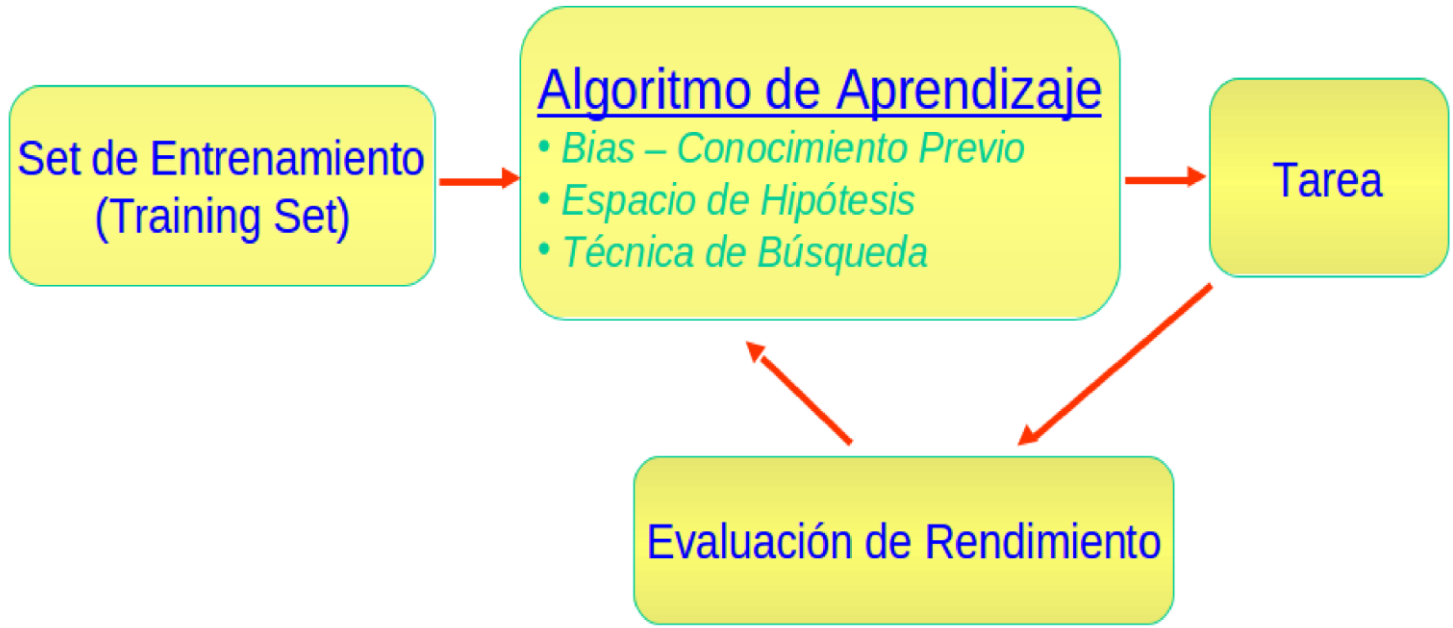
Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación


# IIC2613 – Inteligencia Artificial

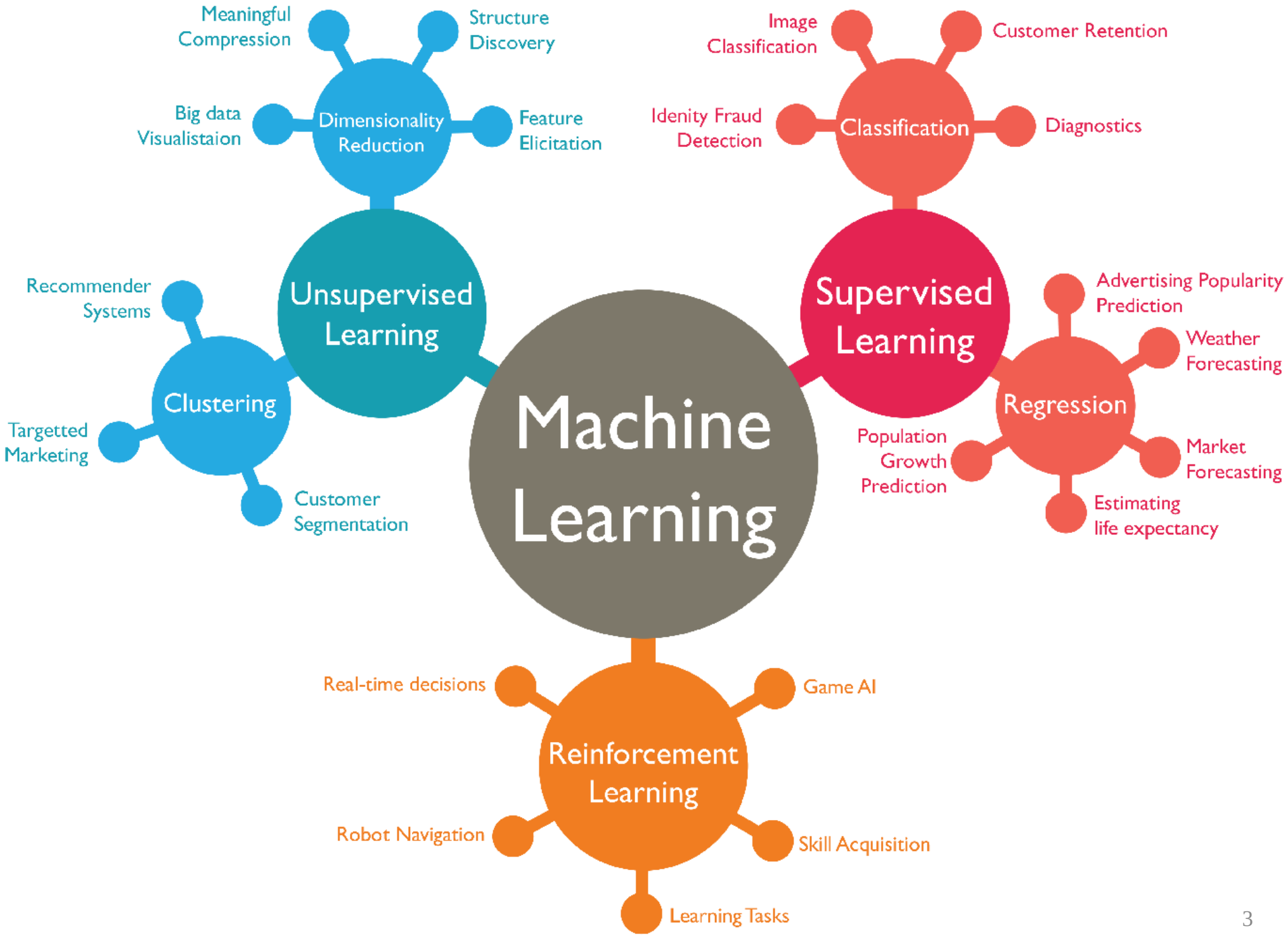
## Fundamentos de ML

**Profesor:** Hans Löbel

Recordemos que Machine Learning se centra en algoritmos que mejoran su rendimiento en una tarea, a través de la experiencia




Buscamos la solución más adecuada en el espacio de hipótesis, usando conocimiento previo y datos de entrenamiento para guiar la búsqueda.


Machine Learning
Unsupervised Learning
Meaningful Compression
Structure Discovery
Dimensionality Reduction
Big data Visualistaion
Feature Elicitation
Clustering
Recommender Systems
Targetted Marketing
Customer Segmentation
Supervised Learning
Image Classification
Customer Retention
Classification
Idenity Fraud Detection
Diagnostics
Regression
Advertising Popularity Prediction
Weather Forecasting
Market Forecasting
Estimating life expectancy
Population Growth Prediction
Reinforcement Learning
Real-time decisions
Game AI
Robot Navigation
Skill Acquisition
Learning Tasks

# Algoritmos de ML trabajan sobre datos multidimensionales

- Cada dato esta caracterizado por una serie de mediciones = atributos = variables.
- La cantidad de variables define la dimensionalidad del dato.
- El espacio donde viven los datos (variables) se conoce como espacio de características (*feature space*).

## Wine Data Set

*Download*: Data Folder, Data Set Description

**Abstract**: Using chemical analysis determine the origin of wines

| | | | | | |
|---|---|---|---|---|---|
| **Data Set Characteristics:** | Multivariate | **Number of Instances:** | 178 | **Area:** | Physical |
| **Attribute Characteristics:** | Integer, Real | **Number of Attributes:** | 13 | **Date Donated** | 1991-07-01 |
| **Associated Tasks:** | Classification | **Missing Values?** | No | **Number of Web Hits:** | 573523 |

Para entrenar = ajustar = calibrar un modelo,
se utiliza un set de entrenamiento

| Typhoon number | Input vectors | | | | Response vector |
|---|---|---|---|---|---|
| | Distance from the eye of the storm (km) | Wind speed at site (m/s) | Pressure deficit at site (hPa) | Forward speed of the eye of the storm (km/h) | Storm surge (cm) |
| 5111 | 96.0 | 20.7 | 20.6 | 27.6 | 47.4 |
| 5114 | 108.5 | 15.4 | 11.0 | 58.9 | 24.5 |
| 5201 | 181.2 | 8.1 | 1.7 | 40.1 | 7.9 |
| 5204 | 245.3 | 5.7 | 6.4 | 29.6 | 5.5 |
| 5209 | 117.5 | 23.3 | 22.0 | 46.6 | 61.7 |
| 5211 | 231.4 | 13.3 | 11.5 | 38.1 | 20.8 |
| 5309 | 293.6 | 4.0 | 7.2 | 35.4 | 5.6 |
| 5508 | 0.6 | 8.5 | 7.0 | 32.2 | 8.7 |
| 5512 | 227.6 | 10.0 | 10.4 | 19.3 | 16.0 |
| 5609 | 257.3 | 11.5 | 15.0 | 44.1 | 10.8 |

Cada dato (fila) del set de entrenamiento, puede considerarse como un vector en el espacio de características.

# Objetivo último es la generalización

Entrenamiento

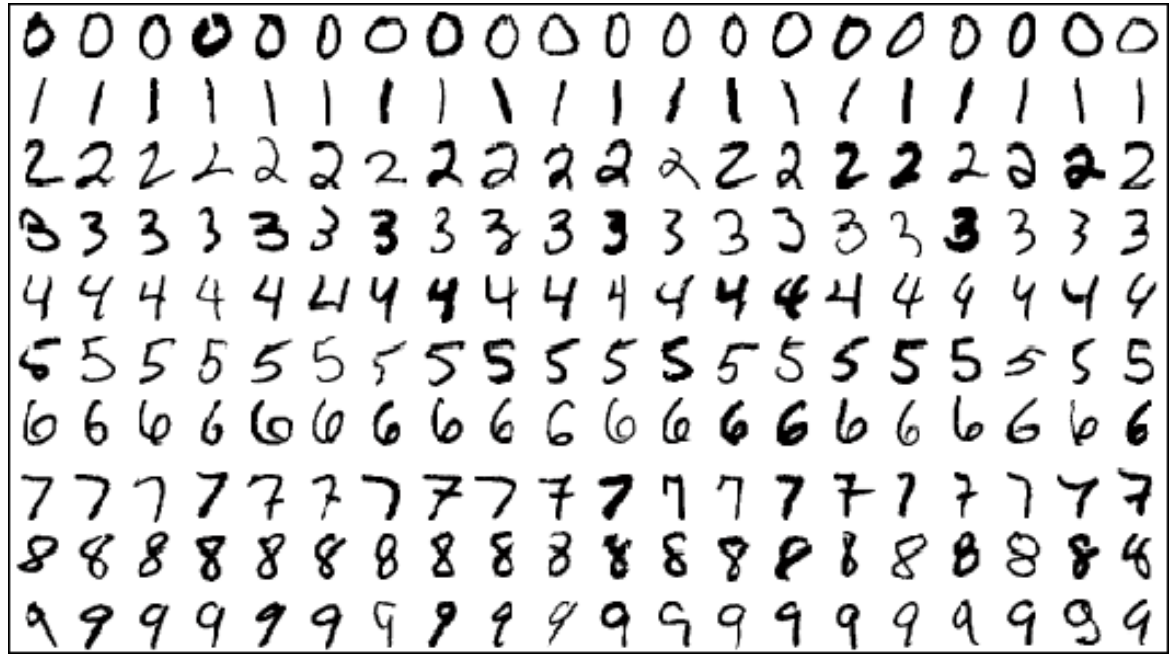
Test

| Typhoon number | Input vectors | | | | Response vector |
|---|---|---|---|---|---|
| | Distance from the eye of the storm (km) | Wind speed at site (m/s) | Pressure deficit at site (hPa) | Forward speed of the eye of the storm (km/h) | Storm surge (cm) |
| 5111 | 96.0 | 20.7 | 20.6 | 27.6 | 47.4 |
| 5114 | 108.5 | 15.4 | 11.0 | 58.9 | 24.5 |
| 5201 | 181.2 | 8.1 | 1.7 | 40.1 | 7.9 |
| 5204 | 245.3 | 5.7 | 6.4 | 29.6 | 5.5 |
| 5209 | 117.5 | 23.3 | 22.0 | 46.6 | 61.7 |
| 5211 | 231.4 | 13.3 | 11.5 | 38.1 | 20.8 |
| 5309 | 293.6 | 4.0 | 7.2 | 35.4 | 5.6 |
| 5508 | 0.6 | 8.5 | 7.0 | 32.2 | 8.7 |
| 5512 | 227.6 | 10.0 | 10.4 | 19.3 | 16.0 |
| 5609 | 257.3 | 11.5 | 15.0 | 44.1 | 10.8 |
| 0209 | 290.6 | 9.5 | 13.6 | 46.9 | |
| 0215 | 245.3 | 10.6 | 14.2 | 77.6 | |
| 0306 | 227.0 | 4.4 | 7.9 | 20.8 | |
| 0314 | 279.1 | 4.4 | 7.8 | 29.5 | |
| 0415 | 266.3 | 8.7 | 8.8 | 32.9 | |
| 0515 | 165.6 | 19.2 | 16.4 | 45.6 | |
| 0601 | 136.5 | 10.7 | 12.2 | 4.6 | |
| 0603 | 207.9 | 4.4 | 8.0 | 14.1 | |

?

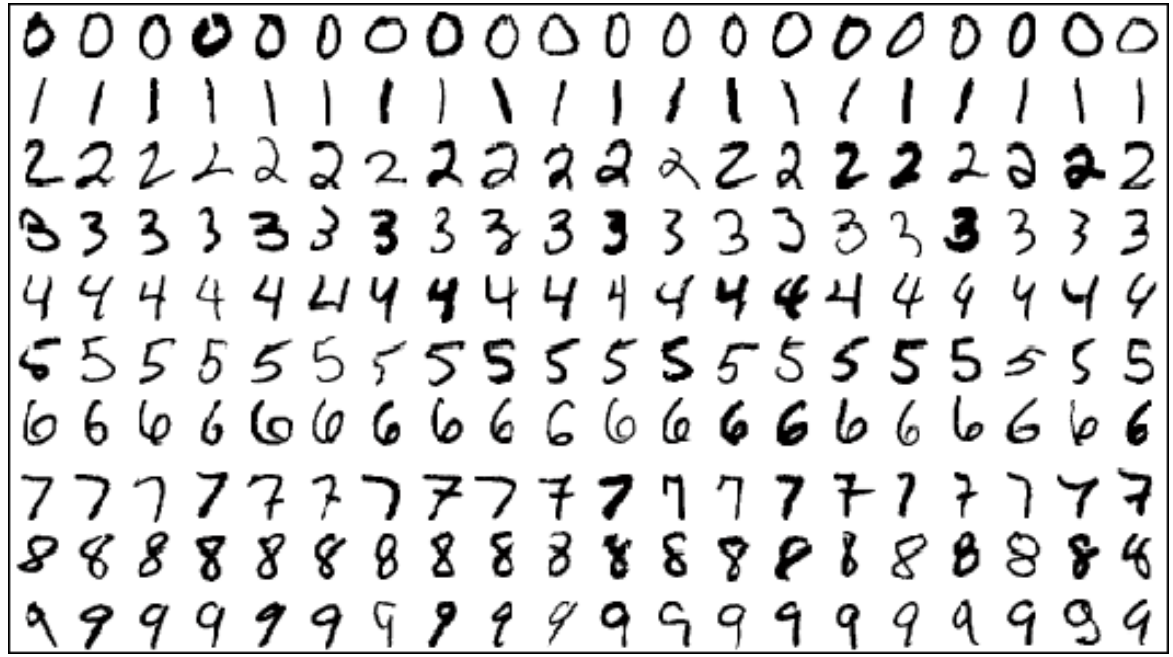
Set de test es útil para evaluar la capacidad de generalización del modelo

# El set de datos MNIST permite construir clasificadores de dígitos a partir de imágenes (OCR)



- MNIST es un set de datos compuesto de imágenes de dígitos escritos a mano.
- Cada imagen muestra un sólo dígito entre 0 y 9. Las imágenes son binarias con una resolución de 28x28 pixeles.
- El dataset consta de 60.000 ejemplos de entrenamiento y 10.000 de test.

El set de datos MNIST permite construir clasificadores de dígitos a partir de imágenes (OCR)



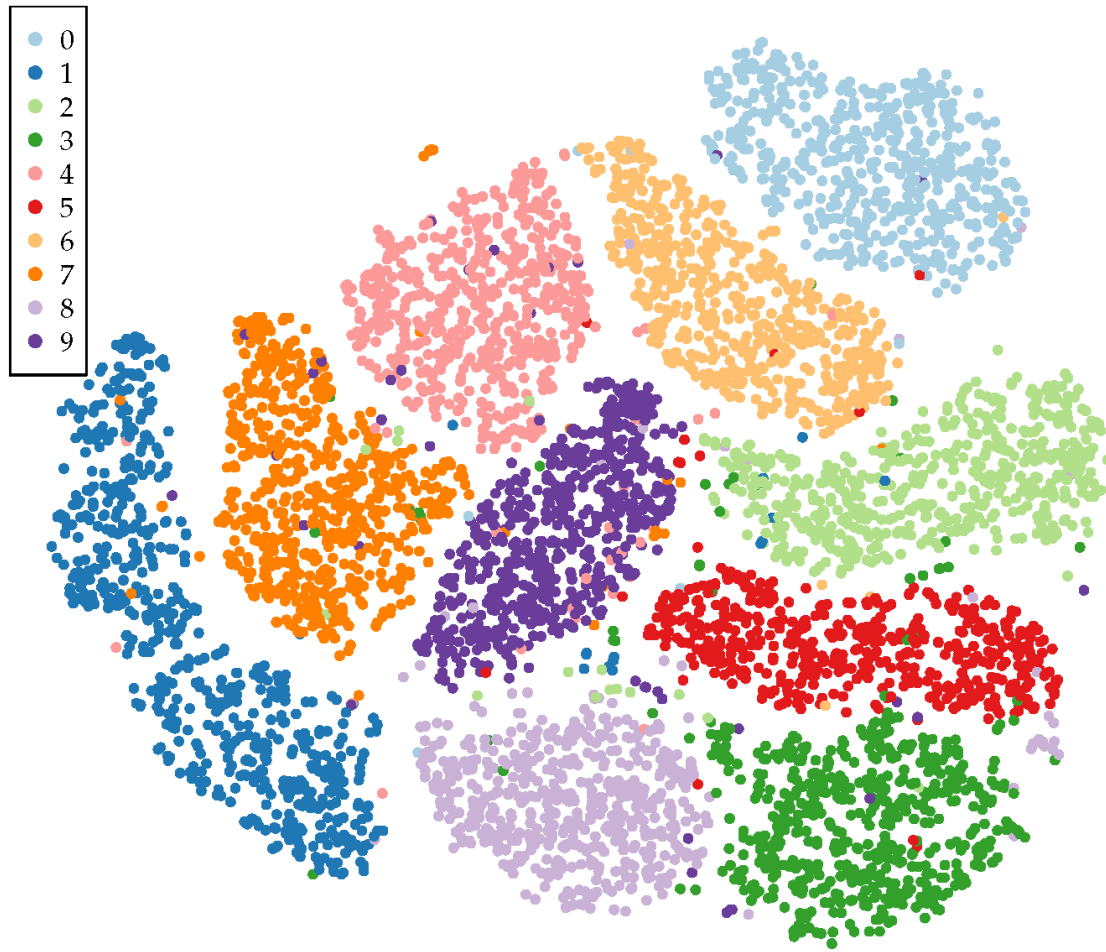
- ¿Cómo podríamos resolver este problema (clasificación de dígitos)?
- Quizá, visualizar el espacio de características nos da una pista.

# ¿Es posible visualizar directamente el espacio de características de MNIST?

NO

¿Por qué?

Usando técnicas de reducción de dimensionalidad (tSNE), es posible transformar el espacio de características




¿Cómo podríamos resolver este problema?

(aka cuál es el algoritmo más simple que podríamos usar)

# Clasificador de k-vecinos cercanos permite realizar la tarea de manera intuitiva

- Dado un dato sin clasificar, su clase se define como el resultado de la votación de los k-vecinos más cercanos.
- Con k=1, se obtienen los siguientes resultados:

Predicción

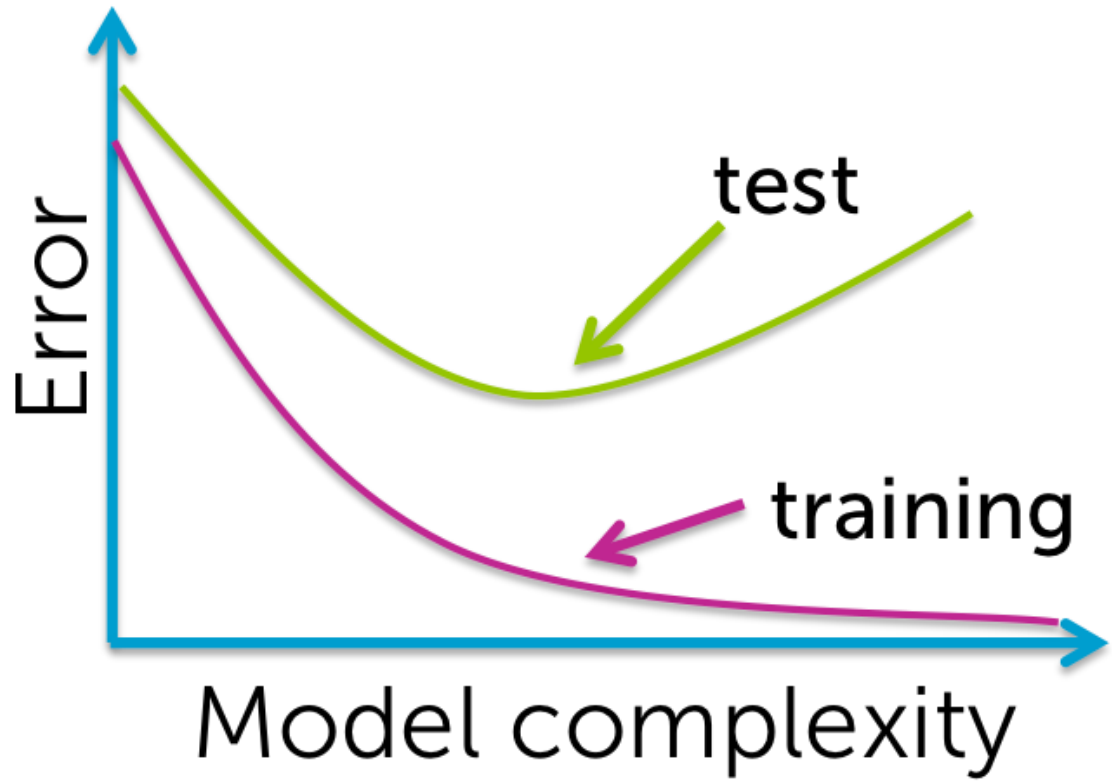
Real

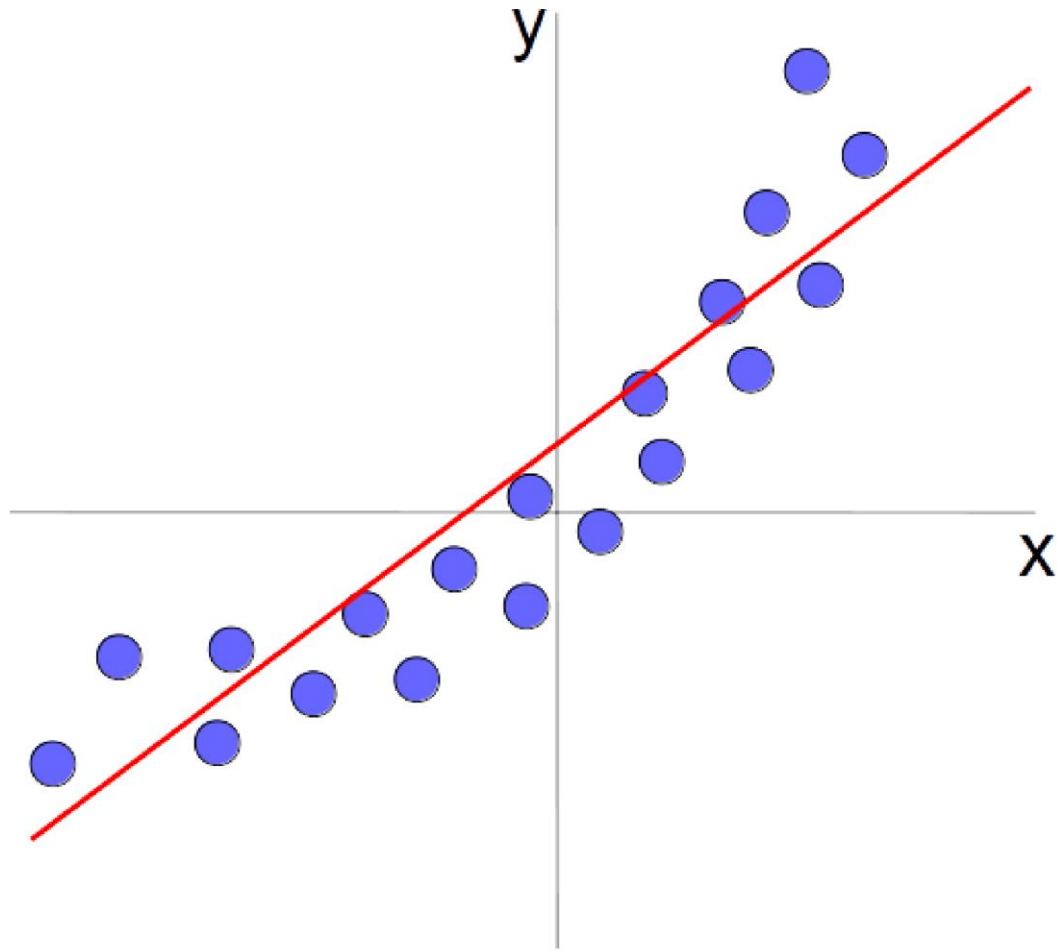
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 972 | 1 | 1 | 0 | 0 | 1 | 3 | 1 | 0 | 0 |
| **1** | 0 | 1129 | 3 | 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| **2** | 7 | 6 | 992 | 5 | 1 | 0 | 2 | 16 | 3 | 0 |
| **3** | 0 | 1 | 2 | 970 | 1 | 19 | 0 | 7 | 7 | 3 |
| **4** | 0 | 7 | 0 | 0 | 944 | 0 | 3 | 5 | 1 | 22 |
| **5** | 1 | 1 | 0 | 12 | 2 | 860 | 5 | 1 | 6 | 4 |
| **6** | 4 | 2 | 0 | 0 | 3 | 5 | 944 | 0 | 0 | 0 |
| **7** | 0 | 14 | 6 | 2 | 4 | 0 | 0 | 992 | 0 | 10 |
| **8** | 6 | 1 | 3 | 14 | 5 | 13 | 3 | 4 | 920 | 5 |
| **9** | 2 | 5 | 1 | 6 | 10 | 5 | 1 | 11 | 1 | 967 |

# En ML, el set de entrenamiento es clave

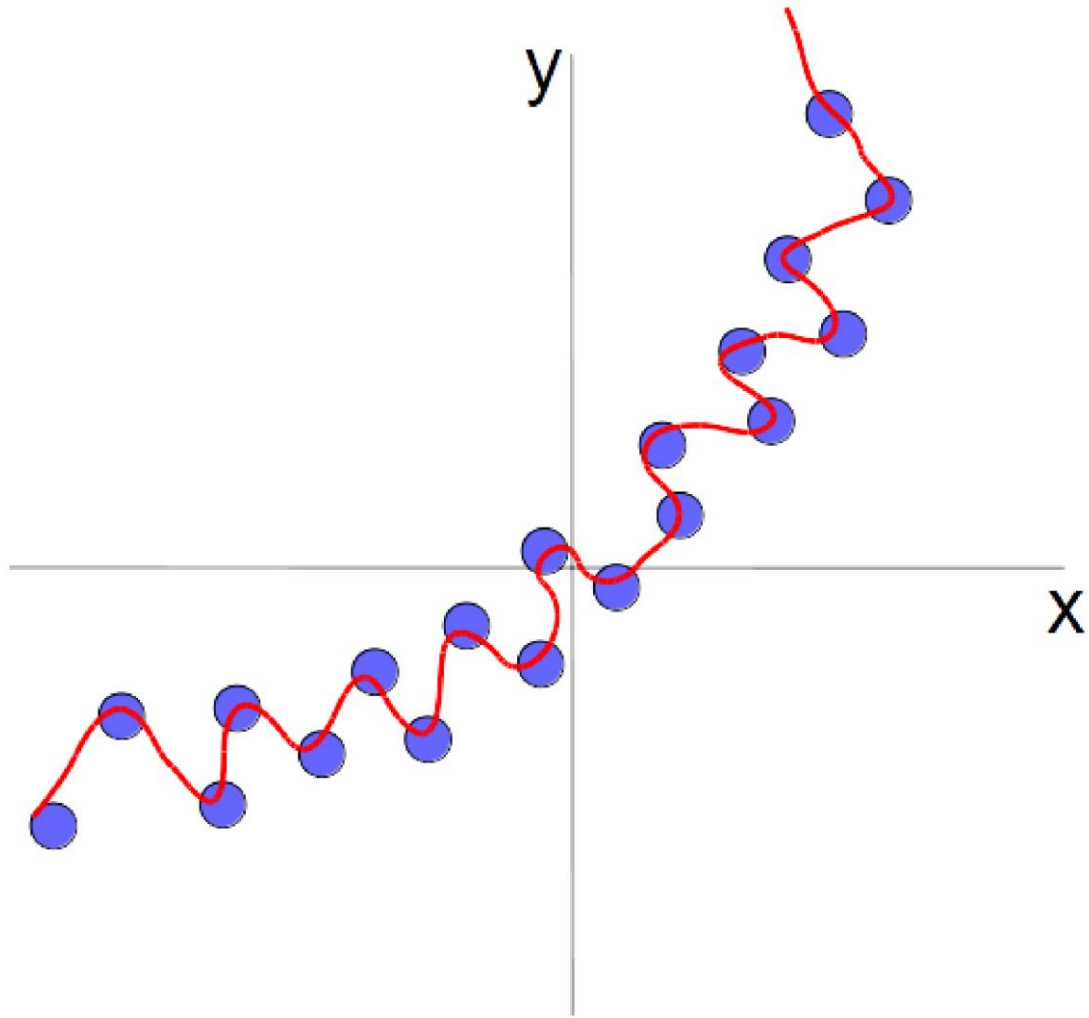
- Este último ejemplo resalta la importancia de un buen set de entrenamiento.
- Los algoritmos de aprendizaje viven y mueren por el set de entrenamiento.
- Lamentablemente, tener un buen set de entrenamiento, no asegura tener buena generalización.
- A continuación, uno de los principales enemigos de ML

El sobreajuste (*overfitting*) ocurre cuando el modelo comienza a memorizar los datos de entrenamiento

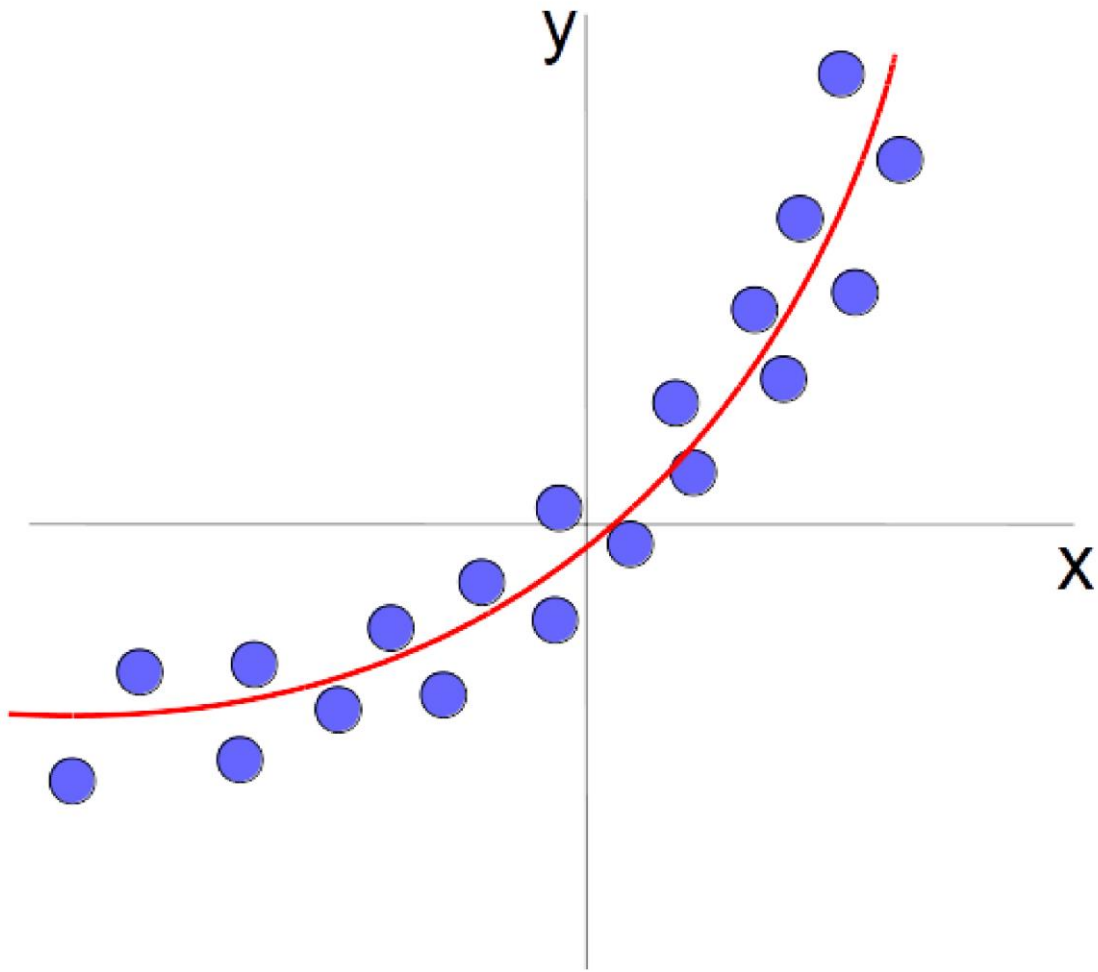





Modelo es demasiado simple para capturar el comportamiento de los datos (*underfitting*).




Modelo es muy complejo, y captura hasta el ruido presente en los ejemplos (*overfitting*).




Modelo tiene la complejidad necesaria para capturar los patrones relevantes, obviando el ruido.

Veamos un ejemplo práctico de esto, usando como base una regresión lineal

- Una regresión lineal busca aproximar una función, a través de una combinación lineal de las variables:

$$\hat{y} = f(x;\theta) = \sum \theta_i x_i$$

- ¿Cómo podemos obtener los valores del parámetro $\theta$ ?
- Si tenemos suficientes datos, pares (x,y), es posible construir una función que nos indique cuán buena es la estimación (mínimos cuadrados).
- El valor de $\theta$ que minimice esta función, entregará la mejor estimación de la función original.

La teoría de aprendizaje busca formalizar el proceso de aprendizaje (inferencia) inductivo

1. Observar un fenómeno
2. Construir un modelo que explique el fenómeno
3. Realizar predicciones usando el modelo

Nada es más práctico que una buena teoría (Vapnik)

# No existe una elección universal de lo que es un buen modelo

*No Free Lunch theorem*

- Si no hay una suposición sobre como se relaciona el pasado (training) con el futuro (test), la predicción es imposible.
- Si no hay una restricción a priori sobre lo esperado, no hay un algoritmo mejor que otro.

Generalización = Datos + Conocimiento

# Es fundamental asumir pseudo estacionariedad en el fenómeno estudiado

1. Observaciones (ejemplos) pasados y futuros son muestreados independientemente desde la misma distribución (i.i.d.).

2. Un resultado de esto, es que mientras más datos tenga, mejor voy a predecir (k-vecinos cercanos)

3. Lamentablemente, esto no se cumple siempre si se tienen conjuntos finitos de datos.

# Formalicemos usando el caso de la clasificación binaria

- Consideremos espacios de entrada y salida $\mathbb{X}$ e $\mathbb{Y}$, respectivamente
- Buscamos construir una función $f : \mathbb{X} \rightarrow \mathbb{Y}$, que prediga $\mathbb{Y}$ a partir de $\mathbb{X}$.
- ¿Qué criterio podemos ocupar para guiar la búsqueda de esta función?

# El concepto de riesgo (o probabilidad de error) nos permite operativizar estas definiciones

- El riesgo de la función g, podemos definirlo como:

$$R(f) = \mathbb{P}(f(\mathbb{X}) \neq \mathbb{Y})$$

- Buscamos una función g, tal que:

$$R(g) = \inf_f R(f)$$

P es desconocida, ¿qué hacemos?

- Utilizamos como proxy el ajuste de las funciones candidatas a los datos, conocido como riesgo empírico:

$$R_n(f) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}_{f(x_i) \neq y_i}$$

# El concepto de riesgo (o probabilidad de error) nos permite operativizar estas definiciones

- Dado lo anterior, sólo bastaría minimizar el riesgo empírico:

$$\hat{f} = \arg\min_{f} R_n(f)$$

¿Cuál es el problema con esto?

- Solución 1: acotar el dominio (empirical risk minimization)

$$\hat{f} = \arg\min_{f \in \mathbb{F}} R_n(f)$$

- Solución 2: penalizar modelos (structural risk minimization)

$$\hat{f} = \arg\min_{f \in \mathbb{F}_d, d \in \mathbb{N}} R_n(f) + pen(d, n)$$

Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación


# IIC2613 – Inteligencia Artificial

## Fundamentos de ML

**Profesor:** Hans Löbel