Desafío - Naive Bayes

- Para realizar este desafío debes haber estudiado previamente todo el material disponibilizado correspondiente a la unidad.
- Una vez terminado el desafío, comprime la carpeta que contiene el desarrollo de los requerimientos solicitados y sube el `.zip` en el LMS.
- Desarrollo desafío:
    - El desafío se debe desarrollar de manera Individual.
    - Para la realización del desafío necesitarás apoyarte del archivo *Apoyo Desafío - Naive Bayes*.

Descripción

- En esta sesión trabajaremos con una serie de base de datos sobre letras musicales de distintos artistas. Cada uno de los `csv` se encuentra en la carpeta `dump` dentro del proyecto.
- Cada csv tiene el nombre del artista a analizar. Los archivos contienen el nombre del artista, el género musical del artista, el nombre de la canción y las letras.
- En base a esta información, se les pide un modelo generativo que pueda predecir el género de una canción a partir de la letra de una canción.
- Existen 4 géneros que se registran en la base de datos, por lo que para esta actividad trabajaremos con un Multinomial Naive Bayes.

Ejercicio 1: Preparar el ambiente de trabajo

- Importe los módulos `numpy`, `pandas`, `matplotlib`, `seaborn`, `glob` y os siguiendo las buenas prácticas. Los últimos dos módulos permitirán realizar la importación de múltiples archivos dentro de la carpeta dump.
- Para ello genere un objeto que guarde en una lista todos los archivos alojados en `dump` utilizando `glob.glob` y `os.getcwd()` para extraer las rutas absolutas. Posteriormente generé un objeto `pd.DataFrame` que contenga todos los `csv`.
- Asegúrese de eliminar la columna Unnamed: 0 que se genera por defecto.

Ejercicio 2: Descripción de los datos

- Utilizando el objeto creado en el Ejercicio 1, genere dos gráficos de barras que resuman la siguiente información:
    - La cantidad de canciones registradas por cada artista, ordenados de mayor a menor.
    - La cantidad de canciones registradas en cada género, ordenados de mayor a menor.
- Comente sobre las principales tendencias.

Ejercicio 3: Matriz de ocurrencias

Digresión: Tokenización de Textos

Para poder trabajar con textos, debemos pasarlos a una **matriz dispersa**, donde cada fila representará una entrada (en este caso, una canción), y cada columna **representará una palabra (token)**. Este es el proceso de tokenización: Identificar la ocurrencia de una palabra específica dentro de un conjunto de textos (corpus).

El tokenizador más simple `sklearn.feature_extraction.text.CountVectorizer` genera una colección de textos a una matriz que representa la frecuencia **dentro del texto** de una palabra específica.

El tokenizador funciona de la siguiente manera:

```
from sklearn.feature_extraction.text import CountVectorizer
# instanciamos un objeto
count_vectorizer=CountVectorizer(stop_words='english')
# Implementamos los pasos fit y transform
count_vectorizer_fit = count_vectorizer.fit_transform(lista_de_textos)
# Extraemos tokens (palabras)
words = count_vectorizer.get_feature_names()
# extraemos frecuencia
words_freq = count_vectorizer_fit.toarray().sum(axis=0)
```

- Importe la clase `CountVectorizer` dentro de los módulos `feature_extraction.text` de la librería `sklearn`. Lea la documentación asociada a ésta. ¿Cuál es el objetivo de esta clase?
- Aplique la clase para extraer las 100 palabras más repetidas en toda la base de datos.
- Genere una función que replique el procedimiento para cada uno de los géneros.
- Comente sobre las principales características de cada género en cuanto a sus palabras.

Ejercicio 4: Entrenamiento del Modelo

Digresión: `sklearn` Pipelines

La clase `Pipeline` del módulo `sklearn.pipeline` permite concatenar múltiples pasos de procesamiento y preprocesamiento en un estimador generado por algún método de `scikit-learn`. En sí, la clase cuenta con los métodos clásicos `fit`, `predict` y `score` y presenta un comportamiento idéntico a los demás objetos de `scikit-learn`. Uno de los usos más comunes es para concatenar pasos de preprocesamiento con un modelo.

Componentes de un `Pipeline`

Imaginemos que deseamos implementar el siguiente modelo. Considerando un conjunto de datos, deseo **Estandarizar**, posteriormente **extraer sus principales componentes** y finalmente aplicar un modelo de **regresión lineal**. Este flujo se puede reexpresar como:

```
pipeline_model = Pipeline([('scale', StandardScaler()),
            ('pca', RandomizedPCA(n_components=3)),
            ('model', LinearRegression())])
```

Algunos de los elementos a considerar:

1. Cada paso se considera como una tupla, donde se declara el nombre del paso y la función a implementar. En este caso, nuestro primer paso es estandarizar la matriz, por lo que asociamos el método `StandardScaler` con el string `scale`.
2. Todos los pasos declarados se incorporan en una lista, donde el orden de ingreso representa el orden de ejecución.

Posteriormente el objeto creado puede utilizarse con los siguientes métodos

```
pipeline_model.fit(X_train, y_train)
y_hat = pipeline_model.predict(y_test)
```

- Importe `MultinomialNB`, `train_test_split`, `Pipeline`, `confusion_matrix` y `classification_report`.
- Genere las muestras de entrenamiento y validación reservando un 40% para validación y declarando una semilla pseudoaleatoria.
- Monte el modelo dentro de un `Pipeline`, donde el primer paso es implementar `CountVectorizer` y el segundo es ejecutar el clasificador `MultinomialNB`.
- A continuación se les presenta una lista de letras, ¿cuáles serían las predicciones correspondientes?

```
['I got a place in the underworld', # Brody Dalle - Underworld
'As veils of ignorance, hatred retains Storm of arrows through karma
Seeking light through samsara', # Gorguts - Forgotten Arrows
"Bye bye Don't want to be a fool for you Just another player in your
game for two You may hate me but it ain't no lie", # N'SYNC - Bye Bye
Bye
'Move bitch, get out the way Get out the way bitch, get out the way Move
bitch, get out the way Get out the way bitch, get out the way', #
Ludacris - Move B*tch
'Sexual violence doesn't start and end with rape It starts in our books
and behind our school gates' # IDLES - Mother,
"Take it from the girl you claimed to love You gonna get some bad karma
I'm the one who had to learn to \
build a heart made of armor From the girl who made you soup and tied
your shoes when you were hurting\
You are not deserving, you are not deserving" #Banks - Drowning]
```

- Genere una predicción implementando la muestra de validación y contraste las predicciones del modelo con las etiquetas verdaderas. Reporte las principales métricas.

Ejercicio 5: Mejora del Modelo

- Proponga una estrategia para mejorar el desempeño del modelo en la categoría con peores métricas.
- Repita los pasos de entrenamiento y reporte de métricas, esta vez incluyendo los nuevos datos suministrados.
- Comente sobre el desempeño general de este.