



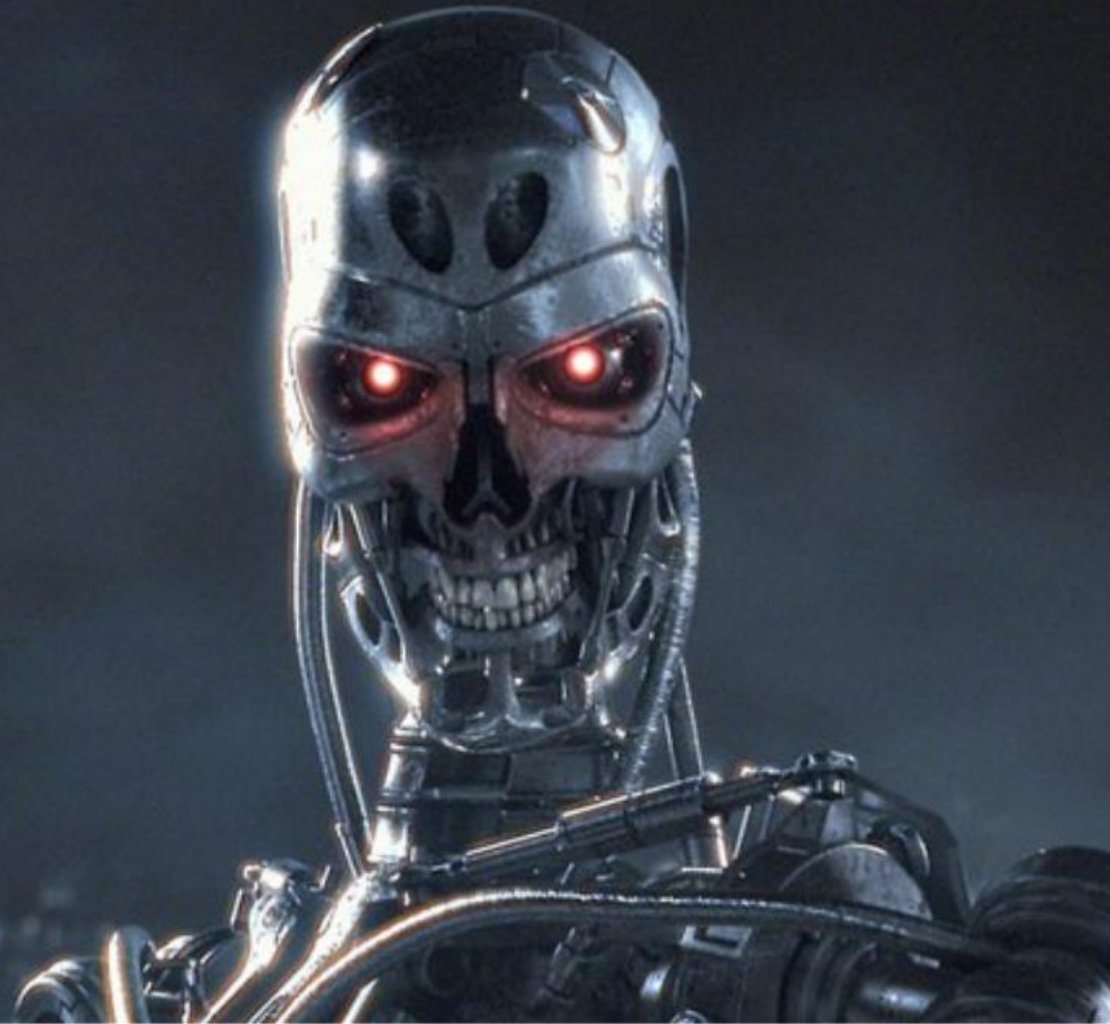
# Aprendizaje de Máquina y Minería de Datos Clase I

Curso Exploratorio de Computación – IIC 1005  
2019-2

# PLAN SEMESTRAL

Week	Fecha semana	Martes	Jueves	Clase Martes	Clase Jueves	Ayudantía
I	5-ago.	6-ago.	8-ago.	Introduccion	Leng. Prog +github+terminal+jupyter	
II	12-ago.	13-ago.	Feriado 15-08-2019	Tecn Web HTML + CSS	Feriado	Git/ Dudas TC1
III	19-ago.	20-ago.	22-ago.	Tecn Web JS	Arquitectura de Computadores	Web
IV	26-ago.	27-ago.	29-ago.	Sistemas Operativos	Redes	JS
V	2-sept.	3-sept.	5-sept.	Bases de Datos	Bases de Datos	
VI	9-sept.	10-sept.	12-sept.	Algoritmos	Ingenieria de Sotware	SQL
VII	16-sept.	17-sept.	Feriado 19-09-2019	Previa- Feriado	Feriado	
VIII	23-sept.	24-sept.	26-sept.	Machine Learning	Machine Learning	
IX	30-sept.	1-oct.	3-oct.	Machine Learning	Visualizacion de Información	ML
X	7-oct.	8-oct.	10-oct.	Computabilidad	Complejidad	Dudas TG2
XI	14-oct.	15-oct.	17-oct.	Programación Logica	Programación Logica	
XII	21-oct.	22-oct.	24-oct.	BPM	BPM II	Turing
XIII	28-oct.	29-oct.	Feriado 31-10-2019	Criptomonedas	Feriado	BPMN
XIV	4-nov.	5-nov.	7-nov.	Guest: CSCW-HCI	MOOC	
XV	11-nov.	12-nov.	14-nov.	Innovación y emprendimiento	Guest: Mobile & Cloud	
XVI	18-nov.	19-nov.	21-nov.	Guest: Educ. TI	Resumen Final	

# Inteligencia Artificial



# Inteligencia Artificial

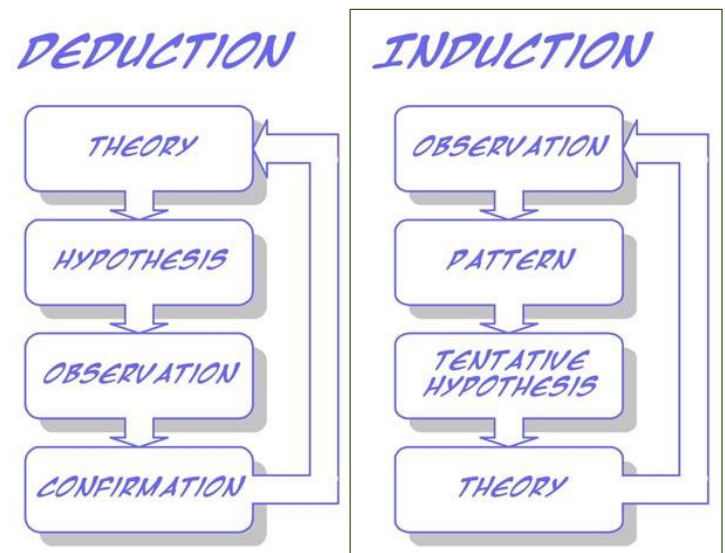
- **Objetivo:** Construir máquinas/software que exhiban comportamiento inteligente.
- Algunas aplicaciones comunes: percepción visual, reconocimiento del habla, toma de decisiones, traducción entre lenguajes.

# Inteligencia Artificial

- El dominio de problemas en el área de Inteligencia Artificial incluye:
  - Representar conocimiento
  - Razonamiento
  - Planning
- En esta clase me enfocaré en métodos estadísticos usados para aprendizaje, lo que se conoce como **Machine Learning**, y en el procedimiento de uso de estos algoritmos para encontrar patrones en colecciones de datos (**Data Mining**)

# Aprendizaje de Máquina (Machine Learning)

- Estudio de algoritmos computacionales que aprenden y mejoran automáticamente a través de la experiencia.
- Algunos investigadores lo llaman también Aprendizaje Estadístico
- Tareas típicas de Machine Learning
  - Descubrimiento de Patrones
  - Clasificación
  - Clustering
  - Regresión
  - Detección de Anomalías/Outliers
  - Reducción de Dimensionalidad



# Logros de Machine Learning

- IBM Watson vence a los campeones de Jeopardy.  
<< ... With all of its processing CPU power, Watson can scan two million pages of data in three seconds.>> E. Nyberg, CMU professor
- Implicancias: Aplicaciones en medicina



<http://www.aaai.org/Magazine/Watson/watson.php>

# Vehículos Autónomos





# Generación de música con estilo

- **Deep learning driven jazz generation**
- <https://github.com/jisungk/deepjazz>
- <https://soundcloud.com/deepjazz-ai>



# Mastering Go

## Google AI algorithm masters ancient game of Go

Deep-learning software defeats human professional for first time.

[Elizabeth Gibney](#)

27 January 2016



[PDF](#)



[Rights & Permissions](#)



The computer that mastered Go

Nature Video

# Imágenes: incluir elementos y estilo

<https://github.com/luanfujun/deep-painterly-harmonization>

README.md

## deep-painterly-harmonization

Code and data for paper "Deep Painterly Harmonization"

### Disclaimer

This software is published for academic and non-commercial use only.

### Setup

This code is based on torch. It has been tested on Ubuntu 16.04 LTS.

Dependencies:

- [Torch](#) (with [loadcaffe](#))
- [Matlab](#) or [Octave](#)

CUDA backend:

- [CUDA](#)
- [cudnn](#)

Download VGG-19:

```
sh models/download_models.sh
```

Compile `cuda_utils.cu` (Adjust `PREFIX` and `NVCC_PREFIX` in `makefile` for your machine):

```
make clean && make
```

### Usage

To generate all results (in `data/`) using the provided scripts, simply run

```
python gen_all.py
```

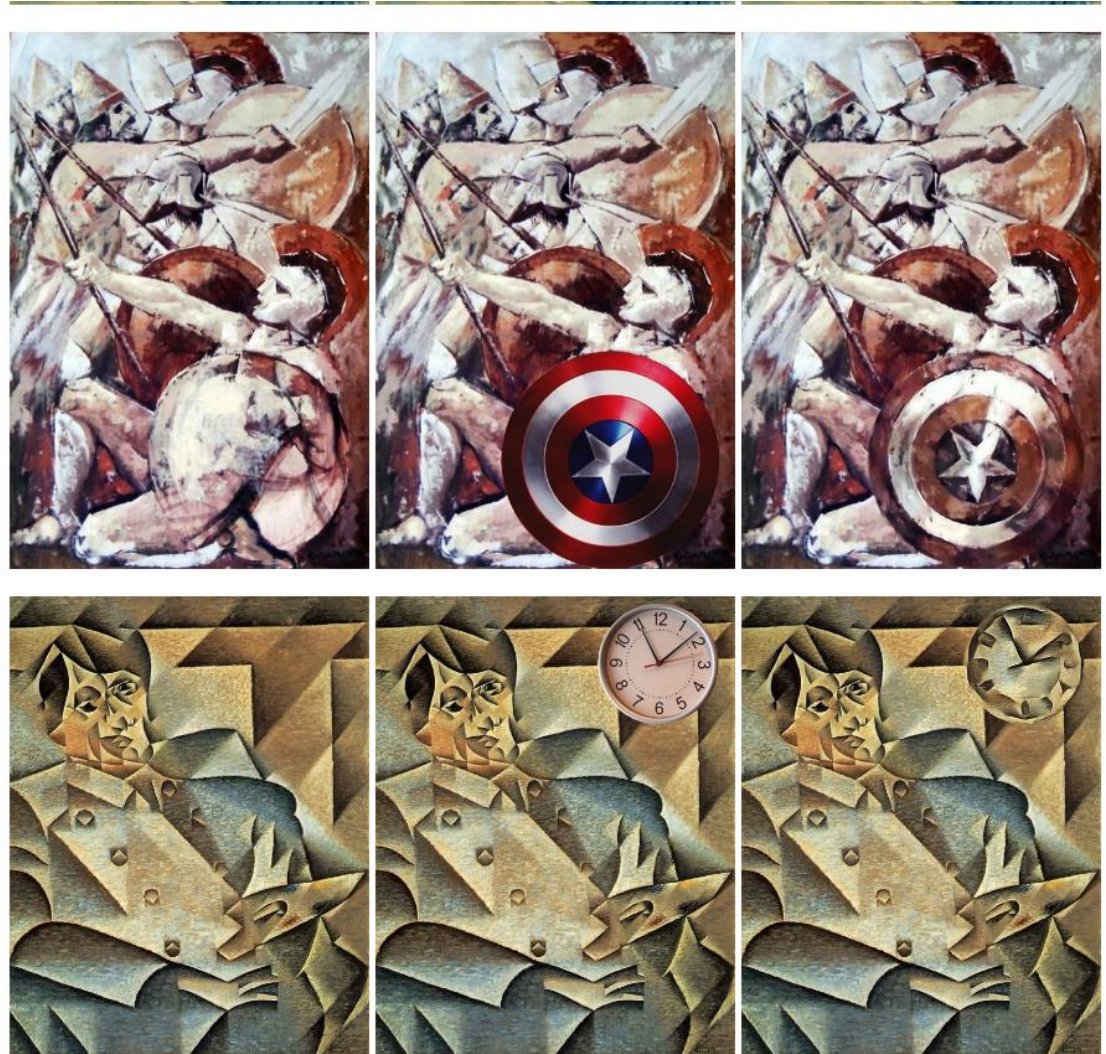
in Python and then

```
run('filt_cnn_artifact.m')
```

in Matlab or Octave. The final output will be in `results/`.

Note that in the paper we trained a CNN on a dataset of 80,000 paintings collected from [wikiart.org](#), which estimates the stylization level of a given painting and adjust weights accordingly. We will release the pre-trained model in the next update. Users will need to set those weights manually if running on their new paintings for now.

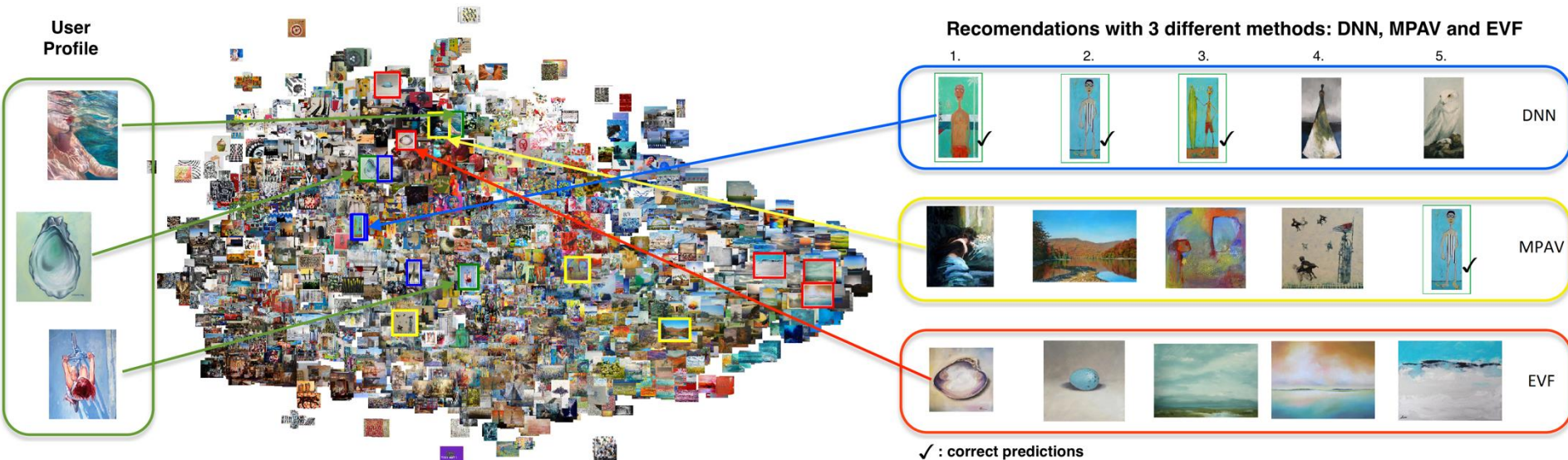
Removed a few images due to copyright issue. Full set [here](#) for testing use only.



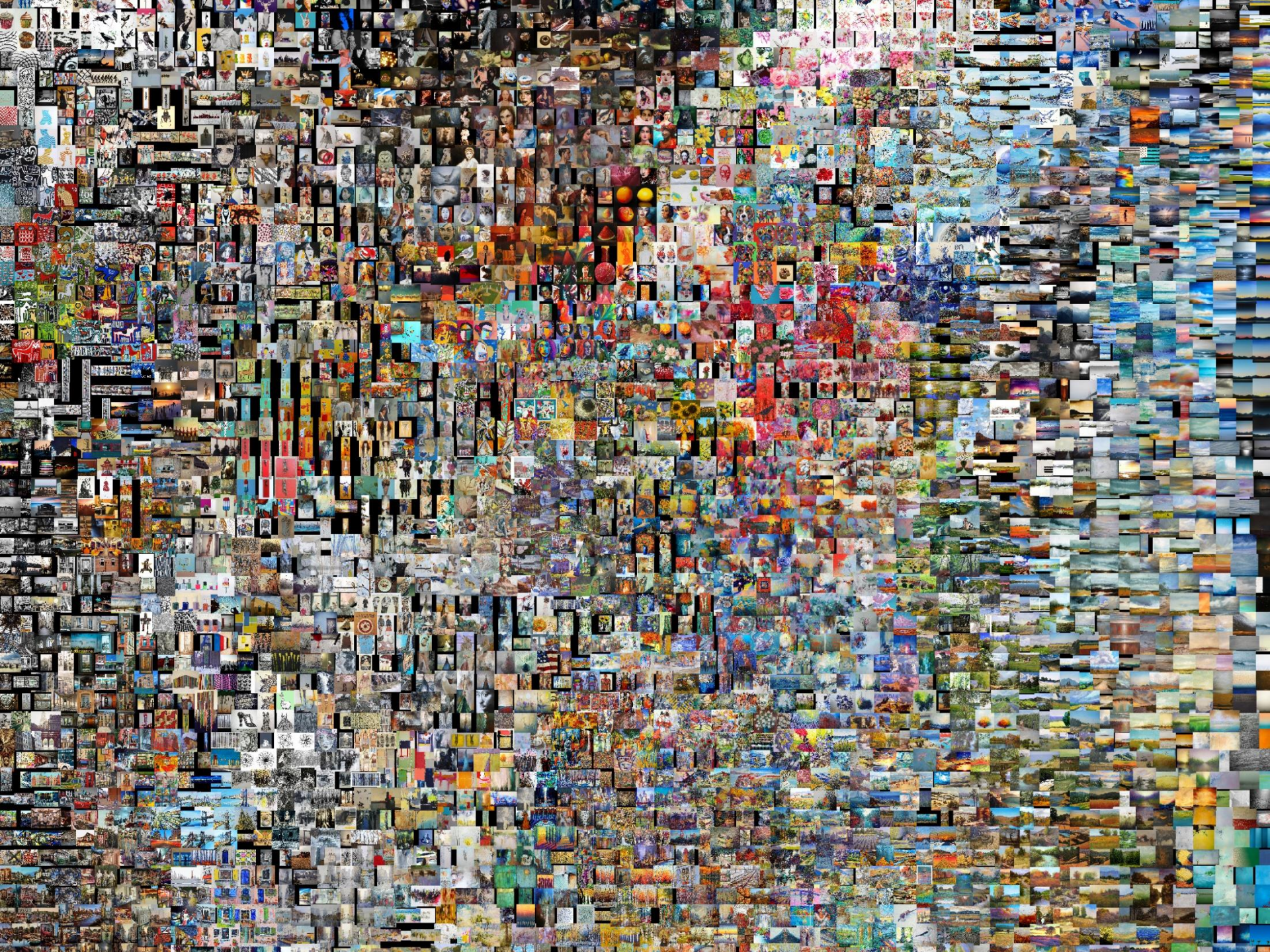


# Una imagen vale más que mil palabras

- Trabajo conjunto con Denis Parra y Pablo Messina









# Clasificación Tradicional de algoritmos de ML

- **Aprendizaje Supervisado:** Los algoritmos reciben ejemplos (datos etiquetados) a partir de los cuales aprenden
- **Aprendizaje No Supervisado:** Los algoritmos no reciben ejemplos etiquetados.

# Aprendizaje Supervisado

- KNN
- Árboles de Decisión
- Naive Bayes
- Regresión Logística
- Support Vector Machines
- Redes Neuronales

# Genéricamente

- Inteligencia de Máquina: **Aprender de los Datos**
- Revisemos de modo conceptual un ejemplo:  
Construir para un banco un sistema que automáticamente apruebe o niegue crédito

Applicant information:

**Rechazar o  
Aprobar credito??**

age	23 years
gender	male
annual salary	\$30,000
years in residence	1 year
years in job	1 year
current debt	\$15,000
...	...



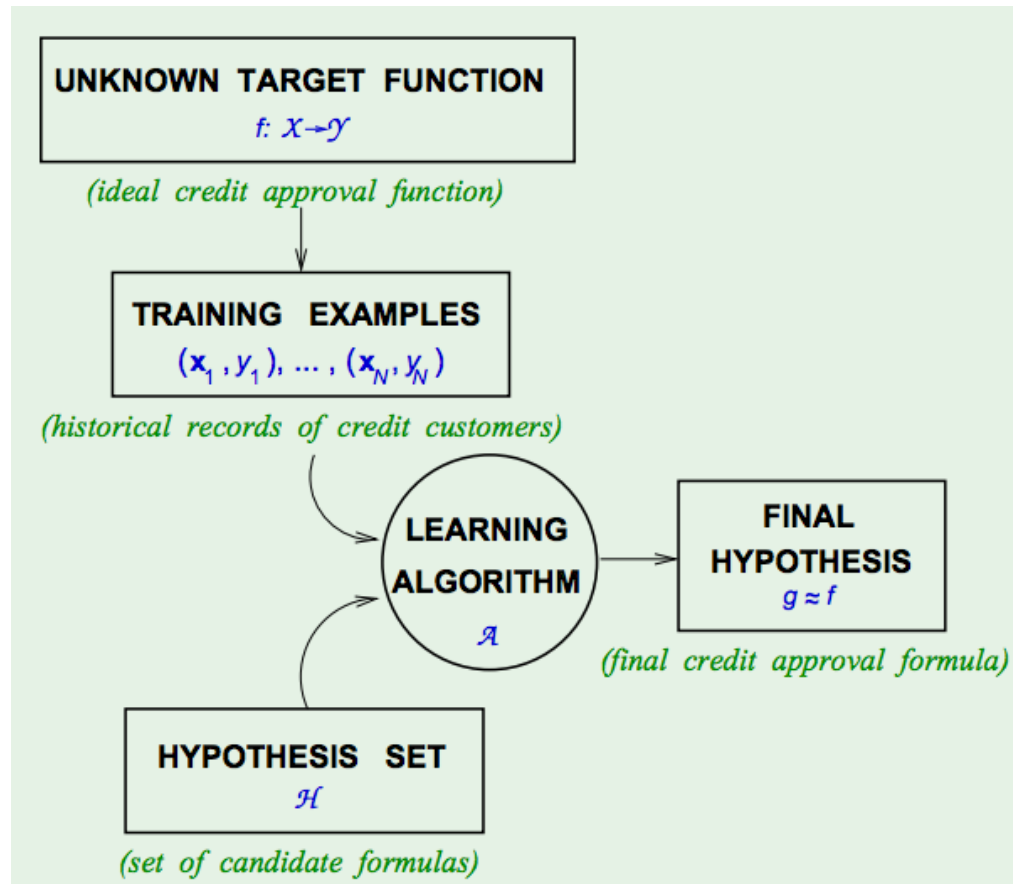
# Formalización del Problema de Aprendizaje

- Encontrar la fórmula de aprobación  $g()$  que se aproxime lo más posible a la fórmula ideal  $f()$

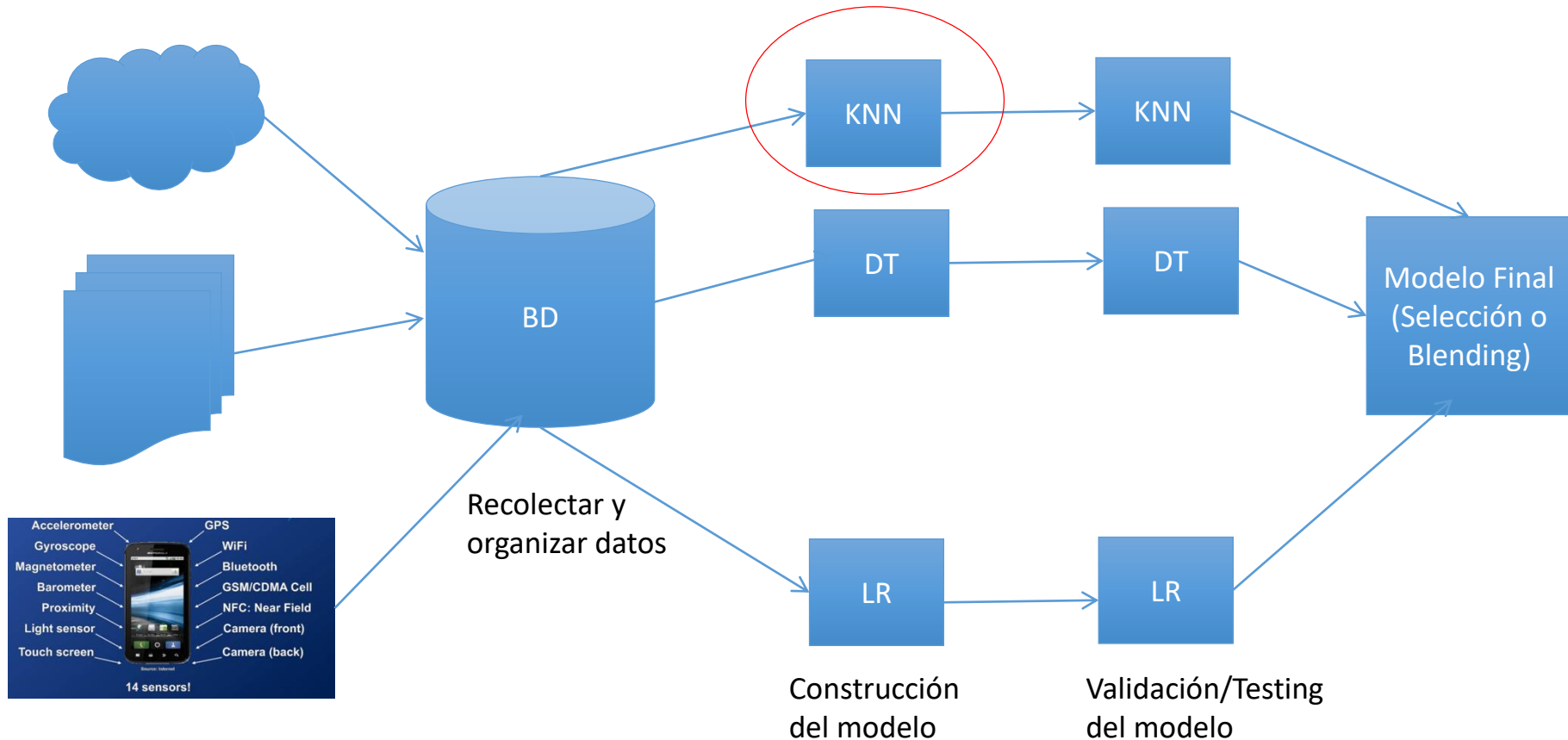
- Input:  $\mathbf{x}$  (*customer application*)
  - Output:  $y$  (*good/bad customer?*)
  - Target function:  $f : \mathcal{X} \rightarrow \mathcal{Y}$  (*ideal credit approval formula*)
  - Data:  $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_N, y_N)$  (*historical records*)
- ↓   ↓   ↓
- Hypothesis:  $g : \mathcal{X} \rightarrow \mathcal{Y}$  (*formula to be used*)

# Formalización del Problema de Aprendizaje

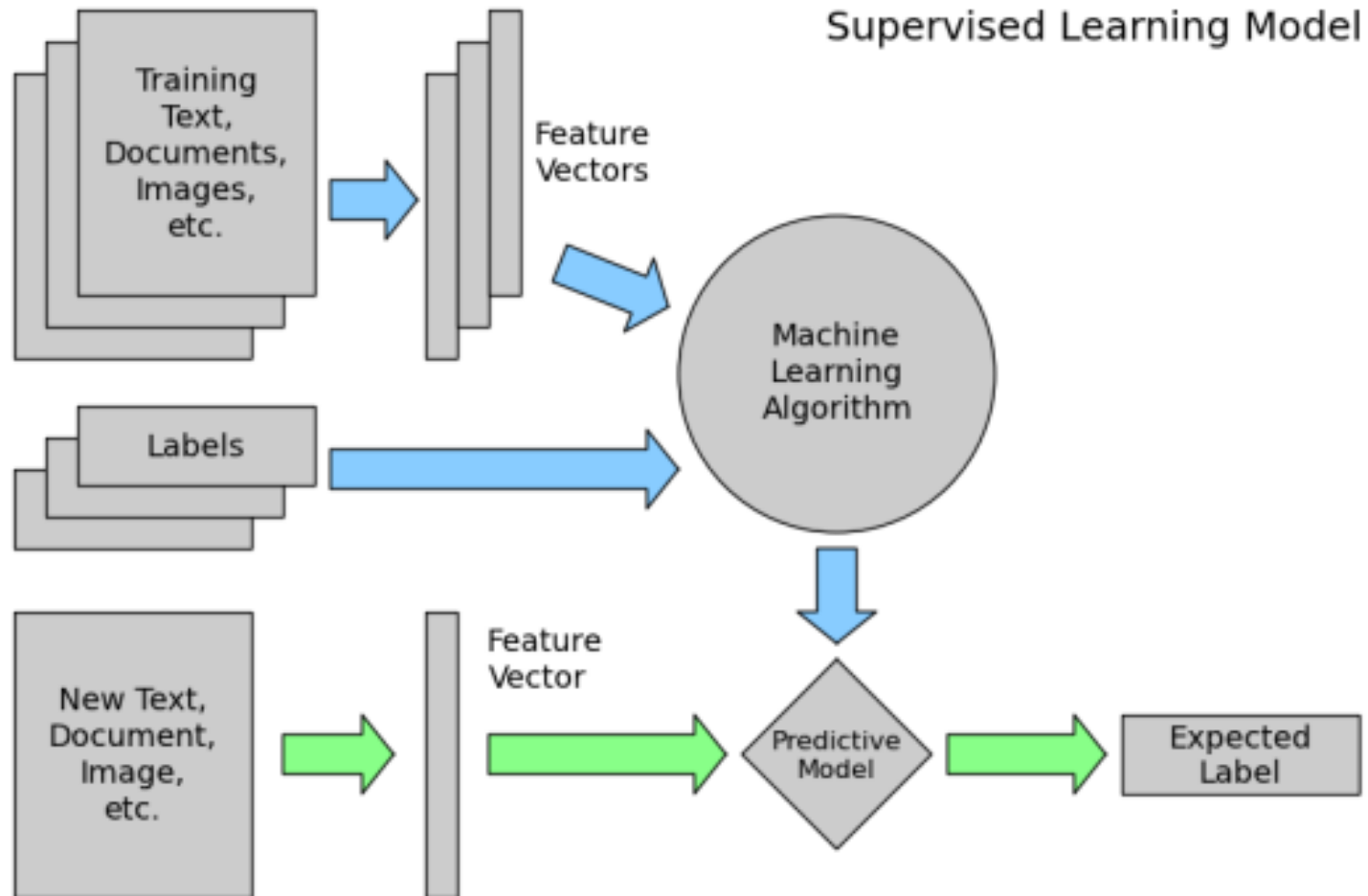
- El conjunto de hipótesis  $H$



# Pasos para realizar Minería de Datos (supervisado)



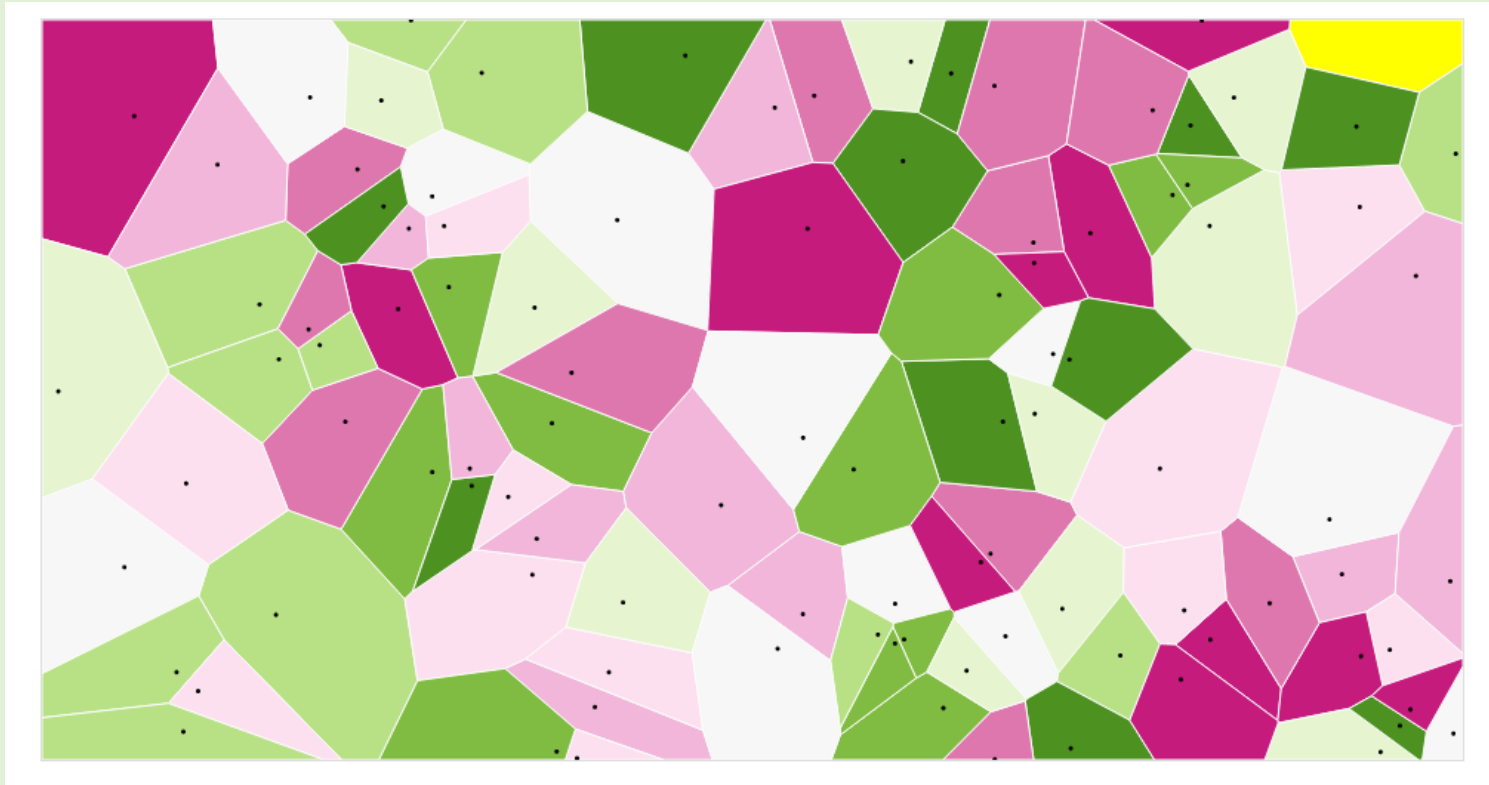
# Modelo de Aprendizaje Supervisado (detalle)



# K-NN

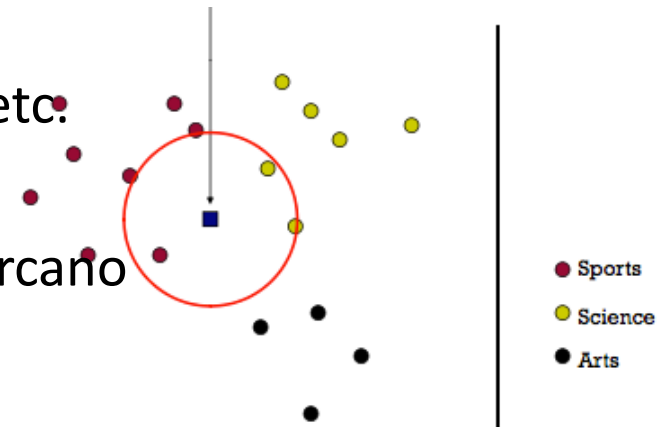
- Significa K-nearest neighbors ( K-vecinos más cercanos )
- La intuición de este modelo es clasificar a una instancia en base a la clasificación ya conocida de las instancias más cercanas.

# Clasificador KNN - K vecinos mas cercanos

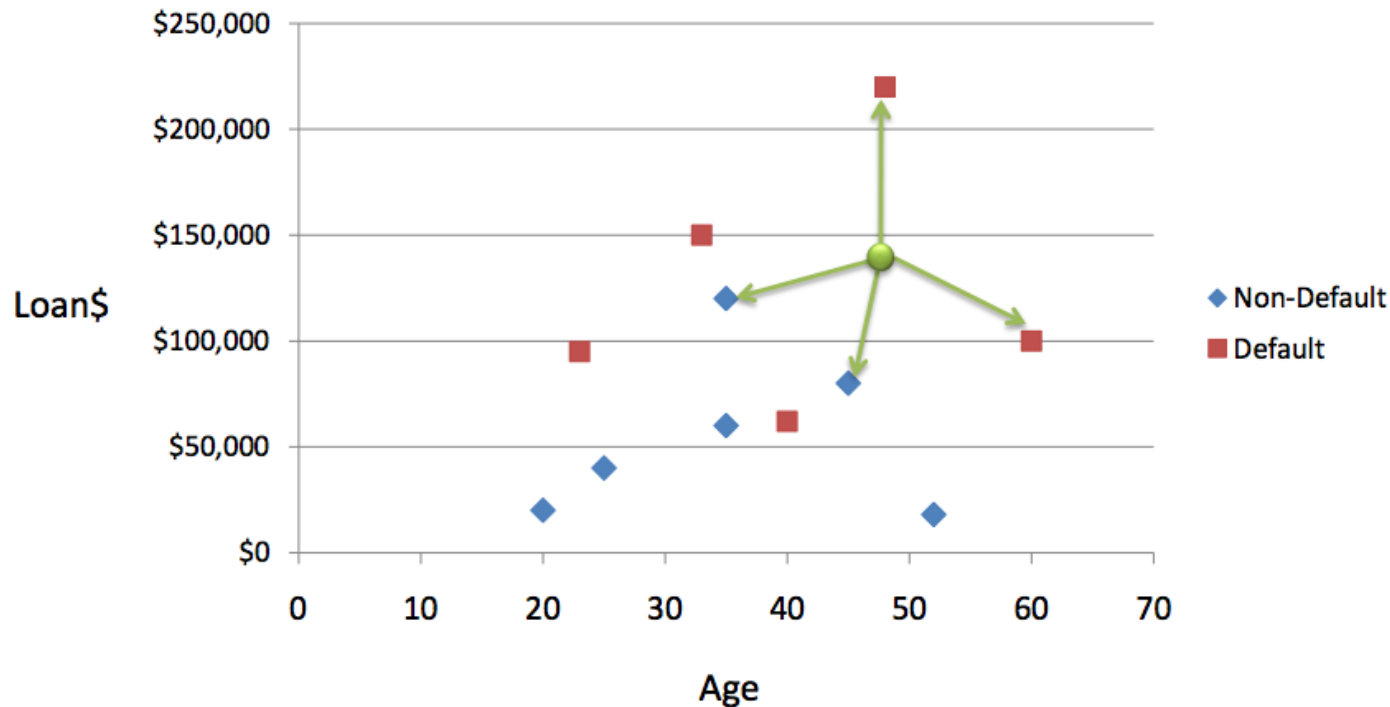


# ¿Qué parámetros considerar con KNN?

- Una métrica de distancia
  - Euclidiana, Manhattan, Correlación, etc.
- ¿Cuántos vecinos considerar?
  - En 1-NN: solo un vecino => el más cercano
- Una función de peso (opcional)
  - $1/d$ ;  $1-d$ ;  $\exp(-d^2)$ , etc.
- ¿Cómo usar los vecinos cercanos para hacer la predicción?
  - El más cercano, votación, el promedio, etc.



# Ejemplo: Clasificar el caso siguiente





# Usemos 1NN

Age	Loan	Default	Distance
25	\$40,000	N	
35	\$60,000	N	
45	\$80,000	N	
20	\$20,000	N	
35	\$120,000	N	
52	\$18,000	N	
23	\$95,000	Y	
40	\$62,000	Y	
60	\$100,000	Y	
48	\$220,000	Y	
33	\$150,000	Y	
<b>48</b>	<b>\$142,000</b>	<b>?</b>	

Euclidean Distance

$$D = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

# Usemos 1NN / 2NN / ... KNN

Age	Loan	Default	Distance
25	\$40,000	N	102000
35	\$60,000	N	82000
45	\$80,000	N	62000
20	\$20,000	N	122000
35	\$120,000	N	22000
52	\$18,000	N	124000
23	\$95,000	Y	47000
40	\$62,000	Y	80000
60	\$100,000	Y	42000
48	\$220,000	Y	78000
33	\$150,000	Y	8000
<b>48</b>	<b>\$142,000</b>	<b>?</b>	

Euclidean Distance

$$D = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

# ¿Cómo lo hacemos en python?

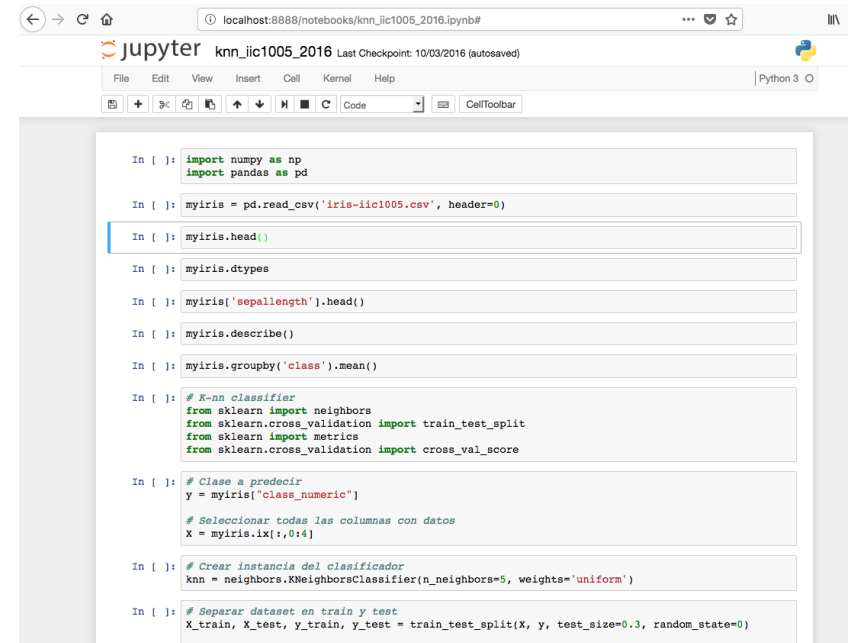
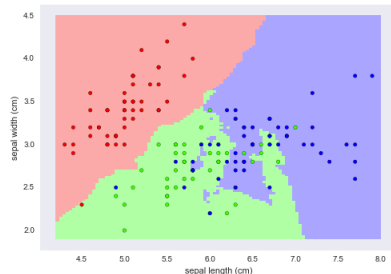
- Biblioteca scikit-learn
- Probemos con un jupyter notebook

```
# What kind of iris has 3cm x 5cm sepal and 4cm x 2cm petal?
X_pred = [3, 5, 4, 2]
result = knn.predict([X_pred, ])

print(iris.target_names[result])
print(iris.target_names)
print(knn.predict_proba([X_pred, ]))

from fig_code import plot_iris_knn
plot_iris_knn()
```

```
['versicolor']
['setosa' 'versicolor' 'virginica']
[[ 0.  0.8 0.2]]
```



```
In [ ]: import numpy as np
import pandas as pd

In [ ]: myiris = pd.read_csv('iris-lic1005.csv', header=0)

In [ ]: myiris.head()

In [ ]: myiris.dtypes

In [ ]: myiris['sepalength'].head()

In [ ]: myiris.describe()

In [ ]: myiris.groupby('class').mean()

In [ ]: # K-nn classifier
from sklearn import neighbors
from sklearn.cross_validation import train_test_split
from sklearn import metrics
from sklearn.cross_validation import cross_val_score

In [ ]: # Clase a predecir
y = myiris['class_numerico']

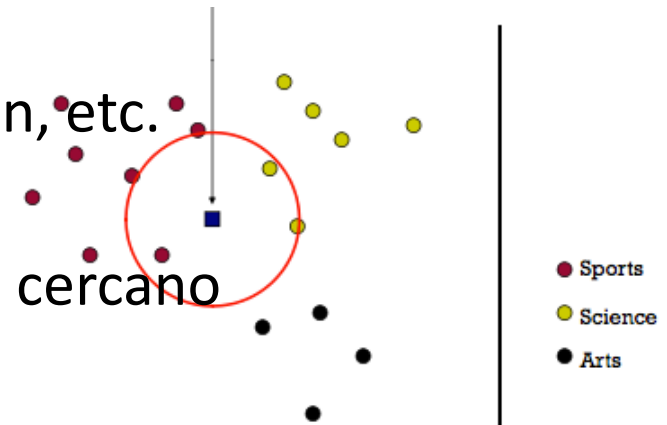
# Seleccionar todas las columnas con datos
X = myiris.ix[:,0:4]

In [ ]: # Crear instancia del clasificador
knn = neighbors.KNeighborsClassifier(n_neighbors=5, weights='uniform')

In [ ]: # Separar dataset en train y test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)
```

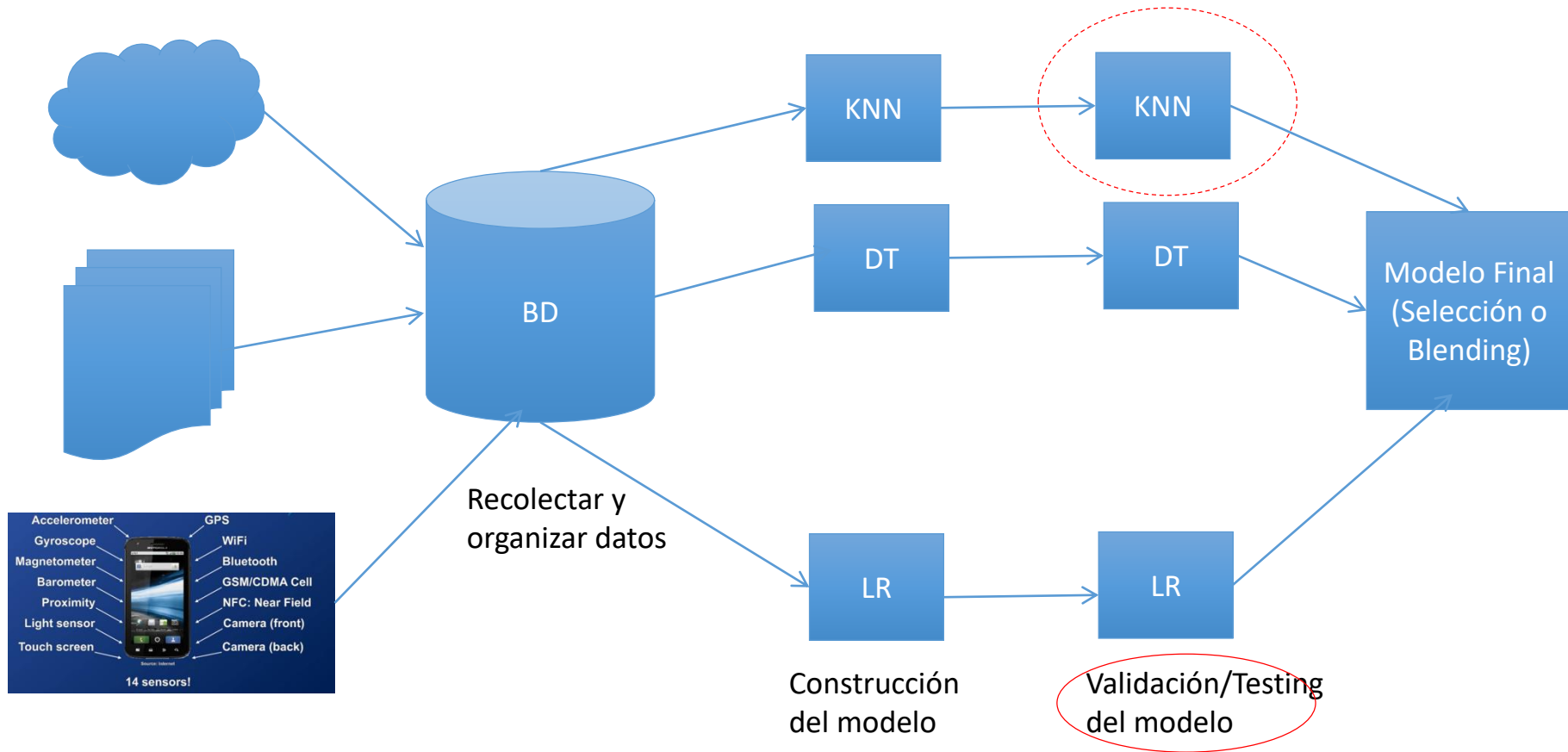
# Recapitulando

- Una métrica de distancia
  - Euclidiana, Manhattan, Correlación, etc.
- ¿Cuántos vecinos considerar?
  - En 1-NN: solo un vecino => el más cercano
- Una función de peso (opcional)
  - $1/d$ ;  $1-d$ ;  $\exp(-d^2)$ , etc.
- ¿Cómo usar los vecinos cercanos para hacer la predicción?
  - El más cercano, votación, el promedio, etc.



**Sin embargo, estamos usando los datos, no un MODELO**

# Pasos para realizar minería de Datos



# Caso uno: clasificación de 2 clases

	Predicción clase A	Predicción clase B	Total de cada Clase
Clase A	104	22	126
Clase B	8	217	225

Supongamos para la tarea que Clase A = vendedor / Clase B = no vendedor

TP: Verdaderos Positivos: 321 - Revisamos cada valor CLASE POR CLASE

$$\text{TP Rate} = \text{TP} / (\text{TP} + \text{FN})$$

Clase A: 104 de 126 ; TP rate =  $104/126 = 0.825$

Clase B: 217 de 225 ; TP rate =  $217/225 = 0.964$

FP: Falso Positivos: 30

$$\text{FP Rate} = \text{FP} / (\text{FP} + \text{TN})$$

Clase A: 8 (que eran de clase B) ; FP rate =  $8 / 225 = 0.035$

Clase B: 22 (que eran de la clase A) ; FP rate =  $22 / 126 = 0.174$

# Caso uno: clasificación de 2 clases

	Predicción clase A	Predicción clase B	Total de cada Clase
Clase A	104	22	126
Clase B	8	217	225

## PRECISION

CLASE A:  $104/112 = 0.928$

CLASE B:  $217/239 = 0.907$

## RECALL

CLASE A:  $104/126 = 0.825$

CLASE B:  $217/225 = 0.964$

F-Measure (Harmonic mean)

$2 * \text{PRECISION} * \text{RECALL} / (\text{PRECISION} + \text{RECALL})$

F-clase A = 0.874

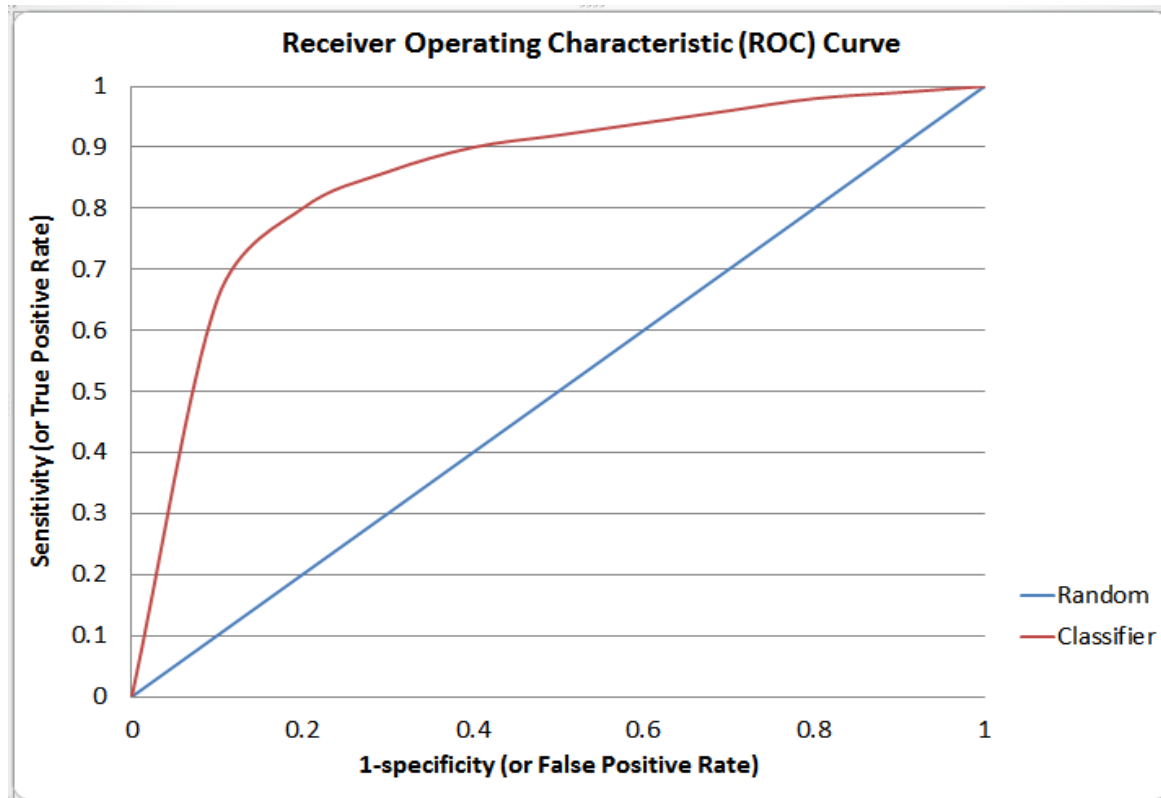
F-clase B = 0.935

ROC Area = 0.892 (revisaremos este en una slides mas adelante)

Accuracy (set completo)  
 $(104 + 217) / 351 = 0.914$

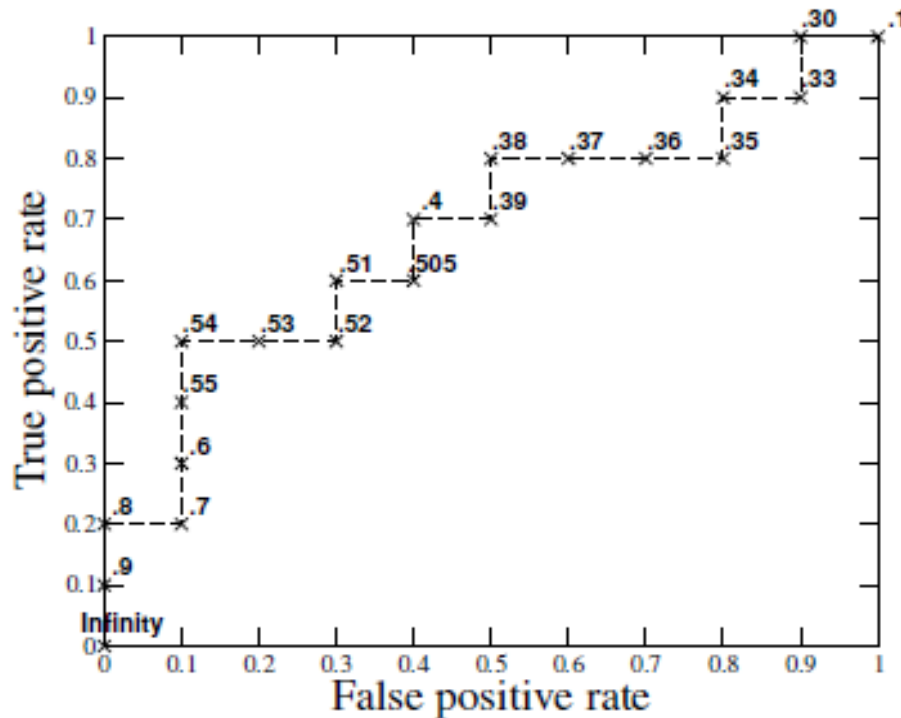
# ROC y AUC – Clasificación Binaria

- Area under the curve





# Construcción de la ROC Curve



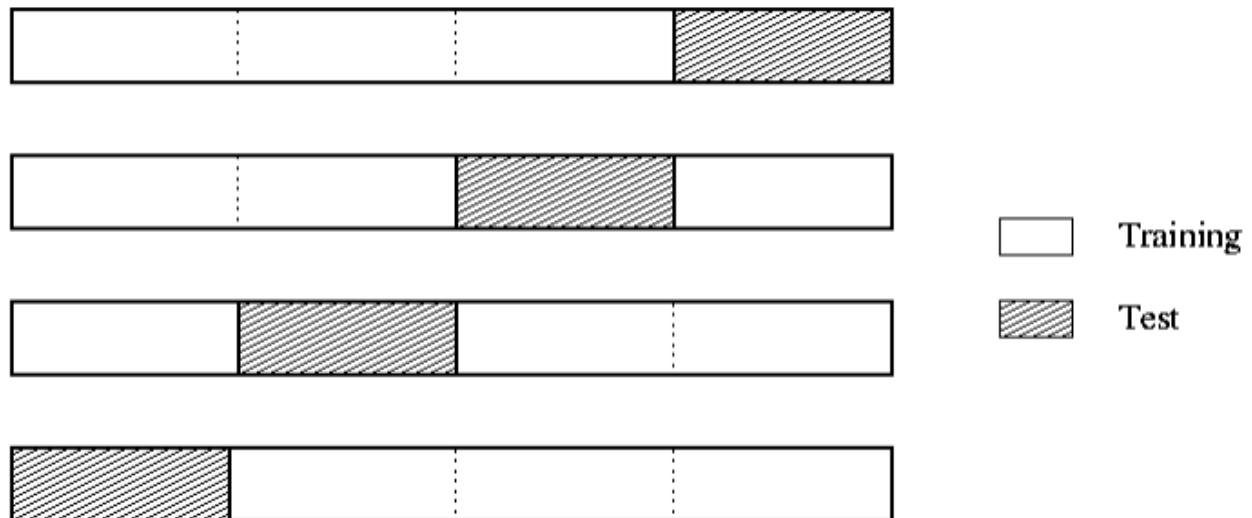
Inst#	Class	Score	Inst#	Class	Score
1	p	.9	11	p	.4
2	p	.8	12	n	.39
3	n	.7	13	p	.38
4	p	.6	14	n	.37
5	p	.55	15	n	.36
6	p	.54	16	n	.35
7	n	.53	17	p	.34
8	n	.52	18	n	.33
9	p	.51	19	p	.30
10	n	.505	20	n	.1

Figure 3: The ROC “curve” created by thresholding a test set. The table at right shows twenty data and the score assigned to each by a scoring classifier. The graph at left shows the corresponding ROC curve with each point labeled by the threshold that produces it.

Revisar: <http://www.hpl.hp.com/techreports/2003/HPL-2003-4.pdf>

# Cross-Validation

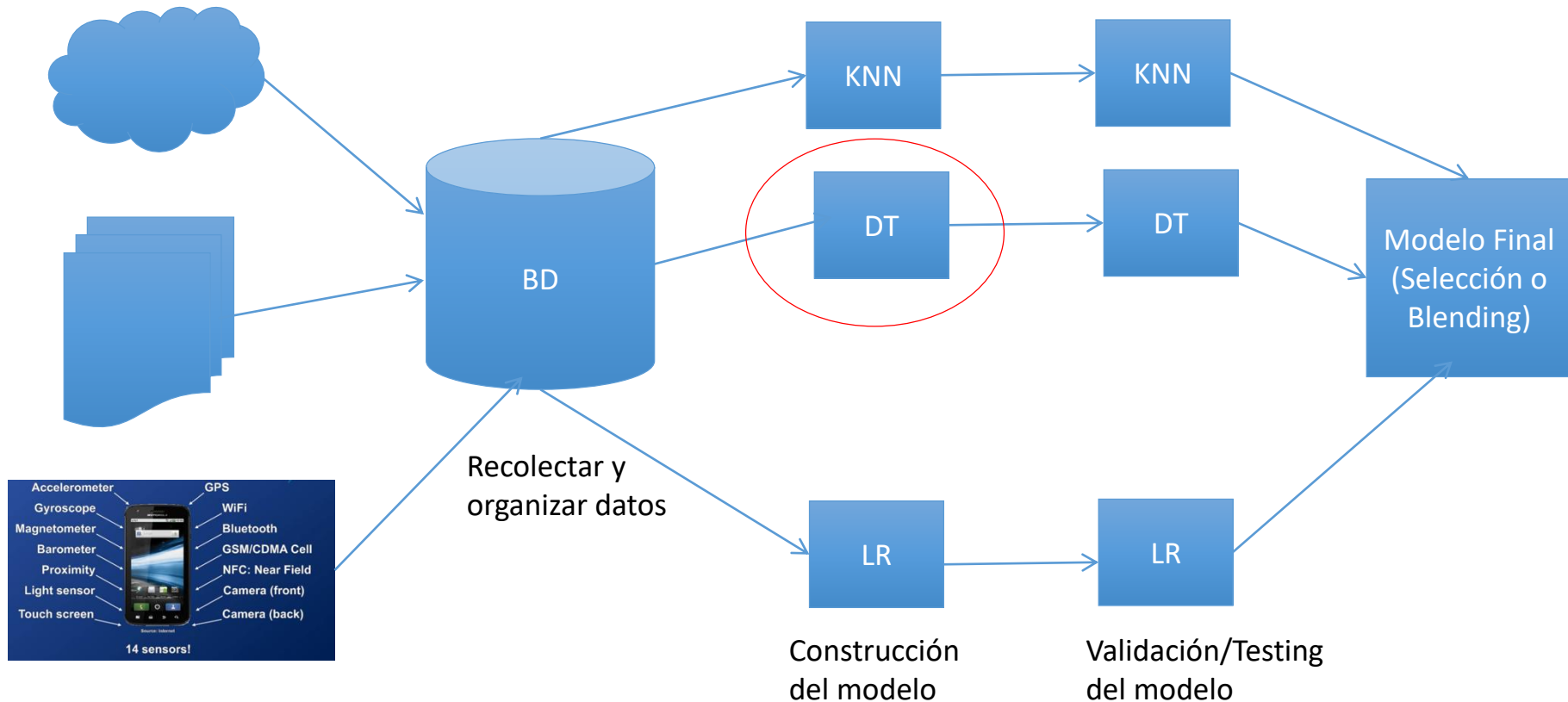
- Partir el dataset en K particiones
- Por cada partición X
  - Usar las otras  $x-1$  particiones para entrenar modelo
- Example: 4-fold cross-validation



# Veamos otro modelo de clasificación

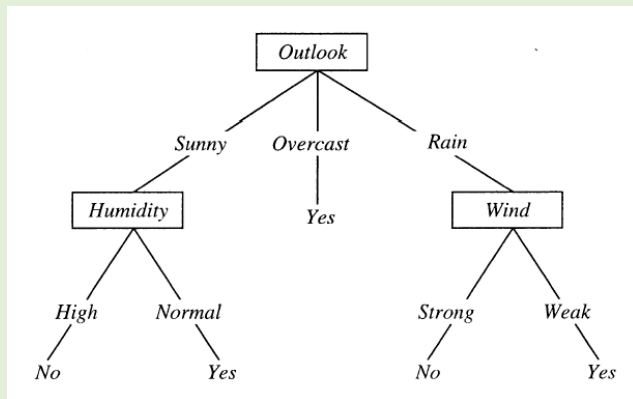
- Árboles de Decisión

# Pasos para realizar Minería de Datos



# Árboles de Decisión

- Objetivo: Obtener de manera automática una serie de reglas que permita clasificar los casos. La representación final del modelo es a través de un árbol



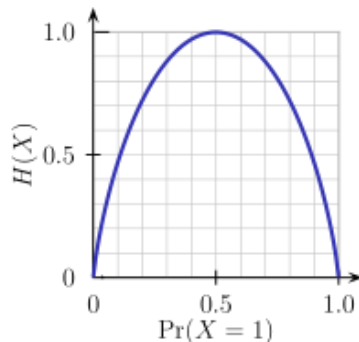
Day	Outlook	Temperature	Humidity	Wind	PlayTenn
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Imagen tomada desde

<http://www.doc.ic.ac.uk/~sgc/teaching/pre2012/v231/lecture11.html>

# Ejemplo de construcción de un árbol de decisión

- Los nodos del árbol representan variables, las ramas representan valores de las variables que permiten clasificar
- Las hojas del árbol corresponden a la clasificación
- En la construcción del árbol, se testea una variable a la vez, usando el concepto de Entropía (número de bits necesarios para transmitir un mensaje - o - nivel de incerteza respecto a un evento)



$$H = -\sum_{i=1}^M P_i \log_2 P_i$$

# ¿Cuál de los siguientes escenarios es más incierto?

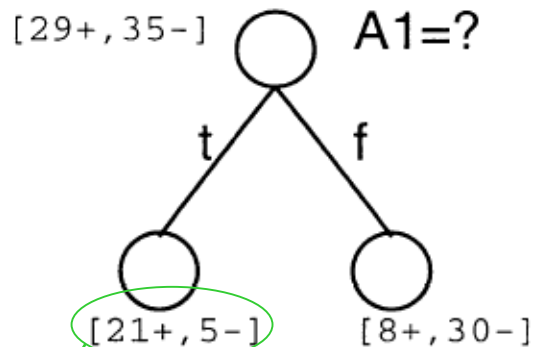
- Predicción del tiempo
  - Lluve: 50%, Sol: 50%
  - Lluve: 100%, Sol: 0%
- Calcular la entropía de cada uno:

$$H = - \sum_{i=1}^M P_i \log_2 P_i$$

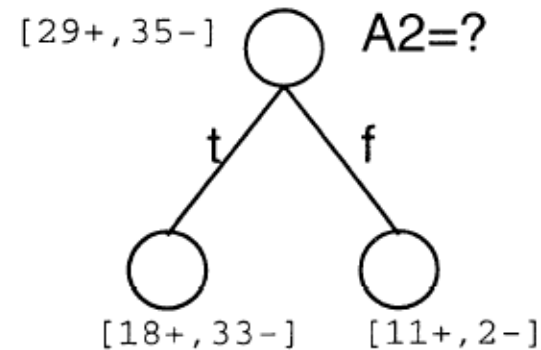
# ¿Qué variable es la mejor como nodo raíz?

- La que reduce en mayor grado la entropía inicial

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$



Proporciones  
en clase objetivo

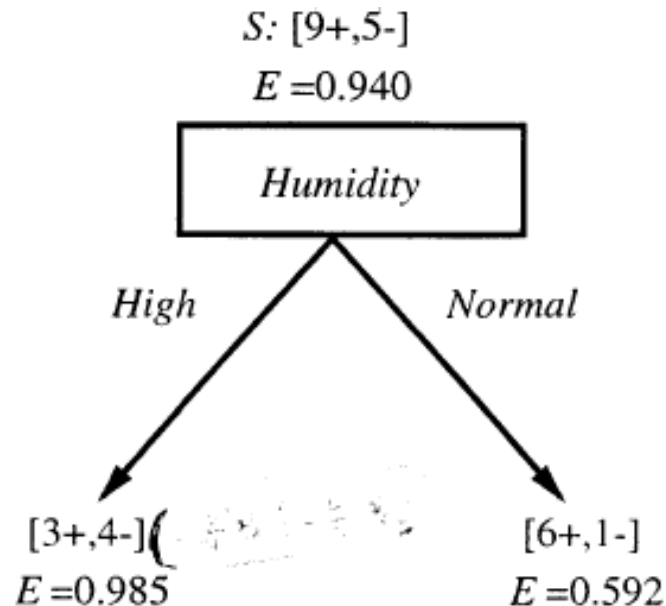




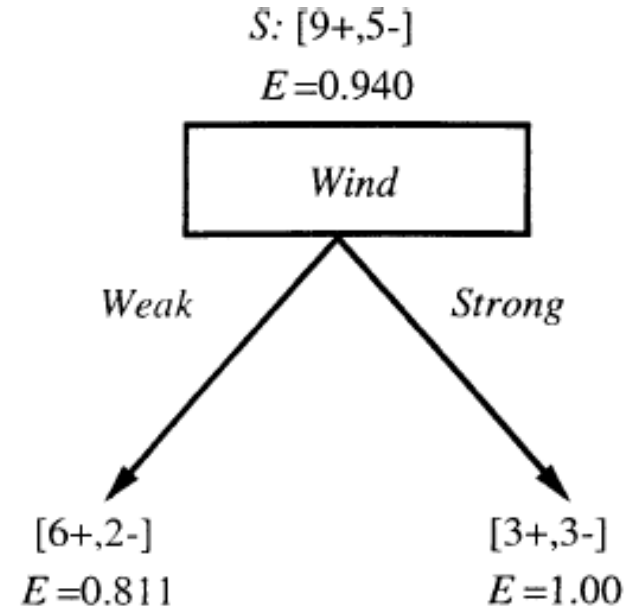
# Ejemplo: ¿Jugar o no jugar tenis?

Day	Outlook	Temperature	Humidity	Wind	PlayTenn
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

# ¿Cuál variable reduce mas la entropía?



$$\begin{aligned} \text{Gain}(S, \text{Humidity}) &= .940 - (7/14) \cdot .985 - (7/14) \cdot .592 \\ &= .151 \end{aligned}$$

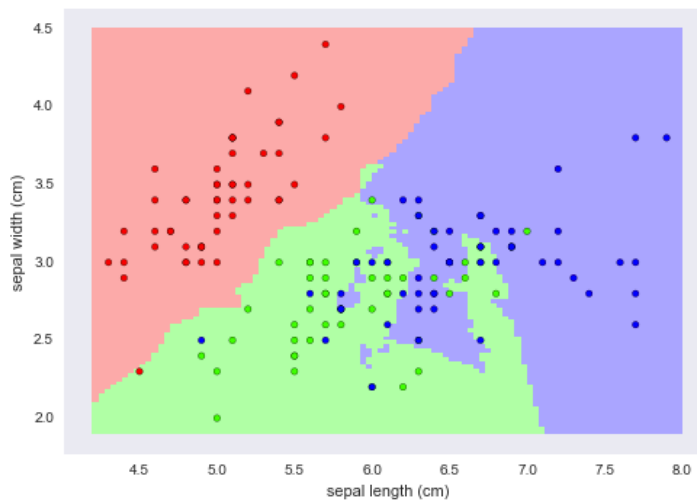


$$\begin{aligned} \text{Gain}(S, \text{Wind}) &= .940 - (8/14) \cdot .811 - (6/14) \cdot 1.0 \\ &= .048 \end{aligned}$$

# El mismo ejemplo anterior...

- Ahora con árboles de decisión

```
# What kind of iris has 3cm x 5cm sepal and 4cm x 2cm petal?  
X_pred = [3, 5, 4, 2]  
result = knn.predict([X_pred, ])  
  
print(iris.target_names[result])  
print(iris.target_names)  
print(knn.predict_proba([X_pred, ]))  
  
from fig_code import plot_iris_knn  
plot_iris_knn()  
  
['versicolor']  
['setosa' 'versicolor' 'virginica']  
[[ 0.  0.8 0.2]]
```



# Resumen

- Cuáles son las “escuelas de aprendizaje” en Inteligencia Artificial:
  - Aprendizaje deductivo (lógica)
  - Aprendizaje inductivo (learning from data)
- En el modelo inductivo, qué tipos de tareas típicas con encontramos:
  - Predicción
  - Clasificación
  - Clustering
- En cuanto a clasificación, aprendiste: K-NN y D.T.