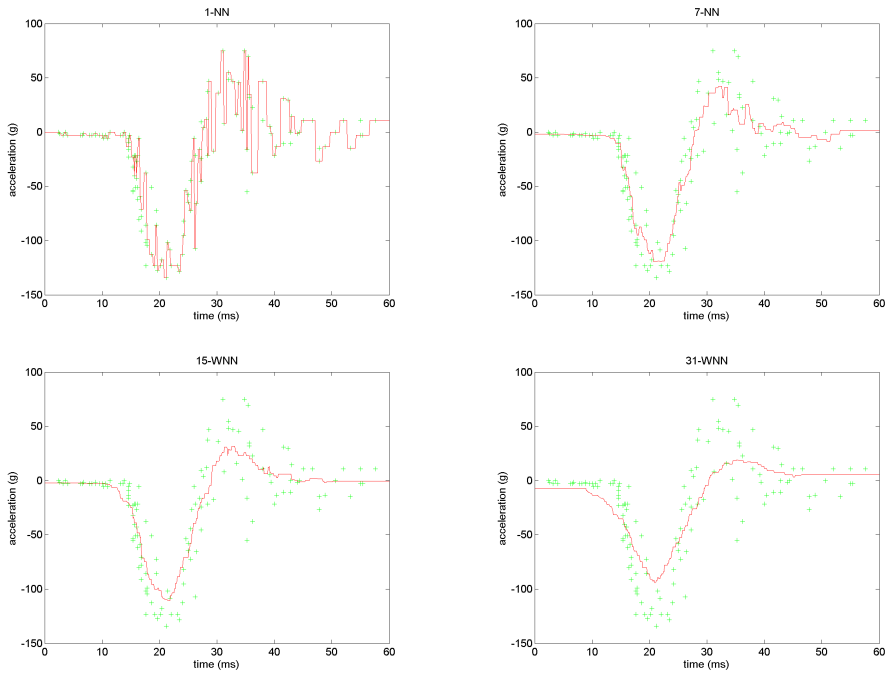
- *k*-NN can be used for real-valued responses as well
  - See e.g. Altman[5].
- Basic *k*-NN regression, output is mean of the responses for the *k* most similar training patterns.
  - Piecewise constant function.
- Weighted *k*-NN regression may be better

$$f(\boldsymbol{x}_q) = \frac{\sum_{j=1}^{k} w_j y_j}{\sum_{j=1}^{k} w_j} \quad \text{where} \quad w_j = \frac{d_{max} - d(\boldsymbol{x}_q, \boldsymbol{x}_j)}{d_{max} - d_{min}}$$

- Closely related to Parzen non-parametric regression.

[5] N.S. Altman, *An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression*, The American Statistician, vol. 46, no. 3., pp. 175-185, August 1992.

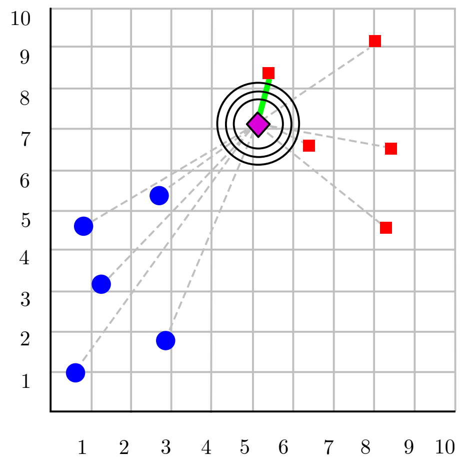
Example - The Motorcycle Dataset



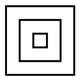
Alternative distance metrics

Even with continuous variables, we are not restricted to Euclidean distance, we can use any Minkowski distance.

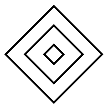
$$D(Q,C) \equiv \sqrt{\sum_{i=1}^{n}(q_i - c_i)^2} \qquad D(Q,C) \equiv \sqrt[p]{\sum_{i=1}^{n}(q_i - c_i)^p}$$



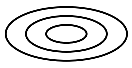
Max ($p = \inf$)



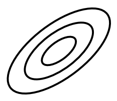
Manhattan ($p = 1$)



Weighted Euclidean



Mahalanobis



More Distance Metrics

Minkowsky

$$D(\mathbf{x},\mathbf{y}) = \left(\sum_{i=1}^{m}|x_i - y_i|^r\right)^{1/r}$$

Euclidean

$$D(\mathbf{x},\mathbf{y}) = \sqrt{\sum_{i=1}^{m}(x_i - y_i)^2}$$

Manhattan/city-block

$$D(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{m}|x_i - y_i|$$

Camberra

$$D(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{m}\frac{|x_i - y_i|}{|x_i + y_i|}$$

Chebychev

$$D(\mathbf{x},\mathbf{y}) = \max_{i=1}^{m}|x_i - y_i|$$

Yet More Distance Metrics

Quadratic

$$\begin{aligned} D(\boldsymbol{x}, \boldsymbol{y}) &= (\boldsymbol{x} - \boldsymbol{y})^T \boldsymbol{Q}(\boldsymbol{x} - \boldsymbol{y}) \\ &= \sum_{j=1}^{m} \left(\sum_{i=1}^{m} (x_i - y_i) q_{ji} \right) (x_j - y_j) \end{aligned}$$

$\boldsymbol{Q}$ is a problem-specific positive definite $m \times m$ weight matrix.

Mahalanobis

$$D(\boldsymbol{x}, \boldsymbol{y}) = [\det \boldsymbol{V}]^{1/m} (\boldsymbol{x} - \boldsymbol{y})^T \boldsymbol{V}^{-1} (\boldsymbol{x} - \boldsymbol{y})$$

$\boldsymbol{V}$ is the covariance of matrix of $A_1 \ldots A_m$, and A_j is the vector of values for attribute j occuring in the training set instances $1 \ldots n$

Even More Distance Metrics

Correlation

$$D(\boldsymbol{x}, \boldsymbol{y}) = \frac{\sum_{i=1}^{m} (x_i - \bar{x}_i)(y_i - \bar{y}_i)}{\sqrt{\sum_{i=1}^{m} (x_i - \bar{x}_i)^2 \sum_{i=1}^{m} (y_i - \bar{y}_i)^2}}$$

$\bar{x}_i = \bar{y}_i$ and is the average value for attribute i occurring in the training set.

Chi-square

$$D(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{m} \frac{1}{\text{sum}_i} \left(\frac{x_i}{\text{size}_x} - \frac{y_i}{\text{size}_y} \right)^2$$

sum_i is the sum of all values for attribute i occurring in the training set, and size_x is the sum of all values in the vector $\boldsymbol{x}$

One Last Distance Metrics

Kendall's Rank Correlation

$$D(\mathbf{x},\mathbf{y}) = 1 - \frac{2}{n(n-1)} \sum_{i=1}^{m} \sum_{j=1}^{i-1} \operatorname{sign}(x_i - x_j)\operatorname{sign}(y_i - y_j)$$

where

$$\operatorname{sign}(x) = \begin{cases} -1 & x < 0 \\ 0 & x = 0 \\ 1 & x > 0 \end{cases}$$

2. Alternative distance metrics (cont)

How about categorical attributes like colour or race, or complex data like text or images?

1. Create dummy boolean variables

Colour
RED
GREEN
RED
BLUE
GREEN
RED

$\longrightarrow$

Red	Green	Blue
1	0	0
0	1	0
1	0	0
0	0	1
0	1	0
1	0	0

2. Use an alternative such as Value Distance Metric or any specialized distance measures exist for DNA strings, time series, images, graphs, videos, sets, fingerprints etc

5. Efficiency Issues

k-NN is slow at classifying new cases, as each requires a linear scan of the training data. It can be made more efficient.

1. Finding neighbours efficiently.
   The cases can be stored in data structures to speed up the R^*-Tree of KD-Tree. This only helps for low dimensional attribute spaces.
2. Finding neighbours approximately
   Heuristics have been employed to find neighbours that are close, but not necessarily the closest.
3. Removing Cases
   Another approach is to remove cases that are not near the decision boundary.

Truncating k-Nearest Neighbour Search

- Bei and Gray[6] suggest a simple method to speed up searches
- Using the sum of squares distance metric

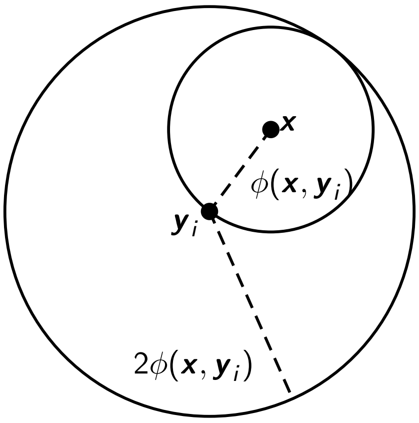
$$d_{\mathrm{SS}}(Q, C) = \sum_{i=1}^{d} [q_i - c_i]^2$$

- Keep a list of the k most similar patterns so far.
- Monitor the partial sum when computing the distance between the query point and a new training point.
- If partial sum is greater than the k^{th} best so far there is no need to compute the remaining terms.

[6]C.-D. Bei and R. Gray, *An Improvement of the Minimum Distortion Encoding Algorithm for Vector Quantization*, IEEE Transactions on Communications, vol. 33, no. 10, pp. 1132-1133, October 1985.

Accelerating Nearest Neigbour Search

- Orchard[7] provides a simple means to accelerate NN search.
- Choose n reference training points representative of the data.
- Pre-compute the distance from every training point to each of the reference points.
- Measure distance from the test point to each reference point.
- If the distance from a training point to a reference point is more than twice the distance between the reference point and the test point, it can't be a nearest neighbour.

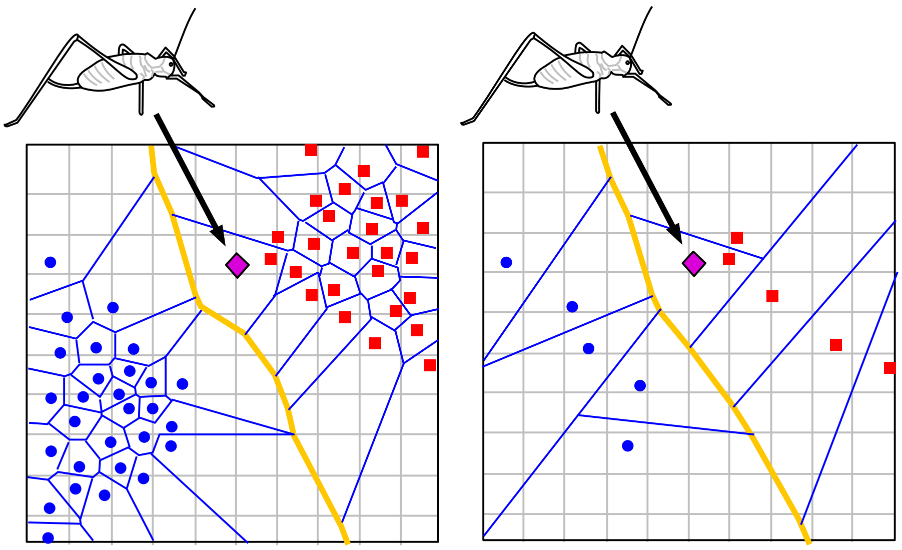


$\boldsymbol{x}$
$\phi(\boldsymbol{x}, \boldsymbol{y}_i)$
$\boldsymbol{y}_i$
$2\phi(\boldsymbol{x}, \boldsymbol{y}_i)$

[7] M.T. Orchard, *A fast nearest-neighbor search algorithm*, in Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP-91), vol.4, pp. 2297–2300, April 14-17 1991.

Removing Patterns

- Not all patterns are required to define the decision boundary.



- The higher the dimensionality of input space generally the less easily editing is done.

Condensing/Reducing/Editing Algorithms

- Condensed 1-NN: Repeatedly cycle through data, if an example is currently misclassified by 1-NN, add it to the training set. Choose first training pattern at random.

  P.E. Hart, *The Condensed Nearest Neighbor Rule*, IEEE Transactions on Information Theory, vol. IT-14, no. 3, pp. 515-516, May 1968.

- Reduced 1-NN: Repeatedly cycle through prototypes. If a prototype can be deleted without causing any training pattern to be misclassified, do so.

  G.W. Gates, *The Reduced Nearest Neighbor Classifier*, IEEE Transactions on Information Theory, vol. IT-18, no. 3, pp. 431-433, May 1972.

- Edited k-NN: If a pattern is classified correctly by k-NN using the rest of the data. Use 1-NN in operation.

  D.L. Wilson, *Asymptotic Properties of Nearest Neighbor Rules using Edited Data*, IEEE Transactions on Systems, Man and Cybernetics, vol. SMC-2, no.3, pp. 408-421, July 1972.

Summary

- Classify query pattern as belonging to the class best represented in the k most similar training examples.
- Advantages:
  - Simple to implement.
  - Low training time.
  - Non-parametric (arbitrary decision surface).
  - Defined for any distance measure.
  - Handles streaming data trivially.
  - Posterior probability estimates available.
  - Asymptotically approaches Bayes optimal decision.
- Disadvantages:
  - Very sensitive to irrelevant features and scale.
  - Slow classification time for large datasets.
  - Works best for real valued datasets.
- Next week: Linear Classifiers.