Instance Based Classifiers

- Similar instances have similar classification

Classification steps

- Training phase (Model construction): a model is constructed from the training instances.
  - classification algorithm finds relationships between predictors and targets
  - relationships are summarised in a model
- Testing phase:
  - test the model on a test sample whose class labels are known but not used for training the model
- Usage phase (Model usage):
  - use the model for classification on new data whose class labels are unknown

Instance-based Classifiers

- No clear separation between these phases of classification
- also called lazy classification, as opposed to eager classification
- Examples:
  - Rote-learner
    - Memorizes entire training data and performs classification only if attributes of record match one of the training examples exactly
  - Nearest neighbor
    - Uses k "closest" points (nearest neighbors) for performing classification

Eager vs Lazy Classification

Eager Classification

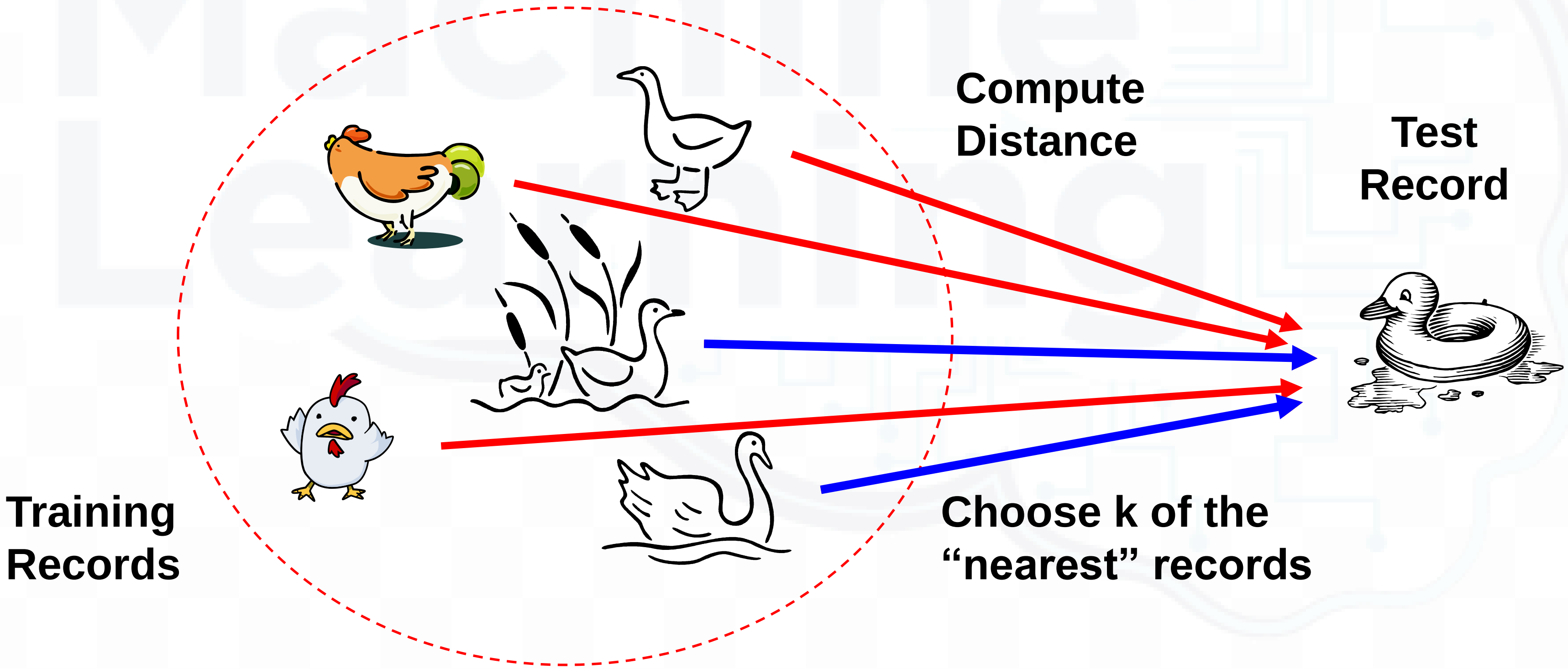
- Model is computed beforeclassification
- Model is independent of the test instance
- Test instance is not included in the training data
- Avoids too much work at classification time
- Model is not accurate for each instance

Lazy Classification

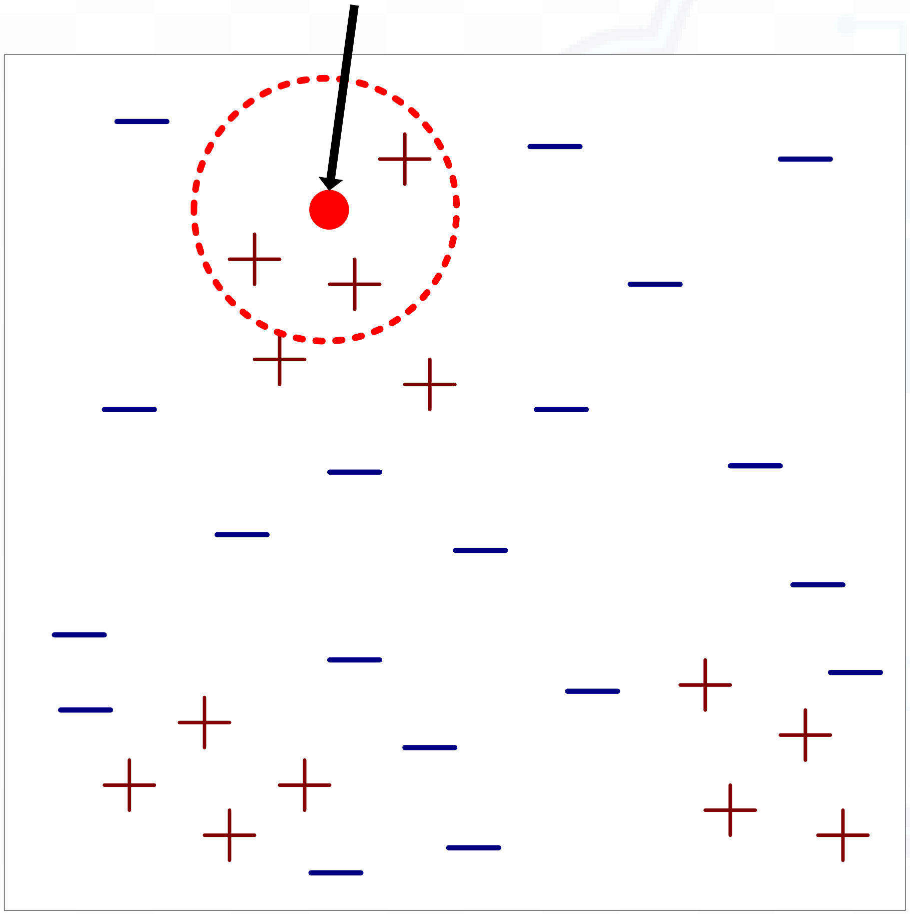
- Model is computed during classification
- Model is dependent on the test instance
- Test instance is included in the training data
- High accuracy for models at each instance level

Nearest Neighbor Classifiers

- Basic idea:
  - If it walks like a duck, quacks like a duck, then it's probably a duck



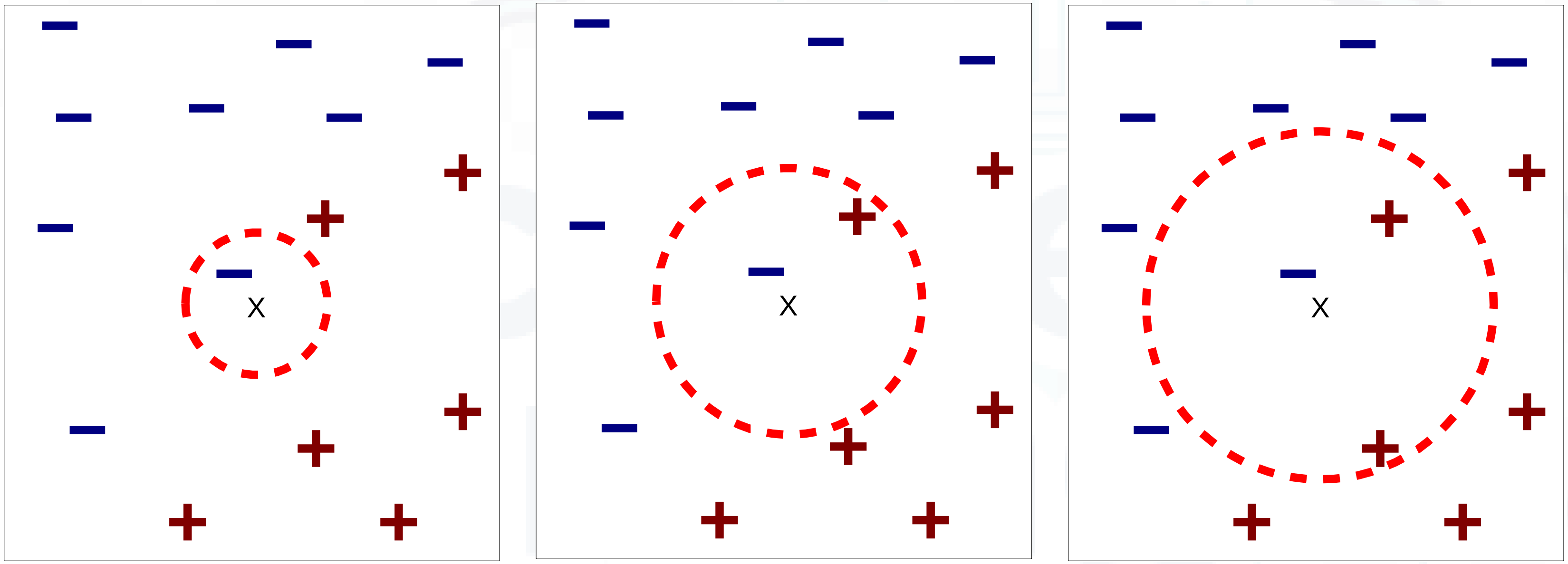
Nearest-Neighbor Classifiers



- Requires three things
  - The set of labeled records
  - Distance Metric to compute distance between records
  - The value of *k*, the number of nearest neighbors to retrieve

- To classify an unknown record:
  - Compute distance to other training records
  - Identify *k* nearest neighbors
  - Use class labels of nearest neighbors to determine the class label of unknown record (e.g., by taking majority vote)

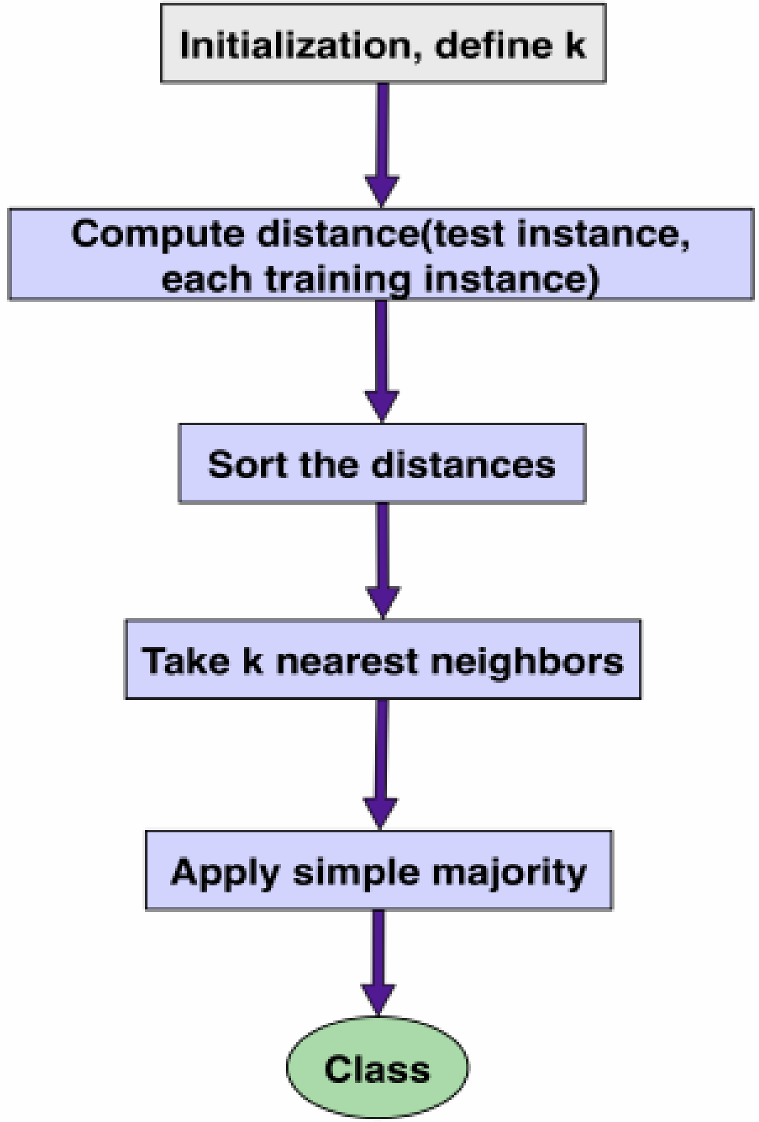
Definition of Nearest Neighbor



(a) 1-nearest neighbor (b) 2-nearest neighbor (c) 3-nearest neighbor

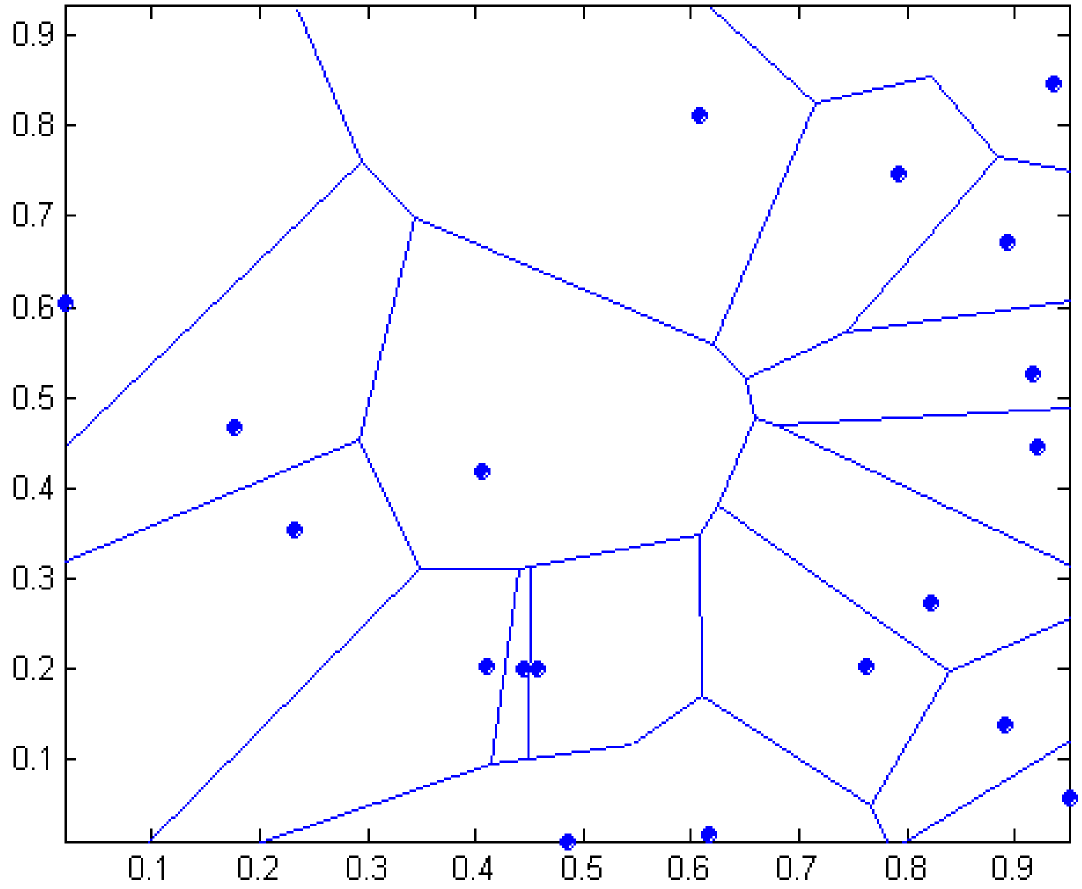
K-nearest neighbors of a record x are data points that have the k smallest distances to x

How does it work?



1 nearest-neighbor

Voronoi Diagram



Predict the same value/class as the nearest instance in the training set

Comparing Objects

- Problem: measure similarity between instances

vs. text similarity

  - different types of data: numbers colours, geolocation, booleans, etc.
- Solution: convert all features of the instances into numerical values
  - represent instances as vectors of features in an n-dimensional space

Distance Metrics

- Euclidean distance:

$$d(x,y)=\sqrt{\sum_{i=1}^{n}(x_i-y_i)^2}$$

- Manhattan distance

$$d(x,y)=\sum_{i=1}^{n}|x_i-y_i|$$

- Minkowski distance

$$d(x,y)=\left(\sum_{i=1}^{n}|x_i-y_i|^p\right)^{1/p}$$

Nearest Neighbor Classification

- Determine the class from nearest neighbor list
  - Take the majority vote of class labels among the k–nearest neighbors

$$y' = \underset{v}{\operatorname{argmax}} \sum_{(x_i, y_i) \in D_z} I(v = y_i)$$

  - Weigh the vote according to distance

$$y' = \underset{v}{\operatorname{argmax}} \sum_{(x_i, y_i) \in D_z} w_i \times I(v = y_i)$$

    - weight factor, $w = 1/d^2$

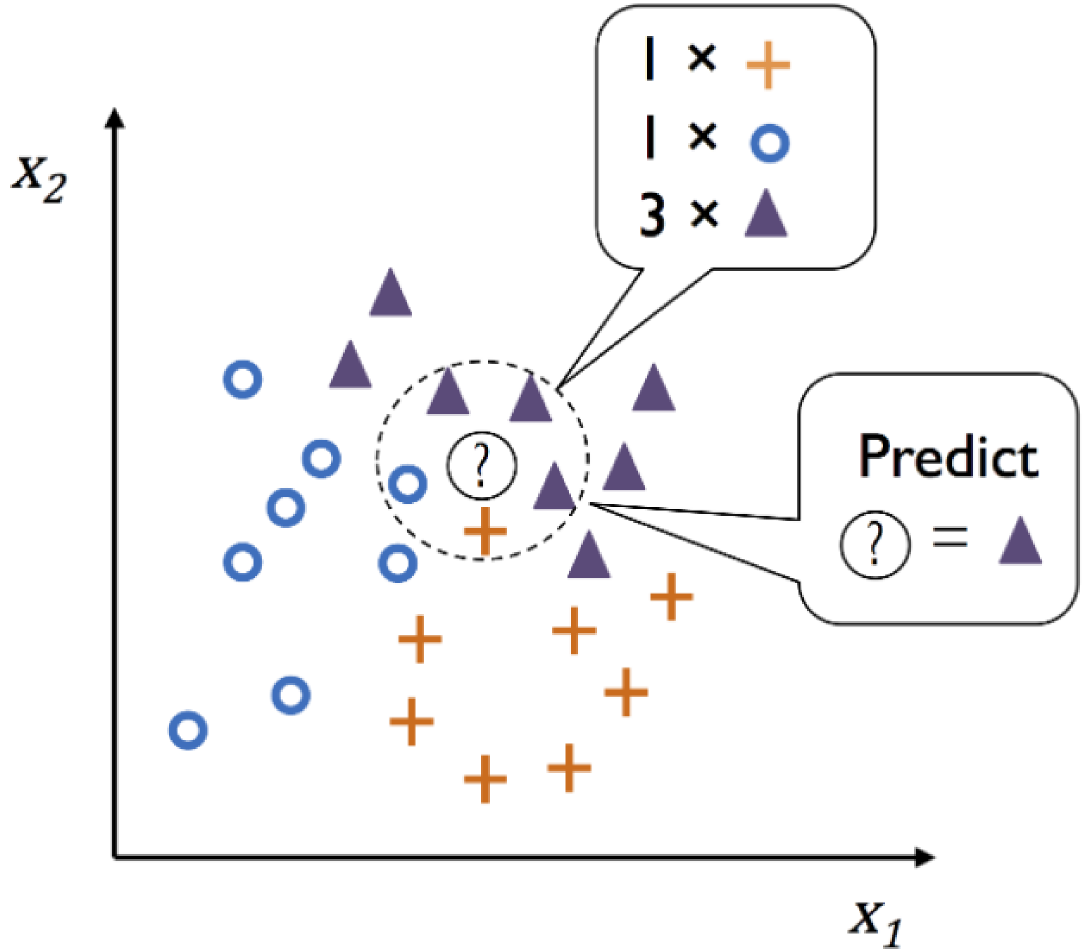
The kNN classification algorithm

- Let k be the number of nearest neighbors and D be the set of training examples.

1. **for** each test example z = (**x**',y') **do**
2. Compute d(**x**',**x**), the distance between z and every example, (**x**,y) ϵ D
3. Select $D_z \subseteq D$, the set of k closest training examples to z.
4. $y' = \underset{v}{\operatorname{argmax}} \sum_{(x_i, y_i) \in D_z} I(v = y_i)$
5. **end for**

Example

- Illustration of kNN for a 3-class problem with k=5

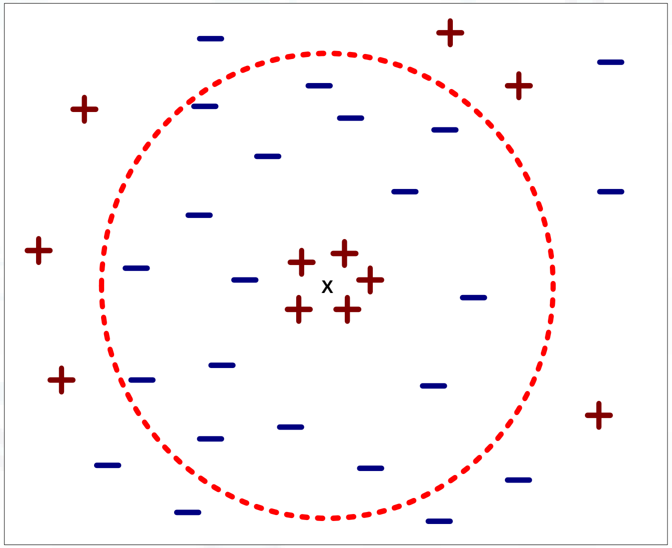


Nearest Neighbor Classification…

- Choosing the value of k: Classification is sensitive to the correct selection of k
  - if k is too small ⇒ overfitting
    - algorithm performs too good on the training set, compared to its true performance on unseen test data

➔ small k? → less stable, influenced by noise
➔ large k? → less precise, higher bias

$$k = \sqrt{n}$$



Nearest Neighbor Classification…

- Scaling issues
  - Attributes may have to be scaled to prevent distance measures from being dominated by one of the attributes
  - Example:
    - height of a person may vary from 1.5m to 1.8m
    - weight of a person may vary from 90lb to 300lb
    - income of a person may vary from $10K to $1M

Nearest Neighbor Classification…

- Selection of the right similarity measure is critical:

1 1 1 1 1 1 1 1 1 1 1 0	vs	0 0 0 0 0 0 0 0 0 0 0 1
0 1 1 1 1 1 1 1 1 1 1 1		1 0 0 0 0 0 0 0 0 0 0 0

Euclidean distance = 1.4142 for both pairs

Solution: Normalize the vectors to unit length

Pros and Cons

- Pros:
  - Simple to implement and use
  - Robust to noisy data by averaging k-nearest neighbours
  - kNN classification is based solely on local information
  - The decision boundaries can be of arbitrary shapes

Pros and Cons

- Cons:
  - Curse of dimensionality: distance can be dominated by irrelevant attributes
  - O(n) for each instance to be classified
  - More expensive to classify a new instance than with a model

FACULTY OF INFORMATION TECHNOLOGY