Can Machine Learning solve my problem?

9th BCD ISHEP Cargèse School - part I

28 March 2024 - Cargèse, France

Emille E. O. Ishida

Laboratoire de Physique de Clermont - Université Clermont-Auvergne
Clermont Ferrand, France

What ***impressive things*** *machine learning can and/or will be able to do?*

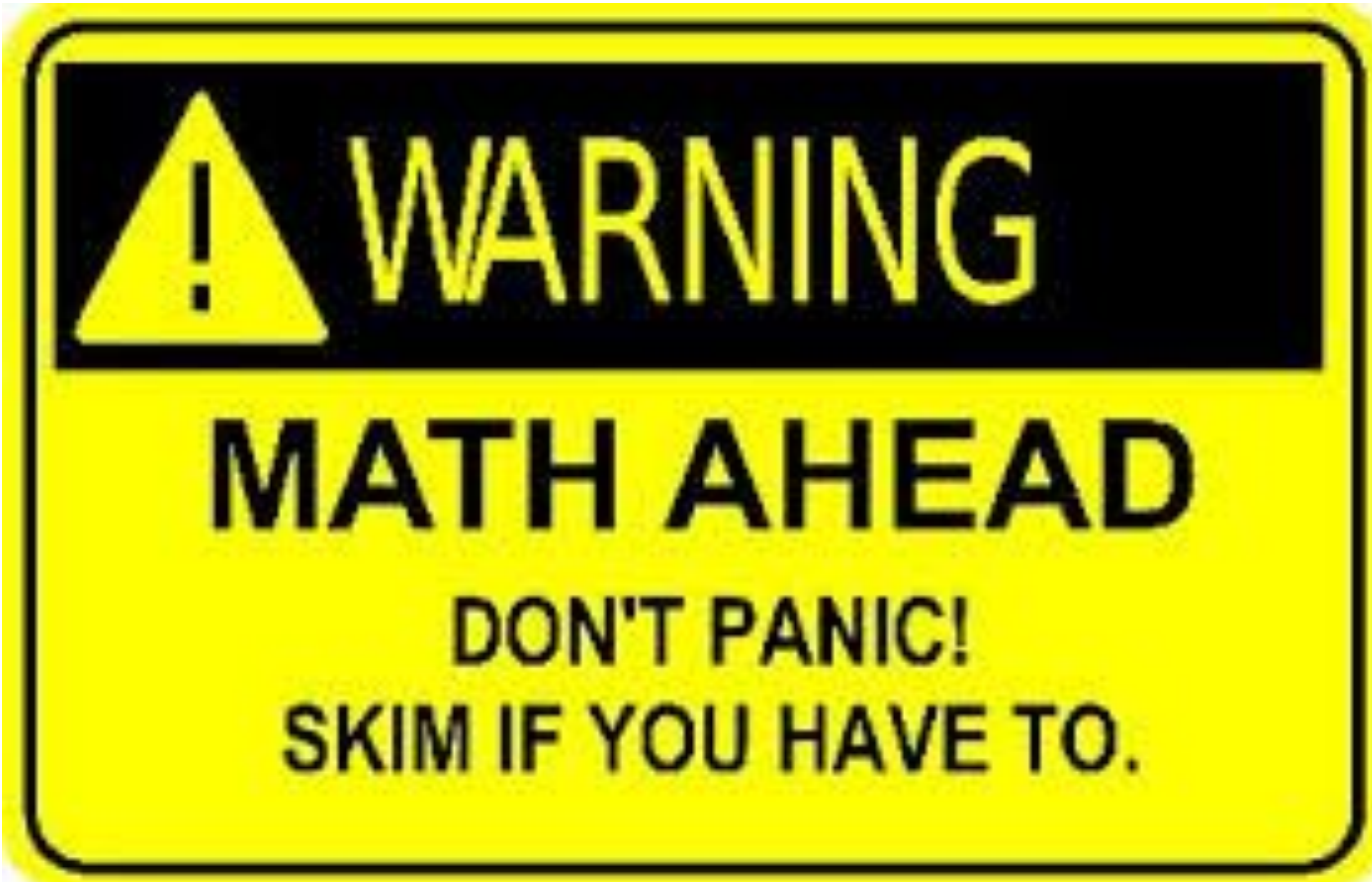
Join at menti.com with code: 7907 6385

Can Machine Learning solve my problem?

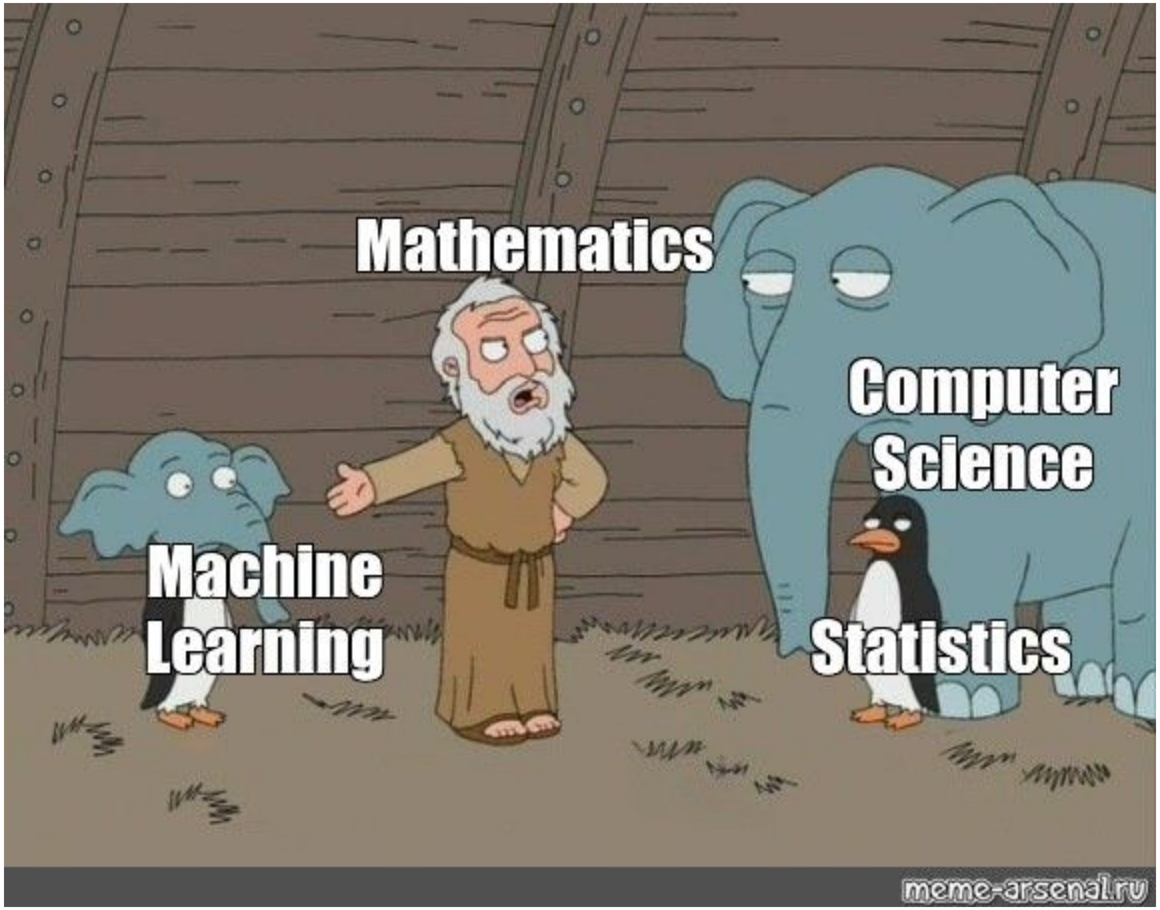
I do not know

9th BCD ISHEP Cargèse School - part I
28 March 2024 - Cargèse, France

Emille E. O. Ishida
Laboratoire de Physique de Clermont - Université Clermont-Auvergne
Clermont Ferrand, France



https://www.general-staff.com/wp-content/uploads/2019/03/MathWarning-300x193.jpg



https://www.meme-arsenal.com/en/create/meme/1868835

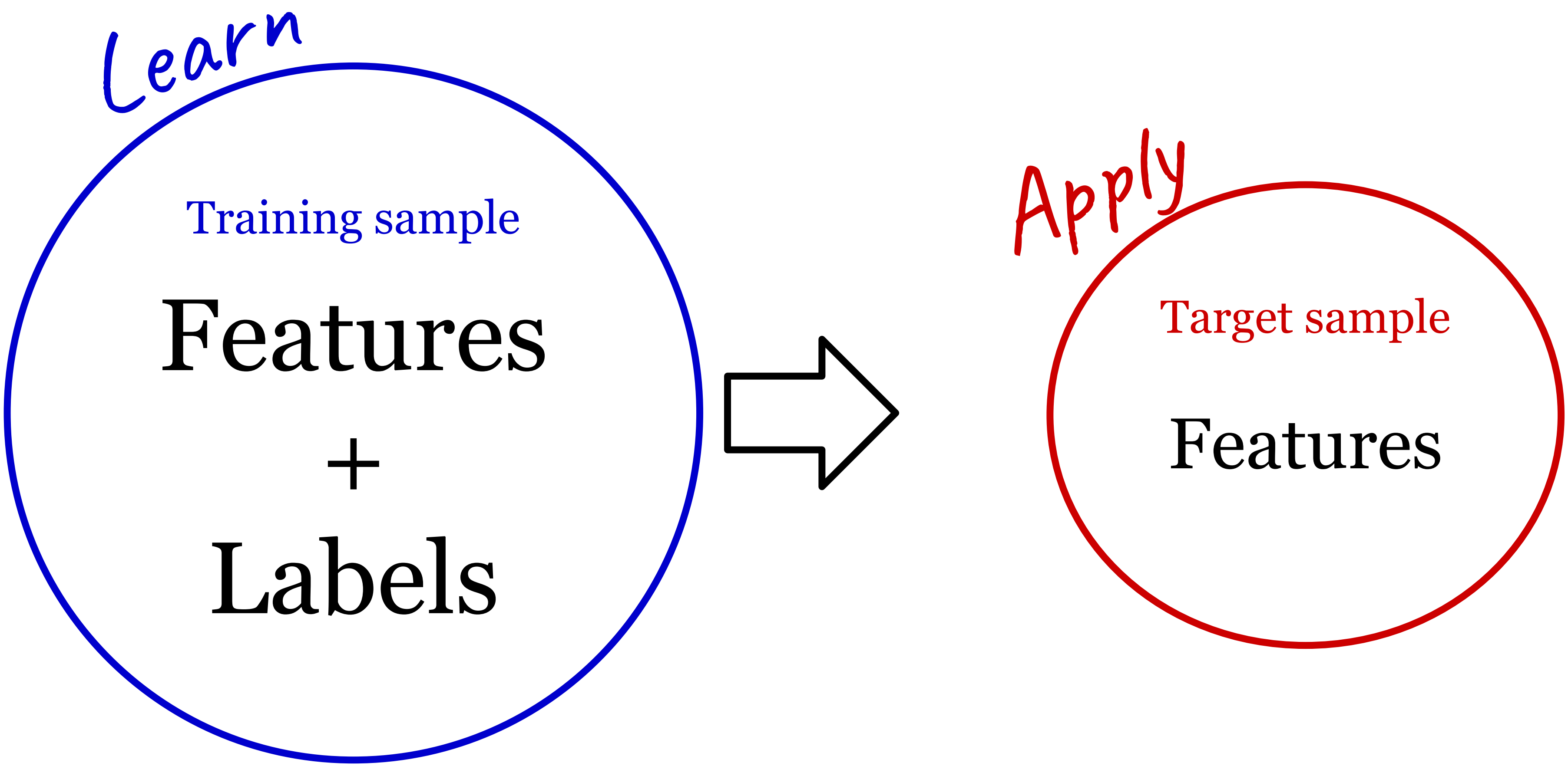
What is learning ?

“A relatively permanent change in behaviour due to past experiences.”

D. Coon, *Introduction to psychology: exploration and application* (1983)

Supervised Learning

Learn by example



Question:

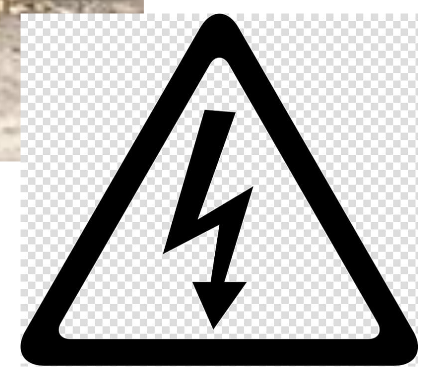
List 2 animals that you believe are capable of learning.

Join at menti.com with code: 7907 6385

Rat bait shyness - I

Rzóska, J. (1953). Bait shyness, a study in rat behavior. British Journal of Animal Behaviour, 1, 128–135

Rat bait shyness - II



Rzóska, J. (1953). Bait shyness, a study in rat behavior. British Journal of Animal Behaviour, 1, 128–135

Question:

- Do you believe the rat will learn the correlation between bad food ⇒ shock and/or sound ⇒ nausea?

Join at menti.com with code: 7907 6385

Question:

- What aspect of the rat learning model prevents it from understanding the input ⇒ output correlation?

Pigeon superstition

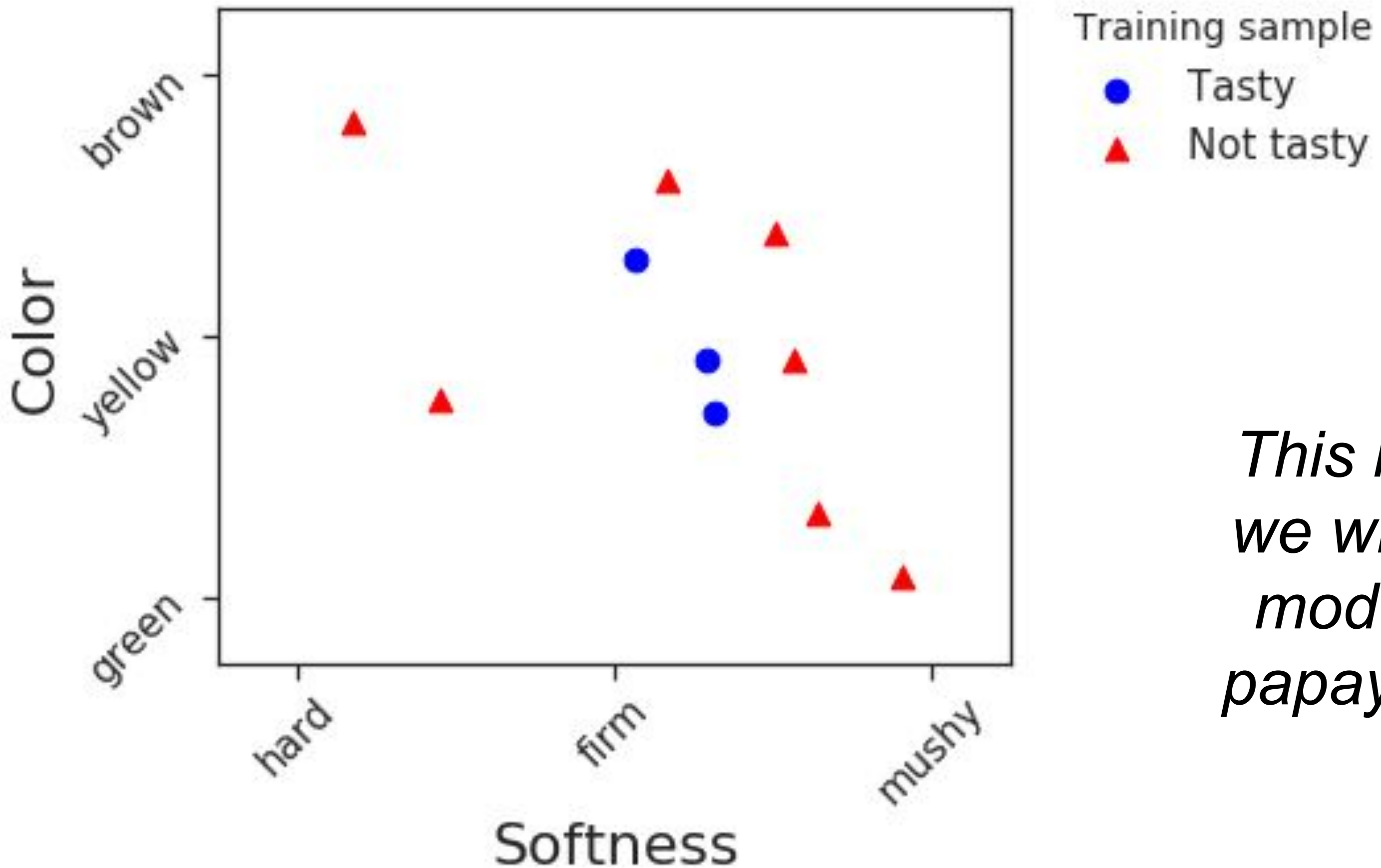
Skinner, B. F. "'Superstition' in the Pigeon", Journal of Experimental Psychology#38, 1947

Take home message

Priors knowledge is crucial for effective learning

Papaya tasting

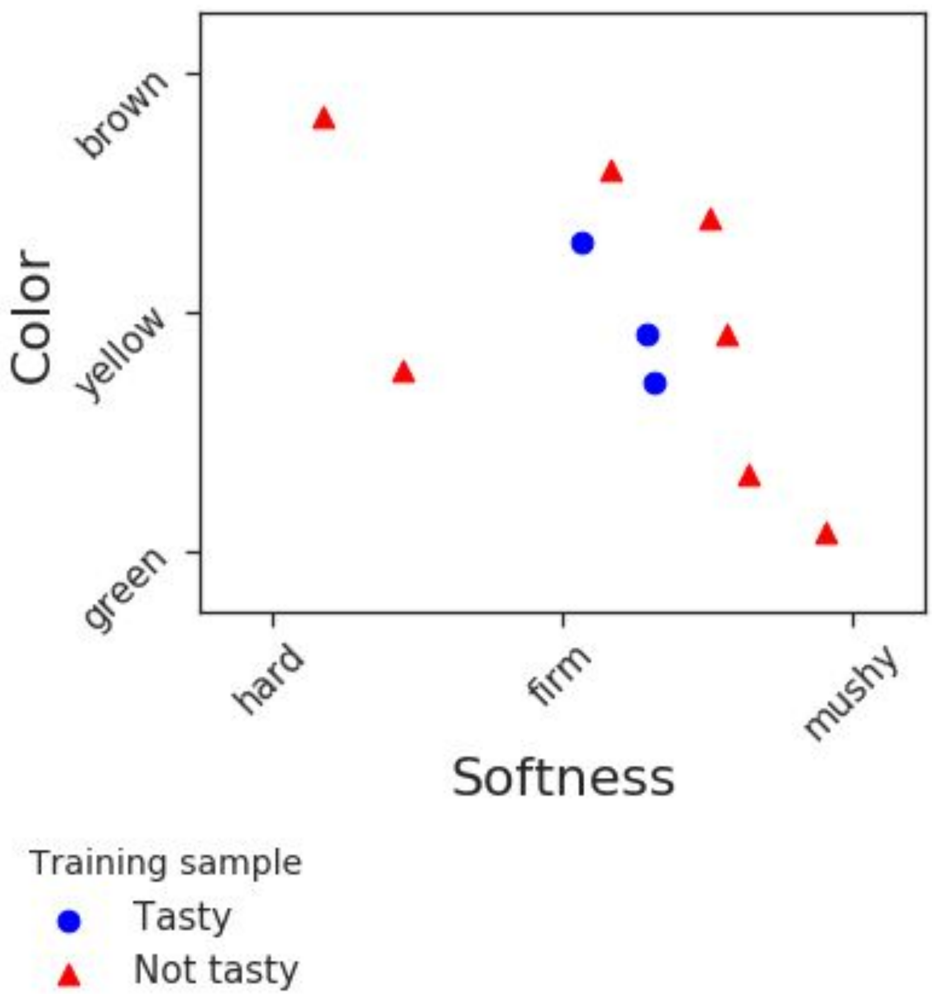
Binary classification



This is all the data we will input to the model about the papayas in the real world!

YouTube class on the papaya testing example:
https://www.youtube.com/watch?v=b5NlRg8SjZg&list=PLPW2keNyw-usgvmR7FTQ3ZRjfLs5jT4BO&index=2&t=0s

Papaya tasting



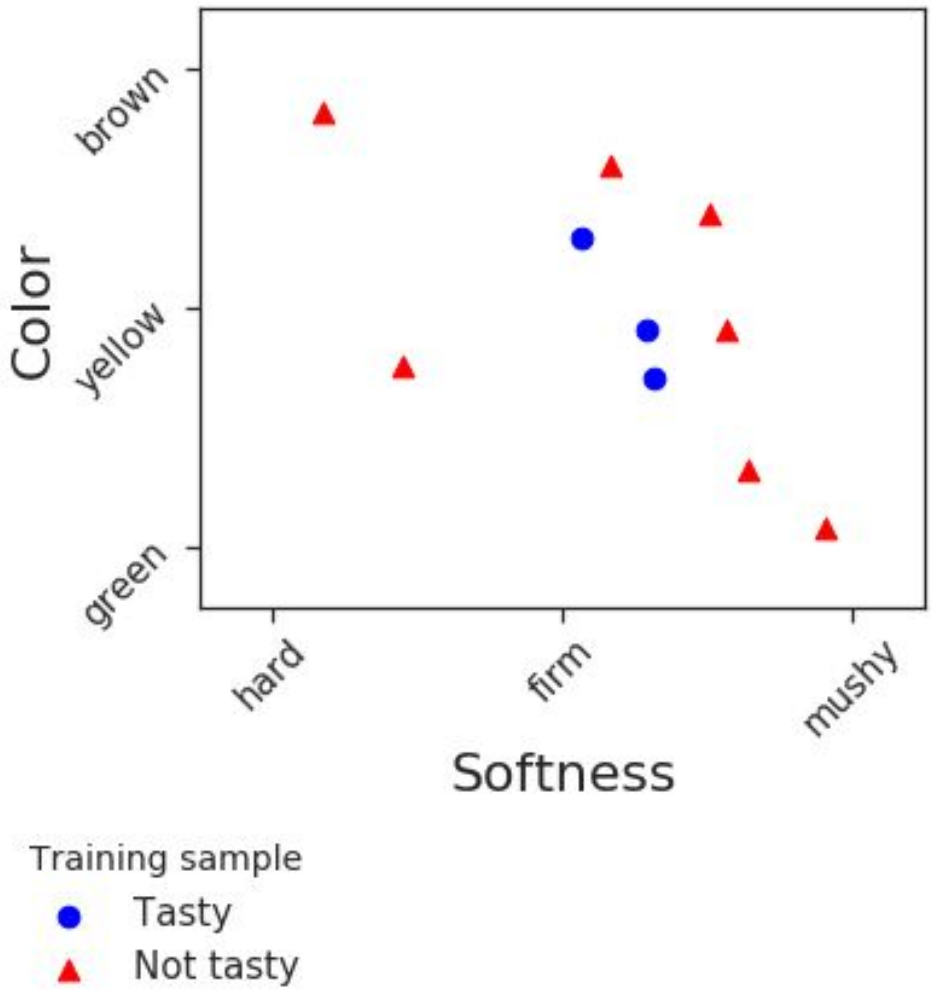
X: set of all features,

x = [softness, color]

Y: set of possible labels,

y = [tasty, not tasty]

Papaya tasting



X: set of all features,

x = [softness, color]

Y: set of possible labels,

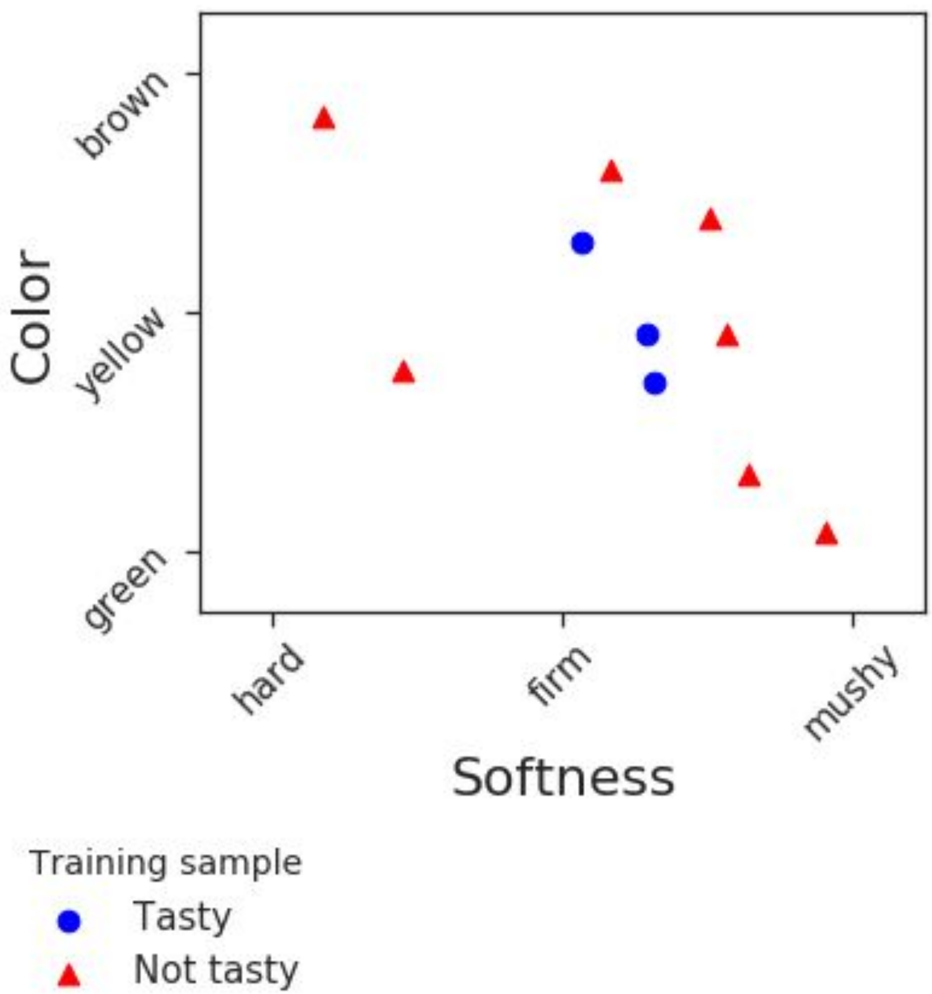
y = [tasty, not tasty]

D: data generation model,

$D \Rightarrow P(X)$

A controlled example:

Papaya tasting



X: set of all features,

x = [softness, color]

Y: set of possible labels,

y = [tasty, not tasty]

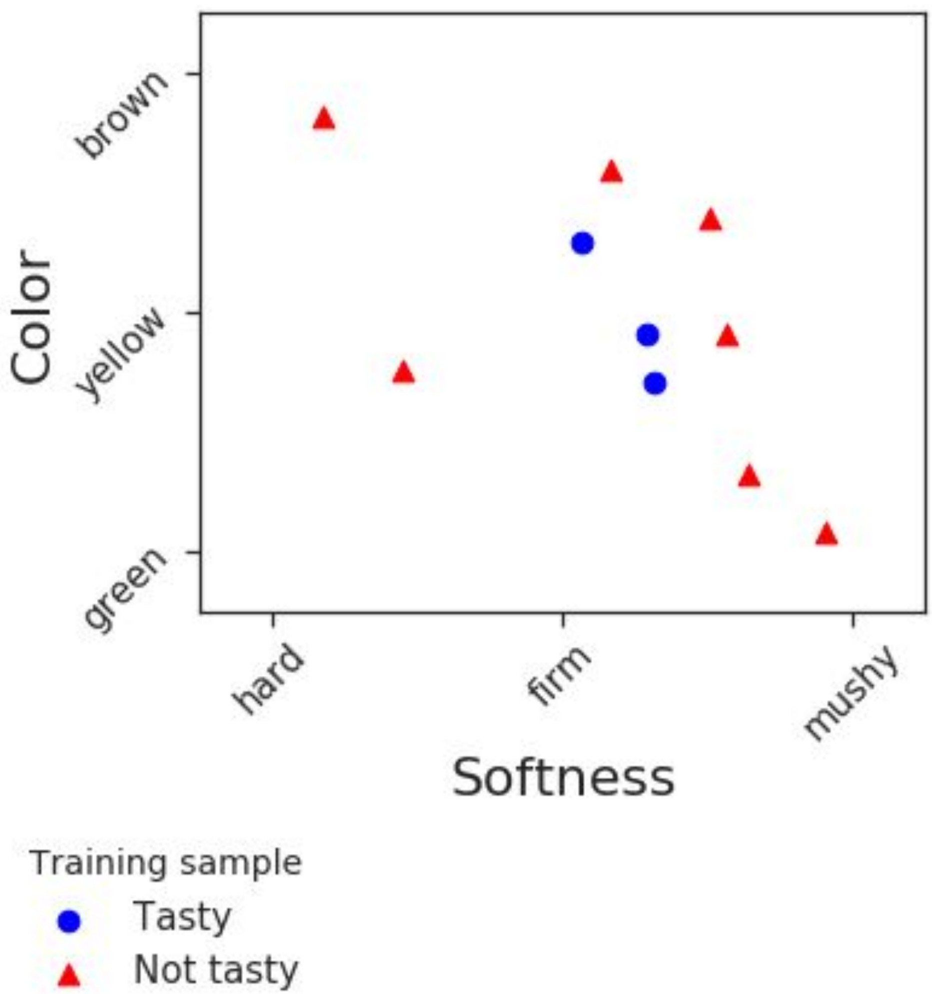
D: data generation model,

$D \Longrightarrow P(X)$

True Labelling function: $y = f(x)$

A controlled example:

Papaya tasting



X: set of all features,

x = [softness, color]

Y: set of possible labels,

y = [tasty, not tasty]

D: data generation model,

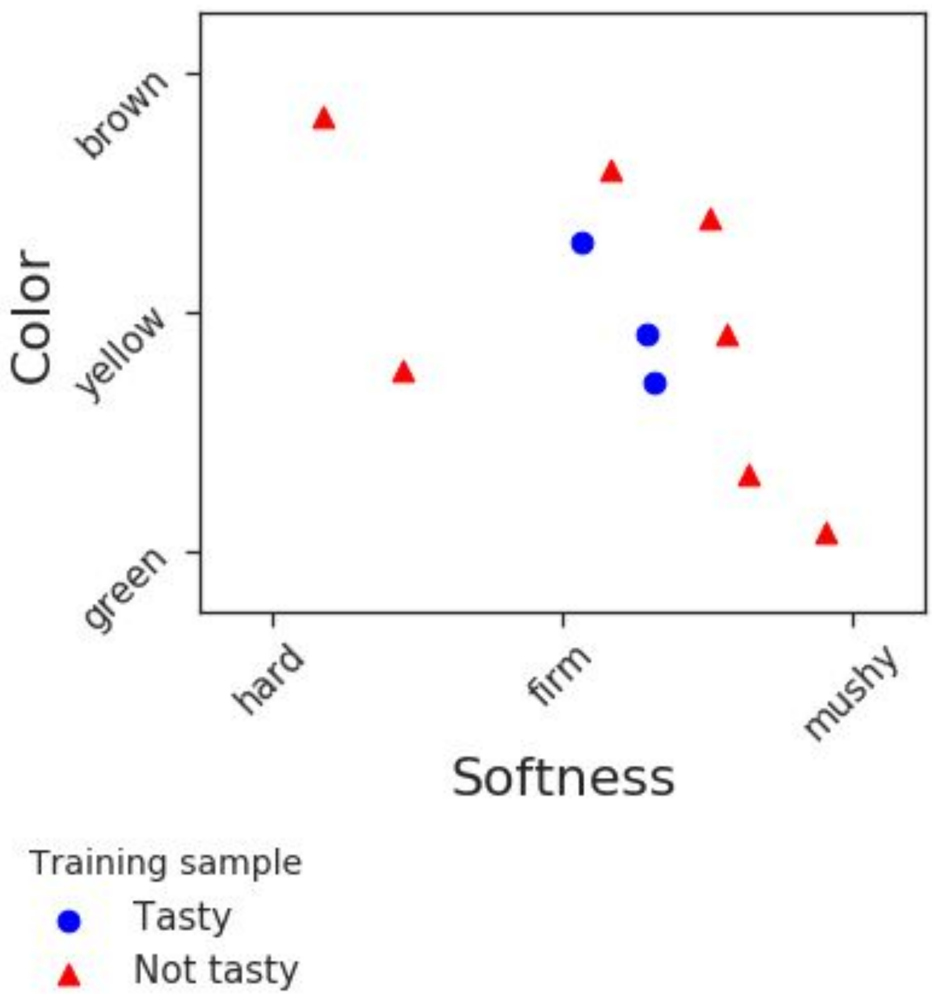
$D \Longrightarrow P(X)$

True Labelling function: $y = f(x)$

S: training sample: $[x_i, y_i]$, $i \in training$

m: number of objects for training

Papaya tasting



X: set of all features,

x = [softness, color]

Y: set of possible labels,

y = [tasty, not tasty]

D: data generation model,

$D \Longrightarrow P(X)$

True Labelling function: $y = f(x)$

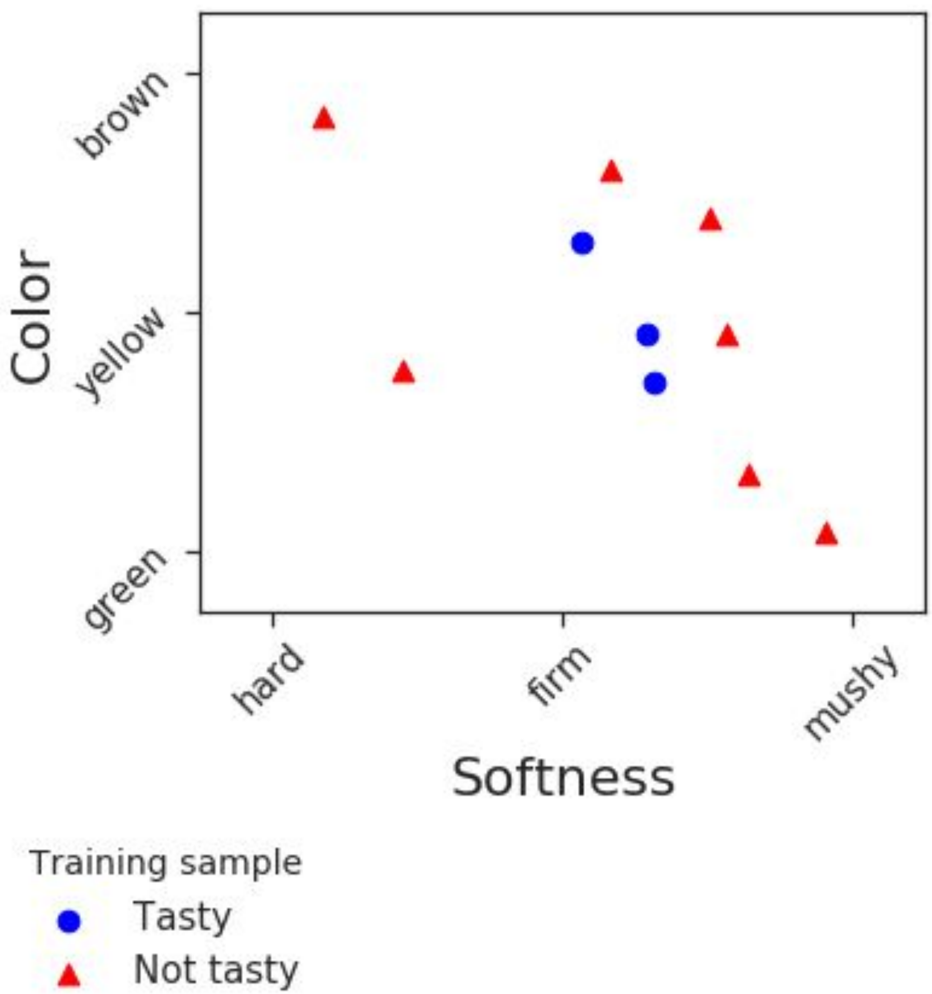
S: training sample: $[x_i, y_i]$, $i \in training$

m: number of objects for training

h_S: learner: $y_{est;i} = h_S(x_i)$

A controlled example:

Papaya tasting



X: set of all features,

x = [softness, color]

Y: set of possible labels,

y = [tasty, not tasty]

D: data generation model,

$D \Rightarrow P(X)$

True Labelling function: $y = f(x)$

S: training sample: $[x_i, y_i]$, $i \in$ *training*

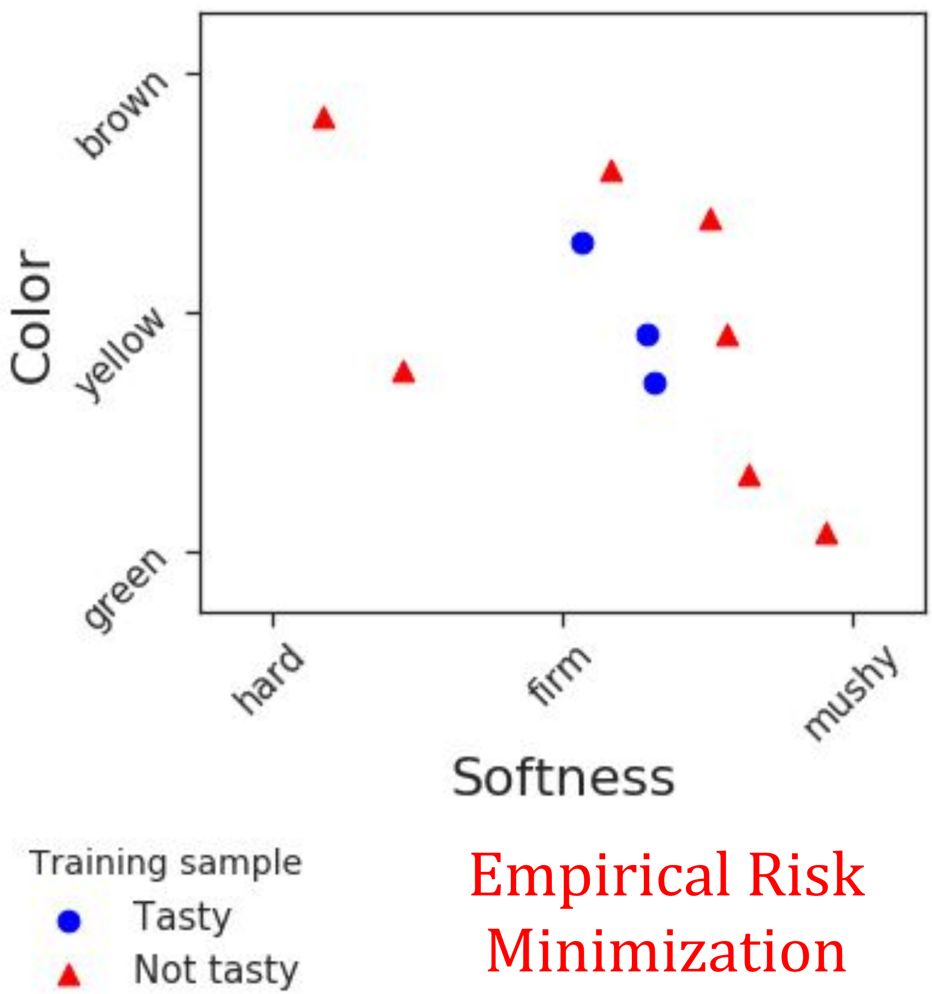
m: number of objects for training

h_S learner: $y_{est;i} = h_S(x_i)$

L metric: $L\,(y_{true;i} - y_{est;i})$, $i \in$ *training*

A controlled example:

Papaya tasting



X: set of all features,

x = [softness, color]

Y: set of possible labels,

y = [tasty, not tasty]

D: data generation model,

$D \Rightarrow P(X)$

True Labelling function: $y = f(x)$

S: training sample: $[x_i, y_i]$, $i \in training$

m: number of objects for training

h_S learner: $y_{est;i} = h_S(x_i)$

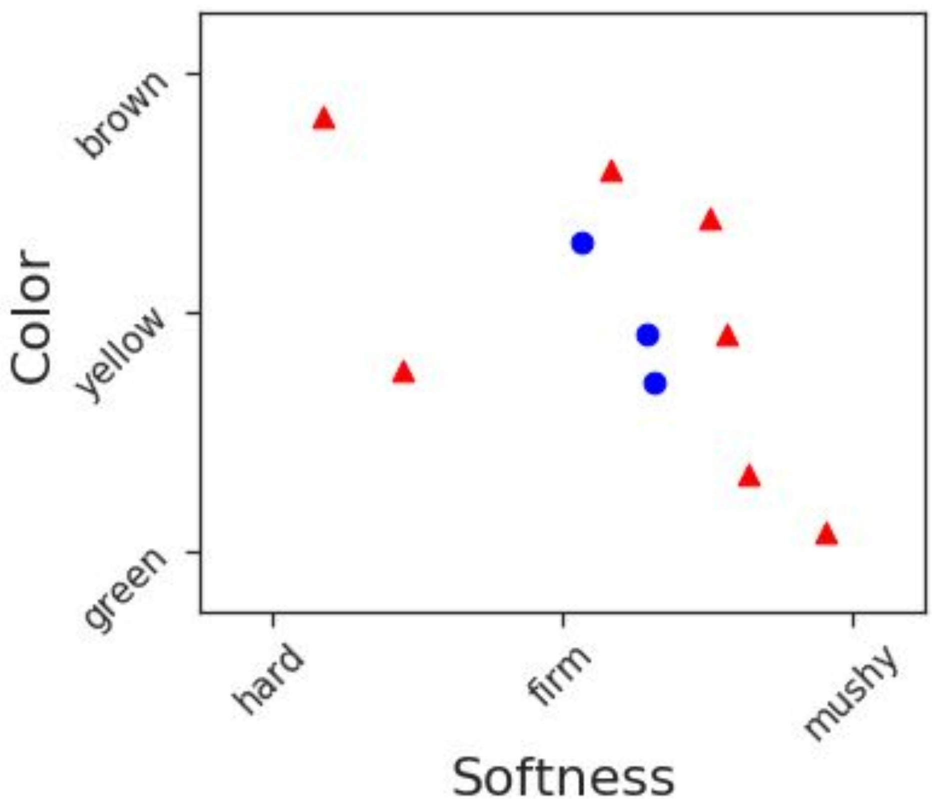
L metric: $L\,(y_{true;i} - y_{est;i})$, $i \in training$

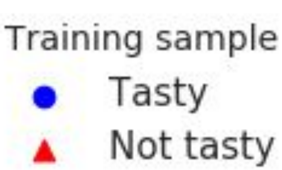
Empirical Risk Minimization (ERM)

$L \rightarrow$ *fraction of incorrect predictions*

A controlled example:

Papaya tasting





X: set of all features,

x = [softness, color]

Y: set of possible labels,

y = [tasty, not tasty]

D: data generation model,

$D \Rightarrow P(X)$

True Labelling function: *y = f(x)*

S: training sample: $[x_i, y_i]$, $i \in$ *training*

m: number of objects for training

h_S learner: $y_{est;i} = h_S(x_i)$

L metric: $L\,(y_{true;i} - y_{est;i})$, $i \in$ *training*

Empirical Risk Minimization (ERM)

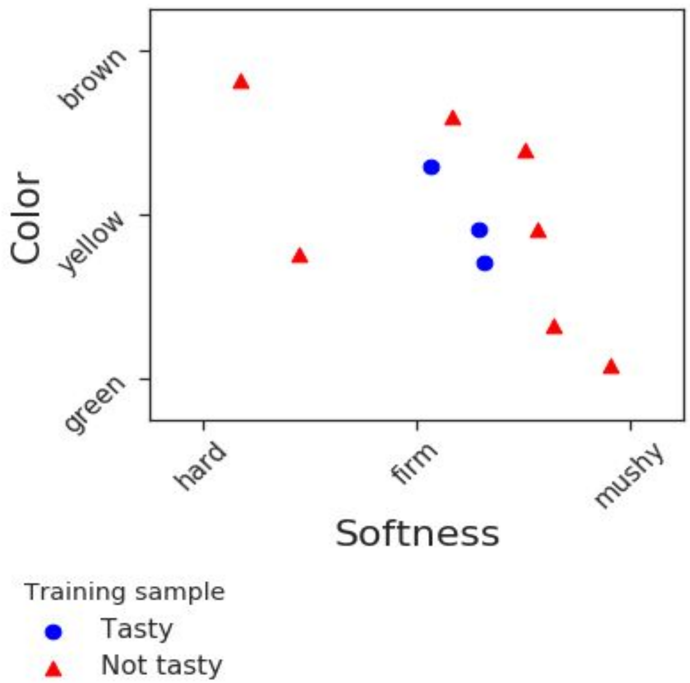
$$L_{\mathcal{D}}(h_S) = \frac{|\{x \in \mathcal{D} : h_S(x) \neq f(x)\}|}{m}$$

A controlled example:

Papaya tasting

Proposed learner:

$$h_S(x) = \begin{cases} y_i & \text{if } x = x_i \mid \{x_i \in S\} \\ 0 & \text{otherwise} \end{cases}$$



X: set of all features,

x = [softness, color]

Y: set of possible labels,

y = [tasty, not tasty]

D: data generation model,

D ⟹ P(X)

True Labelling function: y = f(x)

S: training sample: *$[x_i, y_i]$*, *$i \in$ training*

m: number of objects for training

h_S learner: *$y_{est;i} = h_S(x_i)$*

L metric: *$L(y_{true;i} - y_{est;i})$*, *$i \in$ training*

$$L_{\mathcal{D}}(h_S) = \frac{|\{x \in \mathcal{D} : h_S(x) \neq f(x)\}|}{m}$$

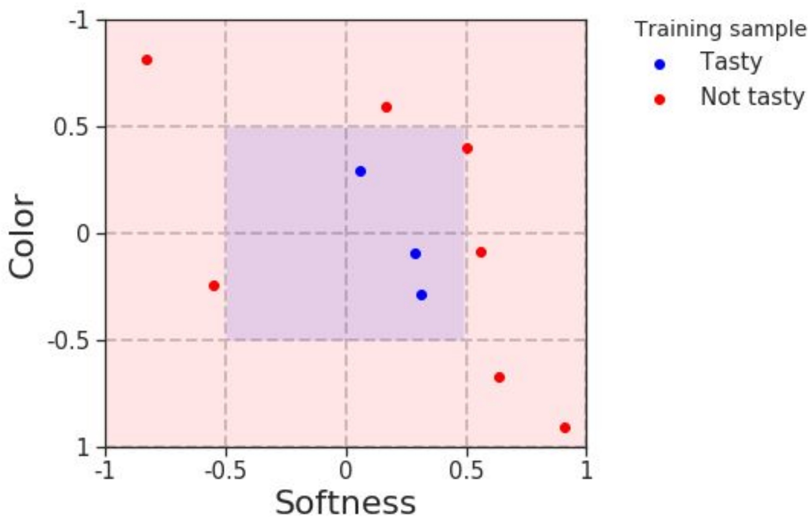
A controlled example:

Papaya tasting

Proposed learner:

$$h_S(x) = \begin{cases} y_i & \text{if } x = x_i \mid \{x_i \in S\} \\ 0 & \text{otherwise} \end{cases}$$

Toy model ...



X: set of all features,

x = [softness, color]

Y: set of possible labels,

y = [tasty, not tasty]

D: data generation model,

D ⟹ P(X)

True Labelling function: y = f(x)

S: training sample: $[x_i, y_i]$, $i \in$ *training*

m: number of objects for training

h_S learner: $y_{est;i} = h_S(x_i)$

L metric: $L\,(y_{true;i} - y_{est;i})$, $i \in$ *training*

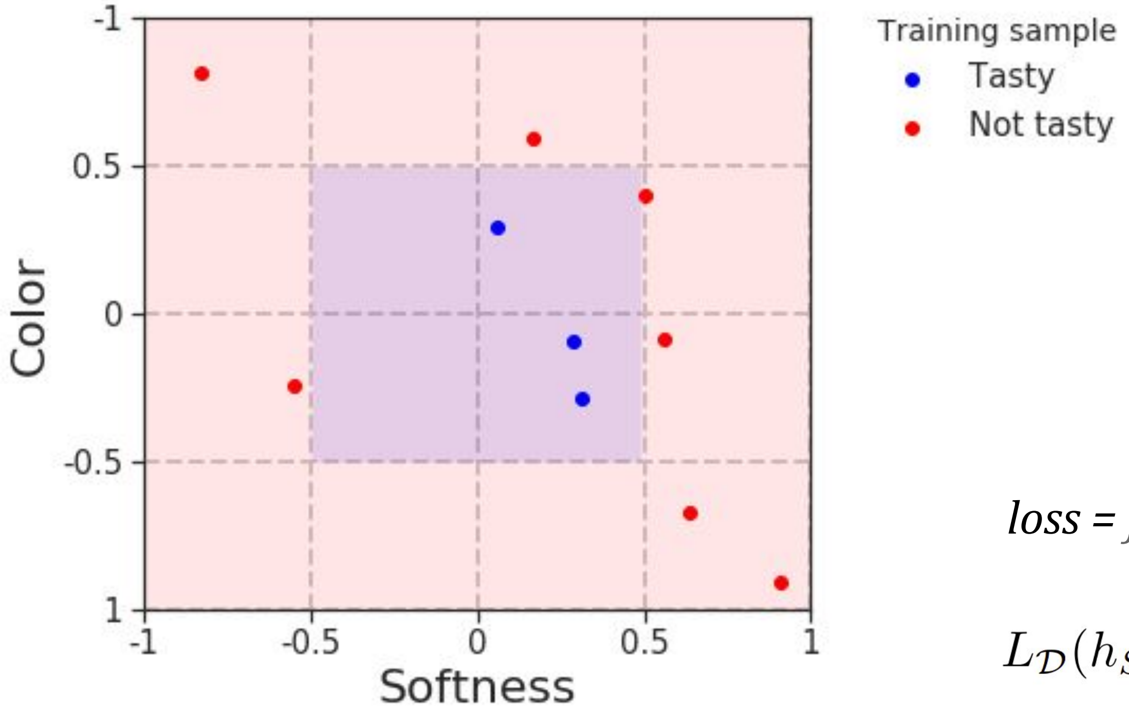
$$L_{\mathcal{D}}(h_S) = \frac{|\{x \in \mathcal{D} : h_S(x) \neq f(x)\}|}{m}$$

A controlled example:

Question:

Proposed learner:

$$h_S(x) = \begin{cases} y_i & \text{if } x = x_i \mid \{x_i \in S\} \\ 0 & \text{otherwise} \end{cases}$$



[tasty, not tasty] = [1, 0]

What is the expected loss when this model is applied to an arbitrary test sample?

Join at menti.com with code: 7907 6385

loss = fraction of incorrect predictions

$$L_{\mathcal{D}}(h_S) = \frac{|\{x \in \mathcal{D} : h_S(x) \neq f(x)\}|}{m}$$

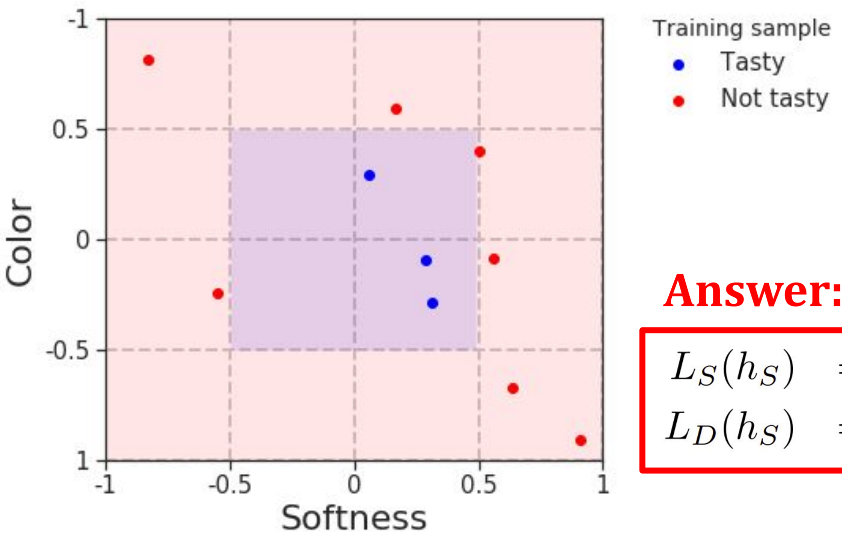
A controlled example:

Papaya tasting

Proposed learner:

$$h_S(x) = \begin{cases} y_i & \text{if } x = x_i \mid \{x_i \in S\} \\ 0 & \text{otherwise} \end{cases}$$

Answer:



Answer:

$$L_S(h_S) = 0.0$$
$$L_D(h_S) = 0.25$$

X: set of all features,

x = [softness, color]

Y: set of possible labels,

y = [tasty, not tasty] = [1, 0]

D: data generation model,

D ⟹ P(X)

True Labelling function: y = f(x)

S: training sample: *[x_i, y_i], i ∈ training*

m: number of objects for training

h_S learner: $y_{est;i} = h_S x_i$

L metric: $L\,(y_{true,i} - y_{est;i})$, *i ∈ training*

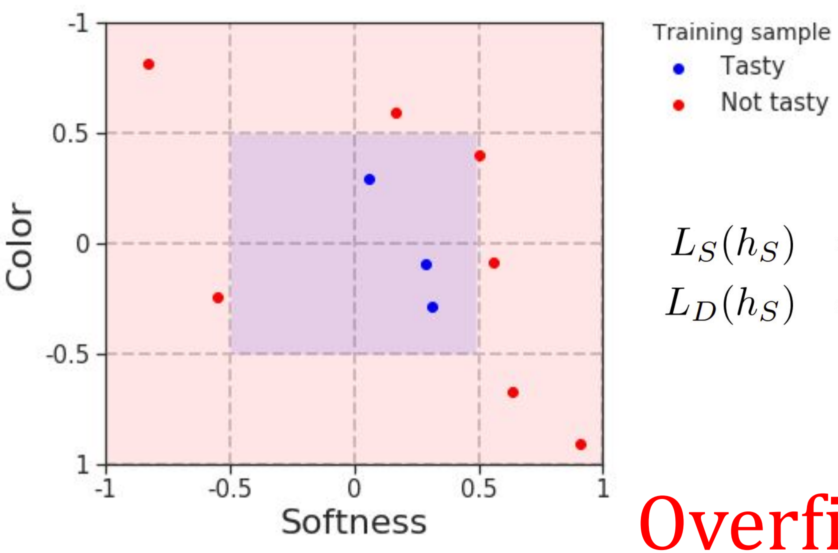
$$L_{\mathcal{D}}(h_S) = \frac{|\{x \in \mathcal{D} : h_S(x) \neq f(x)\}|}{m}$$

Papaya tasting

Proposed learner:

$$h_S(x) = \begin{cases} y_i & \text{if } x = x_i \mid \{x_i \in S\} \\ 0 & \text{otherwise} \end{cases}$$

Answer:



$$L_S(h_S) = 0.0$$
$$L_D(h_S) = 0.25$$

X: set of all features,
x = [softness, color]

Y: set of possible labels,
y = [tasty, not tasty]

D: data generation model,
$D \Longrightarrow P(X)$

True Labelling function: y = f(x)

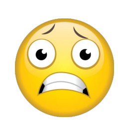
S: training sample: $[x_i, y_i]$, $i \in$ *training*

m: number of objects for training

h_S learner: $y_{est;i} = h_S x_i$

L metric: $L\,(y_{true,i} - y_{est;i})$, $i \in$ *training*

$$L_{\mathcal{D}}(h_S) = \frac{|\{x \in \mathcal{D} : h_S(x) \neq f(x)\}|}{m}$$

Overfitting! 

Question:

- How can we avoid overfitting?

Question:

- How can we avoid overfitting?

by adding prior knowledge ...

Choosing the learner

X: set of all features,

x = [softness, color]

Y: set of possible labels,

y = [tasty, not tasty]

D: data generation model,

$D \Longrightarrow P(X)$

True Labelling function: y = f(x)

S: training sample: $[x_i, y_i]$, $i \in training$

m: number of objects for training

h_S learner: $y_{est;i} = h_S(x_i)$

$$h_S(x) = \begin{cases} y_i & \text{if } x = x_i \mid \{x_i \in S\} \\ 0 & \text{otherwise} \end{cases}$$

L: loss: $L\,(y_{true;i} - y_{est;i})$, $i \in training$

$$L_{\mathcal{D}}(h_S) = \frac{|\{x \in \mathcal{D} : h_S(x) \neq f(x)\}|}{m}$$

Choosing the learner

X: set of all features,

x = [softness, color]

Y: set of possible labels,

y = [tasty, not tasty]

D: data generation model,

$D \Longrightarrow P(X)$

True Labelling function: y = f(x)

S: training sample: $[x_i, y_i]$, $i \in training$

m: number of objects for training

h_S learner: $y_{est;i} = h_S(x_i)$

$$h_S(x) = \begin{cases} y_i & \text{if } x = x_i \mid \{x_i \in S\} \\ 0 & \text{otherwise} \end{cases}$$

L: loss: $L\,(y_{true;i} - y_{est;i})$, $i \in training$

$$L_{\mathcal{D}}(h_S) = \frac{|\{x \in \mathcal{D} : h_S(x) \neq f(x)\}|}{m}$$

Hypothesis class ($\mathcal{H}$):

$$h : \mathcal{X} \longrightarrow \mathcal{Y}; \qquad h \in \mathcal{H}$$

$$\mathrm{ERM}_{\mathcal{H}}(S) \in \underset{h \in \mathcal{H}}{\operatorname{argmin}}\, L_S(h),$$

Choosing the learner

X: set of all features,

x = [softness, color]

Y: set of possible labels,

y = [tasty, not tasty]

D: data generation model,

$D \Longrightarrow P(X)$

True Labelling function: y = f(x)

S: training sample: $[x_i, y_i]$, $i \in training$

h_S learner: $y_{est;i} = h_S(x_i)$

$$h_S(x) = \begin{cases} y_i & \text{if } x = x_i \mid \{x_i \in S\} \\ 0 & \text{otherwise} \end{cases}$$

L: *loss*: $L\,(y_{true;i} - y_{est;i})$, $i \in training$

$$L_{\mathcal{D}}(h_S) = \frac{|\{x \in \mathcal{D} : h_S(x) \neq f(x)\}|}{m}$$

Hypothesis class ($\mathcal{H}$):

$$h : \mathcal{X} \longrightarrow \mathcal{Y}; \qquad h \in \mathcal{H}$$

$$\mathrm{ERM}_{\mathcal{H}}(S) \in \underset{h \in \mathcal{H}}{\operatorname{argmin}}\, L_S(h),$$

- $\mathcal{H}$ is finite, $N_{\mathcal{H}}$ = number of hypothesis
- The true labelling function is part of $\mathcal{H}$:

$$f \in \mathcal{H}$$

Choosing the learner

χ: set of all features,
x = [softness, color]

Y: set of possible labels,
y = [tasty, not tasty]

D: data generation model,
$D \Longrightarrow P(\chi)$

True Labelling function: y = f(x)

S: training sample: $[x_i, y_i]$, $i \in training$

h_S learner: $y_{est;i} = h_S(x_i)$

$$h_S(x) = \begin{cases} y_i & \text{if } x = x_i \mid \{x_i \in S\} \\ 0 & \text{otherwise} \end{cases}$$

L: loss: $L(y_{true;i} - y_{est;i})$, $i \in training$

$$L_{\mathcal{D}}(h_S) = \frac{|\{x \in \mathcal{D} : h_S(x) \neq f(x)\}|}{m}$$

Hypothesis class ($\mathcal{H}$):

$$h : \mathcal{X} \longrightarrow \mathcal{Y}; \qquad h \in \mathcal{H}$$

$$\mathrm{ERM}_{\mathcal{H}}(S) \in \operatorname*{argmin}_{h \in \mathcal{H}} L_S(h),$$

- $\mathcal{H}$ is finite, $N_{\mathcal{H}}$ = number of hypothesis
- The true labelling function is part of $\mathcal{H}$:

  $$f \in \mathcal{H}$$

- *S* is identically independently distributed (*i.i.d.*) from *D*

Choosing the learner

χ: set of all features,
x = [softness, color]
Y: set of possible labels,
y = [tasty, not tasty]
D: data generation model,
$D \Longrightarrow P(\chi)$
True Labelling function: y = f(x)

S: training sample: $[x_i, y_i]$, $i \in$ *training*
h_S learner: $y_{est;i} = h_S(x_i)$

$$h_S(x) = \begin{cases} y_i & \text{if } x = x_i \mid \{x_i \in S\} \\ 0 & \text{otherwise} \end{cases}$$

L: *loss*: $L\,(y_{true;i} - y_{est;i})$, $i \in$ *training*

$$L_{\mathcal{D}}(h_S) = \frac{|\{x \in \mathcal{D} : h_S(x) \neq f(x)\}|}{m}$$

Hypothesis class ($\mathcal{H}$):

$$h : \mathcal{X} \longrightarrow \mathcal{Y}; \qquad h \in \mathcal{H}$$

$$\mathrm{ERM}_{\mathcal{H}}(S) \in \underset{h \in \mathcal{H}}{\operatorname{argmin}}\, L_S(h),$$

- $\mathcal{H}$ is finite, $N_{\mathcal{H}}$ = number of hypothesis
- The true labelling function is part of $\mathcal{H}$:

$$f \in \mathcal{H}$$

- *S* is identically independently distributed (*i.i.d.*) from *D*

Machine Learning:

(a personal favorite)

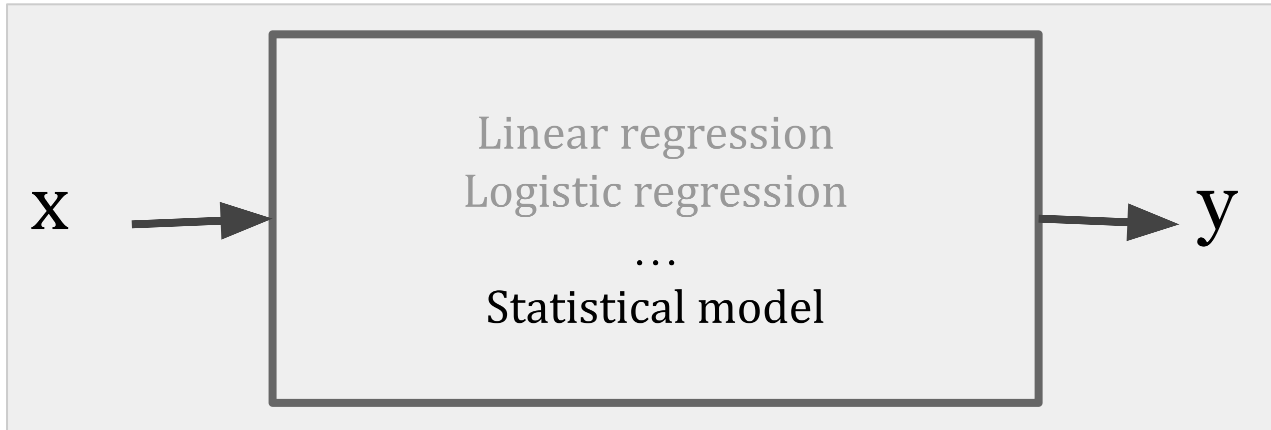
Supervised definition

Hypothesis:

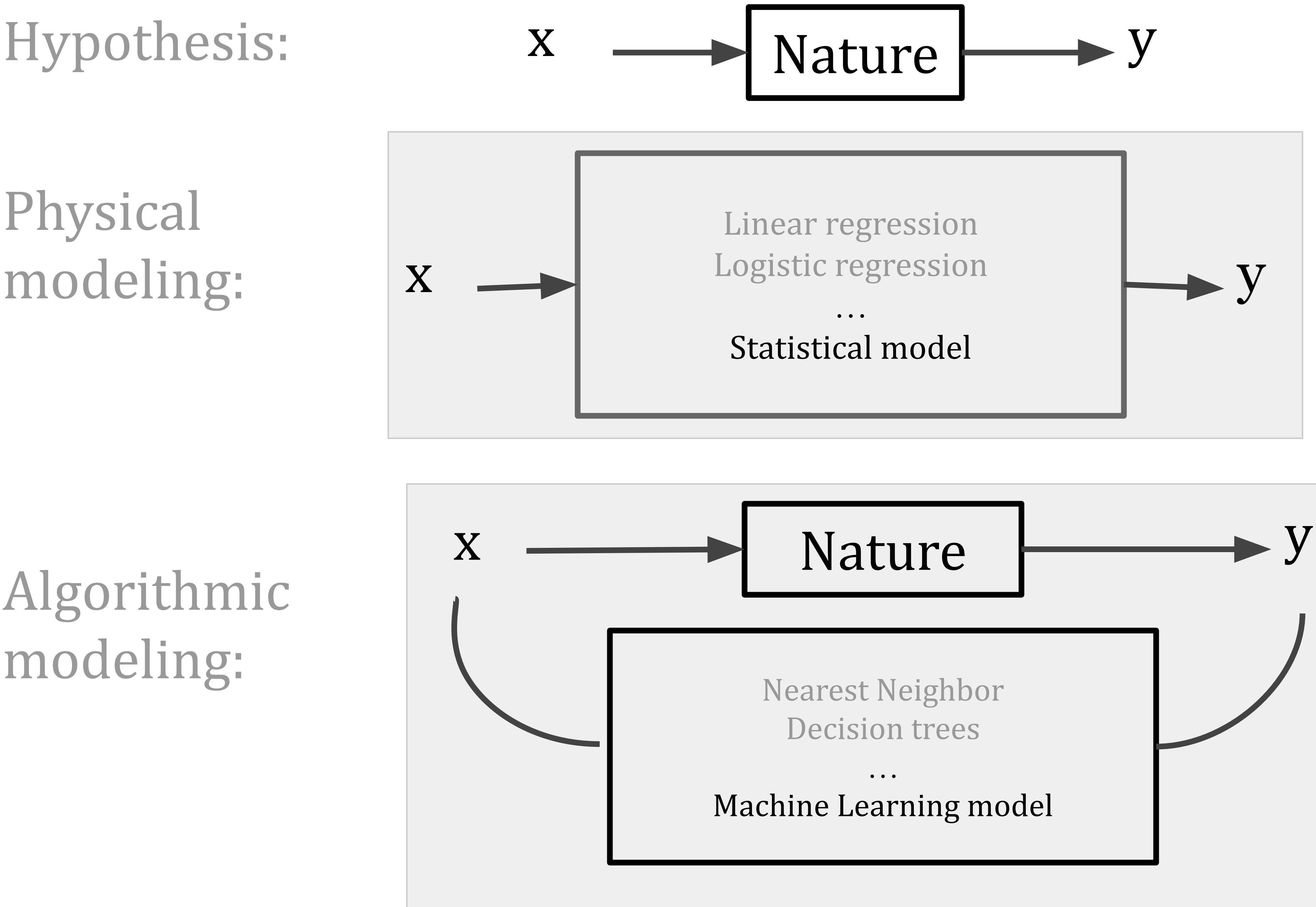
Breiman, L., Statistical Modeling: The Two Cultures, Stat. Sci, Volume 16 (2001)

Hypothesis:

Physical modeling:



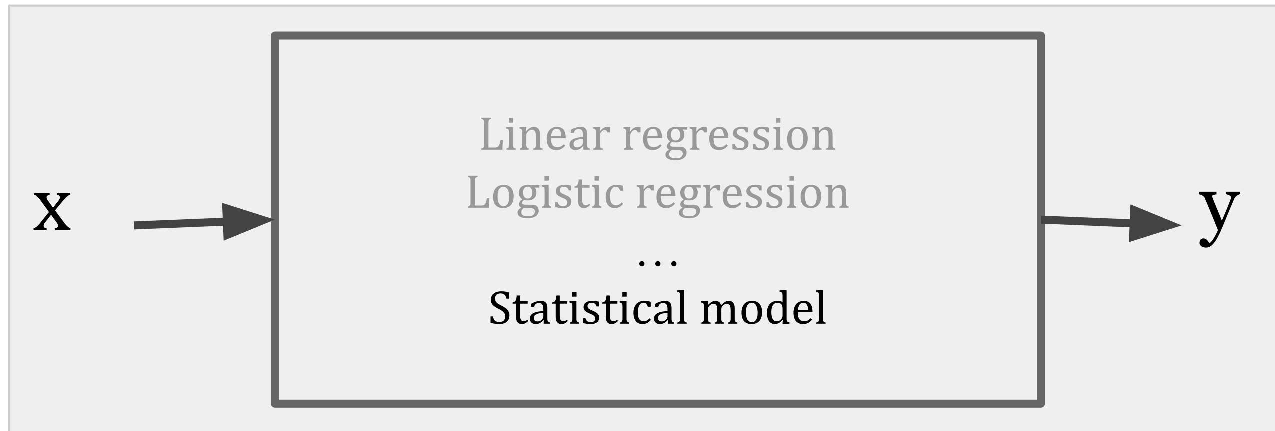
Breiman, L., Statistical Modeling: The Two Cultures, Stat. Sci, Volume 16 (2001)



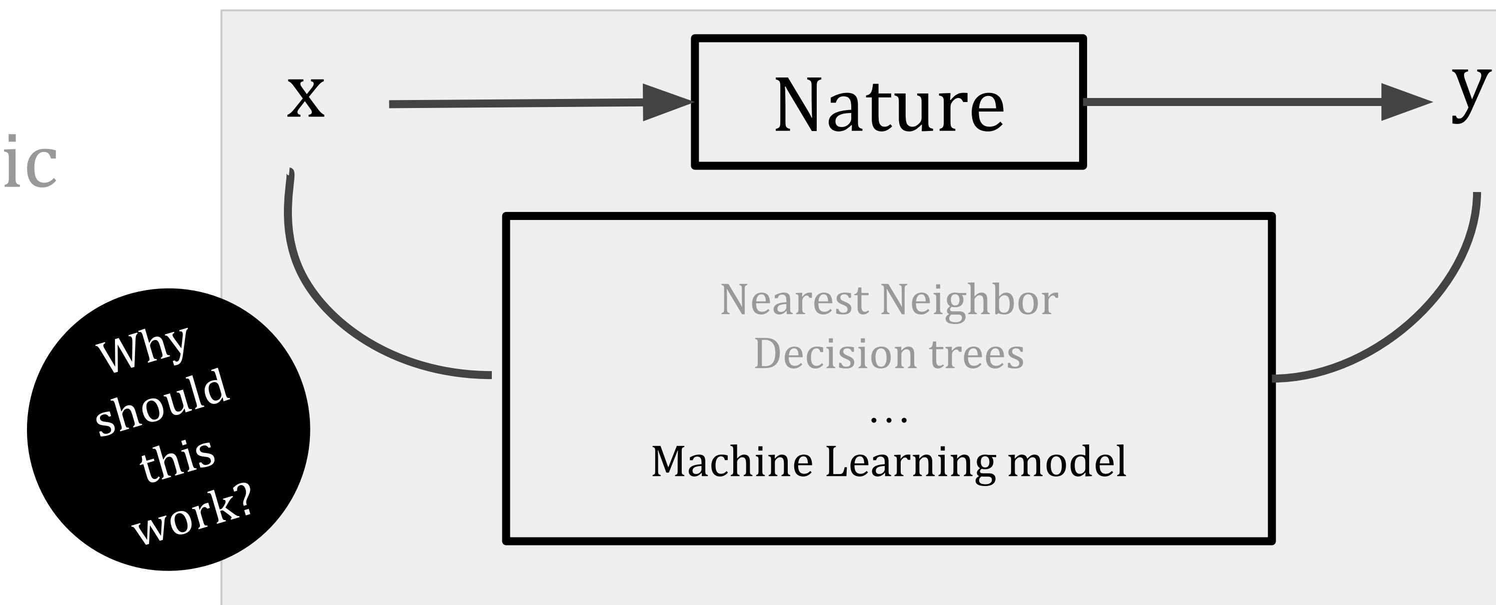
Breiman, L., Statistical Modeling: The Two Cultures, Stat. Sci, Volume 16 (2001)

Hypothesis:

Physical modeling:



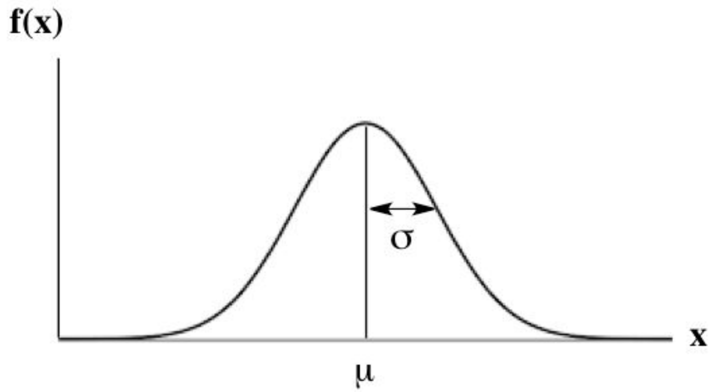
Algorithmic modeling:



Breiman, L., Statistical Modeling: The Two Cultures, Stat. Sci, Volume 16 (2001)

Representativeness

Probability distribution, *P*



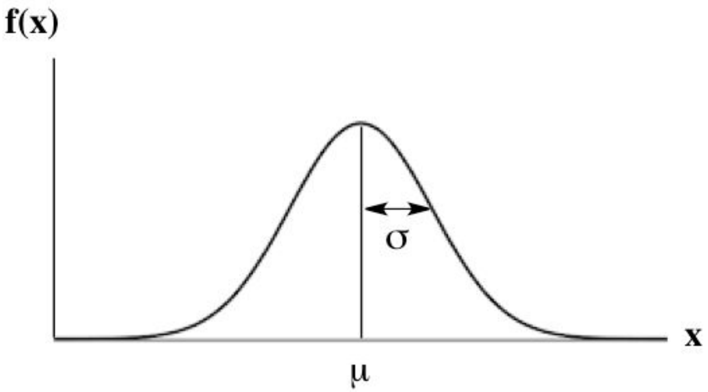
$$(\mu_P, \sigma_P)$$

Sample, S1

$$(\mu_{S_1}, \sigma_{S_1})$$

Representativeness

Probability distribution, *P*



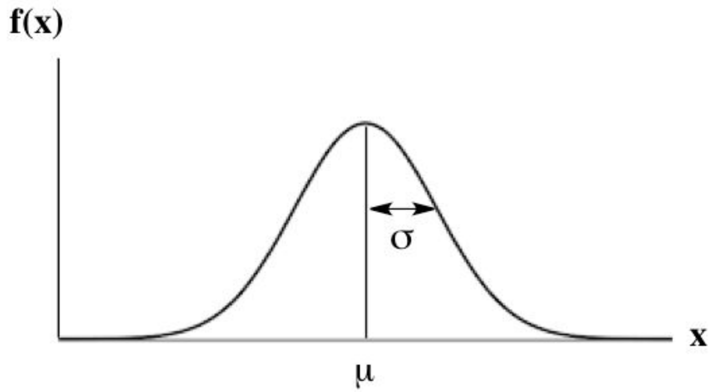
$$(\mu_P, \sigma_P)$$

Sample, S1

$$(\mu_{S_1}, \sigma_{S_1})$$

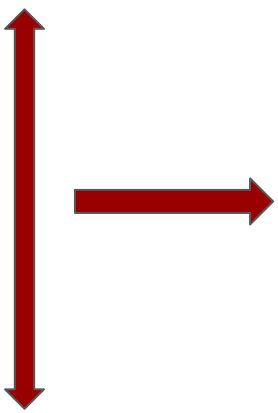
Representativeness

Probability distribution, *P*



Sample, S1

$$(\mu_P, \sigma_P)$$



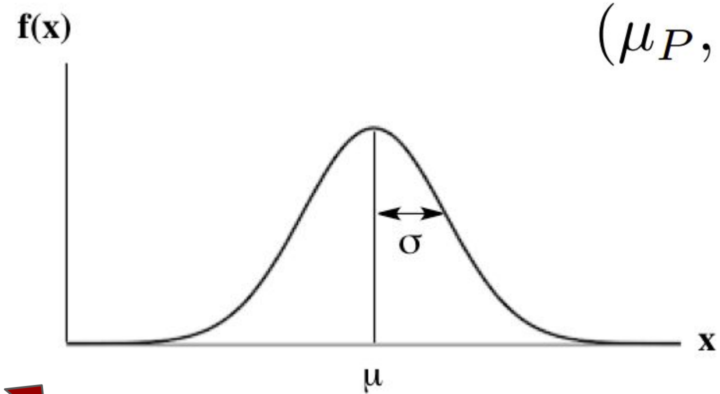
S_1 is *representative* of *P*

$$(\mu_{S_1}, \sigma_{S_1})$$

Representativeness

Probability distribution, *P*

(μ_P, σ_P)



Sample, S_1

$(\mu_{S_1}, \sigma_{S_1})$

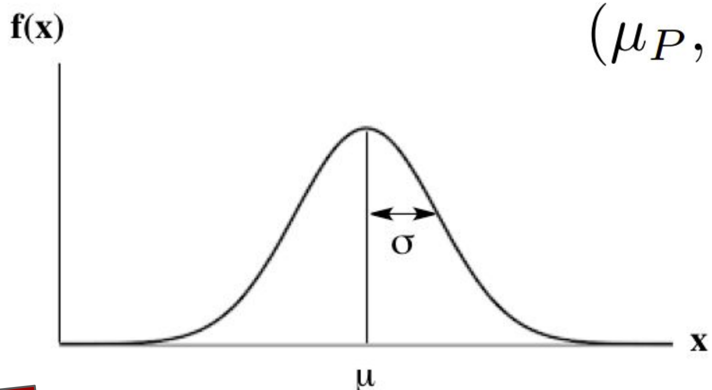
Sample, S_2

$(\mu_{S_2}, \sigma_{S_2})$

Representativeness

Probability distribution, *P*

(μ_P, σ_P)



Sample, S_1

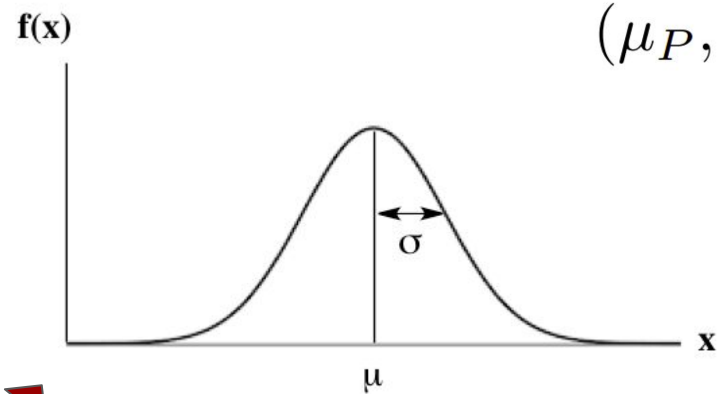
Sample, S_2

$(\mu_{S_1}, \sigma_{S_1})$

$(\mu_{S_2}, \sigma_{S_2})$

Representativeness

Probability distribution, *P*



(μ_P, σ_P)

This is why it works!

Training

$(\mu_{S_1}, \sigma_{S_1})$

Test

$(\mu_{S_2}, \sigma_{S_2})$

Representativeness

- A sample S1 is said to be representative of a probability distribution P if one can draw accurate conclusions about P from S1
- If two samples S1 and S2 are representative of P, S1 and S2 are representative in relation to each other

Representativeness

- A sample S1 is said to be representative of a probability distribution P if one can draw accurate conclusions about P from S1

- If two samples S1 and S2 are representative of P, S1 and S2 are representative in relation to each other

Question:

If a sample S_1 identically independently distributed (i.i.d.) from a distribution P, is this enough to guarantee that S_1 is representative of P?

Join at menti.com with code: 7907 6385

Model assumptions

χ: set of all features,

x = [softness, color]

Y: set of possible labels,

y = [tasty, not tasty]

D: data generation model,

$D \Longrightarrow P(\chi)$

True Labelling function: y = f(x)

S: training sample: $[x_i, y_i]$, $i \in training$

h_S learner: $y_{est;i} = h_S(x_i)$

$$h_S(x) = \begin{cases} y_i & \text{if } \exists i \in [m] \text{ s.t. } x_i = x \\ 0 & \text{otherwise.} \end{cases}$$

L: *loss*: $L(y_{true;i} - y_{est;i})$, $i \in training$

$$L_{\mathcal{D}}(h_S) = \frac{|\{x \in \mathcal{D} : h_S(x) \neq f(x)\}|}{m}$$

Hypothesis class ($\mathcal{H}$):

$$h : \mathcal{X} \longrightarrow \mathcal{Y}; \qquad h \in \mathcal{H}$$

$$\mathrm{ERM}_{\mathcal{H}}(S) \in \underset{h \in \mathcal{H}}{\operatorname{argmin}}\, L_S(h),$$

- $\mathcal{H}$ is finite, $N_{\mathcal{H}}$ = number of hypothesis
- The true labelling function is part of $\mathcal{H}$:

  $$f \in \mathcal{H}$$

- *S* is identically independently distributed (*i.i.d.*) from *D*

Not enough!

Bad samples and hypothesis

Bad samples and hypothesis

$\delta \;\longrightarrow$ probability of non-representative (bad) samples

Bad hypothesis and samples

$\delta \longrightarrow$ probability of non-representative (bad) samples

$1 - \delta \longrightarrow$ confidence parameter

Bad hypothesis and samples

δ ⟶ probability of non-representative (bad) samples

1 - δ ⟶ confidence parameter

ε ⟶ contamination. A failure will occur when $L_D(h_S) \geq \epsilon$

Good hypothesis: $\mathcal{H}_G := [h \in \mathcal{H} : L_S(h_S) = 0 \quad \& \quad L_D(h_S) < \epsilon]$

Bad hypothesis: $\mathcal{H}_B := [h \in \mathcal{H} : L_S(h_S) = 0 \quad \& \quad L_D(h_S) \geq \epsilon]$

Bad hypothesis and samples

δ $\longrightarrow$ probability of non-representative (bad) samples

1 - δ $\longrightarrow$ confidence parameter

ε $\longrightarrow$ contamination. A failure will occur when $L_D(h_S) \geq \epsilon$

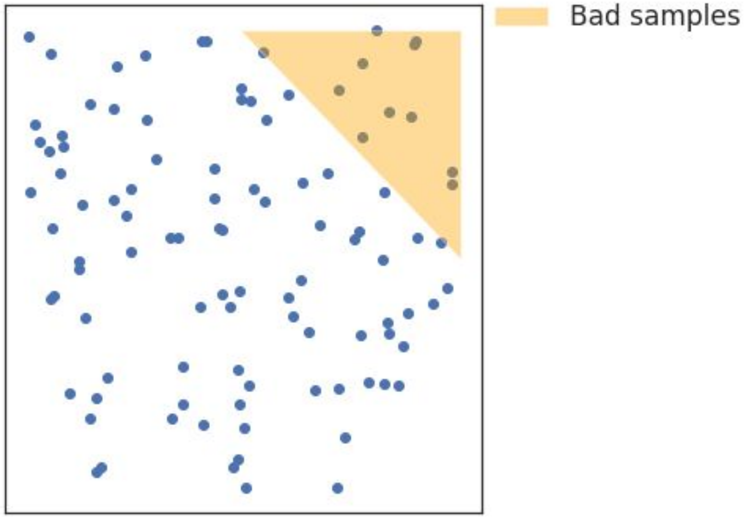
Good hypothesis: $\mathcal{H}_G := [h \in \mathcal{H} : L_S(h_S) = 0 \;\&\; L_D(h_S) < \epsilon]$

Bad hypothesis: $\mathcal{H}_B := [h \in \mathcal{H} : L_S(h_S) = 0 \;\&\; L_D(h_S) \geq \epsilon]$

Realizability assumption, $f \in \mathcal{H}$

Constructing misleading samples

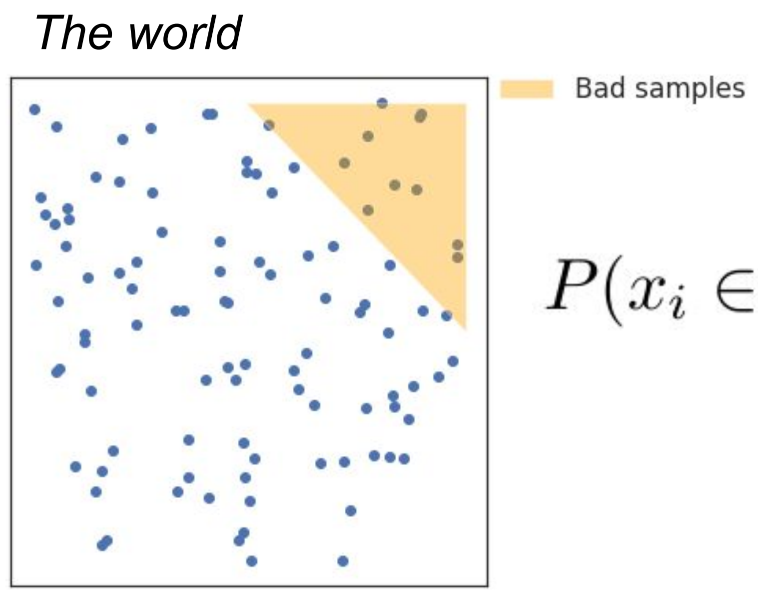
The world



For 1 element in the training sample

$$x_i \quad | \quad h(x_i) \quad = \quad y_i$$

Constructing misleading samples



For 1 element in the training sample

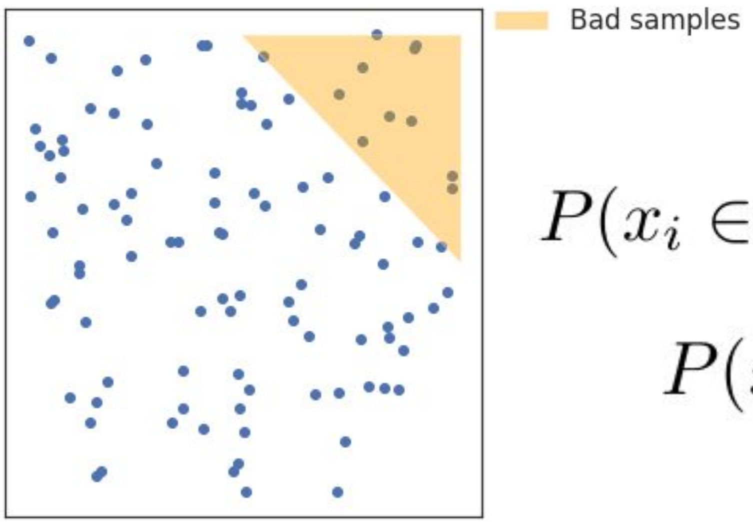
$$x_i \quad | \quad h(x_i) \quad = \quad y_i$$

$$P(x_i \in \mathcal{D} : h(x_i) \quad = \quad y_i) = 1 - L_{\mathcal{D},f}(h)$$

...

Constructing misleading samples

The world



For 1 element in the training sample

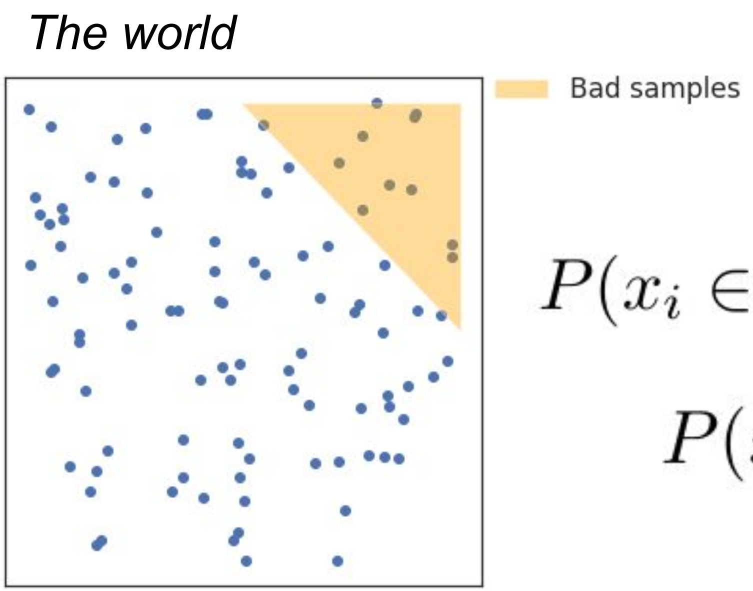
$$x_i \quad | \quad h(x_i) \quad = \quad y_i$$

$$P(x_i \in \mathcal{D} : h(x_i) \quad = \quad y_i) = 1 - L_{\mathcal{D},f}(h)$$

...

$$P(x_i \in \mathcal{D} : h(x_i) \quad = \quad y_i) \leq 1 - \epsilon$$

Constructing misleading samples



For 1 element in the training sample

$$x_i \quad | \quad h(x_i) \quad = \quad y_i$$

$$P(x_i \in \mathcal{D} : h(x_i) \quad = \quad y_i) = 1 - L_{\mathcal{D},f}(h)$$

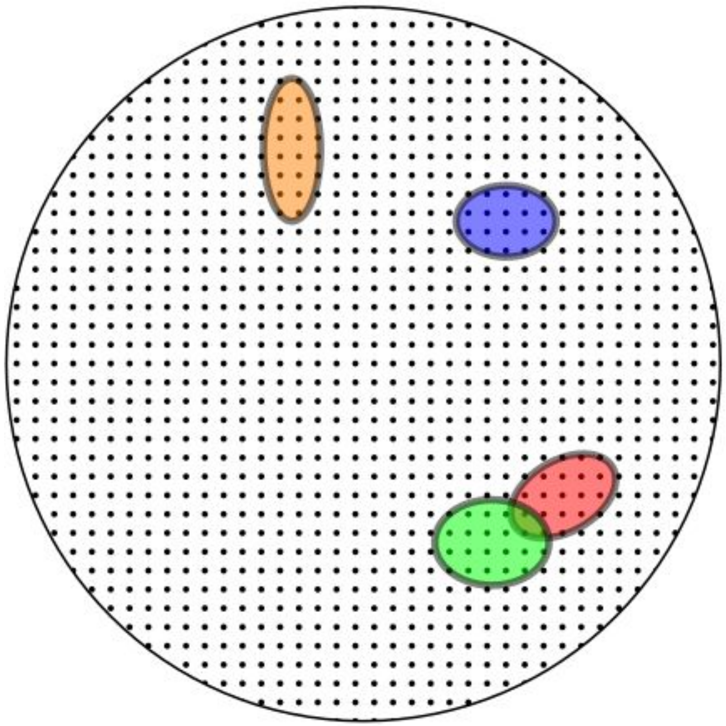
$$P(x_i \in \mathcal{D} : h(x_i) \quad = \quad y_i) \leq 1 - \epsilon$$

*For **m** elements in the training sample*

Since all elements in training are i.i.d.,

$$P(S_m : L_S(h) = 0) \quad \leq \quad \prod_{i=1}^{m} (1 - \epsilon) = (1 - \epsilon)^m$$

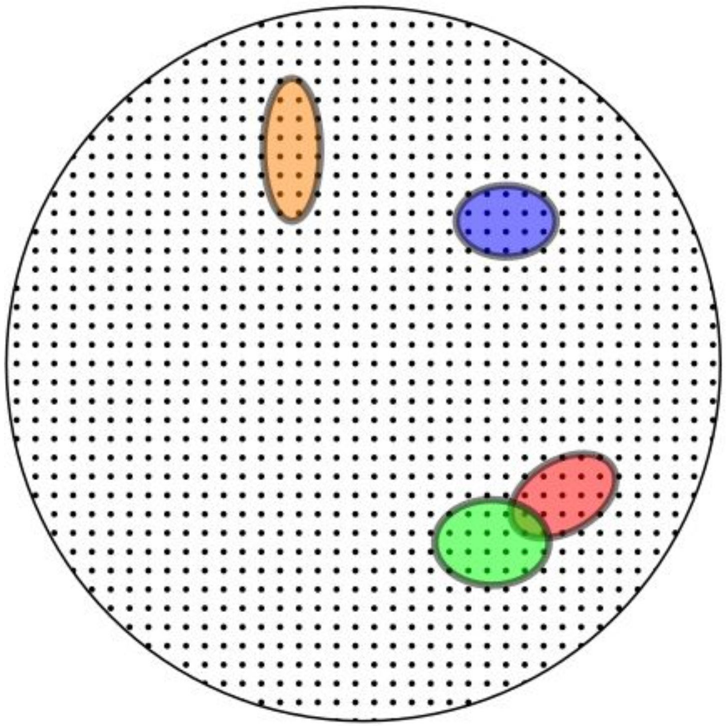
Considering bad hypothesis



For 1 hypothesis

$$P(S_m : L_S(h) = 0) \quad \leq \quad (1 - \epsilon)^m$$

Considering bad hypothesis



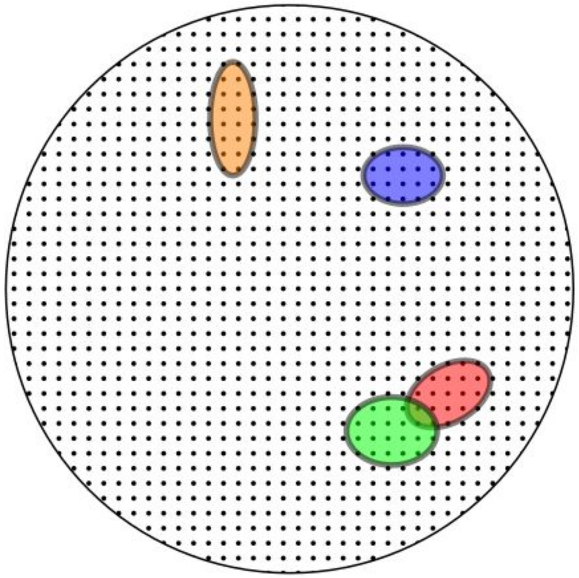
For 1 hypothesis

$$P(S_m : L_S(h) = 0) \leq (1-\epsilon)^m$$

The sum rule

$$P(A \cup B) \leq P(A) + P(B)$$

Considering bad hypothesis



For 1 hypothesis

$$P(S_m : L_S(h) = 0) \quad \leq \quad (1-\epsilon)^m$$

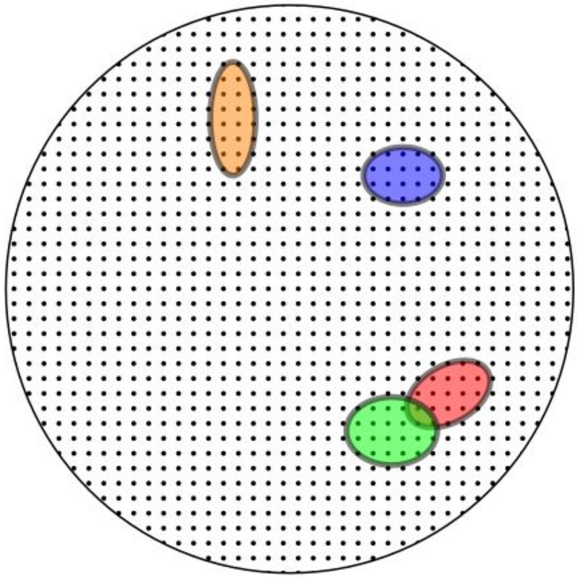
The sum rule

$$P(A \cup B) \quad \leq \quad P(A) + P(B)$$

For all bad hypothesis

$$\delta = P(L_S(h) = 0, \forall h \in \mathcal{H}_B) \quad \leq \quad \sum_{h \in \mathcal{H}_B} (1-\epsilon)^m$$

Considering bad hypothesis



For 1 hypothesis

$$P(S_m : L_S(h) = 0) \quad \leq \quad (1-\epsilon)^m$$

The sum rule

$$P(A \cup B) \quad \leq \quad P(A) + P(B)$$

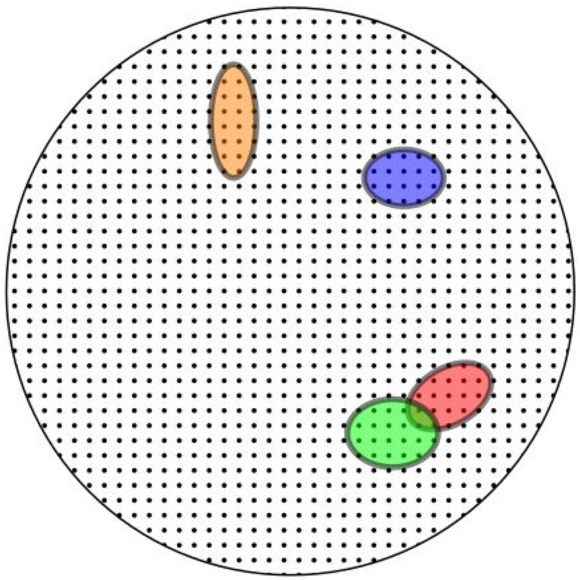
For all bad hypothesis

$$\delta = P(L_S(h) = 0, \forall h \in \mathcal{H}_B) \quad \leq \quad \sum_{h \in \mathcal{H}_B} (1-\epsilon)^m$$

using...

$$(1-x)^y \quad \leq \quad \exp(-xy)$$

Considering bad hypothesis



For 1 hypothesis

$$P(S_m : L_S(h) = 0) \quad \leq \quad (1-\epsilon)^m$$

The sum rule

$$P(A \cup B) \quad \leq \quad P(A) + P(B)$$

For all bad hypothesis

$$\delta = P(L_S(h) = 0, \forall h \in \mathcal{H}_B) \quad \leq \quad \sum_{h \in \mathcal{H}_B} (1-\epsilon)^m$$

using...

$$(1-x)^y \quad \leq \quad \exp(-xy)$$

$$\delta \leq N_{\mathcal{H}} \exp(-\epsilon m)$$

In summary …

$$\delta \leq N_{\mathcal{H}} \exp(-\epsilon m)$$

L. Valiant. A theory of the learnable. Communications of the ACM, 27, 1984.

In summary …

$$\delta \leq N_{\mathcal{H}} \exp(-\epsilon m)$$

If,

every *h* from ERM,

$$m_{\mathcal{H}}(\epsilon, \delta) \geq \frac{\ln(N_{\mathcal{H}}/\delta)}{\epsilon} \Longrightarrow L_{(\mathcal{D},f)}(h_S) \leq \epsilon.$$

L. Valiant. A theory of the learnable. Communications of the ACM, 27, 1984.

PAC learning model

$$\delta \leq N_{\mathcal{H}} \exp(-\epsilon m)$$

If,

$$m_{\mathcal{H}}(\epsilon, \delta) \geq \frac{\ln(N_{\mathcal{H}}/\delta)}{\epsilon}$$

→

every *h* from ERM,

$$L_{(\mathcal{D},f)}(h_S) \leq \epsilon.$$

L. Valiant. A theory of the learnable. Communications of the ACM, 27, 1984.

PAC learning model

$$\delta \leq N_{\mathcal{H}} \exp(-\epsilon m)$$

Probably $\longrightarrow$ with confidence 1 - δ over *m* samples
Approximately $\longrightarrow$ within a contamination level ≤ ε
Correct

If, every *h* from ERM,

$$m_{\mathcal{H}}(\epsilon, \delta) \geq \frac{\ln(N_{\mathcal{H}}/\delta)}{\epsilon} \Longrightarrow L_{(\mathcal{D},f)}(h_S) \leq \epsilon.$$

L. Valiant. A theory of the learnable. Communications of the ACM, 27, 1984.

Remember what is behind this!!

PAC Assumptions

χ: set of all features,

x = [softness, color]

Y: set of possible labels,

y = [tasty, not tasty]

D: data generation model,

$D \Longrightarrow P(\chi)$

True Labelling function: y = f(x)

S: training sample: $[x_i, y_i]$, $i \in training$

m: number of objects for training

h_S learner: $y_{est;i} = h_S(x_i)$

$$h_S(x) = \begin{cases} y_i & \text{if } \exists i \in [m] \text{ s.t. } x_i = x \\ 0 & \text{otherwise.} \end{cases}$$

L: *loss*: $L(y_{true;i} - y_{est;i})$, $i \in training$

$$L_{\mathcal{D}}(h_S) = \frac{|\{x \in \mathcal{D} : h_S(x) \neq f(x)\}|}{m}$$

Hypothesis class ($\mathcal{H}$):

$$h : \mathcal{X} \longrightarrow \mathcal{Y}; \qquad h \in \mathcal{H}$$

$$\text{ERM}_{\mathcal{H}}(S) \in \underset{h \in \mathcal{H}}{\text{argmin}}\, L_S(h),$$

- $\mathcal{H}$ is finite, $N_{\mathcal{H}}$ = number of hypothesis
- The true labelling function is part of $\mathcal{H}$:

  $$f \in \mathcal{H}$$

- *S* is identically independently distributed (*i.i.d.*) from D
- Representativeness

Papaya tasting

χ: set of $x \in$ *[softness, color]*

Y: set $y =$ *[tasty, not tasty]*

D: data generation model: $D \Longrightarrow P(\chi)$

Papaya tasting

χ: set of $x \in$ *[softness, color]*

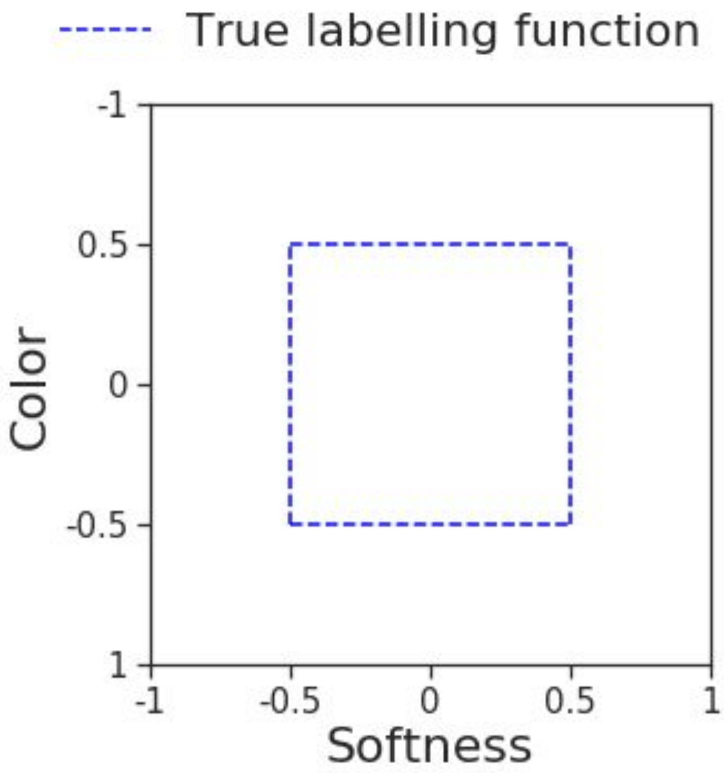
Y: set *y = [tasty, not tasty]*

D: data generation model: $D \Longrightarrow P(\chi)$

True Labelling function:

y = tasty if softness $\in$ *[-0.5, 0.5] and*

color $\in$ *[-0.5, 0.5]*



Papaya tasting

χ: set of $x \in$ *[softness, color]*

Y: set *y = [tasty, not tasty]*

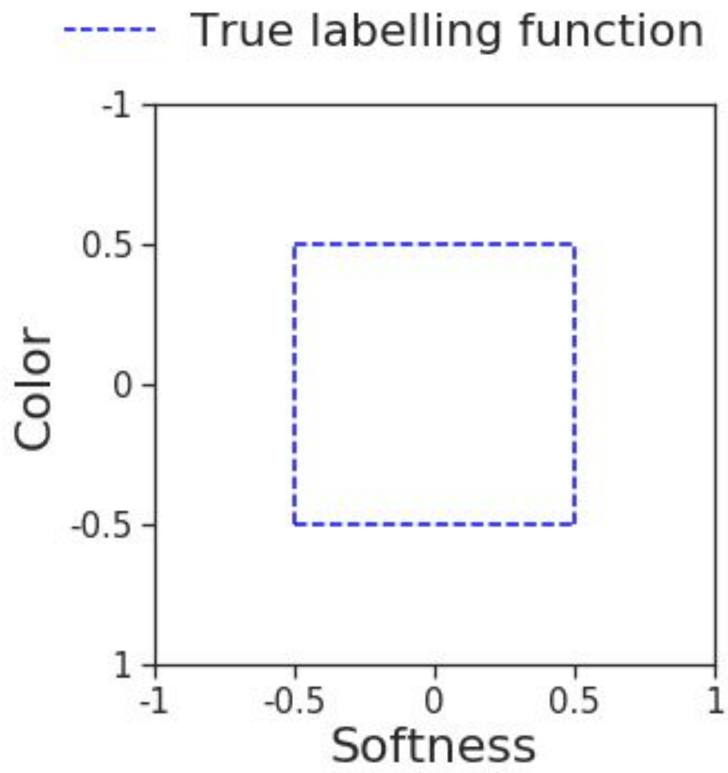
D: data generation model: $D \Longrightarrow P(\chi)$

True Labelling function:

y = tasty if softness $\in$ *[-0.5, 0.5] and*
color $\in$ *[-0.5, 0.5]*

S: training sample: $[x_i, y_i]$, $i \in training$

m: number of objects for training



Papaya tasting

χ: set of $x \in$ *[softness, color]*

Y: set *y = [tasty, not tasty]*

D: data generation model: $D \Rightarrow P(\chi)$

True Labelling function:

y = tasty if softness $\in$ [-0.5, 0.5] and

color $\in$ [-0.5, 0.5]

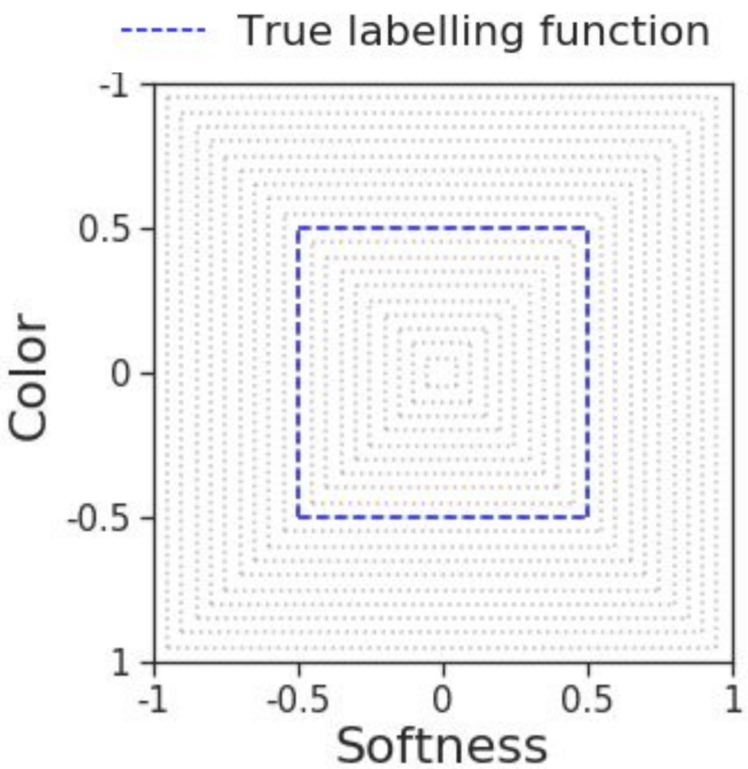
S: training sample: $[x_i, y_i]$, $i \in$ *training*

m: number of objects for training

$\mathcal{H}$: *hypothesis class:*

axis aligned squares in steps of 0.05

$N_H = 20$



Papaya tasting

χ: set of $x \in$ *[softness, color]*

Y: set *y = [tasty, not tasty]*

D: data generation model: $D \Rightarrow P(\chi)$

True Labelling function:

y = tasty if softness $\in$ [-0.5, 0.5] and color $\in$ [-0.5, 0.5]

S: training sample: $[x_i, y_i]$, $i \in$ *training*

m: number of objects for training

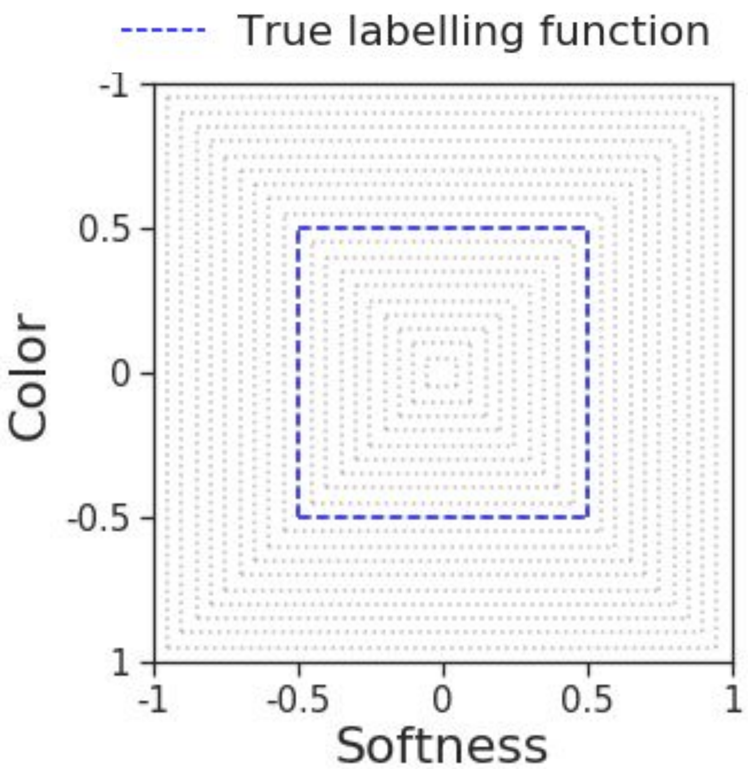
$\mathcal{H}$: *hypothesis class:*

axis aligned squares in steps of 0.05

$N_H = 20$

L: loss: $L(y_{true;i} - y_{est;i})$, $i \in$ *training*

$$L_{\mathcal{D}}(h_S) = \frac{|\{x \in \mathcal{D} : h_S(x) \neq f(x)\}|}{m}$$



Question:

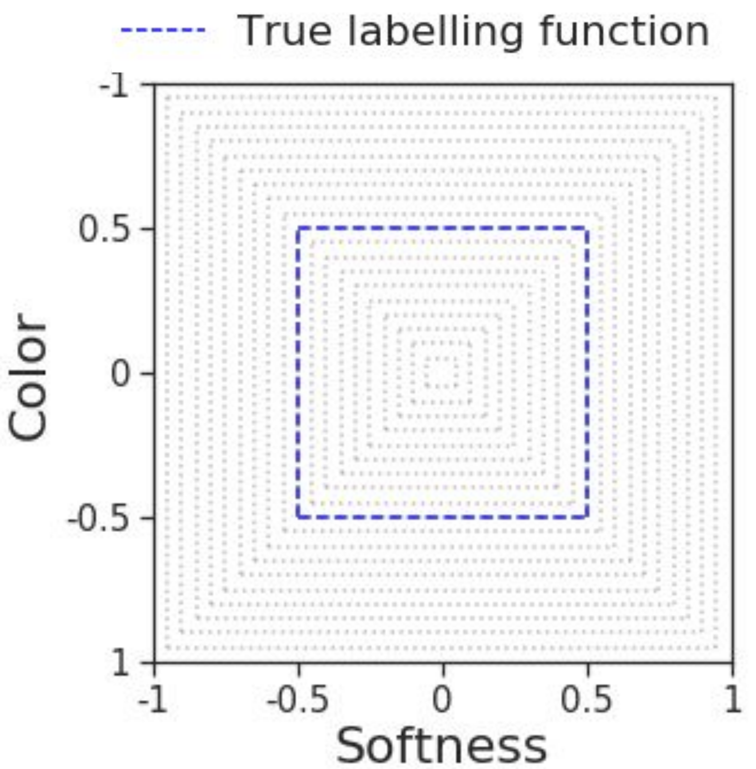
Data model: uniform distribution [-1,1] in both axis

$1 - \delta = 0.95$ ← confidence

$\varepsilon = 0.05$ ← contamination

$N_H = 20$ ← number of possible squares

m = ??



Join at menti.com with code: 7907 6385

What would you guess is the number of examples necessary for training?

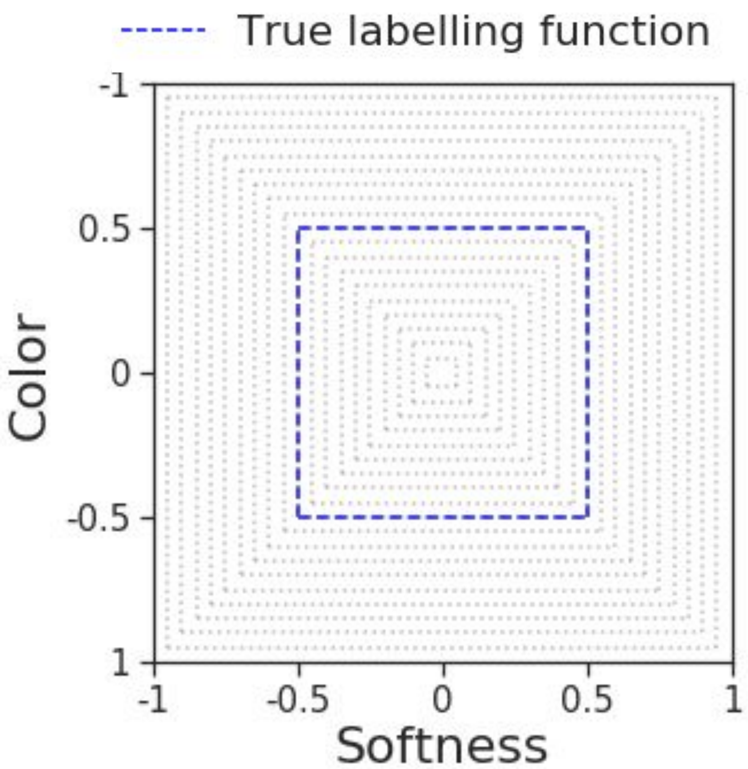
Question:

Data model: uniform distribution [-1,1] in both axis

$1 - \delta = 0.95 \leftarrow$ confidence

$\varepsilon = 0.05 \leftarrow$ contamination

$N_H = 20 \leftarrow$ number of possible squares

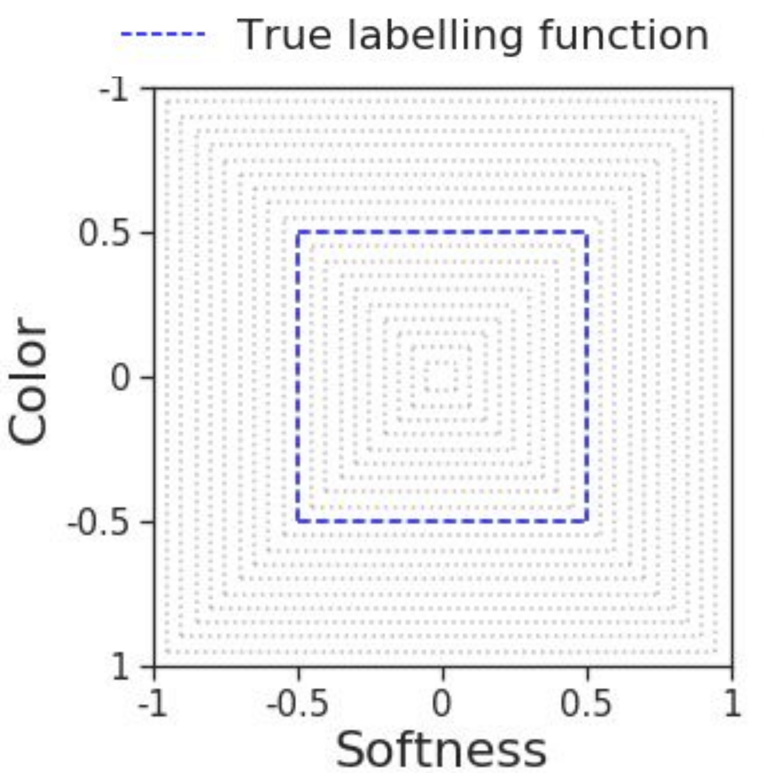


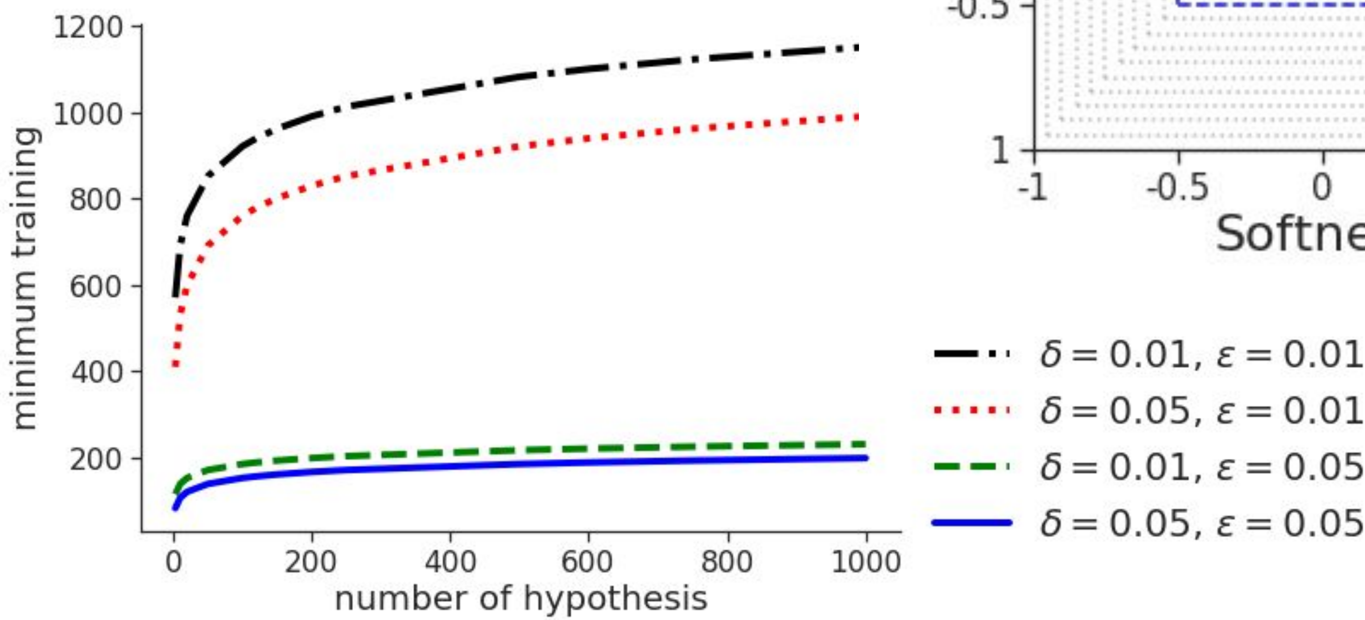
m ~ 120

Question:

Data model: uniform distribution [-1,1] in both axis

1 - δ = 0.95 ← confidence
ε = 0.05 ← contamination
N_H = 20 ← number of possible squares





Agnostic PAC learning

χ: set of all features,

x = [softness, color]

Y: set of possible labels,

y = [tasty, not tasty]

D: data generation model,

$D \Longrightarrow P(\chi, Y)$

True Labelling function: y = f([x,y])

S: training sample: $[x_i, y_i]$, $i \in$ *training*

m: number of objects for training

h_S learner: $y_{est;i} = h_S(x_i, y_i)$

L: loss

$$L_{\mathcal{D}}(h) \stackrel{\text{def}}{=} \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}} (h(x) - y)^2$$

Hypothesis class:

$$h : \mathcal{X} \longrightarrow \mathcal{Y}; \qquad h \in \mathcal{H}$$

$$\text{ERM}_{\mathcal{H}}(S) \in \underset{h \in \mathcal{H}}{\text{argmin}}\, L_S(h),$$

- $m \longrightarrow$ number of objects in training
- $\mathcal{H}$ is finite, N_H = number of hypothesis
- The true labelling function may not be part of $\mathcal{H}$:

$$f \notin \mathcal{H}$$

$$L_{\mathcal{D}}(h) \leq \min_{h' \in \mathcal{H}} L_{\mathcal{D}}(h') + \epsilon,$$

Representativeness

in machine learning

Shalev-Shwartz, S. and Ben-David, S., Understanding Machine Learning - from theory to algorithms, 2014, Cambridge University Press

Important remark!

Representativeness

in machine learning

or

Uniform Convergence

$$\forall h \in \mathcal{H}, \quad |L_S(h) - L_{\mathcal{D}}(h)| \leq \epsilon$$

Shalev-Shwartz, S. and Ben-David, S., Understanding Machine Learning - from theory to algorithms, 2014, Cambridge University Press

Representativeness

in machine learning

or

Uniform Convergence

$$\forall h \in \mathcal{H}, \quad |L_S(h) - L_{\mathcal{D}}(h)| \leq \epsilon$$

It can be shown that, if $\mathcal{H}$ has uniform convergence, $\text{ERM}_{\mathcal{H}}$ is a successful agnostic PAC learner of $\mathcal{H}$.

Shalev-Shwartz, S. and Ben-David, S., Understanding Machine Learning - from theory to algorithms, 2014, Cambridge University Press

Can machine learning solve my problem?

Can machine learning solve my problem?

- If your data satisfy all the necessary conditions;

Can machine learning solve my problem?

- If your data satisfy all the necessary conditions;
- If you have enough training sample to fulfill your expectations;

Can machine learning solve my problem?

- If your data satisfy all the necessary conditions;
- If you have enough training sample to fulfill your expectations;
- If the set of your hypothesis class + loss function + training data has uniform convergence (representativeness)

Can machine learning solve my problem?

- If your data satisfy all the necessary conditions;
- If you have enough training sample to fulfill your expectations;
- If the set of your hypothesis class + loss function + training data has uniform convergence (representativeness)

Then.. probably (1-δ), approximately (ε) : yes

What about practical situations?

*If you are using a classical learner whose class under your training sample and loss function are representative (has uniform convergence), you are probably getting reasonable results… **but not all the time**!*

Many of these requirements are difficult to fulfill, e.g.

What about practical situations?

*If you are using a classical learner whose class under your training sample and loss function are representative (has uniform convergence), you are probably getting reasonable results… **but not all the time**!*

So why does it seem to work in everything around us?

Best guess: *we do not know how to model real data...*

https://www.youtube.com/watch?v=M2BJC0OyI-w

There is plenty room for improvement!

Progress will only be possible through ***interdisciplinary*** *collaboration!*

There is plenty room for improvement!

Progress will only be possible through ***interdisciplinary*** *collaboration!*

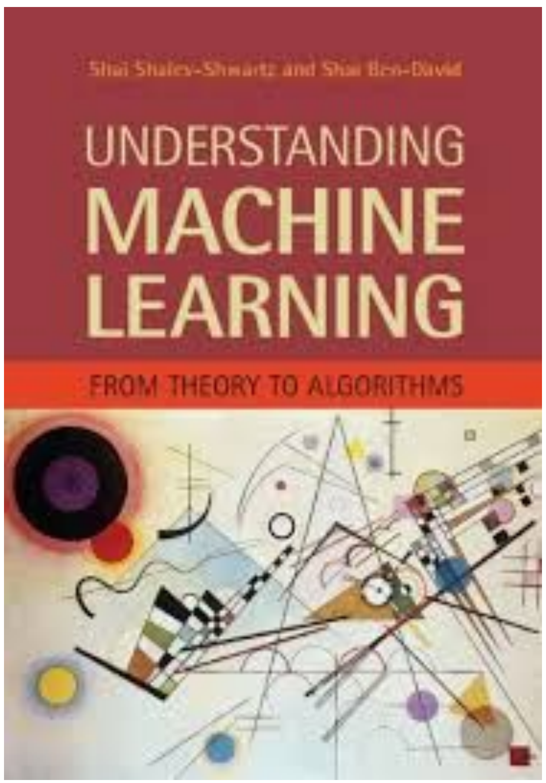
Machine learning is a wonderful field of research, which has already shown its potential in many fields! We should definitely take advantage of its results .. however ...

There is plenty room for improvement!

Progress will only be possible through ***interdisciplinary*** *collaboration!*

Machine learning is a wonderful field of research, which has already shown its potential in many fields! We should definitely take advantage of its results .. however ...

This talk is a rough summary of chapters 1-4:



Free download - with agreement from the editor:

https://www.cse.huji.ac.il/~shais/UnderstandingMachineLearning/index.html

23 lectures of 1.5 hours each on youtube:

https://www.youtube.com/playlist?list=PLPW2keNyw-usgvmR7FTQ3ZRjfLs5jT4BO

Enjoy!

THANK YOU