```
# Importing Data (Same in All notebooks)
```

In [1]:

```
## Importing library
import numpy as np
import pandas as pd
np.random.seed(100)

data = pd.read_csv('/users/rohanchitte/downloads/Dataset_lyrics.csv_lyrics.csv'
```

```
# Data Preprocessing (Same in All notebooks)
```

In [2]:

```
filtered = data[data['lyrics'].notnull()]
filtered
```

Out[2]:

	index	song	year	artist	genre	lyrics
0	0	ego-remix	2009	beyonce-knowles	Pop	Oh baby, how you doing?\nYou know I'm gonna cu...
1	1	then-tell-me	2009	beyonce-knowles	Pop	playin' everything so easy,\nit's like you see...
2	2	honesty	2009	beyonce-knowles	Pop	If you search\nFor tenderness\nIt isn't hard t...
3	3	you-are-my-rock	2009	beyonce-knowles	Pop	Oh oh oh I, oh oh oh I\n[Verse 1:]\nIf I wrote...
4	4	black-culture	2009	beyonce-knowles	Pop	Party the people, the people the party it's po...
...	...	...	...	...	...	...
362232	362232	who-am-i-drinking-tonight	2012	edens-edge	Country	I gotta say\nBoy, after only just a couple of ...
362233	362233	liar	2012	edens-edge	Country	I helped you find her diamond ring\nYou made m...
362234	362234	last-supper	2012	edens-edge	Country	Look at the couple in the corner booth\nLooks ...
362235	362235	christ-alone-live-in-studio	2012	edens-edge	Country	When I fly off this mortal earth\nAnd I'm meas...
362236	362236	amen	2012	edens-edge	Country	I heard from a friend of a friend of a friend ...

266557 rows × 6 columns

In [3]:

```
import nltk
from nltk.corpus import stopwords

cleaned = filtered.copy()

# Remove punctuation
cleaned['lyrics'] = cleaned['lyrics'].str.replace("[-\?.,\/#!$%\^&\*;:{}=\_~()]'

# Remove song-related identifiers like [Chorus] or [Verse]
cleaned['lyrics'] = cleaned['lyrics'].str.replace("\[(.*?)\]", ' ')
cleaned['lyrics'] = cleaned['lyrics'].str.replace("' | '", ' ')
cleaned['lyrics'] = cleaned['lyrics'].str.replace('x[0-9]+', ' ')

# Remove all songs without lyrics (e.g. instrumental pieces)
cleaned = cleaned[cleaned['lyrics'].str.strip().str.lower() != 'instrumental']

# Remove any songs with corrupted/non-ASCII characters, unavailable lyrics
cleaned = cleaned[~cleaned['lyrics'].str.contains(r'[^\x00-\x7F]+')]
cleaned = cleaned[cleaned['lyrics'].str.strip() != '']
cleaned = cleaned[cleaned['genre'].str.lower() != 'not available']

#Selecting Pop, Rock, Country, Jazz
cleaned = cleaned.loc[(cleaned['genre'] == 'Pop') |
            (cleaned['genre'] == 'Country') |
            (cleaned['genre'] == 'Rock') |
            (cleaned['genre'] == 'Hip-Hop') |
            (cleaned['genre'] == 'Jazz') ]
cleaned.reset_index(inplace = True)

cleaned
print(len(cleaned))

from nltk.corpus import stopwords
stop = stopwords.words('english')
#removing stop words from lyrics

cleaned['lyrics'] = cleaned['lyrics'].apply(lambda x: ' '.join([word for word i

#lemmatizing lyrics
import nltk

w_tokenizer = nltk.tokenize.WhitespaceTokenizer()
lemmatizer = nltk.stem.WordNetLemmatizer()

def lemmatize_text(text, flg_lemm=True):
    #Convert string to list (tokenize)
    lst_text = text.split()

    ## Lemmatisation (convert the word into root word)
    if flg_lemm == True:
        lem = nltk.stem.wordnet.WordNetLemmatizer()
        lst_text = [lem.lemmatize(word) for word in lst_text]

    ## back to string from list
    text = " ".join(lst_text)
    return text

#cleaned["lyrics"] = cleaned["lyrics"].apply(lemmatize_text)

```

```
cleaned["lyrics"]  = cleaned["lyrics"].apply(lambda x:  lemmatize_text(x))

df = cleaned.drop(labels=["level_0", "index","song","year","artist"], axis=1)
```

```
185493
```

# Visualize Data (Same in All notebooks)

In [4]:

```
df
```

Out[4]:

	genre	lyrics
0	Pop	Oh baby You know I'm gonna cut right chase Som...
1	Pop	playin everything easy like seem sure still wa...
2	Pop	If search For tenderness It hard find You love...
3	Pop	Oh oh oh I oh oh oh I If I wrote book stand Th...
4	Pop	Party people people party popping sitting arou...
...	...	...
185488	Country	I gotta say Boy couple date You're hand outrig...
185489	Country	I helped find diamond ring You made try everyt...
185490	Country	Look couple corner booth Looks lot like She's ...
185491	Country	When I fly mortal earth And I'm measured depth...
185492	Country	I heard friend friend friend You finally got r...

185493 rows × 2 columns

In [5]:

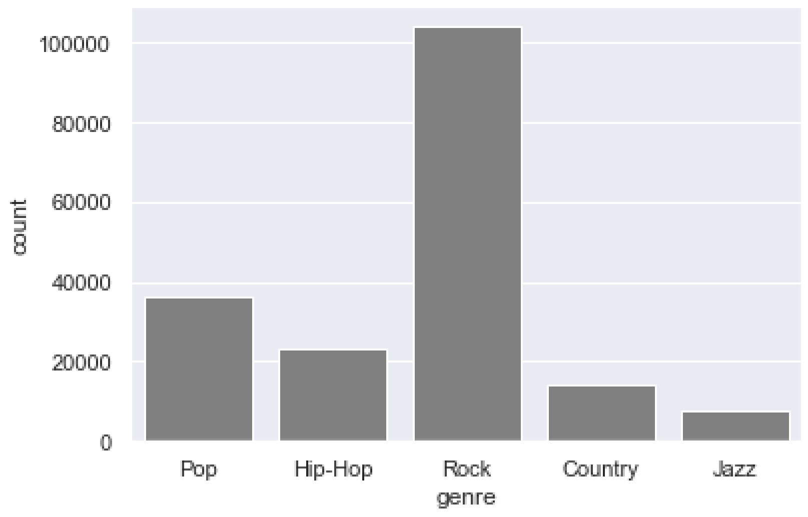
```
import seaborn as sns
sns.set()

sns.countplot(df['genre'], color='gray')
```

```
/opt/anaconda3/lib/python3.8/site-packages/seaborn/_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From v
ersion 0.12, the only valid positional argument will be `data`, and pa
ssing other arguments without an explicit keyword will result in an er
ror or misinterpretation.
  warnings.warn(
```

Out[5]:

```
<AxesSubplot:xlabel='genre', ylabel='count'>
```



```
# One hot Encoding of Genres
```

In [6]:

```
from sklearn.preprocessing import LabelEncoder
Y = df["genre"]
Y = LabelEncoder().fit_transform(Y)
```

In [7]:

```
df['Y'] = Y.tolist()
```

In [ ]:

```
df["Y"]
```

In [ ]:

```
y = df["Y"]
```

```
# Splitting Data into Train and Test Set
```

In [8]:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(df["lyrics"],df["Y"], test_s
```

In [9]:

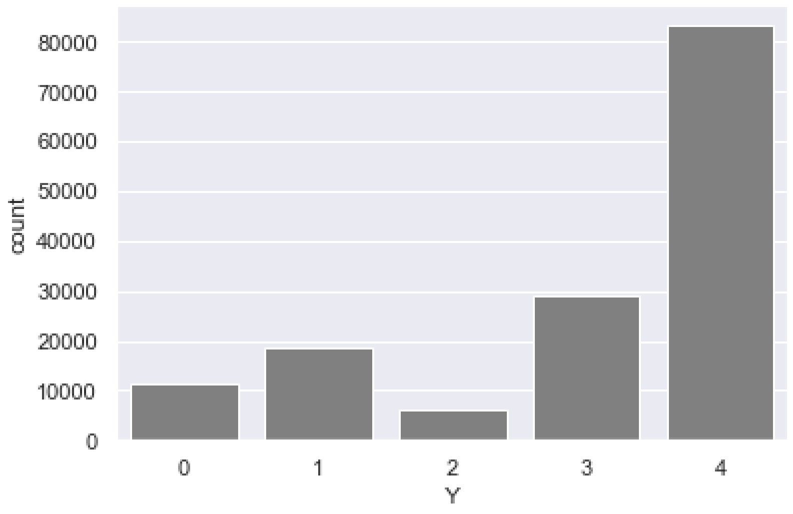
```
#Visualizing Y - Genres of training set
import seaborn as sns
sns.set()

sns.countplot(y_train, color='gray')
```

```
/opt/anaconda3/lib/python3.8/site-packages/seaborn/_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From v
ersion 0.12, the only valid positional argument will be `data`, and pa
ssing other arguments without an explicit keyword will result in an er
ror or misinterpretation.
  warnings.warn(
```

Out[9]:

```
<AxesSubplot:xlabel='Y', ylabel='count'>
```



```
# Bert Model
```

In [10]:

```
import tensorflow as tf
import tensorflow_hub as hub
import tensorflow_text as text
from official.nlp import optimization  # to create AdamW optimizer
from official.nlp import bert
from tensorflow import keras
```

```
/opt/anaconda3/lib/python3.8/site-packages/tensorflow_addons/utils/ens
ure_tf_install.py:53: UserWarning: Tensorflow Addons supports using Py
thon ops for all Tensorflow versions above or equal to 2.3.0 and stric
tly below 2.6.0 (nightly versions are not supported).
 The versions of TensorFlow you are currently using is 2.6.0 and is no
t supported.
Some things might work, some things might not.
If you were to encounter a bug, do not file an issue.
If you want to make sure you're using a tested and supported configura
tion, either change the TensorFlow version or the TensorFlow Addons's
version.
You can find the compatibility matrix in TensorFlow Addon's readme:
https://github.com/tensorflow/addons (https://github.com/tensorflow/ad
dons)
  warnings.warn(
```

In [14]:

```
import spacy
nlp = spacy.load("en_core_web_sm")

def remove_non_ascii(text):
    doc = nlp(text)
    to_return =  " ".join([str(token) for token in doc if token.is_ascii])
    return to_return

X_train = X_train.apply(remove_non_ascii)
```

In [17]:

```
X_test = X_test.apply(remove_non_ascii)
```

In [15]:

```
# selecting BERT encoder having transformer layers(L) = 4, ####dimension of o/p = 51
bert_encoder = 'https://tfhub.dev/tensorflow/small_bert/bert_en_uncased_L-4_H-512_A-

#choosing pre-processor that is compatible with BERT encoder ####selected
bert_pre_process = 'https://tfhub.dev/tensorflow/bert_en_uncased_preprocess/3'
 build_classifier_model():
 text_input = tf.keras.layers.Input(shape=(), dtype=tf.string, name='text')
 preprocessing_layer = hub.KerasLayer(tf_bert_pre_process, name='preprocessing')
 encoder_inputs = preprocessing_layer(text_input)
 encoder = hub.KerasLayer(tf_bert_encoder, trainable=True, name='BERT_encoder')
 outputs = encoder(encoder_inputs)
 net = outputs['pooled_output']
 net = tf.keras.layers.Dropout(0.1)(net)
 net = tf.keras.layers.Dense(5, activation='softmax', name='classifier')(net)
 #net = tf.keras.layers.Dense(1, activation=None, name='classifier')(net)
 return tf.keras.Model(text_input, net)

ssifier_model = build_classifier_model()
```

In [16]:

```
oss = tf.keras.losses.CategoricalCrossentropy(from_logits=True)
etrics = tf.metrics.CategoricalAccuracy()
pochs = 5
teps_per_epoch = 11370
um_train_steps = steps_per_epoch * epochs
um_warmup_steps = int(0.1*num_train_steps)
nit_lr = 3e-5
ptimizer = optimization.create_optimizer(init_lr=init_lr, num_train_steps=num_train_
lassifier_model.compile(optimizer=optimizer,
                        loss=loss,
                        metrics=metrics)
```

In [18]:

```
from sklearn.preprocessing import LabelBinarizer

def get_encoded_labels(topic_clusters):
    encoder = LabelBinarizer()
    encoded_labels = encoder.fit_transform(topic_clusters)
    return encoded_labels

labels = get_encoded_labels(y_train)

labelsval = get_encoded_labels(y_test)
```

```
# Fitting the model
```

In [19]:

```
history = classifier_model.fit(X_train,labels,epochs=epochs,verbose=1,validatio
```

```
Epoch 1/5

/opt/anaconda3/lib/python3.8/site-packages/keras/backend.py:4846: User
Warning: "`categorical_crossentropy` received `from_logits=True`, but
the `output` argument was produced by a sigmoid or softmax activation
and thus does not represent logits. Was this intended?"
  warnings.warn(

4638/4638 [==============================] - 114286s 25s/step - loss:
0.9700 - categorical_accuracy: 0.6271 - val_loss: 0.8630 - val_categor
ical_accuracy: 0.6629
Epoch 2/5
4638/4638 [==============================] - 194980s 42s/step - loss:
0.8062 - categorical_accuracy: 0.6940 - val_loss: 0.7849 - val_categor
ical_accuracy: 0.7047
Epoch 3/5
4638/4638 [==============================] - 154321s 33s/step - loss:
0.7122 - categorical_accuracy: 0.7323 - val_loss: 0.7715 - val_categor
ical_accuracy: 0.7142
Epoch 4/5
4638/4638 [==============================] - 112350s 24s/step - loss:
0.6270 - categorical_accuracy: 0.7662 - val_loss: 0.7914 - val_categor
ical_accuracy: 0.7095
Epoch 5/5
4638/4638 [==============================] - 114761s 25s/step - loss:
0.5467 - categorical_accuracy: 0.7987 - val_loss: 0.8354 - val_categor
ical_accuracy: 0.7138
```

In [ ]:

```
epochs = 2
classifier_model.fit(X_train,labels,epochs=epochs,verbose=1,validation_data =(X_
```

```
Epoch 1/2
4638/4638 [==============================] - 113663s 25s/step - loss:
0.4733 - categorical_accuracy: 0.8270 - val_loss: 0.8779 - val_categor
ical_accuracy: 0.7026
Epoch 2/2
2907/4638 [=================>............] - ETA: 10:58:01 - loss: 0.4
018 - categorical_accuracy: 0.8549
```

Training interrupted as the model was overfitting and the validation accuracy was not improving.