University of Warsaw
Faculty of Economic Sciences

Unsupervised Learning

Winter Semester, 2025/2026

Handling missing values with unsupervised machine learning

Missing data

Cluster the data

Apply clustering algorithm (e.g., k-means, hierarchical clustering, or DBSCAN) to the dataset with missing values (either after imputation or using distance metrics that handle missingness)

Assign each data point to a cluster.

Impute data based on cluster membership

Within each cluster, calculate representative statistics (mean, median, or mode) for the variables with missing values.

Impute the missing values for each data point using the statistics from its assigned cluster.

Missing data

Gower distance

Gower distance-based clustering is particularly suited for datasets that include both numerical and categorical variables, and it handles missing values effectively. The Gower distance metric calculates the average dissimilarity between two data points, considering each feature individually and accounting for the type of data (numerical, categorical, or binary).

When missing values are present, Gower distance ignores the missing entries for that particular feature in the pairwise comparison. Furthermore, the weights used in calculating the Gower distance are automatically normalized to reflect the number of valid (non-missing) features, so missing values do not bias the overall distance measure by reducing it artificially.

This robustness extends to clustering methods that rely on Gower distance, such as hierarchical clustering or k-medoids (e.g., PAM). These algorithms can work directly with the distance matrix generated using Gower distance.

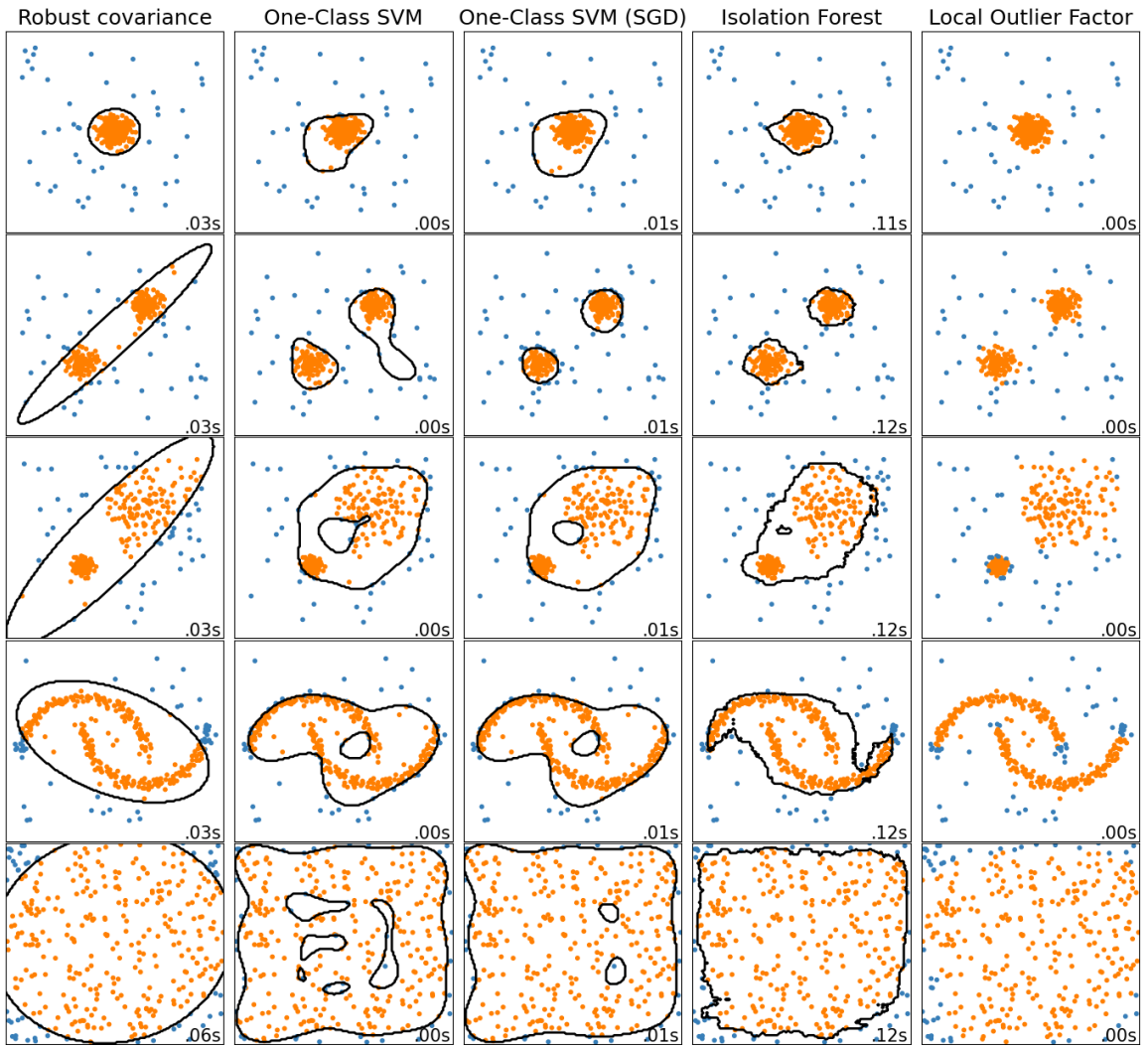
Anomalies

Anomalies

Anomaly detection in the context of unsupervised learning focuses on identifying patterns in data without relying on labeled examples. Since labeled datasets that explicitly distinguish between normal and anomalous data points are often unavailable, unsupervised approaches are commonly employed to tackle the problem. These methods analyze the inherent structure of the data to detect outliers—data points that deviate significantly from the general distribution.

In unsupervised learning for anomaly detection, algorithms typically rely on assumptions about what constitutes "normal" behavior. For instance, they may define normal data as belonging to high-density regions in the feature space, while anomalies are sparse or isolated points. Techniques such as clustering, density estimation, or distance-based methods are frequently used in these scenarios.

One of the key advantages of unsupervised anomaly detection is its flexibility. It can be applied to a wide range of datasets without requiring any prior knowledge or manual labeling. However, this also introduces challenges, as the results are highly dependent on the assumptions made by the algorithm. For example, if the data contains multiple overlapping distributions, the algorithm may struggle to distinguish between legitimate variations and actual anomalies.



Anomalies

SOURCE: SCIKIT LEARN

Anomalies

The Isolation Forest (iForest) algorithm is an unsupervised learning method used for detecting anomalies by isolating data points that significantly differ from the majority of the dataset. It works on the principle that anomalies are rare and distinct, making them easier to isolate compared to normal data points.

The algorithm builds an ensemble of decision trees, known as Isolation Trees, by randomly selecting features and splitting values. These random splits help partition the data, and anomalies, being less frequent and farther from dense regions of the dataset, are typically isolated in fewer splits. This characteristic results in shorter path lengths for anomalies, as they are separated quickly during the tree construction process.

To determine whether a point is an anomaly, the algorithm calculates an anomaly score based on the average path length across all the trees in the forest. Points with shorter average path lengths are assigned higher anomaly scores, indicating they are more likely to be anomalies. Conversely, points that require more splits to be isolated, indicative of being in denser regions, receive lower anomaly scores and are classified as normal.

Isolation Forest is computationally efficient, scalable to large datasets, and does not rely on assumptions about the data's underlying distribution. It performs well in both low-dimensional and high-dimensional spaces.

Anomalies

The Local Outlier Factor (LOF) algorithm is an unsupervised anomaly detection method that identifies data points as anomalies based on their density relative to their local neighborhood. Unlike global approaches, LOF focuses on local density variations, making it particularly effective for datasets where anomalies are context-dependent or where the data has varying densities.

The algorithm works by comparing the density of a data point to the densities of its nearest neighbors. For each data point, LOF calculates its local reachability density (LRD), which measures how densely it is packed relative to its neighbors. The key idea is that anomalies tend to have significantly lower density compared to their neighbors, as they are more isolated and not part of dense clusters.

To quantify how anomalous a point is, LOF computes a score by comparing the LRD of the point to the LRDs of its neighbors. A high LOF score indicates that the point has a substantially lower density than its neighbors, suggesting it is an anomaly. Conversely, points with similar densities to their neighbors have scores close to 1 and are considered normal.
LOF is well-suited for datasets with clusters of varying densities, as it adapts to local data characteristics. However, it is sensitive to parameter tuning, particularly the number of neighbors (k), which determines the size of the local neighborhood. Choosing an appropriate value for k is crucial for accurate anomaly detection.

Anomalies

LOF

LOF measures anomalies based on local density. It calculates how the density of a point compares to the densities of its neighbors. If a point has a much lower density than its neighbors, it is considered an anomaly. This makes LOF sensitive to the local structure of the dataset, making it particularly useful when anomalies are context-dependent or the dataset has varying density regions.

IFOREST

iForest isolates anomalies by randomly splitting features to construct a forest of decision trees. Anomalies are identified as points that require fewer splits (shorter path lengths) to be isolated. This method does not rely on density estimation or distance but rather focuses on the idea that anomalies are rare and distinct.

Unsupervised neural networks

kNN

kNN

simple

effective

fast training

no assumptions upon data distribution

kNN

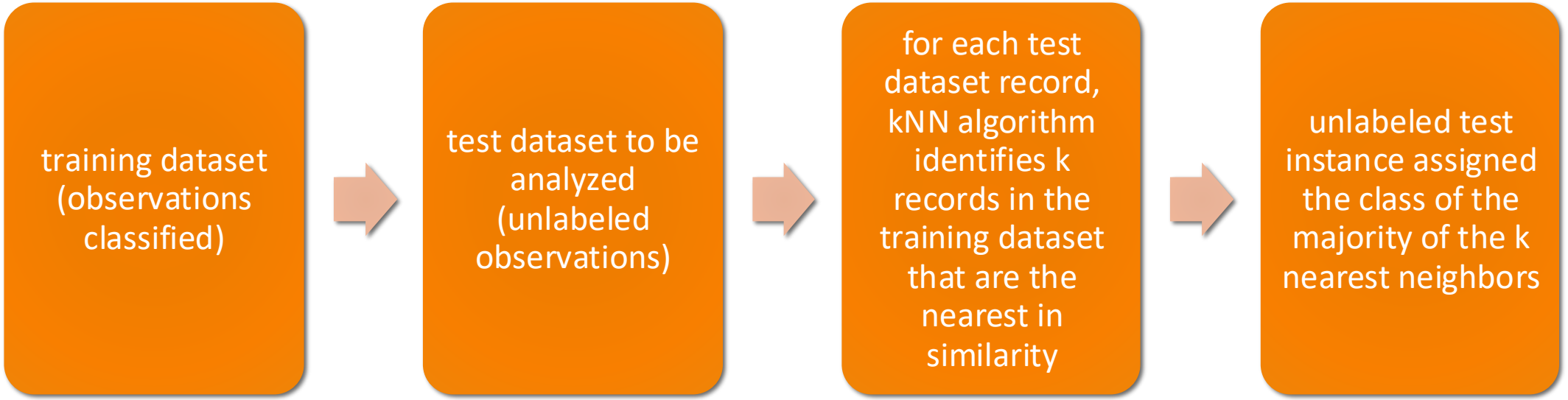
output is not a model

memory consuming

relatively slow classification

missing data or nominal features require additional insight

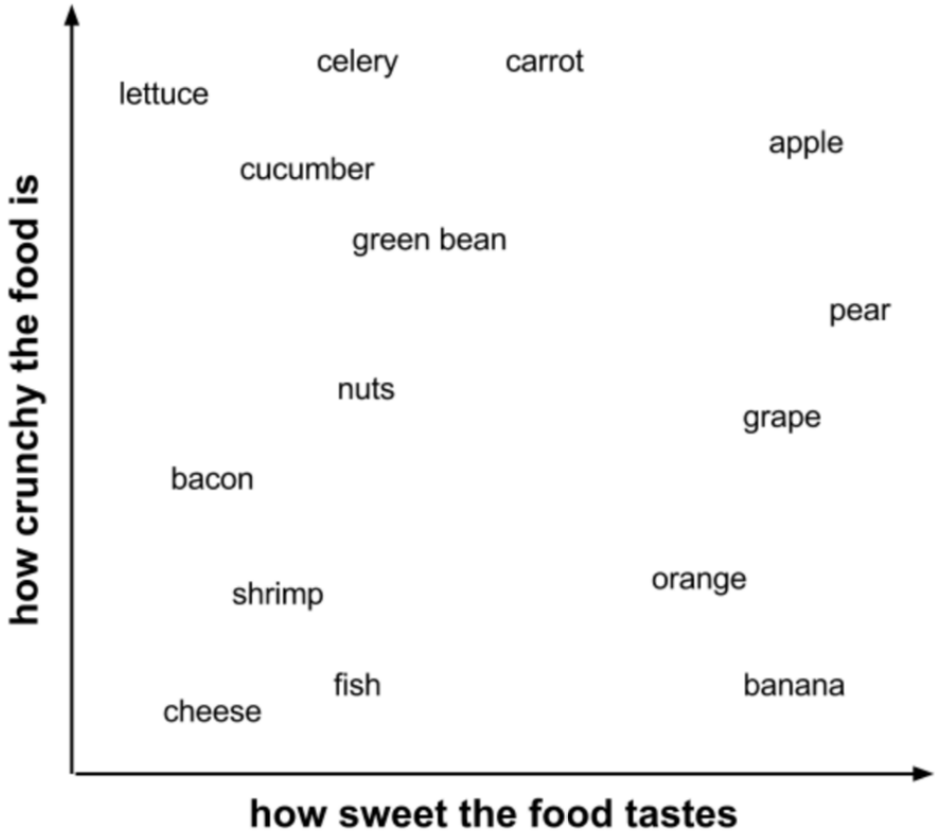
kNN


training dataset (observations classified)
test dataset to be analyzed (unlabeled observations)
for each test dataset record, kNN algorithm identifies k records in the training dataset that are the nearest in similarity
unlabeled test instance assigned the class of the majority of the k nearest neighbors

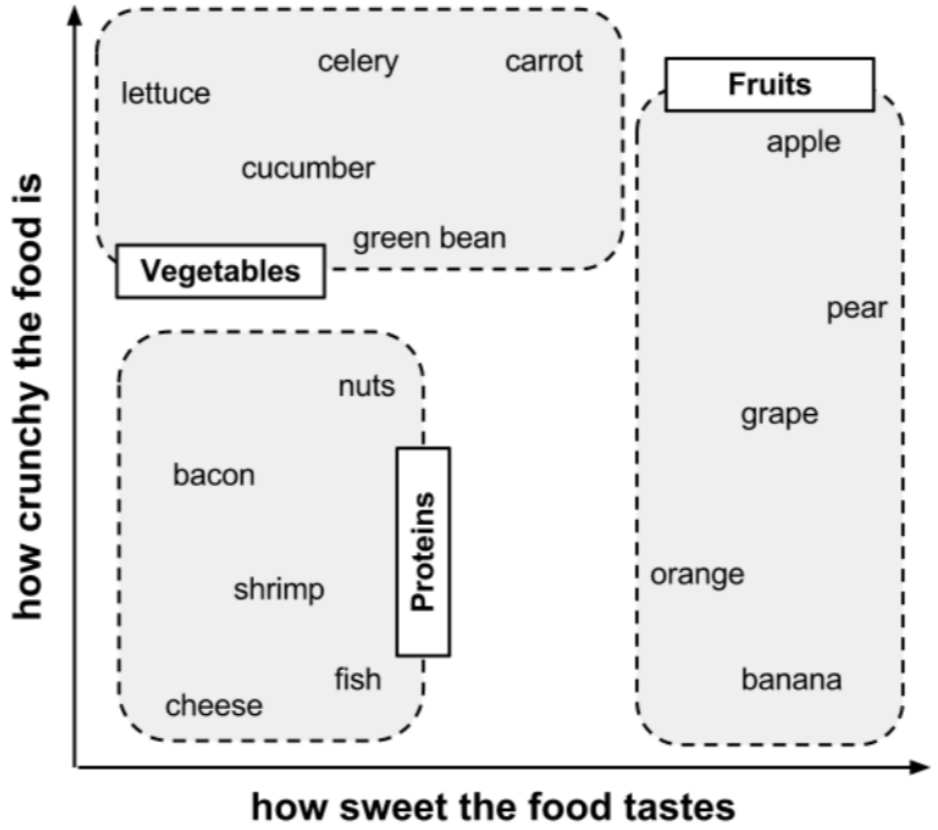
kNN example (Lantz, 2013)

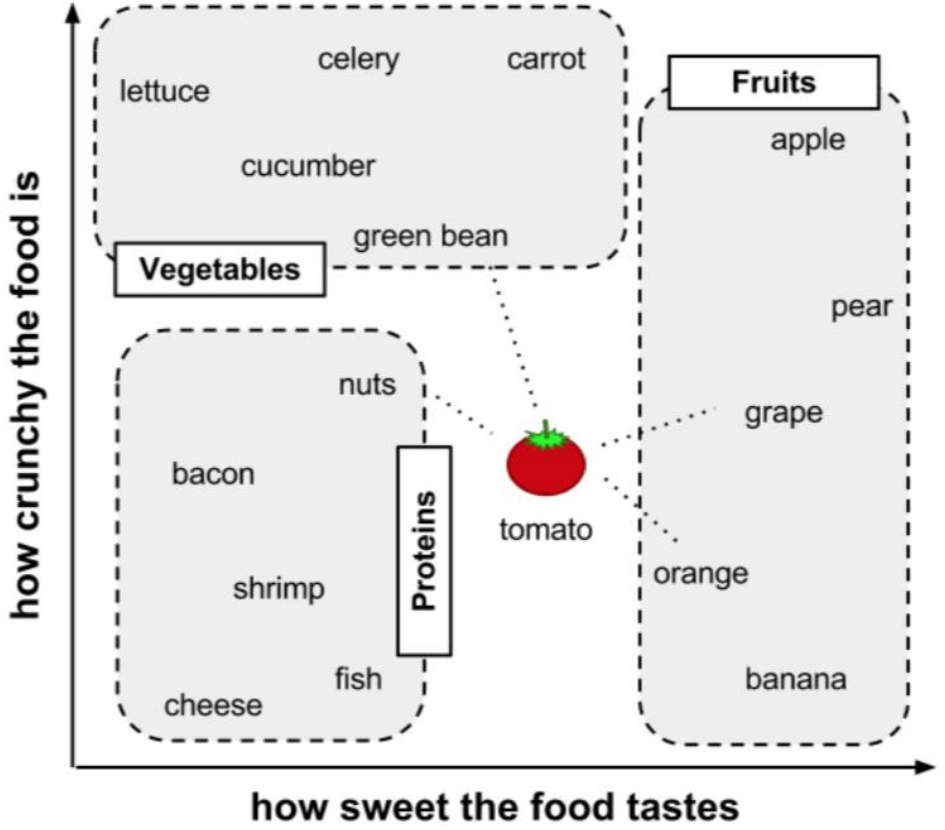
ingredient	sweetness	crunchiness	food type
apple	10	9	fruit
bacon	1	4	protein
banana	10	1	fruit
carrot	7	10	vegetable
celery	3	10	vegetable
cheese	1	1	protein

kNN example (Lantz, 2013)



kNN example (Lantz, 2013)





Distance measures

Distance measures

- in other words: fuction of similarity between objects
- necessary input for kNN and many others
- read more: https://www.sciencedirect.com/topics/computer-science/minkowski-distance
- distance measures (e.g.)
  - Euclidean

$$d(i,j) = \sqrt{\left(x_{i1} - x_{j1}\right)^2 + \left(x_{i2} - x_{j2}\right)^2 + \cdots + \left(x_{ip} - x_{jp}\right)^2}.$$

  - Manhattan

$$d(i,j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \cdots + |x_{ip} - x_{jp}|$$

  - Minkowski

$$d(i,j) = \sqrt[h]{|x_{i1} - x_{j1}|^h + |x_{i2} - x_{j2}|^h + \cdots + |x_{ip} - x_{jp}|^h}.$$

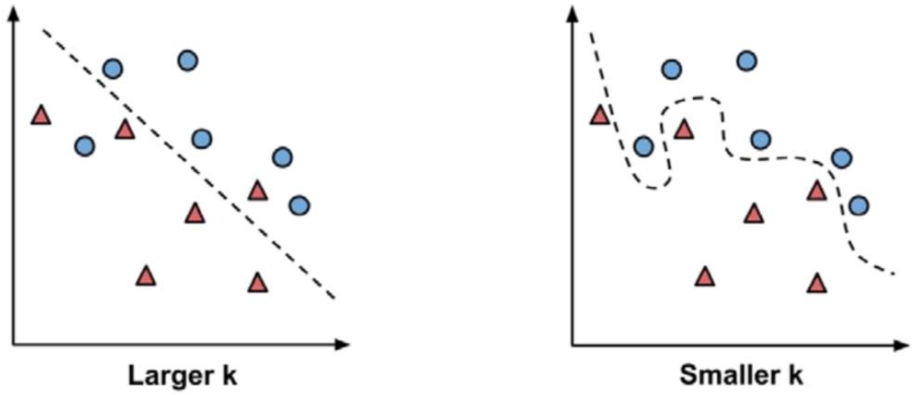
Distance measures example (Lantz, 2013)

ingredient	sweetness	crunchiness	food type	distance to the tomato
grape	8	5	fruit	*sqrt((6 - 8)^2 + (4 - 5)^2) = 2.2*
green bean	3	7	vegetable	*sqrt((6 - 3)^2 + (4 - 7)^2) = 4.2*
nuts	3	6	protein	*sqrt((6 - 3)^2 + (4 - 6)^2) = 3.6*
orange	7	3	fruit	*sqrt((6 - 7)^2 + (4 - 3)^2) = 1.4*

Appropriate k

Appropriate k

- under/overfitting; bias-variance tradeoff
- large k
  - less impact due to noisy data; risk of ignoring small patterns
- depends strongly on the data
- square root of the number of training examples
- various options tested to pick up the best classification performance
- Lantz, 2013:



Data preparation

Preparing data for kNN

- what would happen if some features have much larger values than others?
- we have to adjust scaling in our data
  - min-max normalization (0-1) [how far the original value is along the range between the original minimum and maximum]

$$X_{new} = \frac{X - \min(X)}{\max(X) - \min(X)}$$

  - z-score standardization [how many standard deviations a feature's values fall above or below the mean value]

$$X_{new} = \frac{X - \mu}{\sigma} = \frac{X - \text{Mean}(X)}{\text{StdDev}(X)}$$

  - dummy coding
  - dividing each value by the standard deviation [set of transformed variables with variances of 1, but different means and ranges]

Confusion matrix

	Actual Positives	Actual Negatives
Positive Predictions	True Positives (TP)	False Positives (FP)
Negative Predictions	False Negatives (FN)	True Negatives (TN)

Confusion matrix

- Accuracy=(TP+TN)/N
- Precision=TP/(TP+FP)
- Recall=TP/(TP+FN)
- Error=(FP+FN)/N
- F1=2*Recall*Precision/(Recall+Precision)
- TP+FP=|{retrieved elements}|
- TP=|{relevant elements}∩{retrieved elements}|

Classification problems

- Assume the number of classes
- Binary classification problem
  - Exactly two classes
- Multiclass problem
  - More than two classes and each object falls into exactly one class
- Multi-label problem
  - More than two classses and each object falls into more than one class

Evaluation metrics

- The results may be mis-interpreted
- Reason? Unbalanced number of positive and negative observations, what results in accuracy and error measures dominated by TN
- What to propose then? Probably recall, precision or F1 measures

Evaluation metrics

- How to compute scores in multi-label classification?
  - Macro-averaged measures (average the scores of binary tasks) – dominated by the system's performance on rare categories
  - Micro-averaged measures (sum the FP, TP, TN, FN, N over all the categories and compute each of the metric) – dominated by the system's performance on most common categories

Evaluation metrics

- More popular metrics
  - Mean average precision
  - Mean reciprocal rank
  - Spearman's rank correlation coefficient
  - Geometric mean of average precisiion
  - Discounted cumulative gain
  - R-precision
  - K-precision (precision at K)

Thank you!