# INTRODUCTION TO MACHINE LEARNING

# Data Mining vs Machine Learning

- Data Mining is a cross-disciplinary field that focuses on discovering properties of data sets.
- Different approaches to discovering properties of datasets.
  - Correlation analysis
  - Visualization techniques
  - Machine learning

**Machine Learning is one of the approaches for Data mining**

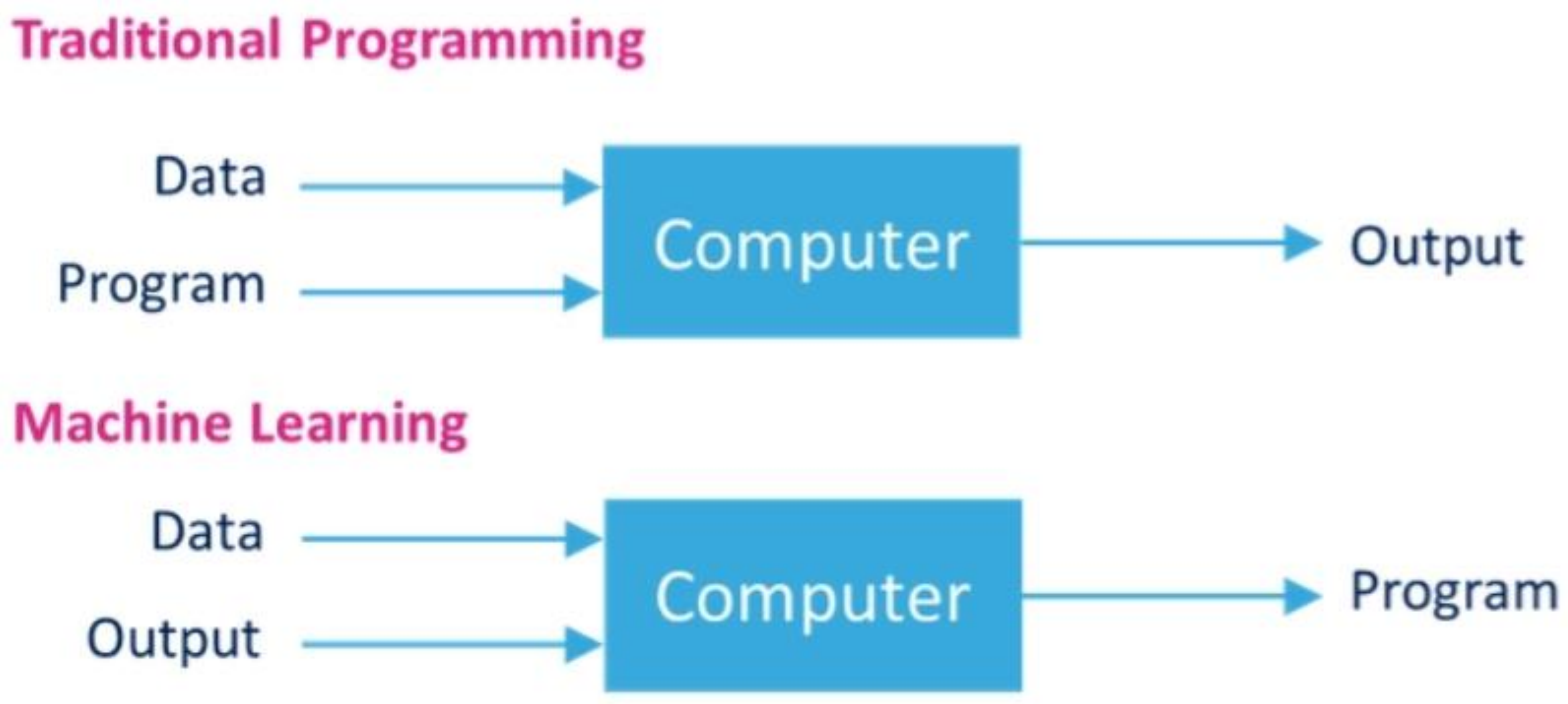
# What is Machine Learning?

- "Learning is any process by which a system improves performance from experience." – Herbert Simon
- "Machine learning is programming computers to optimize a performance criterion using example data or past experience.' Alpaydin, 2004
- Machine learning is about:
  - Learning general models from a data of particular examples.
  - Build a model that is a good and useful approximation to the data.

# What is Machine Learning?

- The essence of machine learning can be pinned down to three main parts:
  - We have the **dataset**.
  - A **pattern** must **exist** in the dataset.
  - We **cannot pin** down the pattern existing in the dataset **mathematically**.
- Learning isn’t always useful:
  - There is no need to “learn” to calculate payroll
- Learning is used when:
  - Human expertise does not exist (navigating on Mars)
  - Humans are unable to explain their expertise (speech recognition)
  - Solution changes in time (routing on a computer network)
  - There is Huge amounts of data

# Learning vs Programming



# The learning problem

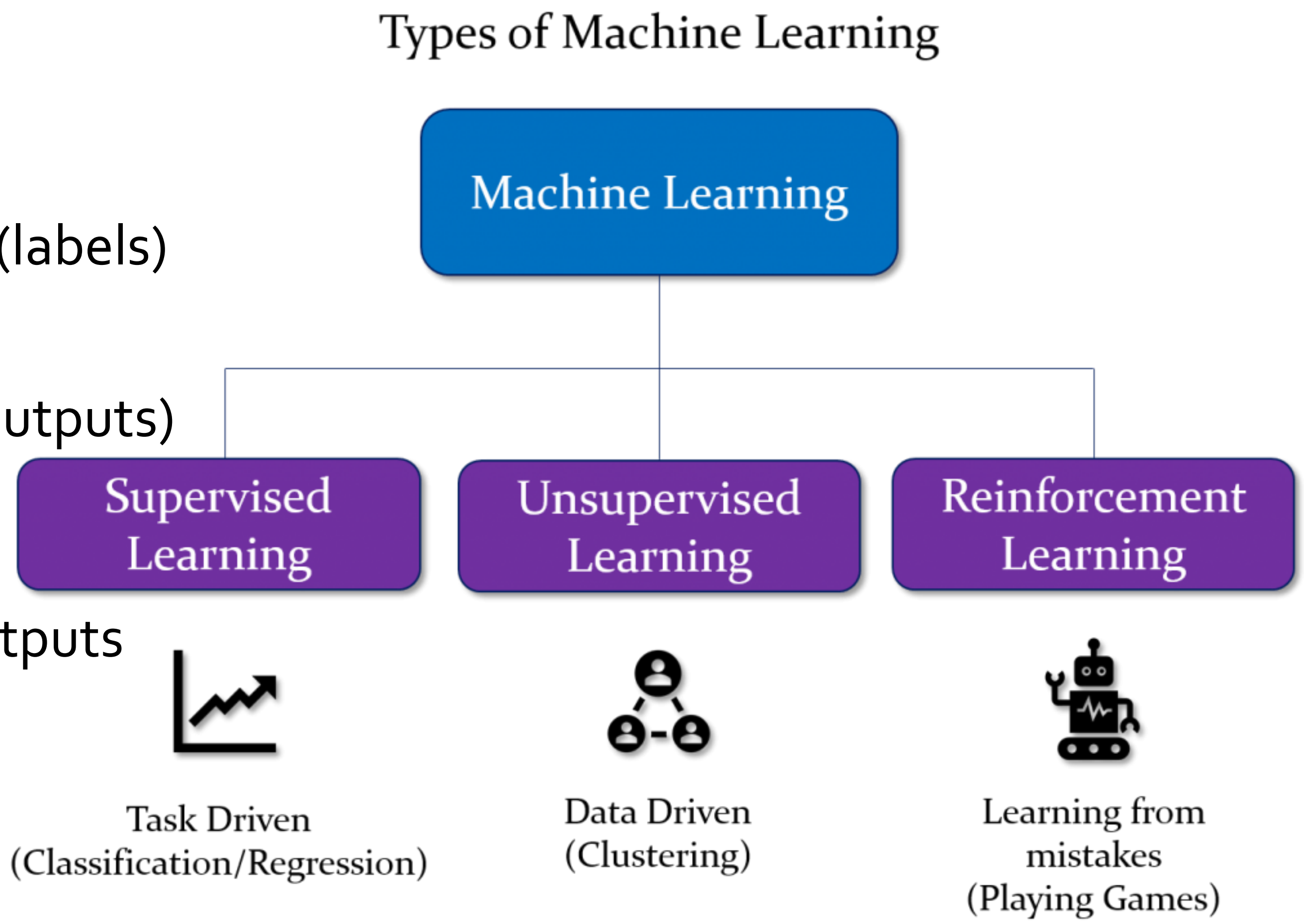
- Given a **set of examples** (the training set)
- Find a **function** that performs a given task
- with respect to a **performance metric**
- Example :
  - Given a Dataset of emails, some with human-given labels (spam or legitimate)
  - Find a function that categorize email messages as spam or legitimate.
  - Using the Percentage of email messages correctly classified as a metric.

# ML main components

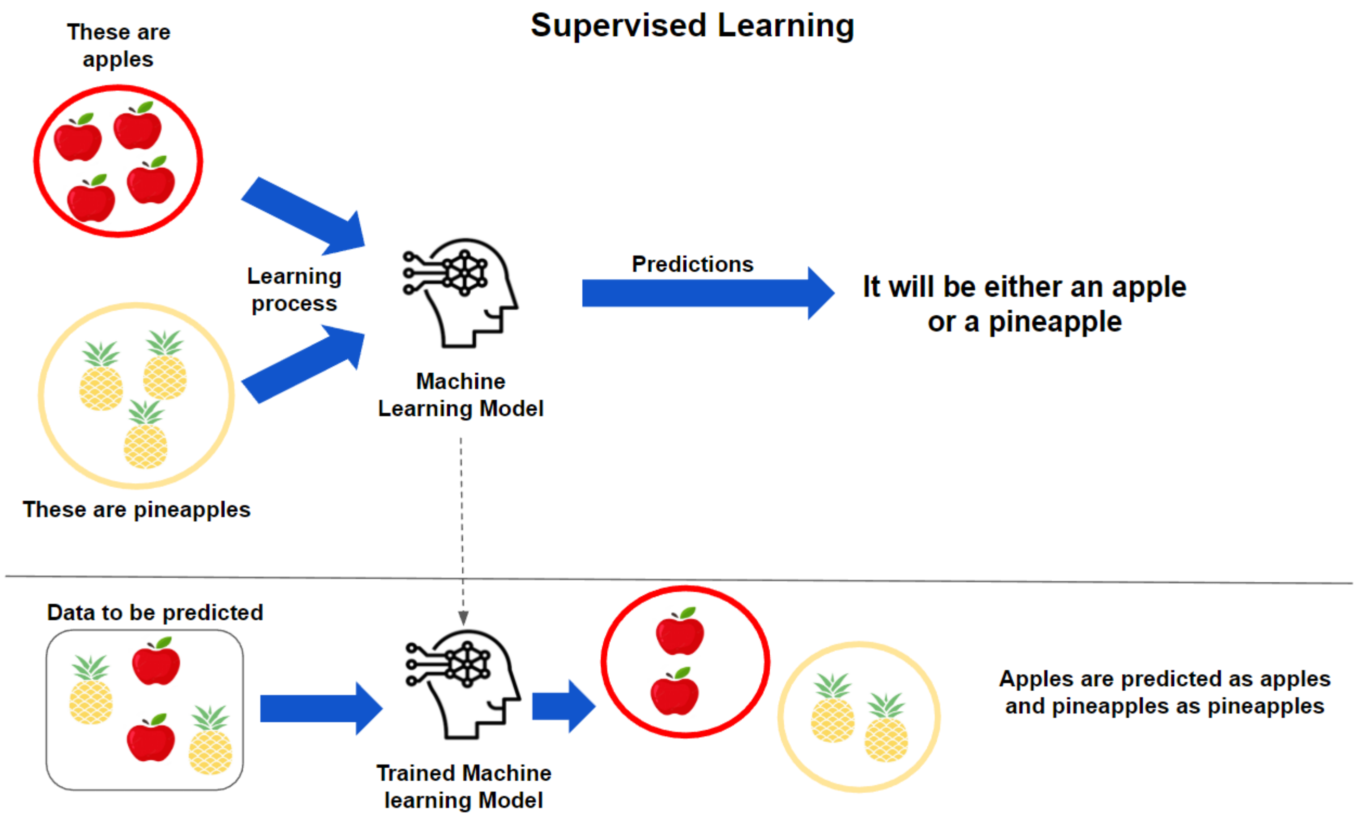
- Model representation (hypothesis space):
  - Structure of the functional form of the knowledge to be extracted (Trees, partition, graph,…)
- Search method (learning algorithm):
  - Strategy used to explore the search space and find the optimal or "good" model (backpropagation, local search, divide-and-conquer, greedy search, …)
- Objectif function (cost function):
  - Measure the quality of the model (Gini, Entropy, RMSE, logloss, …)

# Types of learning

- **Supervised learning**
  - Given: training data + desired outputs (labels)
- **Unsupervised learning**
  - Given: training data (without desired outputs)
- **Semi-supervised learning**
  - Given: training data + a few desired outputs
- **Reinforcement learning**
  - Rewards from sequence of actions

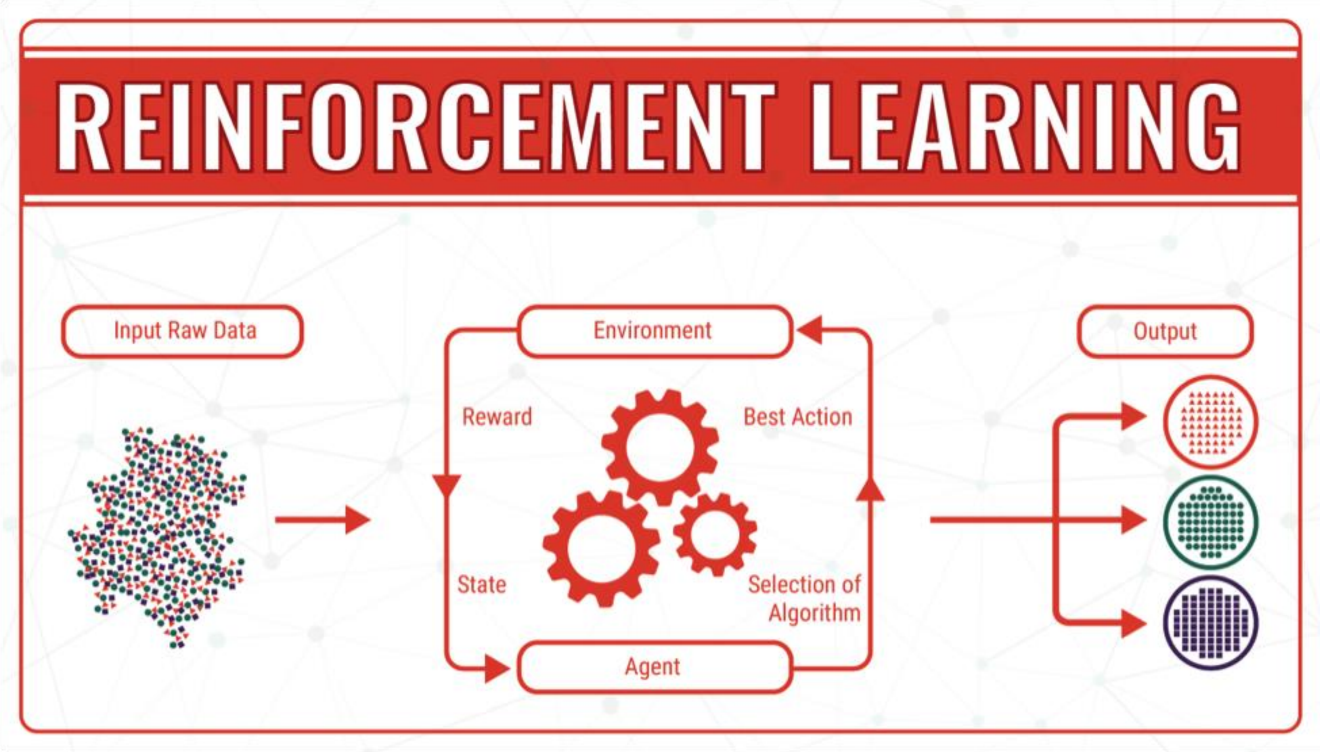


# Supervised learning



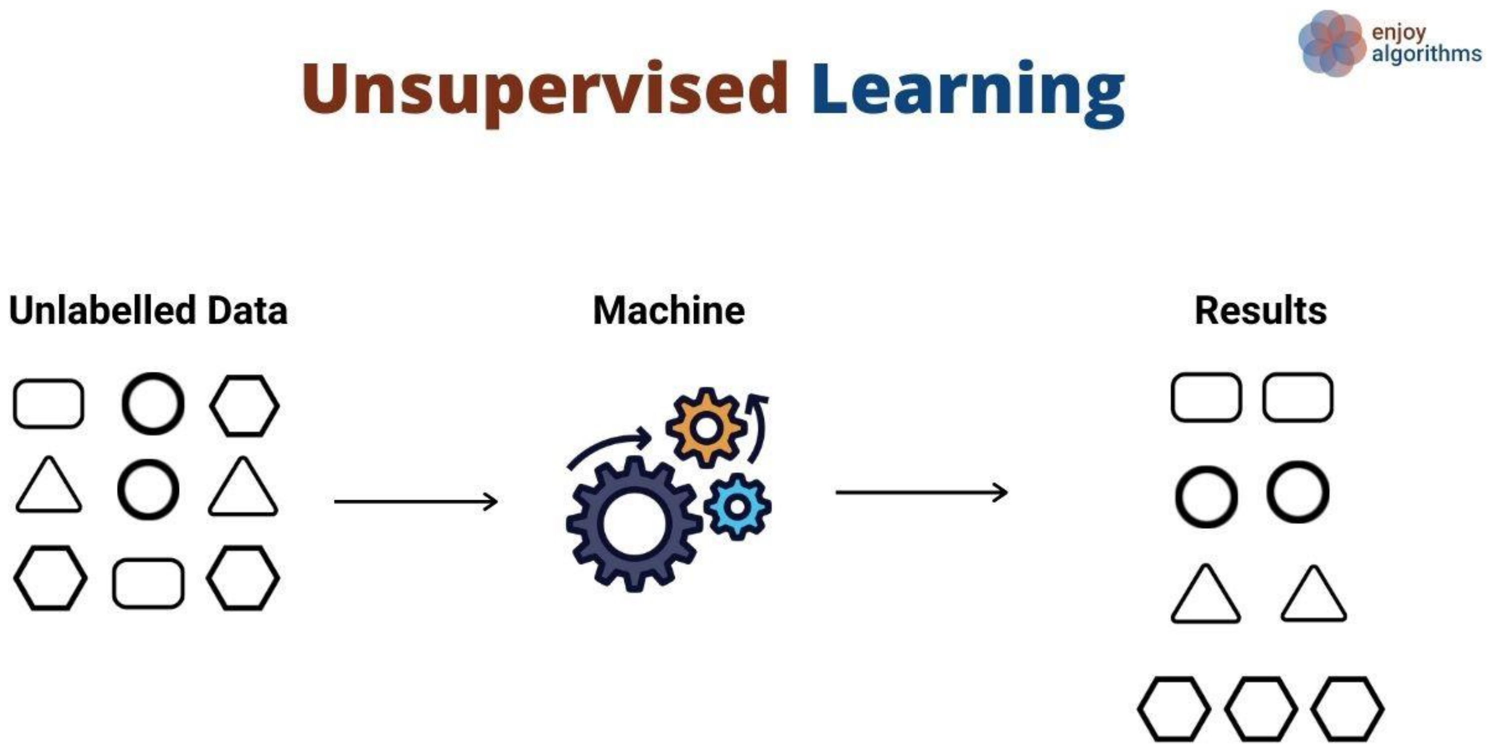

- Learn a function from **labeled data**

# Reinforcement Learning



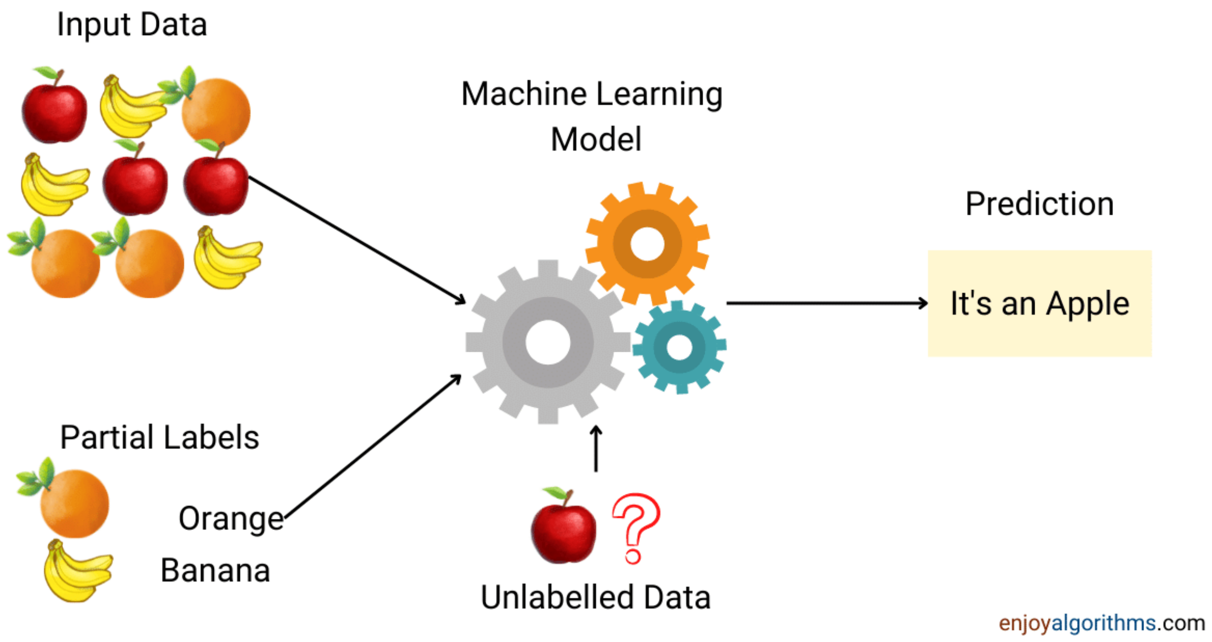

- Given a sequence of states and actions with (delayed) rewards, output a policy
  - Policy is a mapping from states to actions that tells you what to do in a given state

# Unsupervised Learning



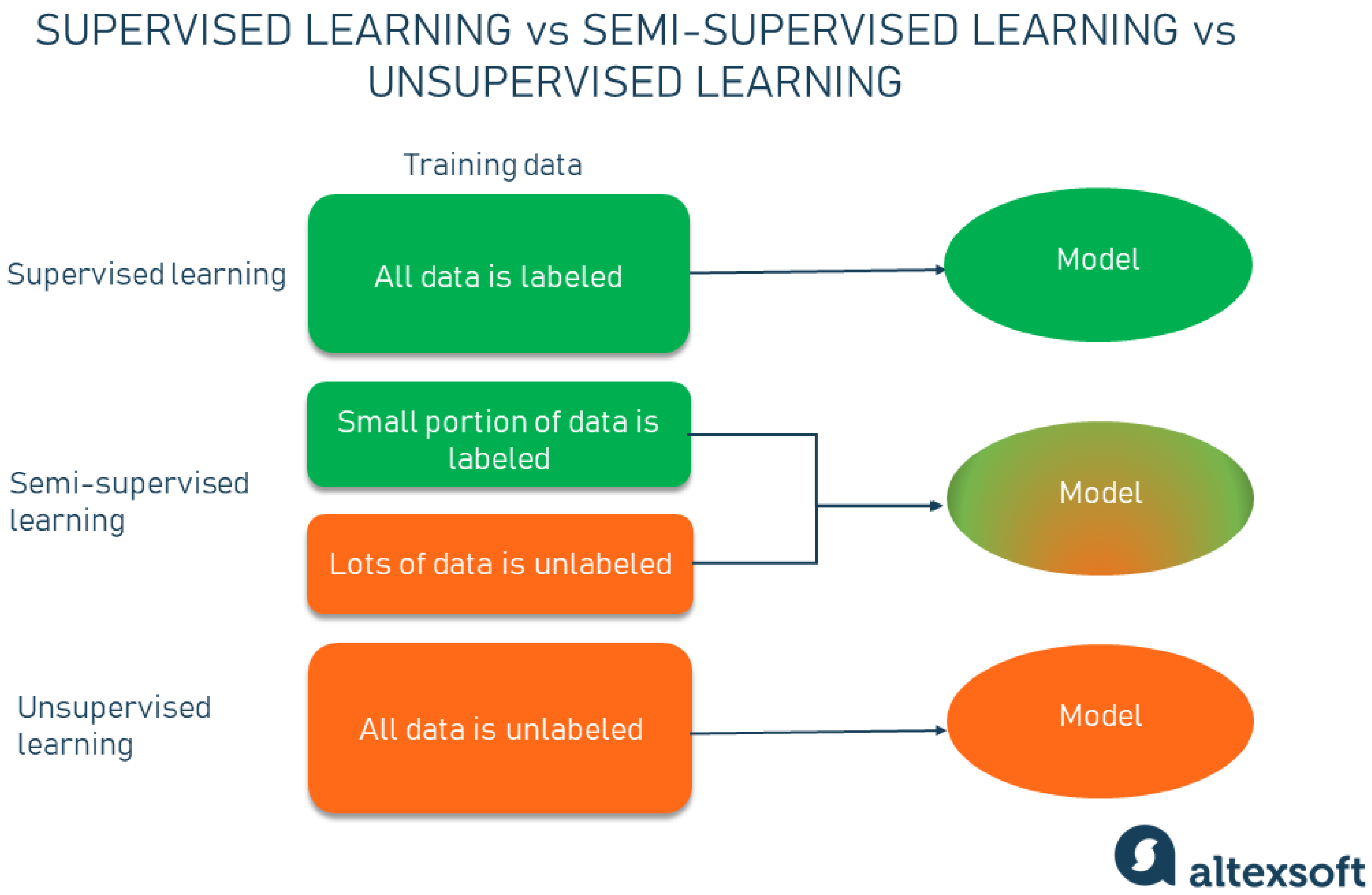

- Learn from unlabeled data.
- A model is fit to observations

# Semi-supervised Learning




- Semi-supervised learning is partially supervised and partially unsupervised
- Only a small portion of data is labeled

# Types of learning

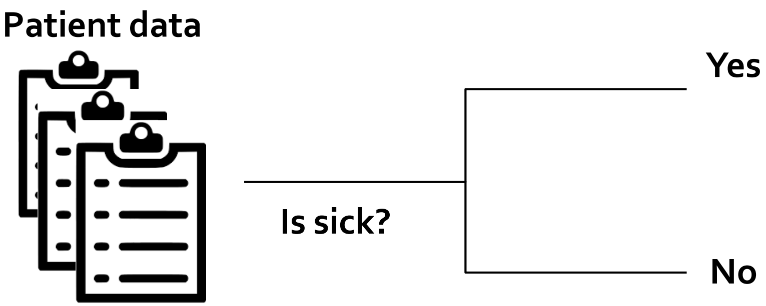


# Designing a learning problem

1. Choose the training experience
2. Choose the target function (what to learn) and how to represent it (e.g. Linear, Tree,…)
3. Choose a learning algorithm
4. Evaluate the entire system

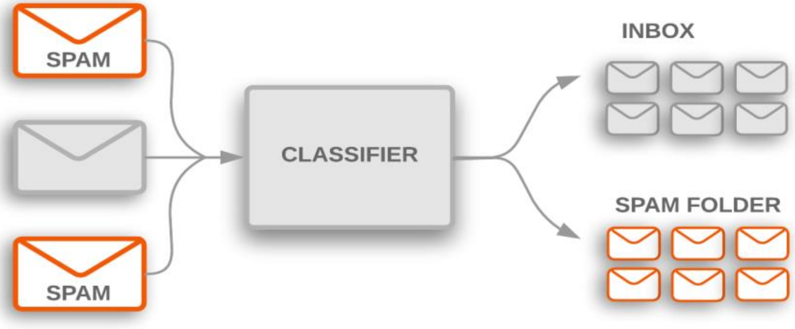
For rest of this course, we will focus on supervised learning

# Supervised Learning : Classification




**Medical Diagnosis**

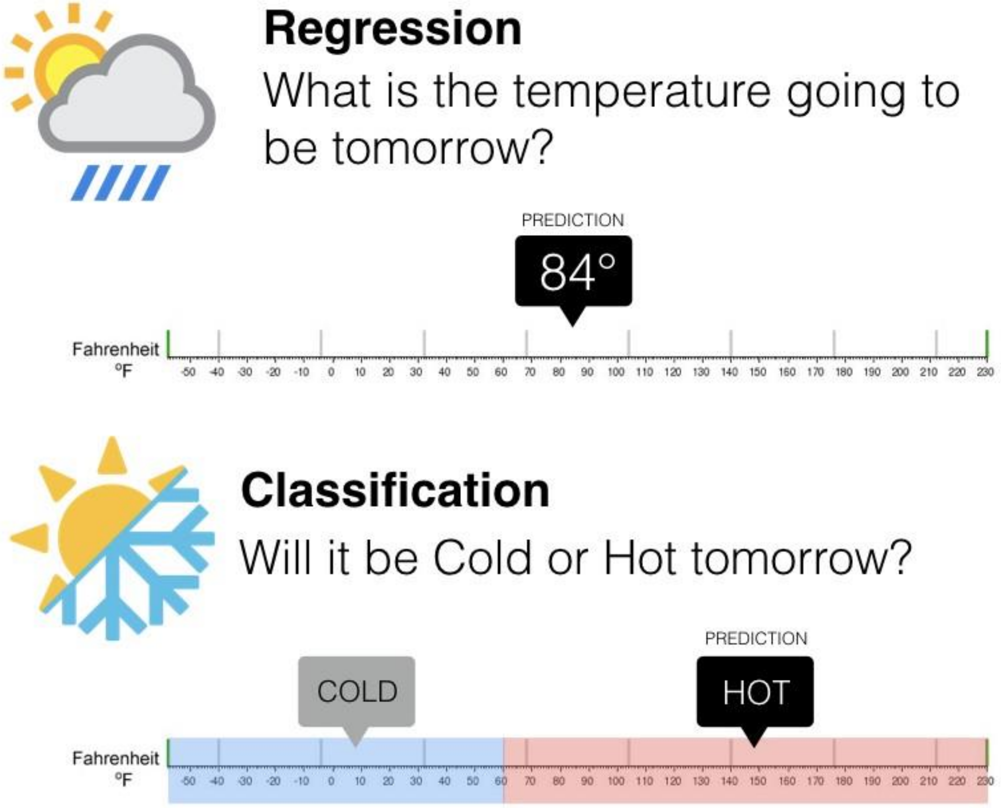
**Text Categorization**


**Spam detection**

Classification assigns data items to target categories or classes

# Classification vs regression




Regression is the task of predicting a continuous quantity

Classification is the task of predicting a discrete class label

# Classification

## Example of application

- Fraudulent credit card transactions detection
- Approach:
  - **Data used**: credit card transactions and information of account holders
    - When does a customer buy, the products he buy and how often he pays on time…
  - **Data labeling**: label past transactions as “fraud” or “fair” (the class attribute)
  - **Modeling**: analyze the data to learn a model for the class of the transactions
  - **Usage**: use the learned model to detect fraud on credit card transactions

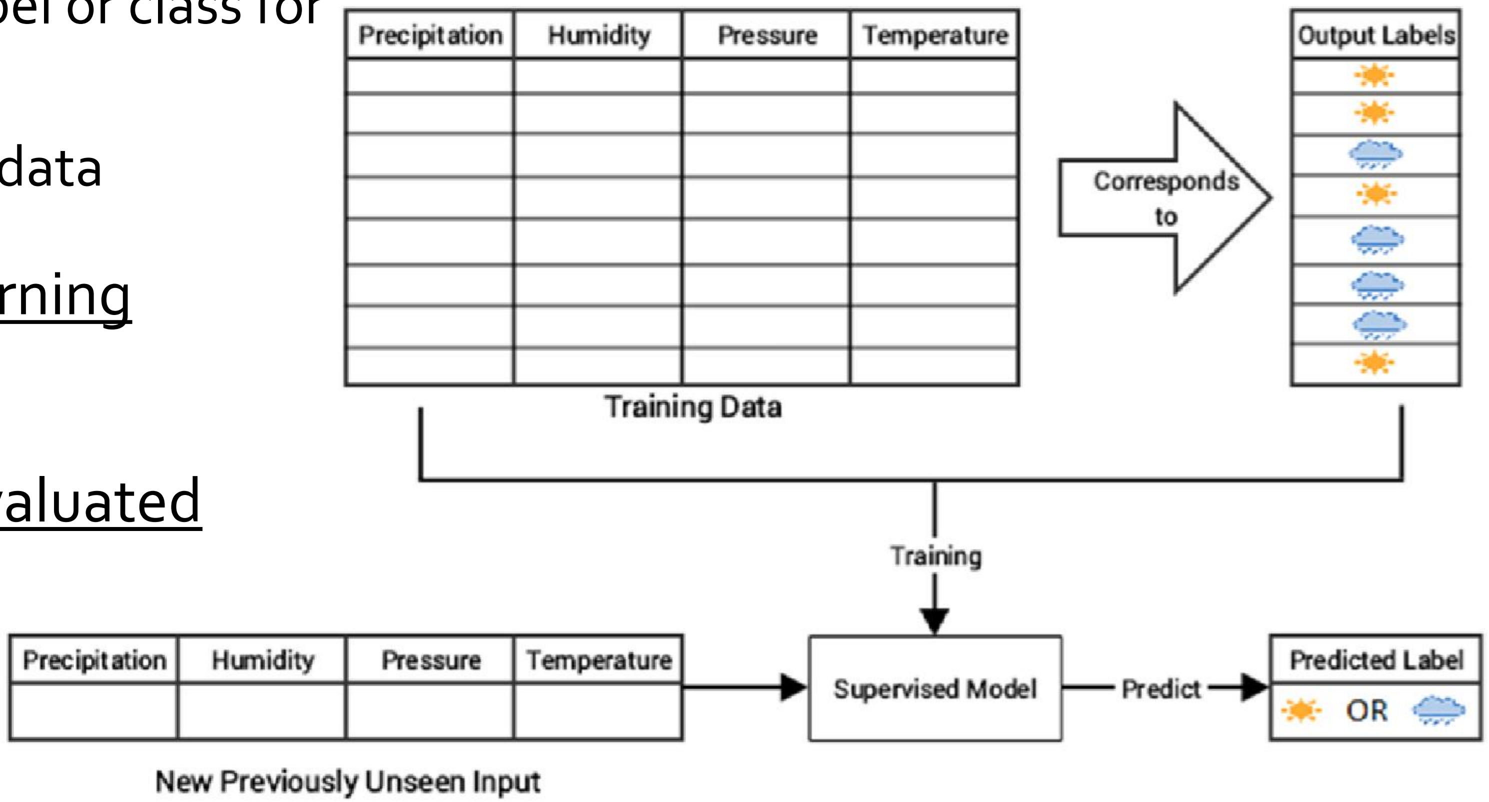
# Classification

Classification uses an algorithm (classifier):

- To automatically predict the label or class for any new data point
- On the basis of a set of labeled data

Classification is a supervised learning method

The quality of classification is evaluated through common metrics.



# Classification methods

Approaches to learn classifiers:

- Linear classifiers: Logistic Regression, Bayesian classifier
- Support Vector Machines (SVM)
- Decision trees
- Random Forest
- K-Nearest Neighbor
- Neural networks
- …

# The Classification problem

The problem of classification is defined as:

- **Given**: A set of training data

  $(x_1, y_1), \dots (x_n, y_n)$ where $x_i \; in \; \mathbb{R}^n$ and $y_i$ is the class label

- **Find :** A classification function

  $f\colon \; \mathbb{R}^n \; \rightarrow \{c_1, \dots, c_k\}$ which classifies well additional samples $\{x_k\}$

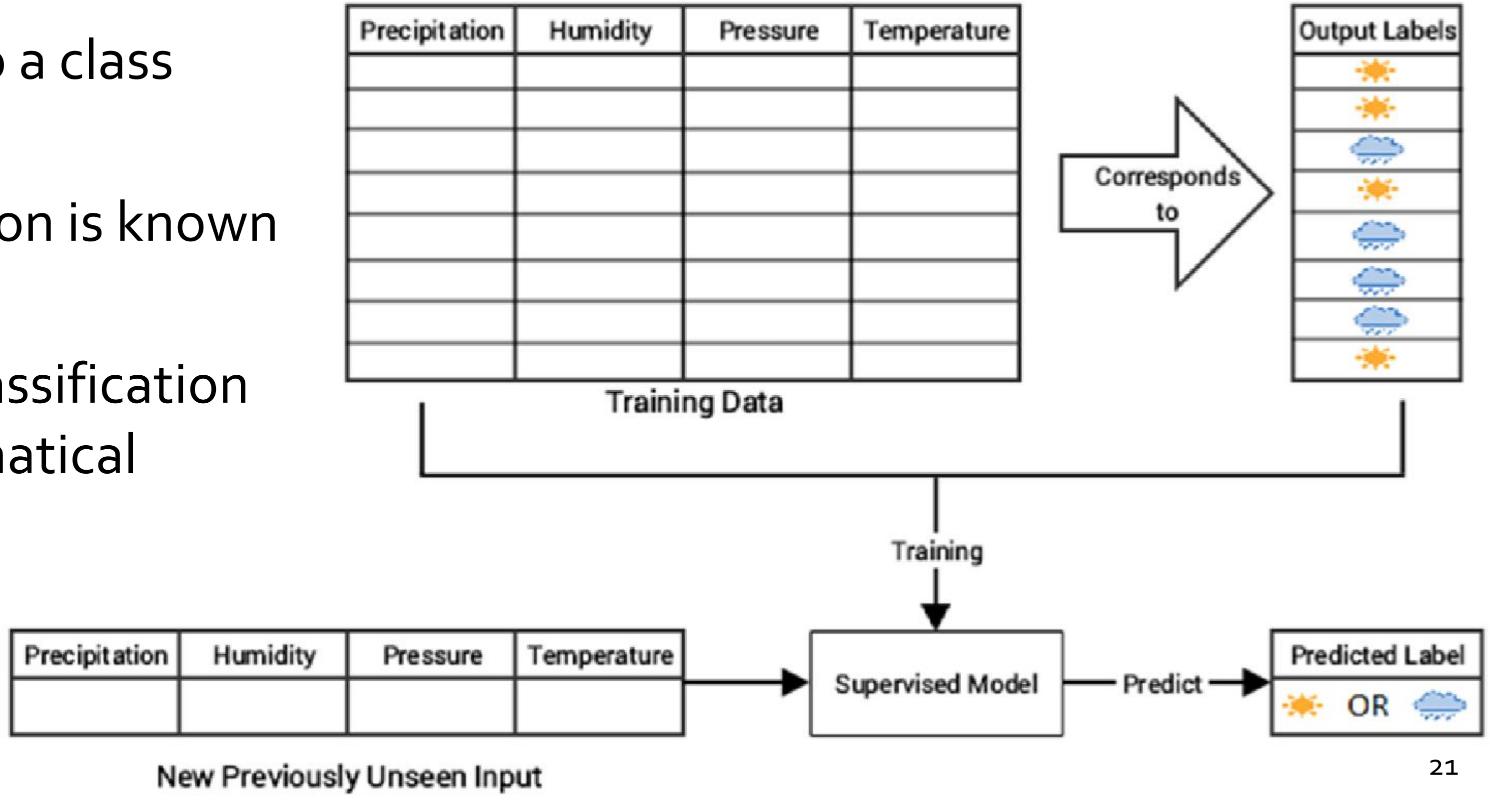
  k is the number of classes

# Classification- A two-step process

Train and test paradigm!

**Step 1: Building the classification model**

- Each data point is associated to a class label
- Data used for model construction is known as the training set
- The model is represented as classification rules, decision trees or mathematical formula

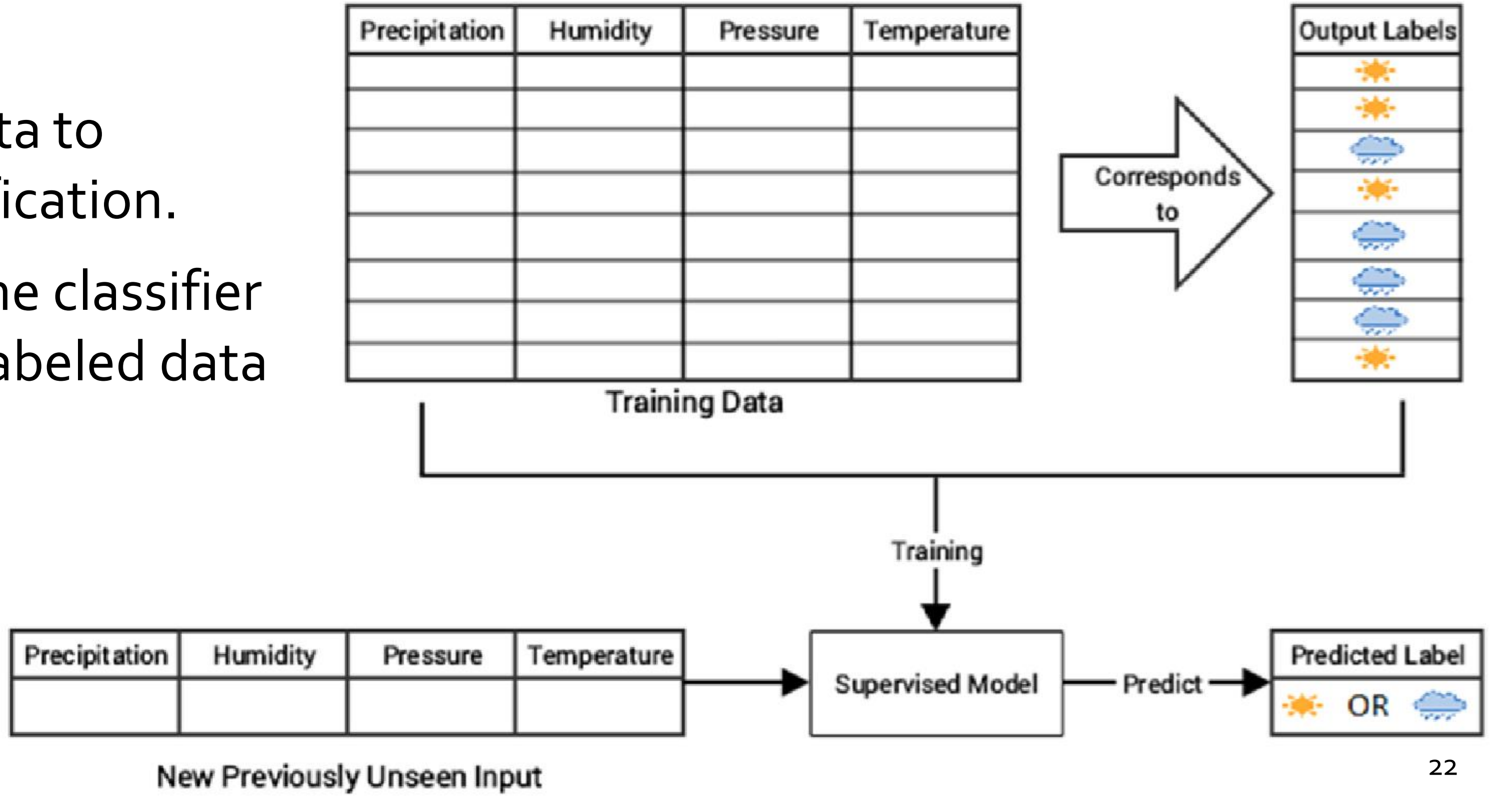


# Classification- A two-step process

**Step 2: Using the classifier for classification**

- Testing the classifier on test data to estimate the accuracy of classification.
- If  the accuracy is acceptable, the classifier can be used to classify new unlabeled data

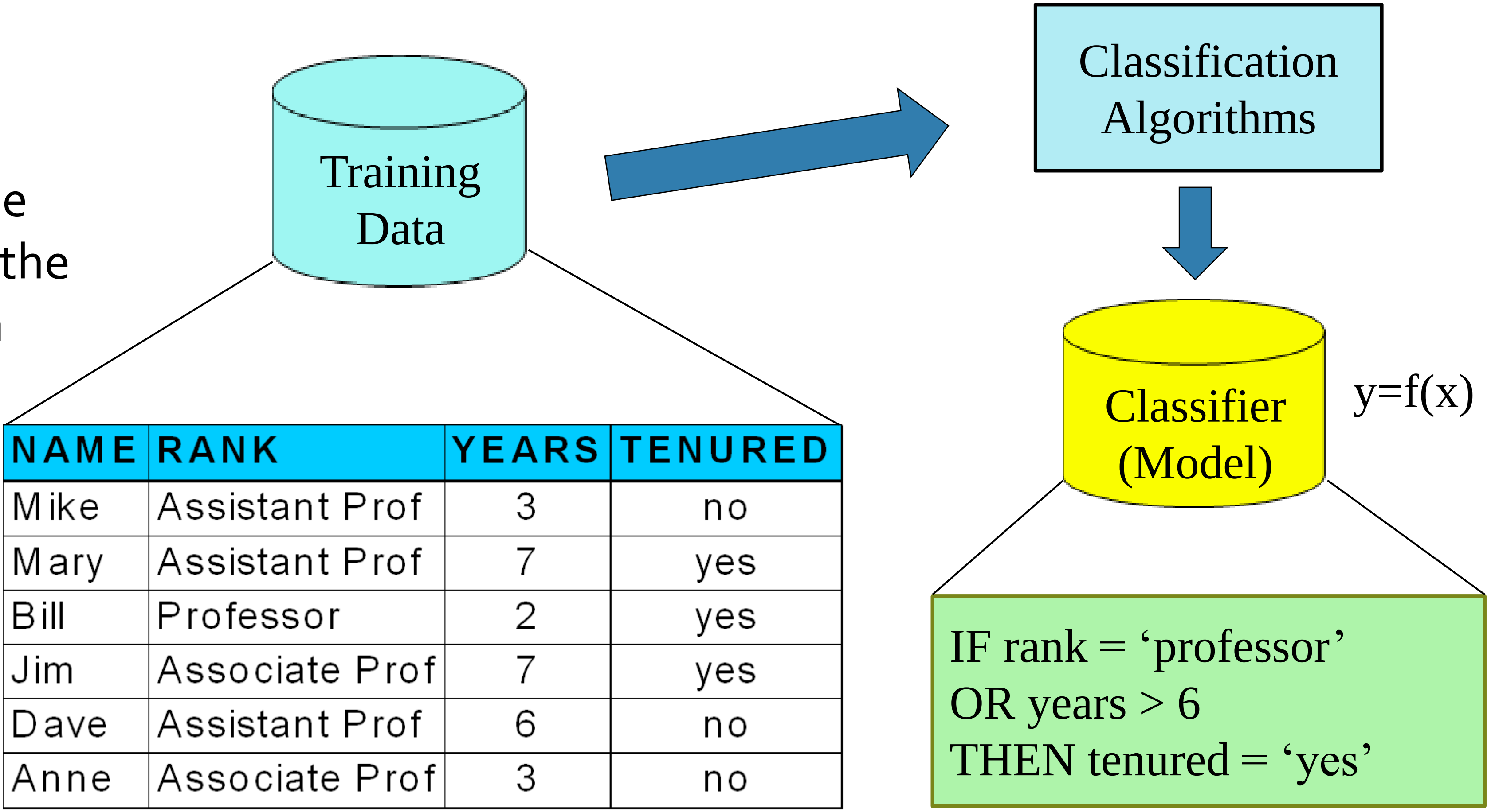
Train and test paradigm!



# Classifier building

**Learning phase**

The classification algorithm analyzes the training data to learn the classification function

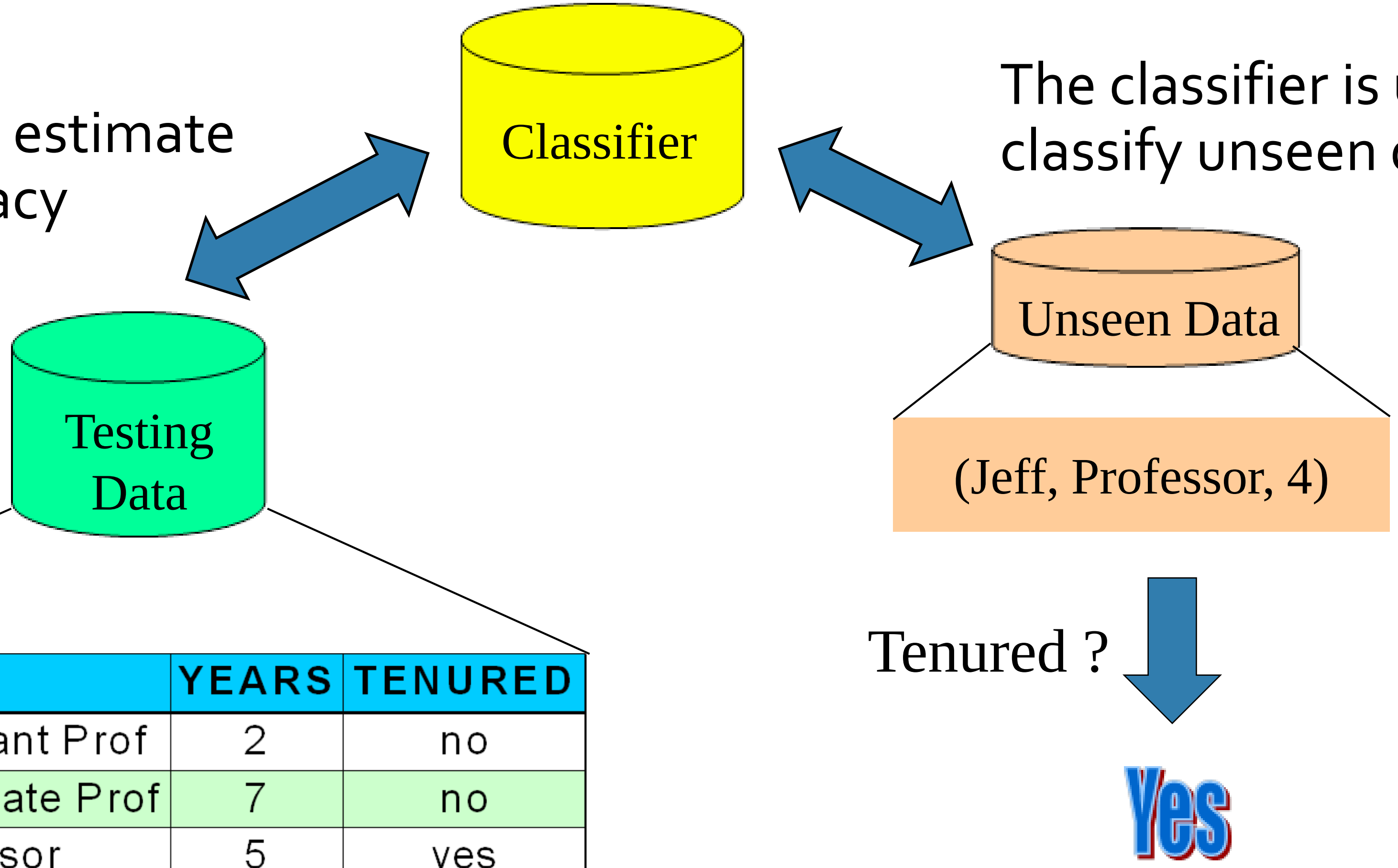



| NAME | RANK | YEARS | TENURED |
|---|---|---|---|
| Mike | Assistant Prof | 3 | no |
| Mary | Assistant Prof | 7 | yes |
| Bill | Professor | 2 | yes |
| Jim | Associate Prof | 7 | yes |
| Dave | Assistant Prof | 6 | no |
| Anne | Associate Prof | 3 | no |

# Classifier usage

Test data is used to estimate the classifier accuracy

The classifier is used to classify unseen data

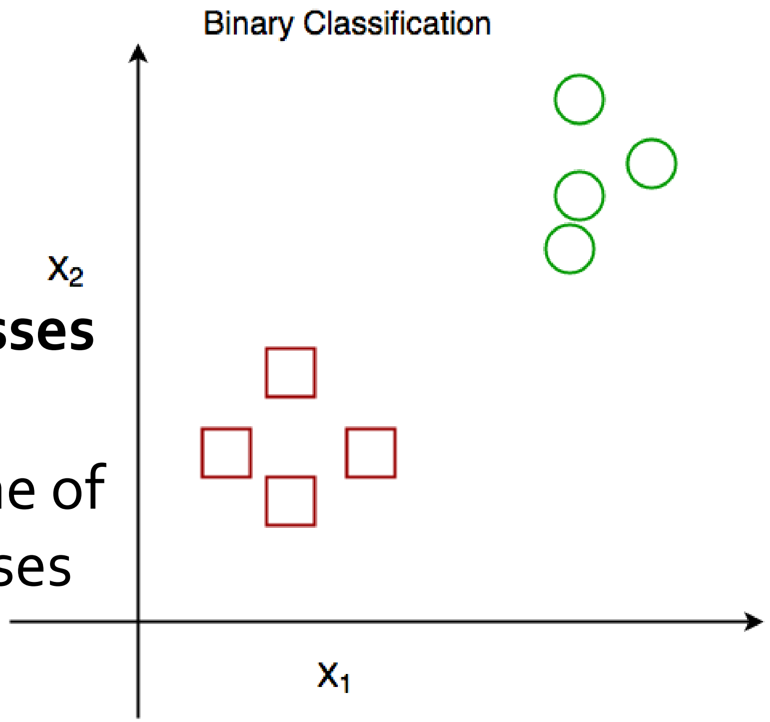


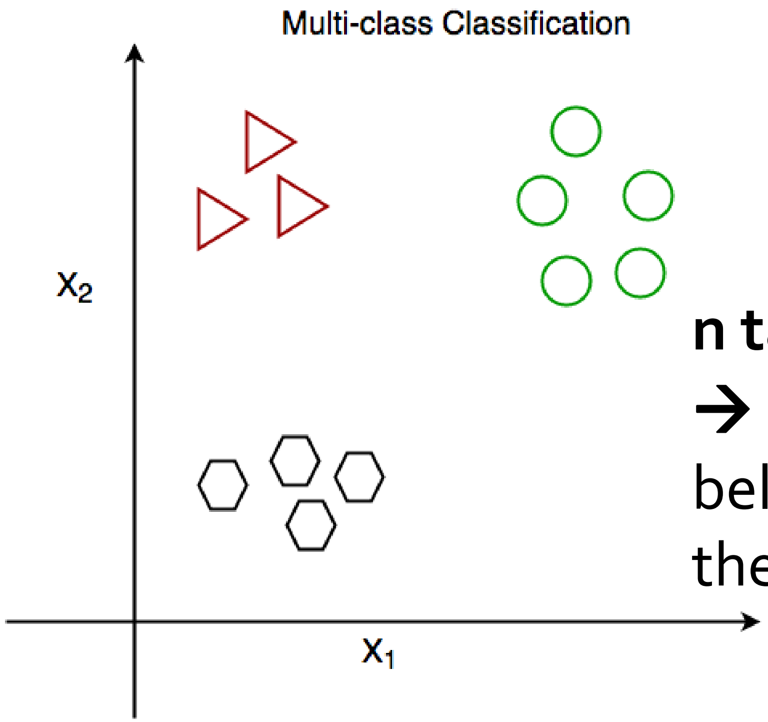

| NAME | RANK | YEARS | TENURED |
|---|---|---|---|
| Tom | Assistant Prof | 2 | no |
| Merlisa | Associate Prof | 7 | no |
| George | Professor | 5 | yes |
| Joseph | Assistant Prof | 7 | yes |

# Types of classification

Binary vs multiclass classification

**2 target classes**
→ Samples belong to one of the two classes



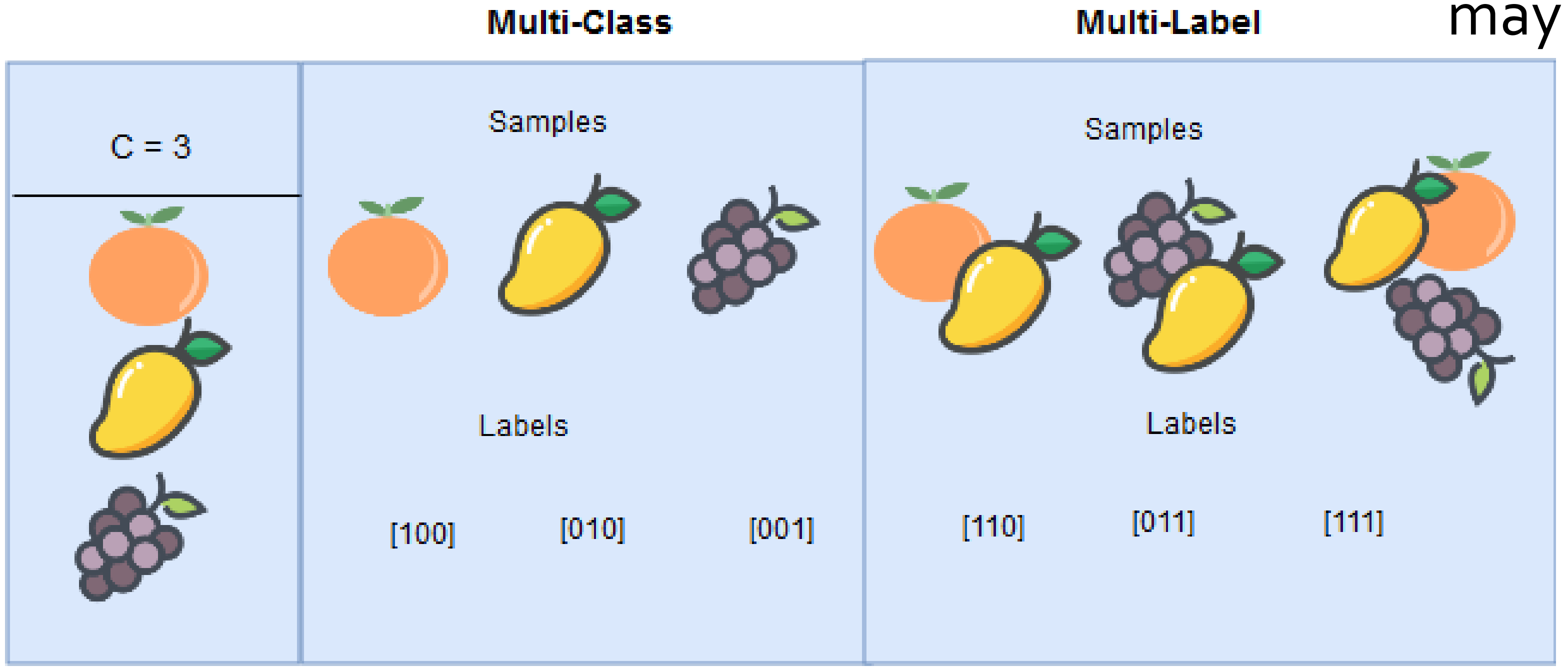

**n target classes**
→ Samples belong to one of the n classes

# Types of classification

## Multiclass vs multilabel classification

**Classes are mutually exclusive**
→ A sample can be assigned to <u>one label only</u>

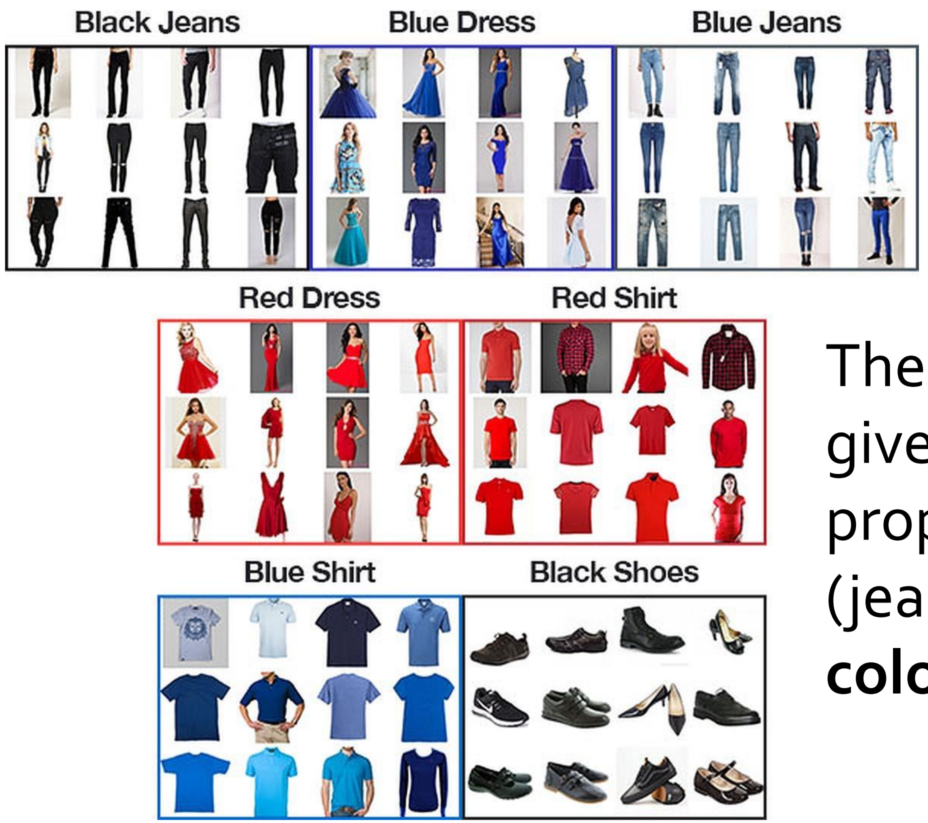
To one sample, <u>multiple labels</u> may be assigned



# Types of classification

## Multioutput-multiclass classification

- Simultaneously outputs a set of labels
- A label is output for each property and each label is one of the possible classes of the corresponding property.




The goal is to predict for a given image two properties: the **category** (jeans, dress, shirt, …) + **color** (Black, blue, … )

# Classifier evaluation

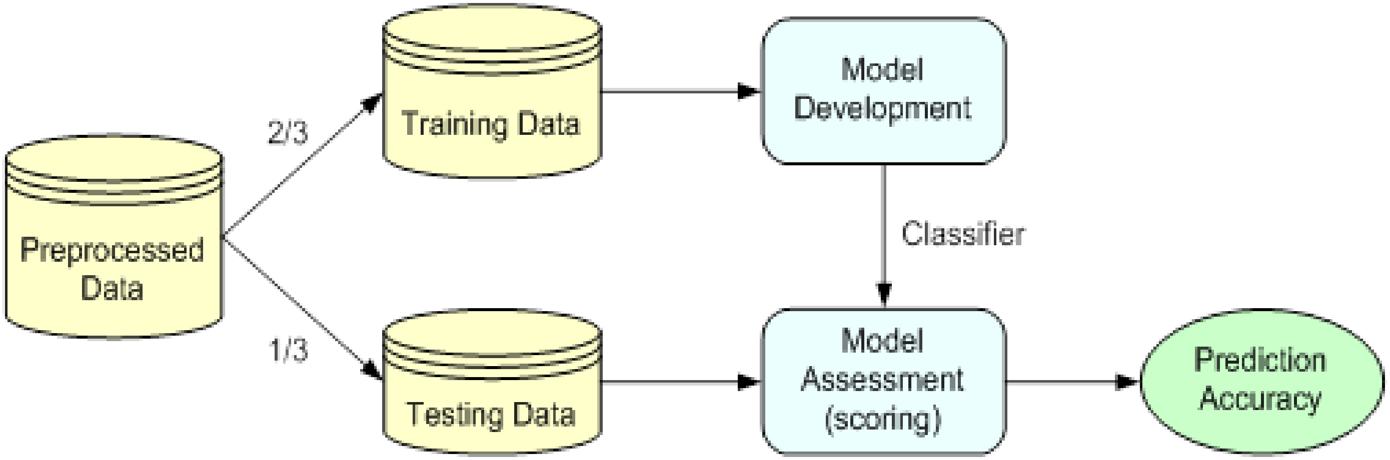
Goals of evaluation

- Measure the classifier performance
- Evaluate the ability of a classifier to generalize what it was learned from training on the new unseen instances
- Choose the most appropriate learning scheme for a specific problem

# Evaluation strategies

## Holdout method

- Split data into 2 independent sets, one third for testing and the rest for training

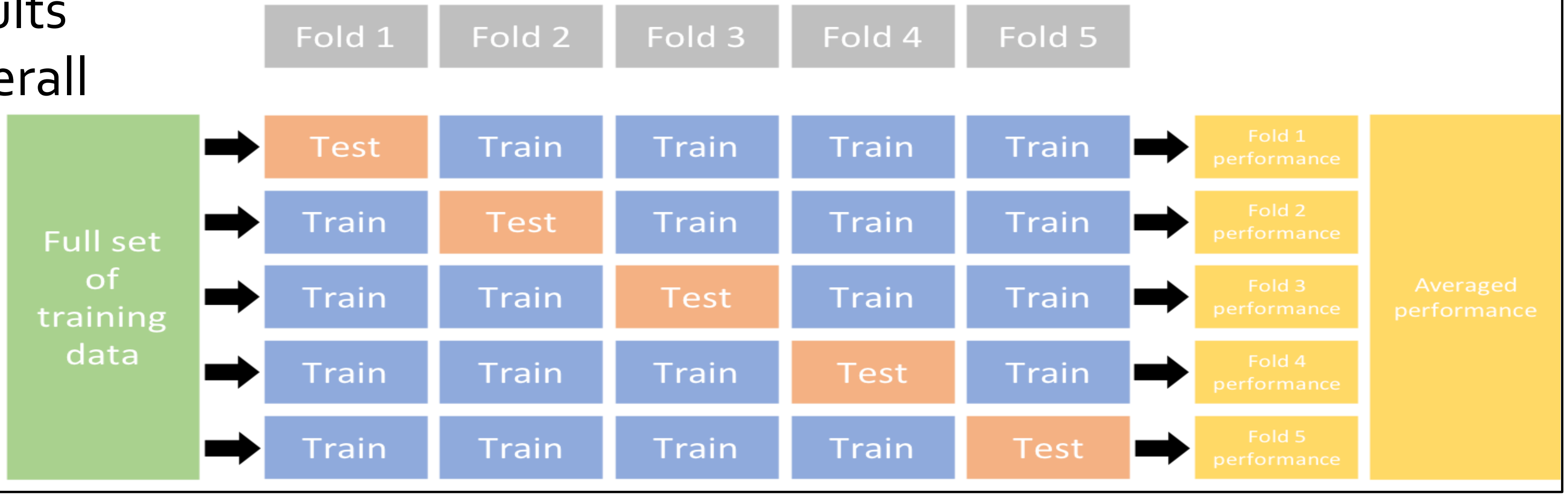



- Repeated holdout method: a variation of holdout
  - Multiple iterations of holdout, the overall accuracy is the average the accuracies obtained at each iteration

# Evaluation strategies

## k-fold cross-validation

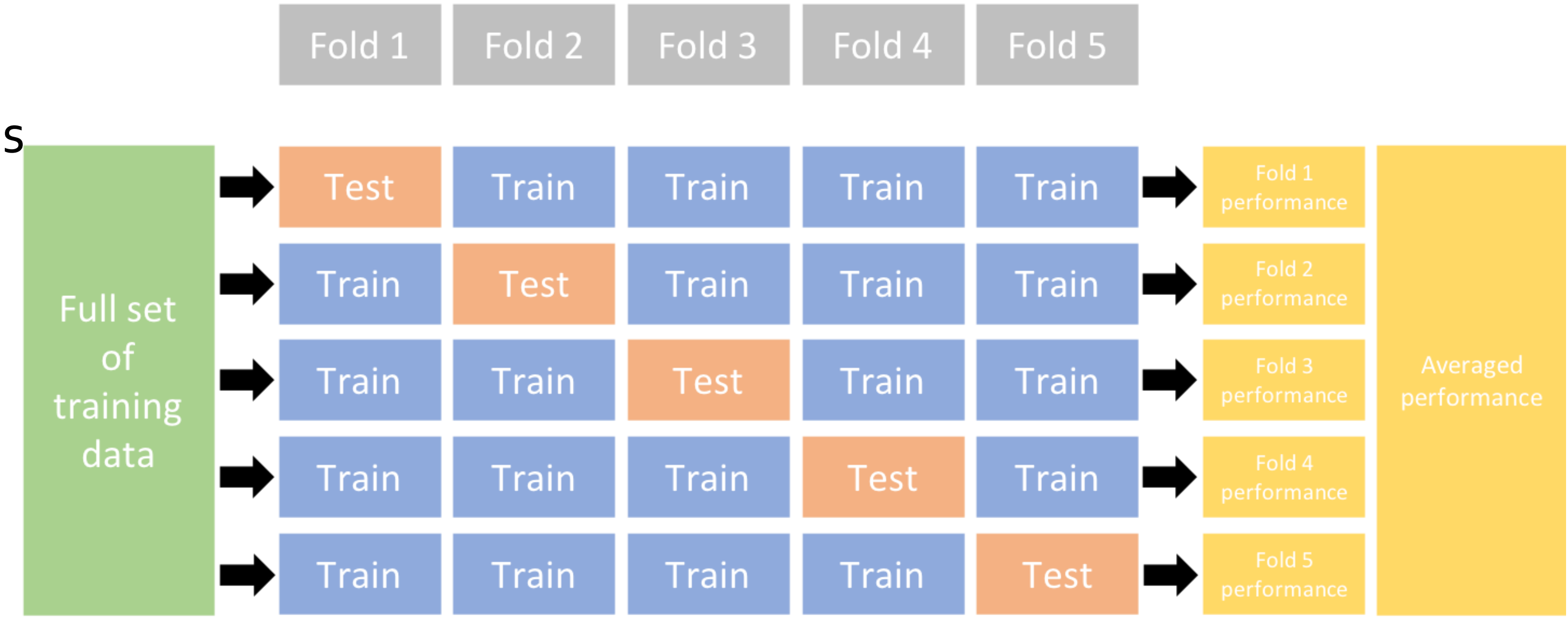
- Randomly split data into k disjoint subsets (folds) of approximately equal size
- Use k-1 subsets as training data and one subset as testing data
- Repeat k times
- Aggregate the results to estimate the overall accuracy



# Evaluation strategies

## Stratified k-fold cross-validation

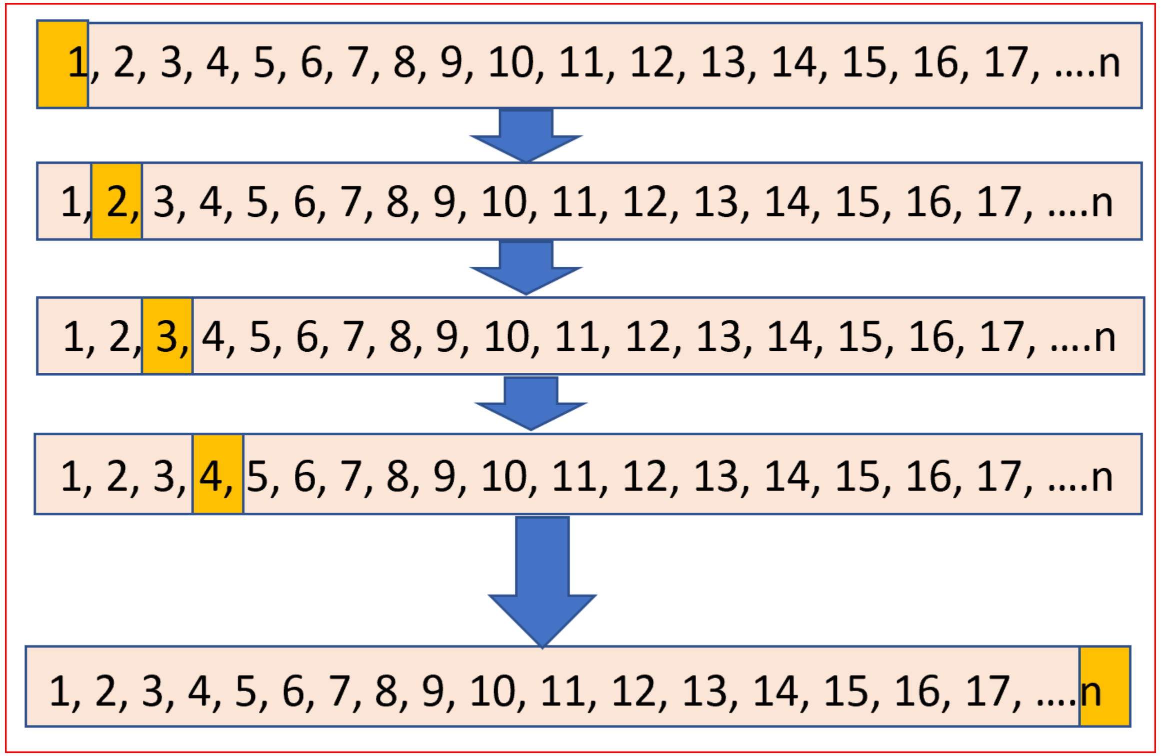
- Stratification ensures that each class is equally represented across each fold
- Useful for imbalanced datasets
- Another variation:
  - Repeated stratified cross validation



# Evaluation strategies

## Leave one out

- A particular form of k-fold cross-validation
- Number of folds = number of training samples
- Mainly used for small datasets



# Classification outputs

- **Class output** : Algorithms like SVM and KNN create a class output. For instance, in a binary classification problem, the outputs will be either 0 or 1.

- **Probability output** : Algorithms like Logistic Regression, Random Forest, Gradient Boosting, Adaboost etc. give probability outputs. Probability outputs can be converted to class output by creating a threshold probability.

# Measures of performance

## Confusion matrix

- A summary of prediction results
- Given ***k*** classes, an entry, $\boldsymbol{CM_{i,j}}$ in a confusion matrix indicates the number of tuples in class ***i*** that were labeled by the classifier as class ***j***

Total predictions $n = 921$

| | Predicted class POSITIVE (spam) | Predicted class NEGATIVE (normal) | |
|---|---|---|---|
| Actual class POSITIVE (spam) | TRUE POSITIVE (TP) 320 | FALSE NEGATIVE (FN) 43 | **363** |
| Actual class NEGATIVE (normal) | FALSE POSITIVE (FP) 20 | TRUE NEGATIVE (TN) 538 | **558** |
| | **340** | **581** | |

# Measures of performance

- Sample S is
  - **Positive** if predicted as spam
    - **True Positive** if it is actually spam (correct)
    - **False Positive** if is actually not spam (error)
  - **Negative** if predicted as not spam
    - **True Negative** if it is actually not spam (correct)
    - **False Negative** if it is actually a spam (error)

| Total predictions $n = 921$ | Predicted class POSITIVE (spam ) | Predicted class NEGATIVE (normal ) | |
|---|---|---|---|
| Actual class POSITIVE (spam ) | TRUE POSITIVE (TP) 320 | FALSE NEGATIVE (FN) 43 | 363 |
| Actual class NEGATIVE (normal ) | FALSE POSITIVE (FP) 20 | TRUE NEGATIVE (TN) 538 | 558 |
| | 340 | 581 | |

Incorrect predictions

Correct predictions

# Measures of performance

- Accuracy: % of samples that are correctly classified

$$Accuracy = \frac{\#\ of\ correct\ predictions}{Total\ predictions} = \frac{TP + TN}{n}$$

- Classification error: % of samples that are incorrectly classified (1- accuracy)

$$Classification\ error = \frac{\#\ of\ incorrect\ predictions}{Total\ predictions} = \frac{FP + FN}{n}$$

| $n = 921$ | Predicted class POSITIVE (spam) | Predicted class NEGATIVE (normal) | |
|---|---|---|---|
| Actual class POSITIVE (spam) | TRUE POSITIVE (TP) 320 | FALSE NEGATIVE (FN) 43 | 363 |
| Actual class NEGATIVE (normal) | FALSE POSITIVE (FP) 20 | TRUE NEGATIVE (TN) 538 | 558 |
| | 340 | 581 | |

Incorrect predictions

Correct predictions

# Measures of performance

- **Precision :** a measure of *exactness*
  - Number of items correctly identified as positive out of total items identified as positive

- Precision = P(Positive sample| sample is classified as positive)

$$\boldsymbol{Precision} = \frac{TP}{TP + FP}$$

| $n = 921$ | Predicted class POSITIVE (spam ) | Predicted class NEGATIVE (normal ) | |
|---|---|---|---|
| Actual class POSITIVE (spam ) | TRUE POSITIVE (TP) 320 | FALSE NEGATIVE (FN) 43 | 363 |
| Actual class NEGATIVE (normal ) | FALSE POSITIVE (FP) 20 | TRUE NEGATIVE (TN) 538 | 558 |
| | 340 | 581 | |

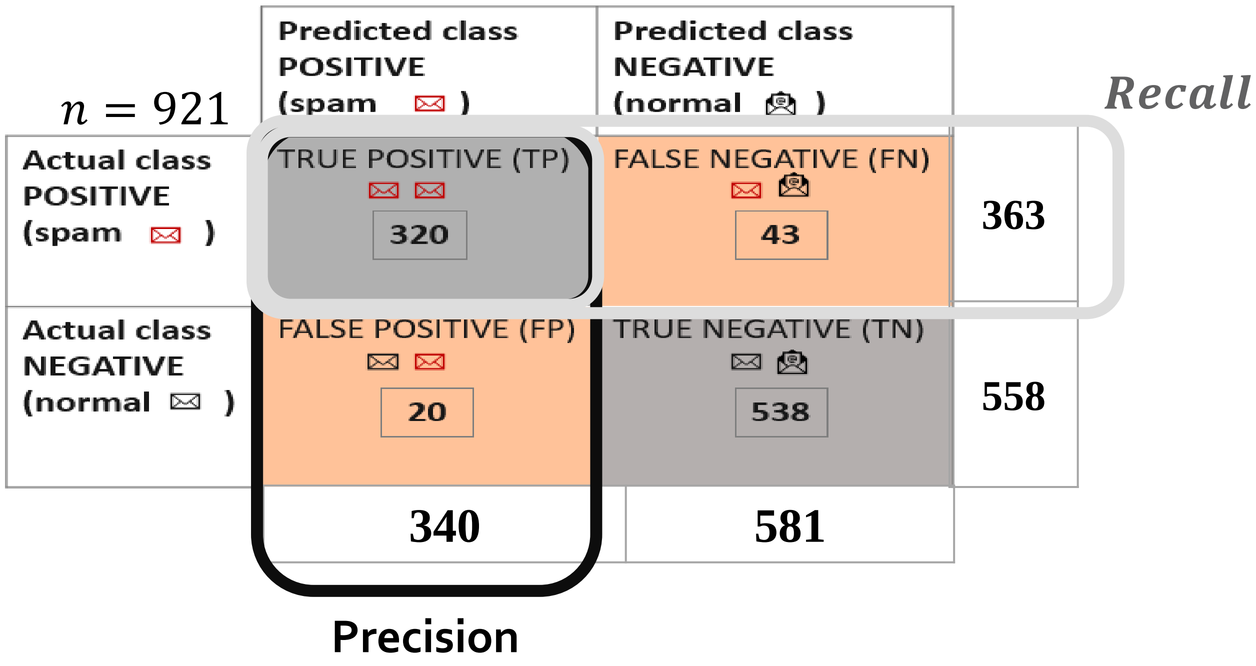
**Precision**

# Measures of performance

- **Recall**: a measure of *completeness*
  - Number of items correctly identified as positive out of total actual positives

- Recall = P(correctly classified | positive sample)

$$\boldsymbol{Recall} = \frac{TP}{TP + FN}$$



# Measures of performance

## Recall / Precision

- Classifier with better precision and recall is a better model
- Inverse relationship

  As FP decrease, FN increase (recall decreases) and vice versa
- **High recall, low precision**: Most of the positive examples are correctly recognized (low FN) but there are a lot of false positives.
- **Low recall, high precision**: We miss a lot of positive examples (high FN) but those we predict as positive are indeed positive (low FP).

$$\boldsymbol{Precision} = \frac{TP}{TP + FP}$$

$$\boldsymbol{Recall} = \frac{TP}{TP + FN}$$

# Measures of performance

## F1 score

- F1 score relies on both recall and precision
  - The mean harmonic of recall and precision
  - Will always be nearer to the smaller value.

$$\boldsymbol{Precision} = \frac{TP}{TP + FP}$$

$$\boldsymbol{Recall} = \frac{TP}{TP + FN}$$

$$\boldsymbol{F1} = 2\ \frac{recall * precision}{recall + precision}$$

# Measures of performance

- Logarithmic loss
  - To be used with probability outputs
  - **Log Loss** takes into account the uncertainty of prediction based on how much it varies from the actual label.
    - It gives a more nuanced view into the performance of the model.

*No. of datapoints*

*Actual class label ('0' or '1')*

$$-\frac{1}{n}\sum_{i=1}^{n}\left\{(\log(p_i) * y_i) + (1-y_i)*\log(1-p_i)\right\}$$

*Prob. score for $i^{th}$ datapoint*

**(a) log-loss formulae for binary classification**

No. of classes

$$-\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{c} y_{ij}\log(p_{ij})$$

Prob. that $x_i \in$ class j

= 1 if $x_i \in$ class j
= 0 otherwise

**(a) log-loss formulae for multi-class classification**

# Measures of performance

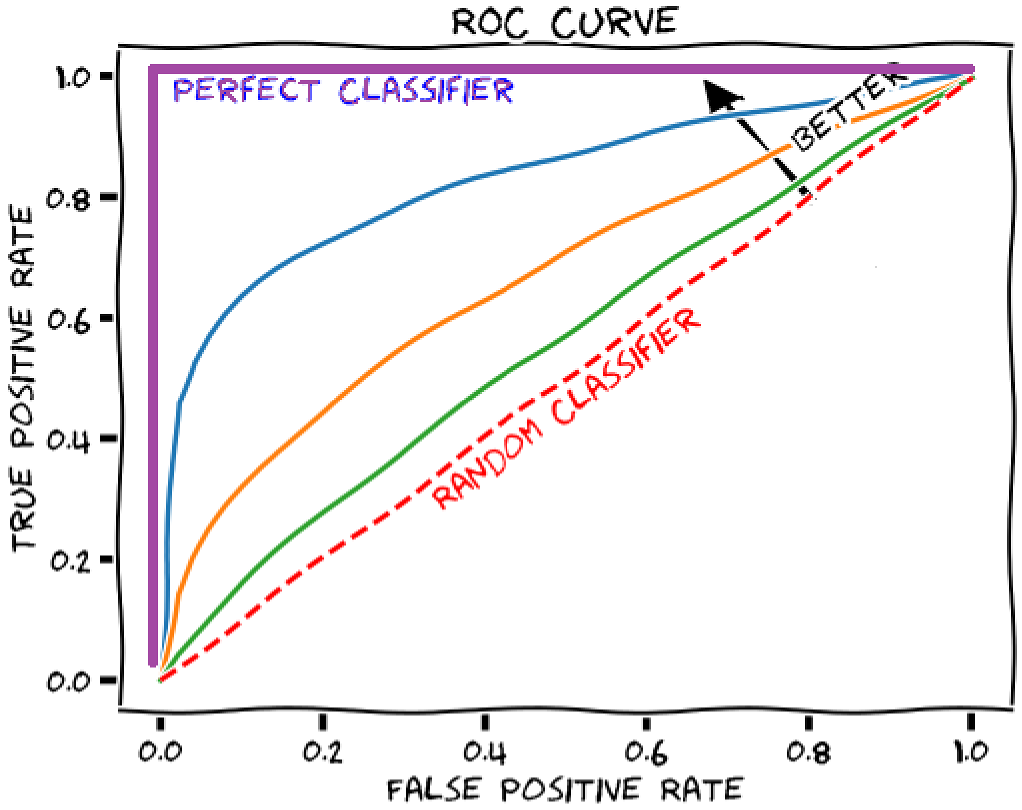
## Receiver Operating Characteristic (ROC) curve

- True Positive Rate, recall or sensitivity

$$\boldsymbol{TPR} = \frac{TP}{P} = \frac{TP}{TP + FN}$$

- True Negative Rate or specificity

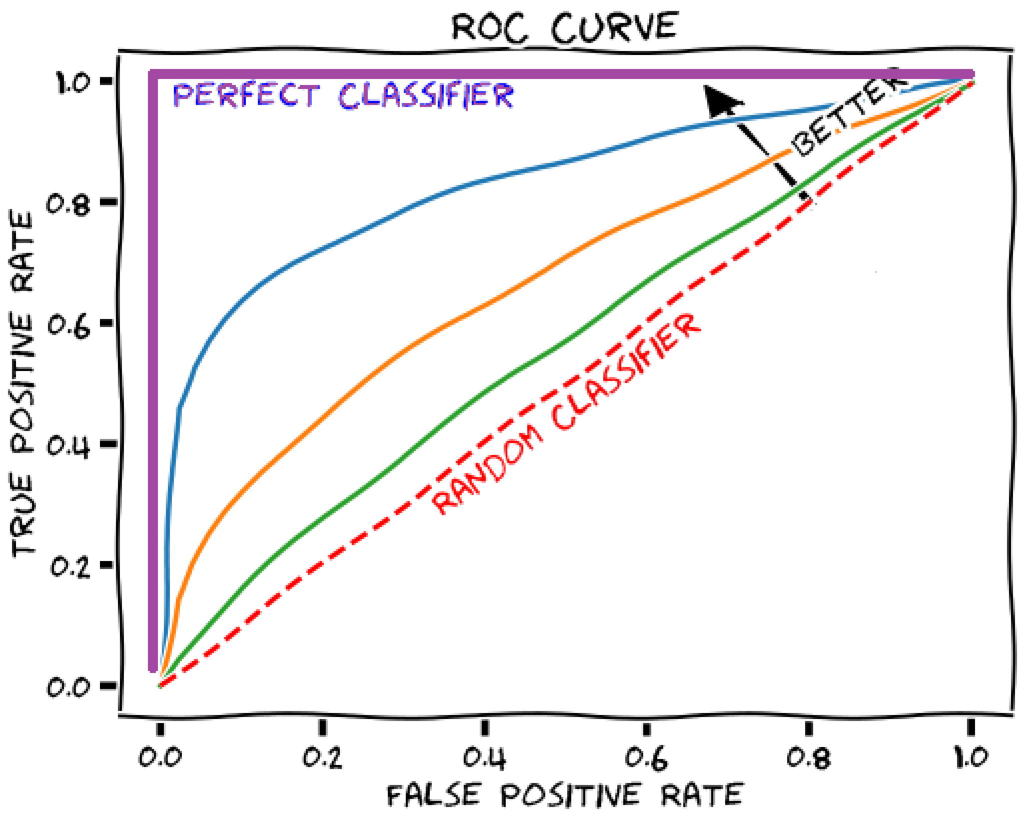
$$\boldsymbol{TNR} = \frac{TN}{N} = \frac{TN}{TN + FP}$$



# Measures of performance

## Receiver Operating Characteristic (ROC) curve

- Performance measured by AUC: area under curve
- Represent the model's ability to discriminate between positives and negatives
- Higher the AUC, better the model is at predicting positives as positives and negatives as negatives.



# Comparing metrics

| | samples | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **Actual class** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **Model 1** | **Predicted probability** | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.6 | 0.6 | 0.5 | 0.5 | 0.9 | 0.9 | 0.9 | 0.9 | 0.9 | 0.9 |
| | **Predicted class (threshold=0.5)** | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| | **Predicted class (threshold=0.7)** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| **Model 2** | **Predicted probability** | 0.6 | 0.6 | 0.6 | 0.6 | 0.6 | 0.6 | 0.6 | 0.6 | 0.7 | 0.7 | 0.7 | 0.7 | 0.7 | 0.8 | 0.8 | 0.8 |
| | **Predicted class (threshold=0.5)** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| | **Predicted class (threshold=0.7)** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

Model 2 performs better than Model 1

# Compring metrics

| Model 1 (threshold=0.5) | Predicted positives | Predicted negatives |
|---|---|---|
| **Actual positives** | 8 | 0 |
| **Actual negatives** | 2 | 6 |

- 14  correct predictions from a total of 16
- 2 incorrect predictions
- Accuracy = 0.875
- Precision = 0.8
- Recall = 1
- F1 = 0.88

| Model 1 (threshold=0.7) | Predicted positives | Predicted negatives |
|---|---|---|
| **Actual positives** | 6 | 2 |
| **Actual negatives** | 0 | 8 |

- 14  correct predictions from a total of 16
- 2 incorrect predictions
- Accuracy = 0.875
- Precision = 1
- Recall = 0.75
- F1 = 0.86

# Compring metrics

| Model 2 (threshold=0.5) | Predicted positives | Predicted negatives |
|---|---|---|
| **Actual positives** | 8 | 0 |
| **Actual negatives** | 8 | 0 |

- 8  correct predictions from a total of 16
- 8 incorrect prediction
- Accuracy = 0.5
- Precision = 0.5
- Recall = 1
- F1 = 0.66

| Model 2 (threshold=0.7) | Predicted positives | Predicted negatives |
|---|---|---|
| **Actual positives** | 8 | 0 |
| **Actual negatives** | 0 | 8 |

- 16  correct predictions from a total of 16
- 0 incorrect prediction
- Accuracy = 1
- Precision = 1
- Recall = 1
- F1 = 1

# Compring metrics

| | F1 (threshold=0.5) | F1 (threshold=0.7) | ROC-AUC | Log-Loss |
|---|---|---|---|---|
| Model 1 | 0.88 | 0.86 | 0.94 | 0.28 |
| Model 2 | 0.66 | 1 | 1 | 0.60 |

*Is better in predicting class probabilities*

*Very big difference from actual labels*

- Log-Loss is useful to compare models in terms of probabilistic outcome
- F1 score and ROC-AUC are useful if we care only about final class predictions.
  - F1 score is sensitive to threshold and should be tuned first before comparing the models
  - AUC score can be used if we don't want to tune threshold.

# Comparing metrics

## Example

| | | Predicted class | | |
|---|---|---|---|---|
| | | POSITIVE (cancer = yes) | NEGATIVE (cancer = no) | Total |
| Actual class | POSITIVE (cancer = yes) | 90 | 210 | 300 |
| | NEGATIVE (cancer = no) | 140 | 9560 | 9700 |
| | Total | 230 | 9770 | 10000 |

Accuracy = 96,5%

Precision = 39,13%

Recall = 30%

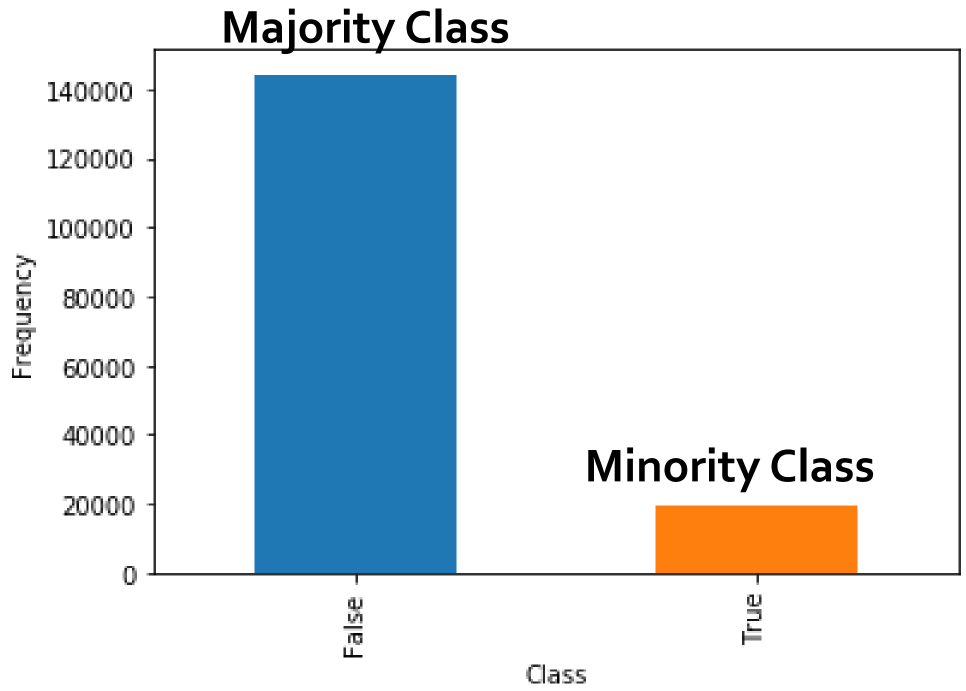
F1-score = 33,96%

Accuracy is a misleading metric

| Class distribution |
|---|
| 3% |
| 97% |

**The class imbalance problem !!**

# Imbalanced classification




**A classification problem where the distribution of examples across the classes is not equal**

**Slight imbalance**: distribution uneven by a small amount

**Severe imbalance**: distribution uneven by a large amount

Ex: Anomaly detection, fraud detection, outlier detection, …

# Imbalanced classification

- Causes of class imbalance
  - A property of the problem domain (some events are naturally of low occurrence )
  - Biased sampling
  - Measurement errors
- The learning process of most classification algorithms is often biased toward the majority class examples, so that minority ones are not well modeled into the final system.
- Approaches to deal with class imbalance
  - Sampling based (oversampling/undersampling)
  - Cost based (redefine cost function such as to penalize FN more that FP)

# Metrics and class imbalance

| S.No. | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Actual (Imbalanced) | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| Predicted (Model 1) | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.9 | 0.9 |
| Predicted (Model 2) | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.9 | 0.9 | 0.9 | 0.9 |

| Model 1 | Predicted positives | Predicted negatives |
|---|---|---|
| Actual positives | 2 | 1 |
| Actual negatives | 0 | 13 |

| Model 2 | Predicted positives | Predicted negatives |
|---|---|---|
| Actual positives | 3 | 0 |
| Actual negatives | 1 | 12 |

We care more about the positive class so Model 2 is better that Model 1

# Metrics and class imbalance

| S.No. | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Actual (Imbalanced) | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| Predicted (Model 1) | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.9 | 0.9 |
| Predicted (Model 2) | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.9 | 0.9 | 0.9 | 0.9 |

| | F1 (threshold=0.5) | ROC-AUC | Log-Loss |
|---|---|---|---|
| **Model 1** | 0.8 | 0.83 | 0.24 |
| **Model 2** | 0.86 | 0.96 | 0.24 |

Log-Loss does not differentiate between the two models. Treats both errors equally

Log-Loss is not a good metric if classes are imbalanced

# Metrics and class imbalance

| S.No. | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Actual (Imbalanced – few positive) | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| Predicted (Model 1) | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.9 | 0.9 |
| Predicted (Model 2) | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.9 | 0.9 | 0.9 | 0.9 |
| | | | | | | | | | | | | | | | | |
| Actual (Imbalanced – few negative) | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Predicted (Model 3) | 0.1 | 0.1 | 0.9 | 0.9 | 0.9 | 0.9 | 0.9 | 0.9 | 0.9 | 0.9 | 0.9 | 0.9 | 0.9 | 0.9 | 0.9 | 0.9 |
| Predicted (Model 4) | 0.1 | 0.1 | 0.1 | 0.1 | 0.9 | 0.9 | 0.9 | 0.9 | 0.9 | 0.9 | 0.9 | 0.9 | 0.9 | 0.9 | 0.9 | 0.9 |

# Metrics and class imbalance

| | F1 (threshold=0.5) | ROC-AUC | Log-Loss |
|---|---|---|---|
| **Model 1** | 0.8 | 0.83 | 0.24 |
| **Model 2** | 0.86 | 0.96 | 0.24 |
| **Model 3** | 0.963 | 0.83 | 0.24 |
| **Model 4** | 0. 96 | 0.96 | 0.24 |

No difference between the two cases (few negatives vs few positives)

Not much difference because of the large number of positives

ROC-AUC is a good metric for imbalance classification if you care about the minority class

F1 gives more importance to the positive class

# Metrics and class imbalance

POSITIVE (cancer = yes)
NEGATIVE (cancer = no)

| **Model 1** | **Predicted POSITIVE** | **Predicted NEGATIVE** |
|---|---|---|
| **Actual POSITIVE** | 270 | 30 |
| **Actual NEGATIVE** | 100 | 9600 |

| **Model 2** | **Predicted POSITIVE** | **Predicted NEGATIVE** |
|---|---|---|
| **Actual POSITIVE** | 15 | 285 |
| **Actual NEGATIVE** | 100 | 9600 |

- Model 1 performs very much better at detecting the positive class (predicts 90% out of 300)
- Model 2 is a **very bad** predictor

| | **F1** | **ROC-AUC** |
|---|---|---|
| Model 1 | 0.80 | 0.94 |
| Model 2 | 0.07 | $\boxed{0.51}$ |

ROC-AUC does not reflect the bad performance of Model 2

F1 is the appropriate metric if you care about the positive class

# Measures of performance

## Confusion matrix : case multiclass classification

| | | Predicted Class | |
|---|---|---|---|
| | | Positive | Negative |
| Actual Class | Positive | TP | FN |
| | Negative | FP | TN |

| | | Predicted Class | | | |
|---|---|---|---|---|---|
| | | $C_1$ | $C_2$ | ... | $C_N$ |
| Actual Class | $C_1$ | $C_{1,1}$ | FP | ... | $C_{1,N}$ |
| | $C_2$ | FN | TP | ... | FN |
| | ... | ... | ... | ... | ... |
| | $C_N$ | $C_{N,1}$ | FP | ... | $C_{N,N}$ |

- One class is defined as positive and the others as negative.
- The performance measures are computed in exactly the same way.
- Recall and precision and F1 are computed for each class

# Measures of performance

Confusion matrix : case of multiclass classification

- Overall performance of the classifier is calculated using the weighted averages
- Macro average: Equal weight is given to all classes
- Weighted average: classes are given different weights
- Micro average: Equal weight is given to samples regardless of their class

# Measures of performance

## Confusion matrix : case of multiclass classification

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Cat | 0.308 | 0.667 | 0.421 | 6 |
| Fish | 0.667 | 0.200 | 0.308 | 10 |
| Hen | 0.667 | 0.667 | 0.667 | 9 |
| accuracy | | | 0.480 | 25 |
| macro avg | 0.547 | 0.511 | 0.465 | 25 |
| weighted avg | 0.581 | 0.480 | 0.464 | 25 |

| label | tp | fp | fn | precision | recall |
|---|---|---|---|---|---|
| $c_1$ | 3 | 2 | 7 | 0,6 | 0,3 |
| $c_2$ | 1 | 7 | 9 | 0,12 | 0,1 |
| total | 4 | 9 | 16 | | |
| Macro-avereged | | | | 0,36 | 0,2 |
| Micro-avereged | | | | 0,31 | 0,2 |

$$Macro_{precision} = \frac{1}{2} \times \left(\frac{3}{3+2} + \frac{1}{1+7}\right) = 0,36$$

$$Micro_{precision} = \frac{4}{4+9} = 0,31$$

# Regression metrics

- Regression models have continuous output. So, we need a metric based on calculating some sort of distance between predicted and ground truth.
- The performance of a Regression model is reported as errors in the prediction.
- In order to evaluate Regression models, we can use the following metrics:
  - Mean Absolute Error (MAE),
  - Mean Squared Error (MSE),
  - Root Mean Squared Error (RMSE),
  - $R^2$ (R-Squared).

# Model performance

## Accuracy is not EVERYTHING

- Other evaluation criteria
  - **Speed**: refers to the computation costs involved in generating and using the model
  - **Robustness**: refers to the ability of the model to make correct predictions given noisy data or data with missing values
  - **Scalability**: refers to the ability to construct the model efficiently given large amount of data
  - **Interpretability**: refers to the level of understanding and insight that is provided by the model