Informe Laboratorio 1 – Agrupación

Integrantes: Juan Pablo Lora Hernández – 202012524

Andrés Francisco Borda – 201729184

Gabriela Vargas Rojas – 202013830

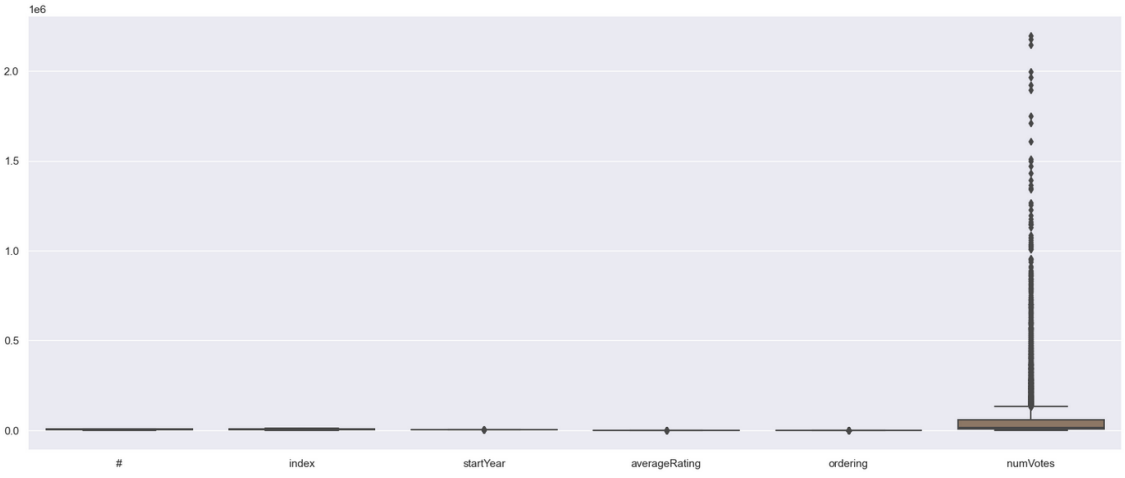
1. Entendimiento de los datos:

1.1. Perfilamiento:

Al explorar los datos encontramos que nos dieron registros de 7471 películas, series y miniseries que se encuentran en la plataforma MovieAlpes. Cada registro cuenta con 18 columnas de las cuales 8 numéricas y 10 categóricas. De estas, 4 columnas nos sirven para identificar cada uno de manera única, 11 columnas nos dan información sobre las características de la cinta cinematográfica y las ultimas 3 nos dan una idea de que tan populares o bien recibidas son en el público. Adicionalmente, encontramos las columnas "isOriginalTitle", "Ordering", "#" e "index" y estas no tienen referencia en el diccionario de los datos.

Se hizo el análisis estadístico de las variables categóricas y adicionalmente se presenta en un boxplot la información de cada una.

	#	index	startYear	averageRating	ordering	numVotes
count	7471.000000	7471.000000	7470.000000	7470.000000	7470.000000	7.470000e+03
mean	4043.482666	5440.458439	2013.157296	7.475676	16.567604	7.282830e+04
std	2192.753689	2752.820924	6.979151	2.771444	12.761147	1.640233e+05
min	1.000000	1.000000	1990.000000	6.500000	1.000000	1.000000e+01
25%	2152.500000	3073.500000	2008.000000	6.900000	6.000000	6.265250e+03
50%	4033.000000	5421.000000	2015.000000	7.300000	14.000000	1.527900e+04
75%	5914.500000	7641.500000	2019.000000	7.800000	24.000000	5.641975e+04
max	7849.000000	10274.000000	2023.000000	92.000000	119.000000	2.197234e+06



Se hizo el análisis estadístico de las variables categóricas. Las variables “isAdult” y “isOriginalTitle” son categóricas, pero son representadas como numéricas por lo que debieron ser analizadas aparte, pero se encontró que ambas columnas tienen únicamente el valor de 0 por lo que solo tienen un valor único.

	tconst	titleType	originalTitle	runtimeMinutes	region	language	types	attributes	main_genre	secondary_genre
count	7471	7471	7471	7470	7470	7470	7470	7470	7470	7470
unique	7466	7	7290	362	4	1	5	1	22	26
top	tt0395843	movie	Home	60	IN	en	imdbDisplay	\N	Drama	Drama
freq	2	4673	3	534	5590	7470	7395	7470	1767	2428

1.2. Análisis de calidad de datos:

1.2.1 Completitud:

Para el análisis de completitud se vio que porcentaje de valores nulos tiene cada una de las columnas de los datos. Se encontró que 5 de las columnas están completas mientras que las demás están 0.000134% incompletas. Esto quiere decir que hay un registro que no tiene los datos completos.

```
numVotes          0.000134
ordering          0.000134
main_genre        0.000134
isOriginalTitle   0.000134
attributes        0.000134
types             0.000134
language          0.000134
region            0.000134
secondary_genre   0.000134
averageRating     0.000134
runtimeMinutes    0.000134
startYear         0.000134
isAdult           0.000134
index             0.000000
originalTitle     0.000000
titleType         0.000000
tconst            0.000000
#                 0.000000
```

1.2.2 Unicidad:

Para el análisis de unicidad primero se buscó si había registros repetidos y se encontró que existen 6 registros que están duplicados. Adicionalmente, como vimos en el perfilamiento de datos existen cuatro columnas que tienen un único valor en todos los registros por lo que la columna no nos da mucha información. Por último, las tres columnas de identificadores únicos nos dan información redundante entre ellas por lo que tal vez solo una debería ser tomada en cuenta.

1.2.3 Consistencia:

Se encontró que la columna de “titleType” cuenta con valores que no están en el diccionario y probablemente son errores de digitación.

```
movie           4673
tvSeries        2340
tvMiniSeries     379
tvMovie           70
MOVIE              5
MOvie              3
Movie              1
Name: titleType, dtype: int64
```

También se encontró que para la columna “averageRating” hay un total de 13 valores que se encuentran fuera de los parámetros ya que esta es una columna numérica float, donde se asigna una calificación a la película hasta un máximo de 10.0 y estos 13 registros poseen valores como 68 o 92, rompiendo los limites en este caso.

1.2.4. Validez:

Para la columna “runtimeMinutes” hay 98 registros que a pesar de no estar vacíos tienen como información “/N” lo cual no es un dato válido teniendo en cuenta que aquí encontramos información sobre la duración de la película o serie. De esta manera, en este campo para esos 98 registros no hay una validez.

2. Preparación de datos

2.2. Corrección de Datos:

- Respecto a la completitud de los datos, como pudimos ver en nuestro análisis para algunas columnas tenemos 1 registro que tiene valores vacíos. De esta forma, consultando a MovieAlpes en todo caso, consideramos que no tomaremos en cuenta este registro al representar un % respecto al total demasiado bajo por lo cual no influirá en nada. De esta forma, se eliminó este registro para el modelo.
- Por otra parte, con los registros que presentaban problemas de consistencia para la columna “averageRating” decidimos que, a pesar de no representar un volumen alto del total de los datos, sí que es una columna que posee información vital para el contexto de MovieAlpes, por ello arreglamos estos datos para que quedaran dentro de los parámetros asumiendo que “68” es 6.8 de calificación y que “92” corresponde a 9.2.
- En el caso de la columna “titleType” según el diccionario de datos proporcionado por MovieAlpes, los valores permitidos eran “movie”, “tvSeries”, “tvMiniSeries” y “tvMovie”, por lo cual los errores encontrados respecto a mayúsculas o minúsculas fueron corregidos y se establecieron valores únicamente acordes a los mencionados en el diccionario.
- Para el tema de los duplicados (6) decidimos tomar un único registro de estos en representación y así eliminar el resto del conjunto para tener datos libres de duplicados en el modelo.

- Otro de los problemas de validez estaba en la columna “runtimeMinutes” donde 98 registros tenían como valor “/N”, así que tomamos estos datos y les asignamos un valor estimado a partir del uso de una media por género acorde a si eran movie, tvSeries, tvMovieSeries o tvMovie, de esta forma tienen valores numéricos que si son válidos para la columna
- Se analizaron las distribuciones de las variables numéricas y se encontró que las variables “numVotes”, “runtimeMinutes” y “ordering” tienen outliers significativos por lo que estos fueron eliminados.

3. Modelamiento

3.1 Selección de Variables:

Teniendo en cuenta el contexto presentado por el negocio y nuestra tarea de aprendizaje (agrupación) decidimos tomar las variables de “titleType”, “averageRating” y “numVotes” para nuestros modelos. Principalmente nuestra elección se basó en las recomendaciones que necesitan los usuarios de MovieAlpes, de tal forma que, la popularidad y la calificación de los productos cinematográficos es de vital importancia para poder hacer un “ranking” de las mejores y así mismo recomendarlas. Sumado a esto, se necesita una descripción para separar si es película o si es serie por lo cual la columna de “titleType” permite discernir en este criterio.

También, durante el análisis de los datos pudimos notar que había relaciones interesantes al comparar estas dos variables numéricas con “titleType” que pueden ser beneficiosas para MovieAlpes.

3.2 Algoritmo K-means (Hecho por Juan Pablo Lora Hernández):

Para el desarrollo del modelo para el algoritmo k-means, lo primero que se realizó fue la conversión de la variable “titleType” de categórica a numérica usando el método OneHotEncoder creando así columnas binarias de “movie”, “tvMovie”, “tvSeries” y “tvMiniSeries”.

Por otra parte, también se hizo una normalización de todas las variables, con la función MinMaxScaler haciendo que todas tuvieran el mismo dominio.

Posteriormente, para la asignación del número de clúster a usar utilizamos el método del codo, que será ahondado en profundidad en la sección de validación, pero como adelanto, este nos dio un punto favorable de baja distorsión para cualquier k>=3.

Así que, los hiperpárametros dispuestos para este algoritmo fueron número de clúster = 3 esto determina la cantidad de grupos a formar, n_init = 10 esto define cuantas veces se ejecutara el algoritmo con diferentes centroides iniciales, max_iter= 300 este indica el número máximo de iteraciones permitidas antes de converger y parar, init= “k-means++” el cual nos permite mejorar la inicialización aleatoria de centroides y random_state=0 que determina un valor numérico como semilla para generar números aleatorios. Se establecieron estos valores después de diferentes exploraciones de los grupos, ya que, estos nos permitieron llegar a un valor de coeficiente de silueta bastante bueno.

¿Cómo Funciona?

K-means básicamente es un método de agrupación que nos permite dividir un conjunto de datos en grupos o clústeres de manera que los elementos de un mismo clúster sean más similares entre sí pero muy diferentes con los elementos de otros clústeres, así que para ello debemos inicializar unos centroides y vamos asignando puntos a estos clústeres donde el centroide este más cerca en términos de una distancia euclidiana, cuando terminamos de asignar todos los puntos se recalcula el centroide de cada clúster a partir de la media de todos los puntos que tiene asignados, esto se repite generalmente hasta que los centroides dejen de cambiar significativamente y es allí donde entramos a analizar los resultados con métricas como el coeficiente de silueta de los clústeres.

3.3 Algoritmo DBSCAN (Hecho por Andrés Borda)

EL algoritmo DBSCAN utiliza una medida de distancia euclidiana entre los diferentes puntos de datos para su funcionamiento por lo que es sensible a que las diferentes variables estén en escalas diferente. Dado esto se utilizó el método de MinMaxScaler para normalizar los datos. Una vez normalizados los datos se procedió a calcular los hiperpárametros óptimos para el algoritmo y nuestra serie de datos. El algoritmo funciona con dos hiperpárametros que son la distancia mínima entre puntos para ser considerados del mismo clúster (ϵ) y el número mínimo de datos para poder formar un clúster (n). Para encontrar los valores óptimos para estos hiperpárametros se realizó un método basado en la iteración, en el cual se intentaron una serie de valores posibles para estos hiperparametros y se escoge la combinación que lleve al coeficiente de silueta que indique unos clústers mejores. Se encontró que los parámetros a utilizar deberían ser $\epsilon = 0.2$ y n = 10. Después de esto se implementó el algoritmo el cual dio como resultado una agrupación en 7 clústers de datos y 7 puntos de "ruido" que no pudieron ser asignados a ninguno.

¿Cómo funciona?

DBSCAN es un algoritmo que se basa en agrupar los datos en áreas de alta densidad separadas por áreas de baja densidad. El algoritmo pasa por todos los puntos y revisa si existen puntos a una distancia menor a ϵ para crear un grupo. En cada punto se comienza a buscar si existen otros puntos que estén al alcance de la distancia objetivo y estos son añadidos al grupo. Si al realizar esta iteración se encuentran por lo menos n-1 puntos dentro del grupo se puede considerar un clúster, si no es el caso no se pueden agrupar. Una vez añadidos los puntos nuevos se realiza la misma comparación con sus vecinos para añadir nuevos puntos. Debido a este funcionamiento los clústers de este algoritmo pueden tener cualquier tipo de forma mientras los puntos que conforman la forma mantengan la densidad planteada. Si existen puntos que no están a una distancia suficientemente cercana como para pertenecer a ninguno de los clústers y que con sus vecinos que si están a distancia no alcanzan a formar un clúster estos son considerados puntos de "ruido" que no pertenecen a ninguna agrupación. Como no importa por qué puntos se comience el algoritmo dado que la distancia de los puntos nunca va a cambiar siempre se llegará a la misma respuesta por lo que este algoritmo es determinístico.

3.4 Algoritmo Spectral Clustering (Hecho por Gabriela Vargas)

El algoritmo Spectral Clustering está enfocado en identificar comunidades de nodos en un grafo basándose en la estructura de similitud entre los datos. Este algoritmo se

desarrolla principalmente en 5 pasos: Primero, se construye una matriz de similitud que logre capturar la afinidad entre las parejas de muestras; segundo, se realiza la matriz laplaciana teniendo en cuenta la matriz de similitud realizada en el paso anterior, con el fin de capturar la estructura de conectividad entre las muestras; tercero, es el cálculo de los vectores propios espectrales; cuarto, la reducción de dimensionalidad, y, por último, se hace el clustering usando el espectro de la matriz laplaciana.

En la realización del algoritmo, primero se realizó el cambio de la variable categórica 'titleType' a numérica, usando OneHotEncoder. Una vez con esto hecho, se normalizó con MinMaxScaler, sin embargo, también se tuvo en mente normalizar con RobustScaler. Al realizar pruebas con RobustScaler, se pudo dar cuenta que, teniendo en cuenta el coeficiente de silueta, el valor era muy pequeño por lo que no se encontraba muy bien el modelo, por eso, se decidió realizar con MinMaxScaler ya que los resultados que arrojó fueron mejor. Seguidamente, se realizó la prueba de codo para ver el número de clústers, el cual nos arrojó que la mejor opción era 3, y con este valor, se implementó en el algoritmo Spectral Clustering. Al realizar la gráfica de silueta, pudimos ver los 3 clústers con su forma adecuada y con un coeficiente exacto de 0.7618

4. Validación:

4.1 Validación Cuantitativa:

- **Algoritmo K-means:**
  A nivel cuantitativo podemos ver que el coeficiente de silueta es de aproximadamente 0.7702 en promedio, en el caso del grupo 3 su máximo coeficiente llega alrededor de 0.65, esto también sucede debido a que hay una menor cantidad de datos en este clúster a comparación de los clústeres 1 y 2.
- **Algoritmo DBSCAN**
  Para evaluar cuantitativamente el algoritmo de DBSCAN se utilizó el método de la silueta y se encontró que es de aproximadamente 0.7697.
- **Algoritmo Spectral Clustering:**
  Se pudo obtener un coeficiente de silueta de 0,7618.

4.2 Validación Cualitativa:

- **Algoritmo K-means:**
  A nivel cualitativo podemos afirmar que los comportamientos en los diferentes grupos se caracterizan por "averageRating" y "numVotes", como información adicional se pudo identificar que en el grupo 1, están la mayoría de "tvSeries", mientras que en el grupo 2 las "movies" y finalmente en el grupo 3 "tvMiniSeries" y unas pocas "tvMovie". Con esto presente, se le puede comunicar a MovieAlpes que el mayor éxito teniendo en cuenta el "averageRating" y el "numVotes" se encuentra en las movies y las tvSeries que son muy vistas y recomendadas por los usuarios a pesar de que, según el análisis arrojado parece ser que apuntan a segmentos del mercado distintos. Sumado a esto, dentro de las variables no usadas en la construcción de los grupos, es importante resaltar el "main_genre" donde sería interesante revisar

los grupos generados en contraste al género cinematográfico al que pertenece, especificando aún más los gustos generales de los usuarios MovieAlpes para iniciar las recomendaciones.

- **Algoritmo DBSCAN**
  Cualitativamente se analizaron los resultados arrojados por el algoritmo DBSCAN por medio de la visualización de los datos y los clústers generados. De los resultados obtenidos podemos determinar que el algoritmo genero cuatro clústers cada uno correspondiente a una de las categorías de la variable "titleType". Adicionalmente, podemos ver que todos los puntos de ruido son "tvMiniSeries" con un número de votos altos dándonos el indicio de que estas miniseries no son vistas por tantas personas. Por último, podemos ver que mientras las series parecen tener un rating más alto, las películas un número de votos más altos por lo que cumplen funciones diferentes en el mercado.

- **Algoritmo-Spectral-Clustering**
  Teniendo en cuenta la gráfica de silueta, los 3 clústeres obtenidos, tuvieron un coeficiente alto, lo cual nos lleva a pensar que tenemos un modelo bien construido.

5. Justificación Modelo Recomendado

Después de construir los respectivos modelos para cada uno de nuestros algoritmos de clustering y analizar los resultados arrojados, principalmente, es pertinente concluir alrededor del coeficiente de silueta que obtuvimos para cada caso, el cual fue muy similar teniendo resultados de 0.7702 para el algoritmo de k-means, 0.7697 para el algoritmo de DBSCAN y 0.7618 para el algoritmo de Spectral clustering. Teniendo esto en cuenta, los 3 modelos fueron bien construidos pero el mejor para las variables seleccionadas y el número de agrupaciones es el algoritmo de K-means. Por lo tanto, este es el modelo recomendado para el sistema que desea implementar MovieAlpes, es importante recordar que las columnas de datos que fueron usadas son importantes pero con el análisis realizado hay una variable importante a tener en cuenta para juntar con los grupos del k-means, y esta es "main_genre" la cual permitirá acoplar aún más las calificaciones de los usuarios de la plataforma y así mismo seccionar de alguna manera las mejores recomendaciones que se pueden ofrecer dentro de la misma.

Finalmente, las principales consideraciones para tener en cuenta tras los resultados del algoritmo de este modelo (k-means), son que la mayor parte del mercado de usuarios que tiene MovieAlpes gira en torno a "movies" y "TvSeries", por lo cual sería importante centrar las recomendaciones a estos grupos tanto por las buenas calificaciones que poseen este tipo de productos como por tener la mayor cantidad de datos relacionados de consumo de los usuarios. Pues, en comparación a las otras categorías de "tvMiniSeries" y "tvMovie" realmente se visualiza un segmento mucho menor y con menos variedad de un valor para el mercado de la plataforma.