Proyecto 3

Procesamiento de Grandes Volúmenes de Datos

Estudiantes

Juan Esteban Cardona
Juan José Hoyos
Jeffrey García
Josue Peña

Pontificia Universidad Javeriana Cali

Diciembre 2020

Índice

Introducción

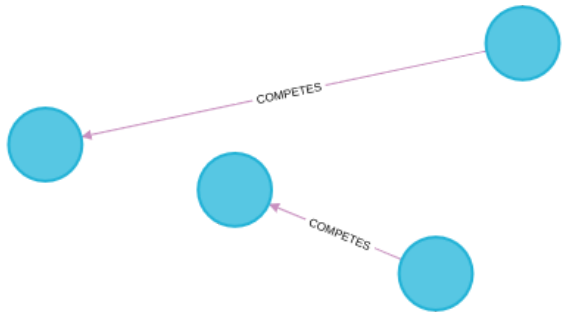
En este proyecto se muestran los resultados de haber considerado el problema desde una perspectiva de análisis topológico del grafo generado con Neo4j. A diferencia de los proyectos anteriores, en este se optó por definir como objetivo de predicción la cantidad de clicks que realiza un usuario a la hora de comprar un artículo de una página web de venta de artículos de ropa y accesorios. Por este motivo, se deja de lado la tarea de clasificación y se escoge realizar la tarea de regresión para el modelo de aprendizaje automático a implementar.

Los atributos considerados en el dataset original fueron: *month, day, order, country, page_1, colour, location, model_photography, price, price_2, page y clicks*, la cantidad de registros del dataset normal luego del filtro es de 154.829 y pesa 4.37MB. El dataset con las medidas topológicas del grafo tiene la misma cantidad de registros y pesa 18MB.

El modelo de aprendizaje automático utilizado fue RandomForestRegressor.

Descripción de la base de datos Neo4j

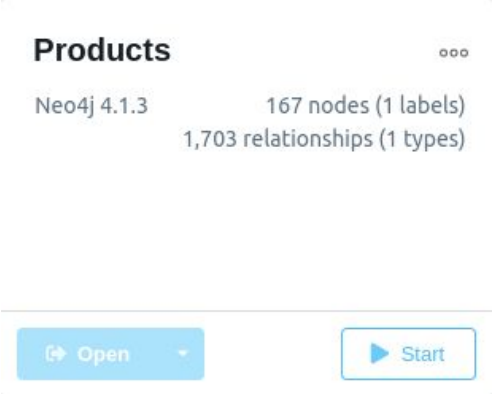
El grafo que se decidió implementar consiste en relacionar pares de productos que reciben clics en la tienda. Cada sesión de usuario en la página de compras cuenta con un número de identificación. Este número es un atributo o columna del archivo de base de datos .csv original.



Con esa información, se define el grafo de la siguiente forma: Cada nodo corresponde a un producto que aparece registrado en la base de datos (según su ID), y este cuenta con los atributos de nombre o ID, categoría, color, y precio. Si dos productos aparecen registrados en la misma sesión, esto quiere decir que el usuario hizo click en ambos, y por lo tanto estos productos compiten entre sí en la tienda.

Se establece entonces la relación de “COMPETES”, la cual es bidireccional. Esta relación cuenta con un atributo de peso, el cual indica la cantidad de instancias en las cuales los dos productos aparecen en una misma sesión de usuario.

Posteriormente, se decide incluir solamente las relaciones que cuenten con un peso (o número de incidencias) mayor o igual a 150. Al realizar este filtro quedan 167 nodos de productos (originalmente habían 217), y 1703 relaciones, como se ve aprecia en la siguiente imagen.



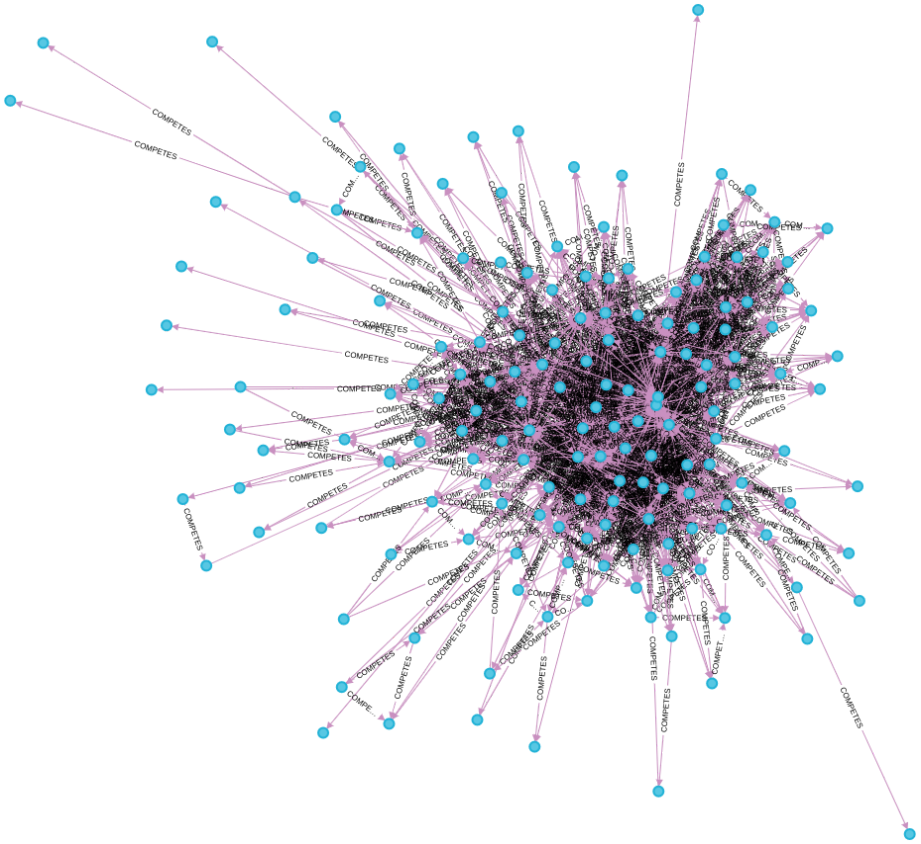
El cálculo de la cantidad de clicks se realiza de forma posterior y es externo a la estructura del grafo dentro de Neo4j.

Para generar el grafo se diseñó el script “***generate_neo4j_graph.py***”, el cual toma la base de datos .csv y a partir de esta genera otro .csv en un formato fácil de procesar mediante Cypher. A continuación se muestran las sentencias de Cypher usadas para generar el grafo y una representación gráfica del mismo.

```
LOAD CSV WITH HEADERS FROM
'http://localhost:11001/project-76d4894d-6d21-418f-953d-a0d3aa0b1d
71/dataset_graph.csv' AS line FIELDTERMINATOR ';'

MERGE (a:Product {id:line.id1, category:toInteger(line.category1),
colour:toInteger(line.colour1),      photo:toInteger(line.photo1),
price:toInteger(line.price1)})

MERGE (b:Product {id:line.id2, category:toInteger(line.category2),
colour:toInteger(line.colour2),      photo:toInteger(line.photo2),
price:toInteger(line.price2)})

MERGE (a)-[r:COMPETES {w: toInteger(line.w)}]-(b)
```

```
-- Eliminar las relaciones menores a 150
MATCH (n)-[r:COMPETES]-()
WHERE r.w < 150
DELETE r

-- Eliminar los nodos que quedan sin ningún tipo de conexión
MATCH (a) WHERE NOT (a)-[:COMPETES]-()
DELETE a
```

Representación gráfica del grafo



Después de generar las medidas topológicas especificadas en la siguiente sesión, se realiza el siguiente query para guardar la tabla como un .csv y utilizar el script "***generate_filtered_dataset.py***" para generar las dos bases de datos (con y sin medidas) que serán usadas para el Machine Learning.

```
-- Query para generar el csv de métricas
MATCH (a:Product)
RETURN a.id as id, a.louvain_community as louvain_comm, a.clustering_coeff as clustering_coeff, a.betweenness as betweenness, a.closeness as closeness, a.x as x, a.y as y, a.z as z
ORDER BY a.id ASC
```

Medidas topológicas usadas

Se tomó un conjunto inicial de medidas topológicas para el grafo construido usando la librería de Graph Data Science de Neo4j. A este conjunto se fueron agregando y eliminando métricas con base en la influencia que ejercían estas en el desempeño del modelo de aprendizaje automático y la similitud (correlación) que existían entre varias de las mismas.

A continuación se describen cada una de las medidas topológicas que se consideraron para realizar el proyecto.

Betweenness: Es una forma de detectar la cantidad de influencia que tiene un nodo sobre el flujo de información en un grafo. A menudo se utiliza para encontrar nodos que sirvan de puente entre una parte de un grafo a otra.

Closeness: La centralidad de proximidad es una forma de detectar nodos que pueden difundir información de manera muy eficiente a través de un grafo.

La centralidad de proximidad de un nodo mide su distancia media (distancia inversa) a todos los demás nodos. Los nodos con una puntuación de proximidad alta tienen las distancias más cortas a todos los demás nodos.

PageRank: El algoritmo PageRank mide la importancia de cada nodo dentro del grafo, en función del número de relaciones entrantes y la importancia de los nodos de origen correspondientes. En términos generales, la suposición subyacente es que una página es tan importante como las páginas que la enlazan.

Degree: La centralidad de grados mide el número de relaciones entrantes y salientes de un nodo. El algoritmo de Degree Centrality de la GDS library puede ayudarnos a encontrar nodos populares en un grafo.

Louvain Community: El método Louvain es un algoritmo para detectar comunidades en grandes redes. Maximiza una puntuación de modularidad para cada comunidad, donde la modularidad cuantifica la calidad de una asignación de nodos a las comunidades. Esto significa evaluar cuánto más densamente conectados están los nodos dentro de una comunidad, en comparación con qué tan conectados estarían en una red aleatoria.

El algoritmo de Louvain es un algoritmo de agrupamiento jerárquico, que fusiona comunidades de forma recursiva en un solo nodo y ejecuta la agrupación de modularidad en los gráficos condensados.

Local Clustering Coefficient: Calcula el coeficiente de agrupación local para cada nodo del grafo. El coeficiente de agrupamiento local Cn de un nodo *n* describe la probabilidad de que los vecinos de n también estén conectados. Para calcular Cn usamos el número de triángulos que un nodo es parte de Tn y el grado del nodo dn.

Node2Vec: Node2Vec es un algoritmo de incrustación de nodos que calcula una representación vectorial de un nodo basándose en recorridos aleatorios en el grafo. El vecindario se muestrea a través de caminatas al azar. Utilizando una serie de muestras de

vecindad aleatorias, el algoritmo entrena una sola red neuronal de capa oculta. La red neuronal está entrenada para predecir la probabilidad de que ocurra un nodo en una caminata en función de la aparición de otro nodo.

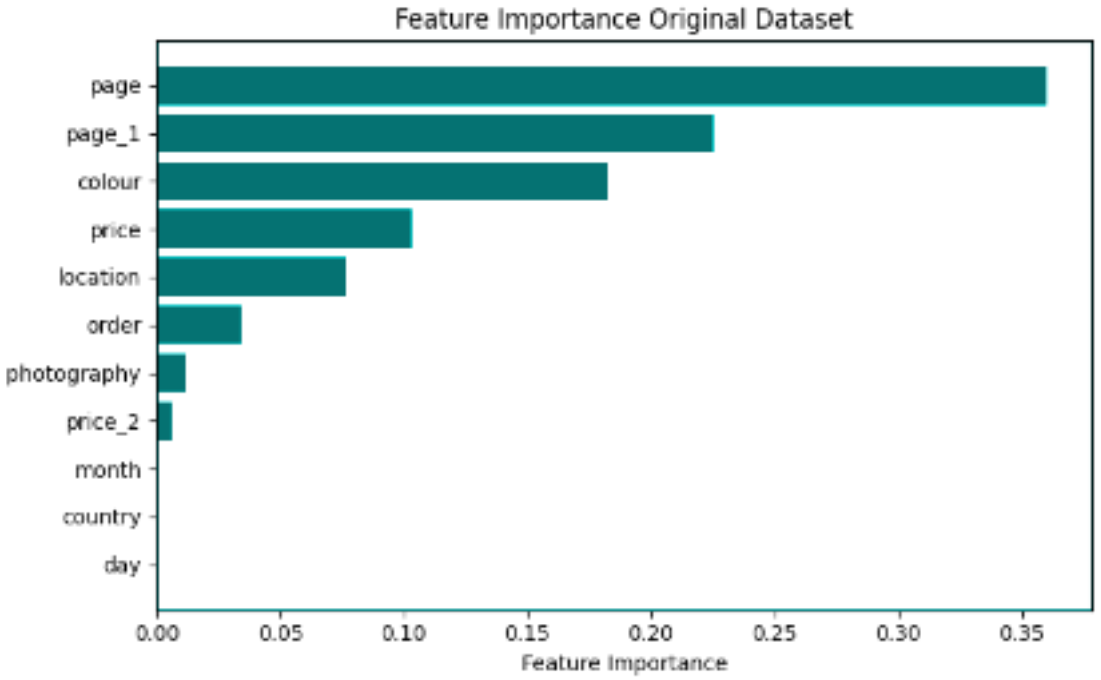
Luego de realizar varios experimentos y revisiones, las medidas topológicas que se usaron en la versión final del proyecto fueron las siguientes:

- Louvain Community
- Local Clustering Coefficient
- Betweenness
- Node2Vec de 3 dimensiones (representados mediante los nombres x, y, & z)

Feature importance

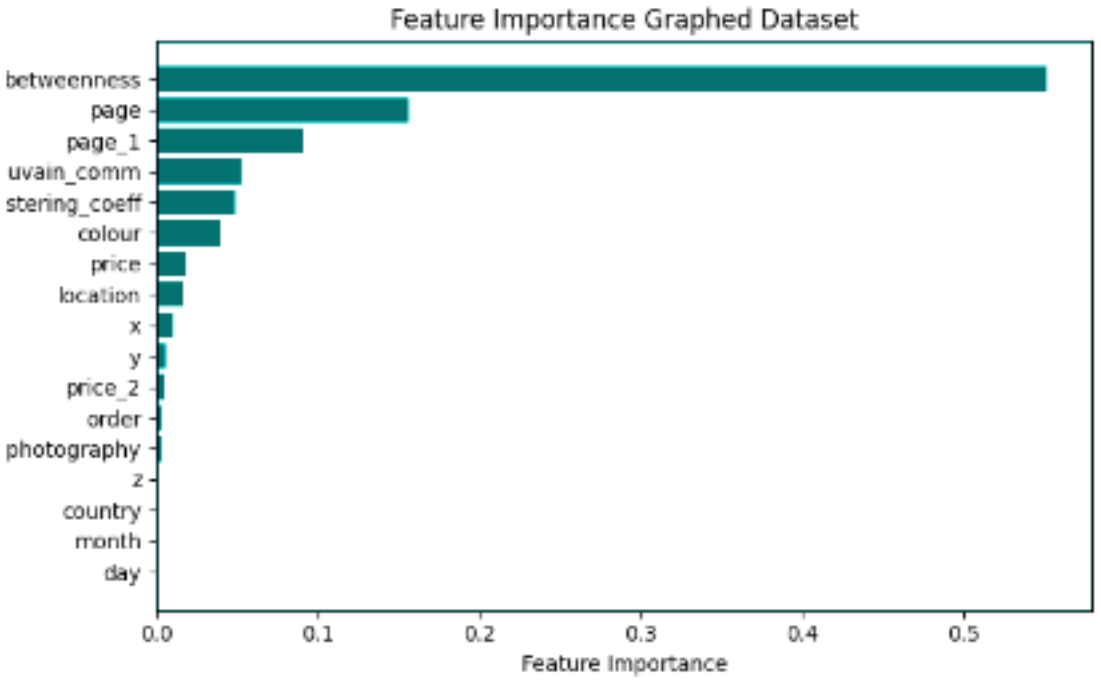
Para identificar la importancia de cada atributo en los resultados de predicción, se usó el método de clase implementado en la librería Pyspark para Machine Learning (pyspark.ml).

A continuación se puede apreciar el gráfico de Feature Importance con el conjunto de datos original (sin medidas topológicas).



El atributo que mayor influencia tiene en la predicción alcanzada es *page*, el cual indica el número de página en la cual se encuentra el producto dentro de la tienda en línea (desde 1 hasta 5), debido a que el número de clicks dados depende en gran medida de cuántas páginas haya avanzado. En segundo lugar tenemos *page_1*, el cual nos indica la categoría a la que está asociada el producto. Y en tercer lugar, *colour*, que designa el color del producto.

A continuación se encuentra el gráfico de Feature Importance con el conjunto de datos que cuenta con las medidas topológicas generadas con Neo4j.



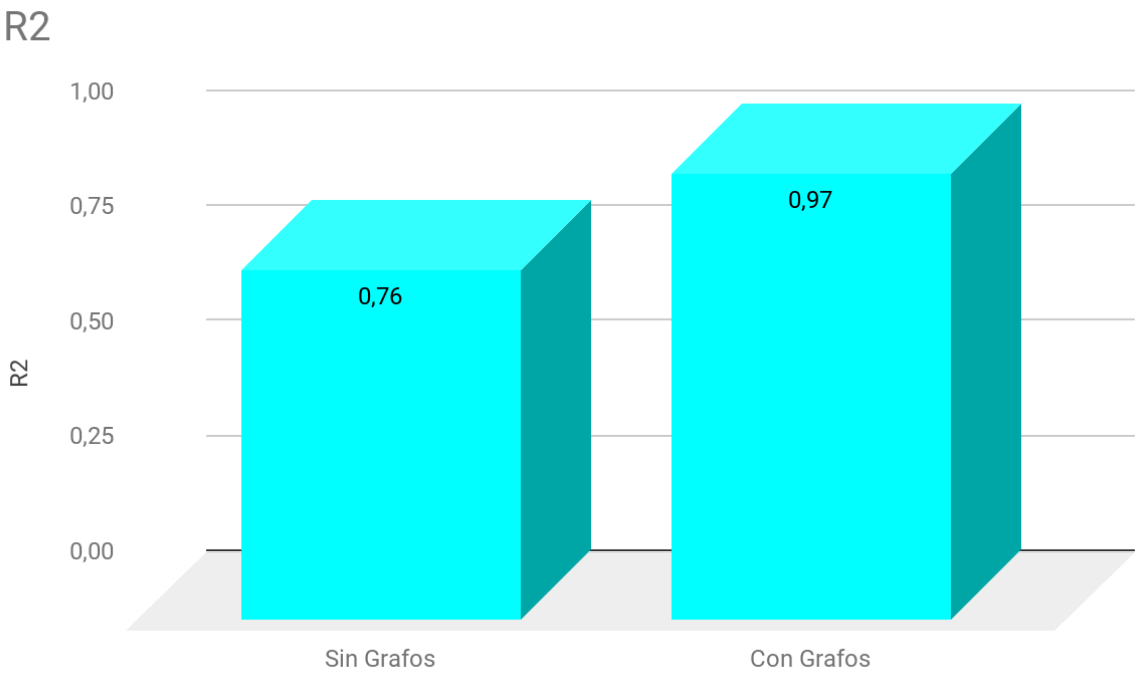
Como se observa, el atributo que mayor influencia tiene en la predicción alcanzada es betweenness. Esta medida topológica nos indica la cantidad de influencia que tiene un nodo sobre el flujo de información en un grafo.

Para calcular esta métrica se requiere hacer el cálculo del camino más corto entre todos los pares de nodos en el grafo. Nodos que pertenezcan a una gran cantidad de caminos cortos recibirán un score de betweenness más alto, mientras que los que no aparecen en muchos caminos cortos recibirán uno más bajo.

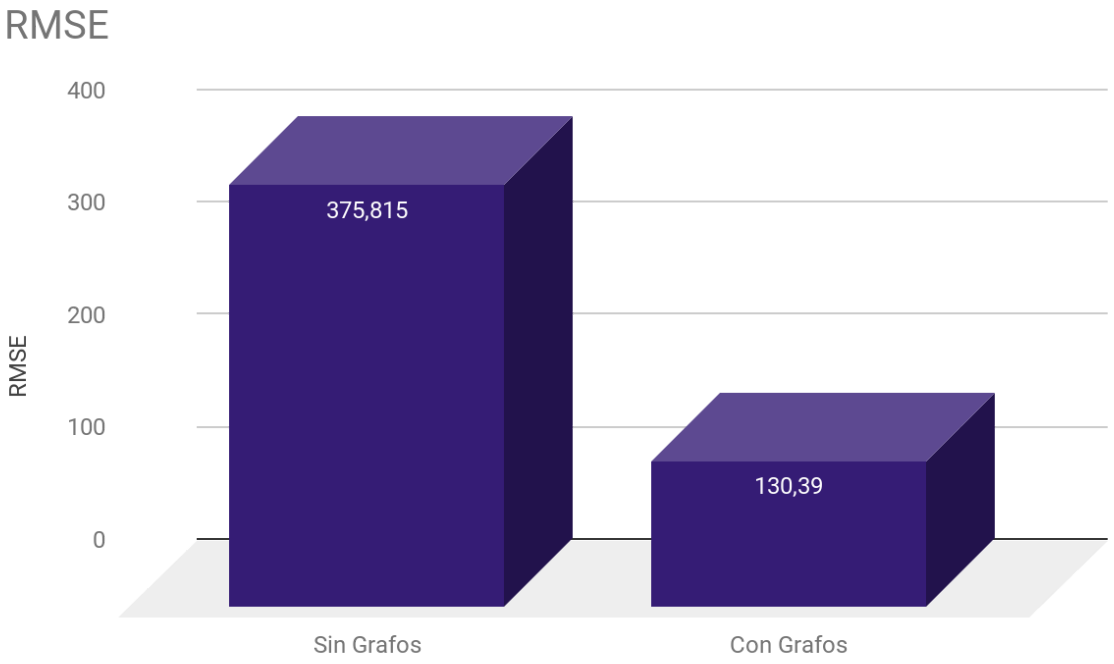
Esto es el mejor predictor para la cantidad de clicks porque el score indica si muchos usuarios hicieron click en un mismo producto durante sus sesiones particulares. Estos productos están conectados con muchos productos en el grafo, y a su vez, son requeridos para atravesar el camino más corto entre ese mismo producto y muchos otros.

Resultados de Predicción con y sin Medidas

R2 Score (coeficiente de determinación)



Error cuadrático medio



Conclusiones de los proyectos

Inicialmente, se consideró una fase exploratoria con el conjunto de datos obtenidos de la tienda online, por este motivo se decidió aplicar tanto regresión como clasificación en el proyecto 1, clasificación para estimar el tipo de foto que se debía poner en diferentes productos y regresión para predecir el precio de venta de un producto en particular. Con esta fase exploratoria se obtuvieron muy buenos resultados en ambas tareas, por lo tanto se cumplió el objetivo de esta fase que en un principio era aplicar algunas técnicas que nos permitieran establecer relaciones entre las variables de interés y del mismo modo probar que tan buena y estable era la tarea de predicción que se lograba hacer con el conjunto de datos. Cabe resaltar que este escenario era completamente estático, pues se trabajó con el conjunto de datos completo que considera 5 meses consecutivos de registro del sitio web en un mismo año.

Posteriormente, se trabajó con un escenario un poco más dinámico, en donde solo se consideró la tarea de clasificación de las que se aplicaron en el proyecto 1, aplicando streaming de datos con el fin de realizar la simulación de un entorno de producción en el cual se estuviesen recolectando los datos del sitio web mes tras mes, y sorprendentemente encontramos que aunque el tamaño de los datos era considerablemente menor se seguían logrando buenas predicciones, sin embargo, concluimos que realizar una predicción sobre si usar una foto frontal o de perfil en la publicidad de un producto, no representa un valor comercial muy interesante, en este orden de ideas un nuevo enfoque se tomó en el proyecto 3.

Finalmente, en función de aportar un modelo que genere valor comercial, se decidió enfocar la tarea de aprendizaje automático en la predicción de la cantidad de clicks que puede tener un producto dentro de la tienda, lo cual es bastante útil porque permite asociar una mayor cantidad de clicks con un mayor interés por el producto y en consecuencia una mayor probabilidad de que dicho producto sea vendido. Para lograr este propósito, se construyó además del modelo de aprendizaje un grafo que permite establecer relaciones entre productos que pueden competir entre sí debido a que son del interés del usuario (ya que este accedió a ambos productos durante una misma sesión), de dicho grafo se extrajeron medidas topológicas que nos permiten categorizar los productos de acuerdo a las relaciones establecidas. Dichas medidas pasaron a ser nuevos atributos a considerar en el modelo de aprendizaje automático, y de manera iterativa pudimos encontrar medidas realmente útiles que permitieron mejorar drásticamente la predicción y disminuir el error de la misma en comparación con lo obtenido con el conjunto de datos original.

Se concluye que es de vital importancia conocer el conjunto de datos y sus características, ya que esto permite, en primer lugar, designar un objetivo de predicción o clasificación para el conjunto de datos, lo cual es un paso fundamental en la realización de un proyecto de procesamiento de datos como los que se trabajaron. En segundo lugar, distinguir la importancia e influencia que puede tener cada una de sus variables en la tarea de predicción, ya que esto conlleva a analizar cómo influyen ciertos atributos en la tarea de

aprendizaje automático seleccionada y, por lo tanto, lleva conocer de una manera más orgánica los datos. Además, tener este conocimiento sobre el conjunto de datos permite realizar procedimientos, como el de formar el grafo, en los cuales se puede aprovechar el potencial de los datos para generar nueva información y aportar a la tarea escogida, y en el mejor de los casos contribuir a su mejoramiento.