CAPSTONE RETAIL

DSMARKET Evaluation

Daniela Castillo, Carlos Esparducer, Diego Leal

ÍNDICE

ANÁLISIS

1. Formato de los datasets
2. Información de las variables
3. Tratamiento de los valores nulos
4. Nuevos datos y variables
5. Extracción de datos

CLUSTERING

Clustering por product

1. Creamos nuevas variables
2. Preparamos el dataset
3. Elbow Curve
4. Segmentación con “K” adecuada
5. Conclusión

Clustering por tienda

1. Creamos nuevas variables
2. Preparamos el dataset
3. Elbow Curve
4. Segmentación con “K” adecuada
5. Conclusión

SALES FORCASTING

USE CASE: REABASTECIMIENTO DE TIENDAS

CONCLUSIONES

ANÁLISIS

1. Formato de los *datasets*

A través de las funciones *info* y *shape* aplicadas a cada una de las tres tablas podemos observar que:

- Sales: tiene 30.490 filas y 1.920 columnas, de las cuales 7 son tipo *object* y 1913 son tipo *int64*.
- Prices: tiene 6.965.706 filas y 5 columnas, de las cuales 3 son de tipo *object* y 2 tipo *float64*.
- Calendar: tiene 1.913 filas y 5 columnas, de las cuales 4 son *object* y 1 *int64*.

2. Información de las variables

Para obtener más detalles aplicamos la función *head*, *value_counts* y *nunique*, así tendremos más información sobre el contenido de cada una de las tablas.

- Sales: analizamos información de cada columna
    - *id*: combinación entre la columna *item* y *store_code*. Tenemos 30.490 valores distintos.
    - *item*: combinación de *department* y un código numérico único que identifica cada item. Cuenta con 3.490 combinaciones.
    - *category*: palabra clave que identifica cada una de las tres categorías (ACCESORIES, HOME_&_GARDEN o SUPERMARKET).
    - *department*: combinación del *category* con un código numérico único. En este caso tenemos 7 posibles combinaciones.
    - store: nombre que identifica cada tienda. En total tenemos 10 tiendas.
    - *store_code*: código único que identifica cada una de las 10 tiendas y está compuesto por el acrónimo de *region* y un código numérico único.
    - *region*: ciudad en la que se ubica la tienda (New York, Boston o Philladelphia).
    - *d_1 a d_1913*: valor numérico que representa la cantidad de ventas diarias de cada *item* para un período de tiempo.

- Prices:
    - *item*: coincide con la columna *item* de la tabla *sales*.
    - *category*: coincide con la columna *category* de la tabla sales.
    - *store_code*: coincide con la columna *store_code* de la tabla sales.
    - *yearweek*: combinación del año (2011 - 2016) y el número de semana respecto al mismo (1 - 52). Formato (yyyyww)
    - *sell_price*: importe de las ventas acumuladas en cada semana por cada *item*.

- Calendar:
    - *date*: fecha formato yyyy-mm-dd. Contiene 1.913 valores distintos, que van desde 2011-01-29 a 2016-04-24.
    - *weekday*: día de la semana (Saturday - Friday).
    - *weekday_int*: código numérico que identifica el día de la semana (Saturday = 1 – Friday = 7).
    - *event:* indica días festivos o eventos que pueden tener impacto en ventas. Contiene 5 valores distintos (SuperBowl, Ramadanstarts, Thanksgiving, NewYear, Easter).

3. Tratamiento de valores nulos

Nuestro siguiente paso es el análisis de valores nulos en cada tabla. Comprobamos que la tabla sales no contiene nulos. Sin embargo, *prices* contiene 243.920 valores nulos en la columna *yearweek* y *calendar* tiene 1887 nulos en la columna *event*.

Para *prices* decidimos eliminar las filas con valores nulos en yearweek. Tomamos esta decisión basándonos en dos factores:

- En la información relevante del contenido de cada tabla proporcionada por la empresa nos indican que si hay valores nulos en la columna *sell_price*, es porque esa semana no hubo ventas de ese ítem en particular. Ahora bien, no hay nulos en esa columna, sino en *yearweek*.

- Si analizamos las filas que contienen nulos de *yearweek*, vemos que los valores de la columna *sell_price* correspondientes a esa filas tienen valores repetidos, es decir, el mismo valor que el de la última columna que no contenía nulos, por lo que no parecen datos reales y caza con la idea de no haber ventas en ese período de tiempo.

En el caso de *calendar*, sustituimos los nulos de *event* por 'No festivo'.

4. Nuevos valores y datos

Decidimos cambiar las columnas *d_1* a *d_1913* de la tabla *sales* por las fechas incluidas en la tabla *calendar*. Para ello pasamos toda la columna *date* de *calendar* a una lista, donde también incluiremos las columnas categóricas de la tabla *calendar* y así hacemos la sustitución total de columnas.

Como también nos interesa tener las ventas anuales, decidimos crear una lista con las fechas de cada año (enero a diciembre) para poder sumar las ventas por ítem anuales. Creamos 6 nuevas columnas, con el nombre de cada año (2011 - 2016).

Finalmente creamos una última columna que contenga la totalidad de ventas por cada id del período llamada *id_cnt_sale*.

Calculamos el precio por item sumando la columna *sell_price* de la tabla *prices* y agrupando por *item*. Sumamos las ventas de todo el período de la columna *id_cnt_sale* de la tabla *sales* agrupada también por cada *item*.

Al dividir el acumulado de ventas entre la cantidad de ítems vendidos obtenemos el precio unitario. Añadimos esta nueva columna llamada *item_price* a la tabla *sales*.

Cambiamos el tipo object que tiene la columna *date* de la tabla calendar a tipo *fecha*.

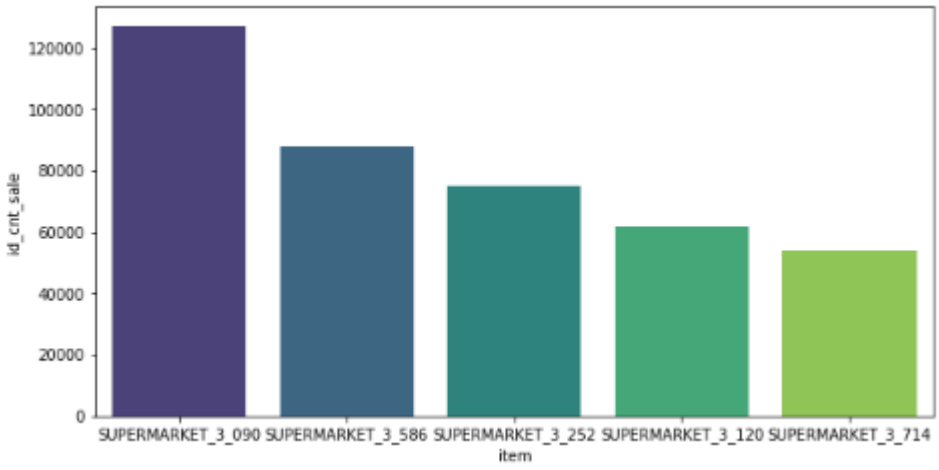
5. Extracción de datos

Trabajaremos con el dataset *sales*:

- Productos más/menos vendidos en el período en todas las tiendas: aplicamos un *sort_values* de la columna *item_cnt_sales*, con el parámetro *ascending* en *False* y obtenemos los productos más vendidos. Aprovechamos para eliminar los duplicados con *drop_duplicates*, pues nos interesa saber los items únicos. Filtramos por *head(5*) para los más vendidos y *tail(5)* para los menos.

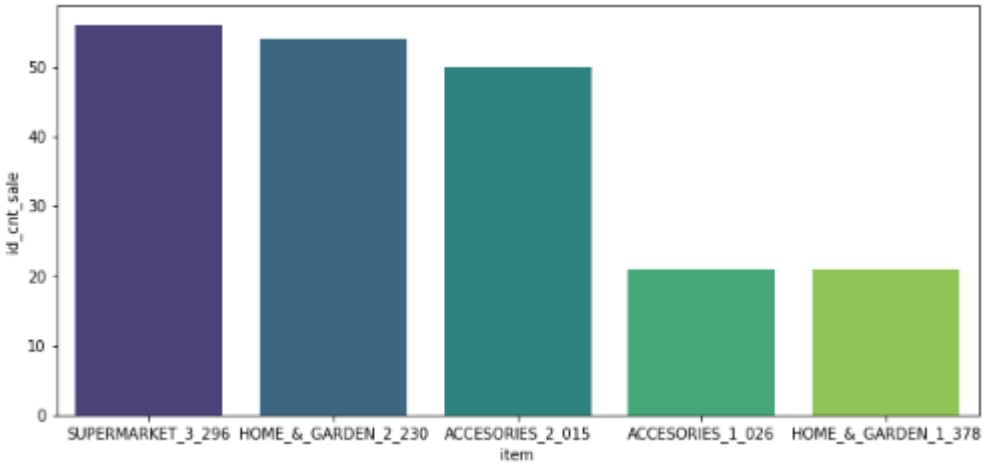
Top 5:

item	item_cnt_sale
SUPERMARKET_3_090	127203
SUPERMARKET_3_586	87691
SUPERMARKET_3_252	74971
SUPERMARKET_3_120	61899
SUPERMARKET_3_714	54080



Bottom 5:

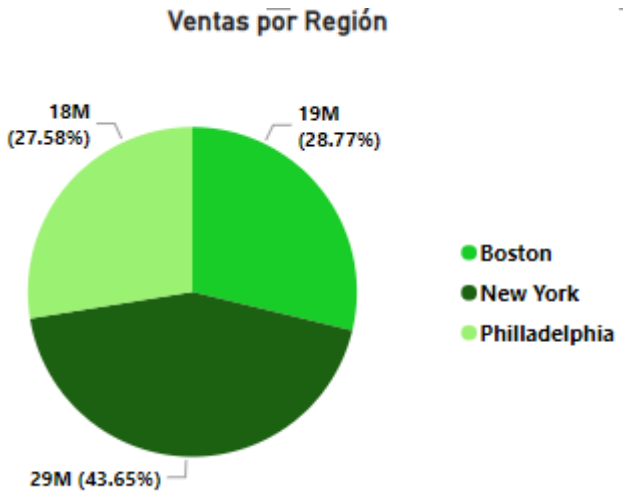
item	item_cnt_sale
SUPERMARKET_3_296	56
HOME_&_GARDEN_2_230	54
ACCESORIES_2_015	50
ACCESORIES_1_026	21
HOME_&_GARDEN_1_378	21



Podemos observar como los 5 productos más vendidos pertenecen a la categoría *Supermarket*. Esto nos da ua idea de la categoría que más ventas genera a la empresa.

- Ventas totales y productos más/menos vendidos por ciudad en todo el período: agrupamos por región y con un *get_group* sobre cada ciudad, ordenamos con *sort_values* en orden descendiente o ascendiente.

  Eliminamos duplicados y reiniciamos el índice de cada una para poder aplicar un *pd.concat* uniendo las 3 tablas.



Top 5:

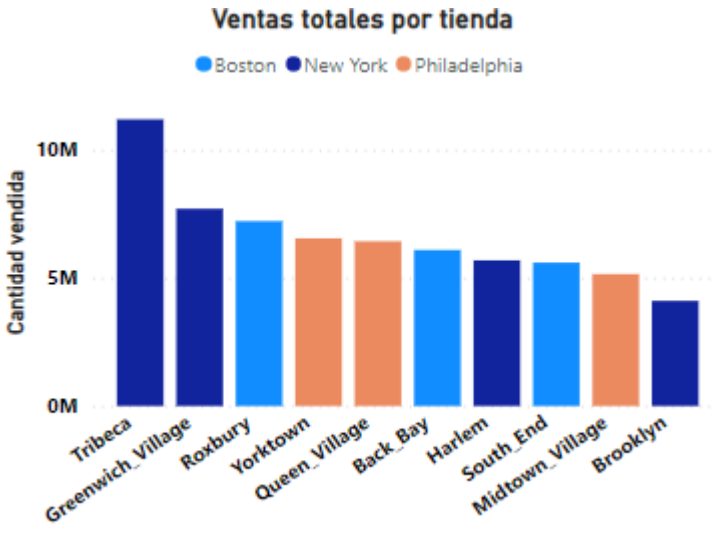
item_NY	item_sales_NY	item_Boston	item_sales_Boston	item_Philly	item_sales_Philly
SUPERMARKET_3_090	250502	SUPERMARKET_3_586	192835.0	SUPERMARKET_3_090	121434
SUPERMARKET_3_586	134386	SUPERMARKET_3_090	119496.0	SUPERMARKET_3_226	98227
SUPERMARKET_3_120	88795	SUPERMARKET_3_252	114153.0	SUPERMARKET_3_586	86080
SUPERMARKET_3_252	81456	SUPERMARKET_3_555	97496.0	SUPERMARKET_3_694	85633
SUPERMARKET_3_541	80491	SUPERMARKET_3_377	73797.0	SUPERMARKET_3_555	57969

Bottom 5:

item_NY	item_sales_NY	item_Boston	item_sales_Boston	item_Philly	item_sales_Philly
ACCESORIES_2_027	105	HOME_&_GARDEN_2_077	63	HOME_&_GARDEN_2_216	64
ACCESORIES_2_084	103	HOME_&_GARDEN_2_245	62	HOME_&_GARDEN_2_396	53
ACCESORIES_2_111	98	SUPERMARKET_2_071	59	HOME_&_GARDEN_2_101	47
HOME_&_GARDEN_2_175	94	SUPERMARKET_3_171	45	ACCESORIES_1_389	46
HOME_&_GARDEN_1_300	90	HOME_&_GARDEN_2_101	45	HOME_&_GARDEN_2_130	41

Entre las ciudades hay coincidencias en cuanto a los productos más vendidos, como el SUPERMARKET_3_90 y SUPERMARKET_3_586. Sin embargo, no ocurre lo mismo con los productos menos vendidos, que varían mucho en cada ciudad.

- Ventas totales por tienda y productos más/menos vendidos de cada una: agrupamos por cada código único de tienda (*store_code*). Ordenamos con *sort_values* en orden descendiente y concatenamos las 3 o 4 tiendas de cada ciudad.



New York: *Greenwich Village*, *Harlem*, *Tribeca* y *Brooklyn*

Top 5:

item_NY1GV	item_sales_NY1GV	item_NY2H	item_sales_NY2H	item_NY3T	item_sales_NY3T	item_NY4B	item_sales_NY4B
SUPERMARKET_3_090	127203	SUPERMARKET_3_586	63416	SUPERMARKET_3_090	250502	SUPERMARKET_3_090	52264
SUPERMARKET_3_586	87691	SUPERMARKET_3_252	57328	SUPERMARKET_3_586	134386	SUPERMARKET_3_586	32557
SUPERMARKET_3_252	74971	SUPERMARKET_3_090	56169	SUPERMARKET_3_120	88795	SUPERMARKET_3_587	24333
SUPERMARKET_3_120	61899	SUPERMARKET_3_555	31001	SUPERMARKET_3_252	81456	SUPERMARKET_3_252	23417
SUPERMARKET_3_714	54080	SUPERMARKET_1_218	28845	SUPERMARKET_3_541	80491	SUPERMARKET_3_808	21282

Bottom 5:

item_NY1GV	item_sales_NY1GV	item_NY2H	item_sales_NY2H	item_NY3T	item_sales_NY3T	item_NY4B	item_sales_NY4B
SUPERMARKET_3_296	56	SUPERMARKET_2_209	30	HOME_&_GARDEN_2_397	34	HOME_&_GARDEN_1_400	22
HOME_&_GARDEN_2_230	54	SUPERMARKET_2_117	29	ACCESORIES_1_052	32	HOME_&_GARDEN_1_183	22
ACCESORIES_2_015	50	SUPERMARKET_1_079	27	HOME_&_GARDEN_1_073	19	HOME_&_GARDEN_2_307	21
ACCESORIES_1_026	21	SUPERMARKET_2_337	25	HOME_&_GARDEN_1_336	18	HOME_&_GARDEN_1_512	20
HOME_&_GARDEN_1_378	21	SUPERMARKET_3_778	12	HOME_&_GARDEN_1_020	10	HOME_&_GARDEN_2_216	19

En New York hay 3 productos estrella que son top en ventas en todas las tiendas, que son SUPERMARKET_3_90, SUPERMARKET_3_586, SUPERMARKET_3_252. Pero los productos menos vendidos no guardan correlación entre cada tienda.

Boston: *South End*, *Roxbury* y *Back Bay*

Top 5:

item_BOS1SE	item_sales_BOS1SE	item_BOS2R	item_sales_BOS2R	item_BOS3BB	item_sales_BOS3BB
SUPERMARKET_3_586	112454	SUPERMARKET_3_586	192835	SUPERMARKET_3_586	150122
SUPERMARKET_3_090	93684	SUPERMARKET_3_090	119496	SUPERMARKET_3_090	114854
SUPERMARKET_3_555	69516	SUPERMARKET_3_252	114153	SUPERMARKET_3_252	86632
SUPERMARKET_3_252	56294	SUPERMARKET_3_555	97496	SUPERMARKET_3_555	77673
SUPERMARKET_3_587	48277	SUPERMARKET_3_377	63174	SUPERMARKET_3_377	73797

Bottom 5:

item_BOS1SE	item_sales_BOS1SE	item_BOS2R	item_sales_BOS2R	item_BOS3BB	item_sales_BOS3BB
SUPERMARKET_3_180	32	ACCESORIES_2_111	49	SUPERMARKET_2_310	34
HOME_&_GARDEN_2_101	29	SUPERMARKET_3_171	45	ACCESORIES_1_217	34
ACCESORIES_2_025	28	HOME_&_GARDEN_2_101	45	ACCESORIES_2_056	32
ACCESORIES_1_217	28	HOME_&_GARDEN_1_378	44	ACCESORIES_1_125	31
HOME_&_GARDEN_2_338	28	SUPERMARKET_2_209	24	SUPERMARKET_2_071	16

Caso similar al de New York, hay 4 productos (SUPERMARKET_3_90, SUPERMARKET_3_586, SUPERMARKET_3_252 y SUPERMARKE_3_555) que coinciden en el top 5 de todas las tiendas de Boston, no siendo así con los menos vendidos.

Philadelphia: *Midtown Village*, *Yorktown* y *Queen Village*

Top 5:

item_PHI1MV	item_sales_PHIMV	item_PHI2Y	item_sales_PHI2Y	item_PHI3QV	item_sales_PHI3QV
SUPERMARKET_3_226	78993	SUPERMARKET_3_226	69966	SUPERMARKET_3_090	121434
SUPERMARKET_3_694	41228	SUPERMARKET_3_694	59235	SUPERMARKET_3_226	98227
SUPERMARKET_3_090	34852	SUPERMARKET_3_007	52020	SUPERMARKET_3_586	86080
SUPERMARKET_3_586	32119	SUPERMARKET_3_234	50043	SUPERMARKET_3_694	85633
SUPERMARKET_3_714	30980	SUPERMARKET_2_360	48043	SUPERMARKET_3_555	57969

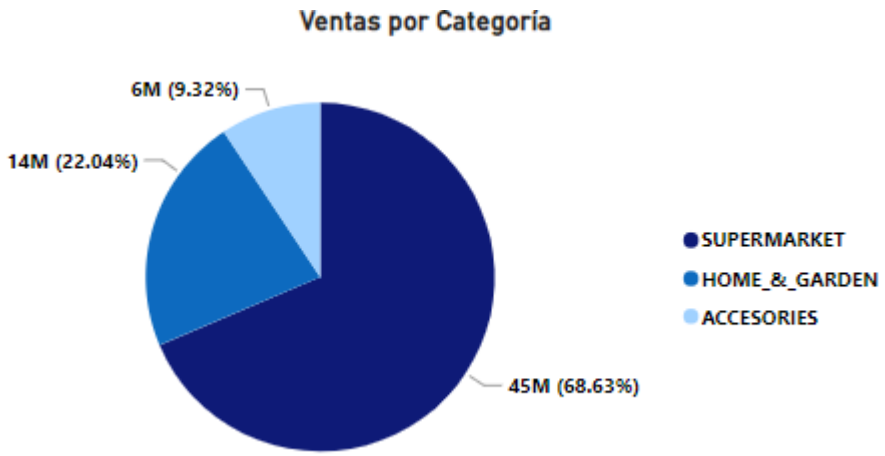
Bottom 5:

item_PHI1MV	item_sales_PHIMV	item_PHI2Y	item_sales_PHI2Y	item_PHI3QV	item_sales_PHI3QV
HOME_&_GARDEN_2_130	41	HOME_&_GARDEN_2_307	28	ACCESORIES_1_186	27
ACCESORIES_2_012	39	HOME_&_GARDEN_2_101	27	HOME_&_GARDEN_2_419	27
ACCESORIES_1_336	38	ACCESORIES_1_212	25	ACCESORIES_2_108	27
HOME_&_GARDEN_2_005	35	HOME_&_GARDEN_2_130	23	HOME_&_GARDEN_2_448	25
ACCESORIES_1_402	28	SUPERMARKET_2_057	16	ACCESORIES_1_170	13

Si bien encontramos coincidencia con 2 de los productos más vendidos a nivel global de ciudad, en Philadelphia hay 2 items más que tiene un nivel alto de ventas en todas tiendas y que no encontramos en el top de New York o Boston, que son SUPERMARKET_3_226 y SUPERMARKET_3_694.

Respecto a los productos menos vendidos, vemos el mismo patrón que con las otras ciudades.

- Ventas totales por categoría y productos más/menos vendidos: realizamos los pasos anteriores, pero esta vez agrupamos por cada una de las 3 categorías: *Accesories*, *Home&Garden* y *Supermarket*. Como no hay una distribución igual entre las 3 categorías, esta vez hemos cambiado el parámetro *ascending* a *True*, puesto que solo aplicar la función *tail(5)* no nos sirve.



Top 5:

item_AC	item_Sales_AC	item_H&G	item_sales_H&G	item_SM	item_sales_SM
ACCESORIES_1_234	29742	HOME_&_GARDEN_1_118	43351	SUPERMARKET_3_090	250502
ACCESORIES_1_348	22744	HOME_&_GARDEN_1_459	36970	SUPERMARKET_3_586	192835
ACCESORIES_1_371	22595	HOME_&_GARDEN_1_334	36811	SUPERMARKET_3_252	114153
ACCESORIES_1_254	20996	HOME_&_GARDEN_1_303	34897	SUPERMARKET_3_226	98227
ACCESORIES_1_268	18904	HOME_&_GARDEN_1_521	30266	SUPERMARKET_3_555	97496

Bottom 5:

item_AC	item_Sales_AC	item_H&G	item_sales_H&G	item_SM	item_sales_SM
ACCESORIES_2_111	112.0	HOME_&_GARDEN_2_175	100.0	SUPERMARKET_3_220	118
ACCESORIES_2_099	122.0	HOME_&_GARDEN_2_005	107.0	SUPERMARKET_2_239	181
ACCESORIES_2_084	125.0	HOME_&_GARDEN_2_392	122.0	SUPERMARKET_2_073	208
ACCESORIES_2_108	127.0	HOME_&_GARDEN_2_060	123.0	SUPERMARKET_3_255	227
ACCESORIES_1_212	133.0	HOME_&_GARDEN_2_487	132.0	SUPERMARKET_3_466	228

El *item* más vendido de la categoría *Supermarket* supera con creces al de la categoría *Home&Garden* o *Accesories.*

- Productos más vendidos por año:

Top 5:

item_2011	sales_2011	item_2012	sales_2012	item_2013	sales_2013
SUPERMARKET_3_586	36574	SUPERMARKET_3_090	74913	SUPERMARKET_3_090	64610
SUPERMARKET_3_090	28466	SUPERMARKET_3_586	41182	SUPERMARKET_3_586	37623
SUPERMARKET_3_587	25464	SUPERMARKET_3_252	23546	SUPERMARKET_3_252	22245
SUPERMARKET_3_555	20968	SUPERMARKET_3_555	22004	SUPERMARKET_3_541	21975
SUPERMARKET_3_252	20133	SUPERMARKET_3_226	21812	SUPERMARKET_3_635	20685

item_2014	sales_2014	item_2015	sales_2015	item_2016	sales_2016
SUPERMARKET_3_090	37605	SUPERMARKET_3_586	33811	SUPERMARKET_3_090	11629
SUPERMARKET_3_586	34255	SUPERMARKET_3_090	33279	SUPERMARKET_3_586	9390
SUPERMARKET_3_607	22172	SUPERMARKET_3_120	31598	SUPERMARKET_3_120	7019
SUPERMARKET_3_252	20757	SUPERMARKET_3_252	21479	SUPERMARKET_1_096	6243
SUPERMARKET_3_120	20541	SUPERMARKET_3_226	17515	SUPERMARKET_3_252	5993

Bottom 5:

item_2011	sales_2011	item_2012	sales_2012	item_2013	sales_2013
HOME_&_GARDEN_2_264	0	ACCESORIES_1_241	0	ACCESORIES_2_126	0
HOME_&_GARDEN_2_265	0	ACCESORIES_1_243	0	ACCESORIES_2_095	0
HOME_&_GARDEN_2_266	0	ACCESORIES_1_253	0	ACCESORIES_2_108	0
HOME_&_GARDEN_2_272	0	ACCESORIES_1_252	0	ACCESORIES_2_109	0
HOME_&_GARDEN_2_276	0	ACCESORIES_1_246	0	ACCESORIES_2_111	0

item_2014	sales_2014	item_2015	sales_2015	item_2016	sales_2016
SUPERMARKET_3_466	0	SUPERMARKET_3_117	9	HOME_&_GARDEN_2_210	6
ACCESORIES_1_344	0	HOME_&_GARDEN_1_183	8	HOME_&_GARDEN_2_202	5
SUPERMARKET_3_500	0	ACCESORIES_1_335	1	ACCESORIES_1_335	0
SUPERMARKET_3_353	0	ACCESORIES_1_112	0	SUPERMARKET_3_441	0
SUPERMARKET_3_366	0	HOME_&_GARDEN_2_162	0	SUPERMARKET_3_444	0

Se puede apreciar que alguno de los ítems más vendidos se mantiene a lo largo del período (SUPERMARKET_3_90, SUPERMARKET_3_586), mientras que hay algún ítem que ha bajado en popularidad (SUPERMARKET_3_555), y otros que, por el contrario, se han hecho más populares, como el SUPERMARKET_3_120.

CLUSTERING

Clustering por producto

1. Creamos nuevas variables

Seguimos trabajando sobre la tabla *sales*.

- Ventas por *id*: generamos una nueva columna llamada *id_sales* que nos agrupe el importe de las ventas por cada uno de los *id* (30.490). Para ello multiplicamos la columna *id_cnt_sale* por la columna *item_price*.

- Ventas por ítem distinguiendo entre días laborables y fines de semana: En este caso utilizaremos la tabla *calendar*, ya que contiene las fechas junto con su respectivo día de la semana.

  Utilizando la función *query* creamos dos DataFrames con fechas filtrando por la columna *weekday_int*. En el caso de los días 1 y 2, que corresponden a sábado y domingo, crearemos una lista llamada lista_fin_semana. Para el resto, es decir, del 3 al 7, crearemos una lista llamada lista_semana.

  Creamos dos nuevas columnas con las ventas por id. La primera será *fines_de_semana,* que sumará la cantidad de ítems vendidos por *id* pero solo en las fechas de fin de semana utilizando la lista que hemos creado. Le segunda será *días_semana* y hará lo propio con las fechas contenidas en la *lista_semana*.

  Tenemos las fechas por id, pero nos interesa que sea por ítem, es por ello que sumamos nuestras nuevas columnas agrupando por cada uno de los 3.049 items.

- Ventas por ciudad y año: agrupamos nuestro *dataset sales* por *region* e *item* y sumamos las ventas de cada uno de los años (2011-2016).

  Hacemos una lista de las ventas de cada año y, a su vez, un *split* de cada lista que corresponda a cada una de las 3 ciudades.

Nos quedamos con las columnas que nos interesan para hacer el *clustering*: *item, category, item_cnt_sale, item_sales, item_price, item_weekends, item_weekdays*. Hacemos un *drop_duplicates* de los ítems ya que cada fila contiene la información que necesitamos.

Ahora hacemos un *merge* con nuestras ventas por ítem en días laborables y fines de semana, aprovechando para crear dos nuevas columnas que representen el porcentaje de las ventas llamadas *%_weekend* y *%_weekday*.

Nos quedamos con estas dos nuevas columnas y eliminaremos *item_weekends* e *item_weekdays*.

Aprovechamos para crear 18 columnas nuevas que corresponden cada una a una ciudad y un año *(Boston_2011 – Boston_2016, NY_2011 – NY_2016, PHI_2011 – PHI_2016*). Utilizamos las listas que nos hemos creado con la agrupación de ventas por ciudad y año.

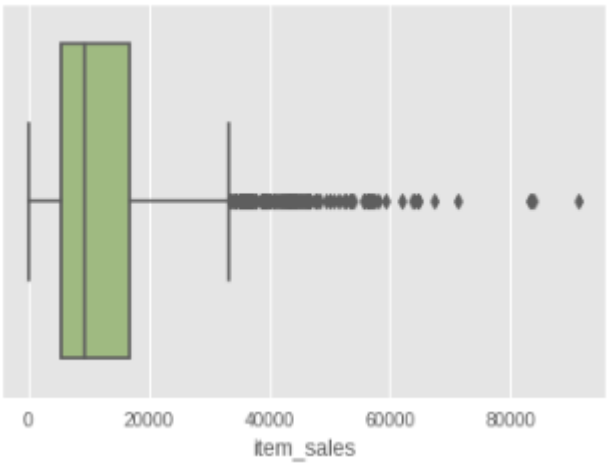
Finalmente calculamos las ventas totales del período de cada ciudad sumando las columnas con las ventas por ciudad y año. Con ello creamos 3 nuevas columnas, que son: *Sales_Boston, Sales_NY y Sales_PHI.*

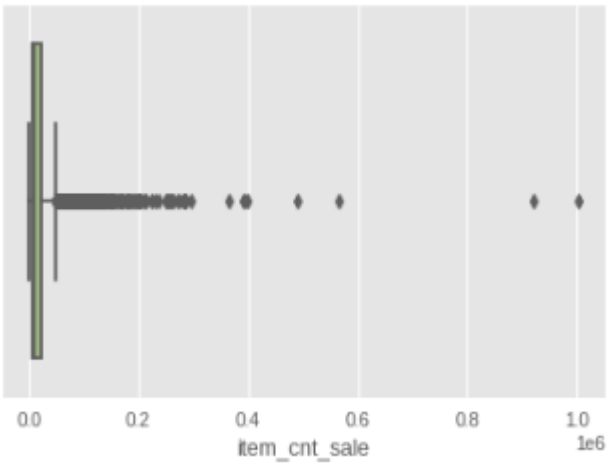
Tenemos listo nuestro *dataset*, que se compone de 3.049 filas y 27 columnas.

Decidimos hacer dos versiones de clustering con variables distintas para ver cuál queda mejor.

2. Preparamos el dataset.

- Outliers: filtramos por *item_sales* y por *item_cnt_sales*





Consideramos adecuado eliminar los ítems que cumplan dos condiciones: 1. Cantidad de ítems vendidos superiores a 250.000 unidades y 2. Importe de ventas superiores a 38.000 $. Con ello pasamos de tener 3.049 filas a 2.918.

- Encoding: en este caso solo tenemos una variable categórica, que es la columna *category*.

  Decidimos aplicar un *OneHotEncoder* ya que son solo 3 categorías y un *LabelEncoding* no sería correcto porque no hay un orden jerárquico entre los valores.

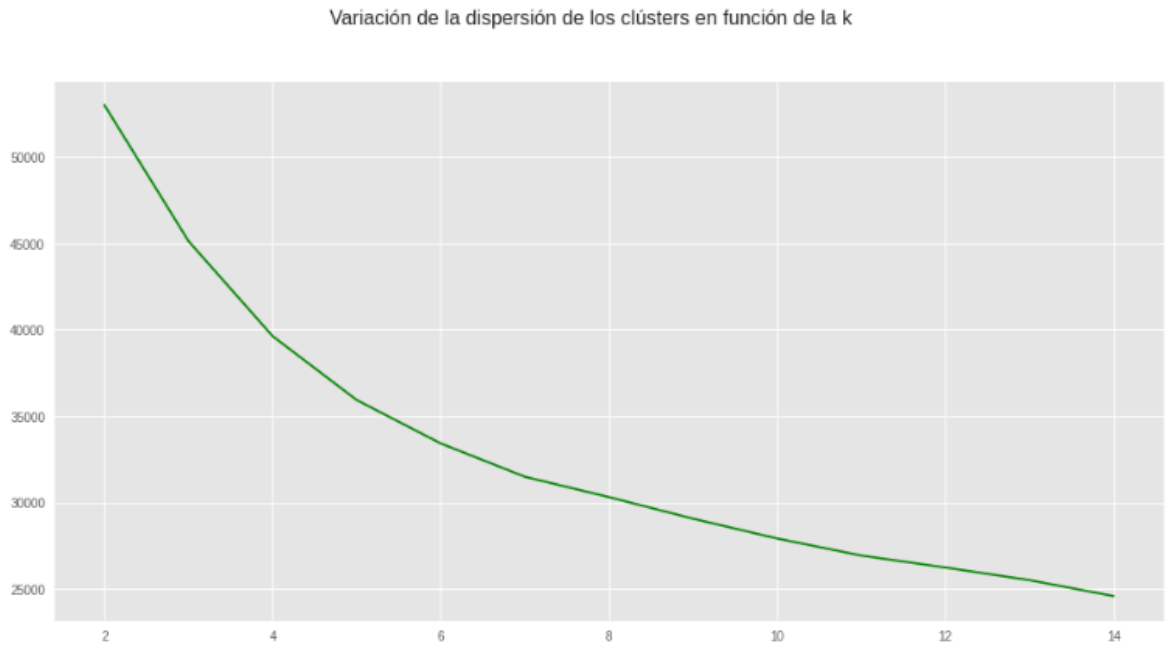
- Escalamos: Utilizando el *StandardScaler*, escalamos nuestro *dataset* sin *outliers* aplicándole un *fit_transform* y, como nos devuelve un *array*, lo pasamos a *DataFrame*.

Versión 1.

Dataset de 27 columnas (29 después de encoding).

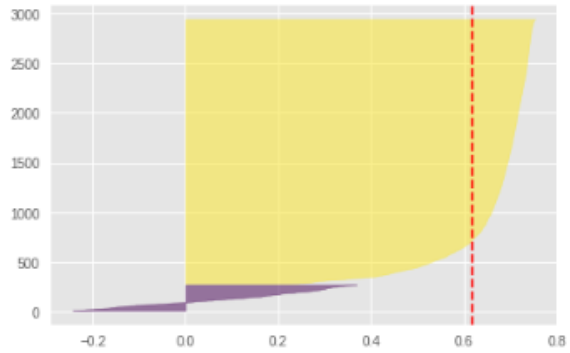
3.1. Elbow Curve

Calculamos la Elbow Curve con nuestro *dataset* escalado. Utilizamos como modelo *K_means y* como métrica *inertia,* que nos dirá la suma total de la distancia que tiene cada punto respecto a su centroide más cercano:

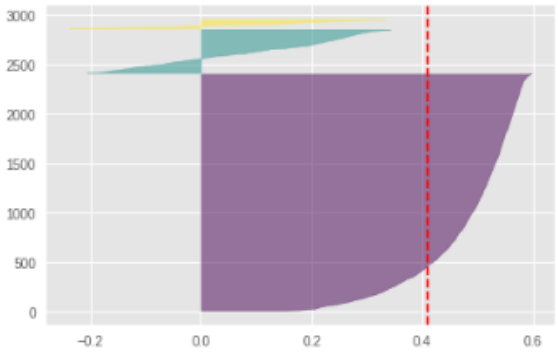


Como no tenemos un codo muy pronunciado, probamos con otra métrica para evaluar: el coeficiente de *Silhouette*. Esta técnica mide lo similar que es un punto respecto a su clúster (cohesión) y la distancia con otros clústeres (separación).

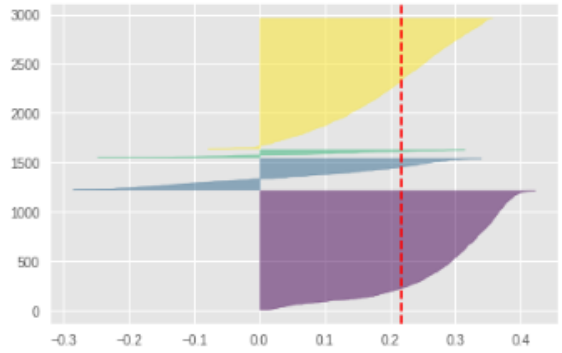
K= 2 Score= 0.61



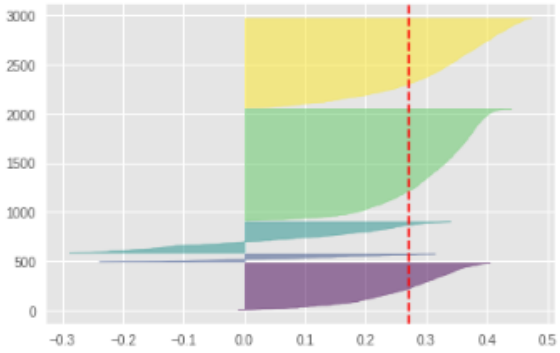
K=3 Score= 0.41



K=4 Score= 0.21



K=5 Score= 0.27



Si bien el mayor score lo tiene K = 2 en esta métrica, no parece idónea la cantidad de clúster, es por ello que nos decantamos por 5 clúster, puesto que según el gráfico todos los clústeres sobrepasan la media y también intuimos del *Elbow Curve* que es el número adecuado.

4.1. Segmentación de productos con la “K adecuada”

Una vez hemos obtenido el número de clústeres adecuado, le indicamos al modelo *Kmeans* lo que queremos (en este caso 5). Aplicamos un *fit* a nuestro *dataset* escalado.

Repetimos los pasos de preparación con nuestro *dataset* original, dejando de lado los *outliers*, porque nos interesa que todos los productos pertenezcan a un clúster. Es decir, solo hacemos encoding de la variable categórica, escalamos y transformamos el array de salida a un *DataFrame*.

Lo que nos queda es aplicar un *predict* sobre este *dataset* con *outliers* y escalado.

Añadimos a nuestro *dataset* inicial (sin escalar, con *outliers*) una nueva etiqueta con el número de clúster para cada producto (0 - 4).

Finalmente creamos una ficha con las variables que han tenido más impacto para agrupar cada clúster:

		cluster	0	1	2	3	4
Grupo Indicadores	Indicador	Estadístico					
General	Clúster	Tamaño	524.000000	1143.000000	109.000000	318.000000	955.000000
Ventas	Precio unitario	Media	4.419771	1.065197	0.033211	0.160566	3.936346
		Desviación	4.892441	1.317881	0.034638	0.150795	5.602440
		Mínimo	0.030000	0.050000	0.000000	0.010000	0.090000
		Perc. 25	1.205000	0.350000	0.010000	0.050000	0.965000
		Perc. 50	2.690000	0.650000	0.020000	0.110000	2.240000
		Perc. 75	5.842500	1.230000	0.040000	0.220000	4.945000
		Máximo	26.990000	12.790000	0.240000	1.150000	71.140000
	Ventas por producto	Media	16773.941393	8661.313578	4888.555596	8017.935377	16020.315194
		Desviación	14095.133061	6020.698213	3556.876802	6390.021868	10923.176058
		Mínimo	202.810000	804.000000	426.680000	842.420000	776.760000
		Perc. 25	6373.512500	4970.840000	2731.100000	3339.982500	7872.030000
		Perc. 50	12686.845000	7151.530000	3695.100000	6467.640000	13821.350000
		Perc. 75	23748.235000	10728.665000	6016.580000	10828.955000	20735.460000
		Máximo	80529.250000	62909.120000	22393.570000	35225.990000	88563.780000
Distribución compra	Laborables	Media	65.779084	66.033255	65.583486	65.770189	64.432021
		Desviación	2.598712	2.048258	1.721006	1.933926	1.848488
		Mínimo	56.670000	58.560000	62.310000	60.490000	57.600000
		Perc. 25	64.090000	64.815000	64.470000	64.440000	63.240000
		Perc. 50	65.800000	65.850000	65.660000	65.825000	64.360000
		Perc. 75	67.522500	67.110000	66.360000	66.880000	65.675000
		Máximo	74.660000	75.130000	75.700000	73.140000	70.320000
	Fines de semana	Media	34.220916	33.966745	34.416514	34.229811	35.567979
		Desviación	2.598712	2.048258	1.721006	1.933926	1.848488
		Mínimo	25.340000	24.870000	24.300000	26.860000	29.680000
		Perc. 25	32.477500	32.890000	33.640000	33.120000	34.325000
		Perc. 50	34.200000	34.150000	34.340000	34.175000	35.640000
		Perc. 75	35.910000	35.185000	35.530000	35.560000	36.760000
		Máximo	43.330000	41.440000	37.690000	39.510000	42.400000

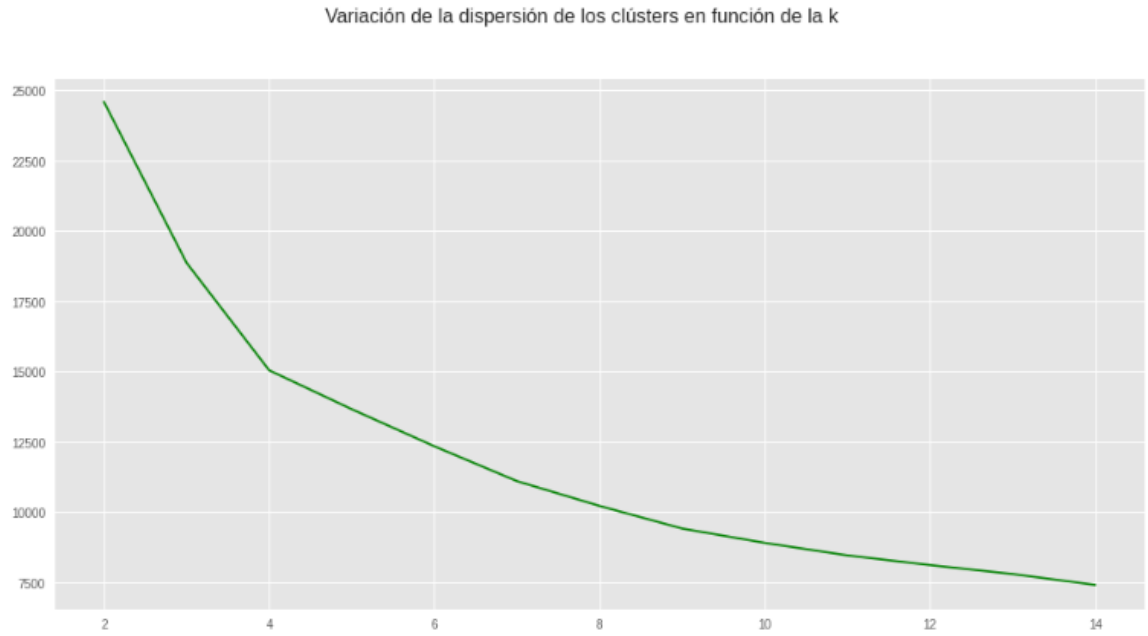
Podemos observar que los clústeres 0 y 4 son los que tienen un mayor peso en el precio unitario y en el importe generado en ventas, con la diferencia de que el clúster 4 agrupa productos que se venden más los fines de semana.

En cuanto al clúster 1 es el que mayor media de importe en ventas tiene para los días laborables, el 2 es el clúster más pequeño, contiene los productos que tienen un precio unitario bajo, pero un mínimo en compra semanales alto y el 3 agrupa los productos de menor precio unitario.

Versión 2

Dataset de 9 columnas (11 después de encoding). Hemos eliminado las 18 columnas correspondientes a ventas por cada ciudad y cada año.

3.1. Elbow Curve



En este caso sí vemos un codo marcado en 4 clústeres, por lo que prescindimos de la métrica de Silhouette.

4.2. Segmentación de productos con la “K adecuada”

Le indicamos al modelo que queremos 4 clústeres y aplicamos un *fit* a nuestro *dataset* escalado.

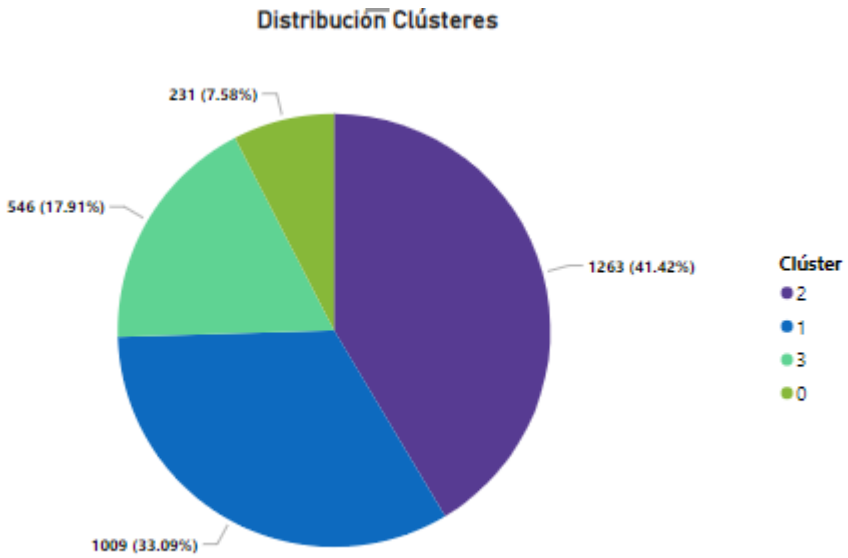
Repetimos los pasos de preparación con nuestro *dataset* original, manteniendo los outliers y aplicamos un *predict*.

Generamos y añadimos a nuestro *dataset* inicial las etiquetas con su correspondiente clúster y creamos nuestra ficha.

		cluster	0	1	2	3
Grupo Indicadores	Indicador	Estadístico				
General	Clúster	Tamaño	231.000000	1009.000000	1263.000000	546.000000
Ventas	Precio unitario	Media	0.065758	3.718692	1.008242	4.224560
		Desviación	0.069734	5.509007	1.372475	4.853612
		Mínimo	0.000000	0.040000	0.030000	0.010000
		Perc. 25	0.020000	0.840000	0.290000	1.005000
		Perc. 50	0.040000	2.000000	0.590000	2.460000
		Perc. 75	0.080000	4.790000	1.160000	5.637500
		Máximo	0.400000	71.140000	13.180000	26.990000
	Ventas por producto	Media	5957.343853	15645.934549	8712.965756	16307.717198
		Desviación	4827.765455	10842.563593	6082.717525	14072.694217
		Mínimo	426.680000	776.760000	804.000000	202.810000
		Perc. 25	3011.325000	7624.620000	4854.900000	5527.412500
		Perc. 50	4489.080000	13686.720000	7143.480000	12238.010000
		Perc. 75	7146.390000	20563.540000	10919.275000	23293.675000
		Máximo	27625.880000	88563.780000	62909.120000	80529.250000
Distribución compra		Máximo	27625.880000	88563.780000	62909.120000	80529.250000
	Laborables	Media	65.705584	64.456492	66.013919	65.842821
		Desviación	1.785628	1.844705	2.040071	2.586248
		Mínimo	60.670000	57.600000	58.560000	56.670000
		Perc. 25	64.680000	63.250000	64.800000	64.162500
		Perc. 50	65.740000	64.360000	65.850000	65.840000
		Perc. 75	66.645000	65.730000	67.100000	67.590000
		Máximo	75.700000	70.320000	75.130000	74.660000
	Fines de semana	Media	34.294416	35.543508	33.986081	34.157179
		Desviación	1.785628	1.844705	2.040071	2.586248
		Mínimo	24.300000	29.680000	24.870000	25.340000
		Perc. 25	33.355000	34.270000	32.900000	32.410000
		Perc. 50	34.260000	35.640000	34.150000	34.160000
		Perc. 75	35.320000	36.750000	35.200000	35.837500
		Máximo	39.330000	42.400000	41.440000	43.330000

En este caso en los clústeres 1 y 3 es donde tienen mayor impacto el precio unitario y el importe acumulado en ventas. Sin embargo, el 3 el que acumula más ventas tanto en días laborables como fines de semana, mientras que el 1 destaca por sus ventas únicamente en fines de semana.

El clúster 0 es el más pequeño y agrupa los productos con menor precio unitario y el clúster 2 es el más grande de todos y destaca por un mínimo en importe en ventas, así como en ventas los fines de semana bastante alto.



5. Conclusión

Parece que el modelo en la primera versión sufre de la llamada “maldición de la dimensionalidad”, ya que tiene 27 variables para 3049 filas y esto hace que la distancia entre los puntos del modelo aumente y le sea más difícil reconocer patrones para agrupar los distintos productos.

Decidimos quedarnos con nuestra segunda versión, ya que queda un poco más clara la distinción de clústeres que hace el modelo.

Clustering por tienda

Antes de empezar a trabajar con las tiendas, debemos partir de la base que tenemos muy pocos datos. Tenemos acceso solo a 10 tiendas y es muy probable que hasta no tener más datos el *clustering* no sea lo más idóneo.

1. Creamos nuevas variables

El punto de partida es nuestra tabla *sales*, como en el caso anterior.

Crearemos 6 columnas nuevas para nuestro *dataset*, que corresponderán a las ventas anuales de cada tienda y una última columna con las ventas totales por tienda para todo el período.

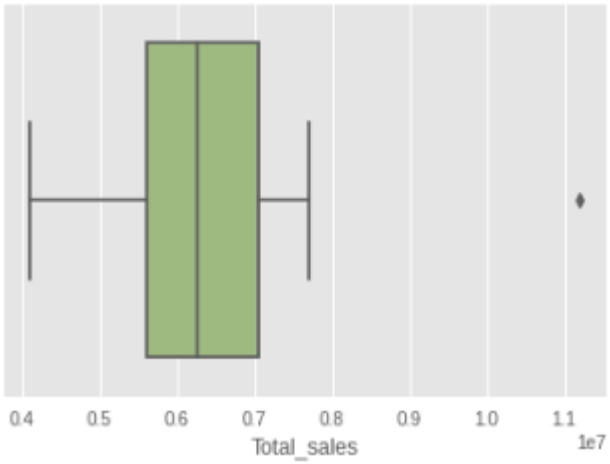
Comenzamos generando 6 listas vacías, una para cada año. Posteriormente construimos una función que agrupe por cada uno de los *store_code (group_by /get_group*) y sume la columna *item_sales_yyyy* de cada año. El resultado se guardará en cada una de las listas que creamos por año.

Aplicamos la función a todos los *store_code* por todos los años y ya podemos asignar cada una de las listas con los resultados a cada una de las nuevas columnas que queremos crear. Para la columna de ventas totales simplemente sumamos las ventas de las columnas de todos los años.

Tenemos nuestro *dataset* listo. Contiene 10 filas (una por tienda) y 8 columnas.

2. Preparamos el *dataset*

- Outliers: solo hay una tienda que sobresale y es por el nivel alto de ventas que tiene, en este caso es NYC_3 y decidimos eliminarla para entrenar.



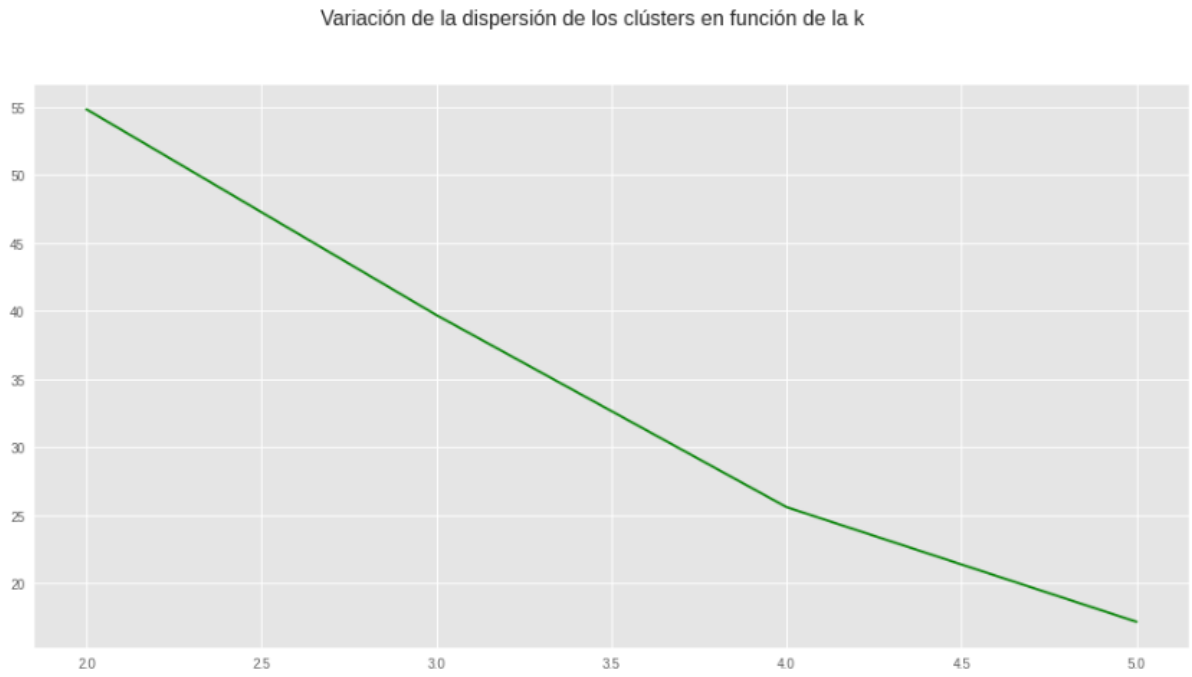
- Encoding: tenemos solo una variable categórica, que es la columna *region*.

  Como hay 3 ciudades y no tienen orden jerárquico entre sí, aplicamos *OneHotEncoding*.

- Escalamos: hacemos uso nuevamente del *StandardScaler* y escalamos nuestro *dataset* sin su *outlier*.

3. Elbow Curve

En este caso decidimos probar con una combinación entre 2 y 5 clústeres.



Vemos un codo claro en 4 clústeres.

4. Segmentación de tiendas con la “K adecuada”

Le indicamos a nuestro modelo *Kmeans* que queremos 4 clústeres. Aplicamos un *fit* a nuestro *dataset* escalado.

Repetimos los pasos de preparación con nuestro *dataset* original, incluyendo nuevamente la tienda outlier, porque nos interesa que todos los productos pertenezcan a un clúster. Es decir, solo hacemos *encoding* de la variable categórica, escalamos y transformamos el array de salida a un *DataFrame*.

Aplicamos un *predict* sobre este *dataset* con su *outlier* y escalado.

Añadimos a nuestro *dataset* inicial una nueva etiqueta con el número de clúster para cada producto (0-3).

Como es un *dataset* pequeño de 10 filas podemos ver a simple vista la distribución que ha hecho

store_code	region	store	Sales_2011	Sales_2012	Sales_2013	Sales_2014	Sales_2015	Sales_2016	Total_sales	cluster
NYC_1	New York	Greenwich_Village	1060793	1406432	1538540	1577931	1606863	507657	7698216	0
NYC_2	New York	Harlem	826083	1032484	1121532	1010213	1208421	486742	5685475	3
NYC_3	New York	Tribeca	1520827	2099706	2249633	2314556	2276753	726705	11188180	0
NYC_4	New York	Brooklin	536099	729865	824096	846176	875101	292339	4103676	2
BOS_1	Boston	South_End	799556	1064111	1105570	1105564	1151251	369240	5595292	3
BOS_2	Boston	Roxbury	1060623	1460222	1536308	1331780	1364799	460652	7214384	0
BOS_3	Boston	Back_Bay	850980	1087198	1136181	1235871	1342873	436227	6089330	3
PHI_1	Philadelphia	Midtown_Village	540021	725388	1021094	1153819	1274924	433816	5149062	1
PHI_2	Philadelphia	Yorktown	566254	1018731	1401490	1460410	1507628	589499	6544012	1
PHI_3	Philadelphia	Queen_Village	1095349	1437700	1201309	1053456	1192198	447770	6427782	3

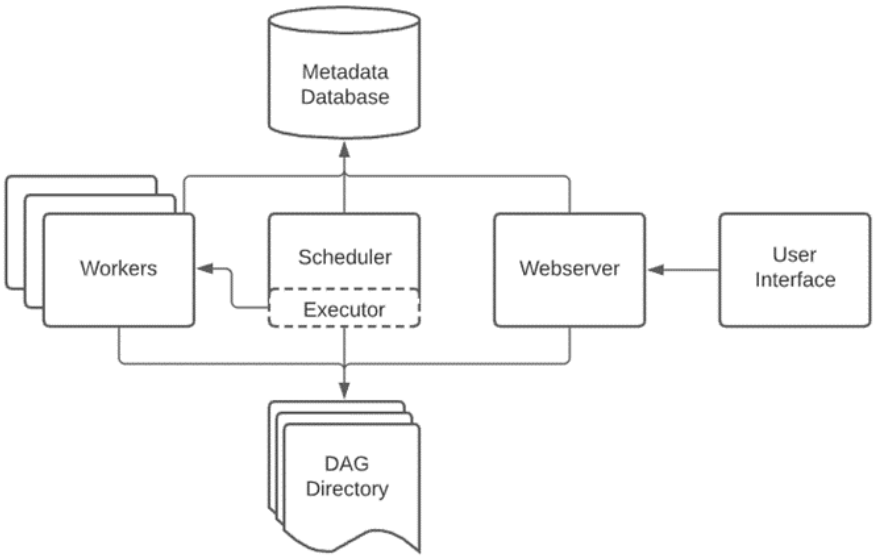
En concreto, el clúster ‘2’ solo contiene una única tienda mientras que el resto agrupan entre 2 y 3 tiendas.

5. Conclusión

La variable que ha tenido más peso es *Total_sales*, es por ello que las 3 tiendas con mayor importe en ventas pertenecen a un mismo clúster, mientras que la tienda con las ventas más bajas es la que ha quedado como única integrante de su clúster.

USE CASE: REABASTECIMIENTO DE TIENDAS (MLOps)

Debido a que estamos hablando de un modelo de forecasting, descartamos la implementación de una API ya que los datos no serán usados en tiempo real.



En este caso la propuesta estaría orientada a implementar un pipeline (DAG) con *Airflow* debido a que nos permite manejar diversas piezas de trabajo individuales denominadas *Tasks* y organizarlas teniendo en cuenta las dependencias entre ellas y el flujo de datos.

Una vez el modelo es ejecutado y se obtiene la predicción para las próximas 4 semanas. Establecemos un *EmailOperator* para que una vez esta predicción sea subida al repositorio, los distintos departamentos reciban un mail notificando que el *forecast* ha sido generado y así puedan trabajar en la toma decisiones pertinentes para optimizar la reposición de las tiendas y evitar problemas de stock (rotura de del stock, sobrestock).

```
with DAG('Forecast') as dag:
  ping = SimpleHttpOperator(enpoint='http://example.com/update/')
  email = EmailOperator(to= 'inventarioBoston@dsmarket.com', subject= 'Forecast Published')

  ping >> email
```

Con respecto a la implementación del modelo, nuestra propuesta es desplegar una prueba piloto para optimizar los procesos de inventario en la tienda de Tribeca (ya que es la tienda que más vende de toda la cadena) y desplegarlo únicamente para los productos de la categoría Supermarket, ya que minimizar el remanente de stock de esta categoría es

particularmente sensible para la empresa ya que hablamos de productos que en su mayoría son perecederos y son los productos más vendidos en todas las tiendas, por ende cualquier rotura en el stock significa pérdida monetaria.

En el caso de rotura de stock de los dos productos más vendidos (SUPERMARKET_3_090 y SUPERMARKET_3_586) en una semana, se dejaría de percibir una media de 26% de los ingresos semanales de la empresa respecto a las 10 tiendas que han sido objeto de estudio.

CONCLUSIONES

Hay una clara diferencia entre regiones respecto a las ventas, siendo Nueva York la que supera en al menos 10 millones de ítems vendidos a las otras dos ciudades.

La categoría más vendida y que engloba más productos es Supermarket, si bien son los de menos precio unitario.

Los productos más vendidos de toda la empresa coinciden con los productos más vendidos de todas las tiendas, no siendo así con los productos que menos se venden.

Consideramos que lo ideal para el tratamiento de productos sería agrupar por compra conjunta. Para ello necesitaríamos información del ticket de compra, poder ver cuáles productos suelen repetirse en un mismo ticket y así crear nuevas categorías.

En lo referente al *clustering*, también sería idóneo aplicarlo sobre clientes. Esto nos permitiría agrupar clientes similares, elaborar perfiles y cruzarlos con los productos (*collaborative filtering*) que más interés generen a clientes del mismo grupo.

Como hemos comentado anteriormente, valoraremos qué tan práctico es el *clustering* sobre tiendas en cuanto obtengamos la información sobre el resto de la red de tiendas que componen la empresa.

Si se quieren agrupar las 10 tiendas sugerimos que, de momento, se haga por ciudad.