UNIVERSIDAD PROVINCIAL DEL SUDOESTE

FACULTAD DE LA MICRO, PEQUEÑA

Y MEDIANA EMPRESA

TECNICATURA UNIVERSITARIA EN TECNOLOGÍAS DE LA PROGRAMACIÓN

TRABAJO FINAL

Autor:
Martín Piampiani

Docente:
Ing. Valentín Barco

Modelo de Aprendizaje para Calificaciones IMDb

Introducción

En el mundo del cine, elegir una película entre la amplia oferta puede ser un desafío. Las preferencias personales, dificultan la toma de decisiones. La falta de indicadores previos sobre la calidad de una película puede llevar a elecciones decepcionantes. Así surge la necesidad de una herramienta que ofrezca a los espectadores una predicción anticipada del puntaje IMDb para facilitar la elección de películas.

La importancia del proyecto radica en simplificar la experiencia al espectador, haciendo una predicción de un puntaje IMDb ofrecemos una herramienta valiosa para tomar decisiones informadas. Esto no solo mejora la satisfacción del espectador, sino que también proporciona datos útiles a la industria cinematográfica sobre las preferencias del publico.

El principal objetivo de este trabajo es desarrollar un modelo de Machine Learning que prediga aproximadamente el puntaje IMDb de una película, con el fin de proporcionar una herramienta tanto como a espectadores como a la industria del cine, la idea es que este modelo se convierta en un recurso para comprender las tendencias de la audiencia y ayudar a la misma a la hora de tomar decisiones.

Metodología

Para este proyecto, utilizamos un conjunto de datos ya preparados por la comunidad, extraído de Kaggle.com. Este dataset contiene aproximadamente información sobre 5000 películas distintas, cada una representada en una fila donde las columnas son los datos que determinaremos en el análisis si son útiles o relevantes a la hora de ver el impacto que tienen sobre el puntaje IMDb de la la película, de ser así o tener alguna relación se utilizara para el modelo de aprendizaje.

Este conjunto de datos posee 28 columnas por película, por lo que tenemos bastante información para analizar, a continuación, mostramos algunas de los datos mas importantes que observamos a simple vista:

- “imdb_score”: Este valor es posiblemente el mas importante de todo el conjunto de datos, contiene los valores del puntaje IMDb de cada película, nuestro parámetro “Y” para los modelos seleccionados.
- “gross”: Esta columna tiene los valores de las ganancias de las películas en el dataset, posiblemente este relacionado con el puntaje de la película.
- “director_facebook_likes”: Número de la cantidad de likes en Facebook que tiene el director de la película, podría estar relacionado o no con el puntaje.

Luego de una vista rápida sobre los datos que tenemos vamos a ver las herramientas y librerías vamos a implementar y utilizar en el transcurso del proyecto:

1. Pandas: Utilizamos esta librería para cargar, manipular y analizar el conjunto de datos, facilitando tareas como la limpieza de datos, la creación de visualizaciones y la preparación de datos para modelos de Machine Learning.
2. Numpy: Lo usamos para realizar operaciones matemáticas en el dataset, facilitando la manipulación y transformación de datos numéricos asociados con el análisis y modelado de películas.
3. Matplotlib: Durante todo el proyecto empleamos esta herramienta para generar gráficos que ayudaron a comprender patrones y tendencias en los datos. Desde gráficos de barras hasta dispersión, nos facilito la representación visual de información relevante para el análisis y presentación de resultados.
4. Seaborn: Esta librería complementa a Matplotlib, permitiéndonos crear visualizaciones más estilizadas, mejorando la interpretación y presentación visual de la información.
5. Scikit-Learn: Utilizamos Scikit-Learn para implementar algunos de los modelos de aprendizaje que esta extensa librería ofrece. También aprovechamos sus utilidades para dividir datos, codificar etiquetas, evaluar el rendimiento de nuestros modelos y hacer ensambles de los mismos. Luego de probar todos los modelos de Machine Learning de regresión que nos ofrece la librería, decidimos conservar los 4 con mejor puntuación:
    - Linear Regression
    - Random Forest Regressor
    - Support Verctor Regression
    - Ridge

Iniciamos el proyecto importando las librerías necesarias mencionadas anteriormente, y lo primero a realizar es importar el dataset utilizando pandas y mostrar una vista de los datos con los que estaremos trabajando y sus columnas. Tras ver las columnas sin mucho detalle pudimos sacar algunas columnas que consideramos no relevantes, algunas de ellas fueron:

- "movie_imdb_link": Columna que contiene un link hacia la película, no relevante para nuestro análisis.
- "movie_title": Nombre de la película en cuestión, también irrelevante para el proyecto ya que el éxito de una película no se debe a esto.

Luego de eliminar las columnas mencionadas y algunas mas, procedemos a ver cuantas celdas con valores "Null" tenemos. Para darnos una idea de como manejar esta falta de datos vamos a hacer gráficos, específicamente gráficos de dispersión sobre el puntaje IMDb y las diferentes columnas con valores numéricos, adicionalmente para estas columnas vamos a otro grafico con los mismos valores pasados por un algoritmo, para reducir el sesgo que podrían llegar a tener, valores extremadamente altos por ejemplo. Para las columnas con valores de tipo "str" vamos a hacer gráficos de barras, para ver cuanta cantidad de cada valor tenemos. Para facilitar esto declaramos 3 funciones distintas, cada una para cada grafico correspondiente, por lo que al llamar la función e indicándole que columna queremos graficar esta nos retornara el grafico deseado.

Ahora que tenemos una idea de los datos que tenemos, vamos a analizar como manejamos las celdas que tienen valores vacíos, por ejemplo, tras ver los gráficos de los valores en las columnas color y lenguaje, Figura 1 y Figura 2 respectivamente, podemos observar que la diferencia es muy grande, por lo que se eliminaron estas columnas, al tener datos que son bastante constantes.

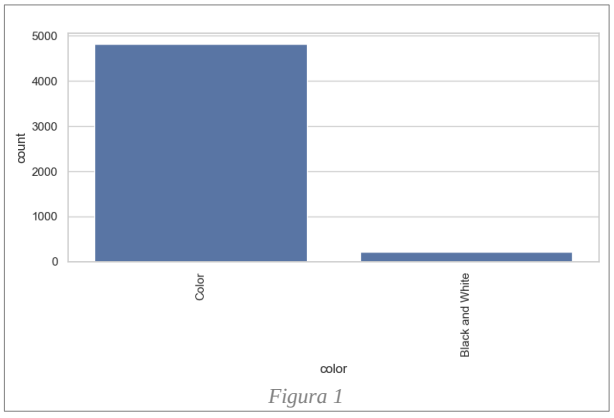

Figura 1

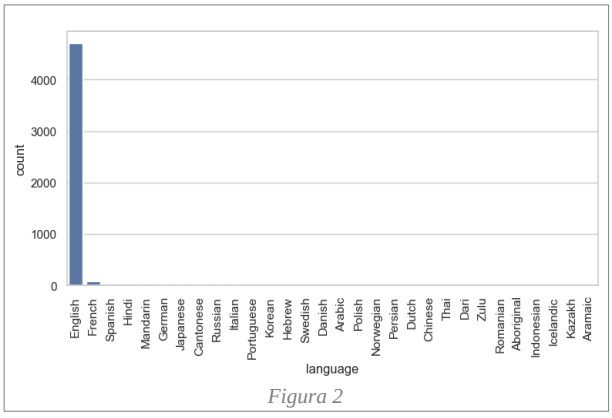

Figura 2

Ademas de la eliminación de las columnas anteriores, se elimino también la columna “plot_keywords”, ya que contenía datos que no creímos importantes (Palabras clave de la trama de cada película). Analizando algunos de los datos pudimos observar que la distribución de países de las películas se centraba en 2 países (Esto lo podemos observar en la Figura 3), USA y UK, y el resto de países en menor cantidad, por lo que agrupamos los restantes quedándonos 3 posibilidades, USA, UK y “others” donde contenemos el resto de los países, esto con el fin de reducir la dimensión de esta variable ya que por lo general suelen ser películas de USA o UK.

Por ultimo, en la limpieza de datos, pudimos detectar que en la columna “content_rating” tenia aproximadamente 300 celdas vaciás, en este caso se opto luego de analizar el grafico en la Figura 4 que lo mejor sería agregar estos valores al que mayores tenga, en este caso “R”, por lo que se rellenaron las 303 celdas vaciás con este.

No se intento recuperar los valores en las celdas de las columnas numéricas como “gross” o “budget” ya que las consideramos unas de las mas importantes para este modelo, intentar llenar todas esas celdas vaciás con valores como por ejemplo la mediana o el promedia podría llegar a sesgar o darnos datos incorrectos. Finalizando este bloque de limpieza de datos se ejecuto un “dropna” sobre el DataFrame para eliminar todas las filas con celdas restantes y avanzamos con la preparación de los datos para el entrenamiento de los modelos.

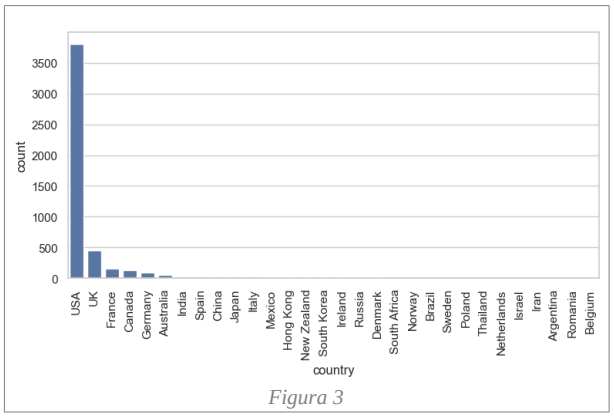

Figura 3

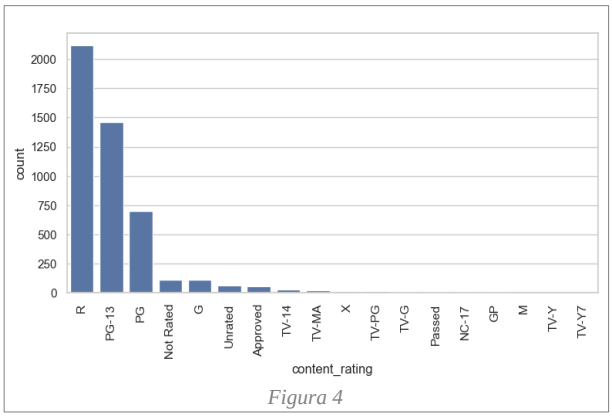

Figura 4

Para finalizar el análisis de los datos, visualizamos en un grafico de torta, en la Figura 5, con el cual podemos ver la cantidad de valores que tenemos en la columna “imdb_score”, gracias a esto pudimos detectar que tenemos la mayor cantidad de películas entre un valor de 6 y 7, mientras que entre 1 y 3 tenemos un 0,7% del DataFrame. Debido a esto decidimos eliminar todas las filas que estén en ese rango bajo, esto tuvo un gran impacto en los modelos, mejorando ampliamente su rendimiento, cabe destacar que restando también los valores de 4 a 5 que representan un 8,1% de nuestro DataFrame la mejora es aun mas, pero consideramos para este caso que es demasiada información perdida, por lo que eliminamos las filas con valores menores a 3.

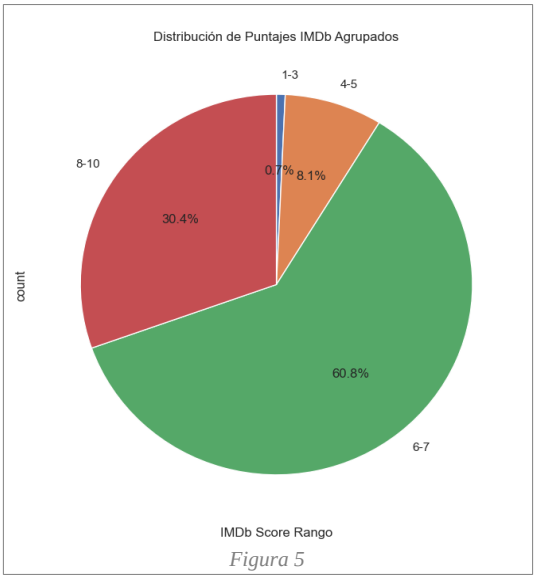

Figura 5

Iniciamos la preparación de datos para el entrenamiento de los distintos modelos, procedemos a separar nuestros datos en “Y” donde se encontrara el puntaje IMDb propio de cada película, y “X” donde estarán el resto de columnas que decidimos conservar. Antes de separar los datos en prueba utilizamos la clase “LabelEnconder” que nos ofrece Scikit-Learn, para poder convertir las etiquetas de texto que se mantienen en algunas columnas en valores numéricos comprensibles para los distintos modelos.

Luego de realizar pruebas exhaustivas en nuestro conjunto de datos, llegamos a la conclusión de que la aplicación de la transformación logarítmica es beneficiosa. Esta estrategia ha demostrado ser eficaz en la normalización de los datos, reduciendo el sesgo y estabilizando la varianza en aquellas columnas que inicialmente presentaban valores extremos. Esta técnica ha demostrado ser una elección valiosa para mejorar la robustez y el rendimiento general de nuestros modelos, por lo que se le fue aplicada a todas las columnas de la variable “X” que eran numéricas antes de la conversión, exceptuando la columna que contiene el año de la película, ya que lo consideramos no necesario.

Hasta este punto ya sabemos que no estamos trabajando con un problema de clustering o de clasificación, estamos viendo un problema de regresión, esto significa que no hay una cantidad de clases definida, no debemos categorizar nada, el objetivo principal de este tipo de modelos es entender como cambia la variable de dependiente (Y) cuando se modifican las variables independientes (Columnas de X), esto fue crucial a la hora de elegir los modelos de Machine Learning que utilizaremos en el proyecto.

Durante la etapa de prueba de los modelos pudimos detectar que algunas columnas no eran necesarias para los mismos, empeoraban el rendimiento, por eso, antes de realizar el “test train split” decidimos encargarnos de estos datos. Columnas que no aportaban datos relevantes a los modelos fueron eliminados, y para los modelos SVR y RFR se pudo detectar que algunas columnas afectaban de manera positiva mientras que en otros modelos de forma negativa, por lo que almacenamos las columnas con las que se debería entrenar a cada modelo.

En esta fase separamos entre datos de prueba y entrenamiento utilizando la herramienta que nos ofrece la librería Scikit-Learn, declaramos una función para facilitar la visualización de los resultados obtenidos por los modelos, graficando y calculando las distintas métricas clave para este tipo de modelo, como por ejemplo el Error Cuadrático Medio (MSE), la Raíz del Error Cuadrático Medio (RMSE), el Coeficiente de Determinación (R2) y algunos mas. De todas las métricas distintas de evaluación, en este proyecto nos enfocamos en conseguir el menor puntaje posible en el MSE, esta evalúa la precisión al medir la magnitud promedio de los errores entre las predicciones del modelo y los valores reales, por lo que con un puntaje alto mayor será la dispersión de los errores, indicando una menor precisión del modelo.

En este punto ya procedemos a entrenar todos los modelos de regresión que nos ofrece Scikit-Learn y conservamos en el proyecto los 4 que nos ofrecieron mejores resultados, una vez ya tenemos los modelos entrenados, utilizamos la función previamente declarada y adicionalmente, unicamente para el modelo de Support Vector Regression, es necesario hacer un ajuste en los datos, utilizando la clase “StandarScaler” (También de Scikit-Learn) realizamos la estandarización de los datos tanto de prueba como entrenamiento, esta transformación garantiza que todas las características tengan una escala comparable.

Finalmente, con los modelos ya entrenados y con sus predicciones almacenadas en variables, realizamos un ensamble promedio, se trata de uno de los mas simples, se suman todas las predicciones de los distintos modelos y lo dividimos en la cantidad de modelos, realizaremos este ensamble con los 4 modelos y otro ensamble aparte mas pero con los mejores 2.

Resultados

En este apartado mostraremos los diferentes resultados obtenidos por los modelos seleccionados durante el proyecto, para esto utilizaremos los distintos gráficos que fuimos realizando, en los cuales se ven reflejadas las predicciones realizadas por los modelos y los valores reales que correspondían. También presentaremos las distintas métricas de evaluación que mencionamos anteriormente.

- Error Cuadrático Medio (MSE): Mide el promedio de los cuadrados de las diferencias entre las predicciones del modelo y los valores reales, cuanto menor sea el MSE, mejor será el modelo.
- Raíz del Error Cuadrático Medio (RMSE): Es la raíz cuadrada del MSE. Proporciona una medida en la misma escala que la variable de respuesta original, lo que nos facilita la interpretación.
- Coeficiente de Determinación (R2): Representa la proporción de la variación en la variable dependiente que es explicada por el modelo. Un R^2 cercano a 1 indica que el modelo explica bien la variabilidad de los datos, mientras que uno cercano a 0 indica que el modelo no explica la variabilidad.
- Error Absoluto Medio (MAE): Mide el promedio de las diferencias absolutas entre las predicciones del modelo y los valores reales. Es menos sensible a valores atípicos que el MSE.
- Error Porcentual Absoluto Medio (MAPE): Mide el promedio de los porcentajes absolutos de error entre las predicciones del modelo y los valores reales.

Modelo	MSE	RMSE	R2	MAE	MAPE	Gráfico
Linear Regression	0,602	0,775	0,400	0,590	9,856	Figura 6
Random Forest Regression	0,429	0,655	0,571	0,488	8,180	Figura 7
Support Vector Regression	0,415	0,644	0,586	0,471	7,999	Figura 8
Ridge	0,602	0,775	0,400	0,590	9,858	Figura 9
Ensamble 1	0,421	0,649	0,579	0,483	8,175	Figura 10
Ensamble 2	0,391	0,625	0,609	0,460	7,805	Figura 11

Analizando las Figuras 6 y 7, podemos ver algunas diferencias a simple vista, el modelo de Linear Regression tiene mucha mas dispersión en la parte inferior, ambos tienden a hacer malas predicciones cuando la película tiene puntajes bajos. En cambio el modelo Random Forest Regressor fue mucho mas preciso al encontrar películas con puntajes mas alto, podemos observar en el grafico correspondiente a la Figura 7 que en la punta superior derecha la dispersión es menor.

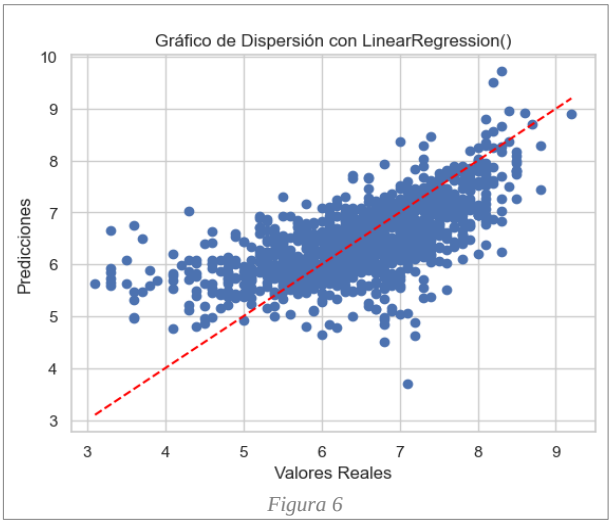

Figura 6

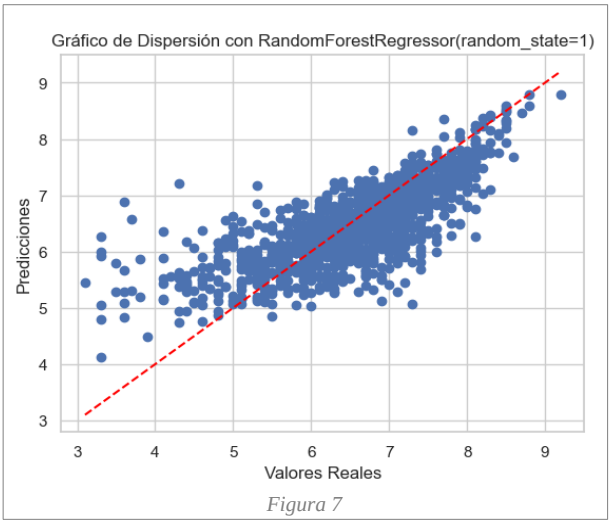

Figura 7

En el caso de los gráficos de las Figuras 8 y 9 siguen los mismos patrones, mientras que el modelo Support Vector Regression dio mejores resultados cuando predecía una película con un puntaje alto (Como el modelo Random Forest Regression), el modelo Ridge tuvo un comportamiento similar al modelo Linear Regression. Observando detenidamente la tabla con los resultados de cada métrica y viendo a detalle cada grafico pudimos ver que los modelos Ridge y Linear Regression eran iguales, mismas predicciones, y por lo tanto mismos gráficos y mismo porcentaje de errores.

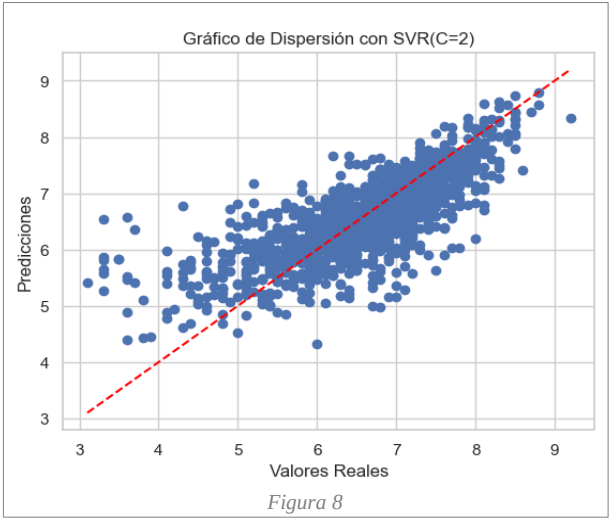

Figura 8

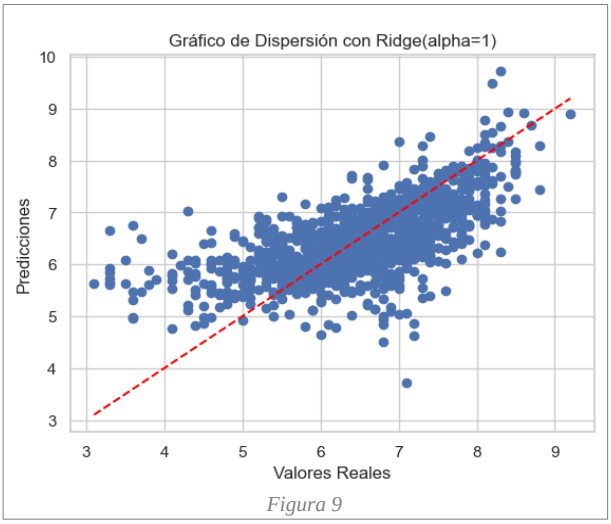

Figura 9

Luego de analizar los modelos se decidió realizar 2 ensambles “promediales”, se suman las predicciones de los modelos seleccionados y se dividen por la cantidad de los mismos, de esta forma sacamos el promedio de las predicciones entre 2 o mas modelos. En nuestro caso, ya que tenemos 2 modelos que nos dan los mismos resultados, no tomamos uno de estos en cuenta (No debería importar cual) para el ensamble de todos los modelos, ya que podría sesgar el resultado.

En la Figura 10 se encuentra el grafico correspondiente al ensamble con 3 modelos (Ensamble 1 en la tabla), ya podemos observar que el grafico es mas compacto, similares a los modelos RFR y SVR, en la tabla obtenemos ligeramente mejores resultados que en los modelos mencionados. Por otro lado, en la Figura 11, hicimos el ensamble de los 2 mejores modelos (RFR y SVR, Ensamble 2 en la tabla), en consecuencia aumentamos la precisión disminuyendo la dispersión en la zona inferior derecha, superando incluso al anterior ensamble. Indicando en verde en la tabla, podemos observar que hay diferencia notable en el rendimiento de el ensamble con el resto de los modelos, por lo que consideramos a este el “mejor ensamble” o “ensamble más eficaz”.

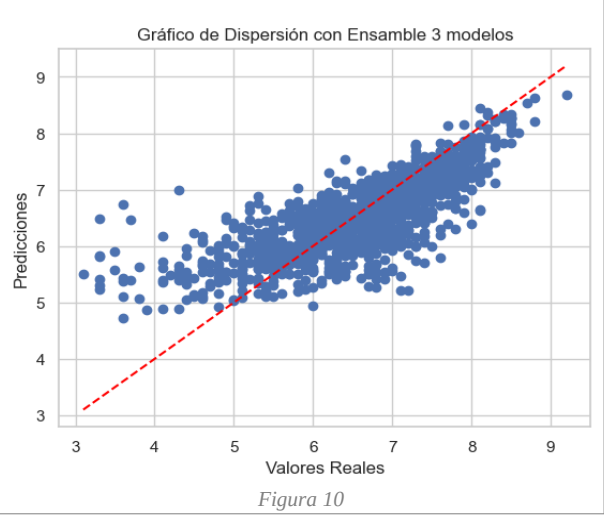

Figura 10

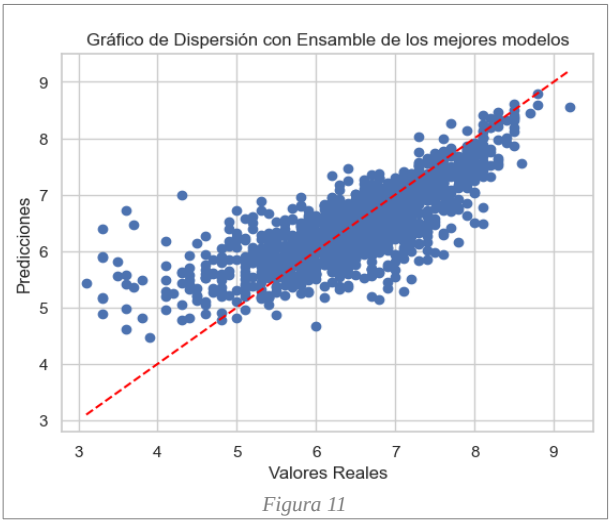

Figura 11

Discusión

Con este proyecto, logramos desarrollar un modelo capaz de predecir con facilidad películas con puntajes altos. Aunque presenta algunos errores al predecir películas que podrían considerarse “menos favorables” y asignarles puntajes como 5 o 6, el modelo ha cumplido con éxito el objetivo que se propuso. Gracias a su eficacia al anticipar películas de alta calidad, proporciona una valiosa guía al espectador, facilitándole la elección de que película debería ver.

Uno de los mayores desafíos fue la preparación de los datos, la eliminación de sesgo y datos con valores extremos, que en su mayoría se pudo resolver luego de investigar a fondo y como resultado utilizamos la transformación logarítmica. También se presento como un reto el pulir los modelos, los hiperparámetros no parecían tener mucho impacto en el rendimiento de los modelos, por lo que se probo combinando distintas columnas columnas y así mejoramos poco a poco el margen de error de los modelos.

Conclusiones

Logramos desarrollar un modelo capaz de predecir con cierta precisión el puntaje IMDb de una película, basándonos en un conjunto de datos que contiene información sobre más de 5000 películas. El objetivo principal de este trabajo fue proporcionar una herramienta útil tanto para los espectadores como para la industria del cine, que les permita elegir o producir películas de calidad.

Los resultados obtenidos por los modelos de regresión que implementamos mostraron que hay una relación entre el puntaje IMDb y algunas variables como el presupuesto, las ganancias, el número de likes en Facebook del director y el año de lanzamiento de la película. El mejor modelo que encontramos fue un ensamble promedio entre el Random Forest Regressor y el Support Vector Regression, que logró un error cuadrático medio de 0,391 y un coeficiente de determinación de 0,609. Estas métricas indican que el modelo tiene una buena capacidad de explicar la variabilidad de los datos y de minimizar los errores de predicción.

En este proyecto, aplicamos los conceptos de Big Data y Machine Learning que aprendimos en el curso. Utilizamos librerías como Pandas, Numpy, Matplotlib, Seaborn y Scikit-Learn para manipular, analizar y visualizar los datos, así como para implementar y evaluar los modelos de aprendizaje. También empleamos técnicas como la transformación logarítmica, la codificación de etiquetas, la estandarización de datos y el ensamble de modelos para mejorar el rendimiento y la robustez de nuestros algoritmos.

Una propuesta significativa para mejorar el proyecto es la implementación de una interfaz de usuario intuitiva. Esta interfaz permitiría a los usuarios ingresar el nombre de una película de su interés, y automáticamente buscaría los detalles de la película para realizar una predicción del puntaje IMDb utilizando nuestro modelo desarrollado. Esta adición transformaría la herramienta en una solución más práctica y personalizada para los espectadores, facilitando la toma de decisiones al seleccionar películas de calidad.

Código

- Link del Colab: https://colab.research.google.com/github/1Pampu/BigData-ML-Proyectos/blob/main/Trabajo%20Final/Trabajo_Final_Colab.ipynb

- Link del repositorio: https://github.com/1Pampu/BigData-ML-Proyectos

Referencias

- Pandas: https://pandas.pydata.org/docs/user_guide/index.html
- Matplotlib: https://matplotlib.org/stable/users/index
- Seaborn: https://seaborn.pydata.org/tutorial.html
- Logarithmic Transformation in Linear Regression: https://dev.to/rokaandy/logarithmic-transformation-in-linear-regression-models-why-when-3a7c
- Linear Regression: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html
- Random Forest Regressor: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html
- Support Vector Regression: https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html
- Ridge: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Ridge.html