Predicción de la popularidad de una canción

1st Daniel Gómez F.
Instituto de Ciencias de la Ingeniería
Ingeniería Civil Eléctrica
Rancagua, Chile
daniel.gomez@pregrado.uoh.cl

2nd Alex Tapia C.
Instituto de Ciencias de la Ingeniería
Ingeniería Civil Eléctrica
Rancagua, Chile
alex.tapia@pregrado.uoh.cl

***Abstract*—En este documento se analiza un conjunto de datos de Spotify, buscando estimar de popularidad que tendría una canción en la plataforma, utilizando algoritmos de predicción. La base de datos utilizada contiene información numérica y categórica de una gran variedad de canciones, como por ejemplo, artista y género musical, entre otras características. Se aplicarán contenidos aprendidos en el curso para el preprocesamiento de los datos y el desarrollo de modelos de aprendizaje supervisado capaces de predecir la popularidad de una canción.**

***Index Terms*—Codificar, Modelo predictivo, Métricas.**

I. Introducción

La popularidad de una canción es un motivo de estudio por los analistas de datos, expertos en marketing y artistas, es por eso que se ha vuelto importante determinar las propiedades más importantes de una canción para generar éxitos musicales. Esto puede ayudar a los artistas y miembros de la industria musical en general, a tomar decisiones basadas en datos certeros sobre las estrategias de composición en las piezas musicales.

El proyecto indaga sobre el impacto de los distintos parámetros de una canción en su recepción y difusión. Ayudando a los músicos a comprender e identificar, mediante el análisis de datos, patrones y tendencias que indican qué atrae al público y cómo mejorar la calidad de su trabajo.

II. Objetivos del proyecto

El motivo del estudio es desarrollar modelos de predicción que sean capaces de predecir de forma precisa la popularidad que conseguirá una canción en base a ciertas características propias de esta.

También se cuantificarán las diferencias entre los modelos desarrollados con respecto a otros donde los datos solo han tenido un preprocesamiento básico y no se hayan realizado las codificaciones.

III. Base de datos

Para este trabajo se utilizará una base de datos de canciones de *Spotify*, el cual cuenta con veinte atributos y 114.000 instancias. Esta base de datos fue elaborada por *Maharshi Pandya*, se llama *Spotify Tracks Dataset* y se encuentra en la página web *Kaggle*.

La base de datos a utilizar contiene información sobre las características de la canción, entre ellas destacan, autor, género y duración, entre otras. A destacar, se añadirán atributos a la base de datos con el fin obtener información importante en base a atributos categóricos, como por ejemplo, el artista.

IV. Análisis preliminar (Preprocesamiento y análisis exploratorio de datos)

A. *Análisis preliminar y codificación*

En primer lugar, se cargan los datos utilizando la librería pandas, se eliminan las filas con valores NaN, las columnas categóricas como el nombre del álbum, id de la canción y el nombre de la misma, y se binariza el atributo *Explicit* para poder trabajar en la base de datos sin ningún problema.

La primera modificación importante al dataset, está relacionada al status que posee el creador de la canción, algunos atributos como *artists* y *track_genre* son determinantes en el éxito de una nueva canción. Si un artista de larga trayectoria anuncia un nuevo lanzamiento, es esperable que la pieza sea esperada con expectación y sea bien recibida por el público. De igual forma, existen géneros musicales con mayor cantidad de adeptos, por lo cual, una canción perteneciente a un género musical con un gran número de seguidores tendrá buena recepción. A continuación se muestra el código utilizado para la codificación.

```
genre_popularity = spotify_data.groupby('track
        _genre')['popularity'].mean().reset_index()

genre_popularity = genre_popularity.sort_values
    (by='popularity', ascending=False)

genre_codes = {}
for i, genre in enumerate(genre_popularity
        ['track_genre']):
    genre_codes[genre] = i + 1

spotify_data.insert(5, 'genre_code',
    spotify_data['track_genre'].map(genre_codes))
```

Debido a lo anterior, los atributos mencionados son codificados desde variables categóricas a valores numéricos bajo el criterio de popularidad promedio, es decir, se realiza un promedio simple de la clase para cada artista. Los artistas, al igual que los géneros musicales, se ordenan según el promedio de popularidad de sus canciones, donde, de acuerdo a la posición en este ranking, es el código numérico que reciben. Así, por ejemplo, un artista o género musical en la primera posición del ranking recibe el número uno. Finalmente, se

agregan al dataframe las columnas *artist_code* y *genre_code*, eliminando las variables categóricas y teniendo así, un set de datos completamente numérico. A continuación, se presenta el dataset a trabajar (por motivos de espacio no se mostraron las columnas restantes)

	popularity	artist_code	mean_popularity	genre_code	duration_ms	explicit	danceability	energy	key	loudness	mode
0	73	4472	58.0	24	230666	0	0.676	0.461	1	-6.746	0
1	55	12429	43.0	24	149610	0	0.420	0.166	1	-17.235	1
2	57	4952	57.0	24	210826	0	0.438	0.359	0	-9.734	1

Fig. 1: Base de datos después del preprocesamiento

B. Matriz de correlación

Además, para conocer la relación que tienen los datos del actual conjunto, se realiza una matriz de correlación, realizada gracias a la librería de python, *seaborn*, esta permite realizar diferentes gráficos en python.

De la matriz de correlación se desprende lo que se esperaba, los atributos *artists_code* y *mean_popularity* están altamente correlacionadas con la clase, es decir, la variable a predecir. Con respecto al resto de los atributos, estos no presentan una alta correlación.

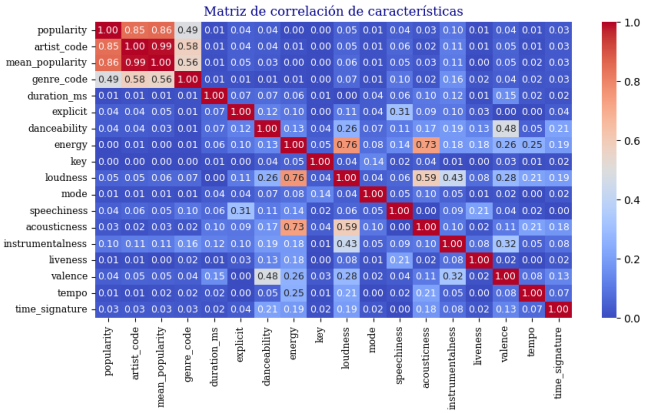


Fig. 2: Matriz de correlación

C. Histogramas

A continuación se observarán los histogramas de los dos atributos más correlacionados, por motivos de espacio no es posible añadir todos.

Analizando el histograma de popularidad, en el eje x se representan los distintos valores de popularidad, los cuales van en el rango desde 0 a 100. El eje y muestra la frecuencia o cantidad de canciones en cada rango de popularidad. Donde se puede apreciar que la mayor cantidad de canciones presentan un bajo rango de popularidad y posteriormente, la curva se ajusta de manera normalmente distribuida a los datos.

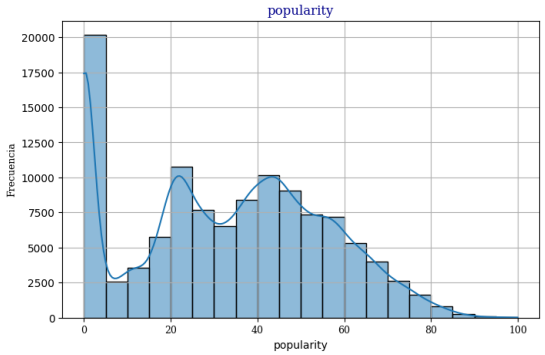


Fig. 3: Histograma popularidad

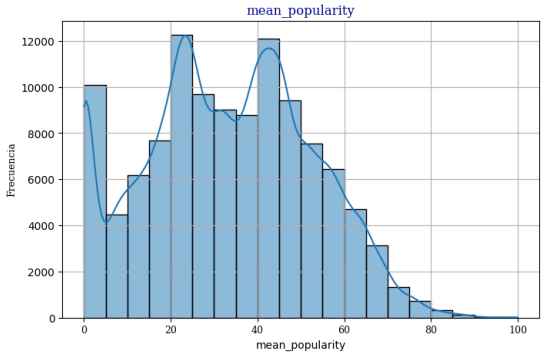


Fig. 4: Histograma popularidad media

Luego, haciendo énfasis en el histograma de popularidad promedio, la distribución se aprecia más uniforme con respecto al histograma anterior. Esto debido a que las canciones de mayor éxito compensan los lanzamientos de popularidad reducida.

D. Estadísticas de los atributos

En base a la imagen es posible observar que ciertos atributos poseen una desviación estándar muy alta, esto indica que antes de desarrollar los modelos de predicción será necesario normalizar los datos y así el modelo se ajuste de mejor forma a los datos.

	promedio	mediana	valor minimo	valor maximo	desviacion estandar
popularity	33.238827	35.000000	0.000	100.000	22.304861
artist_code	17271.451943	18094.000000	1.000	31437.000	9131.837399
mean_popularity	33.236686	33.000000	0.000	100.000	19.099141
genre_code	57.500487	58.000000	1.000	114.000	32.907433
duration_ms	228031.153387	212906.000000	8586.000	5237295.000	107295.587114

Fig. 5: Estadísticas de los atributos

E. Gráfico de dispersión

De igual manera un dato sumamente importante para la elección de algoritmos es el gráfico de dispersión entre los dos atributos correlacionados, es decir, *mean_popularity* y la clase. Del gráfico, es posible observar que se tiene una correlación positiva, es decir, que los valores de ambos aumentan juntos

y que es una correlación fuerte. Sin embargo, no es posible decir con exactitud si se posee correlación lineal.

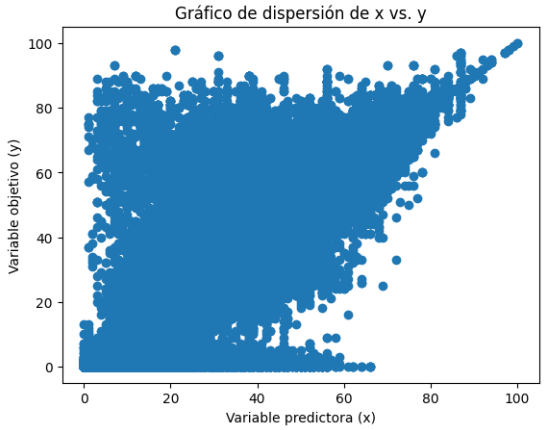


Fig. 6: Gráfico de dispersión

V. Algoritmos y métricas utilizadas

A. Métricas

Para este informe se utilizarán diversos algoritmos de predicción y distintos parámetros, se analizarán todos los resultados en base a métricas, como el error cuadrático medio, error absoluto medio y coeficiente de determinación (R^2), estas son métricas usualmente utilizadas para evaluar la precisión de un modelo de regresión. Las dos primeras métricas, miden la distancia entre el valor predicho y el valor real, por lo tanto, mientras menor sean estos índices, mejor será el modelo. Por otro lado, el R^2 es una métrica que indica la proporción de la varianza de la variable objetivo que es explicada por el modelo. R^2 se encuentra en el rango de 0 a 1, donde 1 indica que el modelo explica toda la varianza y 0 indica que el modelo no proporciona una mejora sobre una predicción constante. A continuación se especifican las formulas de cada uno.

El error cuadrático medio mide la diferencia entre la salida predicha por el modelo y la compara con la salida real y lo eleva al cuadrado, aumentando así el error del modelo, finalmente se divide en la cantidad de datos para obtener un promedio estimado.

$$MSE = \frac{1}{N}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{1}$$

El error medio absoluto sigue la misma lógica, es decir, compara la salida predicha con la real pero no la eleva al cuadrado, simplemente la divide en la cantidad de datos, obteniendo así un rango de error más pequeño que la métrica anterior.

$$MAE = \frac{\sum_{i=1}^{n}|y_i - \hat{y}_i|)}{n} \tag{2}$$

El coeficiente de determinación es la proporción de la varianza total de la variable explicada por la regresión. El coeficiente de determinación, también llamado R cuadrado, refleja la bondad del ajuste de un modelo a la variable que pretender explicar.

$$R^2 = \frac{\sum_{t=1}^{T}(\hat{Y}_t - \bar{Y})^2)}{\sum_{t=1}^{T}(\bar{Y}_t - \hat{Y})^2)} \tag{3}$$

B. Algoritmos utilizados

1) Regresión lineal: Como se pudo observar con anterioridad, no se tiene certeza que el gráfico de dispersión posea una correlación lineal, por lo tanto, se entrenará un modelo de regresión lineal para observar la precisión del algoritmo frente al problema. Este modelo, trata de crear una recta que se parezca a la distribución de los datos. Esto es posible realizarlo buscando los mejores valores para a y b de la siguiente ecuación.

Se utilizará la librería *sklearn* para poder utilizar este modelo.

$$ax + b = 0 \tag{4}$$

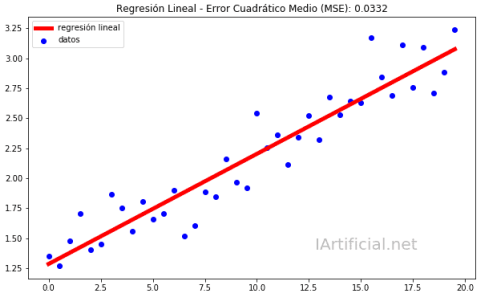


Fig. 7: Regresión lineal [3]

2) XGB: XGBoost, es una técnica donde se utilizan arboles de decisiones, es otra de las opciones elegidas, esto debido a su gran número de ventajas ha demostrado poseer gran velocidad y capacidad para manejar grandes conjuntos de datos, en este caso, se tienen 114.000 datos.

Su funcionamiento se basa en muchos arboles de decisiones que son entrenados para que finalmente se recopile información de cada uno y así poder llegar a uno solución final global.

Para este trabajo se utilizará las librerías *xgboost* para el modelo *xgboost* son modelos basados en árboles de decisiones.

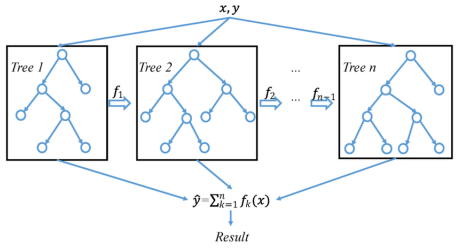


Fig. 8: XGBOOST [4]

3) Random Forest: Finalmente, también se utilizará el modelo Random Forest de la librería *sklearn*, es un algoritmo de aprendizaje automático basado en múltiples arboles de decisiones, este modelo será introducido con el fin de ser comparado a los resultados de los algoritmos *xgboost*. Con respecto a este algoritmo, se pondrán a disposición recursos computacionales para poder encontrar los mejores parámetros para el modelo.

4) SVR: La regresión de vector de soporte (SVR) es un algoritmo de aprendizaje automático utilizado para el análisis de regresión. Es diferente de los métodos tradicionales de regresión lineal, ya que encuentra el hiperplano que mejor se ajusta a los puntos de datos, en lugar de ajustar una línea. El algoritmo SVR tiene como objetivo encontrar el hiperplano que pasa a través de tantos puntos de datos como sea posible dentro de una cierta distancia, llamada margen. Lo cual ayuda a reducir el error de predicción y permite a SVR manejar relaciones no lineales entre las variables de entrada y las variables de salida utilizando una función kernel.

5) Gradient Boosting: La idea principal detrás de este algoritmo es construir modelos secuencialmente y de modo que los modelos posteriores reduzcan los errores del modelo anterior. Se basa en la combinación de modelos predictivos débiles, normalmente árboles de decisión, para crear un modelo predictivo mas robusto. La generación de los árboles de decisión débiles se realiza de forma secuencial, creándose cada árbol de forma que corrija los errores del árbol anterior. Los modelos previos suelen ser árboles "poco profundos", de apenas uno, dos o tres niveles de profundidad, típicamente.

VI. Resultados de los algoritmos

Con respecto a los resultados de los modelos desarrollados y su desempeño en la base de datos utilizada. Para conocer que tan buenas han sido las predicciones, se han realizado muchas otras pruebas, las más importantes están relacionadas con la integración de las columnas *mean_popularity* y *artist_code* y la normalización de los datos. Además se realizarán ciertas comparaciones de los datos obtenidos con otras predicciones realizadas sobre la misma base de datos.

El primer tema a observar es, cómo afectó la inclusión de las columnas más correlacionadas con la clase, para esto, se realizó un código aparte, el cual no tuviera los dos atributos mencionados mientras que uno iba a ser normalizado y el otro no, además de un tercero que no fuera normalizado pero sí se le aplicara PCA. A continuación se mostrarán los resultados (Todos fueron aplicados al mismo modelo).

Primero se observa normalizado, al no tener las columnas agregadas se observa que tuvo un mal rendimiento, si se observa el Error cuadrático medio (MSE).

TABLE I: XGB Normalizado

	MSE	**MAE**
Conjunto Validación	754.92	22.37
Conjunto Prueba	291.76	12.62
Tiempo = 120s		

Observando los resultados del modelo al no normalizar la base de datos, se obtienen resultados peores al anterior, por lo tanto, se decidió que la normalización podría mejorar los modelos. Sin embargo el tiempo de entrenamiento aumentó.

TABLE II: XGB sin normalizar

	MSE	**MAE**
Conjunto Validación	754.21	22.42
Conjunto Prueba	295.63	12.68
Tiempo = 111s		

Después, se aplicó PCA para observar su impacto en el rendimiento del modelo.

TABLE III: XGB PCA

	MSE	**MAE**
Conjunto Validación	623.9	20.61
Conjunto Prueba	378.04	15.76
Tiempo = 56		

Posteriormente, la base de datos se dividió en 3, un 60% para el entrenamiento, un 20% de validación y 20% de prueba. Esto conjuntos fueron normalizados y utilizados para el entrenamiento de los modelos.

Como se estableció en los objetivos, se quiere analizar el impacto de las dos nuevas columnas codificadas, por lo tanto, se tendrán dos subsecciones, los resultados de los modelos cuyo dataset poseen los atributos codificados y los que no.

A. Codificados

1) Regresión lineal: Con respecto a la regresión lineal, este modelo era el más simple, por lo tanto, no se esperaban buenos resultados de la predicción de las clases, sin embargo, a diferencia de algoritmos mucho más robustos, el añadir las dos columnas *mean_popularity* y *artists_code* potenciaron el modelo y se obtuvieron resultados mucho mejores de lo esperado. A continuación se presentan los resultados obtenidos del modelo de predicción.

TABLE IV: Regresión lineal

	MSE	**MAE**	R^2
Conjunto Validación	132.4	6.02	0.73
Conjunto Prueba	133.72	6.08	0.74
Tiempo entrenamiento = 0.06s			

2) XGB: El modelo XGB, se compone de múltiples arboles de decisiones, este algoritmo fue elegido debido a su capacidad para trabajar con una gran cantidad de datos.

TABLE V: XGB

	MSE	**MAE**	R^2
Conjunto Validación	119.68	5.99	0.75
Conjunto Prueba	120.83	6.05	0.75
Tiempo entrenamiento = 8.57s			

3) Gradient Boosting: Al igual que XGB basa su funcionamiento de igual manera en árboles de decisiones. Este método fue elegido ya que posee menor riesgo de sobre ajuste.

TABLE VI: Gradient Boosting

	MSE	**MAE**	R^2
Conjunto Validación	131.07	6.02	0.73
Conjunto Prueba	132.7	6.07	0.73
Tiempo entrenamiento = 33.46s			

4) Random Forest: Como este modelo es más robusto, deberá procesar una gran cantidad de datos, por lo tanto, se realizará PCA(10), es decir, una reducción de dimensionalidad para evitar el sobreconsumo de recursos computacionales. A continuación los resultados:

TABLE VII: Random Forest

	MSE	**MAE**	R^2
Conjunto Validación	115.44	5.31	0.76
Conjunto Prueba	118.51	5.38	0.76
Tiempo entrenamiento = 105s			

5) SVR: Debido a la gran eficiencia de la regresión lineal, se decidió utilizar este modelo debido a que posee un funcionamiento parecido pero a través de un algoritmo mucho más robusto.

TABLE VIII: SVR

	MSE	**MAE**	R^2
Conjunto Validación	134.16	6.06	0.73
Conjunto Prueba	135.55	6.11	0.73
Tiempo entrenamiento = 266.32s			

B. Sin Codificar

A continuación se mostrará el desempeño de los algoritmos donde solo se realizó el preprocesamiento a la base de datos, es decir, no se agregaron las columnas *mean_popularity* y *artists_code*. Esto con el fin de compararlos para poder medir y cuantificar la diferencia del desempeño de cada modelo.

1) Regresión lineal: Nuevamente se realiza una regresión lineal, pero con el dataset original, para así poder medir sus diferencias.

TABLE IX: Regresión lineal sin codificar

	MSE	**MAE**	R^2
Conjunto Validación	487.7	18.43	0.02
Conjunto Prueba	481.6	18.36	0.02
Tiempo entrenamiento = 0.06s			

2) XGB: Se entrena el modelo para medir las diferencias con el otro dataset.

TABLE X: XGB sin codificar

	MSE	**MAE**	R^2
Conjunto Validación	377.78	15.61	0.24
Conjunto Prueba	372.56	15.48	0.24
Tiempo entrenamiento = 10.2s			

3) Gradient Boosting: El modelo se entrena para comparar sus predicciones con otro conjunto de datos.

TABLE XI: Gradient Boosting sin codificar

	MSE	**MAE**	R^2
Conjunto Validación	455.81	17.58	0.08
Conjunto Prueba	449.38	17.47	0.09
Tiempo entrenamiento = 32.38s			

4) Random Forest: Se construye una grilla para buscar los mejores parámetros para el entrenamiento del modelo. Posterior a ello, se realiza PCA (10).

TABLE XII: Random Forest

	MSE	**MAE**	R^2
Conjunto Validación	252.9	11.5	0.5
Conjunto Prueba	256.57	11.54	0.48
Tiempo entrenamiento = 94.8s			

5) SVR: El entrenamiento del modelo permitirá comparar sus predicciones con otro conjunto de datos.

TABLE XIII: SVR

	MSE	**MAE**	R^2
Conjunto Validación	461.54	17.23	0.07
Conjunto Prueba	449.98	17.05	0.09
Tiempo entrenamiento = 262.93s			

C. Comparaciones

1) Comparaciones cuantificadas: Al observar y comparar las tablas es posible notar claras diferencias entre lo modelos de la primera sección y la segunda. Para este caso solo se compararán los resultados obtenidos en el conjunto de prueba. A continuación se presentarán las tablas que cuantificarán las diferencias al utilizar las codificaciones.

TABLE XIV: Diferencias regresión lineal

	MSE	**MAE**	R^2
Conjunto Prueba	347.88	12.28	0.72

TABLE XV: Diferencias XGB

	MSE	**MAE**	R^2
Conjunto Prueba	255.73	9.43	0.51

TABLE XVI: Diferencias Gradient Boosting

	MSE	**MAE**	R^2
Conjunto Prueba	316.68	11.4	0.64

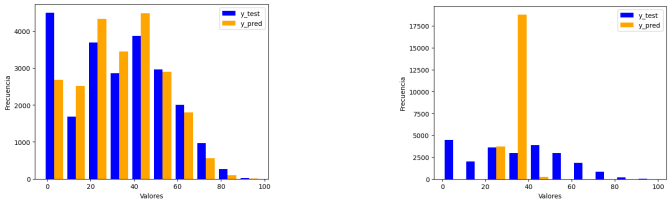
TABLE XVII: Diferencias Random Forest

	MSE	**MAE**	R^2
Conjunto Prueba	138.06	6.16	0.28

TABLE XVIII: Diferencias SVR

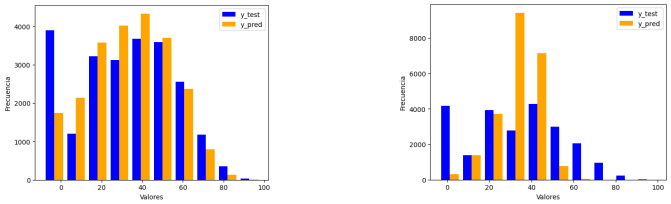
	MSE	**MAE**	R^2
Conjunto Prueba	314.43	10.94	0.64

2) Comparación de gráficos: A continuación se presenta un histograma de las variables *y_test* y *y_pred*, es decir, las etiquetas reales y predecidas, y así poder aterrizar aún más la precisión de la predicción. A la izquierda se encuentra el modelo entrenado con el dataset con atributos codificados y a la derecha el modelo con un preprocesamiento normal.



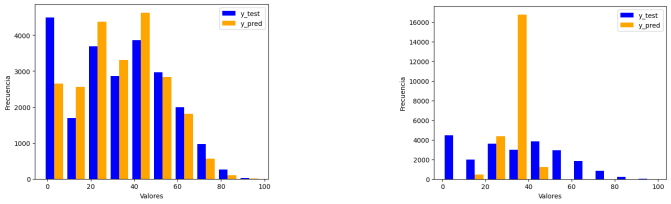
(a) Regresión lineal 1. (b) Regresión lineal 2.

Fig. 9: Comparación regresión lineal.



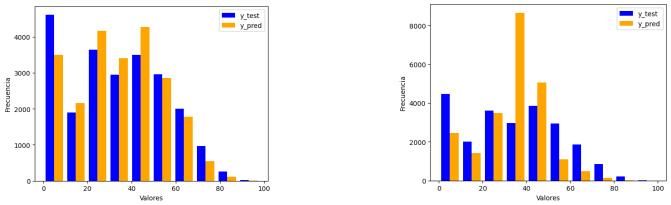
(a) XGB 1. (b) XGB 2.

Fig. 10: Comparación XGB.



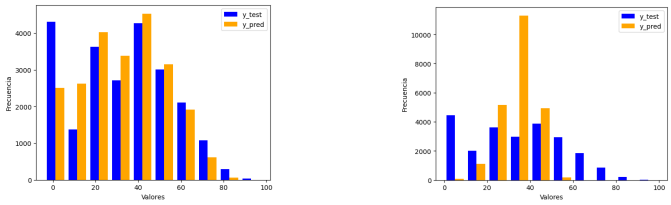
(a) Gradient Boosting 1. (b) Gradient Boosting 2.

Fig. 11: Comparación Gradient Boosting.



(a) Random Forest 1. (b) Random Forest 2.

Fig. 12: Comparación Random Forest.



(a) SVR 1. (b) SVR 2.

Fig. 13: Comparación SVR.

VII. Análisis de resultados

Teniendo los resultados de los modelos y analizando el rendimiento estos, es posible observar que al introducir las dos columnas nuevas *mean_popularity* y *artist_code* los modelos de predicción mejoran significativamente, se observa una mejora de MSE de 347.88, MAE 12.28 y R^2 0.72, en el caso de la regresión lineal.

Al momento de elegir el mejor modelo, es cuestión de observar las métricas utilizadas y los gráficos de las etiquetas presentados. Es posible observar que el modelo Random Forest, fue el mejor en todas las métricas utilizadas, es decir, en MSE, MAE y R^2.

Ahora bien, ¿Qué quiere decir que el MAE del random forest sea 5.3?. En base al rango de la popularidad ([1-100]), el valor MAE del modelo elegido, significa que el algoritmo posee un 10.3% de inexactitud, esto ya que al predecir la clase del dato, tiene un error de ±5.3. Si bien este número es relativamente alto en un problema de regresión, la implementación de las columnas codificadas, mejoró en 6.27 el MAE de la predicción. Por ejemplo, si no se hubieran creado las columnas codificadas, se tendría un error de 23.08%, es decir, si el modelo asigna como popularidad 40, podría existir un rango de inexactitud entre [28.46-51.54]. Mientras que ahora, si el modelo predice una popularidad de 90, el rango de inexactitud es [84.7-95.3], es decir, una mejora muy significativa.

VIII. Conclusiones

Principalmente, se cumplieron los objetivos definidos en un principio, es decir obtener modelos buenos para la base de datos y fue posible cuantificar la mejora de añadir las columnas codificadas.

Si bien algunas desventajas de agregar las columnas codificadas fue que el tiempo que demora en correr el algoritmo es muy alto, en comparación al segundo algoritmo, sin embargo, esto se compensó con una precisión muchísimo mayor.

Finalmente, se obtuvieron mejoras significativas al agregar las dos columnas, fue posible cuantificar y observar la precisión en las predicciones de los modelos. Importante, se implementaron muchos otros modelos mucho más robustos, como redes neuronales, sin embargo los resultados des estos algoritmos no fueron mejor que los vistos en este informe.

REFERENCES

[1] SitioBigData. (2018). "Aprendizaje automático y las métricas de regresión". [Online]. Disponible en: https://sitiobigdata.com/2018/08/27/machine-learning-metricas-regresion-mse/. [Consultado el 23 de Julio del 2023].

[2] SitioBigData. (2019). "Función de pérdida en Machine Learning". [Online]. Disponible en: https://sitiobigdata.com/2019/12/24/funciones-comunes-de-perdida-en-el-aprendizaje-automatico/. [Consultado el 23 de Julio del 2023].

[3] IaArtificial. (2020). "Regresión lineal: Teoría y ejemplos en Python". [Online]. Disponible en: https://www.iartificial.net/regresion-lineal-con-ejemplos-en-python/. [Consultado el 23 de Julio del 2023].

[4] ResearchGate. (2019). "A hybrid ensemble method for pulsar candidate classification". [Online]. Disponible en: https://www.researchgate.net/figure/A-general-architecture-of-XGBoost_fig3_335483097. [Consultado el 23 de Julio del 2023].