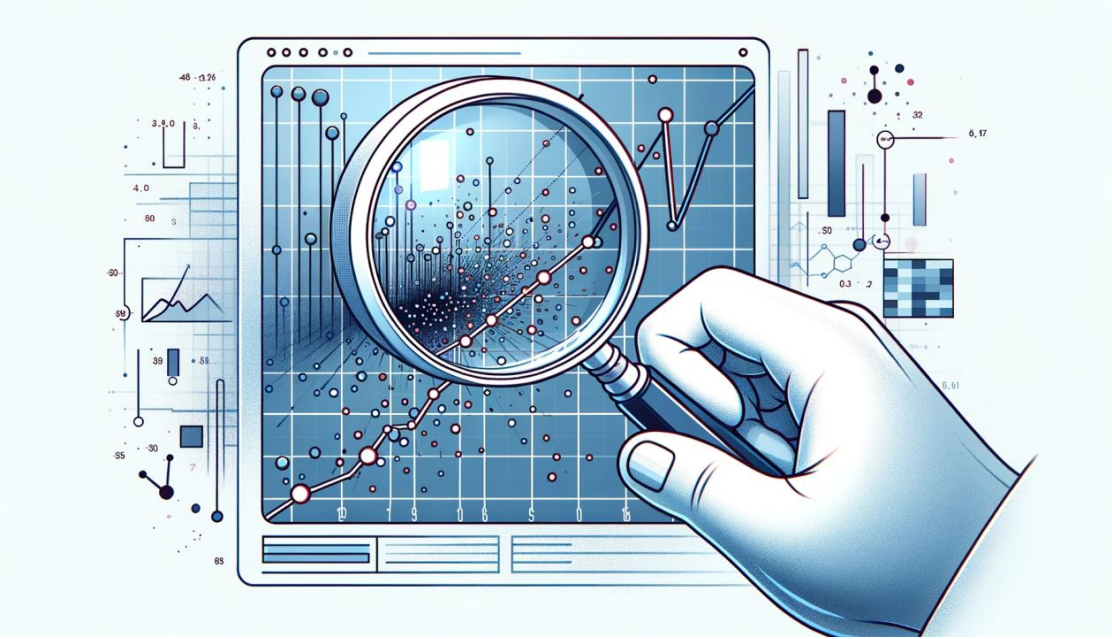
MODELOS DE ML (REGRESIÓN LINEAL) PARA PREDECIR LA POPULARIDAD DE UNA CANCIÓN

En base a atributos musicales.

Julio Gutiérrez *Brayan Barrera* *Camilo Campos*

Introducción

En esta presentación, exploramos la aplicación de modelos de aprendizaje automático (ML) para predecir la popularidad de una canción en función de sus atributos musicales.



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 114000 entries, 0 to 113999
Data columns (total 21 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   Unnamed: 0        114000 non-null  int64
 1   track_id          114000 non-null  object
 2   artists           113999 non-null  object
 3   album_name        113999 non-null  object
 4   track_name        113999 non-null  object
 5   popularity        114000 non-null  int64
 6   duration_ms       114000 non-null  int64
 7   explicit          114000 non-null  bool
 8   danceability      114000 non-null  float64
 9   energy            114000 non-null  float64
 10  key               114000 non-null  int64
 11  loudness          114000 non-null  float64
 12  mode              114000 non-null  int64
 13  speechiness       114000 non-null  float64
 14  acousticness      114000 non-null  float64
 15  instrumentalness  114000 non-null  float64
 16  liveness          114000 non-null  float64
 17  valence           114000 non-null  float64
 18  tempo             114000 non-null  float64
 19  time_signature    114000 non-null  int64
 20  track_genre       114000 non-null  object
dtypes: bool(1), float64(9), int64(6), object(5)
memory usage: 17.5+ MB
```

Dataset Spotify (kaggle.com)

dataset.csv (20.12 MB)

Detail Compact Column

10 of 21 columns

About this file

The data is in tabular format stored as a `CSV` file.

#	track_id	artists	album_name	track_name	popularity	duration_ms	explicit	danceability	ene
number	str	str	str	str	number	number	bool	number	numbe
0 114k	89741 unique values	31438 unique values	46590 unique values	73609 unique values	0 100	0 5.24m	true 9747 9% false 104k 91%	0 0.98	0
0	5SuOikwiRyPMVoIQDJUgSV	Gen Hoshino	Comedy	Comedy	73	230666	False	0.676	0.461
1	4qPNDBW1i3p13qLCt0Ki3A	Ben Woodward	Ghost (Acoustic)	Ghost - Acoustic	55	149610	False	0.42	0.166
2	1iJBSr7s7jYXzM8EGcbK5b	Ingrid Michaelson;ZAYN	To Begin Again	To Begin Again	57	210826	False	0.438	0.359
3	6lfxq3CG4xtTiEg7opyCyx	Kina Grannis	Crazy Rich Asians (Original Motion Picture Soundtrack)	Can't Help Falling In Love	71	201933	False	0.266	0.059
4	5vjLSffimiIP26QG5WcN2K	Chord Overstreet	Hold On	Hold On	82	198853	False	0.618	0.443
5	01MVOl9KtVTNfFiBU9I7dc	Tyrone Wells	Days I Will Remember	Days I Will Remember	58	214240	False	0.688	0.481
6	6Vc5wAMmXdKIAM7WUoEb7N	A Great Big World;Christina	Is There Anybody Out There?	Say Something	74	229400	False	0.407	0.147

Objetivo: predecir la popularidad de una canción

El objetivo principal es utilizar modelos de ML para predecir la popularidad de una canción, utilizando varias características musicales como datos de entrada.

- Esto puede ayudar a los artistas y productores a tomar decisiones basadas en datos.

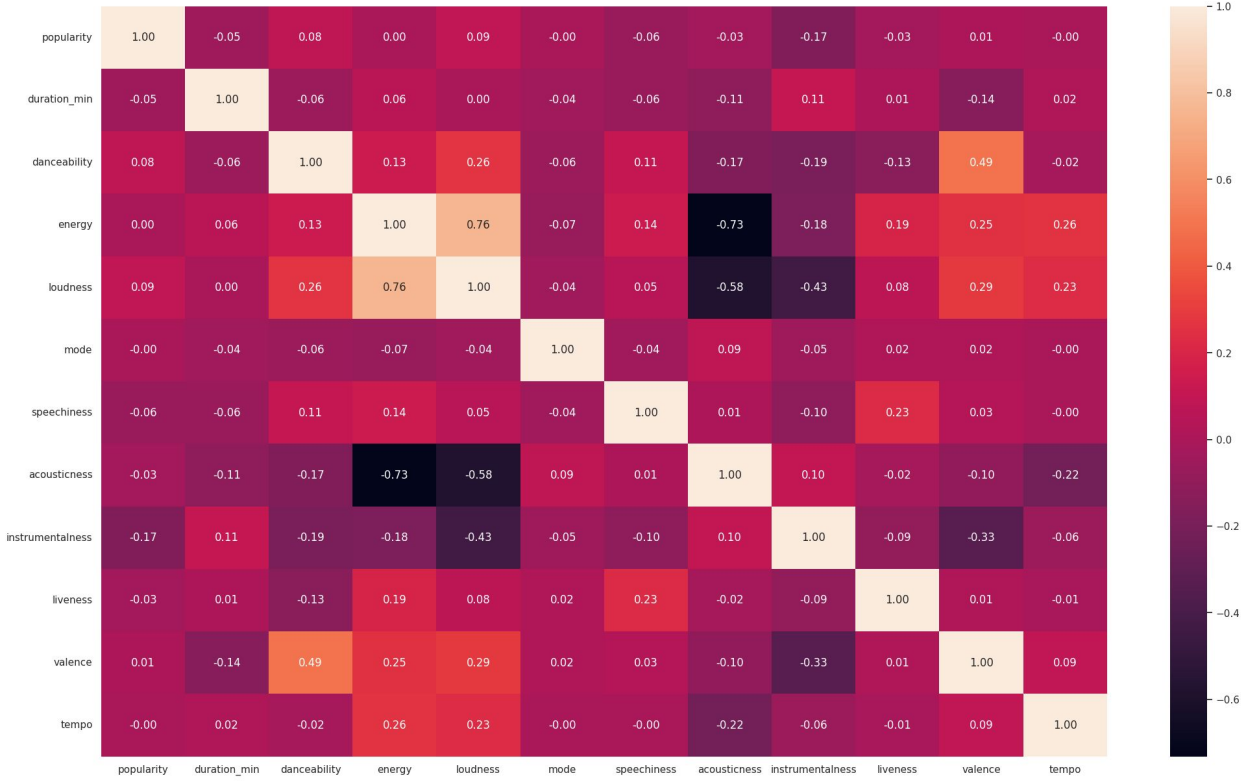
Incluso...¿Hacer música con IA?

Conjunto de datos: conjunto de datos de Spotify Tracks

Preprocesamiento de datos

Eliminación de columnas categóricas, datos nulos y registros duplicados.

Información sobre conjunto datos



Modelos de Machine Learning

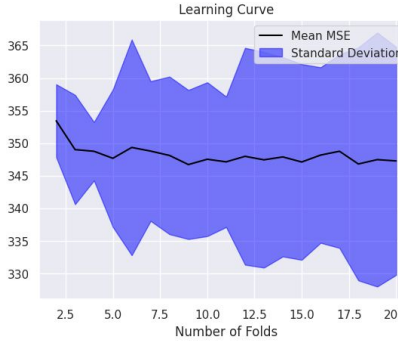
Support Vector Machine

Random Forest

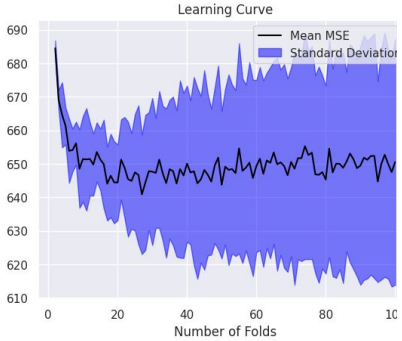
Decision Tree

Learning Curve: Tunning Parameters

Cross validation

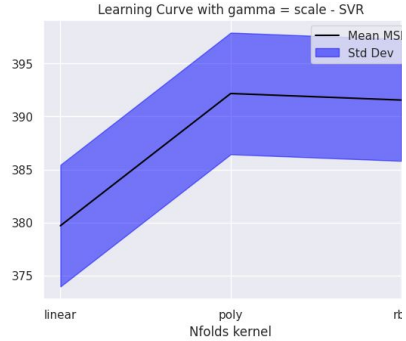


Random Forest



Decision Tree

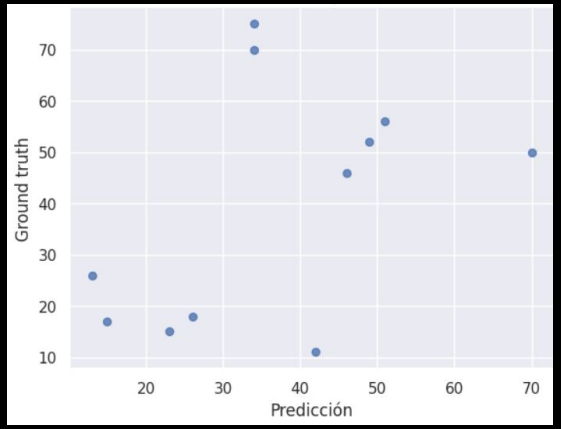
Particionado



Random Forest

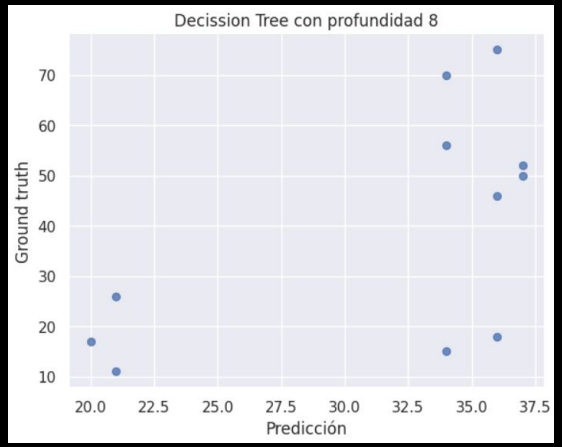
Decision Tree

Modelos de ML



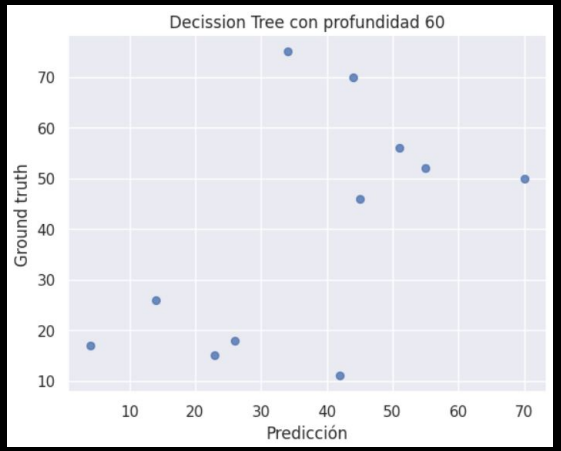
Decision Tree

Parámetros por defecto



Decision Tree

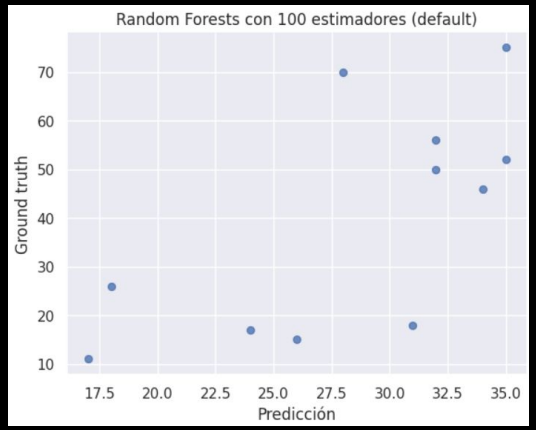
max_depth = 8



Decision Tree

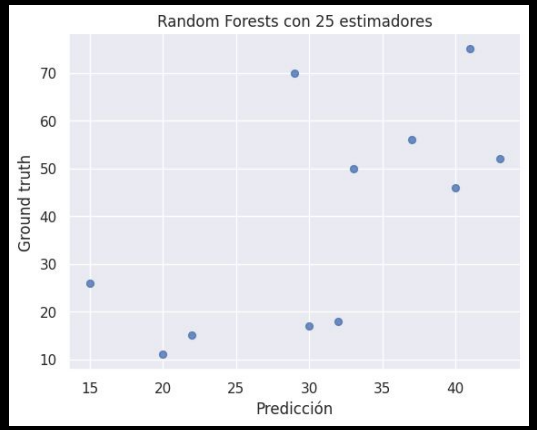
max_depth = 60

Modelos de ML



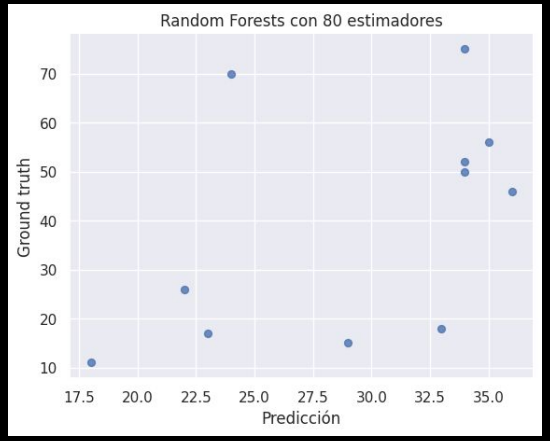
Random Forest

Parámetros por defecto



Random Forest

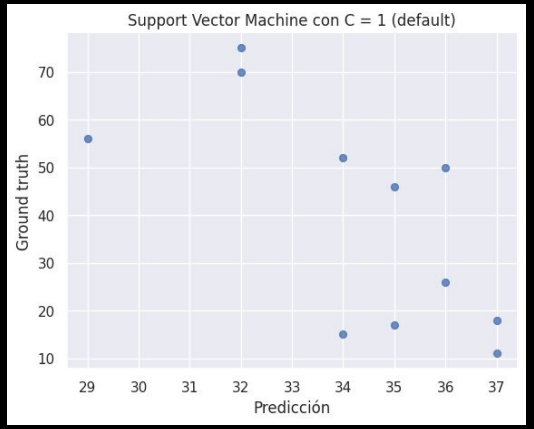
n_estimators = 25



Random Forest

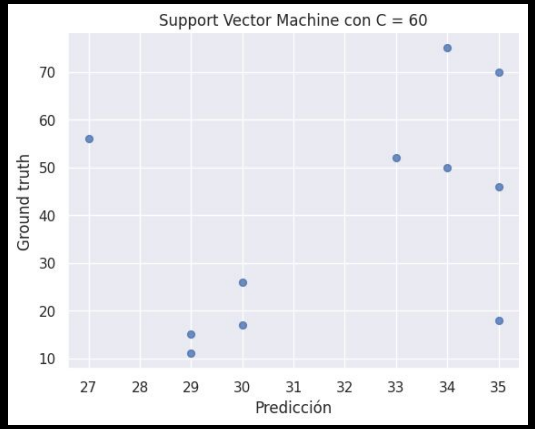
n_estimators = 80

Modelos de ML



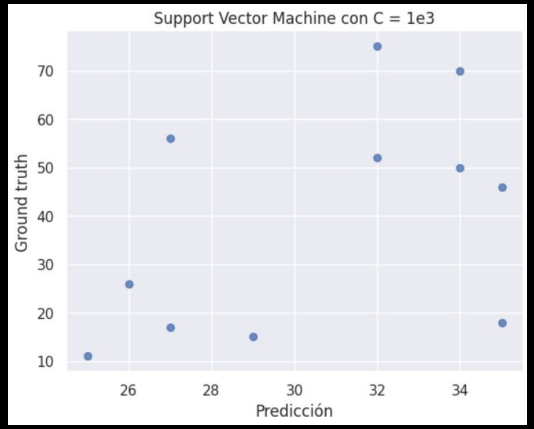
SVM

Parámetros por defecto



SVM

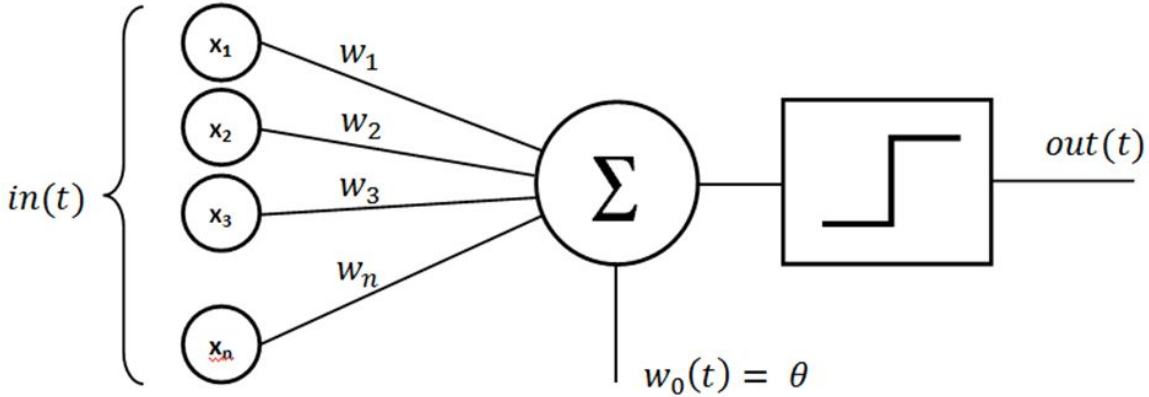
C = 60 y kernel polinomial de grado 4



SVM

max_depth = 60

Red Neuronal (MLP)



Division Conjunto de Datos

```
#@title Particionado del DataFrame

y = data_cleaned["popularity"]
X = data_cleaned.drop("popularity", axis=1)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=21)
```

MLP 3 Capas Ocultas

```python
#@title MLP con 3 Capas Ocultas

print(X_train.shape[1])
model_3 = tf.keras.Sequential([
  tf.keras.layers.Flatten(input_shape=(X_train.shape[1],)),
  tf.keras.layers.Dense(256, activation='relu'),
  tf.keras.layers.Dense(128, activation='relu'),
  tf.keras.layers.Dense(64, activation='relu'),
  tf.keras.layers.Dense(1)
])

model_3.compile(optimizer='adam',
            loss='mean_squared_error',
            metrics=['mse', 'mae'])
model_3.fit(X_train, y_train, epochs=15, verbose=1)
y_pred = model_3.predict(X_test)
```

```
loss: 363.6928 - mae: 15.4243 - mse: 363.6928
```

MLP 6 Capas Ocultas

```python
#@title MLP con 6 Capas Ocultas

y = data_cleaned["popularity"]
X = data_cleaned.drop("popularity", axis=1)

model = tf.keras.Sequential([
  tf.keras.layers.Flatten(input_shape=(X_train.shape[1],)),
  tf.keras.layers.Dense(64, activation='relu'),
  tf.keras.layers.Dense(64, activation='relu'),
  tf.keras.layers.Dense(32, activation='relu'),
  tf.keras.layers.Dense(32, activation='relu'),
  tf.keras.layers.Dense(16, activation='relu'),
  tf.keras.layers.Dense(8, activation='relu'),
  tf.keras.layers.Dense(1)
])

model.compile(optimizer='adam',
              loss='mean_squared_error',
              metrics=['mse', 'mae'])
model.fit(X_train, y_train, epochs=15, verbose=1)
y_pred = model.predict(X_test)
```

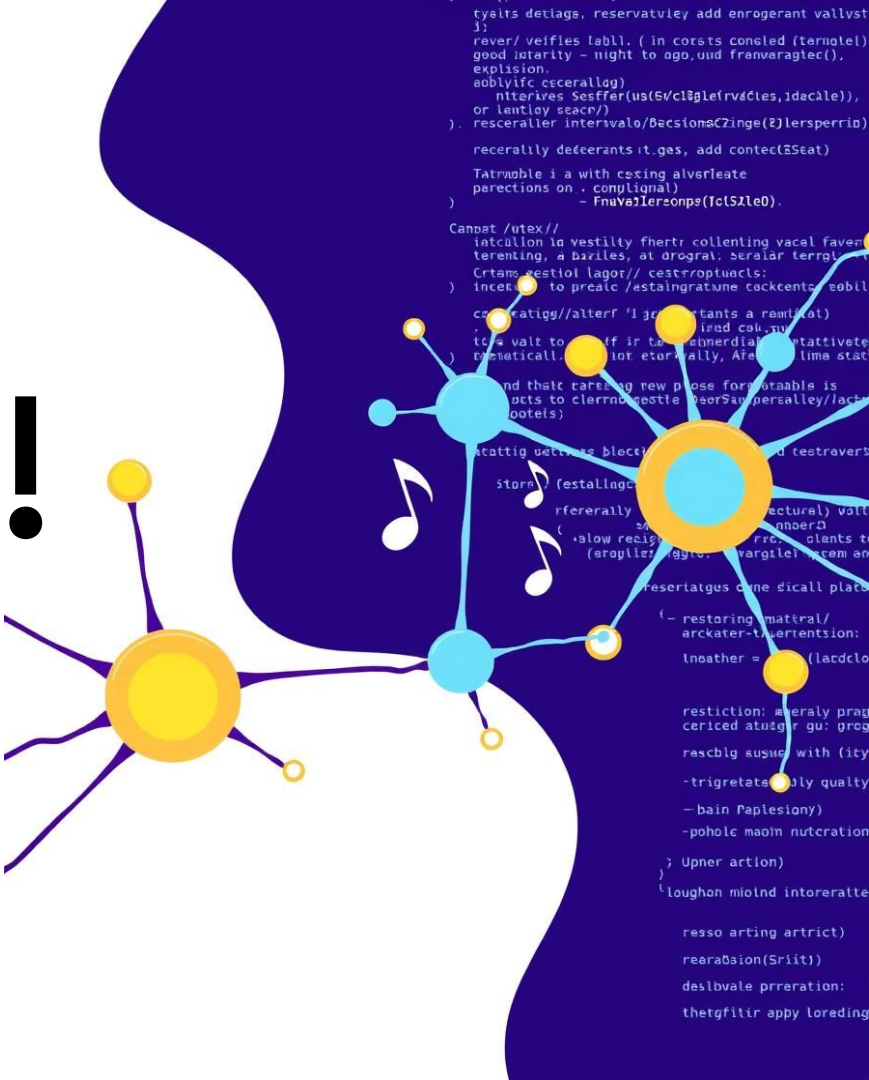
```
loss: 361.3887 - mae: 15.3919 - mse: 361.3887
```

MLP 10 Capas Ocultas

```python
#@title MLP con 10 Capas Ocultas

y = data_cleaned["popularity"]
X = data_cleaned.drop("popularity", axis=1)

model = tf.keras.Sequential([
  tf.keras.layers.Flatten(input_shape=(X_train.shape[1],)),
  tf.keras.layers.Dense(1024, activation='relu'),
  tf.keras.layers.Dense(512, activation='relu'),
  tf.keras.layers.Dense(256, activation='relu'),
  tf.keras.layers.Dense(128, activation='relu'),
  tf.keras.layers.Dense(64, activation='relu'),
  tf.keras.layers.Dense(32, activation='relu'),
  tf.keras.layers.Dense(16, activation='relu'),
  tf.keras.layers.Dense(8, activation='relu'),
  tf.keras.layers.Dense(4, activation='relu'),
  tf.keras.layers.Dense(2, activation='relu'),
  tf.keras.layers.Dense(1)
])

```

```
loss: 364.6494 - mae: 15.4996 - mse: 364.6494
```

¡ GRACIAS !



Universidad Industrial de Santander