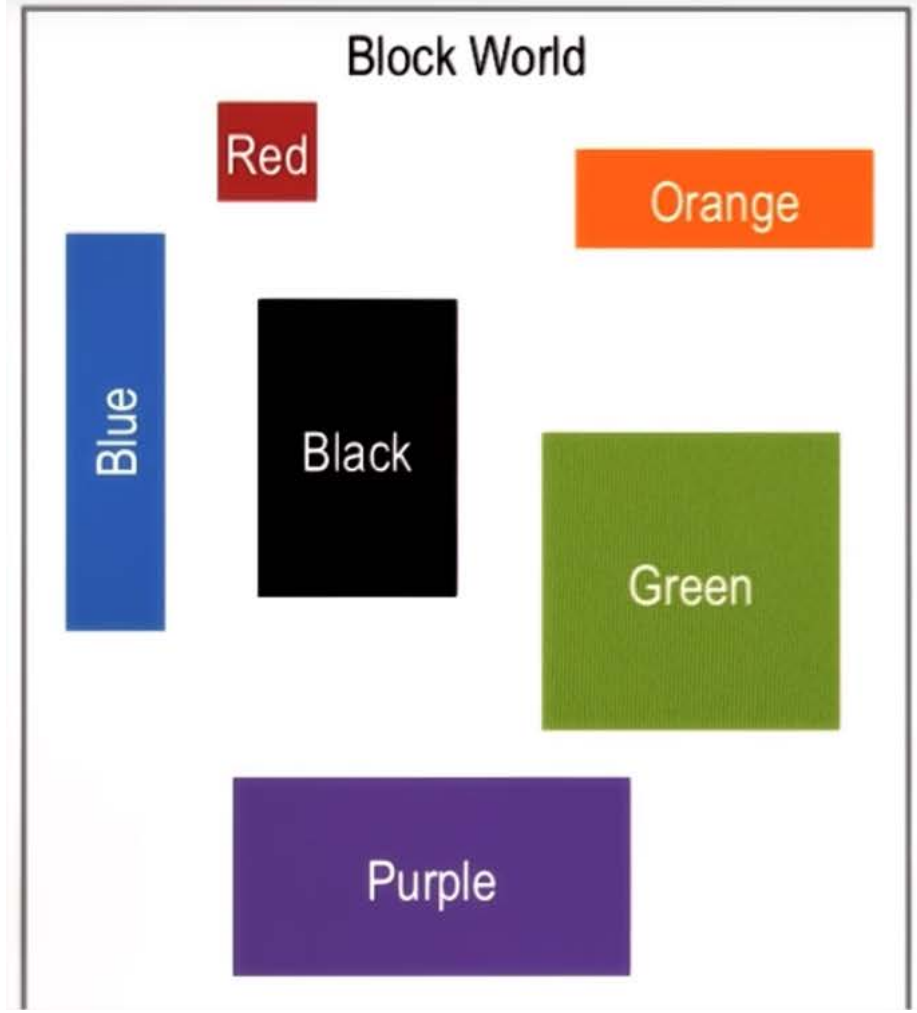
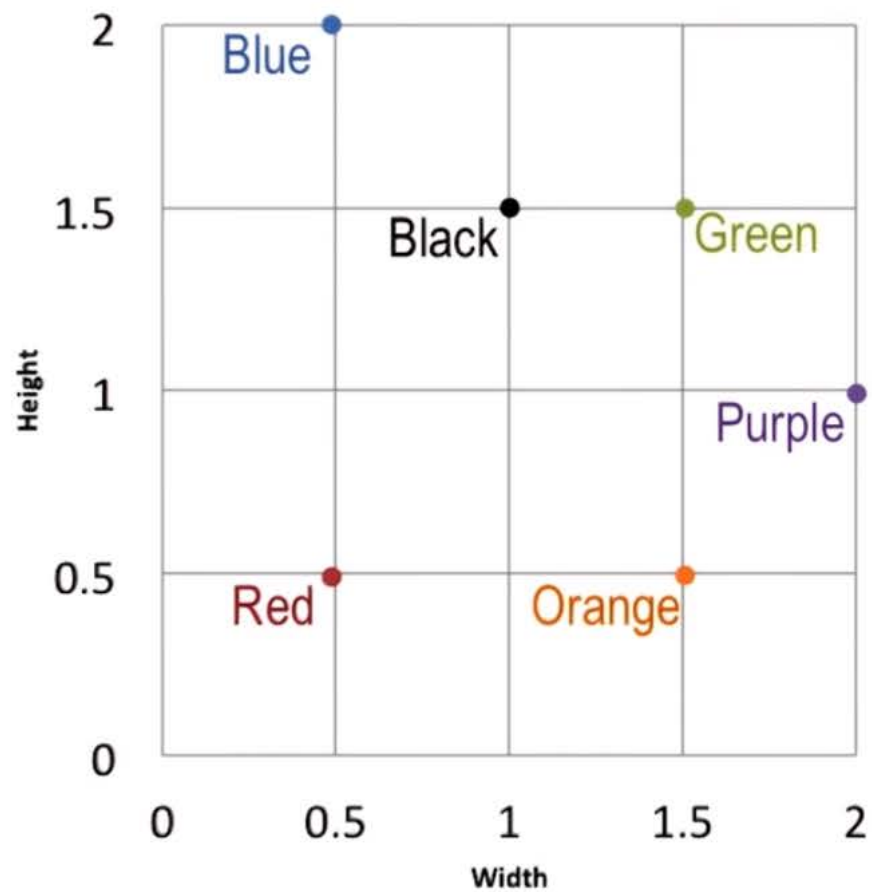
Razonamiento basado en casos

- ¿Cómo conseguir representar mediante hechos y reglas todo lo que un experto –o una comunidad de expertos– sabe?
  - **Cuello de botella de la adquisición de conocimiento**
    - ¿Hay algún experto dispuesto a dedicar el tiempo necesario?
    - ¿El experto y el ingeniero del conocimiento hablan el mismo “idioma”?
    - ¿El ingeniero del conocimiento es capaz de captar todas las sutilezas del dominio para poder así representarlas?
    - ¿Es posible **formalizar** el conocimiento obtenido?
- Adquisición y reutilización de casos de resolución de problemas: Razonamiento por analogía: sistemas basados en casos (CBR, *Case Based Reasoning*)

Razonamiento basado en casos

- Sabiduría heurística en forma de **ejemplos**
- Los seres humanos resolvemos problemas en base a nuestras experiencias pasadas y no a partir de un conocimiento detallado
  - Médicos, ingenieros, programadores expertos, arquitectos, abogados, jugadores de ajedrez, cocineros, …
- El CBR facilita la adquisición de conocimiento
  - A los expertos les resulta más sencillo “contar batallitas” que proporcionar reglas de aplicación general.
  - Facilita el aprendizaje - ***Lazy Learning***
- Un sistema CBR ***sólo*** necesita un conjunto de problemas resueltos (Base de Casos) + conocimiento de similitud + conocimiento de adaptación
  - Los problemas tienden a ser recurrentes
  - Podemos aprender de los errores

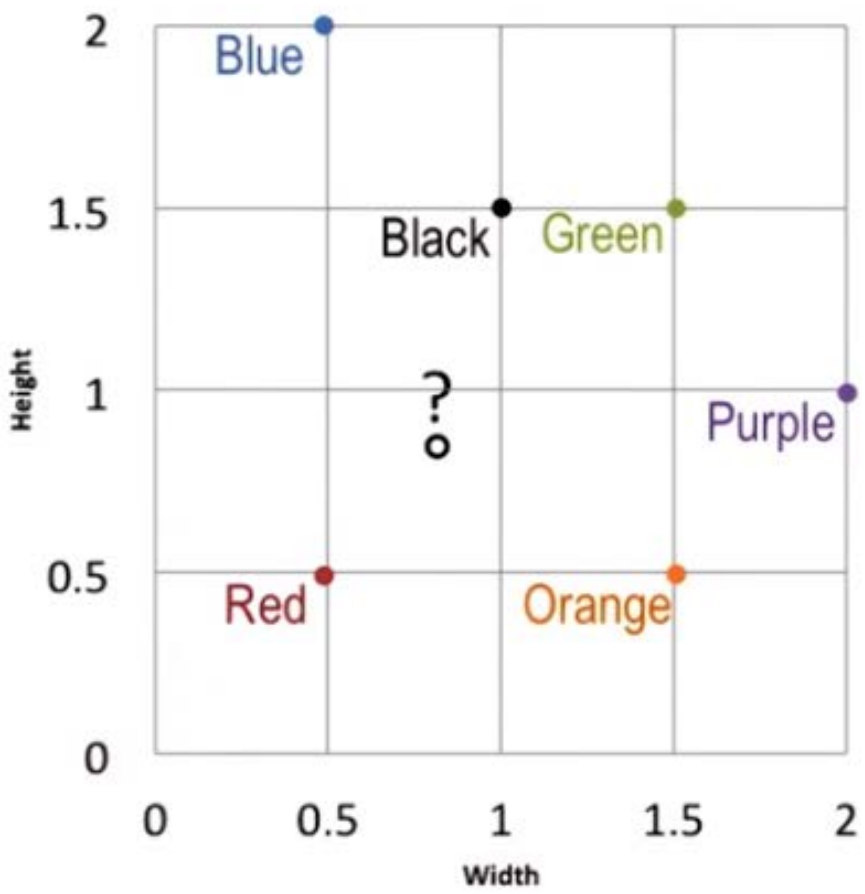

Block World
Red
Orange
Blue
Black
Green
Purple


Blue
Black
Green
Purple
Red
Orange
Height
Width
0
0.5
1
1.5
2

What color is this block?

?

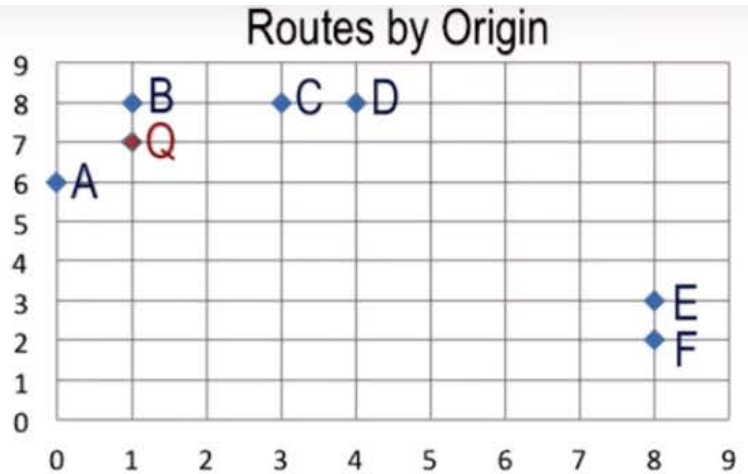
Block	x_c	y_c	x_n	y_n	d
Blue	0.5	2.0	0.8	0.8	1.24
Red	**0.5**	**0.5**	**0.8**	**0.8**	**0.42**
Black	1.0	1.5	0.8	0.8	0.72
Green	1.5	1.5	0.8	0.8	0.98
Orange	1.5	0.5	0.8	0.8	0.76
Purple	2.0	1.0	0.8	0.8	1.22



What
simila

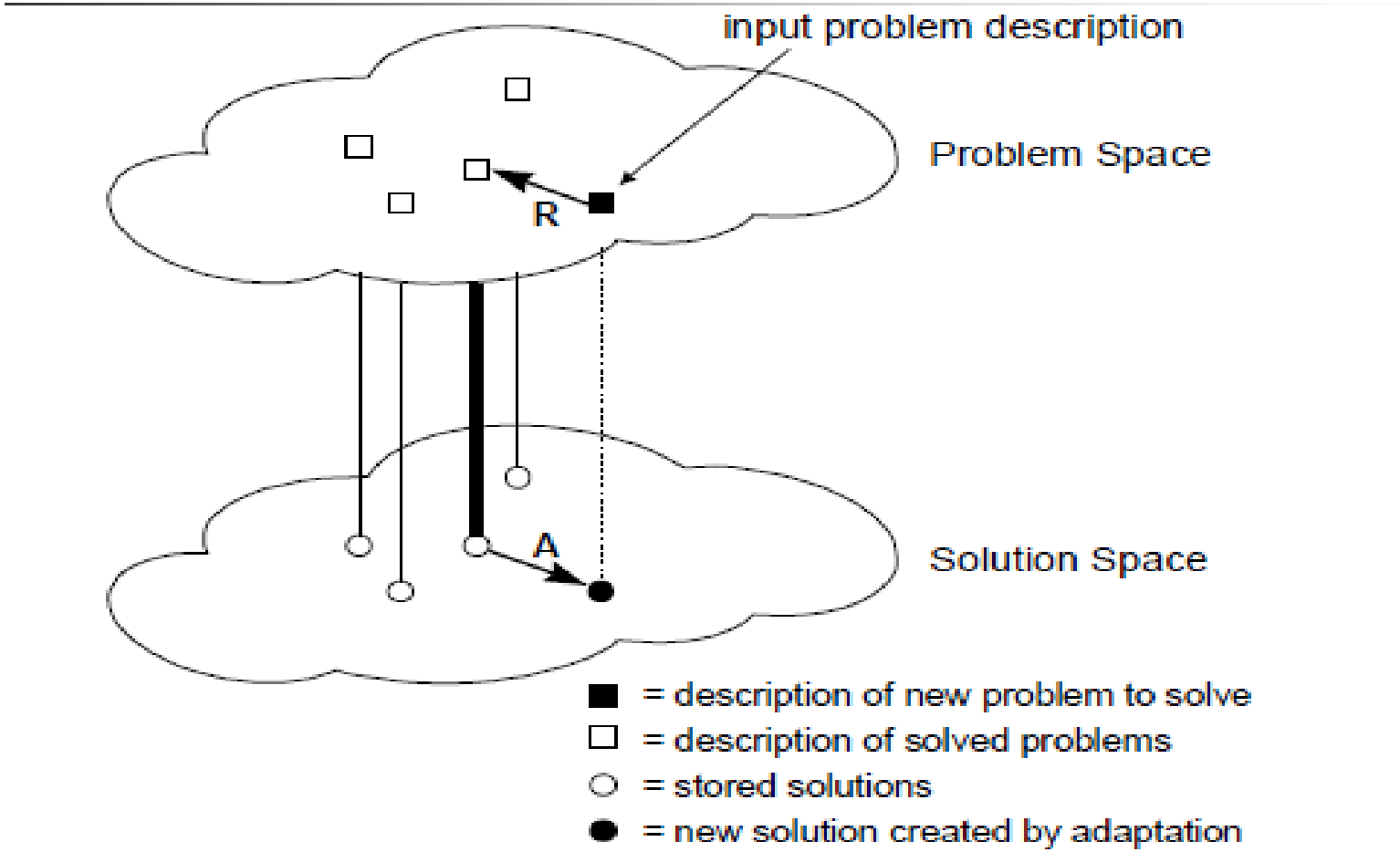
K-Nearest Neighbour method

Route	c_1	c_2	c_3	c_4	d_k
A	0	6	7	9	10.10
B	1	8	9	8	10.68
C	3	8	8	2	7.42
D	**4**	**8**	**4**	**1**	**4.36**
E	8	3	2	1	8.12
F	8	2	9	0	11.80
Q	1	7	1	1	-

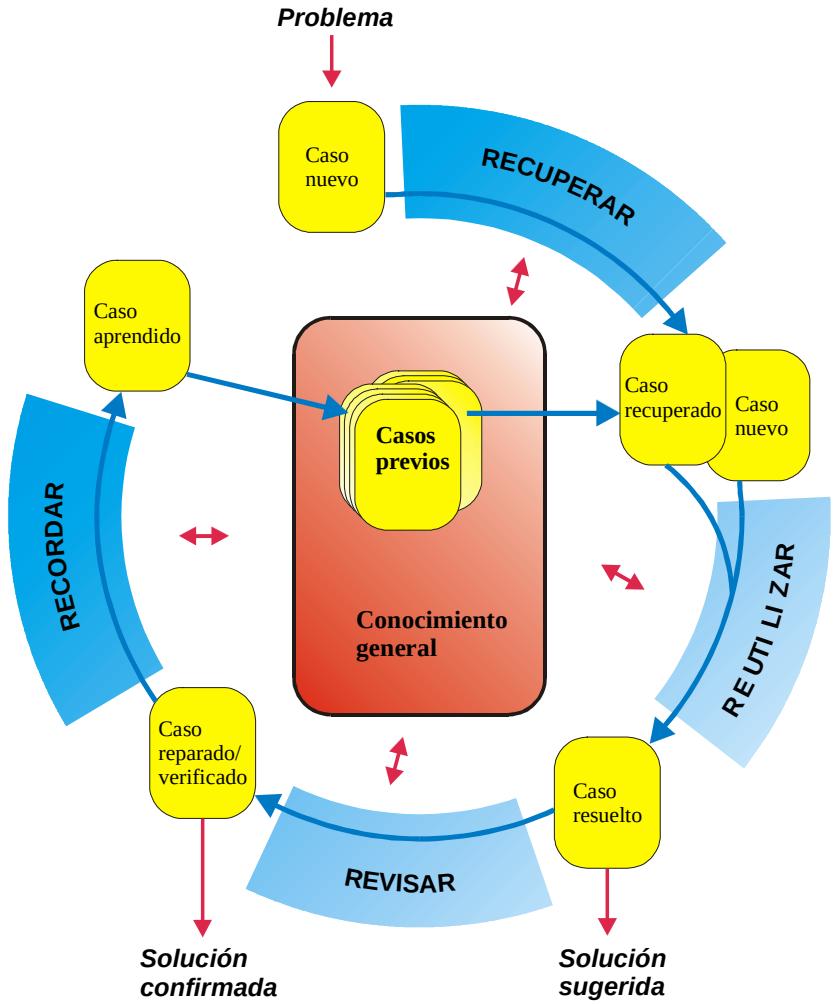


Hipótesis que sustentan el CBR

- Los problemas tienden a recurrir
- Problemas parecidos tienen soluciones parecidas



Ciclo de Razonamiento Basado en Casos (CBR)



📖 **Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches.** *Artificial Intelligence Communications*, 7 (1), Aamodt A., & Plaza E., 1994

Temas de investigación:

- Representación de casos: plana, estructurada, OO, frames, textos
- Indexación de casos
- Similitud
- Medidas de accuracy del proceso completo (testing leave one out)
- CBR conversacional vs one-shot
- Adaptación de soluciones
- Aprendizaje: mantenimiento de bases de casos
  Eficiencia/cobertura
- Sistemas híbridos
- Aplicaciones
- Explicaciones basadas en casos

CASE 1	
Problem (Symptoms)	• Problem: Front light doesn't work • Car: VW Golf II, 1.6 L • Year: 1993 • Battery voltage: 13,6 V • State of lights: OK • State of light switch: OK
Solution	• Diagnosis: Front light fuse defect • Repair: Replace front light fuse

Each case describes one situation

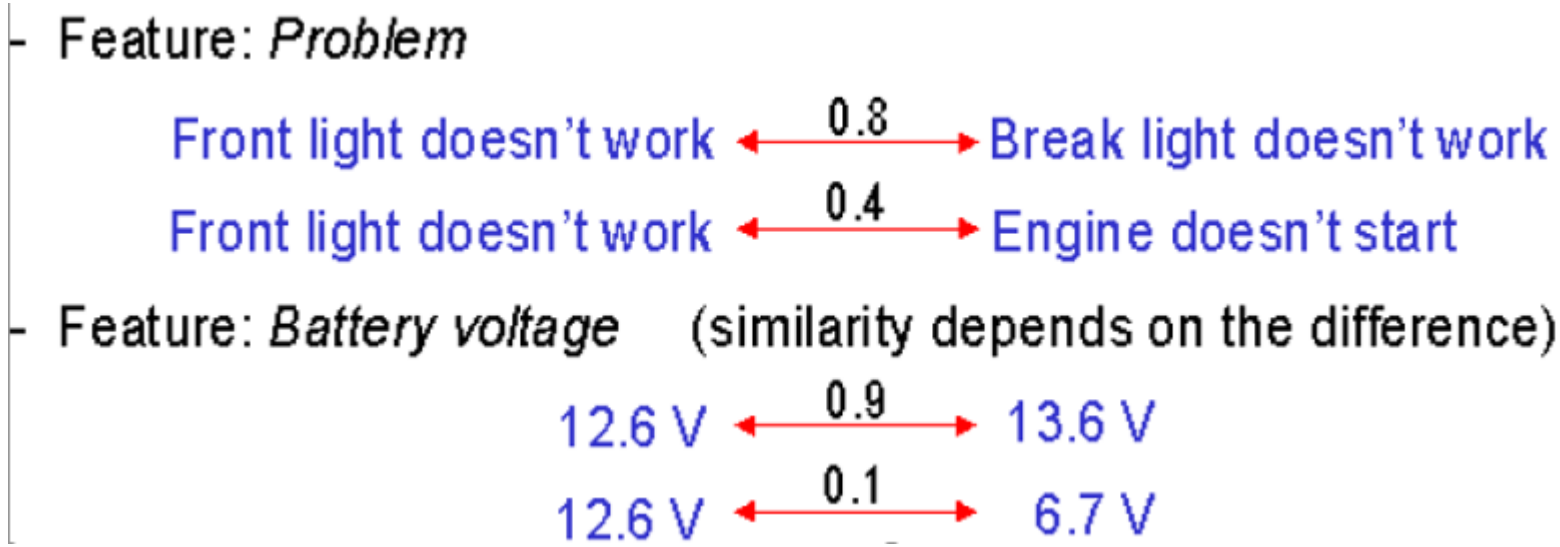
Cases are independent of each other

Case are not rules

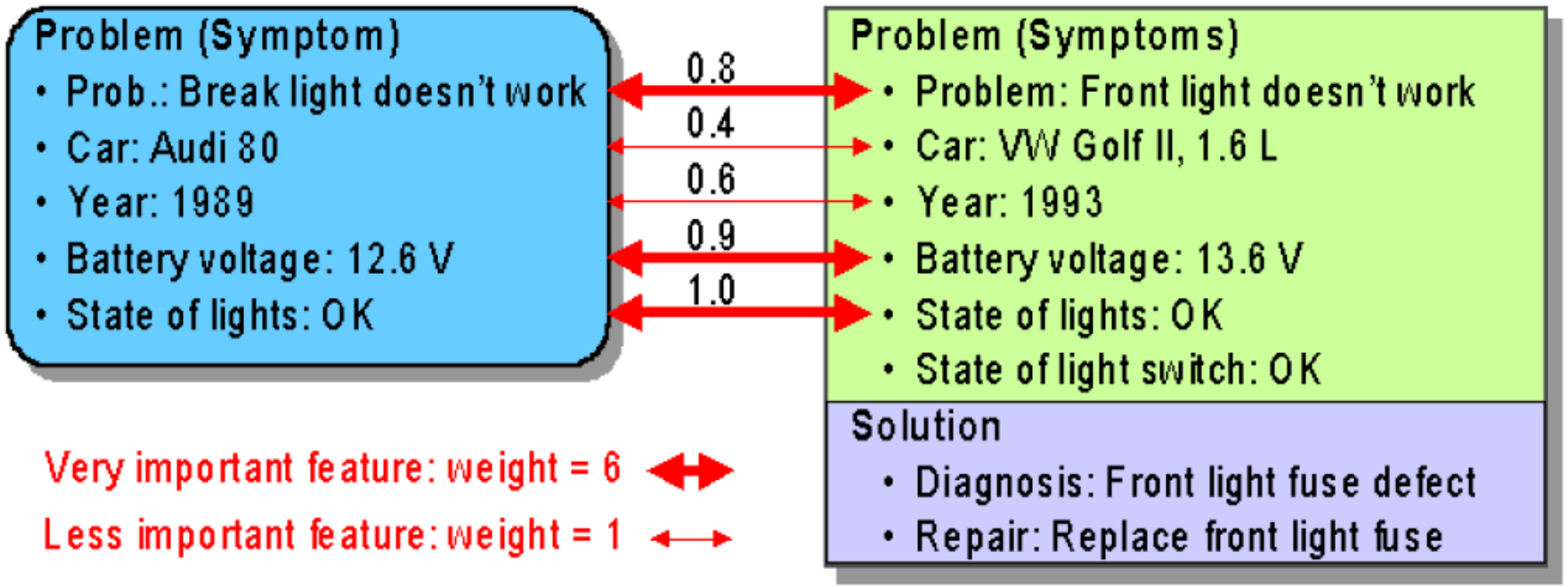
CASE 2	
Problem (Symptoms)	• Problem: Front light doesn't work • Car: Audi A6 • Year: 1995 • Battery voltage : 12,9 V • State of lights: surface damaged • State of light switch: OK
Solution	• Diagnosis: Bulb defect • Repair: Replace front light

Similitud Problema Consulta - Casos

- Se computa la similitud en cada atributo
  - Símbolos, textos, valores, multivalores, ..
  - Cada atributo puede tener una importancia (peso) diferente en la similitud global



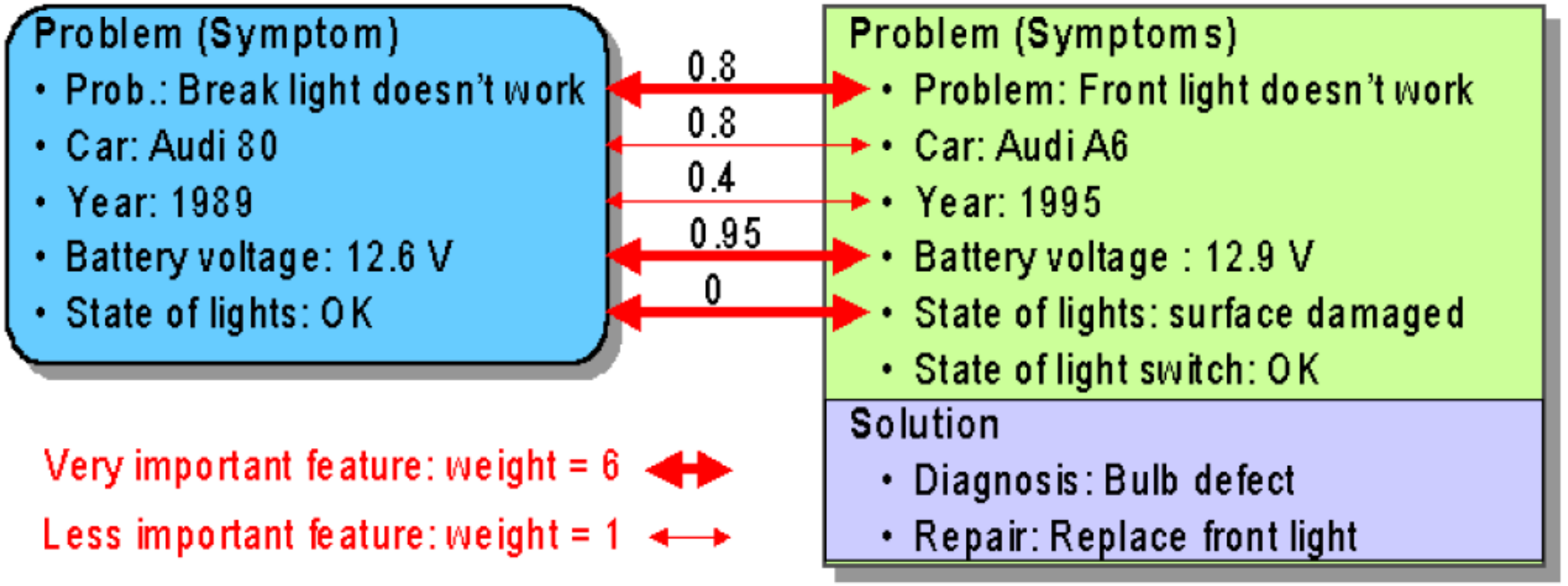
Similitud con caso 1



Similarity Computation by Weighted Average

similarity(new,case 1) = 1/20 * [6*0.8 + 1*0.4 + 1*0.6 + 6*0.9 + 6* 1.0] = **0.86**

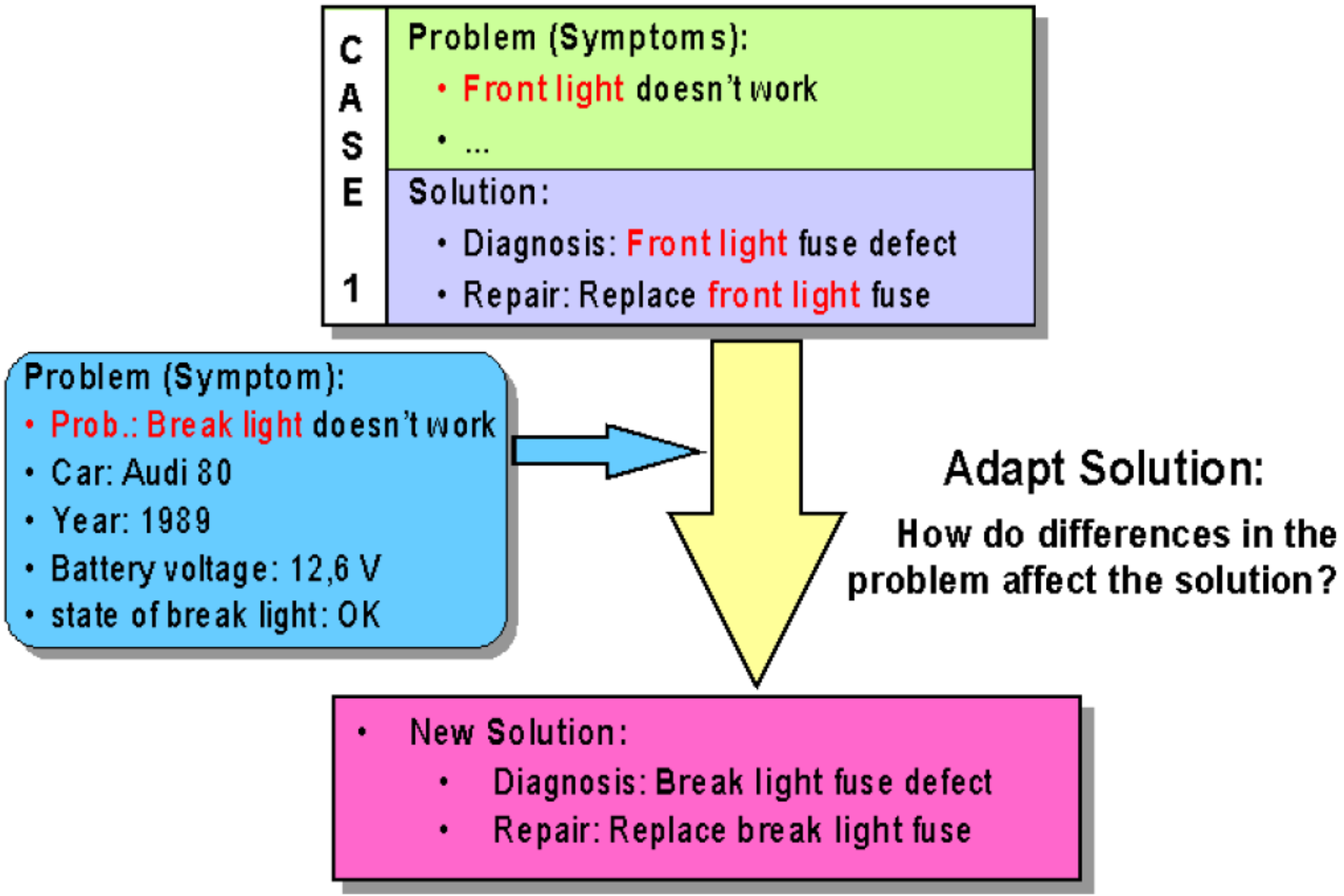
Similitud con caso 2



Similarity Computation by Weighted Average

similarity(new,case 2) = 1/20 * [6*0.8 + 1*0.8 + 1*0.4 + 6*0.95 + 6*0] = **0.585**

Reuse the Solution of Case 1



Store the New Experience

If diagnosis is correct:

store new case in the memory.

CASE 3	
	Problem (Symptoms): • Problem: Break light doesn't work • Car: Audi 80 • Year: 1989 • Battery voltage: 12.6 V • State of break lights: OK • light switch clicking: OK
	Solution: • Diagnosis: break light fuse defect • Repair: replace break light fuse

Cálculo de similitudes

- El modelo KNN se basa en calcular similitud entre dos elementos
- Depende de la representación hay varias opciones
  - Objetos representados como una estructura
    - Igualdad en atributos o funciones locales según el tipo del atributo
    - Similitud semántica (usando recursos externos, ontologías)
    - Distintas formas de combinar las similitudes locales
  - Representación planas tipo lista
    - Cada objeto es una lista plana de descriptores
    - Modelo del espacio vectorial (lo veréis en IA2 relacionado con PLN)
    - Similitud TF/IDF Similitud del coseno, distancia euclidea o correlacion de Pearson.

Similitud entre estructuras

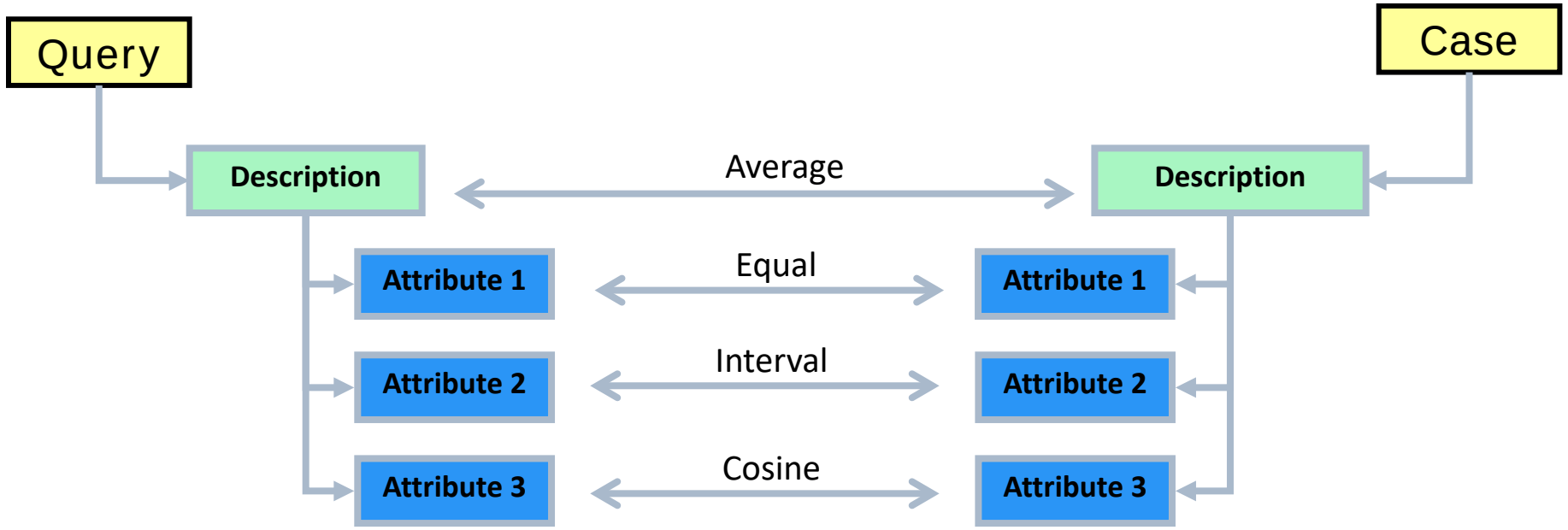
- Propiedades de las funciones de similitud
  - Reflexiva y ¿simétrica? pero NO tiene que ser transitiva

    BMW blanco ↔ Renault blanco ↔ Renault rojo
- Aproximación simplista (igualdad)

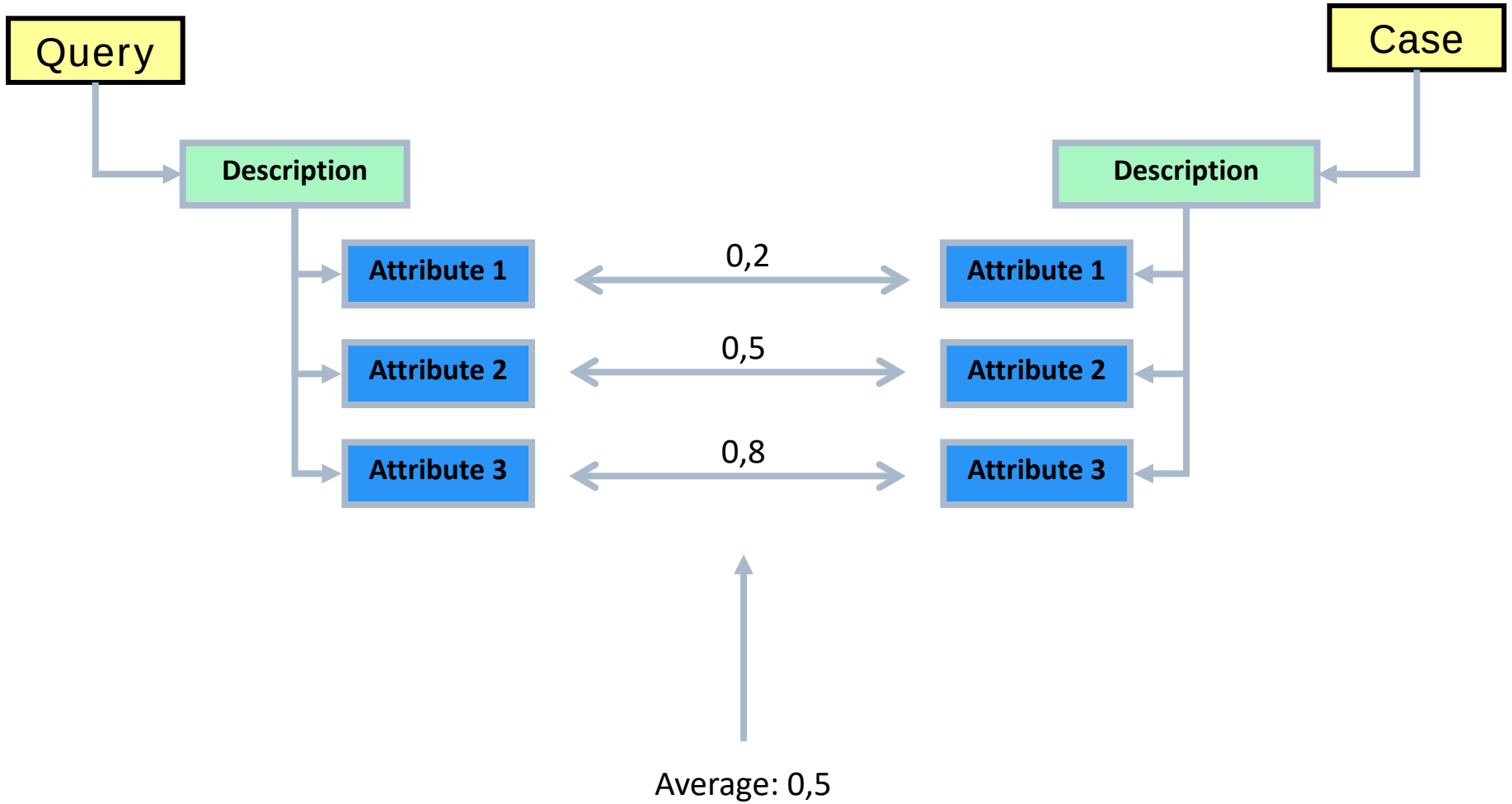
```
leer(C1, C2);
todosAtributos := atributos(C1) U atributos(C2);
tamaño := card(todosAtributos);
valoresIguales := { a | C1.a = C2.a };
iguales := card(valoresIguales);
devolver(iguales/tamaño);
```

- Se puede refinar el cálculo de
  - La similitud entre atributos (similitud local)
  - La similitud entre casos (similitud global)

Similitud entre estructuras



Similitud entre estructuras



Ejercicios

- Plantea el diseño de un SBC para cada uno de los siguientes dominios y problemas utilizando dos aproximaciones:
  - Basada en reglas
    - Estructura de los hechos
    - Plantillas y tipos de reglas
    - Algún ejemplo de entrada y salida
  - Basada en casos
    - Estructura de los casos
    - Función de similitud
    - Adaptación de la solución
    - Algún ejemplo de entrada y salida.
- Explica las ventajas e inconvenientes de cada una de las dos aproximaciones y cuál elegirías en cada caso.

1. SBC de tasación de vehículos de 2º mano.
2. Jugador de parchís

2. SBC de diagnóstico (puede ser de averías de televisiones, ordenadores, o incluso diagnostico médico).
3. SBC entrenador personal
4. Agente de citas online
5. Filtro SPAM
6. Recomendador de películas

Un ejemplo real de CBR

https://gaia.fdi.ucm.es/#research

Intelligent control system for back pain therapy

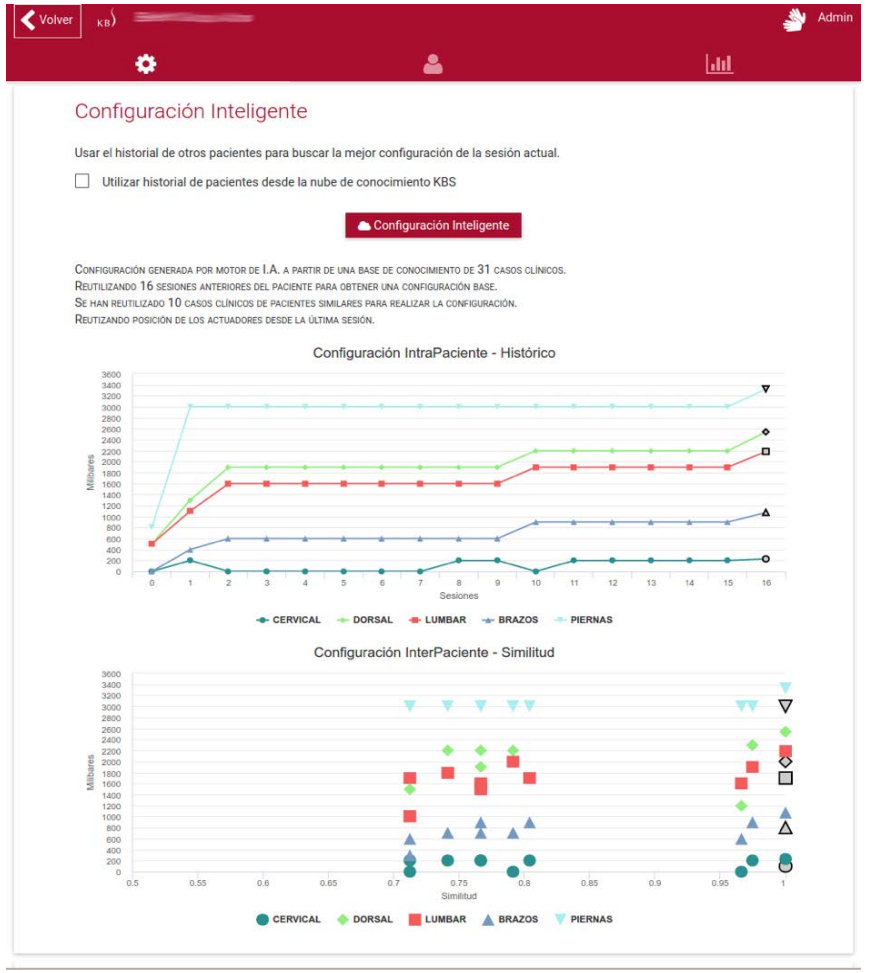
Juan Antonio Recio-García , Belén Díaz-Agudo, José Luis Jorro
Group for Artificial Intelligence Applications
Department of Software Engineering and Artificial Intelligence
Universidad Complutense de Madrid

Alireza Kazemi
Institute of Physiotherapy and Sports of Guadalajara

MINISTERIO DE ECONOMÍA Y COMPETITIVIDAD
CDTI Centro para el Desarrollo Tecnológico Industrial
UNIÓN EUROPEA
Fondo Europeo de Desarrollo Regional (FEDER)
Una manera de hacer Europa

Solución mediante CBR

- No es posible especificar el dominio ni las reglas que rigen la configuración de la máquina.
  - El propio experto no es capaz de definirlo, pero sí contar muchas experiencias con pacientes.
- Aproximación obvia (y única):
  - Recopilar registros de pacientes
  - Recuperar registros similares
  - Generar / Adaptar solución
  - Revisión por el experto
  - Almacenar nueva solución



Sistemas Recomendadores

- ¿Qué son?
- ¿Para qué sirven?
- ¿Qué recomendamos?

- CBR se relaciona con recomendadores
  - Similitud (productos/usuarios)

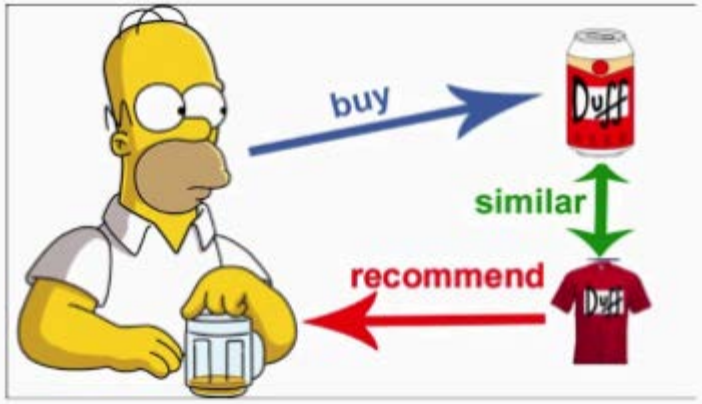


Image ref : https://www.linkedin.com/pulse/collaborative-filtering-apache-spark-juancarlos-olamendy-turruellas/

¿Qué son?

- Los *sistemas recomendadores r*ecomiendan elementos (items) que resulten de **interés** para los usuarios.
  - Interés = preferencias expresadas implícita o explícitamente.
- Ejemplos: comercio electrónico, búsqueda de información
  - "Búsqueda de información interesante"
  - La tecnología de los recomendadores es el nuevo paradigma de búsqueda – ¿aproximada? ¿sin query? → distintas formas de interacción con el usario / con y sin registro / one shot o varios ciclos de interacción

Siste

Los recomendadores ayudan a resolver problemas de sobrecarga de información exponiendo a los usuarios sólo aquellos productos más interesantes, novedosos, relevantes,..

Netflix / Amazon

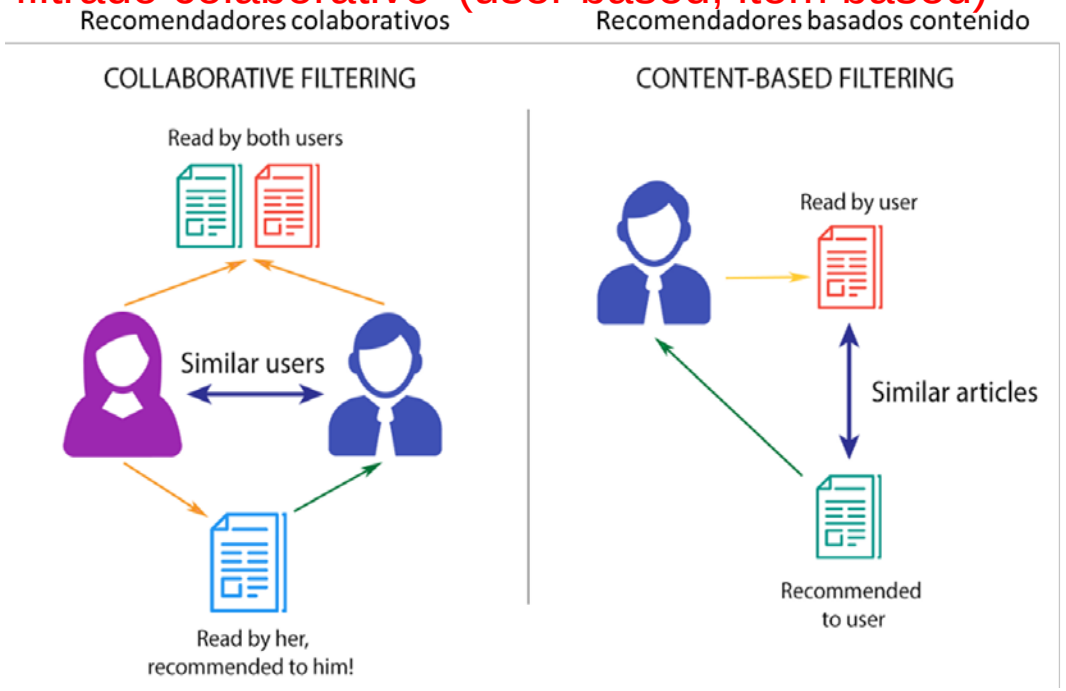
Usa criterios de popularidad + usuarios en **perfiles para** limitarle las opciones
Para generar estos perfiles, el algoritmo no sólo tiene en cuenta nuestros gustos, sino también dónde estamos, en qué momento estamos activos, a través de qué dispositivo nos conectamos e incluso cómo nos comportamos durante la visualización.
Detener o abandonar una serie o película en un momento concreto también da una información.

amazon.com

Para mejorar mis recomendaciones.
Valora un producto comprado: elige una valoración de 1 a 5 estrellas para el producto que has comprado o elige dejar el producto sin valorar. Las valoraciones que envías son privadas. Nunca se comparten con otros clientes de Amazon y no afectan a la opinión media del producto. Estas valoraciones se utilizan únicamente en nuestro sistema de recomendaciones para ofrecerte las recomendaciones más precisas posibles.

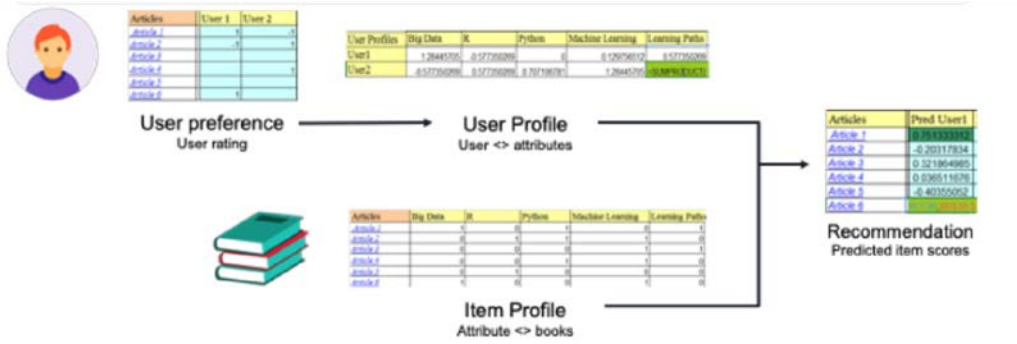
¿Qué técnicas usan los recomendadores?

- **Recomendaciones basadas en popularidad**
  - Se recomiendan productos populares (TOP 10 en Netflix)
- **Recomendaciones personalizadas**
  - Si se usan características del producto → aproximación basada en contenido
  - Si las características provienen de ratings de usuario → aproximación por filtrado colaborativo (user based, item based)



Aproximación basada en contenido

- Construir un perfil de usuario y compararlo con los productos (user profile – item)
  - con elementos explícitos como preguntas directas o escalas sobre gustos,..
    - Basado en sus gustos, preferencias generales,.. Evaluaciones (ratings) hechas sobre los items.
      - Pedirle que ordene una colección de elementos del más favorito al menos favorito
      - Presentarle dos elementos y pedirle que elija el que le parece mejor
      - Pedirle una lista de productos que le gustan
  - con elementos implícitos como observación (qué productos mira y durante cuánto tiempo, qué compra,..)



Recomendadores basados en contenido

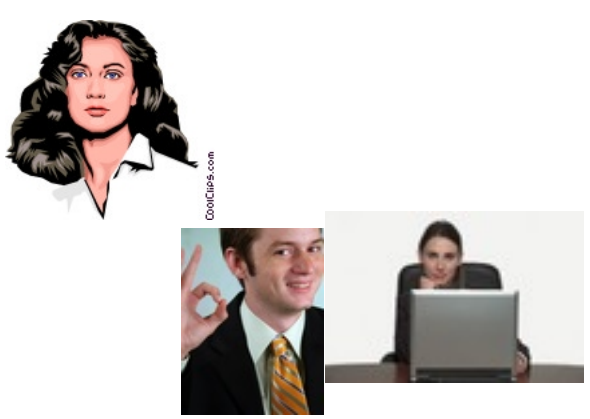
1. Similitud item-item: comparar productos con preferencias basado en los atributos de los productos.
2. Definir una medida de similitud adecuada con **los pesos ajustados.**
3. Computar similitud y recuperar los más parecidos.
   - ***Aproximación Nearest Neighborhood***
     - Recuperación aproximada – recuperar en base a la información de los N vecinos más similares
   - **Filtrado exacto**
4. Tener en cuenta la **diversidad** en los resultados

Recomendadores colaborativos

1. Pesar a los usuarios en función de su **similitud** con el usuario activo.

Recomendadores colaborativos

1. Pesar a los usuarios en función de su **similitud** con el usuario activo.
2. Seleccionar el subconjunto de usuarios que se usará como predictivo.
3. Normalizar los valores de preferencias (ratings) y computar una predicción para el usuario usando una combinación ponderada de los vecinos seleccionados.



Filtrado colaborativo tradicional (basado en usuarios)

- Ejemplo: recomendación colaborativa para películas

	User1	User2	User3	User4	User5	...	UserN
StarWars	1		3	1	2		
Titanic		3	2	3	4		
Dracula	3	3		2			
Aladdin	1			2			

  - **Medir la similitud** con el usuario activo **basada sólo en ratings.**
  - **No tengo información semántica sobre usuarios (edad, sexo, grupos..) ni sobre películas, sólo usamos los ratings numéricos**
  - Seleccionar el subconjunto de usuarios que se usará como predictivo.
    - **Seleccionamos u**suarios que han evaluado al menos el mismo conjunto de películas que yo.
  - Normalizar los valores de preferencias (ratings) y computar una predicción para el usuario usando una **combinación ponderada de los vecinos seleccionados:** Eliminar las películas que ya ha visto el usuario

Filtrado colaborativo tradicional (basada en usuarios)

- Para representar cada usuario se usa el modelo del espacio vectorial (VSM) donde cada cliente se representa como un vector de N dimensiones, donde N es el número de productos distintos del catálogo.
- Cada componente del vector tiene un valor de rating que puede estar ponderado para compensar best-sellers.
  - Por ejemplo, multiplicar las componentes del vector por la inversa del número de clientes que lo han evaluado o comprado.
  - Los productos menos conocidos son más relevantes.
- Observar que son vectores con muchas componentes vacías.

Filtrado colaborativo tradicional (basado en usuarios)

1º Encontrar N vecinos que sean similares al usuario Ua
Similitudes basadas en Coseno o basadas en Correlación

- Dado un conjunto de usuarios similares hay que usar algún método para seleccionar recomendaciones para un usuario Ua partir de los usuarios parecidos
- ¿Cómo medir la **similitud entre dos usuarios A y B**? – varios métodos → Similitud del coseno o correlación Pearson

2º. Predecir el Rating que el usuario Ua dará a un item j.

- Para un usuario y product item j → estimo su rating usando los ratings de usuarios parecidos.
- El item j es un producto no valorado por Ua pero que sí han valorado los usuarios parecidos

Textos modelo TF/IDF

- Con información textual (cada texto es un vector de palabras)
- Se cuenta la frecuencia y la importancia de las palabras en ese texto (por ejemplo, textos de descripcion de los items en RS)
- TF-IDF (Term Frequency-Inverse Document Frequency)

  - Valores que representan la importancia de una palabra de un documento teniendo en cuenta el corpus completo.
- La intuición es que la importancia de una palabra se incrementa proporcionalmente con el número de veces que la palabra aparece pero decrementa si es una palabra muy frecuente en el corpus.

$$similarity(\vec{A}, \vec{B}) = \cos(\vec{A}, \vec{B}) = \frac{\vec{A} \bullet \vec{B}}{\|\vec{A}\| * \|\vec{B}\|}$$

Textos: modelo TF/IDF

- **Frecuencia de los Terminos:**
  - Cada documento se representa con un vector del “peso” de cada palabra del corpus para este documento.
  - Podemos normalizar el valor en función de la frecuencia máxima de cualquier término en el documento

$$\mathrm{Tf}(t) = \frac{\text{Frequency occurence of term t in document}}{\text{Total number of terms in document}}$$

$$\mathrm{Idf}(t) = log_{10}\left(\frac{\text{Total Number of documents}}{\text{Number of documents containing term t}}\right)$$

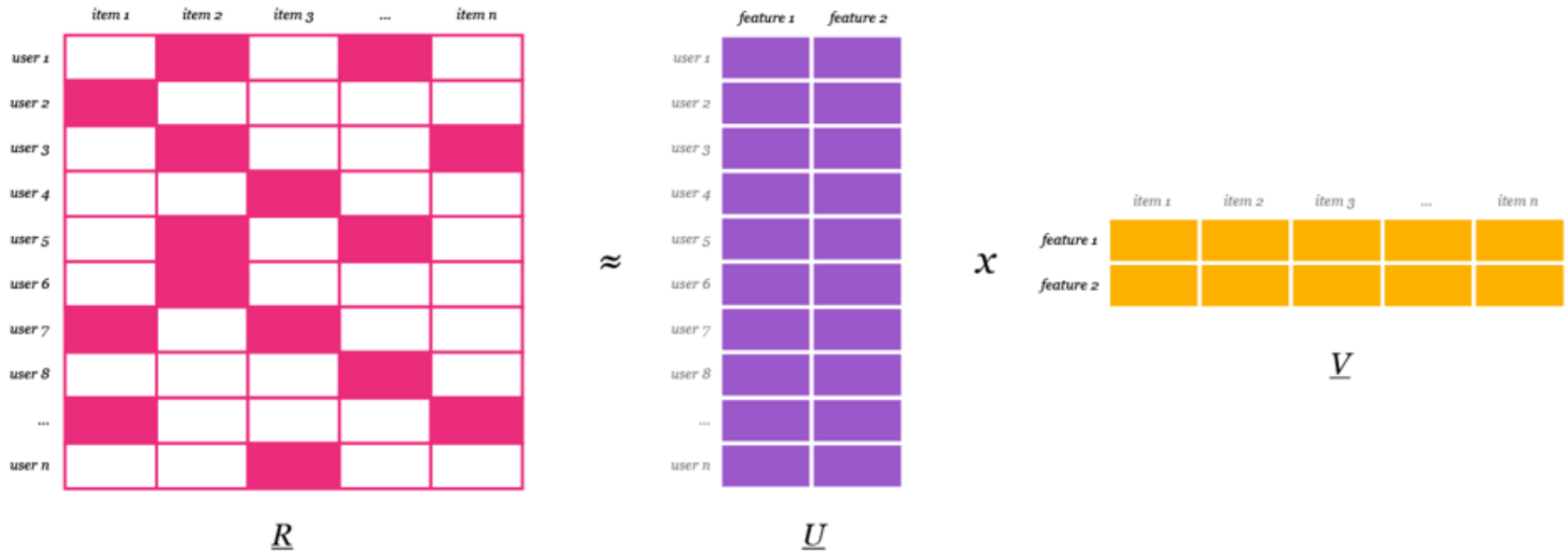
vectores de términos **TF-IDF:**Term Frequency-Inverse Document Frequency

Recomendadores basados en filtrado colaborativo vs contenido

- El filtrado colaborativo tiene algunas desventajas: cold-start, new-item problem, sparcity, transparency, long tail,…
- Ventajas: (de los recomendadores basados en contenidos)
- A diferencia del Filtrado Colaborativo, si los items tienen descripciones suficientes, nos evitamos el “new-item problem””
- Las representaciones del contenido son variadas y permiten utilizar diversas técnicas de procesamiento del texto, uso de información semántica, inferencias, etc.
- Es sencillo hacer un sistema más transparente: usamos el mismo contenido para explicar las recomendaciones.
- Inconvenientes: (de los recomendadores basados en contenido)
- Tienden a la sobre-especialización: va a recomendar items similares a los ya consumidos, creando una tendencia al “filter bubble”
- Ineficiencia e imprecisión
- Los métodos basados en filtrado colaborativo han mostrado ser, empíricamente, **más precisos** al momento de generar recomendaciones

https://medium.com/@rvillalongar/sistemas-recomendadores-basados-en-contenido-ece0227e7005

Mejora: Factorización de matrices (SVN)



Source: https://medium.com/radon-dev/als-implicit-collaborative-filtering-5ed653ba39fe

La descomposición singular de valores (Singular Value Descomposition, SVD) es una técnica de factorización de matrices que permite descomponer una matriz A en otras matrices U, S, V y realizar aproximaciones.

Evaluación de sistemas de recomendación

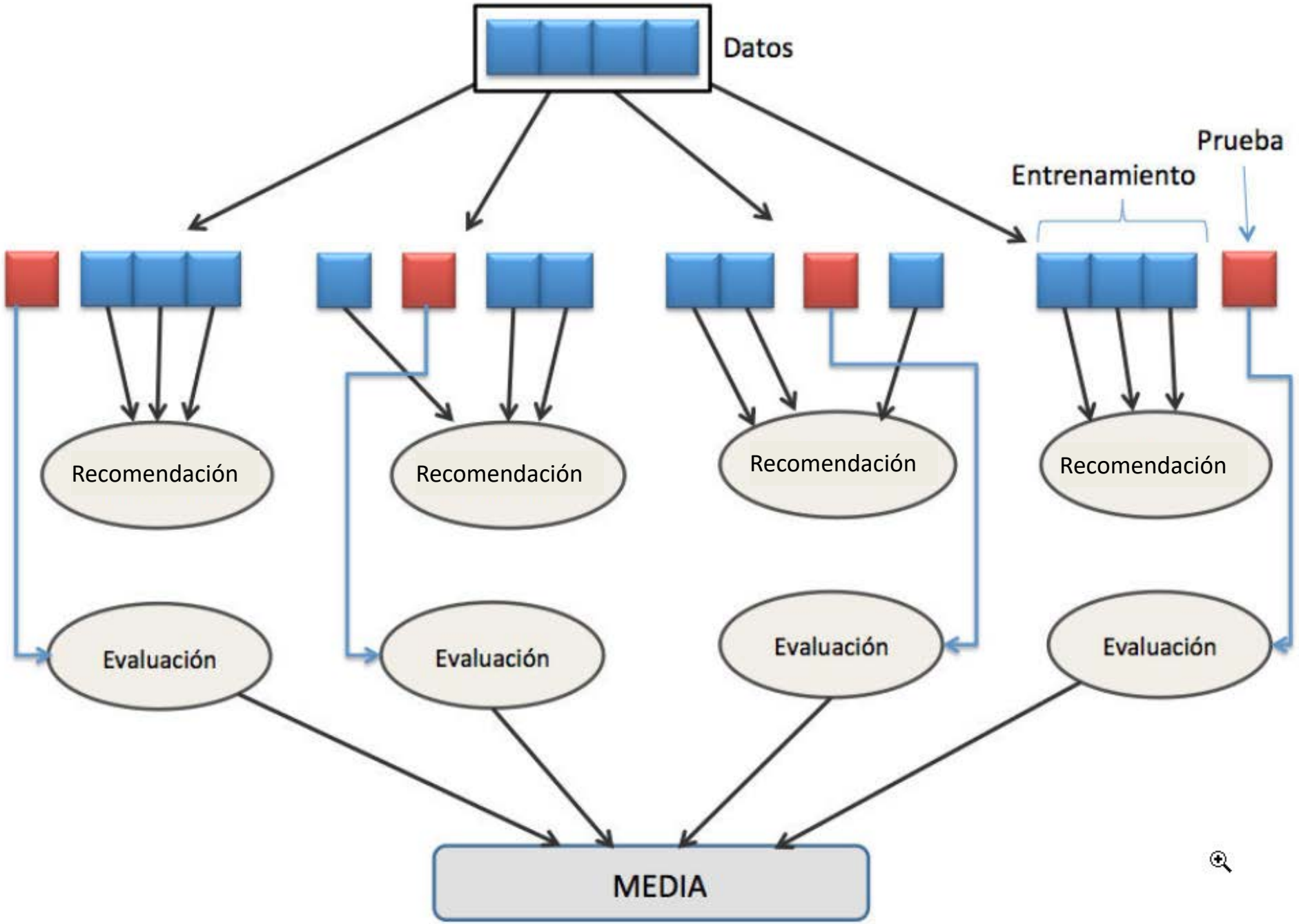
Conceptos

```
Evaluating RMSE, MAE of algorithm SVD on 5 split(s).

                  Fold 1  Fold 2  Fold 3  Fold 4  Fold 5  Mean    Std
RMSE (testset)    0.9362  0.9350  0.9342  0.9402  0.9349  0.9361  0.0022
MAE (testset)     0.7367  0.7355  0.7350  0.7444  0.7385  0.7380  0.0034
Fit time          5.50    5.68    5.51    6.61    5.69    5.80    0.41
Test time         0.27    0.21    0.24    0.21    0.21    0.23    0.02
```

- Las métricas clásicas a Mean Average Error (MAE), Mean Squared Error (MSE) y RMSE (Root Mean Squared Error) nos ayudan evaluar nuestros modelos de predicción y consisten en general en calcular la desviación promedio entre la predicción y el valor real.

- La ***Validación Cruzada* o *k-fold Cross Validation*** consiste en tomar los datos originales y crear a partir de ellos dos conjuntos separados: un primer conjunto de entrenamiento (y prueba), y un segundo conjunto de validación.
- El conjunto de datos originales se divide en *k* subconjuntos.
  - Se entrena con cada *k* subconjunto como conjunto de *prueba* del modelo, mientras que el resto de los datos se tomará como conjunto de *entrenamiento*.
  - Este proceso se repetirá *k* veces, y en cada iteración se seleccionará un conjunto de *prueba* diferente,
- Una vez finalizadas las iteraciones, se calcula la precisión y el error para cada uno de los modelos producidos, y para obtener la precisión y el error final se calcula el promedio de los *k* modelos entrenados.


Datos
Prueba
Entrenamiento
Recomendación
Recomendación
Recomendación
Recomendación
Evaluación
Evaluación
Evaluación
Evaluación
MEDIA

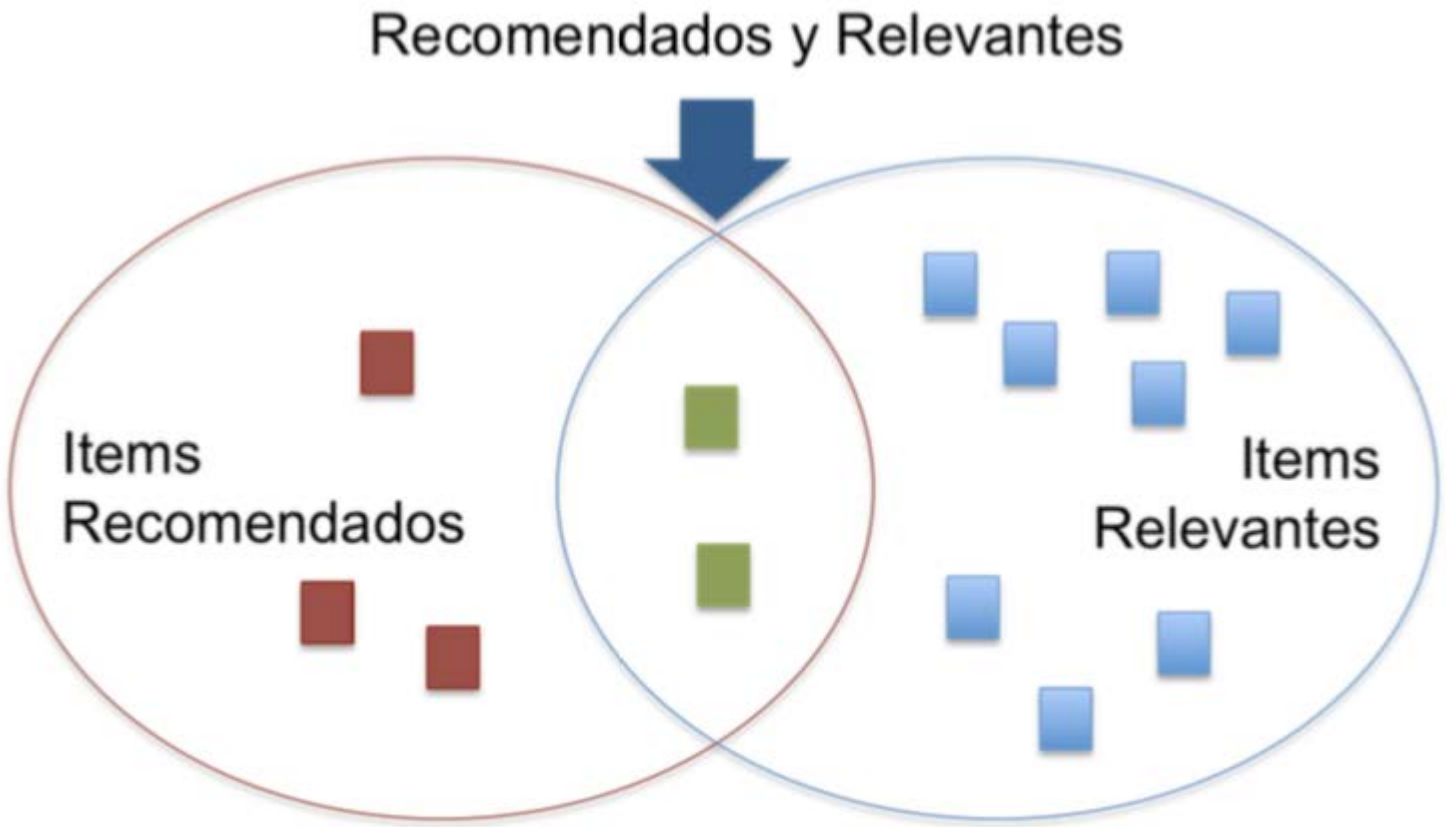
Precision y Recall

- Existen 2 métricas que ayudan a medir la efectividad con respecto a la lista de recomendaciones, *Precision* y *Recall*.
- Precisión es la fracción de los items recomendados que son relevantes, mientras que Recall es la fracción de items relevantes que son recomendados. Suena parecido pero no es lo mismo! Las fórmulas son las siguientes:

$$\text{Precision} = \frac{|\text{Recomendados} \cap \text{Relevantes}|}{|\text{Recomendados}|}$$

$$\text{Recall} = \frac{|\text{Recomendados} \cap \text{Relevantes}|}{|\text{Relevantes}|}$$

Ejemplo



Precision = 2/5
Recall = 2/10 = 1/5

Si sube el Recall, Precision baja, y viceversa.
Universo 1000 items
¿qué significa Recall 100%, Precisión 1%?

Recuperamos todos los elementos (los 1000)

Otra variante de esto es sacar el Precision at N (P@N) que sirve para comparar distintos recomendadores que entregan distintos números de ítems recomendados, midiendo la precisión hasta cierto punto, un N dado.

$$Precision@5 = \frac{2}{5} = 0,4$$

$$Precision@5 = \frac{3}{5} = 0,6$$

- Se pueden comparar recomendadores usando media armónica (F1)

$$F = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

La media armónica del recomendador 1 es: $F = \frac{2 \cdot \frac{2}{5} \cdot \frac{1}{5}}{\frac{2}{5} + \frac{1}{5}} = \frac{\frac{4}{25}}{\frac{3}{5}} = \frac{4}{15} \approx 0.2\overline{6}$

Mientras que la del recomendador 2 es: $F = \frac{2 \cdot 1 \cdot 0.01}{1 + 0.01} \approx 0.0198$

PRACTICA

Práctica: Objetivos

- Comprender los tipos básicos de recomendadores:
  - Recomendación basada en popularidad (parte 1)
  - Recomendación basada en contenido (parte 2)
  - Recomendación colaborativa user-user (parte 3)
- **Primera y segunda partes: recomendadores simples basados en popularidad, básico, con filtro de género y Recomendador basado en contenido**
  - Os dejo un archivo ipynb para hacer un tutorial recomendador basado en contenidos sin librerías con el dataset:
  - https://www.kaggle.com/rounakbanik/the-movies-dataset
  - Se pide hacer pruebas y comentar los resultados y observer las medidas de similitud utilizadas para comparar los objetos.
- **Tercera parte: Recomendador colaborativo usando la librería SURPRISE de Python.**
  - **Comentar los resultados de** evaluación, usando Error Absoluto Medio (MAE por sus siglas en ingles), Error Cuadratico Medio (MSE) y la Raiz del error cuadratico medio (RMSE), Recall y Precisión.

Surprise. http://surpriselib.com/

- Surprise is a Python scikit building and analyzing recommender systems that deal with explicit rating data.
- Surprise fue diseñado para realizar experimentos.
- Tienen datasets incorporados (Movielens, Jester) y algunos propios para hacer pruebas
- Puedes usar sus algoritmos de predicción de ratings: baseline algorithms, neighborhood methods, matrix factorization-based (SVD, PMF, SVD++, NMF), y muchos otros . Incorporan medidas de similitud (cosine, MSD, pearson…) permiten definir algunas propias
- The name *SurPRISE* (roughly :)) stands for Simple Python RecommendatIon System Engine.
- **Please note that surprise does not support implicit ratings or content-based information.**

In [4]:

```
## Ejemplo getting started de la documentación de SURPRISE
##http://surpriselib.com/

from surprise import SVD
from surprise import Dataset
from surprise.model_selection import cross_validate

# Load the movielens-100k dataset (download it if needed).
data = Dataset.load_builtin('ml-100k')

# Use the famous SVD algorithm.
algo = SVD()

# Run 5-fold cross-validation and print results.
cross_validate(algo, data, measures=['RMSE', 'MAE'], cv=5, verbose=True)
```

```
Evaluating RMSE, MAE of algorithm SVD on 5 split(s).

                  Fold 1  Fold 2  Fold 3  Fold 4  Fold 5  Mean    Std
RMSE (testset)    0.9276  0.9235  0.9427  0.9312  0.9445  0.9339  0.0083
MAE (testset)     0.7332  0.7267  0.7420  0.7350  0.7466  0.7367  0.0070
Fit time          23.37   22.35   24.35   28.37   31.42   25.97   3.40
Test time         1.10    0.65    0.97    1.03    1.97    1.15    0.44
```

Evaluación tradicional

- Comparar la predicción del Recomendador con el valor real y medir el error.

MAE: Mean Absolute Error

Error Absoluto Medio

$$MAE = \frac{\sum_{i=1}^{n} |\hat{r}_{ui} - r_{ui}|}{n}$$

MSE: Mean Squared Error

$$MSE = \frac{\sum_{i=1}^{n} (\hat{r}_{ui} - r_{ui})^2}{n}$$

RMSE: Root Mean Squared Error

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} (\hat{r}_{ui} - r_{ui})^2}{n}}$$

Se pueden probar distintos algoritmos y distintas similitudes

- https://surprise.readthedocs.io/en/stable/similarities.html

Similarities module

The similarities module includes tools to compute similarity metrics between users or items.
You may need to refer to the Notation standards, References page.
See also the Similarity measure configuration section of the User Guide.
Available similarity measures:

cosine	Compute the cosine similarity between all pairs of users (or items).
msd	Compute the Mean Squared Difference similarity between all pairs of users (or items).
pearson	Compute the Pearson correlation coefficient between all pairs of users (or items).
pearson_baseline	Compute the (shrunk) Pearson correlation coefficient between all pairs of users (or items) u

Probar y comparar algoritmos basados en knn

prediction_algorithms package

The `prediction_algorithms` package includes the prediction algorithms available for recommendation.

The available prediction algorithms are:

`random_pred.NormalPredictor`	Algorithm predicting a random rating based on the distribution of the traini
`baseline_only.BaselineOnly`	Algorithm predicting the baseline estimate for given user and item.
`knns.KNNBasic`	A basic collaborative filtering algorithm.
`knns.KNNWithMeans`	A basic collaborative filtering algorithm, taking into account the mean rating
`knns.KNNWithZScore`	A basic collaborative filtering algorithm, taking into account the z-score nor
`knns.KNNBaseline`	A basic collaborative filtering algorithm taking into account a *baseline* rating
`matrix_factorization.SVD`	The famous *SVD* algorithm, as popularized by Simon Funk during the Netflix
`matrix_factorization.SVDpp`	The *SVD++* algorithm, an extension of `SVD` taking into account implicit ratir
`matrix_factorization.NMF`	A collaborative filtering algorithm based on Non-negative Matrix Factorizati
`slope_one.SlopeOne`	A simple yet accurate collaborative filtering algorithm.
`co_clustering.CoClustering`	A collaborative filtering algorithm based on co-clustering.