



Proyecto Data Science

Estimación de ratings en animés

Walther Becks

Marco Cordero

Elio Duran

Diego Lizana

Rodrigo Sepúlveda

Víctor Urra

20 de junio de 2023

Profesor Guía
Camilo González Rojas

Santiago, Chile

Índice de Contenidos

Índice de Figuras	iii
Índice de Tablas	iv
1. Introducción	1
2. Motivación.....	1
3. Alcance.....	2
4. Equipo de trabajo.....	2
5. Planificación de la Investigación	3
5.1 Planificación	3
5.2 Pregunta de investigación	3
5.3 Hipótesis de trabajo.....	3
5.4 Definición del vector objetivo	3
5.5 Preprocesamiento de datos.....	3
5.5.1 Procesamiento nulos.....	5
5.5.2 Estandarización y transformación de los datos	6
5.5.3 Tratamiento de outliers.....	12
6. Análisis descriptivo	13
6.1 Análisis univariado.....	16
6.2 Análisis multivariable.....	17
6.3 Análisis de correlaciones	18
7. Creación de datasets para la modelación	19
8.1 Modelos a implementar	21
8.2 Modelación preliminar.....	22
8.3 Elección de los dataframes y modelos candidatos.....	23
8.4 Voting Regressor.....	25
8.5 Red Neuronal	25
9. Segunda Iteración de los modelos candidatos	26
9.1 SVR.....	26
9.2 Gradient Boosting.....	28
9.3 Random Forest.....	30
9.4 Elastic Net	31
9.5 Voting Regressor.....	32
9.6 Elección mejor modelo	33
10. Conclusiones	34
10.1 Árbol de decisiones	36
11. Anexos.....	38

11.1 Carta Gantt.....	38
11.2 Código.....	38
11.3 Enlace a public.tableau.....	38
11.4 Gráficos de dispersión	38
11.5 Boxplots.....	40
11.6 Matriz de correlaciones.....	43

Índice de Figuras

Figura 1: Procedimiento para variables del tipo objeto	4
Figura 2: Procedimiento datos nulos	5
Figura 3: Gráfico episodios sin estandarizar	6
Figura 4: Gráfico miembros sin estandarizar.....	7
Figura 5: Gráfico rating sin estandarizar	7
Figura 6: Proceso estandarizaciones	8
Figura 7: Gráfico episodios con standarscaler	8
Figura 8: Gráfico miembros con standarscaler.....	9
Figura 9: Gráfico episodios con robustscaler	9
Figura 10: Gráfico miembros con robustscaler.....	10
Figura 11: Gráfico episodios con logaritmo	10
Figura 12: Gráfico miembros con logaritmo.....	11
Figura 13: Gráfico episodios con raíz cuadrada	11
Figura 14: Gráfico miembros con raíz cuadrada	12
Figura 15: Proceso estandarizaciones sin outliers	12
Figura 16: EDA	13
Figura 17: Gráfico género vs rating.....	14
Figura 18: Extracto del gráfico genero sin combinar	15
Figura 19: Histograma de la variable members.....	15
Figura 20: Boxplot del dataset sin tratamiento de outliers	16
Figura 21: Boxplot del dataset con tratamiento de outliers	16
Figura 22: Gráfico de dispersión sin tratamiento de outliers.....	17
Figura 23: Gráfico de dispersión con tratamiento de outliers.....	17
Figura 24: Boxplot sin tratamiento de outliers	18
Figura 25: Boxplot con tratamiento de outliers	18
Figura 26: Matriz de correlaciones entre rating, episodes y members.....	19

Figura 27: Métricas para cada dataframe.....	23
Figura 28: Métricas para dataset con raíz cuadrada y jerarquizado	24
Figura 29: Gráficos en relación con el error en el modelo SVR	28
Figura 30: Gráficos en relación con el error en el modelo GB	29
Figura 31: Feature importance	29
Figura 32: Gráficos en relación con el error en el modelo RF.....	30
Figura 33: Gráficos en relación con el error en el modelo EN	31
Figura 34: Gráficos en relación con el error en el modelo VR	32
Figura 35: Métricas de los 5 modelos.....	33
Figura 36: Comparación de modelos GB	34
Figura 37: 10 peores vs 10 mejores animés.....	35
Figura 38: Árbol de decisión	36
Figura 39: Árbol de decisión sin episodios ni miembros	37

Índice de Tablas

Tabla 1: Descripción roles integrantes	2
Tabla 2: Porcentaje de valores nulos en las variables.....	5
Tabla 3: Medidas estadísticas de episodios, miembros y rating	16
Tabla 4: Nivel de Jerarquización para type	20
Tabla 5: Métricas de la modelación preliminar	22
Tabla 6: Métricas de los datasets.....	24
Tabla 7: Métricas del Voting Regressor.....	25
Tabla 8: Métricas de la red neuronal	26
Tabla 9: Parámetros del modelo SVR	27
Tabla 10: Métricas 2da iteración SVR.....	27
Tabla 11: Parámetros del modelo GB.....	28
Tabla 12: Métricas de 2da iteración GB.....	29
Tabla 13: Parámetros del modelo RF	30
Tabla 14: Métricas 2da iteración RF	30
Tabla 15: Parámetros del modelo EN	31
Tabla 16: Métricas 2da iteración EN	31
Tabla 17: Parámetros del modelo VR	32
Tabla 18: Métricas modelo VR	32
Tabla 19: Resumen métricas de los modelos	33
Tabla 20: Extracto Dataframe con predicción del rating.....	35

1. Introducción

La era digital ha revolucionado la forma en que consumimos contenido de entretenimiento, proporcionando acceso inmediato a una gran diversidad de series y películas de todo el mundo. En particular, el Animé, un género originario de Japón ha captado la atención de millones de personas y se ha consolidado como una fuerza significativa en la industria del entretenimiento global.

Somos una plataforma líder para la exploración y valoración de series y películas de Animé, facilitando a los usuarios la elección de contenidos que se adapten a sus gustos y preferencias, desempeñando un papel clave en la toma de decisiones de los fans de este género.

Sin embargo, hemos identificado un reto importante, un porcentaje considerable de nuestra muestra de contenido no tiene valoraciones. Esta falta de información puede afectar la percepción de nuestros usuarios sobre la calidad y el atractivo de nuestro contenido, y potencialmente, su decisión de invertir tiempo en ellas.

2. Motivación

Los animes son una forma popular de entretenimiento con una amplia variedad de géneros, estilos y temáticas de amplia popularidad mundial y el motivo que nos llevó a elegir este problema, fue debido a: la industria expansiva, teniendo hitos importantes como la duplicación de las ganancias de la industria en 10 años, entre el 2009 de cerca de ~12 billones de dólares a ~24 billones de dólares el 2019, junto con el interés en explorar y comprender cómo los diferentes atributos de los animes pueden influir en la calificación dada por los usuarios y la posibilidad de resolver un problema potencialmente importante y significativo, utilizando los aprendizajes en Machine Learning.

Es importante abordar esta brecha y proponemos la creación de un sistema de aprendizaje automático que, basándose en las características de las series y películas, pueda estimar el puntaje que deberían tener. Esto no sólo mejoraría la experiencia de nuestros usuarios, sino que también nos permitirá recopilar datos más precisos y efectivos sobre sus preferencias.

El potencial real de este sistema va más allá de nuestra plataforma, ya que una vez perfeccionado, este producto podría ser comercializado a otros interesados, abriendo nuevas fuentes de ingresos y posicionando a nuestra empresa como líder en la vanguardia tecnológica del mundo del entretenimiento y del Animé en particular.

3. Alcance

In-Scope

- Limpieza de datos.
- Estadística aplicada para la creación de dataset(s).
- Modelos predictivos para entregar recomendaciones a usuarios.
- Presentación del proyecto.

Out-Scope

- No se trabajará con bases de datos distintas a las entregadas.
- La API que podría ser comercializada no será creada en este proyecto.

4. Equipo de trabajo

El equipo de trabajo se conforma por los siguientes integrantes: Víctor Urra, Diego Lizana, Rodrigo Sepúlveda, Walther Becks, Elio Durán y Marco Cordero; de los cuales según tanto de sus fortalezas como de sus debilidades se definieron los roles (tabla 1), dentro de cada uno de estos roles, el integrante deberá cumplir con ciertas responsabilidades que se detallan a continuación.

Tabla 1: Descripción roles integrantes

Nombre	Rol	Descripción del Rol
Víctor Urra	Líder del equipo	Responsable de planificar, coordinar y supervisar el desarrollo de la plataforma. Dirige al equipo, establece objetivos claros, asegura el cumplimiento de los plazos y se comunica con los interesados para garantizar el éxito del proyecto.
Diego Lizana	Ingeniero de datos	Encargado de extraer, transformar y cargar los datos de animes y valoraciones. Diseña y desarrolla pipelines de datos eficientes, realiza limpieza y normalización de datos, y garantiza la calidad y la integridad de la información utilizada en la plataforma.
Rodrigo Sepúlveda	Ingeniero de control de calidad	Responsable de verificar y validar la calidad de los datos en la plataforma. Realiza pruebas y controles para asegurar la precisión y consistencia de las valoraciones y características de los animes, y garantiza que cumplan con los estándares establecidos.
Walther Becks	Ingeniero de análisis de datos	Encargado de analizar los datos para obtener información relevante sobre las preferencias de los usuarios. Utiliza técnicas de análisis estadístico y visualización de datos para descubrir valoraciones de los animes, y proporciona información valiosa para mejorar la experiencia del usuario.
Elio Durán	Ingeniero de modelamiento	Diseña, desarrolla y evalúa modelos de estimación de valoraciones. Selecciona algoritmos de Machine Learning adecuados, entrena y ajusta los modelos con los datos disponibles, evalúa su rendimiento y los utiliza para predecir las valoraciones de los animes sin calificación.
Marco Cordero	Documentador	Responsable de documentar el proceso de desarrollo de la plataforma. Elabora informes, manuales de usuario y documentación técnica que describen los pasos realizados, las decisiones tomadas, los algoritmos utilizados y las métricas de evaluación. Mantiene la documentación actualizada y accesible.

Fuente: Microsoft Excel, elaboración propia

5. Planificación de la Investigación

5.1 Planificación

Se definirán los objetivos del proyecto, como construir un sistema de estimación de valoraciones de Animé y mejorar su precisión. Este es un problema de regresión, ya que se intentará predecir un valor y no una categoría. Estableceremos un cronograma con fechas límite para cada etapa del proyecto, desde la recopilación de datos hasta la implementación y evaluación del modelo. Dicho Cronograma está disponible en el anexo 11.1 de este documento.

5.2 Pregunta de investigación

¿Se puede desarrollar un sistema predictivo para estimar el puntaje de series y películas de Animé que carecen de valoraciones utilizando técnicas de aprendizaje automático, con el objetivo de implementar un modelo de negocios para comercializar a otros usuarios interesados?

5.3 Hipótesis de trabajo

"Si se utiliza un algoritmo de análisis de las características claves de series y películas de Animé, junto con técnicas de aprendizaje automático, es posible desarrollar un sistema de estimación de puntajes preciso que pueda predecir de manera efectiva el puntaje que una serie/película debería tener, incluso en ausencia de valoraciones, lo cual permitiría a la empresa crear una herramienta de valoración confiable y comercializable para otros usuarios interesados."

5.4 Definición del vector objetivo

El vector objetivo, representado por la variable 'rating', es el promedio de valoraciones de los miembros pertenecientes al "grupo" del anime. La finalidad es predecir este puntaje para los animes que no tienen valoración en la muestra. De manera adicional, y una vez que se haya escogido el mejor modelo, los datos faltantes en esta variable serán completados con las predicciones obtenidas.

5.5 Preprocesamiento de datos

Se utilizan métodos de pandas para buscar dentro de las columnas valores que difieran del formato np.nan. Luego se transforman a np.nan para convertirlos de forma masiva dentro del objeto pd.DataFrame.

Métodos de pandas utilizados para encontrar valores distintos a np.nan:

- .value_counts()
- .info()

- `.unique()`
- `.dtypes`

Métodos de pandas utilizados para encontrar valores np.nan:

- `.isnull().value_counts()`
- `.isna().value_counts()`

En conjunto a los métodos de pandas se implementan funciones creadas por el equipo:

- **nulos_en_variable** (df:pd.DataFrame, variable:str):
 - Ingresá un dataframe y la variable. Imprime la cantidad y porcentaje de nulos en dicha variable.
- **graficar_nulos**(df:pd.DataFrame, variable:str):
 - Ingresá un dataframe y la variable a graficar.
- **porcentaje_nulos_en_df**(data:pd.DataFrame, lista_nulos:list):
 - Ingresá una data y una lista de columnas para imprimir los porcentajes de np.nan por columnas, y el porcentaje total de líneas con al menos un np.nan dentro del DataFrame.

Cabe destacar que, tanto para el vector objetivo como para el resto de los atributos, se verifica el tipo de variable que corresponde y dependiendo de esa condición se hace determinado proceso como se muestra en la figura 1.

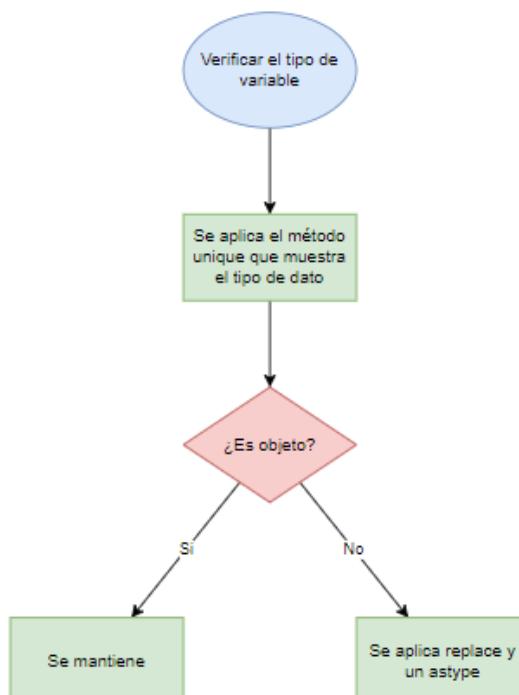


Figura 1: Procedimiento para variables del tipo objeto
Fuente: Visual Studio Code, elaboración propia

5.5.1 Procesamiento nulos

Hay presencia de nulos en las columnas ‘genre’, ‘type’, ‘episodes’ y ‘rating’. Para la columna ‘rating’ que es el vector objetivo, se separa un dataframe independiente donde se encuentren los valores perdidos, con el fin de predecir, una vez se tenga la herramienta de modelación predictiva creada.

Los datos nulos se distribuyen en porcentaje como se muestra en la tabla 2.

Tabla 2: Porcentaje de valores nulos en las variables

Variable predictoras	Porción de valores nulos
Rating	1,8708%
Género	0,3823%
Tipo	0,2034%
Episodios	2,7656%
Valores nulos	3,7742%

Fuente: Microsoft Excel, elaboración propia

Para los atributos se utilizan en conjunto los métodos <métodos>.fillna() y .mode() con la finalidad de imputar los valores por la moda, ya que ningún atributo supera el 5% de datos nulos. En caso de que el porcentaje de nulos hubiese sido mayor al 5% se procede a analizar e identificar una posible solución para cada caso, como una recodificación, por ejemplo. En nuestro caso ningún atributo superó el 5% de datos nulos El procedimiento se muestra en la figura 2.

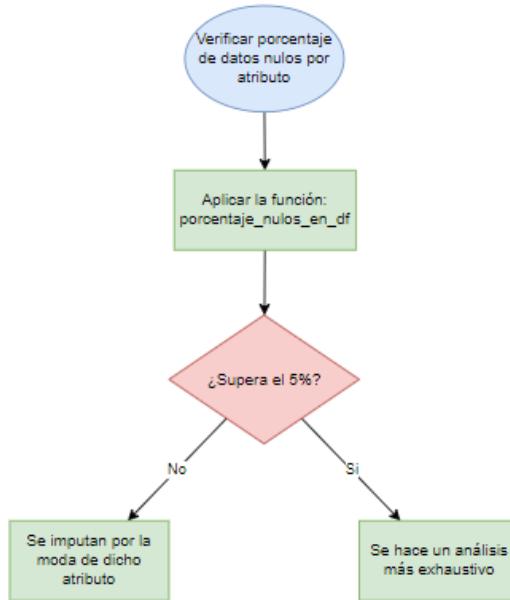


Figura 2: Procedimiento datos nulos
Fuente: Visual Studio Code, elaboración propia

5.5.2 Estandarización y transformación de los datos

En el análisis descriptivo se realizará el estudio estadístico de las variables dentro del dataframe.

Iniciaremos el estudio graficando las variables continuas que tenemos para observar su comportamiento por separado. Una vez obtenidas las gráficas procederemos a estandarizar las variables y volver a graficar. En este caso utilizaremos estos métodos para estandarizar:

- StandardScaler.
- RobustScaler.
- Logaritmo.
- Raíz Cuadrada.

Primero graficaremos las variables continuas sin estandarizar, para observar su comportamiento, estas variables son episodes (figura 3), members (figura 4) y rating (figura 5). De las figuras 3 y 4 concluimos que existe un evidente sesgo hacia los valores bajos, y de la figura 5 se observa una distribución de las observaciones similar a la normal, con una leve tendencia a puntajes sobre la media. Luego de observar las variables continuas sin estandarizar, se lleva a cabo los 4 procesos de estandarización que definimos con anterioridad. En los siguientes capítulos se aplicará un método similar con cada estandarización, cada uno de estos métodos tiene como punto de partida “df_clean”, que no es más que nuestro dataframe con las observaciones imputadas con la moda, como se explicó en el capítulo 5.5.1

Al finalizar este proceso (figura 6) se termina trabajando con 5 dataframes.

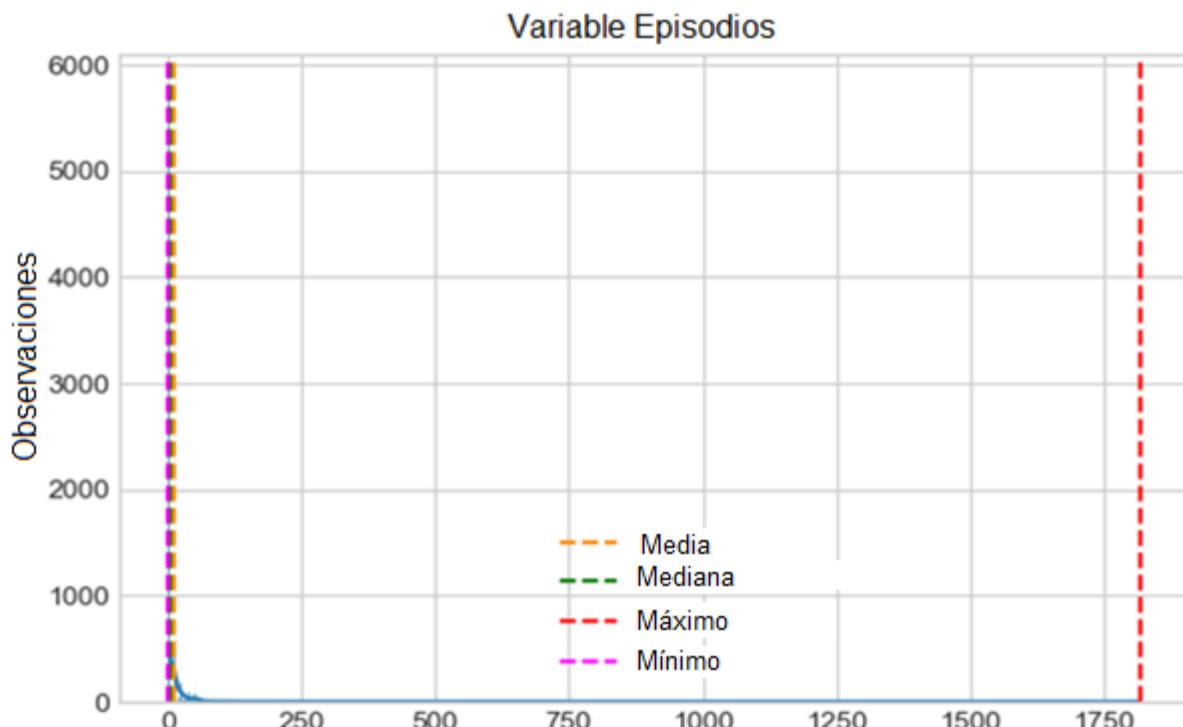


Figura 3: Gráfico episodios sin estandarizar
Fuente: Matplotlib, elaboración propia

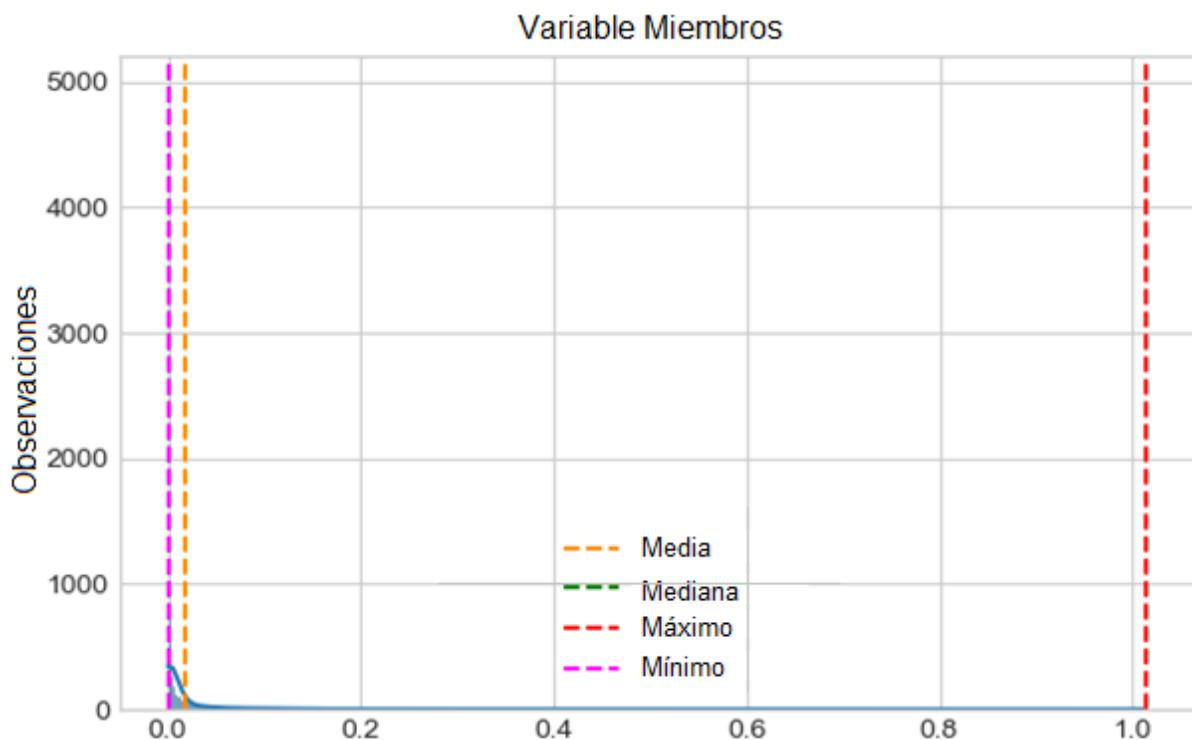


Figura 4: Gráfico miembros sin estandarizar
Fuente: Matplotlib, elaboración propia

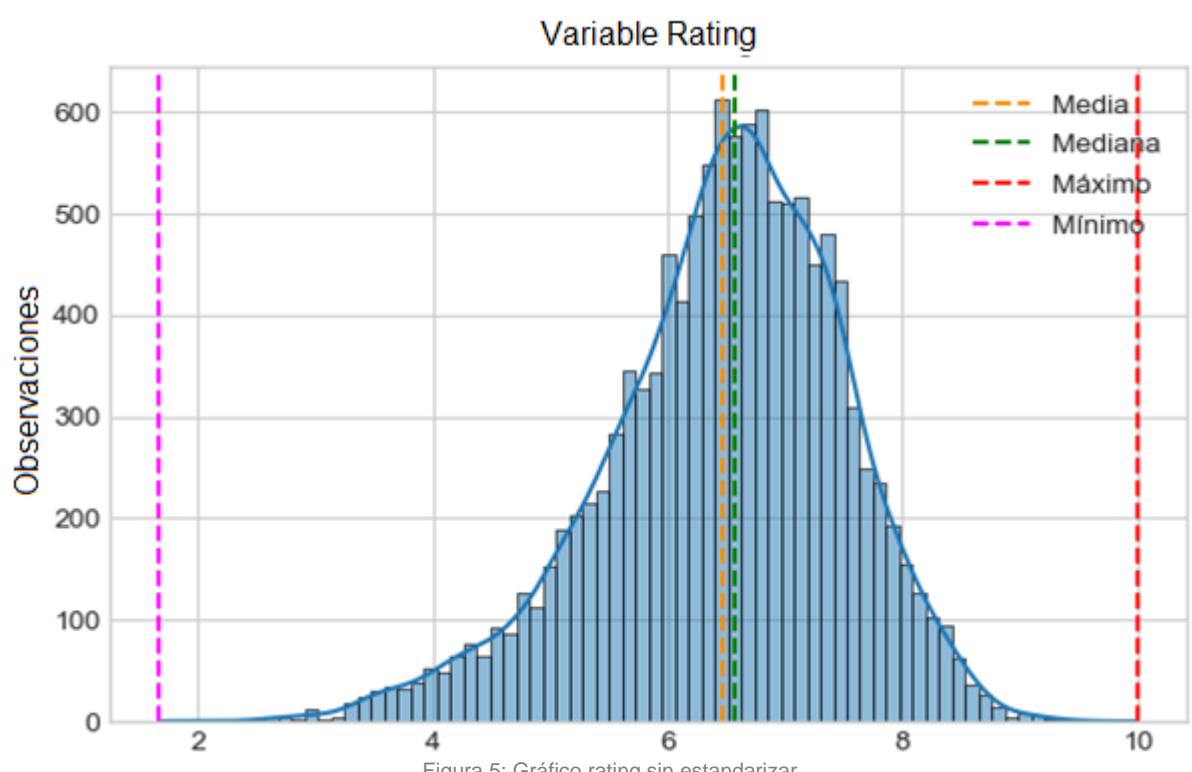


Figura 5: Gráfico rating sin estandarizar
Fuente: Matplotlib, elaboración propia

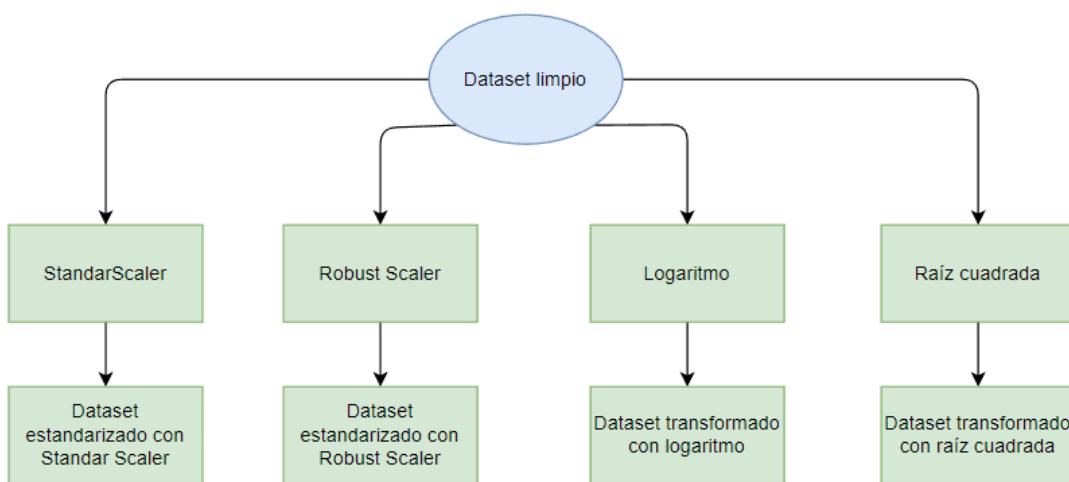


Figura 6: Proceso estandarizaciones
Fuente: Visual Studio Code, elaboración propia

5.5.2.1 StandardScaler

Su objetivo es estandarizar las variables de un conjunto de datos para que tengan una media de cero y una desviación estándar de uno. Esto se logra restando la media y dividiendo por la desviación estándar de cada característica.

En este caso se estandarizaron mediante el StandardScaler únicamente los atributos episodes y members, el resultado de este proceso se muestra en las figuras 7 y 8, respectivamente.

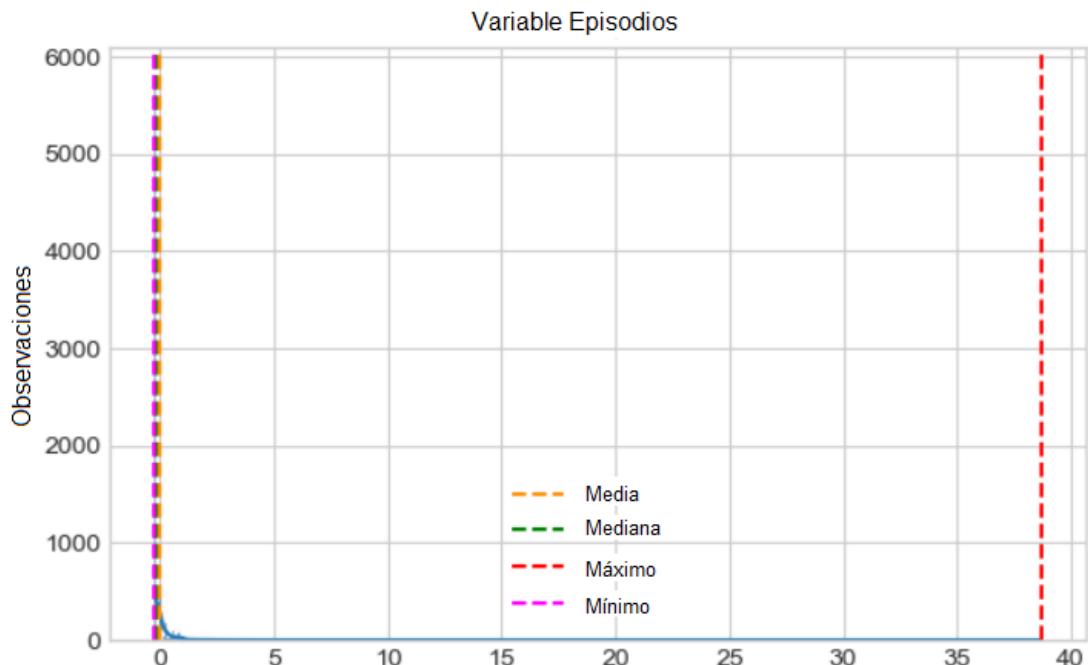


Figura 7: Gráfico episodios con standarscaler
Fuente: Matplotlib, elaboración propia

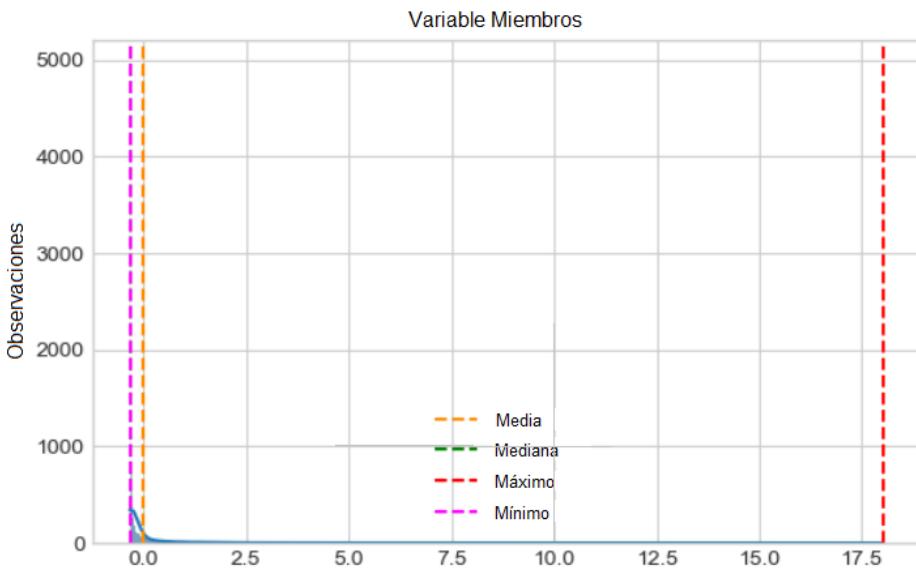


Figura 8: Gráfico miembros con standarscaler
Fuente: Matplotlib, elaboración propia

5.5.2.2 RobustScaler

Es una técnica de escala robusta que utiliza la mediana y el rango intercuartil para escalar variables, es útil cuando los datos contienen valores atípicos y se desea una escala más resistente a ellos en el preprocesamiento de datos para el aprendizaje automático. Esto se logra centralizando los datos restando la mediana a cada muestra, luego, escala las características dividiendo por el IQR (rango intercuartil, corresponde a la diferencia entre el 3er y 1er cuartil).

Al igual que en el capítulo 6.1.1 solo se estandarizaron las variables episodes (figura 9) y members (figura 10).

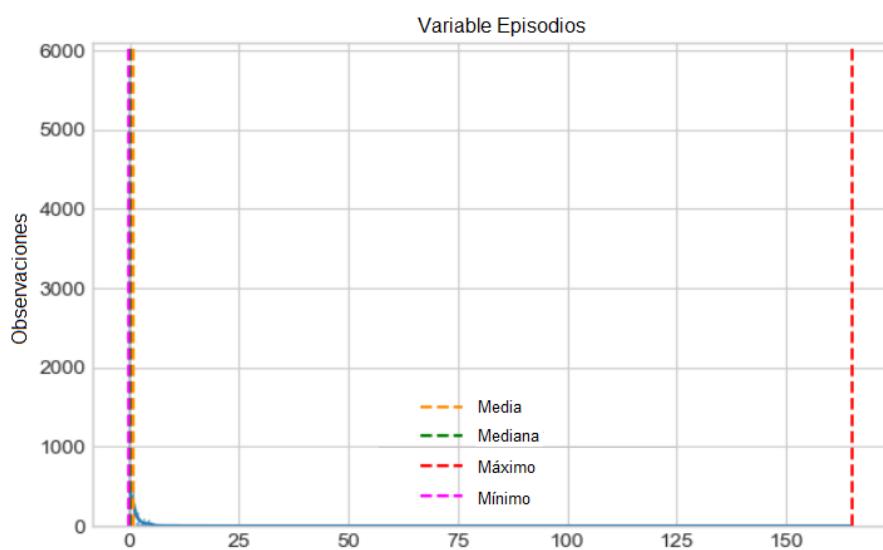


Figura 9: Gráfico episodios con robustscaler
Fuente: Matplotlib, elaboración propia

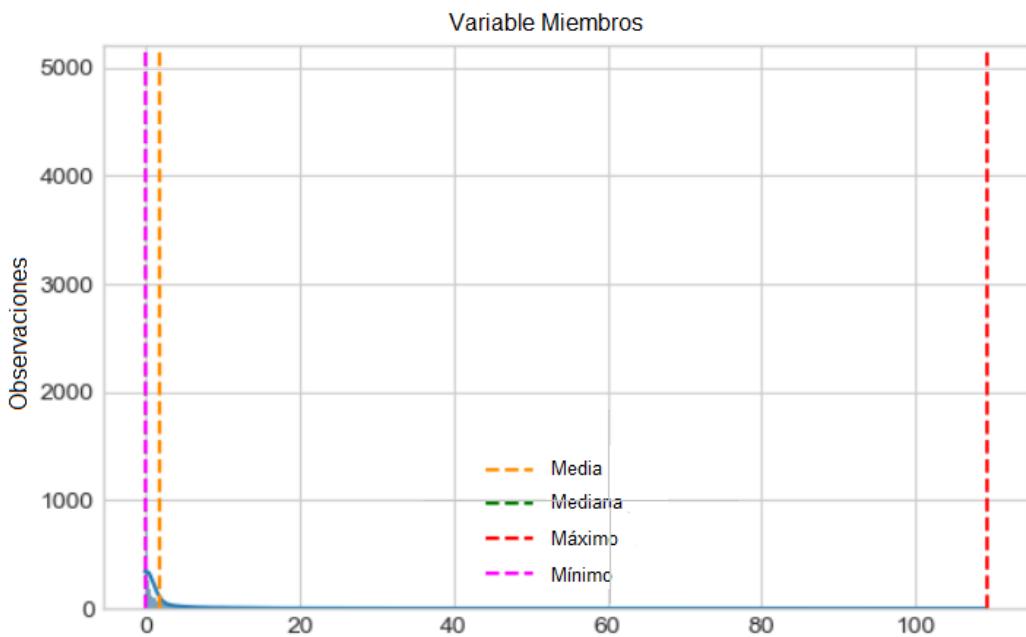


Figura 10: Gráfico miembros con robustscaler
Fuente: Matplotlib, elaboración propia

5.5.2.3 Transformación con logaritmo

Es un proceso en el cual los datos transforman sus características mediante la aplicación del logaritmo, este tipo de transformación es útil cuando se tienen los datos con una distribución sesgada o no sigue una distribución normal, ya que, al aplicar el logaritmo, se reduce la asimetría y se acerca la distribución a una forma más simétrica.

Siguiendo la tendencia solo se aplicará esta técnica de estandarización a los atributos episodes (figura 11) y members (figura 12).

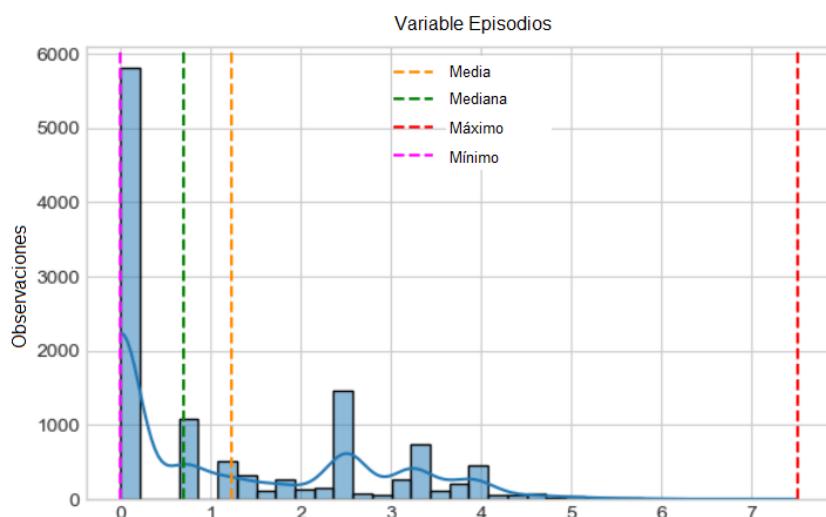


Figura 11: Gráfico episodios con logaritmo
Fuente: Matplotlib, elaboración propia

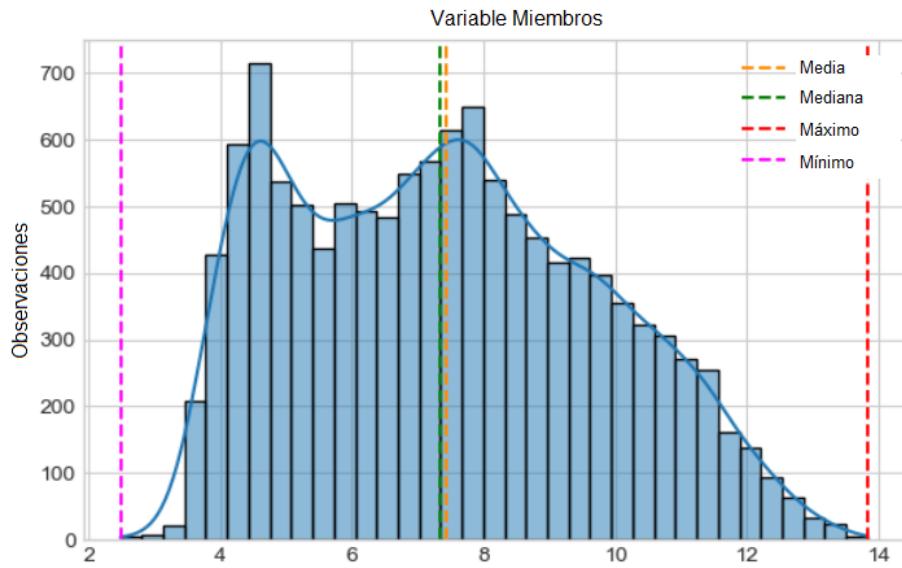


Figura 12: Gráfico miembros con logaritmo
Fuente: Matplotlib, elaboración propia

5.5.2.4 Transformación con raíz cuadrada

Es una técnica que aplica la función de raíz cuadrada a los valores de una característica con el objetivo de reducir la escala y suavizar la distribución de los datos. Esto es útil cuando se requiere una reducción de la influencia de valores altos o bien, una distribución más simétrica en el análisis.

Siguiendo la tendencia solo se aplicará esta técnica de estandarización a los atributos episodes (figura 13) y members (figura 14).

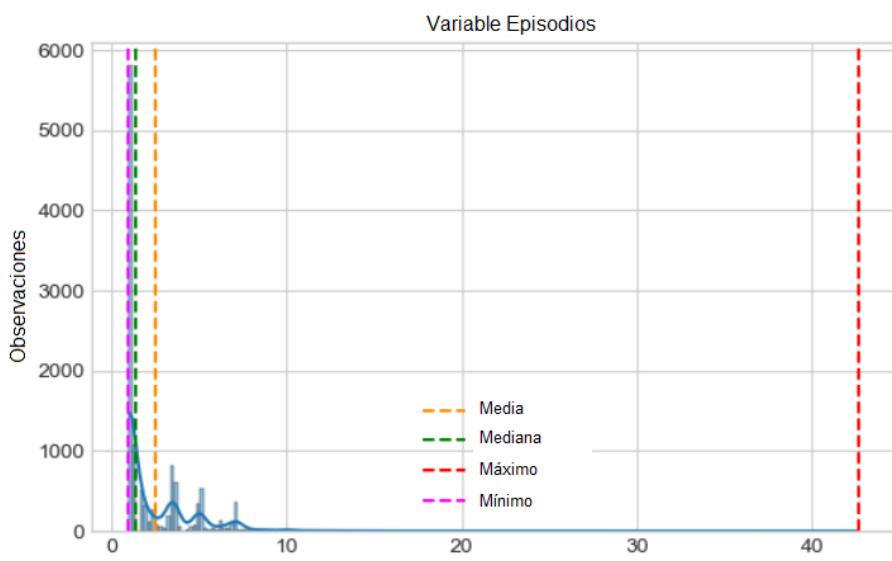


Figura 13: Gráfico episodios con raíz cuadrada
Fuente: Matplotlib, elaboración propia

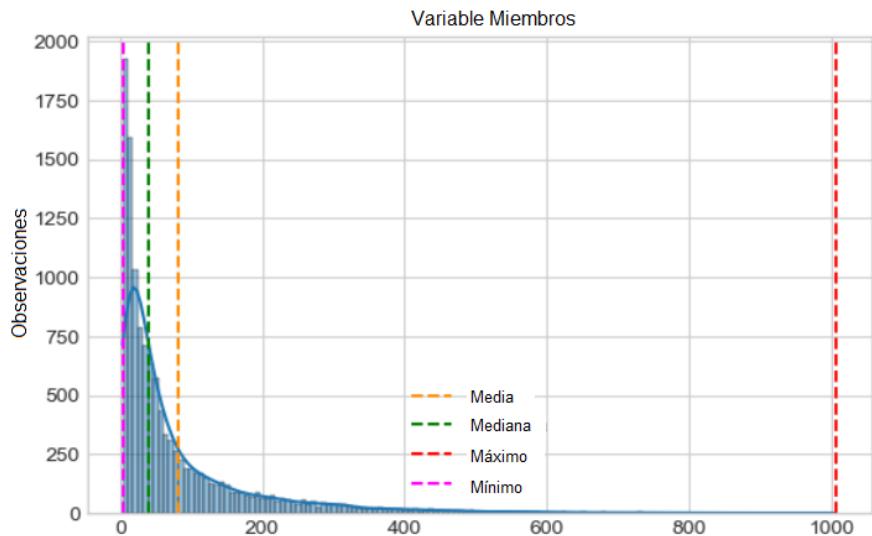


Figura 14: Gráfico miembros con raíz cuadrada
Fuente: Visual Studio Code, elaboración propia

5.5.3 Tratamiento de outliers

Para realizar el tratamiento de outliers fue necesario definir un criterio de selección. Para ellos nos basamos en lo siguiente:

- Aproximadamente el 68% de los datos se encuentran dentro de una desviación estándar de la media.
- Aproximadamente el 95% de los datos se encuentran dentro de dos desviaciones estándar de la media.
- Aproximadamente el 99.7% de los datos se encuentran dentro de tres desviaciones estándar de la media.

Se utilizaron 3 desviaciones estándar como criterio de selección. Como se muestra en la figura 15, a partir del tratamiento de los outliers de 5 dataframes, ahora estamos trabajando con 10.

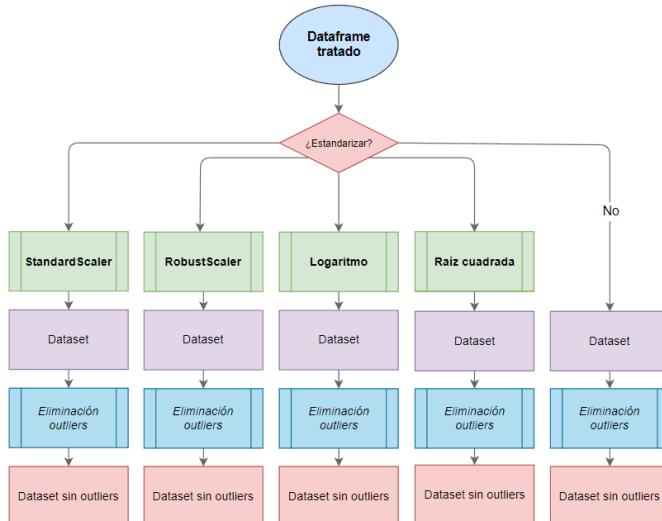


Figura 15: Proceso estandarizaciones sin outliers
Fuente: Visual Studio Code, elaboración propia

6. Análisis descriptivo

Este tipo de análisis es una metodología para poder resumir, organizar y presentar datos de manera significativa y comprensible. Se hizo un análisis inicial de las variables quebradas por tipo, como se aprecian en la figura 16, podemos observar que el rating promedio es de 6,5 aproximadamente, donde el tipo TV es el mejor valorado, seguido de OVA y movie, cabe mencionar que los 12,2k de observaciones están bastante distribuidas entre TV, OVA y movie.

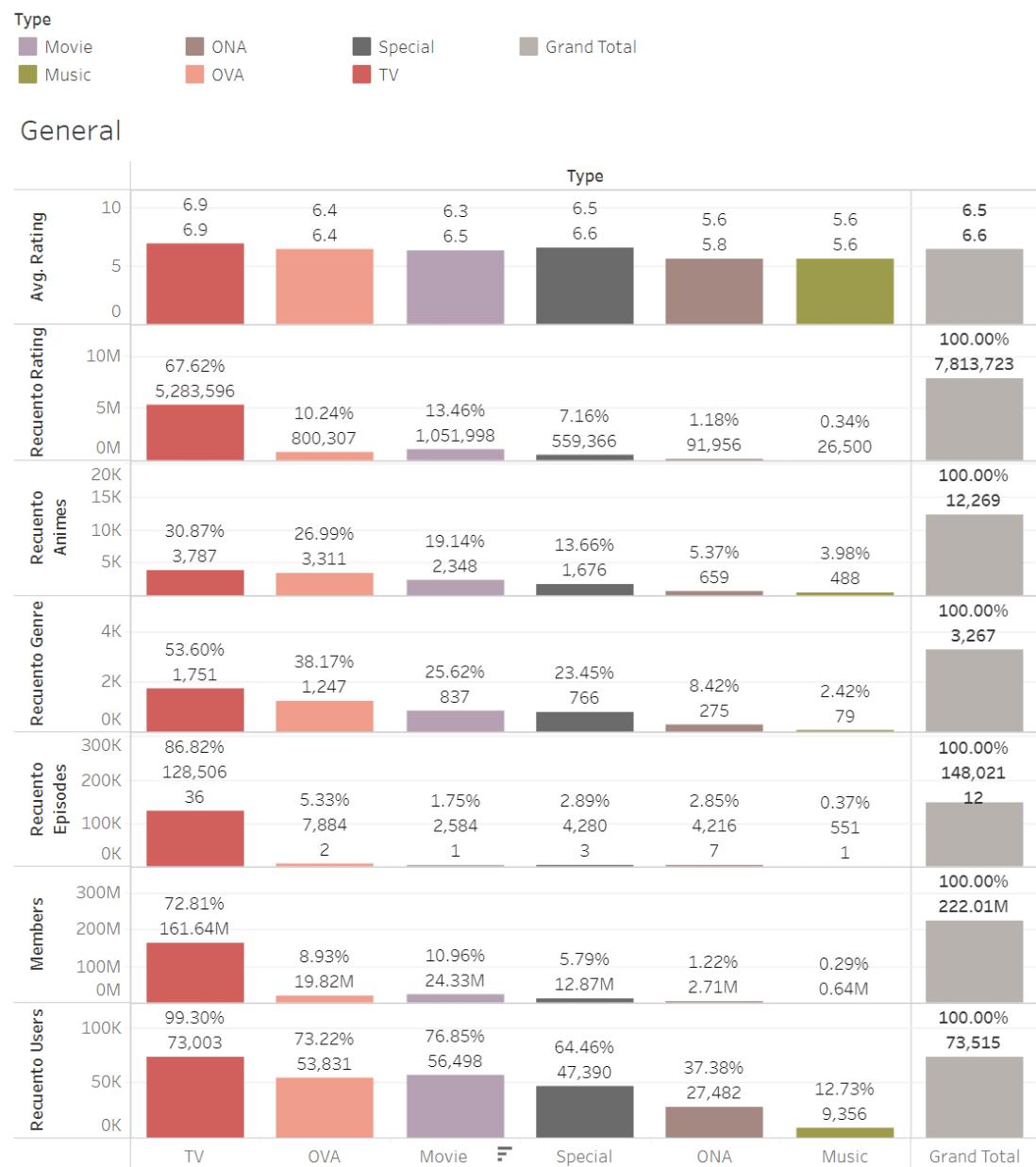


Figura 16: EDA
Fuente: public.tableau, elaboración propia

Además, del gráfico de la figura 16 podemos concluir lo siguiente:

- Rating cantidad: La mayor cantidad de rating (67,6%) es para TV, seguido de Movie. Dejando en claro que para el animé el tipo más fuerte es TV.

- Recuento animés: De un total de 12.2k de animés, la distribución es bastante pareja para TV, OVA y Movie.
- Recuento Genre: Podemos observar que hay una gran cantidad de géneros, debido a que un animé puede tener muchos tipos de géneros en un mismo animé, por lo que el número enorme que observamos es debido a la permutación de combinaciones, se verá este tema más adelante.
- Recuento episodes: Podemos ver que en total promedio hay 12 episodios por animé, y que para TV es 36, esto es un indicador de que su varianza es bastante grande, a diferencia de rating.
- Members: Podemos ver una enorme varianza, en donde la distribución de TV es abismalmente mayor que cualquier otro, confirmando los datos de Recuento Rating.
- Recuento users: Podemos observar que, de un total de 73.515 usuarios únicos, el 99,3% de ellos corresponden a TV, luego como segundo lugar tenemos que el 76,85% de ellos son de Movies.

La figura 17, representa el genre vs el rating, se puede observar cómo, inicialmente el género “hentai” es el top 1, pero como se mencionó antes, al haber miles de géneros, estos siendo combinaciones entre ellos, se produce esta equivocación de estimación.

En el gráfico de la figura 18, se puede observar que “hentai” equivale realmente al 3,31% y no está ni cerca de ser el top 1, en la misma figura se concluye que “comedy” es el género más común con un 12,84%.

Genero vs Rating

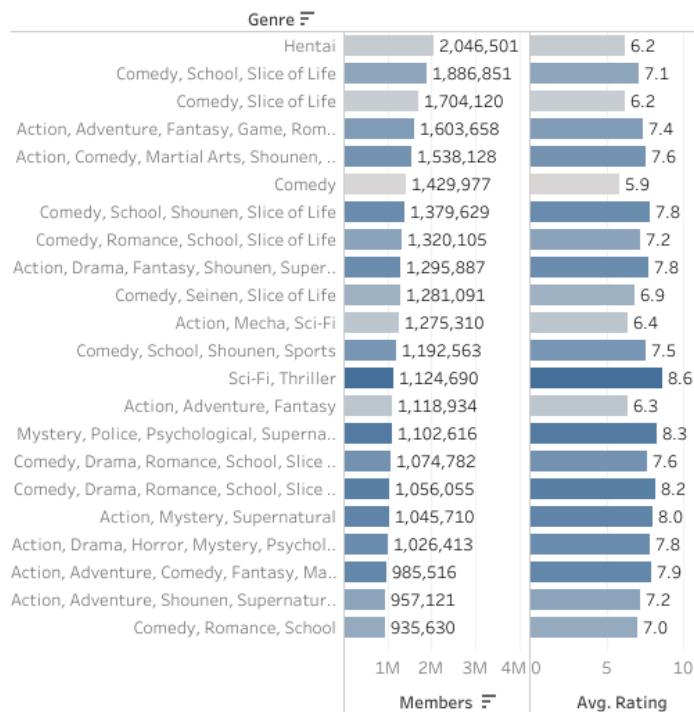


Figura 17: Gráfico género vs rating
Fuente: public.tableau, elaboración propia

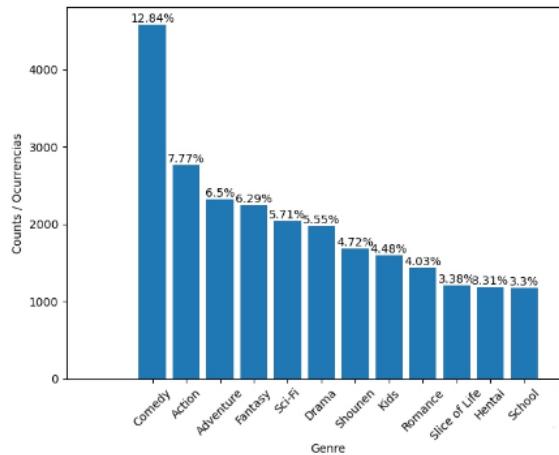


Figura 18: Extracto del gráfico genero sin combinar
Fuente: public.tableau, elaboración propia

En el siguiente heatmap (figura 19) tenemos el rating promedio quebrado por género de animé y tipo. Es interesante observar, que el promedio que se reportó en el análisis inicial, no se ve reflejado, acá podemos observar que el promedio de TV, por ejemplo, es más alto que el reportado de 6,9 pero debido a que hay algunos géneros con 0 de rating, estos ensucian la información. En el anexo 11.3 se encuentra un enlace para más información. Un elemento faltante de alta importancia, es la fecha de estreno y fecha de finalización del animé, ya que podemos ver que por ejemplo hay animés históricos, que han sido aclamados por la crítica pero estos no están en las primeras posiciones, debido a que la mayoría de ellos tienen décadas, por lo que es natural que con tanta gente diversa votando por ellas, hayan personas que no les guste, haciendo que el rating baje, en comparación con estrenos recientemente nuevos, por lo que al no estar estas variables de fechas, estamos sugiriendo que hay animés con rating inicialmente inflados, y que la fecha es una variable crucial para entender mejor el rating real de una obra.

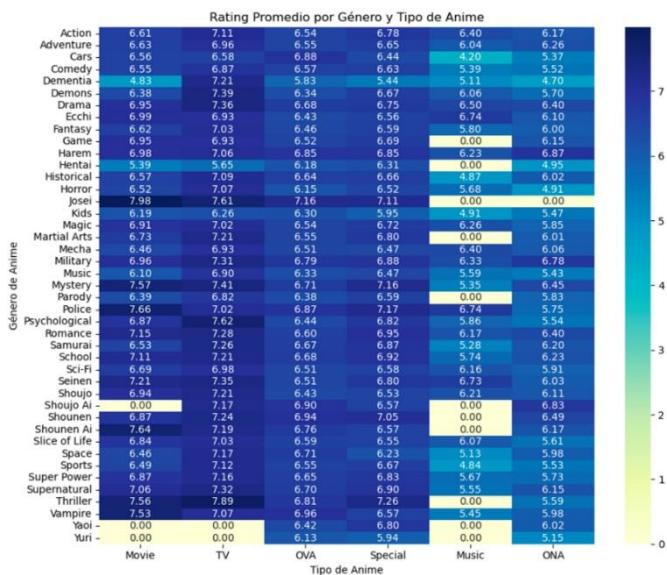


Figura 19: Histograma de la variable members
Fuente: public.tableau, elaboración propia

6.1 Análisis univariado

Este tipo de análisis es una técnica que se utiliza para analizar y comprender una única variable en un conjunto de datos, es decir, se examina una sola variable a la vez, sin considerar su relación con otras variables. El objetivo principal de este tipo de análisis es obtener una descripción detallada de la variable en estudio y explorar sus características básicas.

En el análisis univariado implica el uso de diferentes medidas estadísticas, como la media, mediana, moda y desviación estándar para resumir y describir las características de la variable, también se pueden utilizar gráficos, como histogramas, de dispersión y en el caso de este informe de boxplots. En la tabla 3 observamos las principales medidas estadísticas mencionadas anteriormente (media y desviación estándar) para los atributos episodios, miembros y rating. Además de boxplots, figura 20 y 21, en el que comparamos un dataset sin y con tratamiento de outliers, respectivamente. Se observa una clara disminución de los outliers en la figura 21, cabe destacar que el dataset utilizado en estas figuras corresponde a un conjunto de datos en el que las variables fueron transformadas mediante raíz cuadrada. El resto de los boxplots para los demás datasets utilizados se muestran en el anexo 11.5

Tabla 3: Medidas estadísticas de episodios, miembros y rating

Medida estadística	Episodios	Miembros	Rating
Media	2,5047	80,8112	6,4739
Desv. Std	2,4489	108,3976	1,0267

Fuente: Microsoft Excel, elaboración propia

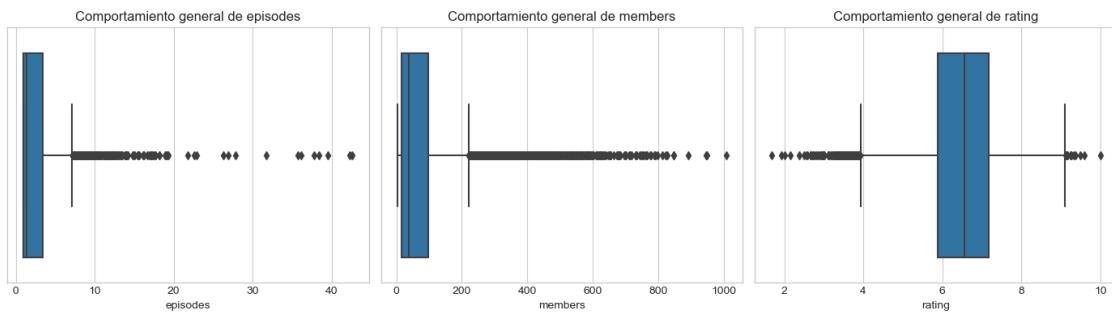


Figura 20: Boxplot del dataset sin tratamiento de outliers
Fuente: Matplotlib, elaboración propia

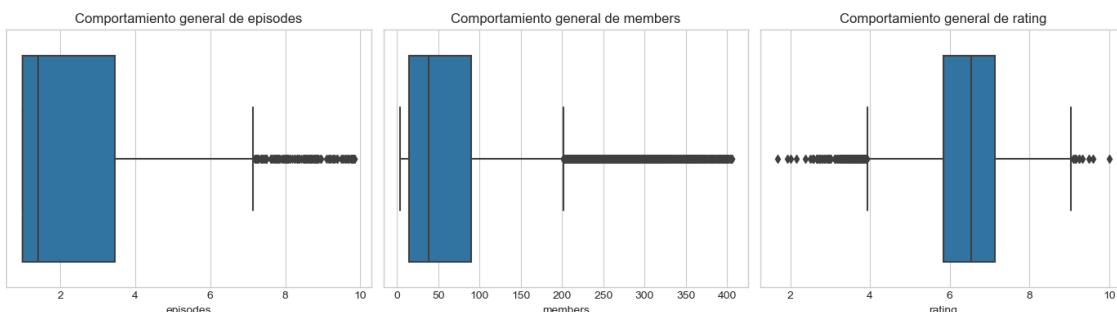


Figura 21: Boxplot del dataset con tratamiento de outliers
Fuente: Matplotlib, elaboración propia

6.2 Análisis multivariable

Después de estandarizar el dataframe utilizando los 4 métodos anteriormente explicados, podemos obtener las gráficas de dispersión para las variables continuas members y episodes con respecto al rating. Obtendremos las gráficas para los valores para escenario con outliers y escenario sin outliers.

A continuación, en las figuras 22 y 23 presentamos dos de las gráficas de dispersión disponibles que muestran la relación entre los atributos miembros vs rating y episodios vs rating, respectivamente, el resto se encuentran en el anexo 11.4. En este caso se muestra en la figura 22 el gráfico de dispersión en un dataset sin tratamiento de outliers y en la figura 23 con tratamiento de outliers utilizando en ambos casos un dataset con un conjunto de datos en que las variables fueron transformadas mediante raíz cuadrada. Se observa una correlación positiva en ambos casos, en cuanto a la tendencia se ajusta de mejor manera luego de tratar los outliers, dado que la recta capta mejor la distribución de los datos.

La figura 24 y 25 representan un boxplots considerando un dataset sin y con tratamiento de outliers, respectivamente y, el dataset se trata de un conjunto de datos en que las variables fueron transformadas mediante logaritmo. Se observa una clara mejora en la mediana de la muestra, para todas las categorías. La transformación logarítmica logra afectar de manera considerable la presencia de outliers, de hecho, con este dataset es el que mejor logra tratar los outliers. El resto de los datasets se mostrarán en el anexo 11.x

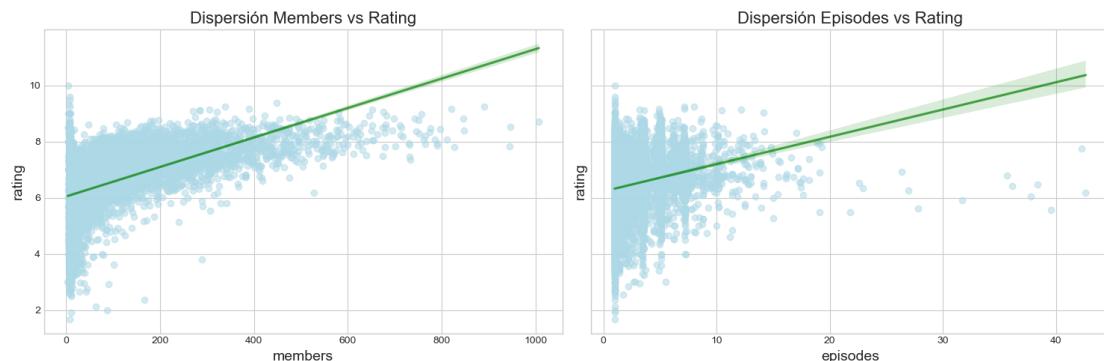


Figura 22: Gráfico de dispersión sin tratamiento de outliers
Fuente: Matplotlib, elaboración propia

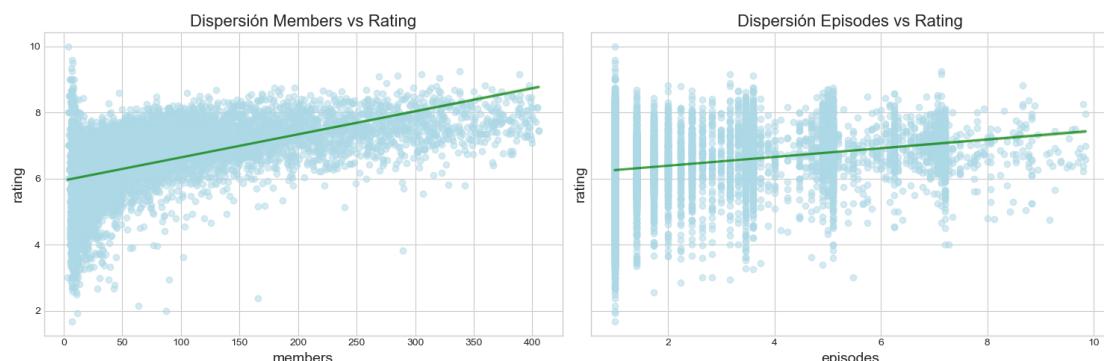


Figura 23: Gráfico de dispersión con tratamiento de outliers
Fuente: Matplotlib, elaboración propia

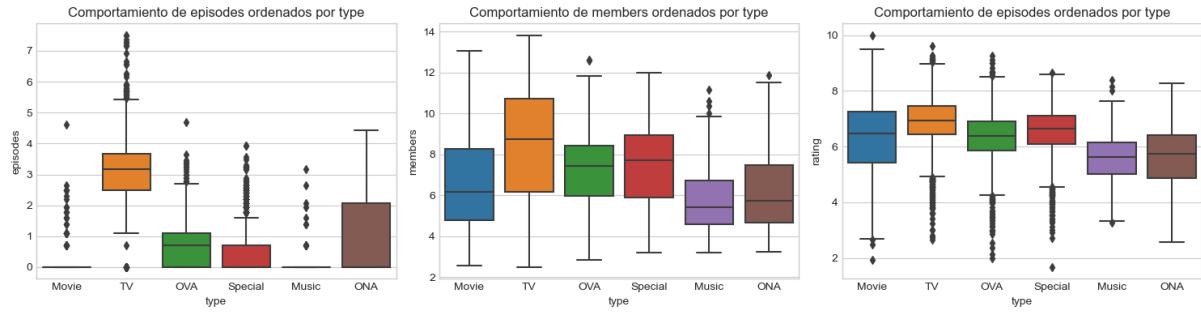


Figura 24: Boxplot sin tratamiento de outliers
Fuente: Matplotlib, elaboración propia

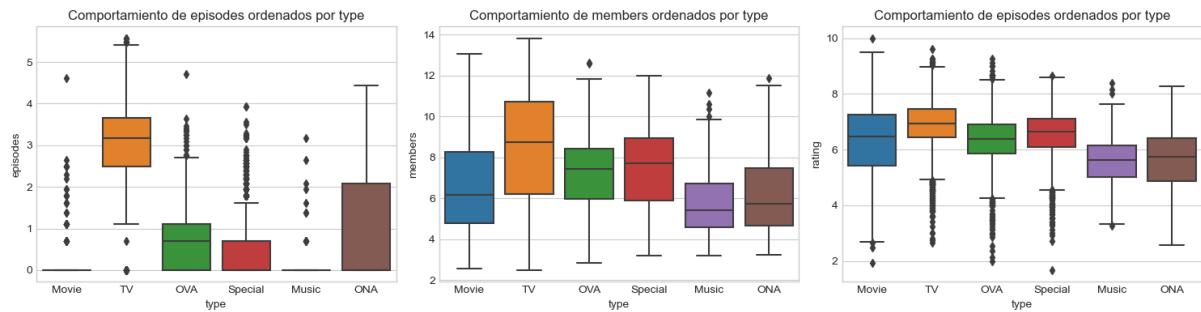


Figura 25: Boxplot con tratamiento de outliers
Fuente: Matplotlib, elaboración propia

6.3 Análisis de correlaciones

Un análisis de correlaciones es una técnica utilizada para explorar la relación entre dos o más variables en un conjunto de datos. El objetivo principal es determinar si existe una asociación estadística entre las variables y evaluar la fuerza y dirección de esa asociación.

A continuación, se muestra en la figura 26 una matriz de correlaciones entre las variables rating, members y episodes. Esta matriz considera un dataset con un conjunto de datos en el que las variables fueron transformadas mediante logaritmo. De manera análoga al capítulo 6.2 solo se muestra para dos datasets con (izquierda) y sin (derecha) tratamiento de outliers para comparar su diferencia. Se puede apreciar una fuerte correlación entre las variables miembros y rating con un coeficiente de 0.65, caso contrario sucede para las variables miembros y episodios con un coeficiente 0.27. No se observa una mayor diferencia de los coeficientes de correlación con respecto al dataset que tiene tratamiento de outliers (derecha) con el que no (izquierda). De todos los coeficientes de correlación, particularmente de este dataset, es el que las variables miembros y rating están más asociados. Para el resto de los datasets se muestran en el anexo 11.6

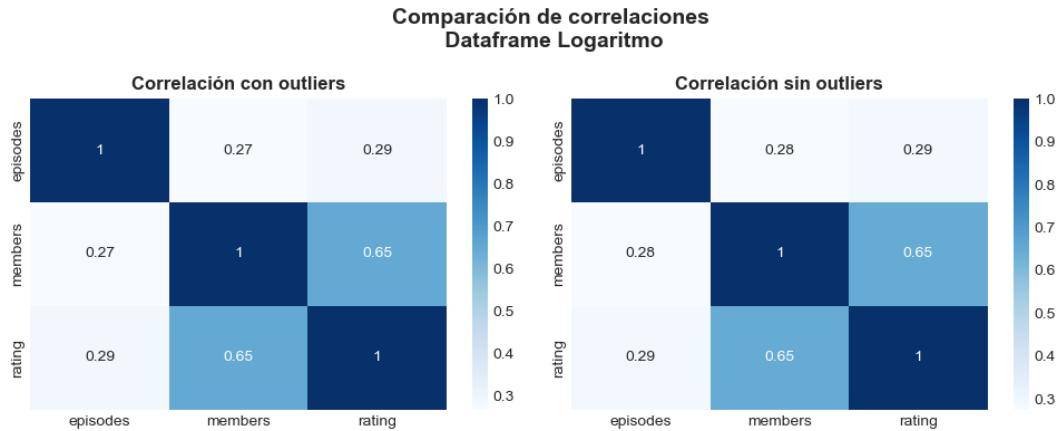


Figura 26: Matriz de correlaciones entre rating, episodes y members
Fuente: Matplotlib, elaboración propia

7. Creación de datasets para la modelación

Para este capítulo, crearemos un diccionario que contenga todos los datasets que serán utilizados para el modelamiento. Debemos recordar que los dataframes son una combinación del dataframe original y sus combinaciones dado el método utilizado para su estandarización y sus versiones antes y después del tratamiento de outliers.

Es importante destacar que para el modelamiento las variables type y genre deben ser llevadas a su versión dummies, por lo que para esos atributos se utilizó One-Hot Encoding (OHE). OHE es una técnica utilizada en el procesamiento para convertir variables categóricas en una representación numérica adecuada para su uso en algoritmos de aprendizaje automático.

Esta técnica lo que hace es convertir cada categoría única en una columna separada y asigna un valor binario (1 o 0) a indicar si una observación pertenece o no a una categoría específica.

Para la modelación tendremos los siguientes datasets:

1. df_clean: Dataframe que no considera ningún tipo de estandarización, solo limpieza de datos.
2. df_std: Dataframe en que sus variables continuas están estandarizadas con StandardScaler.
3. df_robust: Dataframe en que sus variables continuas están estandarizadas con RobustScaler.
4. df_log: Dataframe en que sus variables continuas están estandarizadas con logaritmo.
5. df_sqrt: Dataframe en que sus variables continuas están estandarizadas con la raíz cuadrada.
6. df_clean_sin_out: Dataframe con un tratamiento para disminuir los outliers y en que no considera ningún tipo de estandarización, solo limpieza de datos.

7. df_std_sin_out: Dataframe con un tratamiento para disminuir los outliers y que sus variables continuas están estandarizadas con StandardScaler.
8. df_robust_sin_out: Dataframe con un tratamiento para disminuir los outliers y que sus variables continuas están estandarizadas con RobustScaler.
9. df_log_sin_out: Dataframe con un tratamiento para disminuir los outliers y que sus variables continuas están transformadas con logaritmo.
10. df_sqrt_sin_out: Dataframe con un tratamiento para disminuir los outliers y que sus variables continuas están transformadas con la raíz cuadrada.
11. df_rating_log: Dataframe en que su vector y variables continuas están transformadas con logaritmo.
12. df_rating_sqrt: Dataframe en que su vector y variables continuas están transformadas con la raíz cuadrada.
13. df_rating_log_sin_out: Dataframe con un tratamiento para disminuir los outliers en que su vector y variables continuas están transformadas con logaritmo.
14. df_rating_sqrt_sin_out: Dataframe con un tratamiento para disminuir los outliers en que su vector y variables continuas están transformadas con la raíz cuadrada.
15. df_log_jerarquia: Dataframe en que su vector y variables continuas están transformadas con logaritmo, la variable type esta jerarquizada (no es dummies).
16. df_sqrt_jerarquia: Dataframe en que su vector y variables continuas están transformadas con la raíz cuadrada, la variable type esta jerarquizada (no es dummies).

Posterior a OHE se guardaron todos los dataframes en un diccionario llamado “dict_df_para_modelacion”, dicho diccionario contiene los 16 dataframes antes mencionados con su respectiva codificación OHE. Este diccionario se guardó en un pickle para su posterior utilización en los modelos.

Cabe destacar que la jerarquización de la variable type, se hizo de acuerdo con el rating, en la tabla 4 se muestra la jerarquización de la variable.

Tabla 4: Nivel de Jerarquización para type

type	rating	jerarquización
TV	6,9023	1
Special	6,5235	2
OVA	6,3752	3
Movie	6,3181	4
ONA	5,6433	5
Music	5,589	6

Fuente: Microsoft Excel, elaboración propia

8.1 Modelos a implementar

Los modelos que utilizaremos para modelar serán los siguientes:

- ElasticNetCV
- LinearRegression
- SVR
- DecisionTreeRegressor
- RandomForestRegressor
- AdaBoostRegressor
- GradientBoostingRegressor

A continuación, justificamos la razón de dicha selección de modelos:

- **Elastic Net:** Es un modelo que permite seleccionar características importantes y realizar una regularización que reduce la complejidad del modelo y evita problemas de sobreajuste. Elastic Net es especialmente útil cuando hay variables altamente correlacionadas en el conjunto de datos, ya que puede manejar la multicolinealidad y seleccionar características relevantes.
- **Regresión lineal:** Común para problemas de regresión, ya que permite modelar la relación lineal entre las características del Anime y el rating. Es adecuada cuando se asume una relación lineal entre las variables y no hay una complejidad excesiva en los datos. Además, es fácil de interpretar y entender.
- **Decision Tree Regressor:** El uso de Decision Tree Regressor en el modelamiento de un problema de anime proporciona un enfoque interpretable y flexible para comprender las relaciones entre las características y las calificaciones de los usuarios. Esto permite obtener información valiosa sobre los factores que influyen en la calidad percibida de los animes y puede servir como base para tomar decisiones informadas en la industria del anime y satisfacer las preferencias de los aficionados.
- **Random Forest Regressor:** Este modelo ofrece varias ventajas significativas. En primer lugar, Random Forest es capaz de manejar una amplia variedad de características y datos mixtos presentes en el conjunto de datos de anime, lo que permite capturar la diversidad de géneros, estilos de animación y otros atributos relevantes para el rating de los usuarios. Además, al combinar múltiples árboles de decisión, Random Forest tiene la capacidad de capturar relaciones no lineales y patrones complejos en los datos, lo que es crucial en un dominio como el anime donde las preferencias de los usuarios pueden ser altamente variadas. Además, Random Forest proporciona una estimación robusta y precisa del rating de los animes al reducir el sesgo y la varianza inherentes a los árboles de decisión individuales.
- **AdaBoosting:** Es un algoritmo de ensamble que combina varios clasificadores débiles para formar un modelo más robusto. Al ser capaz de aprender de los errores y enfocarse en las observaciones más difíciles, AdaBoosting puede mejorar la precisión de las predicciones en problemas complejos como el de calificación de animes.
- **Gradient Boosting:** Gradient Boosting es otra técnica de conjunto que mejora la precisión de la predicción mediante la combinación de múltiples modelos débiles. Es especialmente útil cuando hay relaciones complejas y no lineales en los datos. Al ajustar iterativamente los modelos en función de los errores anteriores, Gradient

Boosting puede aprender patrones más sutiles y mejorar el rendimiento de la predicción. Evaluaremos diferentes modelos utilizando métricas como el error cuadrático medio (MAE) o el coeficiente de determinación (R^2).

8.2 Modelación preliminar

En este capítulo se implementarán de manera preliminar los siete modelos antes mencionados, dichos modelos se aplicarán a cada uno de los 16 dataframes que se realizaron, dando un total de 112 modelos. En la tabla 5 se muestran los resultados de estos 112 modelos, de los cuales podemos concluir que las métricas del modelo de Regresión lineal dieron muy malas por lo que se decidió eliminar esos modelos, quedándonos con 60. Como se observa en la tabla 5, hay un recuadro marcado en rojo, que indica algunos resultados, en extremo deficientes, además que todos pertenecen al modelo de regresión lineal, por lo que se decide no trabajar con ese modelo.

Las métricas de medición que escogimos para comparar los modelos fueron:

- Error Absoluto Medio (MAE):** medida de la diferencia promedio entre los valores predichos por un modelo y los valores reales de la variable objetivo.
- R cuadrado (R-Square):** medida que indica la proporción de la varianza de la variable dependiente que es explicada por el modelo de regresión.

Tabla 5: Métricas de la modelación preliminar

Dataframes	Métricas	Modelos						
		Elastic Net CV	Linear Regression	SVR0	Decision Tree	Random Forest	Ada Boost	Gradient Boosting
dataset limpio sin tratamiento de outliers	R*	0.1458	0.3556	0.3905	0.3095	0.5368	0.2368	0.5633
	MAE	0.7293	0.6277	0.5946	0.6054	0.4994	0.6649	0.4933
	MAP	0.1243	0.1053	0.1042	0.1023	0.0848	0.1076	0.0847
dataset limpio con tratamiento de outliers	R*	0.1902	-2.31938E+12	0.3765	0.304	0.52	0.3628	0.5272
	MAE	0.7116	1.07E+05	0.6005	0.611	0.508	0.6257	0.5182
	MAP	0.1245	1.57E+04	0.1072	0.105	0.0885	0.1062	0.0908
dataset con sus variables estandarizadas con standard scaler y sin	R*	0.3508	-1.26E+21	0.5232	0.3029	0.5343	0.3118	0.5641
	MAE	0.6317	7.30E+09	0.5192	0.6095	0.5002	0.6773	0.4931
	MAP	0.1065	1.10E+09	0.0896	0.1024	0.0848	0.108	0.0847
dataset con sus variables estandarizadas con standard scaler y con	R*	0.3731	-4.18E+21	0.4913	0.3042	0.5175	0.3745	0.5267
	MAE	0.6191	1.05E+10	0.5229	0.6114	0.5034	0.6334	0.518
	MAP	0.1074	1.57E+09	0.0939	0.1055	0.089	0.1056	0.0908
dataset con sus variables estandarizadas con robust scaler y sin tratamiento de	R*	0.2344	-2.40E+19	0.5062	0.3045	0.5352	0.2725	0.564
	MAE	0.6612	5.01E+08	0.53	0.6043	0.5003	0.6999	0.4933
	MAP	0.1115	7.45E+07	0.0918	0.1017	0.0845	0.1109	0.0847
dataset con sus variables estandarizadas con robust scaler y con tratamiento	R*	0.3261	-2.37E+20	0.5167	0.3135	0.5201	0.3256	0.5263
	MAE	0.645	1.90E+09	0.5156	0.6066	0.5083	0.6665	0.5184
	MAP	0.1117	2.80E+08	0.0917	0.105	0.0887	0.1104	0.0908
dataset con sus variables transformadas con logaritmo y sin tratamiento	R*	0.5213	-3.70E+21	0.5694	0.2933	0.5327	0.3342	0.5642
	MAE	0.5283	4.60E+03	0.4899	0.6084	0.501	0.6632	0.4932
	MAP	0.0894	6.83E+08	0.0837	0.1028	0.0847	0.1007	0.0847
dataset con sus variables transformadas con logaritmo y con	R*	0.5084	-2.37E+22	0.5532	0.3446	0.545	0.3031	0.5474
	MAE	0.5393	2.20E+10	0.5002	0.5964	0.5022	0.6855	0.5133
	MAP	0.0931	3.30E+09	0.0853	0.101	0.0852	0.1211	0.087
dataset con sus variables transformadas con raíz cuadrada y sin tratamiento	R*	0.3266	-2.68E+18	0.4734	0.2936	0.5353	0.3717	0.5644
	MAE	0.6428	1.07E+08	0.5485	0.6091	0.4931	0.639	0.4931
	MAP	0.1094	1.62E+07	0.094	0.1022	0.0847	0.1052	0.0847
dataset con sus variables transformadas con raíz cuadrada y con	R*	0.3205	0.4166	0.4209	0.2814	0.4664	0.2596	0.5021
	MAE	0.6316	0.5748	0.5575	0.5396	0.5119	0.6865	0.5186
	MAP	0.1081	0.0972	0.096	0.1021	0.0875	0.1062	0.0885
dataset con el vector objetivo transformado con logaritmo y sin tratamiento	R*	0.4951	-4.06E+21	0.5534	0.2273	0.4953	-0.3368	0.5304
	MAE	0.0871	7.80E+08	0.0803	0.1005	0.0821	0.1639	0.0827
	MAP	0.0501	4.06E+08	0.0464	0.0577	0.0475	0.0322	0.0477
dataset con el vector objetivo transformado con logaritmo y con	R*	0.4809	-3.24E+22	0.5371	0.2807	0.5034	0.1015	0.5117
	MAE	0.089	4.05E+03	0.0822	0.0988	0.0828	0.1332	0.085
	MAP	0.0511	2.12E+03	0.0474	0.0568	0.0478	0.0727	0.0483
dataset con el vector objetivo transformado con raíz cuadrada y sin	R*	0.3035	-2.35E+18	0.4655	0.2734	0.5147	0.0768	0.5482
	MAE	0.1306	2.03E+07	0.1036	0.1224	0.101	0.1632	0.1014
	MAP	0.0538	7.98E+06	0.0454	0.0503	0.0417	0.0648	0.0417
dataset con el vector objetivo transformado con raíz cuadrada y con	R*	0.3007	0.3918	0.4192	0.2574	0.4732	0.2664	0.431
	MAE	0.1289	0.1179	0.1114	0.123	0.1037	0.1365	0.1048
	MAP	0.0531	0.0484	0.0461	0.0508	0.0443	0.055	0.0432
dataset con todas las variables transformadas con logaritmo, excepto	R*	0.4748	-2.82E+20	0.5434	0.2534	0.4943	-0.2198	0.5273
	MAE	0.0892	5.25E+08	0.0861	0.0984	0.0819	0.1557	0.0831
	MAP	0.0513	2.75E+08	0.0468	0.0566	0.0474	0.0648	0.0479
dataset con todas las variables transformadas con raíz cuadrada,	R*	0.3209	-1.0597	0.4754	0.2839	0.5165	0.0974	0.5461
	MAE	0.1294	9.25E+06	0.1093	0.1223	0.1007	0.1612	0.1016
	MAP	0.0531	3.55E+06	0.0451	0.0503	0.0415	0.0641	0.0418

Fuente: Microsoft Excel, elaboración propia

8.3 Elección de los dataframes y modelos candidatos

Posterior a la eliminación de este modelo, para la modelación final, se procede a buscar los mejores dataframes según el pre-proceso, como se muestra en la figura 27, adicionalmente se observa que el preprocesamiento de los datos mediante logaritmos logra mejores resultados que los demás, alcanzando un mayor r^2 y un menor MAE. En sus valoraciones promedio logaritmo compite solamente con su par sin outliers, sin embargo, por la valoración de tener datos conocidos dentro del entrenamiento, se opta por continuar y mantener los outliers, posterior al preprocesamiento mediante logaritmos. Además, a partir de los mejores dataframes se evaluaron los mejores modelos según sus métricas, como se ve en la figura 28. Dentro de los modelos con logaritmos, encontramos los siguientes mejores modelos: SVR, Gradient Boost, Random Forest y Elastic Net. Como conclusión los modelos SVR, Gradient Boosting, Random Forest y Elastic Net son considerados como los mejores y serán los llamados modelos candidatos. En conclusión, por los resultados se optó por utilizar el dataframe df_sqrt_jerarquico, ya que con ese dataset se obtuvieron las mejores métricas, según la tabla 6. Cabe señalar que los nombres de los dataset referidos en la tabla 6 están explicados en el capítulo 7.

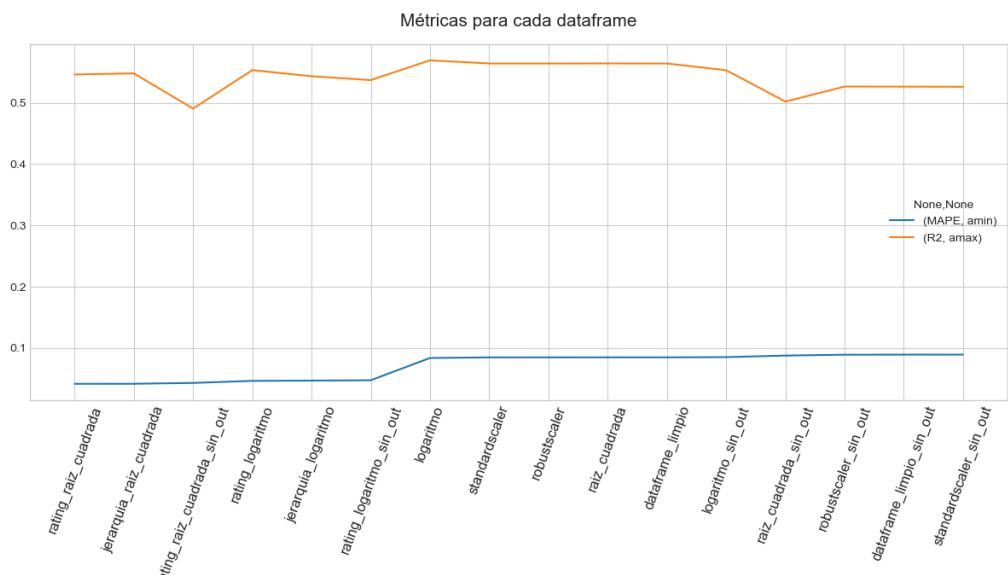


Figura 27: Métricas para cada dataframe
Fuente: Matplotlib, elaboración propia

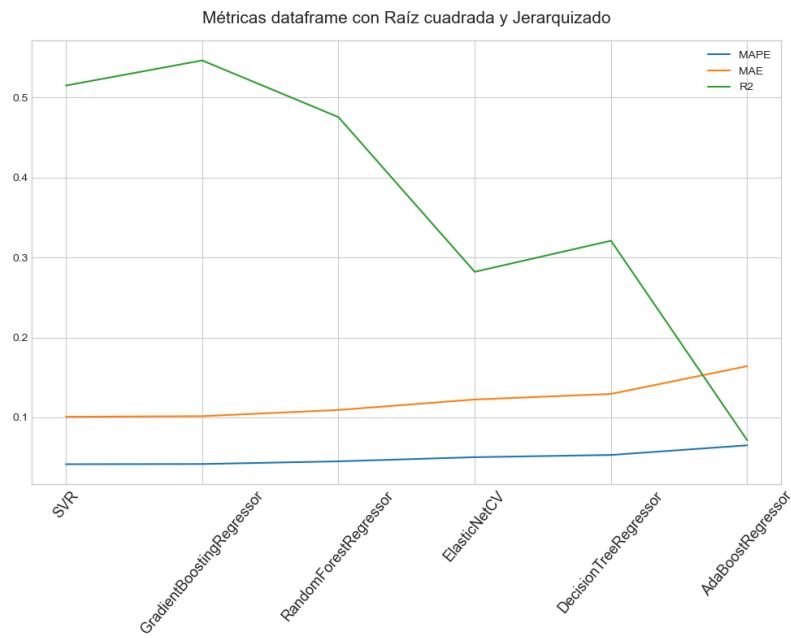


Figura 28: Métricas para dataset con raíz cuadrada y jerarquizado

Fuente: Matplotlib, elaboración propia

Tabla 6: Métricas de los datasets

Dataset	MAPE	R ²
dataframe rating raíz cuadrada	0,0415	0,5461
dataframe jerarquía raíz cuadrada	0,0416	0,5482
dataframe rating raíz cuadrada sin out	0,0427	0,4911
dataframe rating logaritmo	0,0464	0,5534
dataframe jerarquía logaritmo	0,0468	0,5434
dataframe ratinglogaritmo sin out	0,0474	0,5371
dataframe logaritmo	0,0837	0,5694
dataframe standard scaler	0,0847	0,5644
dataframe robust scaler	0,0847	0,5642
dataframe raíz cuadrada	0,0847	0,5642
dataframe limpio	0,0847	0,564
dataframe logaritmo sin out	0,0853	0,5532
dataframe raíz cuadrada sin out	0,0878	0,5022
dataframe robust scaler sin out	0,089	0,5265
dataframe limpio sin out	0,0891	0,5267
dataframe standard scaler sin out	0,0892	0,5269

Fuente: Microsoft Excel, elaboración propia

8.4 Voting Regressor

Voting regressor es un algoritmo de aprendizaje automático utilizado en tareas de regresión. Forma parte de los métodos de ensamble que combinan las predicciones de varios modelos en un solo estimador de regresión para obtener una predicción más precisa y robusta. Su predicción final se calcula como una agregación de las predicciones individuales de los modelos base. La idea detrás del voting regressor es que, al combinar las predicciones de varios modelos, se puedan compensar los errores individuales y obtener una estimación más precisa y generalizada.

En el caso de este proyecto se eligieron como modelos base, los mejores del capítulo 8.3; los cuales son SVR, Gradient Boosting, Random Forest y Elastic Net. En la tabla 7 se muestran los resultados del MAPE, MAE y R². En dicha tabla se aprecia que el comportamiento de voting regressor, considerando los modelos previamente mencionados, muestra ser similar a los modelos previamente testeados, por lo que se suma al abanico de modelos a estudiar, este comportamiento es esperado.

Tabla 7: Métricas del Voting Regressor

Métricas	Resultado
MAPE	0,0426
MAE	0,1034
R ²	0,5275

Fuente: Microsoft Excel, elaboración propia

Por lo que la selección de modelos candidatos quedaría así: SVR, Gradient Boosting, Random Forest, Elastic Net y Voting Regressor.

8.5 Red Neuronal

Las redes neuronales son un tipo de modelo computacional inspirado en el funcionamiento del cerebro humano, están diseñadas para aprender y reconocer patrones complejos a partir de datos, estas redes están compuestas por unidades llamadas “neuronas” interconectadas entre sí, organizadas en capas. En una red neuronal típica, hay una capa de entrada que recibe los datos de entrada, una o varias capas ocultas que procesan la información y una capa de salida que genera las predicciones o resultados deseados. Cada neurona en una capa está conectada a las neuronas de la capa siguiente mediante conexiones llamadas “pesos”.

En la tabla 8 se muestran los resultados de esta red neuronal utilizando la librería keras, como capa de entrada se agrega una capa densa de 32 neuronas con una función de activación “sigmoid” que lo que hace es comprimir los valores en un rango de 0 a 1. Como capa oculta se agrega una segunda capa densa con 16 neuronas y se utiliza la función de activación “softsign”, que produce una salida en el rango de -1 a 1. Como capa de salida se agrega una capa con solo una neurona, ya que es un problema de regresión y se espera una única salida numérica, no se especifica ninguna función de activación, ya que será una

capa lineal que produce una salida sin restricciones. Por último, se entrena con 10 épocas, lo que indica cuántas veces se iterará sobre el conjunto de datos completo durante el entrenamiento, además el tamaño del lote se establece en 32 lo que significa que se actualizarán los pesos después de procesar 32 muestras a la vez.

Tabla 8: Métricas de la red neuronal

Métricas	Resultado
MAPE	0,0419
MAE	0,1014
R ²	0,5373

Fuente: Microsoft Excel, elaboración propia

9. Segunda Iteración de los modelos candidatos

En este capítulo se tomarán los 5 modelos candidatos mencionados en el capítulo 8.3. Cada uno de estos modelos se modelarán con sus respectivas grillas utilizando GridSearch.

El GridSearchCV es una técnica que nos permite automatizar la búsqueda de la combinación óptima de hiperparámetros para nuestro modelo. Funciona al definir un conjunto de valores posibles para cada hiperparámetro que queremos ajustar. Luego, realiza una búsqueda exhaustiva en una cuadrícula de todas las combinaciones posibles de estos valores, evaluando el rendimiento del modelo en cada combinación. Para evaluar el rendimiento, el GridSearchCV utiliza una métrica de evaluación específica, como MAE, MAPE y R². Utilizando técnicas de validación cruzada, el GridSearchCV entrena y evalúa el modelo en diferentes subconjuntos del conjunto de datos para obtener una estimación más robusta del rendimiento.

9.1 SVR

En la tabla 9 se muestran los parámetros utilizados en la búsqueda por grilla de este modelo, estos parámetros son:

- Kernel: Especifica el tipo de función kernel utilizada.
- C: Parámetro de regularización que controla la penalización por errores.
- Epsilon: Controla el margen de error aceptable para las predicciones.
- Gamma: Controla la influencia de cada ejemplo de entrenamiento en el modelo.
- Shrinking: Especifica si se utiliza el enfoque de encogimiento en el modelo.
- Cache_size: Tamaño del caché en memoria para almacenar resultados.

Dentro de la misma tabla 9 se encuentra marcado en rojo los mejores parámetros, además en la tabla 10 se muestran los resultados de sus métricas obtenidas a partir del conjunto de datos de prueba.

Tabla 9: Parámetros del modelo SVR

Parámetro	Valor
kernel	rbf
C	[10, 50, 100]
epsilon	[0.2, 0.5, 0.9]
gamma	auto
shrinking	True
cache_size	[100, 300]

Fuente: Microsoft Excel, elaboración propia

Tabla 10: Métricas 2da iteración SVR

Métricas	Resultado
MAPE	0,0525
MAE	0,1303
R ²	0,3555

Fuente: Microsoft Excel, elaboración propia

En la siguiente figura 29, se muestra de manera ordenada cuatro gráficos, los cuales son:

- Concentración de errores: Este gráfico muestra la concentración de errores en diferentes rangos, en el eje X se representa el rango de valores del vector objetivo, y en el eje Y se muestra el error absoluto correspondiente, este gráfico proporciona una visión general de cómo se distribuyen los errores en relación con los diferentes rangos de valores del vector objetivo.
- Media del error: Este gráfico muestra la media del error en cada rango. En el eje X se representan los rangos de valores del vector objetivo y en el eje Y se muestra la media del error absoluto correspondiente. Este gráfico proporciona una medida resumida del error promedio en cada rango.
- Heatmap del error: Este gráfico muestra un mapa de calor del error promedio en función de los rangos de valores del vector objetivo. Estos rangos se muestran en el eje Y, mientras que los valores reales del vector objetivo se muestran en el eje X. Este gráfico proporciona una visualización más detallada del error promedio para diferentes combinaciones de valores reales y rangos de valores.
- Dispersión del error: Este gráfico muestra la dispersión del error absoluto en función de los valores reales del vector objetivo. Cada punto en el gráfico representa una instancia del conjunto de pruebas, donde el eje X muestra el valor real y el eje Y muestra el error absoluto correspondiente. Los puntos son agrupados por rangos de valores del vector objetivo, que se representan mediante diferentes colores. Este gráfico proporciona observar la relación entre los valores reales y los errores, y cómo se distribuyen en diferentes rangos.

Es importante destacar que estos mismos gráficos serán mostrados en cada una de las modelaciones, por lo que sólo en este capítulo se proporcionará una descripción.

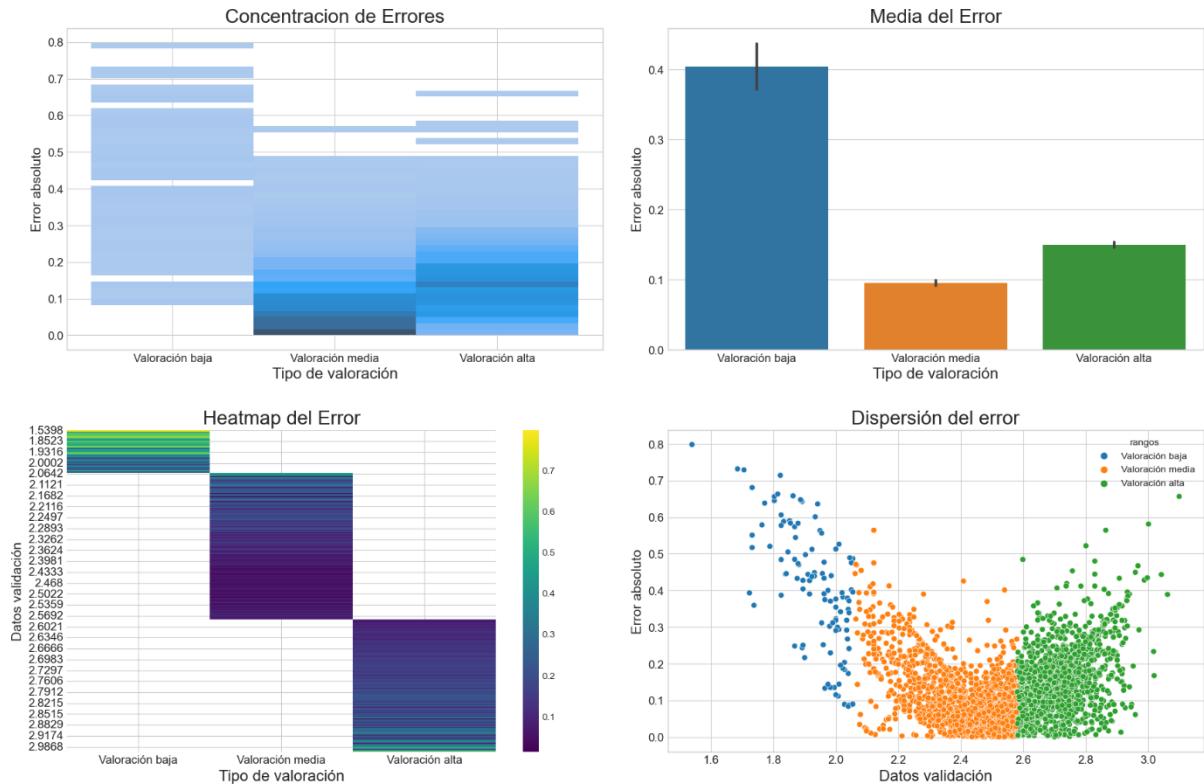


Figura 29: Gráficos en relación con el error en el modelo SVR
Fuente: Matplotlib, elaboración propia

9.2 Gradient Boosting

En la tabla 11 se muestran los parámetros utilizados en la búsqueda por grilla de este modelo, estos parámetros son:

- n_estimators: Especifica el número de estimadores a utilizar.
- learning_rate: Factor de aprendizaje que controla la contribución de cada árbol.
- subsample: Especifica la fracción de muestras para entrenar el modelo.

Dentro de la misma tabla 11 se encuentra marcado en rojo los mejores parámetros, en el caso del subsample el mejor parámetro fue 1.0, además en la tabla 12 se muestran los resultados de sus métricas obtenidas a partir del conjunto de datos de prueba y en la figura 30 se muestran los gráficos del error absoluto.

Tabla 11: Parámetros del modelo GB

Parámetro	Valor
n_estimators	[1, 10, 50, 100]
learning_rate	[0.01, 0.1, 0.5, 1, 5, 10]
subsample	np.linspace(0.1, 1.0, 10)

Fuente: Microsoft Excel, elaboración propia

Tabla 12: Métricas de 2da iteración GB

Métricas	Resultado
MAPE	0,0408
MAE	0,0993
R ²	0,5598

Fuente: Microsoft Excel, elaboración propia

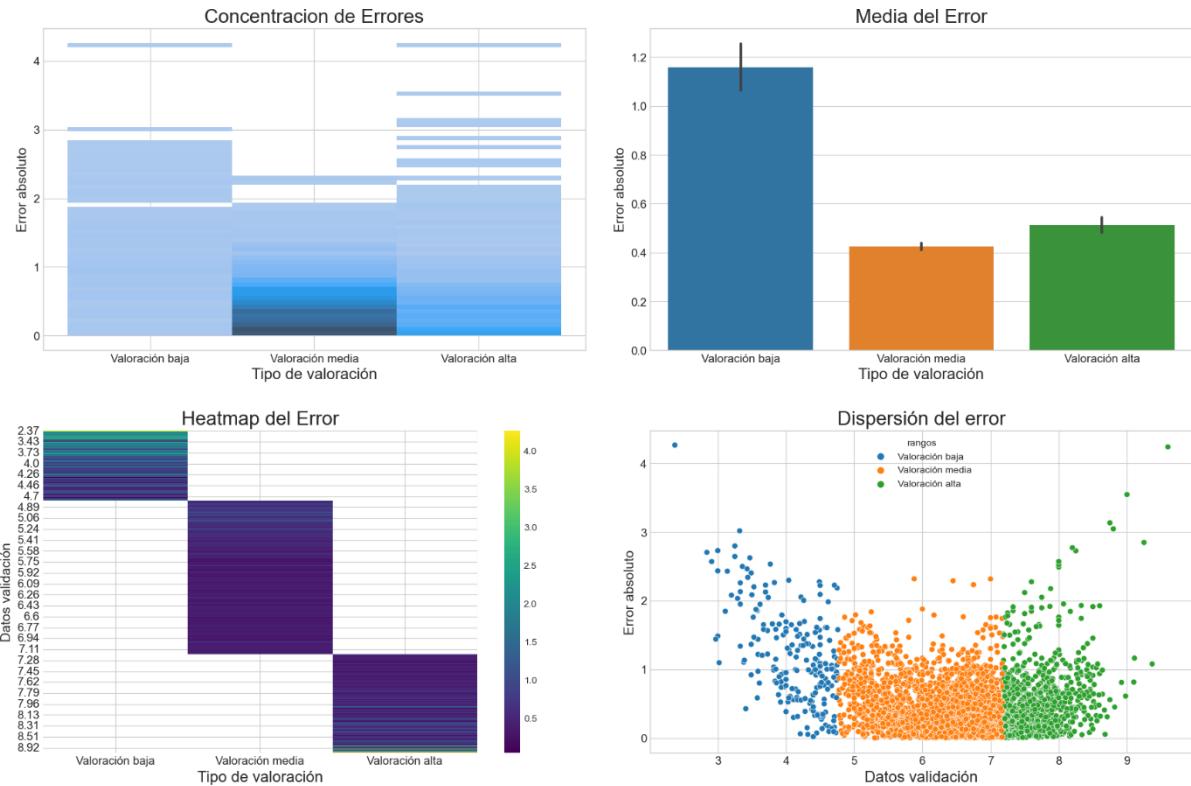


Figura 30: Gráficos en relación con el error en el modelo GB
Fuente: Matplotlib, elaboración propia

Además, con el modelo Gradient Boosting se puede realizar un gráfico que muestra la importancia de los atributos, dicho gráfico se muestra en la figura 31, en la cual se observa como el atributo más importante es “members”.

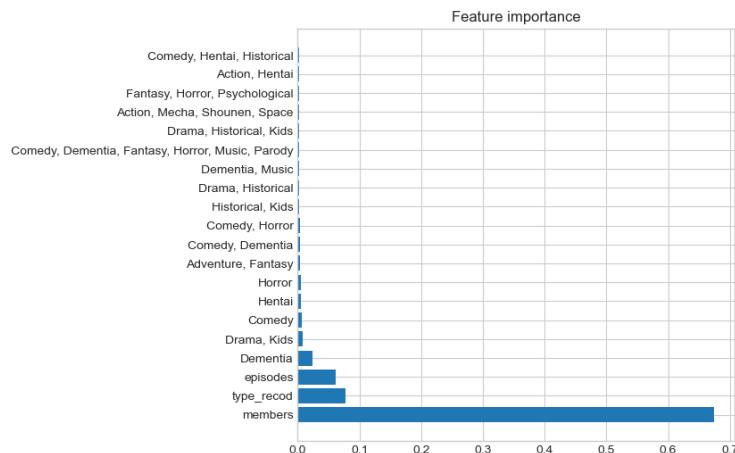


Figura 31: Feature importance
Fuente: Matplotlib, elaboración propia

9.3 Random Forest

En la tabla 13 se muestran los parámetros utilizados en la búsqueda por grilla de este modelo, estos parámetros son:

- `n_estimators`: Especifica el número de árboles de decisión a utilizar.
- `max_depth`: Especifica la profundidad máxima para cada árbol.
- `min_samples_split`: Especifica el número mínimo de muestras requeridas para dividir un nodo interno en un árbol.
- `min_samples_leaf`: Especifica el número mínimo de muestras requeridas en un nodo terminal de un árbol.

Dentro de la misma tabla 13 se encuentra marcado en rojo los mejores parámetros, además en la tabla 14 se muestran los resultados de sus métricas obtenidas a partir del conjunto de datos de prueba y en la figura 32 se muestran los gráficos del error absoluto.

Tabla 13: Parámetros del modelo RF

Parámetro	Valor
<code>n_estimators</code>	[10, 100]
<code>max_depth</code>	[1, 10]
<code>min_samples_split</code>	[0.001, 1]
<code>min_samples_leaf</code>	[0.001, 1]

Fuente: Microsoft Excel, elaboración propia

Tabla 14: Métricas 2da iteración RF

Métricas	Resultado
MAPE	0,0414
MAE	0,1008
R ²	0,5436

Fuente: Visual Studio Code, elaboración propia

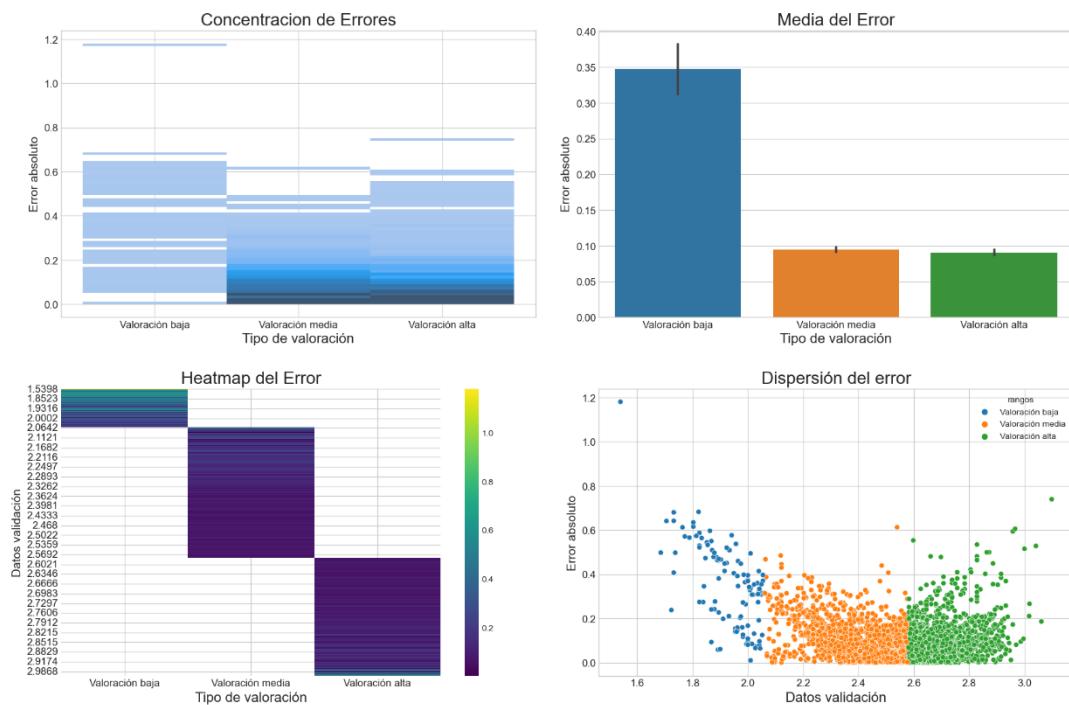


Figura 32: Gráficos en relación con el error en el modelo RF

Fuente: Matplotlib, elaboración propia

9.4 Elastic Net

En la tabla 15 se muestran los parámetros utilizados en la búsqueda por grilla de este modelo, estos parámetros son:

- **eps**: Especifica el valor de la tolerancia para el criterio de convergencia del modelo.
- **fit_intercept**: Especifica si se debe ajustar o no el intercepto en el modelo.
- **n_alphas**: Especifica el número de valores alfas que se generan automáticamente.
- **selection**: Especifica el esquema de selección de características en el modelo.

Dentro de la misma tabla 15 se encuentra marcado en rojo los mejores parámetros, además en la tabla 16 se muestran los resultados de sus métricas obtenidas a partir del conjunto de datos de prueba y en la figura 33 se muestran los gráficos del error absoluto.

Tabla 15: Parámetros del modelo EN

Parámetro	Valor
eps	[0.001, 0.01, 0.1]
fit_intercept	[True, False]
n_alphas	[100, 200, 300]
selection	[cyclic, random]

Fuente: Microsoft Excel, elaboración propia

Tabla 16: Métricas 2da iteración EN

Métricas	Resultado
MAPE	0,0482
MAE	0,1174
R²	0,4284

Fuente: Microsoft Excel, elaboración propia

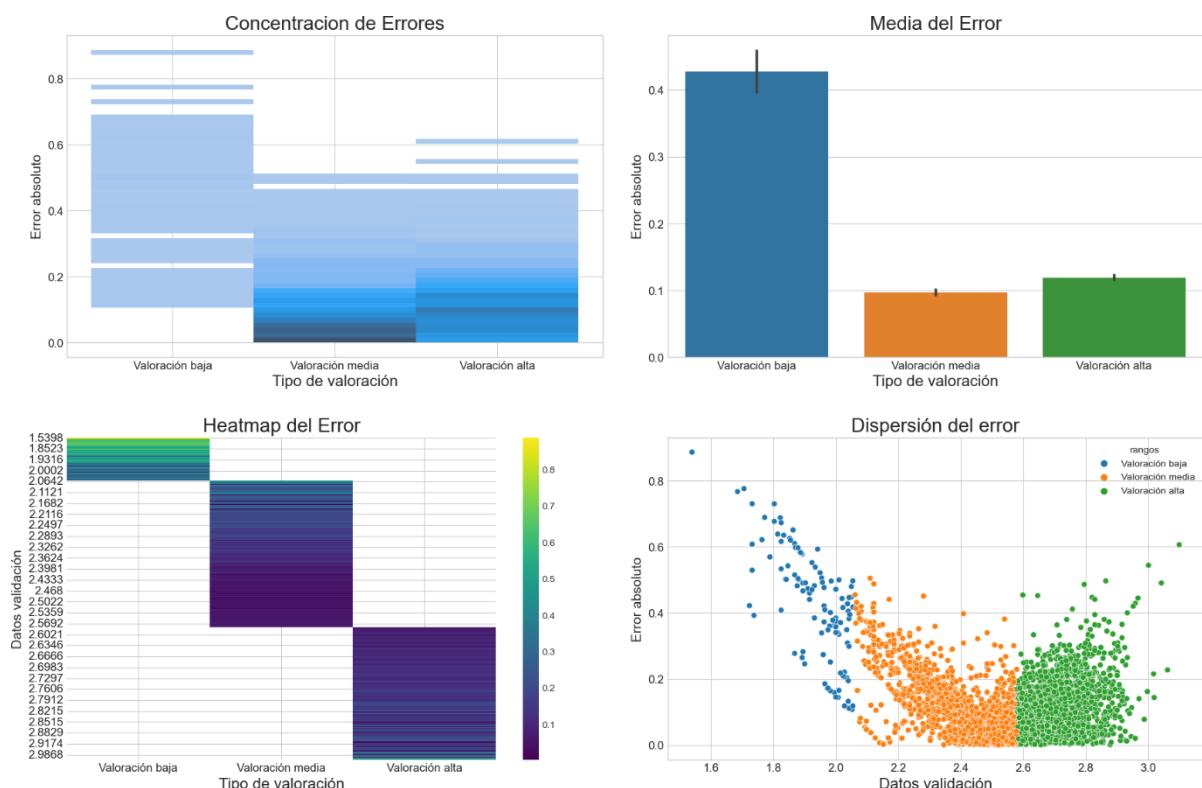


Figura 33: Gráficos en relación con el error en el modelo EN

Fuente: Matplotlib, elaboración propia

9.5 Voting Regressor

En la tabla 17 se muestran los parámetros utilizados en la búsqueda por grilla de este modelo, estos parámetros son:

- `eps`: Especifica el valor de la tolerancia para el criterio de convergencia del modelo.
- `fit_intercept`: Especifica si se debe ajustar o no el intercepto en el modelo.
- `n_alphas`: Especifica el número de valores alfas que se generan automáticamente.
- `selection`: Especifica el esquema de selección de características en el modelo.

Además, en la tabla 18 se muestran los resultados de sus métricas obtenidas a partir del conjunto de datos de prueba y en la figura 34 se muestran los gráficos del error absoluto.

Tabla 17: Parámetros del modelo VR

Parámetro	Valor
SVR	Mejores parámetros del modelo SVR
GB	Mejores parámetros del modelo GB
RF	Mejores parámetros del modelo RF
EN	Mejores parámetros del modelo EN

Fuente: Microsoft Excel, elaboración propia

Tabla 18: Métricas modelo VR

Métricas	Resultado
MAPE	0,0427
MAE	0,1039
R ²	0,5332

Fuente: Microsoft Excel, elaboración propia

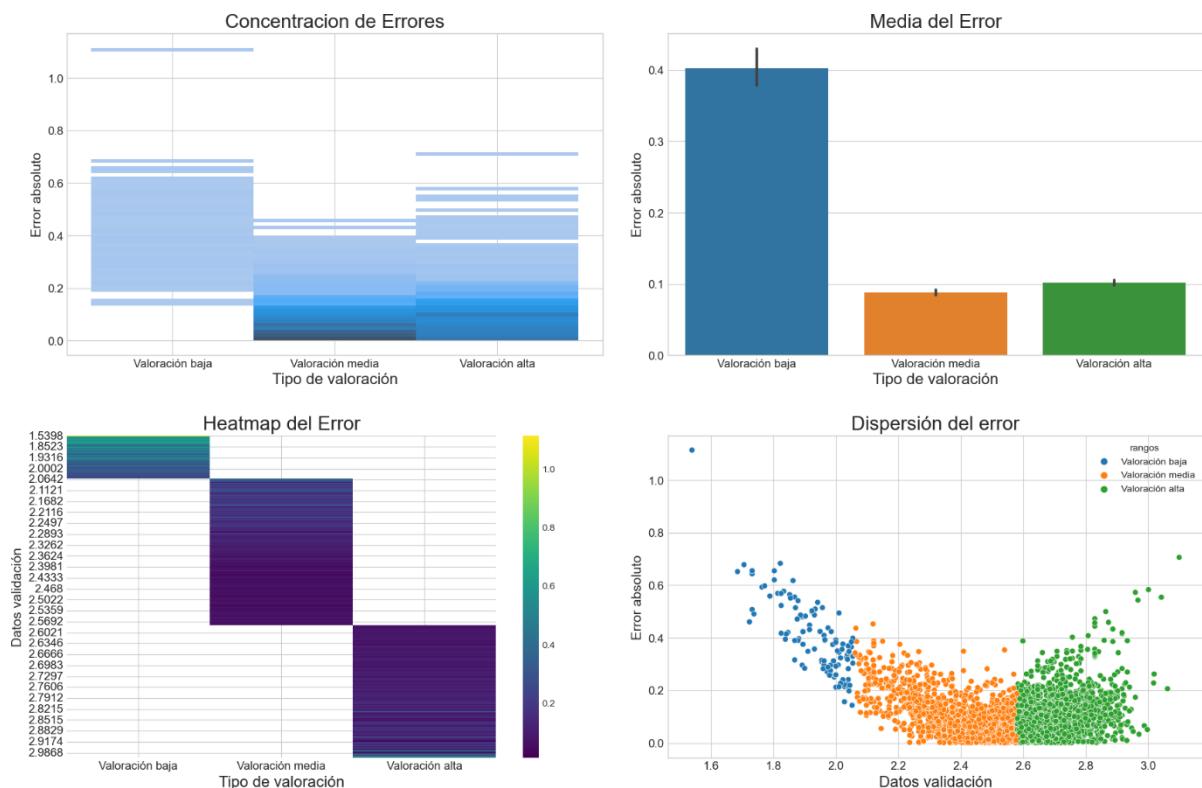


Figura 34: Gráficos en relación con el error en el modelo VR

Fuente: Matplotlib, elaboración propia

9.6 Elección mejor modelo

A continuación, en la figura 35 se presentan las métricas de los cinco modelos previamente calculados con el dataset denominado df_sqrt_jerarquico. Las métricas, principalmente el MAPE muestra un comportamiento estable en todos los modelos. Además, se puede observar que los modelos que utilizan árboles de decisión como base, muestran un mejor desempeño que los demás modelos. Como observación final, el mejor modelo se logra mediante Gradient Boosting Regressor con un error porcentual del 4%. Además, en la tabla 19 se muestra un resumen de las métricas de los 5 modelos utilizados, marcado en rojo el modelo con las mejores métricas, Gradient Boosting.

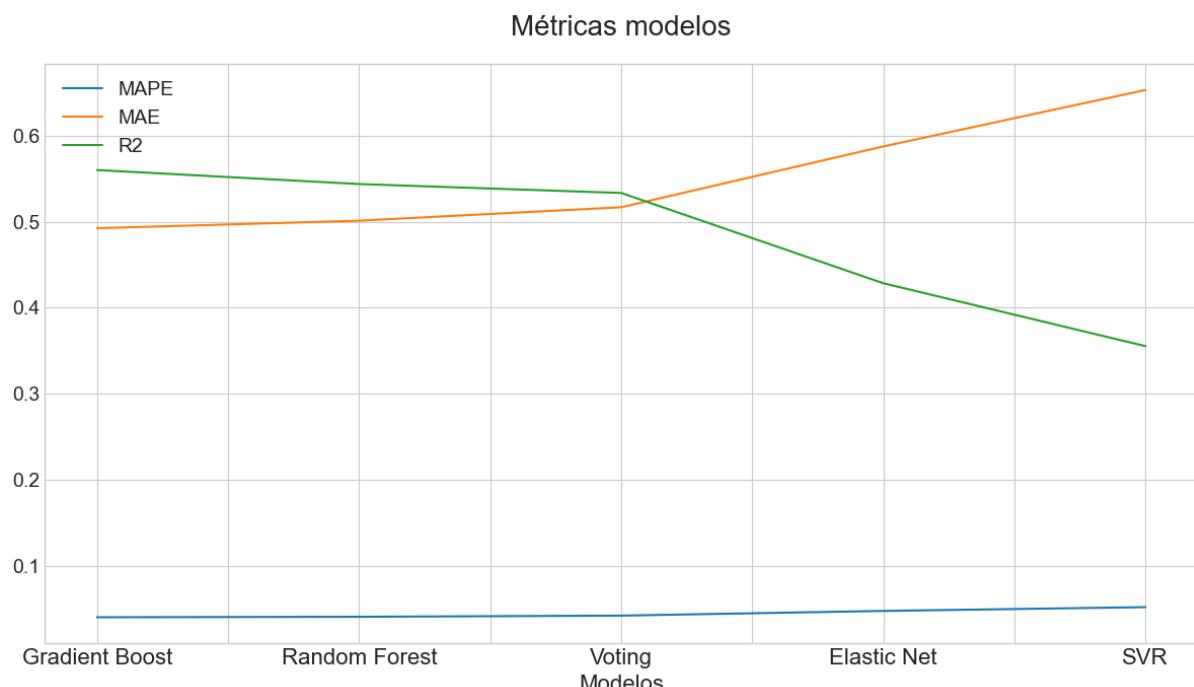


Figura 35: Métricas de los 5 modelos
Fuente: Matplotlib, elaboración propia

Tabla 19: Resumen métricas de los modelos

Modelos	Métricas					
	MAPE		MAE		R ²	
	Train	Test	Train	Test	Train	Test
SVR	0,0497	0,0525	0,615	0,6528	0,4598	0,3555
Gradient Boosting	0,0382	0,0408	0,4626	0,423	0,6407	0,5598
Random Forest	0,0395	0,0414	0,477	0,5011	0,6096	0,5436
Elastic Net	0,0431	0,0482	0,5204	0,5872	0,5446	0,4284
Voting Regressor	0,0424	0,0427	0,5109	0,5167	0,5609	0,5332

Fuente: Microsoft Excel, elaboración propia

10. Conclusiones

En la figura 36 se está representando una comparación de un dataframe sin ninguna transformación de logaritmo, raíz cuadrada ni de jerarquización del atributo type aplicándole un modelo gradient boosting sin hiperparámetros con el modelo que mencionamos anteriormente.

Vemos que se logra reducir el error porcentual en un 4% y a la vez el MAE se logra disminuir en un 1%, lo cual indica una clara mejora. El R2 y el MAE tienen una mejora muy leve del 0,4% y 0,7% respectivamente.

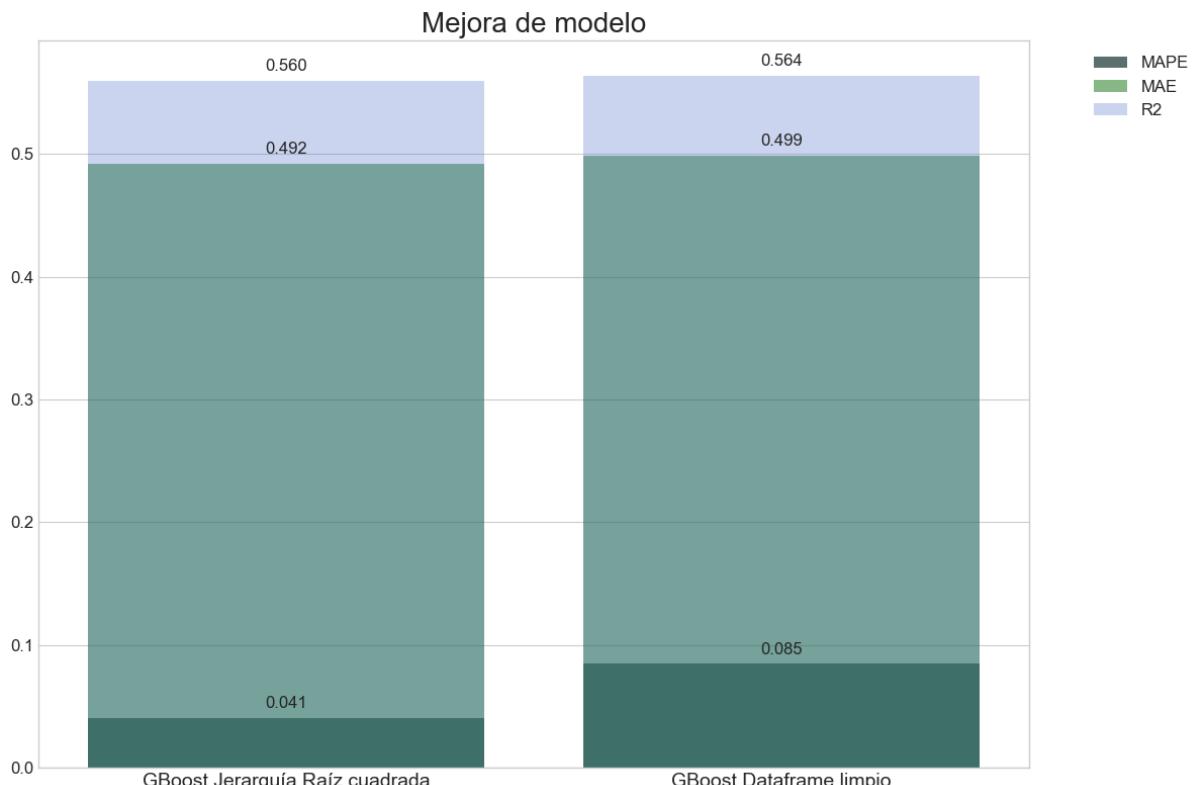


Figura 36: Comparación de modelos GB
Fuente: Matplotlib, elaboración propia

Recordando también la figura 31, del feature importance, que quiere decir la importancia de los atributos en el modelo, el atributo que más influye en el modelo es la cantidad de miembros que están asociados a ese anime en específico, la cantidad de episodios y el tipo de formato; luego estos 3 atributos vienen los géneros, que eso igual coincide con el análisis descriptivo que se realizó en el capítulo 6.

Al principio, se separó una matriz de prueba, en la cual había observaciones en el vector objetivo nulos, los cuales los separamos del dataframe, para que al momento de concluir cual sería nuestro mejor modelo, gradient boosting, completar esos datos faltantes con dicho modelo. La tabla 20 muestra un extracto de aquellas predicciones.

Tabla 20: Extracto Dataframe con predicción del rating

Nombre	Género	Tipo	Episodios	Rating	Predicción Rating
Inazma Delivery	acción, comedia	TV	10	Nulo	6,1816
Nananin no Ayakashi	comedia, super natural	TV	1	Nulo	5,6424
Gintama	acción, comedia	TV	1	Nulo	7,4607
One punch man 2	acción, comedia	TV	1	Nulo	7,3674
Steins gate	Sci-fi, thriller	TV	1	Nulo	7,2639
Nuki doki	Hentai	OVA	1	Nulo	5,7997
Sagurare otome	Hentai	OVA	1	Nulo	5,0001
Saimin Class	Hentai	OVA	1	Nulo	5,7269
Shikkoku no shaga	Hentai	OVA	1	Nulo	5,7269
Taimanin asagi 3	Demon, Hentai	OVA	1	Nulo	5,9649

Fuente: Microsoft Excel, elaboración propia

Además, en la figura 37 se muestran, según las predicciones, los 10 mejores y los 10 peores animés de la data faltante

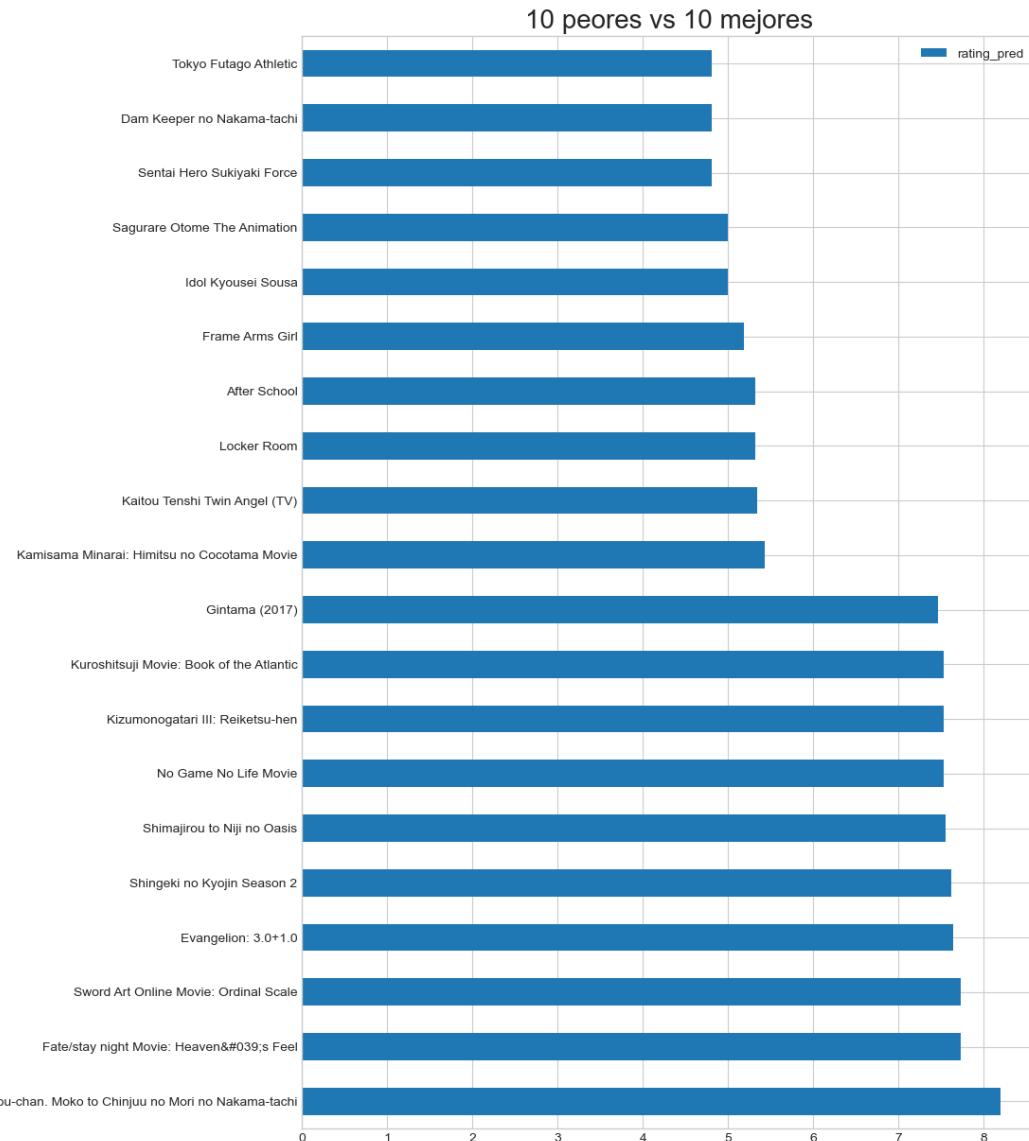


Figura 37: 10 peores vs 10 mejores animés
Fuente: Matplotlib, elaboración propia

10.1 Árbol de decisiones

Un árbol de decisión es una técnica utilizada para tomar decisiones o predecir resultados en base a un conjunto de características o variables. Se representa en forma de un diagrama en forma de árbol, donde cada nodo representa una característica o atributo, cada rama representa una posible opción o valor de esa característica, y cada hoja del árbol representa una decisión o resultado final. El árbol de decisión se construye a partir de un conjunto de datos de entrenamiento, donde se busca encontrar la mejor manera de dividir los datos en diferentes ramas, de modo que se maximice la homogeneidad dentro de cada rama y se maximice la diferencia entre las ramas. Este proceso se repite recursivamente en cada subconjunto de datos hasta alcanzar ciertos criterios de parada, como un número máximo de niveles o una pureza mínima en las ramas. Una vez construido el árbol, se puede utilizar para tomar decisiones o hacer predicciones sobre nuevos datos. Al seguir el camino desde la raíz hasta una hoja del árbol, se aplican las reglas definidas por los nodos en el camino y se llega a una conclusión o decisión final. Los árboles de decisión son ampliamente utilizados en la ciencia de datos debido a su interpretabilidad, facilidad de uso y capacidad para manejar datos numéricos y categóricos. También pueden manejar conjuntos de datos grandes y complejos. Sin embargo, pueden ser propensos al sobreajuste si no se controla correctamente, y a menudo se utilizan técnicas de poda y regularización para evitar este problema.

Para el proyecto se realizaron dos árboles de decisiones, uno considerando todos los atributos, figura 38, y un 2do árbol que no considera los atributos ni episodios ni miembros, que corresponde a la figura 39.

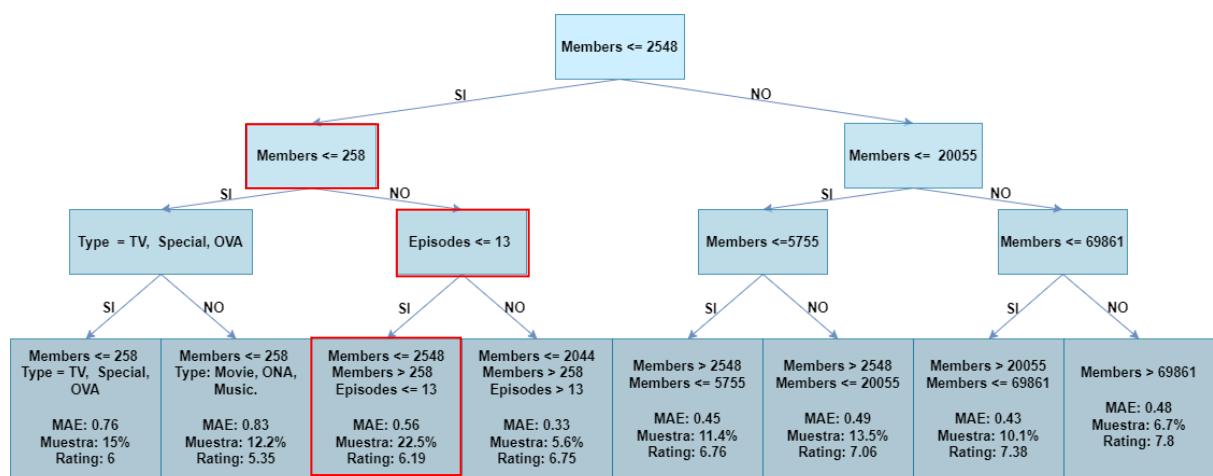


Figura 38: Árbol de decisión
Fuente: Matplotlib, elaboración propia

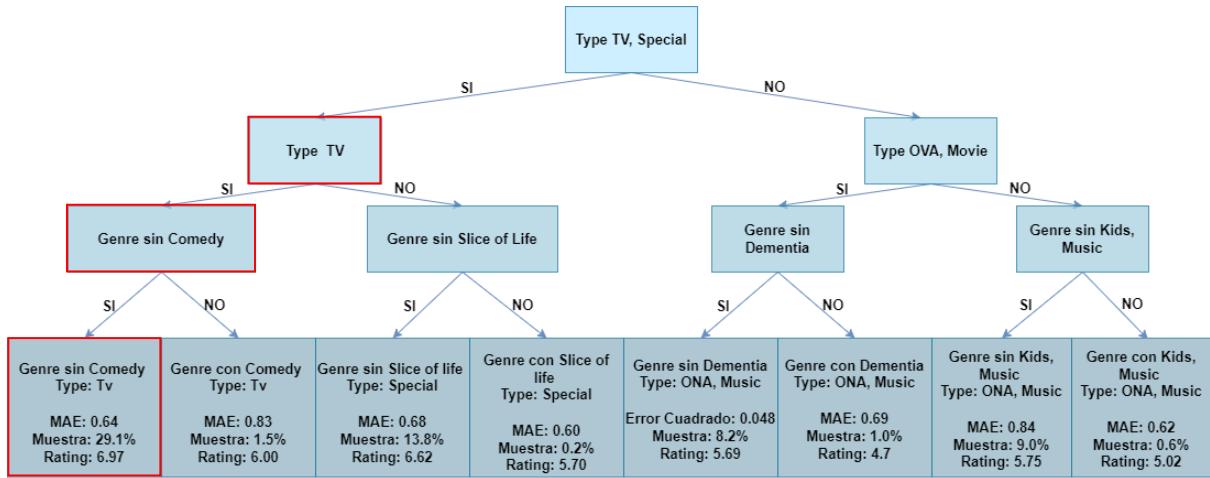


Figura 39: Árbol de decisión sin episodios ni miembros

Fuente: Matplotlib, elaboración propia

Como observamos en la figura 38, en los recuadros marcados en rojo, se trata de una muestra que corresponde al 22,5% del total, logra un error de 0,56 y un rating promedio de 6,19. Este grupo marcado tiene la característica de estar con un número de miembros mayor a 258 y menor o igual a 2548 miembros, además de contar con una cantidad menor o igual a 13 episodios. El funcionamiento del árbol de decisión de la figura 38, es que con todas las variables del dataset se busca minimizar el error con todos los puntos de corte, en este caso el MAE, entonces se va probando la cantidad de miembros desde la cantidad mínima hasta la máxima y va calculando el MAE por dentro, hace lo mismo para episodios y todas las demás variables; y se elige la variable con el punto de corte que tenga el menor MAE, en el caso que se muestra se eligió la variable miembros justo en el corte 2548, que es donde hace la mejor separación, en este punto si cumple o no esta condición se va hacia la izquierda o si no hacia la derecha; este proceso se repite con la misma u otra variable; el número de separaciones se definió en 3 (max depth).

También con este árbol de decisiones se pueden realizar predicciones, estas predicciones se hacen en base a algún animé del cual tenemos los miembros y episodios. Siguiendo el ejemplo de los recuadros marcados en rojo, si hay un animé que tenga más de 13 episodios y que tenga menos de 2538 miembros, va a tener una predicción de rating promedio de 6,19. Entonces en este caso de árbol vamos a tener un total de 8 predicciones distintas.

11. Anexos

11.1 Carta Gantt



carta_gantt_anime.
pdf

11.2 Código



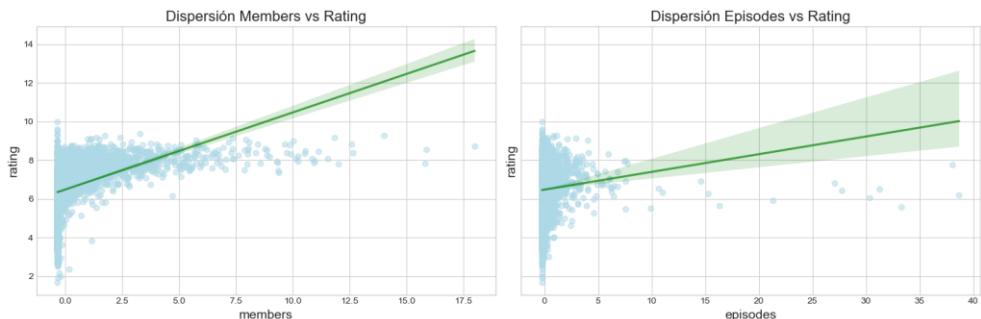
anime1_Hito_4.ipyn
b

11.3 Enlace a public.tableau

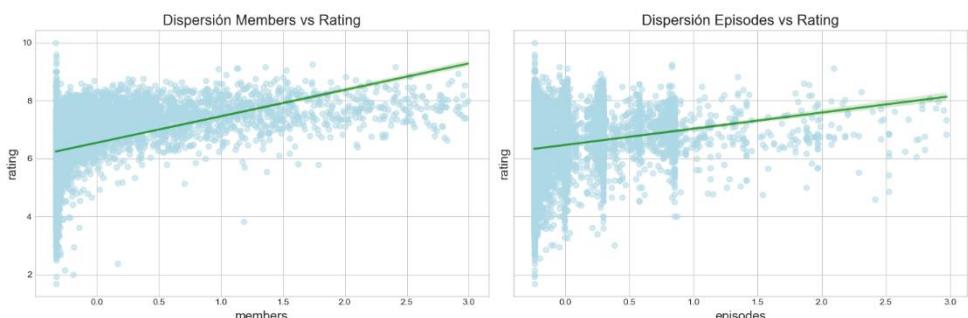
<https://public.tableau.com/app/profile/walther2215/viz/Anime-Anlisisdedatosbasecompleta/ProyectoDataScienceAnime1?publish=yes>

11.4 Gráficos de dispersión

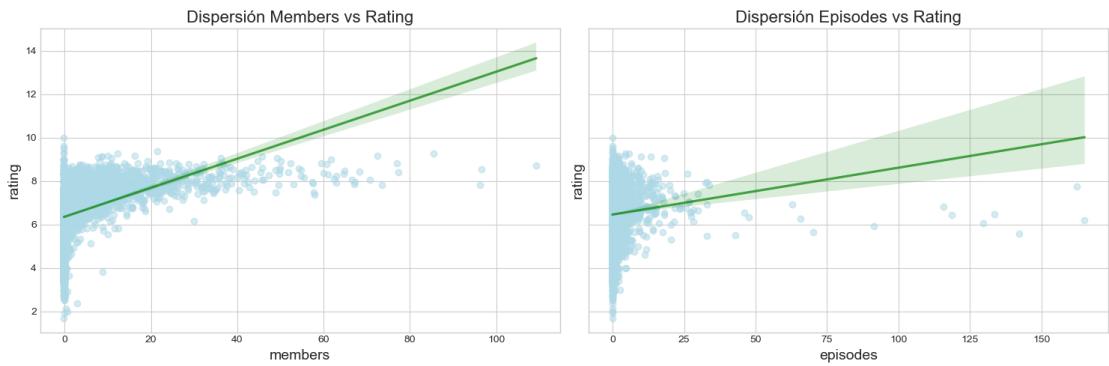
- Dataframe standard scaler:



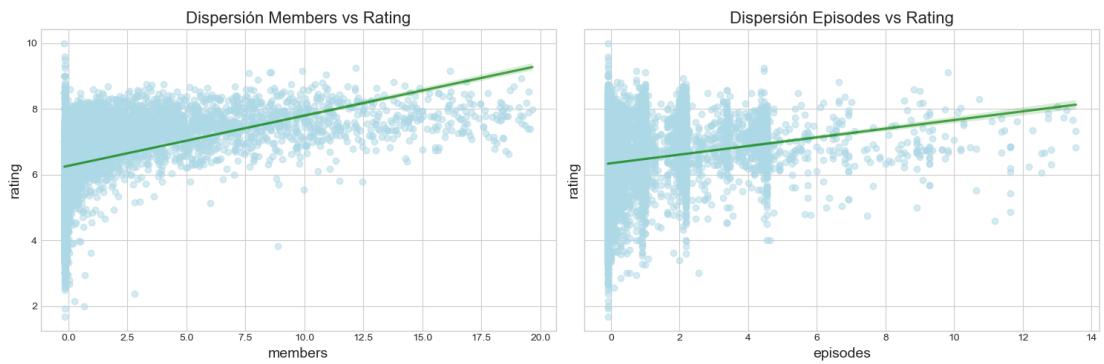
- Dataframe standard scaler sin out:



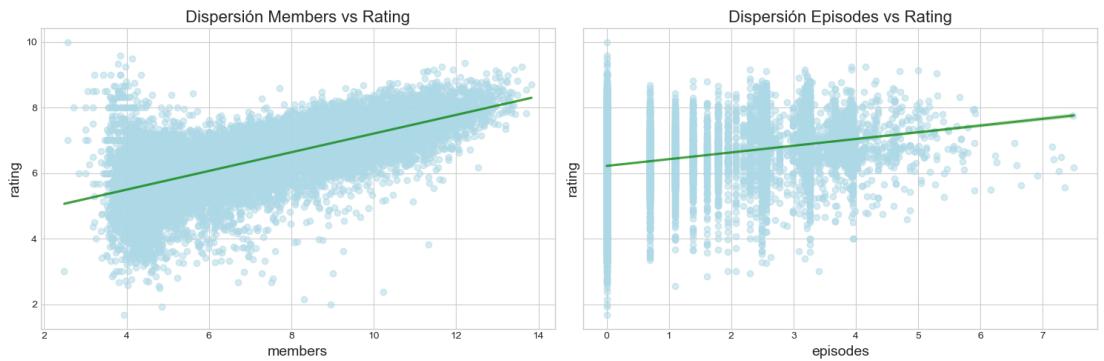
- Dataframe robust scaler:



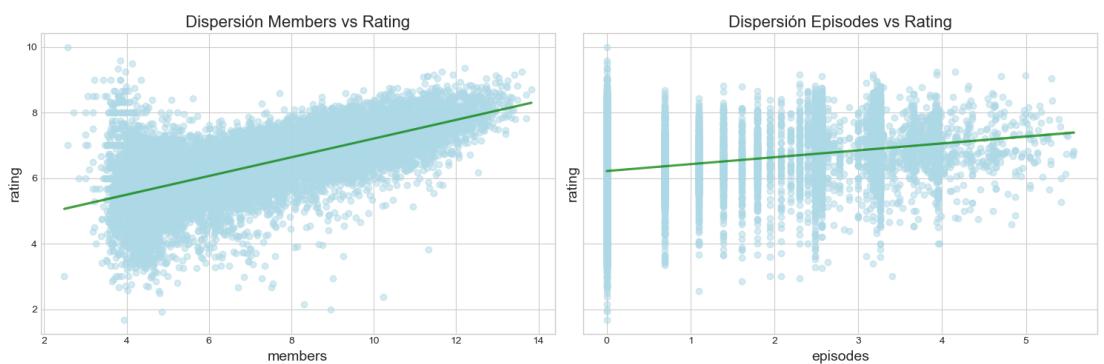
- Dataframe robust scaler sin out:



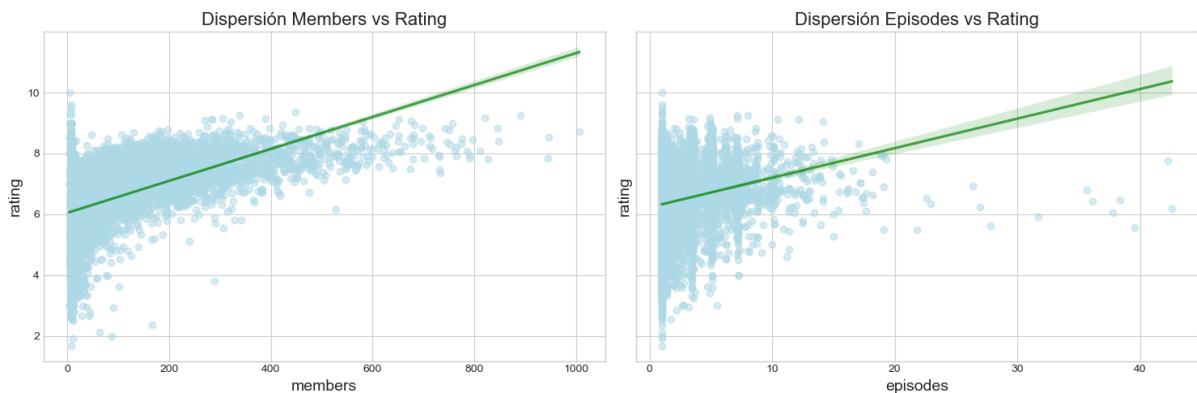
- dataframe logaritmo:



- dataframe logaritmo sin out:



- Dataframe raíz cuadrada:

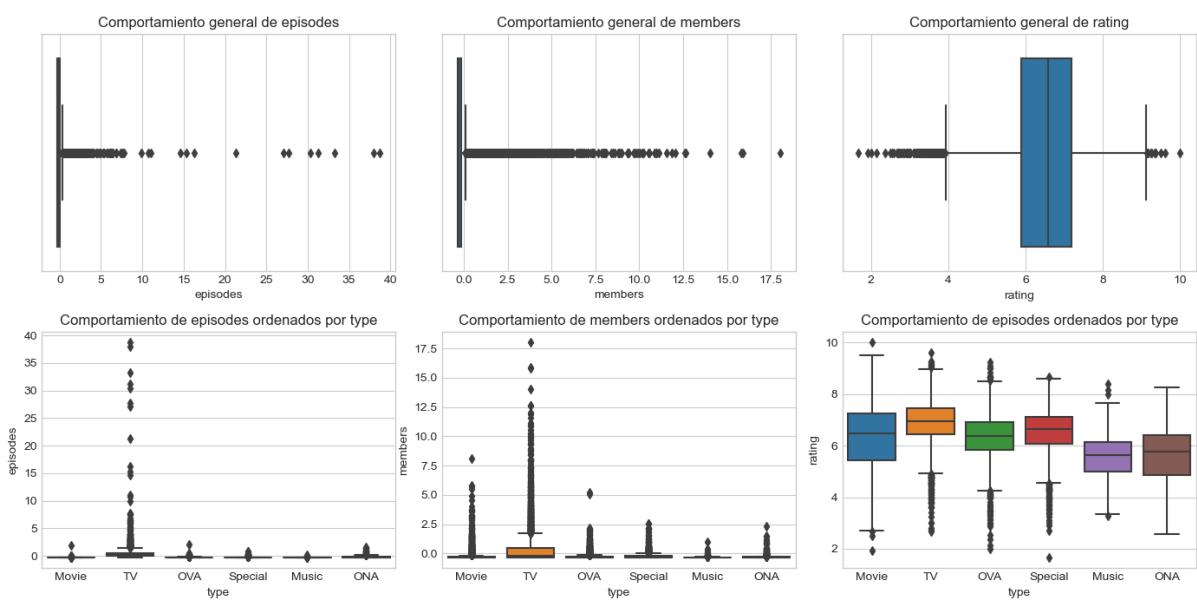


- Dataframe raíz cuadrada sin out:

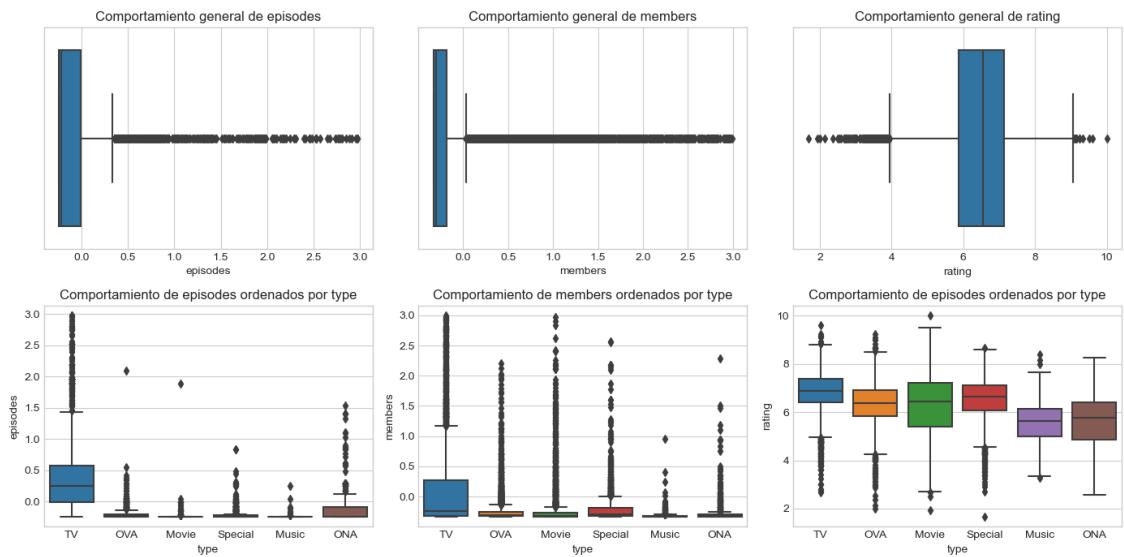


11.5 Boxplots

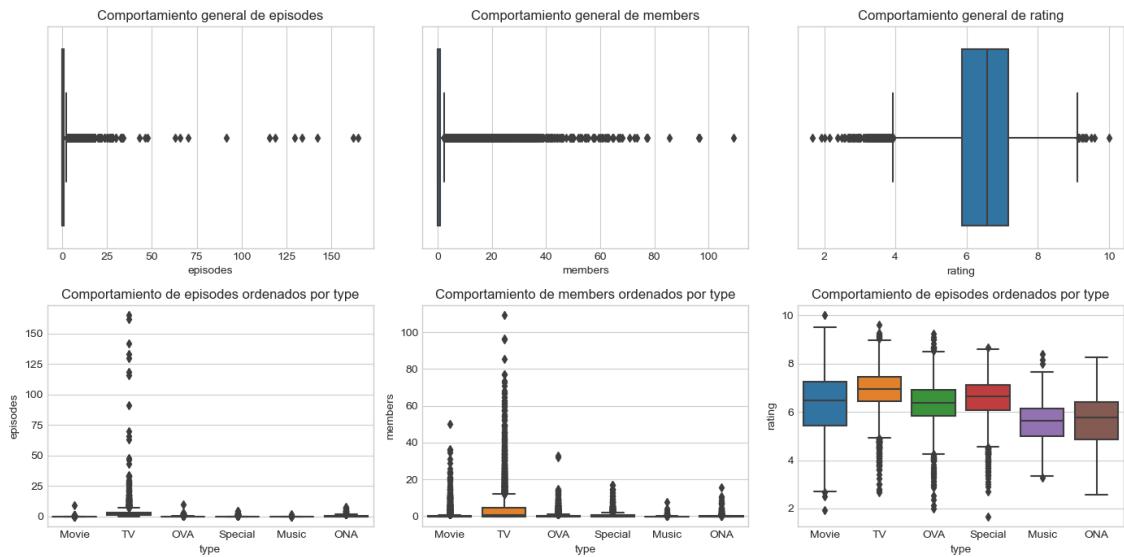
- Dataframe standard scaler:



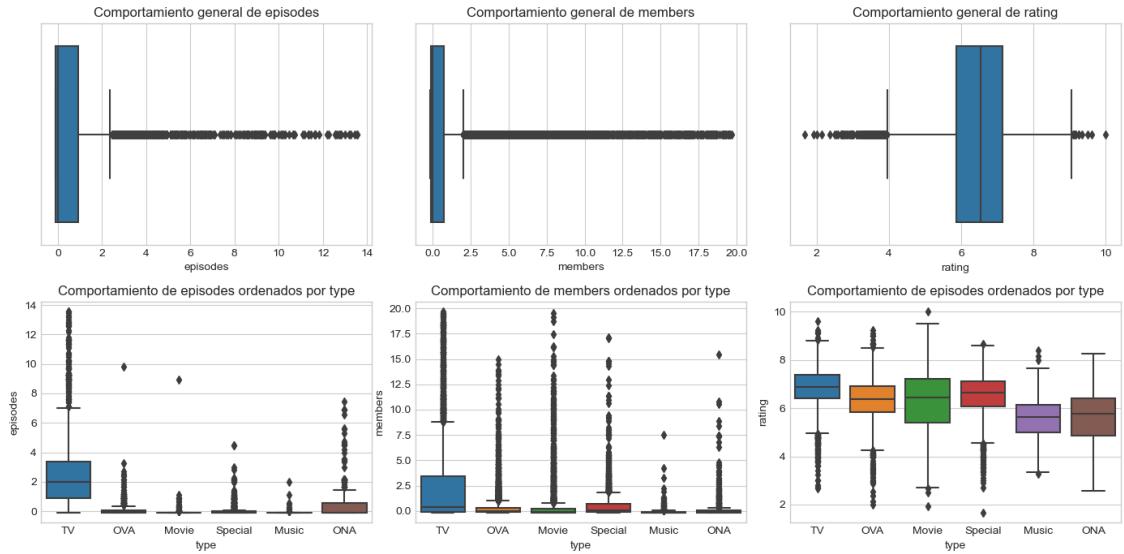
- Dataframe standard scaler sin out:



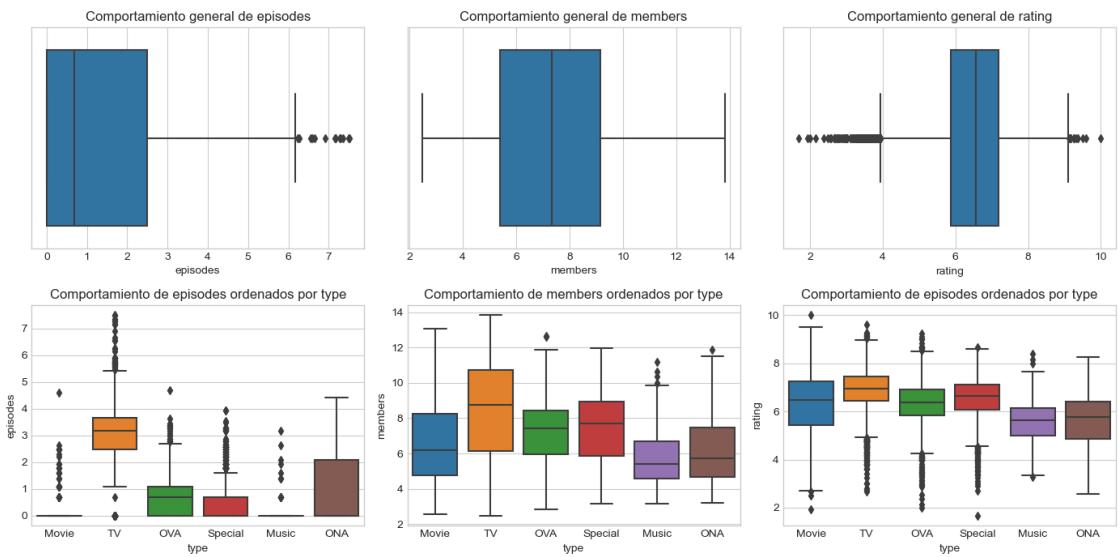
- Dataframe robust scaler:



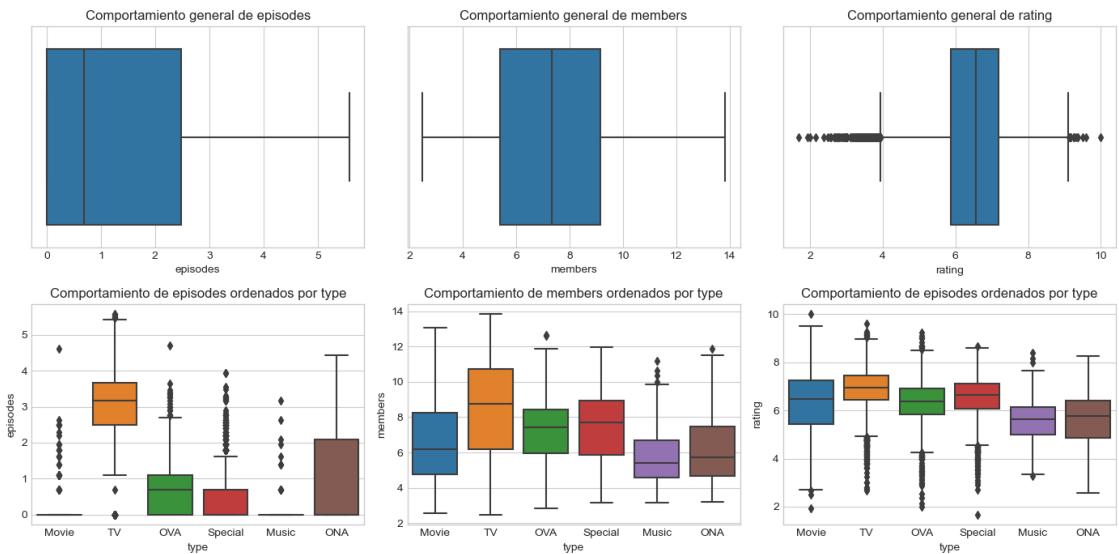
- Dataframe robust scaler sin out:



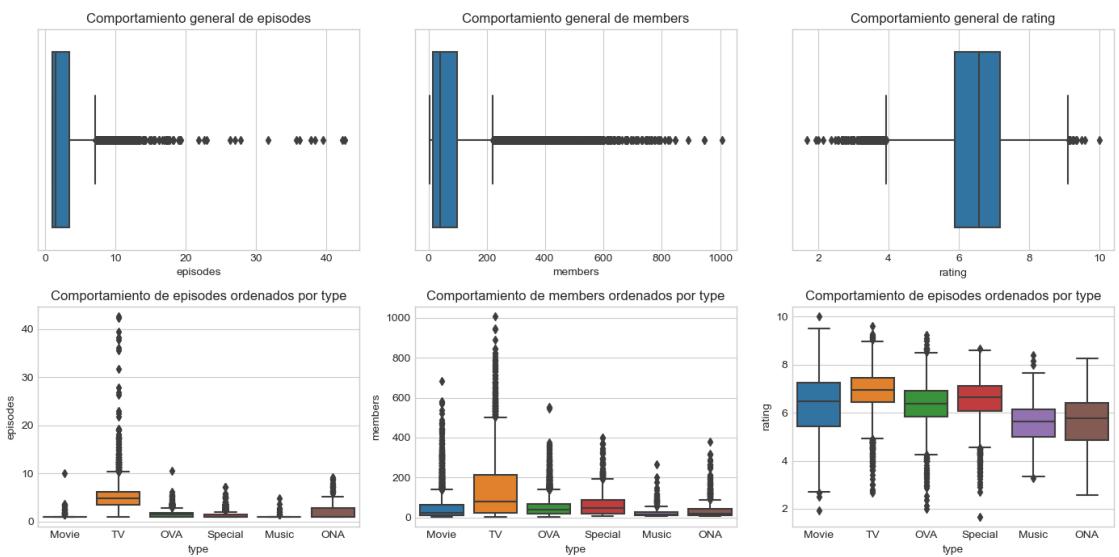
- **dataframe logaritmo:**



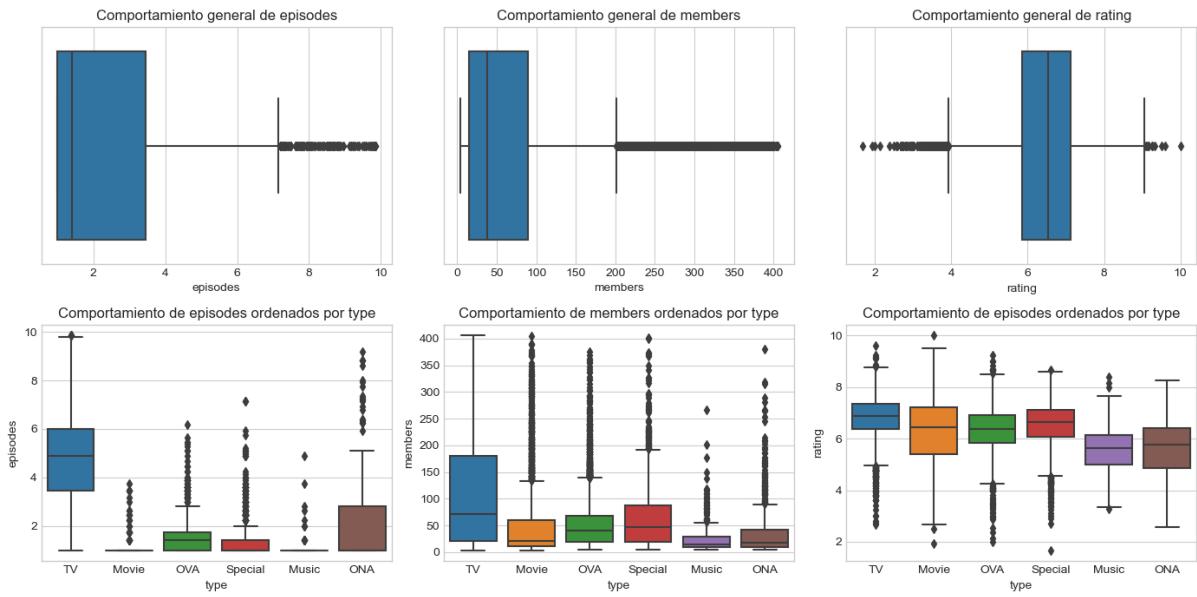
- **dataframe logaritmo sin out:**



- **Dataframe raíz cuadrada:**

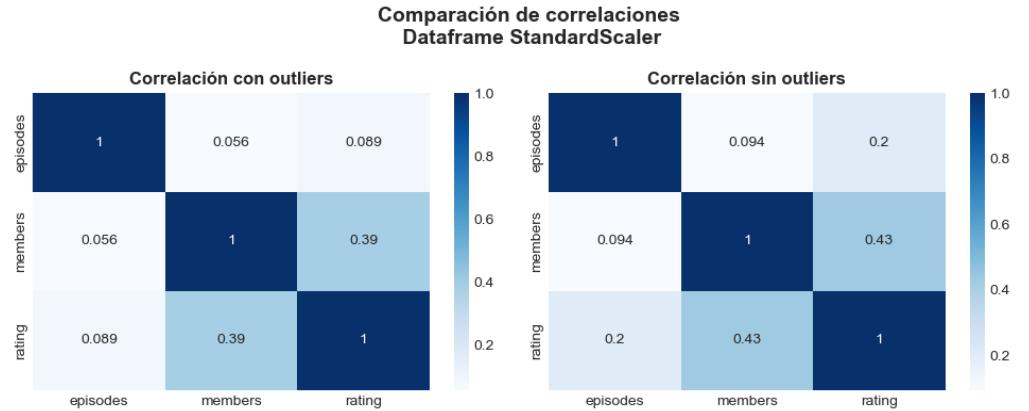


- Dataframe raíz cuadrada sin out:

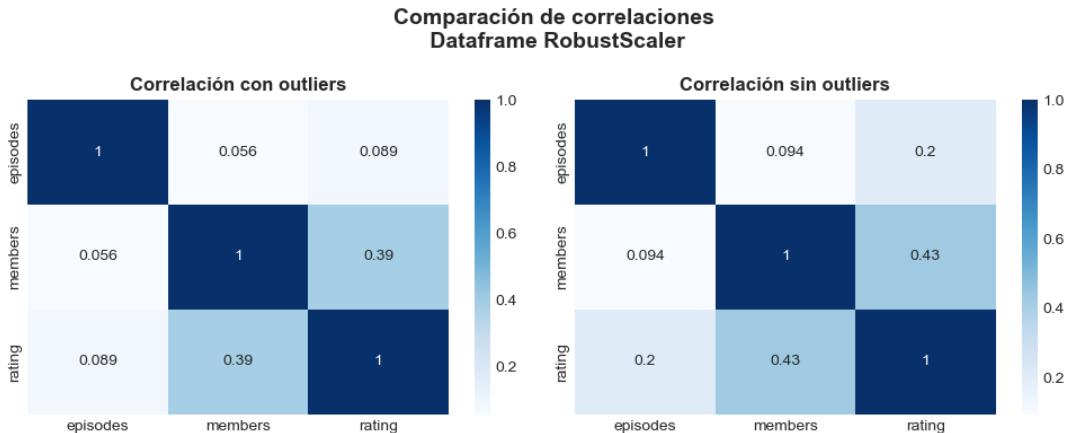


11.6 Matriz de correlaciones

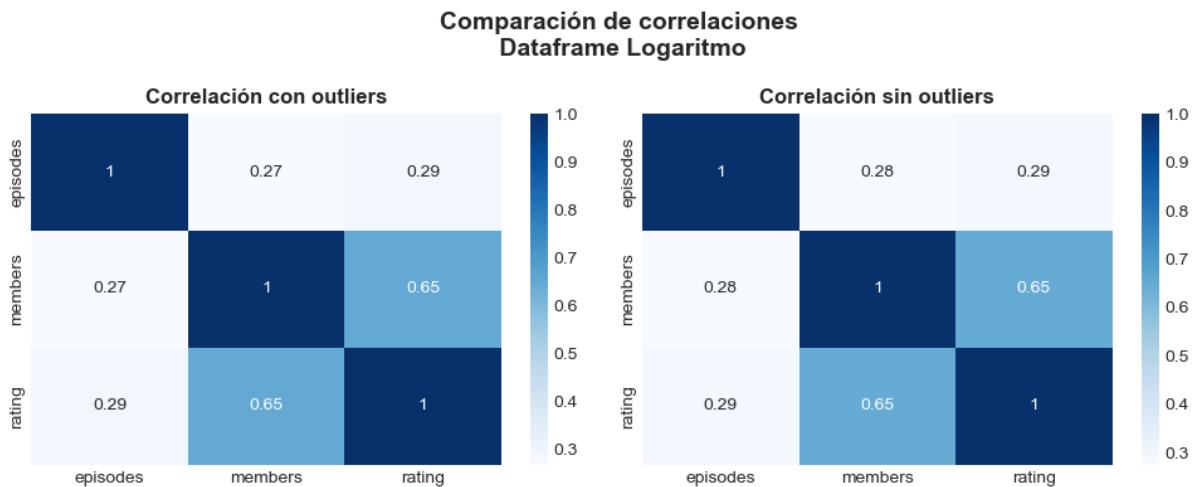
- Dataframe standard scaler con y sin out:



- Dataframe robust scaler con y sin out:



- Dataframe logaritmo con y sin out:



- Dataframe raíz cuadrada con y sin out:

