# Trabajo Final de Master

## DS MARKET

## Capstone Project

Stefano Satireyo
Walter Leone
Joaquín Vettorazzi

nuclio digital school

# I. Introducción

## 1.1 Contexto

**Breve introducción al contexto del proyecto**

DSMarket representa una pequeña cadena de centros comerciales en los Estados Unidos que se ha sumado recientemente a una iniciativa integral de transformación digital. El cambio de nombre de la empresa es el primer paso de un plan de cinco años para revolucionar cada aspecto de sus operaciones. Michelle Huggins, la recién nombrada Directora Digital, aporta más de 15 años de experiencia liderando áreas de Marketing Digital en el sector minorista. El enfoque estratégico de Michelle es aprovechar el activo subutilizado de DSMarket: sus datos.

A pesar de contratar varios especialistas en marketing digital en el primer año, solo se ha incluido a una persona en el equipo con el rol de data scientist, Nicole. Las iniciativas iniciales de ciencia de datos priorizan la estandarización y transformación de las fuentes de datos de la empresa, la migración de procesos a la nube y la incorporación de ingenieros y arquitectos de datos.

## 1.2 Alcance del Trabajo

**Descripción de las tareas y objetivos asignados**

**Objetivos:**

En el contexto mencionado más arriba, el propósito de este trabajo es mejorar las operaciones internas y la toma de decisiones mediante la aplicación de la ciencia de datos. Para lograr esto, la empresa busca alcanzar los siguientes objetivos:

**Realizar un análisis detallado del comportamiento del negocio:**

- Realizar análisis exhaustivos para comprender las tendencias y variaciones en las ventas.
- Identificar productos menos populares y entender las diferencias en la popularidad entre ciudades y tiendas.
- Presentar resultados analíticos a la alta dirección para respaldar la toma de decisiones informada.

**Lograr la agrupación de productos y tiendas:**

- Identificar grupos de productos con comportamientos similares, facilitando la evaluación de campañas de marketing.
- Explorar la posibilidad de agrupar tiendas en función de similitudes en su desempeño o características.

**Optimización de predicciones de ventas:**

- Desarrollar enfoques más avanzados y precisos para prever las ventas de productos.
- Reducir las inexactitudes en las predicciones de ventas, que históricamente han estado basadas en métodos rudimentarios.
- Evaluar y ajustar estrategias de pronóstico a nivel de tienda y producto.

**Implementación de modelos predictivos en operaciones:**

- Aplicar modelos predictivos avanzados para mejorar la eficiencia en la reposición de productos en tiendas.
- Desarrollar una propuesta para la implementación de modelos predictivos, incluyendo detalles de productización y consideraciones para la implementación de un API.

# II. Análisis de Datos

## 2.1 Análisis Exploratorio de Datos (EDA) y Data Cleaning

**Resumen del proceso de limpieza de datos y análisis exploratorio, con énfasis en la identificación de productos populares en diferentes ciudades y tiendas.**

Objetivo:

**Análisis del comportamiento del negocio:**

- Realizar análisis exhaustivos para comprender las tendencias y variaciones en las ventas.
- Identificar productos menos populares y entender las diferencias en la popularidad entre ciudades y tiendas.
- Presentar resultados analíticos a la alta dirección para respaldar la toma de decisiones informada.

Para abordar todo el trabajo, se utilizó Visual Studio Code como nuestro entorno de desarrollo principal para escribir código en Python. La versión a de Python que utilizamos es la 3.10.9, gestionada a través de Anaconda. Se trabajó en local.

Para la primera tarea, se instalan e importan las librerías Pandas y Numpy para poder trabajar y manipular los datos de forma eficiente.

Se recibieron 3 Datasets para trabajar:

1) item_sales.csv

| Item_sales | |
|---|---|
| **id** | id de serie de venta (ítem + código de tienda) |
| **item** | producto |
| **category** | categoría de producto |
| **department** | id de departamento |
| **store** | Nombre tienda |
| **store_code** | id tienda |
| **region** | región |
| **d** | días |
| **quantity** | cantidades vendidas por día |

La tabla "item_sales" almacena información detallada sobre las ventas de productos en diversas tiendas. Cada venta está identificada por un único ID de serie, que se compone del ítem vendido y el código de la tienda. La tabla incluye datos clave, como el producto vendido, su categoría y el departamento al que pertenece. Además, proporciona información sobre la tienda, identificada por su nombre y código, así como la región a la que pertenece. El campo "d" indica el día de la venta, y la columna "quantity" detalla la cantidad de unidades vendidas diariamente. Esta estructura facilita el análisis detallado de las ventas, permitiendo la segmentación y evaluación de rendimiento por producto, tienda, región y otros criterios relevantes.

En esta tabla contamos con tres “claves de unión”: la variable ‘ítem’ y ‘store_code’, serán útiles para poder vincular la tabla de ventas con la de precios y la variable ‘d’ para vincularla con la de calendario.

2) item_prices.csv

| Item_prices | |
|---|---|
| item | ID producto |
| category | categoría de producto |
| store_code | código de la tienda |
| yearweek | fecha de periodo del precio |
| sell_price | Precio para el artículo del producto para el período en año-semana. Los precios se proporcionan por semana (promedio a lo largo de 7 días). Si no están disponibles, significa que no hubo ventas para el producto durante esa semana." |

La tabla "item_prices" contiene información detallada sobre los precios de los productos en diferentes tiendas a lo largo del tiempo. Cada fila de la tabla está asociada a un artículo específico identificado por su ID de producto, perteneciente a una determinada categoría. Además, se especifica el código de la tienda, indicando la ubicación de la tienda donde se aplican los precios. La columna "yearweek" proporciona la fecha correspondiente al período del precio, representado en el formato año-semana. El atributo "sell_price" muestra el precio promedio del artículo durante el período mencionado. Esta tabla resulta fundamental para comprender la variación de precios a lo largo del tiempo y en diferentes ubicaciones, facilitando análisis detallados sobre las estrategias de precios y su impacto en las ventas.

3) daily_calendar_with_events.csv

| Calendar | |
|---|---|
| date | día en formato y-m-d |
| weekday | día de la semana |
| weekday_int | día de la semana (numérico) |
| d | identificador de día |
| event | Si el día incluye un evento, el nombre del evento |

La tabla "daily_calendar_with_events " proporciona información detallada sobre fechas y eventos específicos. La columna "date" contiene el día en formato año-mes-día, mientras que "weekday" indica el día de la semana correspondiente. La columna "weekday_int" presenta el día de la semana en formato numérico. El atributo "d" actúa como un identificador único para cada día, tal como lo vimos en la tabla "item_sales". La columna "event" especifica el nombre del evento si el día correspondiente incluye algún evento especial. Esta tabla es esencial para contextualizar y entender las características temporales de los datos, permitiendo la identificación de patrones y correlaciones en relación con eventos específicos que podrían influir en las ventas.

**Exploración y Preprocesamiento de Datos.**

EL análisis de datos se inició con una fase de exploración y preprocesamiento, para garantizar la calidad, coherencia y preparación adecuada de los conjuntos de datos que se utilizarán para las tareas que se piden abordar. A través de Python utilizando la librería Pandas, se ejecutaron acciones específicas para abordar y entender las características únicas de cada tabla. De esta forma, se presenta a continuación un desglose detallado del procedimiento:

**1. Exploración de la Tabla *'daily_calendar_with_events.csv':***

Visualización Inicial: Se inició con la observación de las primeras filas de la tabla y una revisión general de la información que proporciona. Esto lo hicimos a través del método .head().

Exploración inicial: Aplicamos el método .info() y verificamos lo siguiente:

- La tabla contiene 1.913 filas y las 5 columnas mencionadas previamente.
- La columna "date" no está en formato DateTime, por lo que luego habrá que transformar estos datos para poder trabajar con ellos.
- La columna "week_int" efectivamente tiene datos en formato "Integer".
- Verificamos que la columna "d" tiene un día por fila.
- También verificamos que en la columna "event" hay únicamente 26 filas que no tienen un valor Nulo y 1.887 que sí. Es decir, de los 1.913 días, únicamente 26 tienen un evento especial asignado.

Tratamiento de valores Nulos en 'event': Para deshacernos de los valores nulos de la columna "event", se decidió reemplazar los valores nulos con una etiqueta: 'no event'. Esto lo hicimos con el método .fillna()

**2. Exploración de la Tabla *'item_prices.csv'*:**

Visualización inicial: Se realizó una exploración de las primeras 20 filas para obtener una comprensión inicial de la visualización de la tabla de precios.

Exploración inicial: A través del método .info() podemos observar lo siguiente:

- Contiene 6.965.706 de filas y 5 columnas.
- La tabla cuenta con 3 variables de tipo “object” y dos variables numéricas de tipo “float”
- Se observó que la columna “yearweek” tiene 243.920 valores nulos.

Manejo de valores Nulos: Los valores nulos de la columna 'yearweek' fueron completados utilizando el método de propagación hacia adelante ('ffill') para abordar situaciones donde no había datos disponibles. Esto se decidió, debido a que la tabla está ordenada por fecha (de “yearweek”), por lo tanto, la idea es que cuando se encuentre con un valor nulo, tome el valor no nulo de la fila anterior. Al investigar un poco más sobre este método, se decidió que es un buen mecanismo para lidiar con nulos en series temporales donde se busca mantener la continuidad de los datos, como en este caso.

**3. Exploración de la Tabla *‘item_sales.csv’:***

Visualización y Descriptores Estadísticos: Se procedió a examinar las primeras filas de la tabla, generar estadísticas descriptivas y evaluar la forma general del conjunto de datos.

Se observó lo siguiente:

- El Dataframe contiene 30.490 filas y 1.920 columnas.
- Cada fila representa un producto específico en una tienda y región particular. Las columnas 'd_1' a 'd_1913' indican las cantidades vendidas diariamente para cada día. Se verificó a través del método .nunique() que la tabla cuenta con 3.049 productos únicos.
- Cada producto tiene un identificador único y pertenece a una categoría y departamento específicos.

Transformación de tabla: Con el objetivo de facilitar análisis posteriores, se llevó a cabo una transposición de las columnas relacionadas con los días ('d'), convirtiendo cada día en una fila individual. Esto se realizó con la función de Pandas .melt().

Una vez hecha la transformación, se observó lo siguiente:

- El Dataframe transpuesto tiene 58.327.370 filas y 9 columnas. Para cada día, tenemos la cantidad vendida para cada ítem y para cada tienda.
- Todas las columnas son de tipo “object”, menos la cantidad vendida que es un dato de tipo “Integer”.
- Se verifican nulos en el Dataset y no se encuentran.

**Preprocesamiento y Fusión de Datos:**

Optimización: Antes de hacer merge de las tablas, es necesario optimizar o reducir lo máximo posible las tablas para no tener problemas de memoria, ya que se estará trabajando con tablas muy grandes. Por eso, es fundamental revisar los Dataset y eliminar columnas que no sean esenciales para lograr los objetivos planteados. Dicho esto, se procedió a eliminar columnas innecesarias como por ejemplo 'weekday_int' en la tabla de calendario y "id" en la tabla de ventas.

Fusión de Tablas: La tabla de ventas fue fusionada con la tabla de calendario, utilizando la columna 'd' como clave de unión. Esto se logró con el método .merge() y se obtuvo un nuevo Dataset llamado "merge_sales_date" con 58.327.370 filas y 11 columnas. Al Dataset de ventas se le agregaron las columnas "date", "weekday" y "event".
Lo fundamental del merge, es poder tener los datos de ventas por "date", es decir, armar la serie temporal. Por esto, fueron eliminadas algunas columnas que por ahora son irrelevantes para los objetivos planteados y además logramos una ocupación menor de espacio en memoria.

Para hacer el merge final del nuevo Dataset obtenido con el Dataset de precios, se debe hacer a través de las columnas "item", "store_code" y "yearweek".

Para obtener la columna "yearweek" en el nuevo Dataset "merge_sales_date", se debe manipular y transformar la columna "date" para que coincida con el formato de la tabla de precios. Para esto, es necesario realizar lo siguiente:

- Crear una columna nueva llamada "yearweek" que será una copia de la columna "date".
- Pasar dicha columna a formato Datetime.
- A partir de esta última nueva columna, crear dos nuevas columnas: "Year" para año y "week" para semana. Esto se logra con el método .dt.isocalendar().
- Se combina el año y la semana y se crea una nueva columna con el resultado. La columna se llama "yearweek_numeric".
- Por último, se le cambia el nombre a "yearweek" para poder hacer el merge con el Dataset de precios.

A continuación, se eliminan columnas que no son necesarias. Algunas de ellas únicamente fueron utilizadas para obtener el resultado final de nuestra nueva columna.

**Fusión Final:**

Se llevó a cabo exitosamente el merge de la tabla "merged_sales_date" con la tabla de precios, a través de las columnas 'item', 'store_code' y 'yearweek'.

Nuevamente, fueron eliminadas algunas columnas que no eran esenciales para lograr con los objetivos. También la columna 'sell_price' se la convirtió a tipo “float”, y se implementaron acciones para tratar valores nulos. Puntualmente, a los nulos obtenidos en “sell_price” se les asigno el valor 0.

Realizando un muy breve análisis, se puede observar lo siguiente:

- Existen 3 regiones: New York, Philadelphia y Boston
- Se pueden identificar un total de 10 tiendas
- Existen 3.049 productos que pueden pertenecer a alguna de estas 3 categorías: Supermarket, Home & Garden, o Accesories.
- Los datos van desde 2011 hasta el primer trimestre de 2016.
- El precio máximo del Dataset es USD 134 y el precio promedio es de USD 4.32.

**Exportación del Dataset final:**

Para finalizar con el proceso, se exporta y guarda la última versión del Dataset para, luego, poder importarlo en un siguiente notebook y continuar con las siguientes tareas. Los datos fueron exportados en formato .csv con la función de pandas .to_csv().

## 2.2 Servicio de Business Intelligence (BI)

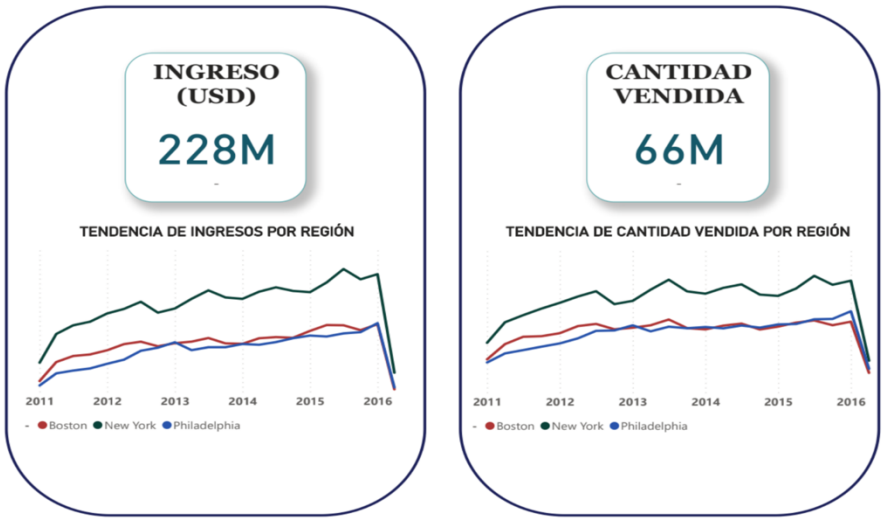
**Descripción de la implementación de un servicio de BI para seguir los principales resultados del análisis de manera regular**

Lo que se propone en esta oportunidad es utilizar la herramienta Power BI, ya que permite visualizar los datos gráficamente, creando informes, facilitando el análisis, y la creación de paneles interactivos para que los usuarios puedan utilizarlo de forma efectiva para tomar decisiones de negocio.

Para utilizar Power Bi, adaptado a este caso, se carga la versión final del Dataset obtenido en el apartado anterior con todos los datos. Una vez cargado el Dataset, se le agrega una nueva columna llamada “income” que resulta de la multiplicación de “quantity” y “sell_price”, ya que se considera un indicador clave para cualquier la toma de decisiones.

Para la tarea número 1 puntualmente, generamos paneles interactivos en los que se puede visualizar la siguiente información (ver paneles en power BI):

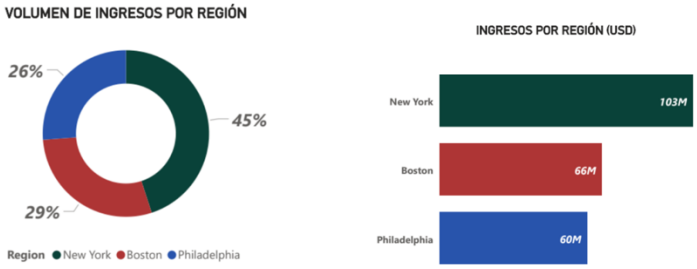
1) En el primer panel interactivo, podemos ver dos indicadores clave que cualquier Gerente general, Gerente comercial, Responsable Regional o de tienda, necesita ver para comprender y analizar el status del negocio. El primer indicador se trata de ver los ingresos generados y el segundo se trata de la cantidad de unidades vendidas. Esto se puede ver en formato tarjeta (número en dólares o unidades) o ver la tendencia en el tiempo. Este dashboard se puede filtrar por fecha (año, trimestre, mes) y también por cualquiera de las 10 tiendas.

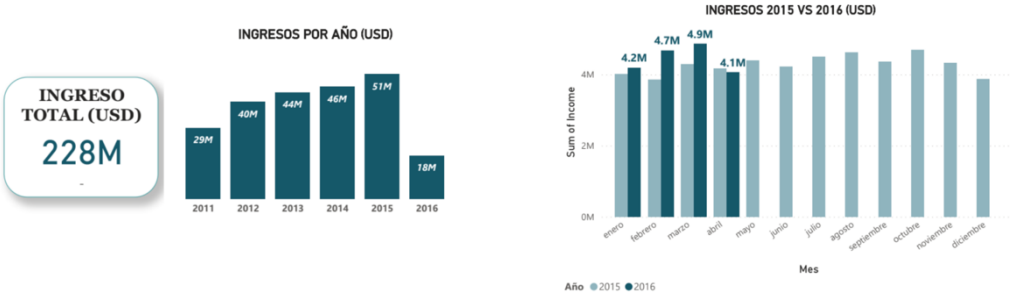



2) Información sobre los ingresos generados. En el siguiente informe se pueden visualizar 4 elementos:

   - Primero, se puede observar cómo se reparte el total de ingresos, en función de las 3 regiones. De esta visualización podemos analizar cuál de las 3 regiones es la responsable del mayor porcentaje de ingresos del negocio.
   - Segundo, se puede observar la misma información, pero a través de números específicos en millones de dólares.
   - Tercero, podemos comprender cómo fue la evolución anual de los ingresos, acompañado por una tarjeta que indica cuál es el ingreso total.
   - Por último, se observa la comparativa de cada mes de 2015, vs los meses del año corriente (2016).

   Todo el dashboard, se puede filtrar por el año o por la región que seleccionemos en el gráfico.






3) En el siguiente panel, podemos analizar dos aspectos importantes del negocio: Los precios y las tiendas.

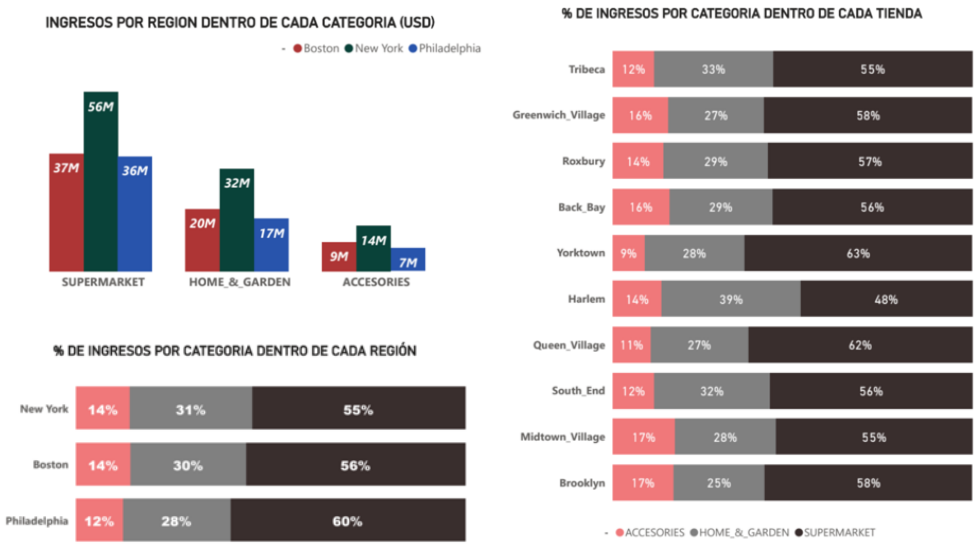
Para empezar, en la parte superior del panel, es posible comparar el precio promedio por categoría en todas las regiones. También se puede comparar precios promedios de cada categoría por región y analizar su variación. También se puede apreciar una tarjeta que indica el precio promedio, lo cual hace posible ver como varía el precio promedio por región, sin discriminar categoría. Al mismo tiempo cuenta con una tarjeta que indica el precio máximo, también pudiéndose filtrar por región y por categoría deseada.

Por otro lado, en la parte inferior del panel, se puede observar las tiendas que más unidades venden y las que más ingresos generan. De la misma forma que las visualizaciones de precio, es posible filtrarlas por región o por categoría. De esta forma y a modo de ejemplo, sería posible obtener rápidamente, cual es la tienda que más vende artículos de “Supermarket” en “New York”.


4) En el cuarto panel, es posible analizar aspectos sobre los productos y la categoría a la que pertenecen. A la izquierda del dashboard se puede observar

un gráfico de torta que representa la totalidad de ingresos, y el mismo está segmentado por cómo se distribuyen dichos ingresos por categoría. Con este gráfico, de manera simple, se puede identificar cuál es la categoría que más ingresos genera.
A su derecha, se encuentran dos tarjetas que indican cuáles son los productos más y menos vendidos. Estas tarjetas se pueden segmentar por categoría, si quisiéramos ver los productos más y menos vendidos de determinada categoría. También es posible filtrarlo por la tienda que se seleccione en el dashboard, o bien, por tienda y por categoría a la vez. Si, por ejemplo, un responsable de tienda quiere saber, dentro de su tienda, cuáles son los ítems más vendidos dentro de determinada categoría, con este dashboard podrá verlo rápidamente.


5) En el último dashboard de análisis de negocio, se pueden observar tres gráficos comparativos. El primero, indica cómo están repartidos los ingresos por región, pero dentro de cada categoría. Debajo de éste, se encuentra un gráfico que muestra lo contrario: dentro de cada región, como está repartido el ingreso total en función de la categoría. Por último, sobre el lado derecho del dashboard, se encuentra una visualización que compara como se reparte la totalidad de ingresos de cada tienda, en función de la categoría.



## III. Agrupamiento (Clustering)

**Resumen del enfoque utilizado para identificar grupos de productos similares.**

A continuación, se comentará el proceso que se llevó adelante en el notebook para realizar el clustering por producto:

1) **Instalación e importación de librerías y carga del CSV.**

En la primera sección del notebook se realizan la instalación e importación librerías necesarias. Dentro de las librerías que importamos están: Numpy, Pandas, Matplotlib, Sklearn, Time. Dentro de la librería Sklearn, importamos clases y funciones específicas que se utilizarán a lo largo del notebook.

Además, se carga el Dataset obtenido en el notebook anterior a través de la función de Pandas .read_csv().

2) **Análisis Exploratorio de Datos (EDA):**

Se realiza un rápido y sencillo análisis exploratorio de datos para entender la estructura y características del conjunto de datos. Esto se realiza a través de una función creada para proporcionar información sobre el DataFrame. También se visualizan las últimas filas y se comprueba la inexistencia de nulos.

3) **Creación de variables:**

La sección de creación de variables en un proceso de clustering es una de las más importantes. En dicha sección, se generan nuevas variables para enriquecer el Dataset y, en consecuencia, aumentar la precisión de la segmentación y mejorar la calidad de los clusters.

Para crear y agregar nuevas variables, se sigue de forma iterativa la siguiente secuencia de pasos:

a) Se parte del Dataset agrupado por ítem. Es decir, una fila por “item”. Esto se realiza con el método groupby() y se le aplica una función de agregación correspondiente (nueva variable).
b) Obtenemos un nuevo Dataset con la nueva variable, que es el resultado del paso anterior.
c) Realizamos merge con el Dataset original (también agrupado por item).

La primera variable que agregamos al Dataset es el ingreso. Esta variable, como se explica más arriba en el documento, es el resultado de multiplicar las cantidades por el precio.

Luego, se le agregan variables relacionadas con precios, como precio máximo, precio medio y la variación del precio. También se le agregan variables que indican cuanto se

vendió en cada ciudad, pero agrupado por producto. Esto se logró, tal como los procesos anteriores, haciendo .groupby(), pero en este caso, se utilizó el método .pivott_table para convertir cada región (New York, Philadelphia y Boston) en una columna independiente, con la cantidad vendida de cada ítem. De la misma forma se procedió a crear variables de cantidades vendidas por cada uno de los años del Dataset (2011, 2012, 2013, 2014, 2015 y 2016).

Por último, se agregaron variables con cantidades vendidas agrupadas por días de semana (lunes a viernes) y fines de semana (sábado y domingo).

Una vez realizados cada una de las agrupaciones correspondientes, contamos con los siguientes Datasets, cada uno de ellos con 3.049 filas (una fila por ítem).

- df_final: Dataset agrupado por ítem, sumando los ingresos.
- df_quanti: Dataset agrupado por ítem, sumando las cantidades vendidas
- prices_agg: Dataset agrupado por ítem, con su precio máximo, su precio medio y su variación de precio.
- pivoted_citty_agg: Dataset agrupado por ítem, con sus cantidades vendidas por región.
- pivoted_ventas_por_año: Dataset agrupado por ítem, con sus cantidades vendidas por año.
- ventas_dias_laborales: Dataset agrupado por ítem, con sus cantidades vendidas los días laborales.
- ventas_fin_de_semana: Dataset agrupado por ítem, con sus cantidades vendidas por fin de semana.

### 4) Merges y preparación del DataFrame final:

A continuación, se realizan los merges necesarios para obtener un único DataFrame (“df_final”) que servirá como entrada para el modelo de clustering.

### 5) Preprocesamiento y escalado de datos:

Se define un pipeline que incluye la imputación de valores nulos usando el algoritmo KNNImputer y el escalado de variables usando RobustScaler. Una de las principales razones por las que se utilizó RobustScaler para realizar el escalado es que éste utiliza la mediana y el rango intercuartílico (y no la desviación estándar o media), lo que lo hace resistente a los valores atípicos que puedan existe en el Dataset. El DataFrame resultante después del preprocesamiento se almacena en “df_final_scaled”.

### 6) Determinación del número de clústers (Elbow Curve):

Se utiliza el método de “Elbow Curve” para determinar el número óptimo de clústeres para el algoritmo KMeans. Se visualiza que la curva se acentúa en el clúster número 4.

Se optó por emplear el algoritmo KMeans debido a su eficacia en la realización de procesos de clustering, como el que abordamos en este caso específico. No solo es un algoritmo simple y de implementación sencilla, sino que también destaca por su versatilidad, siendo aplicable a diversos tipos de datos y casos. Además, ofrece una interpretación intuitiva de los resultados, facilitando la comprensión de los patrones identificados. De la misma forma, su capacidad de escalabilidad le permite manejar conjuntos de datos extensos y con múltiples dimensiones.

¿Cómo funciona K-means?

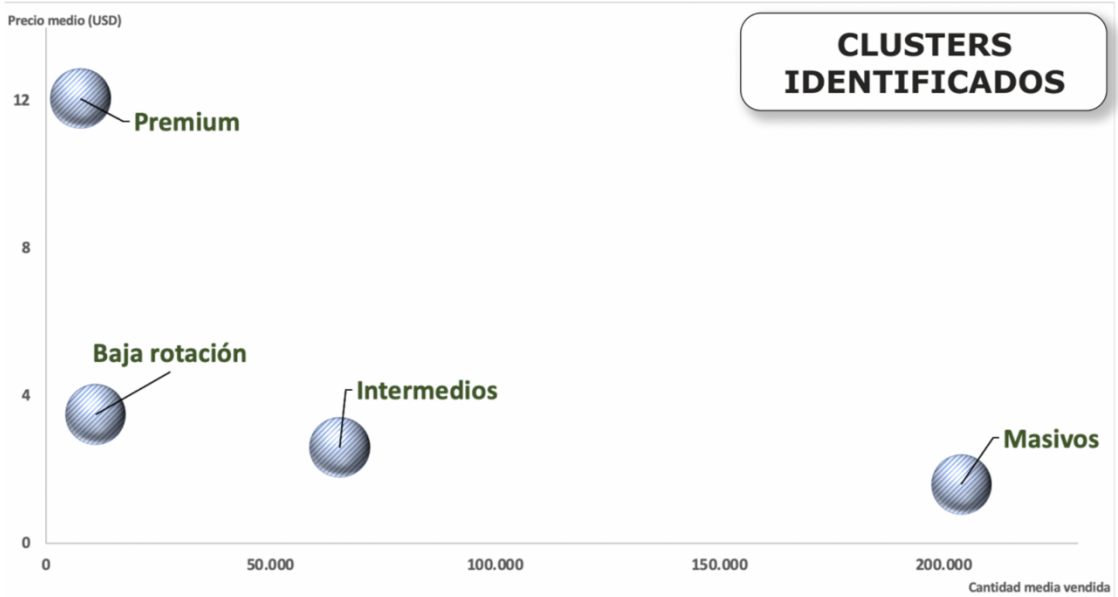
Se seleccionan aleatoriamente k puntos como centroides iniciales (en este caso 4). Cada punto de datos se asigna al clúster cuyo centroide es el más cercano utilizando la distancia euclidiana. Los centroides se actualizan recalculándolos a partir de la media de las observaciones asignadas a cada clúster. Este ciclo de asignación y actualización se repite hasta que los centroides dejan de cambiar significativamente o se alcanza un número predeterminado de iteraciones, marcando la convergencia del algoritmo.

7) **Segmentación de productos con el número adecuado de clústers:**

Una vez analizado el paso anterior, se define un nuevo pipeline que incluye la imputación, el escalado y el modelo KMeans con el número óptimo de clústeres, en este caso 4. Se ajusta el modelo al DataFrame final y se asignan las etiquetas de clúster a cada observación.

8) **Análisis de Resultados y Descripción de Clústeres:**

Se realiza un análisis detallado de cada clúster, describiendo las características y patrones de comportamiento de los productos (ver Power BI).



**CLUSTER 0 : PRODUCTOS INTERMEDIOS 2** - En este clúster se encuentran productos con demanda y precios moderados.

- Clúster más grande con 2315 productos, que representan el 76% del total.
- Ingresos: Este clúster tiene un ingreso promedio de aproximadamente USD 44.431, lo que sugiere ingresos bajo en comparación a los productos de los demás clusters.
- Precios: Este clúster tiene precios moderados (no es el que tiene precios más bajos) por lo que es coherente destacar que los bajos ingresos, se deben a la baja cantidad vendida. El precio promedio de los productos en este clúster es de USD 3.5.
- Cantidad vendida: La cantidad promedio de productos vendidos en este cluster es de 11.021 unidades. Esto indica una demanda baja en relación a los demás clusters.
- Ingresos por ciudad y año: Aunque los ingresos varían de ciudad en ciudad y año a año, no se observan fluctuaciones significativas. Los productos en este clúster mantienen una demanda estable en cada región.
- La cantidad vendida aumenta año a año pero a un ritmo similar que al total de ítems. No hay picos ni bajos considerables.

**CLUSTER 1: PRODUCTOS INTERMEDIOS 1** - Este clúster agrupa productos de bajo valor con una demanda relativamente alta. Contiene productos que tienen precios bajos, logrando una demanda (cantidad vendida) fuerte, lo que se refleja en sus altos ingresos en diferentes ciudades.

- 340 productos, que constituyen el 11% del total.
- Ingresos - Este clúster destaca por tener un ingreso promedio de USD 180.057, siendo en este caso, la cantidad vendida el factor principal de este elevado número.
- Cantidad Vendida: A pesar de los precios bajos, estos productos tienen una cantidad promedio vendida de 65.436 unidades, lo que hace que sea el segundo clúster con más ventas.
- Precio Promedio: El precio promedio en este clúster es de USD 2.69, lo que sugiere, que es el clúster que ocupa el segundo lugar en precios bajos, y lo podemos categorizar como productos económicos.
- Ingresos por ciudad y año: La distribución de ingresos también muestra que estos productos se venden bien en diferentes ciudades y períodos, lo que respalda la afirmación de una relativamente demanda alta. Año a año, este grupo de productos, considerando la media vendida, fluctuó bastante. Esto lo revertió el último año cerrado (2015), alcanzando un buen número nuevamente.

**CLUSTER 2: PROUCTOS PREMIUM -** Los productos en este clúster se caracterizan por ser exclusivos y tener los precios más altos. Aunque se venden en cantidades muy bajas, generan ingresos considerables.

- 329 productos, que representan el 10% del total.
- Ingresos - Este clúster se caracteriza por tener un ingreso promedio de USD 113.223, siendo el segundo clúster de ingreso medio más bajo.
- Cantidad Vendida - La cantidad promedio vendida es muy baja (7.637 unidades), lo que puede ser consecuencia de los precios elevados de los productos, y por ende, una inversión alta para el cliente.
- Precio Promedio - El precio promedio en este clúster es el más alto, de USD 12.07, lo que confirma que son productos exclusivos. Son productos que apuntan a un segmento de clientes específicos, con un poder adquisitivo mayor.
- Ingresos por ciudad y año - A pesar de que la cantidad vendida es baja, año a año esta categoría de productos va en aumento. En las regiones, sigue el patrón de venta que el total de los productos.

**CLUSTER 3: PRODUCTOS MASIVOS -** En este clúster se encuentran productos con los precios más bajos, pero con una demanda significativamente alta comparado con los demás.

- Clúster más chico (61 productos, 3% del total)
- Ingresos - A pesar de tener los precios más bajos, este clúster genera un ingreso promedio de USD 377.872, siendo relativamente alto, lo que se debe a gran demanda, es decir, la cantidad vendida.
- Cantidad vendida La cantidad promedio vendida es de 203.943 unidades. Los precios económicos de estos ítems se traducen en productos que se venden de forma masiva.
- Precio promedio - El precio promedio en este clúster es de aproximadamente USD 1.6 (un 40.74% más barato que el clúster que le sigue en términos de precios bajos)
- Ingresos por ciudad y año - Este clúster tiene los productos más vendidos en cada una de las regiones y en cada uno de los años.

## IV. Pronóstico de Ventas

**Descripción del enfoque propuesto para predecir las ventas a nivel de tienda y producto.**

1. **Importación de librerías:** Se instalan e importan las librerías necesarias, incluyendo pandas, numpy, matplotlib, scikit-learn, statsmodels, y Prophet.

2. **Carga de datos y preprocesamiento:** Se carga el Dataset que exportamos en la tarea número 1 con toda la información necesaria de ventas. La idea es poder realizar las predicciones de forma semanal, por lo tanto, el primer paso es agrupar el Dataset por semana.

3. **Predicción para fechas futuras:** Se genera un conjunto de predicciones para ventas futuras utilizando el modelo Prophet. Se selecciona aleatoriamente un

subconjunto de 500 productos y se itera sobre todas las combinaciones posibles de tiendas y productos. Se generan 9 semanas de predicciones y se almacenan en un Dataframe llamado “predicciones”. De las 9 semanas de predicciones, 5 semanas corresponden a las últimas 5 semanas del Dataset y 4 semanas son a futuro. De esta forma, las 5 semanas que coinciden con el Dataset se pueden tomar como forma de validación y servirán para realizar la medición del error medio.

El algoritmo Prophet es interesante, ya que funciona descomponiendo la serie temporal en 4 elementos:

- la tendencia a corto y largo plazo, es decir, de forma semanal y anual. Ayuda a capturar tendencias y fluctuaciones estacionales.
- Días festivos.
- Se le pueden incorporar efectos estacionales personalizados, por nuestro conocimiento del negocio.
- El ruido. Tiene un buen manejo de irregularidades en los datos.

4. **Manipulación y guardado de predicciones:** Se filtran las columnas relevantes y se almacenan las predicciones en un archivo CSV para no perder la información obtenida. Se le crea identificadores únicos a ambos Datasets (al original agrupado en semanas y al de predicciones). El identificador creado es una nueva columna, la cuál es el resultado de combinar el nombre del ítem, el nombre de la tienda y la fecha. El identificador único servirá para luego realizar el merge de las predicciones con el Dataset original.

5. **Validación y cálculo de RMSE:** Se realiza una validación comparando las predicciones con los datos reales disponibles. Se calcula el error cuadrático medio de la raíz (RMSE) como métrica de rendimiento.

6. **Merge de Dataset de predicciones con Dataset original:** Se crea un nuevo Dataset que combina las predicciones con los datos originales y de validación.

7. **Visualización de Resultados:** Se elige un artículo y una tienda específica, y se crea un gráfico de línea que compara las predicciones con los datos reales para un rango de fechas específico. Este paso se realiza simplemente para tener un panorama inicial de las predicciones.

8. **Guardado del Conjunto de Datos Final:** Se guarda el Dataset final con las predicciones en un archivo CSV.

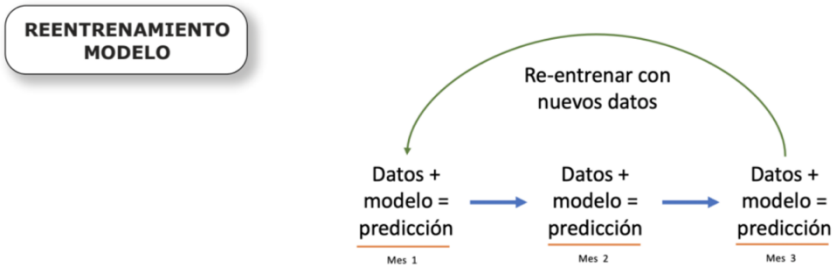
## V. Caso de Uso: Reabastecimiento de Tiendas

**Redacción de una propuesta para la aplicación de modelos predictivos al caso de abastecimiento de tiendas.**

Una vez que existe un modelo consolidado para realizar las predicciones, se propone realizar dichas predicciones al final de cada mes para que los comerciales o

responsables de tiendas puedan consultarlo a lo largo del próximo mes. Este proceso sigue una naturaleza asíncrona, en el cual las predicciones se generan y la información resultante se almacena en una tabla. Los responsables de las tiendas pueden acceder a esta información cuando lo necesiten. En este caso, no se requiere una API RES, ya que no se trata de un proceso síncrono destinado a consultas en tiempo real.

Al ser Retail una industria en la que hay drift (variación de la demanda a lo largo del tiempo) es fundamental reentrenar el modelo con los nuevos datos. El reentrenamiento puede llevarse a cabo de forma trimestral (como se muestra en la imagen), mensual o semanalmente. Siempre es conveniente reentrenar en el menor tiempo posible para que el modelo esté lo más “actualizado” posible y las predicciones sean más acertadas. Sin embargo, el reentrenamiento a baja frecuencia puede ser algo costoso a nivel empresa.

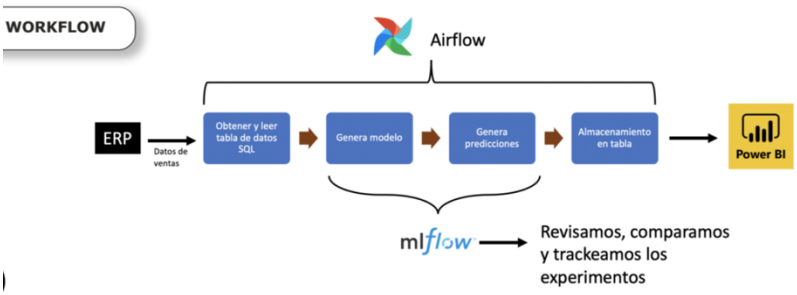



Para incorporar este proceso en el funcionamiento diario de la organización, el flujo de trabajo propuesto se inicia extrayendo los datos del ERP de la empresa, donde los comerciales o equipo de ventas, registran y almacenan la información de ventas. Posteriormente, se propone utilizar la herramienta Airflow para crear un DAG que ejecute las siguientes tareas automatizadas:

1. Obtener y leer la tabla de datos SQL.
2. Generar el modelo.
3. Realizar predicciones.
4. Almacenar los datos de predicción en una tabla.

En los pasos 2 y 3, se sugiere utilizar la herramienta MLflow, ya que facilita la experimentación ágil, el seguimiento y la comparación de modelos, asegurando la elección del mejor modelo y por ende las predicciones más precisas.

Para concluir el proceso, se propone cargar la tabla con las predicciones en Power BI. Esto permitirá que cada vendedor pueda visualizar mensualmente la estimación de ventas para su tienda, sobre cualquier producto.


WORKFLOW
Airflow
ERP
Datos de ventas
Obtener y leer tabla de datos SQL
Genera modelo
Genera predicciones
Almacenamiento en tabla
Power BI
mlflow
Revisamos, comparamos y trackeamos los experimentos