Similarity Based Learning

k-Nearest Neighbour

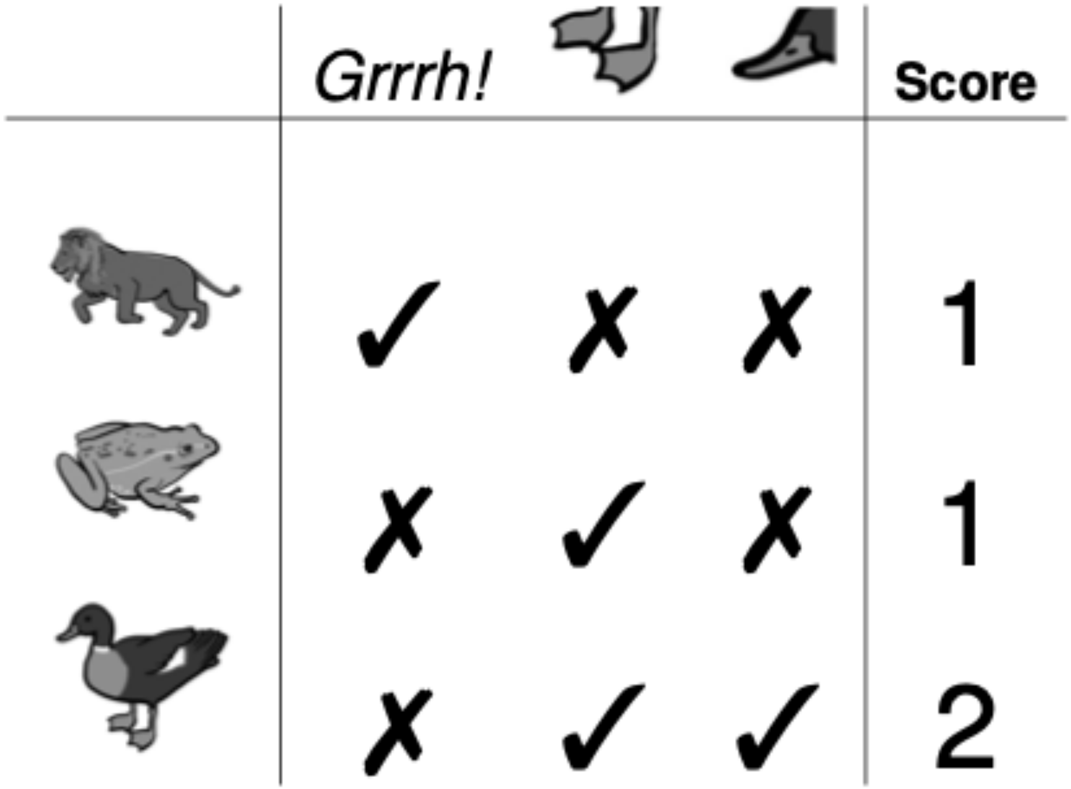
Sarah Jane Delany

Slides adapted from ML for PDA book

Big Idea

- *Looking at what has worked well in the past and make the same (or similar) predictions*

- In 1798, Lieutenant-Colonel David Collins of HMS Calcutta was exploring in NSW when one of his sailors saw a strange animal….



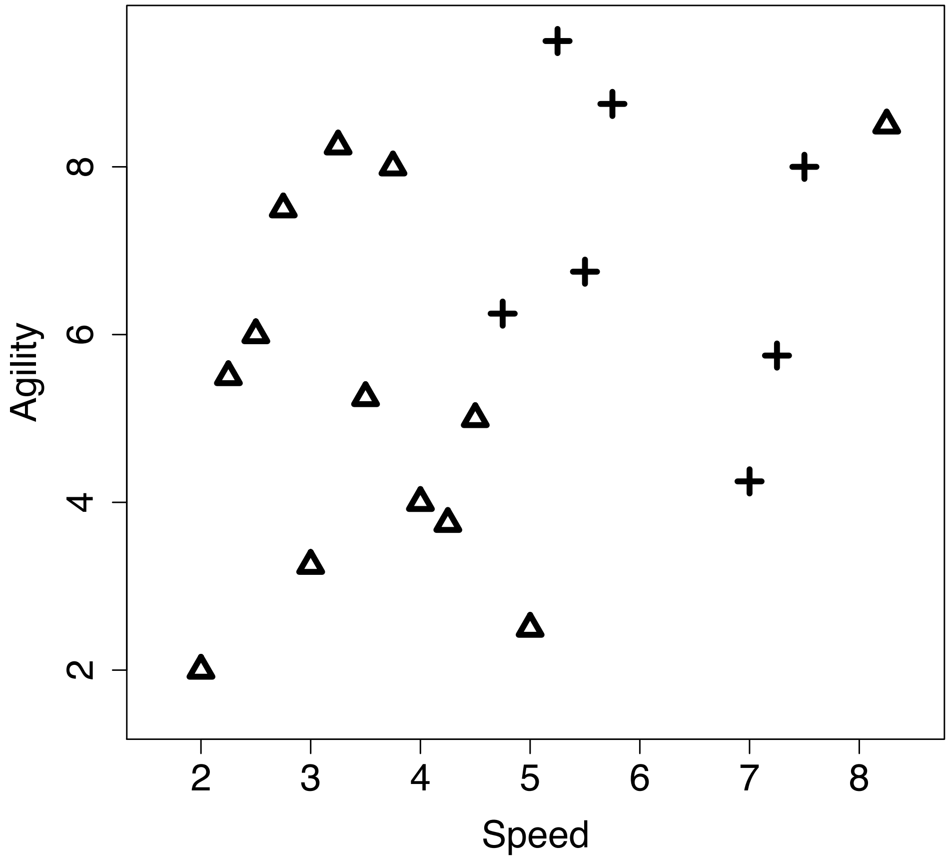
Fundamentals of Similarity Learning

- Feature space - a *D*-dimensional coordinate system used to represent the descriptive features of the instances in the training data, with one axis for each feature
- Similarity metrics - the distance between instances in the feature space is a measure of the similarity of the instances

Feature Space

- Example: 20 college athletes and whether they were drafted by a professional team.
  - Descriptive features are Speed & Agility => 2-d feature space

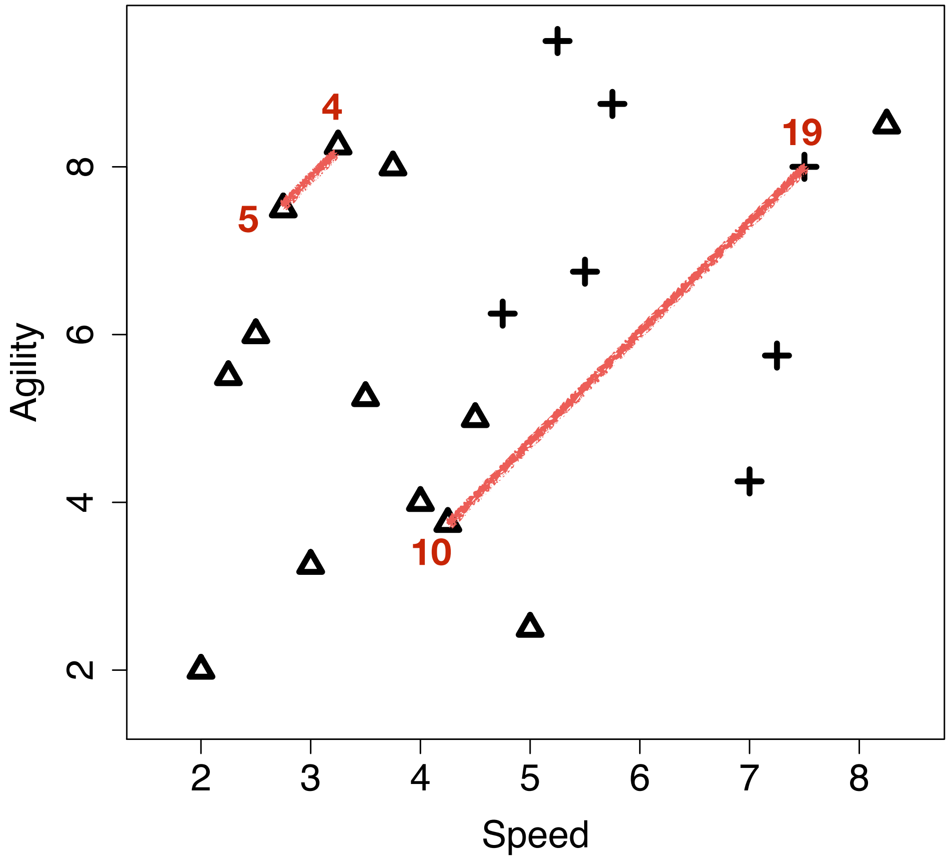
ID	Speed	Agility	Draft	ID	Speed	Agility	Draft
1	2.50	6.00	No	11	2.00	2.00	No
2	3.75	8.00	No	12	5.00	2.50	No
3	2.25	5.50	No	13	8.25	8.50	No
4	3.25	8.25	No	14	5.75	8.75	Yes
5	2.75	7.50	No	15	4.75	6.25	Yes
6	4.50	5.00	No	16	5.50	6.75	Yes
7	3.50	5.25	No	17	5.25	9.50	Yes
8	3.00	3.25	No	18	7.00	4.25	Yes
9	4.00	4.00	No	19	7.50	8.00	Yes
10	4.25	3.75	No	20	7.25	5.75	Yes



Measuring Similarity

- Similarity between instances is measured by the distance between the instances
    - Many distance metrics - no 'best' measure => problem dependent

ID	Speed	Agility	Draft	ID	Speed	Agility	Draft
1	2.50	6.00	No	11	2.00	2.00	No
2	3.75	8.00	No	12	5.00	2.50	No
3	2.25	5.50	No	13	8.25	8.50	No
4	3.25	8.25	No	14	5.75	8.75	Yes
5	2.75	7.50	No	15	4.75	6.25	Yes
6	4.50	5.00	No	16	5.50	6.75	Yes
7	3.50	5.25	No	17	5.25	9.50	Yes
8	3.00	3.25	No	18	7.00	4.25	Yes
9	4.00	4.00	No	19	7.50	8.00	Yes
10	4.25	3.75	No	20	7.25	5.75	Yes



Athletes 4 and 5 are close to each other, low distance (high similarity)

Athletes 10 and 19 are far from each other, high distance (low similiarity)

Distance Metrics

- Mathematically a metric must conform to the following four criteria:

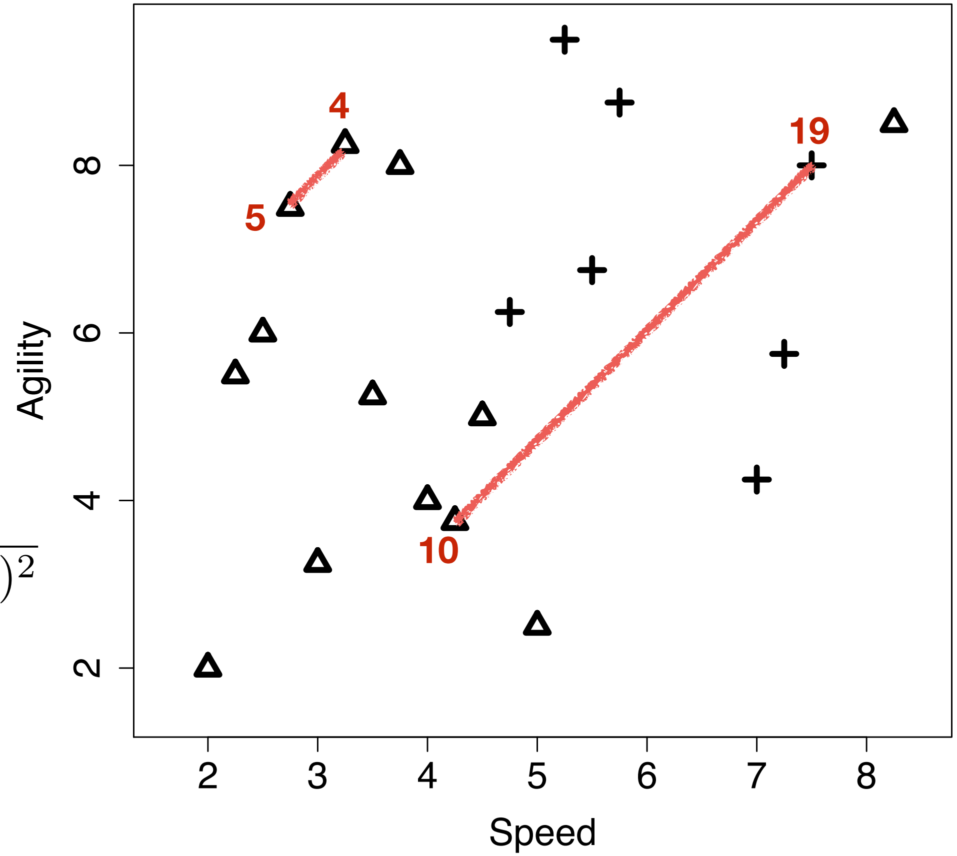
1. Non-negativity: $metric(\mathbf{a}, \mathbf{b}) \geq 0$
2. Identity: $metric(\mathbf{a}, \mathbf{b}) = 0 \iff \mathbf{a} = \mathbf{b}$
3. Symmetry: $metric(\mathbf{a}, \mathbf{b}) = metric(\mathbf{b}, \mathbf{a})$
4. Triangular Inequality:
$metric(\mathbf{a}, \mathbf{b}) \leq metric(\mathbf{a}, \mathbf{c}) + metric(\mathbf{b}, \mathbf{c})$

- Most common distance metric is Euclidean distance which computes the length of a straight line between two points, **a** and **b**:

$$Euclidean(\mathbf{a}, \mathbf{b}) = \sqrt{\sum_{i=1}^{m} (\mathbf{a}[i] - \mathbf{b}[i])^2}$$

Euclidean Distance

$$Euclidean(\mathbf{a}, \mathbf{b}) = \sqrt{\sum_{i=1}^{m} (\mathbf{a}[i] - \mathbf{b}[i])^2}$$



$$\begin{aligned} Euclidean(\mathbf{4}, \mathbf{5}) &= \sqrt{(3.25 - 2.75)^2 + (8.25 - 7.50)^2} \\ &= \sqrt{(0.25 + 0.5625)} \\ &= \sqrt{0.8125} = 0.9014 \end{aligned}$$

$$Euclidean(\mathbf{10}, \mathbf{19}) = ???$$

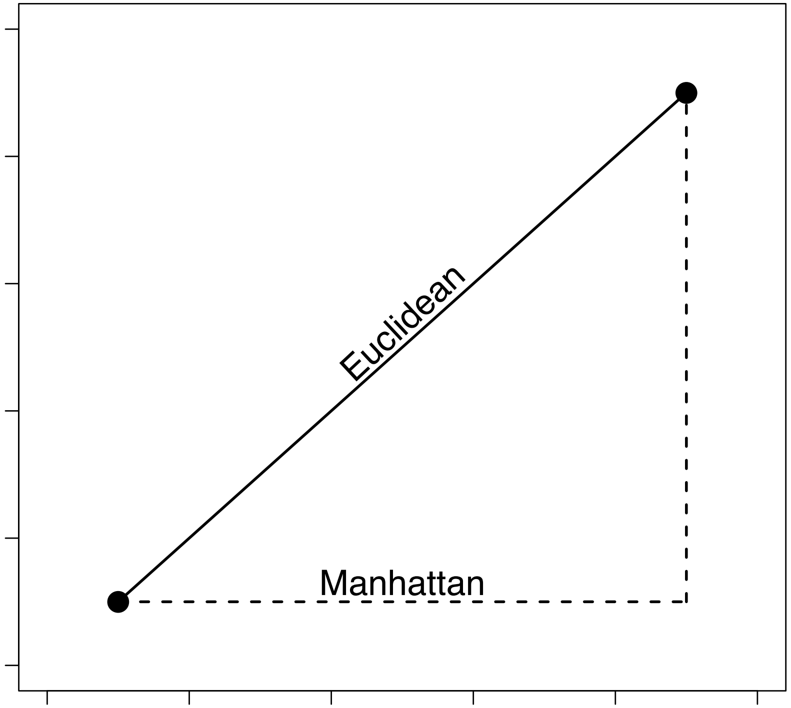
	Speed	Agility
4	3.25	8.25
5	2.75	7.50
10	4.25	3.75
19	7.50	8.00

Note: Distance and similarity have an inverse relationship

Manhattan Distance

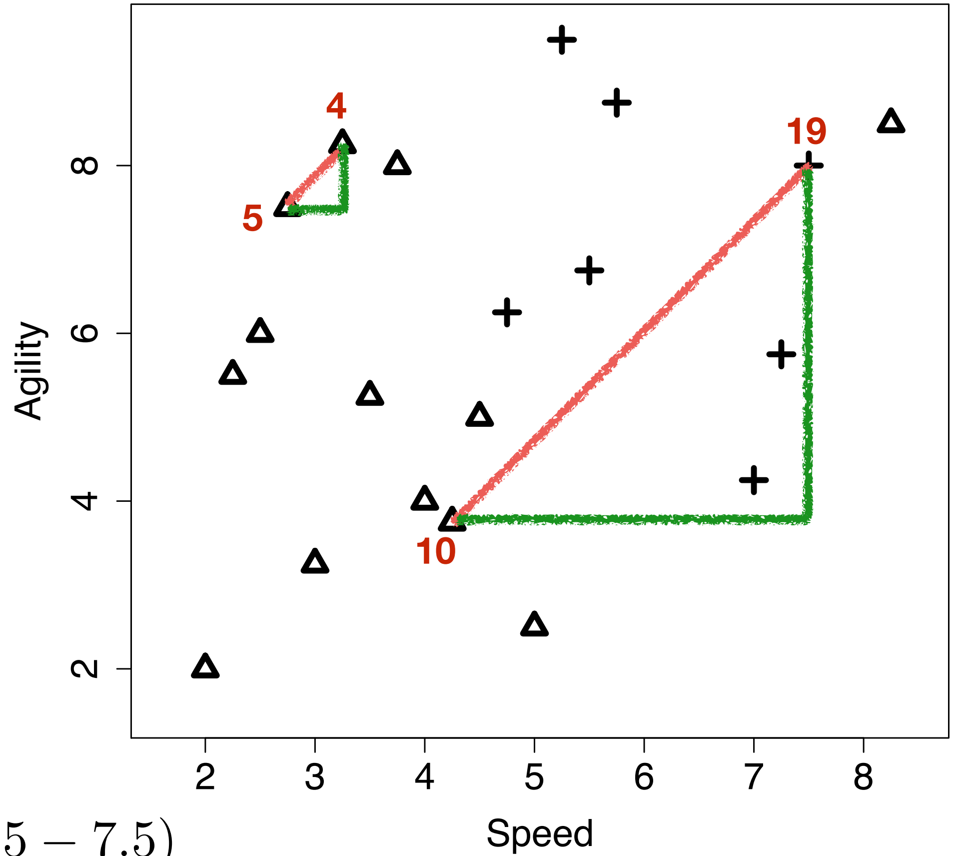
- Another, less well known distance measure is the Manhattan distance or *taxi-cab* distance

$$Manhattan(\mathbf{a}, \mathbf{b}) = \sum_{i=1}^{m} abs(\mathbf{a}[i] - \mathbf{b}[i])$$



Manhattan Distance

	Speed	Agility
4	3.25	8.25
5	2.75	7.50
10	4.25	3.75
19	7.50	8.00



$$Manhattan(\mathbf{a},\mathbf{b}) = \sum_{i=1}^{m} abs(\mathbf{a}[i] - \mathbf{b}[i])$$

$$\begin{aligned} Manhattan(\mathbf{4},\mathbf{5}) &= abs(3.25 - 2.75) + abs(8.25 - 7.5) \\ &= 0.5 + 0.75 = 1.25 \end{aligned}$$

$$Manhattan(\mathbf{10},\mathbf{19}) = ??$$

Note: Manhattan has a slight computational advantage over Euclidean

Minkowski Distance

- Euclidean and Manhattan distances are special cases of Minkowski distance:

$$Minkowski(\mathbf{a}, \mathbf{b}) = \left(\sum_{i=1}^{m} abs(\mathbf{a}[i] - \mathbf{b}[i])^p \right)^{\frac{1}{p}}$$

- Different values of parameter *p* result in different distance measures
  - Manhattan distance for $p = 1$
  - Euclidean distance for $p = 2$
- The larger the value of *p* the more emphasis is placed on features with large differences in values because these differences are raised to the power of *p*

What about non numeric data?

- Binary: Takes only two values - a boolean True/False decision
  e.g. married={True,False}, test_result={Pass,Fail}
- Categorical (Nominal): A feature that takes values from a finite set of values, with no intrinsic ordering to the values
  e.g. blood_group={A,B,AB,O}, nationality={French,Irish,Italian}
- Ordinal: Similar to a categorical variable, but there is a clear ordering of the variables.
  e.g. grade={A,B,C,D,E,F}, dosage={Low,Medium,High}
- Interval: Values that allow ordering and subtraction, but do not allow other arithmetic operations
  e.g date, time

Categorical Data

- Overlap Difference (feature level): Simplest distance measure. Returns 0 if the two values for a feature are equal and 1 otherwise

Athlete	Gender	Nationality
x1	Female	Irish
x2	Male	Irish
x3	Male	Italian

For feature *Gender*

$d_g(x1,x2) = 1$
$d_g(x1,x3) = 1$
$d_g(x2,x3) = 0$

For feature *Nationality*

$d_n(x1,x2) = 0$
$d_n(x1,x3) = 1$
$d_n(x2,x3) = 1$

- Hamming distance: Distance metric for instance represented with categorical data only, = the sum of the overlap differences across all features - i.e. number of features on which two examples disagree.

$d(x1,x2) = 1 + 0 = 1$
$d(x1,x3) = 1 + 1 = 2$
$d(x2,x3) = 0 + 1 = 1$

Overlap distance for *Gender* +
Overlap distance for *Nationality*

Ordinal Data

- For *ordinal features*, calculate the absolute value of the difference between the two positions in the ordered list of possible values.

e.g. Ordinal Feature *Dosage*:
{Low,Medium,High} = {1, 2, 3}

```
diff(Low,High) = |1-3| = 2
diff(Medium,Low) = |2-1| = 1
diff(High,High) = |3-3| = 0
```

Heterogeneous Distance Measures

- In many datasets, the features associated with instances will have different types (e.g. continuous, categorical, ordinal etc).

Athlete	*Speed*	*Agility*	*Gender*	*Nationality*
x1	2.50	6.00	Female	Irish
x2	3.75	8.00	Male	Irish
x3	2.25	5.50	Male	Italian

- Local distance function: Measure the distance between two instances based on a single feature.
  - e.g. distance between **x1** and **x2** in terms of *Speed*?
  - e.g. distance between **x1** and **x3** in terms of *Gender*?
  - e.g. distance between **x1** and **x2** in terms of Nationality?

- Global distance function: Measure the distance between two instances based on the combination of the local distances across all features.

Heterogeneous Distance Functions

- We can create a global measure from different local distance functions, using an appropriate function for each feature.

Athlete	*Speed*	*Agility*	*Gender*	*Nationality*
x1	2.50	6.00	Female	Irish
x2	3.75	8.00	Male	Irish
x3	2.25	5.50	Male	Italian

Use absolute difference for continuous features *Speed & Agility*

Use overlap for categorical features *Gender & Nationality*

```
d(x1,x2) = 1.25 + 2.0 + 1 + 0 = 4.25
d(x1,x3) = 0.25 + 0.5 + 1 + 1 = 2.75
d(x2,x3) = 1.5 + 2.5 + 0 + 1 = 5.0
```

Global distance calculated as sum over individual local distances

- Often domain expertise is required to choose an appropriate distance measure for a particular dataset.

Nearest Neighbour Algorithm (k-NN)

- Have a set of training instances and a query to be classified
- Steps:
  - Iterate across the training instances in memory and find the instance(s) that is/are the most similar (shortest distance) to the query instance in the feature space
  - Make a prediction for the query instance based on the target values of the nearest neighbour(s) of the query instance

- 1-NN - use the most similar/closest training instance
- *k*-NN - use the k most similar/closest training instances

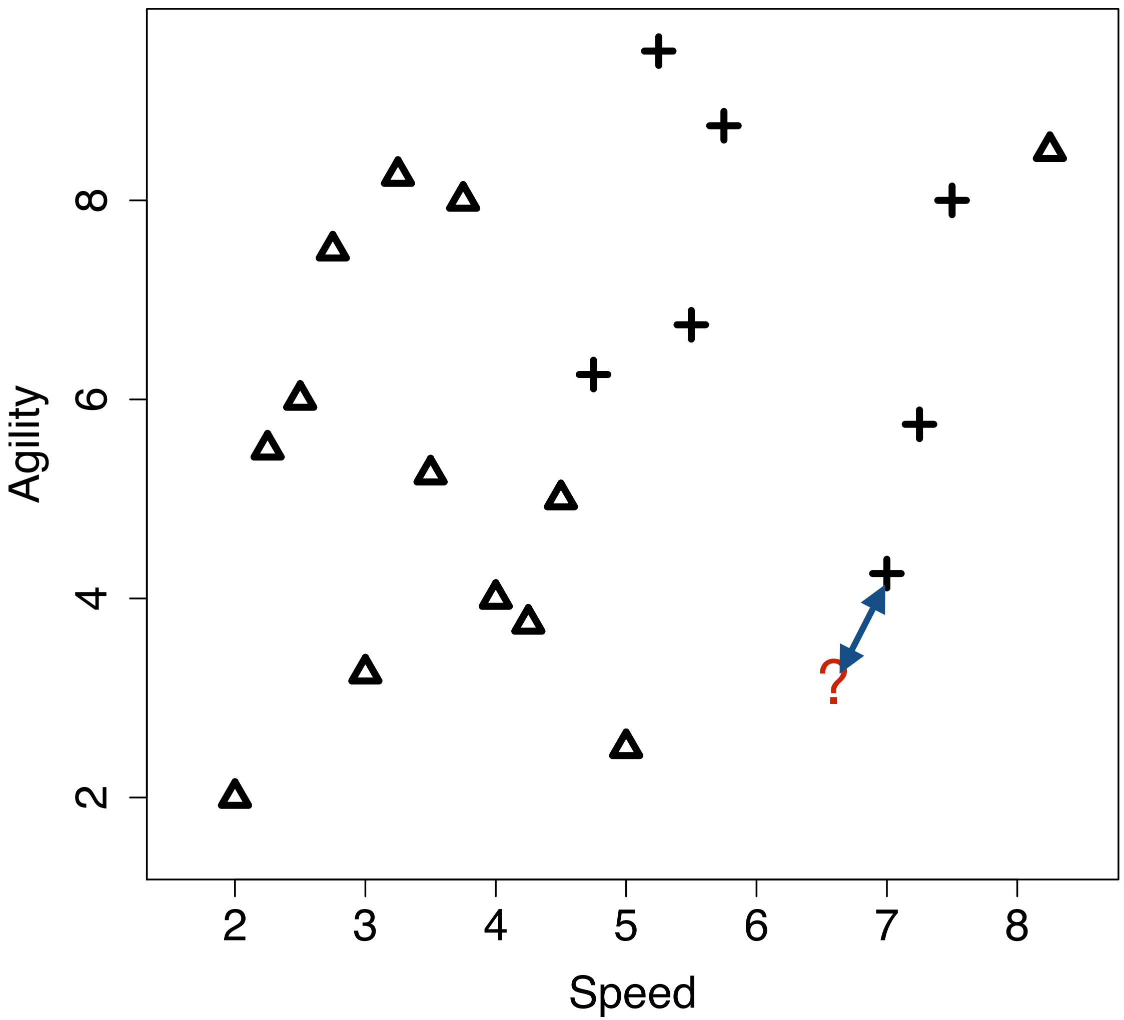
Example

ID	Speed	Agility	Draft	ID	Speed	Agility	Draft
1	2.50	6.00	No	11	2.00	2.00	No
2	3.75	8.00	No	12	5.00	2.50	No
3	2.25	5.50	No	13	8.25	8.50	No
4	3.25	8.25	No	14	5.75	8.75	Yes
5	2.75	7.50	No	15	4.75	6.25	Yes
6	4.50	5.00	No	16	5.50	6.75	Yes
7	3.50	5.25	No	17	5.25	9.50	Yes
8	3.00	3.25	No	18	7.00	4.25	Yes
9	4.00	4.00	No	19	7.50	8.00	Yes
10	4.25	3.75	No	20	7.25	5.75	Yes

- Should we draft an athlete who with the following profile?

Speed = 6.75; Agility = 3

Example



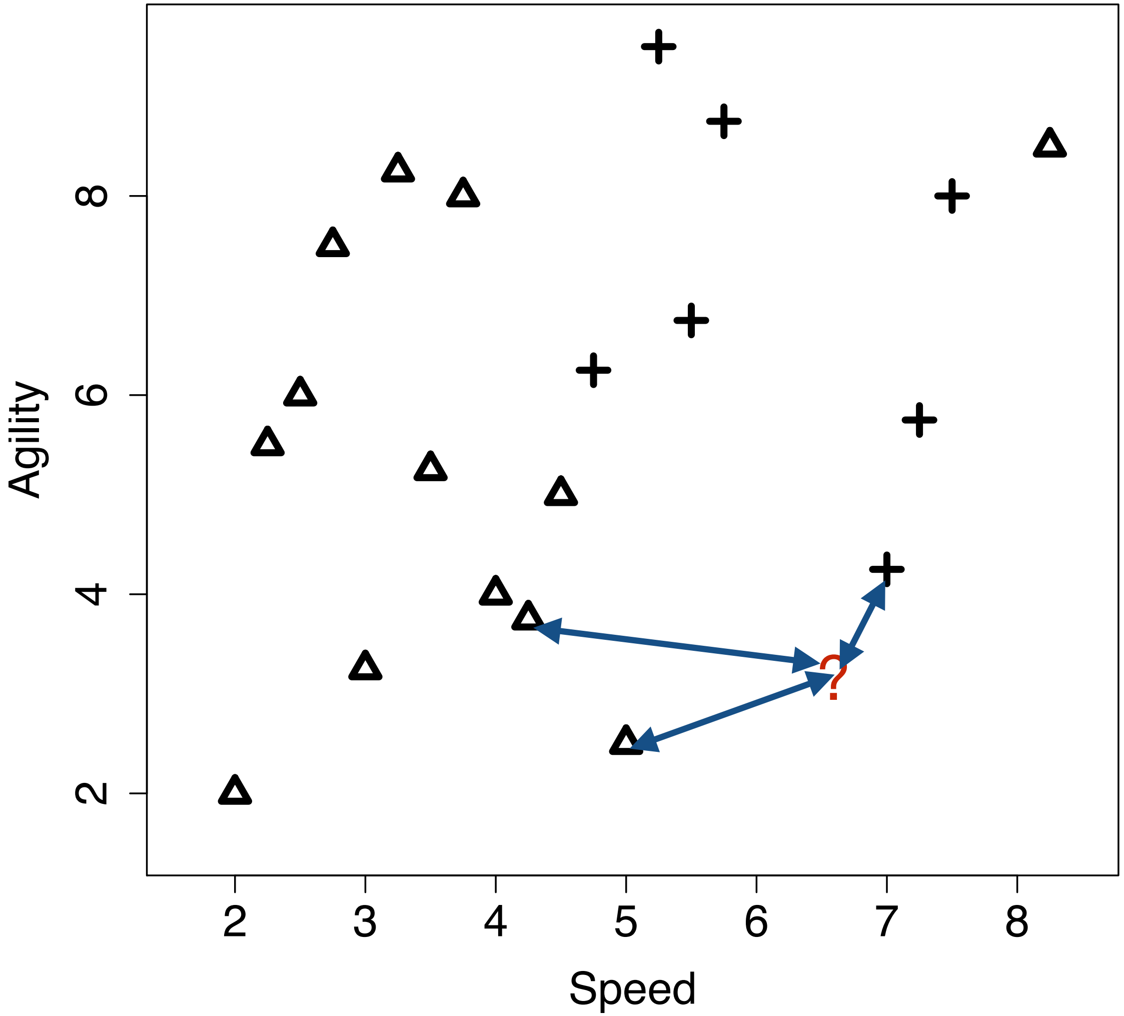
Triangle: No
Plus: Yes

Example: 1-NN

- Compute distance (using selected distance measure) of query to each training instance
- Rank all instances based on calculated distance
- See which is the *closest* to query
  Remember: lowest distance = highest similarity

ID	SPEED	AGILITY	DRAFT	Dist.	ID	SPEED	AGILITY	DRAFT	Dist.
18	7.00	4.25	yes	1.27	11	2.00	2.00	no	4.85
12	5.00	2.50	no	1.82	19	7.50	8.00	yes	5.06
10	4.25	3.75	no	2.61	3	2.25	5.50	no	5.15
20	7.25	5.75	yes	2.80	1	2.50	6.00	no	5.20
9	4.00	4.00	no	2.93	13	8.25	8.50	no	5.70
6	4.50	5.00	no	3.01	2	3.75	8.00	no	5.83
8	3.00	3.25	no	3.76	14	5.75	8.75	yes	5.84
15	4.75	6.25	yes	3.82	5	2.75	7.50	no	6.02
7	3.50	5.25	no	3.95	4	3.25	8.25	no	6.31
16	5.50	6.75	yes	3.95	17	5.25	9.50	yes	6.67

Example: 3-NN



Triangle: No
Plus: Yes

Example: 3-NN

- Compute distance of query to each training instance
- Rank all instances based on distance
- Choose the *k* nearest neighbours
- Determine predicted target by majority vote of target of the *k* nearest neighbours

ID	SPEED	AGILITY	DRAFT	Dist.	ID	SPEED	AGILITY	DRAFT	Dist.
18	7.00	4.25	yes	1.27	11	2.00	2.00	no	4.85
12	5.00	2.50	no	1.82	19	7.50	8.00	yes	5.06
10	4.25	3.75	no	2.61	3	2.25	5.50	no	5.15
20	7.25	5.75	yes	2.80	1	2.50	6.00	no	5.20
9	4.00	4.00	no	2.93	13	8.25	8.50	no	5.70
6	4.50	5.00	no	3.01	2	3.75	8.00	no	5.83
8	3.00	3.25	no	3.76	14	5.75	8.75	yes	5.84
15	4.75	6.25	yes	3.82	5	2.75	7.50	no	6.02
7	3.50	5.25	no	3.95	4	3.25	8.25	no	6.31
16	5.50	6.75	yes	3.95	17	5.25	9.50	yes	6.67

What about 4-NN?

Example: 4-NN

ID	SPEED	AGILITY	DRAFT	Dist.	ID	SPEED	AGILITY	DRAFT	Dist.
18	7.00	4.25	yes	1.27	11	2.00	2.00	no	4.85
12	5.00	2.50	no	1.82	19	7.50	8.00	yes	5.06
10	4.25	3.75	no	2.61	3	2.25	5.50	no	5.15
20	7.25	5.75	yes	2.80	1	2.50	6.00	no	5.20
9	4.00	4.00	no	2.93	13	8.25	8.50	no	5.70
6	4.50	5.00	no	3.01	2	3.75	8.00	no	5.83
8	3.00	3.25	no	3.76	14	5.75	8.75	yes	5.84
15	4.75	6.25	yes	3.82	5	2.75	7.50	no	6.02
7	3.50	5.25	no	3.95	4	3.25	8.25	no	6.31
16	5.50	6.75	yes	3.95	17	5.25	9.50	yes	6.67

- Can break ties
  - Randomly
  - Based on the sum of the nearest neighbour distances for each target class

k-NN algorithm

- *k* nearest neighbours algorithm predicts the target class with the majority vote from the set of *k* nearest neighbours to the query q:

$$\mathbb{M}_k(\mathbf{q}) = \underset{c \in classes(t)}{\operatorname{argmax}} \sum_{i=1}^{k} \delta_{t_i, c}$$

$classes(t)$ is the set of target classes

t_i is the target class for instance i

$\delta_{i,j}$ is Kronecker's delta $\quad \delta_{i,j} = \begin{cases} 1 & \text{when } i = j \\ 0 & \text{when } i \neq j. \end{cases}$

Weighted *k*-NN algorithm

- In distance weighted *k* Nearest Neighbour algorithm some neighbours get higher weight than others

$$\mathbb{M}_k(\mathbf{q}) = \underset{c \in classes(t)}{\arg\max} \sum_{i=1}^{k} w_i \times \delta_{t_i, c}$$

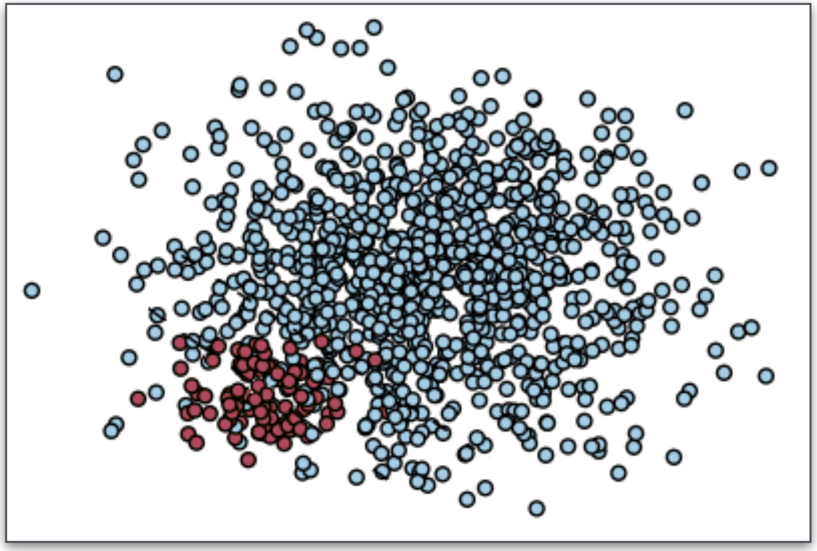
- Instead of a binary vote of 1 for the class of the neighbour, closest neighbours (those that are most similar to the query) get a higher weighting when deciding the prediction for the query
- Remember: similarity = inverse of distance

$$w_i = \frac{1}{dist(\mathbf{q}, \mathbf{d_i})}$$

Tuning for *k*

- A simple 1-NN classifier is easy to implement. But it will be susceptible to "noise" in the data. A misclassification will occur every time a single noisy example is retrieved.
- We might decide to vary the neighbourhood size parameter *k* to improve the predictive performance of *k*-NN.
- Choosing between different settings of an algorithm is often referred to as *hyperparameter tuning* or *model selection*.

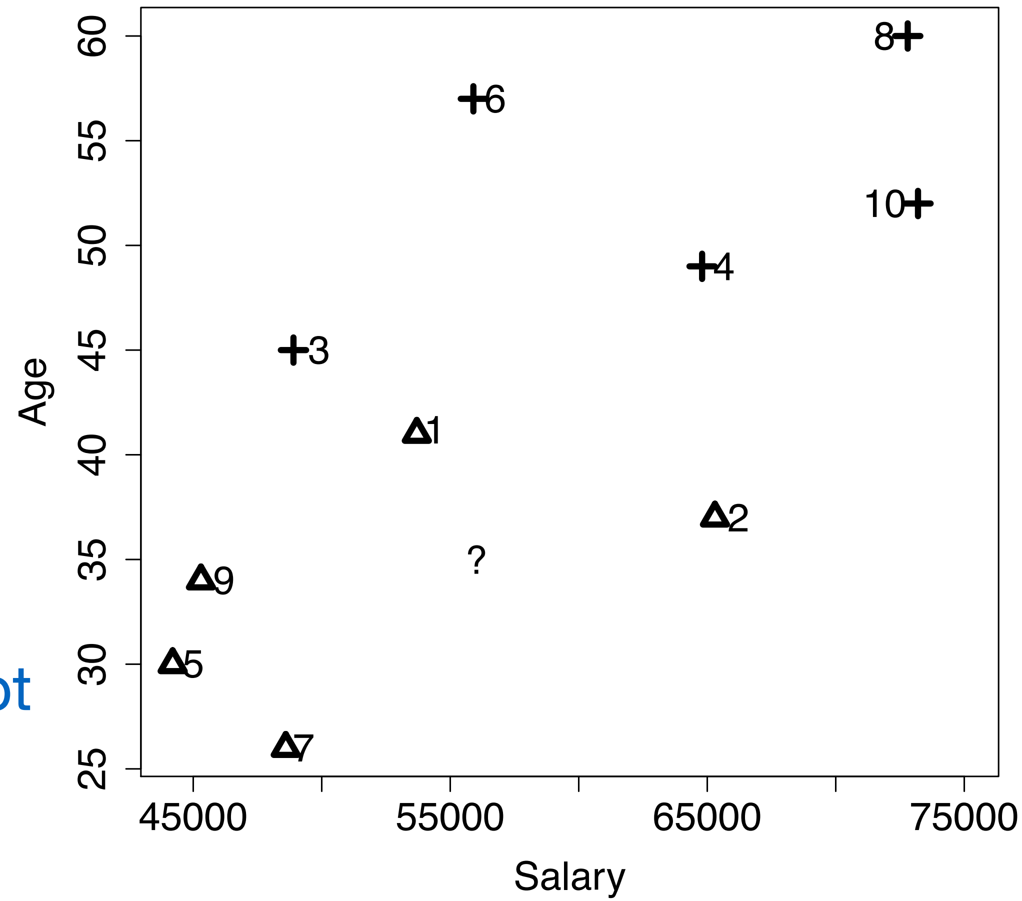
- Using a larger *k* (e.g. $k > 2$) can sometimes make the classifier more robust and overcome this problem.
- But when *k* is large ($k \to N$) and classes are *unbalanced*, we always predict the majority class.



Data Normalisation

- Consider the dataset listing salary and age information for customers and whether or not they purchased a pension plan.

ID	Salary	Age	Purchased
1	53700	41	No
2	65300	37	No
3	48900	45	Yes
4	64800	49	Yes
5	44200	30	No
6	55900	57	Yes
7	48600	26	No
8	72800	60	Yes
9	45300	34	No
10	73200	52	Yes



What should the marketing dept expect for customer aged 35, with salary of 56,000?

Data Normalisation

- Calculate distance using Euclidean distance:

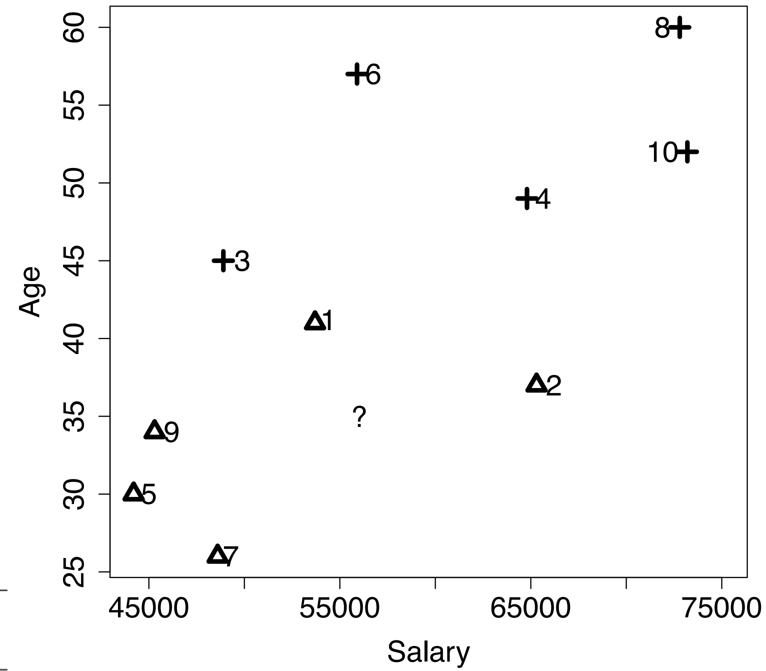
				Salary and Age		Salary Only		Age Only	
ID	Salary	Age	Purch.	Dist.	Rank.	Dist.	Rank.	Dist.	Rank.
1	53700	41	No	2300.0078	2	2300	2	6	4
2	65300	37	No	9300.0002	6	9300	6	2	2
3	48900	45	Yes	7100.0070	3	7100	3	10	6
4	64800	49	Yes	8800.0111	5	8800	5	14	7
5	44200	30	No	11800.0011	8	11800	8	5	3
6	55900	57	Yes	102.3914	1	100	1	22	9
7	48600	26	No	7400.0055	4	7400	4	9	5
8	72800	60	Yes	16800.0186	9	16800	9	25	10
9	45300	34	No	10700.0000	7	10700	7	1	1
10	73200	52	Yes	17200.0084	10	17200	10	17	8

- Salary feature dominates the computation of distance, age feature is virtually ignored
- Due to the larger range in the Salary feature

Data Normalisation

- Use min-max range normalisation to rescale values to the range of [0,1]

$$a_i' = \frac{a_i - min(a)}{max(a) - min(a)} \times (high - low) + low$$
$$= \frac{a_i - min(a)}{max(a) - min(a)}$$



	Normalized Dataset			Salary and Age		Salary Only		Age Only	
ID	Salary	Age	Purch.	Dist.	Rank	Dist.	Rank	Dist.	Rank
1	0.3276	0.4412	No	0.1935	1	0.0793	2	0.17647	4
2	0.7276	0.3235	No	0.3260	2	0.3207	6	0.05882	2
3	0.1621	0.5588	Yes	0.3827	5	0.2448	3	0.29412	6
4	0.7103	0.6765	Yes	0.5115	7	0.3034	5	0.41176	7
5	0.0000	0.1176	No	0.4327	6	0.4069	8	0.14706	3
6	0.4034	0.9118	Yes	0.6471	8	0.0034	1	0.64706	9
7	0.1517	0.0000	No	0.3677	3	0.2552	4	0.26471	5
8	0.9862	1.0000	Yes	0.9361	10	0.5793	9	0.73529	10
9	0.0379	0.2353	No	0.3701	4	0.3690	7	0.02941	1
10	1.0000	0.7647	Yes	0.7757	9	0.5931	10	0.50000	8

Predicting Continuous Targets

- *k* nearest neighbours algorithm predicts the average target value of *k* nearest neighbours to the query q:

$$\mathbb{M}_k(\mathbf{q}) = \frac{1}{k} \sum_{i=1}^{k} t_i$$

t_i is the target feature value for instance i

k-NN Algorithm

- Similarity based learning attempts to mimic a very human way of reasoning - this makes them more explainable - easy to interpret and understand
  - Can give people more confidence in the model
- Lazy learning technique: Model is built at classification time rather than training time
  - Uses a set of local training instances to classify each query
  - Appropriate for heterogeneous data
  - Computationally more expensive as the number of training instances becomes larger
- Easy to add new instances to training data for re-training to handle concept drift
- Important to normalise data (for all prediction algorithms)