**Spotify Examples Part-2**

**Example 11:**  This example uses the Python profanity check library to examine profanity or offensive language in song lyrics. The example saves both profanity prediction scores and their probabilities as new columns to the example8results.csv file.

Original Tutorial:

<https://pypi.org/project/profanity-check/>

pip3 install profanity-check (This installation gave me error when importing the library)

pip3 install alt-profanity-check

import pandas

#I am creating a dictionary here titled inputdata

inputdata={}

#I am assigning the content of the csv file to my dictionary

#header is my row in the csv file that is why header is 0 below

inputdata = pandas.read\_csv('example8results.csv', header=[0], index\_col=0).to\_dict()

#We can use type to check the data type of a variable

#print(type(inputdata))

#I am using the column headers from the csv file to find the data I am interested to analyze

# I created a new dictionary here for the description column in my csv file

lyricsdictionary = inputdata.get('Lyrics')

#print(type(descriptiondictionary))

# I am converting the dictionary to a list so I can analyze the data

lyricslist = list(lyricsdictionary.values())

#print(lyricslist)

from profanity\_check import predict, predict\_prob

#0 means there is no offensive language. 1 is the opposite

profanity\_prediction\_list = predict(lyricslist)

print(profanity\_prediction\_list)

profanity\_probability\_list= predict\_prob(lyricslist)

print(profanity\_probability\_list)

file = pandas.read\_csv('example8results.csv')

file['Profanity Prediction Score'] = profanity\_prediction\_list

file['Profanity Prediction Probability'] = profanity\_probability\_list

#Index is false because example 1.csv file already has an index column

file.to\_csv('example11results.csv', index=False)

print("done")

**Example 12:** This example focuses on calculating the strongest emotion from the lyrics of a song. The example uses the notion of supervised learning from the machine learning class to build a random forest-based model. We need the text.txt file to build the model. Once we build the model, we make predictions based on our scraped song lyrics. The example saves the results into a csv file.

Original Tutorial: <https://thecleverprogrammer.com/2021/02/19/text-emotions-detection-with-machine-learning/>

import re

from collections import Counter

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import accuracy\_score

from sklearn.svm import SVC

from sklearn.svm import LinearSVC

from sklearn.ensemble import RandomForestClassifier

from sklearn.tree import DecisionTreeClassifier

def read\_data(file):

data = []

with open(file, 'r')as f:

for line in f:

line = line.strip()

label = ' '.join(line[1:line.find("]")].strip().split())

text = line[line.find("]")+1:].strip()

data.append([label, text])

return data

#This text file is an input to build an emotion detection classifier.

file = 'text.txt'

data = read\_data(file)

print("Number of instances: {}".format(len(data)))

def ngram(token, n):

output = []

for i in range(n-1, len(token)):

ngram = ' '.join(token[i-n+1:i+1])

output.append(ngram)

return output

def create\_feature(text, nrange=(1, 1)):

text\_features = []

text = text.lower()

text\_alphanum = re.sub('[^a-z0-9#]', ' ', text)

for n in range(nrange[0], nrange[1]+1):

text\_features += ngram(text\_alphanum.split(), n)

text\_punc = re.sub('[a-z0-9]', ' ', text)

text\_features += ngram(text\_punc.split(), 1)

return Counter(text\_features)

def convert\_label(item, name):

items = list(map(float, item.split()))

label = ""

for idx in range(len(items)):

if items[idx] == 1:

label += name[idx] + " "

return label.strip()

emotions = ["joy", 'fear', "anger", "sadness", "disgust", "shame", "guilt"]

X\_all = []

y\_all = []

for label, text in data:

y\_all.append(convert\_label(label, emotions))

X\_all.append(create\_feature(text, nrange=(1, 4)))

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_all, y\_all, test\_size = 0.2, random\_state = 123)

def train\_test(clf, X\_train, X\_test, y\_train, y\_test):

clf.fit(X\_train, y\_train)

train\_acc = accuracy\_score(y\_train, clf.predict(X\_train))

test\_acc = accuracy\_score(y\_test, clf.predict(X\_test))

return train\_acc, test\_acc

from sklearn.feature\_extraction import DictVectorizer

vectorizer = DictVectorizer(sparse = True)

X\_train = vectorizer.fit\_transform(X\_train)

X\_test = vectorizer.transform(X\_test)

svc = SVC()

lsvc = LinearSVC(random\_state=123)

rforest = RandomForestClassifier(random\_state=123)

dtree = DecisionTreeClassifier()

#I am using only the random forest classifier from the tutorial

clifs = [rforest]

# train and test them

print("| {:25} | {} | {} |".format("Classifier", "Training Accuracy", "Test Accuracy"))

print("| {} | {} | {} |".format("-"\*25, "-"\*17, "-"\*13))

for clf in clifs:

clf\_name = clf.\_\_class\_\_.\_\_name\_\_

train\_acc, test\_acc = train\_test(clf, X\_train, X\_test, y\_train, y\_test)

print("| {:25} | {:17.7f} | {:13.7f} |".format(clf\_name, train\_acc, test\_acc))

emoji\_dict = {"joy":"joy", "fear":"fear", "anger":"anger", "sadness":"sadness", "disgust":"disgust", "shame":"shame", "guilt":"quilt"}

import pandas

#Let's use our lyrics data to classify songs with respect to the emotions above

inputdata={}

#I am assigning the content of the csv file to my dictionary

#header is my row in the csv file that is why header is 0 below

inputdata = pandas.read\_csv('example11results.csv', header=[0], index\_col=0).to\_dict()

#We can use type to check the data type of a variable

#print(type(inputdata))

#I am using the column headers from the csv file to find the data I am interested to analyze

# I created a new dictionary here for the description column in my csv file

lyricsdictionary = inputdata.get('Lyrics')

#print(type(descriptiondictionary))

# I am converting the dictionary to a list so I can analyze the data

lyricslist = list(lyricsdictionary.values())

#print(lyricslist)

emotionresults\_list=[]

for text in lyricslist :

features = create\_feature(text, nrange=(1, 4))

features = vectorizer.transform(features)

prediction = clf.predict(features)[0]

print("Emotion:", emoji\_dict[prediction])

emotionresults\_list.append(emoji\_dict[prediction])

file = pandas.read\_csv('example11results.csv')

file['Emotion'] = emotionresults\_list

file.to\_csv('example12results.csv', index=False)

print("done")

**Example 13:** This example finds the explicit words from Rihanna’s scraped songs. The script uses the Google’s explicit word list to accomplish its objective. You need the explicit words txt file for the script to work. The script saves the frequency count of identified explicit words into a csv file.

Original Tutorial: <https://github.com/coffee-and-fun/google-profanity-words>

import pandas

import re

#I am creating a dictionary here titled inputdata

inputdata={}

#I am assigning the content of the csv file to my dictionary

#header is my row in the csv file that is why header is 0 below

inputdata = pandas.read\_csv('example8results.csv', header=[0], index\_col=0).to\_dict()

#We can use type to check the data type of a variable

#print(type(inputdata))

#I am using the column headers from the csv file to find the data I am interested to analyze

# I created a new dictionary here for the description column in my csv file

lyricsdictionary = inputdata.get('Lyrics')

#print(type(descriptiondictionary))

# I am converting the dictionary to a list so I can analyze the data

lyricslist = list(lyricsdictionary.values())

lyrics\_string= str(lyricslist)

#I am cleaning the lyrics text

cleaned\_text = lyrics\_string.replace("\\n","")

cleaned\_text = lyrics\_string.replace("<.\*?>|&([a-z0-9]+|#[0-9]{1,6}|#x[0-9a-f]{1,6});","")

cleaned\_text = lyrics\_string.replace("\\s","")

cleaned\_text = lyrics\_string.replace('\\',"")

#Open explicit words file

explicitwords\_file = open("explicitwords.txt", "r")

explicitwords\_data\_string = explicitwords\_file.read()

pattern = r'[0-9]'

explicitwords\_data\_string = re.sub(pattern, '', explicitwords\_data\_string)

explicitwords\_list= explicitwords\_data\_string.split()

output={"ExplicitWords":[]}

songs\_explicit\_words\_list=[]

for explicitword in explicitwords\_list:

counter=0

if len(explicitword)>2:

if (explicitword not in ['dream','play','make','hit','wild','off','men','her','your','and','love','huge','tea','one','two','are','baby','straight','gone', 'baby' 'men','brown','white','eat','black','girl','hot','jack','come','jacket']):

if explicitword in cleaned\_text:

#print(explicitword)

songs\_explicit\_words\_list.append(explicitword)

output = {"ExplicitWord":[],"Frequency":[]}

counterdictionary={}

for explicitword in songs\_explicit\_words\_list:

output["ExplicitWord"].append(explicitword)

counterdictionary[explicitword] = songs\_explicit\_words\_list.count(explicitword)

output['Frequency'].append(counterdictionary[explicitword])

# Remove duplicate values in dictionary

# Using setdefault() method

results = pandas.DataFrame(output)

results= results.drop\_duplicates(keep='first')

#print(results)

#Explicit Words in Rihanna's Scraped Songs

results.to\_csv('example13results.csv', index=True, index\_label="Index")

print("done")

**Example 14:** This example demonstrates how to draw scatter plot diagrams to visually examine a potential relationship between two variables.

Required Library: pip3 install matplotlib

<https://pypi.org/project/matplotlib/>

import pandas

#I am creating a dictionary here titled inputdata

inputdata={}

#I am assigning the content of the csv file to my dictionary

#header is my row in the csv file that is why header is 0 below

inputdata = pandas.read\_csv('example4results.csv', header=[0], index\_col=0).to\_dict()

#We can use type to check the data type of a variable

#print(type(inputdata))

#I am using the column headers from the csv file to find the data I am interested to analyze

# I created a new dictionary here for the energy column in my csv file

energydictionary = inputdata.get('energy')

# I am converting the dictionary to a list so I can analyze the data

energylist = list(energydictionary.values())

popularitydictionary = inputdata.get('popularity')

popularitylist = list(popularitydictionary.values())

import matplotlib.pyplot as plt

fig, ax = plt.subplots()

ax.scatter(energylist,popularitylist)

plt.xlabel("Energy")

plt.ylabel("Popularity")

plt.show()

**Example 15:**  This example demonstrates how to draw a line chart to visually examine if data follows a trend.

import pandas

#I am creating a dictionary here titled inputdata

inputdata={}

#I am assigning the content of the csv file to my dictionary

#header is my row in the csv file that is why header is 0 below

inputdata = pandas.read\_csv('example4results.csv', header=[0], index\_col=0).to\_dict()

#We can use type to check the data type of a variable

#print(type(inputdata))

#I am using the column headers from the csv file to find the data I am interested to analyze

# I created a new dictionary here for the description column in my csv file

songdictionary = inputdata.get('name')

# I am converting the dictionary to a list so I can analyze the data

songlist = list(songdictionary.values())

danceabilitydictionary = inputdata.get('danceability')

danceabilitylist = list(danceabilitydictionary.values())

import matplotlib.pyplot as plt

x\_axis = songlist

y\_axis = danceabilitylist

plt.plot(x\_axis, y\_axis)

plt.title('Danceability Trend')

plt.xlabel('Danceability')

plt.ylabel('Songs')

plt.show()