Proyecto final: Clasificación Grafos Amazon

[ac] Camilo Andrés González Vargas, [ac] Juan David Valencia Sandoval, [ac] Paula Alejandra Velasco Cuberos, [ac] Pavel Mauricio Dussan Gutiérrez, [ac] Tatiana Roa Ahumada, [ac] Yuri Angélica Reina Guzmán, [bc] Sergio Alberto Mora Pardo

a. Estudiante de Maestría en Analítica para la Inteligencia de Negocios
b. Profesor, Departamento de Ingeniería Industrial
c. Pontificia Universidad Javeriana, Bogotá, Colombia

BUSINESS UNDERSTANDING

Background:

En el ámbito de la minería de grafos, la clasificación semisupervisada de nodos se ha consolidado como una herramienta clave para mejorar las predicciones y recomendaciones en sistemas complejos. Las redes neuronales de grafos (GNN) han demostrado ser particularmente efectivas en estas tareas, logrando resultados sobresalientes en la clasificación de nodos dentro de estructuras de grafos. El uso de divisiones fijas y el ajuste inconsistente de hiperparámetros pueden generar sesgos en los resultados y dificultar comparaciones justas entre distintas arquitecturas, lo que podría subestimar la efectividad de modelos más sencillos.

Amazon, por su parte, enfrenta el desafío de mejorar la precisión en la clasificación de productos. Con un catálogo extenso y en constante crecimiento, su sistema actual de recomendaciones, basado en patrones simples como la co-ocurrencia de compras, no logra capturar adecuadamente las relaciones profundas entre productos ni sus características detalladas. Esta limitación afecta la calidad de las recomendaciones y puede impactar negativamente en la satisfacción del cliente. Para mejorar este aspecto, un sistema más sofisticado es necesario, y las GNN ofrecen la posibilidad de modelar de manera más precisa tanto las relaciones entre productos como sus características específicas, lo que permitiría generar recomendaciones más ajustadas a las necesidades de los usuarios.

El entorno competitivo de Amazon hace que la precisión de las recomendaciones sea crucial para asegurar la lealtad del cliente y el éxito comercial. A medida que el catálogo se expande, la capacidad para generar recomendaciones personalizadas y relevantes se convierte en una ventaja estratégica. Los sistemas tradicionales, basados en patrones de compras pasadas, presentan limitaciones, mientras que un sistema basado en grafos tiene el potencial de capturar la complejidad de las interacciones entre productos, mejorando tanto la calidad como la personalización de las recomendaciones.

Los modelos de grafos son especialmente adecuados para tareas de clasificación y recomendación, ya que permiten representar de manera detallada y compleja las relaciones entre productos. Además, ofrecen una ventaja en términos de escalabilidad, facilitando su adaptación al crecimiento del catálogo sin necesidad de rediseños extensos.

Finalmente, una clasificación adecuada de productos no solo mejora la experiencia del usuario, sino que también optimiza el sistema de recomendaciones, lo cual es fundamental para personalizar la oferta en la plataforma y fomentar la fidelización del cliente, contribuyendo al crecimiento de Amazon en un mercado competitivo.

Determine Business objectives

Business goal:

BG1: Mejorar la precisión de las recomendaciones de productos en Amazon mediante la identificación de las relaciones complejas entre productos y sus características, permitiendo personalizar la oferta de productos según las preferencias individuales de los usuarios y optimizando el descubrimiento de productos relevantes y aumentando la satisfacción del cliente.

Business success criteria:

BG1 – KPI1: Precisión en la clasificación de los productos en las clases definidas de acuerdo con sus características.

Determine Data mining goals

Data mining goal:

DMG1: Modelar las relaciones complejas entre los productos de Amazon y sus características utilizando redes neuronales de grafos (GNN).

Data mining success criteria:

DMG 1 – KPI1: Obtener una métrica de accuracy mínimo de 0.90 en test.

DATA UNDERSTANDING

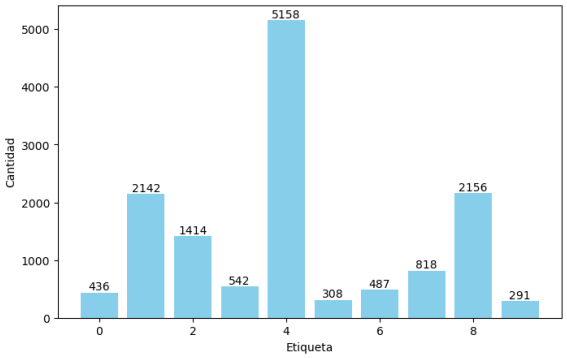
Dataset seleccionado: AmazonComputers (https://pytorch-geometric.readthedocs.io/en/latest/generated/torch_geometric.datasets.Amazon.html#torch_geometric.datasets.Amazon)

Se cuenta con dos conjuntos de datos, un conjunto de asociado a computadores y otro asociado a fotos. Para el actual trabajo se usará concretamente el asociado a computadores. El conjunto de datos contiene 13,752 productos los cuales pueden pertenecer a una de las diez categorías que se proponen. El conjunto de datos propone ya una representación vectorial (embedding), usando el método de Bag Of Words, en el que cada producto cuenta con 767 características (features). Adicionalmente, el conjunto de datos establece 491,722 aristas o enlaces entre los nodos (productos) indicando que los 2 pares de productos son frecuentemente comprados juntos.

En relación con el grafo, se identifican las siguientes características.

- No es un grafo direccionado.
- Posee nodos o productos aislados, es decir que hay productos frecuentemente son comprados solos.
- No tiene bucles (self-loops), lo cual es coherente dado que no tiene sentido que un producto sea frecuentemente comprado con el mismo.

En relación con las categorías de los productos, como se evidencia en la Figura 1, la categoría más grande es la número 4 con 5158 productos (38%) seguido de las categorías 8 y 1 con 2156 productos (16%) y 2142 productos (16%) respectivamente.



Cat	0	1	2	3	4	5	6	7	8	9
Porc	3%	16%	10%	4%	38%	2%	4%	6%	16%	2%

Figura 1. Distribución de los productos por categoría

Analizando el grafo de manera visual, la Figura 2 permite justamente constatar la existencia de los nodos aislados mencionados anteriormente.

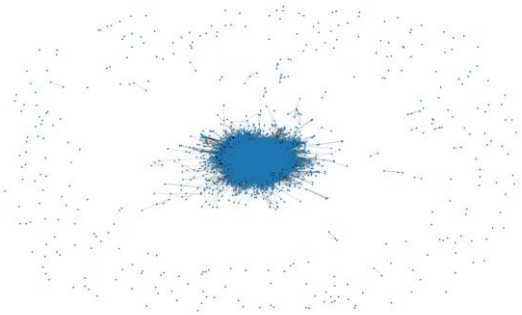


Figura 2. Visualización del grafo de AmazonComputers

A continuación, se realizará un análisis de las propiedades del grafo en relación con sus métricas de integración, de centralidad, de segregación y de resiliencia.

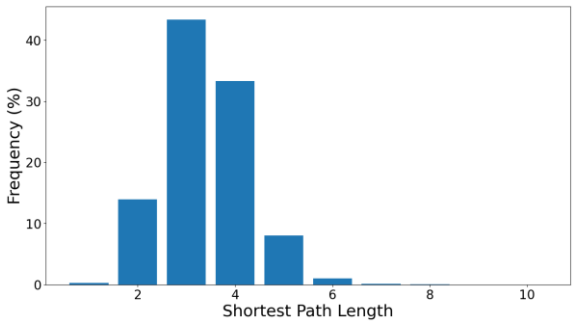
Métricas de integración

Las métricas de integración buscan medir que tan interconectado está el grafo, de manera que ayudan a comprender la estructura y las relaciones existentes dentro de él.

prom grados	Mediana grados	Camino más corto prom	Densidad	Eficiencia global
35.76	22	3.29	0.0026	0.30

Tabla 1. Métricas de integración del grafo de AmazonComputers

En cuanto a los nodos, como se puede observar en la Tabla 1, se obtuvo que el número promedio de grados es de 35.76 y una mediana de 22, concluyendo que el grafo tendría una baja conexión dado la gran cantidad de nodos totales (13,752). Esta conclusión se refuerza al analizar el valor de la densidad muy cercano a 0 y con una eficiencia global de apenas el 0.30.



1	2	3	4	5	6	7	8	9	10
0%	14%	43%	33%	8%	1%	0%	0%	0%	0%

Figura 3. Distribución porcentual de los caminos más cortos del grafo de AmazonComputers

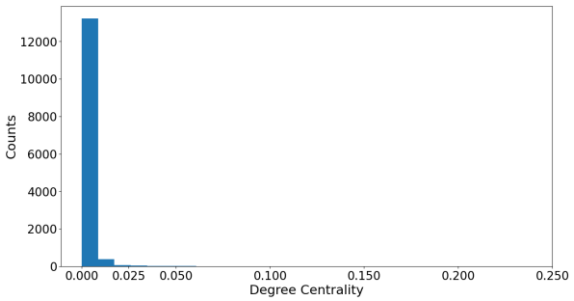
No obstante, analizando el camino más corto promedio del grafo en la Tabla 1 de 3.29 aristas, es posible ver en aquellas partes dónde está conectado el grafo, la información viaja fácil de un nodo a otro, siendo estos subgrafos los que contribuyen a que el grafo obtenga algo de densidad e integración. Adicionalmente, en la Figura 3, se identifica que la gran mayoría de los caminos más cortos posibles son principalmente de 2 a 5 aristas. El mínimo camino más corto es 1 arista y el máximo es de 10 aristas.

De manera general, se evidencia un grafo con baja conectividad lo que afecta a que la información tenga dificultades para ser propagada. Esto refuerza lo ya descrito y observado en la Figura 2.

Métricas de centralidad

Las métricas de centralidad buscan medir que tan importantes son los nodos de manera individual en relación con toda la red.

- **Centralidad de grados:** Mide la importancia de un nodo contando cuántos enlaces tiene. Un nodo con alta centralidad resulta "mejor" conectado y tendría una mayor influencia sobre la red.



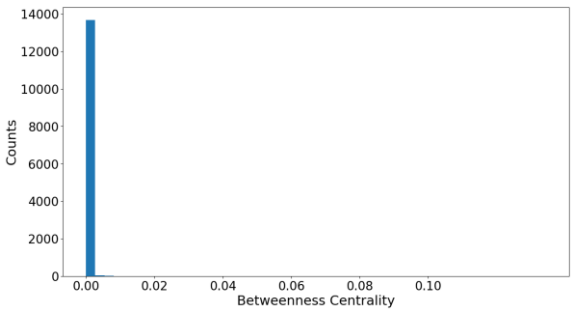
count	mean	std	min	25%	50%	75%
13752.0	0.0034	0.0078	0.0	0.0	0.0003	0.0032

80%	85%	90%	95%	99%	max
0.0034	0.0042	0.0053	0.0073	0.0178	0.2176

Figura 4. Centralidad de grado del grafo de AmazonComputers y sus estadísticos centrales y percentiles.

Analizando la centralidad del grafo de AmazonComputers, la Figura 4 muestra que la gran mayoría de los nodos, 99%, tienen una baja centralidad de grados. Esto implica que son apenas unos nodos que tienen un alto grado de enlaces, los cuales serían los productos más populares.

- **Centralidad de intermediación:** Evalúa cuantos caminos cortos pasar por cada nodo. Indica principalmente que nodos actúan como puente o intermediarios en la red.



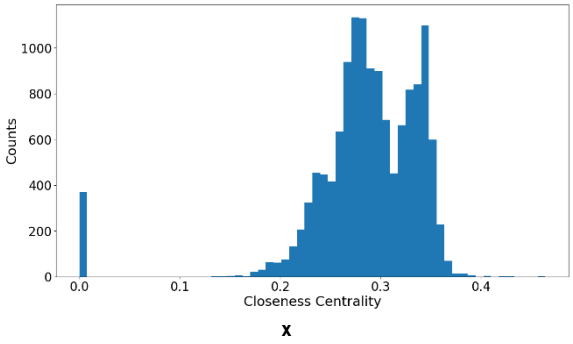
count	mean	std	min	25%	50%	75%
13752.0	0.0002	0.0019	0.0	0.0	0.0	0.0001

80%	85%	90%	95%	99%	max
0.0001	0.0002	0.0003	0.0005	0.0019	0.1344

Figura 5. Centralidad de intermediación del grafo de AmazonComputers y sus estadísticos centrales y percentiles

Analizando la centralidad del grafo de AmazonComputers, la Figura 5 muestra que la gran mayoría de nodos no son nodos puentes o intermediarios. Al igual que la centralidad de grados, son apenas unos productos populares que son vendidos con otros nodos no tan populares.

- **Centralidad de cercanía:** Mide que tan cerca está un nodo del resto y calcula la distancia promedio. Esto permite establecer la eficiencia con la que los nodos pueden comunicarse.



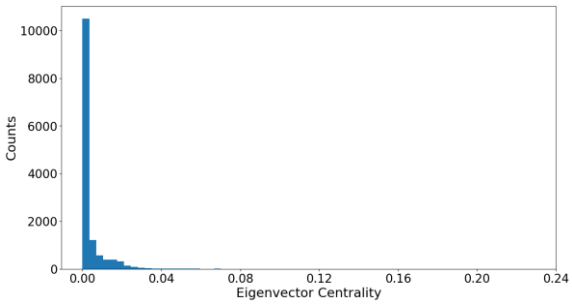
count	mean	std	min	25%	50%	75%
13752.0	0.2855	0.0618	0.0	0.2648	0.2898	0.3278

80%	85%	90%	95%	99%	max
0.334	0.3403	0.3457	0.3509	0.3627	0.4644

Figura 6. Centralidad de cercanía del grafo de AmazonComputers y sus estadísticos centrales y percentiles

Analizando la centralidad del grafo de AmazonComputers, la Figura 6 muestra que un gran grupo de nodos/productos están cerca entre ellos. Esto lo podemos corroborar con la Figura 2 dónde se visualiza que muchos nodos están conectados y cerca. Por otro lado, también podemos identificar aquellos grupos de nodos que tienen un valor de 0 al estar aislados.

- **Centralidad de vector propio:** Adicionalmente de medir las conexiones de un nodo, también mide la influencia de los nodos vecinos. Un nodo con alta centralidad implica que tiene entonces nodos importantes como vencinos.



count	mean	std	min	25%	50%	75%
13752.0	0.0034	0.0078	0.0	0.0	0.0003	0.0032

80%	85%	90%	95%	99%	max
0.0046	0.0069	0.0114	0.0176	0.0303	0.2108

Figura 7. Centralidad de vector propio del grafo de AmazonComputers y sus estadísticos centrales y percentiles

Analizando la centralidad del grafo de AmazonComputers, la Figura 7 muestra que la mayoría de los nodos no cuenta con nodos vecinos importantes. Analizando los percentiles, vemos que la tendencia de que sean apenas unos pocos productos los que concentran las mejores métricas de centralidad también se presenta en este caso. De manera que podemos concluir que son justamente aquellos nodos o productos populares los que están conectados entre sí.

De manera general sobre las métricas de centralidad, se identifican apenas unos pocos nodos muy importantes, ya sea por el número de conexiones o porque están conectados entre sí. No obstante, muchos de los nodos en el grafo se encuentran relativamente cerca.

Métricas de segregación

Las métricas de segregación buscan cuantificar la presencia de grupos de nodos interconectados, los cuales suelen ser denominados comunidades.

- **Coeficiente de agrupación:** Mide la tendencia de los nodos a formar grupos densamente conectados, implicando que los vecinos de un nodo también tienden a ser vecinos entre sí.

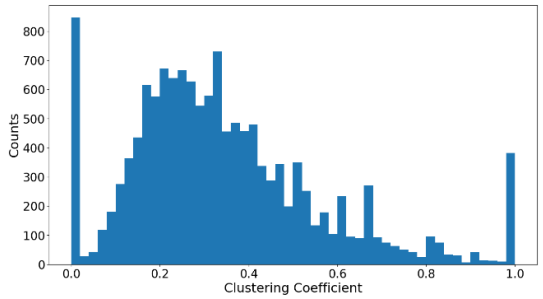


Figura 8. Coeficiente de agrupación del grafo AmazonComputers

Analizando la Figura 8, identificamos unas 4 poblaciones de nodos. En primer lugar, un grupo que cuenta con un coeficiente igual a 0, que corresponde con los nodos aislados o a nodos que no forman triángulos con sus vecinos. Un segundo grupo, correspondería a los nodos que tienen un coeficiente bajo (0.1 -0.4) y que, estimando el área sobre la curva, son el grupo más grande. Este grupo se caracterizaría por tener conexiones poco densas con pequeñas comunidades. Adicionalmente, viendo el coeficiente de agrupación promedio de 0.34, en la Tabla 2, se reafirma la tendencia general de los nodos del grafo. Un tercer grupo, son aquellos nodos con un coeficiente medio (0.4-0.8) dónde habría comunidades moderadamente densas. Por último, se identifica unos nodos con un coeficiente cercano o igual a 1 y que corresponden a comunidades cerradas dónde todos los vecinos están conectados.

Coef agrupación prom	Total triángulos únicos	prom triángulos únicos por nodo	Mediana triángulos únicos por nodo
0.34	1,527,469	333,22	69

Tabla 2. Métricas de segregación del grafo de AmazonComputers

Ampliando el análisis de segregación, vemos que la cantidad de triángulos únicos por nodo en relación con los potenciales triángulos dado la dimensión de nodos es bajo, lo que confirma la baja conexión de los nodos y por tanto una gran cantidad de comunidades resultantes y una gran cantidad de puentes, los cuales se pueden evidenciar en la Tabla 3.

# puentes	# puentes locales	# Comunidades
342	6,406	430

Tabla 3. Métricas de segregación del grafo de AmazonComputers

Lo concluido anteriormente, podemos visualizarlo en la Figura 9, dónde nuevamente se visualiza el grafo de AmazonComputers con las 430 comunidades. Los nodos aislados terminan teniendo una gran influencia en el número de comunidades identificadas.

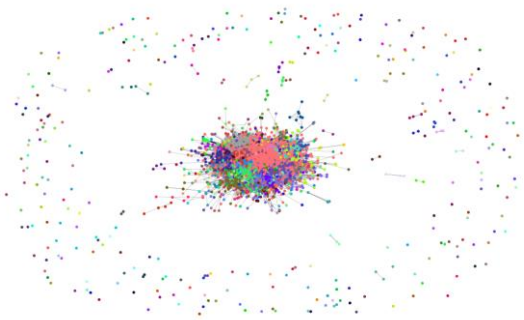


Figura 9. Grafo de AmazonComputers con las comunidades identificadas

Métricas de resiliencia

Las métricas de resiliencia permiten entender la estabilidad y adaptabilidad ante cambios de la red.

- **Coeficiente de asortividad**

El grafo reporta un coeficiente de asortividad igual a -0.056 lo que implicaría que no hay un patrón claro en las conexiones basadas en la popularidad de los productos, ya sea de productos muy populares con no populares o que productos populares se compraran juntos con más frecuencia.

DATA PREPARATION

El conjunto de datos trabajado ya tiene realizado una fase de preprocesamiento. Se aplicó un método de representación vectorial (embedding) usando bag-of-words, teniendo como insumo las descripciones y características de los productos. Este embedding construyó una representación vectorial con 767 atributos.

Por otro lado, si fue necesario realizar las correspondientes divisiones del conjunto de datos en los conjuntos de entrenamiento, de validación y de prueba.

Se realizó una división del conjunto de datos, usando un rango específico de índices teniendo en cuenta el número total de nodos. Esta división del conjunto de datos se implementó en los *modelos 3 – GCN 1 y Modelo 4 – GCN 2.*

Adicionalmente, se implementó una división del conjunto de datos aplicando un random_split. Los modelos a los que se les implementó está división fueron *Modelo 1 – GNN 1, Modelo 2 – GNN 2*, *Modelo 5 – GCN 3*, *Modelo 6 – GAT 1 (Single), Modelo 7 – GAT 2 (Dos capas)* y *Modelo 8 – GAT 3.*

MODELING

Para la etapa de modelamiento se utilizaron tres (3) tipos de redes neuronales para grafos, sobre los cuales se desarrollaron (8) modelos diferentes con las librerías de Pytorch. Los tipos de redes son:

- Graph Neural Network (GNN)
- Graph Convolutional Network (GCN)
- Graph Attention Network (GAT)

A continuación, se describe brevemente cada uno de los modelos implementados para la clasificación de artículos de computación en las 10 categorías predeterminados con sus correspondientes parámetros.

Red A: Graph Neural Network:

El GNN (Modelo de Red Neuronal de grafos) es una arquitectura de red neuronal diseñada para trabajar con datos estructurados como grafos. Las GNN están diseñadas para aprovechar la estructura y las relaciones entre nodos, cada nodo y cada arista puede tener un conjunto de características.

Modelo 1 – GNN 1: Yuri Angélica

Para este modelo se determinó la división de los conjuntos de entrenamiento, test y validación: 70%, 15%, 15% respectivamente. El modelo utiliza dos capas de convolución GCN para procesar las características de los nodos en el grafo, una función de activación ReLU, optimizador **Adam**, que se utiliza para actualizar los parámetros del modelo durante el entrenamiento. Una tasa de aprendizaje lr=0.01 que controla el tamaño de los pasos que da el optimizador para ajustar los pesos del modelo.

Se define una función de pérdida de 'CrossEntropy' y 150 épocas para el desarrollo de entrenamiento. Los anteriores también seleccionados a partir de prueba y error.

La siguiente figura resume la arquitectura del modelo:

```
GNN(
  (conv1): GCNConv(767, 64)
  (conv2): GCNConv(64, 10)
  (relu): ReLU()
)
```

Figura 10. Arquitectura GNN 1

Modelo 2 – GNN 2: Paula Alejandra

Para este modelo se determinó la división de los conjuntos de entrenamiento, test y validación: 60%, 10%, 30% respectivamente. Se crea una matriz de adyacencia normalizada, esencial para estabilizar el aprendizaje en GNN. El modelo utiliza tres capas de tipo SparseLayer, un dropout de 0,5, una función de activación ReLU para las capas 1 y 2 y la tercera capa genera predicciones de salida, aplicando log_softmax para obtener probabilidades logarítmicas. Se define una función de pérdida de 'CrossEntropy' y usa un optimizador **Adam**, que se utiliza para actualizar los parámetros del modelo durante el entrenamiento. Una tasa de aprendizaje lr=0.005 que controla el tamaño de los pasos que da el optimizador para ajustar los pesos del modelo y 100 épocas para el desarrollo de entrenamiento. Los anteriores también seleccionados a partir de prueba y error.

La siguiente figura resume la arquitectura del modelo:

```
GNN(
  (gnn1): SparseLayer(
    (linear): Linear(in_features=767, out_features=64, bias=False)
  )
  (gnn2): SparseLayer(
    (linear): Linear(in_features=64, out_features=32, bias=False)
  )
  (gnn3): SparseLayer(
    (linear): Linear(in_features=32, out_features=10, bias=False)
  )
)
```

Figura 11. Arquitectura GNN 2

Red B: Graph Convolutional Network:

Una red neuronal convolucional para grafos (GCN) aplica la idea de convoluciones en redes neuronales clásicas para ser empleadas en los grafos. Los GCNs permiten realizar la clasificación de nodos, la predicción de enlaces y el procesamiento de las relaciones en los grafos.

En una GCN aplicada a grafos, a diferencia de las redes neuronales convolucionales empleadas en procesamiento de imágenes, en lugar de tener una vecindad definida por una cuadrícula, cada nodo en un grafo tiene una vecindad que depende de sus conexiones con otros nodos. Por lo tanto, la GCN aprovecha dicha estructura para aprender representaciones de cada nodo en función de la información de sus nodos vecinos.

De esta forma, la GCN procesa la información en capas de convolución, donde cada capa transforma y agrega características de los nodos vecinos, lo que permite que la red aprenda representaciones contextuales de cada nodo.

Inicialmente, se definen las máscaras de entrenamiento, validación y prueba

```
data.train_mask = range(11000)
data.val_mask = range(11001, 12000)
data.test_mask = range(12001, 13752)
```

Modelo 3 – GCN 1: Pavel Mauricio

Se emplea la función GCN de PyTorch para el ejecución del modelo. Se definen 2 capas de convolución, la primera toma 767 características de entrada por nodo y produce 16 características de salida empleando la función de activación Relu; la segunda recibe 16 características por nodo, es decir, la salida de conv1 y produce 10 características de salida por nodo que representan la salida final del modelo, aplicando el logaritmo de la función softmax para representar las probabilidades para cada nodo del grafo de pertenecer a cada clase.

```
GCN(
  (conv1): GCNConv(767, 16)
  (conv2): GCNConv(16, 10)
)
```

Figura 12. Arquitectura GCN 1

Modelo 4 – GCN 2: Pavel Mauricio

En búsqueda de mejorar el modelo, se realizaron los siguientes ajustes:

- Añadir una tercera capa a la GCN.
- Aumento del tamaño de las capas ocultas a 32.
- Uso de dropout con una probabilidad de 0.5. para reducir sobreajuste
- Cambio en el optimizador a AdamW
- Reducción del learning rate.
- Aplicación de Early stopping (parada temprana).

```
GCN(
  (conv1): GCNConv(767, 32)
  (conv2): GCNConv(32, 32)
  (conv3): GCNConv(32, 10)
  (dropout): Dropout(p=0.5, inplace=False)
)
```

Figura 13. Arquitectura GCN 2

La primera capa genera 32 características por nodo reduciendo las dimensiones del vector de cada nodo, manteniendo la información relevante. La segunda, refina las características aprendidas en la primera capa manteniendo las 32 características por nodo. La capa final, al igual que en el modelo A produce las características que se usarán para clasificar cada nodo en una de las 10 clases.

Modelo 5 – GCN 3: Tatiana Roa

Para este modelo se definió una capa de convolución para grafos (GCNConv), la capa GCN utiliza parámetros de entrada como dim_in, dim_h, y dim_out, que corresponden a la dimensión de entrada, la dimensión oculta y la dimensión de salida, respectivamente. El modelo utiliza dos capas convolucionales. La primera capa toma entradas de dimensión dim_in y las transforma en una representación de dimensión dim_h. La segunda capa recibe la salida de la primera capa y la convierte en una representación de dimensión dim_out.

Se agregaron capas de regularización por el método de Dropout con un valor de 0.1, la función de pérdida utilizada es CrossEntropyLoss, adecuada para problemas de clasificación utilizando PyTorch. Además, se emplea un optimizador Adam con una tasa de aprendizaje (learning rate) de 0.01 para el entrenamiento. El modelo se entrena durante 180 épocas, lo que permite que la red ajuste sus pesos y minimice la función de pérdida a través de múltiples iteraciones.

```
GCN(
  (gcn1): GCNConv(767, 32)
  (gcn2): GCNConv(32, 10)
)
```

Figura 14. Arquitectura GCN 3

Red C: Graph Attention Network:

El Graph Attention Network (GAT) es un tipo de red neuronal especializada para procesar datos estructurados en grafos, diseñada para mejorar la forma en que un modelo "aprende" sobre la relación entre nodos. A diferencia a las anteriores redes neuronales de grafos, GAT emplea un mecanismo de atención, lo que significa que, durante el proceso de agregación de características entre nodos vecinos, el modelo asigna diferentes pesos a las conexiones (vértices) según su relevancia para la tarea específica. Este enfoque le permite priorizar información importante en el contexto del nodo objetivo, lo cual es particularmente útil en tareas como clasificación de nodos o predicción de enlaces en grafos complejos.

El GAT aplica un mecanismo de atención para asignar diferentes pesos a cada vecino de un nodo, calculando coeficientes de atención que determinan su relevancia. Usando estos coeficientes, se realiza una agregación ponderada de las características de los vecinos, creando una nueva representación enriquecida para cada nodo que refleja tanto sus propios atributos como la influencia de sus vecinos. Si el modelo usa múltiples cabezas de atención, esta agregación se repite en paralelo, capturando diferentes aspectos de las relaciones. Finalmente, las representaciones de los nodos pasan por una capa de clasificación con softmax, generando la predicción para las clases necesitada.

Modelo 6 – GAT 1 (Single): Juan David

Inicialmente, se define una capa de atención (GATv2Conv), que permite procesar información de los nodos y sus vecinos en el grafo. La capa GAT utiliza parámetros de entrada como dim_in, dim_out, y heads, que corresponden a la dimensión de entrada, la dimensión de salida y el número de cabezas de atención, respectivamente. El modelo utiliza un heads=32, lo que indica que emplea 32 cabezas de atención, permitiendo que cada nodo "atienda" a sus vecinos de múltiples maneras, extrayendo patrones más complejos de las relaciones en el grafo. Adicionalmente, se agregó una capa de regularización por el método de 'Dropout' con el fin de reducir la dependencia de la red en características específicas y mejorar la capacidad de generalización del modelo al entrenarlo para que sea menos sensible a data no conocida de validación y de prueba. Por medio de prueba y error, se utilizó un valor de 0.3.

Por último, se define una función de pérdida de 'binary crossentropy', para problemas de clasificación utilizando Pytorch, un optimizador Adam con un 'learning rate' de 0.01 y 200 épocas para el desarrollo de entrenamiento. Los anteriores también seleccionados a partir de prueba y error.

La siguiente figura resume la arquitectura del modelo:

```
SingleGAT(
  (gat_layer): GATv2Conv(767, 32, heads=10)
  (dropout): Dropout(p=0.3, inplace=False)
)
```

Figura 15. Arquitectura GAT 1: Single GAT

Modelo 7 – GAT 2 (Dos capas): Juan David

Para esta estrategia se definieron dos capas de atención para el procesamiento de los atributos de los nodos y sus vecinos. La primera capa toma entradas de dimensión dim_in y las transforma en una representación de dimensión dim_h multiplicada por el número de cabezas de atención (16). La segunda capa recibe la salida de la primera capa y la convierte en una representación de dimensión dim_out utilizando 8 cabezas de atención.

Para regularización, se agregaron dos capas de tipo 'Dropout' con valores de 0.1, permitiendo la utilización del 90% de neuronas.

Por último y al igual que el modelo anterior, se definió una función de pérdida de 'binary crossentropy', para problemas de clasificación utilizando Pytorch, un optimizador Adam con un 'learning rate' de 0.01 y 200 épocas para el desarrollo de entrenamiento.

La siguiente figura resume la arquitectura del modelo:

```
GAT(
  (gat1): GATv2Conv(767, 32, heads=16)
  (gat2): GATv2Conv(512, 10, heads=8)
)
```

Figura 16. Arquitectura GAT 2

Modelo 8 – GAT 3 (Multi): Camilo Andrés

El último modelo define una red de atención de grafos que permite crear modelos con múltiples capas de atención. La primera capa transforma las entradas de dimensión dim_in a una representación de dimensión dim_h con 32 cabezas. La capa intermedia contiene la misma estructura, pero reducen sus cabezas de atención a la mitad. La última capa convierte la representación a la dimensión de salida dim_out utilizando una sola cabeza de atención.

Para regularización, se aplica una capa de dropout con una probabilidad del 30% para prevenir el sobreajuste

Por último y al igual que el modelo anterior, se definió una función de pérdida de 'binary crossentropy', para problemas de clasificación utilizando Pytorch, un optimizador Adam con un 'learning rate' de 0.01 y 160 épocas para el desarrollo de entrenamiento.

La siguiente figura resume la arquitectura del modelo:

```
multiGAT(
  (gat_layers): ModuleList(
    (0): GATv2Conv(767, 32, heads=32)
    (1): GATv2Conv(1024, 32, heads=16)
    (2): GATv2Conv(512, 10, heads=1)
  )
  (dropout): Dropout(p=0.3, inplace=False)
)
```

Figura 17. Arquitectura GAT 3: GAT de dos capas

EVALUATION:

La métrica seleccionada para la evaluación fue el 'Accuracy' que mide la proporción de predicciones correctas respecto al total de predicciones realizadas.

A continuación, se realiza la consolidación de los resultados a través de la tabla 4, para set de entrenamiento, validación y testing:

#	Modelo	Métrica: Accuracy		
		Training	Validation	Test
1	GNN 1	88.61%	87.78%	88.13%
2	GNN 2	84.63%	84.16%	83.59%
3	GCN 1	84.51%	84.78%	84.35%
4	GCN 2	84.53%	84.38%	85.84%
5	GCN 3	89.65%	88.22%	88.37%
6	GAT 1 (Single)	92.46%	90.04%	90.41%
7	GAT 2 (Dos capas)	94.72%	89.89%	90.33%
8	GAT 2 (Multicapa)	88.99%	87.64%	89.17%

Tabla 4. Resultados 'Accuracy' para todos los modelos

En general, los resultados de los modelos según la métrica de 'Accuracy' son buenos, superando en su totalidad el umbral del 80%. También se observa que ninguno de los modelos presenta un sobreajuste relevante, donde los resultados entre entrenamiento, validación y prueba no presentan una varianza alta. Esto implica que los modelos son buenos para generar resultados similares ante información desconocida y que los modelos mantienen un desempeño estable y reproducible.

Por otro lado, los resultados muestran que los modelos con mejor métrica de 'Accuracy' son el modelo 6, 7 y 8, y de estos, solo el 6 y 7 logran superar el umbral de 90%. Si bien la métrica de los modelos basados en GNN y GCN no presentan una diferencia relevante, los modelos basados en GAT logran mejorar la métrica de forma consistente por encima de los otros métodos.

Para el modelo 6, que tuvo el mejor desempeño, se graficó la función de pérdida y la métrica de 'Accuracy' para cada época en los conjuntos de validación y de entrenamiento. Se observan a continuación:

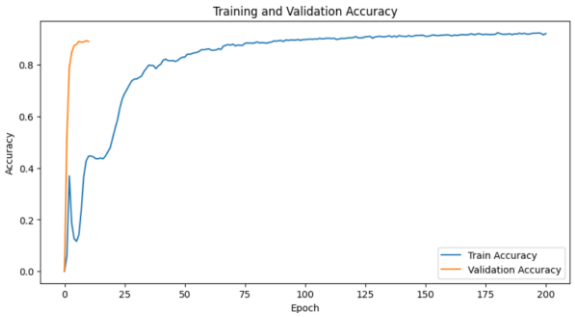


Figura 18. Modelo 6 – 'Accuracy' en entrenamiento y validación

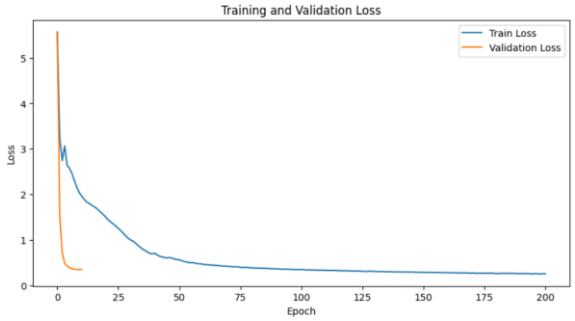


Figura 19. Modelo 6 – Función de Pérdida en entrenamiento y validación

Ambas gráficas demuestran que el modelo aprende con facilidad y sigue refinando el resultado y las predicciones hasta encontrar el mejor resultado.

Por último, se presenta la matriz de confusión del mejor modelo, donde se observa la proporción de predicciones y valores reales acertados. Se muestra en la siguiente figura:

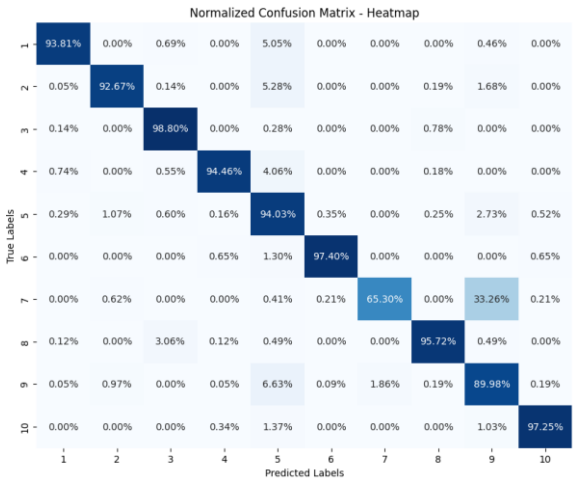


Figura 20. Modelo 6 – Matriz de confusión multiclase

En general, se observa que el modelo presenta en una proporción alta coincidencia entre las predicciones y los valores reales, para la mayoría de las categorías. En este caso, únicamente la categoría 7 presenta una precisión menor al 89%, donde se genera una confusión con la categoría 9.

Evaluación objetivos de minería de datos: Teniendo en cuenta los resultados presentados, se observa que se tiene éxito en el objetivo y criterio planteado, donde dos (2) de los modelos tuvieron un desempeño mayor a 0.90 en 'Accuracy'. En este caso, ambos modelos fueron desarrollados con Graph Attention Network (GAT), de una y dos capas.

RECOMENDACIONES DE NEGOCIO

Se recomiendan las siguientes estrategias a partir de los objetivos de minería de datos y de negocio y la aplicación de los modelos:

- Utilizar el modelo 6 – GAT 1 (Single) para detectar clústeres de productos que compartan características similares o complementarias, en pro de facilitar la personalización de las ofertas y recomendaciones para los usuarios, incrementando la relevancia de los productos sugeridos y mejorando la satisfacción del cliente.
- Realizar pruebas para validar el impacto en la experiencia del usuario y satisfacción en un entorno real, a partir del rendimiento del modelo frente al sistema de recomendación definido.
- Colaborar con equipos multidisciplinarios que incluyan especialistas en la gestión de catálogos y/o en experiencia del usuario para garantizar que el modelo se integre eficazmente en la plataforma y el crecimiento constante de los productos ofertados, de modo que las recomendaciones generadas sean intuitivas y útiles para los usuarios.
- Utilizar las predicciones de productos recomendados para ajustar los niveles de inventario, especialmente para productos que tienden a ser comprados en conjunto o aquellos con una alta probabilidad de conversión.
- Usar los datos generados por el modelo para identificar tendencias en las preferencias de los clientes y anticipar la demanda futura, asegurando que Amazon pueda adaptarse a los cambios en el mercado, reduciendo costos asociados a inventarios excesivos y mejorando la disponibilidad de productos demandados en diferentes épocas.

REPORTE DE TRABAJO EN EQUIPO

Camilo Andrés González Vargas: Desarrollo del Modelo 7 – GAT 2 (Dos capas), código de la descriptiva, data understanding.

Juan David Valencia Sandoval: Desarrollo del Modelo 6 – GAT 1 (Single) y Modelo 8 – GAT 3 (Multi), Evaluation

Paula Alejandra Velasco Cuberos: Desarrollo del Modelo Modelo 2 – GNN 2, Presentación del proyecto, código de descriptiva.

Pavel Mauricio Dussan Gutiérrez: Desarrollo del Modelo Modelo 3 – GCN 1 y Modelo 4 – GCN 2. Presentación del proyecto

Tatiana Roa Ahumada: Desarrollo del Modelo 5 – GCN 3, Business Understanding, recomendaciones de negocio

Yuri Angélica Reina Guzmán: Desarrollo del Modelo 1 – GNN 1, Recomendaciones de negocio

BIBLIOGRAFIA

[1]Scikitlearndevelopers. (n.d.). sklearn.multiclass.OneVsRestClassifier. Scikit-learn 1.3.0 documentation.

https://scikit-learn.org/stable/modules/generated/sklearn.multiclass.OneVsRestClassifier.html

González Mallo, A. (2024). Extracción de características basadas en sinopsis para la clasificación de películas en géneros cinematográficos. Extraído de: https://ruc.udc.es/dspace/handle/2183/39294

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. (2018). **Graph Attention Networks**. *International Conference on Learning Representations (ICLR)*. Recuperado de https://arxiv.org/abs/1710.10903.