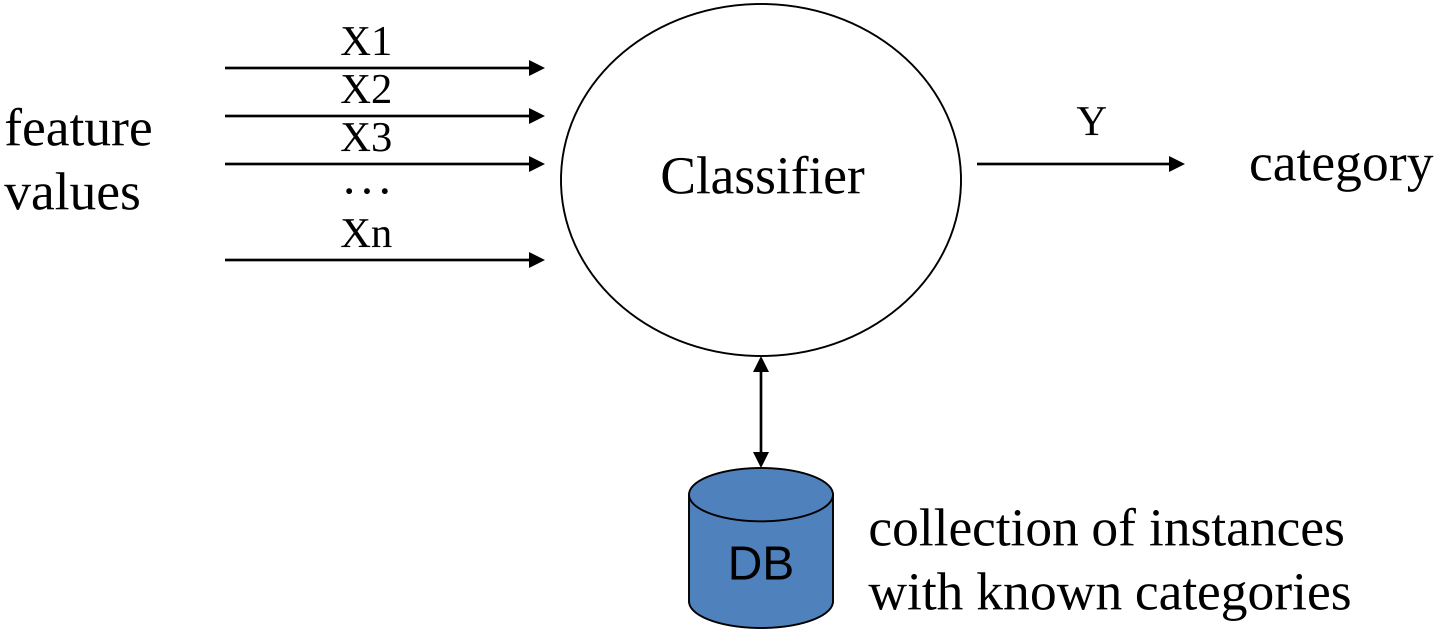
SUPERVISED LEARNING AND K NEAREST NEIGHBORS

Supervised learning and classification

- **Given:** dataset of records with known categories
- **Goal:** using the “knowledge” in the dataset, classify a given instance
  - predict the category of the given instance that is rationally consistent with the dataset

Classifiers



An example application

- An emergency room in a hospital measures 17 variables (e.g., blood pressure, age, etc) of newly admitted patients.

- A decision is needed: whether to put a new patient in an intensive-care unit.

- Due to the high cost of ICU, those patients who may survive less than a month are given higher priority.

- Problem: to predict high-risk patients and discriminate them from low-risk patients.

Another application

- A credit card company receives thousands of applications for new cards. Each application contains information about an applicant,
  - age
  - Marital status
  - annual salary
  - outstanding debts
  - credit rating
  - etc.
- Problem: to decide whether an application should approved, or to classify applications into two categories, approved and not approved.

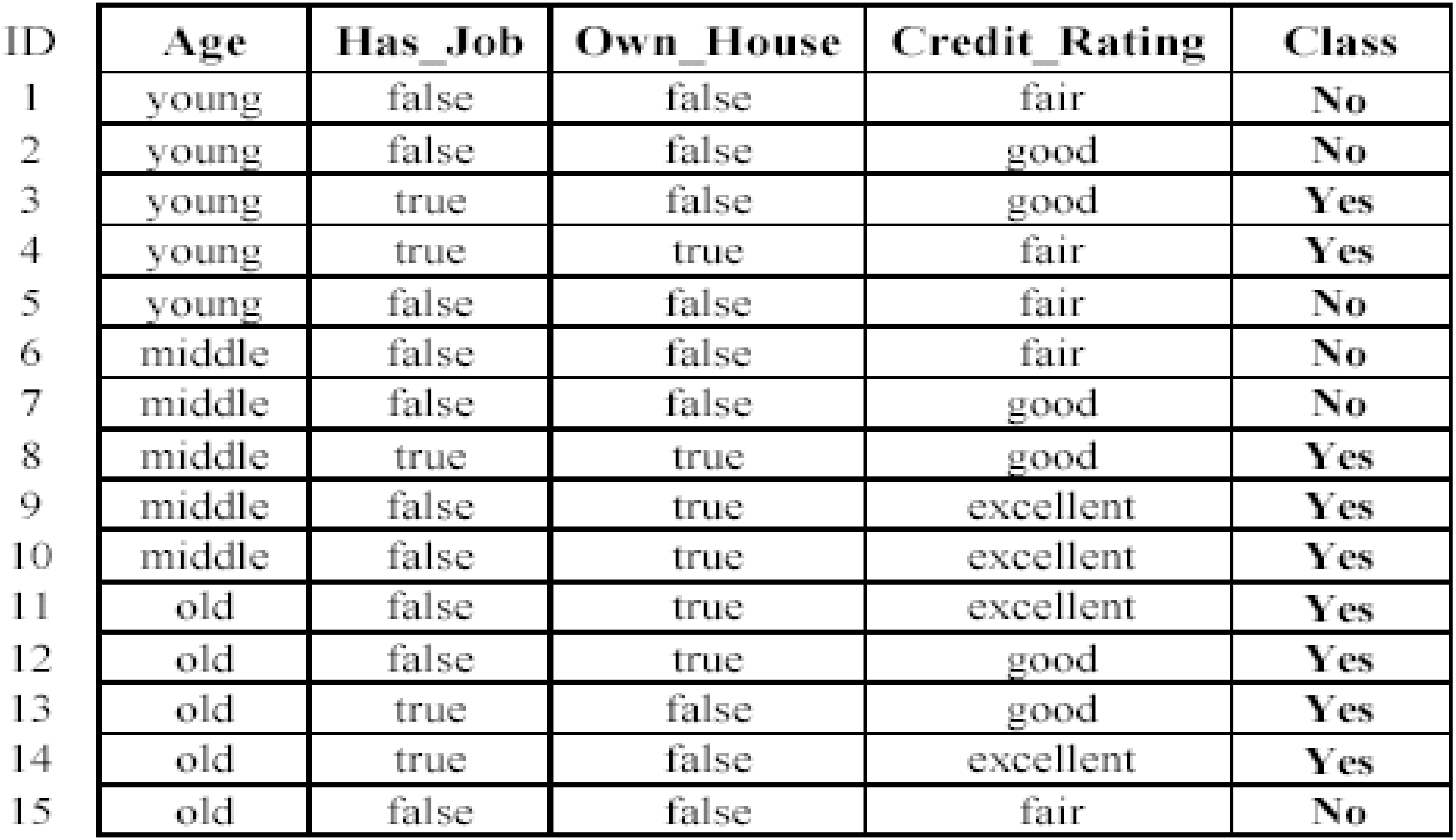
The data and the goal

- **Data:** A set of data records (also called examples, instances or cases) described by
  - k attributes: A_1, A_2, … A_k.
  - a class: Each example is labelled with a pre-defined class.

- **Goal:** To learn a classification model from the data that can be used to predict the classes of new (future, or test) cases/instances.

An example: data (loan application)

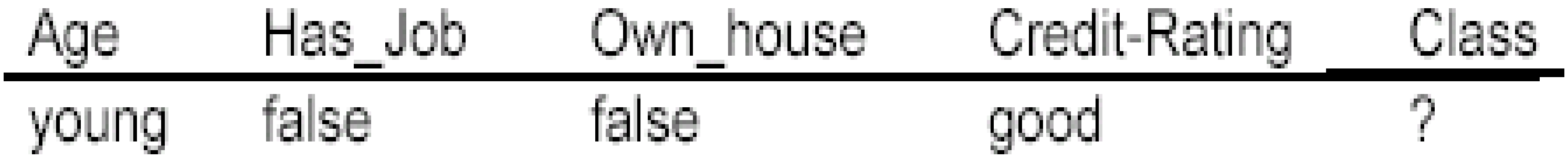
Approved or not



ID	Age	Has_Job	Own_House	Credit_Rating	Class
1	young	false	false	fair	No
2	young	false	false	good	No
3	young	true	false	good	Yes
4	young	true	true	fair	Yes
5	young	false	false	fair	No
6	middle	false	false	fair	No
7	middle	false	false	good	No
8	middle	true	true	good	Yes
9	middle	false	true	excellent	Yes
10	middle	false	true	excellent	Yes
11	old	false	true	excellent	Yes
12	old	false	true	good	Yes
13	old	true	false	good	Yes
14	old	true	false	excellent	Yes
15	old	false	false	fair	No

An example: the learning task

- Learn a classification model from the data
- Use the model to classify future loan applications into
  - Yes (approved) and
  - No (not approved)
- What is the class for following case/instance?

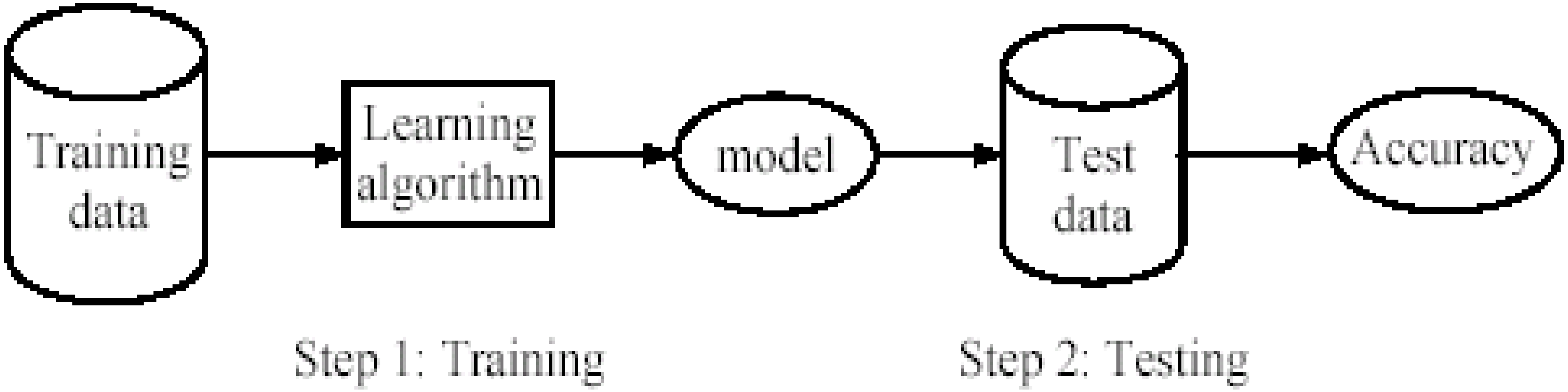


Age	Has_Job	Own_house	Credit-Rating	Class
young	false	false	good	?

Supervised learning process: two steps

- Learning (training): Learn a model using the training data
- Testing: Test the model using unseen test data to assess the model accuracy

$$Accuracy = \frac{\text{Number of correct classifications}}{\text{Total number of test cases}},$$



What do we mean by learning?

- Given
  - a data set D,
  - a task T, and
  - a performance measure M,

- a computer system is said to learn from D to perform the task T if after learning the system’s performance on T improves as measured by M.

- In other words, the learned model helps the system to perform T better as compared to no learning.

An example

- Data: Loan application data
- Task: Predict whether a loan should be approved or not.
- Performance measure: accuracy.

- No learning: classify all future applications (test data) to the majority class (i.e., Yes):
- Accuracy = 9/15 = 60%.

- We can do better than 60% with learning.

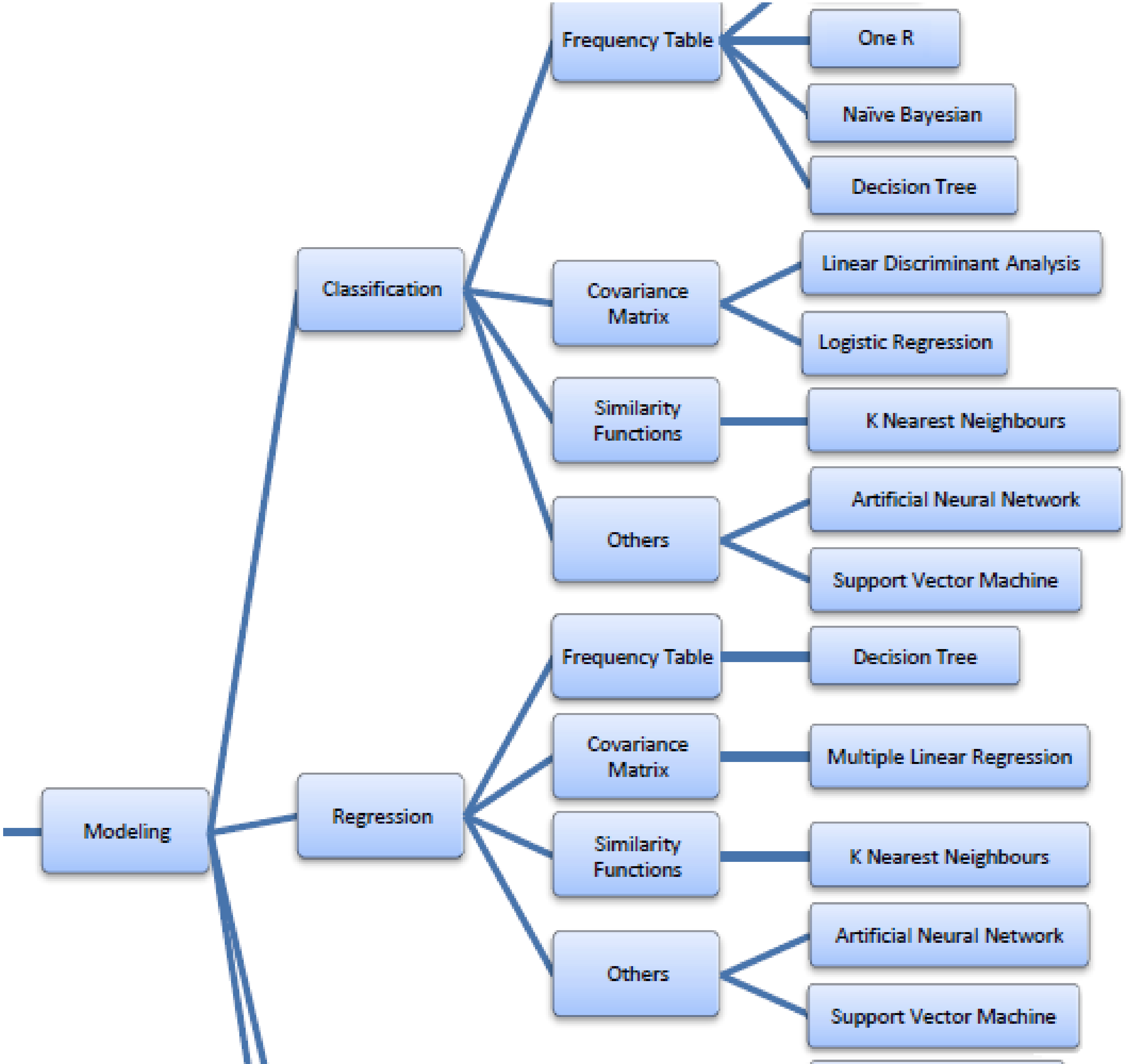
Fundamental assumption of learning

- Assumption: The distribution of training examples is identical to the distribution of test examples (including future unseen examples).

- In practice, this assumption is often violated to certain degree.
- Strong violations will clearly result in poor classification accuracy.
- To achieve good accuracy on the test data, training examples must be sufficiently representative of the test data.

Algorithms

- K Nearest Neighbors (kNN)
- Naïve-Bayes
- Decision trees
- Many others (support vector machines, neural networks, genetic algorithms, etc)



Modeling

Classification

Frequency Table

One R

Naïve Bayesian

Decision Tree

Covariance Matrix

Linear Discriminant Analysis

Logistic Regression

Similarity Functions

K Nearest Neighbours

Others

Artificial Neural Network

Support Vector Machine

Regression

Frequency Table

Decision Tree

Covariance Matrix

Multiple Linear Regression

Similarity Functions

K Nearest Neighbours

Others

Artificial Neural Network

Support Vector Machine

K - Nearest Neighbors

- Assume you are given dataset instances (training examples).

- For a given instance X, get the top k dataset instances that are “nearest” to X
  - Select a reasonable distance measure

- Inspect the category of these k instances, choose the category C that represent the most instances (majority vote).

- Conclude that X belongs to category C

K - Nearest Neighbors

Input: $D = \{(x_1,c_1), \ldots, (x_N, c_N)\}$

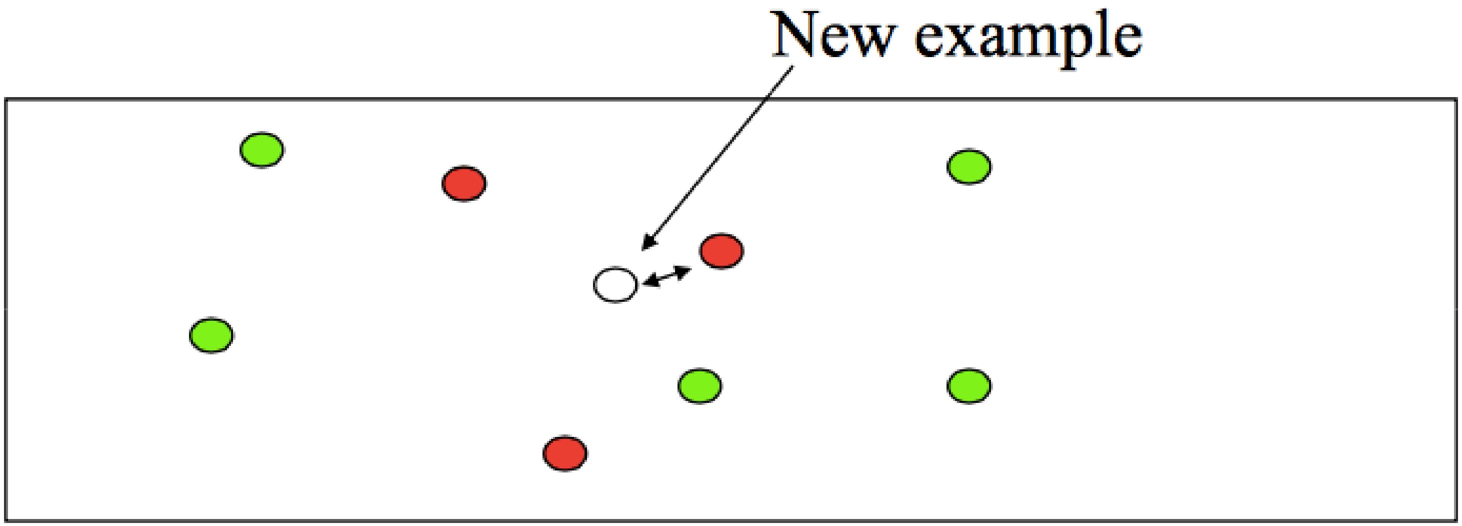
$x = (x_1, \ldots x_f)$ new instance to be classified with f features.

```
KNN( x, D) :
    FOR each labelled instance $(x_i,c_i)$ calculate $d(\mathbf{x}_i,\mathbf{x})$
    Order $d(\mathbf{x}_i,\mathbf{x})$ from lowest to highest, $(i = 1,\ldots,N)$
    Select the $K$ nearest instances to $\mathbf{x}$: $D$
    Assign to $\mathbf{x}$ the most frequent class in $D$
```

KNN Example

- Given training dataset, with height and weight information labeled with gender (F/M).

- Given a new example x, find its closest training example and predict the label / category based on its closest neighbor.



KNN Example (cont.)

- How to measure distance??

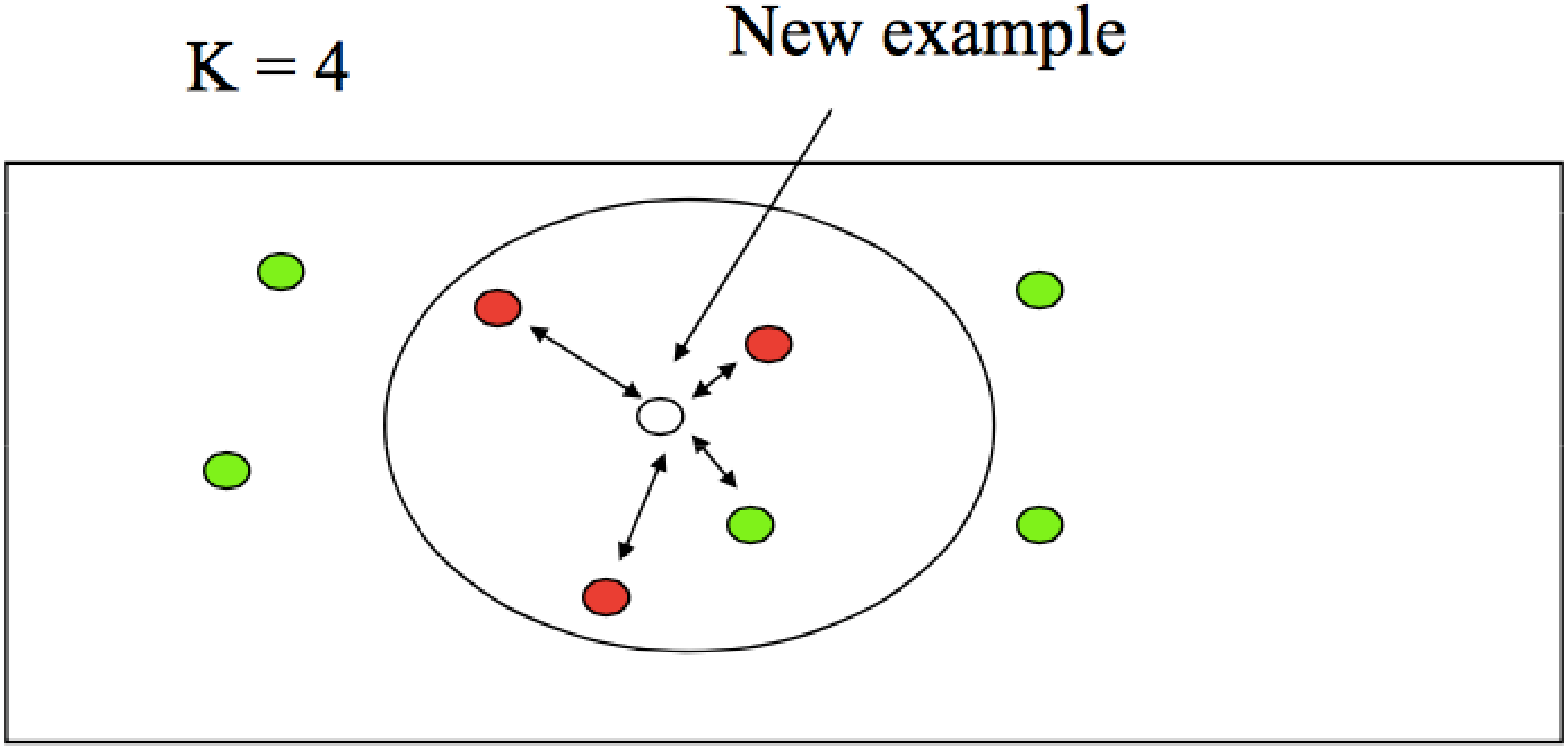
- **Euclidian distance:** squareroot of sum of squares of differences for two features:

$$d(z_1, z_2) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

- Intuition: similar samples should be close to each other
    - May not always apply
      (example: quota and actual sales)

KNN Example

- Find the k nearest neighbors and have them vote.
- This is especially good if there is noise in the class labels.
- Usually its best to choose odd K



Example 1

- Determining decision on scholarship application based on the following features:
  - Household income (annual income in millions of pesos)
  - Number of siblings in family
  - High school grade (on a GPA scale of 1.0 – 4.0)

- Intuition (reflected on data set): award scholarships to high-performers and to those with financial need

Incomparable ranges

- The Euclidian distance formula has the implicit assumption that the different dimensions are comparable

- Features that span wider ranges affect the distance value more than features with limited ranges

- Example:
  - Suppose household income was instead indicated in thousands of pesos per month and that grades are given on a 70-100 scale.

  - Suppose annual income is in dollars, and the other is based on age in years then income will have a much higher influence on the distance calculated.

Incomparable ranges (Cont.)

- One solution is to standardize (or normalize) the training set features.

- **Approach 1:** normalize features to be on the same scale.
  - Linearly scale the range of each feature to be between [0,1]

$$f_{new} = \frac{f_{old} - f_{old}^{\min}}{f_{old}^{\max} - f_{old}^{\min}}$$

Incomparable ranges (Cont.)

- **Approach 2:** normalize features to be on the same scale.
  - Linearly scale to 0 mean and variance 1.
  - For each feature x = (x_1, x_2, … x_n) compute the mean (μ) and the standard deviation (σ) for each feature.

$$f_{new} = \frac{f_{old} - m}{s}$$

  - This is called **Mahalanobis distance** (link). Assumes features are independent of each other.

Non-numeric data

- Feature values are not always numbers

- Example
  - Boolean values: Yes or no, presence or absence of an attribute
  - Categories: Colors, educational attainment, gender

- How do these values factor into the computation of distance?

Dealing with non-numeric data

- Boolean values => convert to 0 or 1
  - Applies to yes-no/presence-absence attributes

- Non-binary characterizations
  - Assign arbitrary numbers but be careful about distances; e.g., color: red, yellow, blue => 1,2,3 … bad approach, why?

  - Given that we want to represent red, yellow, and blue, then represent as a 3-bit vector.

  - This way distance between blue and yellow vs yellow and red is the same.

Preprocessing your dataset

- Dataset may need to be preprocessed to ensure more reliable data mining results

- Conversion of non-numeric data to numeric data

- Calibration of numeric data to reduce effects of disparate ranges
    - Particularly when using the Euclidean distance metric

Evaluating classification methods

- **Predictive accuracy**

$$Accuracy = \frac{\text{Number of correct classifications}}{\text{Total number of test cases}}$$

- Efficiency
  - time to construct the model
  - time to use the model
- Robustness: handling noise and missing values
- Scalability: efficiency in disk-resident databases
- Interpretability:
  - understandable and insight provided by the model
- Compactness of the model: size of the tree, or the number of rules.

Evaluation methods

- **Holdout set**: The available data set *D* is divided into two disjoint subsets,
  - the *training set* D_{train} (for learning a model)
  - the *test set* D_{test} (for testing the model)
- **Important:** training set should not be used in testing and the test set should not be used in learning.
  - Unseen test set provides a unbiased estimate of accuracy.
- This method is mainly used when the data set *D* is large.

Evaluation methods (cont…)

- **n-fold cross-validation**: The available data is partitioned into *n* equal-size disjoint subsets.

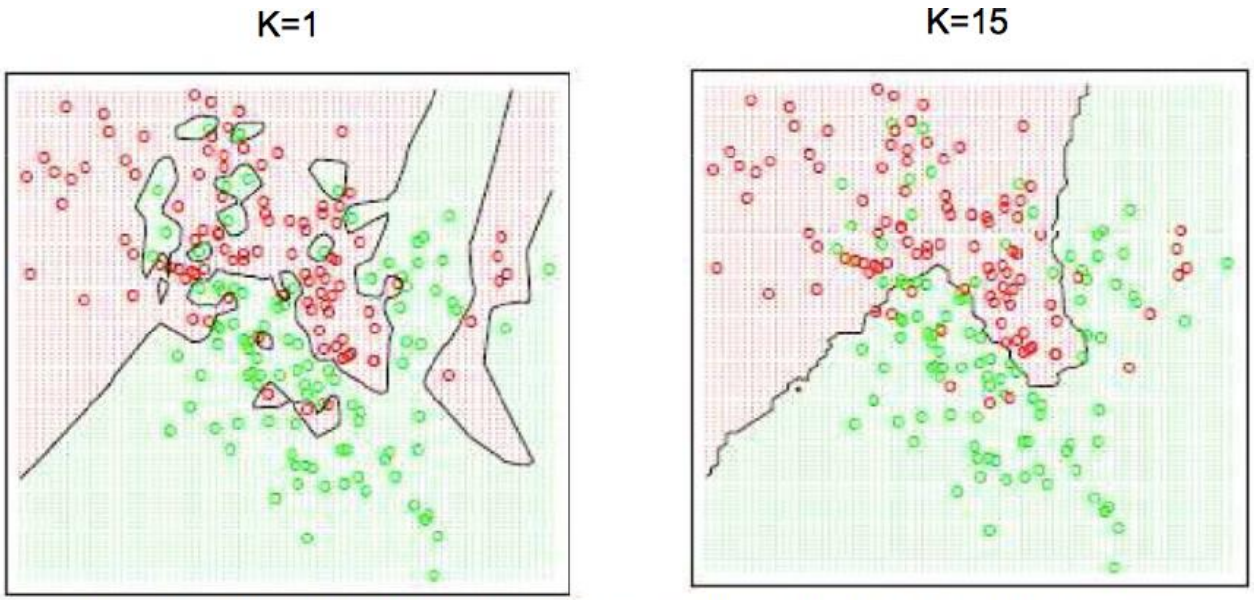
- Use each subset as the test set and combine the rest *n*-1 subsets as the training set to learn a classifier.
- The procedure is run *n* times, which give *n* accuracies.
- The final estimated accuracy of learning is the average of the *n* accuracies.
- 10-fold and 5-fold cross-validations are commonly used.
- This method is used when the available data is not large.

Evaluation methods (cont…)

- Validation set: the available data is divided into three subsets,
  - a training set,
  - a validation set and
  - a test set.
- A validation set is used frequently for estimating parameters in learning algorithms.
- In such cases, the values that give the best accuracy on the validation set are used as the final parameter values.
- Cross-validation can be used for parameter estimating as well.

Effect of k

- If k is too small, its sensitive to noise points
- Larger k produces smoother boundary effect and can reduce impact of class label noise.



Figures from Hastie, Tibshirani and Friedman (Elements of Statistical Learning)

Effect of k

- As a rule of thumb, K is chosen to be equal to the square root of the number of instances.
  - Popularized by the "Pattern Classification" book by Duda et al.

- Cross-validation is a well established technique that can be used to obtain estimates of model parameters that are unknown. It can be used to determine k.

- Divide the data sample into a number of *v* folds.
- For a fixed value of *k*, we apply the *KNN* model to make predictions on the *v*th segment and evaluate the error.
- This process is then applied to all possible choices of *v*.

- At the end of the v folds (cycles), the computed errors are averaged to yield a measure of the stability of the model.
- The above steps are then repeated for various *k* and the value achieving the lowest error is then selected as the optimal value for *k*.

k-NN variations

- Weighted evaluation of nearest neighbors
  - Plain majority may unfairly skew decision
  - Revise algorithm so that closer neighbors have greater “vote weight”
  - i.e. let the closest points among the *K* nearest neighbors have more say in affecting the outcome
  - This can be achieved by introducing a set of weights *W*, one for each nearest neighbor, defined by the relative closeness of each neighbor with respect to the new sample.

$$d(z_1, z_2) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

$$w = 1 / d(z_1, z_2)^2$$

Other distance measures

- City-block distance (Manhattan dist)
  - Add absolute value of differences

- Cosine similarity
  - Measure angle formed by the two samples (with the origin)

- Jaccard distance
  - Determine percentage of exact matches between the samples (not including unavailable data)

- Others

k-NN Time Complexity

- Suppose there are **n** instances and **f** features in the dataset

- Each distance computation involves scanning through each feature value, hence:
  - **O(f)** to compute distance to one example.
  - **O(nf)** to compute distance to all other examples and classify one new sample.
  - Plus, **O(nk)** time to find **k** closest examples.
  - Total time**: O(nk + nf)**

- K-NN is very expensive for a large number of samples.
  - But we need a large number of samples for kNN to work well!!.

K-NN curse of dimensionality

- Consider the problem of classifying text based on authorship.

- May decide to extract features from text by representing a document using n-grams… thus generating thousands of features.

- Some features may be irrelevant…

- “Intrinsic” dimensionality may be smaller than the number of features

Curse of dimensionality

- Datasets typically highly dimensional
  - Vision : 10^4 pixels, texts: 10^6 features

- True dimensionality often much lower
  - True dimensionality refers to features that actually affect classification.

- Dealing with high dimensionality
  - Use domain knowledge to reduce dimensions

  - Reduce the dimensionality of the data (more common approach)

Choosing sets of features

- Score each feature
- Forward/Backward elimination
  - Choose the feature with the highest/lowest score
  - Re-score other features
  - Repeat
- If you have lots of features (like in text)
  - Just select top K scored features