

plain concepts



DÍA 3: AGENDA

- Hadoop In-Memory: Spark
- Machine Learning
 - Azure ML
 - Spark ML
- **Ejercicio práctico del día**

SPARK

HADOOP IN-MEMORY: SPARK

- Spark sobre YARN
- Análisis exploratorio en tiempo real

¿QUÉ ES SPARK?

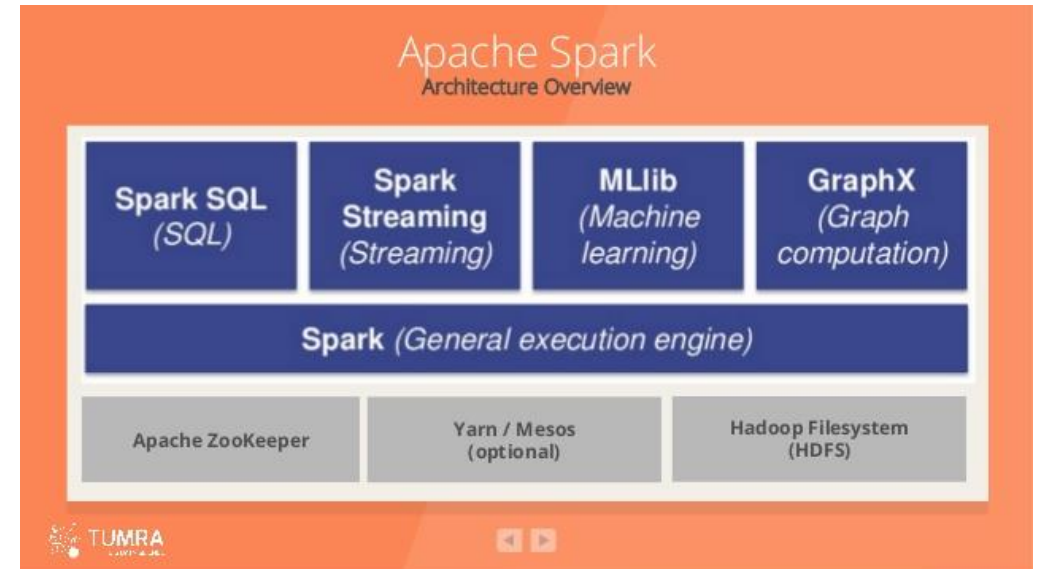
- Spark es un motor de computación de propósito general que soporta operaciones en memoria
- Con Tez, MR y demás nos basamos en un DAG (Grafo Acíclico Dirigido), trabajando de storage a storage
- Esto es ineficiente en casos en los que necesitamos reutilizar un conjunto de datos (working set)
 - Algoritmos iterativos (machine learning)
 - Análisis exploratorio en tiempo real

¿QUÉ ES SPARK?

- Spark trabaja mediante operaciones sobre datasets distribuidos
- Dataset distribuido (RDD, Resilient Distributed Datasets)
 - Colección de objetos repartidos en un clúster, bien en memoria o en disco
 - Construidos mediante transformaciones paralelas
 - Reconstruidos de forma automática si hay un fallo
- Operaciones
 - Transformaciones (map, filter, group by...)
 - Acciones (count, save...)

ARQUITECTURA SPARK

- Spark está dividido en 3 capas principalmente
- La capa de aplicación que es la que el desarrollador usa en función de sus requisitos
- El core de Spark que se encarga de las tareas de ejecución de jobs
- La capa de infraestructura, que depende de cómo ha sido provisionado el cluster

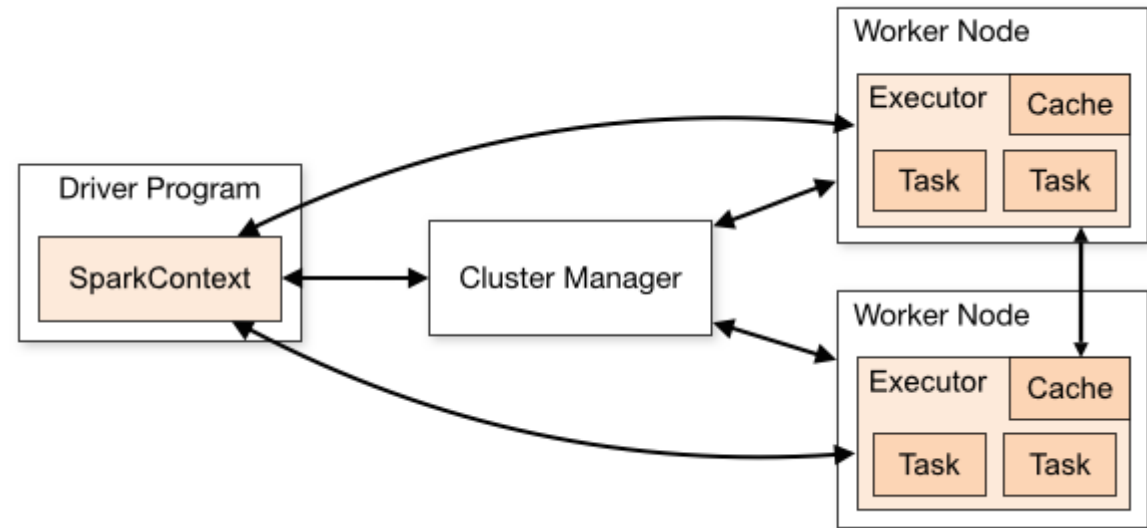


FORMAS DE APROVISIONAR UN CLUSTER

- Standalone
- YARN
- Mesos

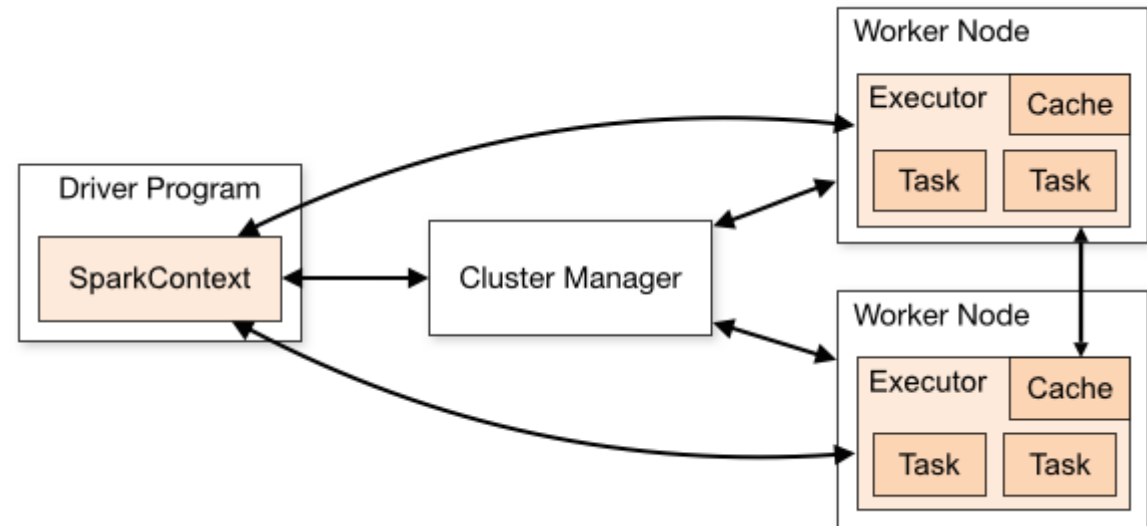
STANDALONE

- La manera más sencilla de montar un clúster de Spark
- Interesante para entornos de pruebas sencillos o pruebas de concepto
- En este caso, el Cluster Manager de la imagen es la propia instancia de Spark



MESOS

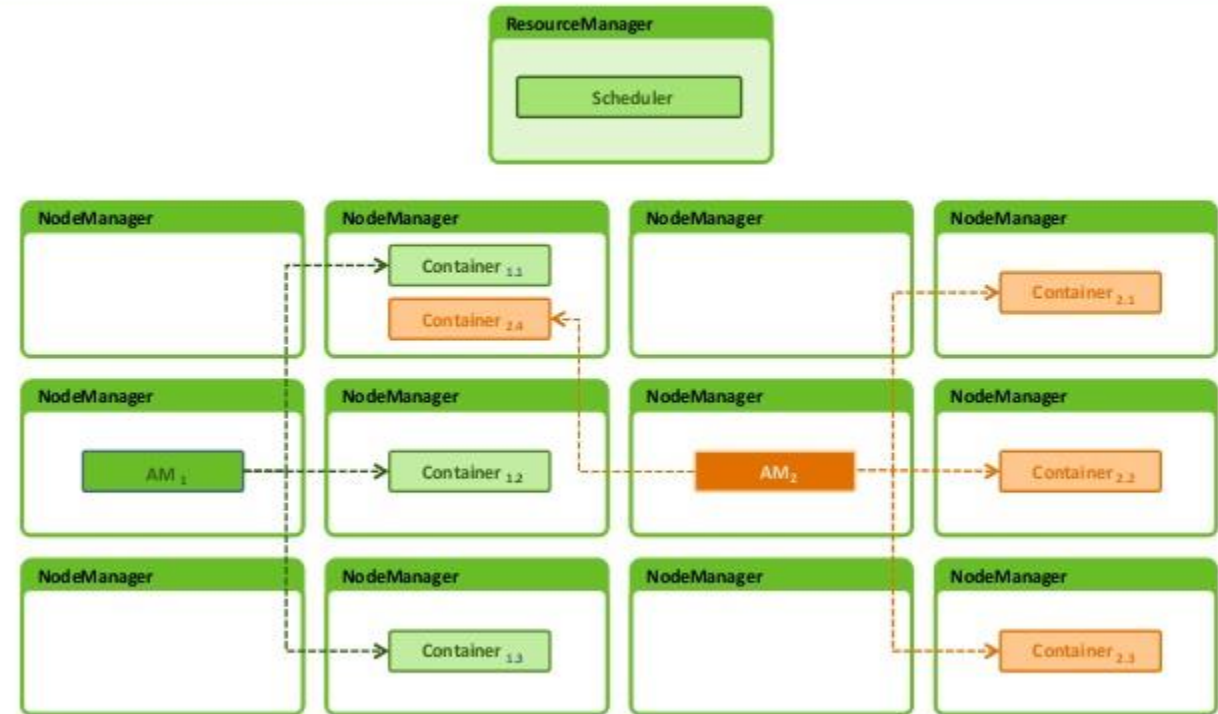
- Mesos modela el cluster como un pool de recursos
- En este caso, el Cluster Manager de la imagen es reemplazado por Mesos
- En teoría, Mesos maneja mejor los recursos ya que cuando va a ejecutar un job, tiene en cuenta otros frameworks corriendo en el cluster.
- Pueden correr diferentes frameworks en el cluster gracias al particionado dinámico de recursos



YARN

- Basicamente, usando YARN delega en él la gestión de los recursos
- Esto es, cuando un job de Spark es encolado para su ejecución, será YARN quien se encargue de llevar a cabo la ejecución

YARN Architecture



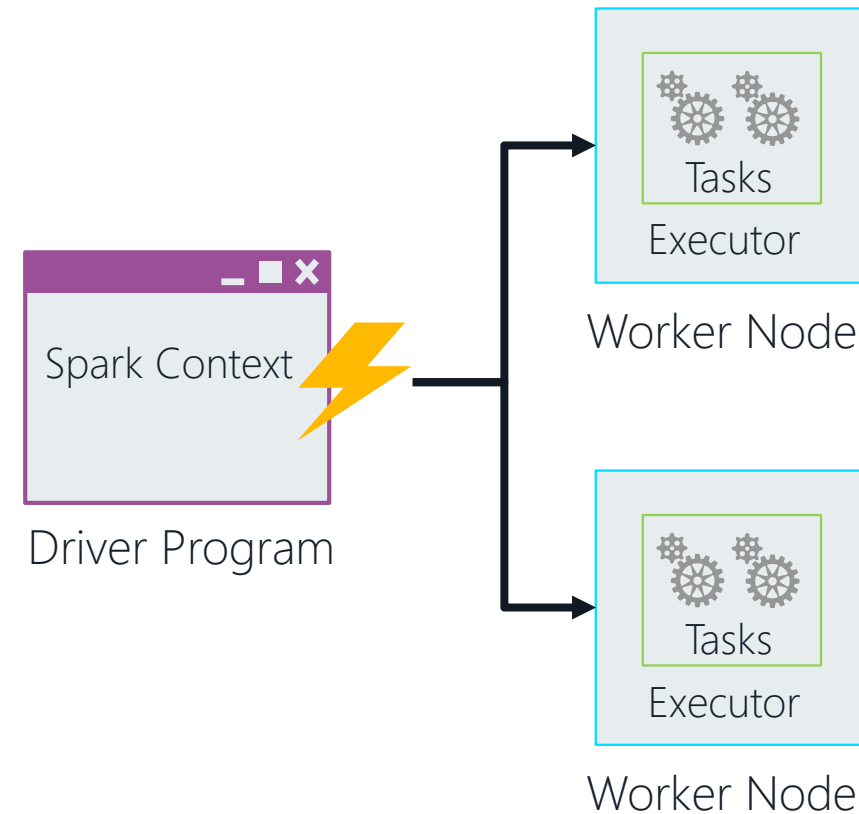
© Hortonworks Inc., 2012

DIFERENCIAS ENTRE YARN Y MESOS

- Las diferencias están en cómo cada uno planifica y maneja los recursos y prioridades
- Mesos está enfocado a datacenters y YARN fue una iteración lógica de Hadoop
- Mesos VS YARN? Depende de lo que necesites. Uno funciona bien gestionando datacenters (Twitter y Airbnb) y YARN funciona bien planificando y manejando Hadoop Jobs
- Pueden ser complementarios? Sí, pero hay un coste: particiones estáticas

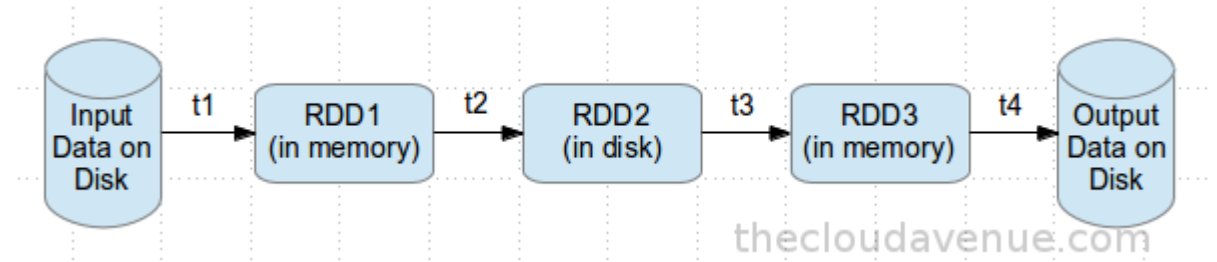
ARQUITECTURA SPARK (JOBS)

- La arquitectura de procesamiento distribuida se compone de
 - Un Driver Program
 - Uno o mas Worker Nodes
- El Driver Program utiliza un context de Spark para conectarse al cluster...
- ...y utiliza los Worker Nodes para realizar operaciones sobre los RDDs



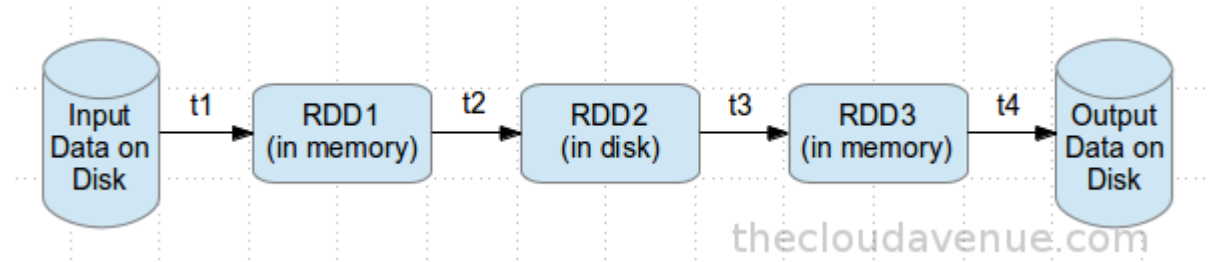
QUÉ SON LOS RDD?

- Resilient Distributed Dataset
 - Inmutable
 - Distribuido
 - Evaluados perezosamente
 - Cacheable
- Pueden estar en memoria o en disco

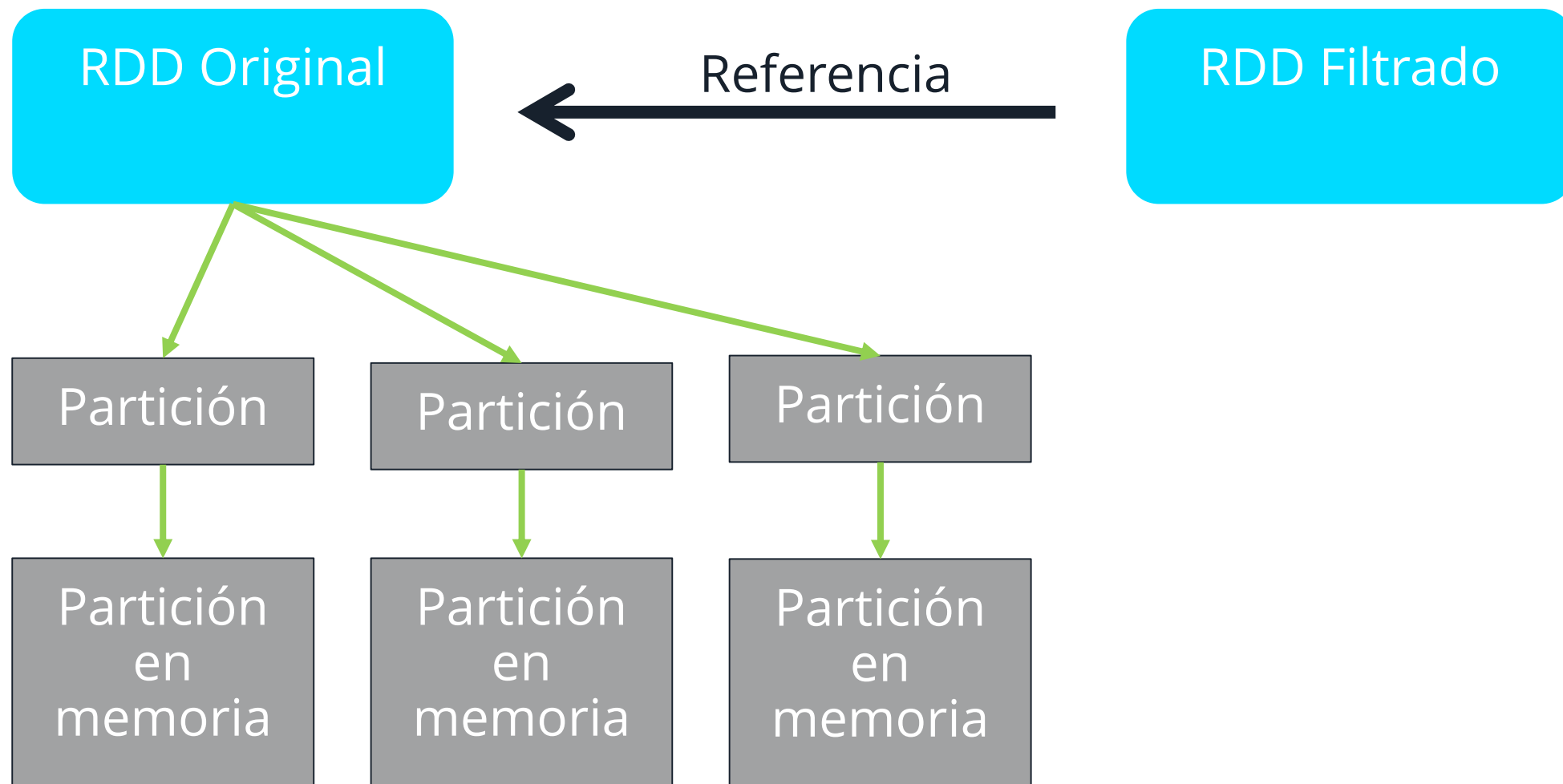


QUÉ SON LOS RDD?

- En realidad, podríamos describir un RDD como una colección de registros de solo lectura y particionada. Los RDD solo pueden ser creados a través de operaciones determinísticas usando datos de una una fuente de datos estable o de otros RDD's
- Surgen como parte de la solución al problema de resolver algoritmos iterativos mediante MapReduce

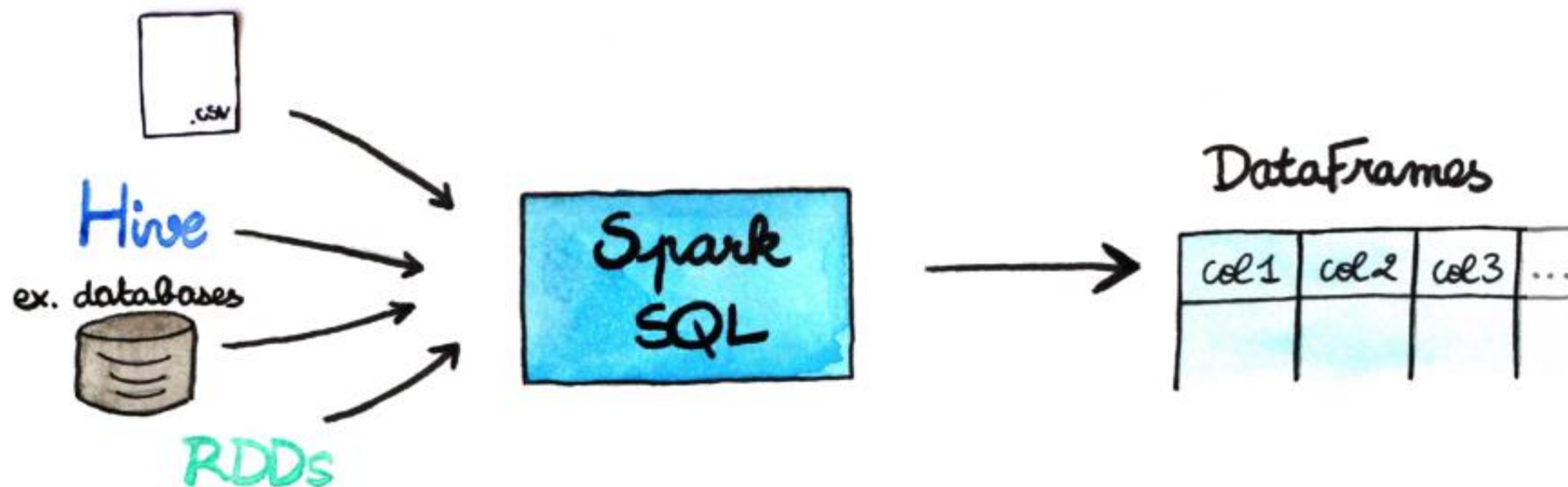


RDD



DATAFRAMES

- Los DataFrames aparece en la versión 1.3 de Spark
- Introducen el concepto de un esquema que describe a los datos
- Son más eficientes puesto que evitan serializaciones innecesarias y pasan los datos de nodo a nodo
- Son más ligeros en memoria (off-heap)



OPERACIONES

- Transformaciones
 - Crean un nuevo RDD al transformar uno existente
- Acciones
 - Devuelven resultados al Driver Program o a un fichero de salida
- Spark utiliza evaluación perezosa
 - Nada se ejecuta hasta llegar a una acción
 - Los RDDs se recomputan en cada acción

OPERACIONES

- La mayoría de operaciones consisten en pasar una función a una transformación o una acción
- Las funciones pueden ser
 - Declaradas de forma explícita
 - Pasadas inline
 - Python usa la keyword lambda
 - Scala usa la sintaxis =>
 - Java usa function classes o lambdas (Java 8)

```
RDD.filter(function)
```

```
def containsMSTag(txt):  
    return "#ms" in txt  
  
msTwts = txtRDD.filter(containsMSTag)
```

```
#Python  
msTwts = txtRDD.filter(lambda txt: "#ms" in txt)
```

```
//Scala  
msTwts = txtRDD.filter(txt => txt.contains("#ms"))
```

TRANSFORMACIONES COMUNES

- filter: Crea un RDD filtrado
- flatMap: Aplica una function a cada elemento, retornando multiples elementos a un nuevo RDD
- map: Aplica una function a cada element, retornan un elemento a un nuevo RDD
- reduceByKey: agrega valores por cada clave en un RDD clave-valor

```
txt = sc.parallelize(["the owl and the pussycat", "went to sea"])
```

```
{["the owl and the pussycat"], ["went to sea"]}
```

```
owlTxt = txt.filter(lambda t: "owl" in t)  
{["the owl and the pussycat"]}
```

```
words = owlTxt.flatMap(lambda t: t.split(" "))  
{["the"], ["owl"], ["and"], ["the"], ["pussycat"]}
```

```
kv = words.map(lambda key: (key, 1))  
{["the",1], ["owl",1], ["and",1], ["the",1], ["pussycat",1]}
```

```
counts = kv.reduceByKey(lambda a, b: a + b)  
{["the",2], ["owl",1], ["and",1], ["pussycat",1]}
```

ACCIONES COMUNES

- reduce: Agrega los elementos de un RDD utilizando una función con dos argumentos
- count: Devuelve el numero de elementos del RDD
- first: Devuelve el primer element del RDD
- collect: Devuelve el RDD como un array
- saveAsTextFile: Almacena el RDD como un fichero de texto en el path proporcionado

```
nums = sc.parallelize([1, 2, 3, 4])
```

{[1], [2], [3], [4]}

```
nums.reduce(lambda x, y: x + y)
```

9

```
nums.count()
```

4

```
nums.first()
```

1

```
nums.collect()
```

[1, 2, 3, 4]

```
nums.saveAsTextFile("/results")
```

/results/part-00000

DESPLEGANDO JOBS

- Los Jobs se envían al clúster mediante una llamada parametrizada a spark-submit

./bin/spark-submit

--class

org.apache.spark.examples.SparkPi

--master local[8]

/path/to/examples.jar 100

- **--class:** El punto de entrada de la aplicación(e.g. org.apache.spark.examples.SparkPi)
- **--master:** La URL del cluster(e.g. spark://23.195.26.187:7077)
- **--deploy-mode:** Para desplegar el job en los worker nodes (cluster) o desplegarlo de manera local (client)
- **--conf:** Configuración arbitraria de Spark, el format es clave-valor key=value format. Si contiene espacios el valor, usar comillas “. Ejemplo: “key=value foo”
- **application-jar:** Ruta hacia la aplicación que se va a ejecutar. Esta aplicación debe ser autocontenida, es decir, todas las dependencias deben estar dentro del jar. Esta ruta debe ser visible desde cualquier parte del cluster, como por ejemplo, un ruta a HDFS (hdfs://) que está presente en todos los nodos
- **application-arguments:** En caso de que existan, argumentos que se le pasen al punto de entrada de la aplicación

plain concepts

SPARK



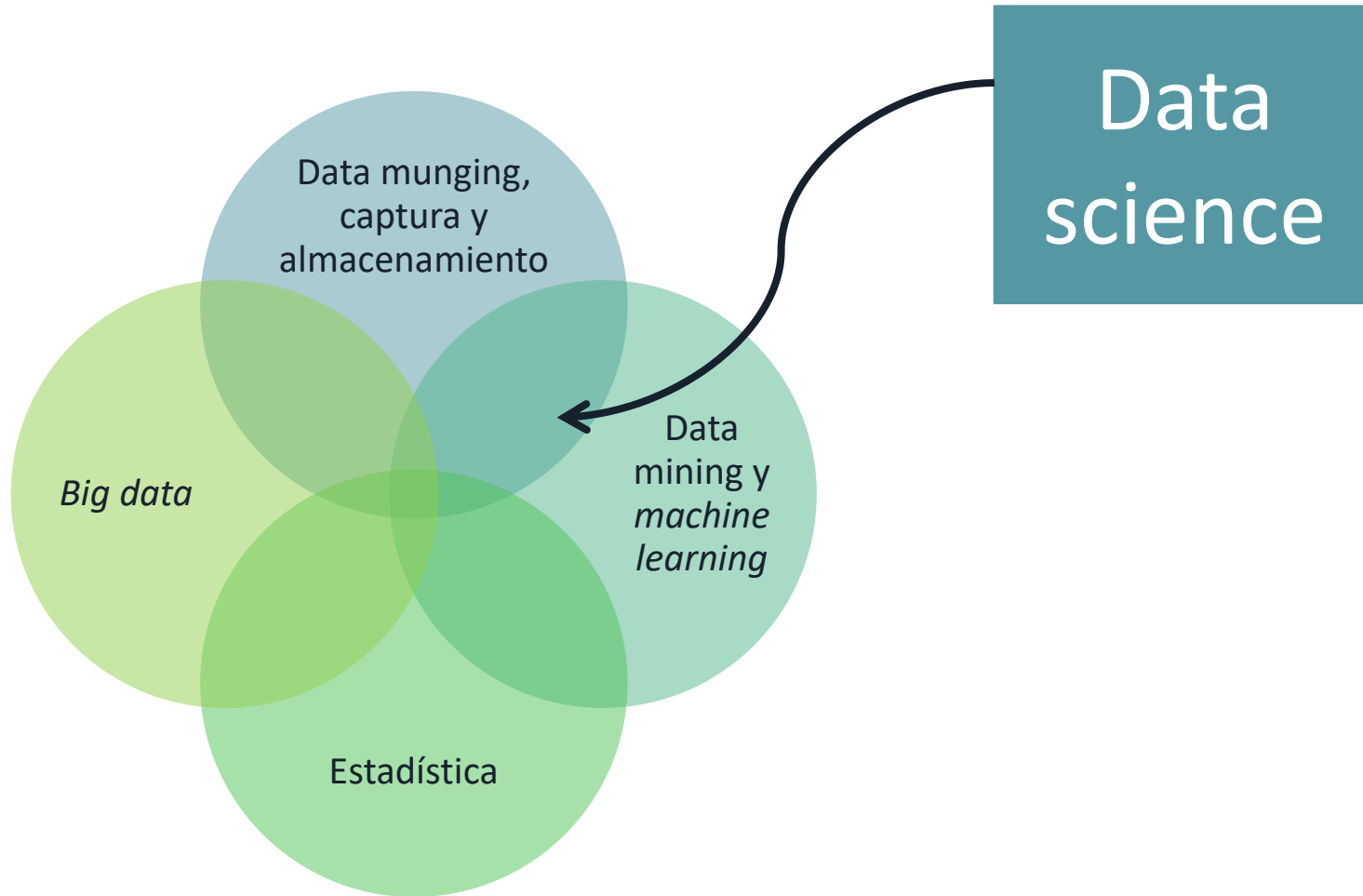
MACHINE LEARNING

MACHINE LEARNING

Algoritmos para
detector patrones
interesantes en los
datos.

Inteligentes y
completamente
automáticos (*¡que más
quisiéramos!*)

MACHINE LEARNING

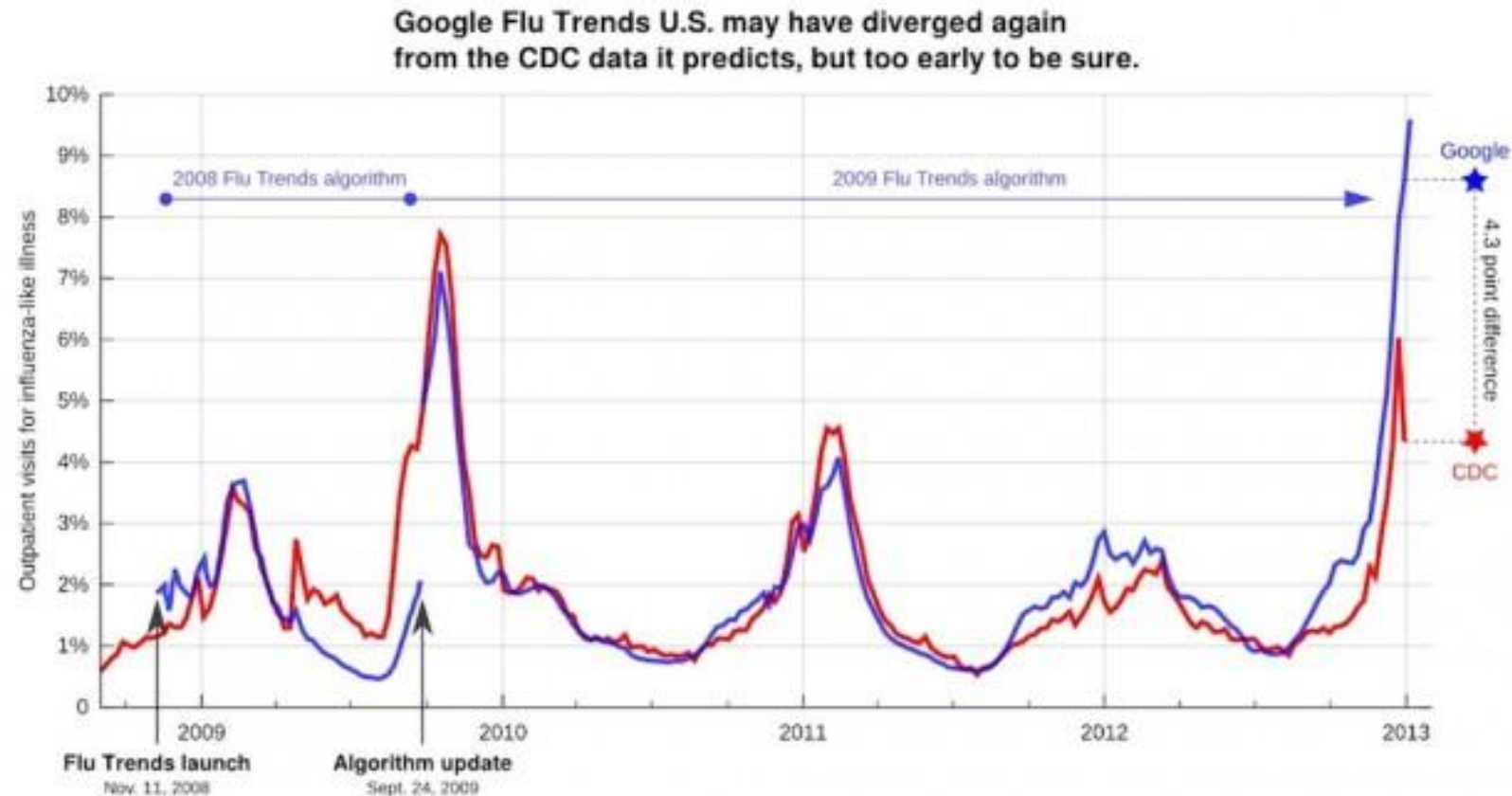


ESCENARIOS COMUNES

Dominio	Escenarios
Servicios Financieros	Modelado de Riesgos
	Análisis de Amenazas y Detección de Fraude
Media y Entretenimiento	Publicidad dirigida
	Motores de Recomendaciones
Comercio	Análisis de Sentimiento
	Análisis de Transacciones en Punto de Venta
Telecomunicaciones	Análisis de CDRs (Call Detail Records)
Gobierno	Monitorización medioambiental
	Congestión y re-routing de tráfico
Sanidad	Investigación (Genómica, Cáncer, etc..)
	Detección temprana de pandemias
Ingeniería	Mantenimiento Predictivo

ANÁLISIS

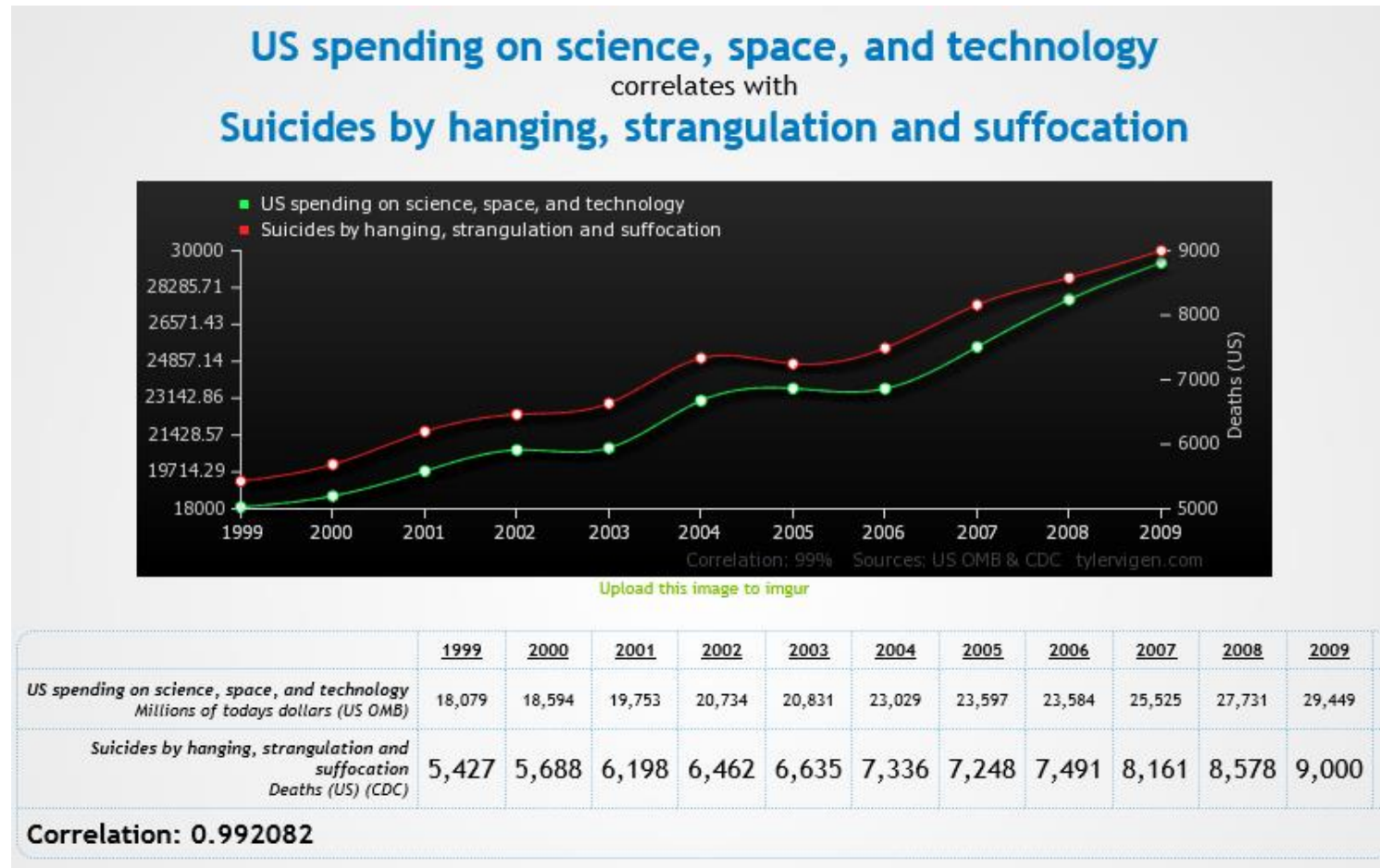
GRIPE 2009



Sources: <http://www.google.org/flutrends/us>, CDC (IIRet data from <http://gis.cdc.gov/grasp/fluview/fluportals/dashboard.html>),
Cook et al. (2011) Assessing Google Flu Trends Performance in the United States during the 2009 Influenza Virus A (H1N1) Pandemic,
PLoS ONE 6(8): e23610, doi:10.1371/journal.pone.0023610.

Data as of Jan. 12, 2013. Keith Winstein (keithw@mit.edu)

CORRELACION != CAUSALIDAD



Fuente: <http://www.tylervigen.com>

VOLVIENDO A ML, ¿LO NECESITAMOS?

¿Tenemos **preguntas** de negocio para las que no tengamos respuestas?

¿Tenemos los **datos** para responderlas?

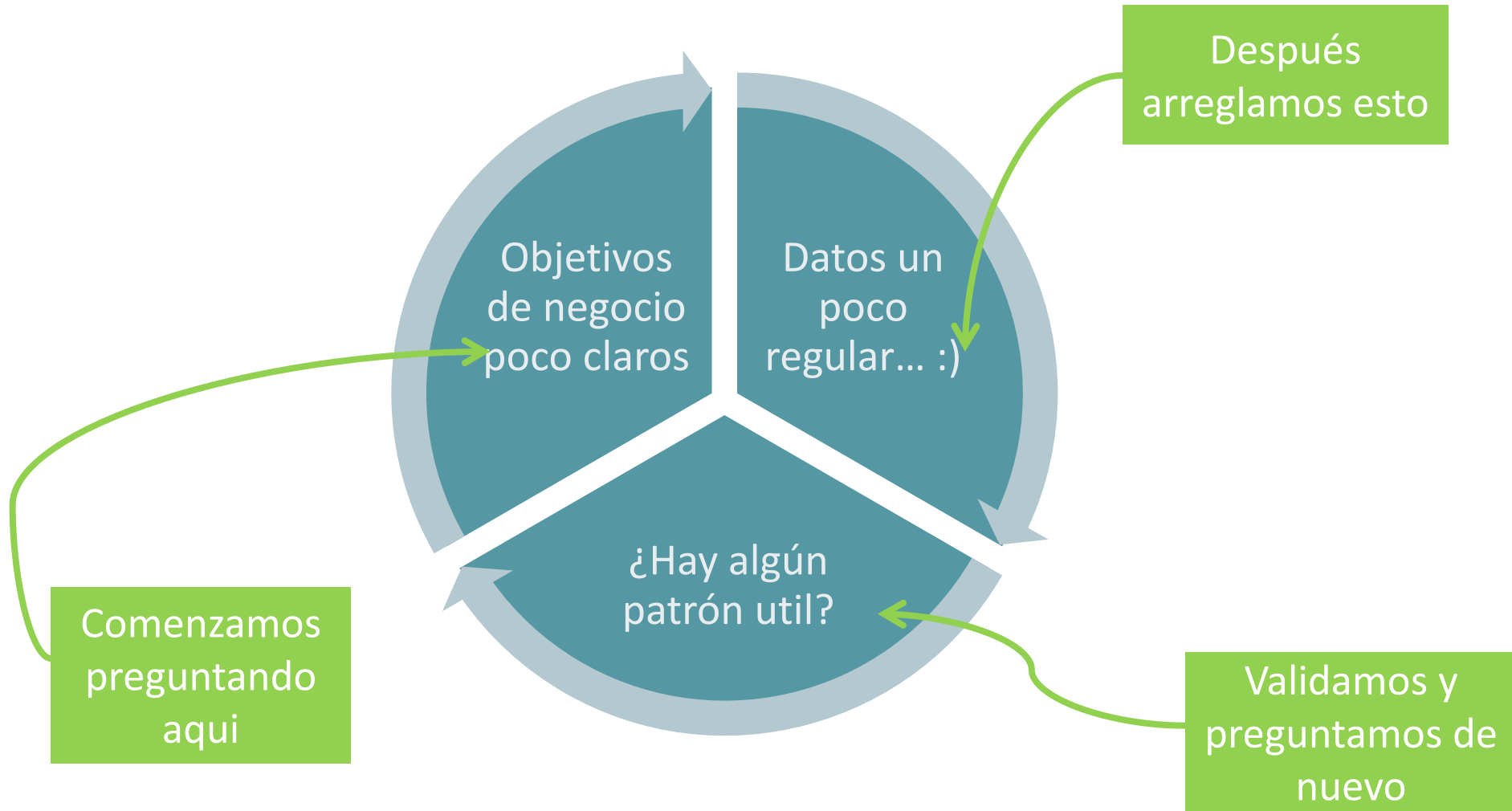
EL PROCESO



LAS PERSONAS INVOLUCRADAS



¿POR DONDE EMPEZAMOS?

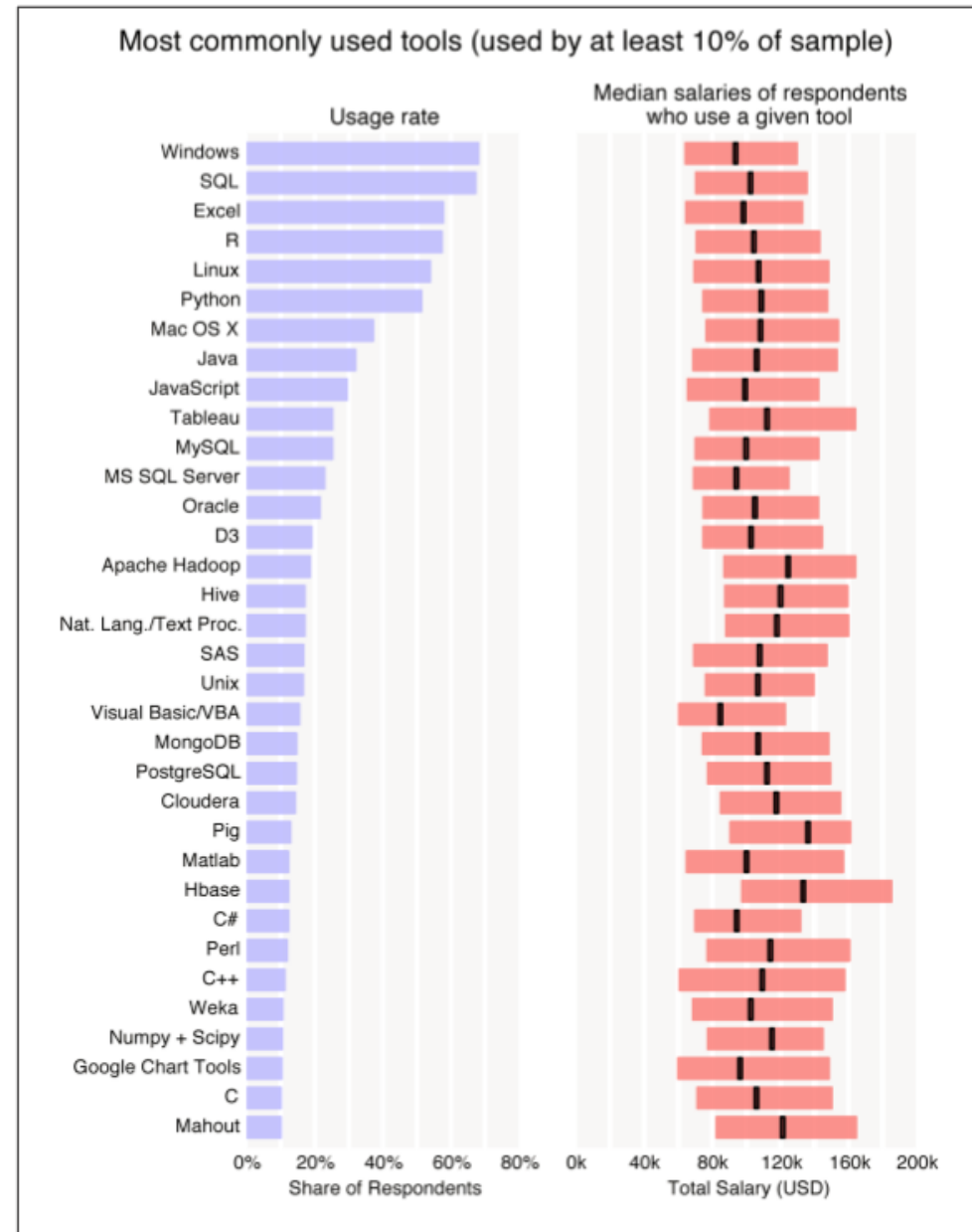


EJEMPLO: DETECCION DE FRAUDE



HERRAMIENTAS

"2014 Data Science Salary Survey"
[O'Reilly]



MIS HERRAMIENTAS

Principales

- SQL Server
- Excel + PowerBI
- AzureML
- R (con Rstudio y Revo)
- Hadoop (Hive y Spark)

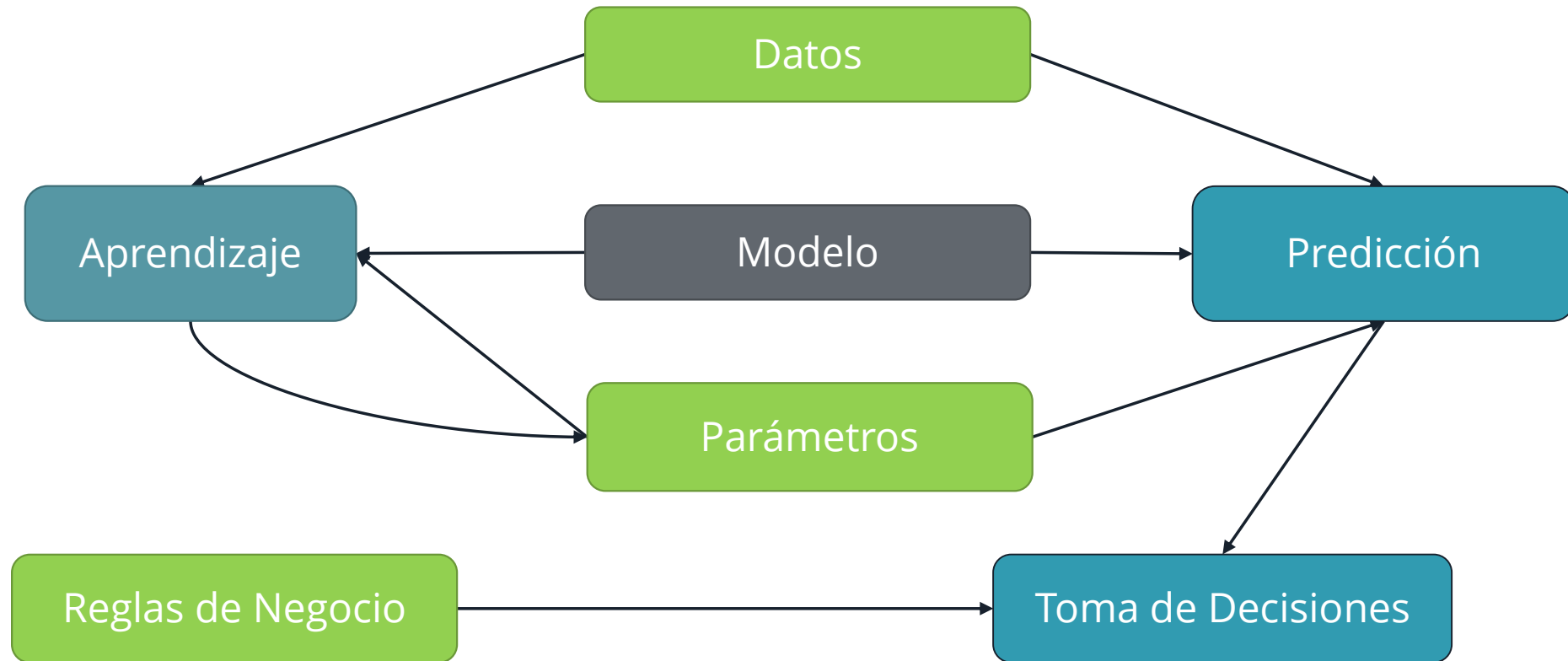
Secundarias

- Python + Pandas

No, si puedo evitarlo :)

- Mahout
- SAS
- SPSS

CONCEPTOS FUNDAMENTALES



CLASES DE PROBLEMAS DE ML

Clasificación

- Asignar una categoría
- Ej: Restaurante (Chino | Indio | Italiano | Japo)

Regresión

- Predicción de un valor real para cada elemento
- Ej: valor de una compra, una temperatura, etc.

Ranking

- Ordenar los elementos de acuerdo a un criterio
- Ej: resultados de una búsqueda en la web

Clustering

- Particionado de los elementos en grupos heterogeneos
- Ej: clustering de posts de twitter posts por temática

Reducción de Dimensionalidad

- Transformación de una representación inicial en una representación de menor dimensionalidad
- Ej: preprocesado de imagines, reconocimiento de voz, etc.

SPARK MLIB ALGORITHMS

- Basic stats
 - Summary stats
 - Correlations
 - Stratified sampling
 - Hypothesis testing
 - Random data generation
- Classification
 - Linear models (SVMs, logistic, linear)
 - Naive Bayes
 - Decision trees
 - Ensemble of trees
 - Random forests
 - Gradient-boosted trees
- Simulation
 - Montecarlo
- Collaborative clustering
 - Alternating least squares
- Clustering
 - K-means
 - Gaussian mixture
 - Power iteration clustering
 - Latent Dirichlet allocation
- Dimensionality reduction
 - SVD
 - PCA

TRES TIPOS

Supervisados

- Predicción
- Clasificación
- Regresión

No Supervisados

- Clustering
- Reducción de Dimensionalidad
- Relacion de Atributos / Selección de Atributos

Refuerzo

- Toma de Decisiones

CLASIFICACIÓN

CLASIFICACION

- Ejemplo de análisis de explosiones de alcantarillas

Modelo de la alcantarilla: [5 3 120 12 1 0]

Número de eventos el año pasado
Número de incidentes serios el año pasado
Número de cables electricos anteriores a 1930
Número de cables eléctricos
Alcantarilla con ventilación?
Inspeccionada?

CLASIFICACION

- Cada observación se representa por un vector de características

Modelo de la alcantarilla :

[5	3	120	12	1	0]	-1
[0	0	89	5	1	1]	1
[1	0	20	0	0	1]	-1



Features, características o X Etiquetas, labels, Y

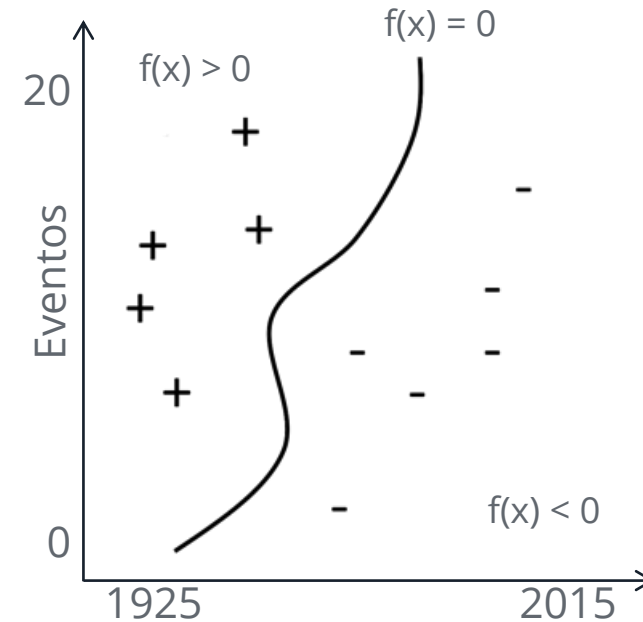
(Predictores,
covariables, variables
independientes)

CLASIFICACION

- Dado un conjunto de entrenamiento (x_i, y_i) para $i=1\dots n$, queremos crear un modelo de clasificación f que pueda predecir una etiqueta y para un valor de x nuevo

Modelo de la alcantarilla: [1925 15]

Fecha de instalación del cable
Número de eventos año pasado



CLASIFICACION

- Binarios
- Multivariados
- Casos de Uso
 - Reconocimiento automatizado de escritura
 - Detección de SPAM
 - Detección de Fraudes
 - Customer Churn (fuga de clientes)
 - Reconocimiento Vocal
 - Reconocimiento de Imagenes
 - Etc.

REGRESIÓN

REGRESION

- Util para predecir valores reales:
 - ¿Cuántas conversions vamos a tener en esta campaña esta semana?
 - ¿Cuántas televisiones venderemos el año que viene?
 - ¿Cuánto gana esta persona en base a su información demográfica?

REGRESION

- Cada observación se representa por un vector de características

Modelamos una persona así:

[5	3	120	12	1	0]	83
[0	0	89	5	1	1]	32
[1	0	20	0	0	1]	-10



Features, características o X



Etiquetas, labels, Y

(Predictores,
covariables, variables
independientes)

REGRESION

- Cada observación se representa por un vector de características

Modelamos una persona así:

$$\begin{bmatrix} 5 \\ 0 \\ 1 \end{bmatrix}$$


Una única feature

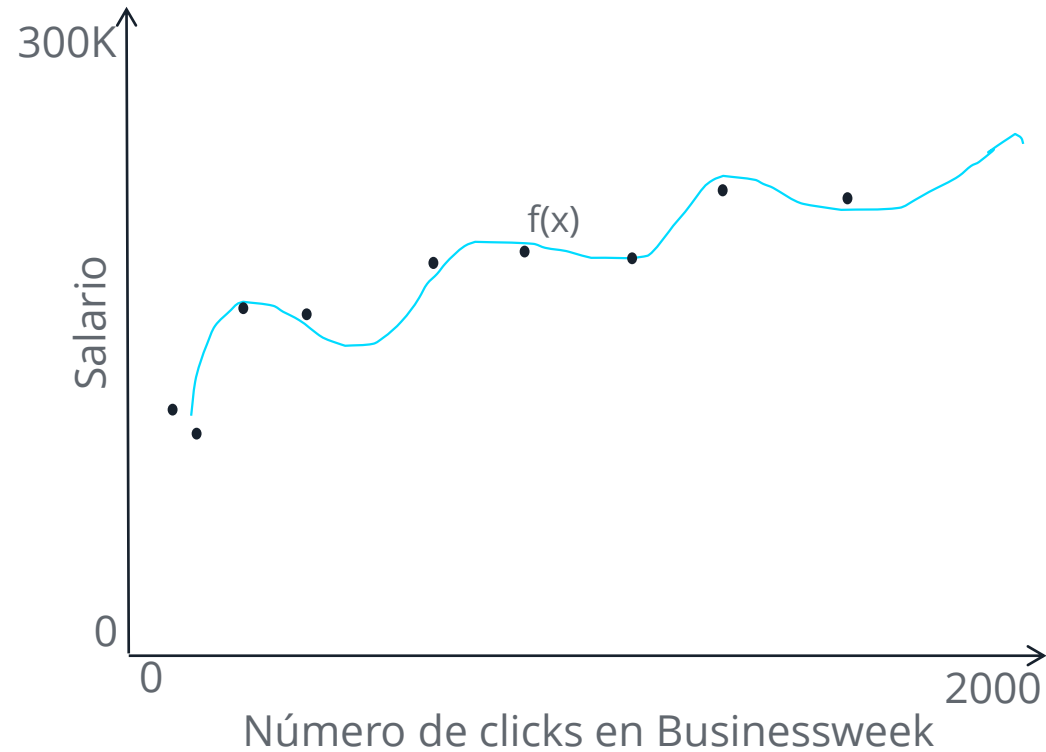
$$\begin{matrix} 83 \\ 32 \\ -10 \end{matrix}$$


Y

REGRESION

$f(x) = \text{funcion}(\text{Número de clicks en BusinessWeek})$

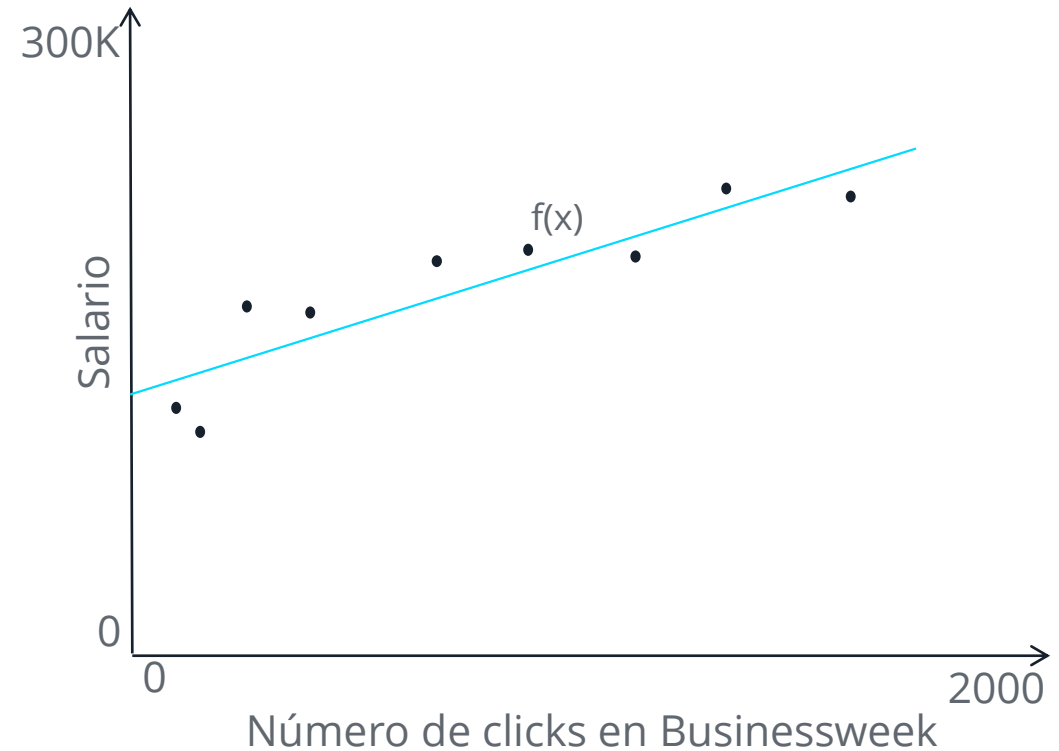
(Overfitting?)



REGRESION

$$\begin{aligned} f(x) &= \text{function}(\text{Número de clicks en BusinessWeek}) \\ &= 5K * \text{Número de clicks en BusinessWeek} + 100K \end{aligned}$$

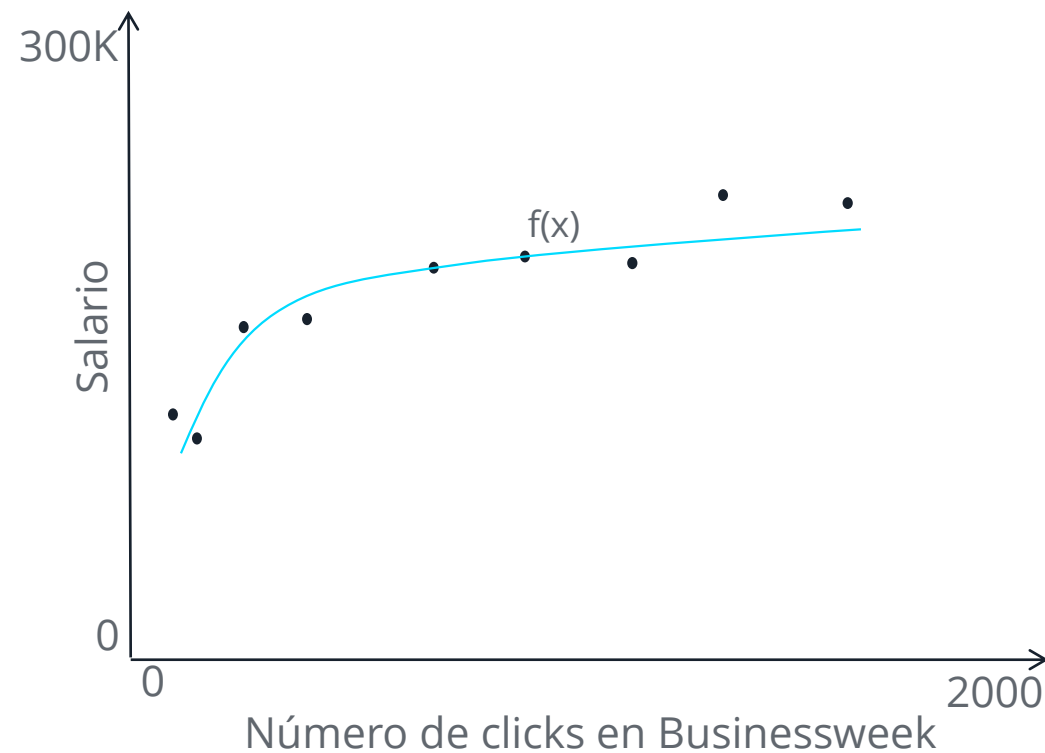
(Underfitting?)



REGRESION

$f(x) = \text{functionpol}(\text{Número de clicks en BusinessWeek})$

(Perfecta?)

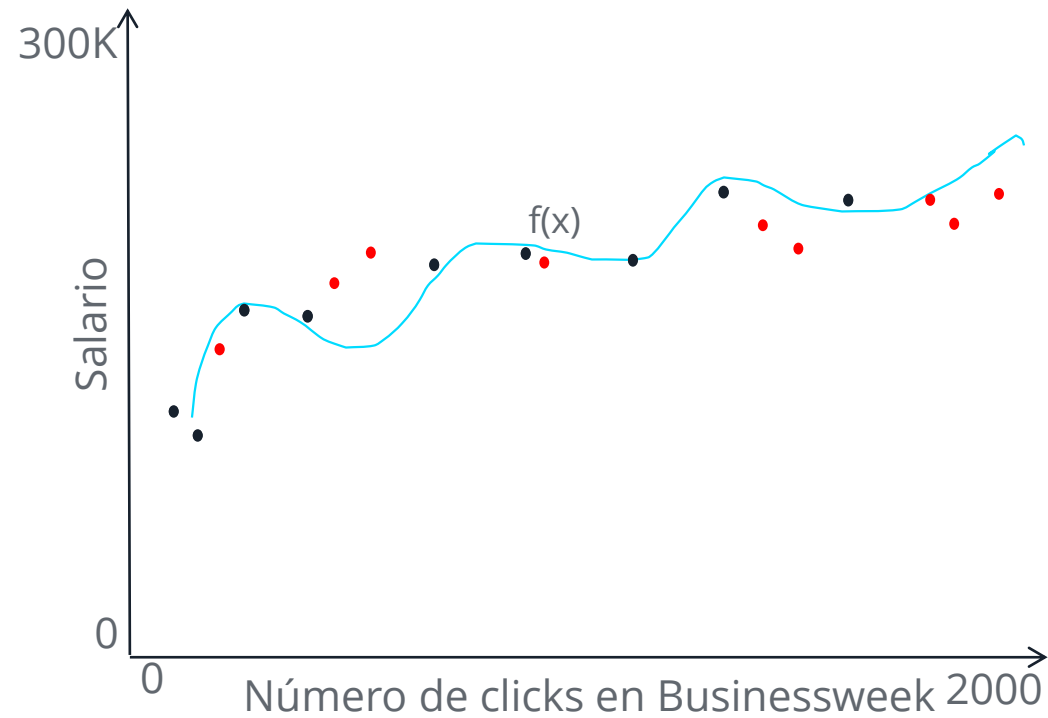


REGRESION

- Salario Estimado
 - $f(x) = \text{funcion}(\text{número de visitas a sitios de muebles, número de clicks en Businessweek, número de gente distinta a la que se envía emails por día, número de compras por encima de 5K en el ultimo mes, número de visitas a aerolíneas})$
- Por Ejemplo:
 - $f(x) = 3 * \text{número de visitas a sitios de muebles}$
 - $+10 * \text{número de clicks en Businessweek}$
 - $+100 * \text{número de gente distinta a la que se envía emails por día}$
 - $+2 * \text{número de compras por encima de 5K en el ultimo mes}$
 - $+10 * \text{número de visitas a aerolíneas}$

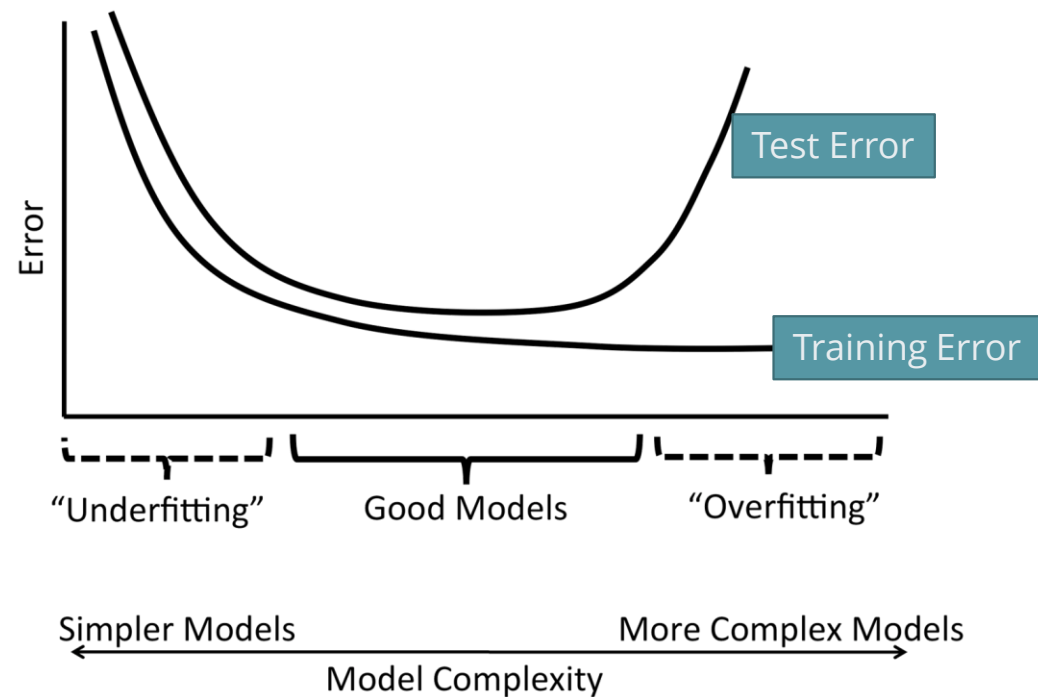
OVERFITTING Y UNDERFITTING

OVERFITTING Y UNDERFITTING



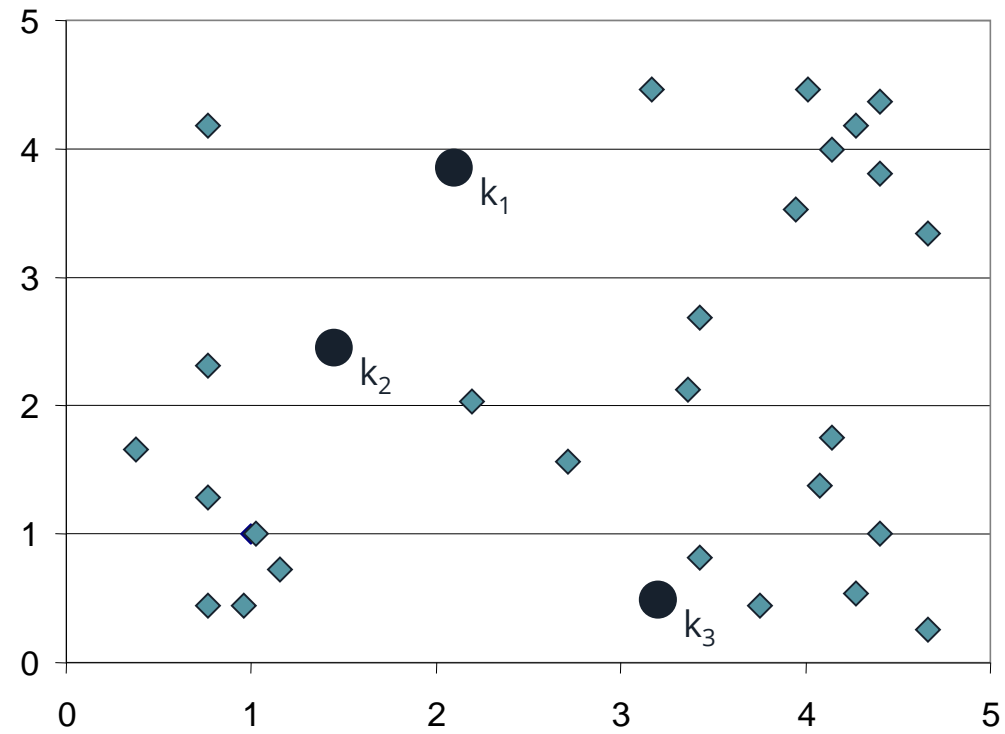
OVERFITTING Y UNDERFITTING

- La navaja de Occam:
 - Los mejores modelos son los más simples que se amolden bien a los datos

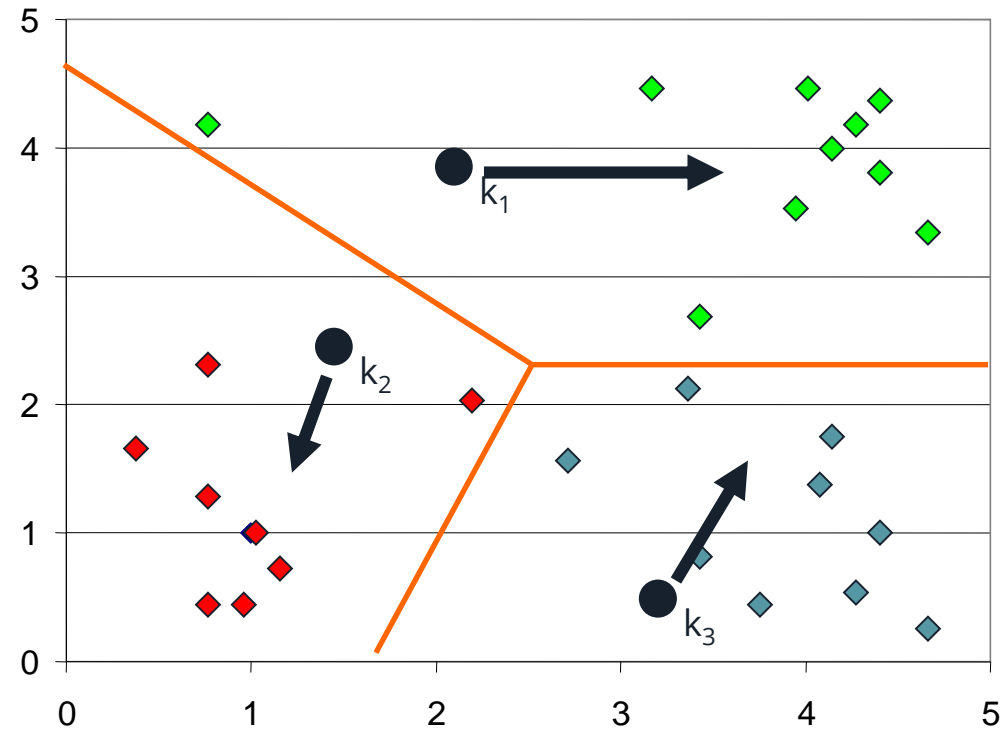


CLUSTERING

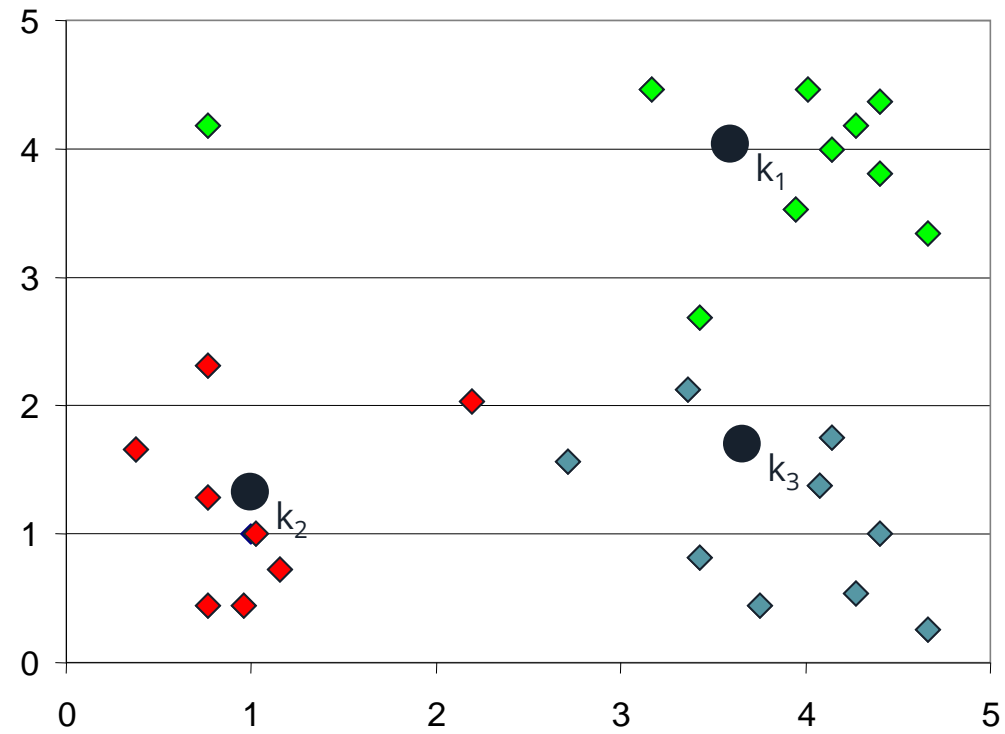
K-MEANS



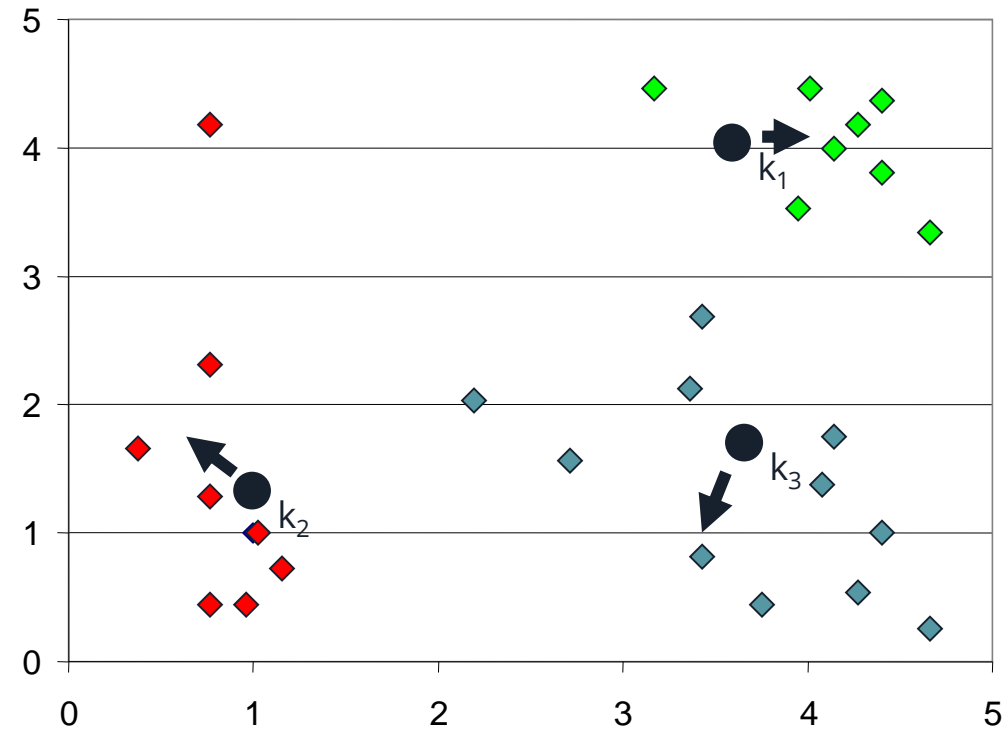
K-MEANS



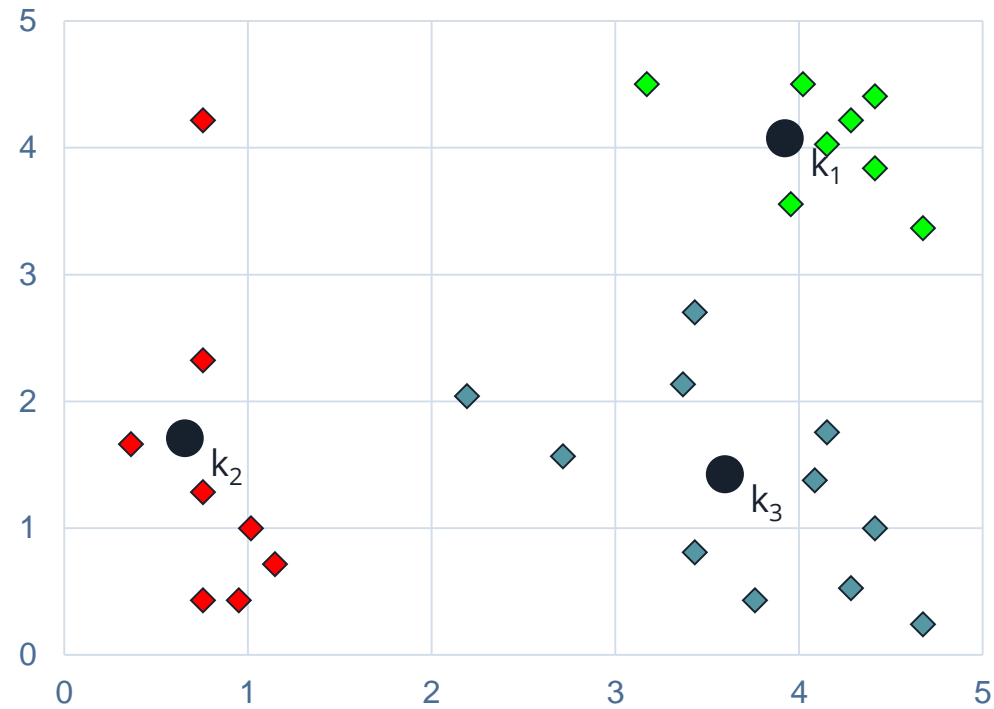
K-MEANS



K-MEANS



K-MEANS



ARBOLES DE DECISION

EJEMPLO: PARTIDO DE TENIS

Día	Situación	Temperatura	Humedad	Viento	Partido
1	Soleado	Calor	Alta	Debil	No
2	Soleado	Calor	Alta	Fuerte	No
3	Nublado	Calor	Alta	Debil	Si
4	Lluvia	Templado	Alta	Debil	Si
5	Lluvia	Frio	Normal	Debil	Si
6	Lluvia	Frio	Normal	Fuerte	No
7	Nublado	Frio	Normal	Fuerte	Si
8	Soleado	Templado	Alta	Debil	No
9	Soleado	Frio	Normal	Debil	Si
10	Lluvia	Templado	Normal	Debil	Si
11	Soleado	Templado	Normal	Fuerte	Si
12	Nublado	Templado	Alta	Fuerte	Si
13	Nublado	Calor	Normal	Debil	Si
14	Lluvia	Templado	Alta	Fuerte	No

¿Que pasará el día 15?

Llueve, alta temperatura,
alta humedad y poco
viento.

EJEMPLO: PARTIDO DE TENIS

Dia	Situación	Temperatura	Humedad	Viento	Partido
1	Soleado	Calor	Alta	Debil	No
2	Soleado	Calor	Alta	Fuerte	No
3	Nublado	Calor	Alta	Debil	Si
4	Lluvia	Templado	Alta	Debil	Si
5	Lluvia	Frio	Normal	Debil	Si
6	Lluvia	Frio	Normal	Fuerte	No
7	Nublado	Frio	Normal	Fuerte	Si
8	Soleado	Templado	Alta	Debil	No
9	Soleado	Frio	Normal	Debil	Si
10	Lluvia	Templado	Normal	Debil	Si
11	Soleado	Templado	Normal	Fuerte	Si
12	Nublado	Templado	Alta	Fuerte	Si
13	Nublado	Calor	Normal	Debil	Si
14	Lluvia	Templado	Alta	Fuerte	No

Datos de Entrenamiento:14 filas

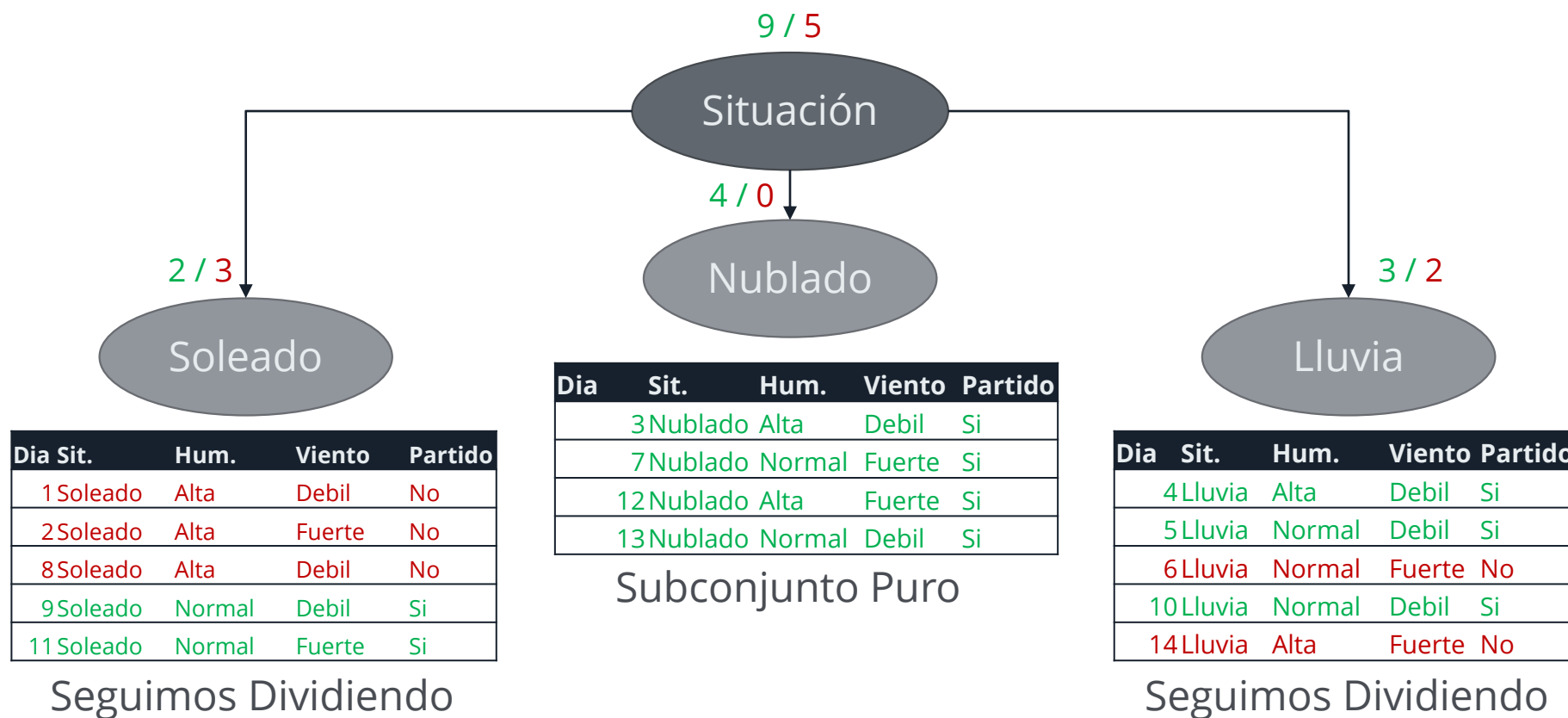
EJEMPLO: PARTIDO DE TENIS

Dia	Situación	Temperatura	Humedad	Viento	Partido
1	Soleado	Calor	Alta	Debil	No
2	Soleado	Calor	Alta	Fuerte	No
3	Nublado	Calor	Alta	Debil	Si
4	Lluvia	Templado	Alta	Debil	Si
5	Lluvia	Frio	Normal	Debil	Si
6	Lluvia	Frio	Normal	Fuerte	No
7	Nublado	Frio	Normal	Fuerte	Si
8	Soleado	Templado	Alta	Debil	No
9	Soleado	Frio	Normal	Debil	Si
10	Lluvia	Templado	Normal	Debil	Si
11	Soleado	Templado	Normal	Fuerte	Si
12	Nublado	Templado	Alta	Fuerte	Si
13	Nublado	Calor	Normal	Debil	Si
14	Lluvia	Templado	Alta	Fuerte	No

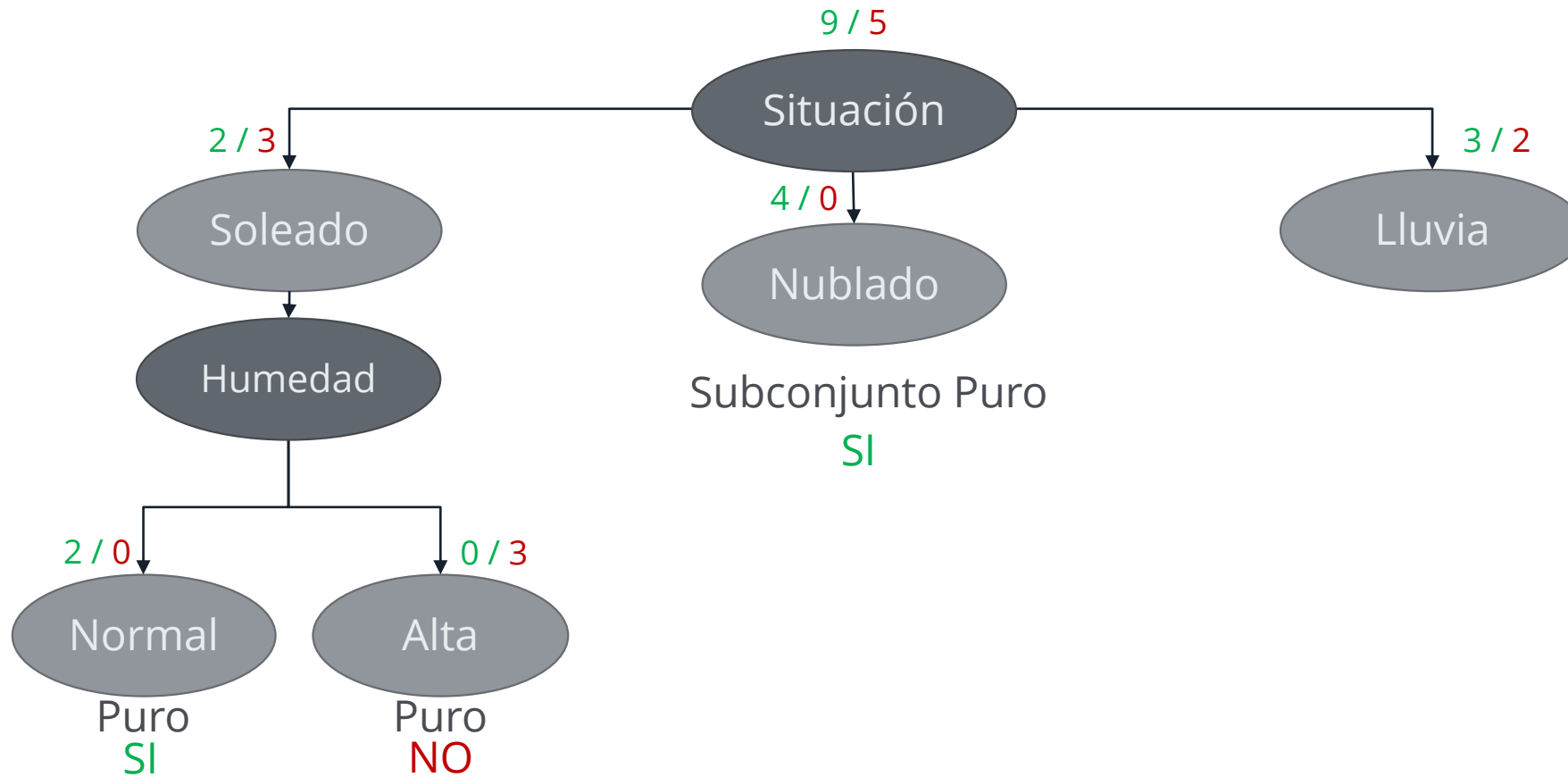
Datos de Entrenamiento:

9 SI
5 NO

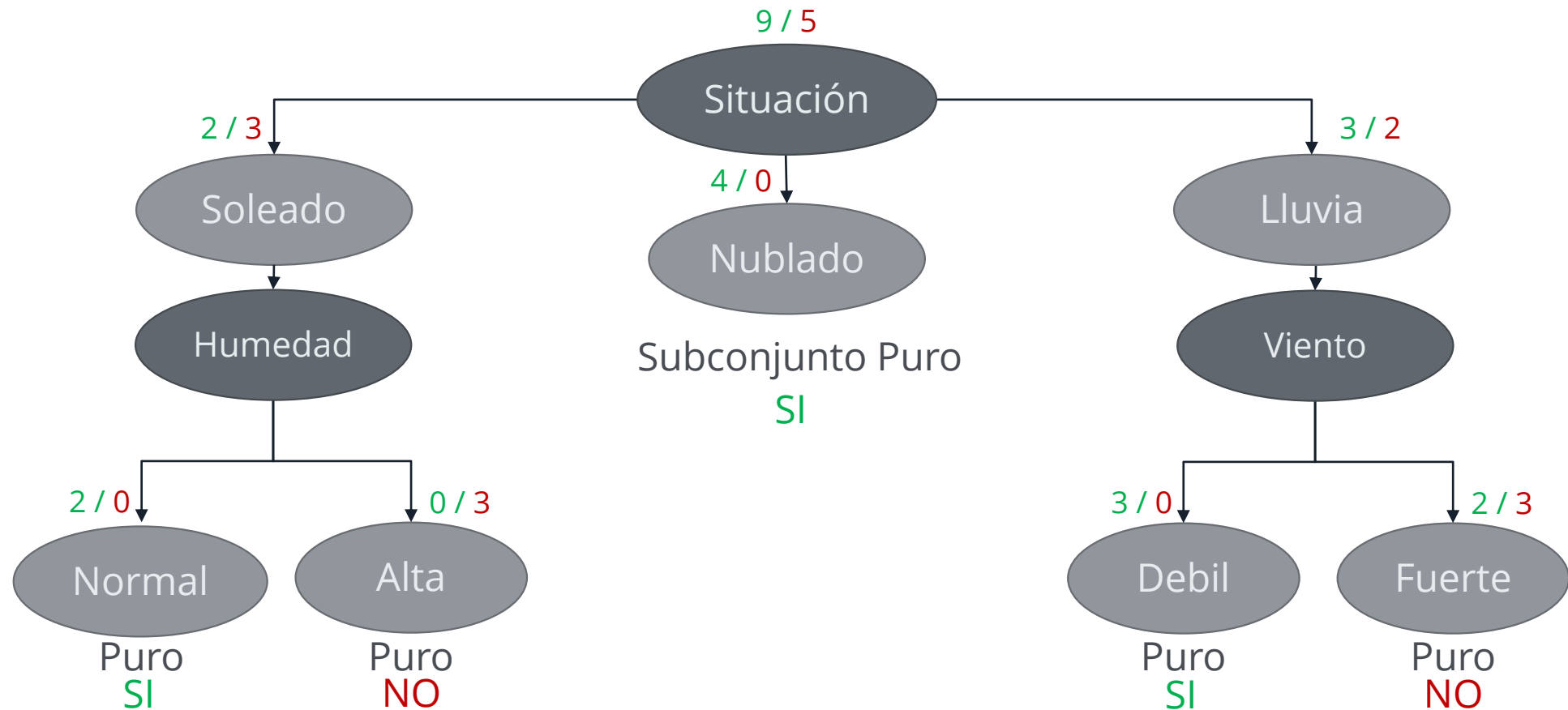
EJEMPLO: PARTIDO DE TENIS



EJEMPLO: PARTIDO DE TENIS

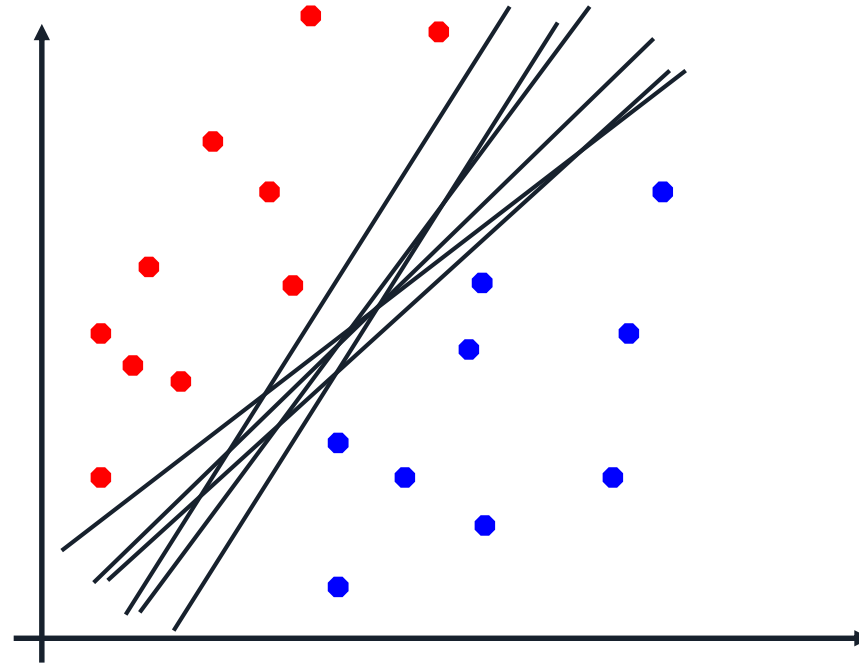


EJEMPLO: PARTIDO DE TENIS

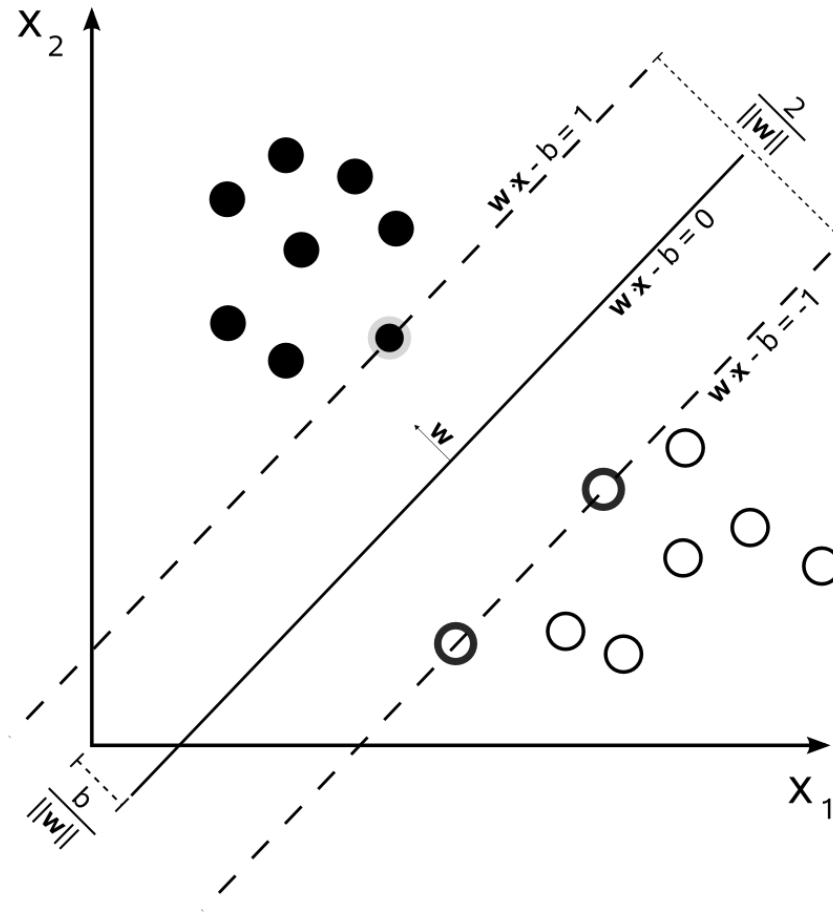


SUPPORT VECTOR MACHINES

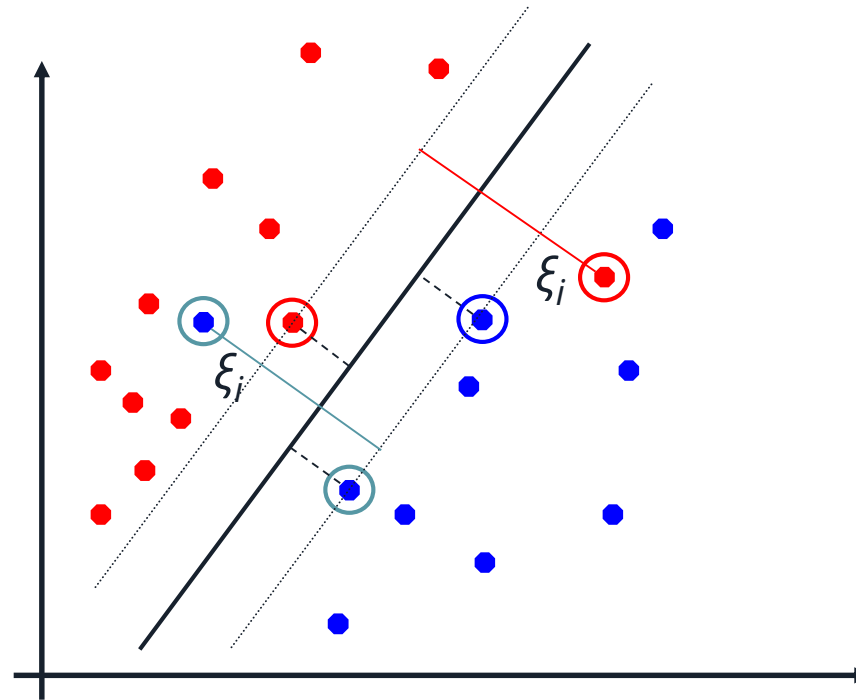
CLASIFICACION BINARIA



MARGEN DE CLASIFICACION



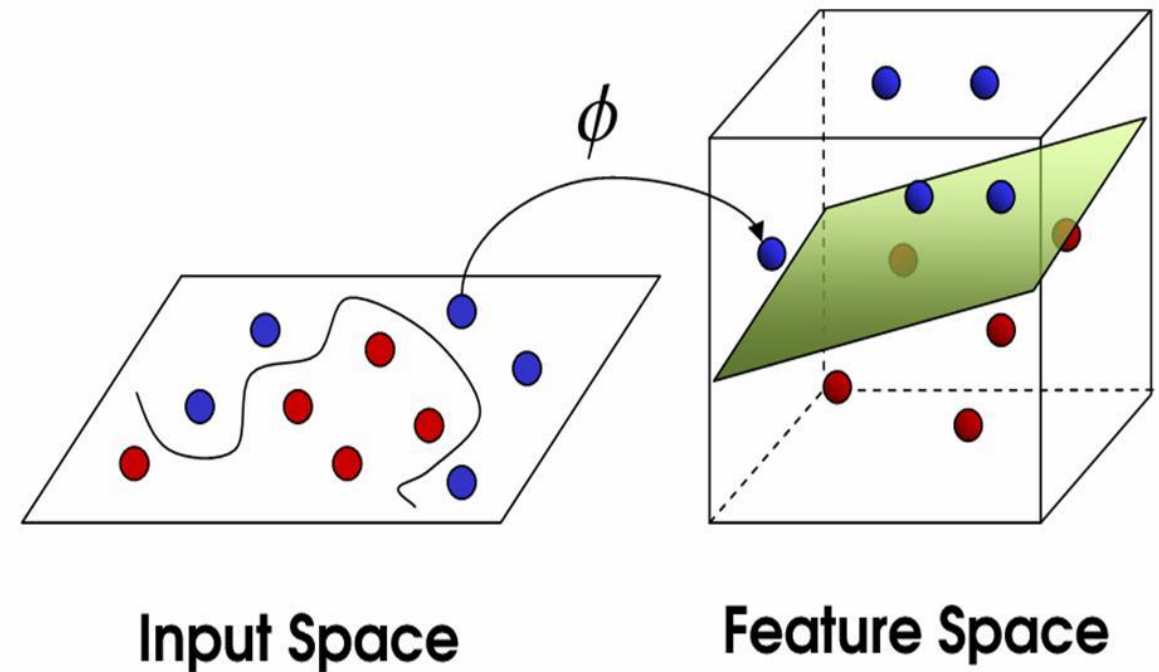
ERROR



SVM

- Es básicamente un problema de optimización
- La idea es buscar el plano que divida a las dos clases y al mismo tiempo maximizar la distancia a cada uno de los puntos más cercanos a cada lado del plano
- Si conseguimos el plano perfecto, se le llama “perceptron de estabilidad optima”
- Recordad que se está haciendo una simplificación (2 clases, un plano)
- En realidad nos referimos al plano como hiperplano

Principle of Support Vector Machines (SVM)



VENTAJAS DE SVM

- Maximizando el margen se reduce el overfitting
- Eficiente: $O(n^3 \cdot m)$
- Sirve para escenarios lineales y no lineales

AZURE ML

AZUREML

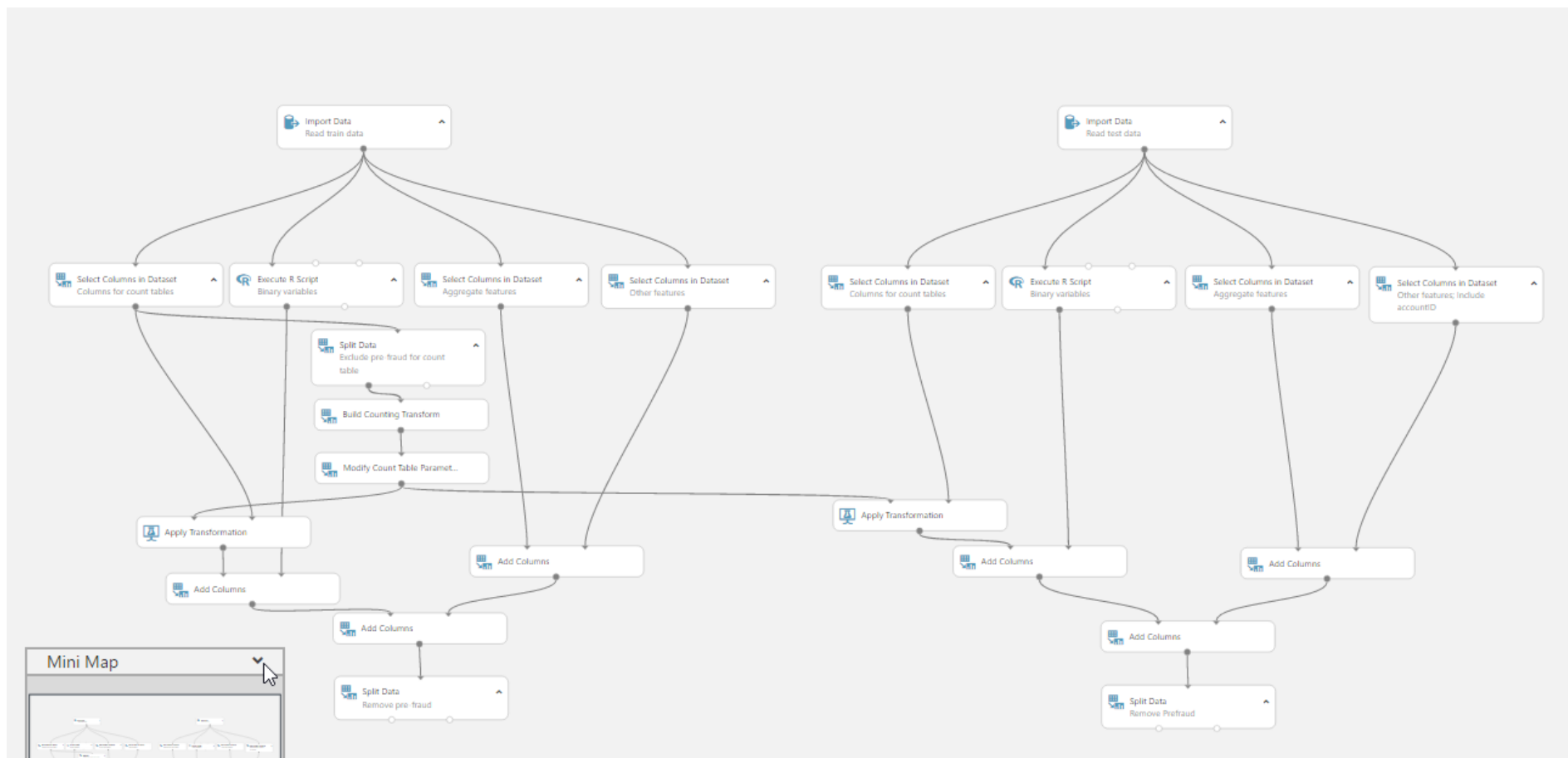


Azure
Machine Learning

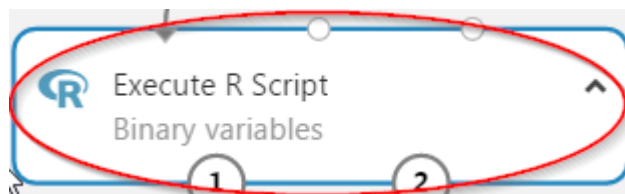
AZUREML

- Objetivo: Reducir la complejidad
- Accesible a través del navegador
- Trabajo colaborativo a través del Azure Workspace
- Workflow visual
- Gran cantidad de algoritmos excelentes de ML
- Extensible gracias al soporte para R

EASY TO USE



POSIBILIDAD DE USAR R



Execute R Script

R Script

```
4 #generate binary variables
5
6 is_highAmount = df$transactionAmountUSD > 150
bVars.df = as.data.frame(is_highAmount)
7
8
9 #addresss mismatch flags
10 bVars.df$acct_billing_address_mismatchFlag = as.character(df$pa
11 bVars.df$acct_billing_postalCode_mismatchFlag = as.character(d
12 bVars.df$acct_billing_country_mismatchFlag = as.character(df$pa
13 bVars.df$acct_billing_name_mismatchFlag = as.character(df$payme
14
15 bVars.df$acct_shipping_address_mismatchFlag = as.character(df$
16 bVars.df$acct_shipping_postalCode_mismatchFlag = as.character(
17 bVars.df$acct_shipping_country_mismatchFlag = as.character(df$
18
19 bVars.df$shipping_billing_address_mismatchFlag = as.character(
20 bVars.df$shipping_billing_postalCode_mismatchFlag = as.character
21 bVars.df$shipping_billing_country_mismatchFlag = as.character(
22
23 data.set = bVars.df
```

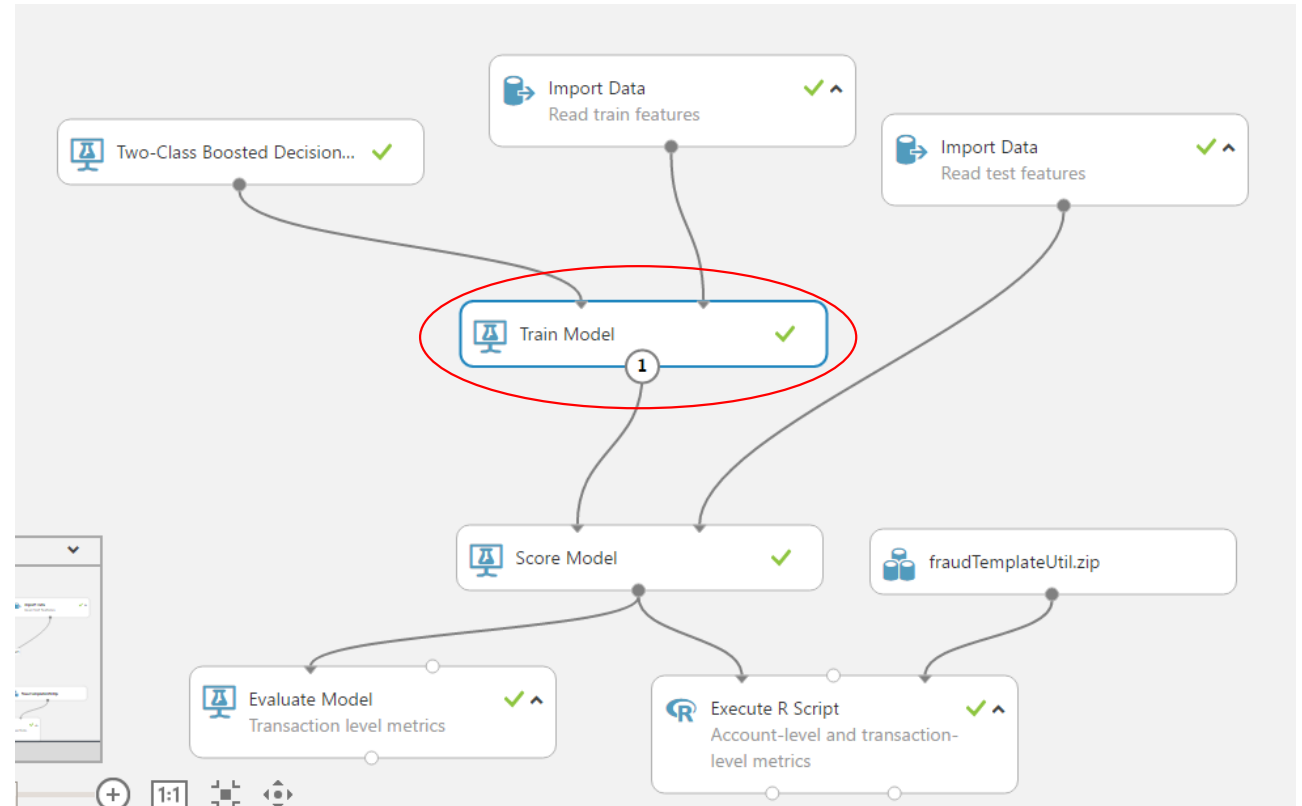
Random Seed

R Version

CRAN R 3.1.0

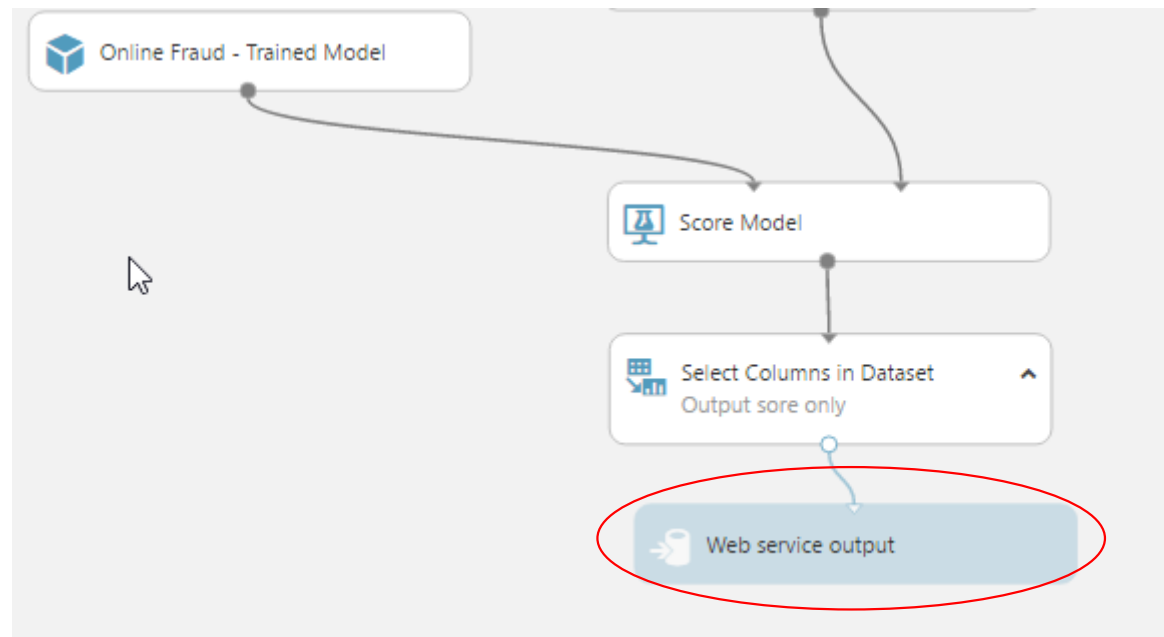
TRAINING A MODEL?

- Evaluate
- ▾ Initialize Model
 - Anomaly Detection
 - ▾ Classification
 - Multiclass Decision Forest
 - Multiclass Decision Jungle
 - Multiclass Logistic Regression
 - Multiclass Neural Network
 - One-vs-All Multiclass
 - Two-Class Averaged Perceptron
 - Two-Class Bayes Point Machine
 - Two-Class Boosted Decision Tree
 - Two-Class Decision Forest
 - Two-Class Decision Jungle
 - Two-Class Locally-Deep Support V...
 - Two-Class Logistic Regression
 - Two-Class Neural Network
 - Two-Class Support Vector Machine
 - Clustering
 - Regression
- Score
- Train



Y COMO USO ESTO EN MI PROYECTO?

- Via REST api, se pueden consumir los servicios de Azure ML



PLANES

Quick Evaluation	Most Popular	Enterprise Grade
Guest Workspace	Free Workspace	Standard Workspace
8-hour trial	\$0/month	\$9.99/month
No sign-in required.	Don't already have a Microsoft account? Simply sign up here .	Azure subscription required Other charges may apply. Read more .
Enter	Sign In	Create Workspace
<ul style="list-style-type: none">▪ No hassle instant access▪ Stock sample datasets▪ ML models built in minutes▪ Full range of ML algorithms	<ul style="list-style-type: none">▪ Free access that never expires▪ 10 GB storage on us▪ R and Python scripts support▪ Predictive web services	<ul style="list-style-type: none">▪ Full SLA Support▪ Bring your own Azure storage▪ Parallel graph execution▪ Elastic Web Service endpoints

plain concepts

AZURE ML



SPARK ML

SPARK MACHINE LEARNING



WHAT IS SPARKML?

- Es una librería de Machine Learning que viene con Spark
- Viene incluida en la instalación de Spark
- Se puede programar con Scala, Java y Python
- Viene con una serie de algoritmos comunes incluidos:
 - Clasificación, regresión, clustering
 - Filtrado colaborativo, reducción de dimensionalidad

WORKFLOW USUAL

- Paso 1: Carga los datos
- Paso 2: Prepara los datos
- Paso 3: Entrénalos
- Paso 4: Predice

PASO 1: CARGA DE DATOS

- Usando SparkSQL podemos cargar nuestros datos de manera sencilla:

```
df= sqlContext.read  
.format("com.databricks.spark.csv")  
.option("header","true")  
.option("inferSchema","true")  
.load("yourFolder/foo.csv")
```

PASO 2: PREPARACIÓN DE LOS DATOS

- En este paso se pueden realizar acciones intermedias para preparar los datos como pueden ser:
 - Conversiones de fechas
 - Convertir cadenas a valores numéricos
 - Concatenar campos
 - Extraer features
 - Etc

PASO 3: ENTRENAR EL MODELO

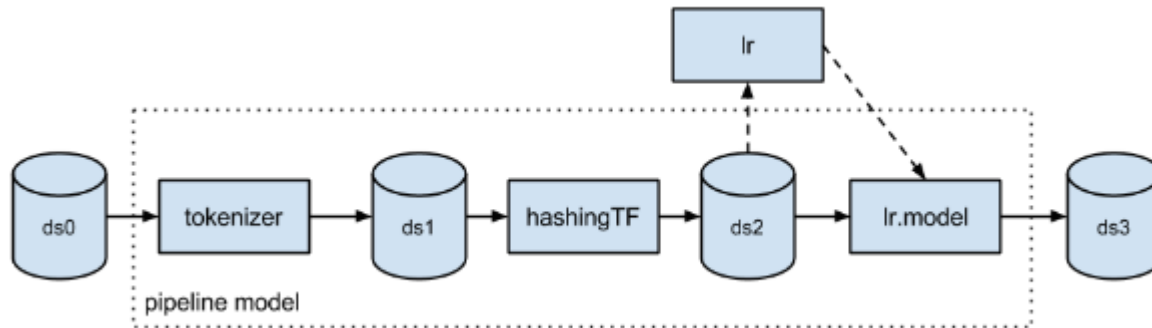
- La complejidad de este paso fluctuarán en función de lo que se quiera hacer, pero SparkML trae un conjunto de técnicas de ML comunes para trabajar con ellas
- Por ejemplo, para un Decision Tree simplemente hay que llamar a la función *DecisionTreeClassifier* con los parámetros pertinentes.

PASO 4: PREDECIR

- Predecir también bastante sencillo (siempre dependiendo de lo que se quiera hacer)
- Un ejemplo: `model.transform(df).select("prediction", "category-index").take(15)`

EL CONCEPTO DE PIPELINE

- Los pasos descritos anteriormente han sido implementados en SparkML a través de la abstracción conocida como *Pipeline*



- El pipeline empieza en el paso 1 y acaba en el 3, dejando listo el sistema para hacer predicciones

plain concepts

SPARKML



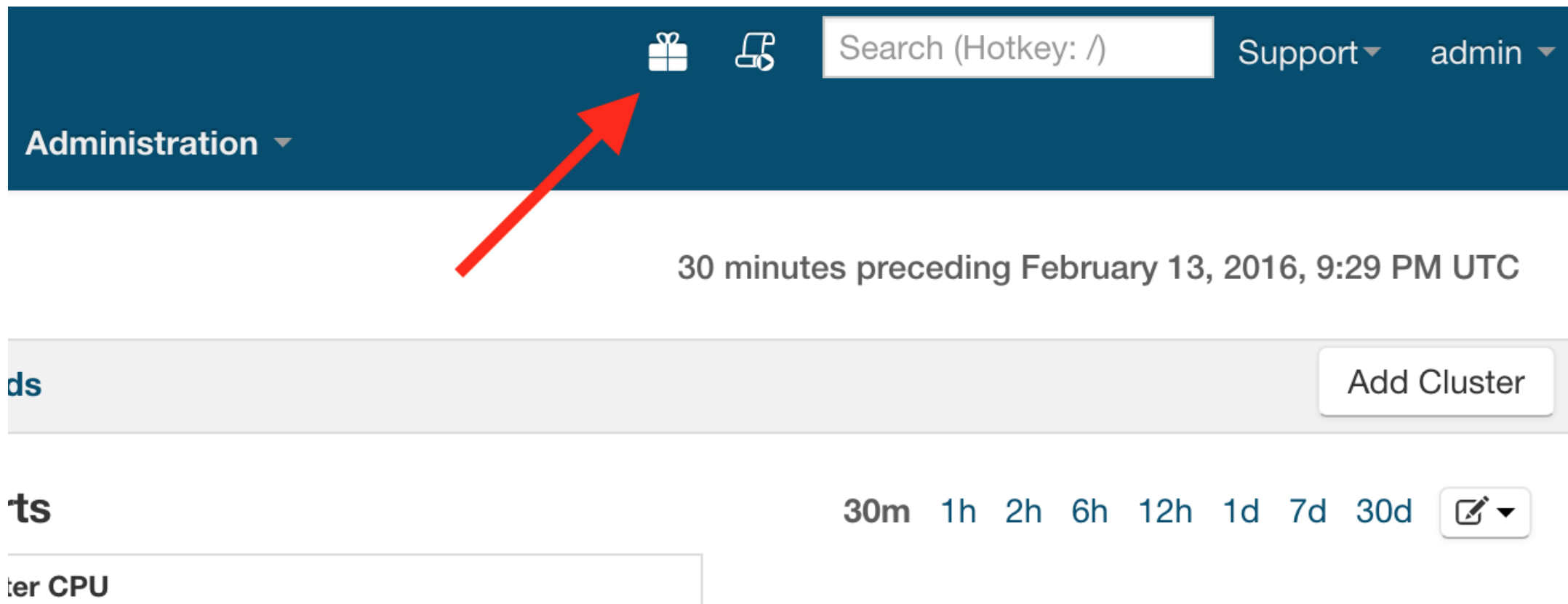
¿PREGUNTAS?

plain concepts

EJERCICIO PRÁCTICO



INSTALAR ANACONDA



The screenshot shows the Databricks Administration interface. At the top, there is a dark blue navigation bar containing the text "Administration" with a dropdown arrow, a gift icon, a laptop icon, a search bar with the placeholder text "Search (Hotkey: /)", and links for "Support" and "admin" with dropdown arrows. A red arrow points from the bottom left towards the gift icon. Below the navigation bar, the main content area displays the text "30 minutes preceding February 13, 2016, 9:29 PM UTC". On the left side of this area, the text "ds" is visible. On the right side, there is a button labeled "Add Cluster". Below this, the text "ts" is visible. To the right of "ts", there is a row of time range filters: "30m", "1h", "2h", "6h", "12h", "1d", "7d", and "30d". To the right of these filters is a small icon of a notepad with a pencil and a dropdown arrow. At the bottom left, the text "ter CPU" is visible.

INSTALAR ANACONDA

Save Changes

1 Edited Value

Show All Descriptions

Local Parcel Repository Path

?

Parcel Update Frequency

hour(s) ▾

?

Remote Parcel Repository URLs

+

-

C

+

-

?

INSTALAR ANACONDA

Cluster 1			
Parcel Name	Version	Status	Actions
Anaconda	2.5.0	Available Remotely	<button>Download</button>

Cluster 1			
Parcel Name	Version	Status	Actions
Anaconda	2.5.0	Downloaded	<button>Distribute</button> ▼

Cluster 1			
Parcel Name	Version	Status	Actions
Anaconda	2.5.0	Distributed	<button>Activate</button> ▼



GRACIAS

Barcelona



Bilbao



Madrid



Sevilla



Dubai



London



Seattle