Similarity-based Learning

Chapter 5, Sections 5.1, 5.2, 5.3

Péter Molnár[1]

J. Mack Robinson College of Business
Georgia State University

MSA8150 – Spring 2016

[1] Original slides from John Kelleher, Brian Mac Namee, and Aoife D'Arcy

Big Idea

The story goes like this ...

- The year is 1798 and you are Lieutenant-Colonel David Collins of HMS Calcutta who is exploring the region around Hawkesbury River, in New South Wales.
- After an expedition up the river one of the men tells you that *he saw a strange animal near the river*.
- You ask him to *describe the animal* to you and he explains that he didn't see it very well but that he did notice that it had *webbed feet* and a *duck-bill snout*, and that it *growled* at him.
- In order to plan the expedition for the next day you decide that you need to classify the animal so that you can figure out whether it is *dangerous* to approach it or not.

	Grrrh!			**Score**
	✓	✗	✗	1
	✗	✓	✗	1
	✗	✓	✓	2

Figure: Matching animals you remember to the features of the unknown animal described by the sailor. Image source: Jan Gillbank (`http://www.e4ac.edu.au`)

- The process of classifying an unknown animal by matching the features of the animal against the features of animals you can remember neatly encapsulates the big idea underpinning similarity-based learning:

if you are trying to classify something then you should search your memory to find things that are similar and label it with the same class as the most similar thing in your memory

- One of the simplest and best known machine learning algorithms for this type of reasoning is called the nearest neighbor algorithm.

Fundamentals

Table: The speed and agility ratings for 20 college athletes labelled with the decisions for whether they were drafted or not.

ID	Speed	Agility	Draft
1	2.50	6.00	No
2	3.75	8.00	No
3	2.25	5.50	No
4	3.25	8.25	No
5	2.75	7.50	No
6	4.50	5.00	No
7	3.50	5.25	No
8	3.00	3.25	No
9	4.00	4.00	No
10	4.25	3.75	No

ID	Speed	Agility	Draft
11	2.00	2.00	No
12	5.00	2.50	No
13	8.25	8.50	No
14	5.75	8.75	Yes
15	4.75	6.25	Yes
16	5.50	6.75	Yes
17	5.25	9.50	Yes
18	7.00	4.25	Yes
19	7.50	8.00	Yes
20	7.25	5.75	Yes

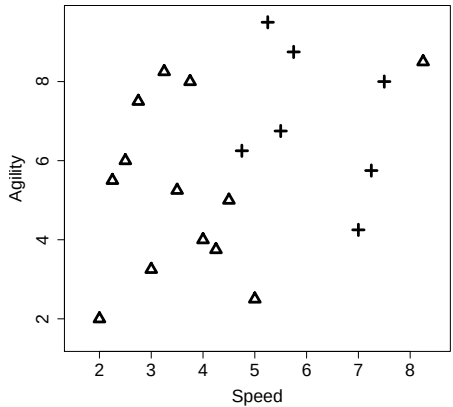


Figure: A feature space plot of Table 2 [18]. Triangles represent *'Non-draft'* and crosses *'Draft'* instances.

- A similarity metric measures the similarity between two instances according to a feature space
- Mathematically, a metric must conform to the following four criteria:
  1. Non-negativity: $\mathit{metric}(\mathbf{a}, \mathbf{b}) \geq 0$
  2. Identity: $\mathit{metric}(\mathbf{a}, \mathbf{b}) = 0 \iff \mathbf{a} = \mathbf{b}$
  3. Symmetry: $\mathit{metric}(\mathbf{a}, \mathbf{b}) = \mathit{metric}(\mathbf{b}, \mathbf{a})$
  4. Triangular Inequality: $\mathit{metric}(\mathbf{a}, \mathbf{b}) \leq \mathit{metric}(\mathbf{a}, \mathbf{c}) + \mathit{metric}(\mathbf{b}, \mathbf{c})$

  where $\mathit{metric}(\mathbf{a}, \mathbf{b})$ is a function that returns the distance between two instances $\mathbf{a}$ and $\mathbf{b}$.

- One of the best known metrics is Euclidean distance which computes the length of the straight line between two points. Euclidean distance between two instances **a** and **b** in a *m*-dimensional feature space is defined as:

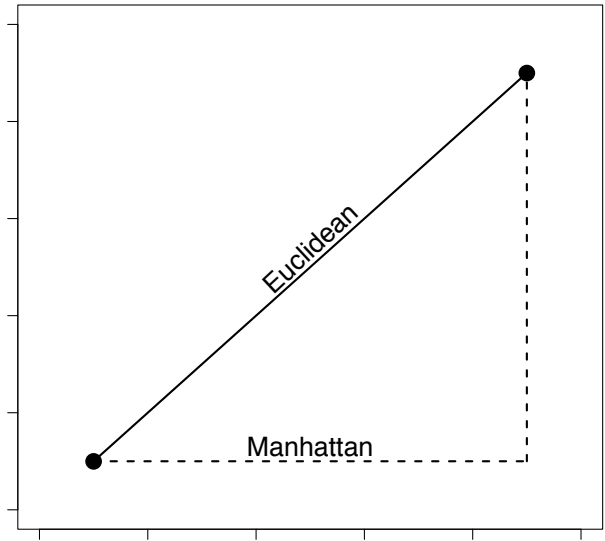
$$Euclidean(\mathbf{a}, \mathbf{b}) = \sqrt{\sum_{i=1}^{m} (\mathbf{a}[i] - \mathbf{b}[i])^2} \tag{1}$$

Example: the Euclidean distance between instances d_{12} (SPEED= 5.00, AGILITY= 2.5) and d_5 (SPEED= 2.75,AGILITY= 7.5) in Table 2 [18] is:

- Another, less well known, distance measure is the Manhattan distance or taxi-cab distance.
- The Manhattan distance between two instances **a** and **b** in a feature space with m dimensions is:

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

Example: The Manhattan distance between instances d_{12} (SPEED= 5.00, AGILITY= 2.5) and d_5 (SPEED= 2.75,AGILITY= 7.5) in Table 2 [18] is:

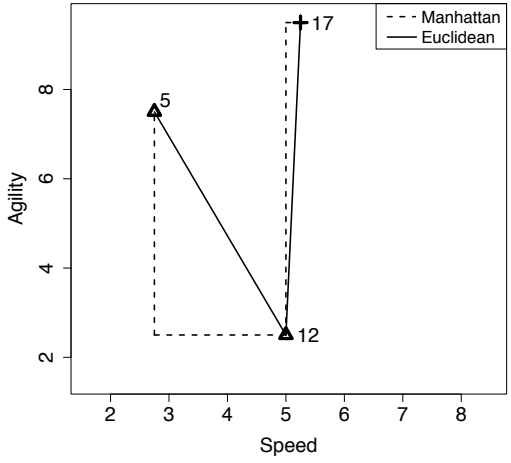

Euclidean
Manhattan

- Euclidean and Manhattan distances are special cases of Minkowski distance
- The Minkowski distance between two instances **a** and **b** in a feature space with m descriptive features is:

$$Minkowski(\mathbf{a}, \mathbf{b}) = \left(\sum_{i=1}^{m} abs(\mathbf{a}[i] - \mathbf{b}[i])^p \right)^{\frac{1}{p}} \quad (3)$$

- Minkowski distance with $p = 1$ is Manhattan, $p = 2$ is the Euclidean.
- The larger the value of p the more emphasis is placed on the features with large differences in values.

Instance ID	Instance ID	Manhattan (Minkowski p=1)	Euclidean (Minkowski p=2)
12	5	7.25	5.4829
12	17	7.25	8.25



Standard Approach: The Nearest Neighbor Algorithm

The Nearest Neighbour Algorithm

Require: set of training instances
Require: a query to be classified

1: Iterate across the instances in memory and find the instance that is shortest distance from the query position in the feature space.
2: Make a prediction for the query equal to the value of the target feature of the nearest neighbor.

Table: The speed and agility ratings for 20 college athletes labelled with the decisions for whether they were drafted or not.

ID	Speed	Agility	Draft
1	2.50	6.00	No
2	3.75	8.00	No
3	2.25	5.50	No
4	3.25	8.25	No
5	2.75	7.50	No
6	4.50	5.00	No
7	3.50	5.25	No
8	3.00	3.25	No
9	4.00	4.00	No
10	4.25	3.75	No

ID	Speed	Agility	Draft
11	2.00	2.00	No
12	5.00	2.50	No
13	8.25	8.50	No
14	5.75	8.75	Yes
15	4.75	6.25	Yes
16	5.50	6.75	Yes
17	5.25	9.50	Yes
18	7.00	4.25	Yes
19	7.50	8.00	Yes
20	7.25	5.75	Yes

Example

- Should we draft an athlete with the following profile:

$$\text{SPEED} = 6.75,\ \text{AGILITY} = 3$$

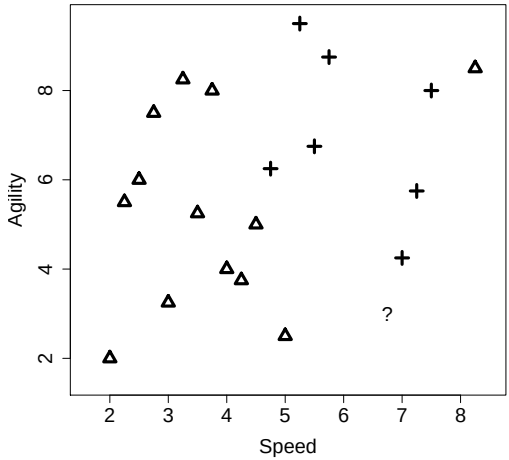
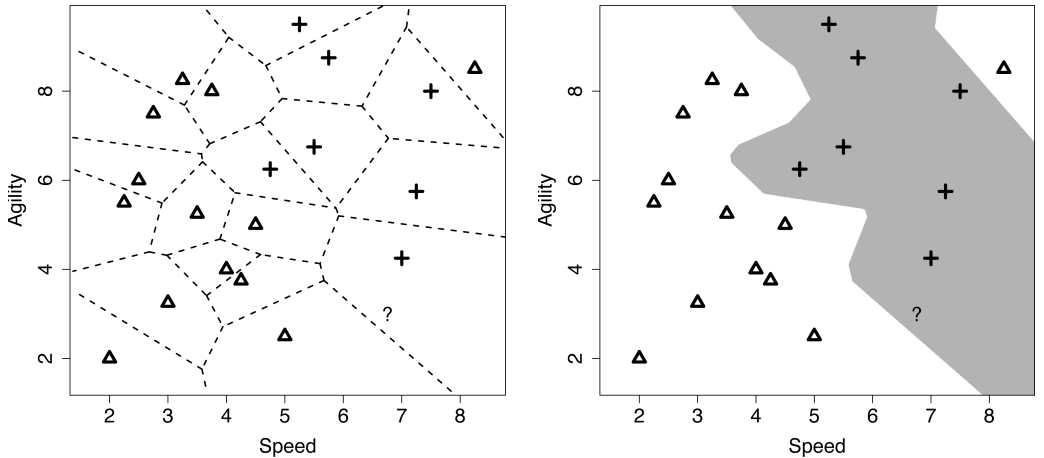


Figure: A feature space plot of the data in Table 2 [18] with the position in the feature space of the query represented by the ? marker. The triangles represent *'Non-draft'* instances and the crosses represent the *'Draft'* instances.

Table: The distances (Dist.) between the query instance with SPEED = 6.75 and AGILITY = 3.00 and each instance in Table 2 [18].

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



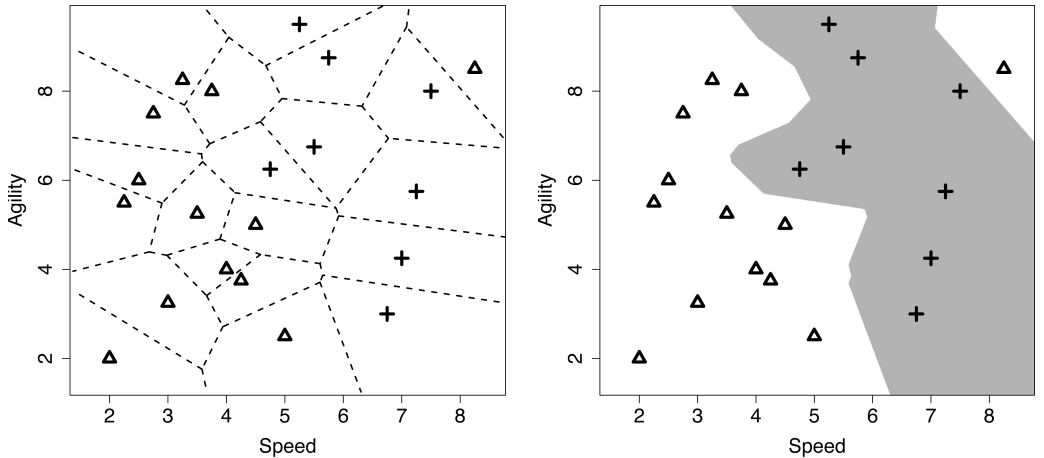
(a) Voronoi tessellation (b) Decision boundary ($k = 1$)

Figure: (a) The Voronoi tessellation of the feature space for the dataset in Table 2 [18] with the position of the query represented by the ? marker; (b) the decision boundary created by aggregating the neighboring Voronoi regions that belong to the same target level.

- One of the great things about nearest neighbour algorithms is that we can add in new data to update the model very easily.

Table: The extended version of the college athletes dataset.

ID	SPEED	AGILITY	DRAFT
1	2.50	6.00	no
2	3.75	8.00	no
3	2.25	5.50	no
4	3.25	8.25	no
5	2.75	7.50	no
6	4.50	5.00	no
7	3.50	5.25	no
8	3.00	3.25	no
9	4.00	4.00	no
10	4.25	3.75	no
11	2.00	2.00	no
12	5.00	2.50	no
13	8.25	8.50	no
14	5.75	8.75	yes
15	4.75	6.25	yes
16	5.50	6.75	yes
17	5.25	9.50	yes
18	7.00	4.25	yes
19	7.50	8.00	yes
20	7.25	5.75	yes
21	6.75	3.00	yes



(a) Voronoi tessellation (b) Decision boundary ($k = 1$)

Figure: (a) The Voronoi tessellation of the feature space when the dataset has been updated to include the query instance; (b) the updated decision boundary reflecting the addition of the query instance in the training set.

Handling Noisy Data

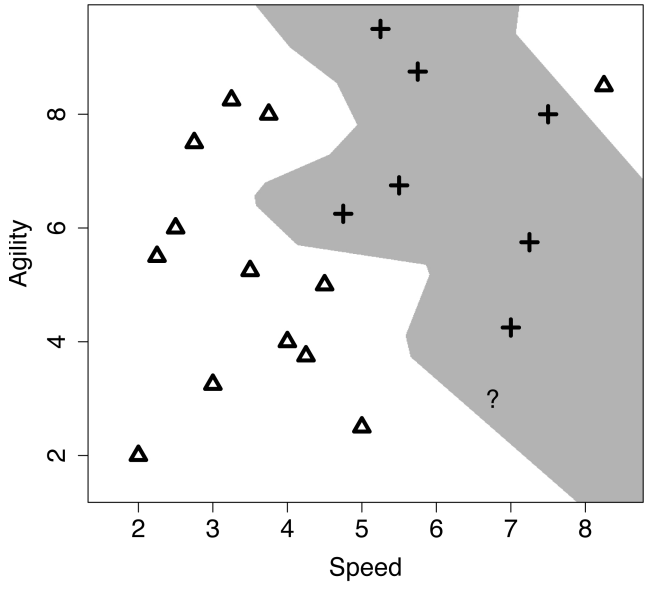


Figure: Is the instance at the top right of the diagram really *noise*?

- The k nearest neighbors model predicts the target level with the majority vote from the set of k nearest neightbors to the query $\mathbf{q}$:

$$\mathbb{M}_k(\mathbf{q}) = \underset{l \in \mathit{levels}(t)}{\operatorname{argmax}} \sum_{i=1}^{k} \delta(t_i, l) \tag{4}$$

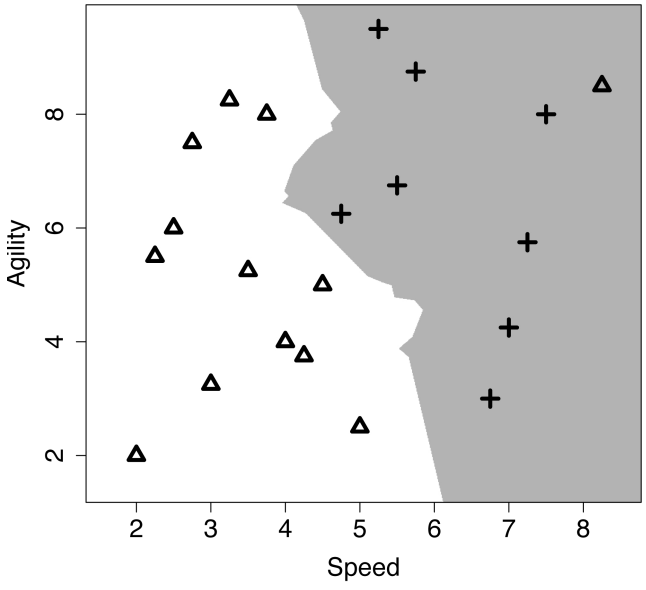


Figure: The decision boundary using majority classification of the nearest 3 neighbors.

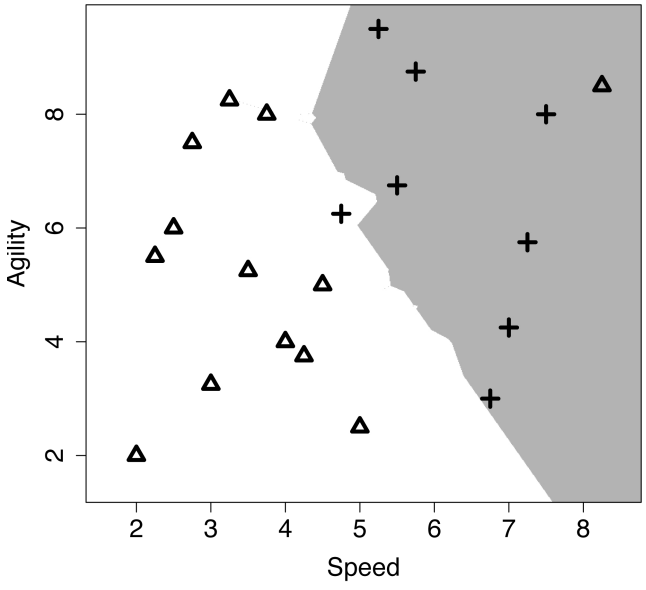


Figure: The decision boundary using majority classification of the nearest 5 neighbors.

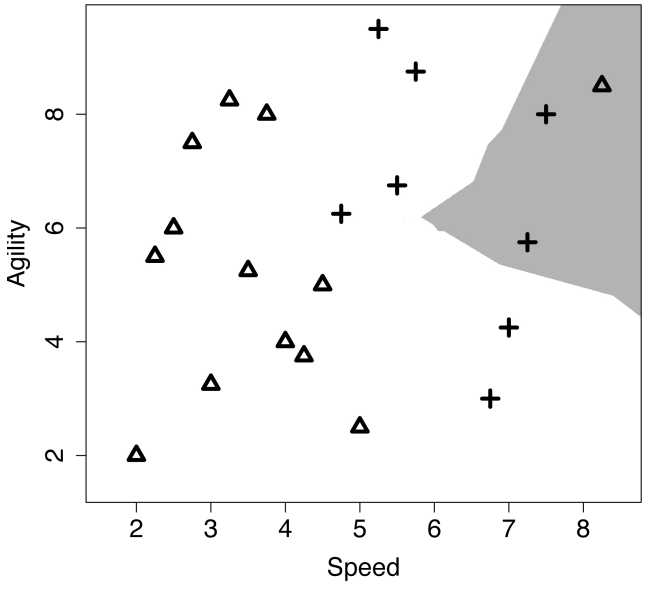


Figure: The decision boundary when k is set to 15.

- In a distance weighted k nearest neighbor algorithm the contribution of each neighbor to the classification decision is weighted by the reciprocal of the squared distance between the neighbor $\mathbf{d}$ and the query $\mathbf{q}$:

$$\frac{1}{dist(\mathbf{q}, \mathbf{d})^2} \tag{5}$$

- The weighted k nearest neighbor model is defined as:

$$\mathbb{M}_k(\mathbf{q}) = \underset{l \in levels(t)}{\text{argmax}} \sum_{i=1}^{k} \frac{1}{dist(\mathbf{q}, \mathbf{d_i})^2} \times \delta(t_i, l) \tag{6}$$

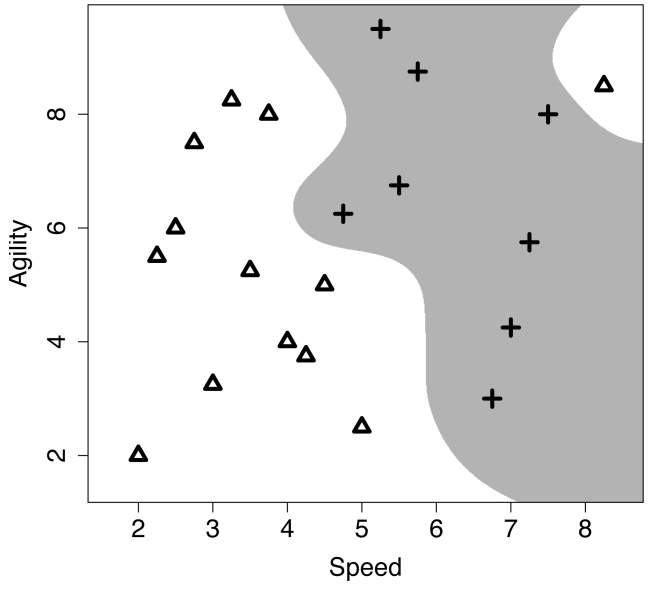


Figure: The weighted k nearest neighbor model decision boundary.

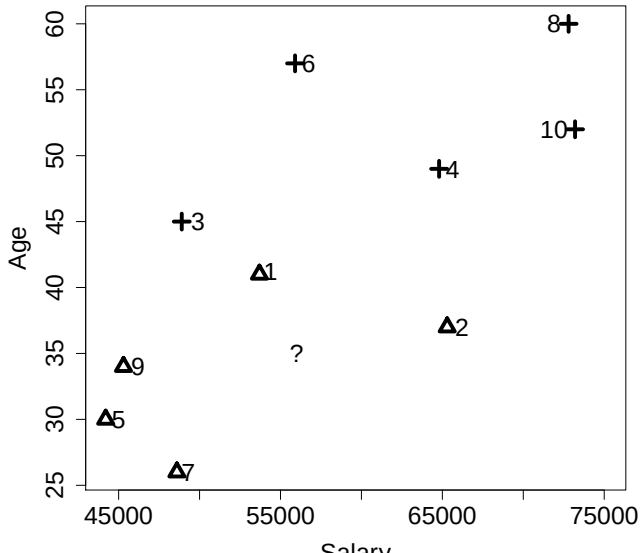
Data Normalization

Table: A dataset listing the salary and age information for customers and whether or not the 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

- The marketing department wants to decide whether or not they should contact a customer with the following profile:

$$\langle \text{SALARY} = 56,000, \text{AGE} = 35 \rangle$$


Age
Salary
25
30
35
40
45
50
55
60
45000
55000
65000
75000
8
6
10
4
3
1
2
?
9
5
7

				Salary and Age		Salary Only		Age Only	
ID	**Salary**	**Age**	**Purch.**	**Dist.**	**Neigh.**	**Dist.**	**Neigh.**	**Dist.**	**Neigh.**
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	5
6	55900	57	Yes	102.3914	1	100	1	22	9
7	48600	26	No	7400.0055	4	7400	4	9	3
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

- This odd prediction is caused by features taking different ranges of values, this is equivalent to features having different variances.
- We can adjust for this using normalization; the equation for range normalization is:

$$a_i^{'} = \frac{a_i - min(a)}{max(a) - min(a)} \times (high - low) + low \tag{7}$$

	Normalized Dataset			Salary and Age		Salary Only		Age Only	
ID	Salary	Age	Purch.	Dist.	Neigh.	Dist.	Neigh.	Dist.	Neigh.
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

- Normalizing the data is an important thing to do for almost all machine learning algorithms, not just nearest neighbor!

Predicting Continuous Targets

- Return the average value in the neighborhood:

$$\mathbb{M}_k(\mathbf{q}) = \frac{1}{k} \sum_{i=1}^{k} t_i \tag{8}$$

Table: A dataset of whiskeys listing the age (in years) and the rating (between 1 and 5, with 5 being the best) and the bottle price of each whiskey.

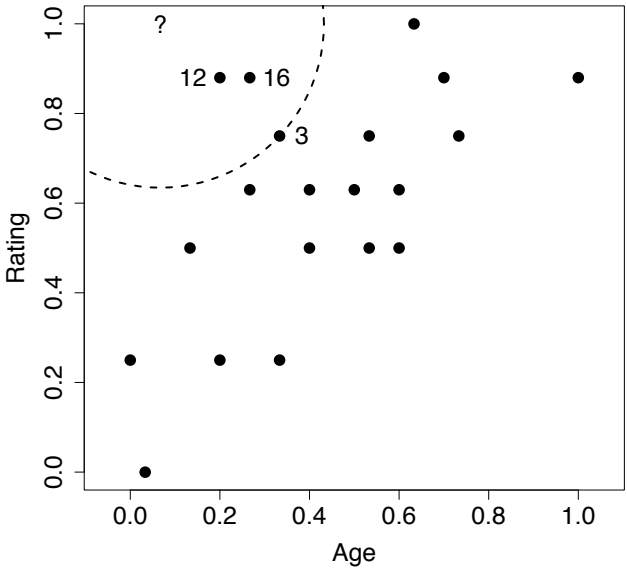
ID	Age	Rating	Price
1	0	2	30.00
2	12	3.5	40.00
3	10	4	55.00
4	21	4.5	550.00
5	12	3	35.00
6	15	3.5	45.00
7	16	4	70.00
8	18	3	85.00
9	18	3.5	78.00
10	16	3	75.00

ID	Age	Rating	Price
11	19	5	500.00
12	6	4.5	200.00
13	8	3.5	65.00
14	22	4	120.00
15	6	2	12.00
16	8	4.5	250.00
17	10	2	18.00
18	30	4.5	450.00
19	1	1	10.00
20	4	3	30.00

Table: The whiskey dataset after the descriptive features have been normalized.

ID	Age	Rating	Price
1	0.0000	0.25	30.00
2	0.4000	0.63	40.00
3	0.3333	0.75	55.00
4	0.7000	0.88	550.00
5	0.4000	0.50	35.00
6	0.5000	0.63	45.00
7	0.5333	0.75	70.00
8	0.6000	0.50	85.00
9	0.6000	0.63	78.00
10	0.5333	0.50	75.00

ID	Age	Rating	Price
11	0.6333	1.00	500.00
12	0.2000	0.88	200.00
13	0.2667	0.63	65.00
14	0.7333	0.75	120.00
15	0.2000	0.25	12.00
16	0.2667	0.88	250.00
17	0.3333	0.25	18.00
18	1.0000	0.88	450.00
19	0.0333	0.00	10.00
20	0.1333	0.50	30.00


?
12
16
3
Rating
Age
0.0
0.2
0.4
0.6
0.8
1.0

- The model will return a price prediction that is the average price of the three neighbors:

$$\frac{200.00 + 250.00 + 55.00}{3} = 168.33$$

- In a weighted k nearest neighbor the model prediction equation is changed to:

$$\mathbb{M}_k(\mathbf{q}) = \frac{\sum_{i=1}^{k} \frac{1}{dist(\mathbf{q}, \mathbf{d}_i)^2} \times t_i}{\sum_{i=1}^{k} \frac{1}{dist(\mathbf{q}, \mathbf{d}_i)^2}} \tag{9}$$

Table: The calculations for the weighted k nearest neighbor prediction

ID	Price	Distance	Weight	Price $\times$ Weight
1	30.00	0.7530	1.7638	52.92
2	40.00	0.5017	3.9724	158.90
3	55.00	0.3655	7.4844	411.64
4	550.00	0.6456	2.3996	1319.78
5	35.00	0.6009	2.7692	96.92
6	45.00	0.5731	3.0450	137.03
7	70.00	0.5294	3.5679	249.75
8	85.00	0.7311	1.8711	159.04
9	78.00	0.6520	2.3526	183.50
10	75.00	0.6839	2.1378	160.33
11	500.00	0.5667	3.1142	1557.09
12	200.00	0.1828	29.9376	5987.53
13	65.00	0.4250	5.5363	359.86
14	120.00	0.7120	1.9726	236.71
15	12.00	0.7618	1.7233	20.68
16	250.00	0.2358	17.9775	4494.38
17	18.00	0.7960	1.5783	28.41
18	450.00	0.9417	1.1277	507.48
19	10.00	1.0006	0.9989	9.99
20	30.00	0.5044	3.9301	117.90
		Totals:	99.2604	16 249.85

Other Measures of Similarity

Table: A binary dataset listing the behavior of two individuals on a website during a trial period and whether or not they subsequently signed-up for the website.

ID	Profile	FAQ	Help Forum	Newsletter	Liked	Signup
1	1	1	1	0	1	Yes
2	1	0	0	0	0	No

Who is q more similar to d_1 or d_2?

$$\mathbf{q} = \langle \text{PROFILE:1}, \text{FAQ:0}, \text{HELP FORUM:1}, \text{NEWSLETTER:0}, \text{LIKED:0} \rangle$$

ID	Profile	FAQ	Help Forum	Newsletter	Liked	Signup
1	1	1	1	0	1	Yes
2	1	0	0	0	0	No

		q	
		Pres.	Abs.
$\mathbf{d}_1$	Pres.	CP=2	PA=0
	Abs.	AP=2	CA=1

		q	
		Pres.	Abs.
$\mathbf{d}_2$	Pres.	CP=1	PA=1
	Abs.	AP=0	CA=3

Table: The similarity between the current trial user, $\mathbf{q}$, and the two users in the dataset, $\mathbf{d}_1$ and $\mathbf{d}_2$, in terms of co-presence (CP), co-absence (CA), presence-absence (PA), and absence-presence (AP).

Russel-Rao

- The ratio between the number of co-presenses and the total number of binary features considered.

$$sim_{RR}(\mathbf{q}, \mathbf{d}) = \frac{CP(\mathbf{q}, \mathbf{d})}{|\mathbf{q}|} \tag{10}$$

Russel-Rao

$$sim_{RR}(\mathbf{q}, \mathbf{d}) = \frac{CP(\mathbf{q}, \mathbf{d})}{|\mathbf{q}|} \tag{11}$$

$\langle$PROFILE:1, FAQ:0, HELP FORUM:1, NEWSLETTER:0, LIKED:0, $\rangle$

		q	
		Pres.	Abs.
$\mathbf{d}_1$	Pres.	CP=2	PA=0
	Abs.	AP=2	CA=1

		q	
		Pres.	Abs.
$\mathbf{d}_2$	Pres.	CP=1	PA=1
	Abs.	AP=0	CA=3

Russel-Rao

$$sim_{RR}(\mathbf{q}, \mathbf{d}) = \frac{CP(\mathbf{q}, \mathbf{d})}{|\mathbf{q}|} \tag{12}$$

Example

$$sim_{RR}(\mathbf{q}, \mathbf{d}_1) = \frac{2}{5} = 0.4$$
$$sim_{RR}(\mathbf{q}, \mathbf{d}_2) = \frac{1}{5} = 0.2$$

- The current trial user is judged to be more similar to instance $\mathbf{d}_1$ then $\mathbf{d}_2$.

Sokal-Michener

- Sokal-Michener is defined as the ratio between the total number of co-presences and co-absences, and the total number of binary features considered.

$$sim_{SM}(\mathbf{q}, \mathbf{d}) = \frac{CP(\mathbf{q}, \mathbf{d}) + CA(\mathbf{q}, \mathbf{d})}{|\mathbf{q}|} \tag{13}$$

Sokal-Michener

$$sim_{SM}(\mathbf{q}, \mathbf{d}) = \frac{CP(\mathbf{q}, \mathbf{d}) + CA(\mathbf{q}, \mathbf{d})}{|\mathbf{q}|} \tag{14}$$

$\langle$PROFILE:1, FAQ:0, HELP FORUM:1, NEWSLETTER:0, LIKED:0, $\rangle$

		q	
		Pres.	Abs.
$\mathbf{d}_1$	Pres.	CP=2	PA=0
	Abs.	AP=2	CA=1

		q	
		Pres.	Abs.
$\mathbf{d}_2$	Pres.	CP=1	PA=1
	Abs.	AP=0	CA=3

Sokal-Michener

$$sim_{SM}(\mathbf{q}, \mathbf{d}) = \frac{CP(\mathbf{q}, \mathbf{d}) + CA(\mathbf{q}, \mathbf{d})}{|\mathbf{q}|} \tag{15}$$

Example

$$sim_{SM}(\mathbf{q}, \mathbf{d}_1) = \frac{3}{5} = 0.6$$
$$sim_{SM}(\mathbf{q}, \mathbf{d}_2) = \frac{4}{5} = 0.8$$

- The current trial user is judged to be more similar to instance $\mathbf{d}_2$ then $\mathbf{d}_1$.

Jaccard

- The Jacquard index ignore co-absences

$$sim_J(\mathbf{q}, \mathbf{d}) = \frac{CP(\mathbf{q}, \mathbf{d})}{CP(\mathbf{q}, \mathbf{d}) + PA(\mathbf{q}, \mathbf{d}) + AP(\mathbf{q}, \mathbf{d})} \tag{16}$$

Jaccard

$$sim_J(\mathbf{q}, \mathbf{d}) = \frac{CP(\mathbf{q}, \mathbf{d})}{CP(\mathbf{q}, \mathbf{d}) + PA(\mathbf{q}, \mathbf{d}) + AP(\mathbf{q}, \mathbf{d})} \tag{17}$$

$\langle$PROFILE:1, FAQ:0, HELP FORUM:1, NEWSLETTER:0, LIKED:0, $\rangle$

		q	
		Pres.	Abs.
$\mathbf{d}_1$	Pres.	CP=2	PA=0
	Abs.	AP=2	CA=1

		q	
		Pres.	Abs.
$\mathbf{d}_2$	Pres.	CP=1	PA=1
	Abs.	AP=0	CA=3

Jaccard

$$sim_J(\mathbf{q}, \mathbf{d}) = \frac{CP(\mathbf{q}, \mathbf{d})}{CP(\mathbf{q}, \mathbf{d}) + PA(\mathbf{q}, \mathbf{d}) + AP(\mathbf{q}, \mathbf{d})} \tag{18}$$

Example

$$sim_J(\mathbf{q}, \mathbf{d}_1) = \frac{2}{4} = 0.5$$
$$sim_J(\mathbf{q}, \mathbf{d}_2) = \frac{1}{2} = 0.5$$

- The current trial user is judged to be equally similar to instance $\mathbf{d}_1$ and $\mathbf{d}_2$!

- Cosine similarity between two instances is the cosine of the inner angle between the two vectors that extend from the origin to each instance.

Cosine

$$sim_{COSINE}(\mathbf{a}, \mathbf{b}) = \frac{(\mathbf{a}[1] \times \mathbf{b}[1]) + \cdots + (\mathbf{a}[m] \times \mathbf{b}[m])}{\sqrt{\sum_{i=1}^{m} \mathbf{a}[i]^2} \times \sqrt{\sum_{i=1}^{m} \mathbf{b}[i]^2}}$$

- Calculate the cosine similarity between the following two instances:

$$\mathbf{d}_1 = \langle \text{SMS} = 97, \text{VOICE} = 21 \rangle$$

$$\mathbf{d}_2 = \langle \text{SMS} = 18, \text{VOICE} = 184 \rangle$$

$$sim_{COSINE}(\mathbf{a}, \mathbf{b}) = \frac{\sum_{i=1}^{m} (\mathbf{a}[i] \times \mathbf{b}[i])}{\sqrt{\sum_{i=1}^{m} \mathbf{a}[i]^2} \times \sqrt{\sum_{i=1}^{m} \mathbf{b}[i]^2}}$$

- Calculate the cosine similarity between the following two instances:

$$\mathbf{d}_1 = \langle \text{SMS} = 97, \text{VOICE} = 21 \rangle$$

$$\mathbf{d}_2 = \langle \text{SMS} = 18, \text{VOICE} = 184 \rangle$$

$$sim_{COSINE}(\mathbf{d}_1, \mathbf{d}_1) = \frac{(97 \times 181) + (21 \times 184)}{\sqrt{97^2 + 21^2} \times \sqrt{181^2 + 184^2}}$$

$$= 0.8362$$

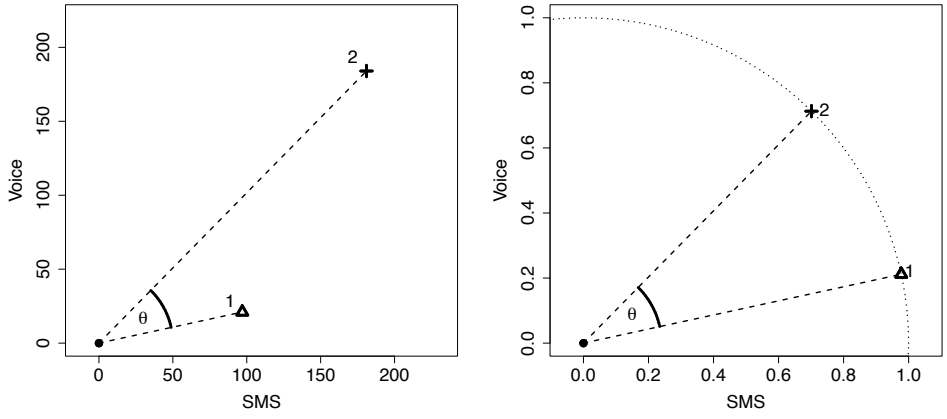


Figure: (a) The θ represents the inner angle between the vector emanating from the origin to instance $\mathbf{d}_1$ $\langle \text{SMS} = 97, \text{VOICE} = 21 \rangle$ and the vector emanating from the origin to instance $\mathbf{d}_2$ $\langle \text{SMS} = 181, \text{VOICE} = 184 \rangle$; (b) shows $\mathbf{d}_1$ and $\mathbf{d}_2$ normalized to the unit circle.

- Calculate the cosine similarity between the following two instances:

$$\mathbf{d}_1 = \langle \text{SMS} = 97, \text{VOICE} = 21 \rangle$$

$$\mathbf{d}_3 = \langle \text{SMS} = 194, \text{VOICE} = 42 \rangle$$

$$sim_{COSINE}(\mathbf{a}, \mathbf{b}) = \frac{\sum_{i=1}^{m} (\mathbf{a}[i] \times \mathbf{b}[i])}{\sqrt{\sum_{i=1}^{m} \mathbf{a}[i]^2} \times \sqrt{\sum_{i=1}^{m} \mathbf{b}[i]^2}}$$

- Calculate the cosine similarity between the following two instances:

$$\mathbf{d}_1 = \langle \text{SMS} = 97, \text{VOICE} = 21 \rangle$$

$$\mathbf{d}_3 = \langle \text{SMS} = 194, \text{VOICE} = 42 \rangle$$

$$sim_{COSINE}(\mathbf{d}_1, \mathbf{d}_1) = \frac{(97 \times 194) + (21 \times 42)}{\sqrt{97^2 + 21^2} \times \sqrt{194^2 + 42^2}}$$

$$= 1$$

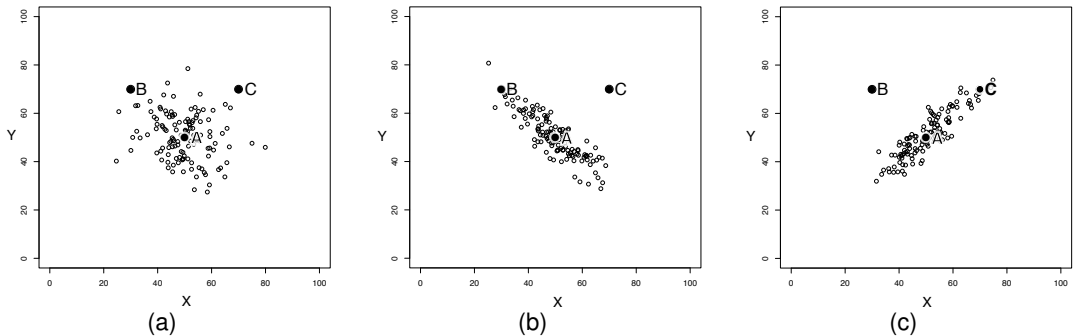


Figure: Scatter plots of three bivariate datasets with the same center point A and two queries B and C both equidistant from A. (a) A dataset uniformly spread around the center point. (b) A dataset with negative covariance. (c) A dataset with positive covariance.

- The mahalanobis distance uses covariance to scale distances so that distances along a direction where the dataset is spreadout a lot are scaled down and distances along directions where the dataset is tightly packed are scaled up.

$$Mahalanobis(\mathbf{a}, \mathbf{b}) = \sqrt{[\mathbf{a}[1] - \mathbf{b}[1], \ldots, \mathbf{a}[m] - \mathbf{b}[m]] \times \sum^{-1} \times \begin{bmatrix} \mathbf{a}[1] - \mathbf{b}[1] \\ \ldots \\ \mathbf{a}[m] - \mathbf{b}[m] \end{bmatrix}} \tag{19}$$

- Similar to Euclidean distance, the mahalanobis distance squares the differences of the features.
- But it also rescales the differences (using the inverse covariance matrix) so that all the features have unit variance and the effects of covariance is removed.

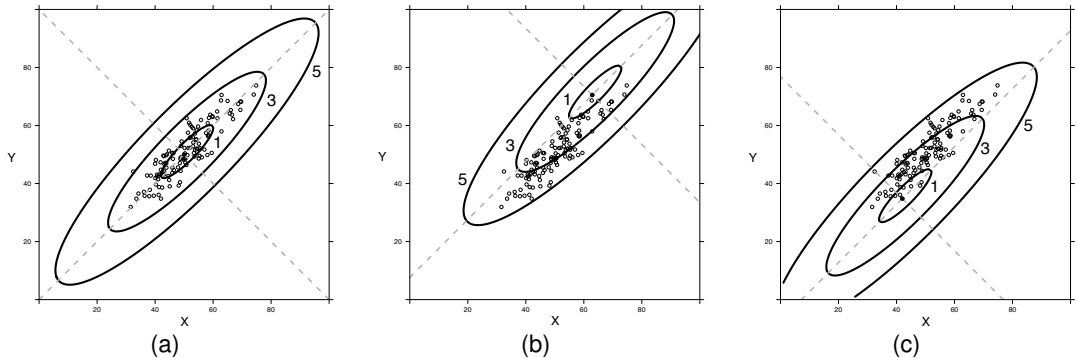


Figure: The coordinate systems defined by the mahalanobis metric using the co-variance matrix for the dataset in Figure 15(c) [69] using three different origins: (a) $(50, 50)$, (b) $(63, 71)$, (c) $(42, 35)$. The ellipses in each figure plot the 1, 3, and 5 unit distances contours from each origin.

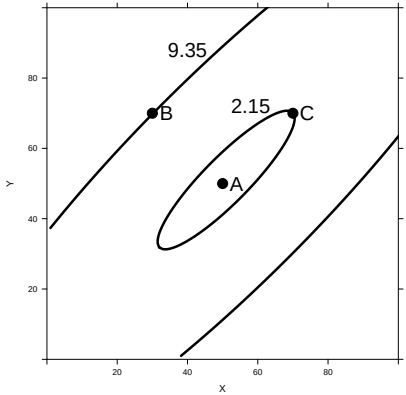


Figure: The effect of using a mahalanobis versus euclidean distance. Point A is the center of mass of the dataset in Figure 15(c) [69]. The ellipses plot the mahalanobis distance contours from A that B and C lie on. In euclidean terms B and C are equidistant from A, however using the mahalanobis metric C is much closer to A than B.

Feature Selection

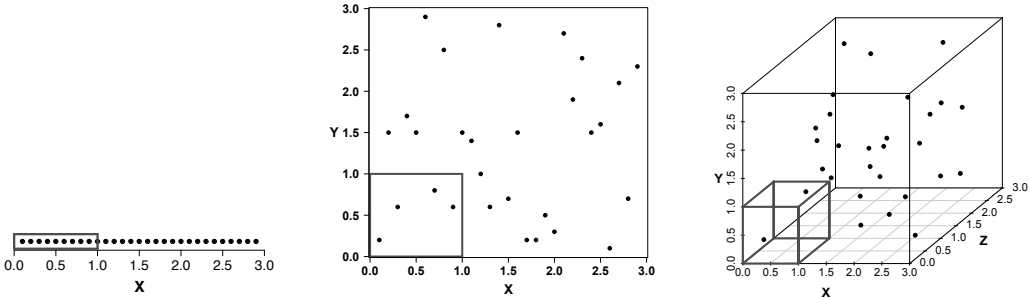
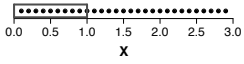
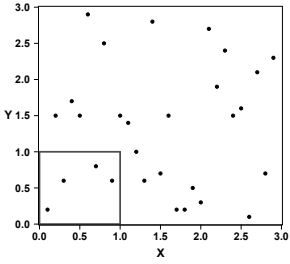


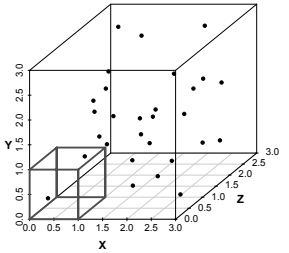
Figure: A set of scatter plots illustrating the curse of dimensionality. Across figures (a), (b) and (c) the density of the marked unit hypercubes decreases as the number of dimensions increases.


0.0 0.5 1.0 1.5 2.0 2.5 3.0
X

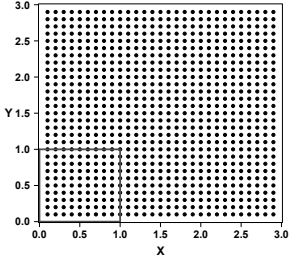
(a)


3.0 2.5 2.0 1.5 1.0 0.5 0.0
Y
0.0 0.5 1.0 1.5 2.0 2.5 3.0
X

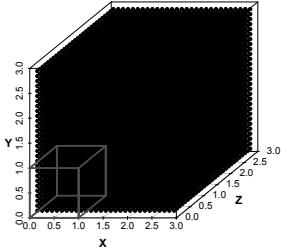
(b)


Y
X
Z

(c)


3.0 2.5 2.0 1.5 1.0 0.5 0.0
Y
0.0 0.5 1.0 1.5 2.0 2.5 3.0
X

(d)


Y
X
Z

(e)

- During our discussion of feature selection approaches it will be useful to distinguish between different classes of descriptive features:
  1. Predictive
  2. Interacting
  3. Redundant
  4. Irrelevant

- When framed as a local search problem feature selection is defined in terms of an iterative process consisting of the following stages:
  1. Subset Generation
  2. Subset Selection
  3. Termination Condition
- The search can move through the search space in a number of ways:
  - Forward sequential selection
  - Backward sequential selection

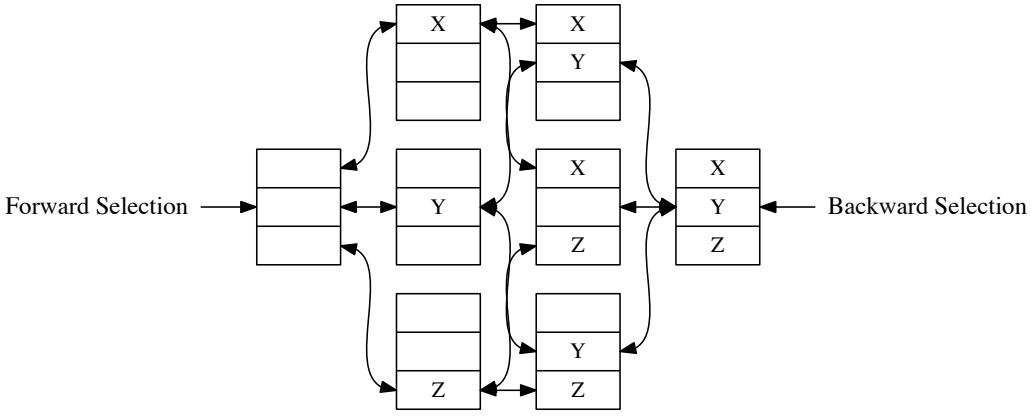


Figure: Feature Subset Space for a dataset with 3 features X, Y, Z.

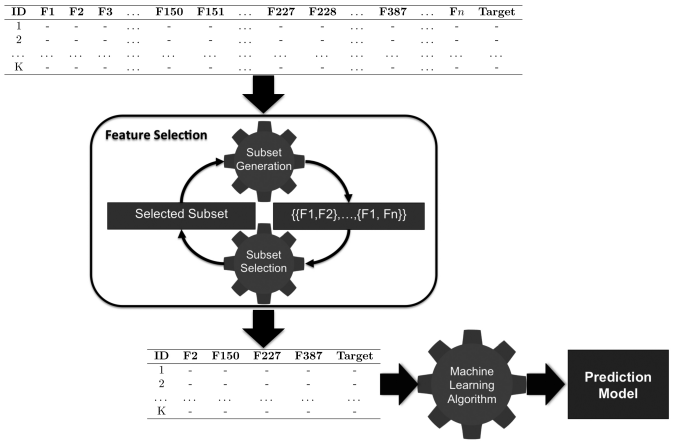


Figure: The process of model induction with feature selection.

Efficient Memory Search

- Assuming that the training set will remain relatively stable it is possible to speed up the prediction speed of a nearest neighbor model by investing in some one-off computation to create an index of the instances that enables efficient retrieval of the nearest neighbors.
- The k-d tree, which is short for k-dimensional tree, is one of the best known of these indexes.

Example

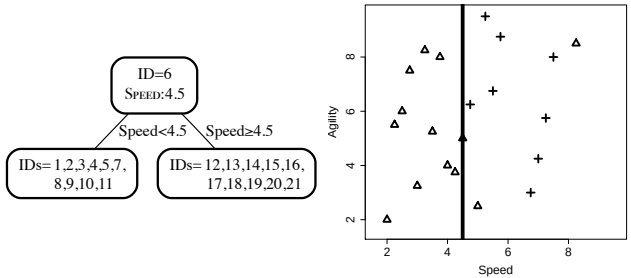
Let's build a k-d tree for the college athlete dataset

Table: The speed and agility ratings for 22 college athletes labelled with the decisions for whether they were drafted or not.

ID	Speed	Agility	Draft
1	2.50	6.00	No
2	3.75	8.00	No
3	2.25	5.50	No
4	3.25	8.25	No
5	2.75	7.50	No
6	4.50	5.00	No
7	3.50	5.25	No
8	3.00	3.25	No
9	4.00	4.00	No
10	4.25	3.75	No
11	2.00	2.00	No
12	5.00	2.50	No
13	8.25	8.50	No
14	5.75	8.75	Yes
15	4.75	6.25	Yes
16	5.50	6.75	Yes
17	5.25	9.50	Yes
18	7.00	4.25	Yes
19	7.50	8.00	Yes
20	7.25	5.75	Yes
21	6.75	3.00	Yes

Example

First split on the SPEED feature

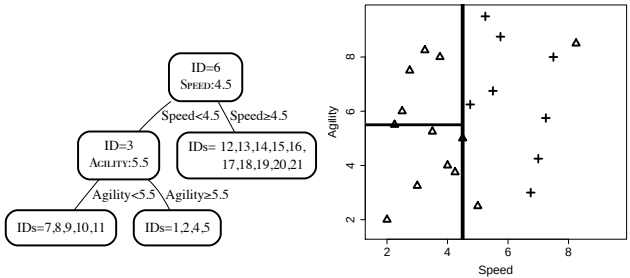


(a) (b)

Figure: The k-d tree generated, for the dataset in Table 11 [83] after the initial split using the SPEED feature. (b) the partitioning of the feature space by the k-d tree in (a).

Example

Next split on the AGILITY feature

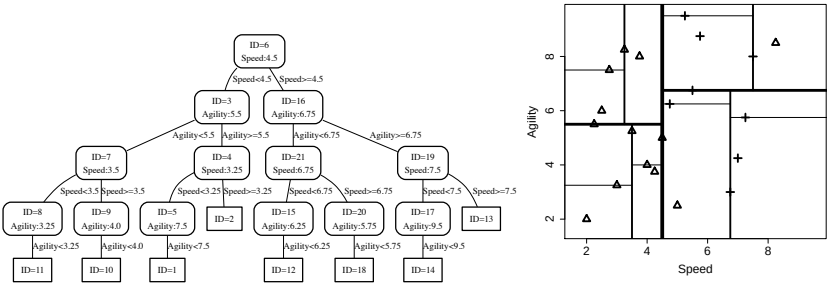


(a) (b)

Figure: (a) the k-d tree after the dataset at the left child of the root has been split using the AGILITY feature with a threshold of 5.5. (b) the partitioning of the feature space by the k-d tree in (a)

Example

After completing the tree building process

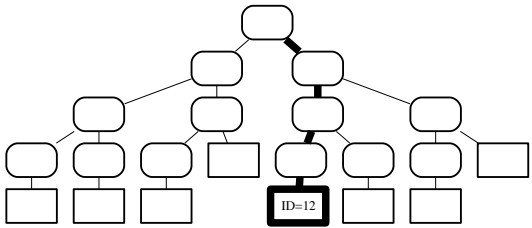


(a) (b)

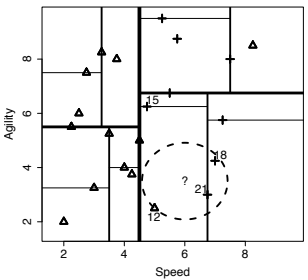
Figure: (a) The complete k-d tree generated, for the dataset in Table 11 [83]. (b) the partitioning of the feature space by the k-d tree in (a)

Example

Finding neighbors for query SPEED= 6, AGILITY= 3.5.



(a)



(b)

Example

Finding neighbors for query SPEED= 6, AGILITY= 3.5.

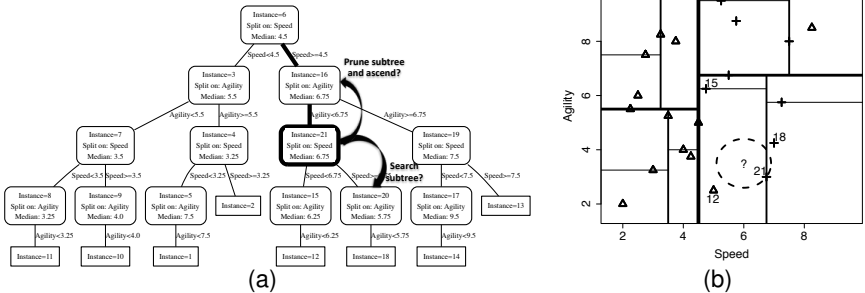


Figure: (a) the state of the retrieval process after instance $\mathbf{d}_{21}$ has been stored as *current-best*. (b) the dashed circle illustrates the extent of the target hypersphere after *current-best-distance* has been updated.

Example

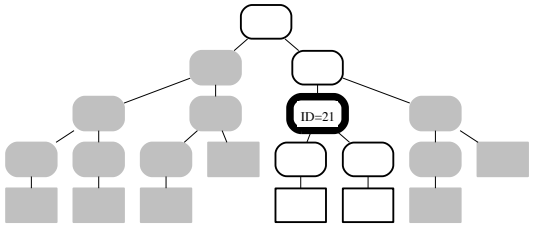


Figure: The greyed out branches indicate the portions of the k-d tree pruned from the search. The node with the bold outline represents the instance ($\mathbf{d}_{21}$) that was returned as the nearest neighbor to the query.

Summary

- Similarity-based prediction models attempt to mimic a very human way of reasoning by basing predictions for a target feature value on the most similar instances in memory—this makes them easy to interpret and understand.
- This advantage should not be underestimated as being able to understand how the model works gives people more confidence in the model and, hence, in the insight that it provides.
- The inductive bias underpinning similarity-based classification is that things that are similar (i.e., instances that have similar descriptive features) belong to the same class.
- The nearest neighbor algorithm creates an implicit global predictive model by aggregating local models, or neighborhoods.
- The definition of these neighborhoods is based on proximity within the feature space to the labelled training instances.

- Queries are classified using the label of the training instance defining the neighborhood in the feature space that contains the query. item Nearest neighbor models are very sensitive to noise in the target feature the easiest way to solve this problem is to employ a ***k* nearest neighbor**.
- **Normalization** techniques should almost always be applied when nearest neighbor models are used.
- It is easy to adapt a nearest neighbor model to **continuous targets**.
- There are many different measures of **similarity**.
- **Feature selection** is a particularly important process for nearest neighbor algorithms it alleviates the **curse of dimensionality**.
- As the number of instances becomes large, a nearest neighbor model will become slower—techniques such as the ***k-d* tree** can help with this issue.