Feasibility of the learning problem

Oriol Pujol

Machine Learning

August 25, 2022

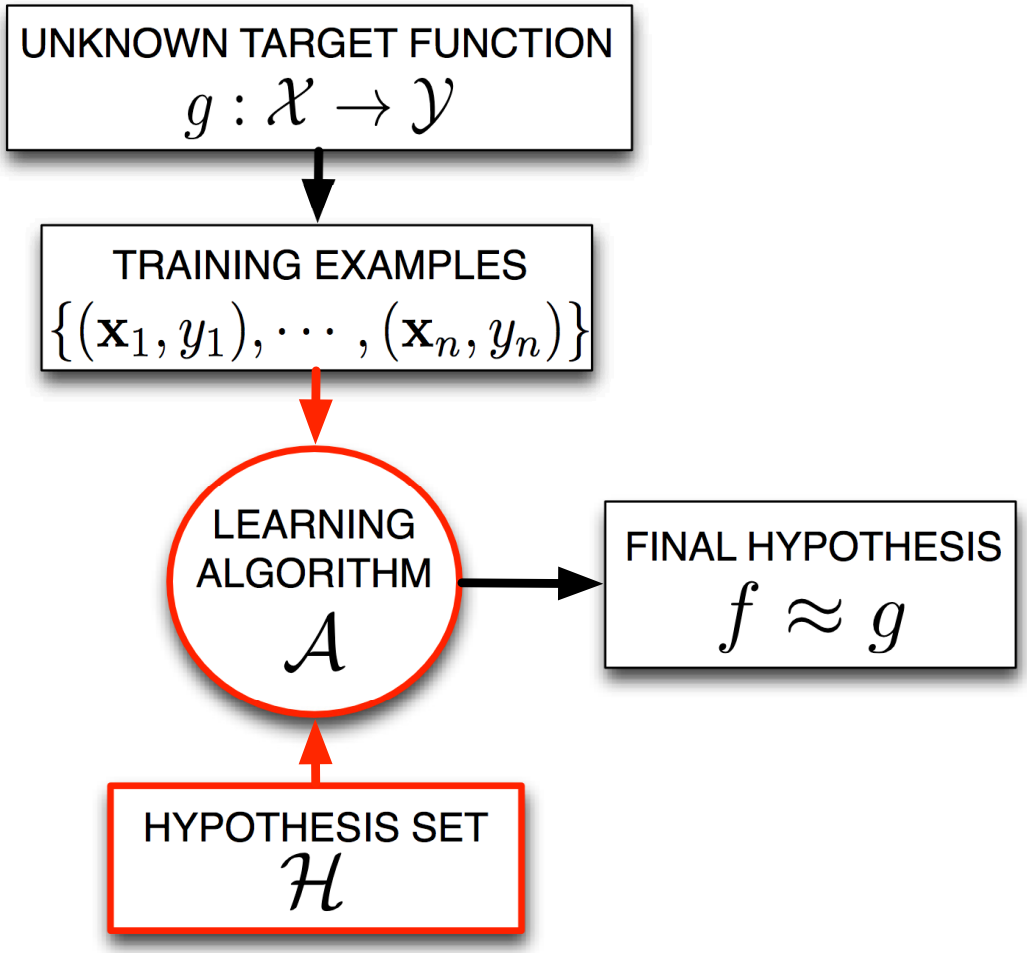
Acknowledgements

This work is inspired in the courses of T. Jaakkola, M. Collins, L. Kaelbling, and T. Poggio at MIT, Andrew Ng at Stanford, Y. Abu-Mostafa at CalTech, E. Xing at CMU, P. Domingos at UWA, Alexander Gray at Georgia Tech, and all my mentors and people who made me realize Machine Learning is one of my passions.

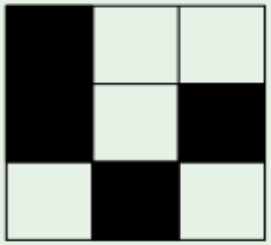
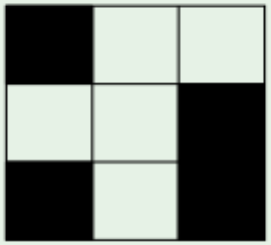
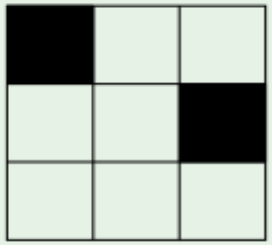
Outline

- Is learning feasible?
- Building the diagram of supervised learning
- On unknown/missing attributes
- A whirlwind tour on machine learning terms

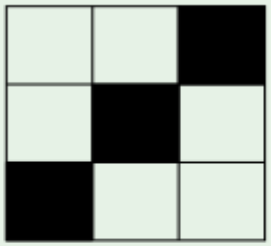
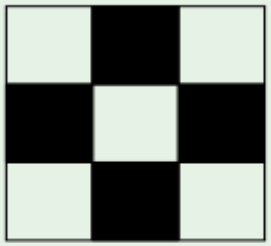
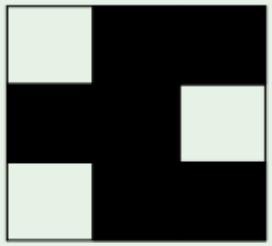
The supervised classification problem.



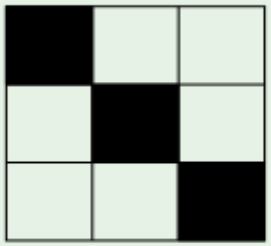
Puzzle

$f = -1$

$f = +1$



$f = ?$

Is learning feasible?

But if the true function $f(x)$ is completely unknown, is learning feasible?

Is learning feasible?

But if the true function $f(x)$ is completely unknown, is learning feasible?

Short answer: NO!
If $f(x)$ can take any value outside the sample, how should I know how to predict given a new data point?

Is learning feasible?

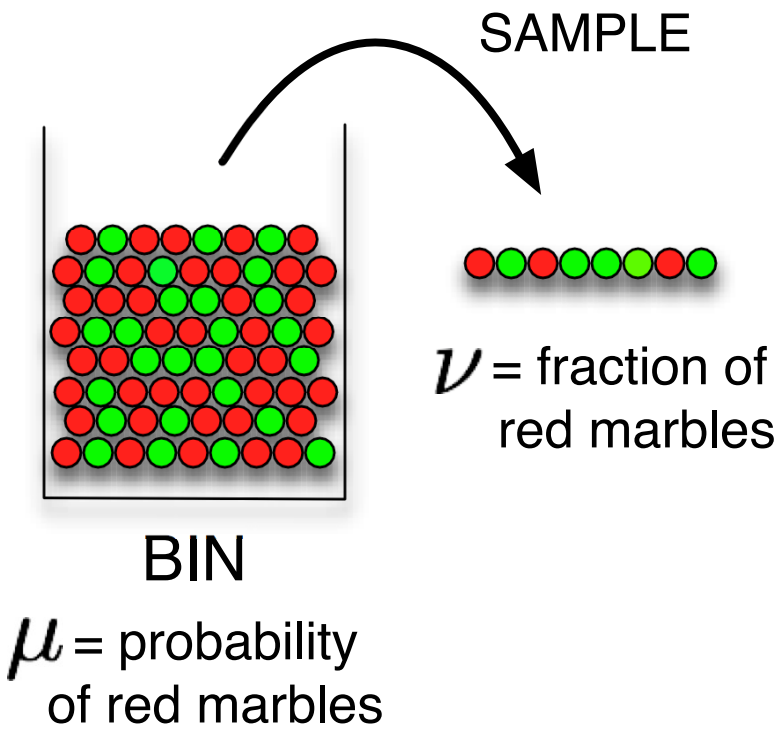
But if the true function $f(x)$ is completely unknown, is learning feasible?

Short answer: NO!
If $f(x)$ can take any value outside the sample, how should I know how to predict given a new data point?

Long answer: YES! (note that Yes is longer than No ;-))

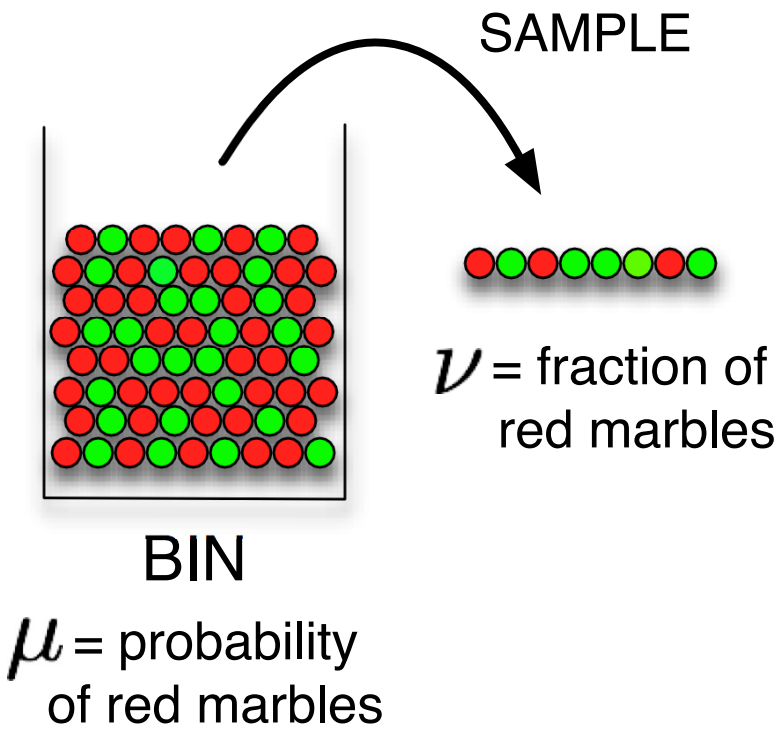
A related experiment.

- Consider a 'bin' with red and green marbles.
  P[picking a red marble] $= \mu$
  P[picking a green marble] $= 1 - \mu$



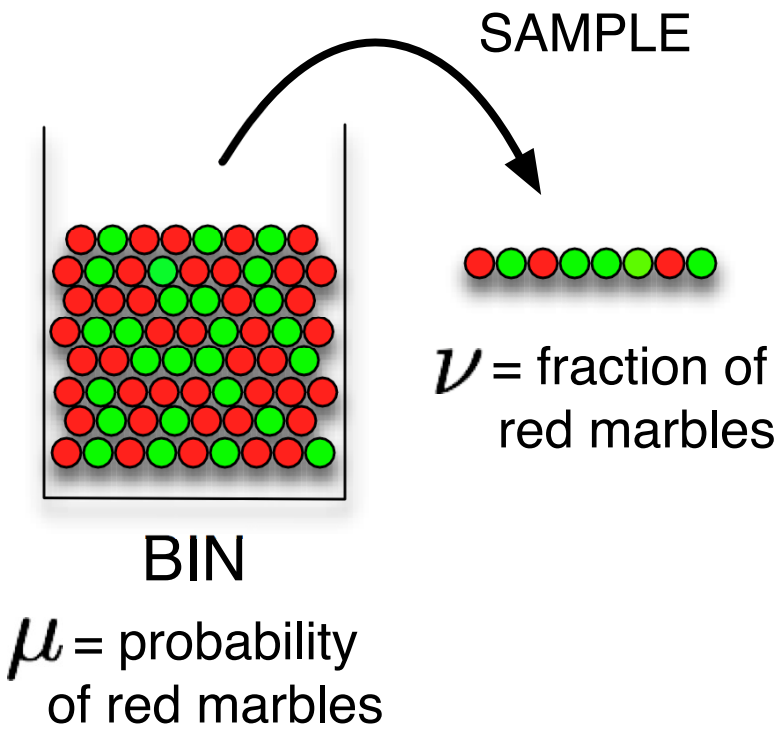
A related experiment.

- Consider a 'bin' with red and green marbles.
  P[picking a red marble] $= \mu$
  P[picking a green marble] $= 1 - \mu$
- The value of μ is unknown to us.



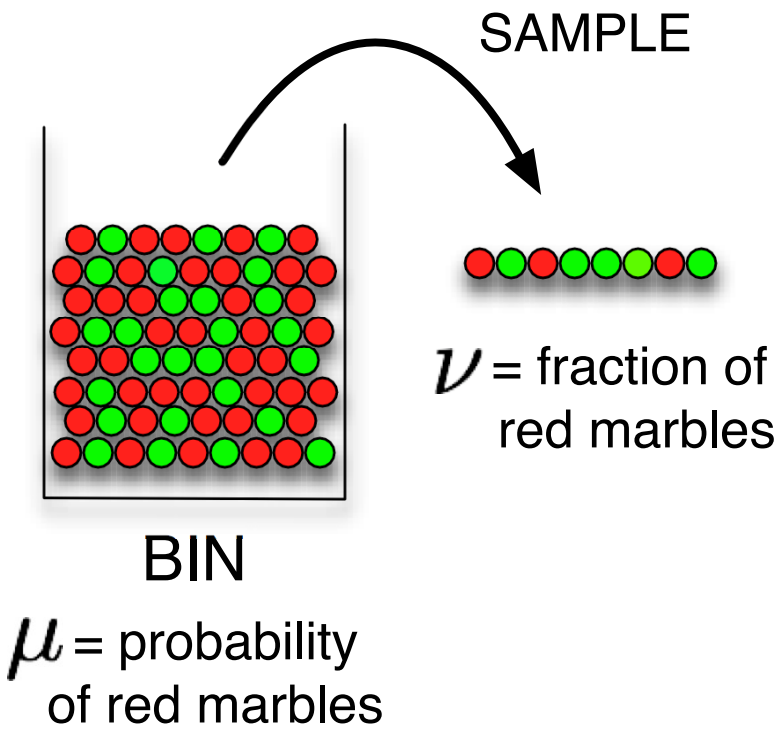
A related experiment.

- Consider a 'bin' with red and green marbles.
  P[picking a red marble] $= \mu$
  P[picking a green marble] $= 1 - \mu$
- The value of μ is unknown to us.
- We pick N marbles independently.



A related experiment.

- Consider a 'bin' with red and green marbles.
  P[picking a red marble] $= \mu$
  P[picking a green marble] $= 1 - \mu$
- The value of μ is unknown to us.
- We pick N marbles independently.
- The fraction of red marbles in sample $= \nu$



Does ν say anything about μ?

Short answer: NO!

Sample can be mostly green while bin is mostly red

Does ν say anything about μ?

Short answer: NO!
Sample can be mostly green while bin is mostly red

Long answer: YES!
Sample frequency ν is likely close to bin frequency μ

Does ν say anything about μ?

Short answer: NO!
Sample can be mostly green while bin is mostly red

Long answer: YES!
Sample frequency ν is likely close to bin frequency μ

Although any configuration is *possible*, not all of them are equally *probable*

Does ν say anything about μ?

In a big sample , ν is probably close to μ

Does ν say anything about μ?

In a big sample (large N), ν is probably close to μ (within ϵ).

Does ν say anything about μ?

In a big sample (large N), ν is probably close to μ (within ϵ).

Formally,

Does ν say anything about μ?

In a big sample (large N), ν is probably close to μ (within ϵ).

Formally,

$$\mathbb{P}[\text{bad event}] \leqslant$$

Does ν say anything about μ?

In a big sample (large N), ν is probably close to μ (within ϵ).

Formally,

$$\mathbb{P}[|\nu - \mu| > \epsilon] \leqslant$$

Does ν say anything about μ?

In a big sample (large N), ν is probably close to μ (within ϵ).

Formally,

$$\mathbb{P}[|\nu - \mu| > \epsilon] \leqslant e^{-} \quad ^{N} \leftarrow \text{GOOD NEWS!}$$

Does ν say anything about μ?

In a big sample (large N), ν is probably close to μ (within ϵ).

Formally,

$$\mathbb{P}[|\nu - \mu| > \epsilon] \leqslant e^{-\epsilon^2 N} \leftarrow \text{THE PRICE!}$$

Does ν say anything about μ?

In a big sample (large N), ν is probably close to μ (within ϵ).

Formally,

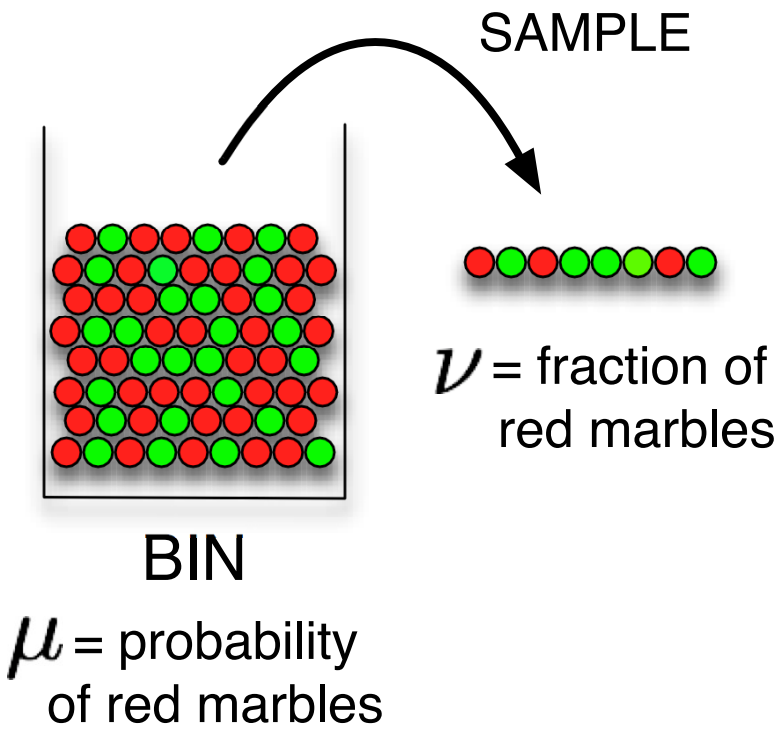
$$\mathbb{P}[|\nu - \mu| > \epsilon] \leqslant 2e^{-2\epsilon^2 N}$$

This is called Hoeffding's Inequality

In other words, the statement "$\mu = \nu$" is Probably Approximately Correct (P.A.C.)

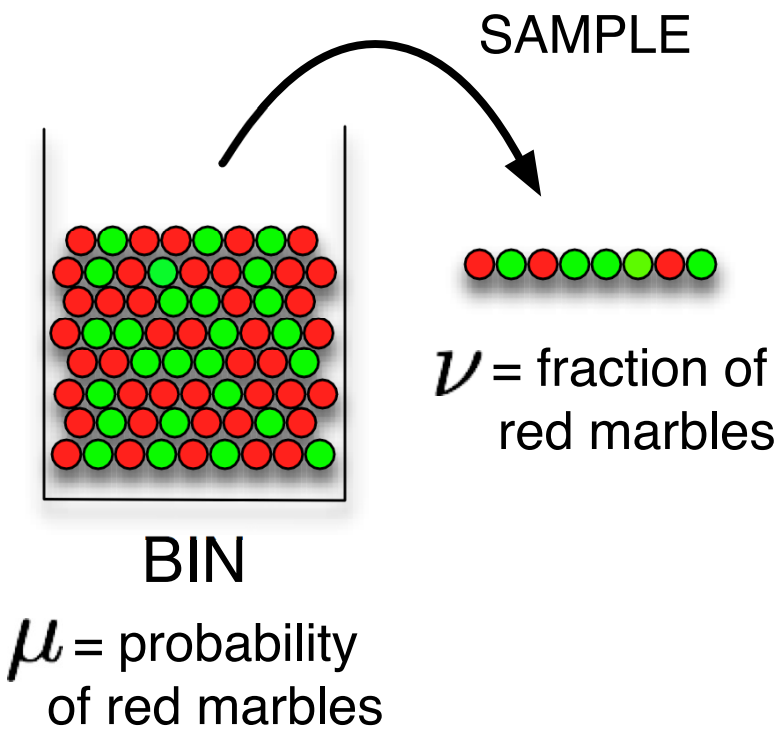
$$\mathbb{P}[|\nu - \mu| > \epsilon] \leqslant 2e^{-2\epsilon^2 N}$$

- Valid for all N and ϵ



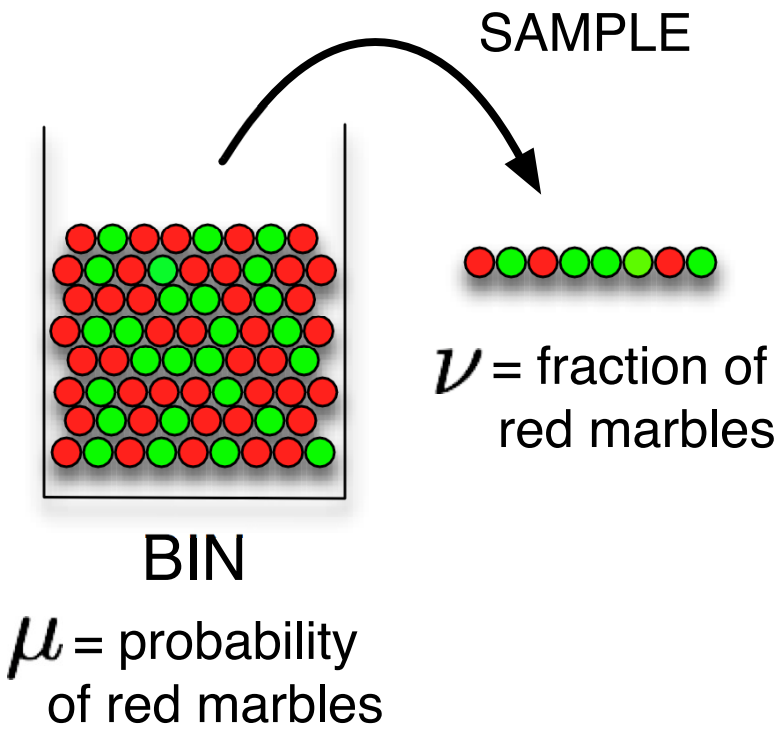
$$\mathbb{P}[|\nu - \mu| > \epsilon] \leqslant 2e^{-2\epsilon^2 N}$$

- Valid for all N and ϵ
- Bound does not depend on μ



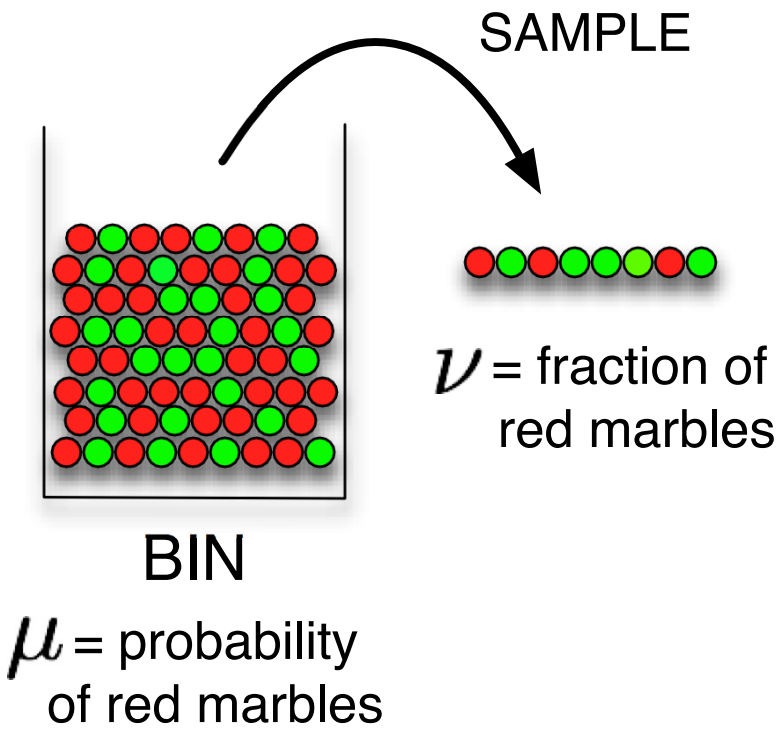
$$\mathbb{P}[|\nu - \mu| > \epsilon] \leqslant 2e^{-2\epsilon^2 N}$$

- Valid for all N and ϵ
- Bound does not depend on μ
- Tradeoff: N, ϵ, and the bound



$$\mathbb{P}[|\nu - \mu| > \epsilon] \leqslant 2e^{-2\epsilon^2 N}$$

- Valid for all N and ϵ
- Bound does not depend on μ
- Tradeoff: N, ϵ, and the bound
- $\nu \approx \mu \implies \mu \approx \nu$. Because it is symmetric we can say things about ν given μ, but also about μ given ν.



Connection to learning

Bin: The unknown is a number μ

Connection to learning

Bin: The unknown is a number μ

Learning: The unknown is a function $g : \mathcal{X} \to \mathcal{Y}$
Each marble • is a point $x \in \mathcal{X}$
• : Hypothesis got it right $h(\mathbf{x}) = g(\mathbf{x})$
• : Hypothesis got it wrong $h(\mathbf{x}) \neq g(\mathbf{x})$

Connection to learning

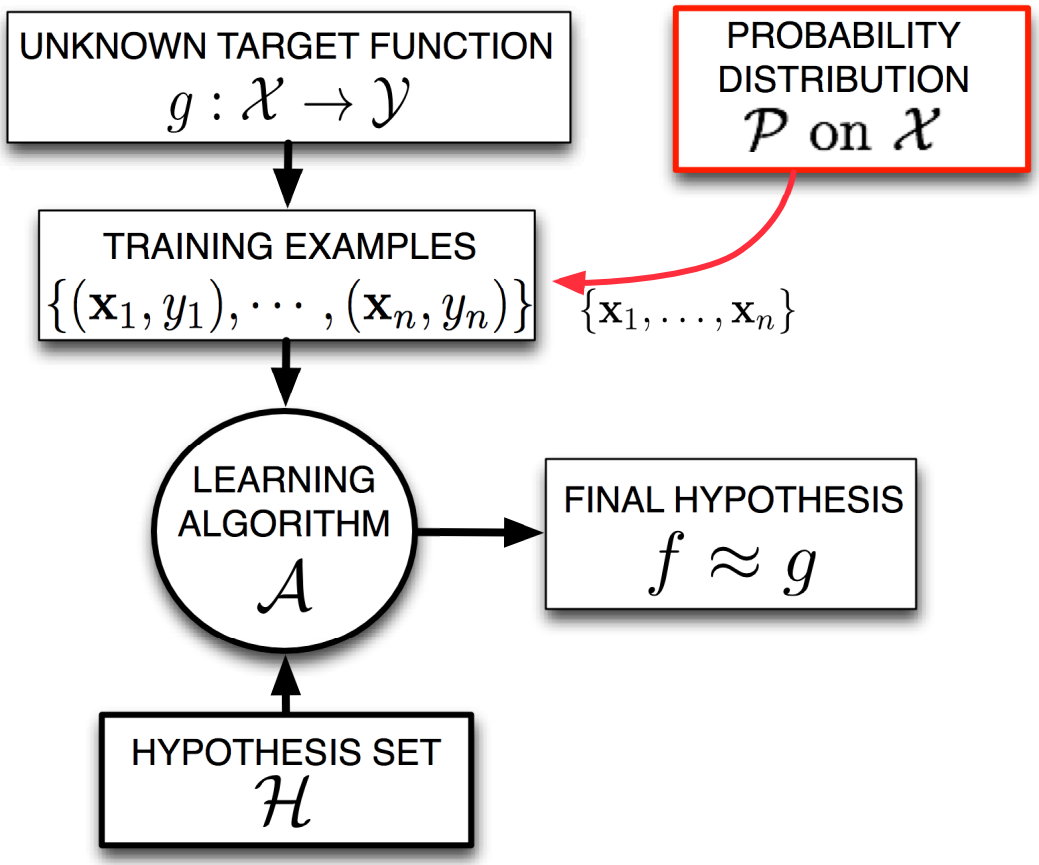
Bin: The unknown is a number μ

Learning: The unknown is a function $g : \mathcal{X} \to \mathcal{Y}$
Each marble • is a point $x \in \mathcal{X}$
• : Hypothesis got it right $h(\mathbf{x}) = g(\mathbf{x})$
• : Hypothesis got it wrong $h(\mathbf{x}) \neq g(\mathbf{x})$

Notation:
ν is 'in sample error' (frequency of the hypothesis getting it wrong) denoted by $E_{in}(h)$
μ is 'out of sample error' (expected error) denoted by $E_{out}(h)$

The supervised classification problem.

Introducing the probability over the data samples will grant feasibility to



the learning problem.

Are we done?

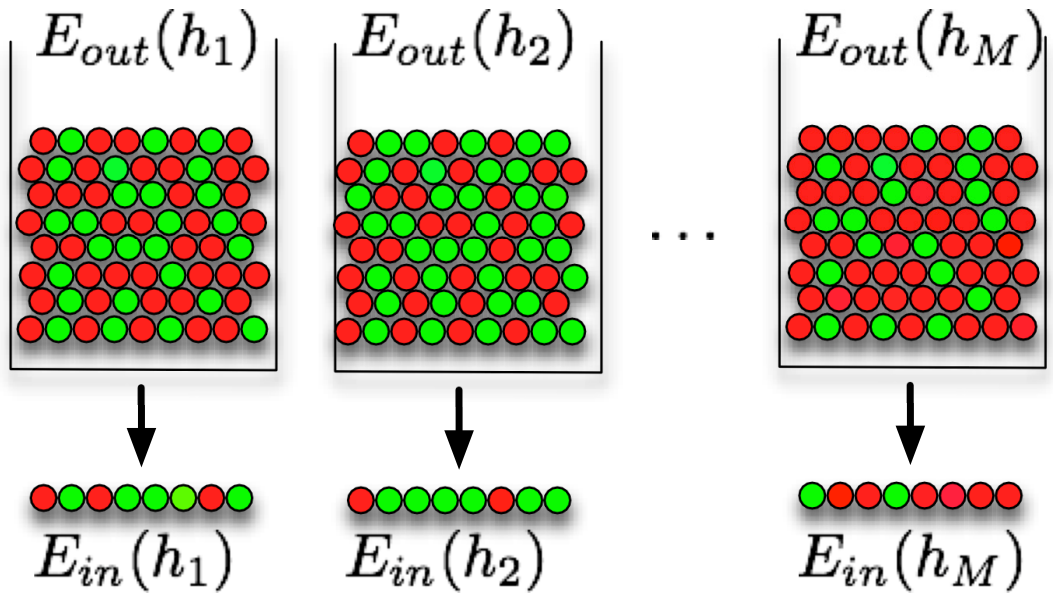
- ATTENTION: The bin analogy requires a particular hypothesis $h(\cdot)$.
- This guarantees that the in sample error will be close to the out of sample error.

Are we done?

- ATTENTION: The bin analogy requires a particular hypothesis $h(\cdot)$.
- This guarantees that the in sample error will be close to the out of sample error.
- This is not learning.

Are we done?

- ATTENTION: The bin analogy requires a particular hypothesis $h(\cdot)$.
- This guarantees that the in sample error will be close to the out of sample error.
- This is not learning.
- Learning corresponds to the process of assuming that the in sample error is an indicator of the out of sample error, select/chose/find the hypothesis with the smaller out of sample error.



The right bound

PROBLEM: Hoeffding doesn't apply to multiple "bins".

The right bound

PROBLEM: Hoeffding doesn't apply to multiple "bins".
What we really want is a bound that holds for all hypothesis in our set

$$\mathbb{P}[\max_{h \in \mathcal{H}} |E_{in}(h) - E_{out}(h)| > \epsilon] \leqslant \delta$$

The right bound

PROBLEM: Hoeffding doesn't apply to multiple "bins".
What we really want is a bound that holds for all hypothesis in our set

$$\mathbb{P}[\max_{h\in\mathcal{H}} |E_{in}(h) - E_{out}(h)| > \epsilon] \leqslant \delta$$

Using the union bound $\left(P(A_1 \cup A_2 \cdots \cup A_M) = P(A_1) + P(A_2) + \cdots + P(A_M)\right)$

$$\mathbb{P}[\max_{h\in\mathcal{H}} |E_{in}(h) - E_{out}(h)| > \epsilon] \leqslant \sum_{h\in\mathcal{H}} \mathbb{P}[|E_{in}(h) - E_{out}(h)| > \epsilon]$$

The right bound

PROBLEM: Hoeffding doesn't apply to multiple "bins".
What we really want is a bound that holds for all hypothesis in our set

$$\mathbb{P}[\max_{h\in\mathcal{H}} |E_{in}(h) - E_{out}(h)| > \epsilon] \leqslant \delta$$

Using the union bound $\left(P(A_1 \cup A_2 \cdots \cup A_M) = P(A_1) + P(A_2) + \cdots + P(A_M)\right)$

$$\mathbb{P}[\max_{h\in\mathcal{H}} |E_{in}(h) - E_{out}(h)| > \epsilon] \leqslant \sum_{h\in\mathcal{H}} \mathbb{P}[|E_{in}(h) - E_{out}(h)| > \epsilon]$$

$$\mathbb{P}[\max_{h\in\mathcal{H}} |E_{in}(h) - E_{out}(h)| > \epsilon] \leqslant 2|\mathcal{H}|e^{-2\epsilon^2 N}$$

KEY IDEA: The bound gets worse with the "complexity" of the hypotheses space.

The right bound

$$2|\mathcal{H}|e^{-2\epsilon^2 N} = \delta \implies \epsilon = \sqrt{\frac{\log|\mathcal{H}| + \log(2/\delta)}{2n}}$$

And, then we can say that with probability $1-\delta$ over the training set[1],

$$E_{out}(h) \leqslant E_{in}(h) + \sqrt{\frac{\log|\mathcal{H}| + \log(2/\delta)}{2n}}, \quad \forall h \in \mathcal{H}$$

[1] Observe that $P(a - b > \epsilon)$ holds with probability δ, then with probability $1-\delta$ we have the complementary $a - b \leqslant \epsilon$

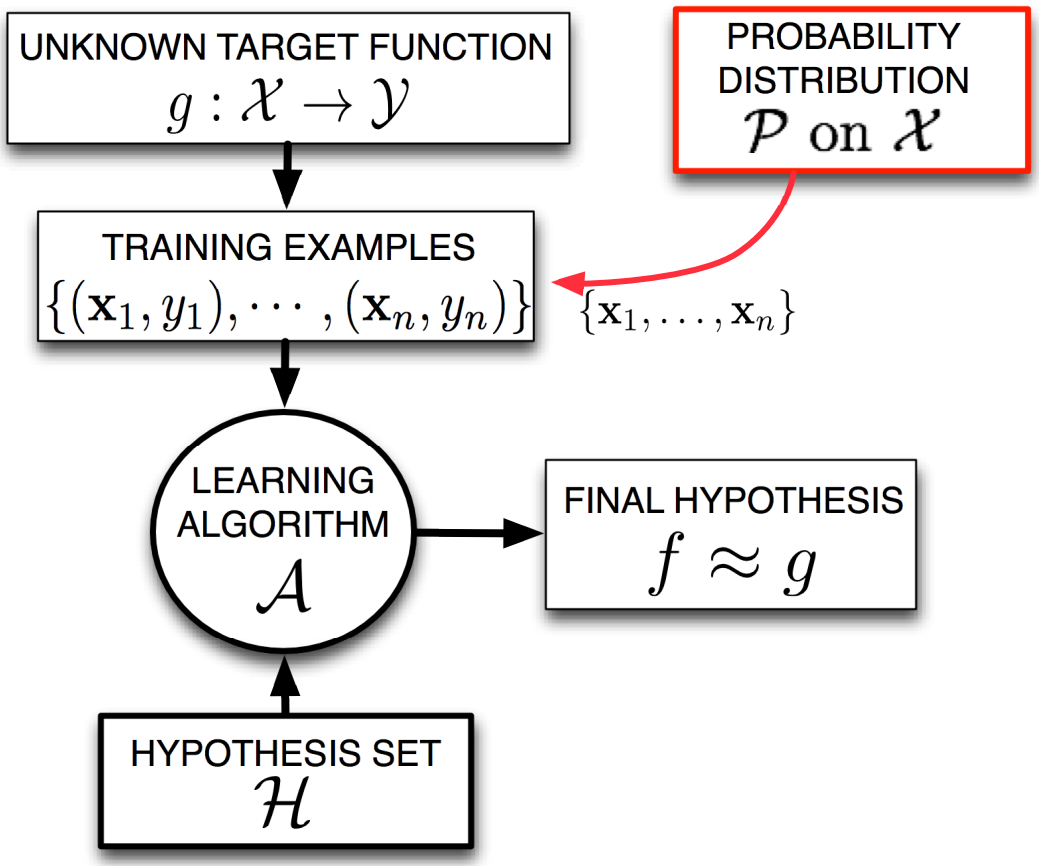
Using the bound

This can be used for model selection, we plug-in the in sample error achieved by different models and their corresponding complexities and then select that with the minimum bound.
Suppose we set $\delta = 0.05$ and would like any classifier that achieves zero training error to have at most 10% generalization error. Let's solve for the number of training examples we would need for such a guarantee within model $\mathcal{H}$. We want

$$E_{out}(h) \leqslant 0 + \sqrt{\frac{\log|\mathcal{H}| + \log(2/0.05)}{2n}} \leqslant 0.10$$

$$n = \frac{\log|\mathcal{H}| + \log(2/0.05)}{2(0.10)^2}$$

The supervised classification problem.



Error measures

What does "$h \approx g$" mean ?

Error measure: $E(h, g)$

Almost always *point wise definition* : $e(h(\mathbf{x}), g(\mathbf{x}))$

Examples:

$$\text{Squared error: } e(h(\mathbf{x}), g(\mathbf{x})) = (h(\mathbf{x}) - g(\mathbf{x}))^2$$

Error measures

What does "$h \approx g$" mean ?

Error measure: $E(h, g)$

Almost always *point wise definition* : $e(h(\mathbf{x}), g(\mathbf{x}))$

Examples:

$$\text{Squared error: } e(h(\mathbf{x}), g(\mathbf{x})) = (h(\mathbf{x}) - g(\mathbf{x}))^2$$
$$\text{Binary error: } e(h(\mathbf{x}), g(\mathbf{x})) = [\![h(\mathbf{x}) \neq g(\mathbf{x})]\!]$$

From point wise errors to overall

Overall error $E(h, g)$ = average of point wise errors $e(h(x), g(x))$.

From point wise errors to overall

Overall error $E(h, g)$ = average of point wise errors $e(h(x), g(x))$.

In sample error:

$$E_{in}(h) = \frac{1}{N} \sum_{n=1}^{N} e(h(\mathbf{x_n}), g(\mathbf{x_n}))$$

From point wise errors to overall

Overall error $E(h, g)$ = average of point wise errors $e(h(x), g(x))$.

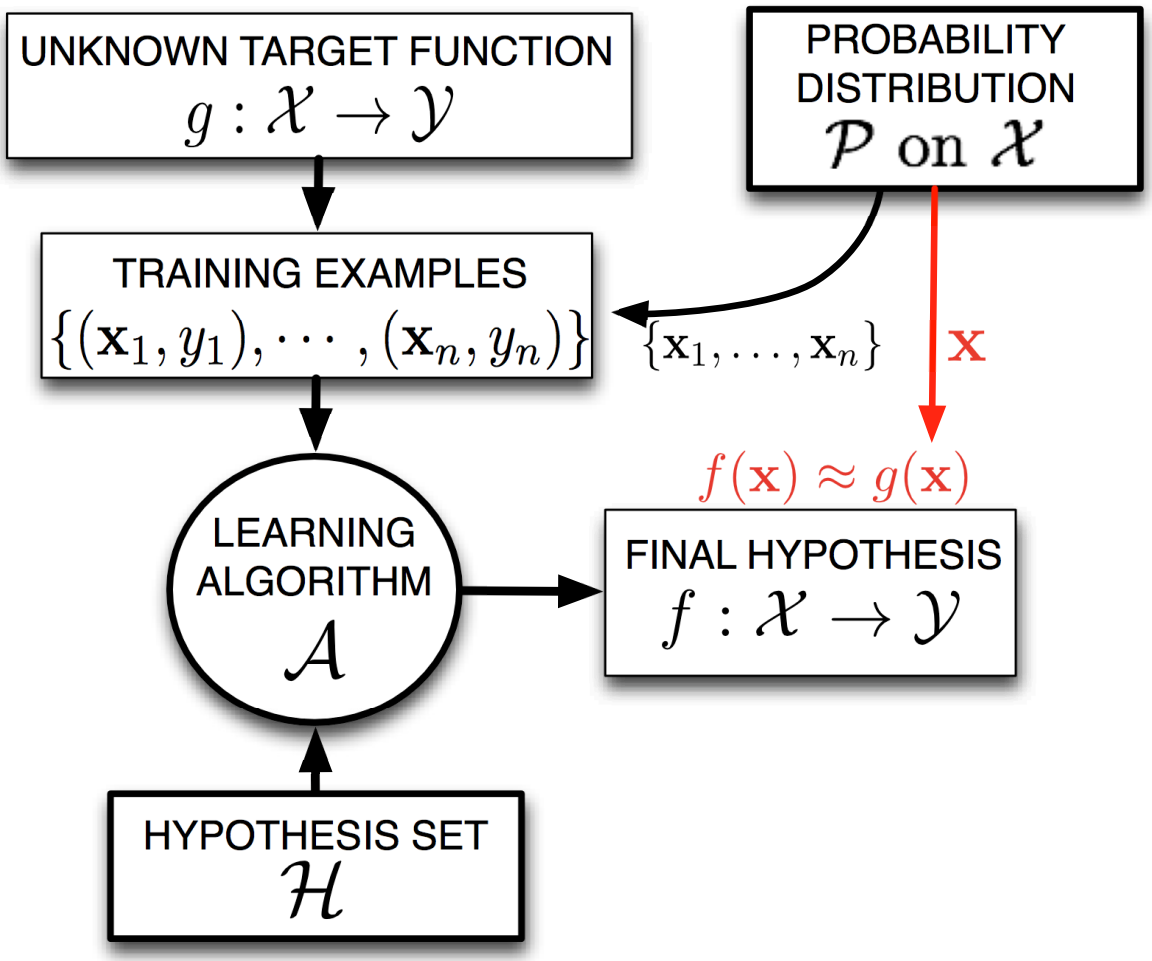
In sample error:

$$E_{in}(h) = \frac{1}{N} \sum_{n=1}^{N} e(h(\mathbf{x_n}), g(\mathbf{x_n}))$$

Out of sample error:

$$E_{out}(h) = \mathbb{E}_{\mathbf{x}}\Big[e(h(\mathbf{x}), g(\mathbf{x}))\Big]$$

The supervised classification problem.



How to chose the error measure.

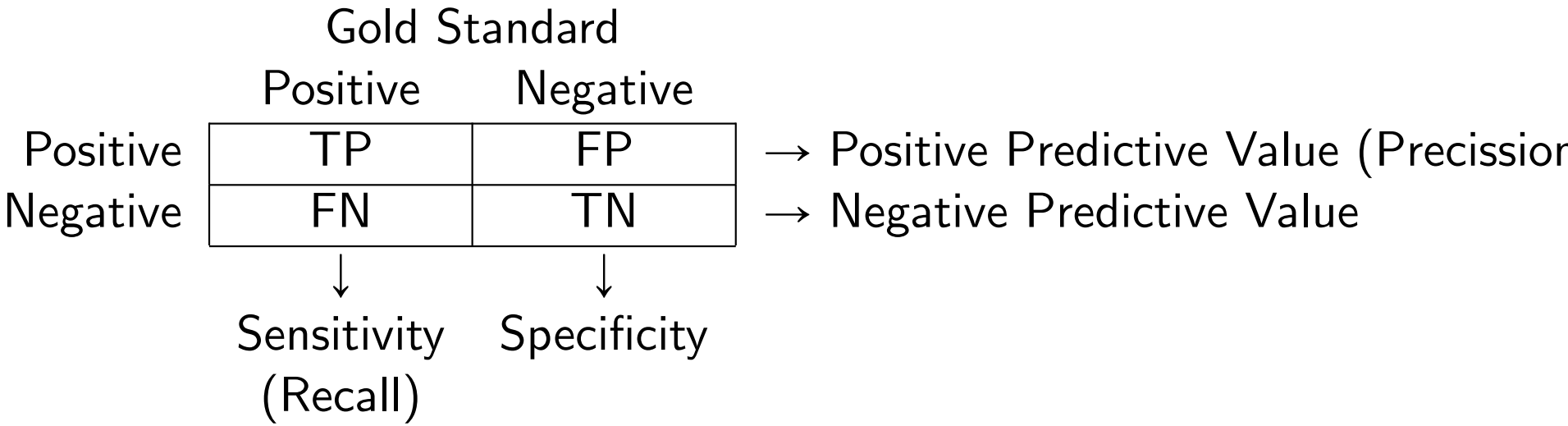
The error measure should be specified by the user/client/stakeholder. Which is the business value/loss of making a mistake? Has the same loss value a FP than a FN? Example: Consider a fingerprint verification system

in a supermarket vs CSI scenarios.

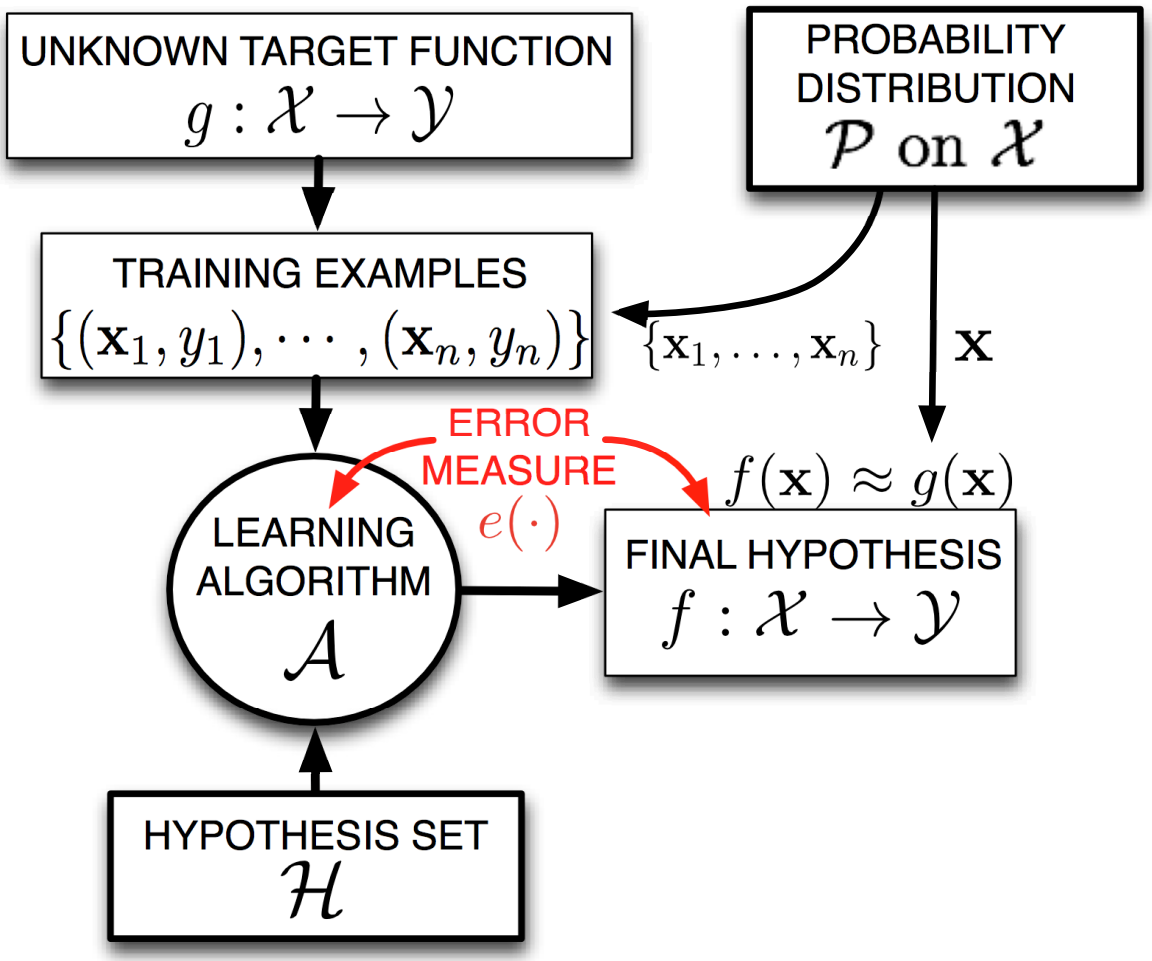
Not always possible. Alternatives:

- **Plausible measures:** squared error $\Longrightarrow$ We assume Gaussian noise corruption
- **Friendly measures:** closed-form solution, convex measures $\Longrightarrow$ We can computationally easily solve the problem.

Error concepts.



The learning diagram - with error measure



The last consideration.

Technically speaking, the *'target function'* is not always a *function*.

The feature vector may describe two items exactly with the same attributes but the true label can be different. Example: Suppose that we are classifying different kind of terrains in an image. We use a very simple feature vector, the RGB components of the image. And we ask ourselves the question "Is a certain pixel sea or sky?".

Two identical observations $\rightarrow$ two different behaviors. This is not a function.

Target 'distribution'

Instead of $y = g(\mathbf{x})$, we have to refer to a target *distribution*

$$P(y|\mathbf{x})$$

Observe now that the pair $(\mathbf{x}, y)$,

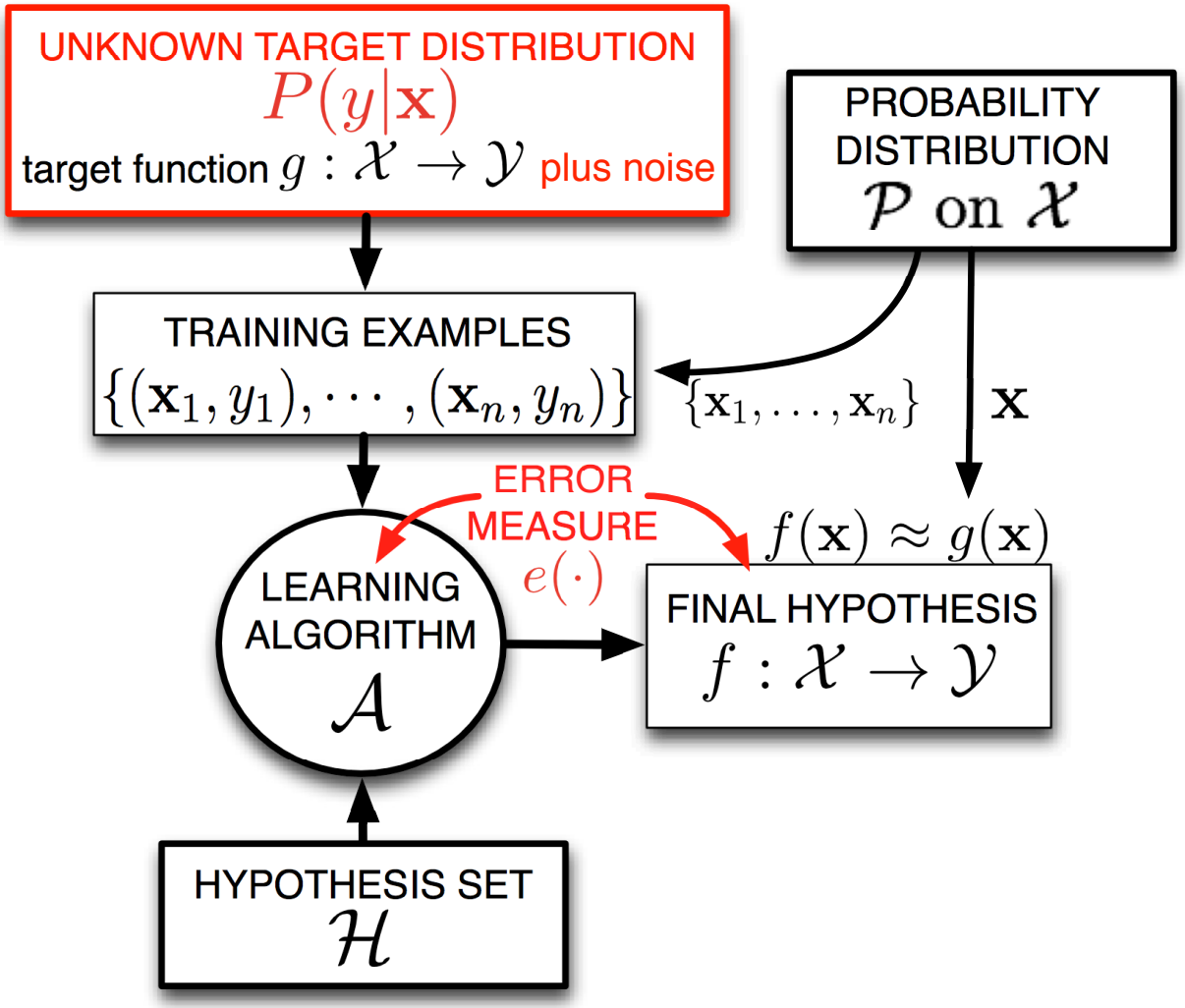
- $\mathbf{x}$ is still generated by $P(\mathbf{x})$, which ensures we can use Hoeffding's
- Instead of y being deterministic with $\mathbf{x}$, it is generated by $P(y|\mathbf{x})$

Thus, $(\mathbf{x}, y)$ are generated by the joint distribution

$$P(\mathbf{x})P(y|\mathbf{x})$$

We can relate this to the deterministic scenario by considering that the target is noisy. In this scenario $g(\mathbf{x}) = \mathbb{E}(y|\mathbf{x})$ and the noise is $y - g(\mathbf{x})$ (noise account for the difference of the real y with respect to the expected one given the data).

The FINAL learning diagram - including noisy target



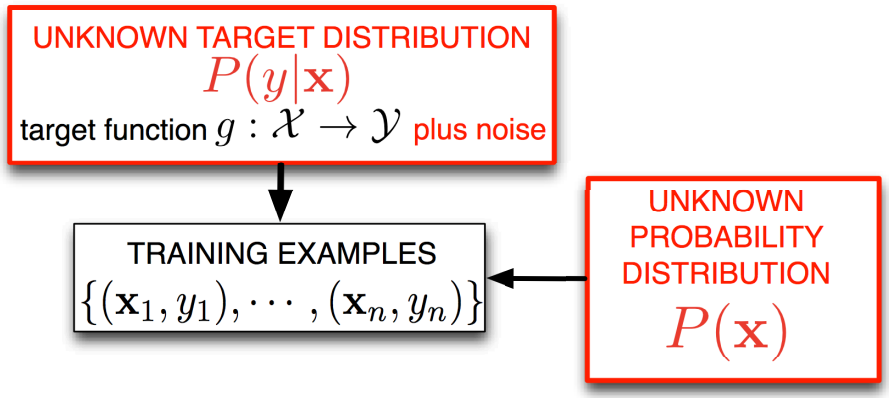
Distinction between $P(y|\mathbf{x})$ and $P(\mathbf{x})$

Both convey probabilistic aspects of $\mathbf{x}$ and y

The target distribution $P(y|\mathbf{x})$ is what we are trying to learn

The input distribution $P(\mathbf{x})$ quantifies the relative importance of $\mathbf{x}$

Merging $P(\mathbf{x})P(y|\mathbf{x})$ as $P(\mathbf{x}, y)$ mixes the two concepts. This is not the target of supervised learning.



Summarizing what we know so far.

Learning is feasible. Thanks to Hoeffdings and variants we know that it is likely that

$$E_{out}(f) \approx E_{in}(f)$$

Is this learning?

Summarizing what we know so far.

Learning is feasible. Thanks to Hoeffdings and variants we know that it is likely that

$$E_{out}(f) \approx E_{in}(f)$$

Is this learning?

We need $f \approx g$, which means

$$E_{out}(f) \approx 0$$

The two questions of learning

$E_{out}(f) \approx 0$ is achieved through:

$$E_{out}(f) \approx E_{in}(f) \text{ and } E_{in}(f) \approx 0$$

Learning is split into 2 questions:

1. Can we make sure that $E_{out}(f)$ is close enough to $E_{in}(f)$? This is almost solved, except for one technical detail, using the variants of the Hoeffding's inequality.
2. Can we make $E_{in}(f)$ small enough? This is the role of the model and the learning algorithm.

The technical "detail".

Recall the Hoeffding's inequality adapted to the learning problem in which we have a finite set of hypothesis,

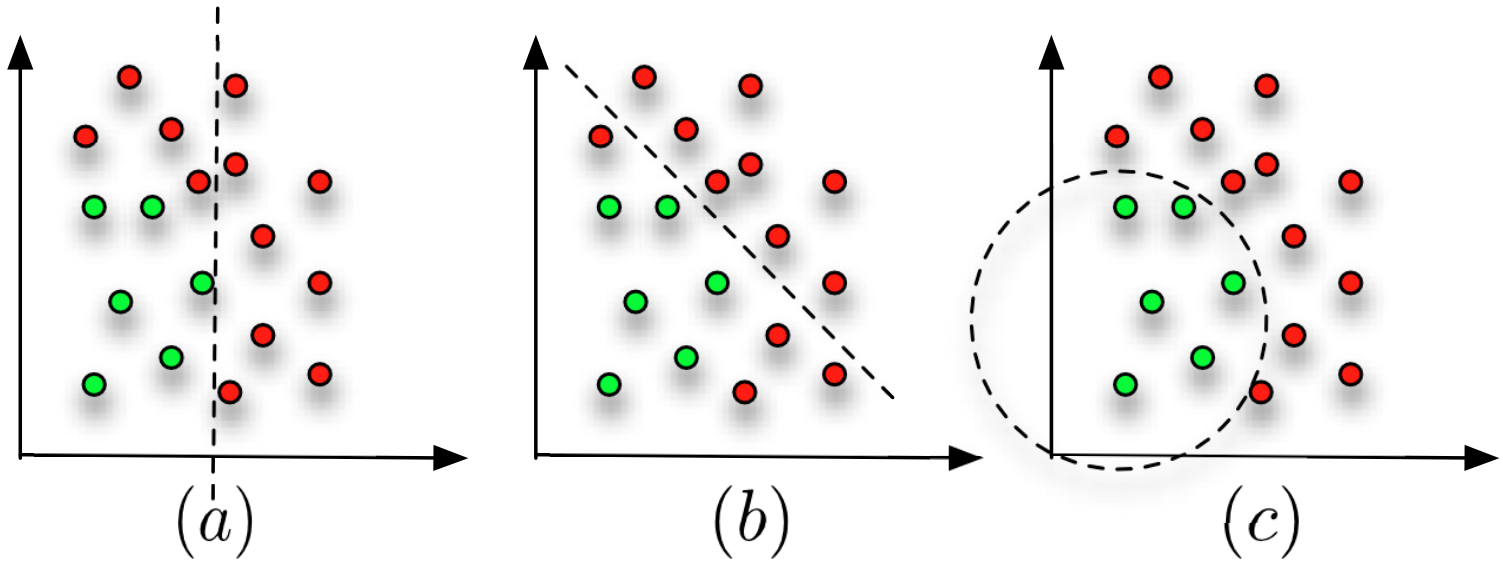
$$\mathbb{P}[\max_{h \in \mathcal{H}} |E_{in}(h) - E_{out}(h)| > \epsilon] \leqslant 2|\mathcal{H}|e^{-2\epsilon^2 N}$$

Now consider a linear model as our base hypothesis. Which is the cardinality of the hypotheses space? ∞

It is clear that the notion of complexity has to be redefined.

The technical "detail"

Let us put our intuition into work.



Which of the classifiers is more complex? Why? Has it anything to do with the discrimination power? Why do you consider one is more powerful than the rest?

Vapnik-Chervonenkis dimension

The model with more capability of discriminating any set of points seems to be more complex.

We say that a hypotheses set $\mathcal{H}$ *shatters* a set of k points

if at least one of the individual hypothesis in the set is able to generate all possible labels over some configuration of k number of points.

Vapnik-Chervonenkis dimension

The VC-dimension is defined as the maximum number of points that a classifier can shatter.

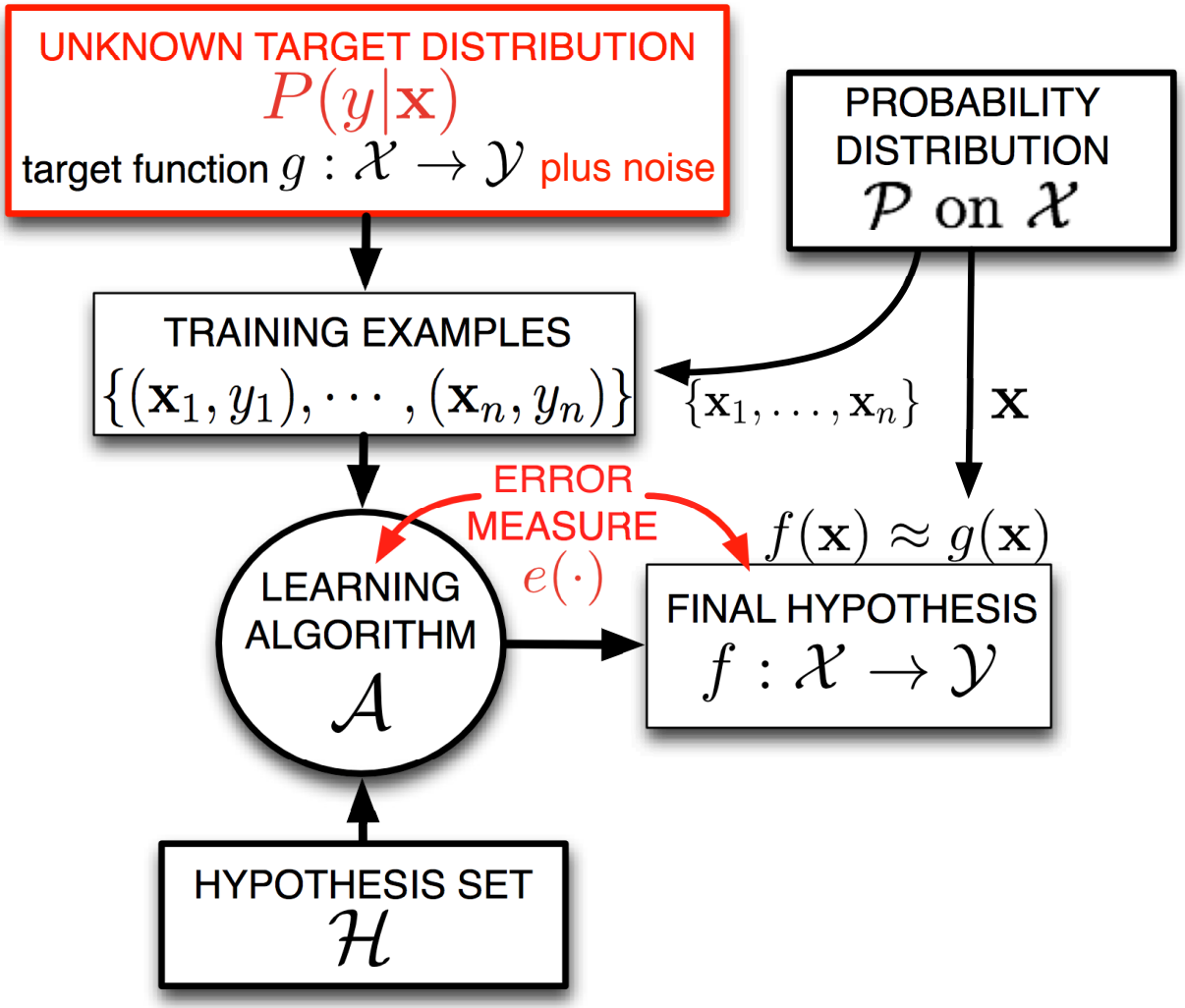
Which is the VC-dimension of a perceptron in 2D? And in 3D?

The final result

The VC-dimension generalizes the previous results and takes the place of the logarithm of the number of classifiers in our set.

$$E_{out}(h) \leqslant E_{in}(h) + \sqrt{\frac{d_{VC}(\log(2n/d_{VC}) + 1) + \log(2/\delta)}{2n}}$$

The FINAL learning diagram - including noisy target



Wrapping this lecture.

- Which hypothesis space $\mathcal{H}$ to choose?
- How to measure the degree of fit?
- How to trade-off degree of fit vs. complexity?
  - Occam's razor
- How do we find a good h?
- How do we know if a good h will predict well?

On unknown/missing attribute values

What if some examples are missing values? Use training example anyway,

- Assign the most common value of the attribute.
- Assign the most common value of the attribute among the examples of the same class.
- Split the example in as many examples as possible values of the missing attribute and assign a probability to each one, e.g. assign the fraction of examples that has that particular value of the attribute.

Basic concepts of machine learning

- **How do I learn a simple Gaussian?** Probability, random variables, distributions; estimation, convergence and asymptotics, confidence interval.
- **How do I learn a mixture of Gaussians (MoG)?** Likelihood, Expectation-Maximization (EM); generalization, model selection, cross-validation; k-means, hidden markov models (HMM)
- **How do I learn any density?** Parametric vs. non-Parametric estimation, Sobolev and other functional spaces; $\ell - 2$ error; Kernel density estimation (KDE), optimal kernel, KDE theory

Basic concepts of machine learning

- **How do I predict a continuous variable (regression)?** Linear regression, regularization, ridge regression, and LASSO; local linear regression; conditional density estimation.
- **How do I predict a discrete variable (classification)?** Bayes classifier, naive Bayes, generative vs. discriminative; perceptron, weight decay, linear support vector machine; nearest neighbor classifier and theory

Basic concepts of machine learning

General theory and model frameworks of Machine Learning

- **Which loss function should I use?** Maximum likelihood estimation theory; $\ell - 2$ estimation; Bayessian estimation; minimax and decision theory, Bayesianism vs frequentism
- **Which model should I use?** AIC and BIC; Vapnik-Chervonenskis theory; cross-validation theory; bootstrapping; Probably Approximately Correct (PAC) theory; Hoeffding-derived bounds.

Basic concepts of machine learning

General theory and model frameworks of Machine Learning

- **How can I learn fancier (combined) models?** Ensemble learning theory; boosting; bagging; stacking.
- **How can I learn fancier (nonlinear) models?** Generalized linear models, logistic regression; Kolmogorov theorem, generalized additive models; kernelization, reproducing kernel Hilbert spaces, non-linear SVM, Gaussian process regression
- **How can I learn fancier (compositional) models?** Recursive models, decision trees, hierarchical clustering; neural networks, back propagation, deep belief networks; graphical models, mixtures of HMMs, conditional random fields, max-margin Markov networks; log-linear models; grammars

Basic concepts of machine learning

Further common machine learning problems and solutions

- **How do I reduce or relate features?** Feature selection vs dimensionality reduction, wrapper methods for feature selection; causality vs correlation, partial correlation, Bayes net structure learning
- **How do I create new features?** principal component analysis (PCA), independent component analysis (ICA), multidimensional scaling, manifold learning, supervised dimensionality reduction, metric learning
- **How do I reduce or relate the data?** Clustering, bi-clustering, constrained clustering; association rules and market basket analysis; ranking/ordinal regression; link analysis; relational data

Basic concepts of machine learning

Further common machine learning problems and solutions

- **How do I treat time series?** ARMA; Kalman filter and stat-space models, particle filter; functional data analysis; change-point detection; cross-validation for time series
- **How do I treat non-ideal data?** covariate shift; class imbalance; missing data, irregularly sampled data, measurement errors; anomaly detection, robustness

Basic concepts of machine learning

General computational frameworks for machine learning

- **How do I optimize the parameters?** Unconstrained vs constrained/Convex optimization, derivative-free methods, first- and second-order methods, backfitting; natural gradient; bound optimization and EM
- **How do I optimize linear functions?** computational linear algebra, matrix inversion for regression, singular value decomposition (SVD) for dimensionality reduction
- **How do I optimize with constraints?** Convexity, Lagrange multipliers, Karush-Kuhn-Tucker conditions, interior point methods, SMO algorithm for SVM

Basic concepts of machine learning

General computational frameworks for machine learning

- **How do I evaluate deeply-nested sums?** Exact graphical model inference, variational bounds on sums, approximate graphical model inference, expectation propagation
- **How do I evaluate large sums and searches?** Generalized N-body problems (GNP), hierarchical data structures, nearest neighbor search, fast multiple method; Monte Carlo integration, Markov Chain Monte Carlo, Monte Carlo SVD
- **How do I treat even larger problems?** Parallel/distributed EM, parallel/distributed GNP; stochastic subgradient methods, online learning

Basic concepts of machine learning

Real-world application of machine learning

- **How do I apply all this in the real world?** Overview of the parts of the ML, choosing between the methods to use for each task, prior knowledge and assumptions; exploratory data analysis and information visualization; evaluation and interpretation, using confidence intervals and hypothesis test, ROC curves; where the research problems in ML are

Your turn

Any special request or advanced topic?