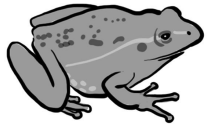
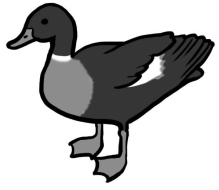
Similarity-based Learning

- Big Idea
- Fundamentals
  1. Feature Space
  2. Distance Metrics
- Standard Approach: The Nearest Neighbour Algorithm
- Handling Noisy Data
- Data Normalization
- Other Measures of Similarity
- Feature Selection

Big Idea

- It's 1798, and as Lieutenant-Colonel David Collins aboard HMS Calcutta, you're exploring near the Hawkesbury River in New South Wales.
- Following a river expedition, a crew member reports seeing an unusual animal with webbed feet, a duck-bill snout, and a growl.
- To prepare for tomorrow's journey, you aim to identify the animal to assess any potential danger.

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

Matching animals you remember to the features of the unknown animal described by the sailor.

- To classify something, search your memory for similar items and assign it the class of the most similar item.

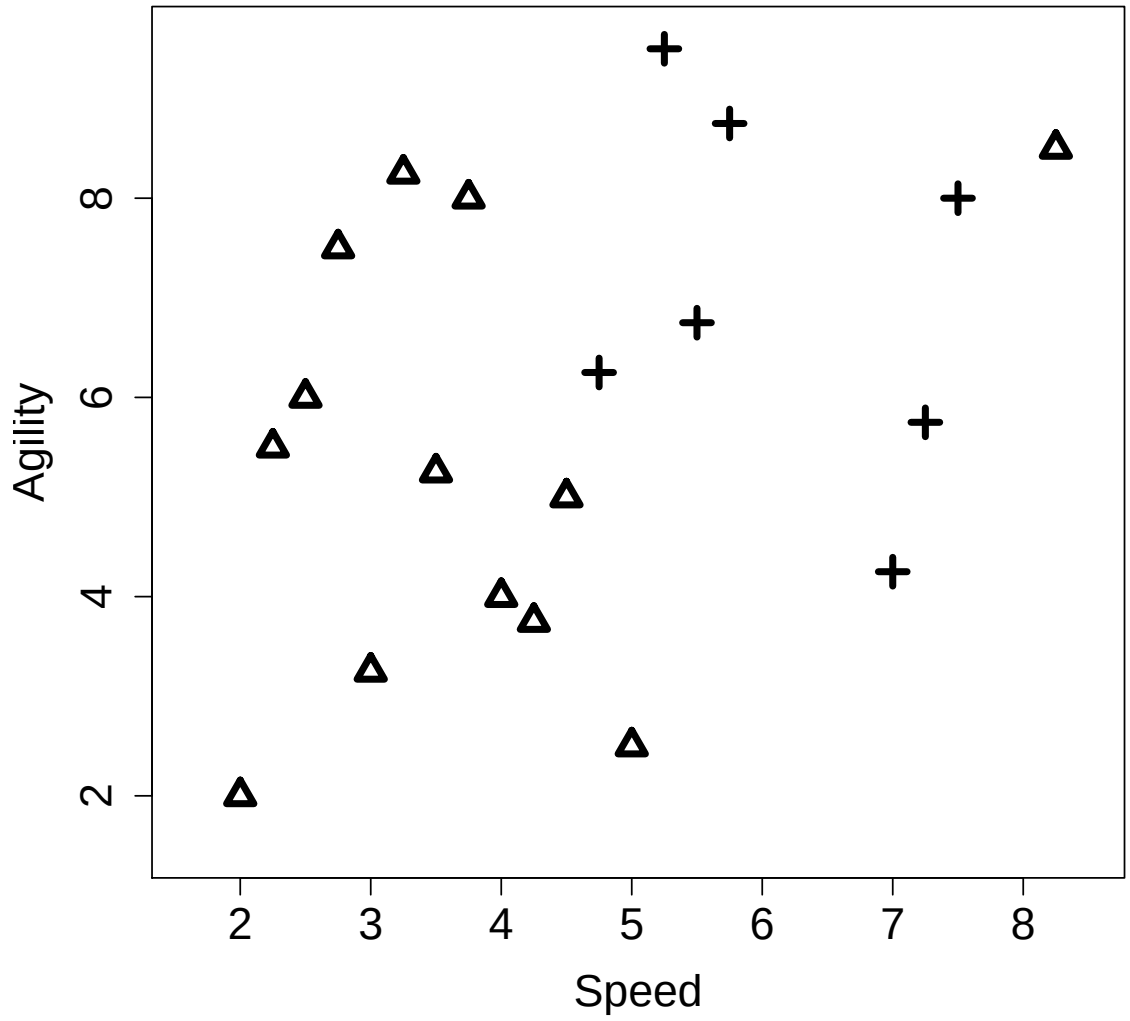
Fundamentals

- The fundamentals of similarity-based learning are:
  - Feature space: n-dimensional abstract space formed by using descriptive features of as axes, with each dataset instance represented as a point based on its feature values.
  - Similarity metrics: measures similarity between two instances according to a feature space

Feature Space

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

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



A feature space plot. The triangles represent *'Non-draft'* instances and the crosses represent the *'Draft'* instances.

Distance Metrics

- Mathematically, a metric must conform to the following four criteria:
  1. Non-negativity: $metric(\mathbf{a}, \mathbf{b}) \geq 0$
  2. Identity: $metric(\mathbf{a}, \mathbf{b}) = 0 \Longleftrightarrow \mathbf{a} = \mathbf{b}$
  3. Symmetry: $metric(\mathbf{a}, \mathbf{b}) = metric(\mathbf{b}, \mathbf{a})$
  4. Triangular Inequality: $metric(\mathbf{a}, \mathbf{b}) \leq metric(\mathbf{a}, \mathbf{c}) + metric(\mathbf{b}, \mathbf{c})$

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

$$\begin{aligned} \mathit{Euclidean}(\langle 5.00, 2.50\rangle, \langle 2.75, 7.50\rangle) &= \sqrt{(5.00-2.75)^2 + (2.50-7.50)^2} \\ &= \sqrt{30.0625} = 5.4829 \end{aligned}$$

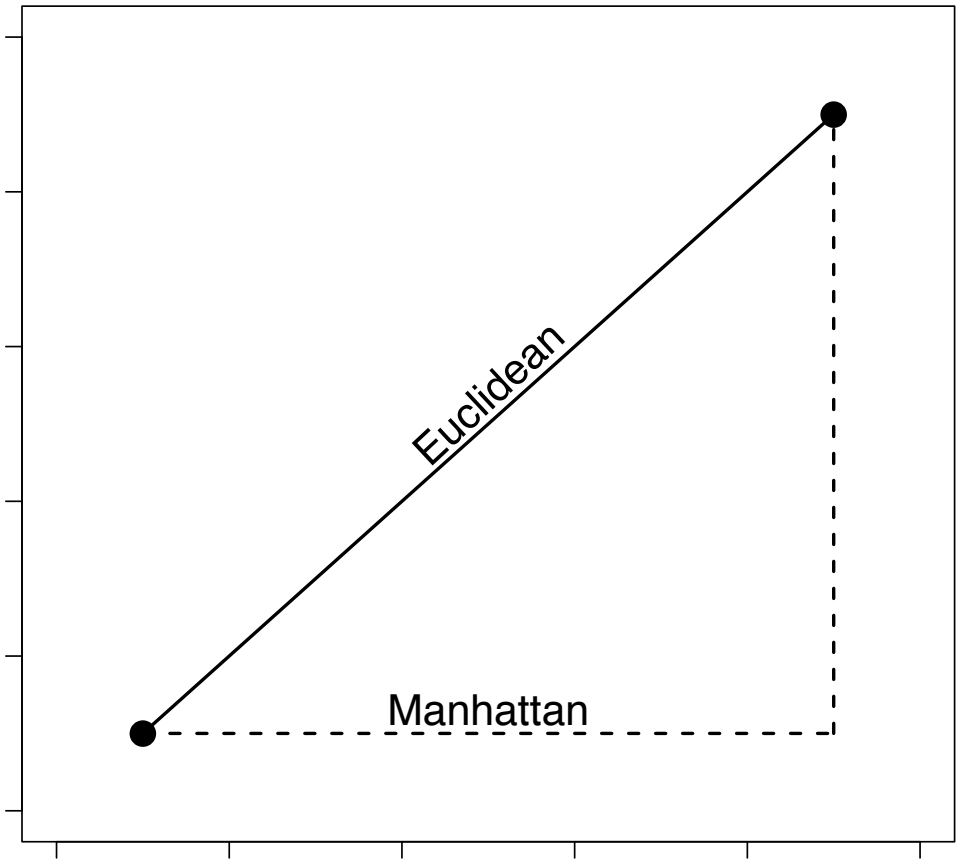
- Manhattan distance or taxi-cab distance between two instances **a** and **b** in a feature space with m dimensions is:

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

Example

The Manhattan distance between instances d_{12} (SPEED= 5.00, AGILITY= 2.5) and d_5 (SPEED= 2.75,AGILITY= 7.5)is:

$$\begin{aligned} Manhattan(\langle 5.00, 2.50\rangle, \langle 2.75, 7.50\rangle) &= abs(5.00 - 2.75) + abs(2.5 - 7.5) \\ &= 2.25 + 5 = 7.25 \end{aligned}$$



The Manhattan and Euclidean distances between two points.

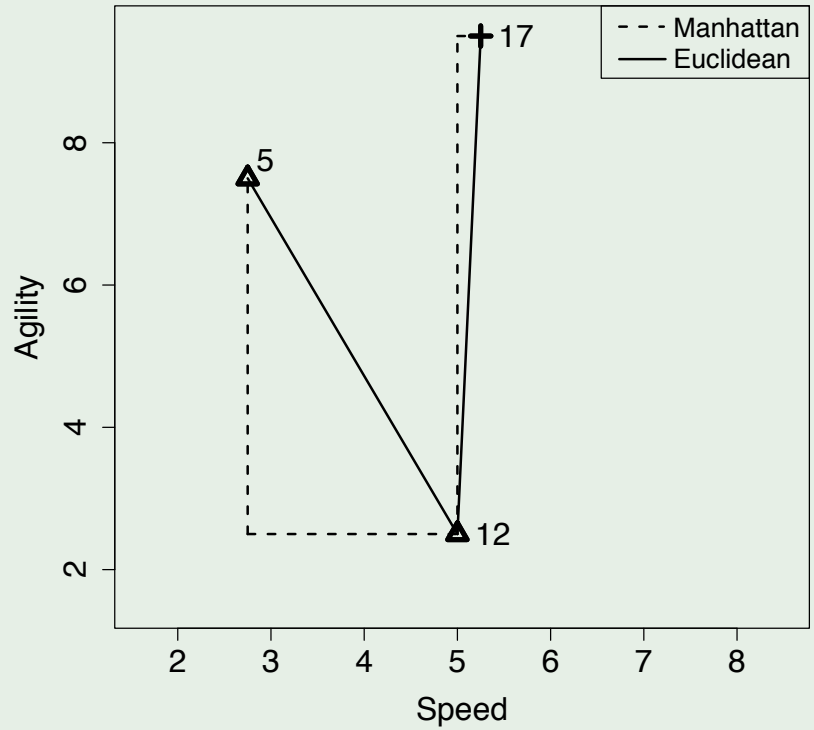
- The Euclidean and Manhattan distances are special cases of Minkowski distance
- The Minkowski distance between two instances **a** and **b** in a feature space with m descriptive features is:

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

  where different values of the parameter p result in different distance metrics
- The Minkowski distance with $p = 1$ is the Manhattan distance and with $p = 2$ is the Euclidean distance.
- The larger the value of p the more emphasis is placed on the features with large differences in values because these differences are raised to the power of p.

Example

Instance ID	Instance ID	Manhattan (Minkowski p=1)	Euclidean (Minkowski p=2)
12	5	7.25	5.4829
12	17	7.25	7.0045



The Manhattan and Euclidean distances between instances $\mathbf{d}_{12}$ (SPEED= 5.00, AGILITY= 2.5) and $\mathbf{d}_5$ (SPEED= 2.75, AGILITY= 7.5) and between instances $\mathbf{d}_{12}$ and $\mathbf{d}_{17}$ (SPEED= 5.25, AGILITY= 9.5).

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.

A Worked Example

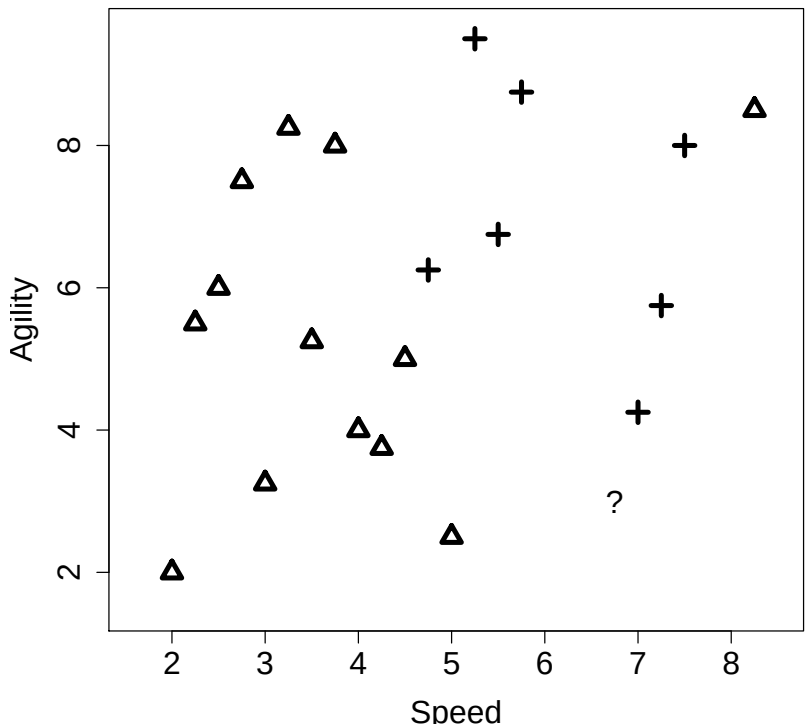
The speed and agility ratings for 20 college athletes labelled with the decisions for whether they were drafted or not.

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

Example

- Should we draft an athlete with the following profile:

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

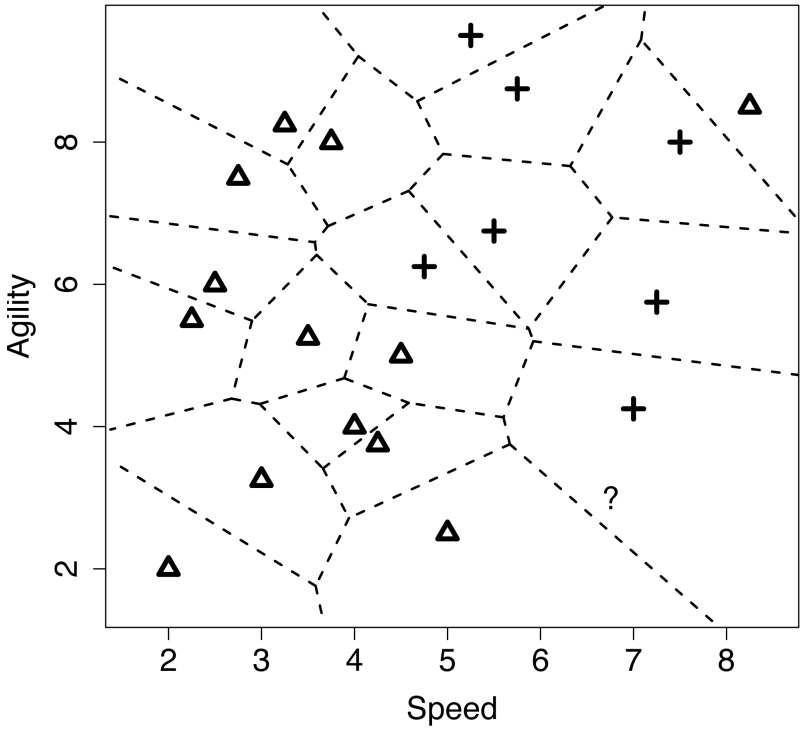


A feature space plot with the position in the feature space of the query represented by the ? marker. The triangles represent *'Non-draft'* instances and the crosses represent the *'Draft'* instances.

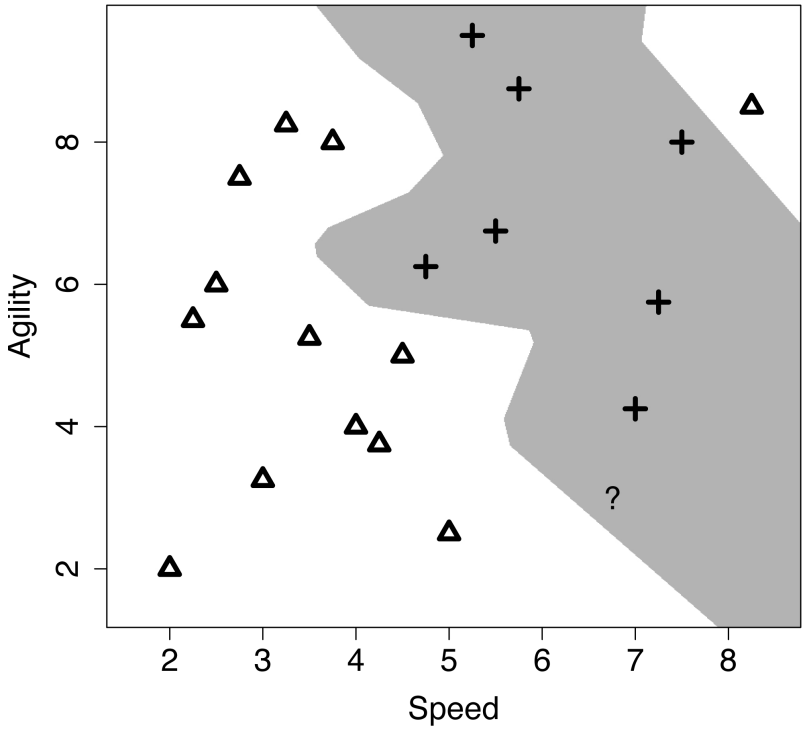
Dist. between the query SPEED $= 6.75$ and AGILITY $= 3.00$ and each instance

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

ID	SPEED	AGILITY	DRAFT	Dist.
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



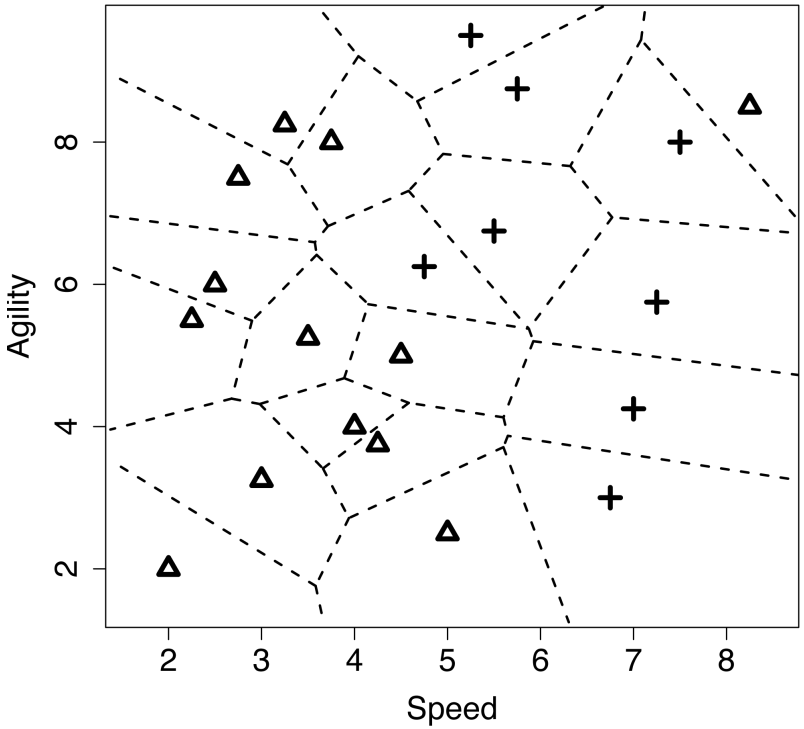
(b) Decision boundary ($k = 1$)

(a) The Voronoi tessellation of the feature space with the position of the query represented by the ? marker; (b) the decision boundary by aggregating the neighboring Voronoi regions that belong to the same target level.

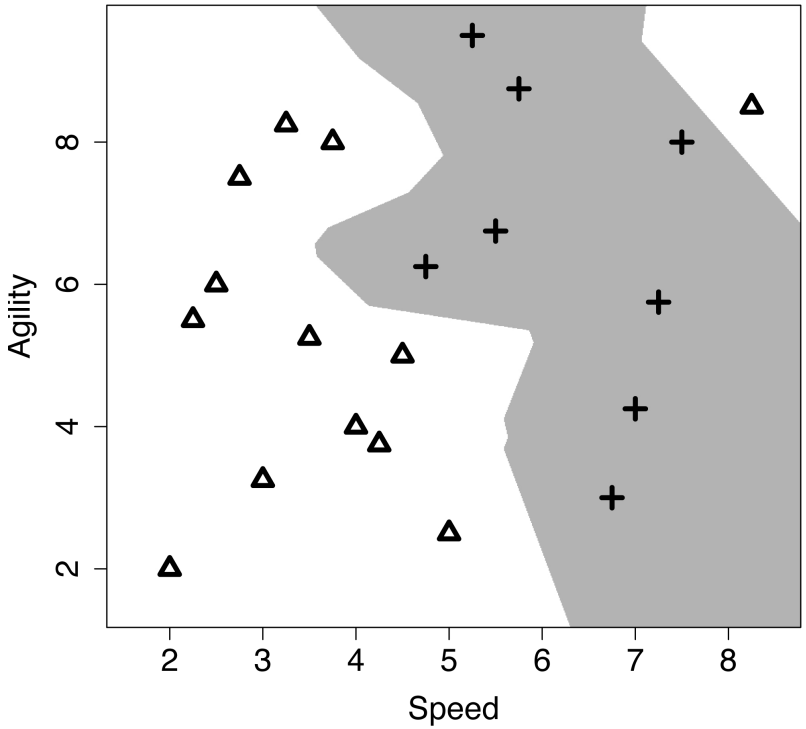
- We can add in new data to update the model easily.

The extended version of the college athletes dataset.

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



(c) Voronoi tessellation

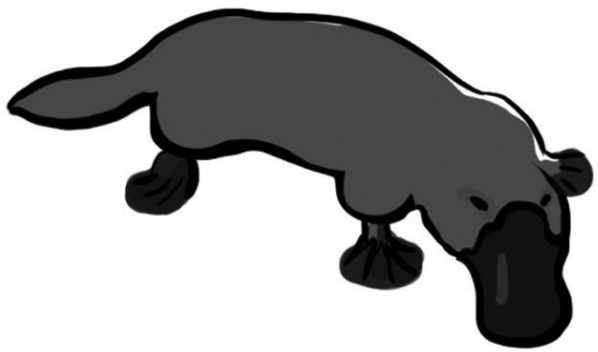


(d) Decision boundary ($k = 1$)

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

Epilogue

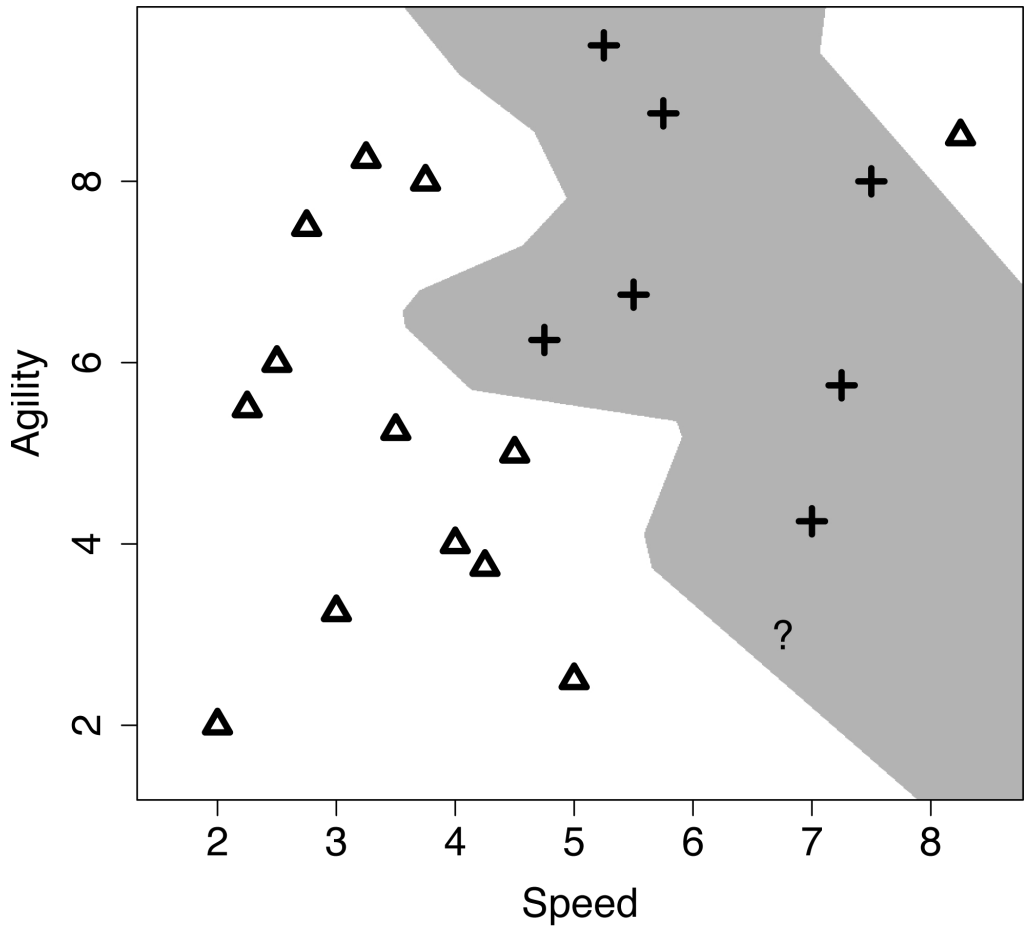
- Returning to 1798 and HMS Calcutta, it turns out that you are the first Europeans to encounter a platypus.



A duck-billed platypus.

- ML is based on the stationarity assumption which states that the data doesn't change - remains stationary - over time.
- ML creates models based on classes they are induced from. If a classification model is trained to distinguish between lions, frogs and ducks, the model will classify a query as being either a lion, a frog or a duck; even if the query is actually a platypus.

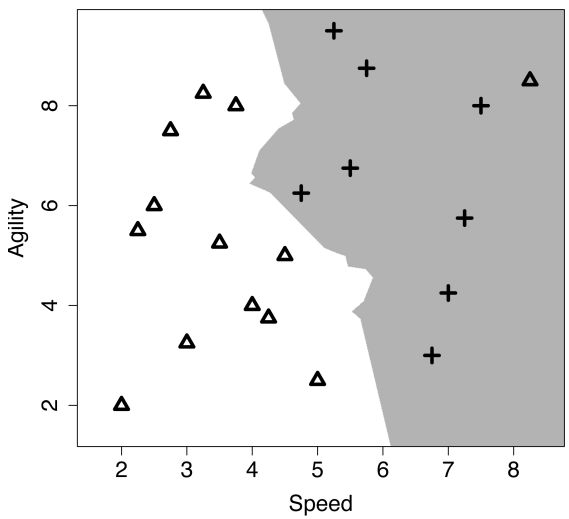
Handling Noisy Data



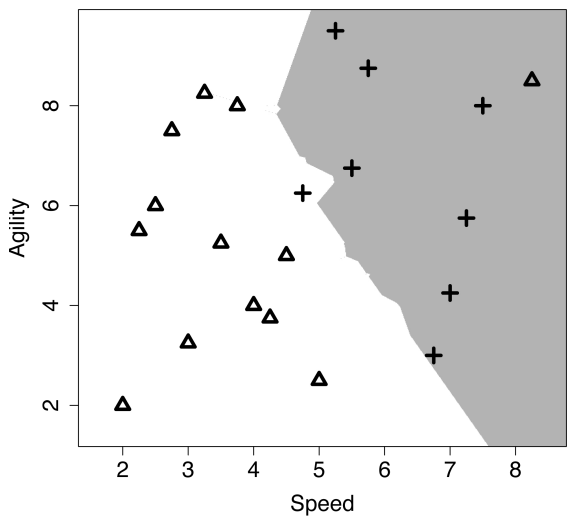
Is the instance at the top right of the diagram really *noise*?

- The k nearest neighbors model predicts the target level with the majority vote from the set of k nearest neightbors to the query **q**:

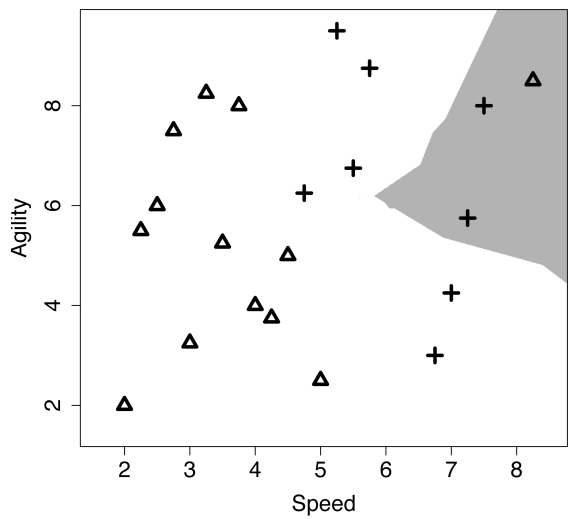
$$\mathbb{M}_k(\mathbf{q}) = \underset{l \in levels(t)}{\text{argmax}} \sum_{i=1}^{k} \delta(t_i, l) \tag{4}$$



$k = 3$



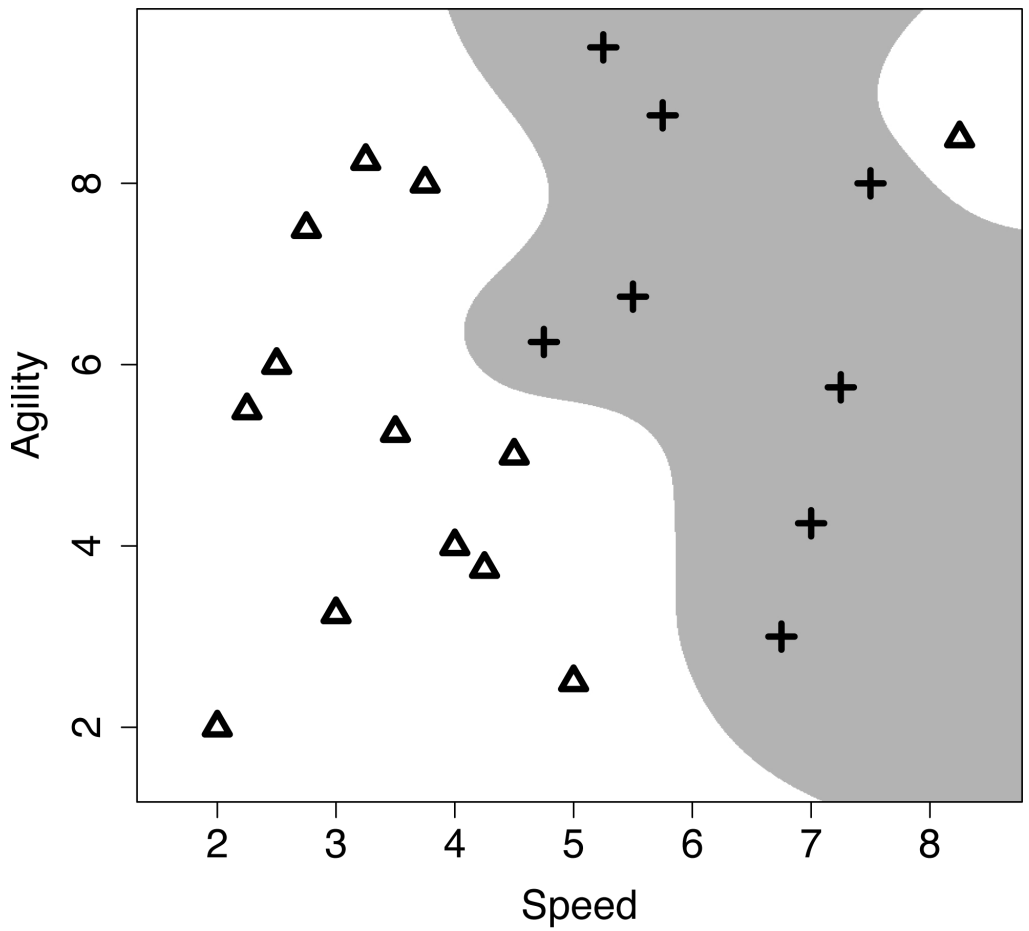
$k = 5$



$k = 15$

- The weighted k nearest neighbor model is defined as:

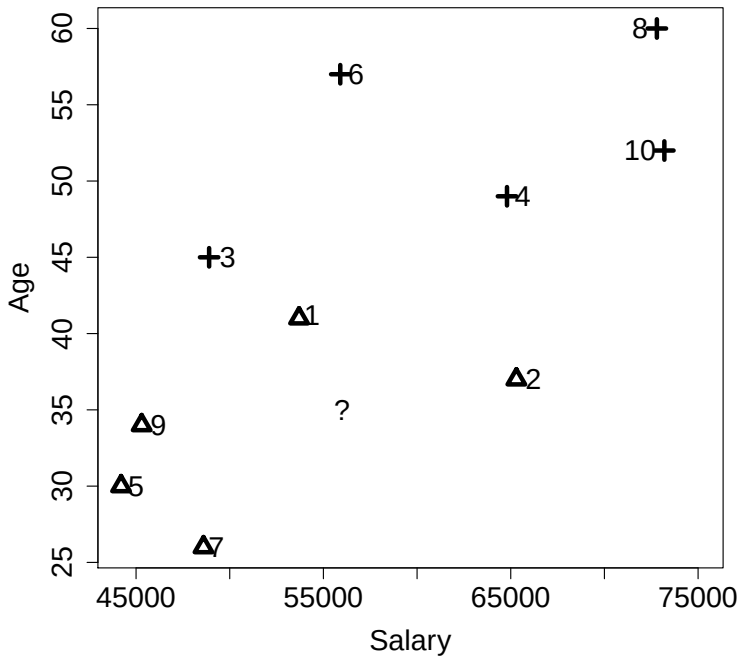
$$\mathbb{M}_k(\mathbf{q}) = \underset{l \in levels(t)}{\text{argmax}} \sum_{i=1}^{k} \frac{1}{dist(\mathbf{q}, \mathbf{d_i})^2} \times \delta(t_i, l) \tag{5}$$



Data Normalization

Pension plan dataset

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 they should contact a customer with the following profile:

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

				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 with:

$$a_i' = \frac{a_i - min(a)}{max(a) - min(a)} \times (high - low) + low \tag{6}$$

	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!

Other Measures of Similarity

A binary dataset listing the behavior of two individuals and whether they 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?

$\mathbf{q} = \langle$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

The similarity between **q**, and $\mathbf{d}_1$ and $\mathbf{d}_2$, in terms of co-presence (CP), co-absence (CA), presence-absence (PA), and absence-presence (AP).

Jaccard

- The Jacquard index ignore co-absences

$$sim_J(\mathbf{q}, \mathbf{d}) = \frac{CP(\mathbf{q}, \mathbf{d})}{CP(\mathbf{q}, \mathbf{d}) + PA(\mathbf{q}, \mathbf{d}) + AP(\mathbf{q}, \mathbf{d})} \tag{7}$$

⟨PROFILE:1, FAQ:0, HELP FORUM:1, NEWSLETTER:0, LIKED:0, ⟩

		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

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 trial user is equally similar to instance $\mathbf{d}_1$ and $\mathbf{d}_2$!

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

Cosine

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

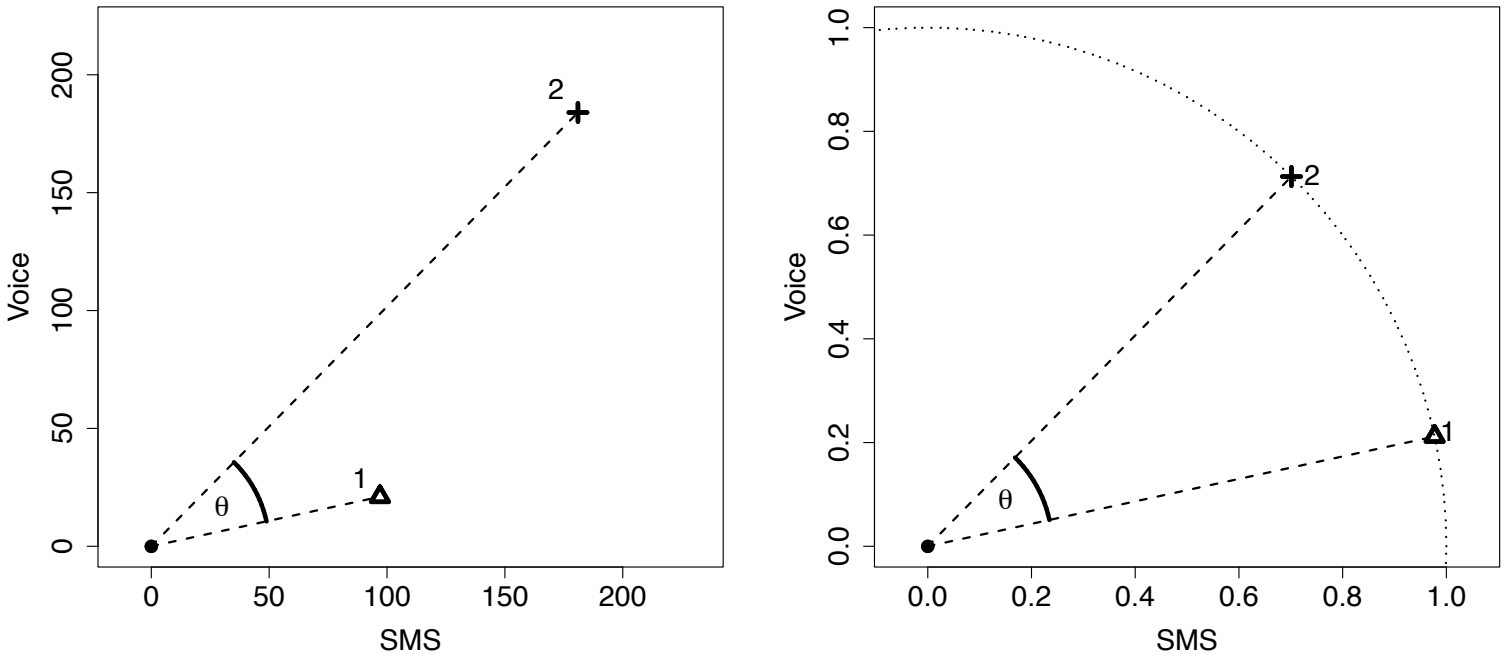
- Calculate the cosine similarity between the following two instances:

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

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

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

$$= 0.8362$$



(e) (f)

(e) 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 $\mathbf{d}_2$ $\langle \text{SMS} = 181, \text{VOICE} = 184 \rangle$; (f) shows $\mathbf{d}_1$ and $\mathbf{d}_2$ normalized to the unit circle.

- Calculate the cosine similarity between the following two instances:

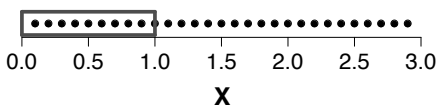
$$\mathbf{d}_1 = \langle \text{SMS} = 97, \text{VOICE} = 21 \rangle$$

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

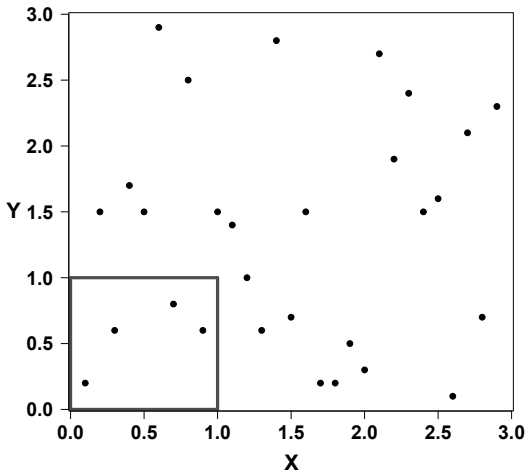
$$sim_{COSINE}(\mathbf{d}_1, \mathbf{d}_1) = \frac{(97 \times 194) + (21 \times 42)}{\sqrt{97^2 + 21^2} \times \sqrt{194^2 + 42^2}}$$

$$= 1$$

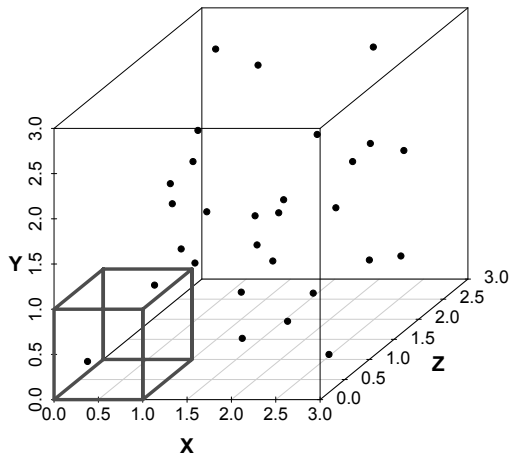
Feature Selection



(g)

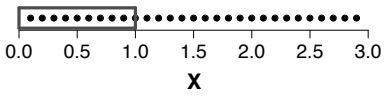


(h)

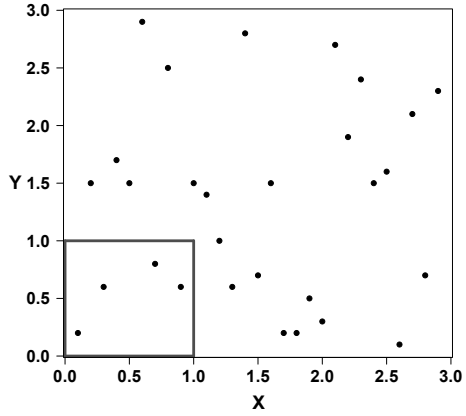


(i)

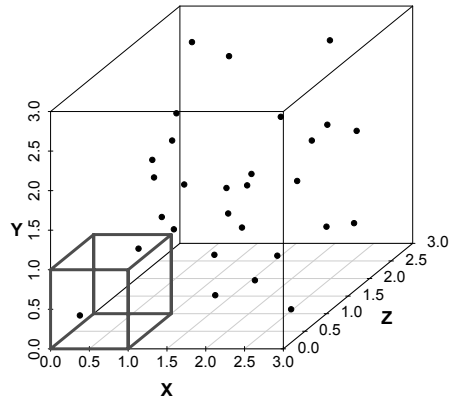
A set of scatter plots illustrating the curse of dimensionality. Across figures (g), (h) and (i) the density of the marked unit hypercubes decreases as the number of dimensions increases.



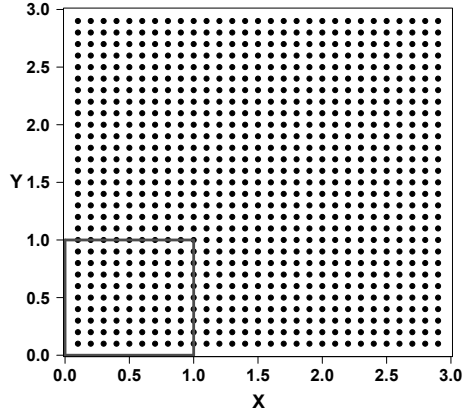
(j)



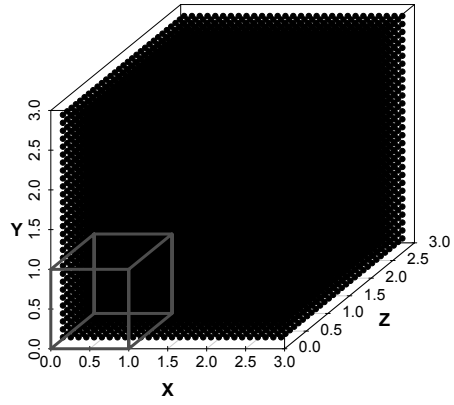
(k)



(l)



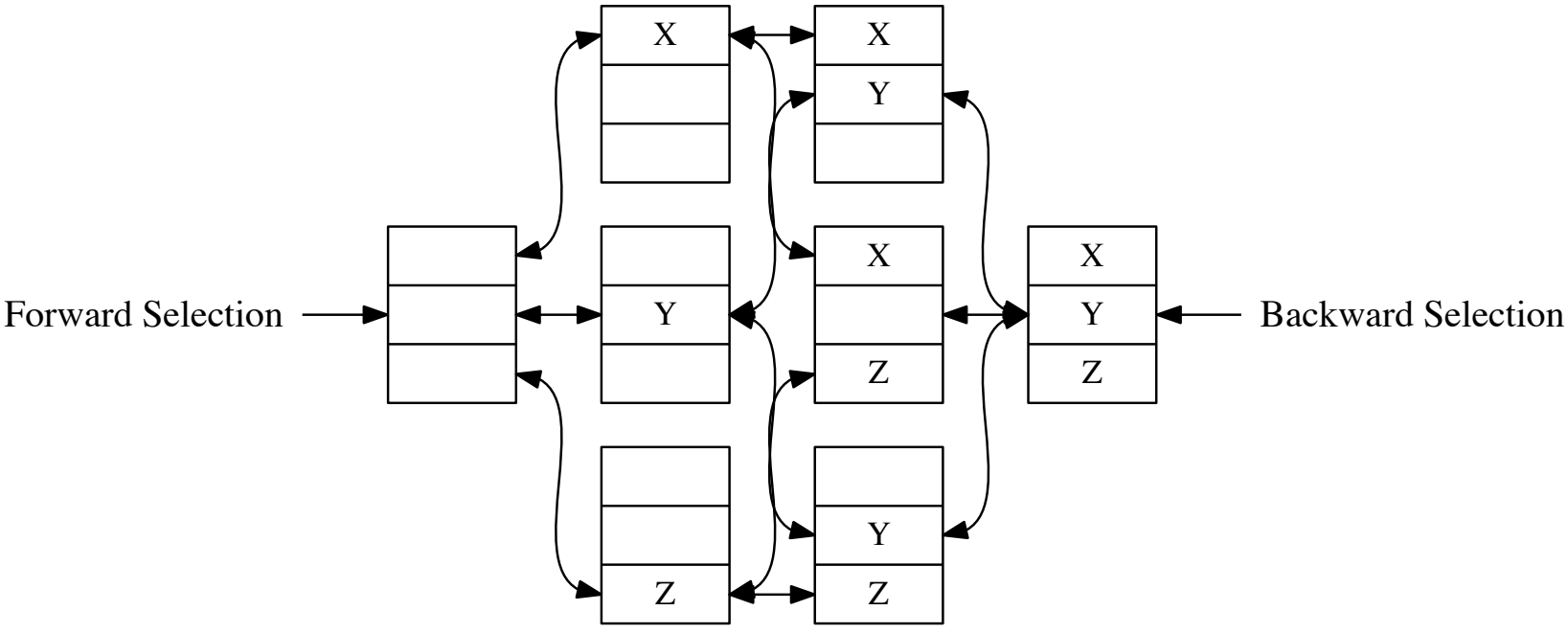
(m)



(n)

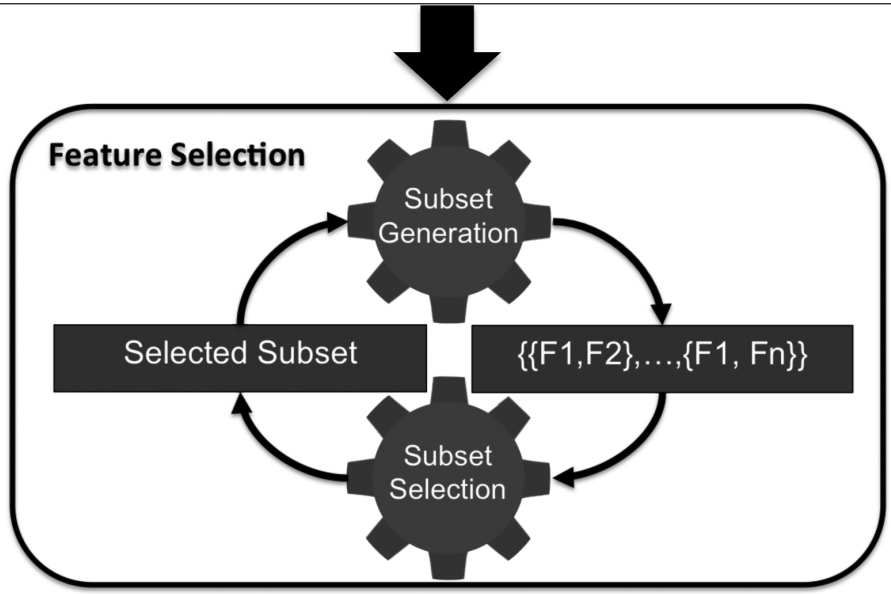
Figures (m) and (n) illustrate the cost we must incur if we wish to maintain the density of the instances in the feature space as the dimensionality of the feature space increases.

- Different classes of descriptive features:
  1. Predictive
  2. Interacting
  3. Redundant
  4. Irrelevant
- When viewed as a local search problem, feature selection is an iterative process with 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

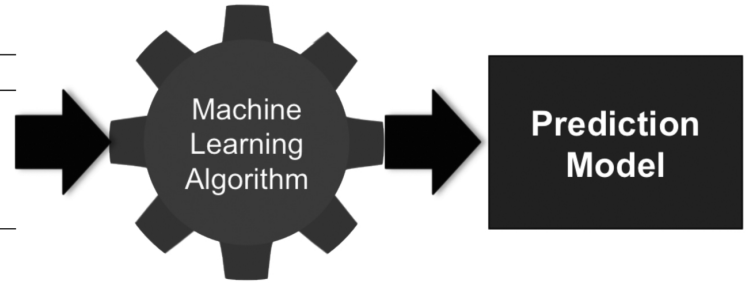


Feature Subset Space for a dataset with 3 features X, Y, Z.

ID	F1	F2	F3	...	F150	F151	...	F227	F228	...	F387	...	Fn	Target
1	-	-	-	...	-	-	...	-	-	...	-	...	-	-
2	-	-	-	...	-	-	...	-	-	...	-	...	-	-
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
K	-	-	-	...	-	-	...	-	-	...	-	...	-	-



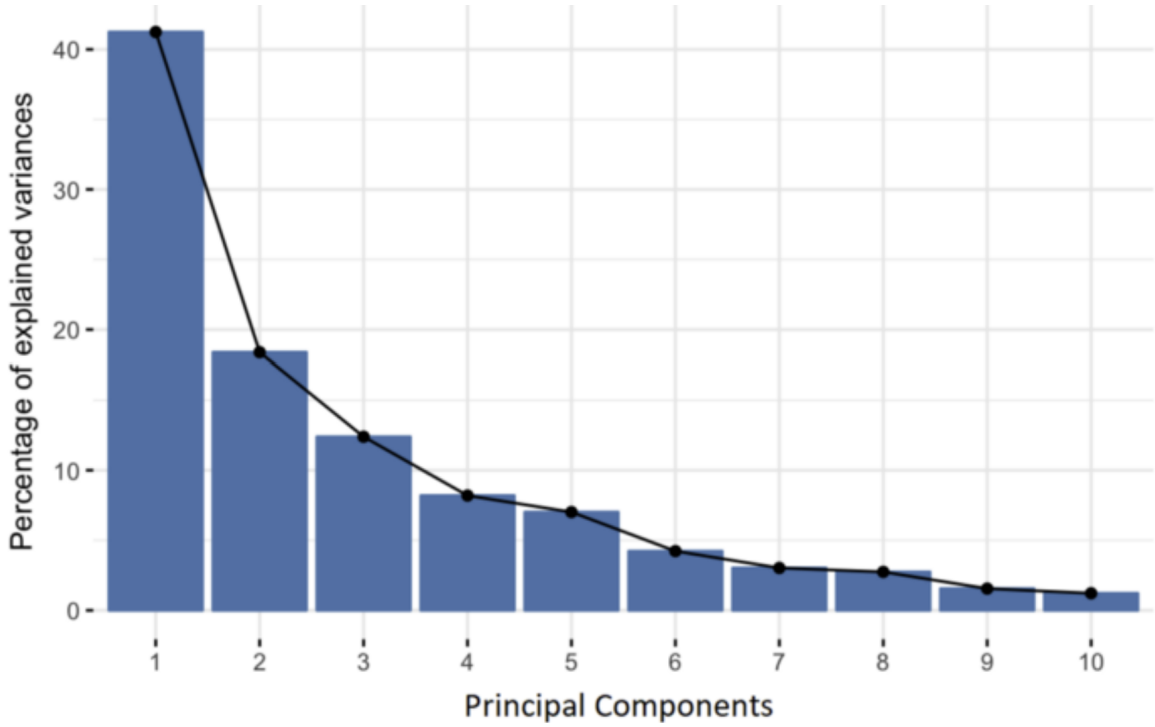
ID	F2	F150	F227	F387	Target
1	-	-	-	-	-
2	-	-	-	-	-
...	...	...	...	...	...
K	-	-	-	-	-



The process of model induction with feature selection.

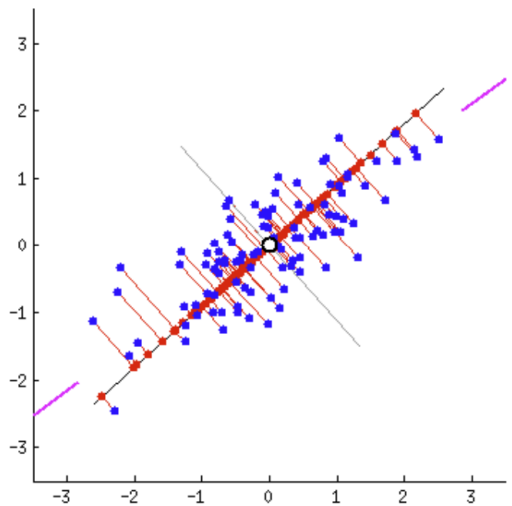
What Is Principal Component Analysis?

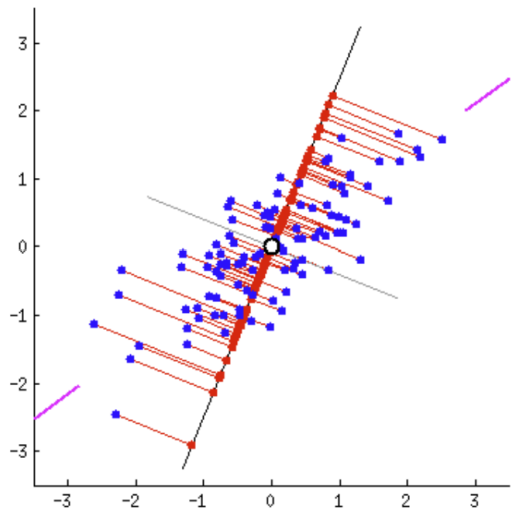
- PCA is a dimensionality reduction method.
- Transforms a large set of variables into a smaller one while retaining most of the information.
- Balances the trade-off between accuracy and simplicity.
- Reduces the number of variables while preserving as much information as possible.



Principal Components

- New variables created as linear combinations of the original variables.
- Principal components are uncorrelated and capture maximum variance.
- First principal component captures the most variance, followed by the second, and so on.
- Principal components are less interpretable but represent directions of maximum variance.



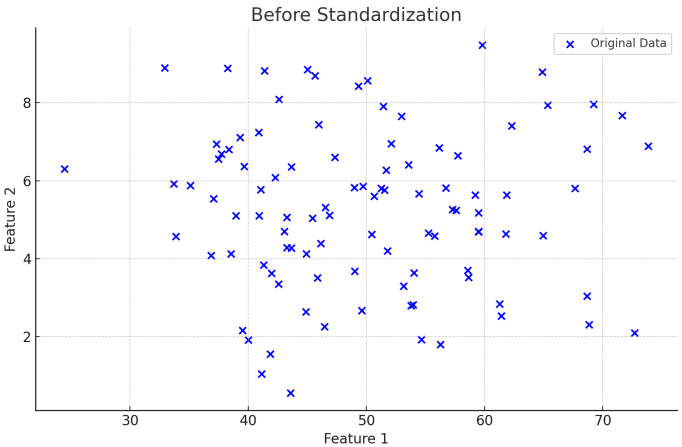


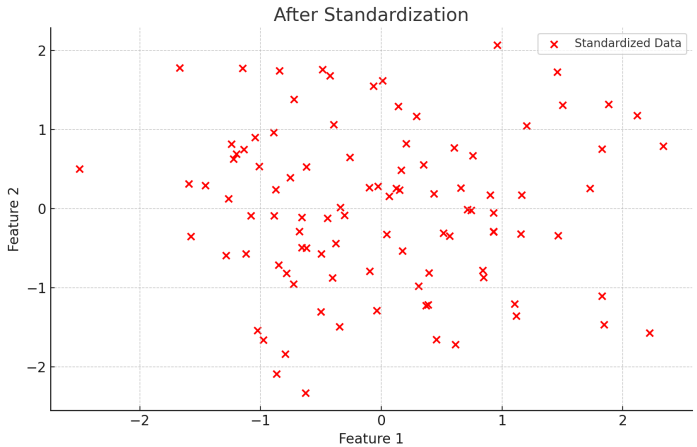
Step 1: Standardization

- Standardizes the range of continuous variables.
- Ensures each variable contributes equally to the analysis.
- Prevents variables with larger ranges from dominating the analysis.
- Formula:

$$z_{ij} = \frac{x_{ij} - \bar{x}_j}{s_j}$$

Step 1: Standardization (cont.)





- After standardization, each feature should have:
  - Mean = 0
  - Standard Deviation = 1
- Calculated means: [0.0, 0.0]
- Calculated standard deviations: [1.0, 1.0]

Step 2: Covariance Matrix Computation

- Understands how variables vary with respect to each other.
- Identifies correlations between variables.
- Covariance matrix formula:

$$\mathbf{C} = \frac{1}{n-1} \sum_{i=1}^{n} (\mathbf{z}_i - \bar{\mathbf{z}})(\mathbf{z}_i - \bar{\mathbf{z}})^T$$

- Symmetric matrix with variances on the diagonal.

$$\begin{bmatrix} Cov(x,x) & Cov(x,y) & Cov(x,z) \\ Cov(y,x) & Cov(y,y) & Cov(y,z) \\ Cov(z,x) & Cov(z,y) & Cov(z,z) \end{bmatrix}$$

Covariance Matrix for 3-Dimensional Data.

- Example covariance matrix:

$$\mathbf{C} = \begin{pmatrix} 5 & 2 \\ 2 & 2 \end{pmatrix}$$

Step 3: Compute Eigenvalues and Eigenvectors

- Compute eigenvalues and eigenvectors from the covariance matrix.
- Eigenvectors determine the principal components.
- Eigenvalues indicate the magnitude of variance in the direction of the eigenvector.

$$\mathbf{Cv} = \lambda \mathbf{v}$$

- Principal components are ranked by eigenvalues in descending order.

Step 4: Form the Feature Vector

- Select the top k eigenvectors to form the feature vector.
- The feature vector reduces dimensionality while retaining most of the information.

$$\mathbf{V}_k = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k]$$

- Simplifies the data by focusing on the most important components.

- Consider a covariance matrix:

$$\mathbf{C} = \begin{pmatrix} 3 & 1 \\ 1 & 2 \end{pmatrix}$$

- Eigenvalues (λ) and corresponding eigenvectors ($\mathbf{v}$):

$$\lambda_1 = 3.618, \qquad \mathbf{v}_1 = \begin{pmatrix} 0.850 \\ 0.526 \end{pmatrix}$$

$$\lambda_2 = 1.382, \qquad \mathbf{v}_2 = \begin{pmatrix} -0.526 \\ 0.850 \end{pmatrix}$$

- Ranking by eigenvalues:
  - $\lambda_1 = 3.618$, $\mathbf{v}_1 = \begin{pmatrix} 0.850 \\ 0.526 \end{pmatrix}$
  - $\lambda_2 = 1.382$, $\mathbf{v}_2 = \begin{pmatrix} -0.526 \\ 0.850 \end{pmatrix}$
- Principal components:
  - PC1: Eigenvector $\mathbf{v}_1$ with eigenvalue λ_1
  - PC2: Eigenvector $\mathbf{v}_2$ with eigenvalue λ_2

Step 5: Recast the Data Along Principal Components

- Transform the original dataset using the feature vector.

$$\mathbf{Z}_{\text{new}} = \mathbf{Z}\mathbf{V}_k$$

- The transformed data is now in terms of the principal components.
- This new representation helps in visualizing and analyzing the data effectively.

Transforming the Dataset Using the Feature Vector

- Original dataset ($\mathbf{Z}$):

$$\mathbf{Z} = \begin{pmatrix} 2 & 3 \\ 3 & 4 \\ 4 & 5 \end{pmatrix}$$

- Feature vector ($\mathbf{V}_k$) (top principal component):

$$\mathbf{V}_k = \begin{pmatrix} 0.707 \\ 0.707 \end{pmatrix}$$

- Transform the original dataset using the feature vector:

$$\mathbf{Z}_{\text{new}} = \mathbf{Z}\mathbf{V}_k = \begin{pmatrix} 2 & 3 \\ 3 & 4 \\ 4 & 5 \end{pmatrix} \begin{pmatrix} 0.707 \\ 0.707 \end{pmatrix} = \begin{pmatrix} 3.535 \\ 4.949 \\ 6.364 \end{pmatrix}$$

- The transformed data is now in terms of the principal component.
- This new representation helps in visualizing and analyzing the data effectively.