INTELIGENCIA ARTIFICIAL

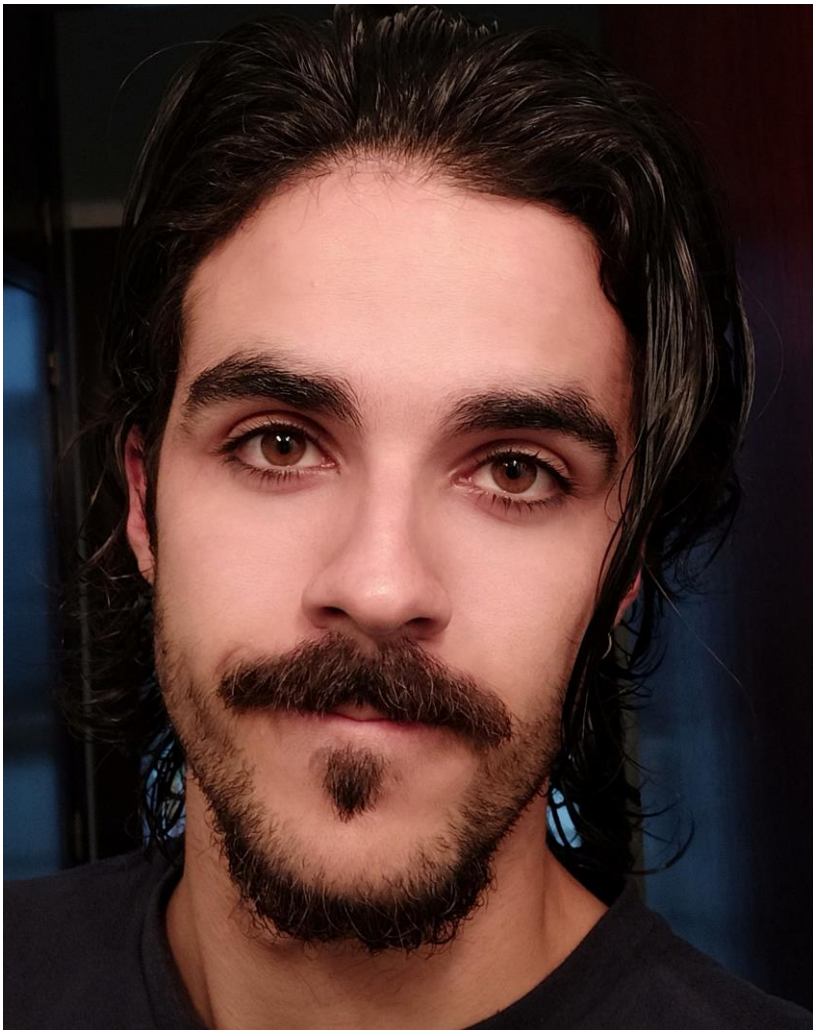
PROYECTO

Rafael Rodriguez Vazquez

New Technology School

Tokio.

New
Technology
School



Rafael Rodriguez Vazquez

RESPONSABLE DEL DESARROLLO DE LA APLICACIÓN

Fecha: 12 / 02 / 2024

INTELIGENCIA ARTIFICIAL
PROYECTO FINAL
Nombre y Apellidos

Tokio.

DESARROLLO

****Realmente el proyecto está desarrollado y bien explicado en el fichero de jupyter. No obstante, por buenas prácticas haré un repaso con imágenes y copiare las conclusiones para entregar este formato****

Generalidades:

En este proyecto vamos a trabajar con un conjunto de datos musicales que consta de canciones registradas en Spotify, la plataforma de música online más reconocida por oyentes y artistas; los datos han sido obtenidos de la plataforma de Kaggle: (https://www.kaggle.com/datasets/maharshipandya/-spotify-tracks-dataset).

En nuestros datos podemos encontrar hasta 125 géneros musicales diferentes, acompañados de 20 columnas de datos con las características que acompañan al género y la popularidad que ha alcanzado.

Objetivo:

El principal objetivo del proyecto es comprobar que datos hacen que una canción alcance más popularidad, teniendo a esta como nuestra variable objetivo. La idea principal de este objetivo es que un artista pueda identificar qué características hacen que una canción pueda ser más o menos popular, debido a que entendemos que popularidad = éxito, por lo que realmente estamos comprobando cuanto éxito alcanzara un artista según las características que ofrece en sus canciones.

Dentro del desarrollo del mismo proyecto, resolvemos diferentes objetivos técnicos como puede ser:

- Classificación, ya que clasificamos y categorizamos nuestros datos.
- Correlación, comparamos entre sí las diferentes características, teniendo como eje la popularidad para determinar si la relación entre ellas es directamente proporcional a la solución de nuestra variable objetivo.
- Conversión de datos, algo necesario para la preparación de los datos de cara a entrenar el modelo.
- Gráficos, establecer criterios visuales que nos permitan entender mejor los datos que tratamos.

Modelo:

Queremos ver si los modelos seleccionados son capaces de predecir que música o canciones tienen más probabilidad de tener éxito, según la relación y predicción de sus características.

Esto permitirá a artistas direccionar las capacidades creativas para establecer un balance al realizar canciones propias, donde el arte y los conjuntos estadísticos forman parte del equipo necesario para que alcancen un mayor número de oyentes.

In [2]:
```
# Establecemos una ruta hacia el csv que contiene la informacion
ruta_spoty="C:\\Users\\Kocxi\\AnacondaProjects\\Inteligencia artificial\\Proyecto final\\spoty_dataset.csv"
spoty_data=pd.read_csv(ruta_spoty, delimiter=',', index_col=0)
# [index_col=0] hace referencia a poner la primera columna como nuestro indice, porque aparecia una columna generada
# automaticamente de nombre unnamed, lo que duplicaba el indice.

# Vamos a mostrar los 10 primeros registros de nuestro dataset.
spoty_data.head(10)
```

Out[2]:

	track_id	artists	album_name	track_name	popularity	duration_ms	explicit	danceability	energy	key	loudness	mode	sp
0	5SuOikwiRyPMVoIQDJUgSV	Gen Hoshino	Comedy	Comedy	73	230666	False	0.676	0.4610	1	-6.746	0	
1	4qPNDBW1i3p13qLCt0Ki3A	Ben Woodward	Ghost (Acoustic)	Ghost - Acoustic	55	149610	False	0.420	0.1660	1	-17.235	1	
2	1iJBSr7s7jYXzM8EGcbK5b	Ingrid Michaelson;ZAYN	To Begin Again	To Begin Again	57	210826	False	0.438	0.3590	0	-9.734	1	
3	6lfxq3CG4xtTiEg7opyCyx	Kina Grannis	Crazy Rich Asians (Original Motion Picture Sou...	Can't Help Falling In Love	71	201933	False	0.266	0.0596	0	-18.515	1	
4	5vjLSffimiIP26QG5WcN2K	Chord Overstreet	Hold On	Hold On	82	198853	False	0.618	0.4430	2	-9.681	1	
5	01MVOl9KtVTNfFiBU9I7dc	Tyrone Wells	Days I Will Remember	Days I Will Remember	58	214240	False	0.688	0.4810	6	-8.807	1	

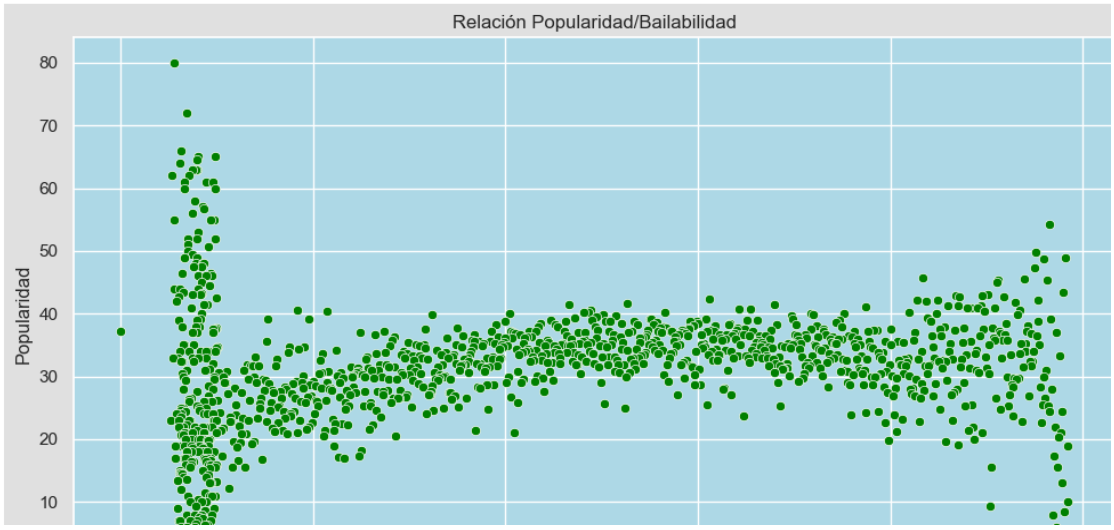
Revisión del dataframe.

In [4]:
```
# Comprobamos informacion general del dataset.
spoty_data.info()
```

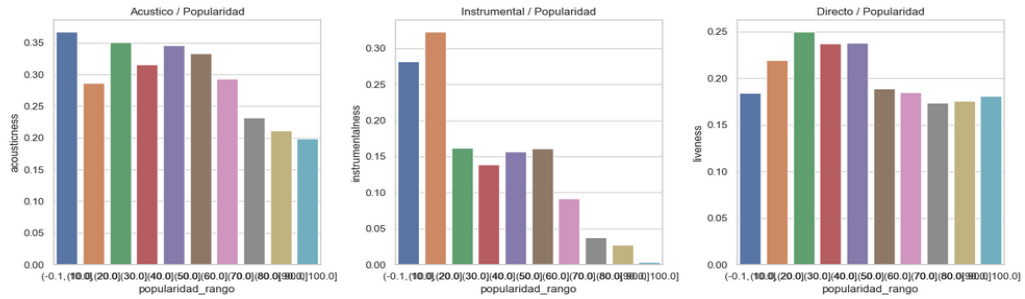
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 114000 entries, 0 to 113999
Data columns (total 20 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   track_id          114000 non-null  object
 1   artists           113999 non-null  object
 2   album_name        113999 non-null  object
 3   track_name        113999 non-null  object
 4   popularity        114000 non-null  int64
 5   duration_ms       114000 non-null  int64
 6   explicit          114000 non-null  bool
 7   danceability      114000 non-null  float64
 8   energy            114000 non-null  float64
 9   key               114000 non-null  int64
 10  loudness          114000 non-null  float64
 11  mode              114000 non-null  int64
 12  speechiness       114000 non-null  float64
 13  acousticness      114000 non-null  float64
```

Comprobar la información genérica del dataframe.

In [9]:
```
# Relacionamos las canciones que son bailables con la popularida que obtinenen:

danceability_popularity=spoty_data[['danceability','popularity']].groupby(['danceability'], as_index=False).mean().sort_v

sns.set(rc={"axes.facecolor":"#ADD8E6","figure.facecolor":"#E0E0E0"})
plt.figure(figsize=(10, 6))
sns.scatterplot(x='danceability', y='popularity', color='green', data=danceability_popularity)
plt.xlabel('Bailabilidad')
plt.ylabel('Popularidad')
plt.title('Relación Popularidad/Bailabilidad')
plt.tight_layout()
plt.show()

# Pd:por la gran cantidad de datos reflejados en esta correlacion, el grafico ha sido cambiado de barras a gráfivo de dis
# (scatter plot)

danceability_popularity
```




```
# Crear subplots para cada característica
fig, axes = plt.subplots(1, 3, figsize=(15, 5))

# Creamos los gráficos de barras:
sns.barplot(x='popularidad_rango', y='acousticness', data=popularity_avg, ax=axes[0])
axes[0].set_title('Acustico / Popularidad')

sns.barplot(x='popularidad_rango', y='instrumentalness', data=popularity_avg, ax=axes[1])
axes[1].set_title('Instrumental / Popularidad')

sns.barplot(x='popularidad_rango', y='liveness', data=popularity_avg, ax=axes[2])
axes[2].set_title('Directo / Popularidad')

plt.tight_layout()
plt.show()

acou_inst_live_popularity

# y la otra de valencia y con eso concluir
```



Out[12]:

	acousticness	instrumentalness	liveness	popularity	popularidad_rango
0	0.01300	0.000005	0.2660	100.0	(90.0, 100.0]
1	0.01250	0.033000	0.2300	99.0	(90.0, 100.0]
2	0.58300	0.000002	0.2180	98.0	(90.0, 100.0]

Relación de características.

In [16]:
```
# En esta celda de código vamos a transformar los géneros de track_genre, para ello vamos a generar una columna
# para cada genero(114 generos), a valores numericos aptos apra el modelo,
# he decidido separarlo en otra celda porque es un poco más especial que lo anterior.

spoty_data_modelo_encoded = pd.get_dummies(spoty_data_modelo, columns=['track_genre'])

# Aunque solo sea una línea de codigo he utilizado la codificacion one-hot encoding que trae pandas (.get_dummies),
# para transformar las variables categóricas, y así tratar con un gran nº de categorias en los datos.
# De este modo es apto para entrenar a mi modelo. De modo que nuestros datos para modelaje quedarian de la siguiente manera:

spoty_data_modelo_encoded
```

Out[16]:

	popularity	explicit	danceability	energy	key	acousticness	instrumentalness	liveness	valence	track_genre_acoustic	...	track_genre_spanish	track_
0	1	0	0	0	0	0	0	0	0	1	...	0	
1	0	0	0	0	0	0	0	0	0	1	...	0	
2	0	0	0	0	0	0	0	0	0	1	...	0	
3	1	0	0	0	0	0	0	0	0	1	...	0	
4	1	0	0	0	0	0	0	0	0	1	...	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...
113995	0	0	0	0	1	0	0	0	0	0	...	0	
113996	0	0	0	0	0	0	0	0	0	0	...	0	
113997	0	0	0	0	0	0	0	0	0	0	...	0	
113998	0	0	0	0	1	0	0	0	0	0	...	0	
113999	0	0	0	0	0	0	0	0	0	0	...	0	

114000 rows × 123 columns

Preparación de los datos para el modelo.

In [18]:
```
# Dividimos nuestros datos para poder estblecer la prediccion del modelo:

# En Y_train/test, guardamos unicamente la columna de popularidad que es nuestro valor objetivo que queremos predeci
Y_train=spoty_model["popularity"]
Y_test=spoty_model["popularity"]

# En X_train/test, extraemos la columna de popularidad y dejamos todas las caracteristicas vistas anteiormente para
X_train=spoty_model.drop(["popularity"], axis=1)
X_test=spoty_model.drop(["popularity"], axis=1)

# Mostramos nuestro conjunto de datos:
X_train.shape, Y_train.shape, X_test.shape, Y_test.shape
```

Out[18]: ((114000, 122), (114000,), (114000, 122), (114000,))

División de datos de entrenamiento y test.

In [20]:
```
# Modelo: Random Forest

random_forest=RandomForestClassifier(n_estimators=100)

random_forest.fit(X_train,Y_train)

random_forest_prediccion=random_forest.predict(X_test)

print("\nMatriz de confusión:\n")
confusion_matrix_rndfo=confusion_matrix(Y_test, random_forest_prediccion)
print(confusion_matrix_rndfo)

print("\nReporte de clasificacion - Modelo Random Forest:\n")
reporte=classification_report(Y_test, random_forest_prediccion)
print(reporte)


f1=f1_score(Y_test, random_forest_prediccion, average='micro')
print("\nF1-score promedio obtenido de Random Forest:\n")
print(f1)

data_f1_rndfo=round(f1*100, 2)
print("\nResultado guardado para comparar:",data_f1_rndfo)
```

```
Matriz de confusión:

[[99548   882]
 [12231  1339]]

Reporte de clasificacion - Modelo Random Forest:

              precision    recall  f1-score   support

           0       0.89      0.99      0.94    100430
           1       0.60      0.10      0.17     13570

    accuracy                           0.88    114000
   macro avg       0.75      0.54      0.55    114000
weighted avg       0.86      0.88      0.85    114000
```

Entrenamiento de algunos modelos(concretamente Random Forest).

In [24]:
```
# Creamos el nuevo dataframe con los resultados de nuestros modelos, y lo ordenamos segun su resultado más alto
comparacion_modelos=pd.DataFrame({
    "Modelo":["Logistic Regression", "Random Forest","K-Nearest Neighbors(KNN)","Gaussian Naive Bayes","Linear Support Vector Mac
    "Resultado (sobre 100)":[data_f1_logreg, data_f1_rndfo, data_f1_knn, data_f1_gaussian, data_f1_lsvc]
})

comparacion_modelos=comparacion_modelos.sort_values(by="Resultado (sobre 100)", ascending=False)

# Reseteamos el indice para detallarlo más claramente
comparacion_modelos.reset_index(drop=True, inplace=True)

# Mostramos El Dframe.
comparacion_modelos
```

Out[24]:

	Modelo	Resultado (sobre 100)
0	Random Forest	88.50
1	Logistic Regression	88.40
2	Linear Support Vector Machine(LSVC)	88.36
3	K-Nearest Neighbors(KNN)	68.43
4	Gaussian Naive Bayes	52.48

Comparación de rendimiento de los modelos.

CONCLUSIONES

El objetivo principal del proyecto consiste en evaluar e intentar predecir la popularidad dentro del conjunto de canciones que tenemos. Entendiendo la popularidad como el éxito que posiblemente tendrá una canción.

Para ello, y después de hacer un filtrado de la información, hemos utilizado diferentes modelos para relacionar si son capazces de establecer que popularidad tendrá una canción teniendo en cuenta las características trabajadas.

Según podemos ver, en el gráfico superior, dentro de los 5 modelos con los que hemos trabajado ha habido tres que han sobresalido notoriamente alcanzando un resultado de hasta un **88% de acierto**. Entendemos que estos modelos son capaces de tener una gran tasa de acierto a la hora de predecir si una canción tendrá o no exito.

Los modelos destacados han sido:

- **Random Forest, con un 88,5% de acierto**, entendemos que este modelo ha sido el que mejor resultado ha mostrado para la predicción debido a la gran cantidad de datos y las realciones complejas entre las características que presenta el conjunto de datos con el que hemos trabajado.
- **Logistic Regression, con un 88,4% de acierto**, muy cercano en cuanto a resultado al modelo anterior, no ha superado el resultado del modelo anterior debido a que funciona muy bien en condiciones binarias, y en este caso se ha encontrado con varias carácterísticas a tener en cuenta para poder establecer la predicción, aún así, ha sido el segundo mejor resultado de nuestro proyecto.
- **Linear Support Vector Machine, con un 88.36% de acierto**, el resultado de la predicción de este modelo me ha sorprendido debido a que ha sido más alto de lo que esperaba, ya que debe de relacionar las carácterísticas entre sí diferenciandolas, pero tras el resultado obtenido podemos definir que la relación entre las características y la variable objetivo (popularidad) se establecen, en mayor medida, de modo lineal.

Por otro lado, tenemos dos modelos que no han obtenido un buen rendimiento en la predicción como los anteriores, con lo que concluimos:

- **K-Nearest Neighbours, con un 68,43% de acierto**, tiene una tasa de acierto más moderada que los modelos anteriores, esto puede deberse a que la popularidad en una cancion no se debe directamente a su relacion con canciones del mismo genero, si no a el conjunto propio de características de una canción.
- **Gaussian Naive Bayes, con un 52,48% de acierto**, es el modelo con un rendimiento más mediocre, posiblemente el resultado debe relacionarse con que este modelo establece que las características son independientes entre sí, y como hemos visto en los modelos anteriores que relacionan las carácteristicas, no es tan válido para este conjunto de datos establecer ese criterio.

Teniendo en cuenta todo el código desarrollado, podemos interpretar ciertas mejoras en el procesamiento de datos, con el fin de mejorar el rendimiento de los modelos. Estas mejoras podrían ser:

- Simplificar más aun los datos con los que opera el modelo, por ejemplo eliminar la columna Key (tonalidad), que puede interpretarse también como un concepto más técnico.
- Establecer una nueva columna que relacione dos o más columnas de datos entre sí, para acotar los datos con los que el modelo trabaja y así poder ser más eficaz en la predicción.
- Elegir otros modelos que operen con gran cantidad de características para poder mejorar el resultado.

Para finalizar, considero que el procesamiento de datos previo al modelaje y predicción ha permitido adquirir un alto rango de aciertos en el conjunto de datos; siendo mi primer proyecto de machine learning esperaba conseguir una tasa de acierto menor, por lo que estoy satisfecho con los resultados.