Supervised machine learning

UNIVERSITÀ DI PISA

Supervised Machine Learning

UNIVERSITÀ DI PISA

A machine is said to *learn* if, when tackling a task, it is able to improve its own performance through experience.

- Task T: the problem we are trying to solve, e.g., predict the cancer risk of a patient
- Experience E: the experience on the task provided to the model, e.g., some dataset
- Performance P: a measure of success, e.g., the success rate in predicting cancer
- Model F: a function $f(e)$ solving the task with a learning algorithm A

Performance is often akin to *loss*, or *error*.

Machine Learning: tasks

Università di Pisa

Machine learning tackles two families of tasks: those whose solutions we do not know, or can't express algorithmically.

Task	Predict	Example
Binary classification	one of two discrete labels	Is the patient at high risk of developing cancer, or not?
Multilabel classification	any of several discrete labels	Of all the possible syndromes, which is the patient going to develop?
Multiclass classification	one of several discrete labels	The student is going to major in...?
Regression	a continuous label	The student's grade is going to be...?

Task categorization admits also *unsupervised* and *reinforce* tasks, and several variations of supervised tasks.

Experience

Usually a dataset (X, Y) of data X and labels Y.

Task	Description	Experience
Binary classification	Is the patient at high risk of developing cancer, or not?	Patients' clinical history
Multiclass classification	The student will major in...?	Students interests, grades in different subjects, and majors
Regression	The student's grade will be...?	Grades, hours studying, and interest

Experience

Università di Pisa

Task	Description	Experience
Binary classification	Is the patient at high risk of developing cancer, or not?	Patients' clinical history

An example dataset with label *Risk*.

Surname	Birth year	BMI	Blood pressure	VO_2max	Risk
Pogacar	1998	21.3	140	89	Low
...	...	...	...	...	...

Experience

Università di Pisa

Task	Description	Experience
Regression	The student's grade will be...?	Grades, hours studying, and interest

An example dataset with label *Grade*.

Surname	Birth year	Course	Interest	Hours	GPA	Grade
Beretta	1988	Greek history	3	123	3.8	27
...	...	...	...		...	...

A running example

We want to develop a model to detect early onset of cancer. We are given an historic dataset of cancer patients and their biomarkers (the experience E), and wish to create a statistical model (the model F) which predicts early cancer onset (task T), minimizing the risk of undetected cancer cases (performance P).

With supervised learning, we aim to *learn* a model that solves the task on any **unknown** experience. In our example, we want to learn a function f that, given a patient x, computes a label $y = f(x)$ stating whether the patient will develop cancer or not.

Models and generalization

Università di Pisa

Models are not developed to aid on known experiences, rather on *unknown* ones. The performance of a model **can't** be uniquely measured on its performance on the given experience, but rather on *novel* experiences which the model was not preview to. We want to achieve a low **generalization** error.

Optimization
Maximizes performance on the given experience E: *optimization* performance

vs

Machine Learning
Maximizes performance on the given, and expected *non-given*, experience E and $\bar{E}$: *generalization* performance

Experience... not given?

If we do not know the non-given experience $\bar{E}$... how can we ever expect to be effective on it?

Equal distribution assumption. It is assumed that the given experience E, and the non-given experience $\bar{E}$ are sampled from the same distribution $\Pr^{E}$.

Given a learning algorithm A, can I always expect the performance to transfer?

The random process of data sampling

I can't. Sampling from the data distribution is a random process, and the resulting models learned end up erring in terms of:

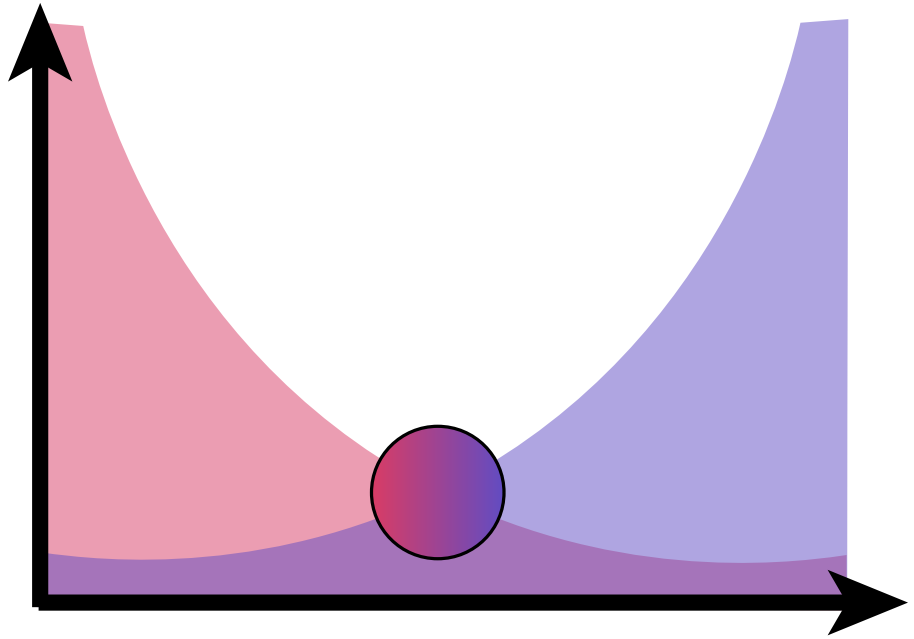
- **Bias**: the expected performance decrease w.r.t. the best model
- **Variance**: the variance with respect to different samples

Ideally, we want to have learning algorithms with low enough bias and variance.

A capacity's view

Two categories improper models:

- **Underfit.** High bias, high variance. Models which have poor performance, regardless of samples. They have *not learnt enough*!
- **Overfit.** Low bias, high variance. Models which have overfit on the given experience, and ought to improve their generalization performance. They have *learnt too much*!

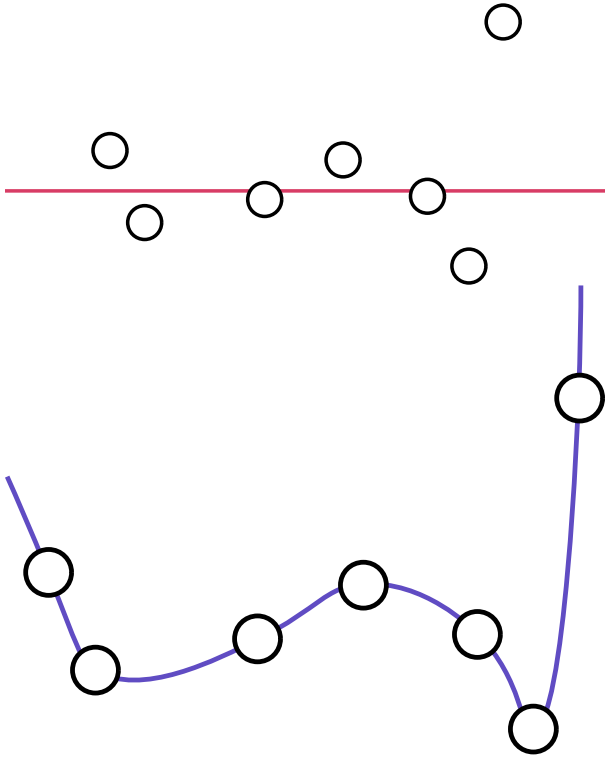


Bias (red) and variance (blue) as a decomposition of model performance.

A capacity's view

Two categories improper models:

- **Underfit.** High bias, high variance. Models which have poor performance, regardless of samples. They have *not learnt enough*!
- **Overfit.** Low bias, high variance. Models which have overfit on the given experience, and ought to improve their generalization performance. They have *learnt too much*!



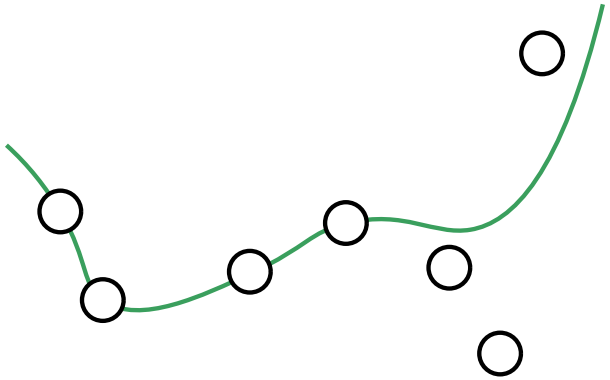
Two models approximating a dataset: one model has too low a capaticy and underfits the data (in red), while another has too high a capacity and overfits the data (in blue).

A capacity's view

Università di Pisa

Two categories improper models:

- **Underfit.** High variance, low bias. Models which ought to improve their performance on the given experience E
- **Overfit.** Low variance, high bias. Models which ought to improve their generalization performance on the unknown experience $\bar{E}$



A properly fit model: it approximates the data well, without being too strict, thus leaving space for some variance in the data.

Searching for models

UNIVERSITÀ DI PISA

- **Tackling the generalization gap.** Data-based strategies to maximize generalization performance
- **Performance evaluation.** How to measure model performance/error
- **Parameterization.** Exploring the space of models

Tackling the generalization gap through regularization

UNIVERSITÀ DI PISA

An obvious solution would be to reduce the capacity of a model. Thus, purposefully constraining the model to reduce variance. This approach, named *regularization*, is highly model-specific, and will be tackled on a model-by-model basis later.

Instead, why not tackle this directly with model-agnostic approaches?

Tackling the generalization gap through data

We design two phases

- **Model selection.** A learning phase wherein, among all possible models in a model space $\mathcal{F}$, we *select* a model f
- **Model validation.** An evaluation phase wherein the generalization performance of the selected model f is estimated

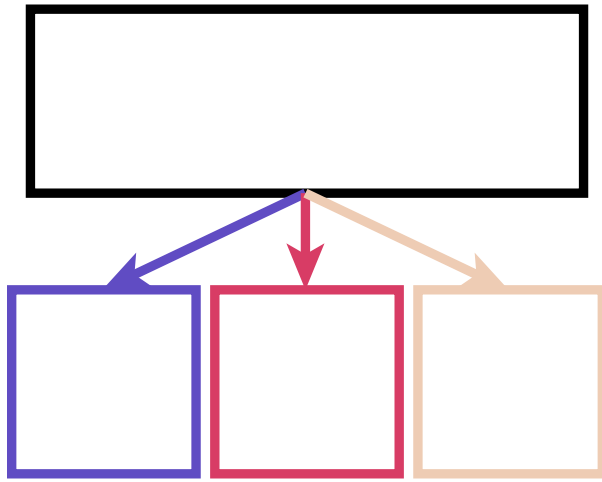
Model validation **cannot** affect model selection: it only goes one way, from selection to validation. Thus, we need to incorporate in model selection some strategy to avoid under/overfitting.

Model selection, data-only

Università di Pisa

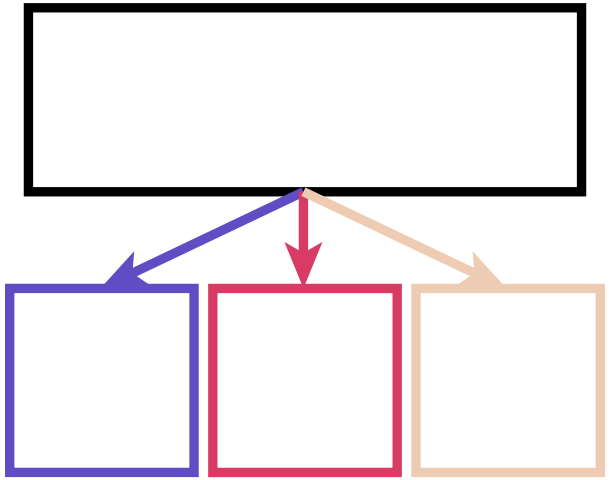
The standard approach is model agnostic: we can apply this to any learning algorithm or family of models we want. We operate a tripartite partitioning of (X, Y):

- **Training dataset** (X^{tr}, Y^{tr}): search through the models' space $\mathcal{F}$
- **Validation dataset** (X^{vl}, Y^{vl}): guesstimate the generalization performance of candidate models $f_1, \ldots, f_k$
- **Test dataset** (X^{ts}, Y^{ts}): estimate the generalization performance of the selected model f_i



A partition of the dataset (in black) in training (blue), validation (red), and test (beige).

Model selection, data-only



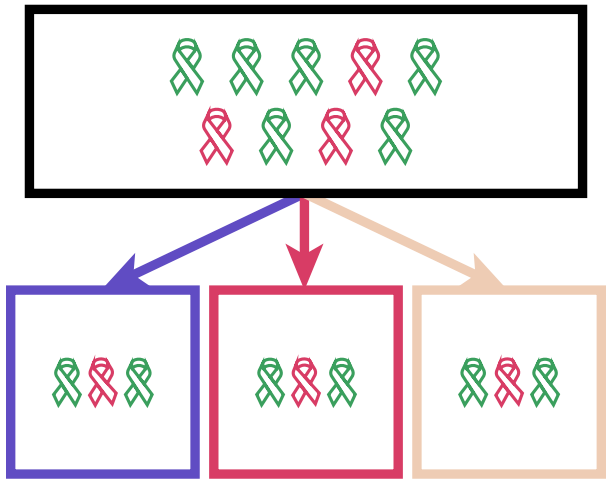
A partition of the dataset (in black) in training (blue), validation (red), and test (beige).

Task	Training	Validation	Test
Disease diagnosis	Biology lectures	Homework	Exam
Learning a language	Duolingo	Exchange student	Living abroad
Pandemic diffusion	Black plague	Ebola	Covid

Model selection, data-only

How to choose the partition?

- **Size.** Test and validation set of similar size, training set of much larger size, e.g., a ratio of 4:1. Some learning algorithms are more data-hungry, so this is a starting baseline.
- **Distribution.** Ideally, same distribution for all three datasets. Random *stratified* sampling is used



A partition of the dataset (in black) in training (blue), validation (red), and test (beige). Patients with (red cross) and without (green cross) cancer are split evenly among the blocks.

Model selection: hold-out

Hold-out leverages the three-blocks partition train-validation test.

1. Learn candidate models $f_1, \ldots, f_k$ on the training set
2. Evaluate them on the validation set
3. Estimate the generalization error on the test set

Model selection: cross validation

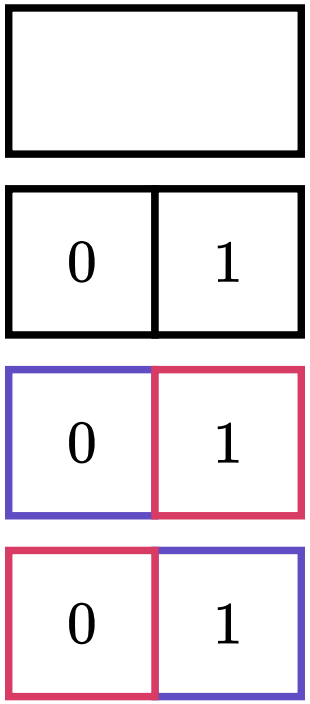
Università di Pisa

Stretching hold-out, we aim to further increase the size of the validation set.

We partition a given set of data, e.g., the training dataset, in two *folds*:

- in set 1, block 1 is a training dataset, block 2 is the validation dataset
- in set 2, block 2 is a training dataset, block 1 is the validation dataset

Now I can learn and guesstimate a model f_i on both folds!



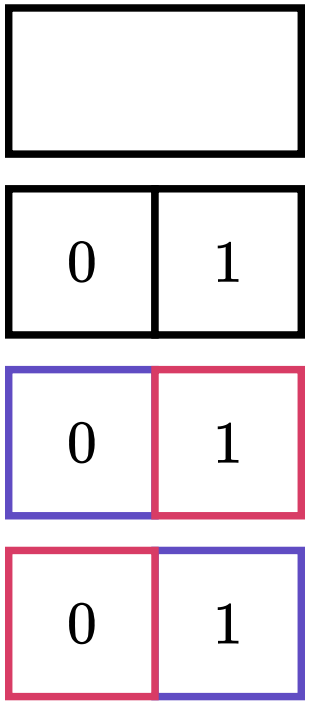
A set partitioned in two blocks, and the resulting *folds*. In each fold, a block acts as validation set (in red), and the other as training set (in blue).

Model selection: cross validation

UNIVERSITÀ DI PISA

- Stronger guesstimates of generalization performance
- Embarrassingly parallel problem

- Reduce validation data available
- For a learning over all folds, requires models trained on different folds to be comparable*



A set partitioned in two blocks, and the resulting *folds*. In each fold, a block acts as validation set (in red), and the other as training set (in blue).

*Rather, models' parameters. We will see parametric models further in the course.

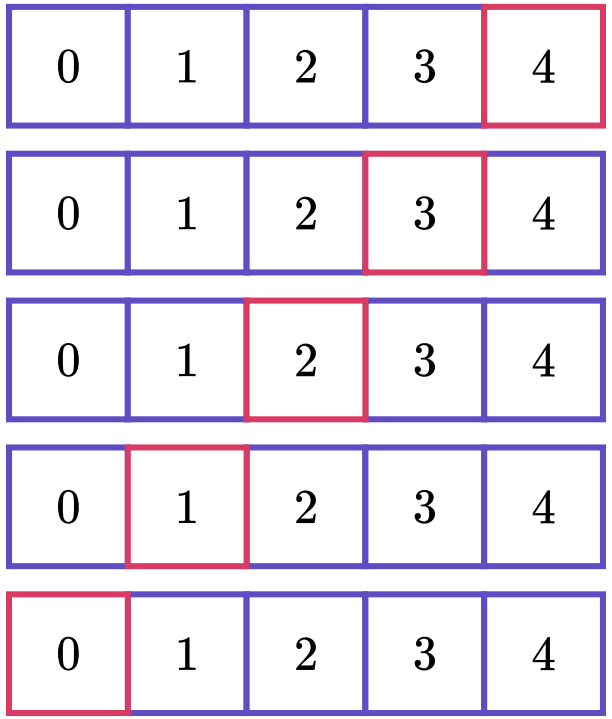
Model selection: k-fold cross validation

Università di Pisa

Why limit ourselves to two folds, when we can employ k?

- Stronger guesstimates of generalization performance
- Embarrassingly parallel problem

- Reduced validation data available
- Large computational cost
- For a learning over all folds, requires models trained on different folds to be comparable*



A set partitioned in $k = 5$ blocks, and the resulting *folds*. In each fold, a block acts as validation set (red), and the other as training set (blue).

*Rather, models' parameters. We will see parametric models further in the course.

Performance evaluation

With proper partitioning, we are now able to feed experiences (data) to the candidate models $f_1, \ldots, f_k$ we wish to select. How do we evaluate them?

Classification. Classification tasks generally aim to measure a Hamming distance ($\ominus$) between the gold labels, and the labels given by the model. Either measured as error (lower is better $\downarrow$) or performance (higher is better $\uparrow$).

Reminder: we indicate the vector of n gold labels with $Y \in \mathcal{Y}$, the model of interest with f, its prediction on an instance x^i with $f(x^i)$, and the indicator function with $\mathbb{1}$.

Hamming distance

Performance evaluation: classification

UNIVERSITÀ DI PISA

Task	Measure	Formulation
Binary, Multiclass	Accuracy	$\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}(Y_i = f(x^i))$ or $1 - Error\ rate$
Binary, Multiclass	Error rate	$\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}(Y_i \neq f(x^i))$ or $1 - Accuracy$
Multilabel	Jaccard similarity	$\frac{1}{n}\sum_{i=1}^{n} \frac{Y_i \cap f(x^i)}{Y_i \cup f(x^i)}$
Multilabel	Hamming error	$\frac{1}{n}\sum_{i=1}^{n} 1 - (Y_i \ominus f(x^i))$

Performance evaluation: classification

For unbalanced data, where labels are not equally probable, it is sensible to adjust model performance by computing metrics on each separate class of labels, then aggregate the results, e.g., through average (balanced accuracy), minimum, maximum, etc.

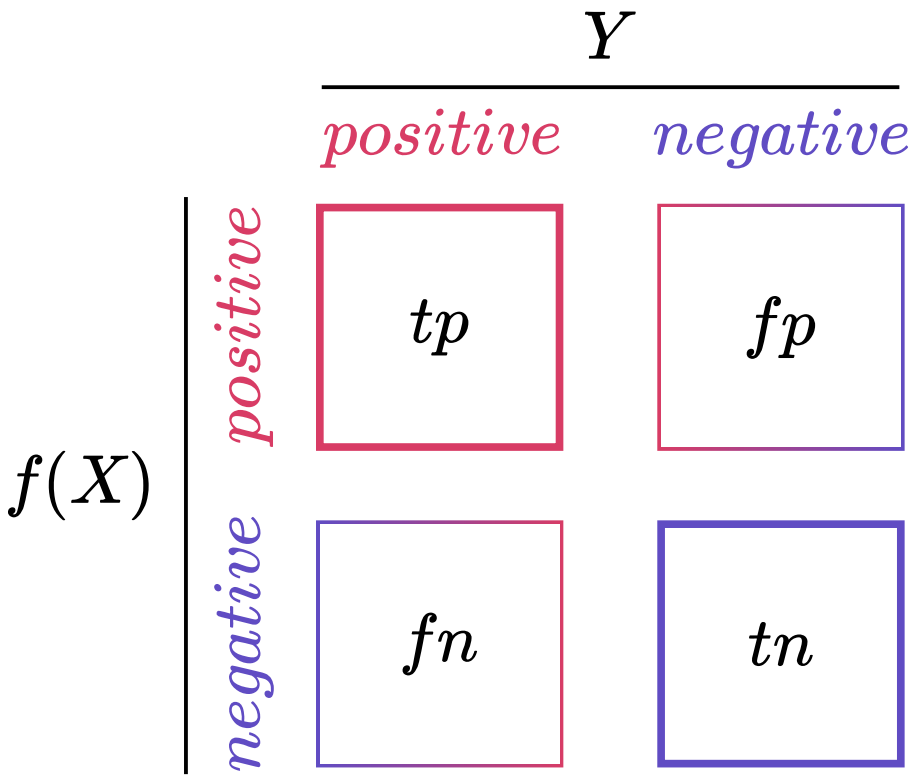
We indicate with $p(Y^i, f)$ the performance p of f on the subset of Y with label i.

Task	Measure	Formulation
Binary, Multiclass	Balanced Accuracy	$\frac{1}{\mid \mathcal{Y} \mid} \sum_{i=1}^{\mid \mathcal{Y} \mid} Accuracy(Y^i, f)$
Binary, Multiclass	Error rate	$1 - Balanced\ Accuracy$
...	...	...

Performance evaluation: classification

Università di Pisa

In some binary cases, one label y is for us of interest (positive label), e.g., patients we predict will have cancer, while the other is not (negative label). We can construct a *confusion matrix* out of the predictions $f(x^i)$ of the model, that we can then leverage to define more performance measures.



A confusion matrix: columns defined by the gold labels Y, and rows defined by the predicted labels $f(X)$.

Performance evaluation: classification

Università di Pisa

Measures derived by the confusion matrix.

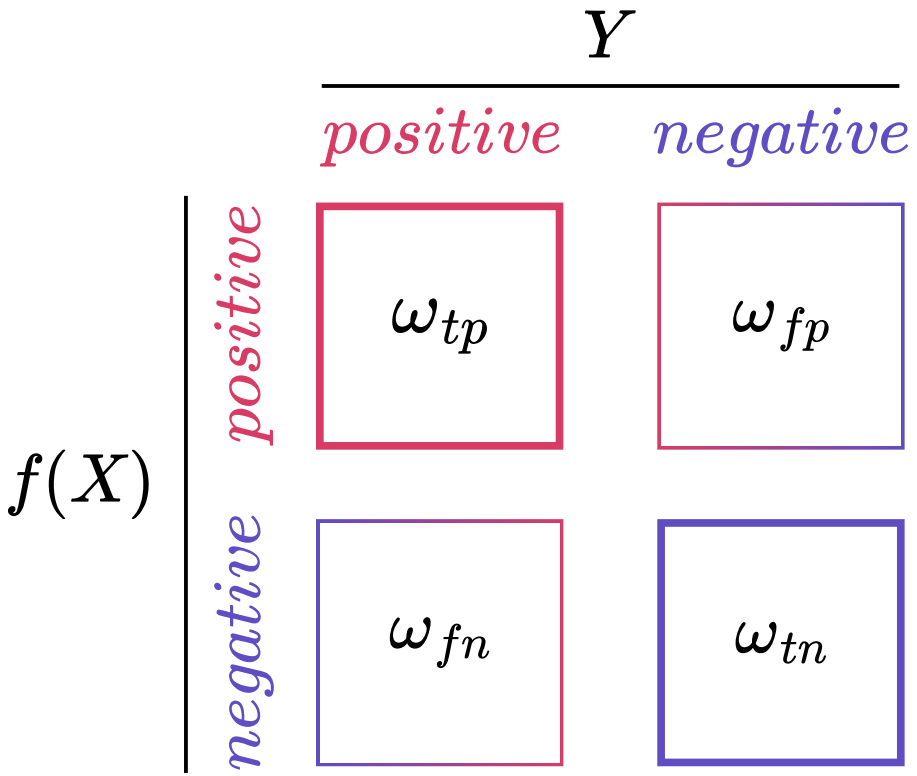
Task	Measure	Formulation	Description
Binary	Precision	$\frac{tp}{tp + fp}$	Of all the positive predictions, how many, in proportion, are correct?
Binary	Recall	$\frac{tp}{tp + fn}$	Of all the positive instances, how many, in proportion, have been correctly predicted?
Binary	$f1$-score	$h(precision, recall)$	Harmonic mean of precision and recall
Binary	Accuracy	$\frac{tp + tn}{tp + tn + fp + fn}$	Accuracy

h indicates the harmonic mean.

Performance evaluation: classification

Università di Pisa

Confusion matrices are limited in their weighting, as any entry, e.g., true positives (tp), has a unitary weight in all performance measures. Yet, in some cases, some false weigh heavier than others, e.g., diagnosing a false positive cancer is far worse than diagnosing a false negative. Thus, we introduce the **cost matrix**, holding one weight per each entry in the confusion matrix, weighing it in performance measures.



A cost confusion matrix: each entry weighs the correspondent entry in the confusion matrix.

Performance evaluation: classification

Binary measures can be extended to the multiclass case through a *one-VS-all* approach: iterating over all possible labels, define the current label as positive, and all others as negative, thus defining a multiclass problem as a set of binary ones. Then, aggregate the performances.

Performance evaluation: classification

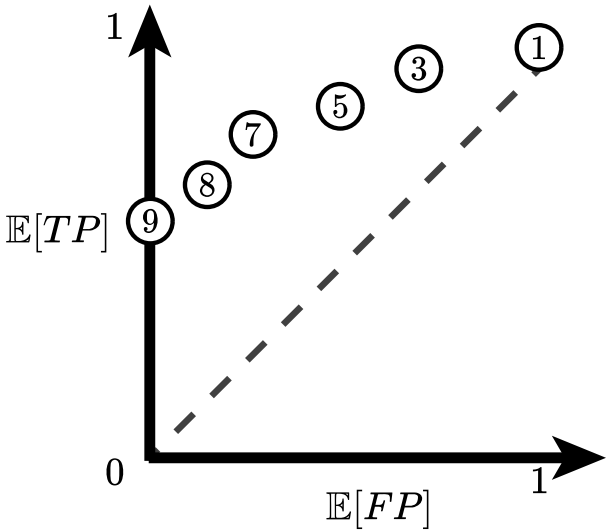
UNIVERSITÀ DI PISA

Aggregation	Descripton
Micro average	Global average
Macro average	Actual per-label average
Weighted average	Macro average, but introduces weights given by proportion of size of the label set

Performance evaluation: classification

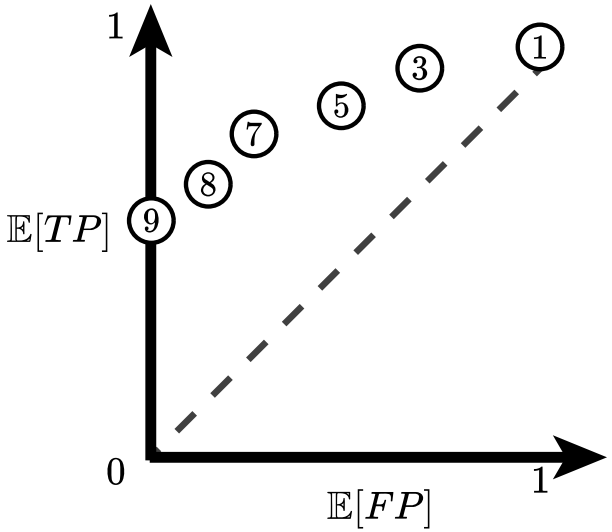
Università di Pisa

Among all possible model solving binary classification tasks, some do not compute a binary label per se, rather a score or probability α of a label, e.g., the positive label. To go from score to label we need to *threshold* such probability, and different thresholds τ induce different labellings, and thus different confusion matrices and probabilities of true or false positives.



A ROC curve, numbers indicating the different thresholds that have generated them. $\mathbb{E}[TP], \mathbb{E}[FP]$, indicate the *rates* of true positives and false positives.

Performance evaluation: classification



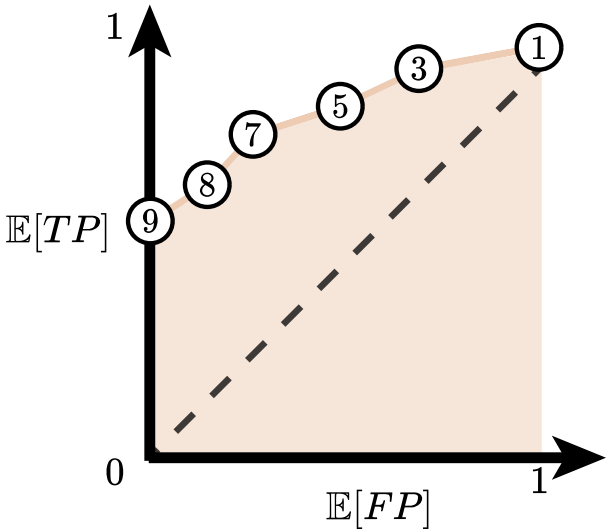
A ROC curve, numbers indicating the different thresholds that have generated them. $\mathbb{E}[TP], \mathbb{E}[FP]$, indicate the *rates* of true positives and false positives.

Points of interest

- $(1, 1)$ Model with no true positives
- $(0, 1)$ Model with no false positives: the best model
- (x, x) Same probability of true and false positives: random models
- $(x, x - \delta)$ Models with flipped labels

Performance evaluation: classification

A ROC curve also allows us to compute a single scalar aggregating the performances at different thresholds: the Area Under the ROC Curve (AUC).



A ROC curve, numbers indicating the different thresholds that have generated them. $\mathbb{E}[TP]$, $\mathbb{E}[FP]$, indicate the *rates* of true positives and false positives. The Area Under the ROC Curve (AUC) in beige.

Performance evaluation: regression

Unlike classification, regression estimates on a *continuous set*, thus we can't leverage the same measures.

Performance	Formulation	Description
Mean Squared Error ↓	$\frac{1}{n}\sum_{i=1}^{n} \mid\mid f(x_i) - Y_i \mid\mid_2^2$	Mean error per instance
Max Squared Error ↓	$\max\{\sum_{i=1}^{n} \mid\mid f(x_i) - Y_i \mid\mid_2^2\}$	Maximum error
R squared ↑	$1 - \frac{e^2}{\sigma^2}$	Error over default model

Performance evaluation: regression

R^2. The two terms of R^2 are

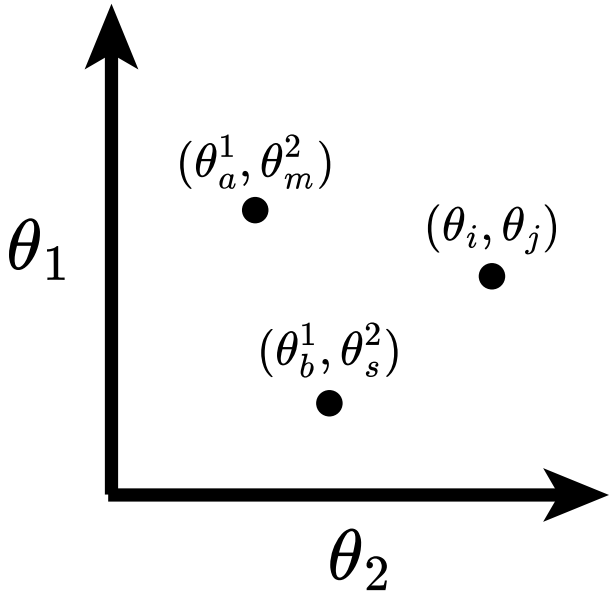
- the model error $e^2 = \sum_{i=1}^{n} || f(x_i) - Y_i ||_2^2$
- the variance of the data σ^2. This would also be the error of a simple model predicting the average

Thus, their ratio compares the error of a model with the one of a default model. The lower such ratio, the better the model, and the closer to 1 the R^2 score. Viceversa, the higher the ratio, the lower the R^2 score.

Supervised learning

We now know...

- given a dataset, how to define what task it describes
- given a dataset, how to partition it into separate blocks to ease generalization in learning
- given a block, what role it serves, and why
- given a task, how to compare different models, choosing the better one(s)



The space of parameters Θ: we need to search this space to find a suitable model.

The last missing ingredient: **defining and optimizing models**.