Inteligencia Artificial
Act 12: Programando Arbol de Decisión en Python

Estudiante: José Luis Calderón Galarza - 2132939
Docente: Luis Ángel Gutiérrez Rodríguez

1 Introducción

Los árboles de decisión son un método de aprendizaje supervisado utilizado tanto para clasificación como para regresión. Su estructura jerárquica permite tomar decisiones basadas en un conjunto de reglas derivadas de los datos de entrenamiento. Son ampliamente utilizados en análisis de datos y predicciones debido a su interpretabilidad y facilidad de implementación.

2 Metodología

En esta investigación, se implementó un **árbol de decisión** para analizar datos de artistas en el ranking de Billboard. El proceso metodológico seguido se detalla a continuación:

2.1 Importación de librerías y carga de datos

Para la manipulación de los datos se utilizó la librería `pandas`, mientras que `seaborn` y `matplotlib` se emplearon para la visualización. La clasificación se realizó mediante `scikit-learn`.

```
import numpy as np
import pandas as pd
import seaborn as sb
import matplotlib.pyplot as plt
from sklearn import tree
from sklearn.tree import plot_tree
from sklearn.model_selection import KFold

# Configuración inicial de los gráficos
plt.rcParams['figure.figsize'] = (16, 9)
plt.style.use('ggplot')

# Carga del dataset
artists_billboard = pd.read_csv("artists_billboard_fix3.csv")
print(artists_billboard.shape)
print(artists_billboard.head())

top_count = artists_billboard.groupby('top').size()
print(top_count)
```

2.2 Exploración y visualización de datos

Se realizaron diferentes visualizaciones para comprender mejor la distribución de las variables.

```
sb.catplot(x='top', hue='top', data=artists_billboard, kind="count")
plt.show()

sb.catplot(x='genre', hue='genre', data=artists_billboard, kind="count", aspect=3)
plt.show()
```

2.3 Preprocesamiento de datos

Se mapearon variables categóricas a valores numéricos y se imputaron datos faltantes para preparar la data para el modelo de aprendizaje automático.

```
artists_billboard['moodEncoded'] = artists_billboard['mood'].map({...}).astype(int)
artists_billboard['genreEncoded'] = artists_billboard['genre'].map({...}).astype(int)
# Otras transformaciones necesarias
```

Se también aplicó una estrategia para manejar valores nulos en la edad de los artistas.

```
artists_billboard['edad_en_billboard'] = artists_billboard.apply(lambda x:calcula_edad(x['aniol

age_avg = artists_billboard['edad_en_billboard'].mean()
age_std = artists_billboard['edad_en_billboard'].std()
age_null_count = artists_billboard['edad_en_billboard'].isnull().sum()
age_null_random_list = np.random.randint(age_avg - age_std, age_avg + age_std, size=age_null_c

artists_billboard.loc[np.isnan(artists_billboard['edad_en_billboard']), 'edad_en_billboard'] =
artists_billboard['edad_en_billboard'] = artists_billboard['edad_en_billboard'].astype(int)
```

2.4 Entrenamiento del árbol de decisión

El modelo se entrenó usando el criterio de entropía y se probó con diferentes profundidades.

```
decision_tree = tree.DecisionTreeClassifier(criterion='entropy', max_depth=4)
decision_tree.fit(X_train, y_train)
```

Se realizó una búsqueda de la mejor profundidad a partir de una validación cruzada:

```
cv = KFold(n_splits=10)
accuracies = []
max_attributes = len(list(artists_encoded))
depth_range = range(1, max_attributes + 1)

for depth in depth_range:
    fold_accuracy = []
    tree_model = tree.DecisionTreeClassifier(criterion='entropy', max_depth=depth)
    for train_fold, valid_fold in cv.split(artists_encoded):
        f_train = artists_encoded.loc[train_fold]
```

```
        f_valid = artists_encoded.loc[valid_fold]

        model = tree_model.fit(X=f_train.drop(['top'], axis=1), y=f_train["top"])
        valid_acc = model.score(X=f_valid.drop(['top'], axis=1), y=f_valid["top"])
        fold_accuracy.append(valid_acc)

    avg = sum(fold_accuracy) / len(fold_accuracy)
    accuracies.append(avg)
```

2.5 Evaluación del modelo

Se evaluó la precisión del modelo entrenado y se generó una visualización del árbol de decisión.

```
plt.figure(figsize=(20,10))
plot_tree(decision_tree, feature_names=list(artists_encoded.drop(['top'], axis=1)), class_names
plt.show()
```

El modelo también se usó para realizar predicciones sobre artistas específicos:

```
# Predecir artista Camila Cabello
x_test = pd.DataFrame(columns=('top','moodEncoded', 'tempoEncoded', 'genreEncoded','artist_type
x_test.loc[0] = (1,5,2,4,1,0,3)
y_pred = decision_tree.predict(x_test.drop(['top'], axis=1))
print("Prediccion: " + str(y_pred))
```

3 Resultados

3.1 Distribución de artistas según si llegaron al top o no

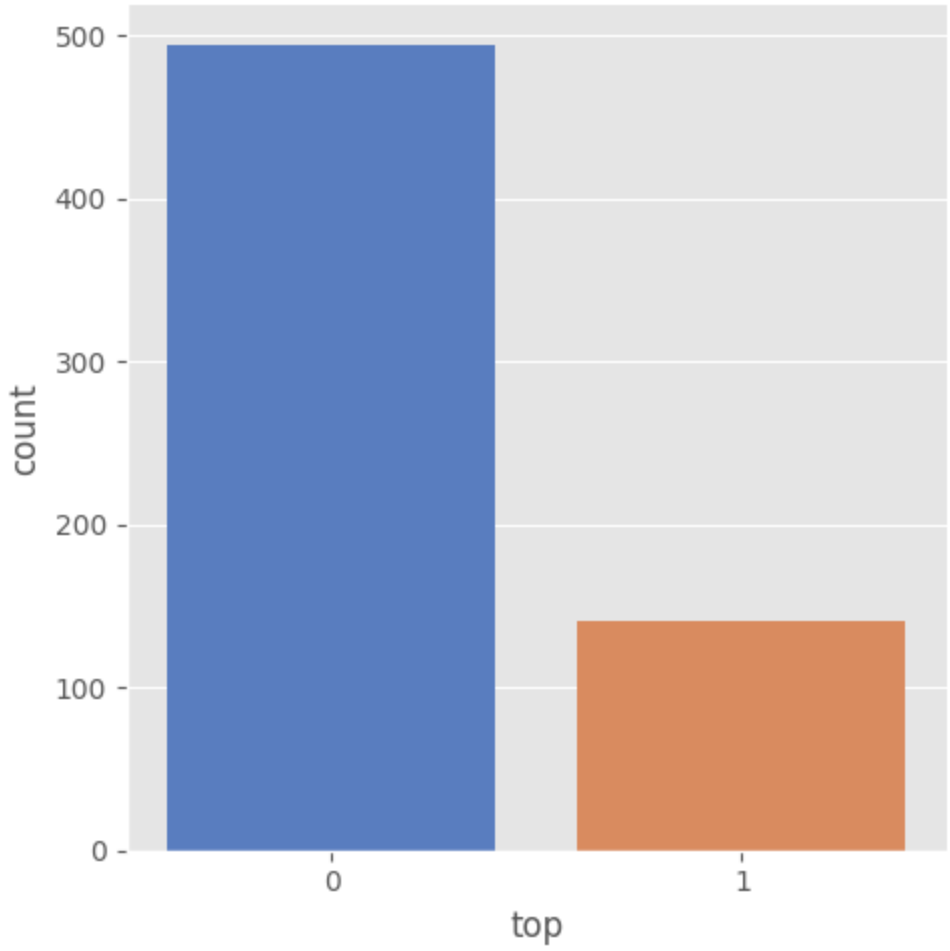

Figure 1: Distribución de artistas según si llegaron al top de Billboard.

3.2 Distribución de artistas por tipo de artista

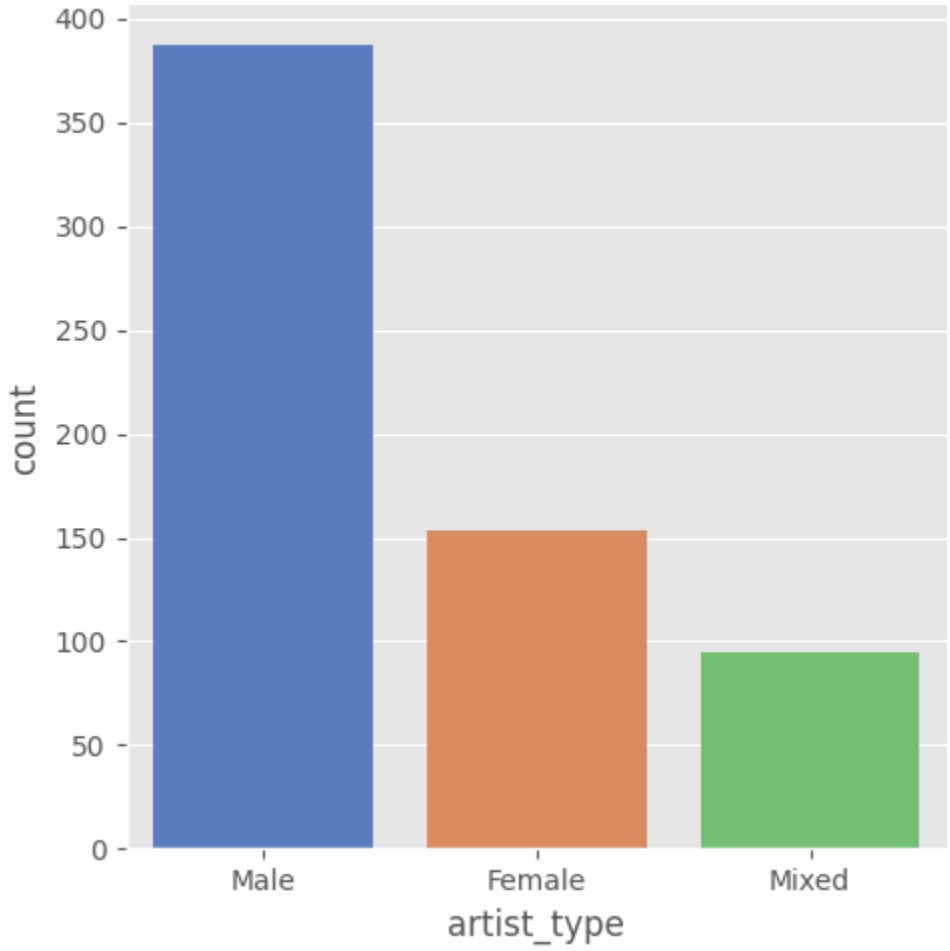

Figure 2: Distribución de artistas por tipo de artista (masculino, femenino, mixto).

3.3 Distribución de artistas según estado de ánimo

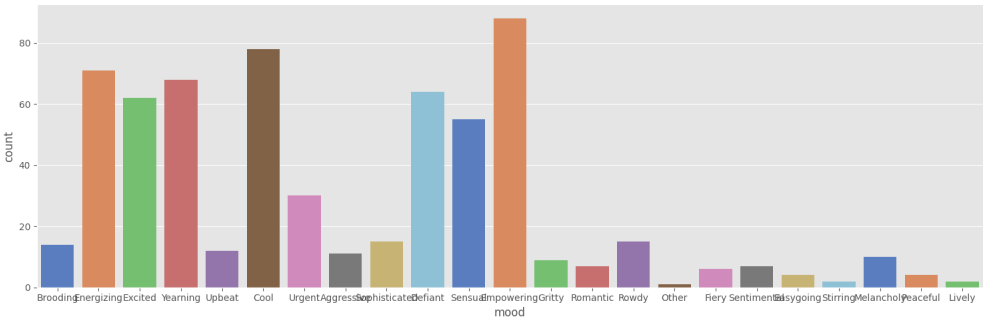

Figure 3: Distribución de artistas según estado de animo.

3.4 Distribución de artistas según tempo de la canción

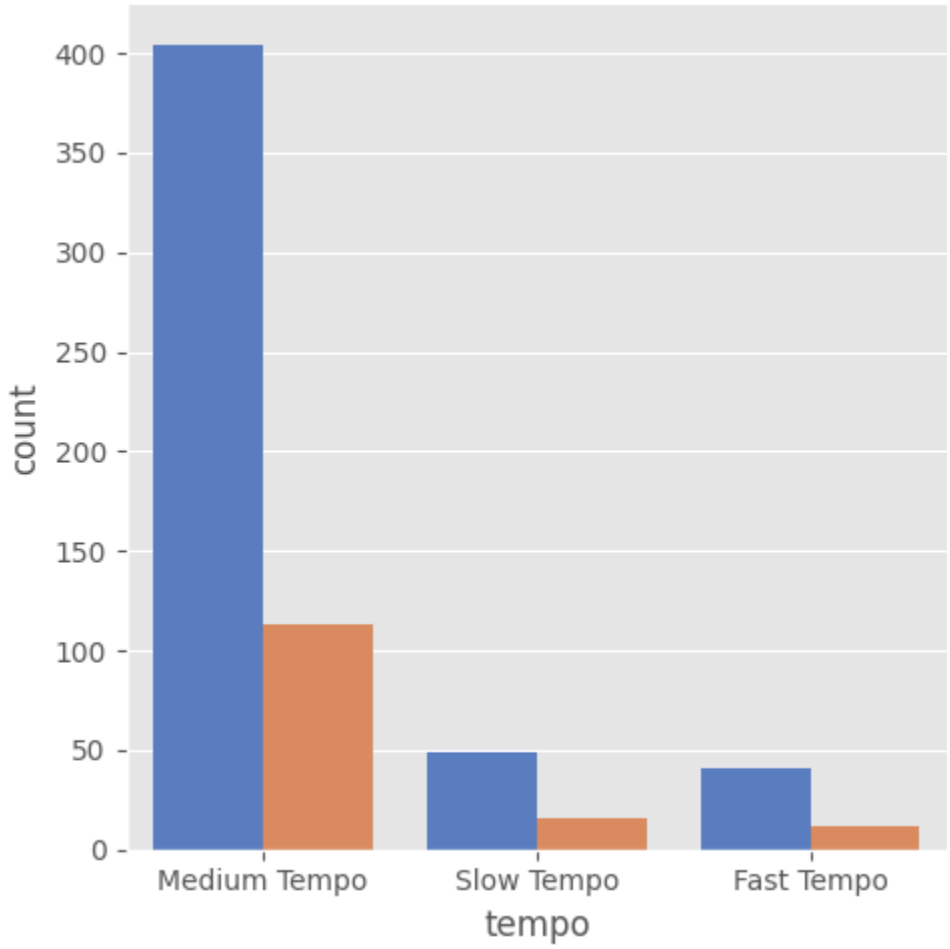

Figure 4: Distribución de artistas según tempo de la canción.

3.5 Distribución de artistas según género musical

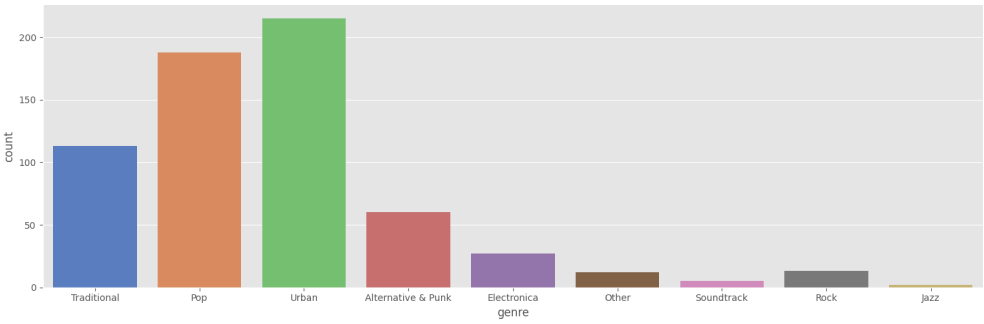

Figure 5: Distribución de artistas según género musical.

3.6 Distribución de artistas según año de nacimiento

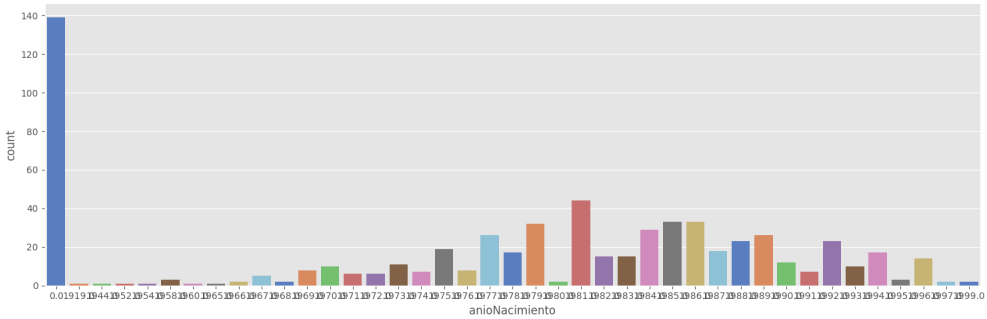

Figure 6: Distribución de artistas según año de nacimiento.

3.7 Relación entre fecha en Billboard y duración de la canción

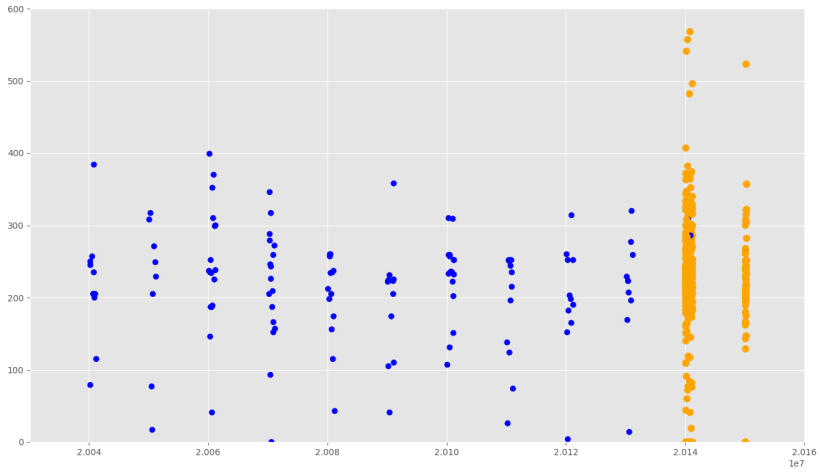

Figure 7: Relación entre la fecha en Billboard y la duración de la canción.

3.8 Visualización de edades con asignación de valores nulos

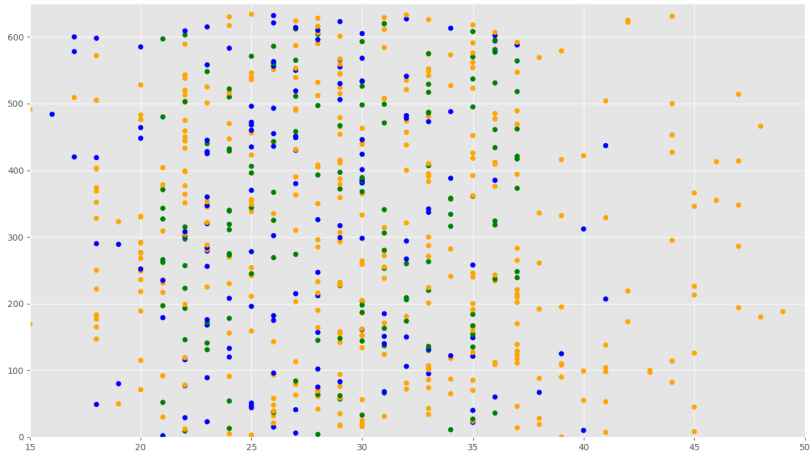

Figure 8: Visualización de edades con asignación de valores nulos.

3.9 Gráfico del árbol de decisión entrenado

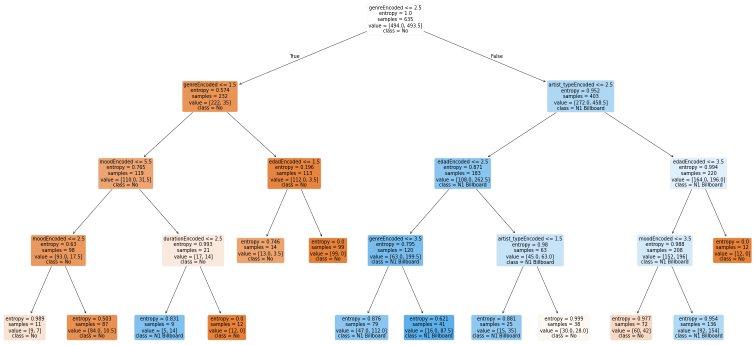

Figure 9: Gráfico del árbol de decisión entrenado.

4 Conclusión

En esta actividad se implementó un árbol de decisión para analizar el desempeño de artistas en el ranking de Billboard. Se observó que la profundidad óptima del árbol mejora la precisión del modelo y que ciertas variables, como el género musical y el estado de ánimo, tienen una influencia significativa en la clasificación. La metodología aplicada permite obtener un modelo interpretable y de alto valor analítico.