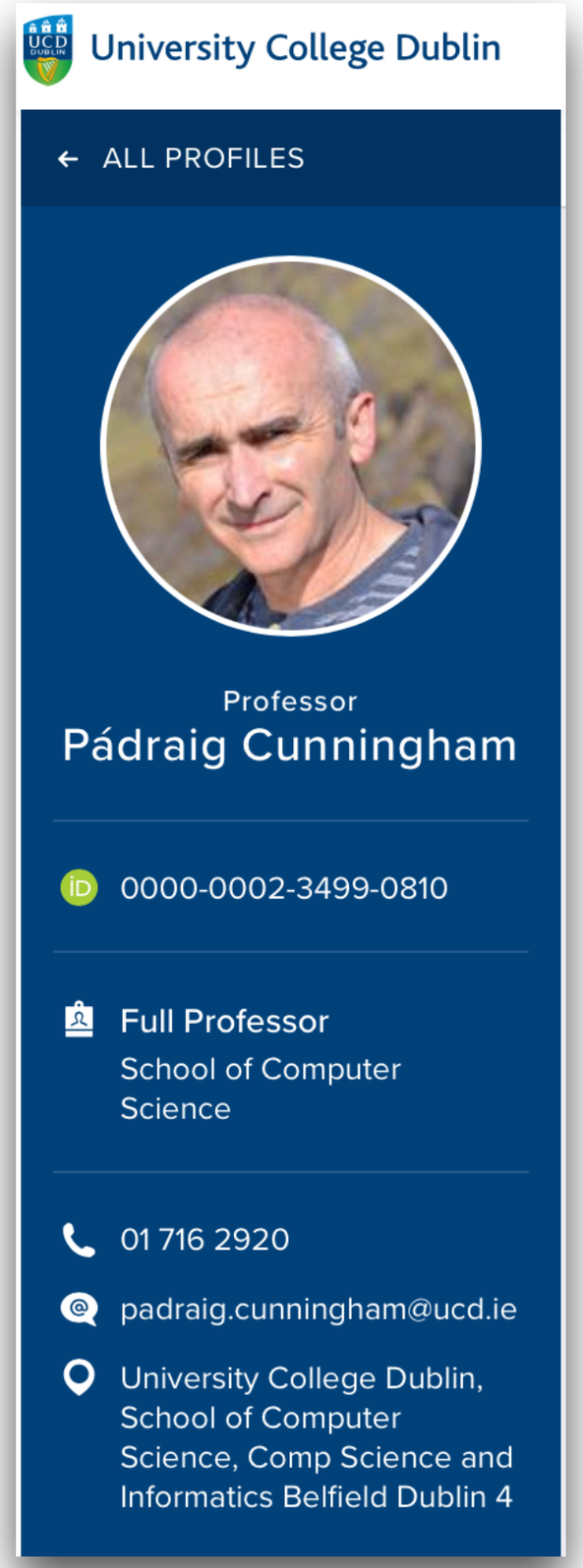
Time-Series Classification

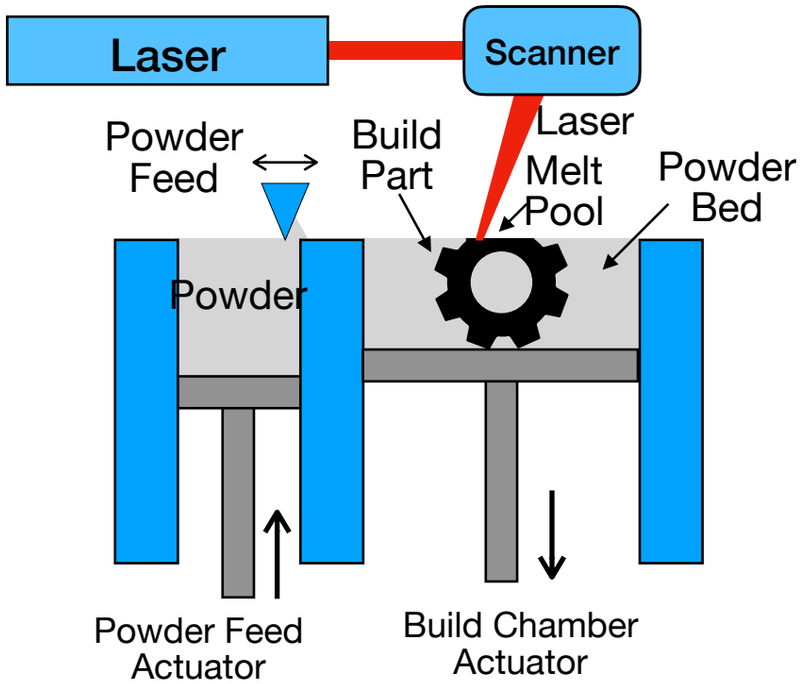
I-Form
Advanced Manufacturing Research Centre

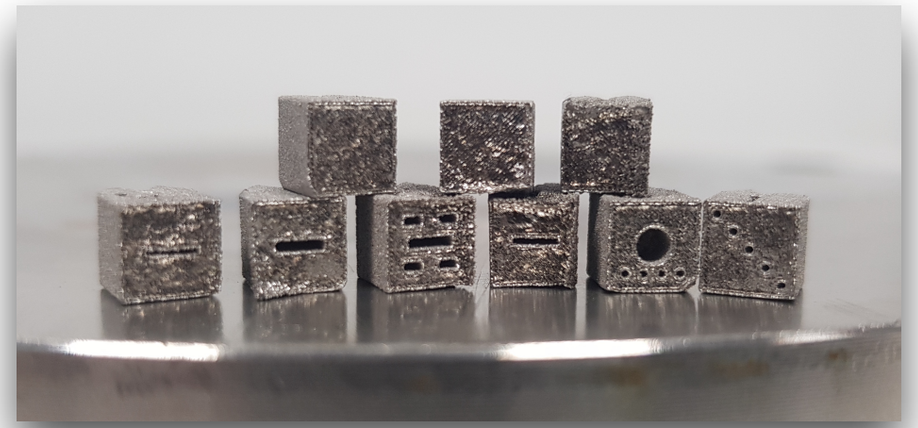
- Manufacturing Context
- Classification in ML
  - Challenges with time-series data
- The *k*-NN solution
  - Euclidean distance
  - Dynamic Time Warping
- Evaluation
- Exercise
- Model Selection



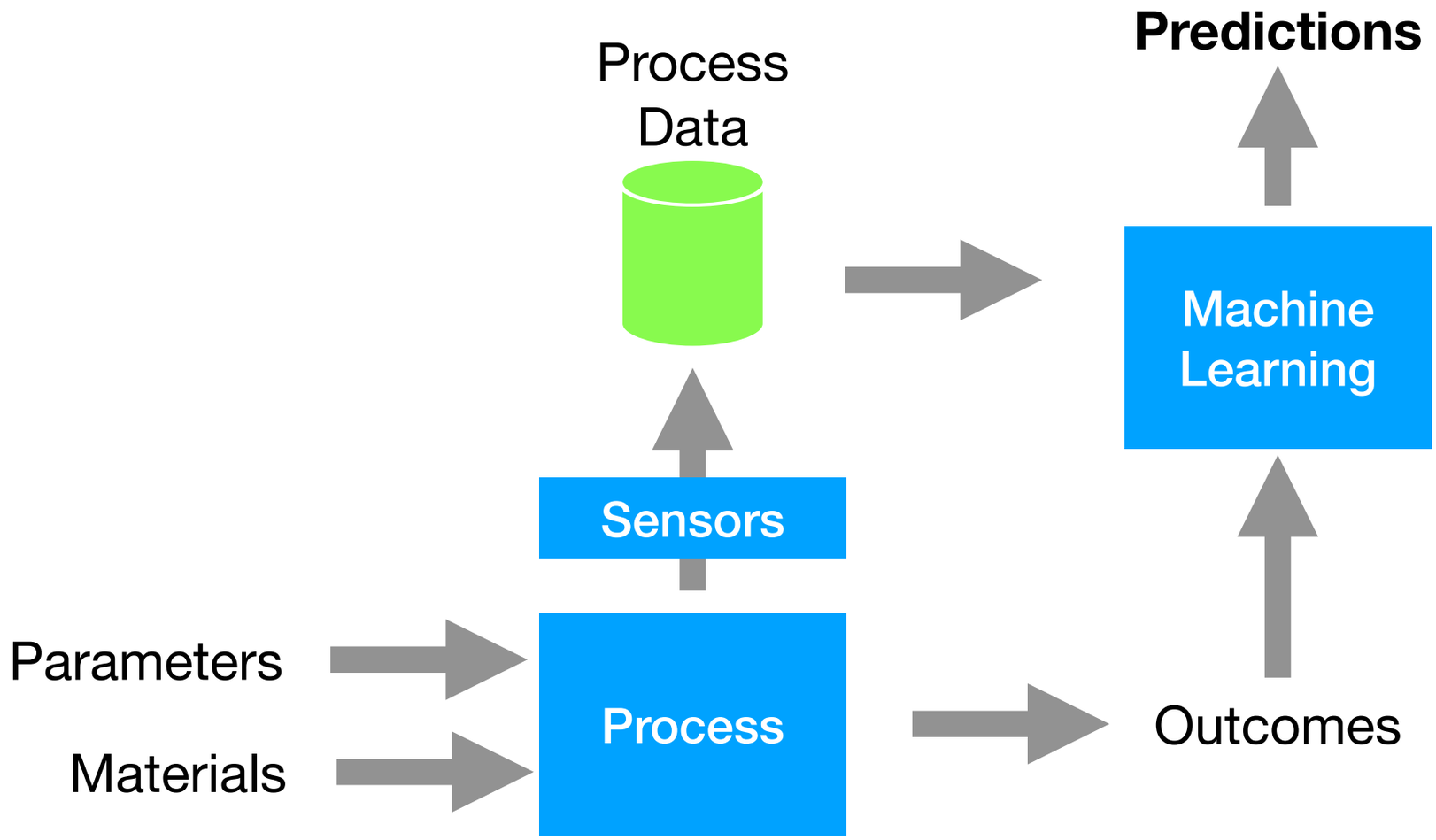
Science Foundation Ireland sfi For what's next

Metal 3D Printing



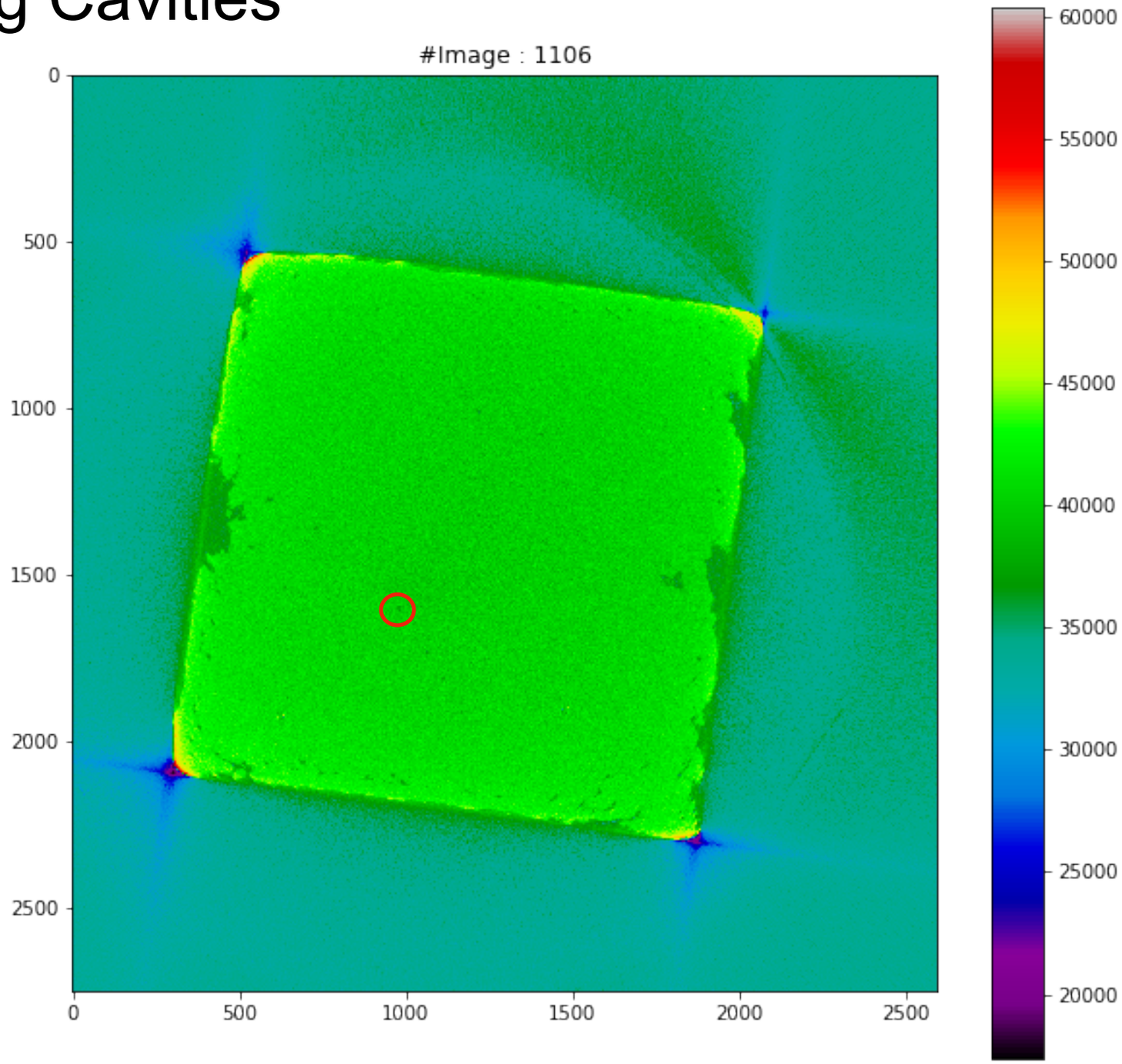


ML & Process Monitoring



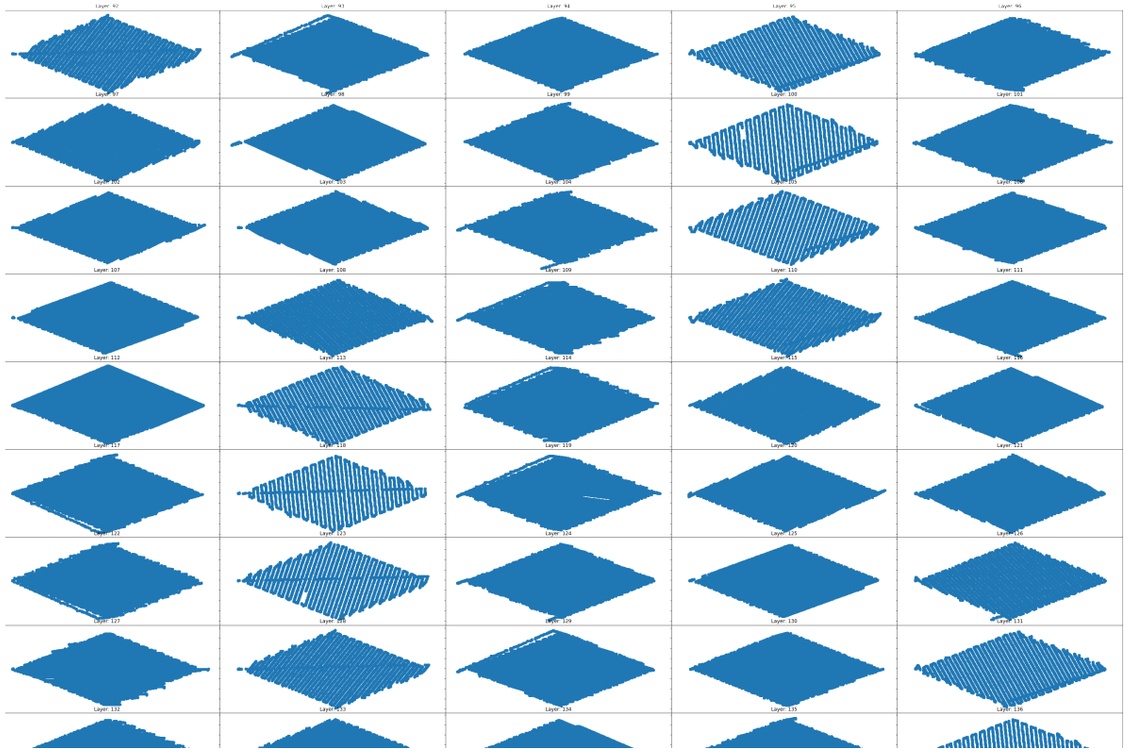
Example: 3D Printing

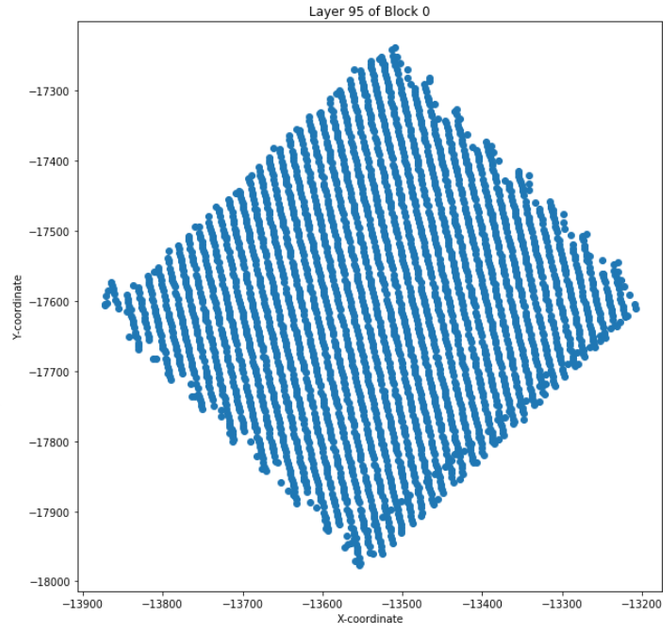
- Predicting Cavities
  - CT Scan

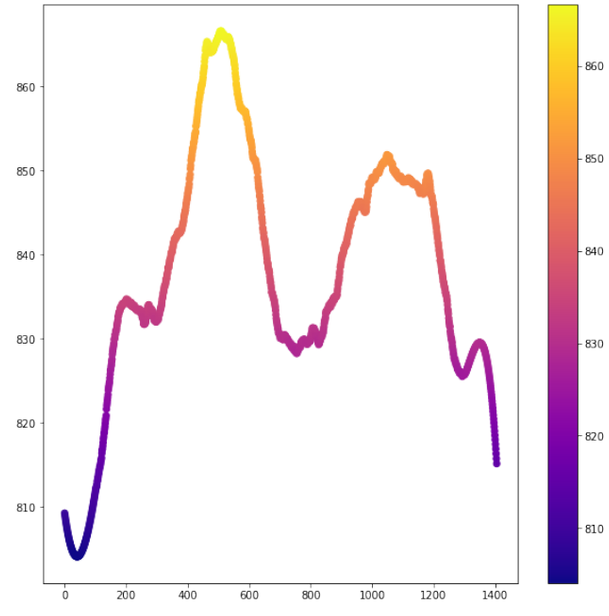


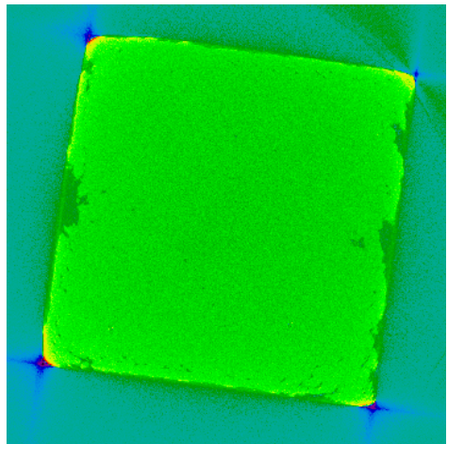
Example: 3D printing

- Learn temperature profiles associated with anomalies







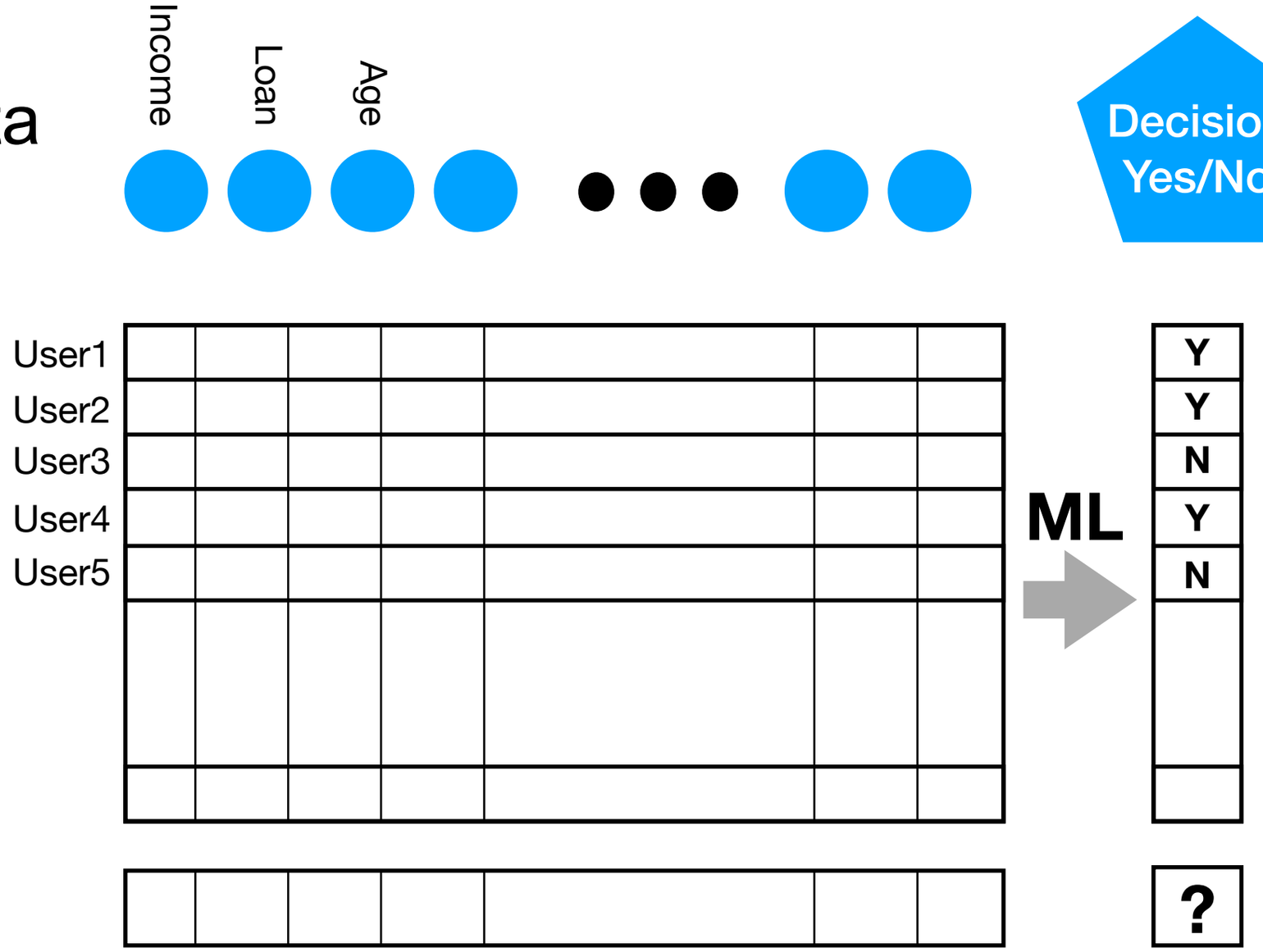


Time-Series Classification

- Manufacturing Context
- Classification in ML
  - Challenges with time-series data
- The *k*-NN solution
  - Euclidean distance
  - Dynamic Time Warping
- Evaluation
- Exercise
- Model Selection

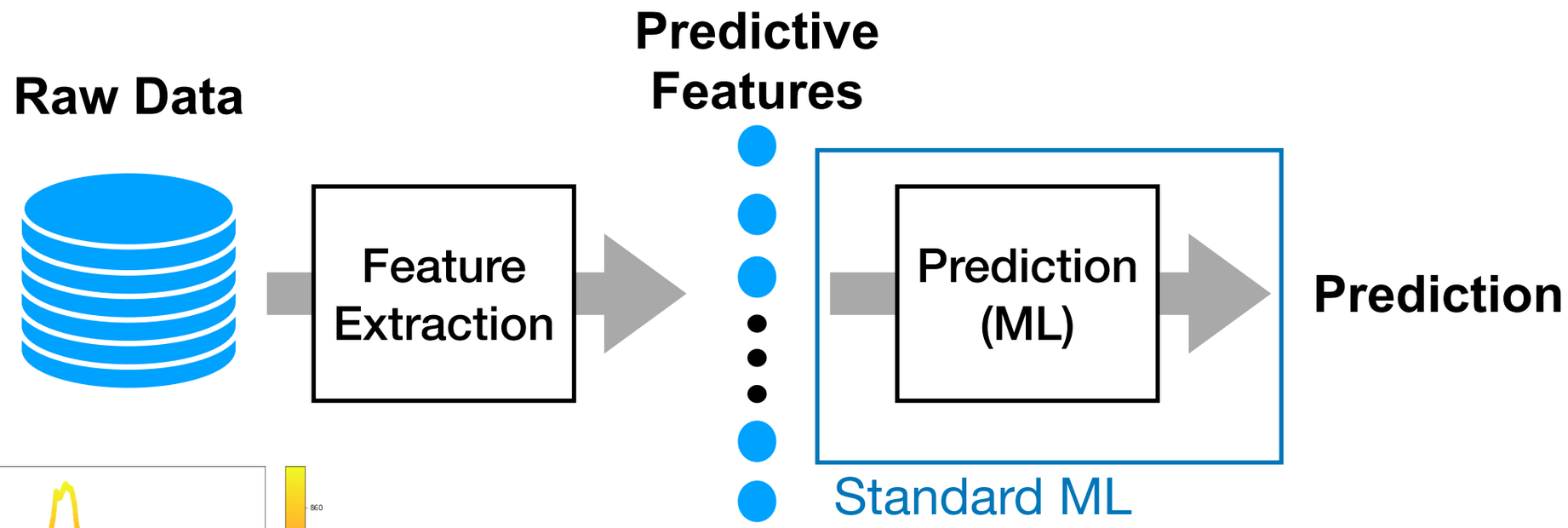
Learning from Data

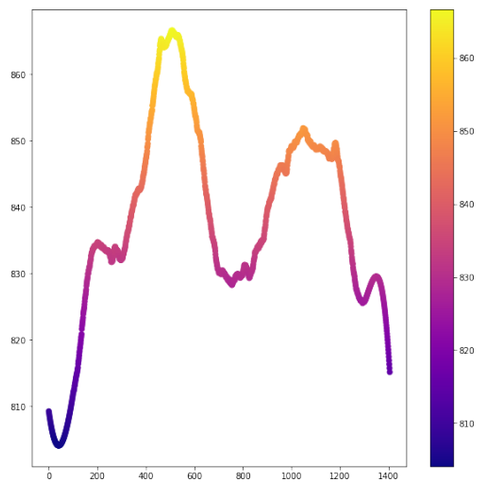
- Table of historic data
- Each row
  - description of case
  - decision (Yes/No)



- Predict decision for new case

Supervised ML (aka Predictive Analytics)





Time-series data:
Information lost in converting to feature vector

Time-Series Classification

- Manufacturing Context
- Classification in ML
  - Challenges with time-series data
- The *k*-NN solution
  - Euclidean distance
  - Dynamic Time Warping
- Evaluation
- Exercise
- Model Selection

Eager v Lazy Classifiers

- Eager Learning Classification Strategy
  - Classifier builds a full model during an initial training phase, to use later when new query examples arrive.
  - More offline setup work, less work at run-time.
  - Generalise before seeing the query example.
- Lazy Learning Classification Strategy
  - Classifier keeps all the training examples for later use.
  - Little work is done offline, wait for new query examples.
  - Focus on the local space around the examples.
- Distance-based Models: Many learning algorithms are based on generalising from training data to unseen data by exploiting the distances (or similarities) between the two.

Example: Athlete Selection

- Training set of performance ratings for 20 college athletes, where each athlete is described by 2 continuous features: *speed*, *agility*.
- Each athlete has a target class label indicating whether they were selected for the university athletics team: 'Yes' or 'No'.

Athlete	*Speed*	*Agility*	*Selected*
x1	2.50	6.00	No
x2	3.75	8.00	No
x3	2.25	5.50	No
x4	3.25	8.25	No
x5	2.75	7.50	No
x6	4.50	5.00	No
x7	3.50	5.25	No
x8	3.00	3.25	No
x9	4.00	4.00	No
x10	4.25	3.75	No

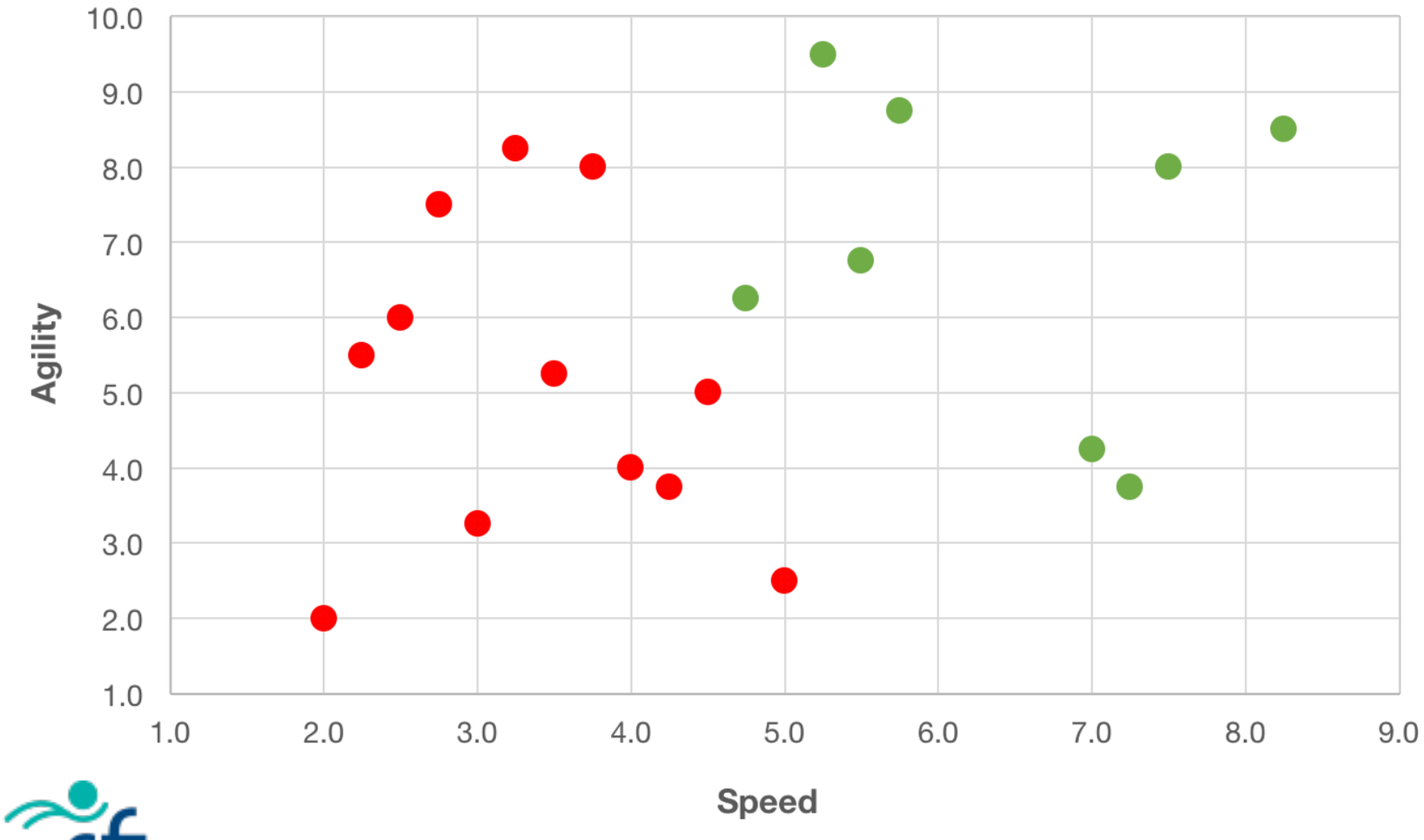
Athlete	*Speed*	*Agility*	*Selected*
x11	2.00	2.00	No
x12	5.00	2.50	No
x13	8.25	8.50	Yes
x14	5.75	8.75	Yes
x15	4.75	6.25	Yes
x16	5.50	6.75	Yes
x17	5.25	9.50	Yes
x18	7.00	4.25	Yes
x19	7.50	8.00	Yes
x20	7.25	3.75	Yes

Q. Will a new athlete **q** be selected: 'Yes' or 'No'?

Athlete	*Speed*	*Agility*	*Selected*
q	3.00	8.00	???

Feature Spaces

- A feature space is a *D*-dimensional coordinate space used to represent the input examples for a given problem, with one coordinate for each descriptive feature.
- **Example:** Use a feature space to visually position the 20 athletes in a 2-dimensional coordinate space (i.e. *agility* versus *speed*):

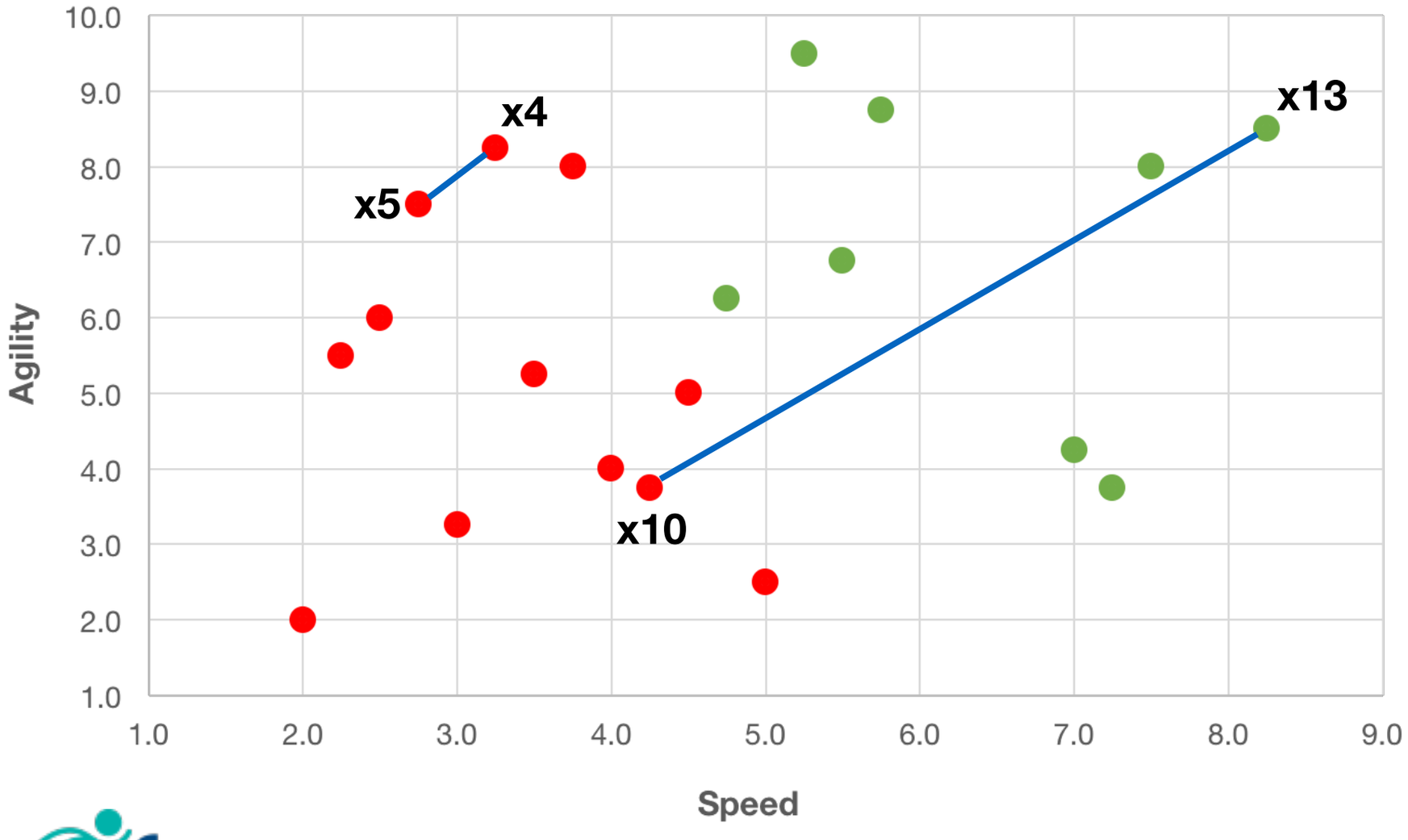


Training set of 20 examples (athletes)

Each example described by 2 feature values: *agility* & *speed*

Measuring Distance

- Measuring the distance (or similarity) between two examples is fundamental to many ML algorithms.
- Many measures can be used to calculate distance. There is no "best" distance measure. The choice is highly problem-dependent.



Examples *x4* and *x5* have a low distance (high similarity)

Examples *x10* and *x13* have a high distance (low similarity)

Measuring Distance

- Euclidean distance: Most common measure used to quantify distance between two examples with numeric features.
- Given by the "straight line" distance between two points in a Euclidean coordinate space - e.g. a feature space.
- Calculated as the square root of sum of squared differences for each feature *f* representing a pair of examples.
- The output is a real value ≥ 0, where a larger value indicates two examples are more distant (i.e. less similar to one another).

Input:
2 examples
p and **q**

$$\mathrm{ED}(\mathbf{p}, \mathbf{q}) = \sqrt{\sum_{f \in F} (q_f - p_f)^2}$$

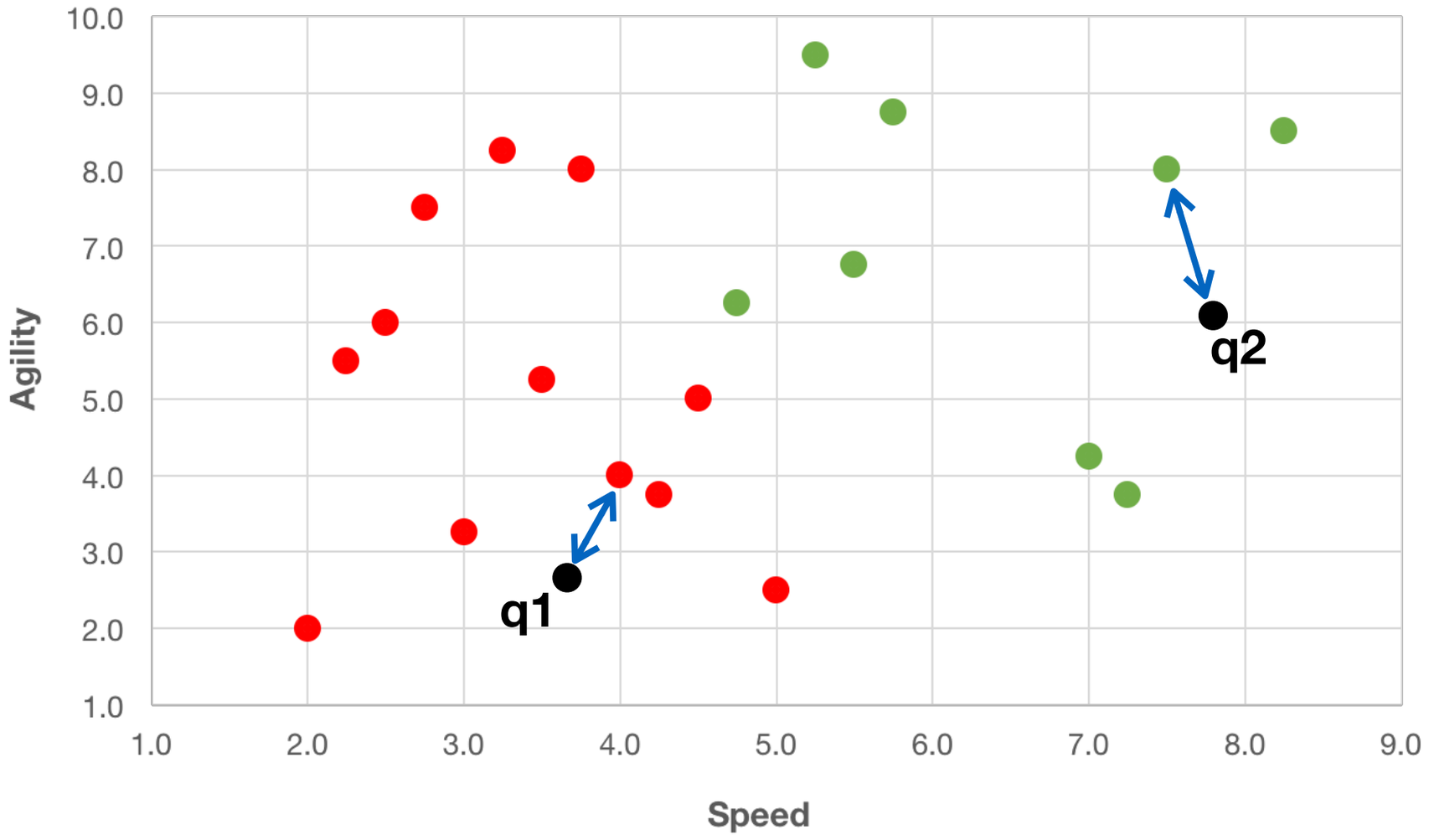
Calculate square of the difference between the examples on feature *f*

For each feature *f* in the full set of features *F*

Nearest Neighbour Classifier

Lazy learning approach: Do not build a model for the data. Identify most similar previous example(s) from the training set for which a label has already been assigned, using some distance function.

Nearest neighbour rule (1NN): For a new query input **q**, find a single labelled example **x** closest to **q**, and assign **q** the same label as **x**.



Assign **q1** to 'No' class

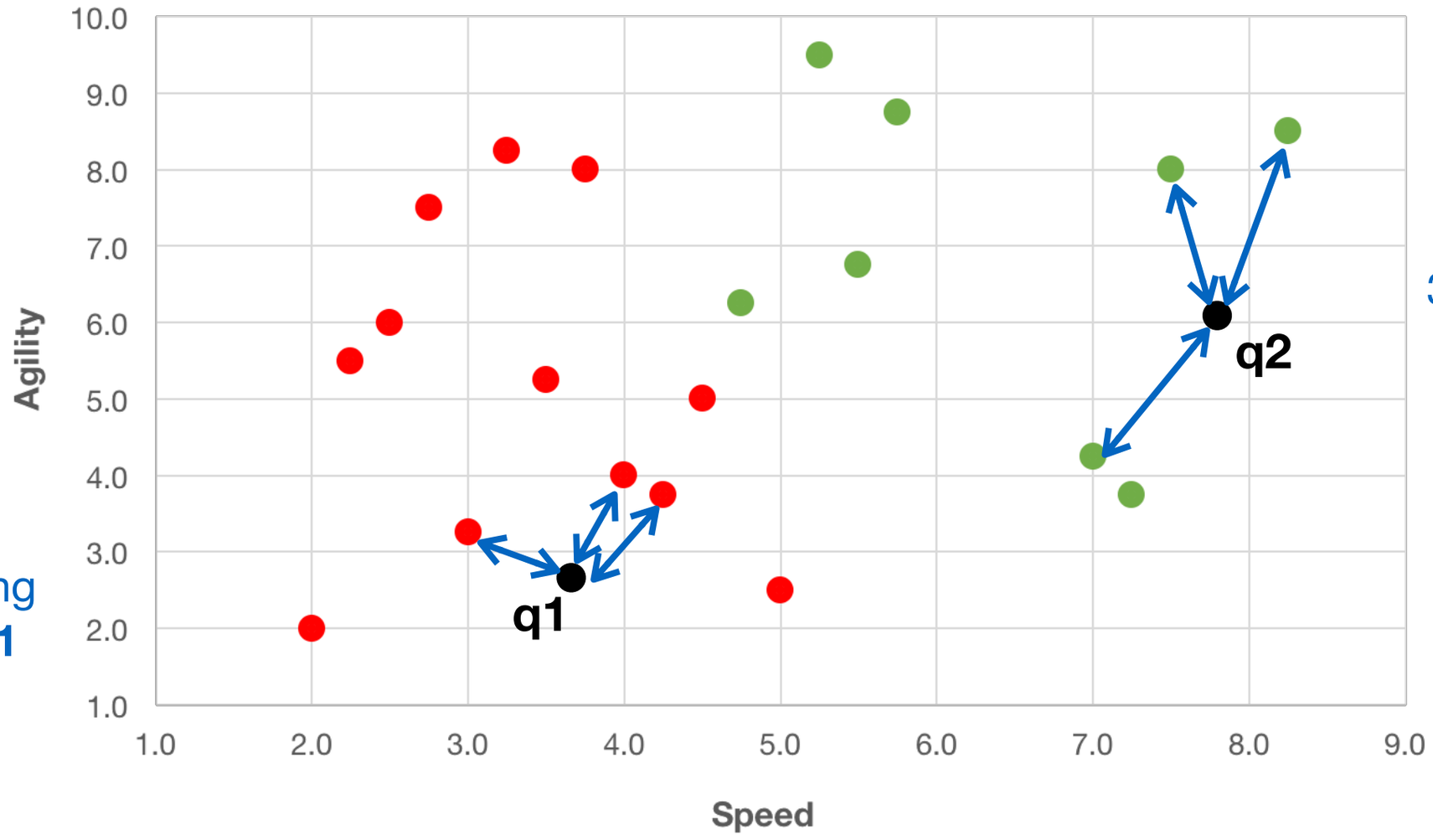
Assign **q2** to 'Yes' class

k-Nearest Neighbour Classifier

k-Nearest neighbours (kNN): The NN approach naturally generalises to the case where we use *k* nearest neighbours from the training set to assign a label to a new query input.

Example: For new query inputs, calculate distance to all training examples. Find *k=3* nearest examples (i.e. with smallest distances).

3 nearest training examples to **q1**

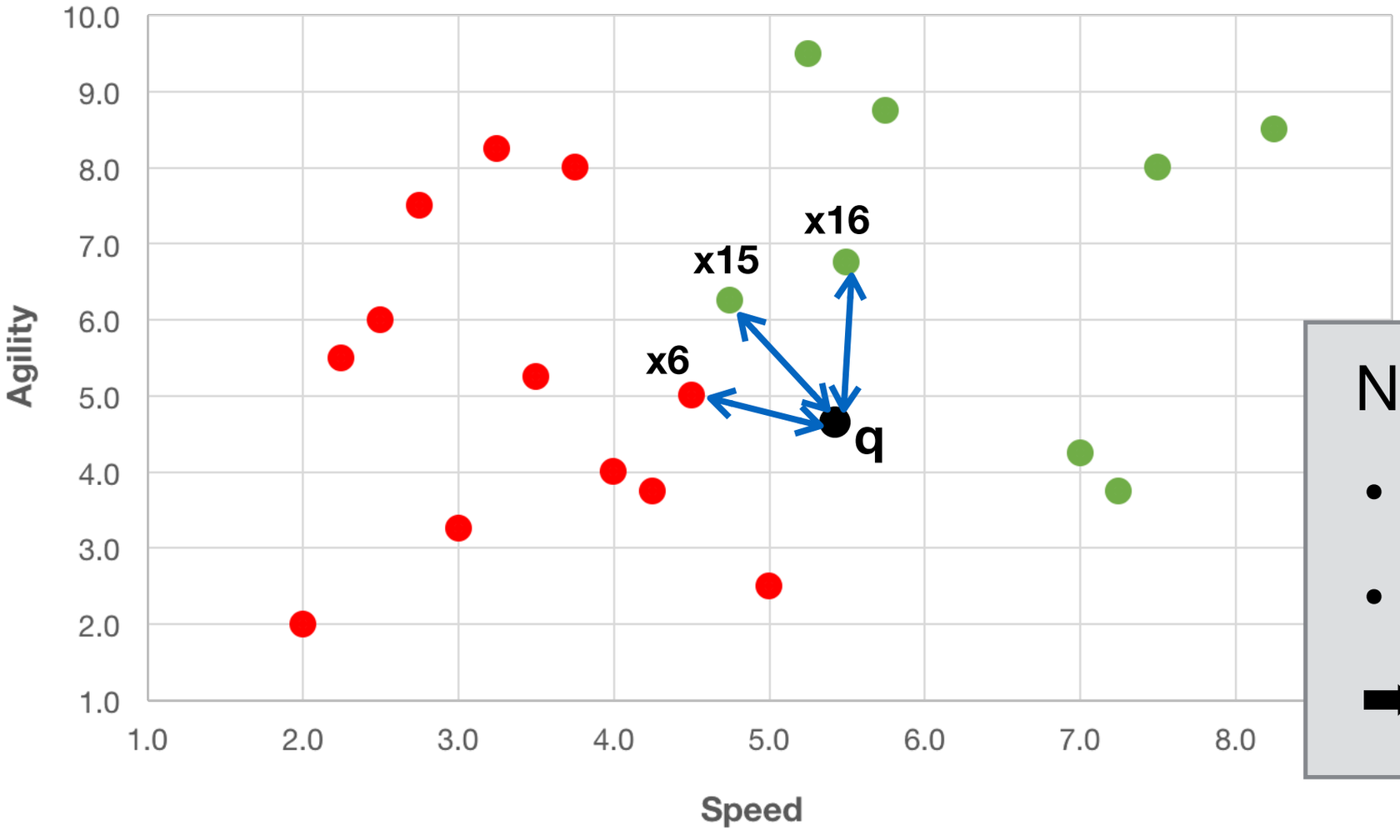


3 nearest training examples to **q2**

k-Nearest Neighbour Classifier

Majority voting: The decision on a label for a new query example is decided based on the "votes" of its *k* nearest neighbours. The label for the query is the majority label of its neighbours.

Example: Measure distance from **q** to all training examples.
Find the *k=3* nearest examples, and use their labels as votes.



Neighbour counts

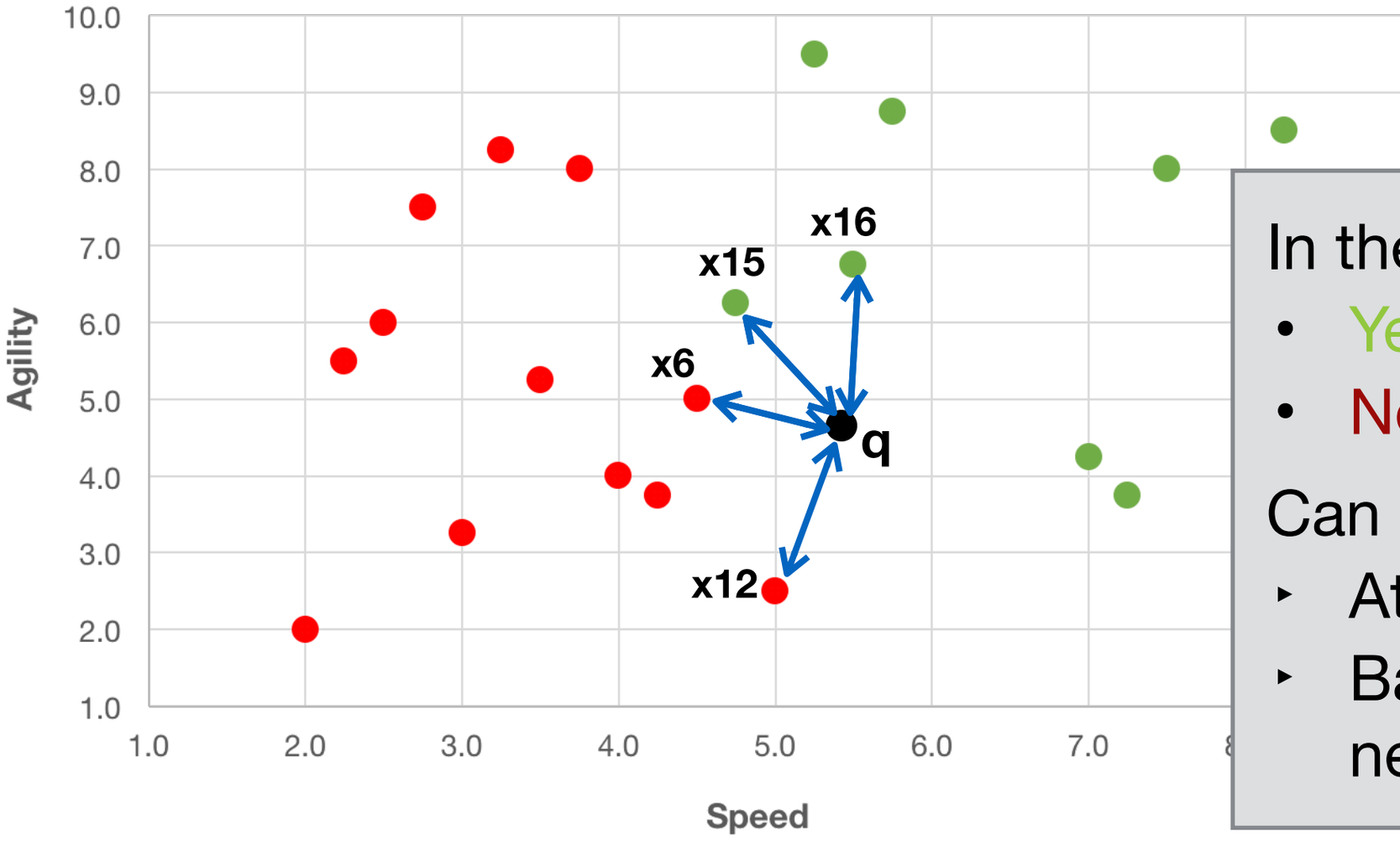
- Yes = 2 votes
- No = 1 vote

➡ Majority says Yes!

k-Nearest Neighbour Classifier

Majority voting: The decision on a label for a new query example is decided based on the "votes" of its *k* nearest neighbours. The label for the query is the majority label of its neighbours.

Example: Measure distance from **q** to all training examples.
Find the *k=4* nearest examples, and use their labels as votes.



In the case that…

- Yes = 2 votes
- No = 2 votes

Can break ties…

- At random
- Based on sum of neighbour distances

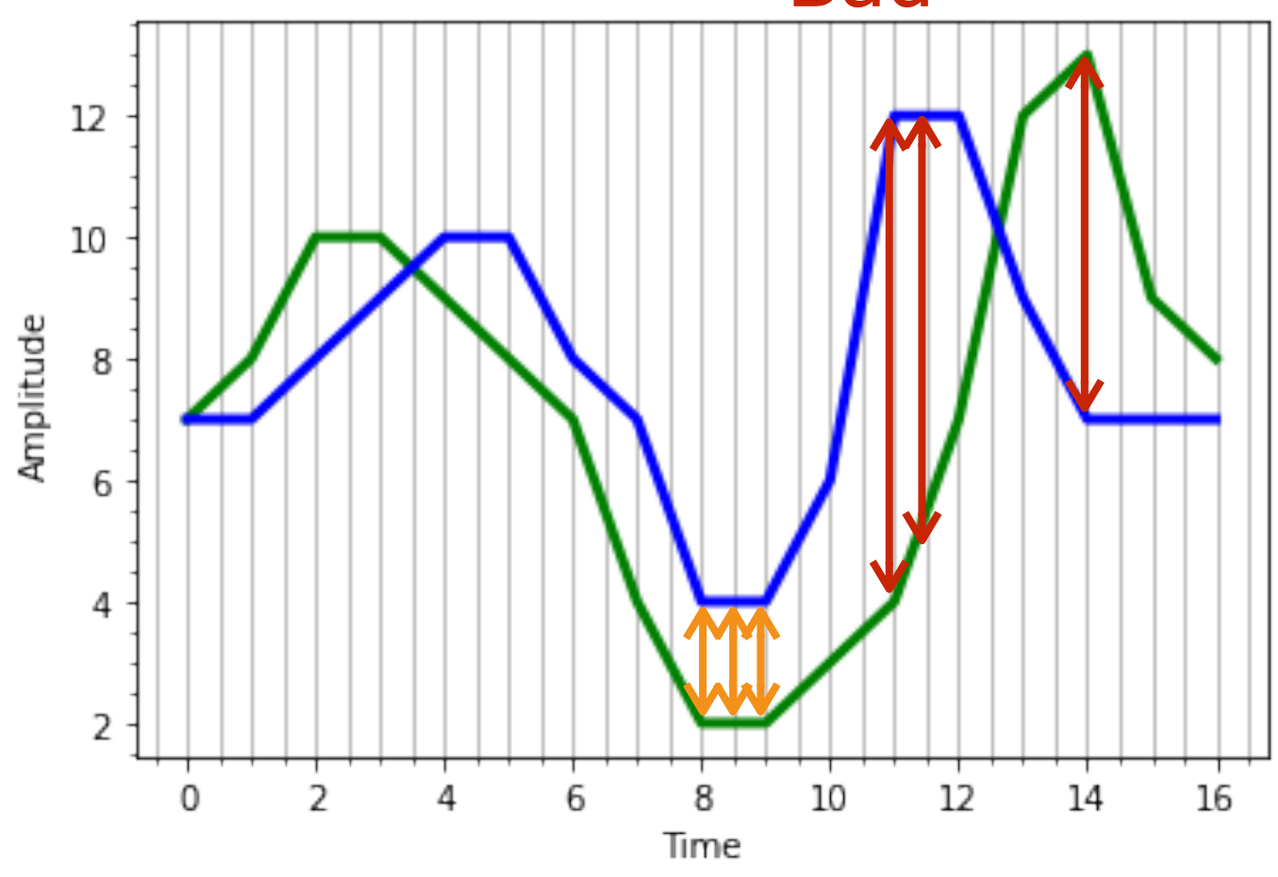
Time-Series Classification

- Manufacturing Context
- Classification in ML
  - Challenges with time-series data
- The *k*-NN solution
  - Euclidean distance
  - Dynamic Time Warping
- Evaluation
- Exercise
- Model Selection

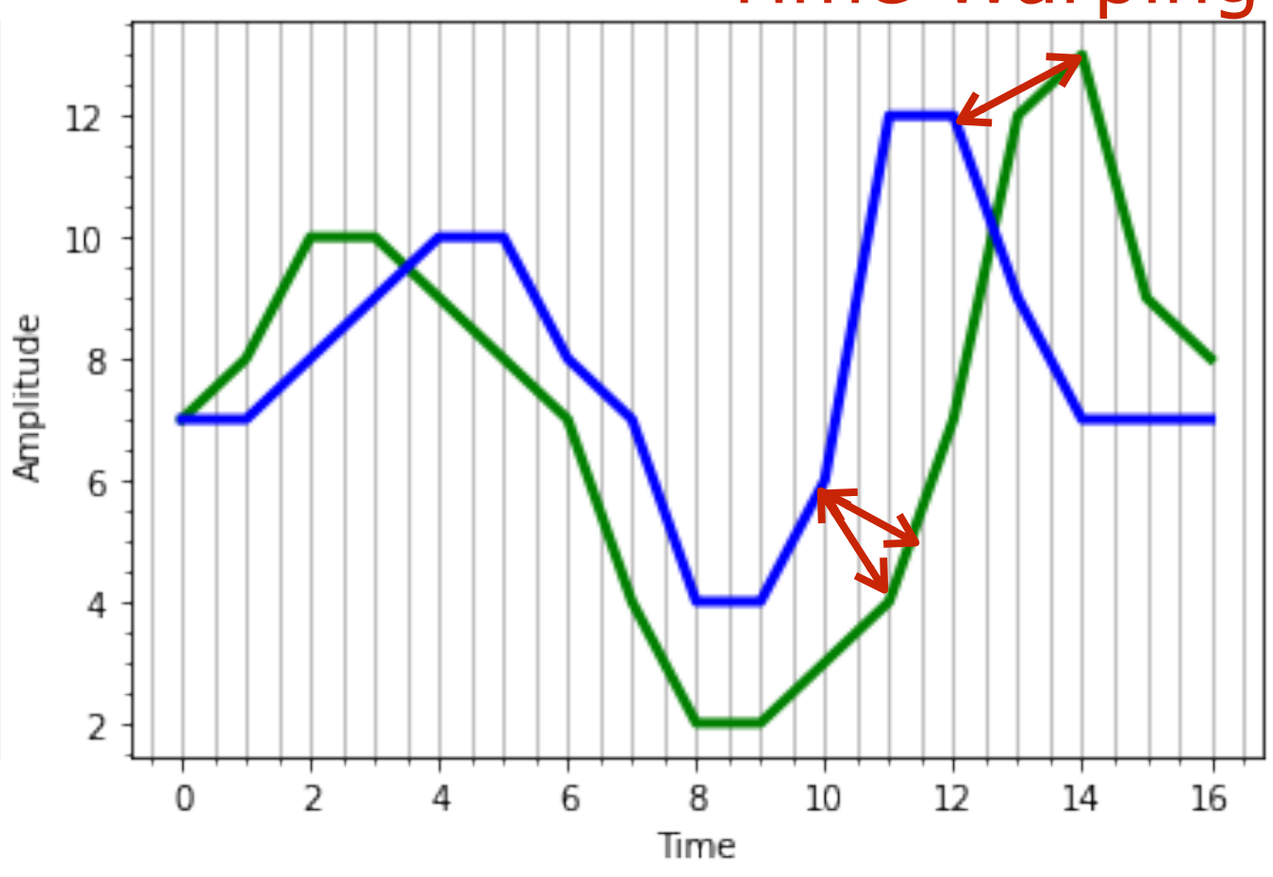
Time-Series as Feature Vectors

- The problem with Euclidean distance:
  - Time-series length 17 ⇒ 17D space

Bad



Time Warping

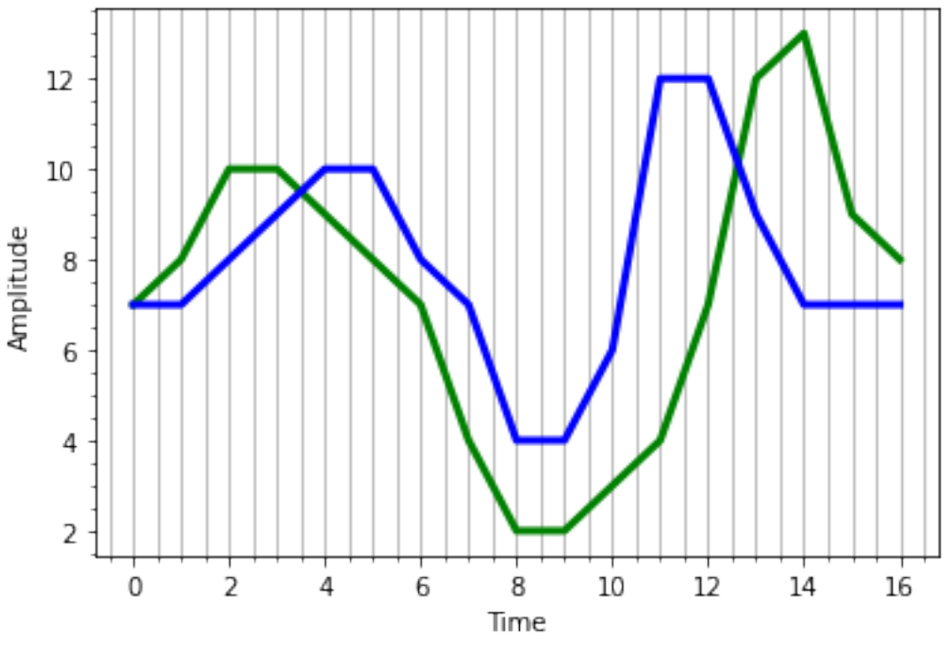


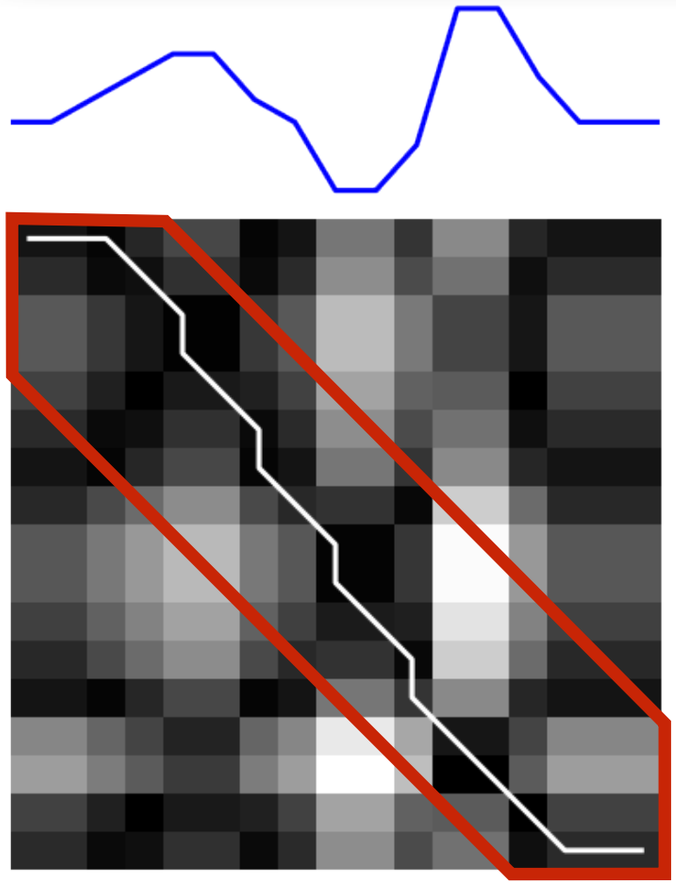
Good

Dynamic Time Warping

- Allow each point to ‘map’ to any point in the other series
  - Or within a window
- n x n matrix - find optimum mapping
- $O(n^2)$
- Euclidean distance is O(n)

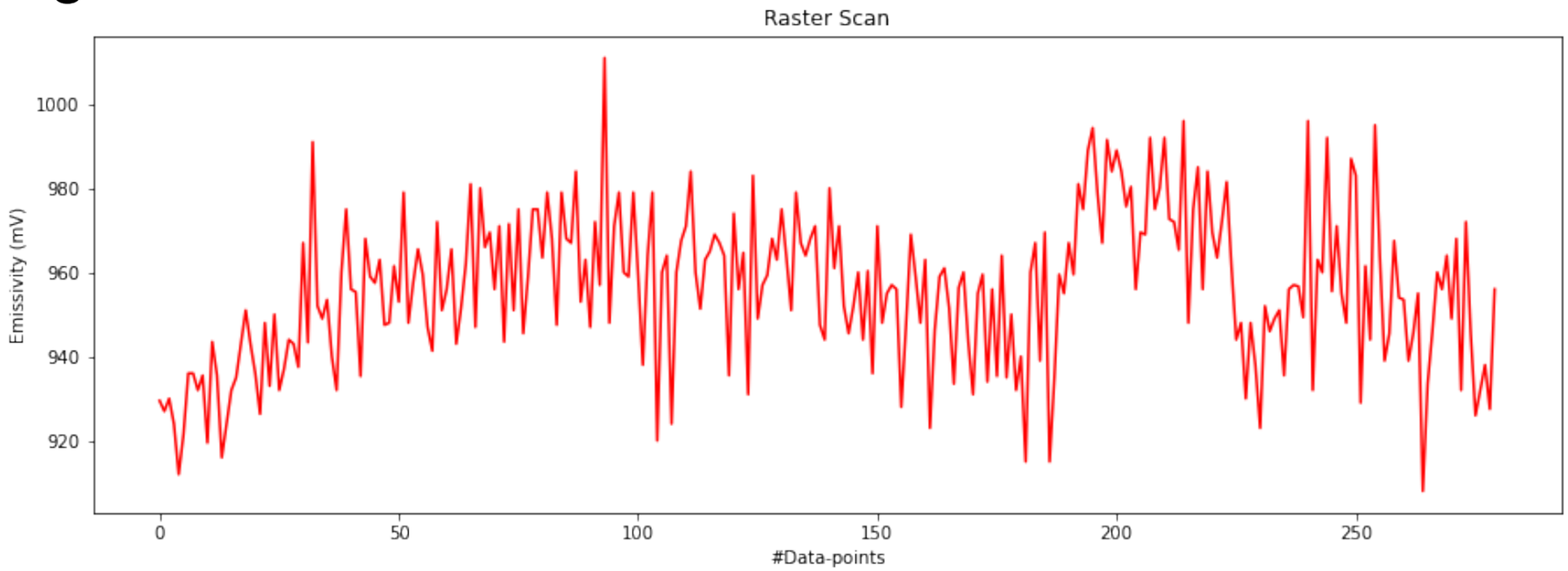
Aside:
k-NN (and perhaps SVMs) will work with any data format if a similarity/distance measure can be defined.



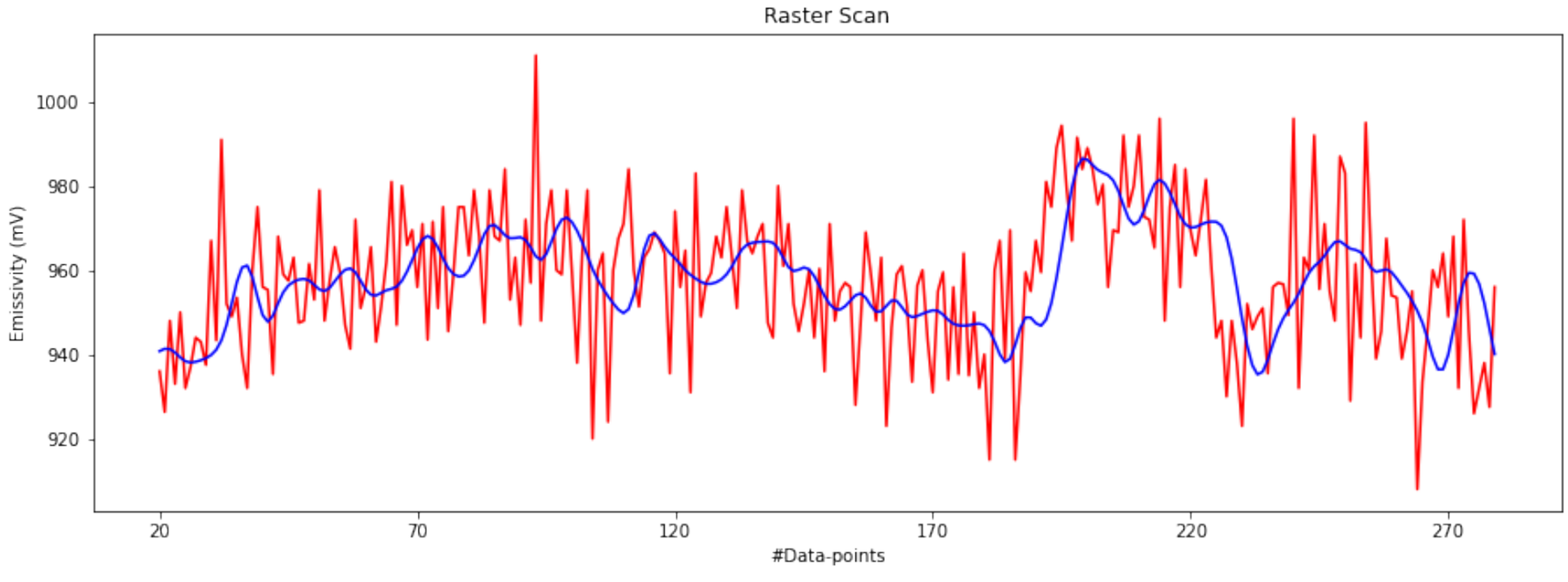


Cleaning the Signal

- Raw Signal



- Butterworth Filter



Time-Series Classification

- Manufacturing Context
- Classification in ML
  - Challenges with time-series data
- The *k*-NN solution
  - Euclidean distance
  - Dynamic Time Warping
- Evaluation
- Exercise
- Model Selection

Performance Measures

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN}$$

$$= \frac{4 + 3}{10} = 0.7$$

$$\text{TPRate} = \frac{TP}{TP + FN} = \frac{4}{4 + 1} = 0.8$$

$$\text{FPRate} = \frac{FP}{FP + TN} = \frac{2}{2 + 3} = 0.4$$

$$\text{TNRate} = \frac{TN}{FP + TN} = \frac{3}{2 + 3} = 0.6$$

$$\text{Precision} = \frac{TP}{TP + FP} = \frac{4}{4 + 2} = 0.667$$

$$\text{Recall} = \frac{TP}{TP + FN} = \frac{4}{4 + 1} = 0.8$$

	Label	Prediction	Correct?	Outcome
1	pore	non-pore	✗	FN
2	pore	pore	✓	TP
3	non-pore	non-pore	✓	TN
4	pore	pore	✓	TP
5	non-pore	pore	✗	FP
6	non-pore	non-pore	✓	TN
7	pore	pore	✓	TP
8	non-pore	pore	✗	FP
9	non-pore	non-pore	✓	TN
10	pore	pore	✓	TP

$$\text{F1} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

$$F1 = \frac{2 \times 0.667 \times 0.8}{0.667 + 0.8} = 0.727$$

ROC Analysis

- Varying the decision threshold value θ can lead to different results, and so to different confusion matrices.

	Label	Score	> 0.5?	Prediction	Outcome
1	pore	0.1	N	non-pore	FN
2	pore	0.8	Y	pore	TP
3	non-pore	0.4	N	non-pore	TN
4	pore	0.9	Y	pore	TP
5	non-pore	0.8	Y	pore	FP
6	non-pore	0.2	N	non-pore	TN
7	pore	0.8	Y	pore	TP
8	non-pore	0.6	Y	pore	FP
9	non-pore	0.1	N	non-pore	TN
10	pore	0.9	Y	pore	TP

Decision Threshold

$\theta = 0.5$

Predicted Class

Non	Pore	
TN=3	**FP=2**	Non
FN=1	**TP=4**	Pore

Real Class

	Label	Score	> 0.7?	Prediction	Outcome
1	pore	0.1	N	non-pore	FN
2	pore	0.8	Y	pore	TP
3	non-pore	0.4	N	non-pore	TN
4	pore	0.9	Y	pore	TP
5	non-pore	0.8	Y	pore	FP
6	non-pore	0.2	N	non-pore	TN
7	pore	0.8	Y	pore	TP
8	non-pore	0.6	N	non-pore	TN
9	non-pore	0.1	N	non-pore	TN
10	pore	0.9	Y	pore	TP

Decision Threshold

$\theta = 0.7$

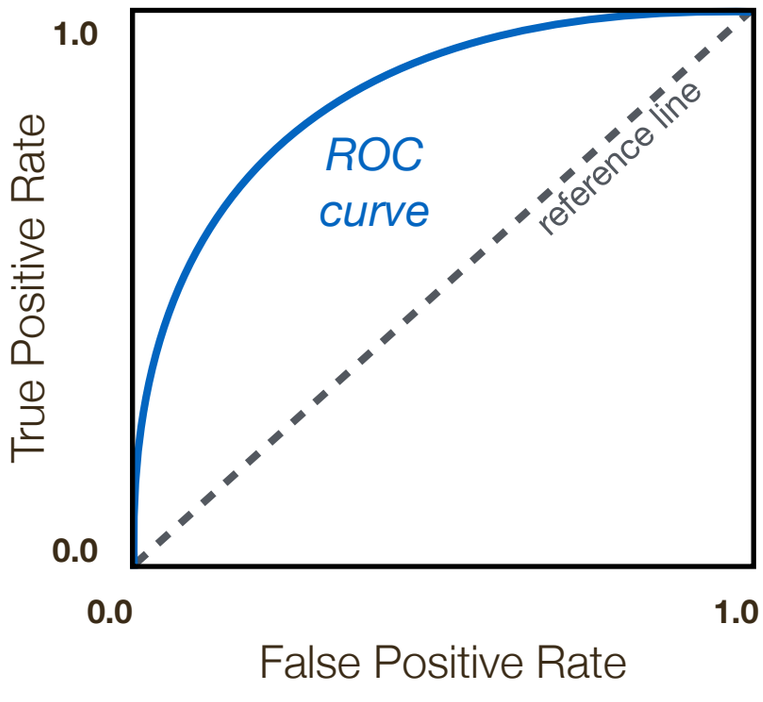
Predicted Class

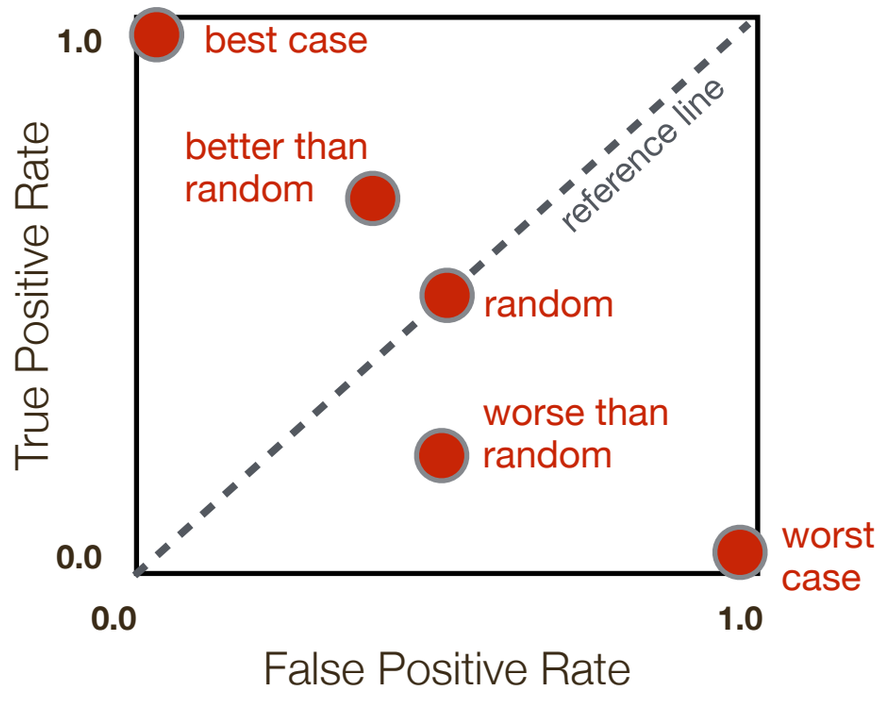
Non	Pore	
TN=4	**FP=1**	Non
FN=1	**TP=4**	Pore

Real Class

ROC Analysis

- We often want to compare the performance of classifiers at many different decision thresholds (i.e. summarise many confusion matrices).
- A Receiver Operating Characteristic (ROC Curve) is a graphical plot of how the true positive rate and false positive rate change over many different thresholds. The curve is drawn by plotting a point for each feasible threshold and joining them.
- A trained classifier should always be above the “random” reference line. The strength of the classifier increases as the ROC curve moves further from the line (i.e. closer to top left corner).





ROC Example

- Given a ranking classifier which will score test samples with a probability P of belonging to the positive class.
- Decision threshold θ controls whether a sample will be classified as positive or negative - i.e. $P > \theta$

Single example $\theta = 0.5$

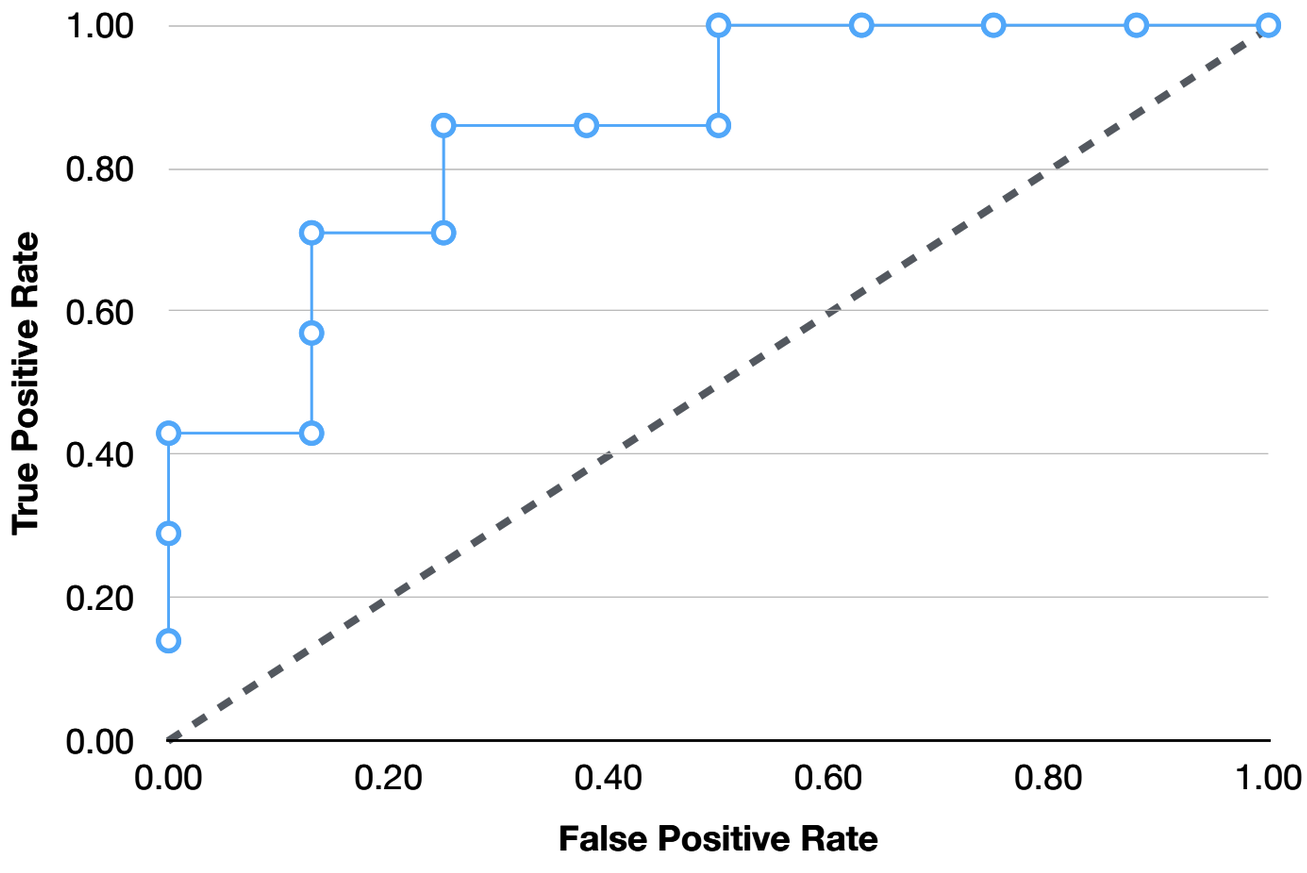
P	> 0.5	Real
0.99	1	1
0.90	1	1
0.85	1	1
0.80	1	0
0.70	1	1
0.70	1	1
0.65	1	0
0.60	1	1
0.45	0	0
0.45	0	0
0.40	0	1
0.30	0	0
0.20	0	0
0.20	0	0
0.20	0	0

ROC Example

- Given a ranking classifier which will score test samples with a probability P of belonging to the positive class.
- Decision threshold θ controls whether a sample will be classified as positive or negative - i.e. $P > \theta$

Single example $\theta = 0.5$

P	> 0.5	Real	FP	TP	FPr	TPr
0.99	1	1	0	1	**0.00**	**0.14**
0.90	1	1	0	2	**0.00**	**0.29**
0.85	1	1	0	3	**0.00**	**0.43**
0.80	1	0	1	3	**0.13**	**0.43**
0.70	1	1	1	4	**0.13**	**0.57**
0.70	1	1	1	5	**0.13**	**0.71**
0.65	1	0	2	5	**0.25**	**0.71**
0.60	1	1	2	6	**0.25**	**0.86**
0.45	0	0	3	6	**0.38**	**0.86**
0.45	0	0	4	6	**0.50**	**0.86**
0.40	0	1	4	7	**0.50**	**1.00**
0.30	0	0	5	7	**0.63**	**1.00**
0.20	0	0	6	7	**0.75**	**1.00**
0.20	0	0	7	7	**0.88**	**1.00**
0.20	0	0	8	7	**1.00**	**1.00**



Time-Series Classification

- Manufacturing Context
- Classification in ML
  - Challenges with time-series data
- The *k*-NN solution
  - Euclidean distance
  - Dynamic Time Warping
- Evaluation
- Exercise
- Model Selection

Exercise

Check to see if smoothing the data (Butterworth Filter) has actually helped classification accuracy.
Update the ROC curves with results for Euclidean distance and DTW on the unfiltered data.
Hint: Simply replace `X_fltr` with the original `X` data.

Time-Series Classification

- Manufacturing Context
- Classification in ML
  - Challenges with time-series data
- The *k*-NN solution
  - Euclidean distance
  - Dynamic Time Warping
- Evaluation
- Exercise
- Model Selection

Model Selection

- AKA: hyper-parameter tuning
- Euclidean distance:
  - *k* neighbours
  - Distance weighting ?
- DTW:
  - *k* neighbours
  - Distance weighting
  - Warping constraint:
    - Mechanism, e.g. Sakoe-Chiba
    - Parameters, e.g. Sakoe-Chiba parameters
- Preprocessing
  - Butterworth Filter
  - Data cleaning: missing value imputation, outlier removal

Grid Search

	k = 1	3	5	7	9	11	13	15	17	19
Uniform										
Distance										

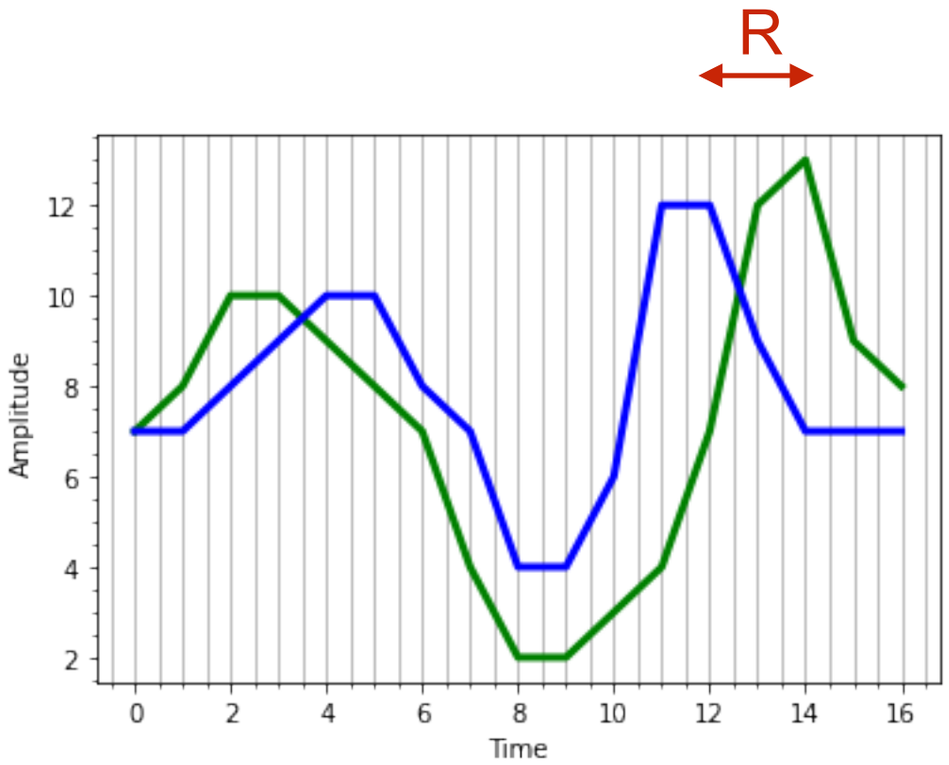
Grid Search

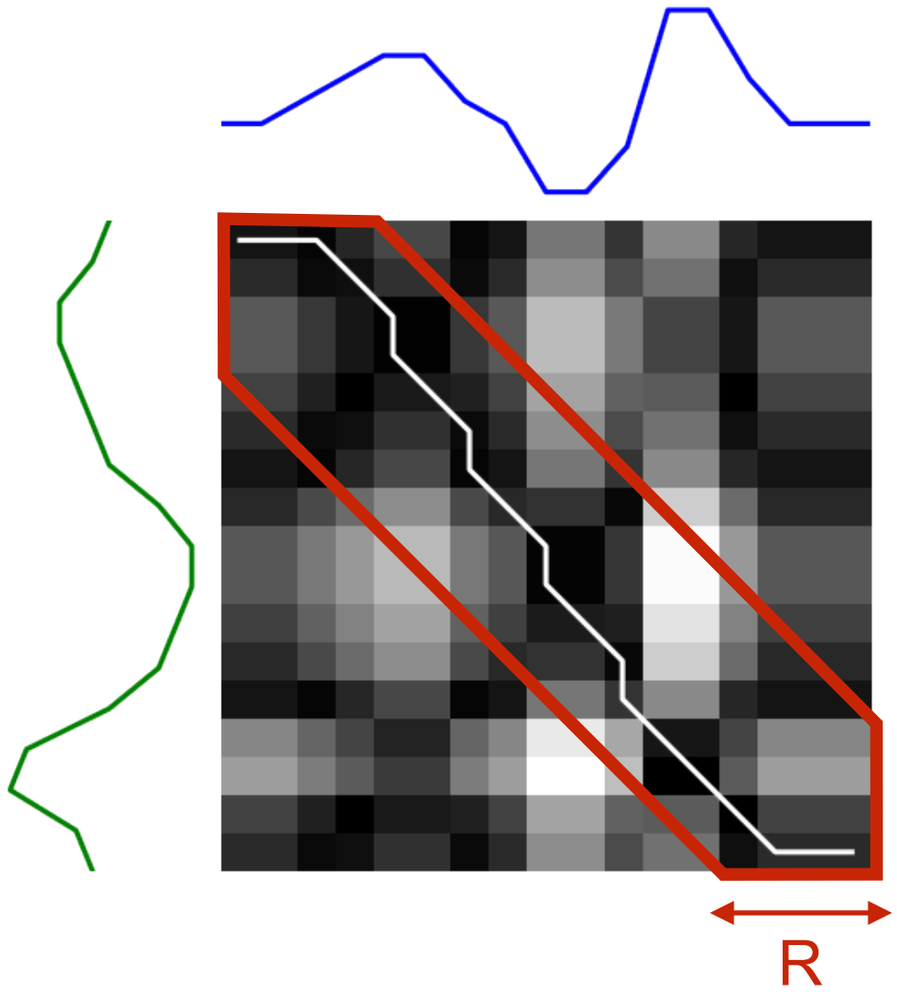
	k									
	1	3	5	7	9	11	13	15	17	19
Uniform										
Distance										

- Evaluate alternatives on the training data
- Test data not used in Grid Search (Model Selection)

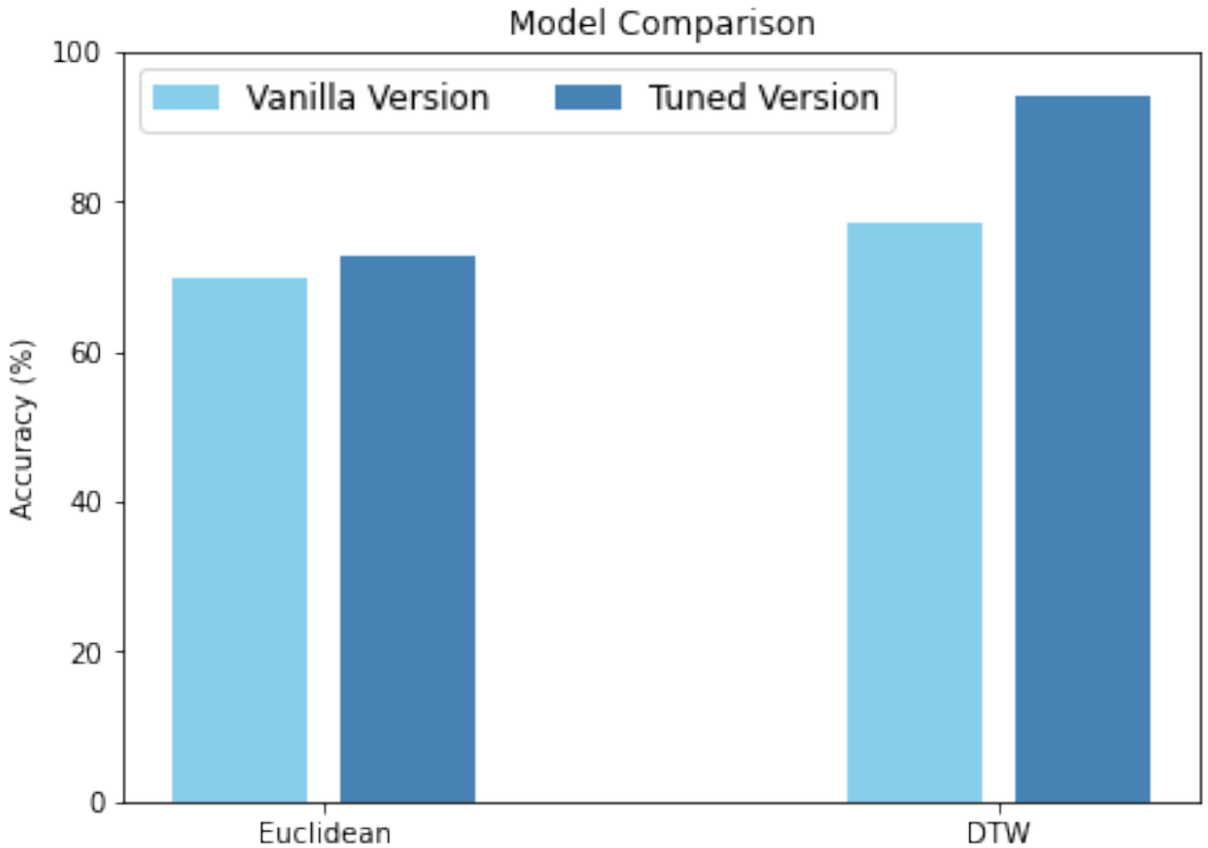
DTW Parameters

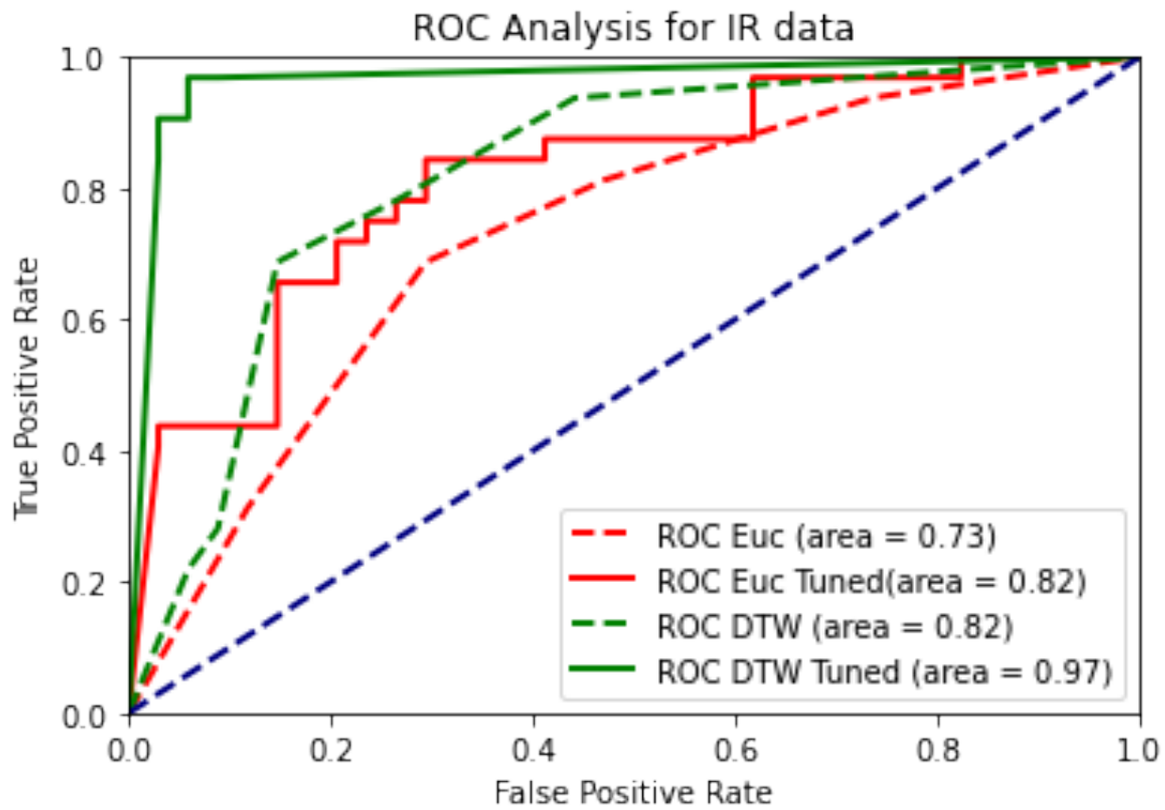
- Constrain the warping
- Depends on the nature of the data





Final Results





Time-Series Classification

- Manufacturing Context
- Classification in ML
  - Challenges with time-series data
- The *k*-NN solution
  - Euclidean distance
  - Dynamic Time Warping
- Evaluation
- Exercise
- Model Selection