Computational Learning Theory

- Under what condition is successful learning possible and impossible.
- Under what condition is a particular learning algorithm assured of learning successfully

(OR)

- What kind of task is required for learning.
- What kind of data is required for learnability

CleverInsight

PAC Learning

Probably approximately correct learning

- PAC learning, or Probably Approximately Correct learning is a framework for mathematical analysis of machine learning
- Goal of PAC: With high probability (“**Probably**”), the selected hypothesis will have low error ("**Approximately Correct**")
- Assume there is no error (noise) on data

PAC receives some samples and it must select some hypothesis from Hypothesis space.

The main goals of computational theory :

1. Sample complexity: How many training examples are needed for a learner to converge (with high probability) to a successful hypothesis ?

2. Computational complexity: How much computational effort is needed for a learner to converge (with high probability) to a successful hypothesis ?

3. Mistake bound: How many training examples will the learner misclassify before converging to successful hypothesis ?

Generalized Laws Constraints:

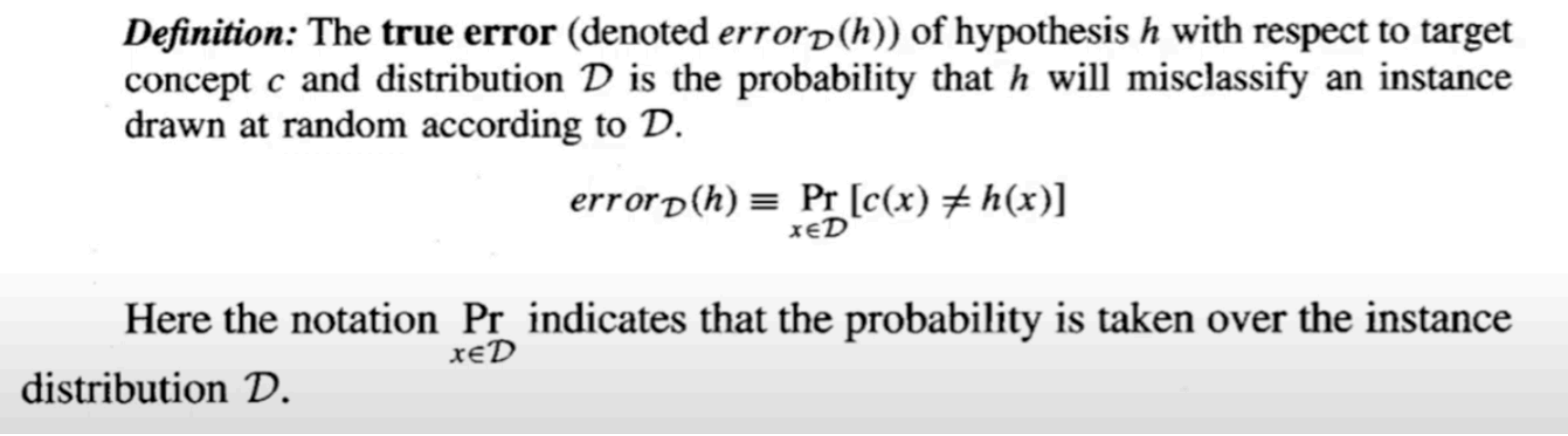
1. Probability of successful learning
2. Number of training examples
3. Complexity of Hypothesis space
4. Accuracy to which target concept is approximated
5. Manner in which training examples presented.

CleverInsight

Error of Hypothesis

True error of Hypothesis:

The true error of h is just the error rate we expect when applying h to future instances drawn according to the probability distribution.

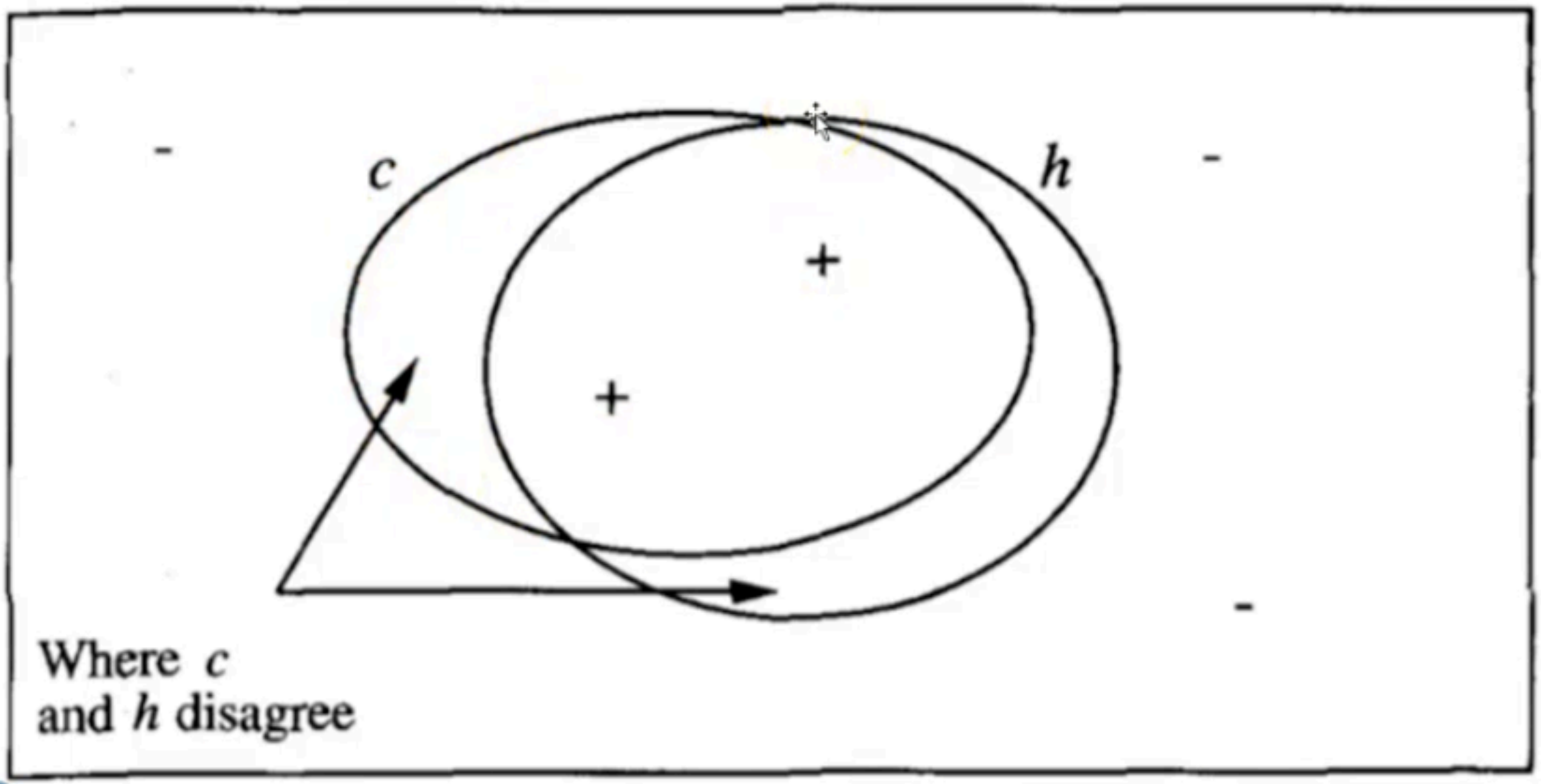


Definition: The **true error** (denoted $error_{\mathcal{D}}(h)$) of hypothesis h with respect to target concept c and distribution $\mathcal{D}$ is the probability that h will misclassify an instance drawn at random according to $\mathcal{D}$.

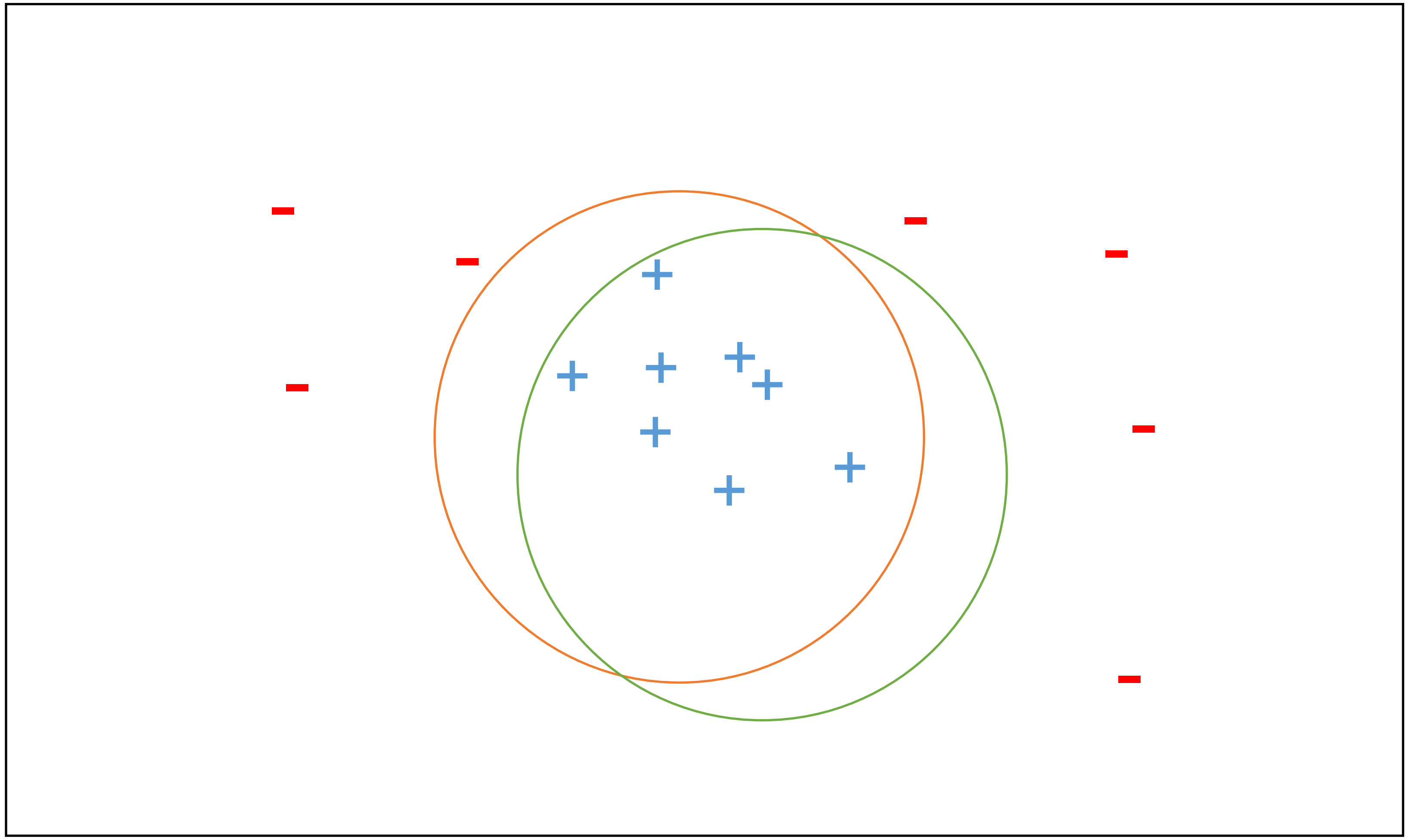
$$error_{\mathcal{D}}(h) \equiv \Pr_{x \in \mathcal{D}}[c(x) \neq h(x)]$$

Here the notation $\Pr_{x \in \mathcal{D}}$ indicates that the probability is taken over the instance distribution $\mathcal{D}$.

Instance space X



CleverInsight



PAC Learning wants the error region of C and H to be minimal.

Notations:

Given:

- Set of instances X
- Set of hypothesis H
- Set of possible target concepts C
- Training instances generated by a fixed, unknown probability distribution ***D*** over X.

Learner observes sequences **D** of training examples {x , c(x)}, for some target concept c $\in$ C.

- Instances x are drawn from distribution **D**.

CleverInsight

Concept Class:

Training examples.

$$\{(x_i, y_i)\}_{i=0}^{m}$$

Concept Class

$$\{(x_i, c(x_i))\}_{i=0}^{m}$$

$$y = c(x)$$

$$\hat{y} = h(x)$$

$c \in C$

$h \in H$

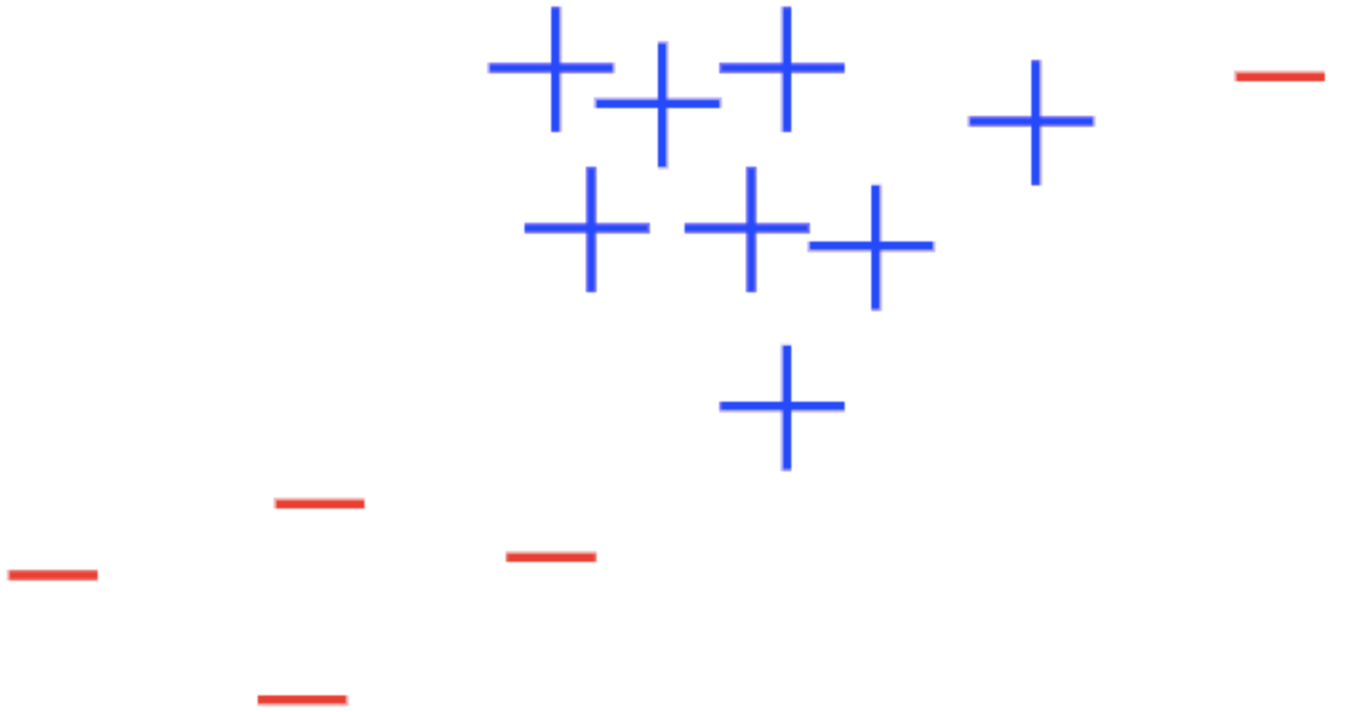
PAC learning: finite hypothesis space

- As we see before, training error underestimates the true error
- In PAC learning, we seek theory to relate:
  - The number of training samples: m
  - Te gap between training and true errors
    - $\text{error}_D(h) \leq \text{error}_{train}(h) + \varepsilon$
  - Complexity of the hypothesis space: $|H|$
  - Confidence of the relation: at least $(1 - \delta)$

CleverInsight

Example (PAC)

- Concept: Average body-size person
- Inputs: for each person:
  - height
  - weight
- Sample: labeled examples of persons
  - label + : average body-size
  - label - : not average body-size
- Two dimensional inputs



Example (PAC)

- Assumption: target concept is a rectangle.
- Goal:
  - Find a rectangle that “approximate” the target.
- Formally:
  - With high probability
  - output a rectangle such that
  - its error is low.

Example (Modeling)

- Assume:
  - Fixed distribution over persons.
- Goal:
  - Low error with respect to THIS distribution!!!
- How does the distribution look like?
  - Highly complex.
  - Each parameter is not uniform.
  - Highly correlated.

PAC approach

- Assume that the distribution is fixed.
- Samples are drawn are i.i.d.
  - independent
  - identical
- Concentrate on the decision rule rather than distribution.

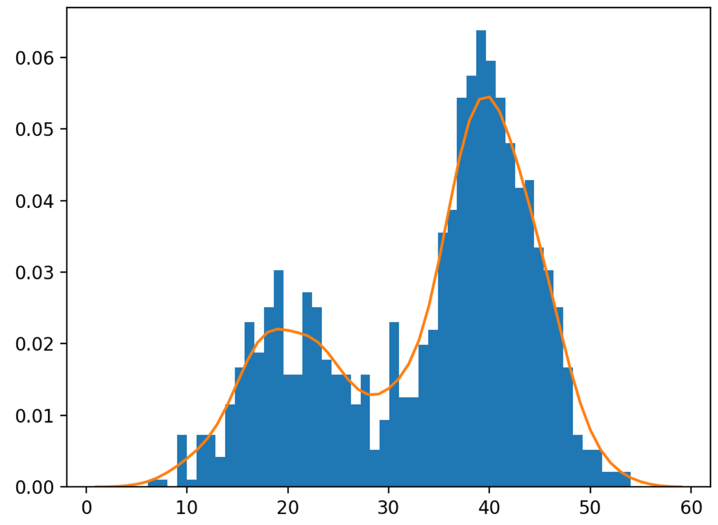
Problem of Probability Density Estimation

Density estimation is the problem of estimating the probability distribution for a sample of observations from a problem domain.

A common modeling problem involves how to estimate a joint probability distribution for a dataset. For example, given a sample of observation (*X*) from a domain (*x1*, *x2*, *x3*, …, *xn*), where each observation is drawn independently from the domain with the same probability distribution.

Density estimation involves selecting a probability distribution function and the parameters of that distribution that best explain the joint probability distribution of the observed data (*X*).

- How do you choose the probability distribution function?
- How do you choose the parameters for the probability distribution function?



- Sample of (X) drawn from the population is small and has noise.
- Any evaluation of an estimated probability density function and its parameter will has errors.

Most important techniques to solve it:

1. Maximum a Posteriori (MAP), a Bayesian method.
2. Maximum Likelihood Estimation (MLE), a Frequentist method.

The main difference is that MLE assumes that all solutions are equally likely beforehand, whereas MAP allows prior information about the form of the solution to be harnessed.

Maximum Likelihood Estimation (MLE)

Maximum Likelihood Estimation involves treating the problem as an optimization or search problem, where we seek a set of parameters that results in the best fit for the joint probability of the data sample (*X*).

Video Explanation: https://youtu.be/XepXtl9YKwc?t=34

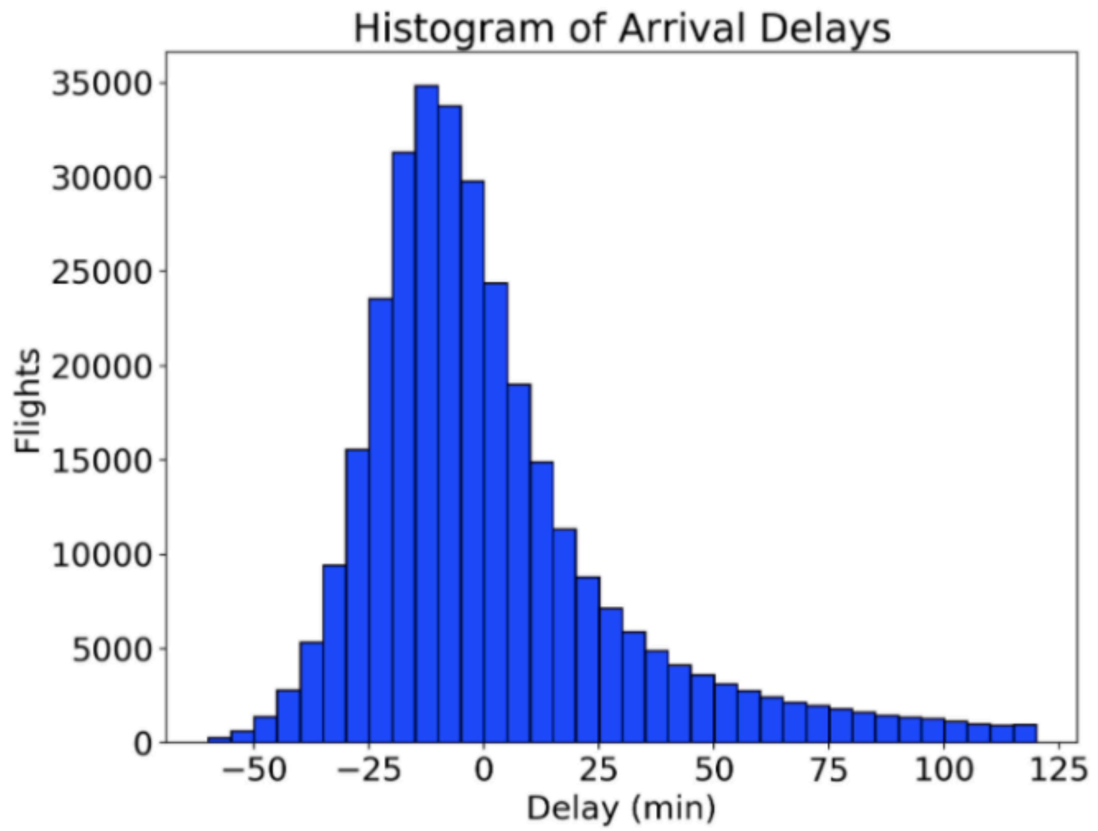
Maximum Likelihood Estimation (MLE), a Frequentist method.

This problem of density estimation is directly related to applied machine learning.
We can frame the problem of fitting a machine learning model as the problem of probability density estimation.

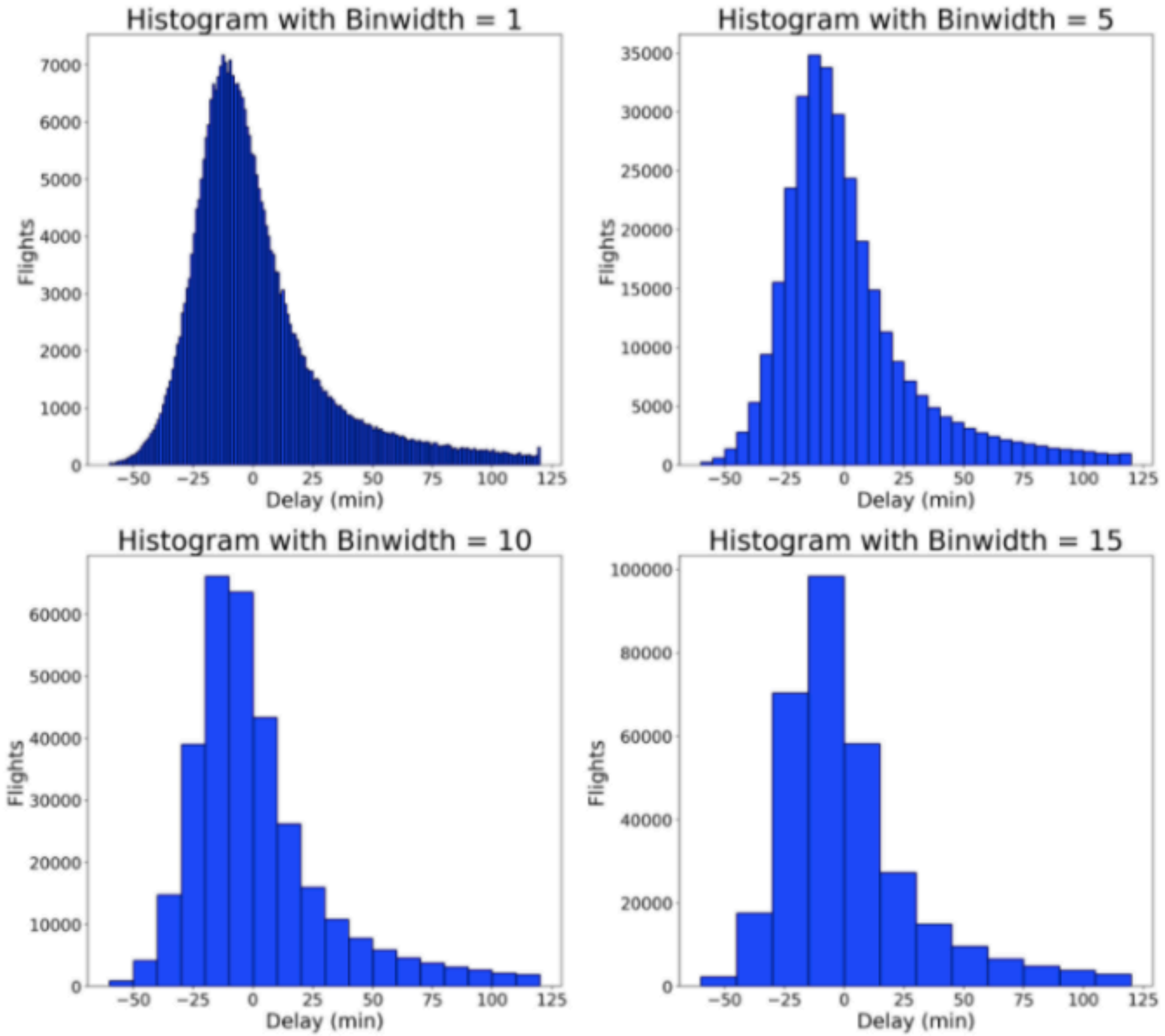
Specifically, the choice of model and model parameters is referred to as a modeling hypothesis *h*, and the problem involves finding *h* that best explains the data *X*.

CleverInsight

Histogram Estimator



A great way to get started exploring a single variable is with the histogram.
A histogram divides the variable into bins, counts the data points in each bin, and shows the bins on the x-axis and the counts on the y-axis.



Histograms with Different Binwidths

Histogram Kernel Density Estimator

https://www.youtube.com/watch?v=x5zLaWT5KPs

Discrete Probability Distributions (PMF)

Discrete probability distributions are used in machine learning, most notably in the modeling of binary and multi-class classification problems, but also in evaluating the performance for binary classification models, such as the calculation of confidence intervals, and in the modeling of the distribution of words in text for natural language processing.

Knowledge of discrete probability distributions is also required in the choice of activation functions in the output layer of deep learning neural networks for classification tasks and selecting an appropriate loss function.

A random variable is the quantity produced by a random process.

Discrete Probability Distributions (PMF)

A random variable that can have one of a finite set of specific outcomes.

Two types in ML:

- **Binary Random Variable**: x in {0, 1}
- **Categorical Random Variable**: x in {1, 2, …, K}

Each outcome or event for a discrete random variable has a probability.

The relationship between the events for a discrete random variable and their probabilities is called the discrete probability distribution and is summarized by a probability mass function, or PMF for short.

PMF: Probability Mass Function, returns the probability of a given outcome.
CDF: Cumulative Distribution Function, returns the probability of a value less than or equal to a given outcome.
PPF: Percent-Point Function, returns a discrete value that is less than or equal to the given probability.

Discrete Probability Distributions (PMF)

There are many common discrete probability distributions

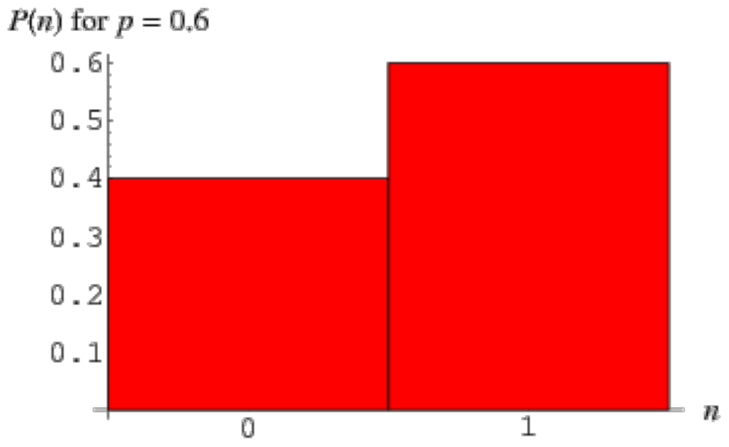
The most common are the Bernoulli and Multinoulli distributions for binary and categorical discrete random variables respectively, and the Binomial and Multinomial distributions that generalize each to multiple independent trials.

- **Binary Random Variable**: Bernoulli Distribution
- **Sequence of a Binary Random Variable**: Binomial Distribution
- **Categorical Random Variable**: Multinoulli Distribution
- **Sequence of a Categorical Random Variable**: Multinomial Distribution

CleverInsight

Bernoulli Distribution

The Bernoulli distribution is a discrete probability distribution that covers a case where an event will have a binary outcome as either a 0 or 1.



x in {0, 1}

In the case of flipping a fair coin, the value of *p* would be 0.5, giving a 50% probability of each outcome.

A common example of a Bernoulli trial in machine learning might be
a binary classification of a single example as the first class (0) or the second class (1).

The distribution can be summarized by a single variable ***p*** that defines the probability of an outcome 1. Given this parameter, the probability for each event can be calculated as follows:

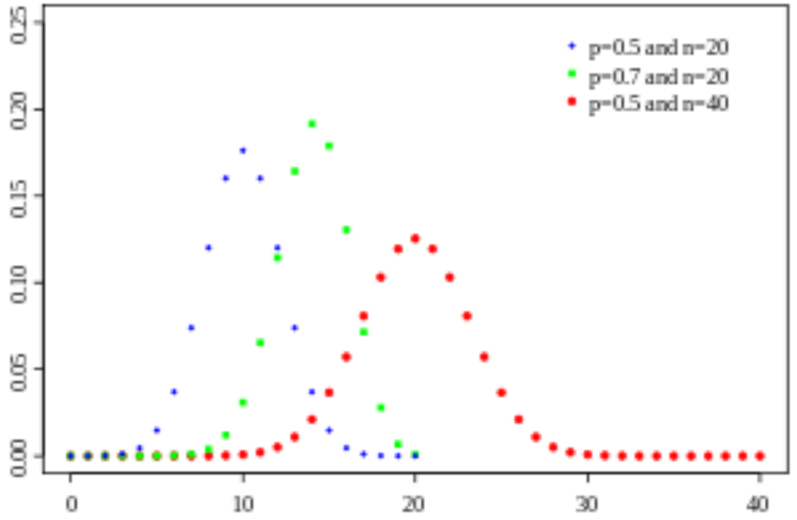
$P(x=1) = p$

$P(x=0) = 1 – p$

Binomial Distribution

The repetition of multiple independent Bernoulli trials is called a Bernoulli process. The outcomes of a Bernoulli process will follow a Binomial distribution. As such, the Bernoulli distribution would be a Binomial distribution with a single trial.

Some common examples of Bernoulli processes include:
- A sequence of independent coin flips.
- A sequence of independent births.



The performance of a machine learning algorithm on a binary classification problem can be analyzed as a Bernoulli process, where the prediction by the model on an example from a test set is a Bernoulli trial (correct or incorrect).

The Binomial distribution summarizes the number of successes in a given number of Bernoulli trials *k*, with a given probability of success for each trial *p*.

Multinoulli Distribution

The Multinoulli distribution, also called the categorical distribution, covers the case where an event will have one of K possible outcomes.

It is a generalization of the Bernoulli distribution from a binary variable to a categorical variable, where the number of cases *K* for the Bernoulli distribution is set to 2, *K=2*.

x in {1, 2, 3, …, K}

Some common examples of Multinoulli distribution is:
- A single roll of a die that will have an outcome in {1, 2, 3, 4, 5, 6}, e.g. K=6.

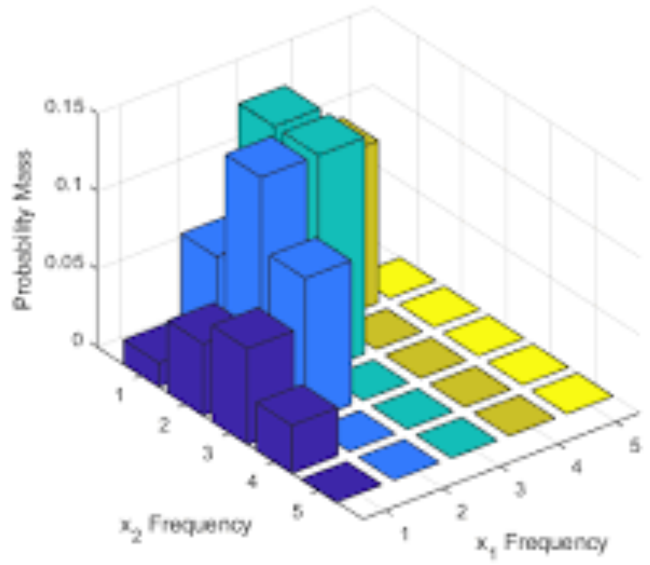
A common example of a Multinoulli distribution in machine learning might be a multi-class classification of a single example into one of *K* classes, e.g. one of three different species of the iris flower.

Multinomial Distribution

The repetition of multiple independent Multinoulli trials will follow a multinomial distribution.

The multinomial distribution is a generalization of the binomial distribution for a discrete variable with K outcomes.

An example of a multinomial process includes a sequence of independent dice rolls.



Some common examples of Multinomial distribution is:

- A common example of the multinomial distribution is the occurrence counts of words in a text document, from the field of natural language processing..

Take away

“The power of machine learning comes from its ability to learn patterns from large amounts of data.”