INTELIGENCIA DE NEGOCIO

2018 - 2019

- Tema 1. Introducción a la Inteligencia de Negocio
- Tema 2. Minería de Datos. Ciencia de Datos
- Tema 3. Modelos de Predicción: Clasificación, regresión y series temporales
- Tema 4. Preparación de Datos
- Tema 5. Modelos de Agrupamiento o Segmentación
- Tema 6. Modelos de Asociación
- Tema 7. Modelos Avanzados de Minería de Datos.
- Tema 8. Big Data

Modelos avanzados de Minería de Datos

Objetivos:

- Analizar diferentes extensiones del problema de clasificación clásico de acuerdo a diferentes problemas reales que plantean un nuevo escenario en los problemas de clasificación.
- Introducir brevemente estas extensiones.

Inteligencia de Negocio

TEMA 7. Modelos Avanzados de Minería de Datos

1. Clases no balanceadas/equilibradas
2. Características intrínsecas de los datos en clasificación
3. **Problemas no estándar de clasificación: MIL, MLL, SSL...**
4. Detección de anomalías
5. Deep Learning
6. Análisis de Sentimientos

Nuevos problemas de clasificación

- TÉCNICAS DE CLASIFICACIÓN: Árboles decisión: C4.5, Sistemas basados en reglas, Clasificación basada en instancias (k-NN, ...), regresión logística, SVM, RNN, One-class, modelos probabilísticos,

- Técnicas avanzadas: Ensembles (Bagging, Boosting), Pruning, ...
- Multiclases: OVA, OVO

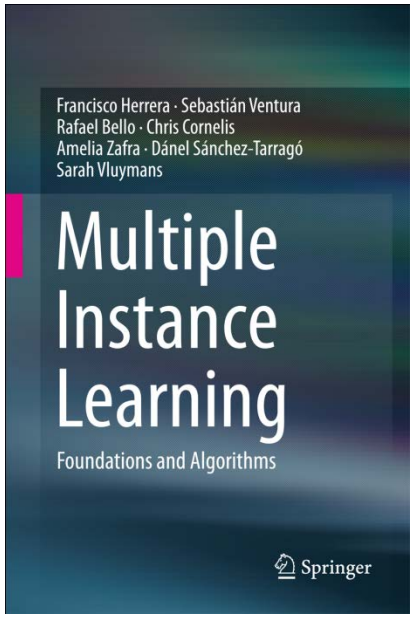
- Datos imperfectos: Valores perdidos, Ruido de clase y variable
- Clases no equilibradas
  Baja densidad de datos– small disjuncts
  Overlapping entre clases
  Dataset Shift - Particionamiento
  Medidas de Complejidad

- Discretización
- Selección de características
- Selección de instancias
- Reducción de la dimensionalidad

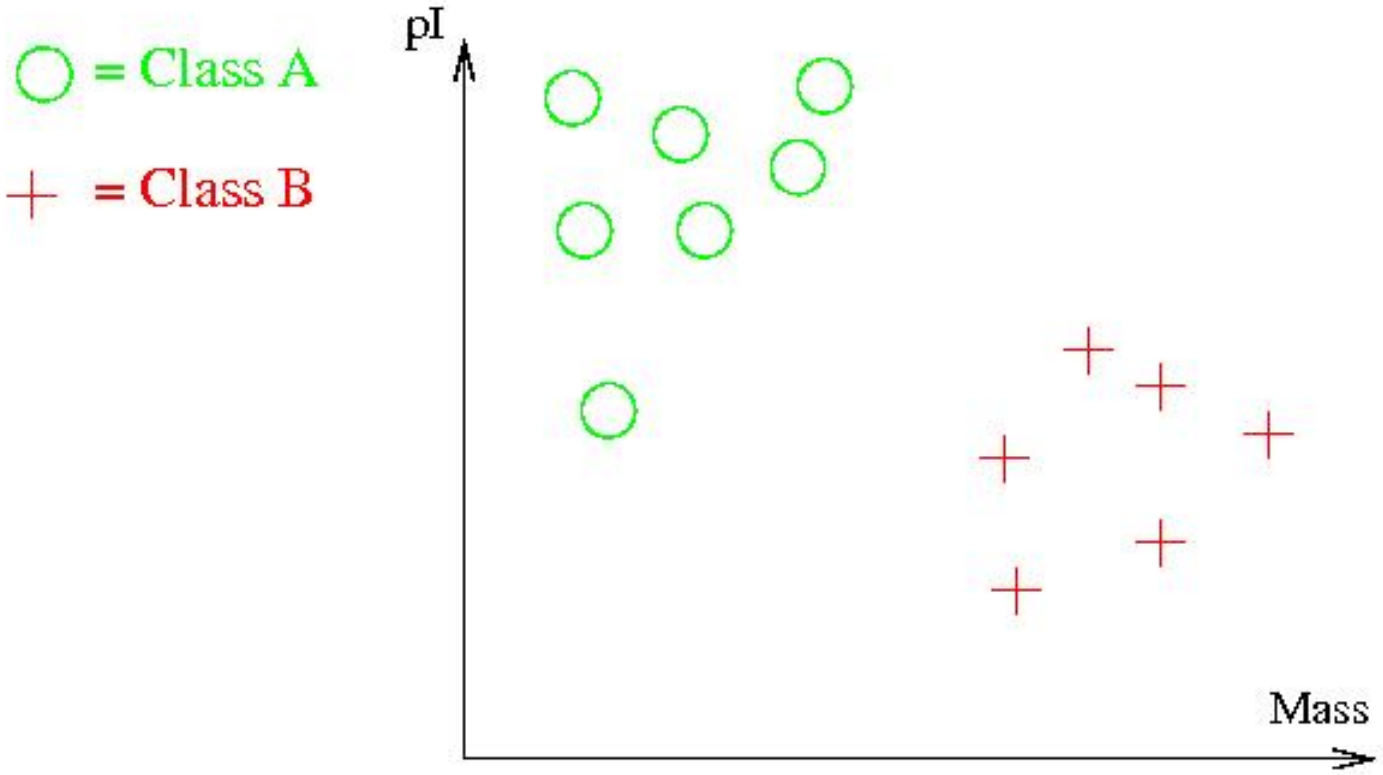
Múltiples etiquetas
Múltiples instancias
Ranking de clases
Clasificación ordinal y monotónica, semisupervisada, multiview learning, ...

Nuevos problemas de clasificación

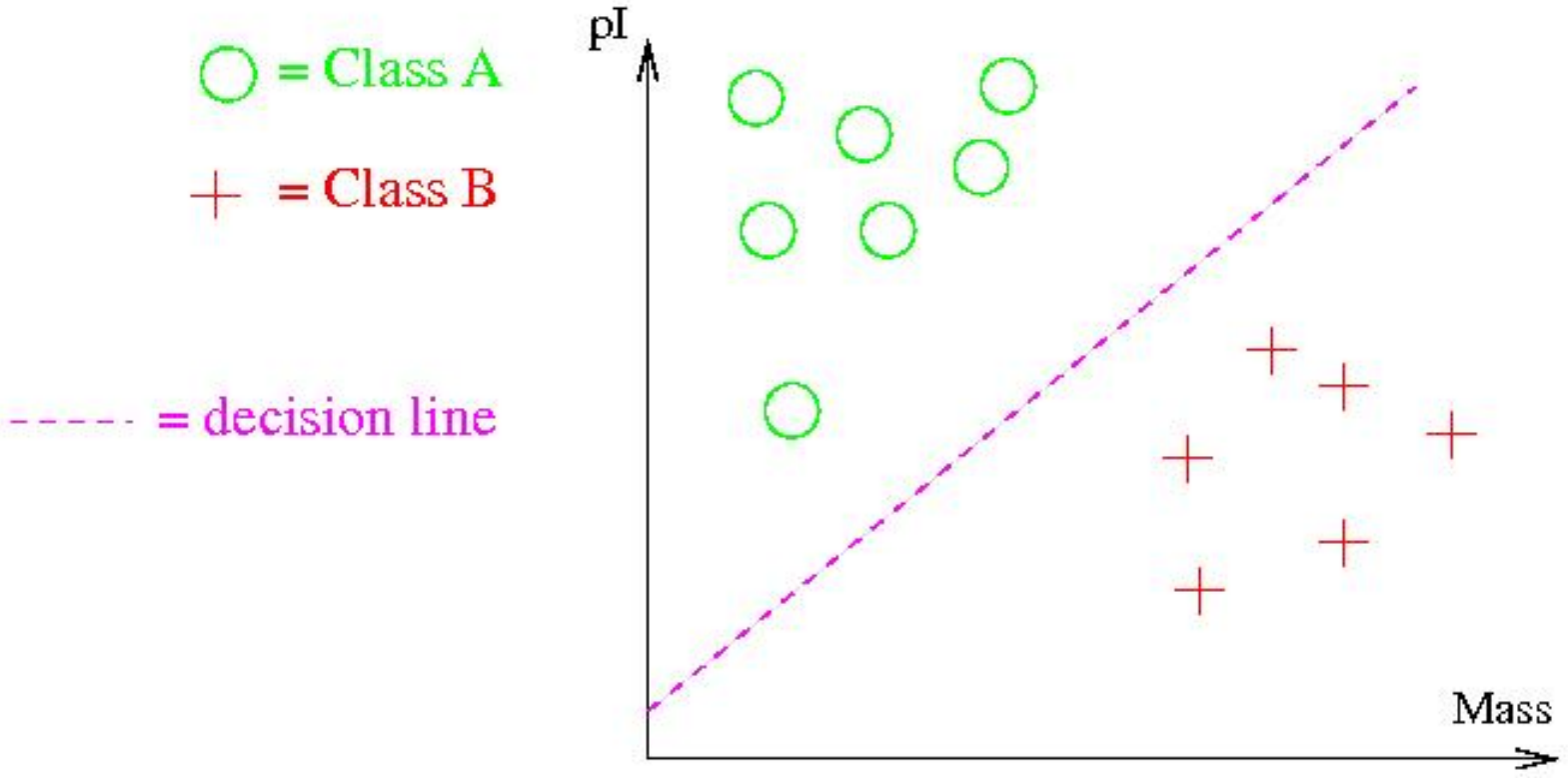
- **MIL: Multi-instance learning**
- ML: Multi-label classification
- Monotonic Classification
- Semisupervised Learning



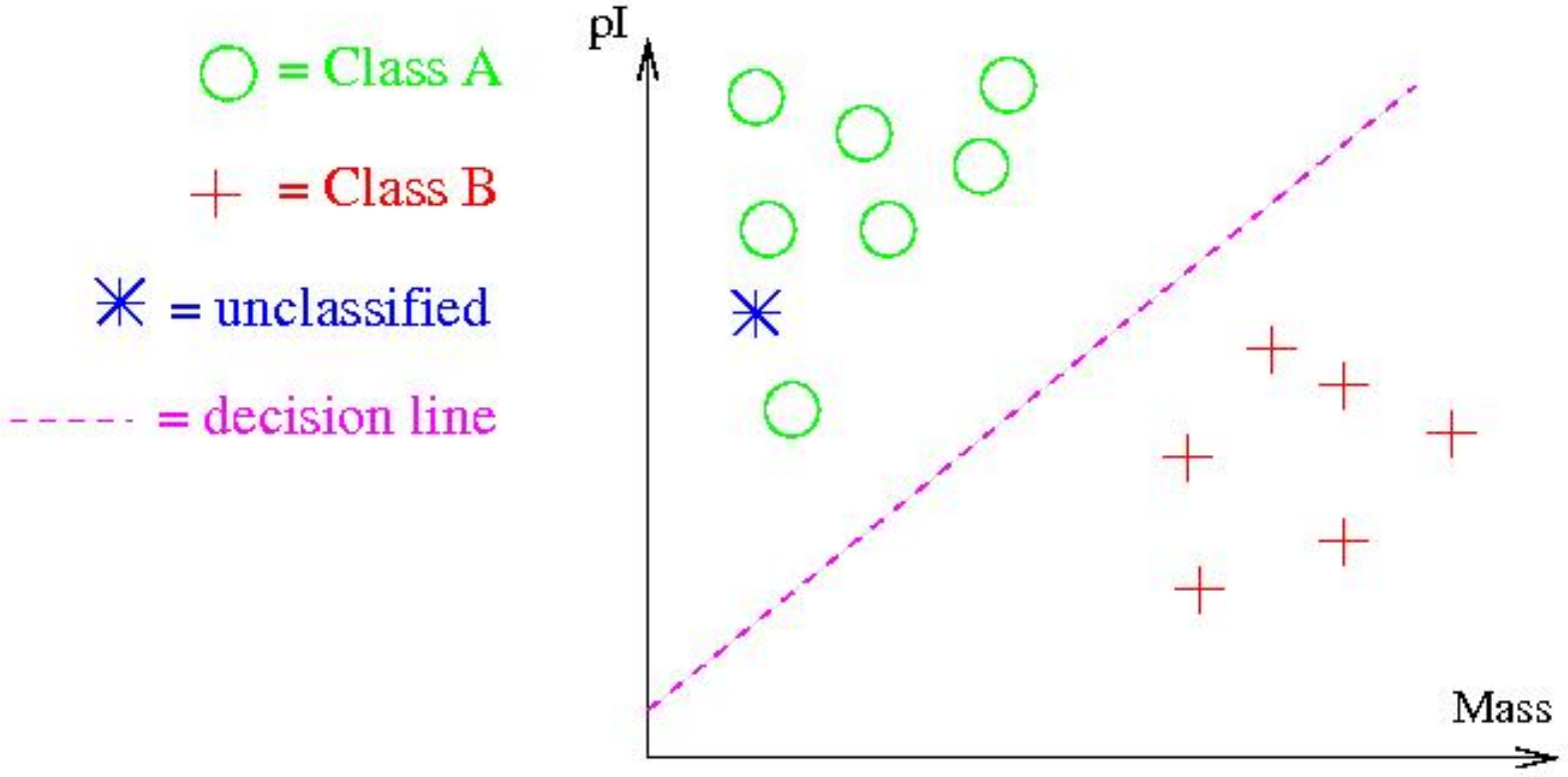
From ML to MIL:
Conventional Machine Learning Model
Decision Problem



From ML to MIL:
Conventional Machine Learning Model
Decision Problem



From ML to MIL:
Conventional Machine Learning Model
Decision Problem



Important decision: Selection of good representation (features) for problem

From ML to MIL: Multi-Instance Learning

- Generalizes conventional machine learning
- Now each example consists of a set (*bag*) of instances
- Single label for entire bag is a function of individual instances' labels

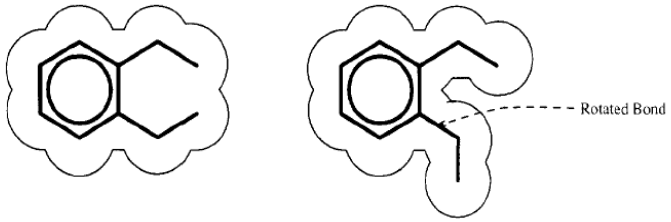
From ML to MIL: Multi-Instance Learning

Originated from the research on drug activity prediction

[Dietterich et al. AIJ97]

Drugs are small molecules working by binding to the target area

- For molecules qualified to make the drug, one of its shapes could tightly bind to the target area
- A molecule may have many alternative shapes



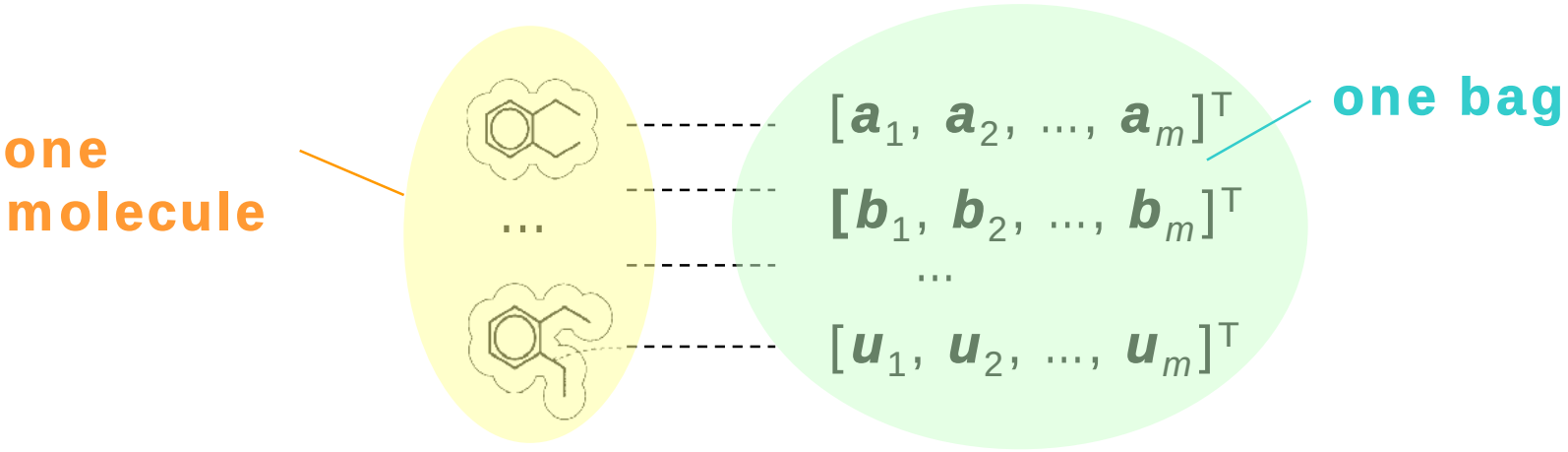
The difficulty:

Biochemists know that whether a molecule is qualified or not, but do not know which shape responses for the qualification

Figure reprinted from [Dietterich et al., AIJ97] [Dietterich *et al.*, 1997] T. G. Dietterich, R. H. Lathrop, T. Lozano-Perez. Solving the Multiple-Instance Problem with Axis-Parallel Rectangles. *Artificial Intelligence Journal,* 89, 1997.

From ML to MIL: Multi-Instance Learning

Each shape can be represented by a feature vector, i.e., an instance



Thus, a molecule is a **bag** of instances

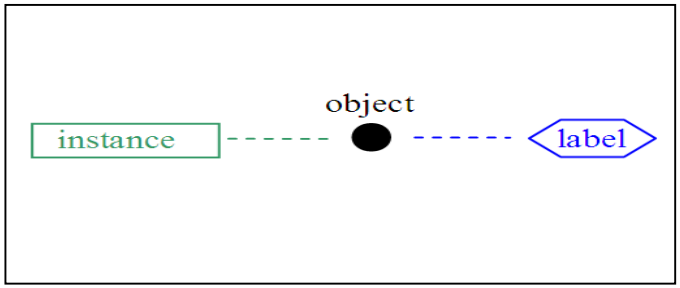
- A bag is positive if it contains at least one positive instance; otherwise it is negative
- The labels of the training bags are known
- The labels of the instances in the training bags are unknown

Figure reprinted from [Zhi-Hua Zhou et al., icml09]

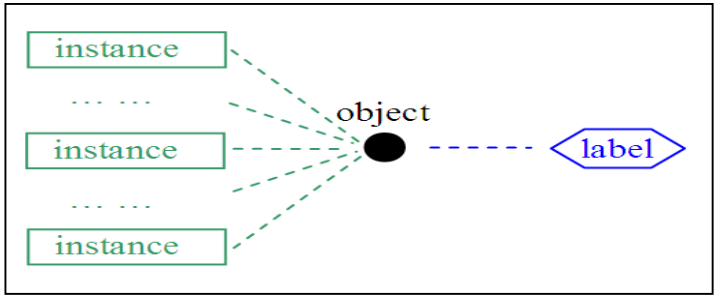
From ML to MIL: Multi-Instance Learning

MIL attempts to learn from a training set consists of bags each containing many instances

- A bag is positive if it contains at least one positive instances; otherwise negative.
- The labels of training bags are known, however, the labels of instances in the bags are unknown.



Traditional supervised learning



Multi-instance learning

In MIL, identifying positive instances is an important problem

✓**understanding the relation between the bag and input patterns.**

From ML to MIL: Multi-Instance Learning

- Multiple-instances Single table
- Examples as sets
- Each instance is a person
- Each set describes a family

Table 3.2. A multi-instance example.

Gene1	Gene2	Gene3	Gene4	Class
aa	aa	aa	AA	negative
aa	aa	aa	aa	
AA	aa	aa	AA	positive
aA	AA	aa	AA	
aA	aA	AA	AA	
aA	aA	AA	aa	
AA	aA	AA	aa	negative
aa	AA	aa	AA	
aa	aA	AA	AA	
aA	AA	AA	AA	positive
aa	AA	AA	aa	
AA	AA	aa	aa	
AA	aa	AA	AA	

Examples, e.g.

```
class(neg) :- person(aa,aa,aa,AA),
                    person(aa,aa,aa,aa).
```

or

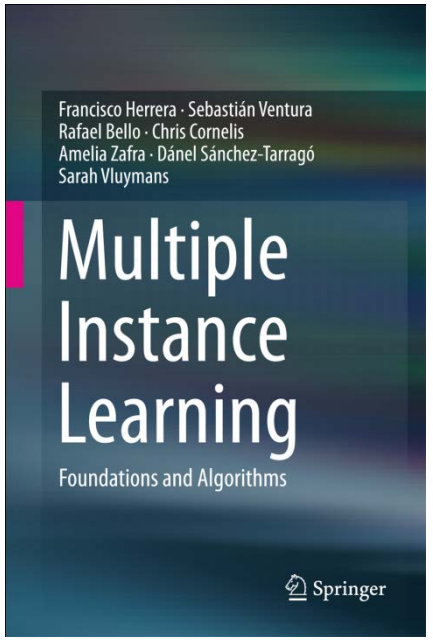
{person(aa,aa,aa,AA), person(aa,aa,aa,aa) }

Multiple-Instance Learning

Learning approaches

- ✓ **Citation kNN**
- ✓ **Support Vector Machine for multi-instance learning**
- ✓ **Multiple-decision tree**
- ✓ **… …**

See: http://link.springer.com/book/10.1007%2F978-3-319-47759-6



Citation K-NN

The popular k Nearest Neighbor (k-NN) approach can be adapted for MIL problems if the distance between bags is defined.

In [Wang and Zucker, 2000], the *minimum Hausdorff distance* was used as the bag-level distance metric, defined as the shortest distance between any two instances from each bag.

$$Dist(A,B) = \min_{\substack{1\le i\le n \\ 1\le j\le m}} (Dist(a_i, b_j) = \min_{a\in A}\min_{b\in B} \|a-b\|$$

where A and B denote two bags, and a_i and b_j are instances from each bag.

Using this bag-level distance, we can predict the label of an unlabeled bag using the k-NN algorithm.

K-NN

However, in a MIL setting, sometimes the majority label of the *K* nearest neighbors of an unlabeled bag is *not* the true label of that bag, mainly because the underlying prediction-generation scheme of *k*NN, *majority voting*, can be easily confused by the false positive instances in positive bags.

The citation approach is used to overcome this weakness, which considers not only the bags as the nearest neighbors (known as references) of a bag *B*, but also the bags that count *B* as their neighbors (known as citers) based on the minimum Hausdorff distance.

Thus, *citation-kNN* predicts the label of a bag based on the labels of both the references and citers of that bag, which is empirically proved to be more robust than the *kNN* based on only references. Another alternative of the majority voting scheme is the Bayesian method, which computes the posterior probabilities of the label of an unknown bag based on labels of its neighbors.

Multiple-Instance Learning

Applications

- ✓ **Drug activity prediction**
- ✓ **Content-based image retrieval and classification**
- ✓ **… …**

Multiple-Instance Learning

Software

- ✓ Iterated-discrim APR [Dietterich *et al.*, 1997]
- ✓ Diverse Density (DD) [Maron and Lozano-Perez, 1998]
- ✓ EM-DD [Zhang and Goldman, 2001]
- ✓ Two SVM variants for MIL [Andrews *et al.*, 2002]
- ✓ Citation-kNN for MIL [Wang and Zucker, 2000]
- ✓ … …

http://www.cs.cmu.edu/~juny/MILL/index.html

MILL: A Multiple Instance Learning Library

Developed by:

Jun Yang
School of Computer Science
Carnegie Mellon University

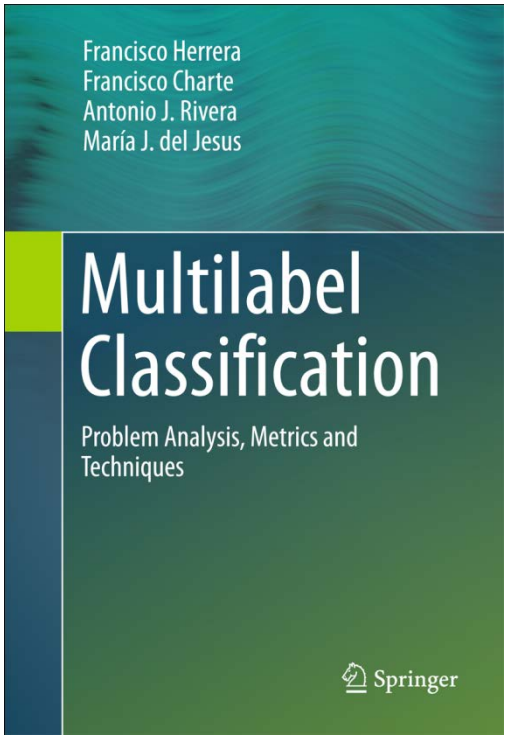
Nuevos problemas de clasificación

- MIL: Multi-instance learning
- **ML: Multi-label classification**
- Monotonic Classification
- Semisupervised Learning

A Tutorial on Multilabel Learning

EVA GIBAJA and SEBASTIÁN VENTURA, Department of Computer Science and Numerical Analysis, University of Córdoba, Spain

ACM Computing Surveys, Vol. 47, No. 3, Article 52, Publication date: April 2015.



Motivation: Multi-label objects

- Text classification is everywhere
  - Web search
  - Business ication
  - Email classification
- Many text data ar

Politics

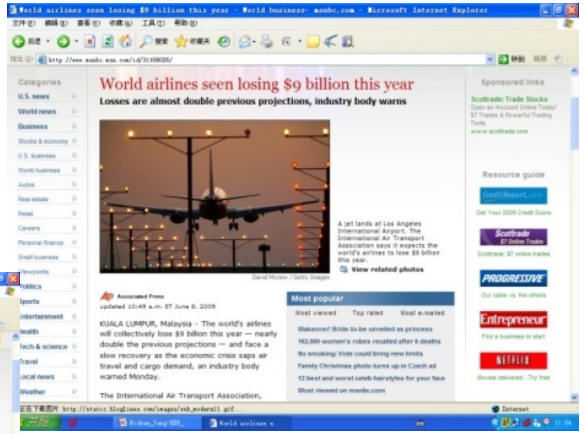
Travel

World news

Entertainment

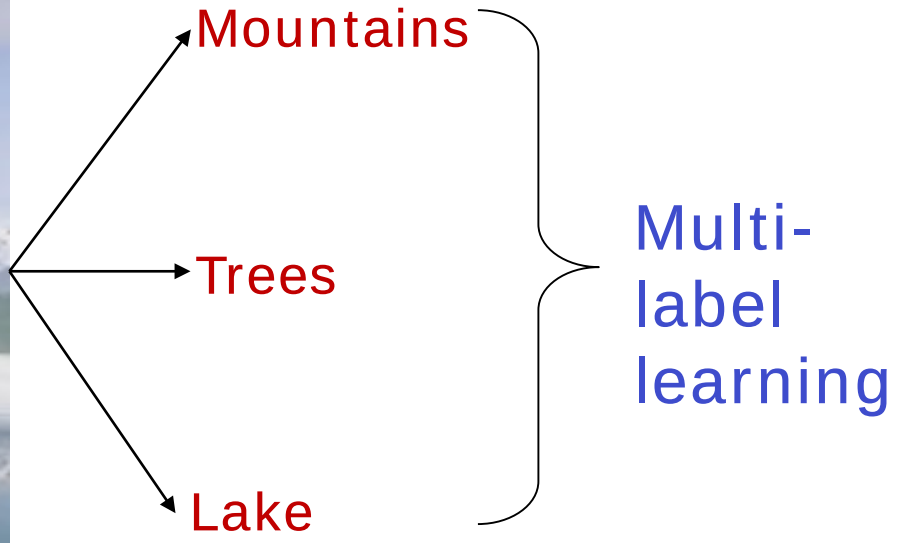
Local news

... ...



Motivation: Multi-Label Objects

e.g. natural scene image



Documents, Web pages, Molecules……

Multi-label classification

- Traditional *single-label* classification is concerned with learning from a set of examples that are associated with a single label l from a set of disjoint labels L, |L| > 1.

- In *multi-label* classification, the examples are associated with a set of labels Y in L.

- In the past, multi-label classification was mainly motivated by the tasks of text categorization and medical diagnosis. Nowadays, we notice that multi-label classification methods are increasingly required by modern applications, such as protein function classification, music categorization and semantic scene classification.

Multi-label classification Formal Definition

Settings:

- $\square$ *d*-dimensional input space $\square^d$
- $\square$ the finite set of possible labels or classes
- H: $\square \rightarrow 2^{\square}$, the set of multi-label hypotheses

Inputs:

S: i.i.d. multi-labeled training examples $\{(x_i, Y_i)\}$ (i=1,2,...m) drawn from an unknown distribution D, where $x_i \in \square$ and $Y_i \subseteq \square$

Outputs:

h: $\square \rightarrow 2^{\square}$, a *multi-label* predictor; or

f : $\square \times \square \rightarrow \square$, a *ranking* predictor, where for a given instance x, the labels in $\square$ are ordered according to $f(x,\cdot)$

Multi-label classification Evaluation Metrics

Given:

S: a set of multi-label examples $\{(x_i, Y_i)\}$ $(i=1,2,\ldots m)$, where $x_i \in \square$ and $Y_i \subseteq \square$

$f : \square \times \square \rightarrow \square$ a ranking predictor (h *is* the corresponding multi-label predictor)

Definitions:

Hamming Loss:

$$\text{hamloss}_S(f) = \frac{1}{m \times k} \sum_{i=1}^{m} \left| h(x_i) \Delta Y_i \right|$$

One-error:

$$\text{one-err}_S(f) = \frac{1}{m} \sum_{i=1}^{m} \left| \{ i \mid H(x_i) \notin Y_i \} \right|, \text{ where } H(x) = \underset{l \in Y}{\text{argmax}}\ f(x,l)$$

Coverage

$$\text{coverage}_S(f) = \frac{1}{m} \sum_{i=1}^{m} \max_{y \in Y_i} rank_f(x_i, y) - 1$$

Ranking Loss:

$$\text{rankloss}_S(f) = \frac{1}{m} \sum_{i=1}^{m} \frac{1}{\left|Y_i\right| \left|\overline{Y_i}\right|} \left| \left\{ (l_0, l_1) \in \overline{Y_i} \times Y_i \mid f(x_i, l_1) \leq f(x_i, l_0) \right\} \right|$$

Average Precision:

$$\text{avgprec}_S(f) = \frac{1}{m} \sum_{i=1}^{m} \frac{1}{|Y_i|} \sum_{l \in Y_i} \frac{\left| \left\{ l' \in Y_i \mid f(x_i, l') > f(x_i, l) \right\} \right|}{\left| \left\{ j \in \{1, \ldots, k\} \mid f(x_i, j) > f(x_i, l) \right\} \right|}$$

Learning approaches, applications, software

Text Categorization

- BoosTexter
  - Extensions of AdaBoost
  - Convert each multi-labeled example into many binary-labeled examples
- Maximal Margin Labeling
  - Convert MLL problem to a multi-class learning problem
  - Embed labels into a similarity-induced vector space
  - Approximation method in learning and efficient classification algorithm in testing
- Probabilistic generative models
  - Mixture Model + EM
  - PMM

Learning approaches, applications, software

Extended Machine Learning Approaches

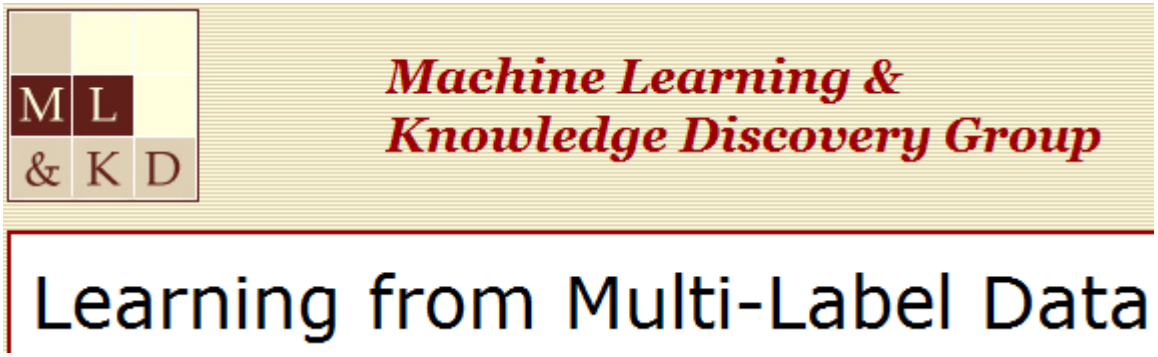
- ADTBoost.MH
  - Derived from AdaBoost.MH [Freund & Mason, ICML99]
    and ADT (Alternating Decision Tree) [Freund & Mason, ICML99]
  - Use ADT as a special weak hypothesis in AdaBoost.MH
- Rank-SVM
  - Minimize ranking loss criterion while at the same have a large margin
- Multi-Label C4.5
  - Modify the definition of entropy
  - Learn a set of accurate rules, not necessarily a set of complete classification rules
- ML k-NN

Learning approaches, applications, software

Software: Mulan: An Open Source Library for Multi-Label Learning

- ✓ Java library for **Mu**lti-**la**bel lear**n**ing, called **Mulan**
- ✓ Mulan is hosted at SourceForge, so you can grab latest releases from there, as well as the latest development source code from the project's public SVN repository.
- ✓ There is a collection of several multilabel datasets, properly formatted for use with Mulan.
- ✓ … …

http://mlkd.csd.auth.gr/multilabel.html



Nuevos problemas de clasificación

- MIL: Multi-instance learning
- ML: Multi-label classification
- **Monotonic Classification**
- Semisupervised Learning

New Generation Computing, 33(2015)367-388
Ohmsha, Ltd. and Springer

Monotonic Random Forest with an Ensemble Pruning Mechanism based on the Degree of Monotonicity

Sergio GONZÁLEZ, Francisco HERRERA and Salvador GARCÍA
Department of Computer Science and Artificial Intelligence,
University of Granada, 18071, Granada, SPAIN
sergio.gvz@gmail.com, {herrera, salvagl}@decsai.ugr.es

Monotonic Classification

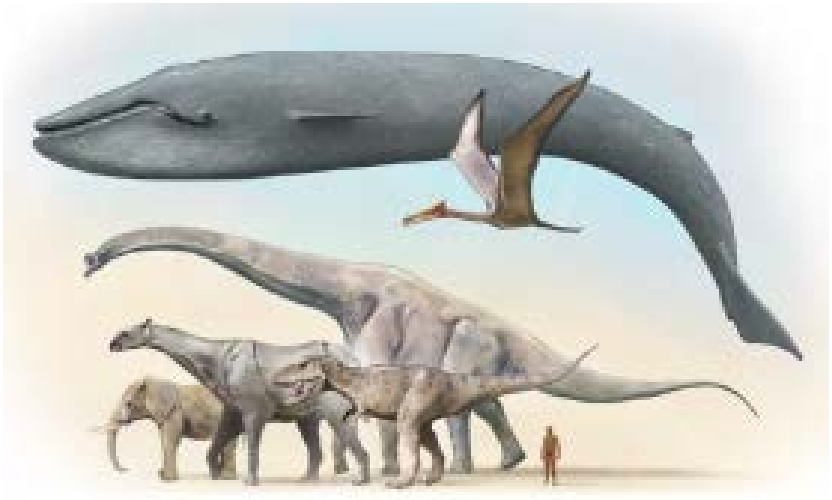
Escenario común

Economía: Los bancos para la asignación de créditos a empresas, toman como base los estados financieros y de ellos un subconjunto de parámetros que permitan clasificarlas como buenas. De allí que si una empresa es evaluada en lo particular como buena (algún parámetro) en lo general debería también ser buena (resto de parámetros).

Algebra: Consideremos la monotonía de la suma donde: si $a > b$ tenemos que $a+c > b+c$

Monotonic Classification

Biología: Mientras más grandes evolucionan los animales de un cierto grupo, entonces más pequeño se vuelve el número de individuos. Una condición que permitiría evaluar a los dinosaurios por parte de los paleontólogos.



Monotonic Classification

Restricción monotónica

Dado un conjunto parcialmente ordenado de instancias denotado por $\boldsymbol{X}$, y un conjunto finito de clases ordenado denotado por $\boldsymbol{C}$. La relación de orden de $\boldsymbol{X}$ y $\boldsymbol{C}$ es definida por el operador $\leq$, donde la regla de asignación $\boldsymbol{f}: \boldsymbol{X} \rightarrow \boldsymbol{C}$ asigna una clase de $\boldsymbol{C}$ a cada instancia de $\boldsymbol{X}$.

En esta definición un **problema de clasificación** es el encontrar una clase que etiquete a $\boldsymbol{f}$ satisfaciendo la **restricción de monotonía**:

$$\boldsymbol{x} \leq \boldsymbol{x}' \Rightarrow \boldsymbol{f}(\boldsymbol{x}) \leq \boldsymbol{f}(\boldsymbol{x}')$$

Es decir, para este caso $\boldsymbol{f}$ es una función no-decreciente monotonamente en $\boldsymbol{X}$ para cada elemento $\boldsymbol{x}, \boldsymbol{x}' \in \boldsymbol{X}$

En concreto, la restricción monotónica determina que aquellos objetos que tengan mejores vectores de características no deberían ser asignados a peores clases. (Hu *et al*, 2012)

Monotonic Classification

A monotonic classifier is one that will not violate monotonicity constraints. Informally, the monotonic classification implies that the assigned class values are monotonically nondecreasing (in ordinal order) with the attribute values.

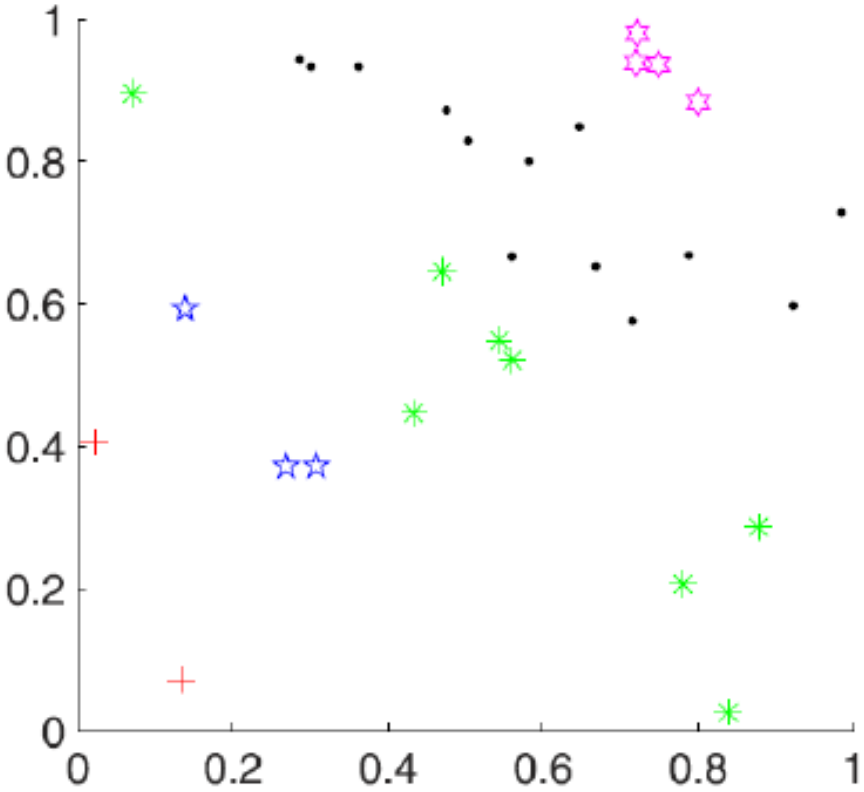
More formally, let $\{\mathbf{x}_i, \text{class}(\mathbf{x}_i)\}$ denote a set of examples with attribute vector $\mathbf{x}_i = (\mathbf{x}_{i,1}, \ldots, \mathbf{x}_{i,m})$ and a class, $\text{class}(\mathbf{x}_i)$, being n the number of instances and m the number of attributes. Let $\mathbf{x}_i \succeq \mathbf{x}_h$ if $\forall_{j=1,\ldots,m}, \mathbf{x}_{i,j} \geq \mathbf{x}_{h,j}$. A data set $\{\mathbf{x}_i, \text{class}(\mathbf{x}_i)\}$ is monotonic if and only if all the pairs of examples i, h are monotonic with respect to each other[13] (see Equation 1).

$$\mathbf{x}_i \succeq \mathbf{x}_h \implies \text{class}(\mathbf{x}_i) \geq \text{class}(\mathbf{x}_h), \forall_{i,h} \tag{1}$$

Monotonic Classification

Ejemplo

Conjunto de 30 muestras artificiales definidas por dos atributos (A1 y A2) teniendo como pertenencia de las muestras a 5 clases diferentes:



Nuevos problemas de clasificación

- MIL: Multi-instance learning
- ML: Multi-label classification
- Monotonic Classification
- **Semisupervised Learning**

Knowl Inf Syst (2015) 42:245–284
DOI 10.1007/s10115-013-0706-y

SURVEY PAPER

Self-labeled techniques for semi-supervised learning: taxonomy, software and empirical study

Isaac Triguero · Salvador García · Francisco Herrera

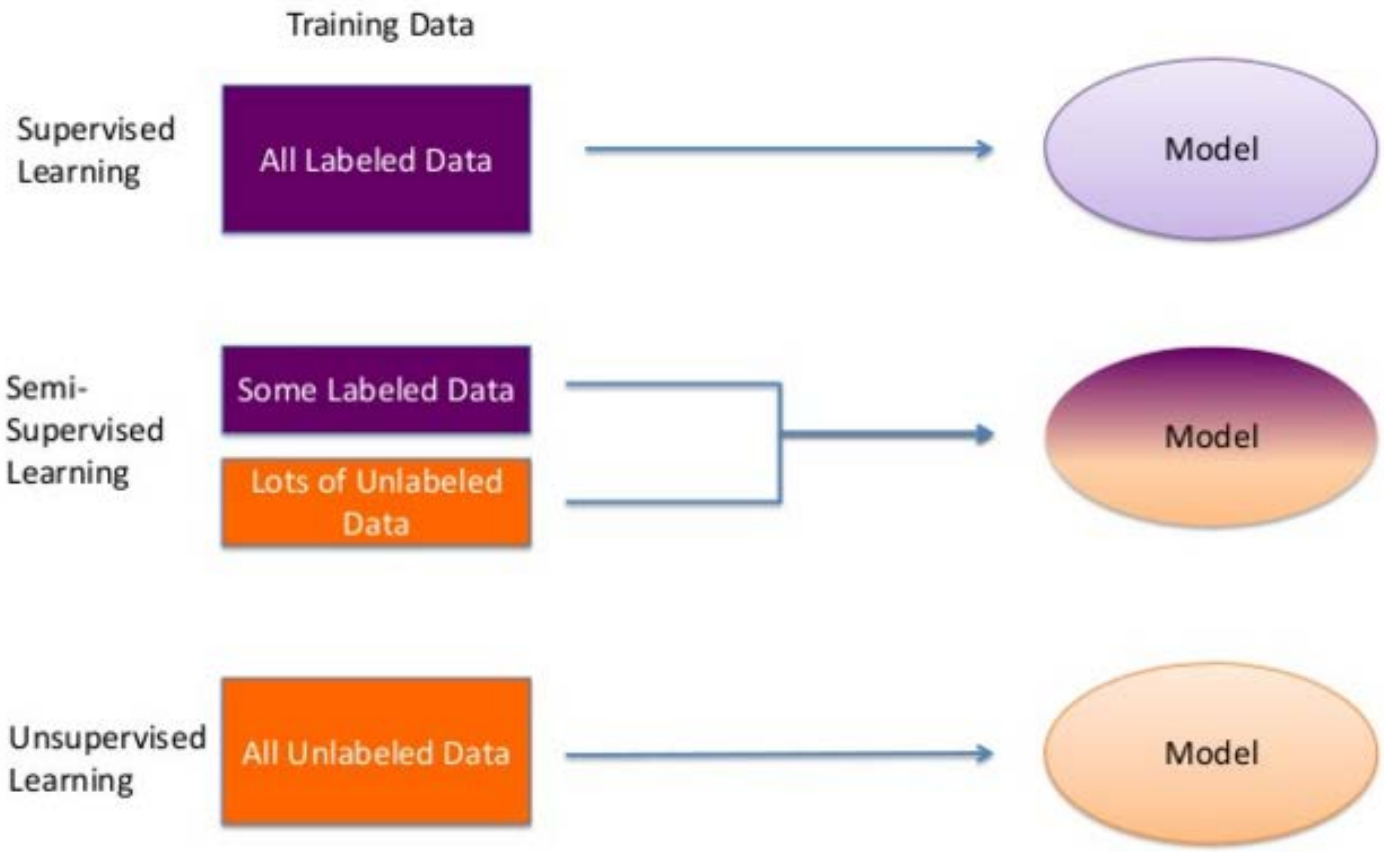
Semisupervised Learning

SSL is a learning paradigm concerned with the design of models in the presence of both labeled and unlabeled data. Essentially, SSL methods use unlabeled samples to either modify or reprioritize the hypothesis obtained from labeled samples alone.

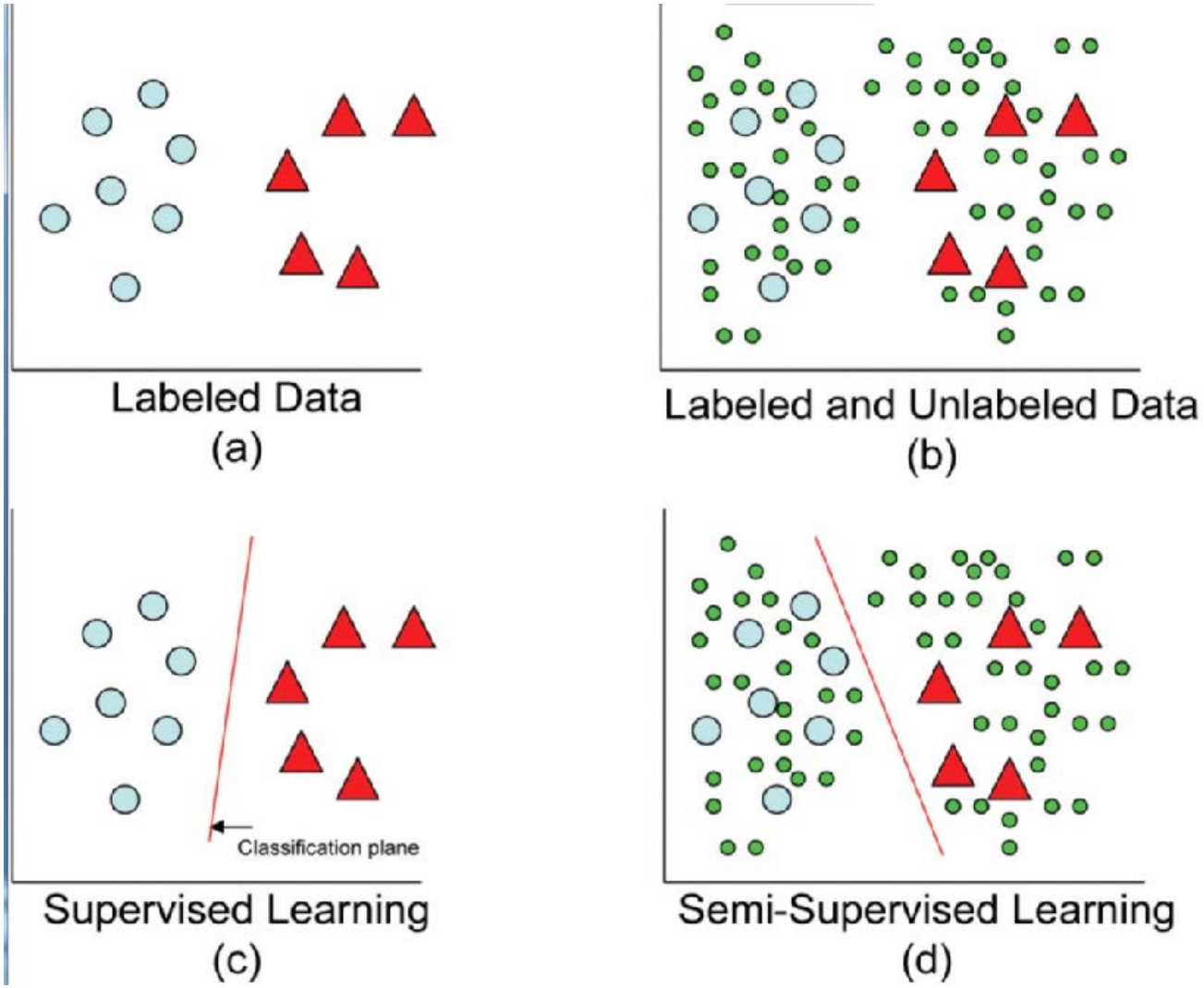
SSL is an extension of unsupervised and supervised learning by including additional information typical of the other learning paradigm.

A successful methodology to tackle the SSC problem is based on traditional supervised classification algorithms. These techniques aim to obtain one (or several) enlarged labeled set(s), based on their most confident predictions, to classify unlabeled data. We denote these algorithms self-labeled techniques.

Semisupervised Learning



Semisupervised Learning



http://bioinformatics.oxfordjournals.org/content/24/6/783/F1.large.jpg

Nuevos problemas de clasificación

- Others: Multi-view learning

Neural Comput & Applic (2013) 23:2031–2038
DOI 10.1007/s00521-013-1362-6

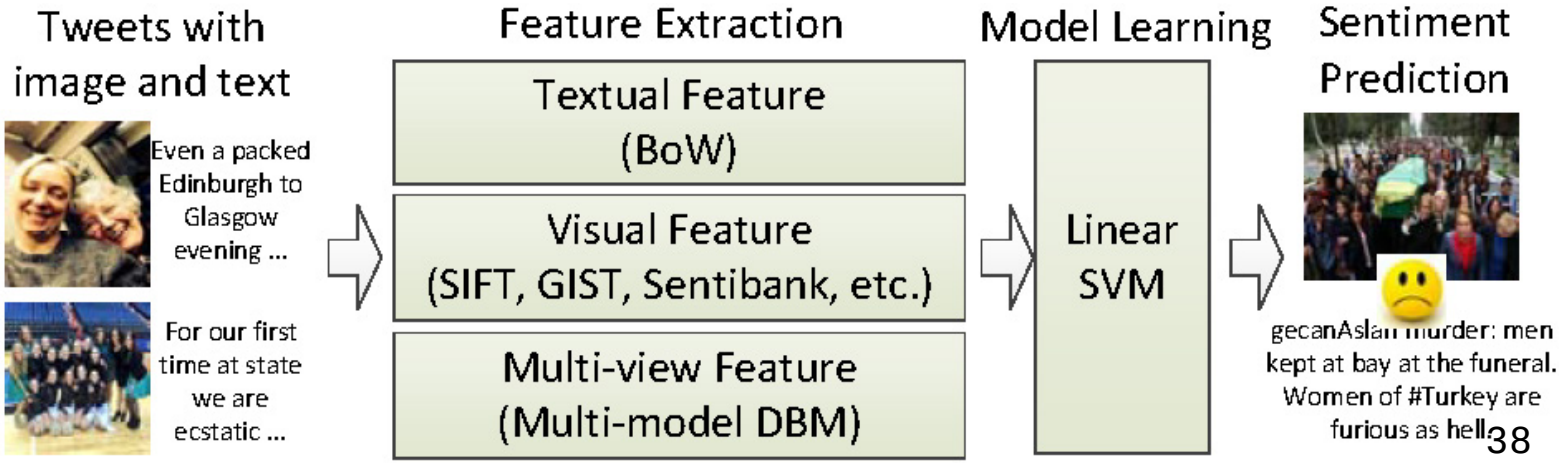
INVITED REVIEW

A survey of multi-view machine learning

Shiliang Sun

Multi-view learning is concerned with the problem of machine learning from data represented by multiple distinct feature sets.

Example: http://www.mcrlab.net/wp-content/uploads/2015/08/framework.jpg
multi-view sentiment analysis.



Nuevos problemas de clasificación

- Others: **Weak supervision and other non-standard classification problems: A taxonomy**

Pattern Recognition Letters 69 (2016) 49–55

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Weak supervision and other non-standard classification problems: A taxonomy☆

Jerónimo Hernández-González*, Iñaki Inza, Jose A. Lozano

Intelligent Systems Group, University of the Basque Country UPV/EHU, P. Manuel Lardizabal 1, 20018 Donostia, Spain

Comentario final: Estas son algunas de las extensiones al problema clásico de clasificación.

Muchas aparecen como consecuencia de nuevos problemas reales que requieren de un nuevo planteamiento de clasificación.

Inteligencia de Negocio

TEMA 7. Modelos Avanzados de Minería de Datos

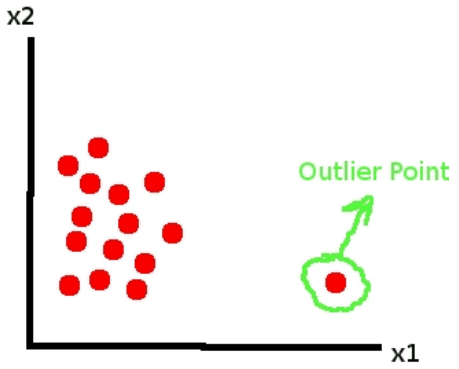
1. Clases no balanceadas/equilibradas
2. Características intrínsecas de los datos en clasificación
3. Problemas no estándar de clasificación: MIL, MLL, SSL…
4. **Detección de anomalías**
5. Deep Learning
6. Análisis de Sentimientos

Outline

- What are anomalies?
- Anomaly Detection: Taxonomy
- Nearest Neighbor Based Techniques
- One-Class to tackle the Fault Detection
- Concluding Remarks

What are anomalies?

- Anomaly is a pattern in the data that does not conform to the expected behavior
- Also referred to as outliers, exceptions, peculiarities, surprise, etc.
- Anomalies translate to significant (often critical) real life entities
  - Cyber intrusions
  - Credit card fraud
  - Faults in a System



What are anomalies?

Real World Anomalies

- Credit Card Fraud
  - An abnormally high purchase made on a credit card

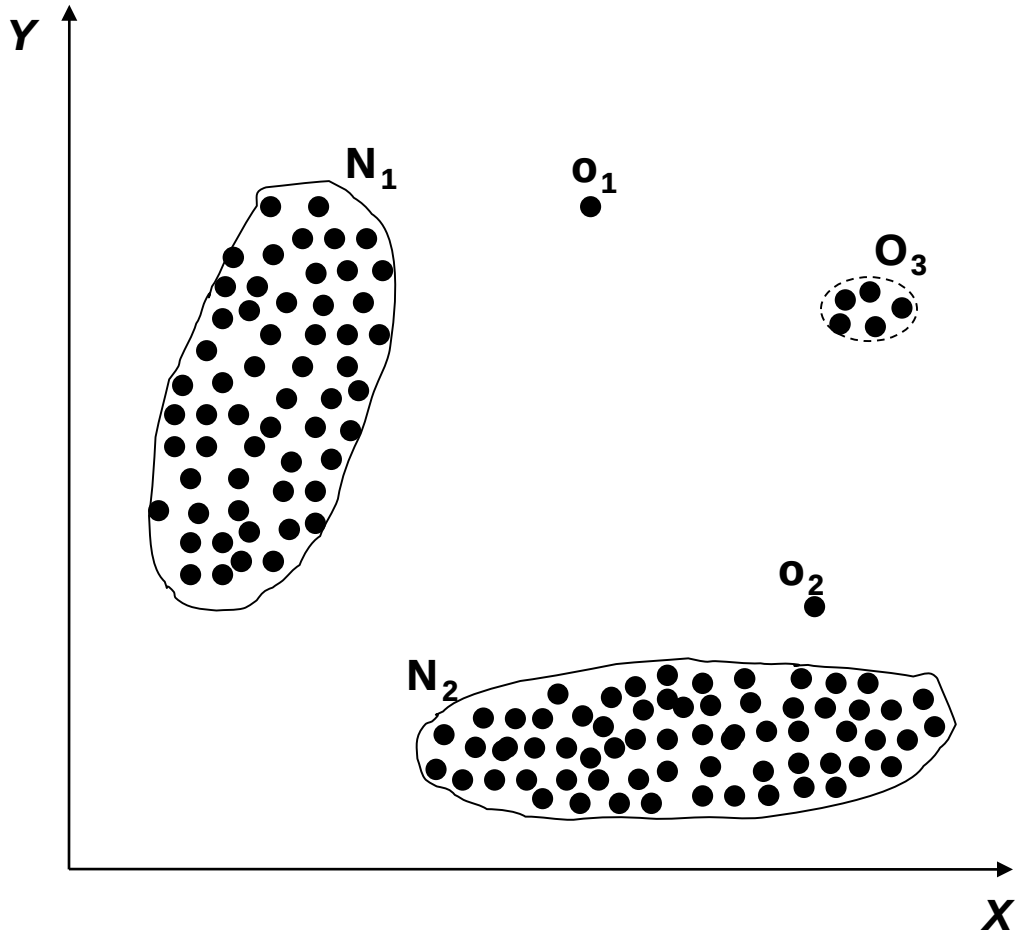
- Cyber Intrusions
  - A web server involved in *ftp* traffic

- Faults in a system
  - An abnormal values from sensors

What are anomalies?

Simple Example

- N_1 and N_2 are regions of normal behavior
- Points o_1 and o_2 are anomalies
- Points in region O_3 are anomalies



What are anomalies?

Related problems

- Rare Class Mining (high imbalanced classes)
- Chance discovery
- Novelty Detection
- Exception Mining
- Noise Removal
- Black Swan*

* N. Talleb, The Black Swan: The Impact of the Highly Probable?, 2007

What are anomalies?

Key Challenges

- Defining a representative normal region is challenging
- The boundary between normal and outlying behavior is often not precise
- The exact notion of an outlier is different for different application domains
- Availability of labeled data for training/validation
- Malicious adversaries
- Data might contain noise
- Normal behavior keeps evolving

What are anomalies?

Aspects of Anomaly Detection Problem

- Nature of input data
- Availability of supervision
- Type of anomaly: point, contextual, structural
- Output of anomaly detection
- Evaluation of anomaly detection techniques

What are anomalies?

Type of Anomaly

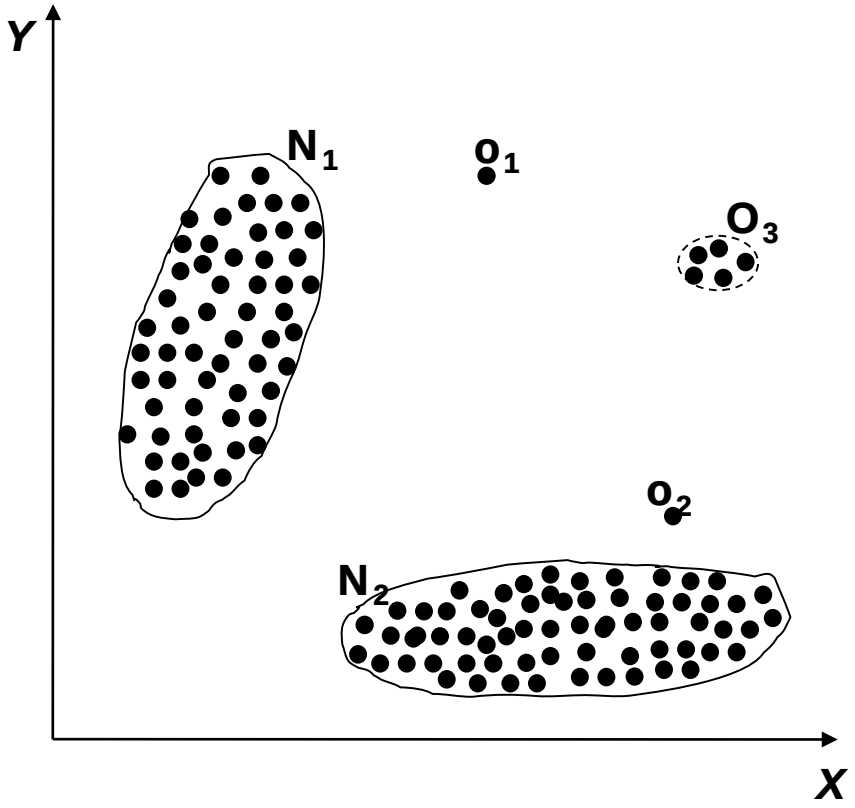
- Point Anomalies
- Contextual Anomalies
- Collective Anomalies

V. CHANDOLA, A. BANERJEE, and VI. KUMAR. **Anomaly Detection: A Survey**
ACM Computing Surveys, Vol. 41, No. 3, Article 15, Publication date: July 2009.
http://doi.acm.org/10.1145/ 1541880.1541882

What are anomalies?

Point Anomalies

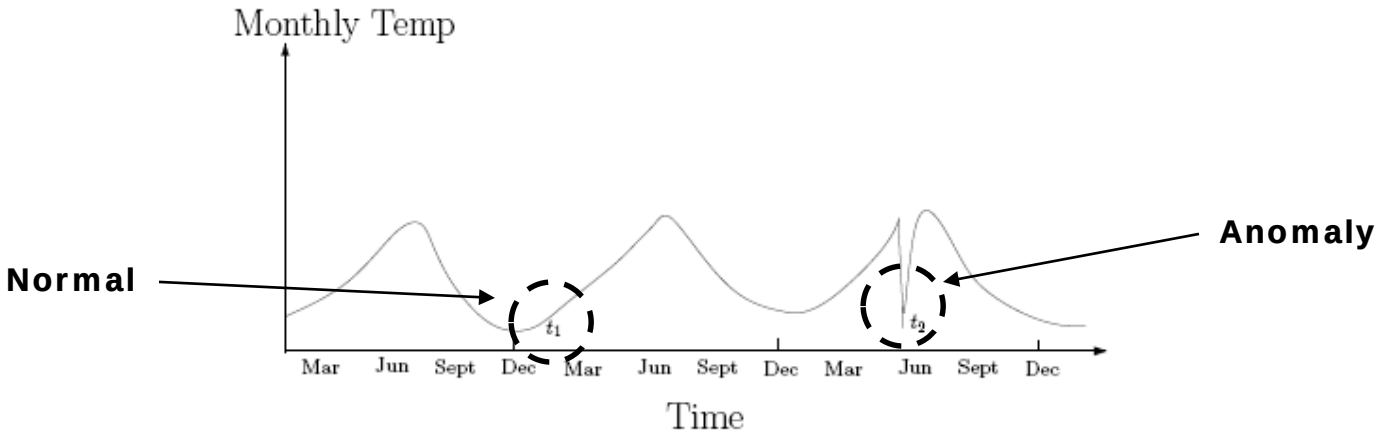
- An individual data instance is anomalous w.r.t. the data



What are anomalies?

Contextual Anomalies

- An individual data instance is anomalous within a context
- Requires a notion of context
- Also referred to as conditional anomalies*

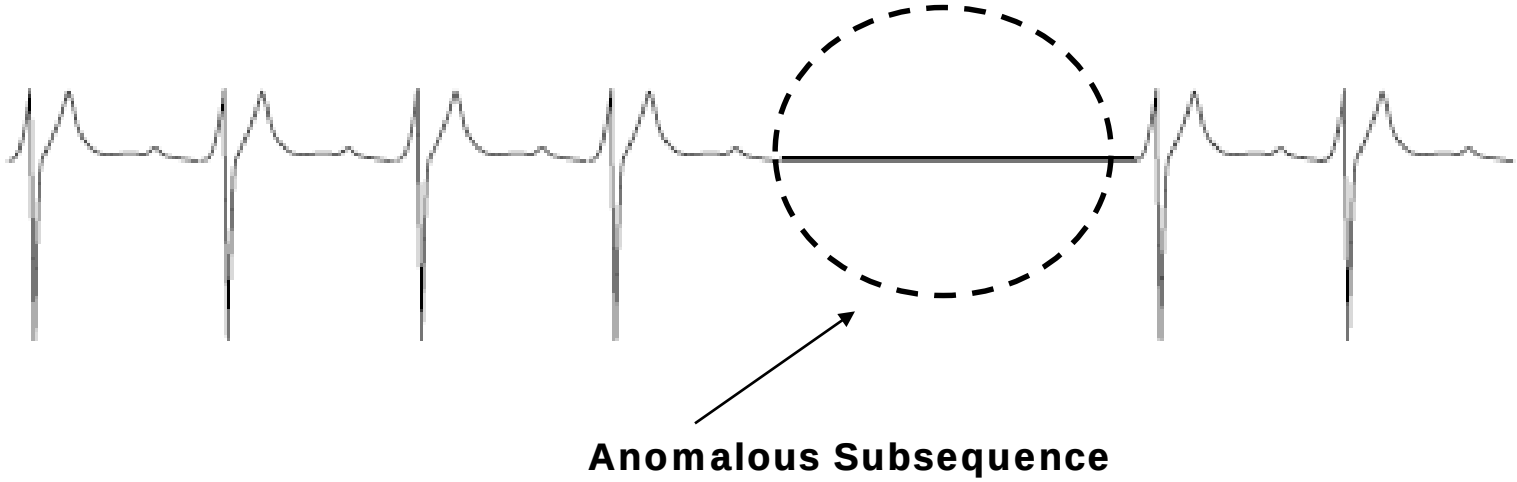


* Xiuyao Song, Mingxi Wu, Christopher Jermaine, Sanjay Ranka, Conditional Anomaly Detection, IEEE Transactions on Data and Knowledge Engineering, 2006.

What are anomalies?

Collective Anomalies

- A collection of related data instances is anomalous
- Requires a relationship among data instances
  - Sequential Data
  - Spatial Data
  - Graph Data
- The individual instances within a collective anomaly are not anomalous by themselves



What are anomalies?

Applications of Anomaly Detection

- Network intrusion detection
- Insurance / Credit card fraud detection
- Healthcare Informatics / Medical diagnostics
- Industrial Damage Detection
- Image Processing / Video surveillance
- Novel Topic Detection in Text Mining
- ...

What are anomalies?

Industrial Damage Detection

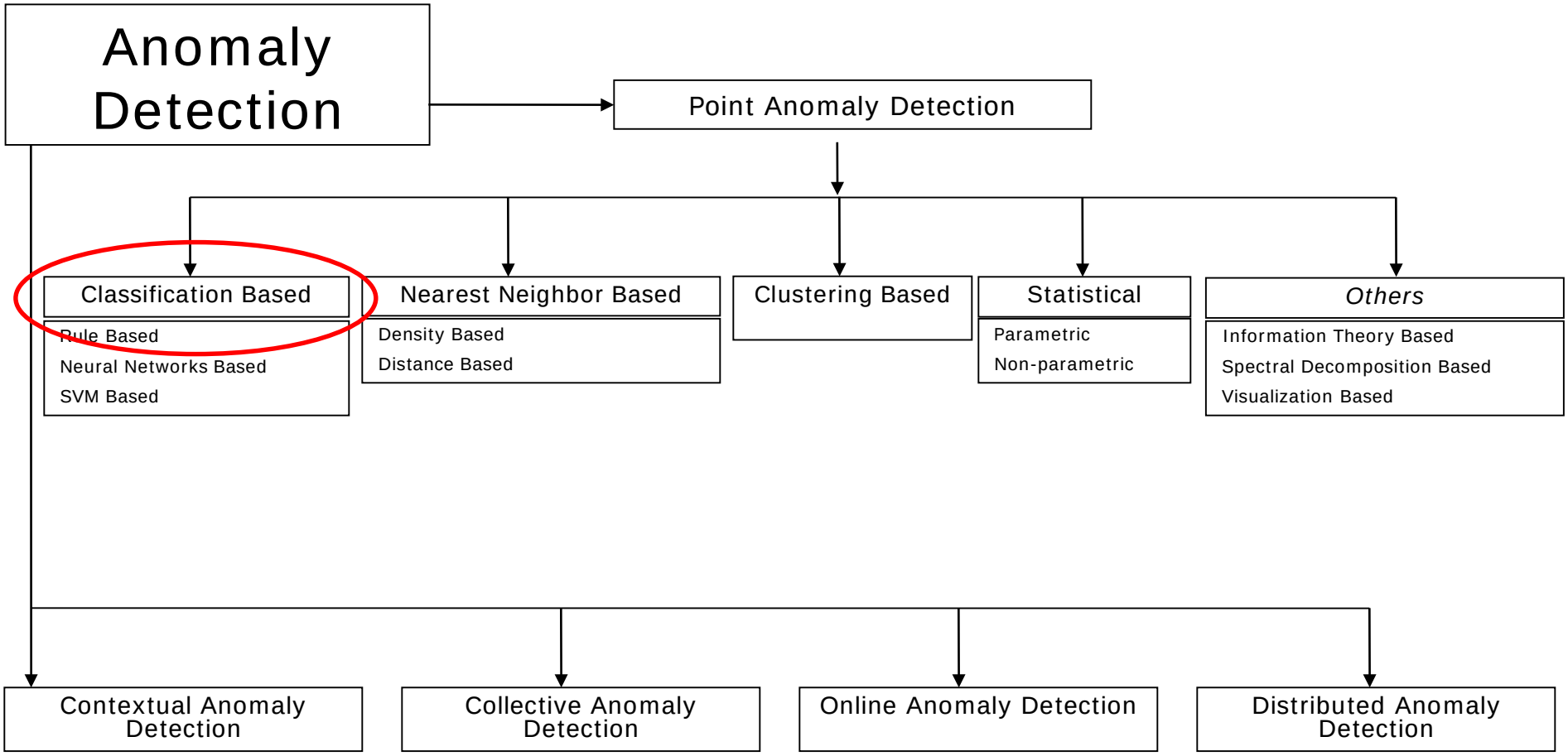
- Industrial damage detection refers to detection of different faults and failures in complex industrial systems, structural damages, intrusions in electronic security systems, suspicious events in video surveillance, abnormal energy consumption, etc.
  - Example: Wind Turbines
    - Fault detection / Anomalies in performance
  - Example: Aircraft Safety
    - Anomalous Aircraft (Engine) / Fleet Usage
    - Anomalies in engine combustion data
    - Total aircraft health and usage management

- **Key Challenges**
  - **Data is extremely huge, noisy and unlabelled**
  - **Most of applications exhibit temporal behaviour**
  - **Detecting anomalous events typically require immediate intervention**

Outline

- What are anomalies?
- Anomaly Detection: Taxonomy
- Nearest Neighbor Based Techniques
- One-Class to tackle the Fault Detection
- Concluding Remarks

Anomaly Detection: Taxonomy



* Anomaly Detection – A Survey, Varun Chandola, Arindam Banerjee, and Vipin Kumar, ACM Computing Surveys, Vol. 41, No. 3, Article 15, Publication date: July 2009.

Anomaly Detection: Taxonomy

Classification Based Techniques

- **Main idea:** build a classification model for normal (and anomalous, rare) events based on labeled training data, and use it to classify each new unseen event
- Classification models must be able to handle skewed (imbalanced) class distributions
- Categories:
  - *Supervised classification techniques*
    - Require knowledge of both ***normal*** and ***anomaly*** class
    - Build classifier to distinguish between normal and known anomalies
  - *Semi-supervised classification techniques*
    - Require knowledge of ***normal*** class only!
    - Use modified classification model to learn the normal behavior and then detect any deviations from normal behavior as anomalous

Anomaly Detection: Taxonomy

Classification Based Techniques

- **Advantages:**
  - ***Supervised classification techniques***
    - Models that can be easily understood
    - High accuracy in detecting many kinds of known anomalies
  - ***Semi-supervised classification techniques***
    - Models that can be easily understood
    - Normal behavior can be accurately learned
- **Drawbacks:**
  - ***Supervised classification techniques***
    - Require both labels from both normal and anomaly class
    - Cannot detect unknown and emerging anomalies
  - ***Semi-supervised classification techniques***
    - Require labels from normal class
    - Possible high false alarm rate - previously unseen (yet legitimate) data records may be recognized as anomalies

Anomaly Detection: Taxonomy

Supervised Classification Techniques

- Rule based techniques
- Model based techniques
  - Neural network based approaches
  - Support Vector machines (SVM) based approaches
  - Bayesian networks based approaches
- Imbalanced classification
  - Manipulating data records (oversampling / undersampling / generating artificial examples)
  - Cost-sensitive classification techniques
  - Ensemble based algorithms (SMOTEBoost, RareBoost)

Anomaly Detection: Taxonomy

Rule Based Techniques

- **Creating new rule based algorithms**
- **Adapting existing rule based techniques**
  - Robust C4.5 algorithm [John95]
  - Adapting multi-class classification methods to single-class classification problem
- **Association rules**
  - Rules with support higher than pre specified threshold may characterize normal behavior
  - Anomalous data record occurs in fewer frequent itemsets compared to normal data record
  - Frequent episodes for describing temporal normal behavior [Lee00,Qin04]
- **Case specific feature / rule weighting**
  - Increasing the rule strength for all rules describing the rare class or features strength for highlighting the minority class.

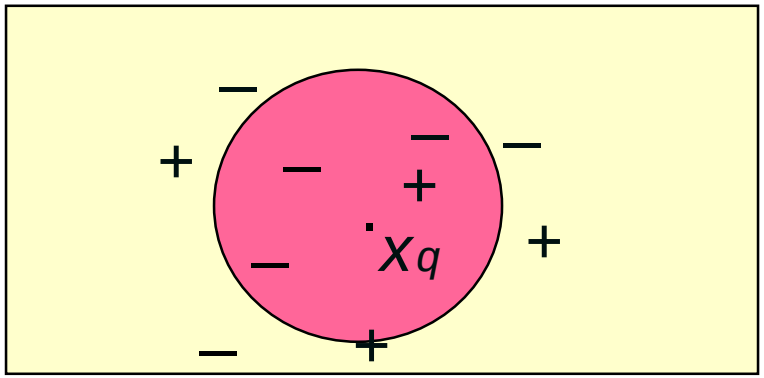
Outline

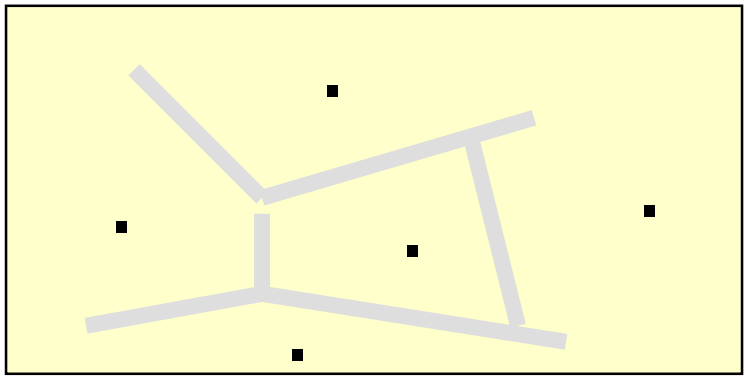
- What are anomalies?
- Anomaly Detection: Taxonomy
- Nearest Neighbor Based Techniques
- One-Class to tackle the Fault Detection
- Concluding Remarks

Nearest Neighbor Based Techniques

K Nearest Neighbor (KNN)

- All instances correspond to points in the n-D space.
- The nearest neighbor are defined in terms of Euclidean distance.
- The target function could be discrete- or real- valued.
- For discrete-valued, the *k*-NN returns the most common value among the k training examples nearest to x_q.
- Voronoi diagram: the decision surface induced by 1-NN for a typical set of training examples.





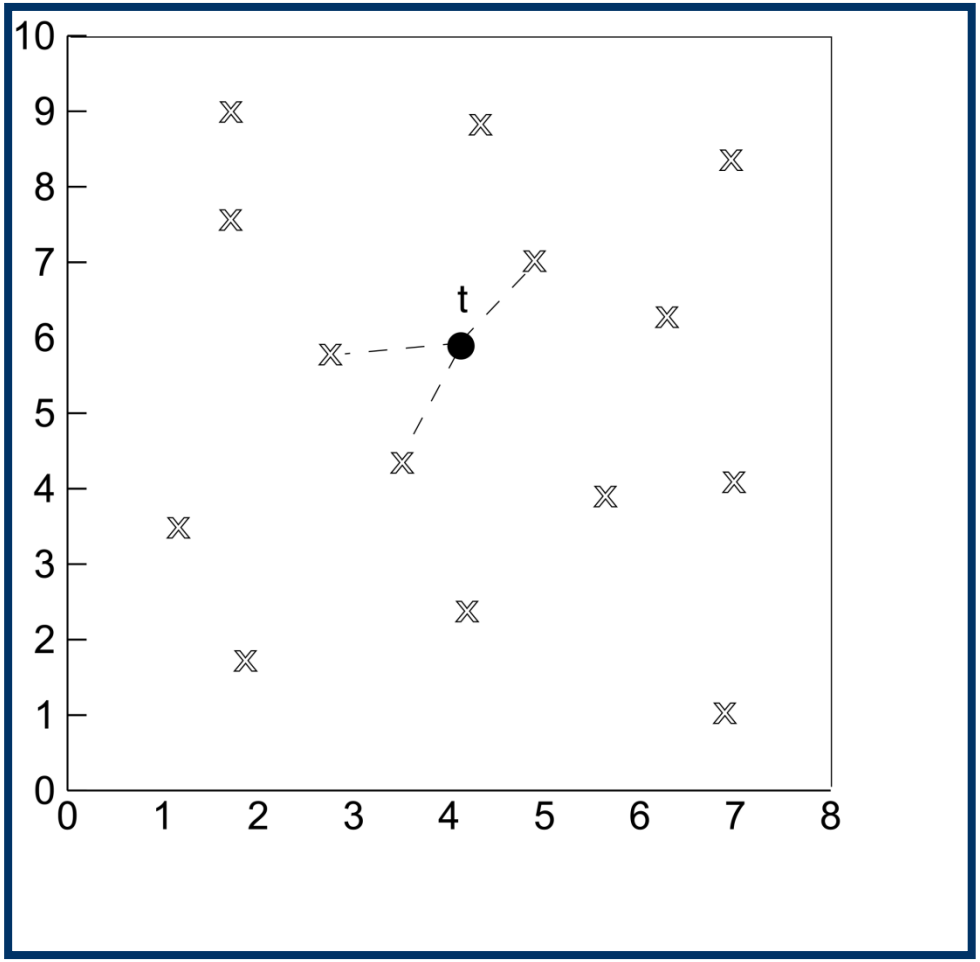
Nearest Neighbor Based Techniques

K Nearest Neighbor (KNN)

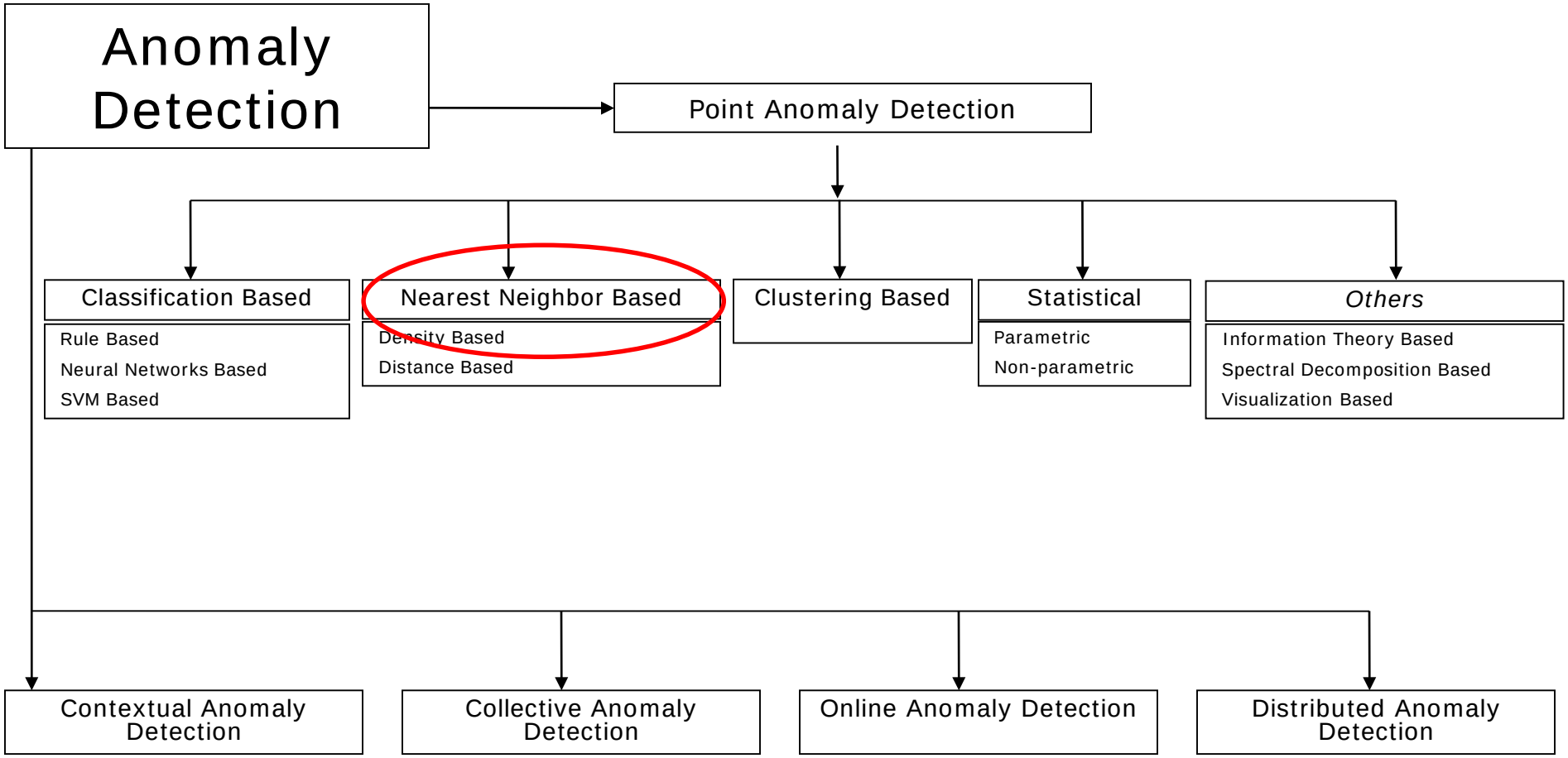
- Training set includes classes.
- Examine K items near item to be classified.
- New item placed in class with the most number of close items.
- O(q) for each tuple to be classified. (Here q is the size of the training set.)

Nearest Neighbor Based Techniques

K Nearest Neighbor (KNN)



Nearest Neighbor Based Techniques



* Anomaly Detection – A Survey, Varun Chandola, Arindam Banerjee, and Vipin Kumar, ACM Computing Surveys, Vol. 41, No. 3, Article 15, Publication date: July 2009.

Nearest Neighbor Based Techniques

- ***Key assumption***: normal points have close neighbors while anomalies are located far from other points
- General two-step approach
  1. Compute neighborhood for each data record
  2. Analyze the neighborhood to determine whether data record is anomaly or not
- **Categories:**
  - Distance based methods
    - Anomalies are data points most distant from other points
  - Density based methods
    - Anomalies are data points in low density regions

Nearest Neighbor Based Techniques

- **Advantage**
  - Can be used in unsupervised or semi-supervised setting (do not make any assumptions about data distribution)
- **Drawbacks**
  - If normal points do not have sufficient number of neighbors the techniques may fail
  - Computationally expensive
  - In high dimensional spaces, data is sparse and the concept of similarity may not be meaningful anymore. Due to the sparseness, distances between any two data records may become quite similar => Each data record may be considered as potential outlier!

Nearest Neighbor Based Techniques

- **Distance based approaches**
  - A point *O* in a dataset is an *DB*(*p*, *d*) outlier if at least fraction *p* of the points in the data set lies greater than distance *d* from the point *O**
- **Density based approaches**
  - Compute local densities of particular regions and declare instances in low density regions as potential anomalies
  - Approaches
    - Local Outlier Factor (LOF)
    - Connectivity Outlier Factor (COF)
    - Multi-Granularity Deviation Factor (MDEF)

Nearest Neighbor Based Techniques

Distance based Outlier Detection

- *Nearest Neighbor (NN) approach*
  - For each data point *d* compute the distance to the *k-th* nearest neighbor d_k
  - Sort all data points according to the distance d_k
  - Outliers are points that have the largest distance d_k and therefore are located in the more sparse neighborhoods
  - Usually data points that have top *n*% distance d_k are identified as outliers
    - *n* – user parameter
  - Not suitable for datasets that have modes with varying density

Nearest Neighbor Based Techniques

Density Based Approaches: Local Outlier Factor (LOF)

- For each data point *q* compute the distance to the *k*-th nearest neighbor (*k-distance*)
- Compute *reachability distance* (*reach-dist*) for each data example *q* with respect to data example *p* as:

$$reach\text{-}dist_k(q, p) = \max\{k\text{-}distance(p), d(q,p)\}$$

- Compute *local reachability density* (*lrd*) of data example *q* as inverse of the average reachabaility distance based on the *MinPts* nearest neighbors of data example *q*

$$lrd(q) = \frac{MinPts}{\sum_{p} reach\_dist_{MinPts}(q, p)}$$

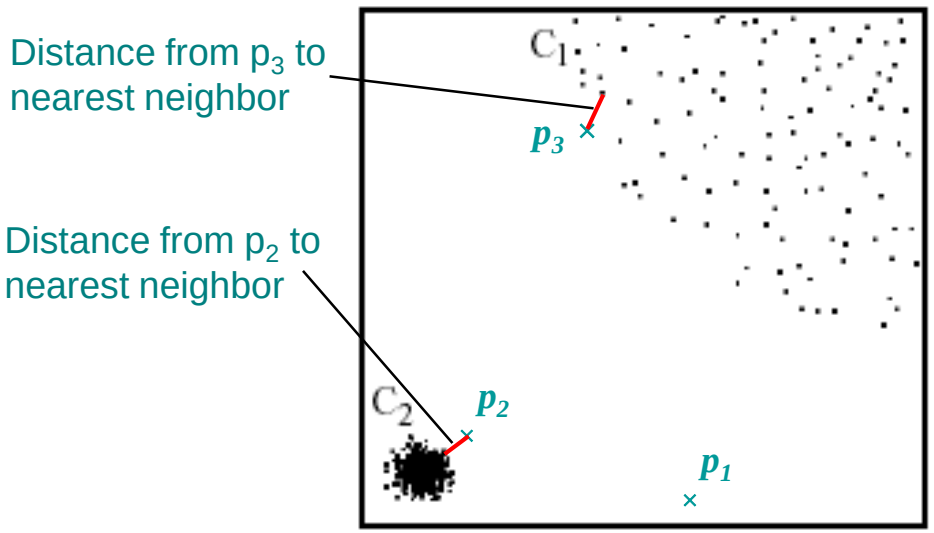
- Compute *LOF*(*q*) as ratio of average local reachability density of *q*'s *k*-nearest neighbors and local reachability density of the data record *q*

$$LOF(q) = \frac{1}{MinPts} \cdot \sum_{p} \frac{lrd(p)}{lrd(q)}$$

Nearest Neighbor Based Techniques

Advantages of Density based Techniques

- *Local Outlier Factor (LOF) approach*
  - Example:



In the *NN* approach, p_2 is not considered as outlier, while the *LOF* approach finds both p_1 and p_2 as outliers

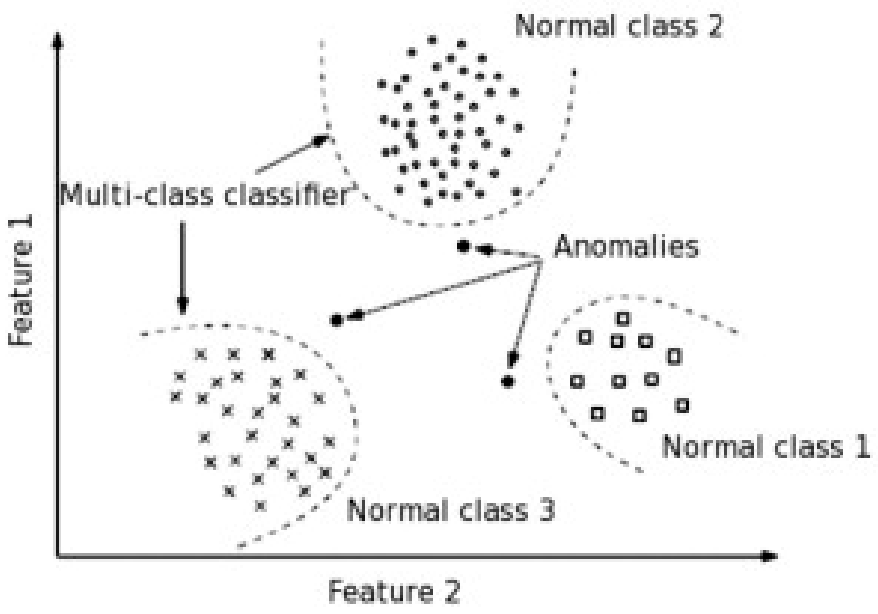
NN approach may consider p_3 as outlier, but LOF approach does not

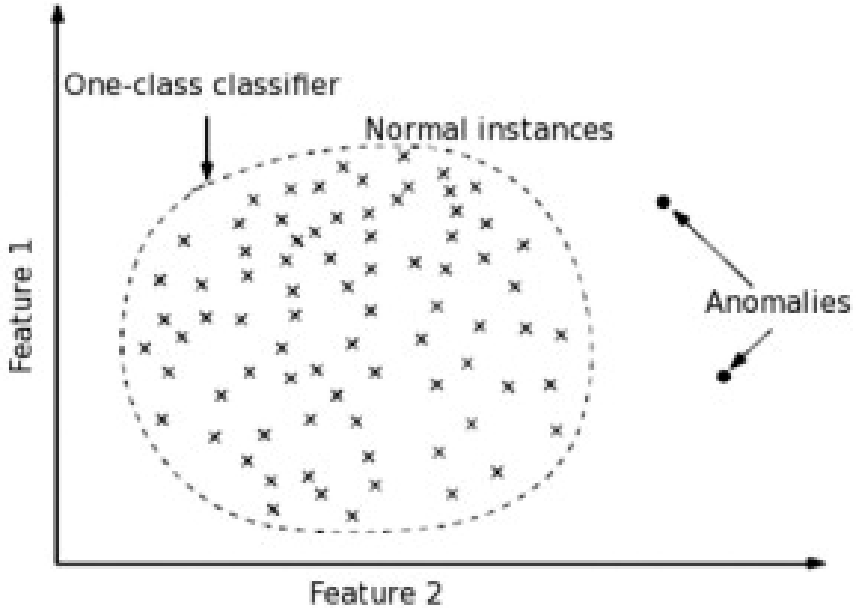
Outline

- What are anomalies?
- Anomaly Detection: Taxonomy
- Nearest Neighbor Based Techniques
- One-Class to tackle the Fault Detection
- Concluding Remarks

One-Class to tackle the Fault Detection

Several classes vs One-class classification



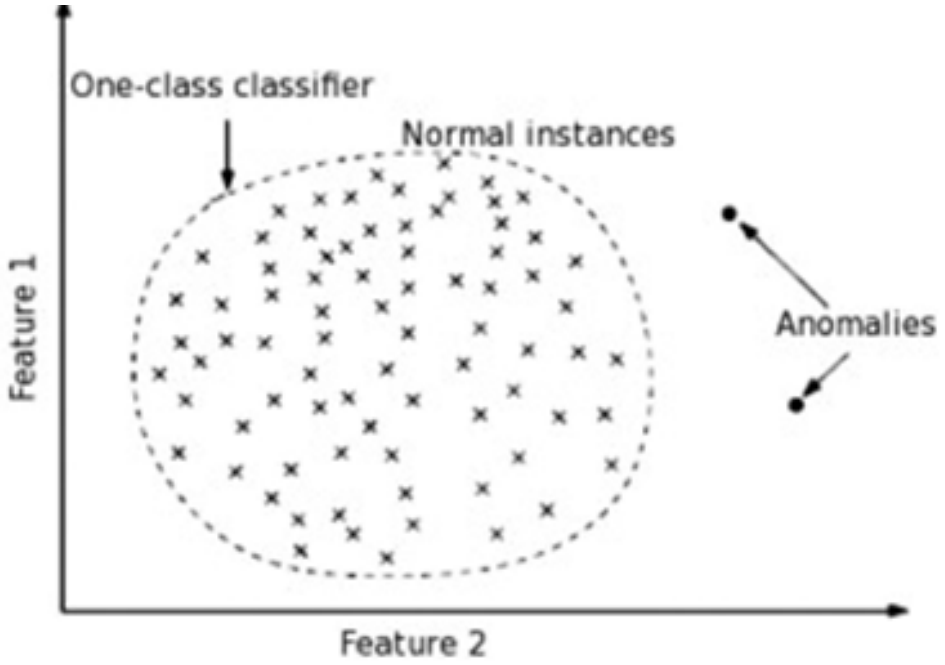


One-Class to tackle the Fault Detection

- **Advantages:** **Classification Based Techniques**
  - ***Supervised classification techniques***
    - Models that can be easily understood
    - High accuracy in detecting many kinds of known anomalies
  - ***Semi-supervised classification techniques (One-class)***
    - Models that can be easily understood
    - Normal behavior can be accurately learned
- **Drawbacks:**
  - ***Supervised classification techniques***
    - Require both labels from both normal and anomaly class
    - Cannot detect unknown and emerging anomalies
  - ***Semi-supervised classification techniques (One-class)***
    - Require labels from normal class
    - Possible high false alarm rate - previously unseen (yet legitimate) data records may be recognized as anomalies

One-Class to tackle the Fault Detection

One-class 1-NN is a semi-supervised algorithm that learns a decision function for novelty detection: classifying new data as similar or different to the training set.



One-Class to tackle the Fault Detection

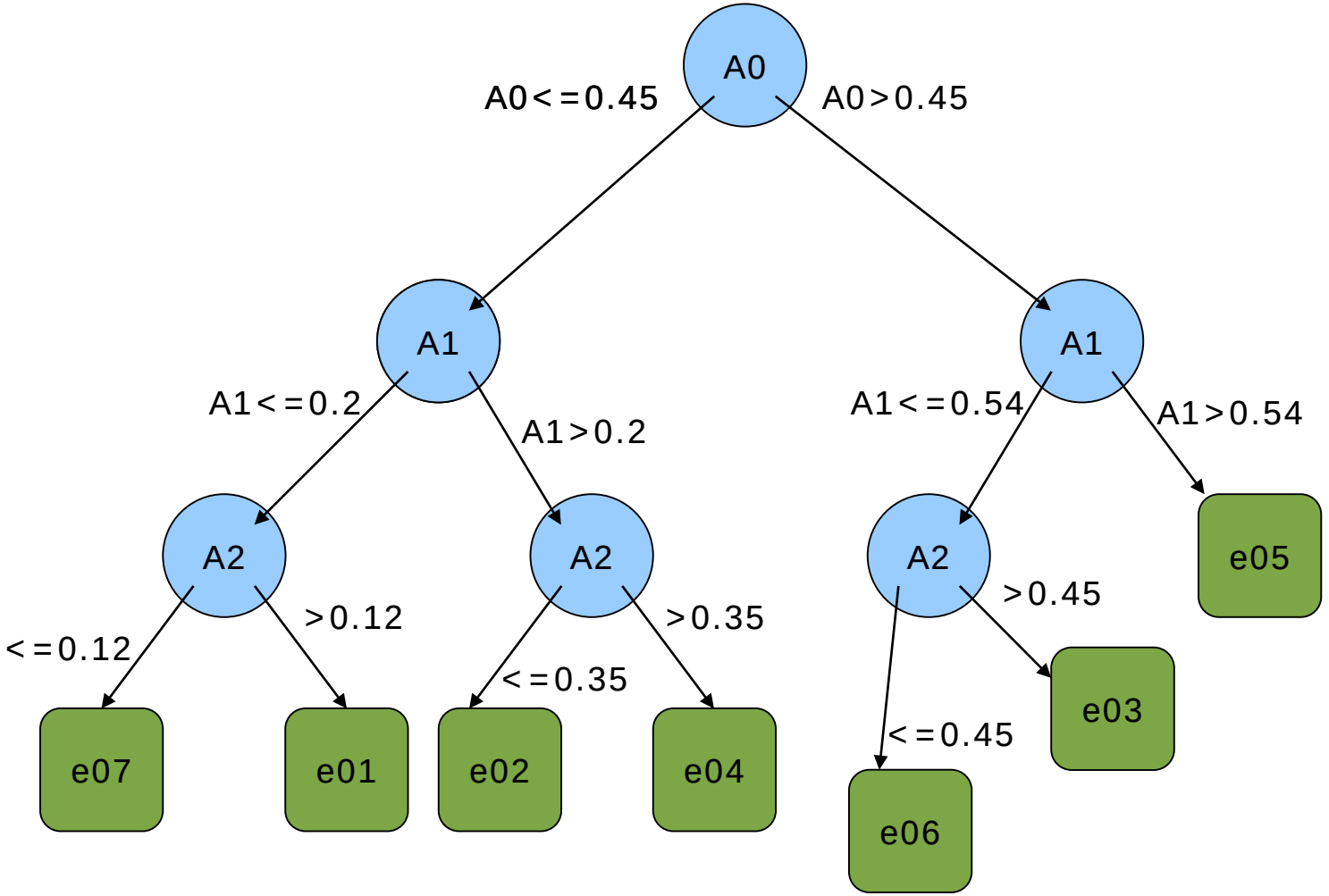
Pseudocode of one-class kNN

When a new test example A needs to be tested

1.- Find its nearest neighbor (NN), which we call B, by using a fast NN technique: **k-d tree***.
2.- The tentative class of A is the class of B.
3.- Find the nearest neighbor of B in the training set using a **k-d tree***, call it C.
4.- For each attribute *attr* in the dataset, perform the following calculations:
If (abs(A[*attr*]–B[*attr*]) > threshAttr*abs(B[*attr*]-C[*attr*])):
    Example A does not belong to any class and is considered an anomaly,
Otherwise, it is assigned to its tentative class.

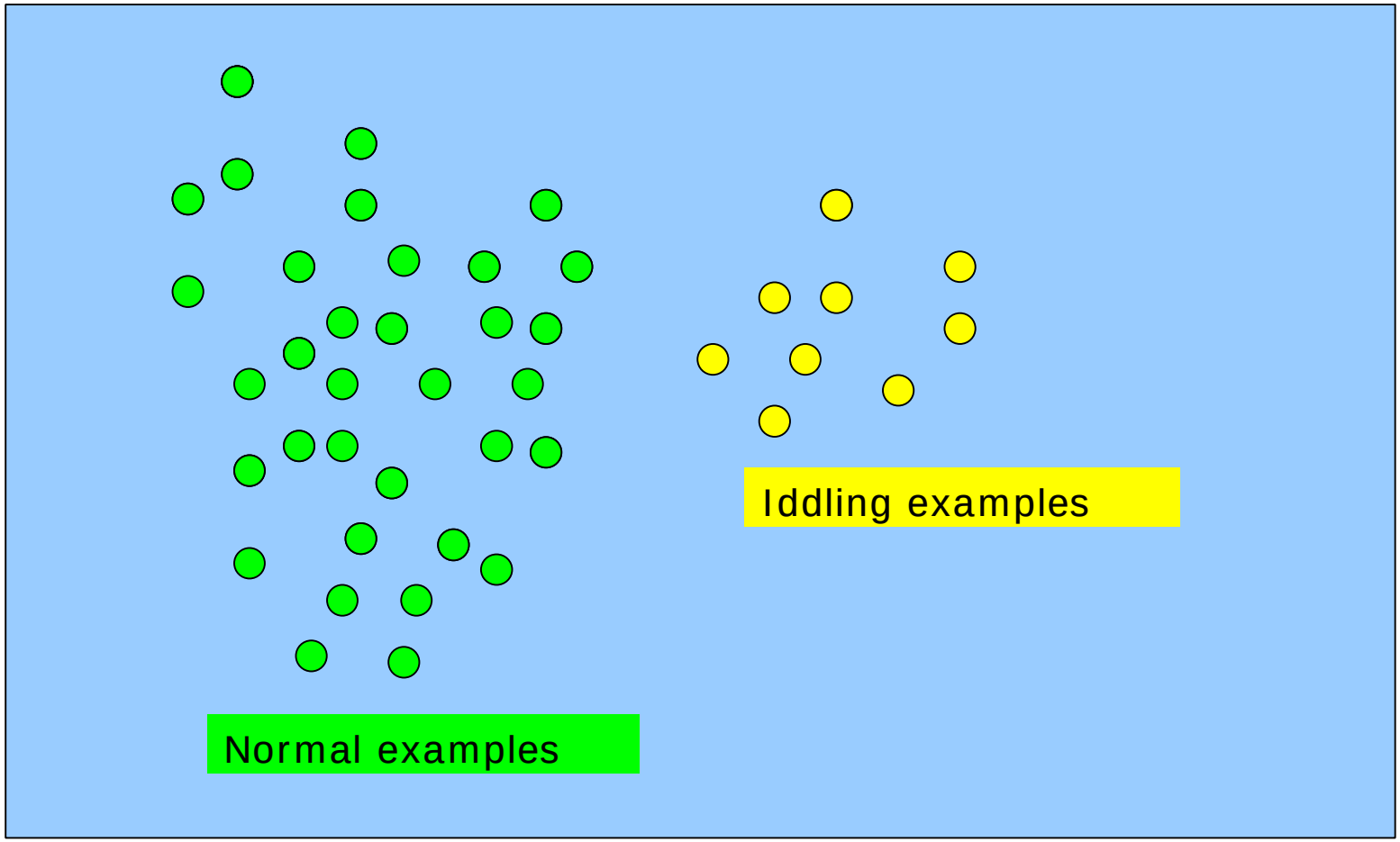
One-Class to tackle the Fault Detection

Constructing a k-d tree



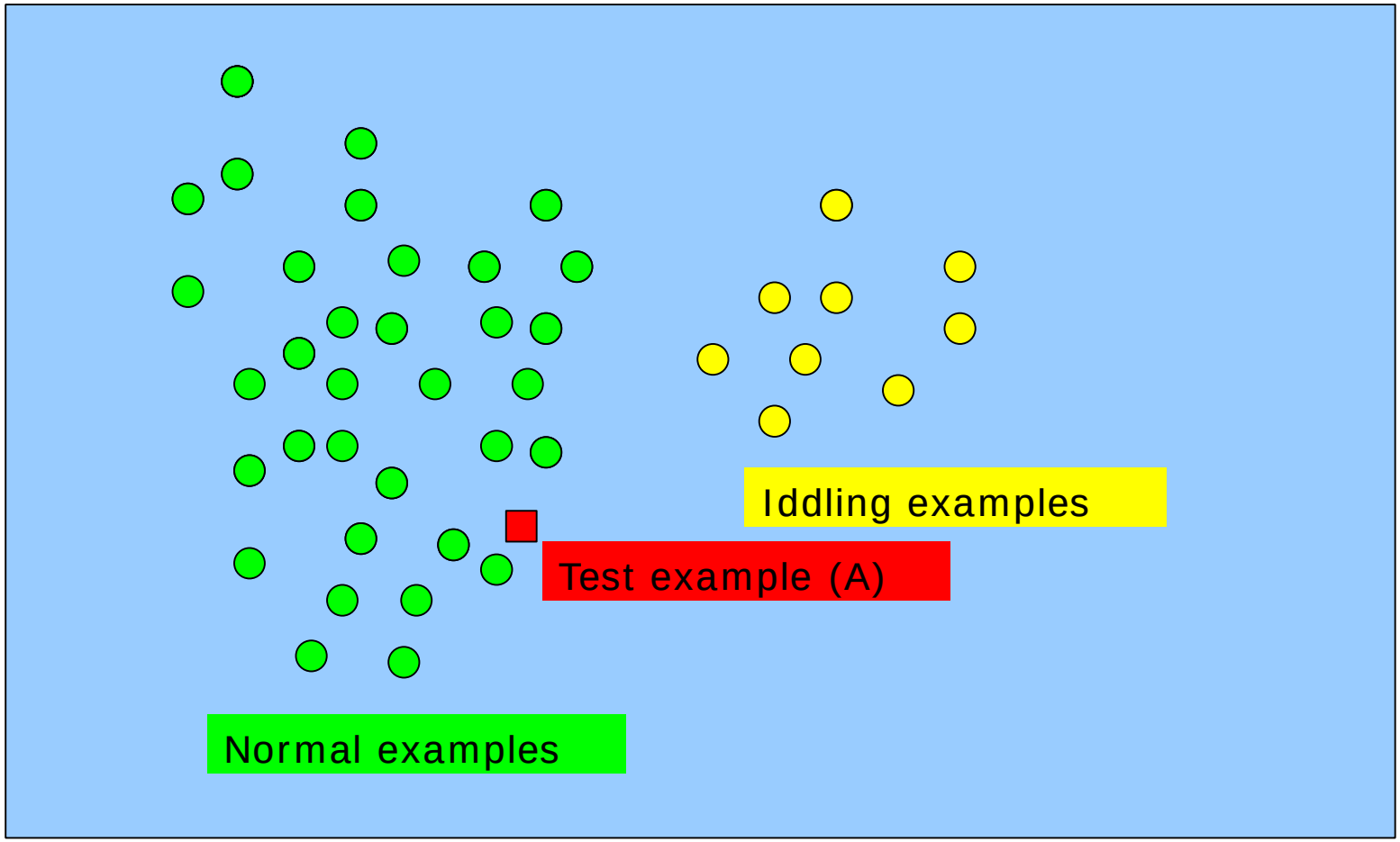
One-Class to tackle the Fault Detection

Visually: Training examples



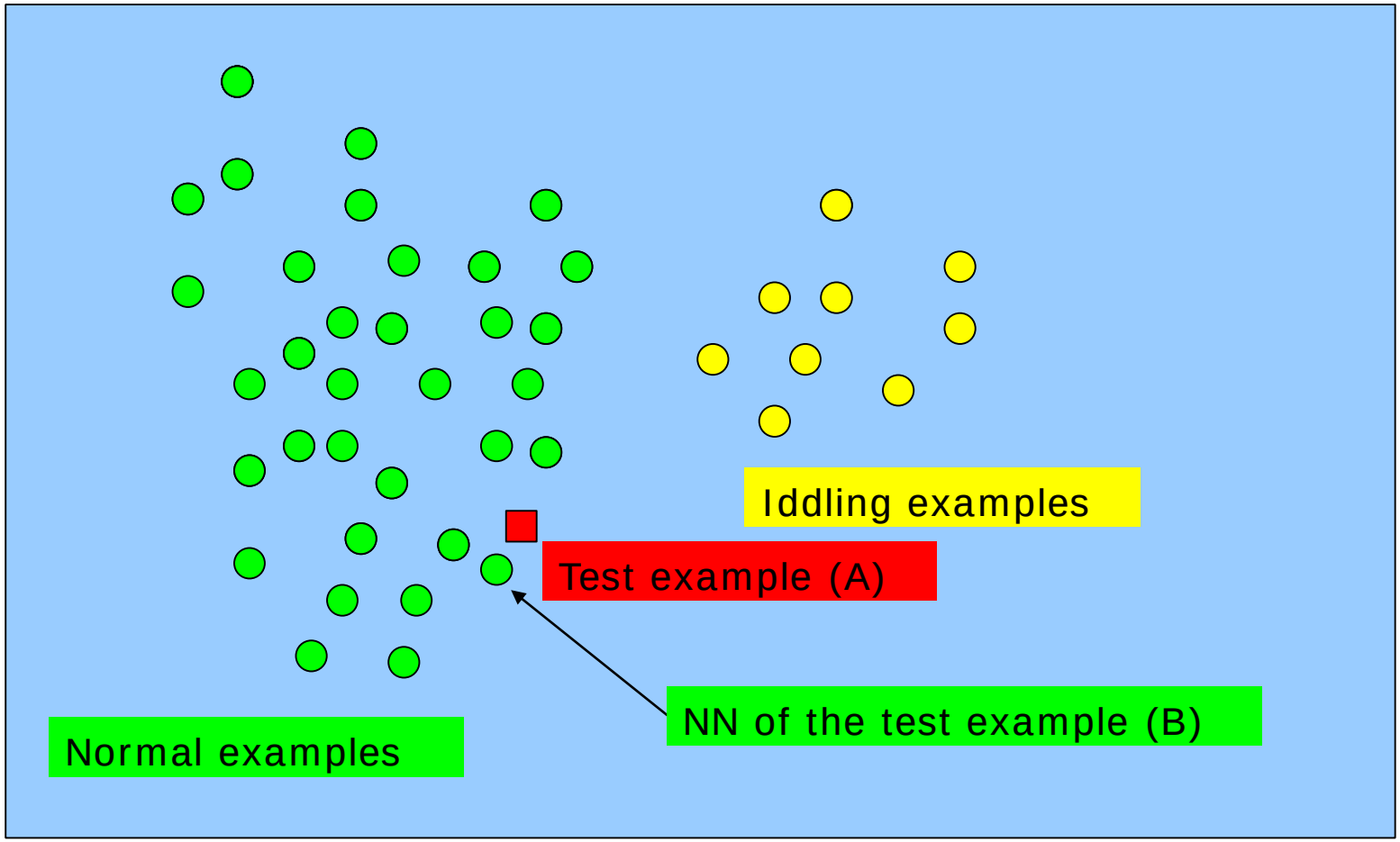
One-Class to tackle the Fault Detection

Visually: Training + 1 test example



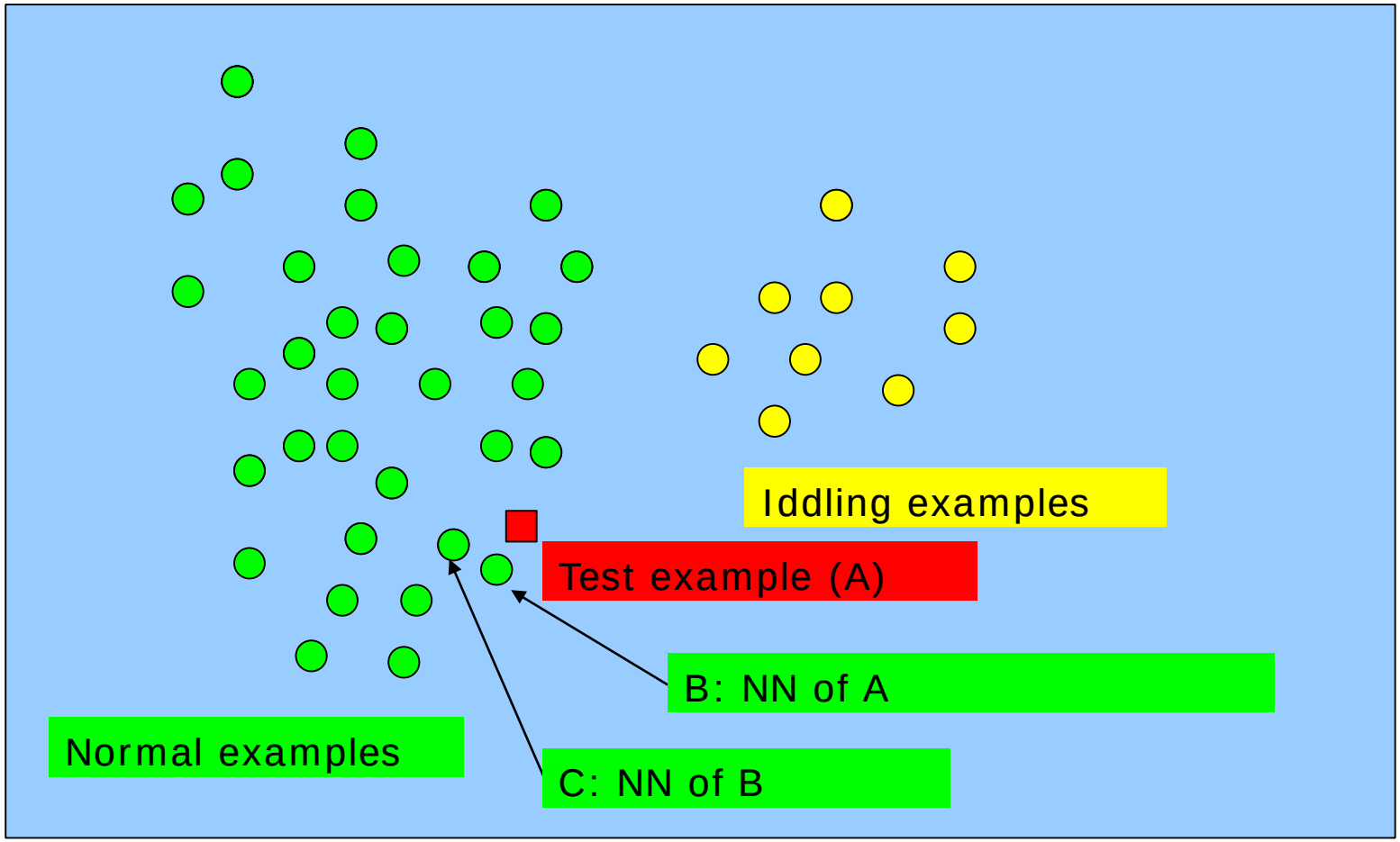
One-Class to tackle the Fault Detection

Visually: Finding the NN of the test example



One-Class to tackle the Fault Detection

Visually: Finding the NN of B (finding C)



One-Class to tackle the Fault Detection

One-class kNN: Reading the output

Test example 24199 has been found to be an anomaly

These are the values for all the attributes on the test example:

225.125 1500 364.523 41.8 42.3

Values for all attributes of example B

Its NN in training is example 57679,
223.575 1497.06 370.553 41.6 42.2

This test example is labeled as an anomaly because
Attribute 3 should be in range [364.553 , 376.553], but its actual value is 364.523

Range where the attribute should be. It is calculated as:
[B[*attr*]-threshAttr*abs(B[*attr*]-C[*attr*]), B[*attr*]+threshAttr*abs(B[*attr*]-C[*attr*])]

One-Class to tackle the Fault Detection

Brief tutorial on k-d trees

Basic idea: binary tree where each node splits the data in two subgroups with roughly half the size (divide and conquer)

How? Take an attribute, split the data points by the median value: The examples with value under or equal to the median are placed on the subtree to one side, those with values over the median go to the subtree on the other side.

The size of the tree is O(n), the average time to find a match (a Nearest Neighbor, the process is explained in the next slide) is O(log(n)). In this context, n refers to the number of examples in the training set.

The time to find a match is on average O(log(n)) only when the k-d tree works well.
For a k-d tree to work well, the number of examples must be much larger than the number of attributes (n should be >= 2^{nAttr}), and said examples should be approximately randomly distributed.

One-Class to tackle the Fault Detection

Constructing a k-d tree (I)

Example data (each row corresponds to an example, each column is an attribute):

Example ID	A0	A1	A2	A3
e01	0.10	0.06	0.20	0.30
e02	0.30	0.33	0.35	0.51
e03	0.50	0.65	0.54	0.45
e04	0.45	0.14	0.56	0.89
e05	0.52	0.17	0.67	0.64
e06	0.53	0.40	0.45	0.11
e07	0.29	0.54	0.12	0.54

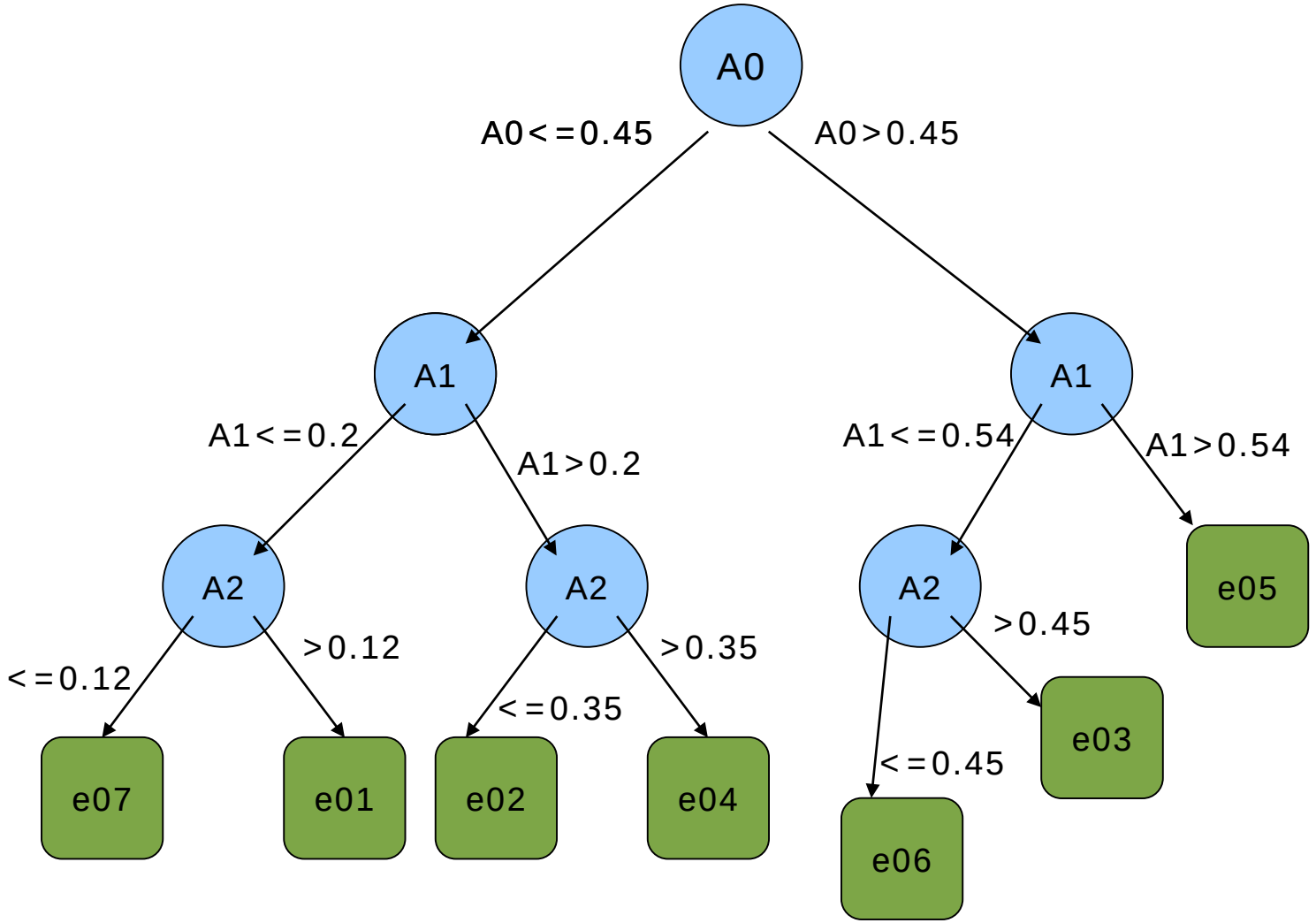
Root node: take attribute A0. Median = 0.45. e01, e02, e04 and e07 go to the left subtree, e03, e05 and e06 to the right one.

Second level: attribute A1. On the left subtree the median is 0.20, e01 and e07 go left and e02, e04 go right. On the right subtree, the median is 0.54. e03 and e06 go left, e05 goes right.

Repeat the process until all examples are on leaves.

One-Class to tackle the Fault Detection

Constructing a k-d tree (II)



One-Class to tackle the Fault Detection

Finding a Nearest Neighbor in the k-d tree

1.- Starting with the root node, the algorithm moves down the tree recursively: it goes left or right depending on whether the point is less than or greater than the current node in the split dimension.

2.- Once the algorithm reaches a leaf node, it saves that node point as the "current best"

3.- Now, it unwinds the recursion of the tree, performing the following steps at each node:

3.1.- If the current node is closer than the current best, then it becomes the current best.

3.2.- The algorithm checks whether there could be any points on the other side of the splitting plane that are closer to the search point than the current best. In concept, this is done by intersecting the splitting hyperplane with a hypersphere around the search point that has a radius equal to the current nearest distance. Since the hyperplanes are all axis-aligned this is implemented as a simple comparison to see whether the difference between the splitting coordinate of the search point and current node is less than the distance (overall coordinates) from the search point to the current best.

3.2.1.- If the hypersphere crosses the plane, there could be nearer points on the other side of the plane, so the algorithm must move down the other branch of the tree from the current node looking for closer points, following the same recursive process as the entire search.

3.2.2.- If the hypersphere doesn't intersect the splitting plane, then the algorithm continues walking up the tree, and the entire branch on the other side of that node is eliminated.

4.- When the algorithm finishes this process for the root node, then the search is complete

Outline

- What are anomalies?
- Anomaly Detection: Taxonomy
- Nearest Neighbor Based Techniques
- One-Class to tackle the Fault Detection
- Concluding Remarks

Conclusions

- Anomaly detection can detect critical information in data
- Highly applicable in various application domains
- Nature of anomaly detection problem is dependent on the application domain
- Need different approaches to solve a particular problem formulation
- The nearest neighbor based techniques are very appropriate for different problems, but they need to be tuned to this problem.

Conclusions

- Related topic: Novelty detection

Contents lists available at ScienceDirect

Signal Processing

journal homepage: www.elsevier.com/locate/sigpro

Review

A review of novelty detection ☆

Marco A.F. Pimentel, David A. Clifton, Lei Clifton, Lionel Tarassenko

Institute of Biomedical Engineering, Department of Engineering Science, University of Oxford, Oxford OX3 7DQ, UK

A B S T R A C T

Novelty detection is the task of classifying test data that differ in some respect from the data that are available during training. This may be seen as "one class classification", in which a model is constructed to describe "normal" training data. The novelty detection approach is typically used when the quantity of available "abnormal" data is insufficient to construct explicit models for non normal classes. Application includes inference in datasets from critical systems, where the quantity of available normal data is very large, such that "normality" may be accurately modelled. In this review we aim to provide an updated and structured investigation of novelty detection research papers that have appeared in the machine learning literature during the last decade.

Inteligencia de Negocio

TEMA 7. Modelos Avanzados de Minería de Datos

1. Clases no balanceadas/equilibradas
2. Características intrínsecas de los datos en clasificación
3. Problemas no estándar de clasificación: MIL, MLL, SSL…
4. Detección de anomalías
5. **Deep Learning**
6. Análisis de Sentimientos

INTELIGENCIA DE NEGOCIO

2018 - 2019

- Tema 1. Introducción a la Inteligencia de Negocio
- Tema 2. Minería de Datos. Ciencia de Datos
- Tema 3. Modelos de Predicción: Clasificación, regresión y series temporales
- Tema 4. Preparación de Datos
- Tema 5. Modelos de Agrupamiento o Segmentación
- Tema 6. Modelos de Asociación
- Tema 7. Modelos Avanzados de Minería de Datos
- Tema 8. Big Data