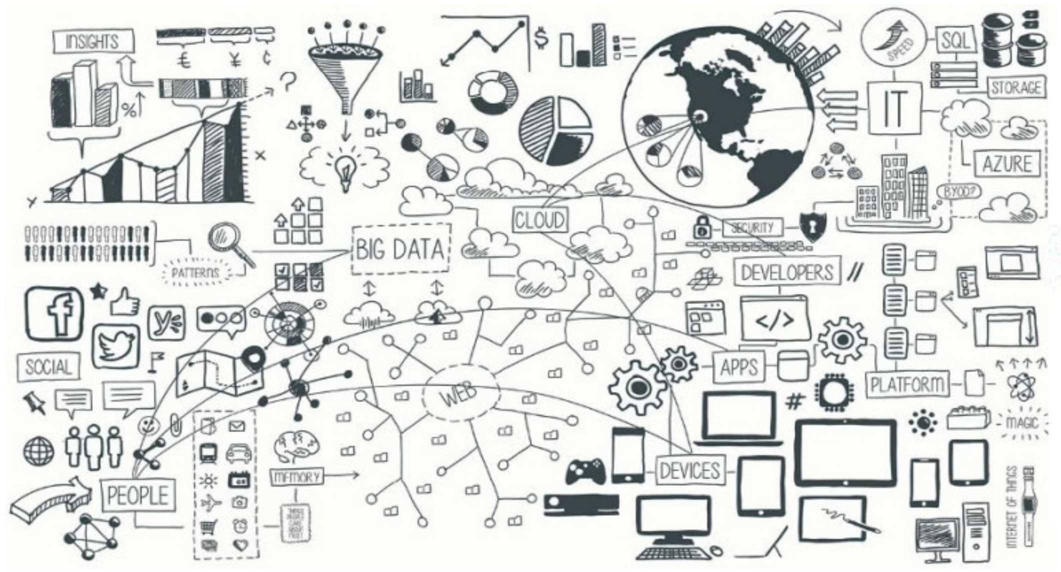
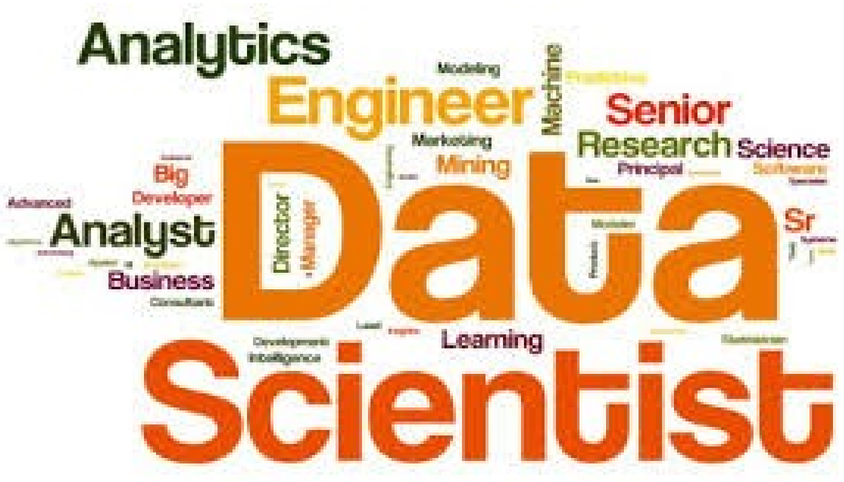
Data Mining (Minería de Datos)

Evaluación, sobreajuste y validación cruzada (cross-validation)





Sixto Herrera

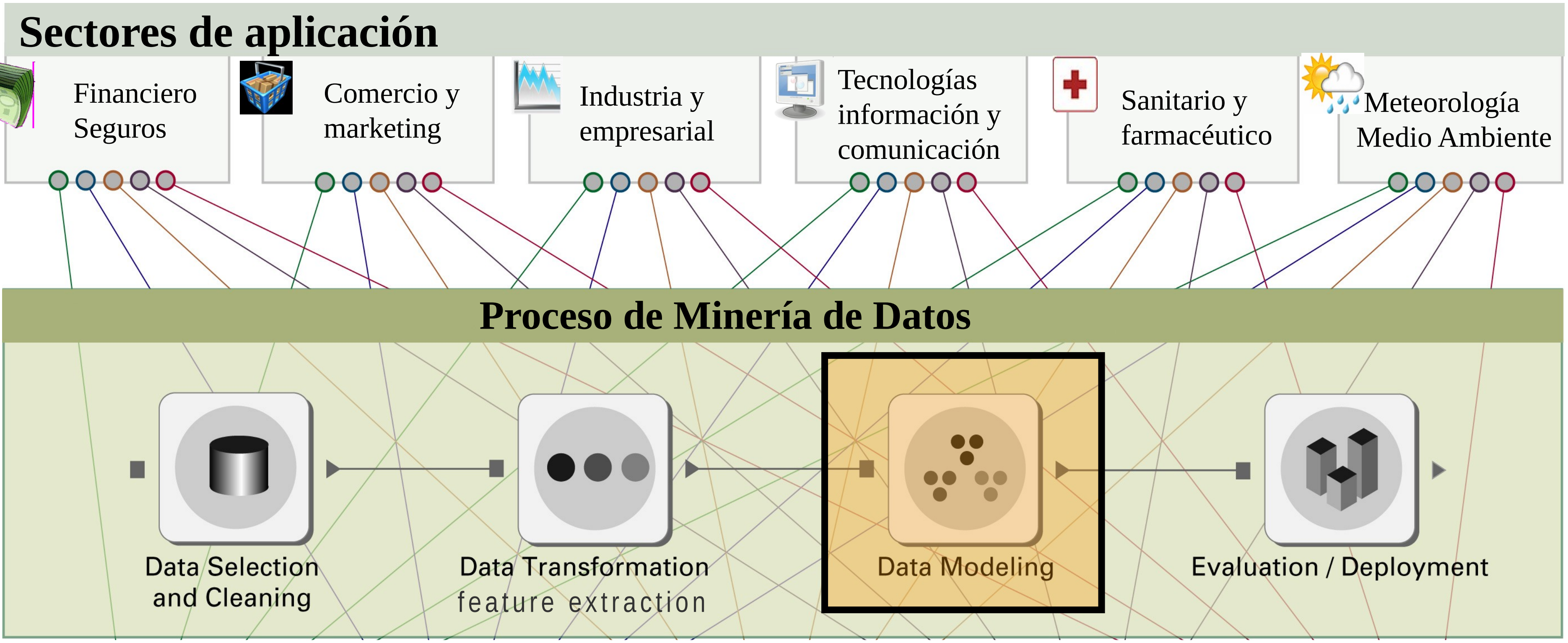
Rodrigo G. Manzanas

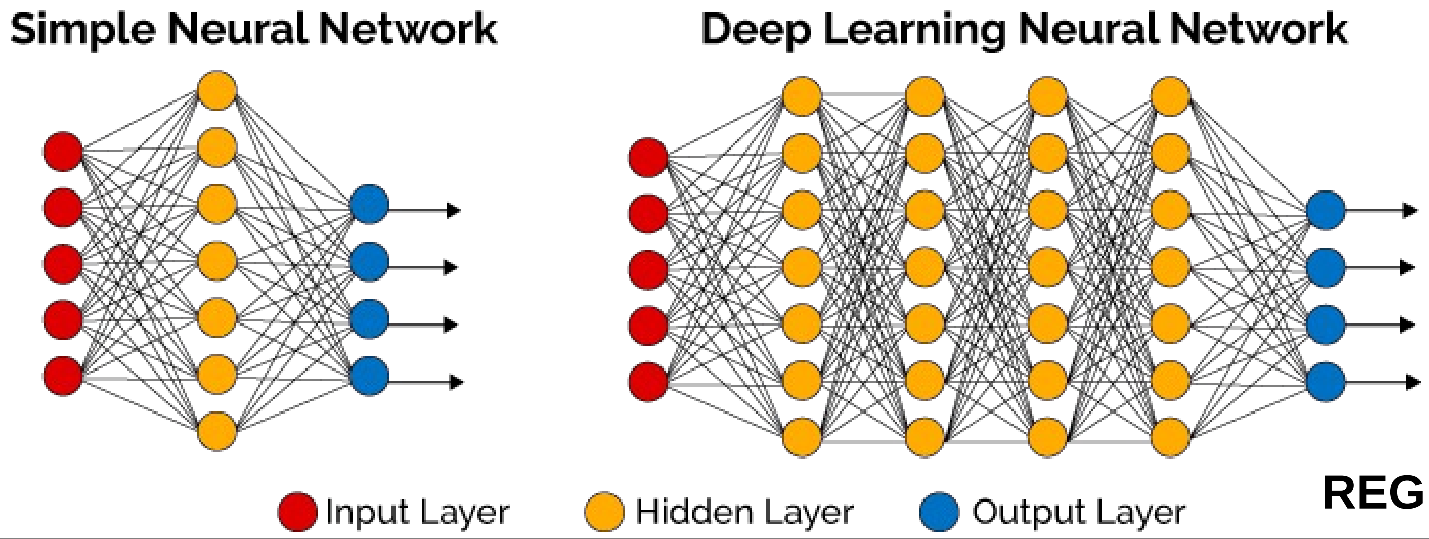
Grupo de Meteorología

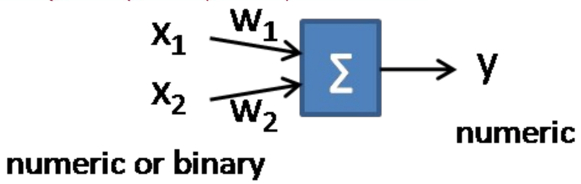
Univ. de Cantabria – CSIC
MACC / IFCA

Nov	2	Presentación, introducción y perspectiva histórica
	4	Paradigmas, problemas canonicos y data challenges
	9	Reglas de asociación
	11	Practica: Reglas de asociación
	16	**Evaluación, sobrejuste y crossvalidacion**
	18	**Practica: Crossvalidacion**
	23	Árboles de clasificacion y decision
	25	Practica: Árboles de clasificación
		T01. Datos discretos
	30	**Técnicas de vecinos cercano (k-NN)**
Dic	2	Práctica: Vecinos cercanos
	9	Comparación de Técnicas de Clasificación.
	14	Reducción de dimensión no lineal
	16	Reducción de dimensión no lineal
		T02. Clasificación
	17	Árboles de clasificación y regresion (CART)
	20	Práctica: Árboles de clasificación y regresion (CART)
	21	Practica: El paquete CARET
		T03. Prediccion
Ene	11	Ensembles: Bagging and Boosting
	13	Random Forests y **Gradient boosting**
	14	**Técnicas de agrupamiento**
	20	**Técnicas de agrupamiento**
	24	Predicción Condicionada
	26	Sesión de refuerzo/repaso.
Ene	27	Examen

NOTA: Las líneas de código de R en esta presentación se muestran sobre un fondo gris.







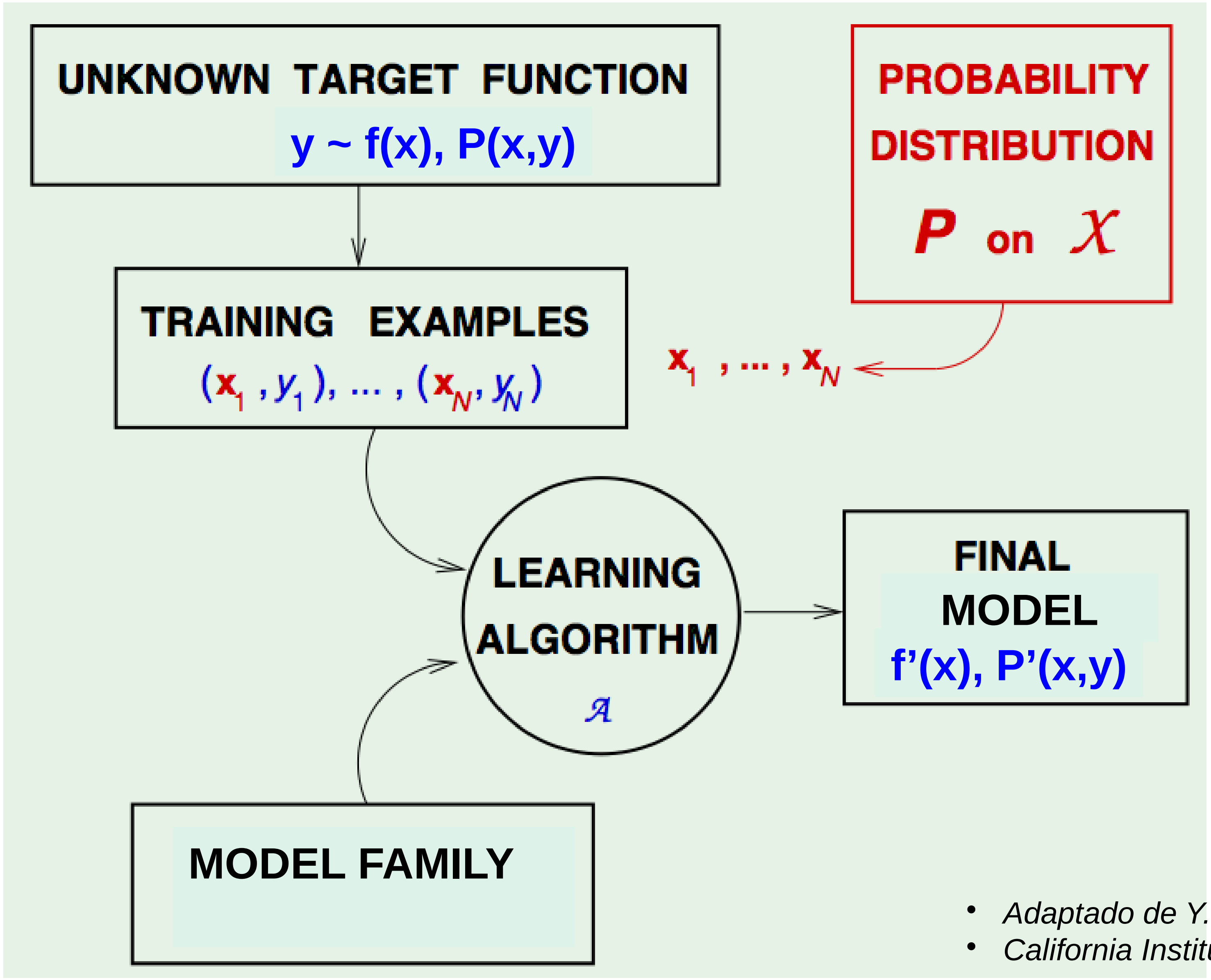
$$y = w_0 + w_1x_1 + w_2x_2$$

$$y = f(X,W) = X^T.W$$

REGRESSION $W = \begin{pmatrix} w_0 \\ w_1 \\ w_2 \end{pmatrix}$ $X = \begin{pmatrix} 1 \\ x_1 \\ x_2 \end{pmatrix}$

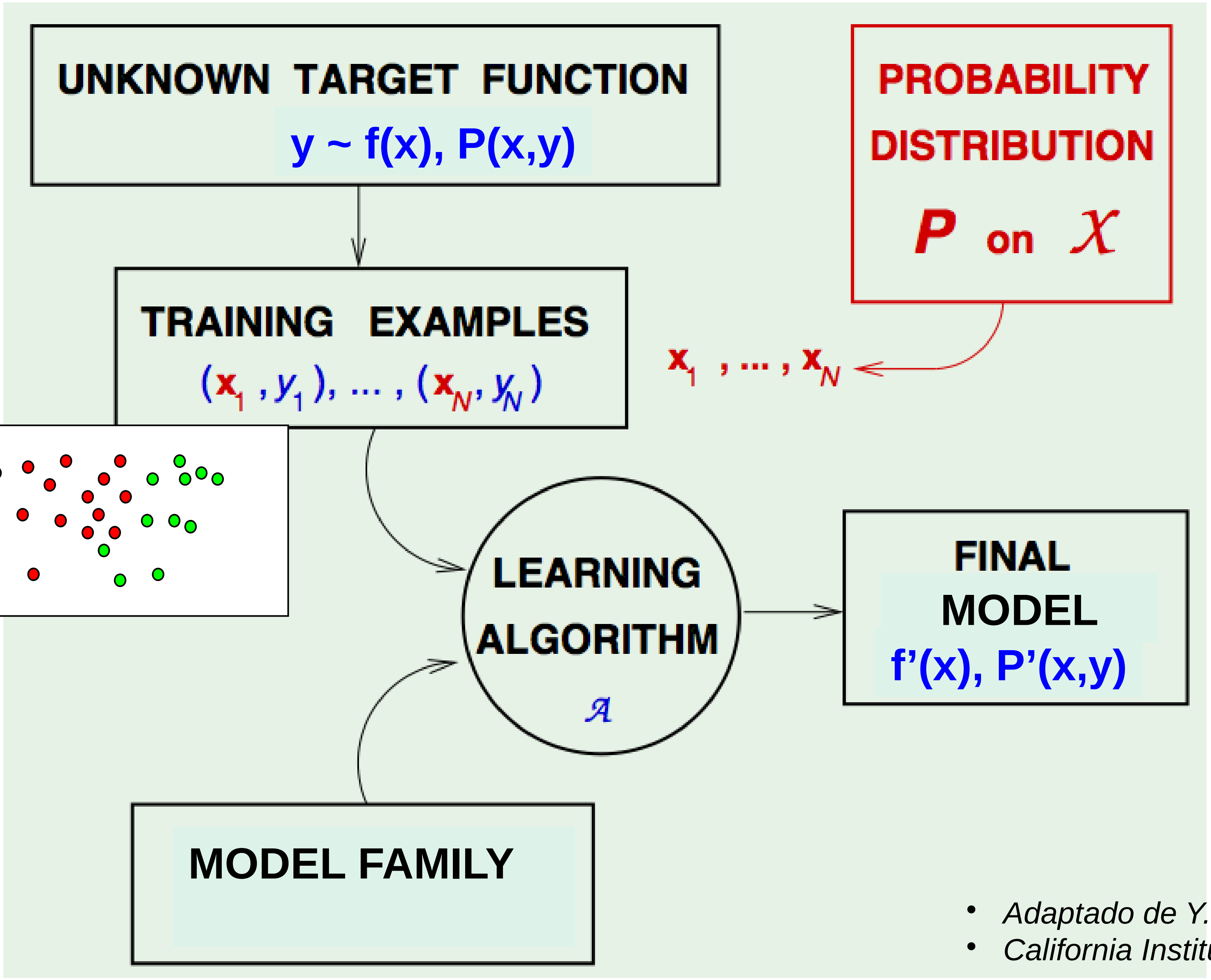
Cross-Validation

Learning is the automatic process of building (adjusting) a model from a data set which is representative from the full population.



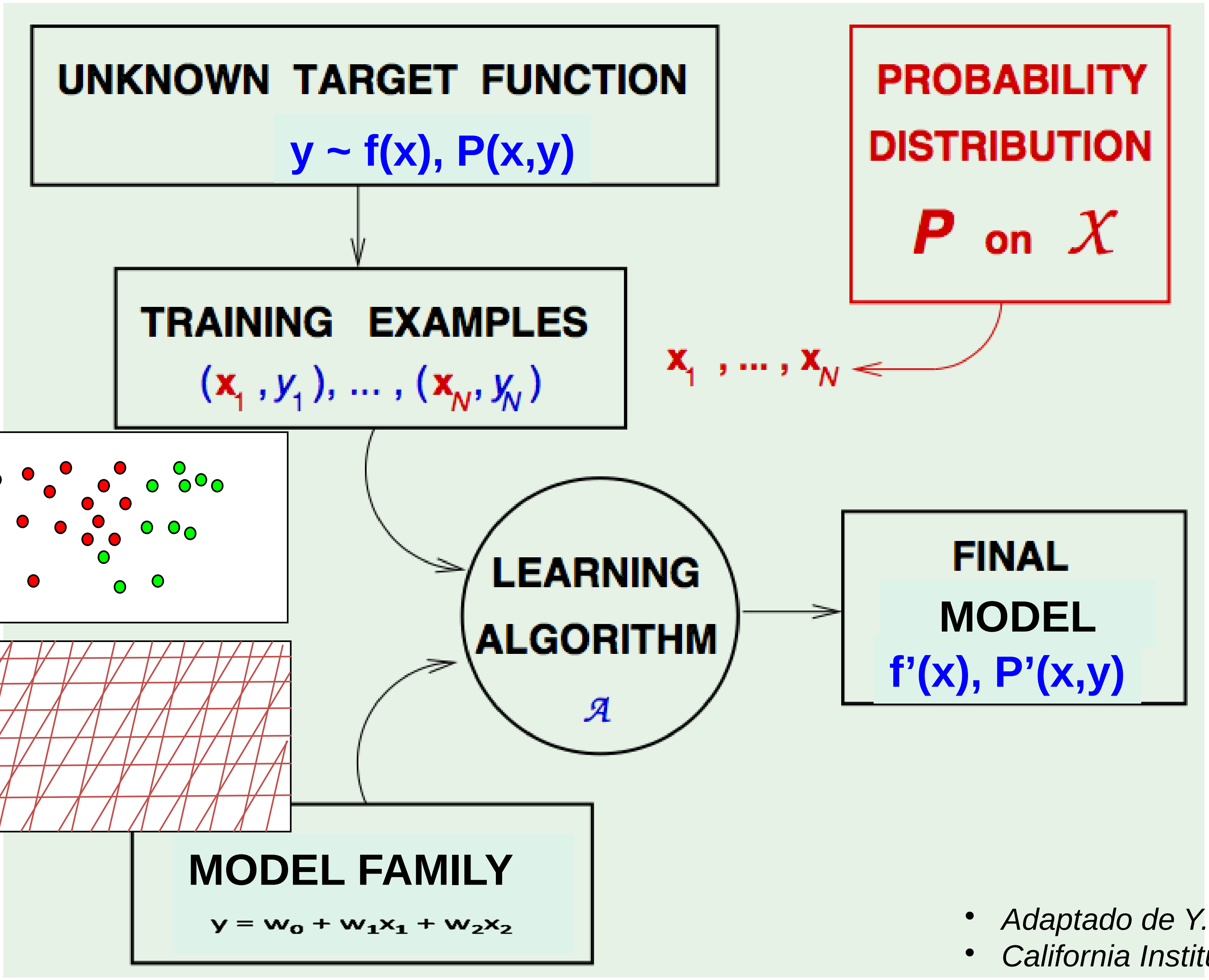
- *Adaptado de Y.S. Abu-Mostafa*
- *California Institute of Technology*

Learning is the automatic process of building (adjusting) a model from a data set which is representative from the full population.



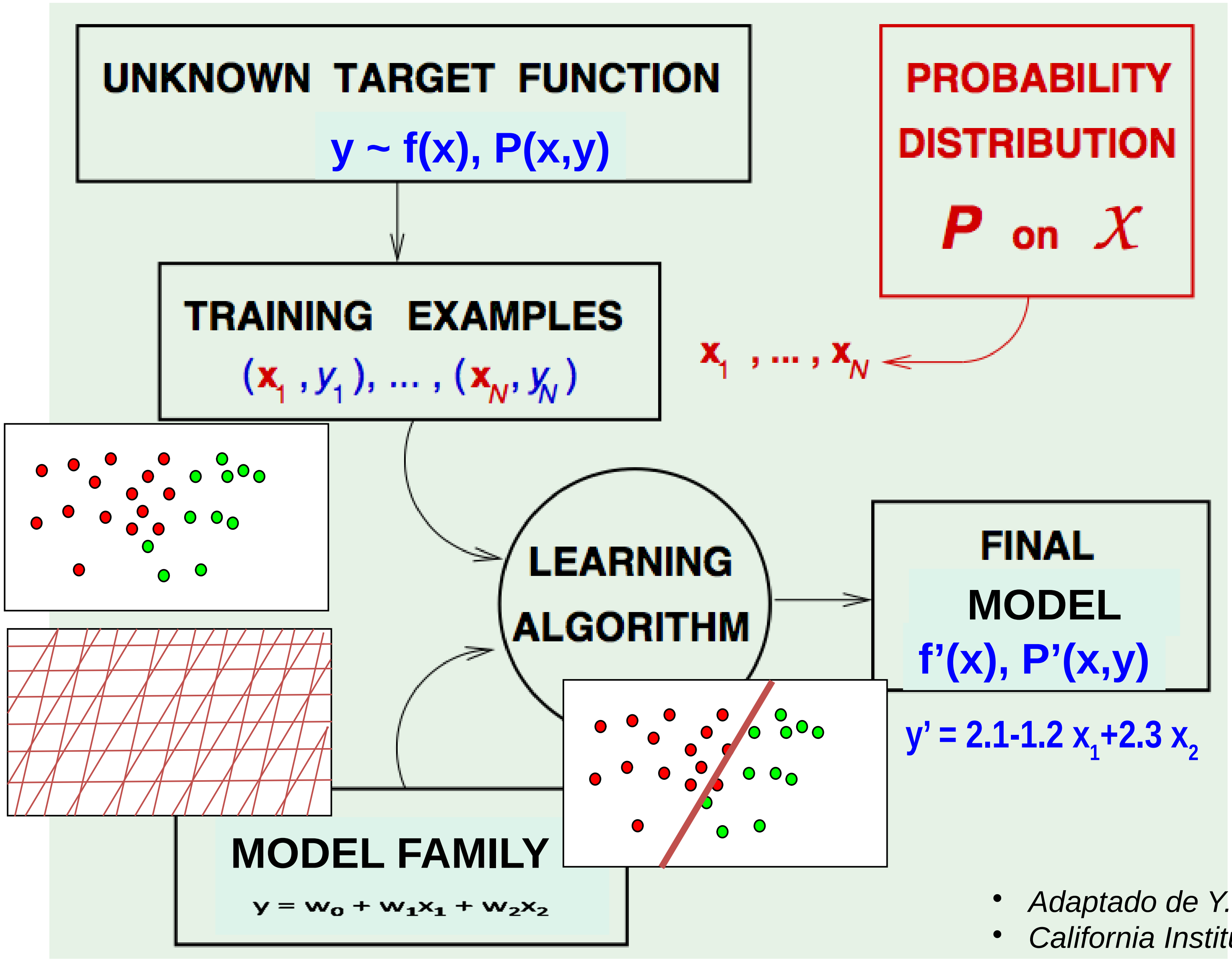
- *Adaptado de Y.S. Abu-Mostafa*
- *California Institute of Technology*

Learning is the automatic process of building (adjusting) a model from a data set which is representative from the full population.



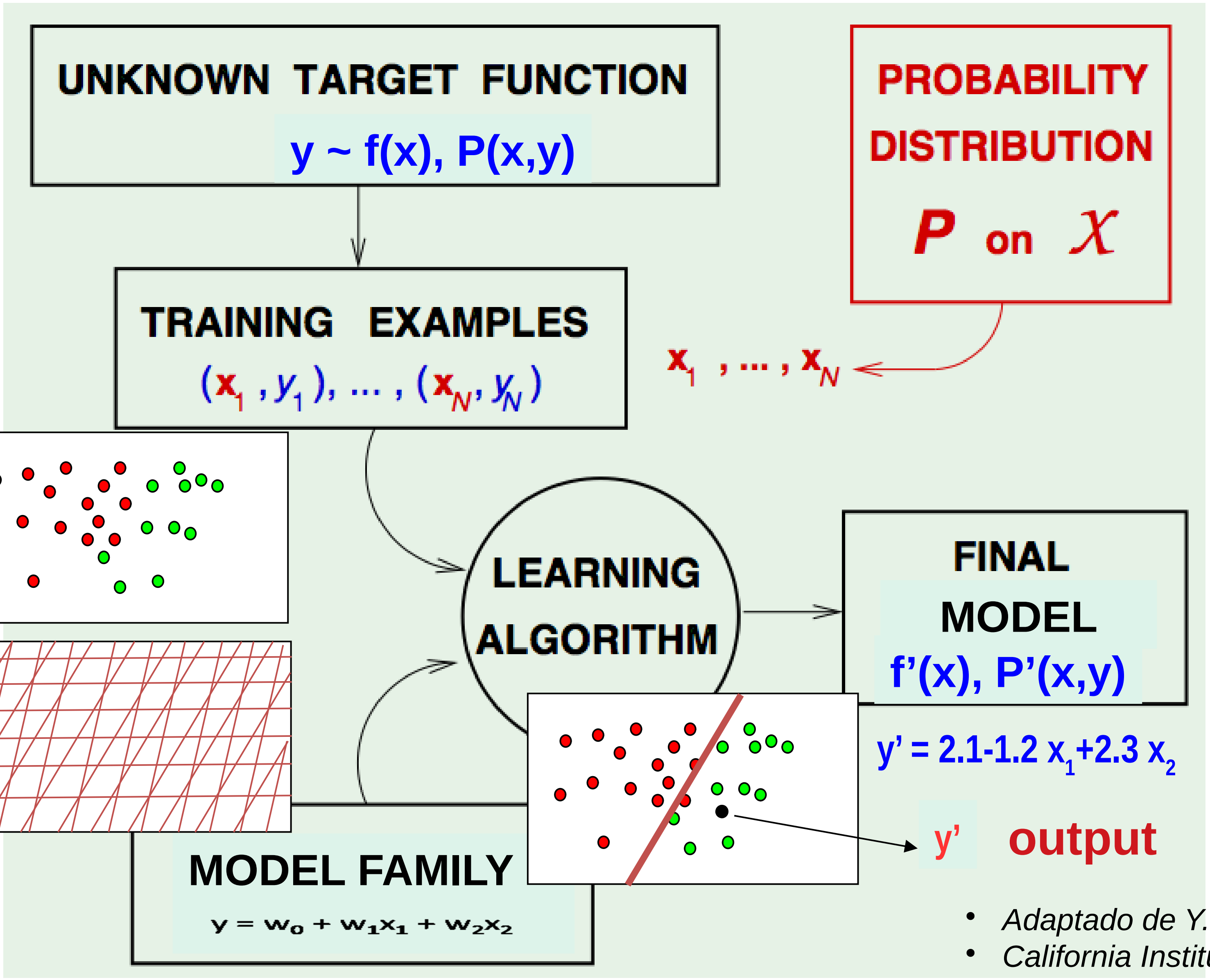
- *Adaptado de Y.S. Abu-Mostafa*
- *California Institute of Technology*

Learning is the automatic process of building (adjusting) a model from a data set which is representative from the full population.



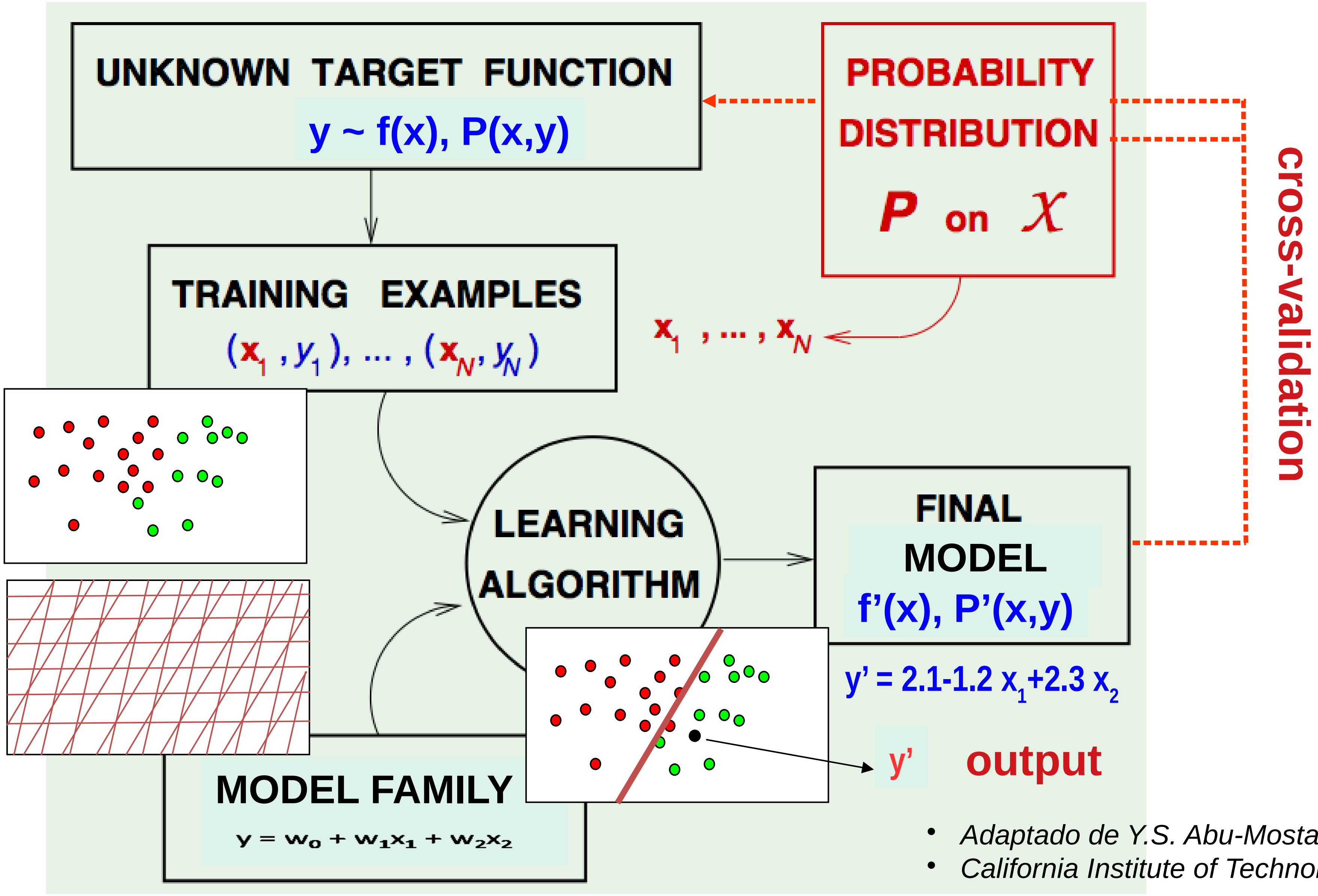
- *Adaptado de Y.S. Abu-Mostafa*
- *California Institute of Technology*

Learning is the automatic process of building (adjusting) a model from a data set which is representative from the full population.

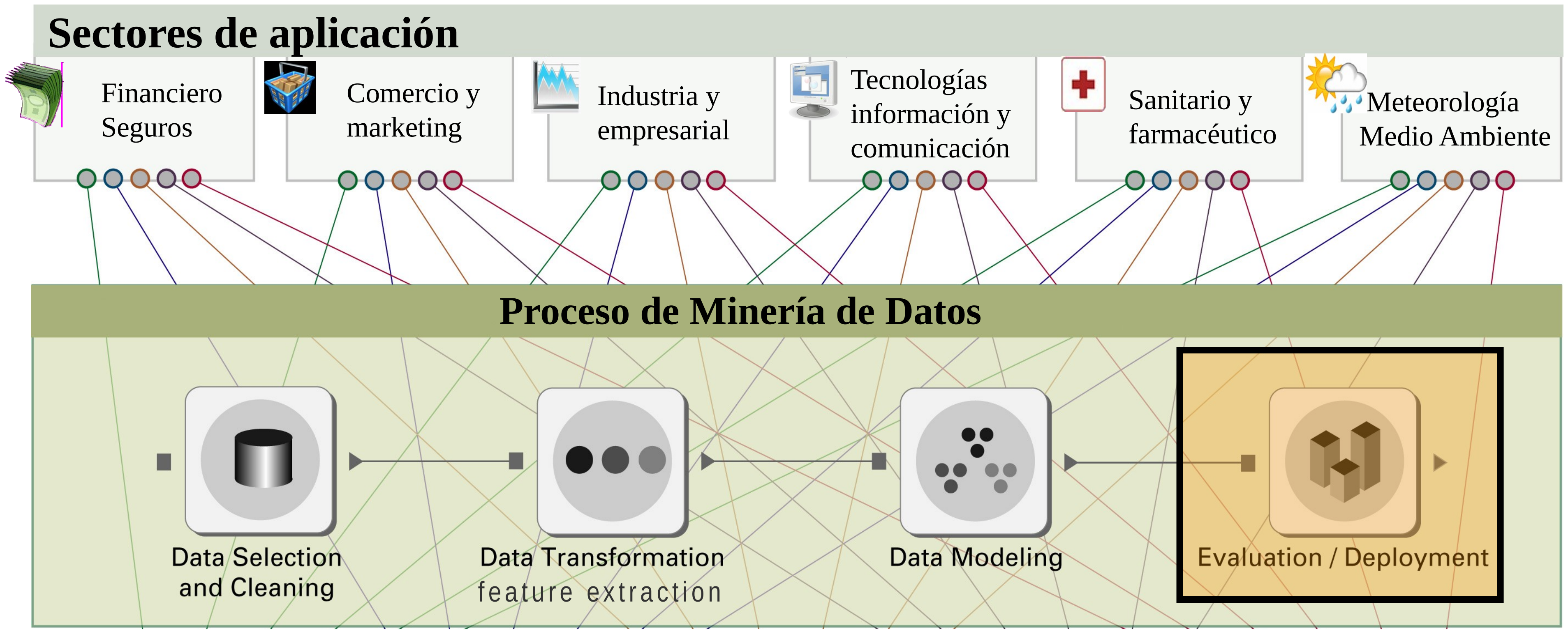


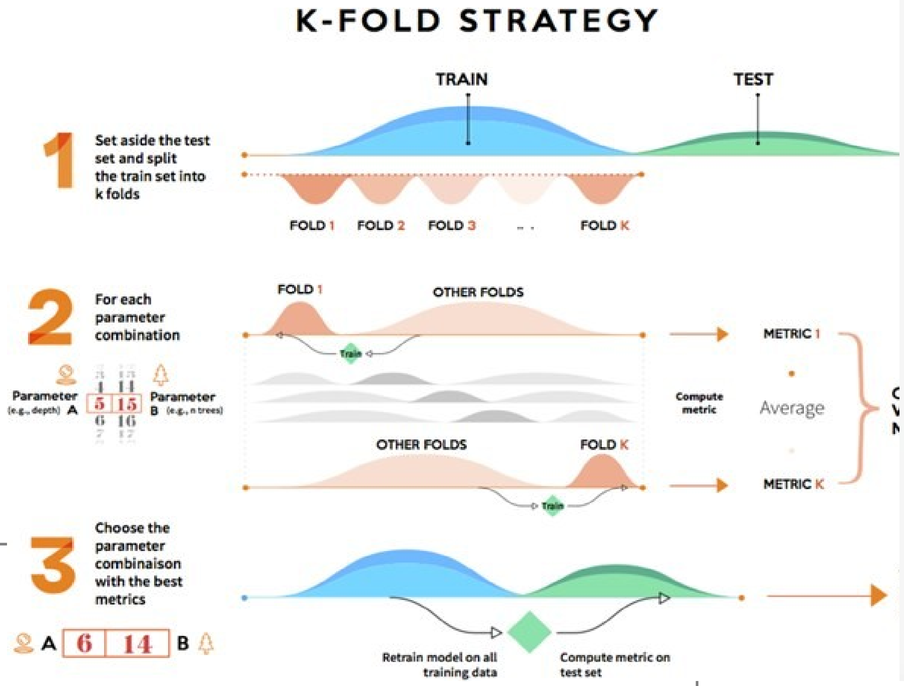
- *Adaptado de Y.S. Abu-Mostafa*
- *California Institute of Technology*

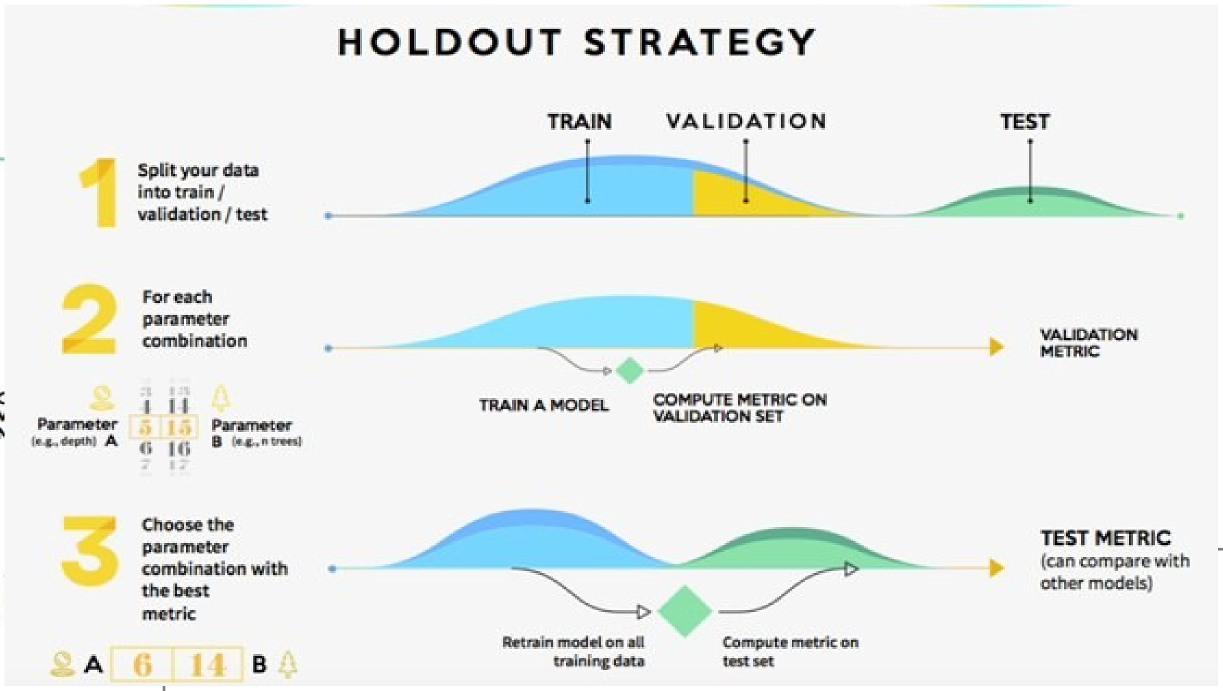
Generalization is the most important feature for data driven systems: They must perform “well” when applied to new data (**cross-validation**).

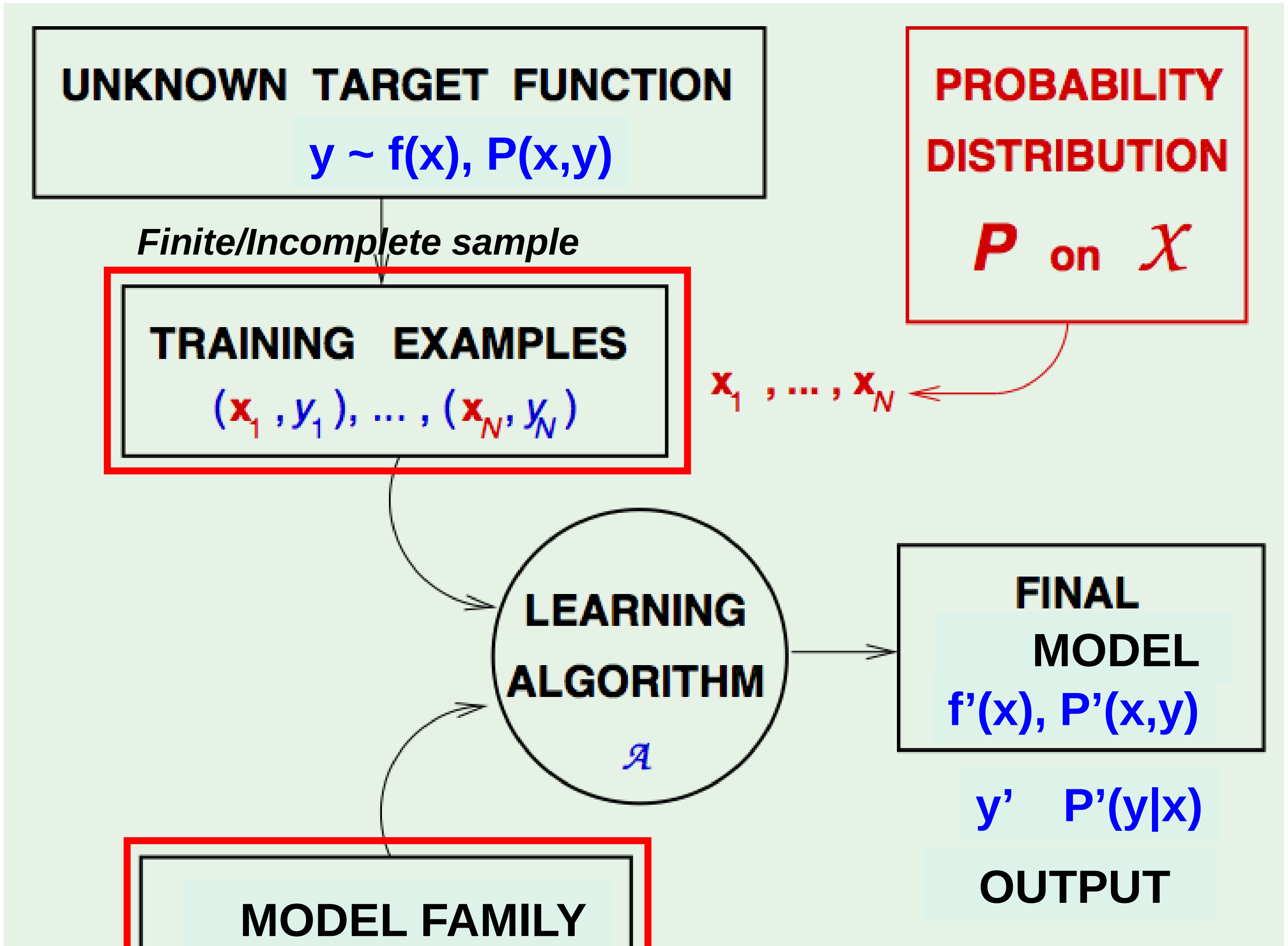


- *Adaptado de Y.S. Abu-Mostafa*
- *California Institute of Technology*


Sectores de aplicación
Financiero Seguros
Comercio y marketing
Industria y empresarial
Tecnologías información y comunicación
Sanitario y farmacéutico
Meteorología Medio Ambiente
Proceso de Minería de Datos
Data Selection and Cleaning
Data Transformation feature extraction
Data Modeling
Evaluation / Deployment


K-FOLD STRATEGY
TRAIN
TEST
1
Set aside the test set and split the train set into k folds
FOLD 1
FOLD 2
FOLD 3
FOLD K
2
For each parameter combination
FOLD 1
OTHER FOLDS
Train
METRIC 1
Compute metric
Average
OTHER FOLDS
FOLD K
METRIC K
Parameter (e.g., depth) A
5 15
Parameter B (e.g., n trees)
3
Choose the parameter combinaison with the best metrics
A 6 14 B
Retrain model on all training data
Compute metric on test set

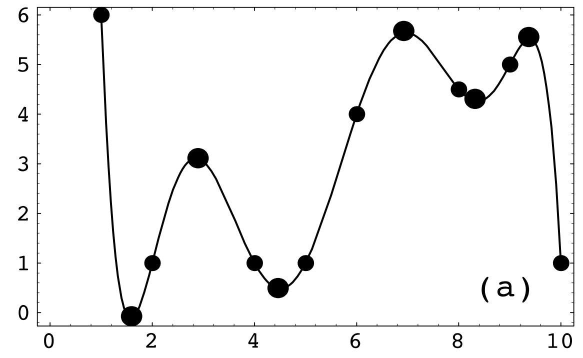

HOLDOUT STRATEGY
TRAIN
VALIDATION
TEST
1
Split your data into train / validation / test
2
For each parameter combination
VALIDATION METRIC
TRAIN A MODEL
COMPUTE METRIC ON VALIDATION SET
Parameter (e.g., depth) A
5 15
Parameter B (e.g., n trees)
3
Choose the parameter combination with the best metric
TEST METRIC (can compare with other models)
A 6 14 B
Retrain model on all training data
Compute metric on test set



Adaptado de Y.S. Abu-Mostafa
California Institute of Technology

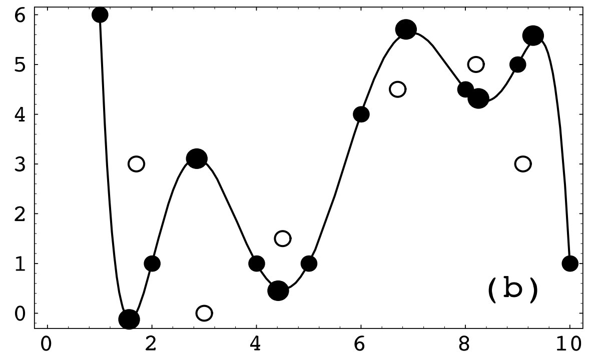
Model complexity <-> degrees of freedom

$$y = \sum_{i=0}^{N} (a_i * x^i)$$



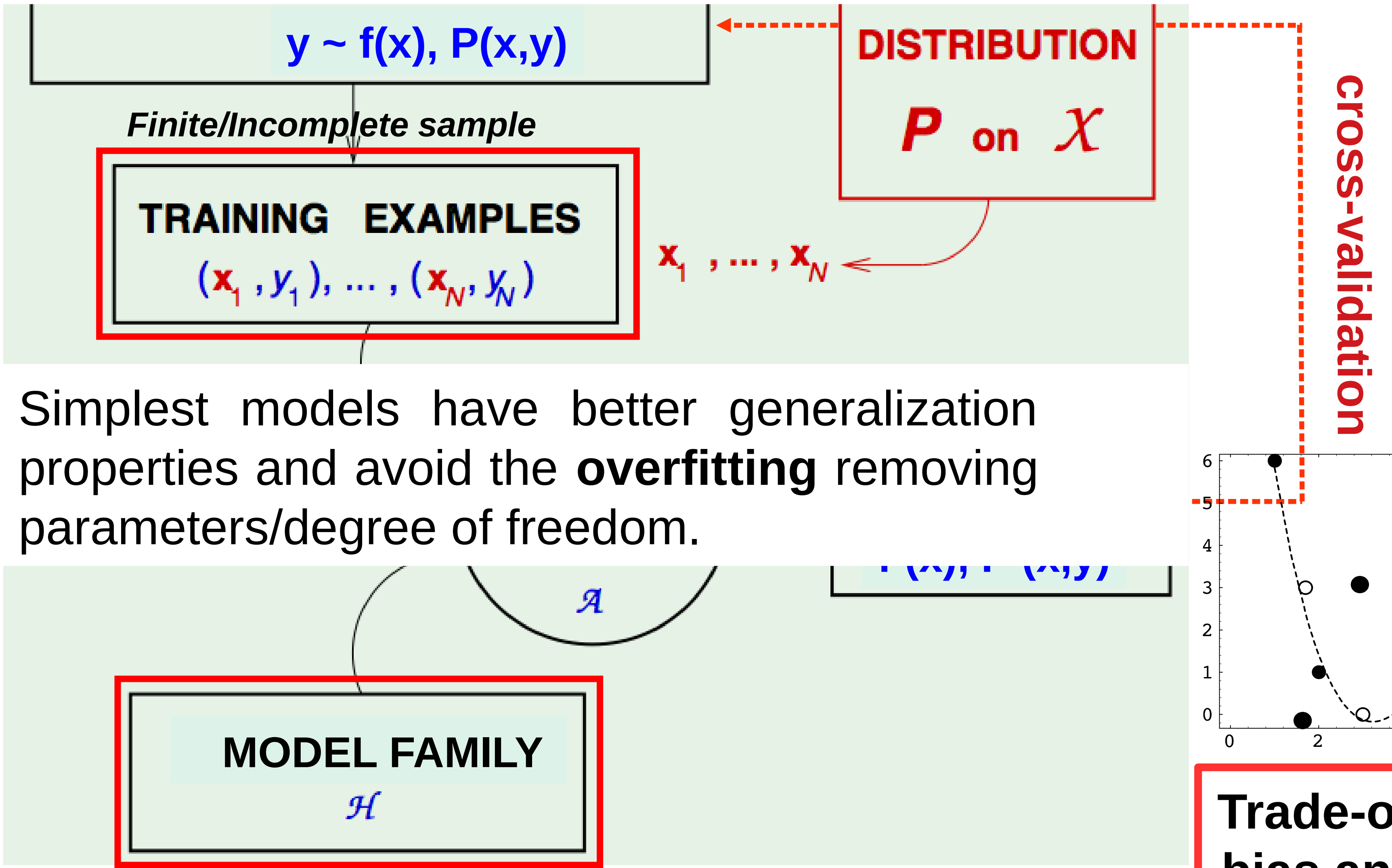
New sample

Overfitting

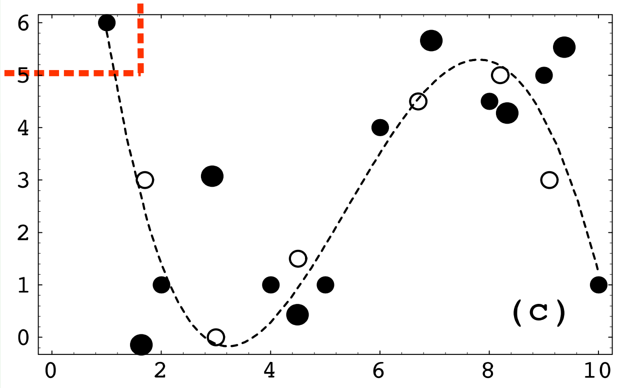


Trade-off between bias and variance

Generalization is the most important feature for data driven systems: They must perform “well” when applied to new data (**cross-validation**).

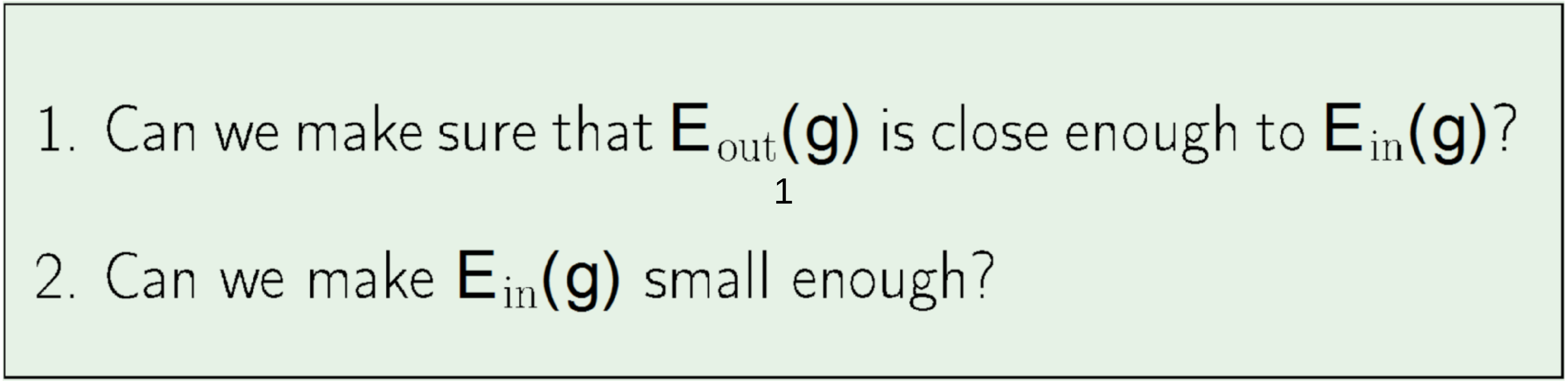


Simplest models have better generalization properties and avoid the **overfitting** removing parameters/degree of freedom.



Trade-off between bias and variance

Generalization is the most important feature for data driven systems: They must perform “well” when applied to new data (**cross-validation**).



1. Can we make sure that $E_{\text{out}}(g)$ is close enough to $E_{\text{in}}(g)$?

1

2. Can we make $E_{\text{in}}(g)$ small enough?

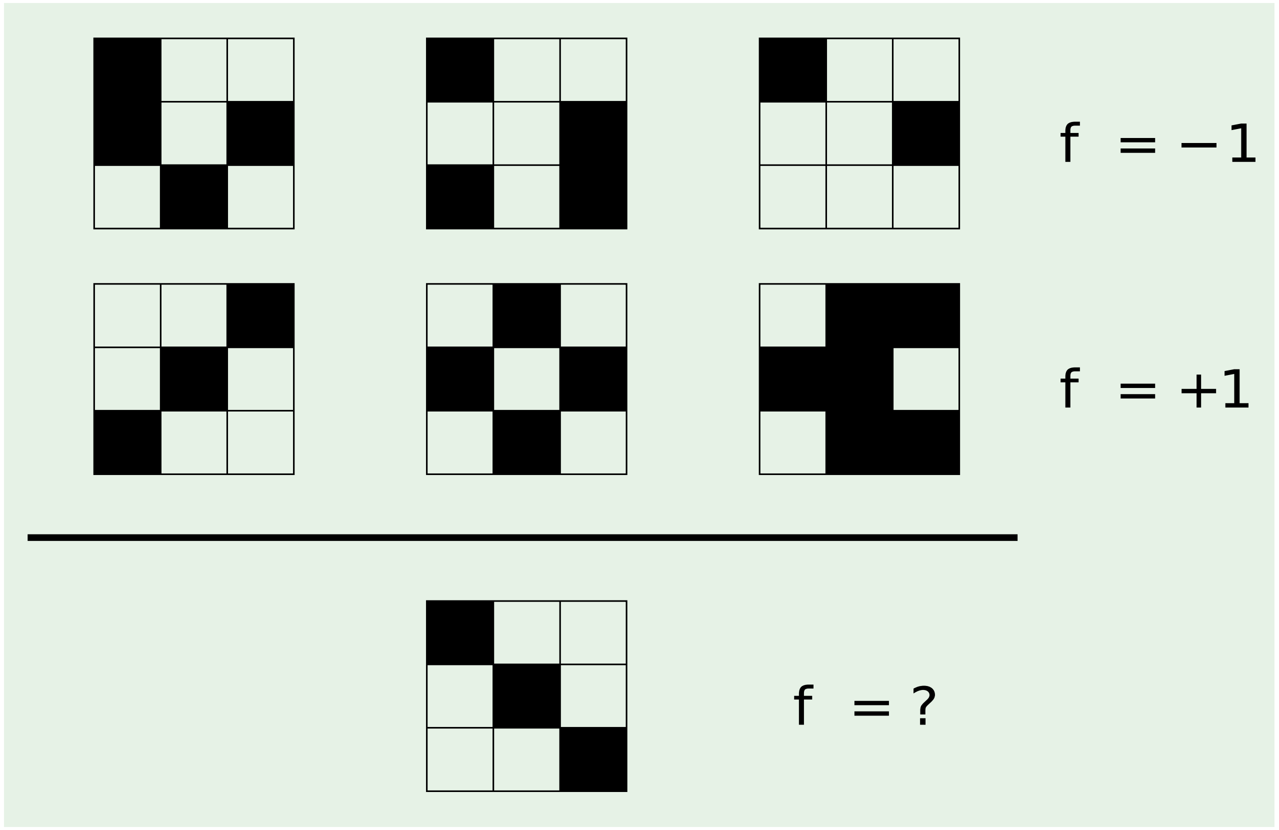
Generalization is the most important feature for data driven systems: They must perform "well" when applied to new data (**cross-validation**).

1. Can we make sure that $E_{out}(g)$ is close enough to $E_{in}(g)$?
2. Can we make $E_{in}(g)$ small enough?

1

The (***in-sample***) error is the unique which can be estimated:

$$E_{in}(h) = \frac{1}{N}\sum_{n=1}^{N}(h(x_n) - y_n)^2$$



		f	
		+1	−1
h	+1	no error	*false accept*
	−1	*false reject*	no error

$$E_{out}(h)=E(f,h)$$

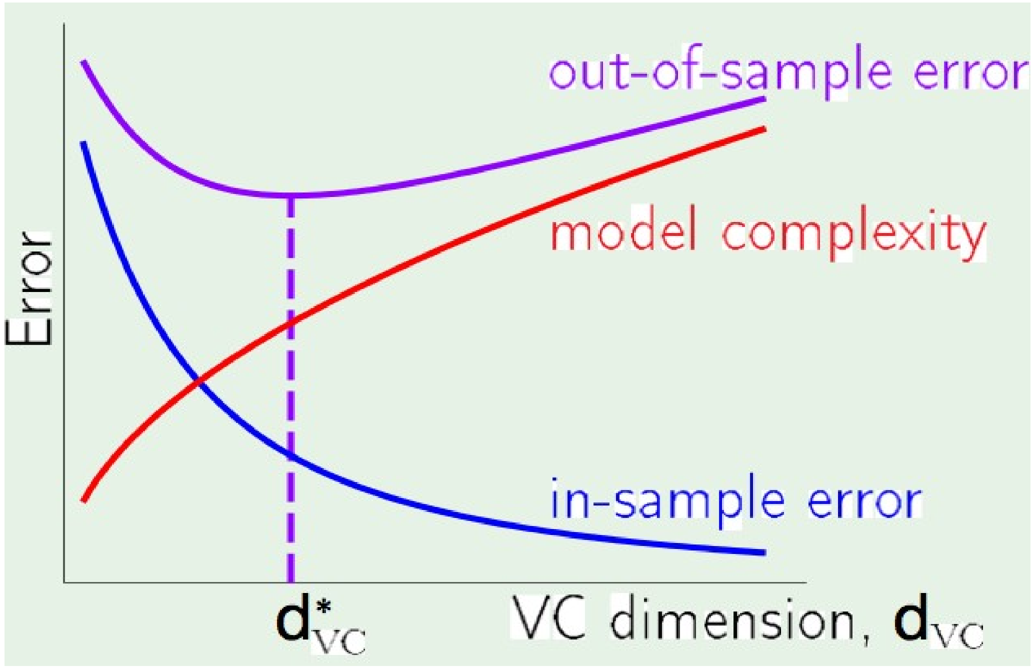
Generalization is the most important feature for data driven systems: They must perform “well” when applied to new data (**cross-validation**).

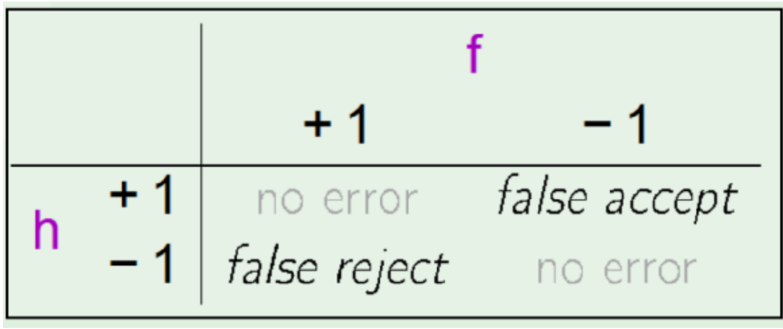
1. Can we make sure that $E_{out}(g)$ is close enough to $E_{in}(g)$?
1
2. Can we make $E_{in}(g)$ small enough?

The (***in-sample***) error is the unique which can be estimated:

$$E_{in}(h) = \frac{1}{N}\sum_{n=1}^{N}(h(x_n) - y_n)^2$$

Vapnik-Chervonenkis (VC) Dimension





		f	
		+1	−1
h	+1	no error	*false accept*
	−1	*false reject*	no error

$$E_{out}(h)=E(f,h)$$

Generalization is the most important feature for data driven systems: They must perform “well” when applied to new data (**cross-validation**).

1. Can we make sure that $E_{out}(g)$ is close enough to $E_{in}(g)$? 1
2. Can we make $E_{in}(g)$ small enough?

The (***in-sample***) error is the unique which can be estimated:

$$E_{in}(h) = \frac{1}{N}\sum_{n=1}^{N}(h(x_n) - y_n)^2$$

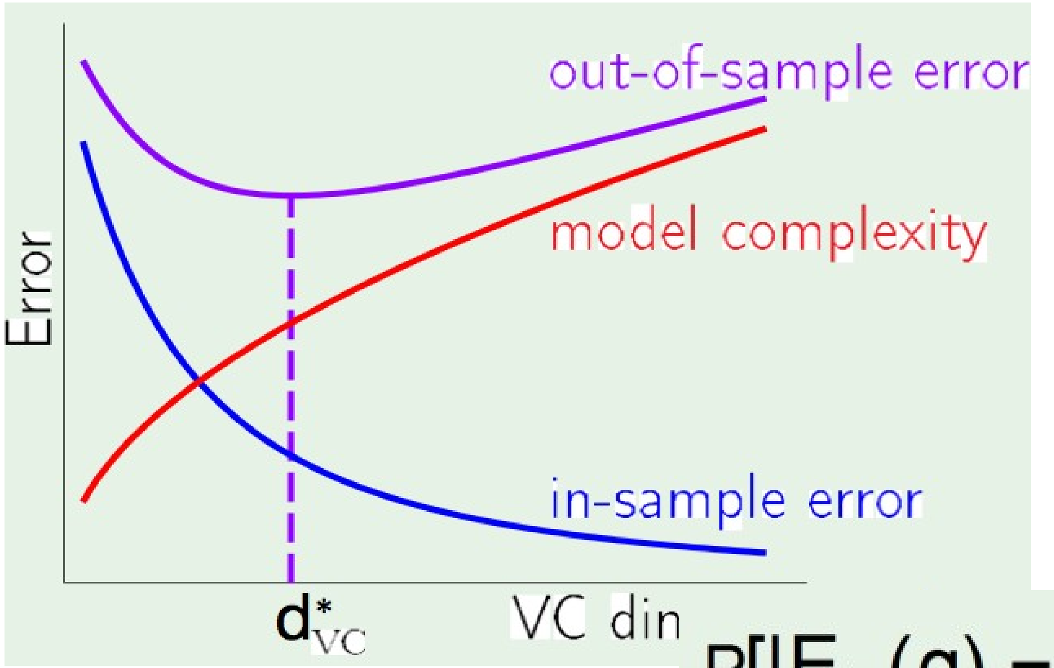
$$E_{out}(h)=E(f,h)$$

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

N=sample size

M=model complexity

Vapnik-Chervonenkis (VC) Dimension

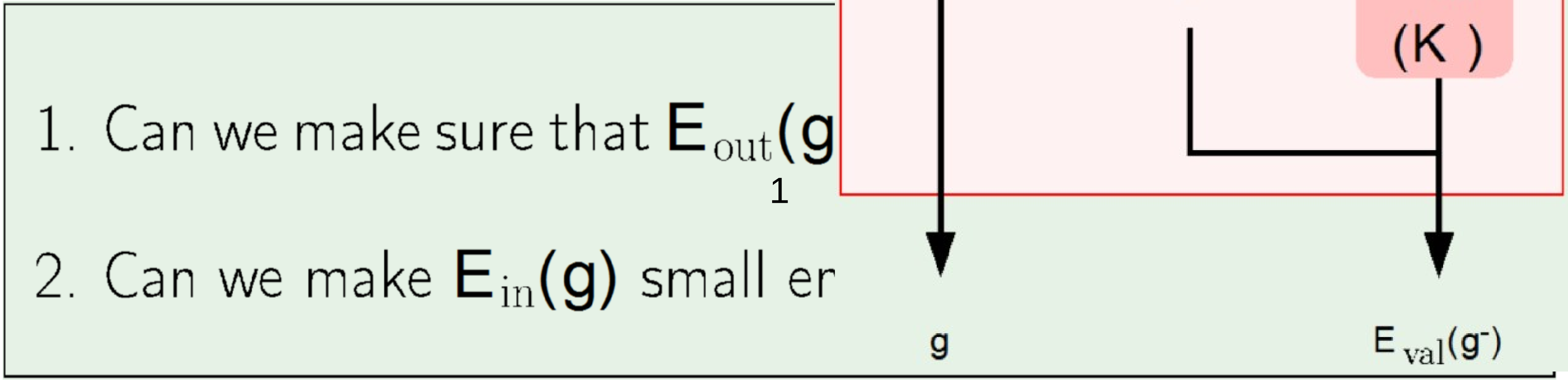


$$P[|E_{in}(g) - E_{out}(g)| > \epsilon] \leq 2M e^{-2\epsilon^2 N}$$

Generalization is the most important feature for data driven systems: They must perform “well” when applied to new data (**cross-validation**).

The sample is divided in two subsets: **train** and **test**.

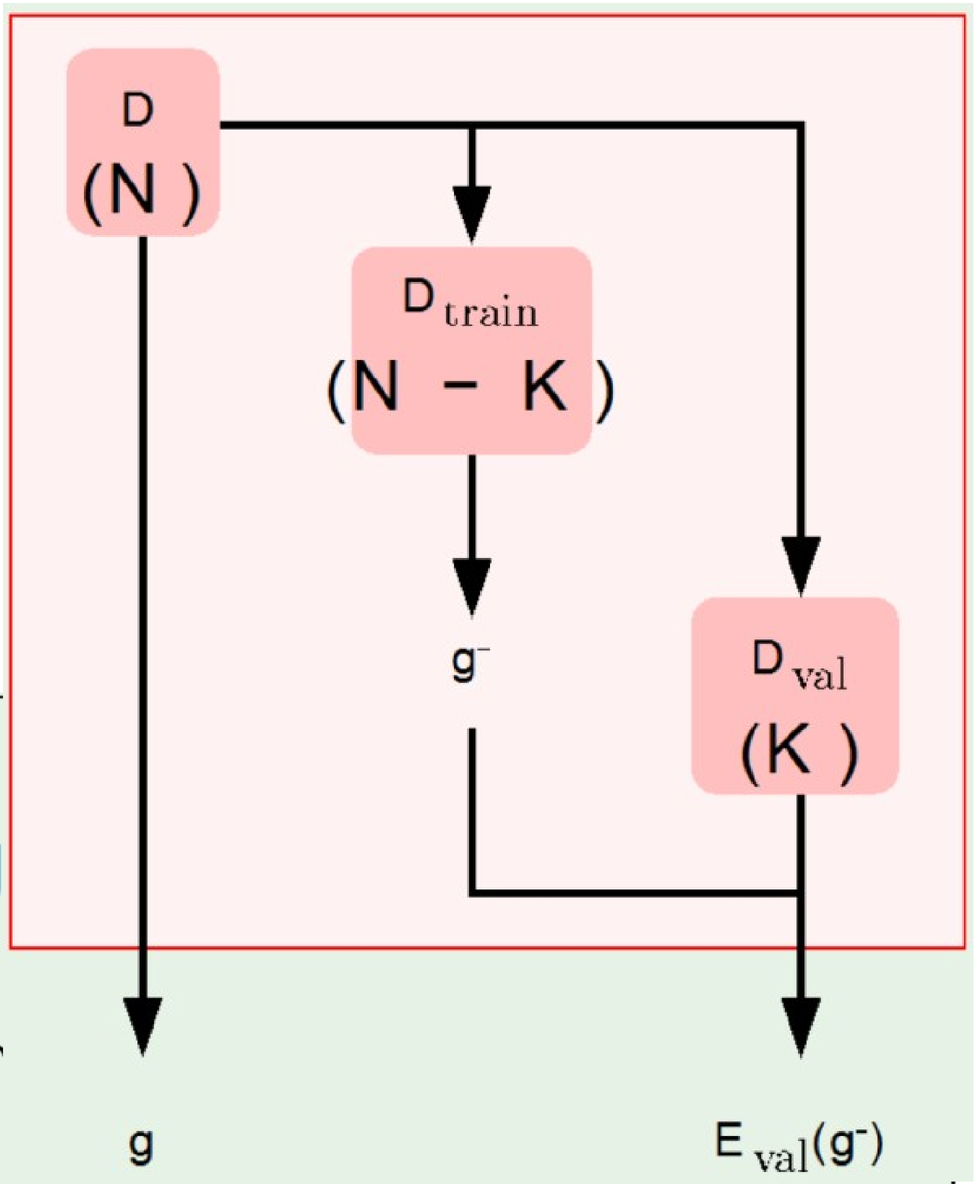
- Hold-out
- Leave-one-out
- K-fold

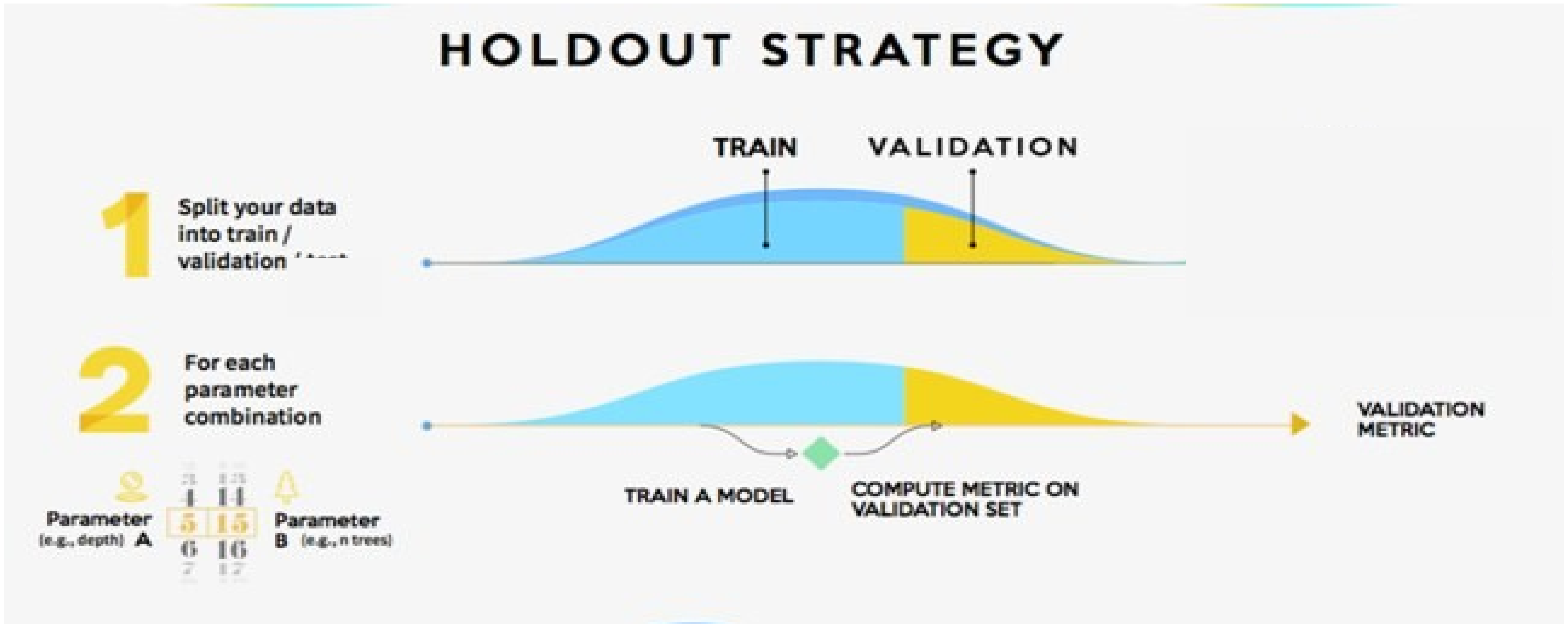


1. Can we make sure that $E_{out}(g$

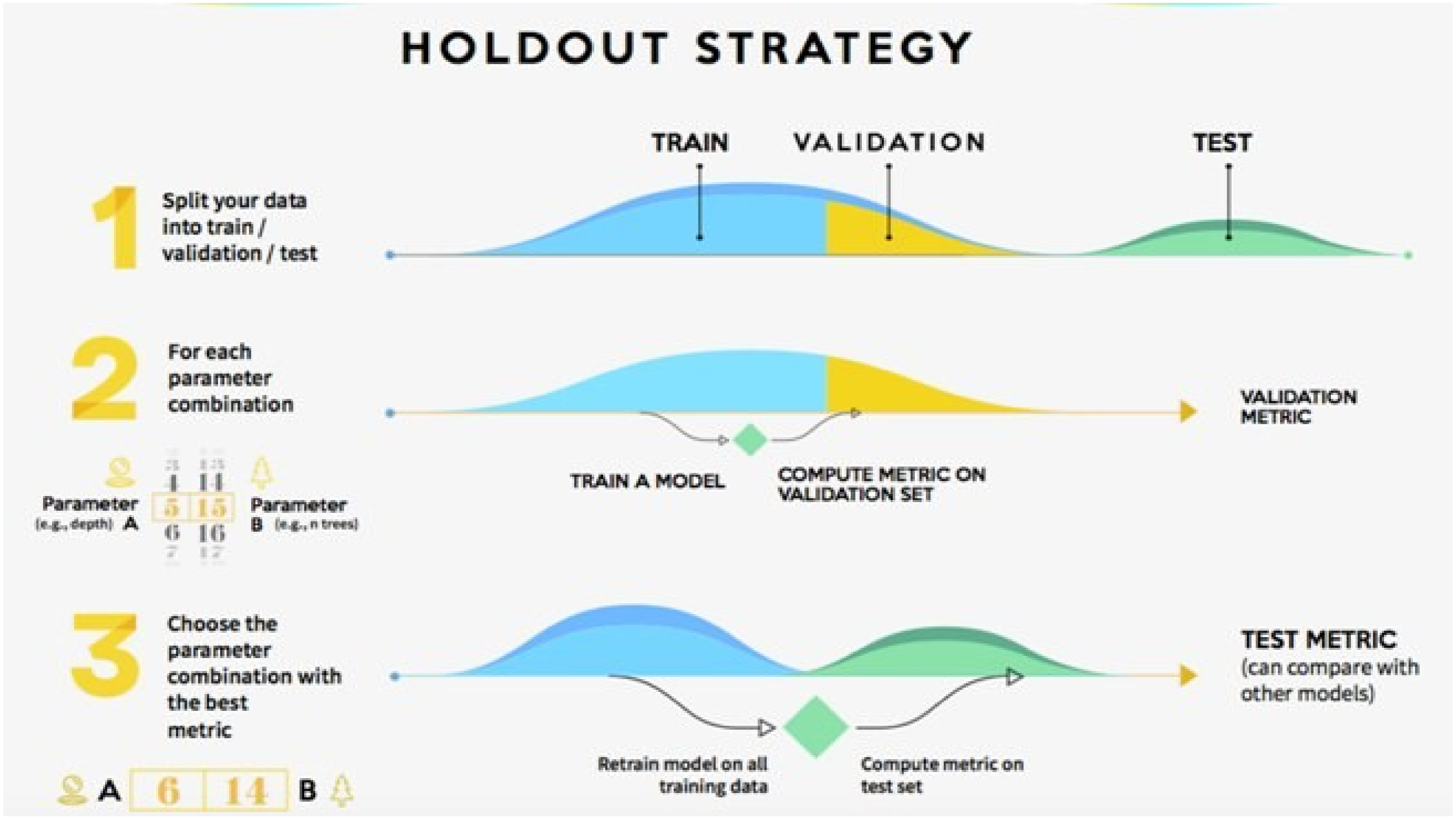
1

2. Can we make $E_{in}(g)$ small er

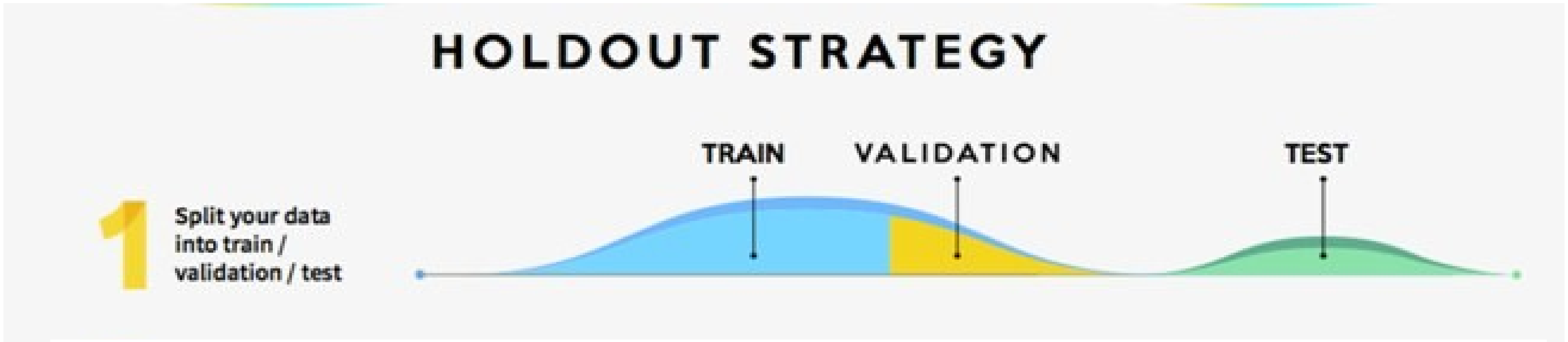



HOLDOUT STRATEGY
TRAIN
VALIDATION
1
Split your data into train / validation
2
For each parameter combination
VALIDATION METRIC
TRAIN A MODEL
COMPUTE METRIC ON VALIDATION SET
Parameter (e.g., depth) A
Parameter B (e.g., n trees)

Source: Robert Kelley

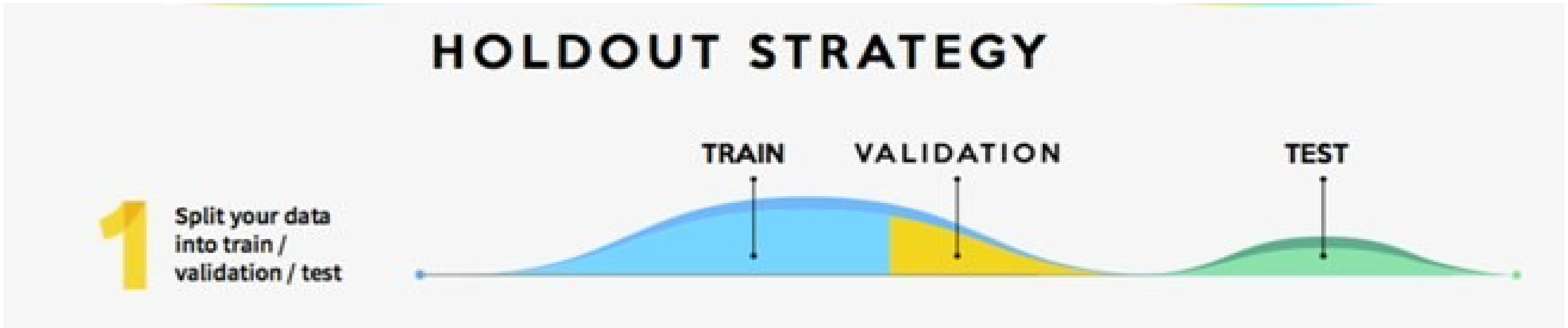

HOLDOUT STRATEGY
TRAIN
VALIDATION
TEST
1 Split your data into train / validation / test
2 For each parameter combination
VALIDATION METRIC
TRAIN A MODEL
COMPUTE METRIC ON VALIDATION SET
Parameter (e.g., depth) A
Parameter B (e.g., n trees)
3 Choose the parameter combination with the best metric
A 6 14 B
TEST METRIC (can compare with other models)
Retrain model on all training data
Compute metric on test set

Source: Robert Kelley



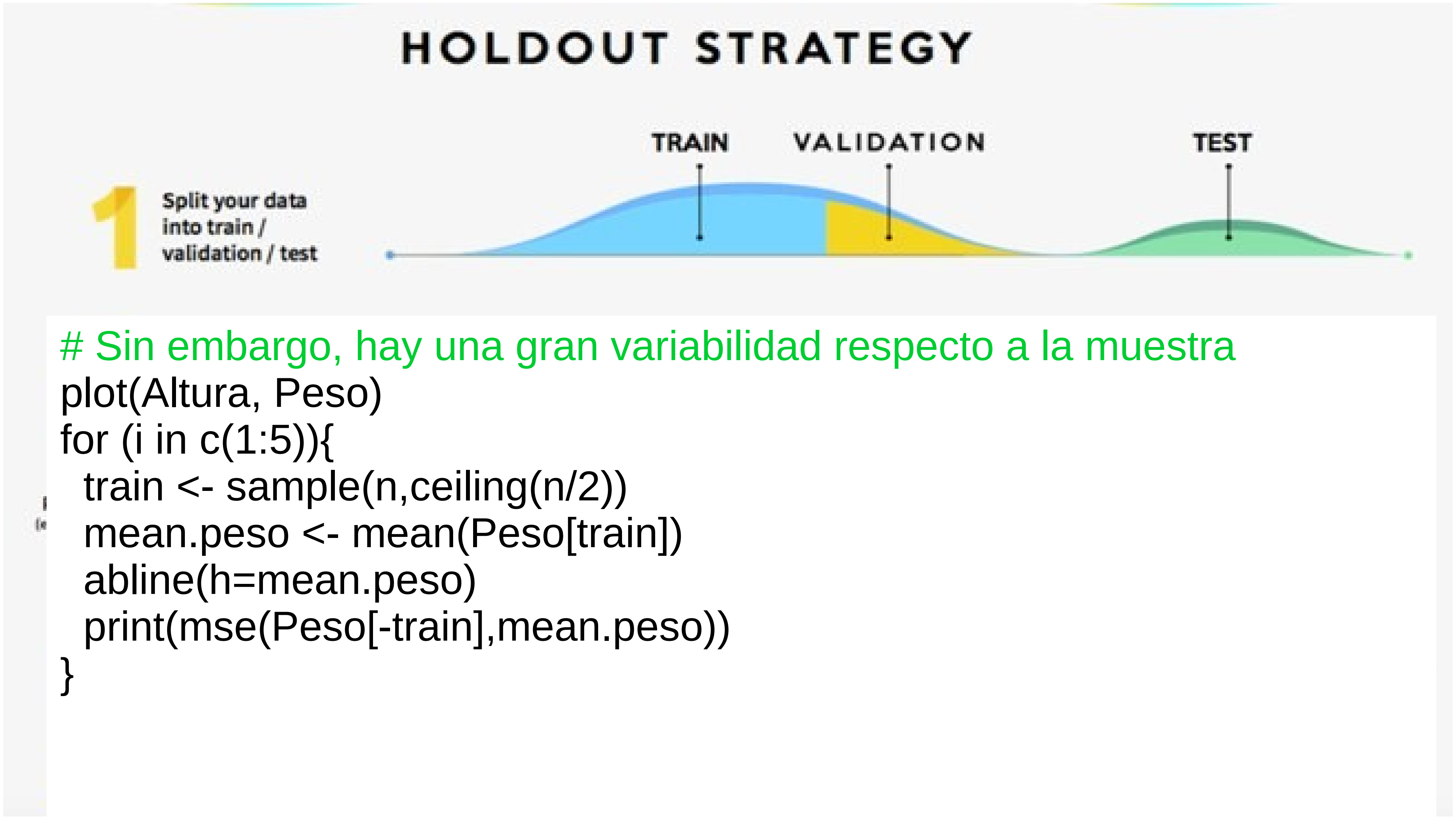
```
plot(Altura, Peso)
train <- 1:ceiling(n/2)
order.index <- order(Peso)
Peso.sort <- Peso[order.index]
Altura.sort <- Altura[order.index]
points(Altura.sort[train], Peso.sort[train], pch=16, col="red")
mean.peso <- mean(Peso.sort[train])
abline(h=mean.peso)
# El error de test es mucho mayor ya que el modelo no generaliza.
mse.train <- mse(Peso.sort[train],mean.peso); mse.train
mse.test <- mse(Peso.sort[-train],mean.peso); mse.test
```

Source: Robert Kelley



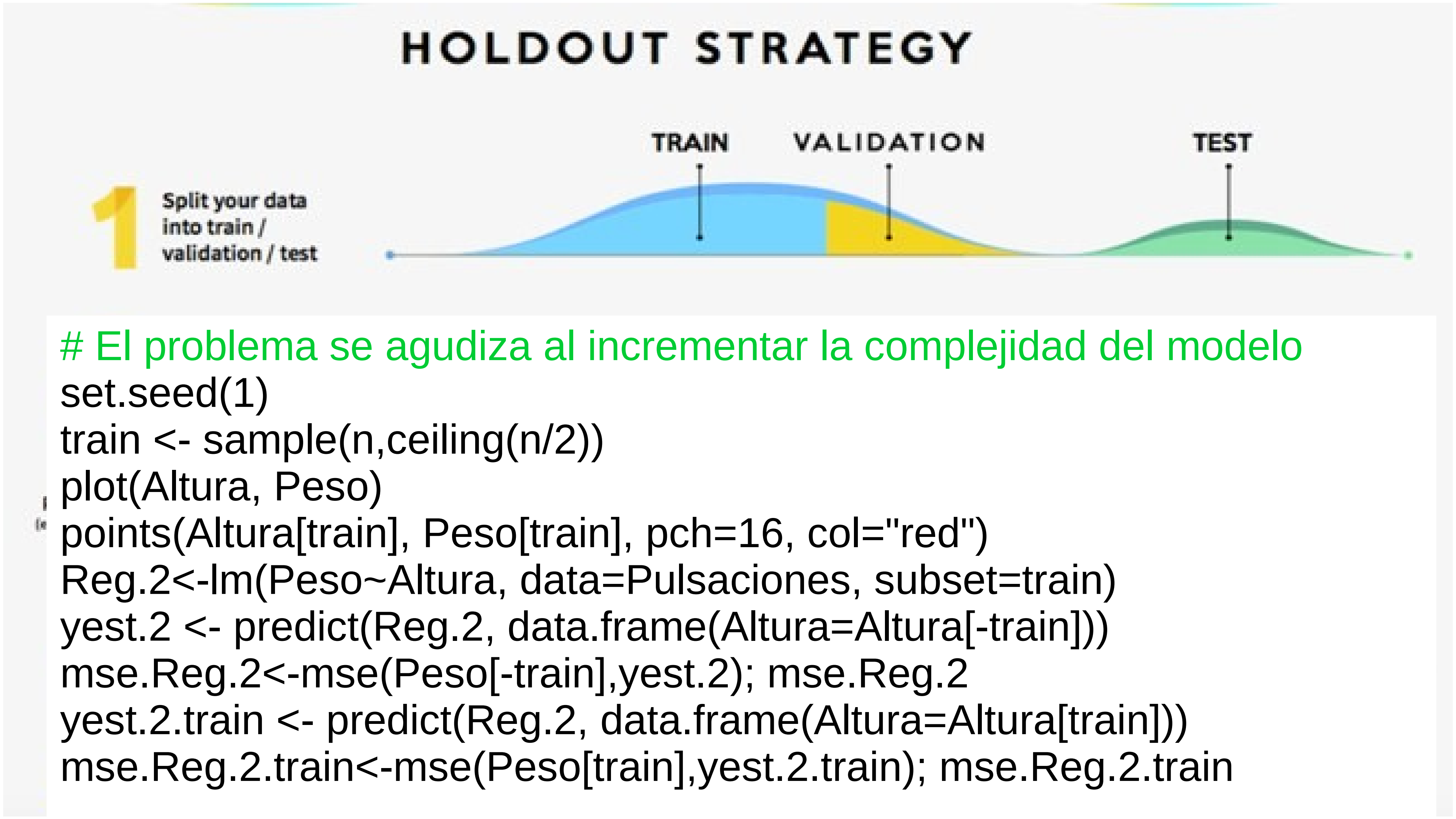
```
# Mejor si cogemos los datos aleatoriamente.
set.seed(1) # Para obtener el mismo valor fijamos la semilla
train <- sample(n,ceiling(n/2))
plot(Altura, Peso)
points(Altura[train], Peso[train], pch=16, col="red")
#y.est=cte esa cte es la media de la variable y selecionada en train
mean.peso <- mean(Peso[train])
abline(h=mean.peso)
mse.train <- mse(Peso[train],mean.peso); mse.train
mse.test <- mse(Peso[-train],mean.peso); mse.test
```

Source: Robert Kelley



```
# Sin embargo, hay una gran variabilidad respecto a la muestra
plot(Altura, Peso)
for (i in c(1:5)){
  train <- sample(n,ceiling(n/2))
  mean.peso <- mean(Peso[train])
  abline(h=mean.peso)
  print(mse(Peso[-train],mean.peso))
}
```

Source: Robert Kelley



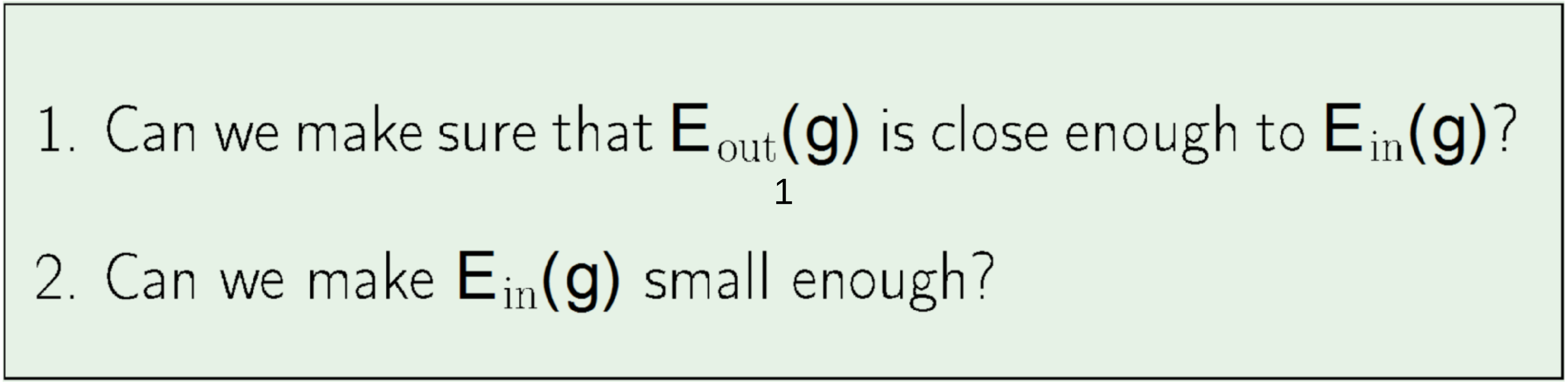
```
# El problema se agudiza al incrementar la complejidad del modelo
set.seed(1)
train <- sample(n,ceiling(n/2))
plot(Altura, Peso)
points(Altura[train], Peso[train], pch=16, col="red")
Reg.2<-lm(Peso~Altura, data=Pulsaciones, subset=train)
yest.2 <- predict(Reg.2, data.frame(Altura=Altura[-train]))
mse.Reg.2<-mse(Peso[-train],yest.2); mse.Reg.2
yest.2.train <- predict(Reg.2, data.frame(Altura=Altura[train]))
mse.Reg.2.train<-mse(Peso[train],yest.2.train); mse.Reg.2.train
```

Source: Robert Kelley

Generalization is the most important feature for data driven systems: They must perform “well” when applied to new data (**cross-validation**).

The sample is divided in two subsets:
train and **test**.

- **Hold-out:**
  - Variability related with the train-test splitting.
  - The sample size of the test sample leads to conservative results.
  - The sample size of the train sample limits the complexity of the model.
- Leave-one-out
- K-fold



1. Can we make sure that $\mathsf{E}_{\text{out}}(\mathsf{g})$ is close enough to $\mathsf{E}_{\text{in}}(\mathsf{g})$?

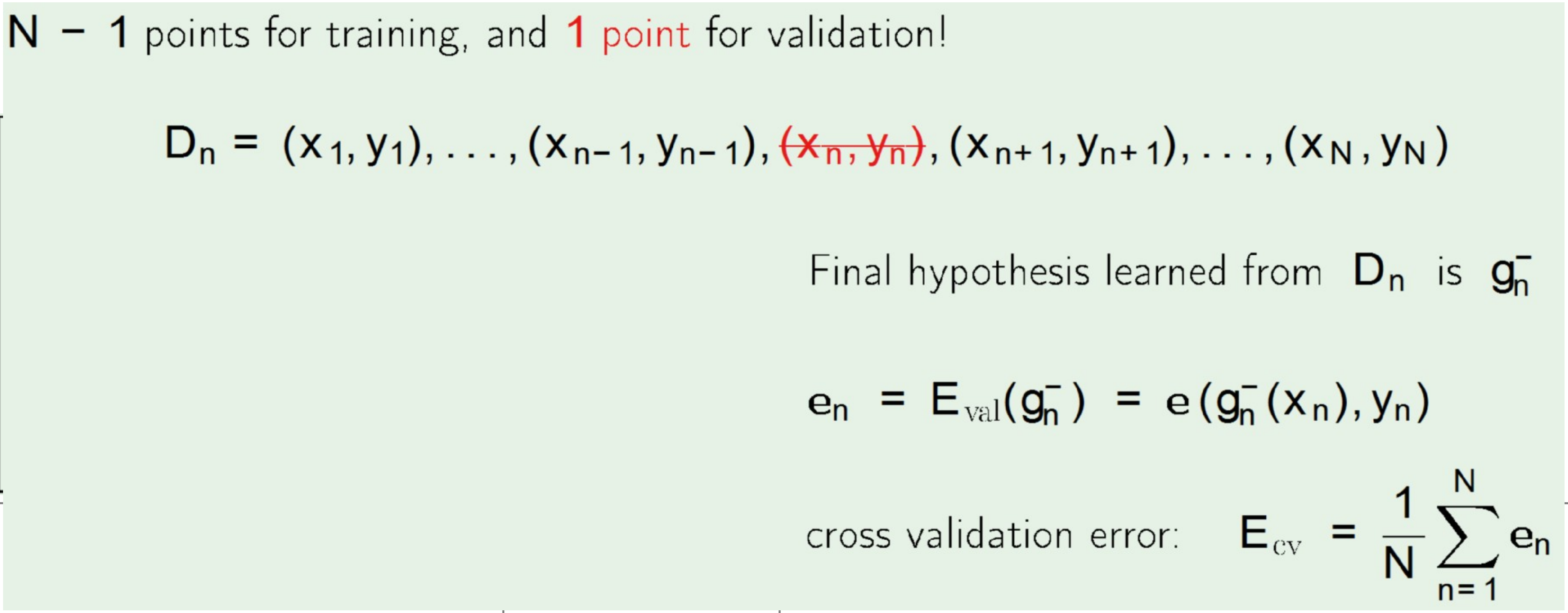
1

2. Can we make $\mathsf{E}_{\text{in}}(\mathsf{g})$ small enough?

Generalization is the most important feature for data driven systems: They must perform “well” when applied to new data (**cross-validation**).

The sample is divided in two subsets:
train and **test**.

- Hold-out:
  - Variability related with the train-test splitting.
  - The sample size of the test sample leads to conservative results.
  - The sample size of the train sample limits the complexity of the model.
- **Leave-one-out**
- K-fold



$N-1$ points for training, and 1 point for validation!

$$\mathcal{D}_n = (x_1, y_1), \ldots, (x_{n-1}, y_{n-1}), \cancel{(x_n, y_n)}, (x_{n+1}, y_{n+1}), \ldots, (x_N, y_N)$$

Final hypothesis learned from $\mathcal{D}_n$ is g_n^-

$$e_n = E_{\mathrm{val}}(g_n^-) = e(g_n^-(x_n), y_n)$$

cross validation error: $E_{\mathrm{cv}} = \dfrac{1}{N}\displaystyle\sum_{n=1}^{N} e_n$

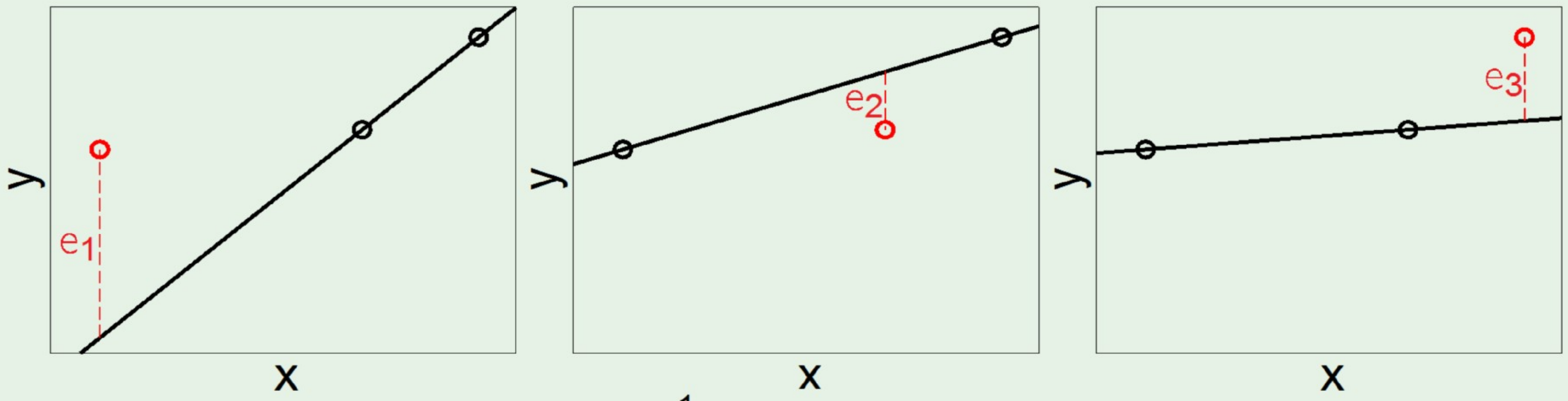
Generalization is the most important feature for data driven systems: They must perform "well" when applied to new data (**cross-validation**).

The sample is divided in two subsets:
train and **test**.

- Hold-out:
  - Variability related with the train-test splitting.
  - The sample size of the test sample leads to conservative results.
  - The sample size of the train sample limits the complexity of the model.
- **Leave-one-out**
- K-fold

$N-1$ points for training, and 1 point for validation!

$$D_n = (x_1, y_1), \dots, (x_{n-1}, y_{n-1}), \cancel{(x_n, y_n)}, (x_{n+1}, y_{n+1}), \dots, (x_N, y_N)$$



$$E_{cv} = \frac{1}{3}(e_1 + e_2 + e_3)$$

Generalization is the most important feature for data driven systems: They must perform “well” when applied to new data (**cross-validation**).

The sample is divided in two subsets:
train and **test**.

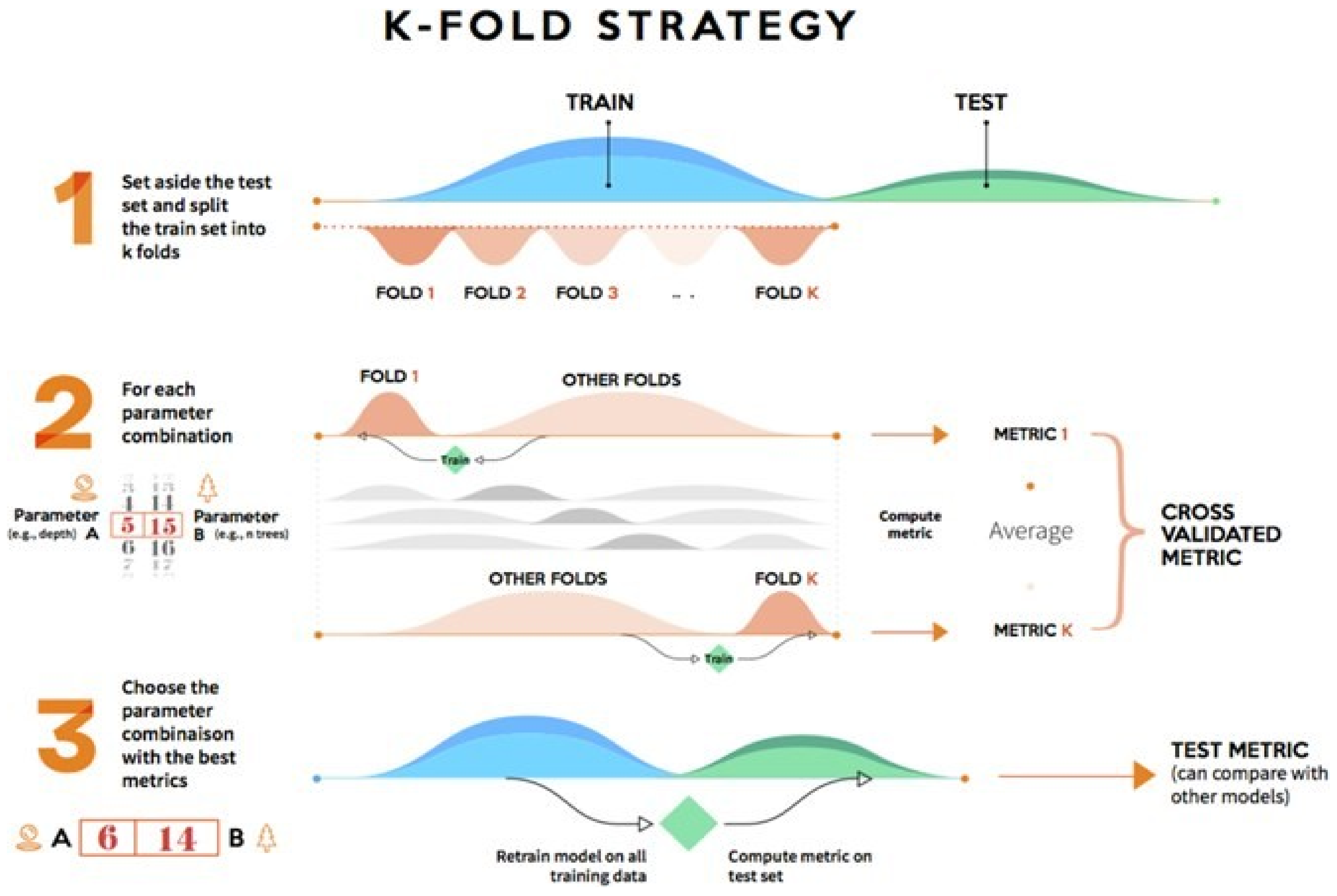
- Hold-out:
  - Variability related with the train-test splitting.
  - The sample size of the test sample leads to conservative results.
  - The sample size of the train sample limits the complexity of the model.
- **Leave-one-out:**
  - High computational cost (**small samples**).
- K-fold

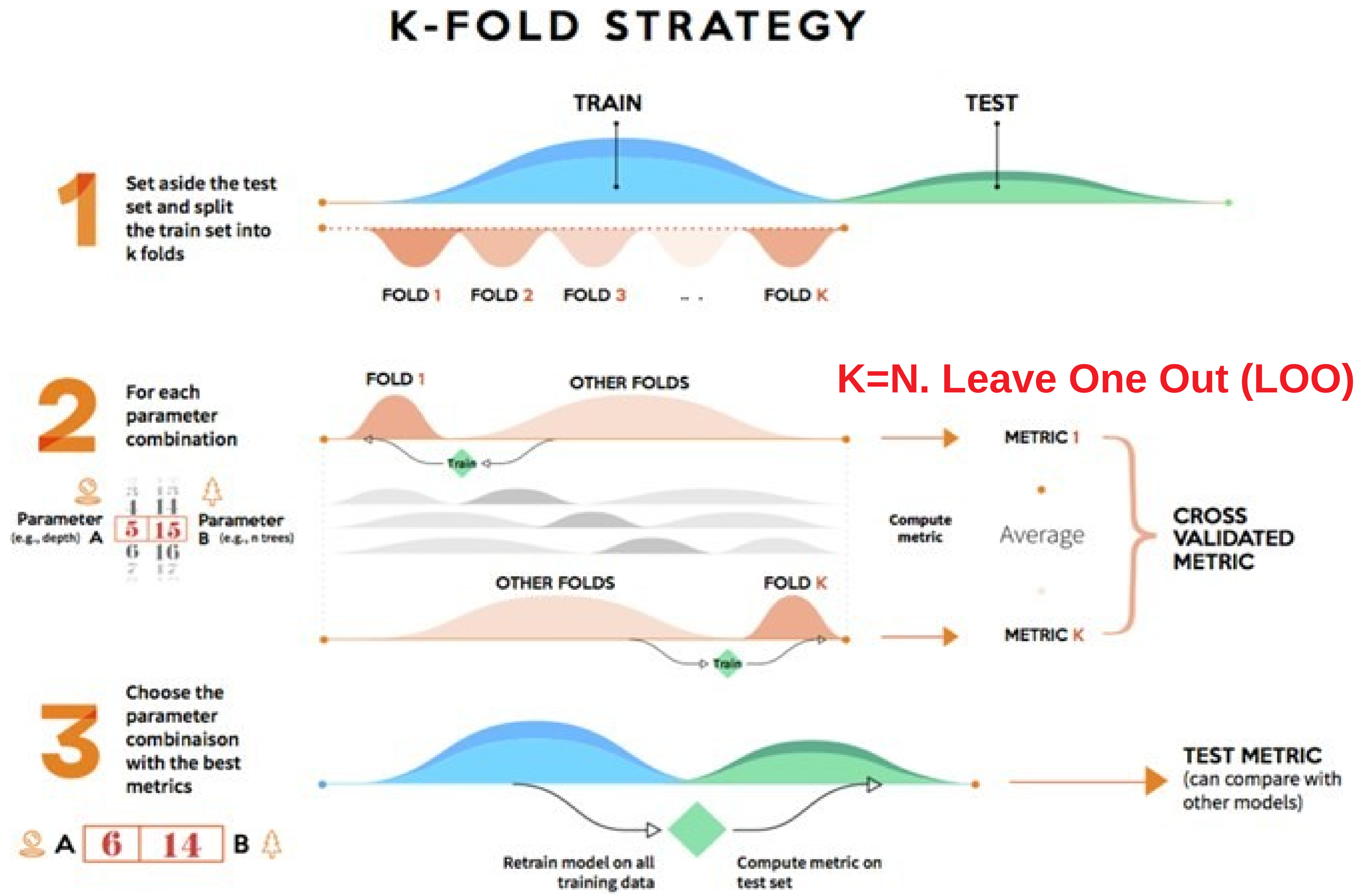
Generalization is the most important feature for data driven systems: They must perform "well" when applied to new data (**cross-validation**).

The sample is divided in two subsets:
train and **test**.

- Hold-out:
  - Variability related with the train-test splitting.
  - The sample size of the test sample leads to conservative results.
  - The sample size of the train sample limits the complexity of the model.
- **Leave-one-out:**
  - High computational cost (**small samples**).
- K-fold

```
# Leave-One-Out Cross-Validation
yest.3<-rep(NA, length(train)) # La actualización es ineficiente
train <- 1:n
for (i in train){
  Reg.i<-lm(Peso~Altura, data=Pulsaciones, subset=train[-i])
  yest.3[i]<-predict(Reg.i,data.frame(Altura=Altura[i]))
}
mse.Reg.3<-mse(Peso,yest.3); mse.Reg.3
```


K-FOLD STRATEGY
TRAIN
TEST
1
Set aside the test set and split the train set into k folds
FOLD 1
FOLD 2
FOLD 3
...
FOLD K
2
For each parameter combination
Parameter (e.g., depth) A
Parameter B (e.g., n trees)
FOLD 1
OTHER FOLDS
Train
METRIC 1
Compute metric
Average
CROSS VALIDATED METRIC
OTHER FOLDS
FOLD K
Train
METRIC K
3
Choose the parameter combinaison with the best metrics
A 6 14 B
Retrain model on all training data
Compute metric on test set
TEST METRIC (can compare with other models)

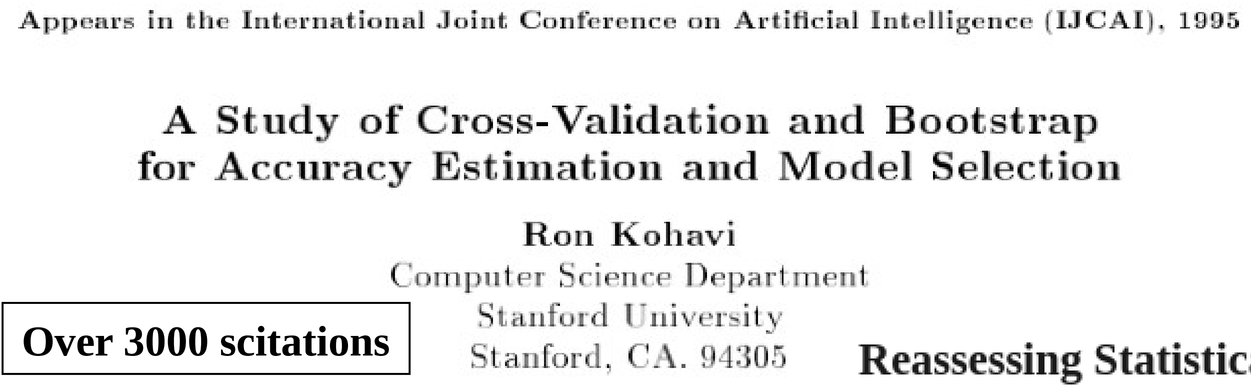

K-FOLD STRATEGY
TRAIN
TEST
1
Set aside the test set and split the train set into k folds
FOLD 1
FOLD 2
FOLD 3
...
FOLD K
2
For each parameter combination
FOLD 1
OTHER FOLDS
Train
Parameter (e.g., depth) A
5 15
Parameter B (e.g., n trees)
Compute metric
METRIC 1
Average
METRIC K
CROSS VALIDATED METRIC
OTHER FOLDS
FOLD K
Train
3
Choose the parameter combinaison with the best metrics
A 6 14 B
Retrain model on all training data
Compute metric on test set
TEST METRIC (can compare with other models)

K=N. Leave One Out (LOO)

Generalization is the most important feature for data driven systems: They must perform “well” when applied to new data (**cross-validation**).

The sample is divided in two subsets:
train and **test**.

- Hold-out:
  - Variability related with the train-test splitting.
  - The sample size of the test sample leads to conservative results.
  - The sample size of the train sample limits the complexity of the model.
- Leave-one-out:
  - High computational cost (**small samples**).
- **K-fold:**
  - Symilar results than leave-one-out with low number of folds.
  - Statistical analysis of the validation measures.

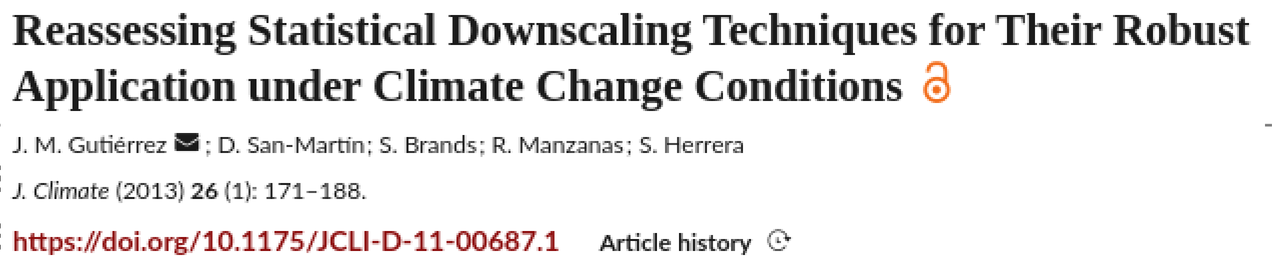

Appears in the International Joint Conference on Artificial Intelligence (IJCAI), 1995

A Study of Cross-Validation and Bootstrap
for Accuracy Estimation and Model Selection

Ron Kohavi
Computer Science Department
Stanford University
Stanford, CA. 94305

Over 3000 scitations

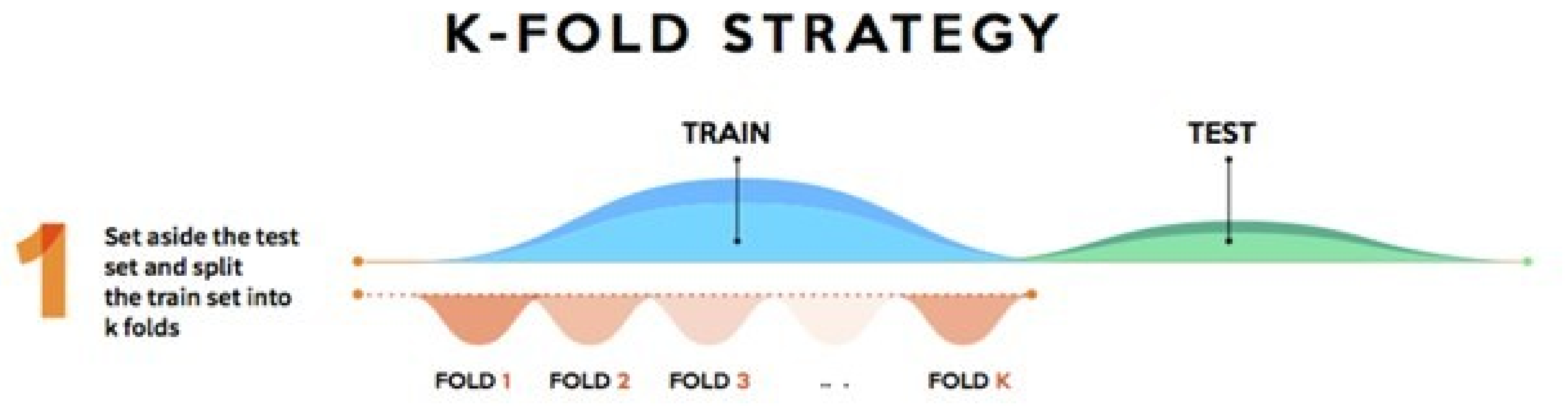
Cross
Valid


Reassessing Statistical Downscaling Techniques for Their Robust Application under Climate Change Conditions

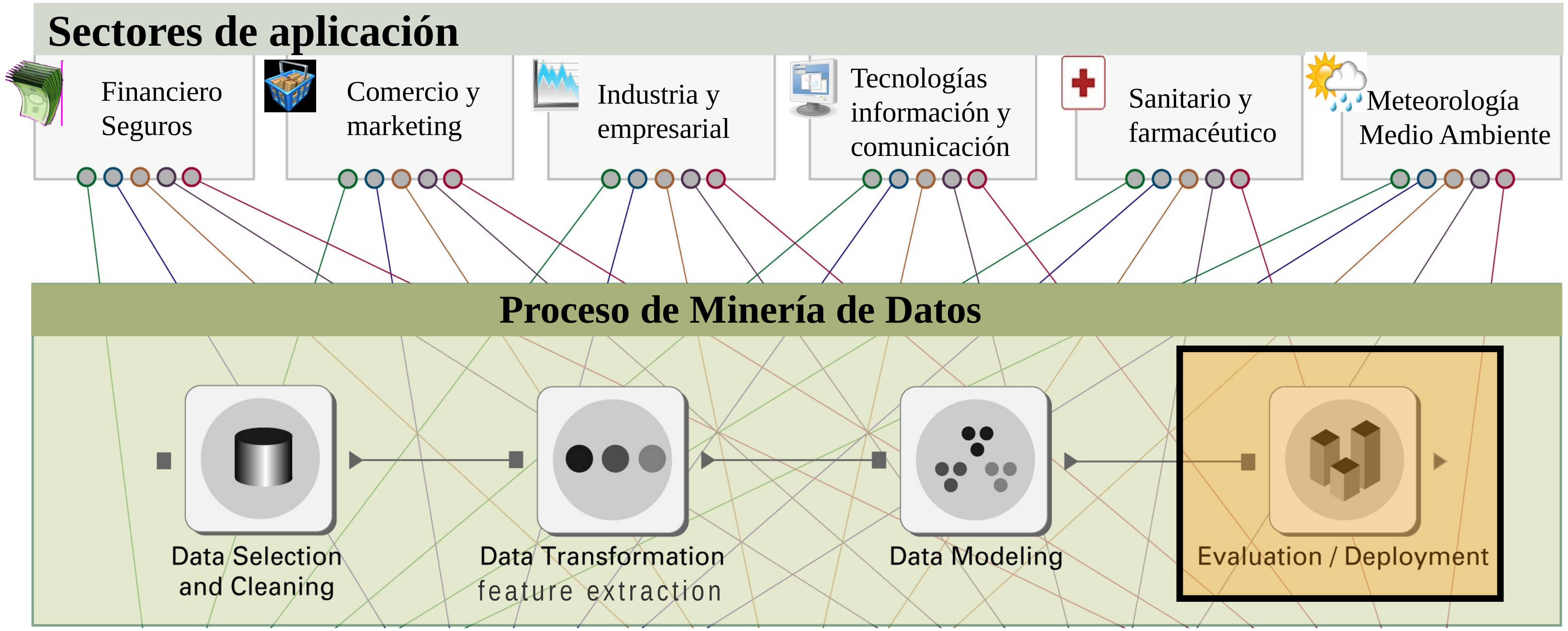
J. M. Gutiérrez; D. San-Martin; S. Brands; R. Manzanas; S. Herrera

J. Climate (2013) **26** (1): 171–188.

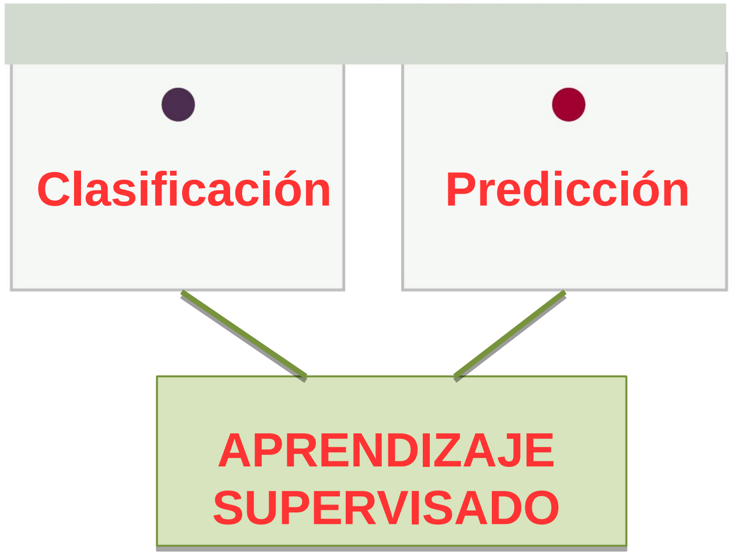
https://doi.org/10.1175/JCLI-D-11-00687.1 Article history



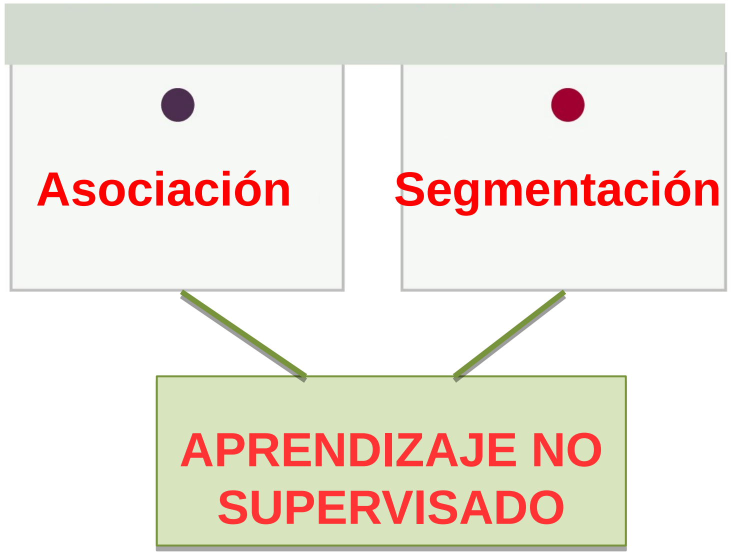
```
# 10-Fold Cross-Validation
idx.aleatorios <- sample(1:n,n,replace=F)
K <- 10
tam <- ceiling(n/K)
yest4 <- rep(NA, length(train)) # La actualización es ineficiente
for (i in 0:(K-1)){
  idx.test <- idx.aleatorios[(i*tam+1):((i+1)*tam)]
  idx.test <- idx.test[!is.na(idx.test)]
  lm4 <- lm(Peso~Altura, subset=-idx.test)
  yest4[idx.test] <- predict(lm4, data.frame(Altura=Altura[idx.test]))
}
mse4 <- mse(Peso,yest4); mse4
```

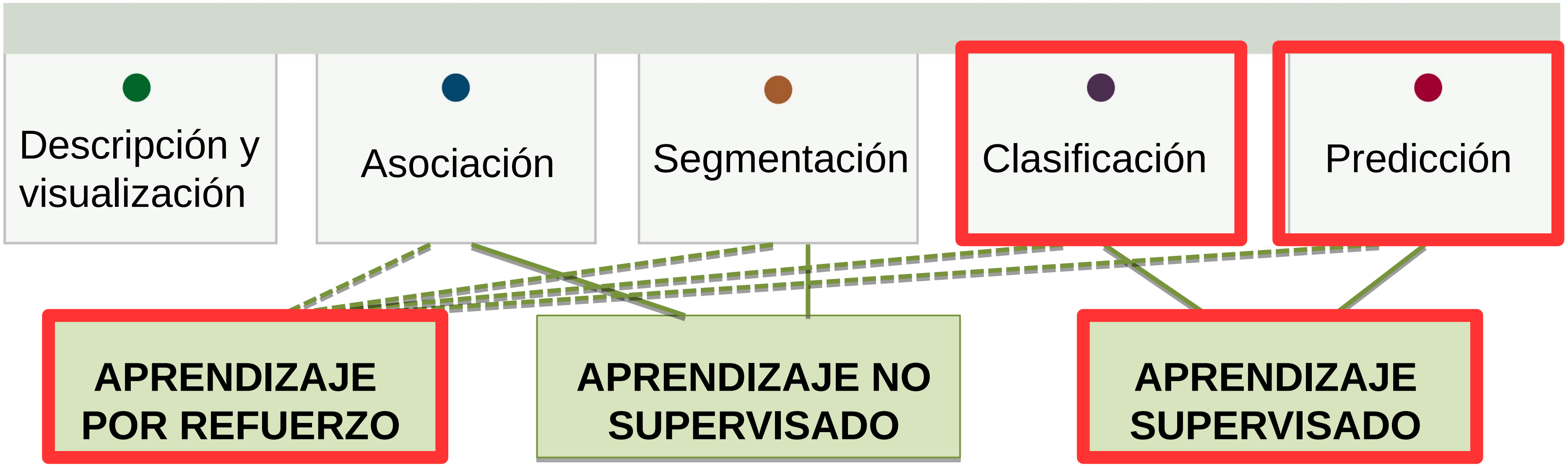

Sectores de aplicación
Financiero Seguros
Comercio y marketing
Industria y empresarial
Tecnologías información y comunicación
Sanitario y farmacéutico
Meteorología Medio Ambiente
Proceso de Minería de Datos
Data Selection and Cleaning
Data Transformation feature extraction
Data Modeling
Evaluation / Deployment

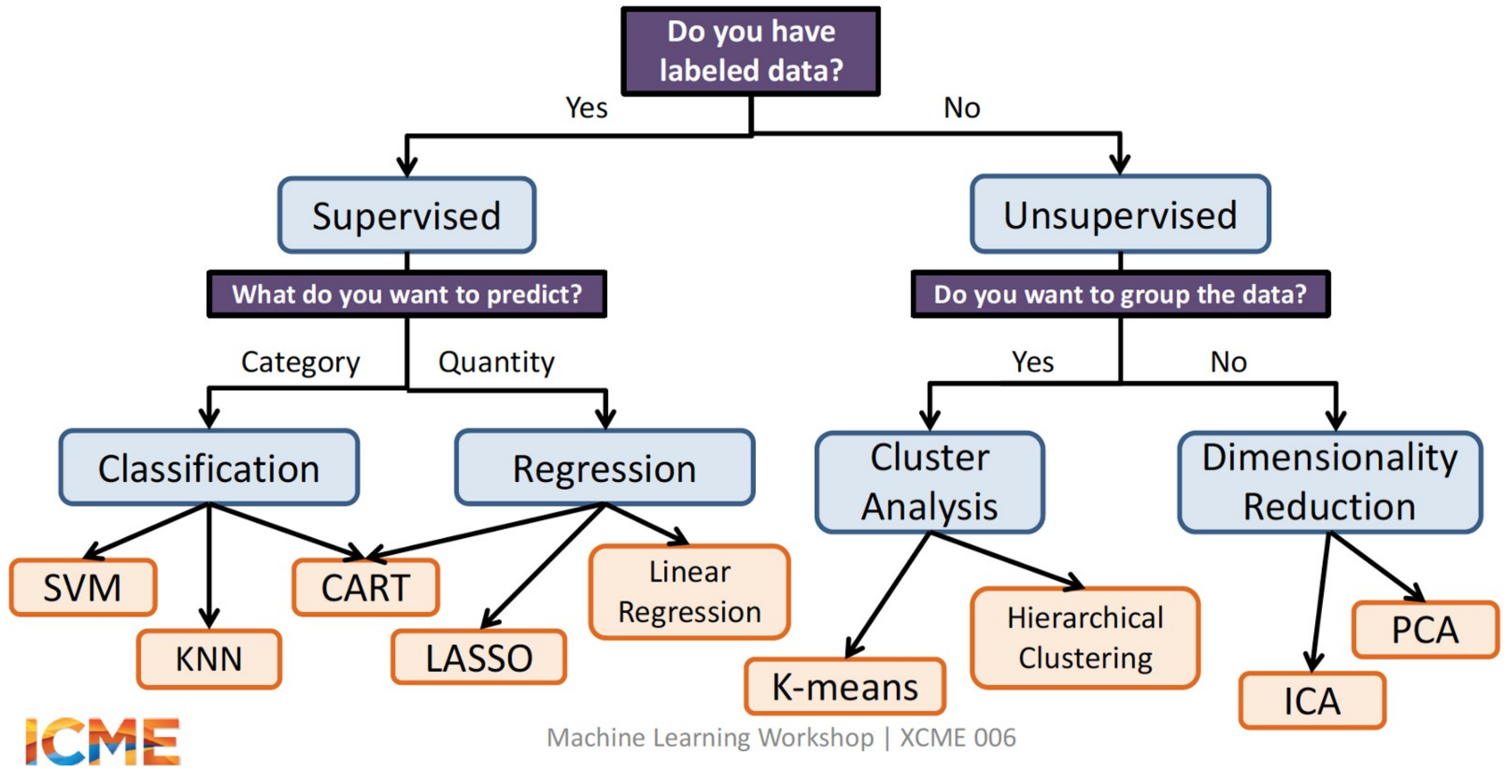
- Target Variable:
  - *Y (discrete/factor* or *continuous)*
- Predictive Model → $Y = f(X_1, X_2, \ldots, X_N)$



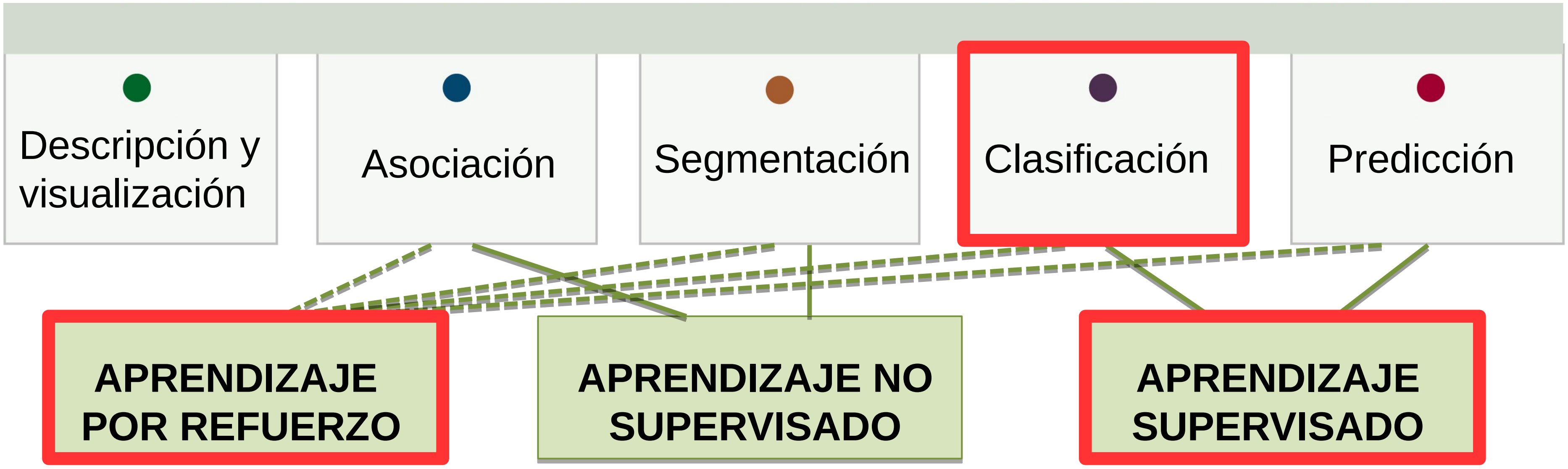
- There is no target variable:
  - *Association* or *segmentation*
- Predictive Model → Algorithmic

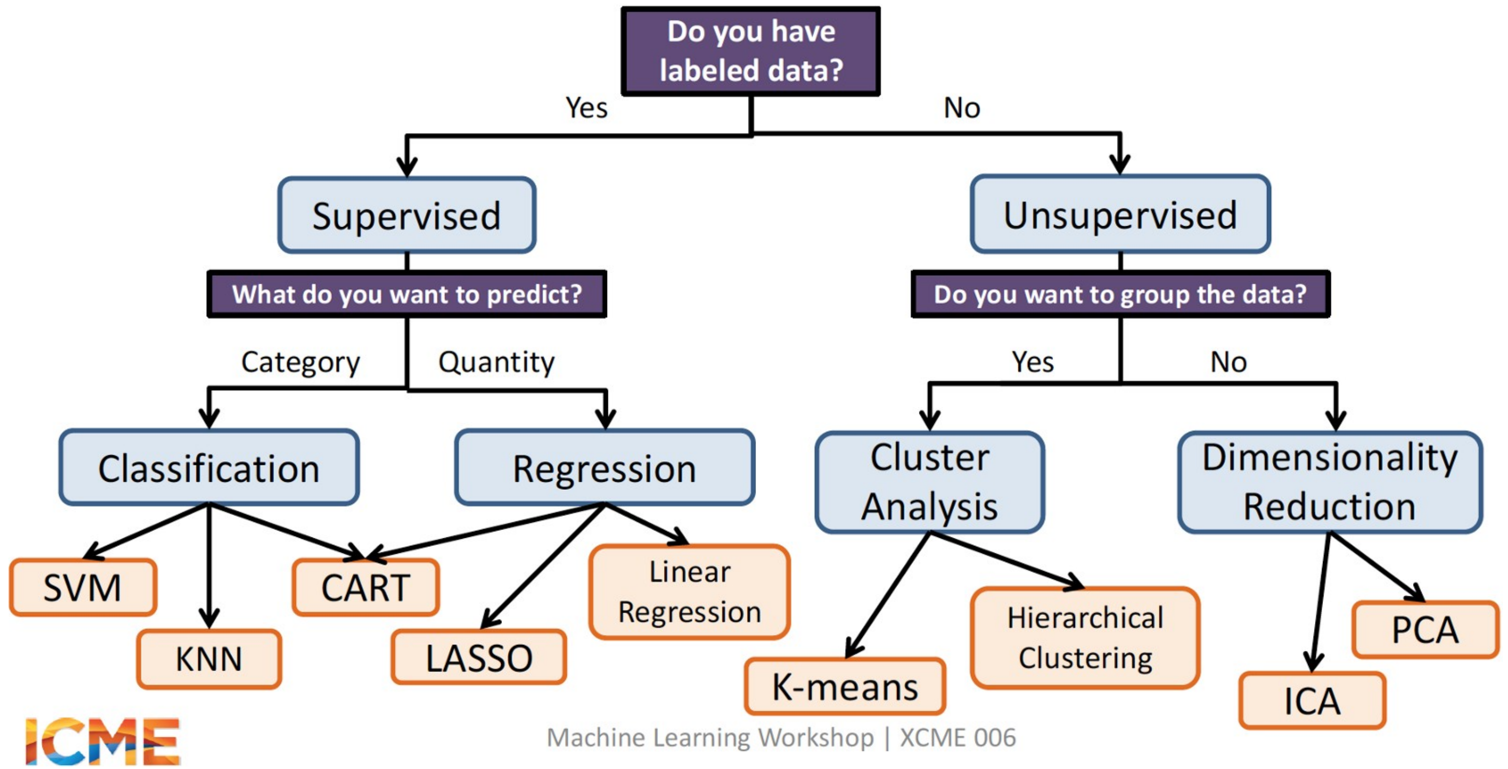




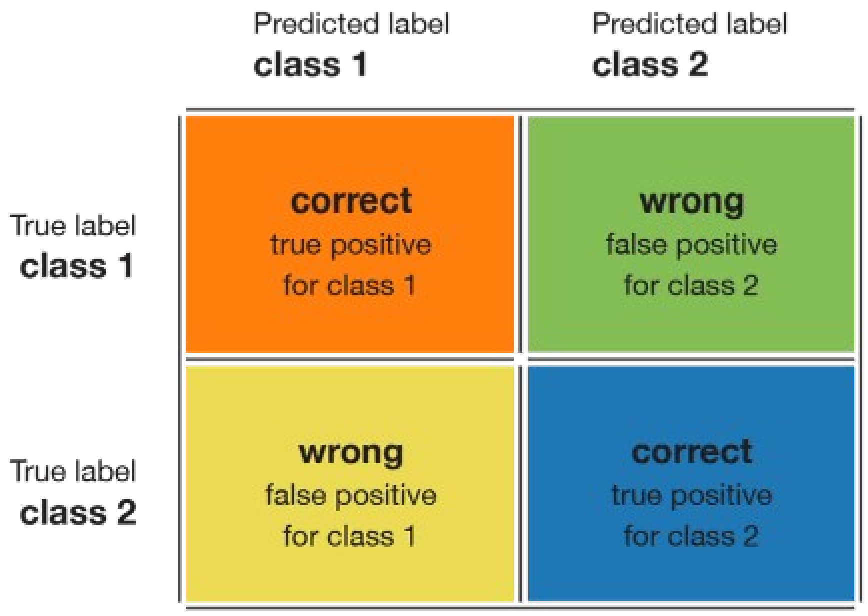


Machine Learning Workshop | XCME 006


Descripción y visualización
Asociación
Segmentación
Clasificación
Predicción
APRENDIZAJE POR REFUERZO
APRENDIZAJE NO SUPERVISADO
APRENDIZAJE SUPERVISADO


Do you have labeled data?
Yes
No
Supervised
Unsupervised
What do you want to predict?
Do you want to group the data?
Category
Quantity
Yes
No
Classification
Regression
Cluster Analysis
Dimensionality Reduction
SVM
KNN
CART
LASSO
Linear Regression
K-means
Hierarchical Clustering
PCA
ICA
ICME
Machine Learning Workshop | XCME 006

Confusion Matrix

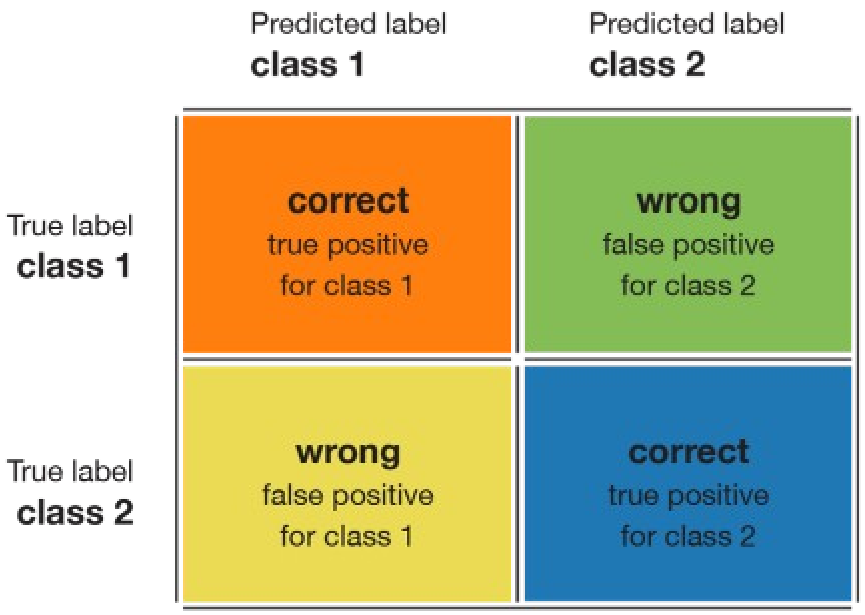


Predicted label class 1
Predicted label class 2
True label class 1
True label class 2
correct true positive for class 1
wrong false positive for class 2
wrong false positive for class 1
correct true positive for class 2

$$\text{accuracy} = \frac{🟧 + 🟦}{🟧 + 🟨 + 🟦 + 🟩}$$

Source: https://towardsdatascience.com/handling-imbalanced-datasets-in-machine-learning-7a0e84220f28

Confusion Matrix



	Predicted label class 1	Predicted label class 2
True label class 1	**correct** true positive for class 1	**wrong** false positive for class 2
True label class 2	**wrong** false positive for class 1	**correct** true positive for class 2

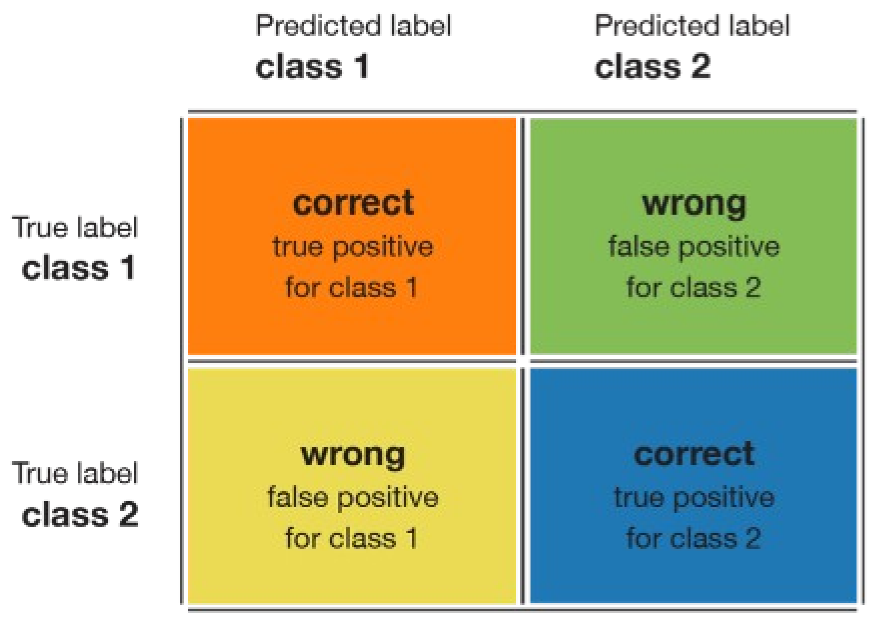
accuracy = (🟧 + 🟦) / (🟧 + 🟨 + 🟦 + 🟩)

Fatal Genetic Defect

10 out of every 100000 babies

Source: https://towardsdatascience.com/handling-imbalanced-datasets-in-machine-learning-7a0e84220f28

Confusion Matrix



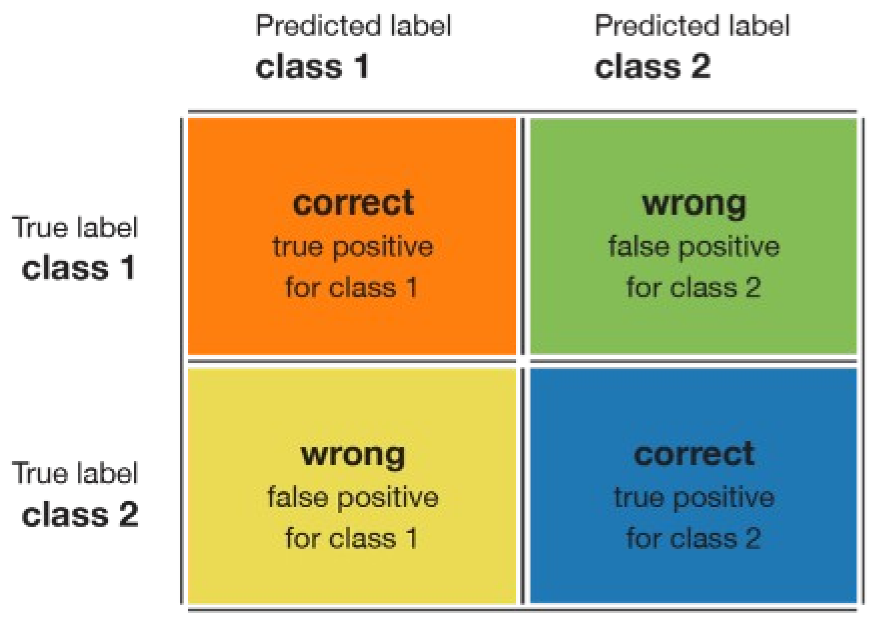
	Predicted label **class 1**	Predicted label **class 2**
True label **class 1**	**correct** true positive for class 1	**wrong** false positive for class 2
True label **class 2**	**wrong** false positive for class 1	**correct** true positive for class 2

accuracy = (🟧 + 🟦) / (🟧 + 🟨 + 🟦 + 🟩)

$$\begin{tikzcd}[column sep=small]
 & \textbf{Fatal Genetic Defect} \quad \textit{10 out of every 100000 babies} \arrow[d] & \\
 & \textbf{Model predicting always No Defect} \arrow[dl] \arrow[dr] & \\
\textbf{99.99\% TP} & & \textbf{100\% FN}
\end{tikzcd}$$

Source: https://towardsdatascience.com/handling-imbalanced-datasets-in-machine-learning-7a0e84220f28

Confusion Matrix


Predicted label class 1
Predicted label class 2
True label class 1
True label class 2
correct true positive for class 1
wrong false positive for class 2
wrong false positive for class 1
correct true positive for class 2

accuracy = (🟧 + 🟦) / (🟧 + 🟨 + 🟦 + 🟩)

Fatal Genetic Defect

10 out of every 100000 babies

↓

Model predicting always No Defect

99.99% TP **100% FN**

↓

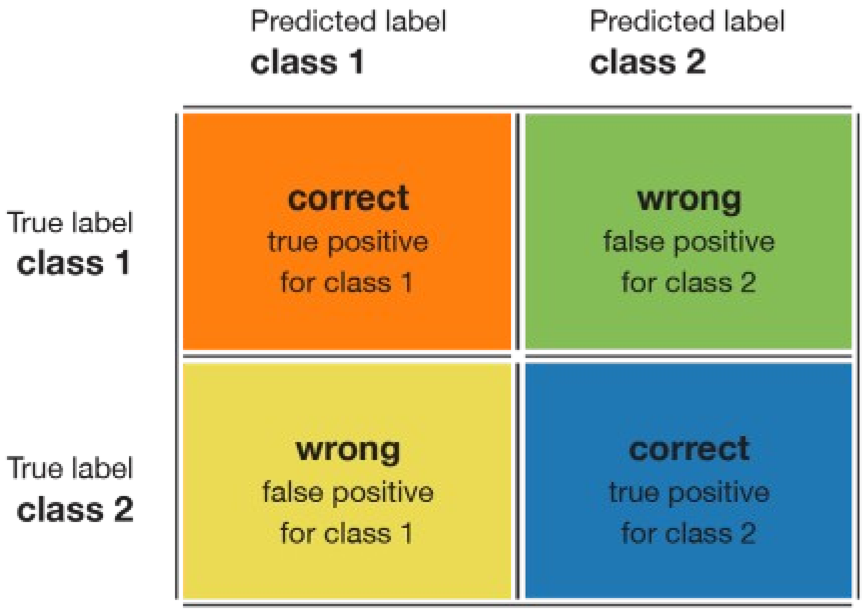
All the new born babies with the fatal genetic defect are wrongly predicted.

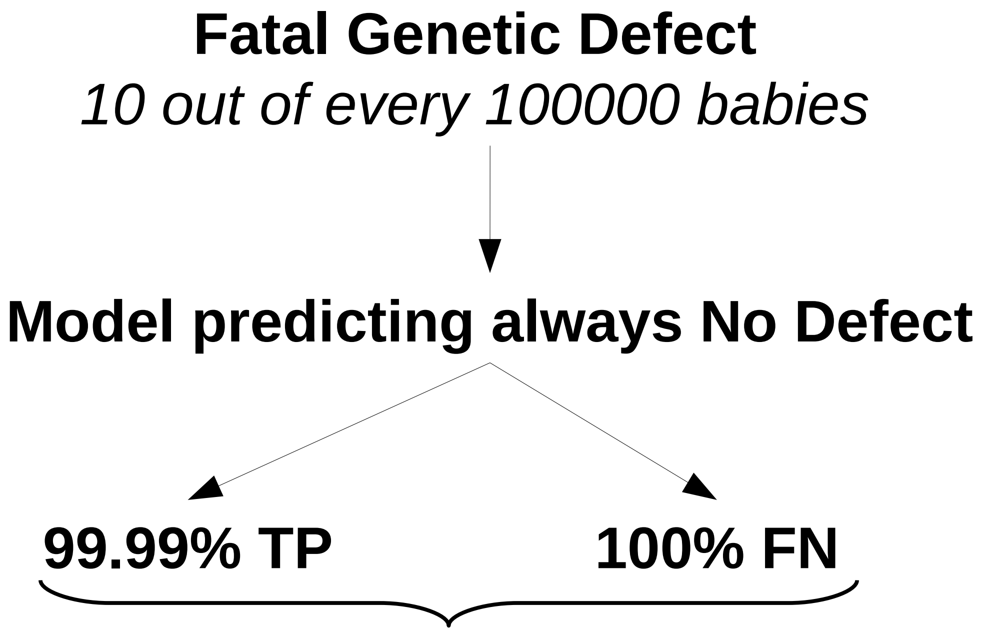
←

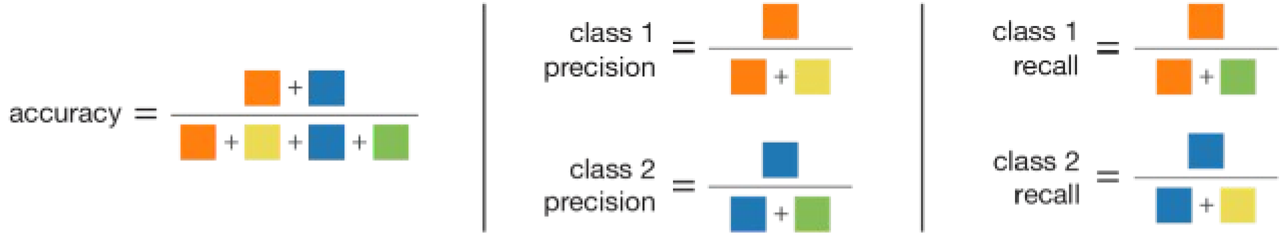
Are there any other validation measure?

Source: https://towardsdatascience.com/handling-imbalanced-datasets-in-machine-learning-7a0e84220f28

Confusion Matrix


Predicted label class 1
Predicted label class 2
True label class 1
True label class 2
correct true positive for class 1
wrong false positive for class 2
wrong false positive for class 1
correct true positive for class 2


Fatal Genetic Defect
10 out of every 100000 babies
Model predicting always No Defect
99.99% TP
100% FN

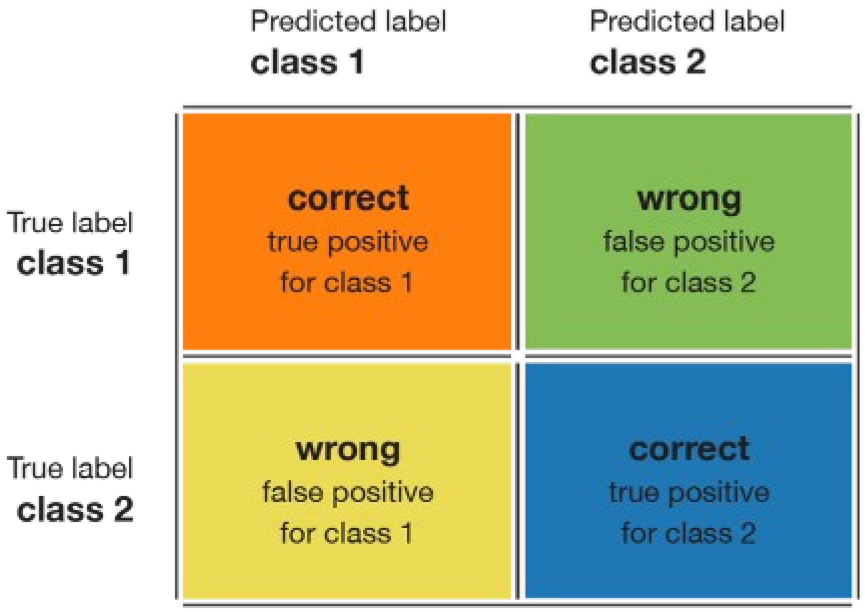

accuracy = (🟧 + 🟦) / (🟧 + 🟨 + 🟦 + 🟩)
class 1 precision = 🟧 / (🟧 + 🟨)
class 2 precision = 🟦 / (🟦 + 🟩)
class 1 recall = 🟧 / (🟧 + 🟩)
class 2 recall = 🟦 / (🟦 + 🟨)

Precision: *define how trustable is the result*

Recall: *expresses how well the model is able to detect that class*

Source: https://towardsdatascience.com/handling-imbalanced-datasets-in-machine-learning-7a0e84220f28

Confusion Matrix



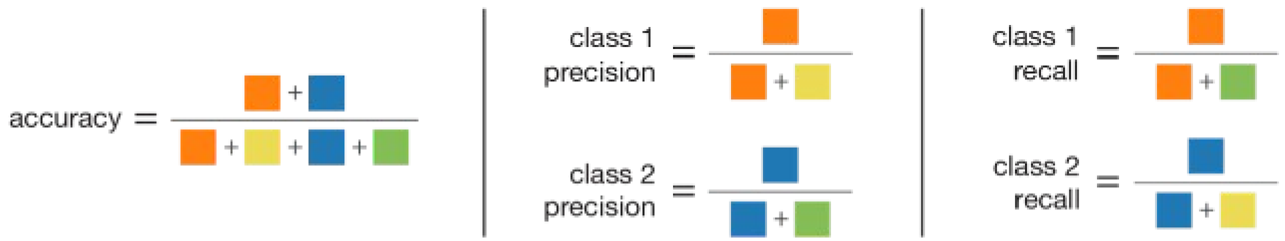
Predicted label class 1
Predicted label class 2
True label class 1
True label class 2
correct true positive for class 1
wrong false positive for class 2
wrong false positive for class 1
correct true positive for class 2

Fatal Genetic Defect

10 out of every 100000 babies

Model predicting always No Defect

$$\underbrace{\textbf{99.99\% TP} \qquad\qquad \textbf{100\% FN}}$$



accuracy = (🟧 + 🟦) / (🟧 + 🟨 + 🟦 + 🟩)
class 1 precision = 🟧 / (🟧 + 🟨)
class 2 precision = 🟦 / (🟦 + 🟩)
class 1 recall = 🟧 / (🟧 + 🟩)
class 2 recall = 🟦 / (🟦 + 🟨)

HR/HP: class is perfectly handled by the model

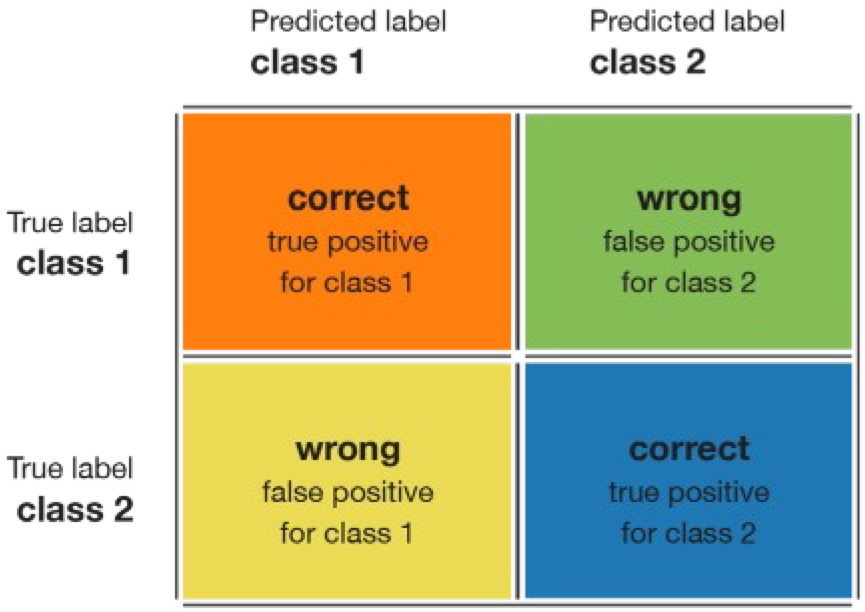
LR/HP: model can’t detect the class well but is highly trustable when it does

HR/LP: class is well detected but the model include points of other classes in it

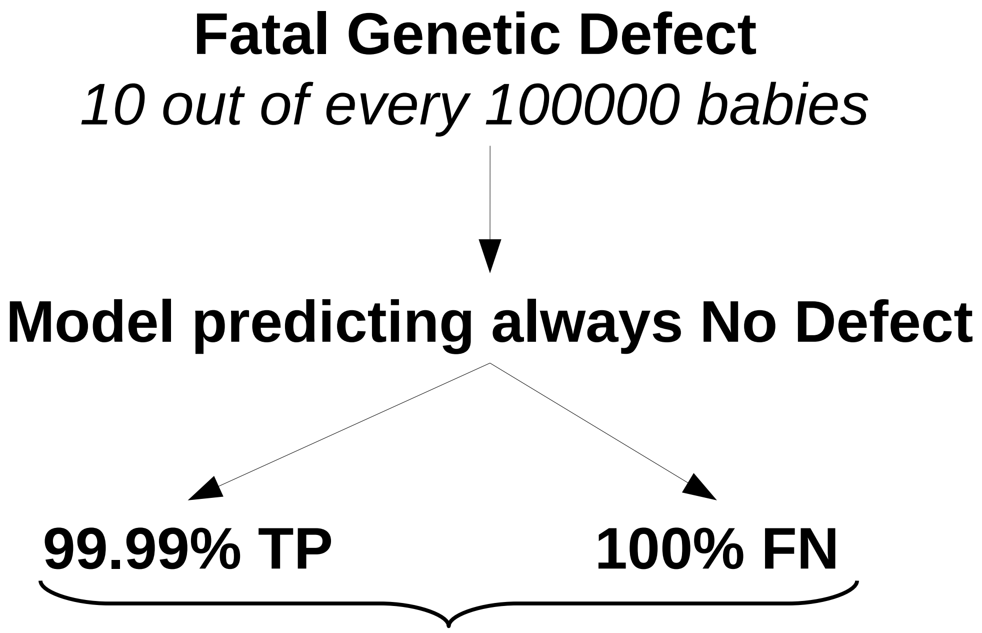
LR/LP: class is poorly handled by the model

Source: https://towardsdatascience.com/handling-imbalanced-datasets-in-machine-learning-7a0e84220f28

Confusion Matrix



	Predicted label **class 1**	Predicted label **class 2**
True label **class 1**	**correct** true positive for class 1	**wrong** false positive for class 2
True label **class 2**	**wrong** false positive for class 1	**correct** true positive for class 2

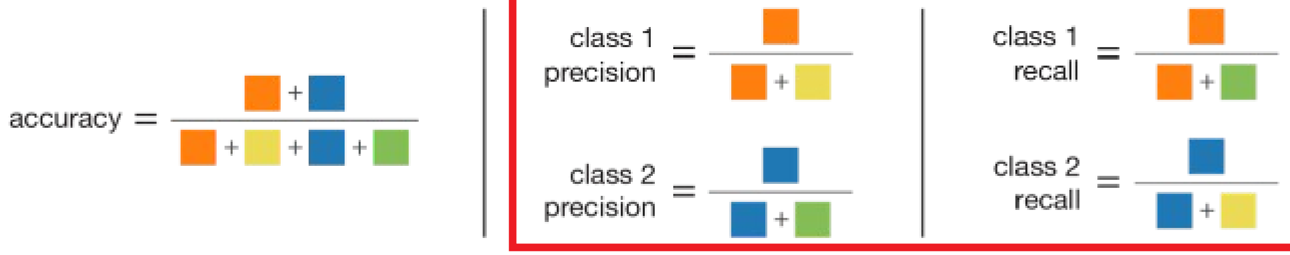


Fatal Genetic Defect

10 out of every 100000 babies

Model predicting always No Defect

$$\underbrace{\textbf{99.99\% TP} \qquad\qquad \textbf{100\% FN}}$$



accuracy = (🟧 + 🟦) / (🟧 + 🟨 + 🟦 + 🟩)

class 1 precision = 🟧 / (🟧 + 🟨)

class 1 recall = 🟧 / (🟧 + 🟩)

class 2 precision = 🟦 / (🟦 + 🟩)

class 2 recall = 🟦 / (🟦 + 🟨)

F1-Score:

$$\frac{2*precision*recall}{(precision+recall)}$$

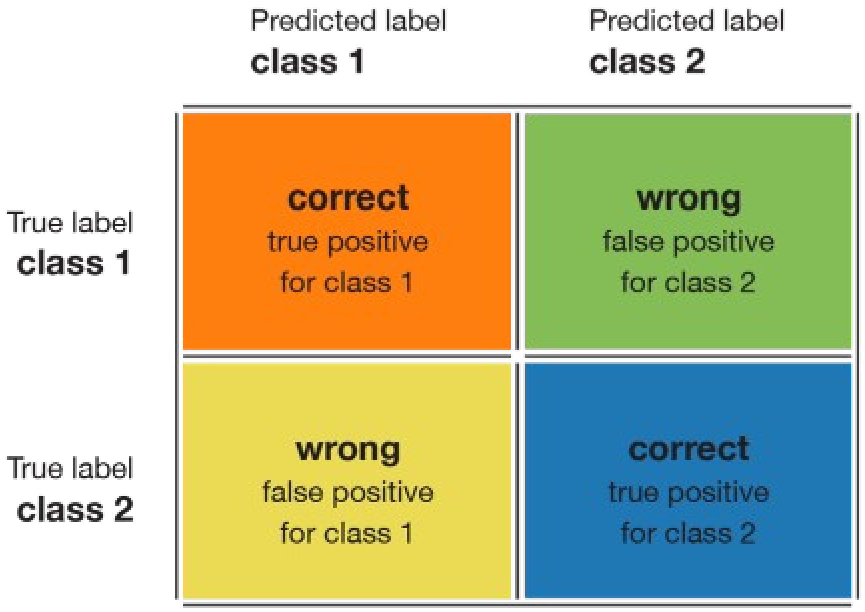
Precision: *define how trustable is the result*

Recall: *expresses how well the model is able to detect that class*

F1: *combines precision and recall of a class in one metric*

Source: https://towardsdatascience.com/handling-imbalanced-datasets-in-machine-learning-7a0e84220f28

Confusion Matrix



	Predicted label class 1	Predicted label class 2
True label class 1	**correct** true positive for class 1	**wrong** false positive for class 2
True label class 2	**wrong** false positive for class 1	**correct** true positive for class 2

accuracy = (🟧 + 🟦) / (🟧 + 🟨 + 🟦 + 🟩)

class 1 precision = 🟧 / (🟧 + 🟨)

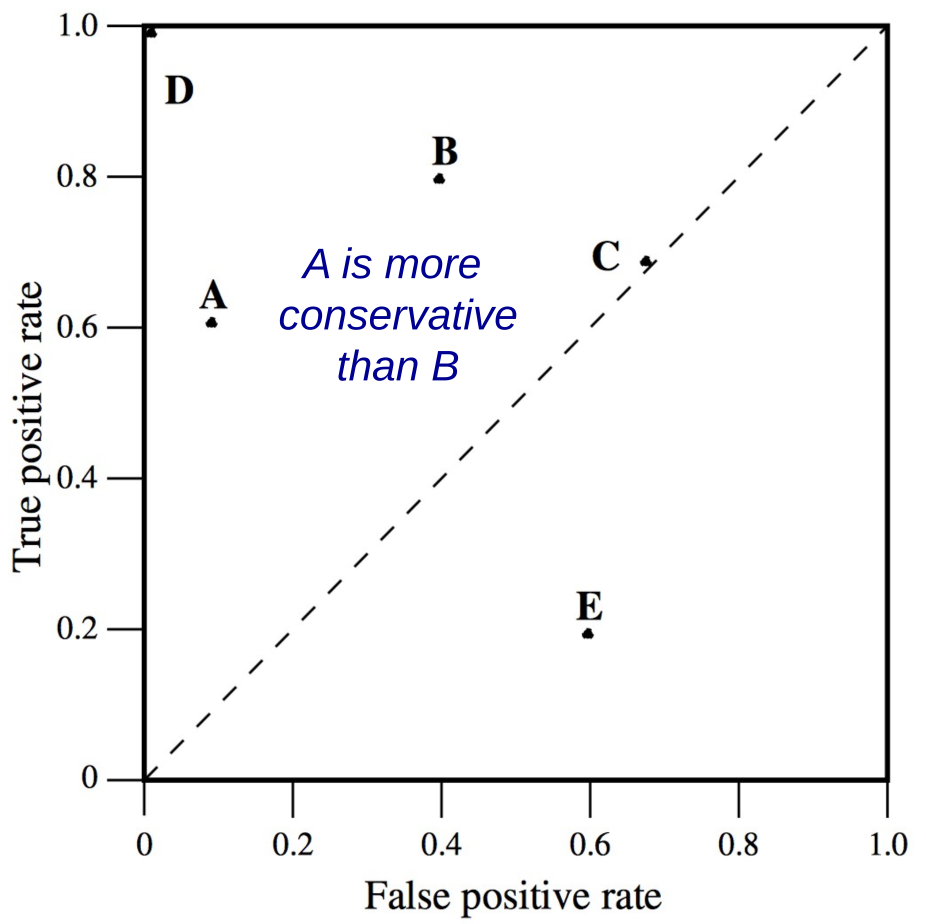
class 2 precision = 🟦 / (🟦 + 🟩)

class 1 recall =

class 2 recall =

False Alarm Rate (**FAR**) Hit Rate (**HIR**)

$$\text{fp rate} = \frac{FP}{N} \qquad \text{tp rate} = \frac{TP}{P}$$



Which systems yield?

HIR = FAR = 0 → Never predicting

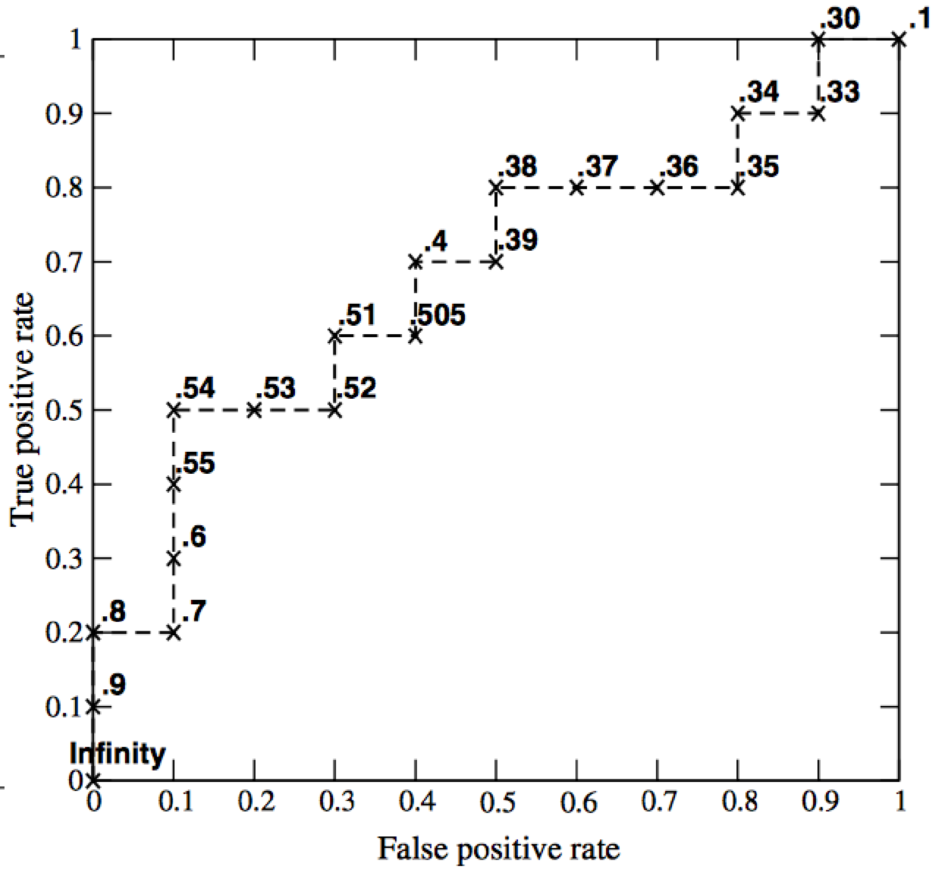
HIR = FAR = 1 → Always predicting

Fawcett, T. (2006) An introduction to ROC analysis, In Pattern Recognition Letters, 27, 861-874, https://doi.org/10.1016/j.patrec.2005.10.010.

Summarizes the performance of the system over all possible probability thresholds.

```
library(pROC)
obs<-c(rep(0,50),rep(1,50));
prd<-obs+2*(runif(100)-0.5);
prd[which(prd<0)]<-0; prd[which(prd>1)]<-1;
plot(roc(obs,prd), print.auc=TRUE)
hist(prd)
```

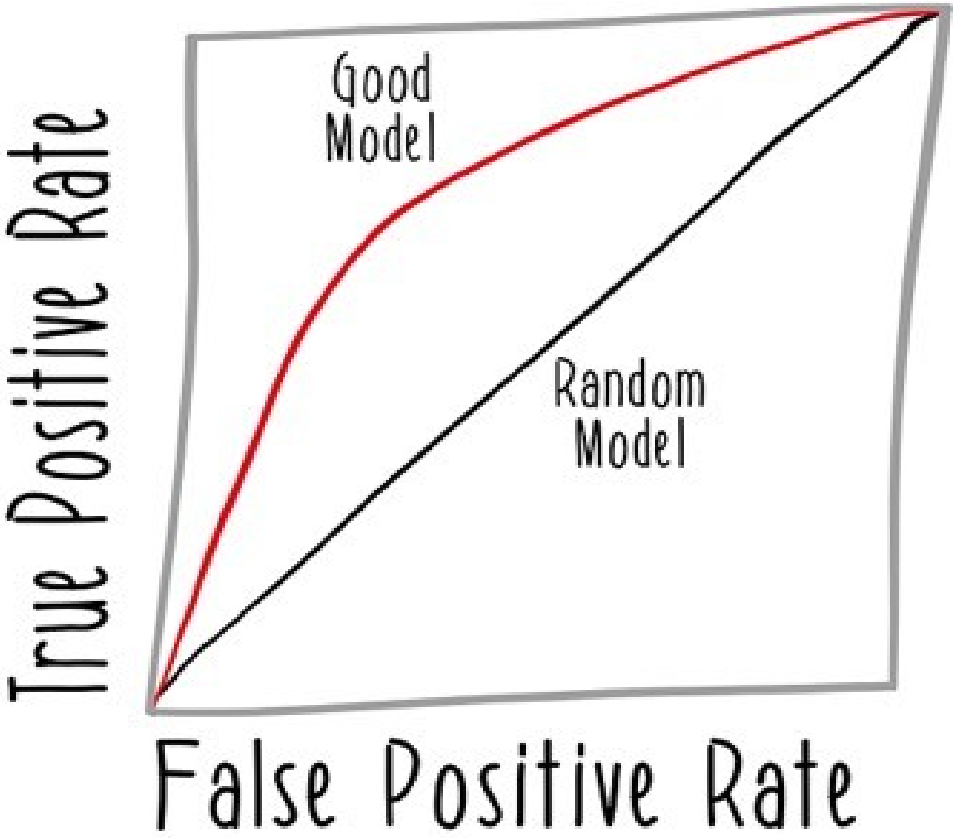
Inst#	Class	Score	Inst#	Class	Score
1	p	.9	11	p	.4
2	p	.8	12	n	.39
3	n	.7	13	p	.38
4	p	.6	14	n	.37
5	p	.55	15	n	.36
6	p	.54	16	n	.35
7	n	.53	17	p	.34
8	n	.52	18	n	.33
9	p	.51	19	p	.30
10	n	.505	20	n	.1



Summarizes the performance of the system over all possible probability thresholds.

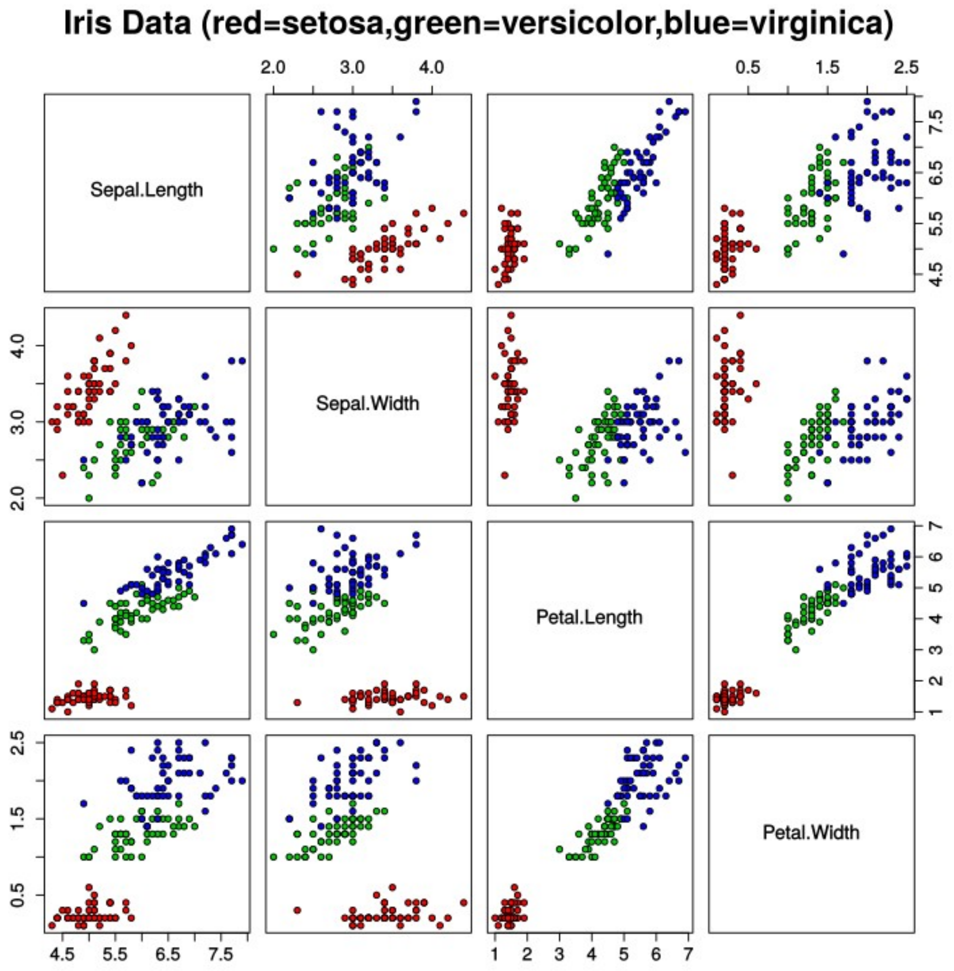
```
library(pROC)
obs<-c(rep(0,50),rep(1,50));
prd<-obs+2*(runif(100)-0.5);
prd[which(prd<0)]<-0; prd[which(prd>1)]<-1;
plot(roc(obs,prd), print.auc=TRUE)
hist(prd)
```

Inst#	Class	Score	Inst#	Class	Score
1	p	.9	11	p	.4
2	p	.8	12	n	.39
3	n	.7	13	p	.38
4	p	.6	14	n	.37
5	p	.55	15	n	.36
6	p	.54	16	n	.35
7	n	.53	17	p	.34
8	n	.52	18	n	.33
9	p	.51	19	p	.30
10	n	.505	20	n	.1



https://www.kdnuggets.com/2018/01/machine-learning-model-metrics.html

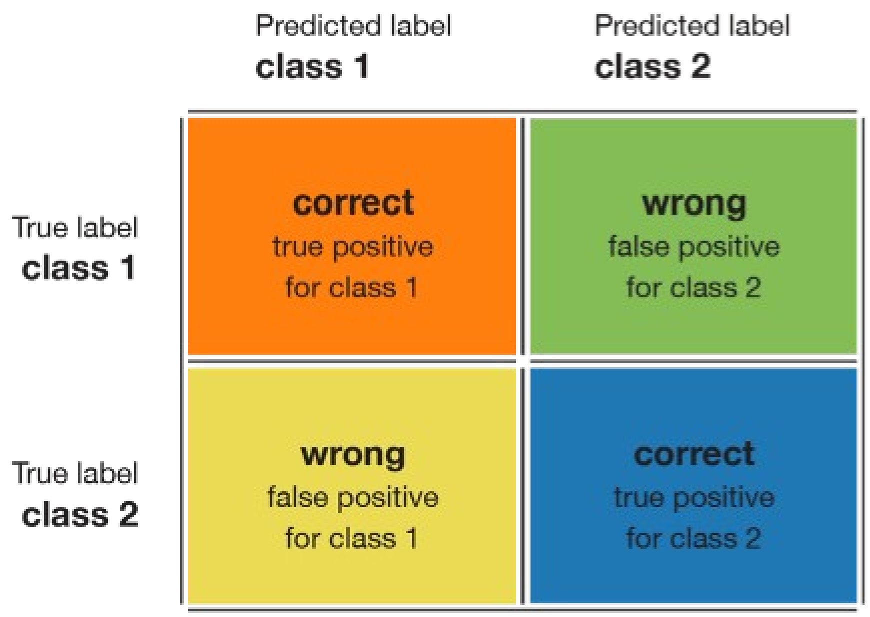
```
data(iris)
fitControl <- trainControl(method="none",
                           number=1,
                           repeats=1,
                           verboseIter=TRUE)
modelFit <- train(Species ~ ., data=iris, method="knn", trControl=fitControl)
pred <- predict(modelFit, newdata = iris[,-5])
acc<-confusionMatrix(iris$Species,pred)
print(acc)
```



Confusion Matrix

Fatal Genetic Defect

10 out of every 100000 babies

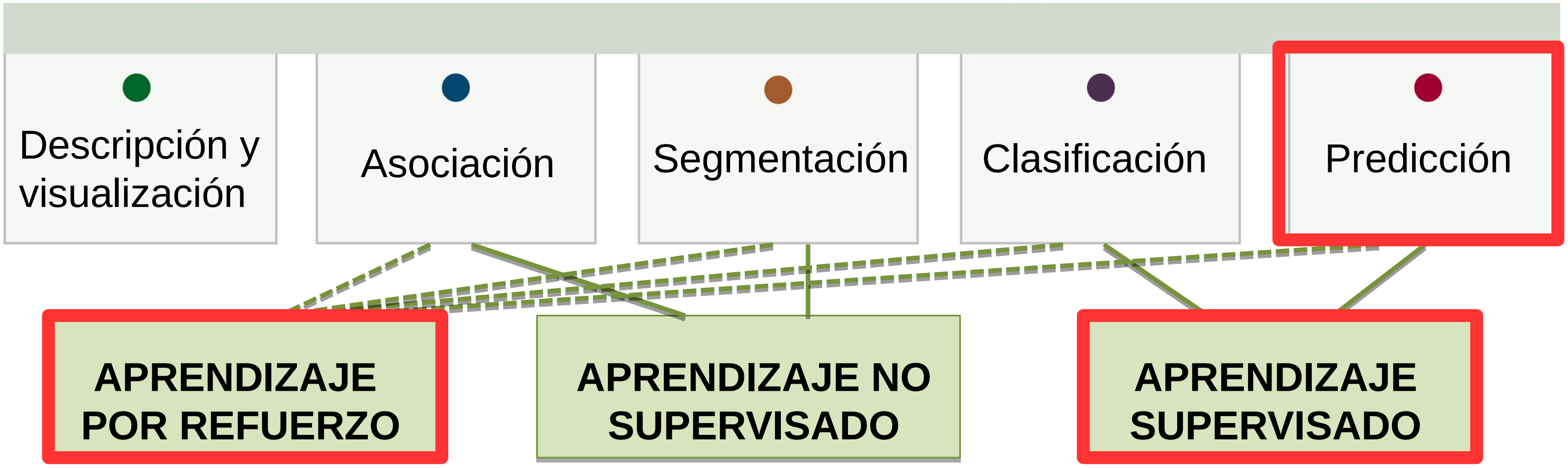

Predicted label class 1
Predicted label class 2
True label class 1
True label class 2
correct true positive for class 1
wrong false positive for class 2
wrong false positive for class 1
correct true positive for class 2

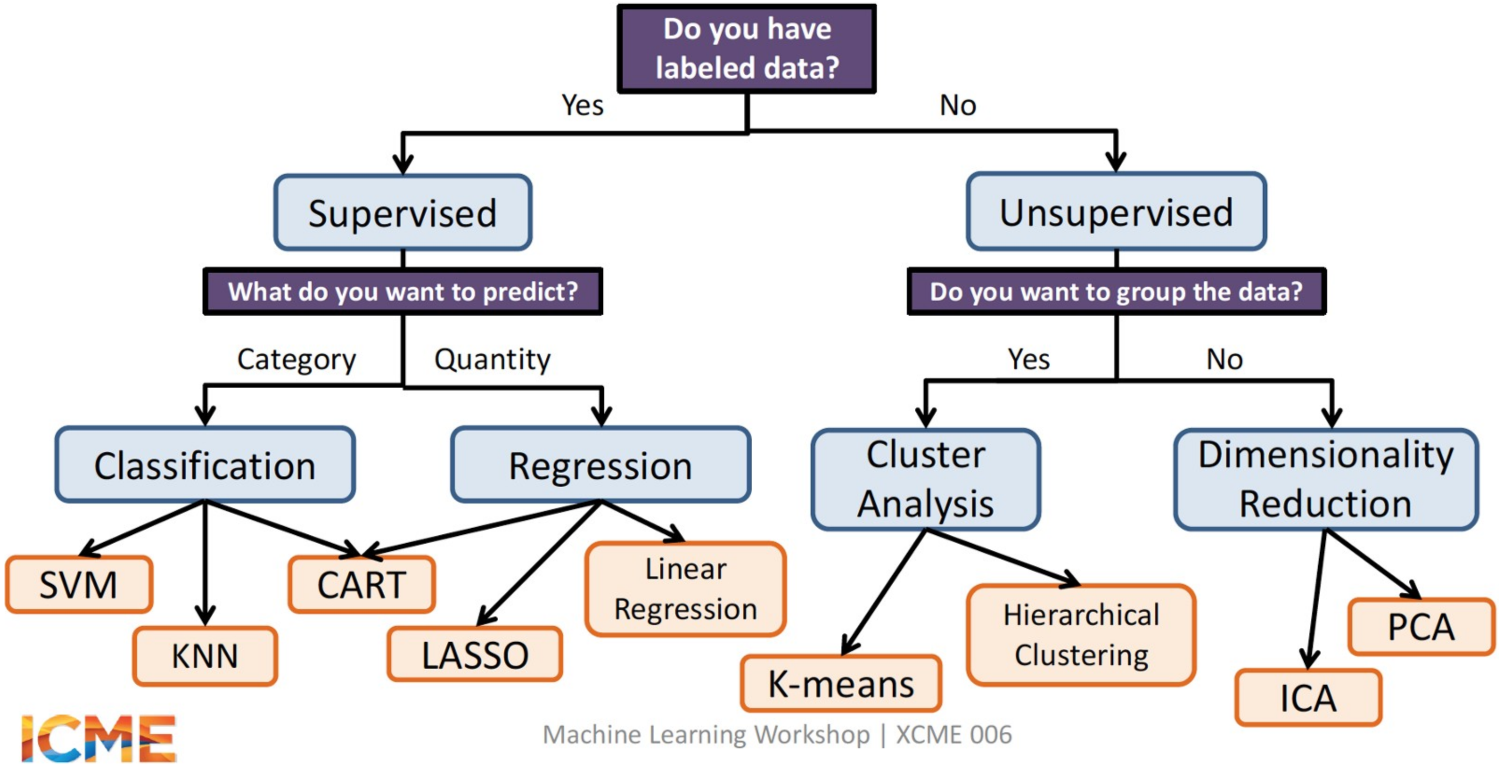
accuracy = $\frac{🟧 + 🟦}{🟧 + 🟨 + 🟦 + 🟩}$

Dealing with unbalanced data in machine learning

https://shiring.github.io/machine_learning/2017/04/02/unbalanced

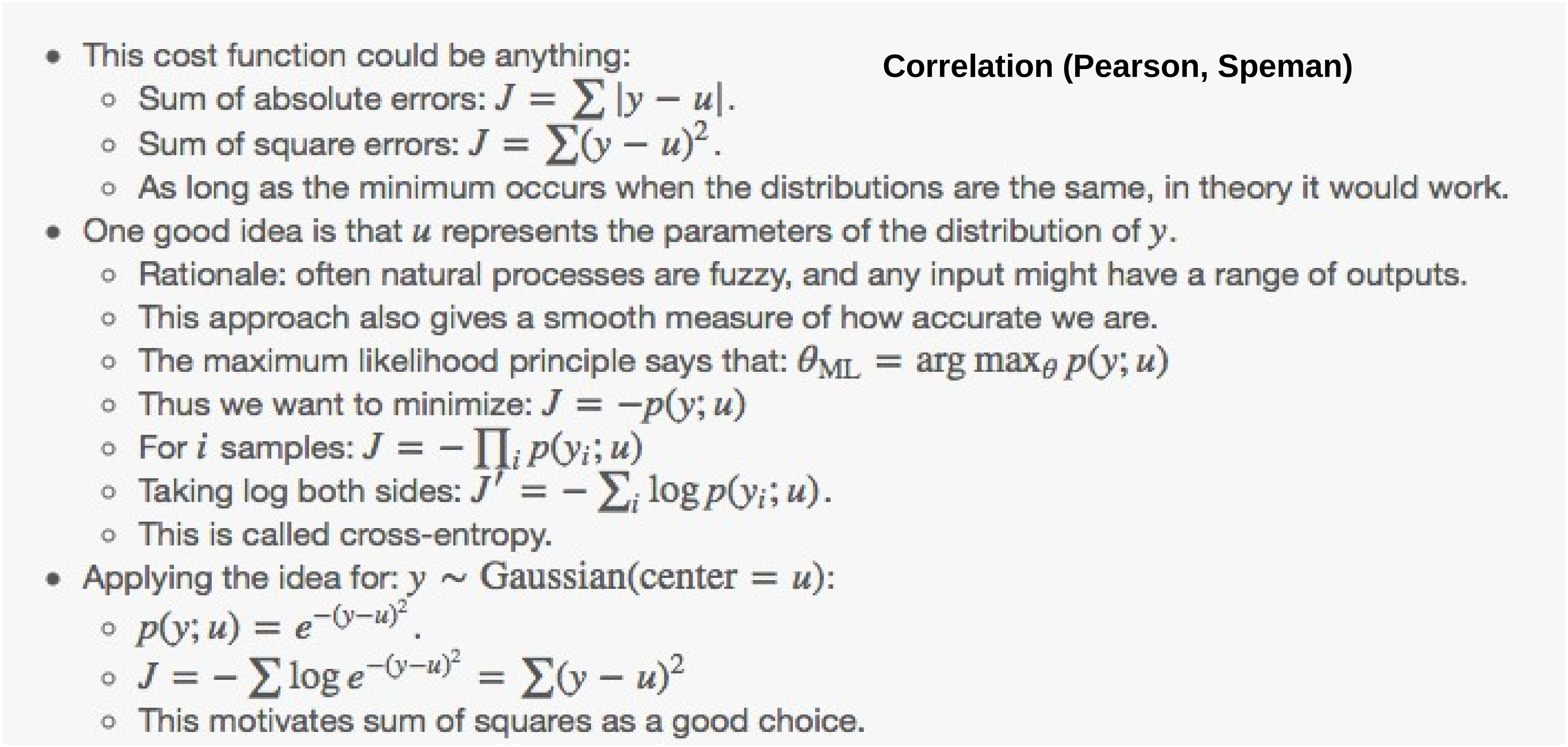
Source: https://towardsdatascience.com/handling-imbalanced-datasets-in-machine-learning-7a0e84220f28


Descripción y visualización
Asociación
Segmentación
Clasificación
Predicción
APRENDIZAJE POR REFUERZO
APRENDIZAJE NO SUPERVISADO
APRENDIZAJE SUPERVISADO


Do you have labeled data?
Yes
No
Supervised
Unsupervised
What do you want to predict?
Do you want to group the data?
Category
Quantity
Yes
No
Classification
Regression
Cluster Analysis
Dimensionality Reduction
SVM
KNN
CART
LASSO
Linear Regression
K-means
Hierarchical Clustering
PCA
ICA
ICME
Machine Learning Workshop | XCME 006

Model accuracy (training and validation).

Some models are trained using an **empirical error (cost) function**, which measures **model accuracy** as the difference between the predicted and the actual value. In this cas, this a natural **validation measure**.



Correlation (Pearson, Speman)

- This cost function could be anything:
  - Sum of absolute errors: $J = \sum |y - u|$.
  - Sum of square errors: $J = \sum (y - u)^2$.
  - As long as the minimum occurs when the distributions are the same, in theory it would work.
- One good idea is that u represents the parameters of the distribution of y.
  - Rationale: often natural processes are fuzzy, and any input might have a range of outputs.
  - This approach also gives a smooth measure of how accurate we are.
  - The maximum likelihood principle says that: $\theta_{\mathrm{ML}} = \arg\max_\theta p(y; u)$
  - Thus we want to minimize: $J = -p(y; u)$
  - For i samples: $J = -\prod_i p(y_i; u)$
  - Taking log both sides: $J' = -\sum_i \log p(y_i; u)$.
  - This is called cross-entropy.
- Applying the idea for: $y \sim \mathrm{Gaussian}(\mathrm{center} = u)$:
  - $p(y; u) = e^{-(y-u)^2}$.
  - $J = -\sum \log e^{-(y-u)^2} = \sum (y - u)^2$
  - This motivates sum of squares as a good choice.

Model performance: Validation diagnostics and metrics.

There are several domain-dependent diagnostics (computed separately for prediction ‘p’ and observation ‘o’) and metrics/errors for validating model performance.

Distributional consistency: evaluates the model capability to reproduce the distribution of the observed data.

- **Bias** = mean p – mean o
- **Variance ratio** = var p / var o
- **Distributional similarity:** ks-score, Von Misses, pdf-score, etc.

The **quantile-quantile plot** is a typical tool to evaluate, in a graphical way, the distributional similarity of the order statistics (e.g. **percentiles**).

Different diagnostics for different fields.

Accuracy: assess the correspondence of the simulated and observed sequences. Two typical scores are usally used: Root Mean Square Error (**RMSE**) and the (Pearson/Spearman/Kendall) **Correlation**.

$$RMSE=\sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i-y_i)^2}$$

$$r=\frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i-\bar{y})^2}}$$

Distributional consistency: evaluates the model capability to reproduce the distribution of the observed data. The most popular are the **bias** (mean difference) or the **ratio of variances/standard deviation**. In addition, there are hypothesis tests to evaluate in a global way the similarity of the observed and simulated series (e.g. **Kolmogorov-Smirnov**, **Perkins**, **Von Misses**, etc).

The **quantile-quantile plot** is a typical tool to evaluate, in a graphical way, the distributional similarity of the order statistics (e.g. **percentiles**).

```
## Example with R:
?qqplot
require(graphics)
y<-rt(200,df=5)
qqnorm(y)
qqline(y,col=2)
qqplot(y,rt(300,df=5))
```

Accuracy: assess the correspondence of the simulated and observed sequences. Two typical scores are usally used: Root Mean Square Error (**RMSE**) and the (Pearson/Spearman/Kendall) **Correlation**.

$$RMSE=\sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i-y_i)^2} \qquad r=\frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i-\bar{y})^2}}$$

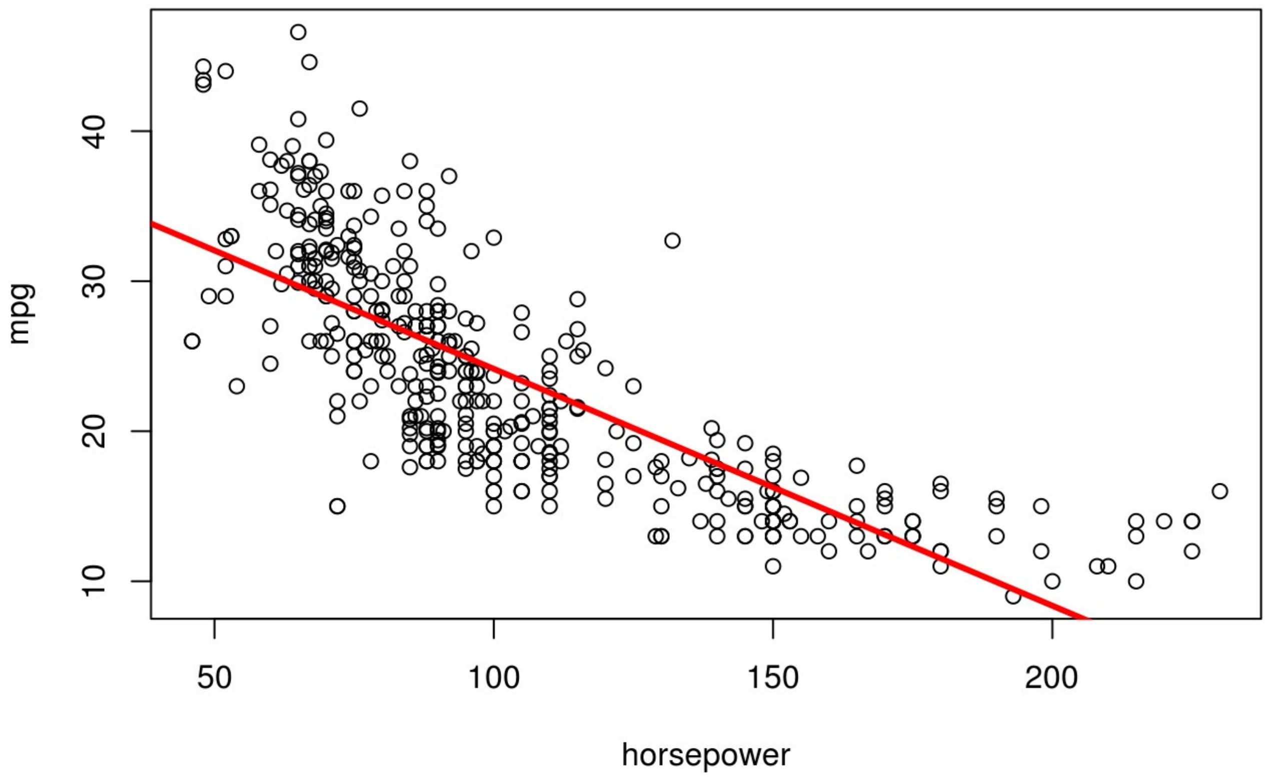
Distributional consistency: evaluates the model capability to reproduce the distribution of the observed data. The most popular are the **bias** (mean difference) or the **ratio of variances/standard deviation**. In addition, there are hypothesis tests to evaluate in a global way the similarity of the observed and simulated series (e.g. **Kolmogorov-Smirnov**, **Perkins**, **Von Misses**, etc).

The **quantile-quantile plot** is a typical tool to evaluate, in a graphical way, the distributional similarity of the order statistics (e.g. **percentiles**).

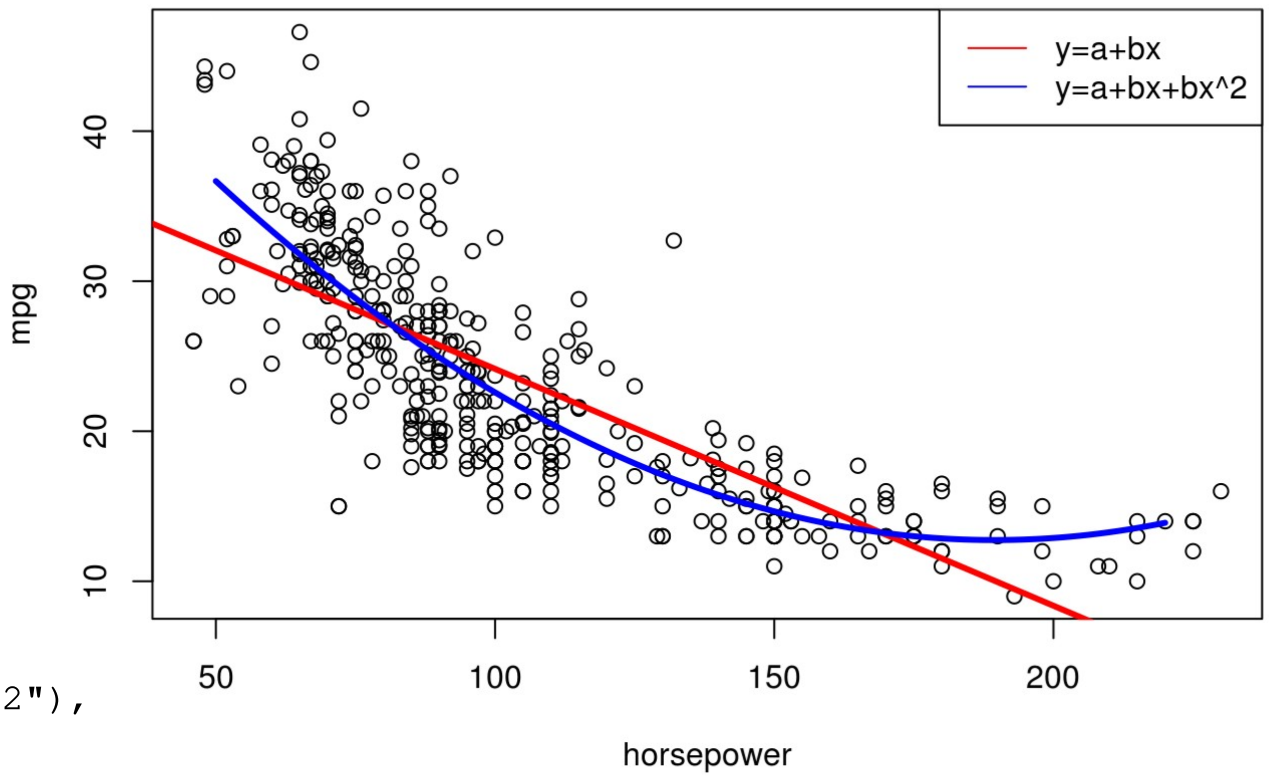
```
## Example with R:
?qqplot
require(graphics)
y<-rt(200,df=5)
qqnorm(y)
qqline(y,col=2)
qqplot(y,rt(300,df=5))
```

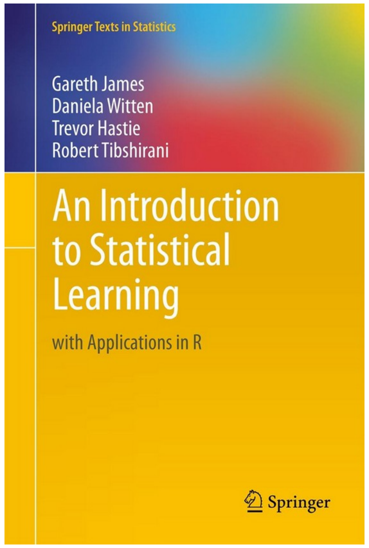
$\rightarrow$ **How to create and use our own functions, including validation measures in R?**

```
# install.packages('ISLR')
library(ISLR)
attach(Auto)
summary(Auto)
n <- length(mpg)
plot(horsepower, mpg)
lm1 <- lm(mpg~horsepower)
abline(lm1, col="red", lwd=3)
summary(lm1)
```



```
# install.packages('ISLR')
library(ISLR)
attach(Auto)
summary(Auto)
n <- length(mpg)
plot(horsepower, mpg)
lm1 <- lm(mpg~horsepower)
abline(lm1, col="red", lwd=3)
summary(lm1)
plot(horsepower, mpg)
abline(lm1, col="red", lwd=3)
lm2 <- lm(mpg~poly(horsepower,2))
xs <- seq(50,220,length=100)
ys <- predict(lm2,
    data.frame(horsepower=xs))
lines(xs,ys, type="l",
    lwd=3, col="blue")
legend("topright",
    legend=c("y=a+bx","y=a+bx+bx^2"),
    lty=1 , col=c("red", "blue"))
summary(lm2)
```





An Introduction to Statistical Learning: With Applications in R

James, G., Witten, D., Hastie, T., Tibshirani, R.

Springer (2013)

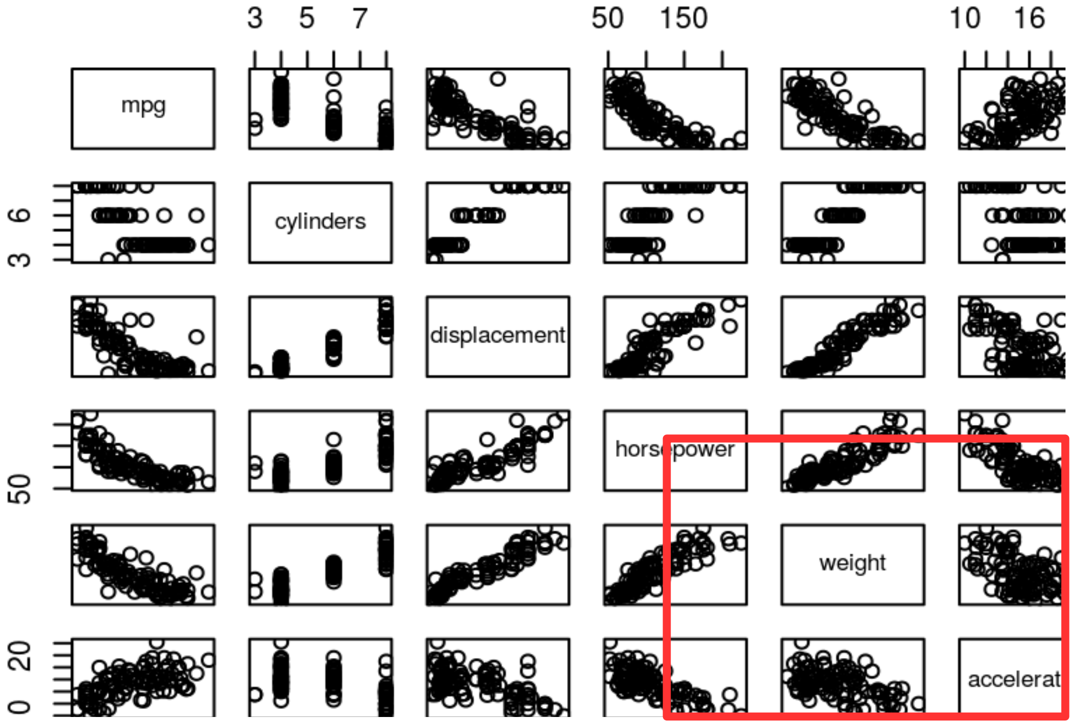
http://www-bcf.usc.edu/~gareth/ISL

```
install.packages("ISLR")
library("ISLR")
library(help = "ISLR")
```

```
> data(Auto)
> str(Auto)
```

```
'data.frame':    392 obs. of  9 variables:
 $ mpg         : num  18 15 18 16 17  ...
 $ cylinders   : num  8 8 8 8 8 8 8 8 ...
 $ displacement: num  307 350 318 304 ...
 $ horsepower  : num  130 165 150 150 ...
 $ weight      : num  3504 3693 3436  ...
 $ acceleration: num  12 11.5 11 12   ...
 $ year        : num  70 70 70 70 70  ...
 $ origin      : num  1 1 1 1 1 1 1 1 ...
 $ name        : Factor w/ 304 levels ...
```

```
> pairs(Auto)
```



CARET (ClAssification and REgresion Training) is a wrapper of a number of standard machine learning packages which performs model tunning (optimization of the model parameters) and cross-validation strategies.
`http://topepo.github.io/caret/index.html`

```
> modelLookup(model = "lm")
  model parameter     label forReg forClass probModel
     lm intercept intercept   TRUE    FALSE     FALSE
```

```
trainControl(method , number, …)
    method: "none", "cv", "LOOCV"
    number: For "cv" (2 => hold-out, 10 => 10-fold)
```

```
> ctrl <- trainControl(method = "LOOCV")
> mod <- train(weight ~ horsepower,
               data = Auto,
               method = "lm",
               trControl = ctrl)
          # metric="RMSE",
          # preProc = c("center", "scale")
```

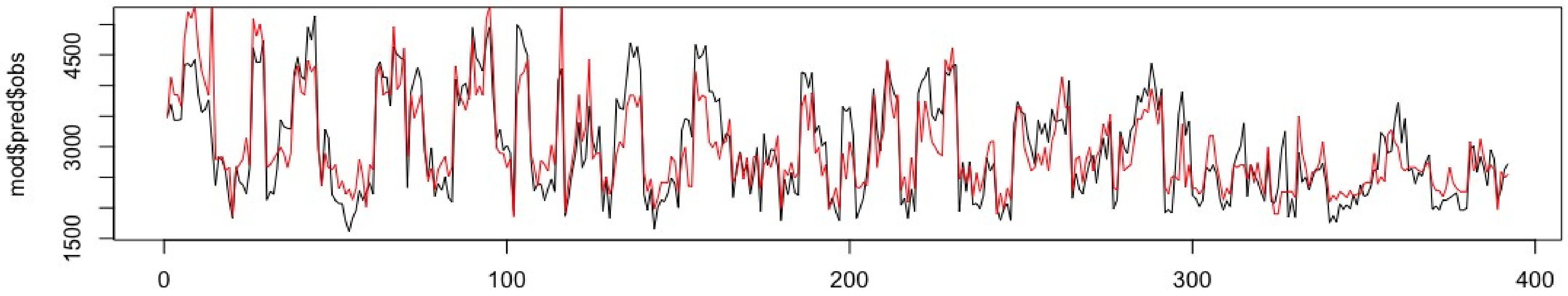
```
> mod
```

```
Linear Regression | 392 samples | 1 predictor | No pre-
processing
Resampling: Leave-One-Out Cross-Validation
Summary of sample sizes: 391, 391, 391, 391, 391, 391, ...
Resampling results:
  RMSE      Rsquared   MAE
  429.5254  0.7436498  347.5039
```

```
> str(model$control$index$Fold001)
```

```
int [1:391] 2 3 4 5 6 7 8 9 10 11 ...
```

```
> plot(mod$pred$obs, type="l");
       lines(1:392,mod$pred$pred,col="red")
```


mod$pred$obs