

**UNIVERSITYHACK 2017®**  
**DATAATHON**

# Reto Microsoft Predictive Modelling



---

Grupo DATAPAK



**Universidad  
Europea**

LAUREATE INTERNATIONAL UNIVERSITIES

Autores:

**M<sup>a</sup> Lorena Prieto**

**Luis Nadal**

**Raúl Pingarrón**

**10/03/2017**

# DATAPAK

## Tabla de contenido

Introducción y esquema global de la solución	3
Descripción del motor de recomendación de DATAPAK	4
Pretratamiento de los datos	4
Comprobación de los datasets	4
Generación de rating para los datos de trabajo	4
Entrada de datos para el modelo	6
(1) Conjunto de entrenamiento	6
(2) Conjunto de test	6
Preparación de los datos dentro del modelo	7
(3) Edición de metadatos	7
(4) Edición de metadatos	8
Transformaciones de los datasets	10
(5) Transformación SQL	10
(6) Transformación SQL	11
(7) Transformación SQL	12
(8) Transformación SQL	13
(9) Eliminación de duplicados	14
(10) Eliminación de duplicados	14
Entrenamiento del modelo	15
(11) Train Matchbox Recommender	15
Predicción	16
(12) Score Matchbox Recommender	16
(13) Score Matchbox Recommender	18
Evaluación del modelo	20
(14) Evaluate Recommender	20
Salida del sistema	21
(15), (16), (17) Cambiar metadatos y convertir a CSV para generar el dataset de entrega	21
Visualización de resultados en Notebook Jupyter	22

## Introducción y esquema global de la solución

El equipo DATAPAK de la Universidad Europea de Madrid ha afrontado el desafío generando un motor de recomendación de Market Basket Analysis que es capaz de predecir con extrema exactitud cuál será el próximo producto financiero a contratar por el cliente.

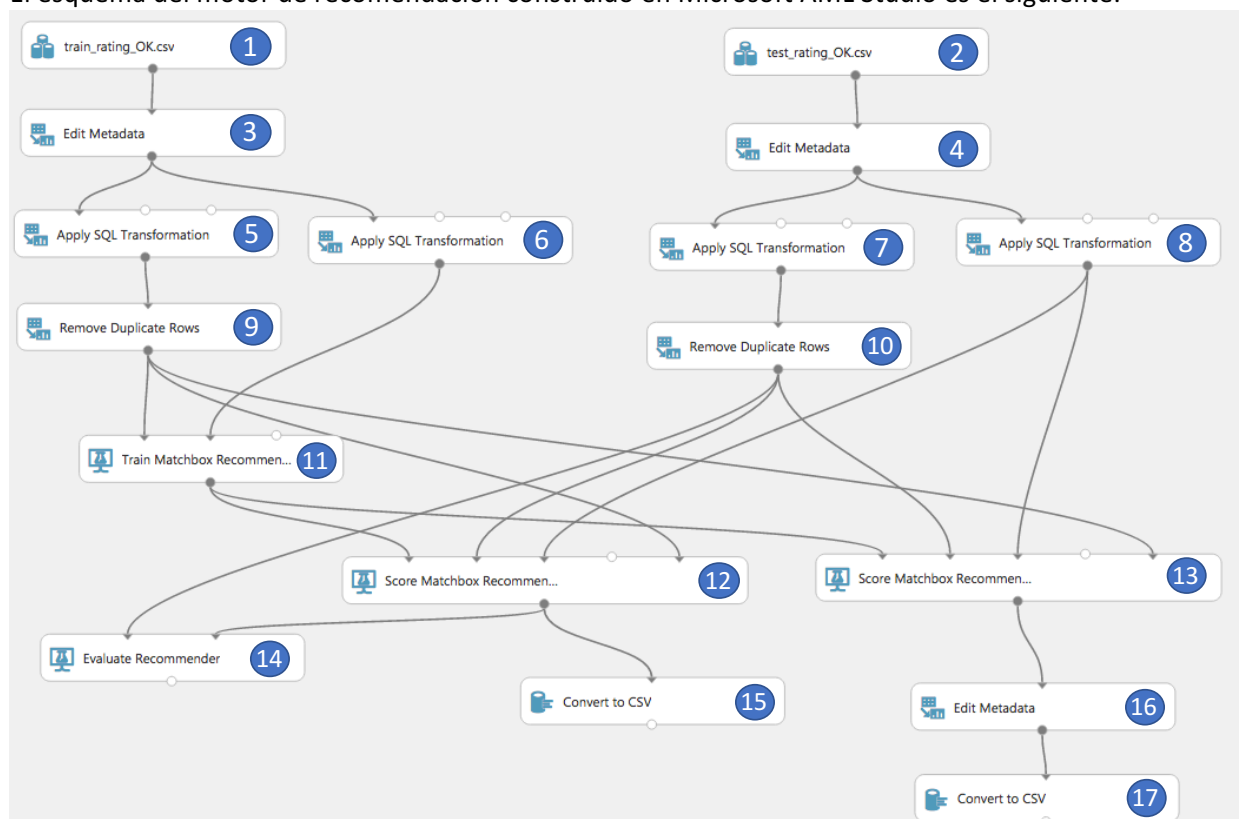
Para construir el motor de recomendación nos hemos apoyado en la herramienta **Microsoft Azure Machine Learning Studio** utilizando uno de los algoritmos de Machine Learning que incorpora: **MatchBox**, utilizándose una aproximación híbrida (Content-Based y Collaborative-Filtering) ya que se han utilizado características de los usuarios y de los elementos.

Matchbox hace uso de la información de contenido en forma de metadatos de usuario y elemento en combinación con la información de filtrado colaborativo del comportamiento del usuario anterior con el fin de predecir el valor de un elemento para un usuario. Los usuarios y elementos están representados por vectores de características que se mapean en un "atributo de espacio" de baja dimensión en el que la similitud se mide en términos de productos internos.

El modelo puede ser entrenado a partir de diferentes tipos de retroalimentación para aprender las preferencias de los elementos de usuario. En concreto se ha utilizado la alimentación basada en un conjunto de clasificaciones ordinales a escala individual.

El Paper que detalla el funcionamiento del algoritmo se puede consultar en la siguiente URL: <https://www.microsoft.com/en-us/research/wp-content/uploads/2009/01/www09.pdf>

El esquema del motor de recomendación construido en Microsoft AML Studio es el siguiente:



## Descripción del motor de recomendación de DATAPAK

### Pretratamiento de los datos

#### Comprobación de los datasets

Dado que los datos del dataset entregado se encontraban en un fichero de texto donde los campos se encontraban separados por el símbolo del pipe "|", fue necesario realizar un parseo de los mismos para convertir los ficheros a CSV (UTF-8, separados por coma). Para ello utilizamos la utilidad "sed" desde un sistema \*NIX de la siguiente manera:

```
muerdecables:~ raul$> sed 's/|/,/g' train2.txt > train2_csv.txt
muerdecables:~ raul$> sed 's/|/,/g' test2.txt > test2_csv.txt
```

Con los ficheros en formato CSV el tratamiento e importación a un servidor SQL y a Microsoft AML Studio era ya posible.

Antes de cargar los dataset a la plataforma de MSFT AML Studio para su uso en la generación del modelo predictivo, se analizaron el datasets resultantes del paso anterior en un gestor de bases de datos SQL (se utilizó una instancia local de servidor SQL) para comprobar la validez de los datos (repetidos, etc.) y realizar una preparación de los mismos.

#### Generación de rating para los datos de trabajo

De cara a calificar los datos obtenidos, se ha generado un rating que asocia un valor de 1 a 5 en función de la temporalidad (anual) de los mismos, y con ello ponderar así su relevancia. La motivación para incluir este criterio es que se puede considerar que los datos del intervalo más próximo en el tiempo (calificados con un 5) son más representativos que aquellos más lejanos en el tiempo.

Dividiendo el tiempo total en 5 partes, quedaría:

Año inicial: 1954 (obtenido al analizar los datos en el gestor de BBDD SQL)

Año final: 2017 (obtenido al analizar los datos en el gestor de BBDD SQL)

Tamaño del intervalo =  $(2017-1954)/5 = 12,6$  años

Se puede definir el **rating** referido de la siguiente manera:

$\text{rating} = 5 - \text{ParteEntera}((2017 - t)/12,6)$

donde  $t$  es el año de un registro dado

Se aplica esta calificación al conjunto completo de entrenamiento, y también al de test, es decir, generaremos unos nuevos datasets para entrenamiento y test que incluyan un campo adicional de rating.

Para ello, y habiendo importado previamente los ficheros `test2_csv.txt` y `train2_csv.txt` en el gestor de BBDD SQL (tablas `DATATHON.test_rating` y `DATATHON.train_rating` respectivamente), ejecutamos las siguientes *queries* para generar los ficheros:

```
insert into DATATHON.test_rating (ID_Customer, Cod_Prod, Cod_Fecha,
Socio_Demo_01, Socio_Demo_02,Socio_Demo_03,Socio_Demo_04,Socio_Demo_05,rating)
SELECT
ID_Customer, Cod_Prod, Cod_Fecha, Socio_Demo_01,
Socio_Demo_02,Socio_Demo_03,Socio_Demo_04,Socio_Demo_05,5-floor( (2017 -
cast(left(Cod_Fecha,4) as unsigned ))/12.6) as rating
FROM DATATHON.test
```

```
insert into DATATHON.train_rating (ID_Customer, Cod_Prod, Cod_Fecha,
Socio_Demo_01, Socio_Demo_02,Socio_Demo_03,Socio_Demo_04,Socio_Demo_05,rating)
SELECT
ID_Customer, Cod_Prod, Cod_Fecha, Socio_Demo_01,
Socio_Demo_02,Socio_Demo_03,Socio_Demo_04,Socio_Demo_05,5-floor( (2017 -
cast(left(Cod_Fecha,4) as unsigned ))/12.6) as rating
FROM DATATHON.train
```

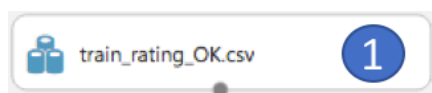
```
select
'ID_Customer','Cod_Prod','Cod_Fecha','Socio_Demo_01','Socio_Demo_02','Socio_De
mo_03','Socio_Demo_04','Socio_Demo_05','rating'
union all
SELECT
ID_Customer, Cod_Prod, Cod_Fecha, Socio_Demo_01,
Socio_Demo_02,Socio_Demo_03,Socio_Demo_04,Socio_Demo_05,rating
FROM DATATHON.train_rating
INTO OUTFILE 'train_rating_OK.csv'
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n';
```

```
select
'ID_Customer','Cod_Prod','Cod_Fecha','Socio_Demo_01','Socio_Demo_02','Socio_De
mo_03','Socio_Demo_04','Socio_Demo_05','rating'
union all
SELECT
ID_Customer, Cod_Prod, Cod_Fecha, Socio_Demo_01,
Socio_Demo_02,Socio_Demo_03,Socio_Demo_04,Socio_Demo_05,rating
FROM DATATHON.test_rating
INTO OUTFILE 'test_rating_OK.csv'
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n';
```

Como resultado de las dos últimas *queries* obtenemos los ficheros `test_rating_OK.csv` y `train_rating.csv`, en formato UTF-8 y CSV (campos separados por comas), que básicamente son los datasets de partida con una columna adicional de rating cuyo valor ha sido calculado según se ha explicado anteriormente, y que utilizaremos como base para construir nuestro modelo predictivo.

## Entrada de datos para el modelo

### (1) Conjunto de entrenamiento

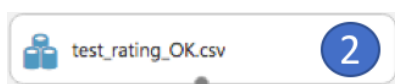


#### ▲ train\_rating\_OK.csv

SUBMITTED BY	21649158
SIZE	107 MB
FORMAT	GenericCSV

Una vez cargado el dataset “train\_rating\_OK.csv” previamente tratado a la plataforma de AML, se introduce como entrada del sistema.

### (2) Conjunto de test




#### ▲ test\_rating\_OK.csv

SUBMITTED BY	21649158
SIZE	36.7 MB
FORMAT	GenericCSV

Se procede de manera análoga con el conjunto de test “test\_rating\_OK.csv” previamente tratado.

## Preparación de los datos dentro del modelo

### (3) Edición de metadatos



**Edit Metadata**

Column

**Selected columns:**  
**Column names:**  
ID\_Customer, Cod\_Prod, Cod\_Fecha, Socio

Launch column selector

Data type











String

Categorical

Make categorical

Los módulos de recomendación basados en MatchBox necesitan que todos los campos del dataset sean de tipo STRING y CATEGÓRICO, por tanto, en este paso seleccionamos todas las columnas para realizar la conversión necesaria, de manera que el sistema no interprete que los códigos de producto, o los identificadores de usuario o de diferentes propiedades puedan responder a tipologías continuas, y puedan tomar valores no discretos.

Muestra del dataset en este punto:

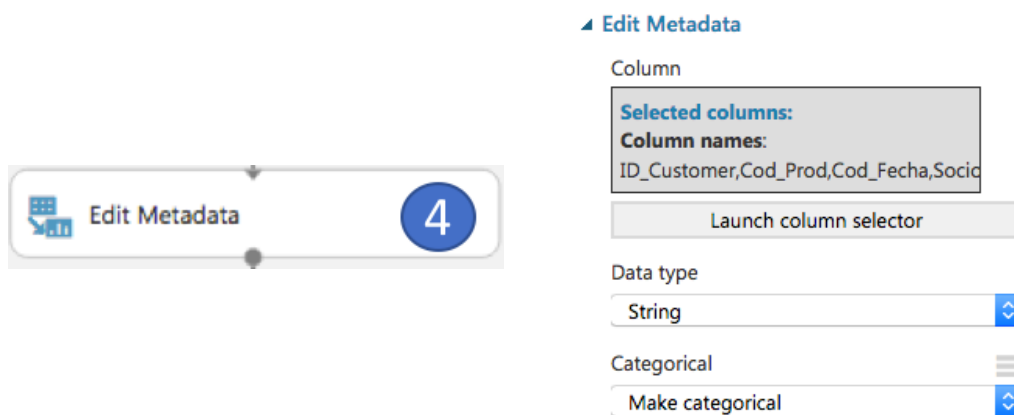
rows	columns									
3350601	9									
	ID_Customer	Cod_Prod	Cod_Fecha	Socio_Demo_01	Socio_Demo_02	Socio_Demo_03	Socio_Demo_04	Socio_Demo_05	rating	
view as										
	A0000001	601	5/1/2007 12:00:00 AM	5	4	3	1	0	5	
	A0000001	704	4/1/2013 12:00:00 AM	5	4	3	1	0	5	
	A0000001	2501	3/1/2006 12:00:00 AM	5	4	3	1	0	5	
	A0000001	2503	3/1/2006 12:00:00 AM	5	4	3	1	0	5	
	A0000001	1011	4/1/2011 12:00:00 AM	5	4	3	1	0	5	
	A0000002	601	6/1/1998 12:00:00 AM	5	5	1	1	0	4	
	A0000002	801	2/1/2006 12:00:00 AM	5	5	1	1	0	5	
	A0000002	9992	2/1/2015 12:00:00 AM	5	5	1	1	0	5	
	A0000002	301	3/1/1995 12:00:00 AM	5	5	1	1	0	4	
	A0000003	601	11/1/1985 12:00:00 AM	5	5	5	2	0	3	
	A0000003	301	5/1/2012 12:00:00 AM	5	5	5	2	0	5	
	A0000003	2501	12/1/2011 12:00:00 AM	5	5	5	2	0	5	
	A0000003	2503	12/1/2011 12:00:00 AM	5	5	5	2	0	5	
	A0000003	201	2/1/2016 12:00:00 AM	5	5	5	2	0	5	
	A0000004	601	5/1/2008 12:00:00 AM	5	5	3	1	0	5	
	A0000004	2301	5/1/2016 12:00:00 AM	5	5	3	1	0	5	
	A0000004	301	10/1/2008 12:00:00 AM	5	5	3	1	0	5	
	A0000004	201	12/1/2014 12:00:00 AM	5	5	3	1	0	5	

Se puede ver que el conjunto de entrenamiento dispone de más de 3.000.000 registros, con distribuciones no uniformes en todos los campos, a excepción del género (Socio\_Demo\_04), y el ID (cuya distribución es irrelevante).

De manera más pormenorizada, tenemos lo siguiente:

- **Productos más contratados:** 601 (650k registros), 301 (400k registros), 201 (350k registros).
- **Edad - Socio\_Demo\_01:** los tipos 4 (entre 45 y 65 años) y 3 (entre 30 y 45 años) suman 2/3 del total de registros.
- **Antigüedad - Socio\_Demo\_02:** más del 70% de los registros corresponden con clientes con 10 o más años de antigüedad.
- **Ingresos - Socio\_Demo\_03:** 2/3 de los registros se corresponden con niveles de renta entre 6.000 y 24.000 euros.
- **Género - Socio\_Demo\_04:** 56% hombres, 44% mujeres
- **Segmento - Socio\_Demo\_05:** la práctica totalidad de los registros se corresponden con el segmento de particulares.
- **Rating:** la gran mayoría de los registros han obtenido un rating de 5, lo que los ubica en los últimos 12,6 años.

#### (4) Edición de metadatos



Al igual que en el caso del dataset de entrenamiento, con el de test se procede a convertir todos los campos a STRING de tipo categórico, para asegurar que son percibidos como valores discretos por el sistema.



Muestra de los datos:

rows	columns								
1147873	9								
	ID_Customer	Cod_Prod	Cod_Fecha	Socio_Demo_01	Socio_Demo_02	Socio_Demo_03	Socio_Demo_04	Socio_Demo_05	rating
view as									
	B0891376	601	1/1/1954 12:00:00 AM	4	3	3	1	0	0
	B0889436	601	9/1/1954 12:00:00 AM	4	5	3	1	0	0
	B0889461	601	1/1/1957 12:00:00 AM	5	5	3	1	0	1
	B0889491	601	2/1/1959 12:00:00 AM	4	5	3	2	0	1
	B0889492	601	2/1/1959 12:00:00 AM	4	5	2	2	3	1
	B0889440	601	1/1/1960 12:00:00 AM	5	5	2	1	0	1
	B0875726	301	7/1/1960 12:00:00 AM	5	5	3	1	0	1
	B0875740	301	7/1/1960 12:00:00 AM	5	5	4	1	0	1
	B0875879	301	7/1/1960 12:00:00 AM	5	5	2	1	0	1
	B0875901	301	7/1/1960 12:00:00 AM	5	5	2	1	0	1
	B0875906	301	7/1/1960 12:00:00 AM	5	5	3	1	0	1
	B0875913	301	7/1/1960 12:00:00 AM	5	5	2	2	0	1
	B0875914	301	7/1/1960 12:00:00 AM	5	5	2	1	0	1

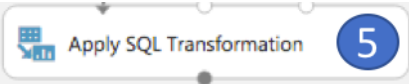
Se puede ver que el conjunto de test dispone de más de 1.000.000 registros, con distribuciones no uniformes en todos los campos, a excepción del género (Socio\_Demo\_04), y el ID (cuya distribución es irrelevante).

De manera más pormenorizada, tenemos lo siguiente:

- **Productos más contratados:** 601 (240k registros), 301 (140k registros), 201 (100k registros).
- **Edad - Socio\_Demo\_01:** los tipos 4 (entre 45 y 65 años) y 3 (entre 30 y 45 años) suman más del 70% del total de registros.
- **Antigüedad - Socio\_Demo\_02:** más del 80% de los registros corresponden con clientes con 10 o más años de antigüedad.
- **Ingresos - Socio\_Demo\_03:** 2/3 de los registros se corresponden con niveles de renta entre 6.000 y 24.000 euros.
- **Género - Socio\_Demo\_04:** 64% hombres, 36% mujeres
- **Segmento - Socio\_Demo\_05:** la práctica totalidad de los registros se corresponden con el segmento de particulares.
- **Rating:** la gran mayoría de los registros han obtenido un rating de 5, lo que los ubica en los últimos 12,6 años.

## Transformaciones de los datasets

### (5) Transformación SQL



Apply SQL Transformation

SQL Query Script




```
1 select (ID_Customer||'-'||Socio_Demo_01||'-'||Socio_Demo_02||'-'||Socio_Demo_03||'-'||Socio_Demo_04||'-'||Socio_Demo_05) as iduser,
2 Cod_Prod, rating from t1;
```

En este punto vamos a agregar todas las propiedades socio-demográficas al usuario, de manera que generemos una nueva variable llamada "iduser" que contenga tanto el ID\_Customer, como los campos socio-demográficos, generándose así iduser únicos para cada contratación realizada por cada usuario. Para su entrada en el modelo, se van a eliminar, en este caso, las columnas que hacen referencia a los campos Socio\_Demo.


El código SQL empleado para ello es el siguiente:

```
select (ID_Customer||'-'||Socio_Demo_01||'-'||Socio_Demo_02||'-'||Socio_Demo_03||'-'||Socio_Demo_04||'-'||Socio_Demo_05) as iduser,
Cod_Prod, rating from t1;
```

Muestra del resultado de la transformación:

rows	columns			
3350601	3			
	iduser	Cod_Prod	rating	
view as				
	A0000001-5-4-3-1-0	601	5	
	A0000001-5-4-3-1-0	704	5	
	A0000001-5-4-3-1-0	2501	5	
	A0000001-5-4-3-1-0	2503	5	
	A0000001-5-4-3-1-0	1011	5	
	A0000002-5-5-1-1-0	601	4	
	A0000002-5-5-1-1-0	801	5	
	A0000002-5-5-1-1-0	9992	5	
	A0000002-5-5-1-1-0	301	4	
	A0000003-5-5-5-2-0	601	3	
	A0000003-5-5-5-2-0	301	5	
	A0000003-5-5-5-2-0	2501	5	

## (6) Transformación SQL



Apply SQL Transformation

6

Apply SQL Transformation

SQL Query Script








```
1 select (ID_Customer||'-'||Socio_Demo_01||'-'||
2 '-'||Socio_Demo_03||'-'||Socio_Demo_04||'-'||
3 Socio_Demo_01, Socio_Demo_02, Socio_Demo_0
4 Socio_Demo_04, Socio_Demo_05 from t1
5 group by ID_Customer,Socio_Demo_01,Socio_D
6
```

En este caso, la transformación es la misma en lo que a la creación de la columna "iduser", sin embargo, en esta tabla nos quedaremos con los atributos de cada uno de estos usuarios, por lo que nos quedaremos con las columnas "iduser" y todas las características Socio\_Demo.

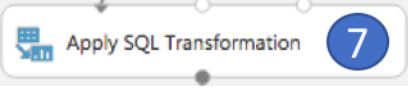
Para ello emplearemos el siguiente código:

```
select (ID_Customer||'-'||Socio_Demo_01||'-'||Socio_Demo_02||
'-'||Socio_Demo_03||'-'||Socio_Demo_04||'-'||Socio_Demo_05) as iduser,
Socio_Demo_01, Socio_Demo_02, Socio_Demo_03,
Socio_Demo_04, Socio_Demo_05 from t1
group by ID_Customer, Socio_Demo_01, Socio_Demo_02, Socio_Demo_03,
Socio_Demo_04, Socio_Demo_05;
```

Y esta sería una muestra del resultado obtenido:

rows	columns						
676370	6						
		iduser	Socio_Demo_01	Socio_Demo_02	Socio_Demo_03	Socio_Demo_04	Socio_Demo_05
view as							
		A0000001-5-4-3-1-0	5	4	3	1	0
		A0000002-5-5-1-1-0	5	5	1	1	0
		A0000003-5-5-5-2-0	5	5	5	2	0
		A0000004-5-5-3-1-0	5	5	3	1	0
		A0000005-5-5-3-1-0	5	5	3	1	0
		A0000006-5-5-3-1-0	5	5	3	1	0
		A0000007-5-5-2-2-0	5	5	2	2	0
		A0000008-3-5-2-1-3	3	5	2	1	3
		A0000009-4-5-2-1-0	4	5	2	1	0
		A0000010-5-5-2-2-0	5	5	2	2	0
		A0000011-4-5-4-2-0	4	5	4	2	0
		A0000012-4-5-2-2-0	4	5	2	2	0
		A0000013-3-5-2-2-0	3	5	2	2	0

## (7) Transformación SQL



**Apply SQL Transformation**

SQL Query Script



```
1 select (ID_Customer||'-'||Socio_Demo_01||'-'
2         '-'||Socio_Demo_03||'-'||Socio_Demo_04
3         Cod_Prod, rating from t1;
```

De manera análoga a lo realizado con el dataset de entrenamiento, procedemos con el de test, por lo que de nuevo vamos a agregar todas las propiedades socio-demográficas al usuario, de manera que generemos una nueva variable llamada "iduser" que contenga tanto el ID\_Customer, como los campos socio-demográficos, generándose así iduser únicos para cada contratación realizada por cada usuario. Para su entrada en el modelo, se van a eliminar, en este caso, las columnas que hacen referencia a los campos Socio\_Demo.


El código SQL empleado para ello es el siguiente:

```
select (ID_Customer||'-'||Socio_Demo_01||'-'||Socio_Demo_02||
        '-'||Socio_Demo_03||'-'||Socio_Demo_04||'-'||Socio_Demo_05) as iduser,
        Cod_Prod, rating from t1;
```

Muestra del resultado de la transformación:

rows	columns		
1147873	3		
		iduser	Cod_Prod rating
view as			
		B0891376-4-3-3-1-0	601 0
		B0889436-4-5-3-1-0	601 0
		B0889461-5-5-3-1-0	601 1
		B0889491-4-5-3-2-0	601 1
		B0889492-4-5-2-2-3	601 1
		B0889440-5-5-2-1-0	601 1
		B0875726-5-5-3-1-0	301 1
		B0875740-5-5-4-1-0	301 1
		B0875879-5-5-2-1-0	301 1
		B0875901-5-5-2-1-0	301 1
		B0875906-5-5-3-1-0	301 1
		B0875913-5-5-2-2-0	301 1

## (8) Transformación SQL


Apply SQL Transformation
8

**Apply SQL Transformation**

SQL Query Script







```
1 select (ID_Customer||'-'||Socio_Demo_01||'-'||
2 '||Socio_Demo_03||'-'||Socio_Demo_04||'-'||
3 Socio_Demo_01, Socio_Demo_02, Socio_Demo_0
4 Socio_Demo_04, Socio_Demo_05 from t1
5 group by ID_Customer, Socio_Demo_01, Socio_D
6
```

Al igual que en el caso del conjunto de entrenamiento, esta transformación es la misma que la aplicada en la Transformación SQL (7) en lo que a la creación de la columna "iduser", sin embargo, en esta tabla nos quedaremos con los atributos de cada uno de estos usuarios, por lo que nos quedaremos con las columnas "iduser" y todas las características Socio\_Demo.

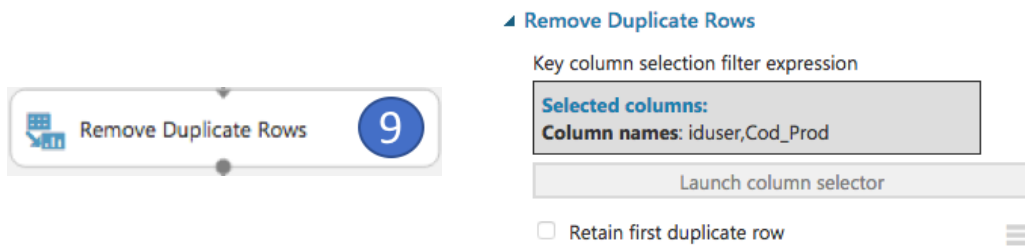
Para ello emplearemos el siguiente código:

```
select (ID_Customer||'-'||Socio_Demo_01||'-'||Socio_Demo_02||
'-'||Socio_Demo_03||'-'||Socio_Demo_04||'-'||Socio_Demo_05) as iduser,
Socio_Demo_01, Socio_Demo_02, Socio_Demo_03,
Socio_Demo_04, Socio_Demo_05 from t1
group by ID_Customer, Socio_Demo_01, Socio_Demo_02, Socio_Demo_03,
Socio_Demo_04, Socio_Demo_05;
```

Y esta sería una muestra del resultado obtenido:

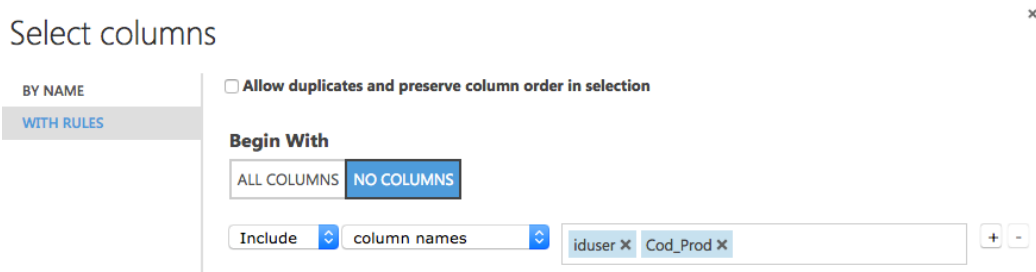
rows		columns				
258989		6				
	iduser	Socio_Demo_01	Socio_Demo_02	Socio_Demo_03	Socio_Demo_04	Socio_Demo_05
view as						
	B0676372-5-5-2-1-0	5	5	2	1	0
	B0676373-5-5-2-1-0	5	5	2	1	0
	B0676374-3-5-3-1-3	3	5	3	1	3
	B0676376-5-5-3-2-0	5	5	3	2	0
	B0676377-5-5-3-2-0	5	5	3	2	0
	B0676378-4-5-2-2-0	4	5	2	2	0
	B0676379-5-5-2-2-0	5	5	2	2	0
	B0676381-4-5-5-2-2	4	5	5	2	2
	B0676382-3-5-2-2-0	3	5	2	2	0
	B0676383-5-5-1-2-0	5	5	1	2	0
	B0676384-5-5-2-2-0	5	5	2	2	0
	B0676385-4-5-2-2-0	4	5	2	2	0

## (9) Eliminación de duplicados



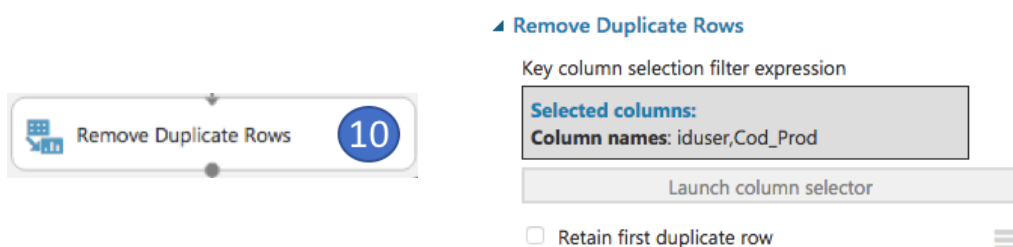
Una vez procesada la Transformación SQL (5), se procede a la eliminación de posibles duplicados, dejando fuera el rating, es decir que se comprueba que no haya valores duplicados de `iduser` y `Cod_Prod`, en cada categoría de rating.

Para ello se seleccionan las columnas `iduser` y `Cod_Prod` en el selector de columnas:



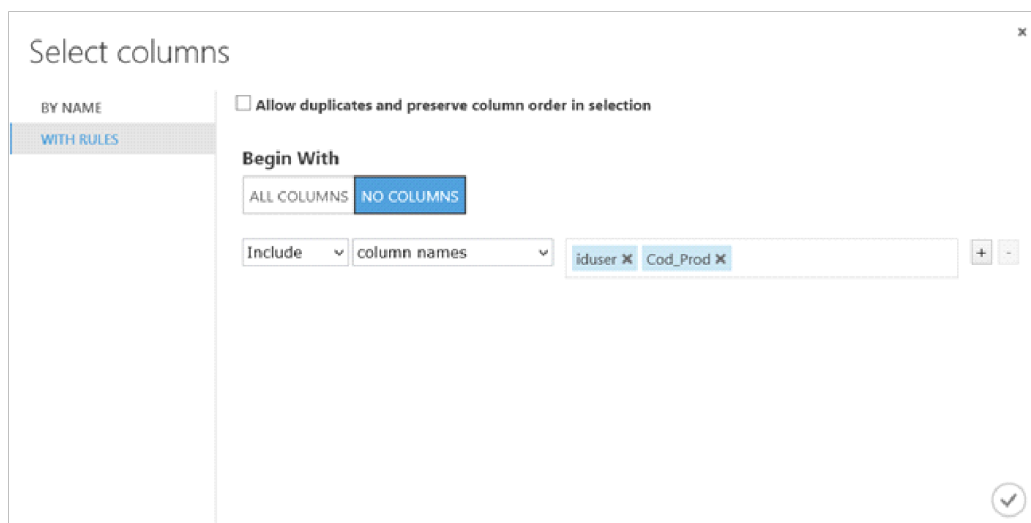
Al procesarlo y analizar el resultado se puede ver que no hay duplicados, pues el dato de salida contiene 3.350.601 filas, igual que la salida de la Transformación SQL (5).

## (10) Eliminación de duplicados



Una vez más, el esquema de trabajo es el mismo para la línea que sigue el dataset de test. En este caso, una vez procesada la Transformación SQL (7), se procede a la eliminación de posibles duplicados, dejando fuera el rating, es decir que se comprueba que no haya valores duplicados de `iduser` y `Cod_Prod`, en cada categoría de rating.

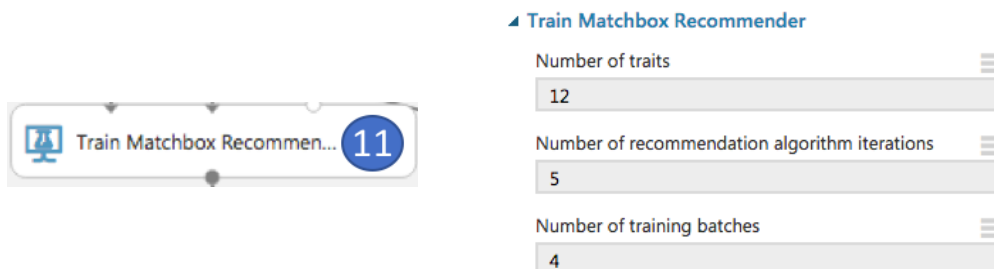
Para ello se seleccionan las columnas `iduser` y `Cod_Prod` en el selector de columnas:



Al procesarlo y analizar el resultado se puede ver que en este caso sí se han dado duplicados, pues el dato de salida contiene 1.147.687 filas, mientras que la salida de la Transformación SQL (7) contenía 1.147.873.

## Entrenamiento del modelo

### (11) Train Matchbox Recommender



Como se indica en la propia documentación de Microsoft, el Train Matchbox Recommender es un recomendador híbrido, es decir, combina el filtrado colaborativo con un enfoque basado en contenido. En casos como el que nos ocupa, en el que se pueden dar nuevos usuarios que accedan a los productos de Cajamar, las predicciones (es decir, la recomendación) se mejoran haciendo uso de la información de las características del usuario, en nuestro caso las variables `Socio_Demo_01-05`. No obstante, para el caso de usuarios ampliamente documentados (a través de su propio historial) el sistema hace una recomendación personalizada, basada en el rating de sus elecciones anteriores.

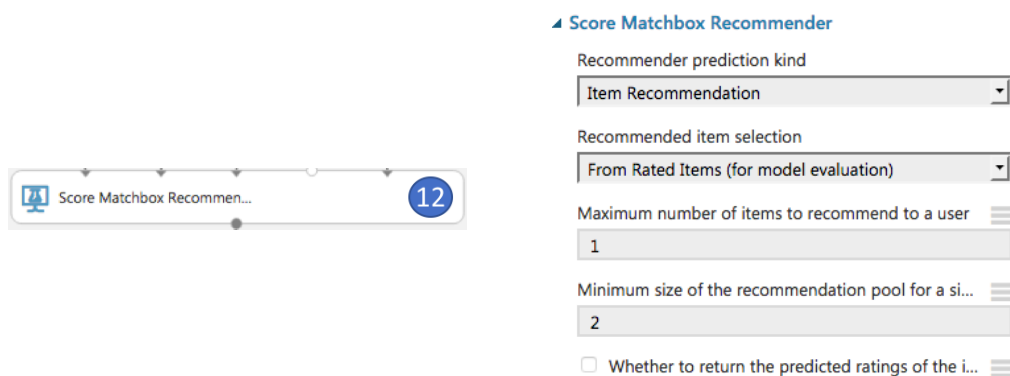
Tal y como funciona el Train Matchbox Recommender, se produce una transición entre un escenario y otro, adaptándose a la realidad de cada petición de predicción.

**Configuración:**

La configuración por defecto era con 10 traits (o propiedades), 2 iteraciones del algoritmo de recomendación y 4 procesos por lote para training.

Hemos estado experimentando con estos valores ya que aumentando el número de traits a considerar en el algoritmo, así como el número de iteraciones mejora significativamente la precisión (pero evidentemente prolonga el tiempo de ejecución y evaluación del modelo).

Finalmente hemos optado por una solución de compromiso configurando el Train Recommender con **12 Traits, 5 iteraciones del algoritmo y 4 batches de entrenamiento**. Con ello hemos conseguido mejorar la precisión original de 0.991107 a 0.995791.

**Predicción****(12) Score Matchbox Recommender**

Una vez obtenido el modelo, se pueden obtener recomendaciones a través del Score Matchbox Recommender, alimentándolo con el modelo obtenido de (11) Train Matchbox Recommender, y las salidas que provienen del tratamiento del dataset de test.

En este caso, especificamos la siguiente configuración:

Tipo de predicción del recomendador: recomendación de ítem (propuesta de producto a contratar).

Selección de ítem recomendada: de los ítems evaluados (para la evaluación del modelo).

Número máximo de ítems a recomendar por usuario: 1.

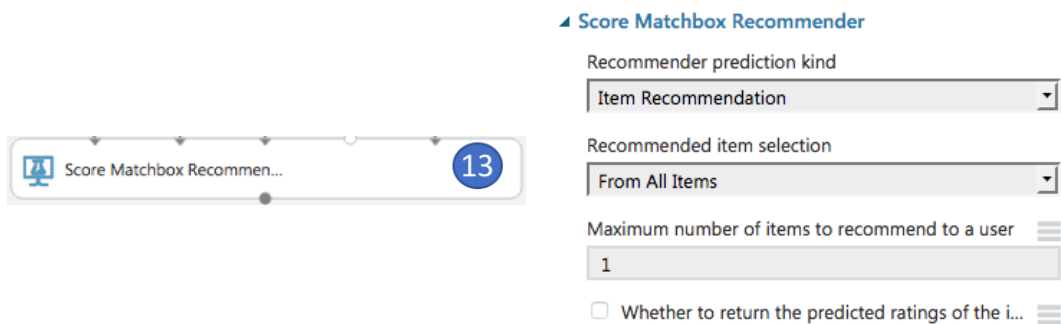
Tamaño mínimo del pool de recomendaciones para un usuario único: 2.

Esta configuración es la que tenemos que emplear para poder evaluar la bondad del modelo predictivo que hemos generado en base a los datasets recibidos.





### (13) Score Matchbox Recommender



**Score Matchbox Recommender**

Recommender prediction kind  
Item Recommendation

Recommended item selection  
From All Items

Maximum number of items to recommend to a user  
1

☐ Whether to return the predicted ratings of the i...

De nuevo, la estructura de este bloque (13) es muy similar a la del (12), este (13) Score Matchbox Recommender también se alimenta con el modelo obtenido de (11) Train Matchbox Recommender, y las salidas que provienen del tratamiento del dataset de test. sin embargo, la configuración es diferente, pues los resultados buscados son otros.

#### Configuración:

Tipo de predicción del recomendador: recomendación de ítem (propuesta de producto a contratar).





Selección de ítem recomendada: Para todos los ítems.

Número máximo de ítems a recomendar por usuario: 1.

En este caso, estamos obteniendo la mejor recomendación para la muestra de test a través de nuestro modelo predictivo. La salida de este bloque es el resultado del proyecto, pues se trata de la mejor recomendación para todos los casos recogidos en el dataset de test.

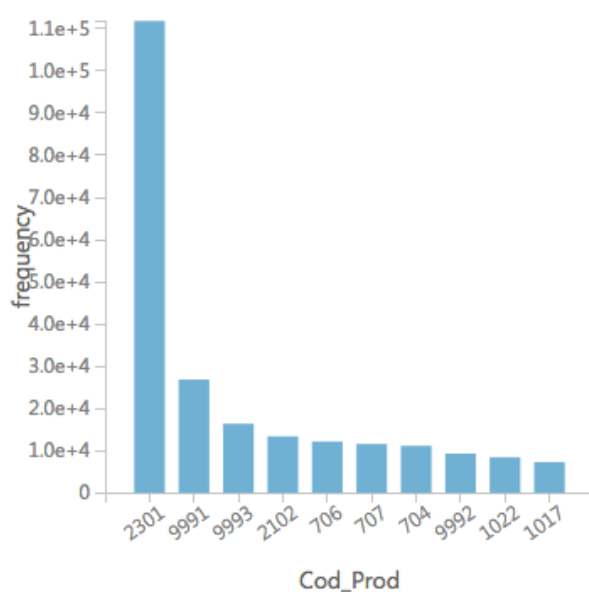
Muestra del resultado:

DATATHON Universityhack 2017 (DATAPAK) » Score Matchbox Recommender » Scored dataset

rows	columns
258989	2
view as	
 	 
B0891376-4-3-3-1-0	2102
B0889436-4-5-3-1-0	2301
B0889461-5-5-3-1-0	706
B0889491-4-5-3-2-0	9991
B0889492-4-5-2-2-3	2301
B0889440-5-5-2-1-0	707
B0875726-5-5-3-1-0	706
B0875740-5-5-4-1-0	706
B0875879-5-5-2-1-0	707
B0875901-5-5-2-1-0	707
B0875906-5-5-3-1-0	706
B0875913-5-5-2-2-0	9993
B0875914-5-5-5-1-0	706

Con respecto de estos resultados, cabe destacar **el producto más recomendado para el dataset entregado es el 2301** con un 43% de los casos, seguido del 9991 (un 10% de los casos), el 9993 (un 6,3%) y el 2102 (con un 5,2%)

A continuación se muestra el histograma de las recomendaciones:



## Evaluación del modelo

### (14) Evaluate Recommender


Evaluate Recommender
14

Minimum number of items that the query user and ...

2

Minimum number of users that the query item and ...

2

Este módulo sirve específicamente para evaluar la precisión de la predicción obtenida por el modelo. Las entradas son los resultados del (12) Score Matchbox Recommender, y el dataset de test, de manera que pueda cruzar las predicciones con los datos del dataset, y generar una métrica adecuada.

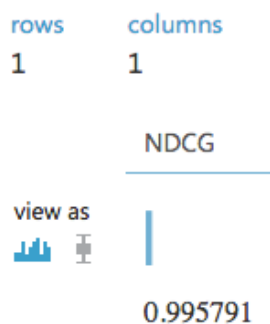
Configuración:

Mínimo número de items que el usuario buscado y el relacionado deben tener en común: 2

Mínimo número de usuarios que el item buscado y el relacionado deben tener en común: 2

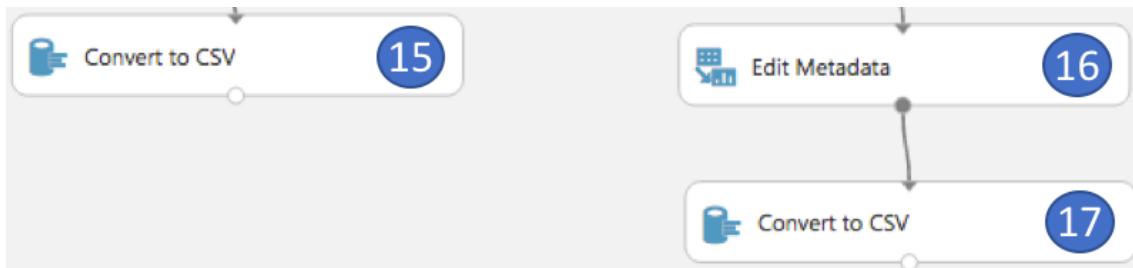
En nuestro caso, el valor de la métrica (precisión de nuestro modelo) resultó el siguiente:

DATATHON Universityhack 2017 (DATAPAK) > Evaluate Recommender > Metric



## Salida del sistema

(15), (16), (17) Cambiar metadatos y convertir a CSV para generar el dataset de entrega



Finalmente, sacamos los resultados de las predicciones de (12) y (13) Score Matchbox Recommender a sendos archivos CSV (botón derecho sobre la caja “Conver to CSV” y “Save as Dataset”).

Dado que el resultado final (el dataset entregable) solo ha de contener los campos ID\_Customer y Cod\_Prod, de manera que cada registro identifique a un cliente y al producto sugerido por el modelo generado a través del motor de recomendación construido, y que el formato del fichero a enviar ha de estar codificado en UTF-8 y tener el símbolo del pipe “|” como separador, tenemos que editar los metadatos antes de exportar el CSV para cambiarle el nombre a las columnas:

**Edit Metadata**

Column

**Selected columns:**  
Column names: User,Item 1

Launch column selector

Data type

Unchanged

Categorical

Unchanged

Fields

Unchanged

New column names

ID\_Customer,Cod\_Prod

El resultado será un CSV como sigue:

ID_Customer	Cod_Prod
B0891376-4-3-3-1-0	2102
B0889436-4-5-3-1-0	2301
B0889461-5-5-3-1-0	706
B0889491-4-5-3-2-0	9991
B0889492-4-5-2-2-3	2301
B0889440-5-5-2-1-0	707
B0875726-5-5-3-1-0	706

Y por tanto tendremos que “limpiar” el campo “ID\_Customer” para que se identifique con el campo “ID\_Customer” del dataset “Test.txt”. Para ello, creamos una nueva tabla en nuestro servidor SQL, importamos el CSV y mediante una query SQL hacemos esta transformación final obteniendo fichero “Test\_Mission.csv”.

```
LOAD DATA LOCAL INFILE 'Test_Mission.csv'
INTO TABLE test_mission
FIELDS TERMINATED BY ','
LINES terminated BY '\n'
IGNORE 1 ROWS -- ignora las cabeceras
(ID_Customer,Cod_Prod);
```

```
SELECT 'ID_Customer','Cod_Prod' UNION ALL SELECT
substring(ID_Customer,1,8),Cod_Prod FROM DATATHON.test_mission
INTO OUTFILE 'Test_Mission_ok.csv'
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n';
```

Por último, tenemos que pasarlo a TXT con separador por pipe “|”. Para ello lo parseamos utilizando la utilidad “sed” desde un sistema \*NIX de la siguiente manera:

```
muerdecables:~ raul$> sed 's/,/|/g' Test_Mission_ok.csv > Test_Mission.txt
```

Con esto ya tenemos el entregable en el formato que se nos pide.

## Visualización de resultados en Notebook Jupyter

Una de las ventajas de Microsoft AML es que permite sacar el resultado a un Notebook de Jupyter para poder realizar un procesamiento posterior de los datos con el fin de realizar visualizaciones y analíticas adicionales.

Hemos creado el siguiente Notebook a modo de muestra:

[https://nbviewer.jupyter.org/github/raul-pingarron/CAJAMAR-DATATHON-2017/blob/master/DATATHON\\_CAJAMAR\\_2017.ipynb](https://nbviewer.jupyter.org/github/raul-pingarron/CAJAMAR-DATATHON-2017/blob/master/DATATHON_CAJAMAR_2017.ipynb)