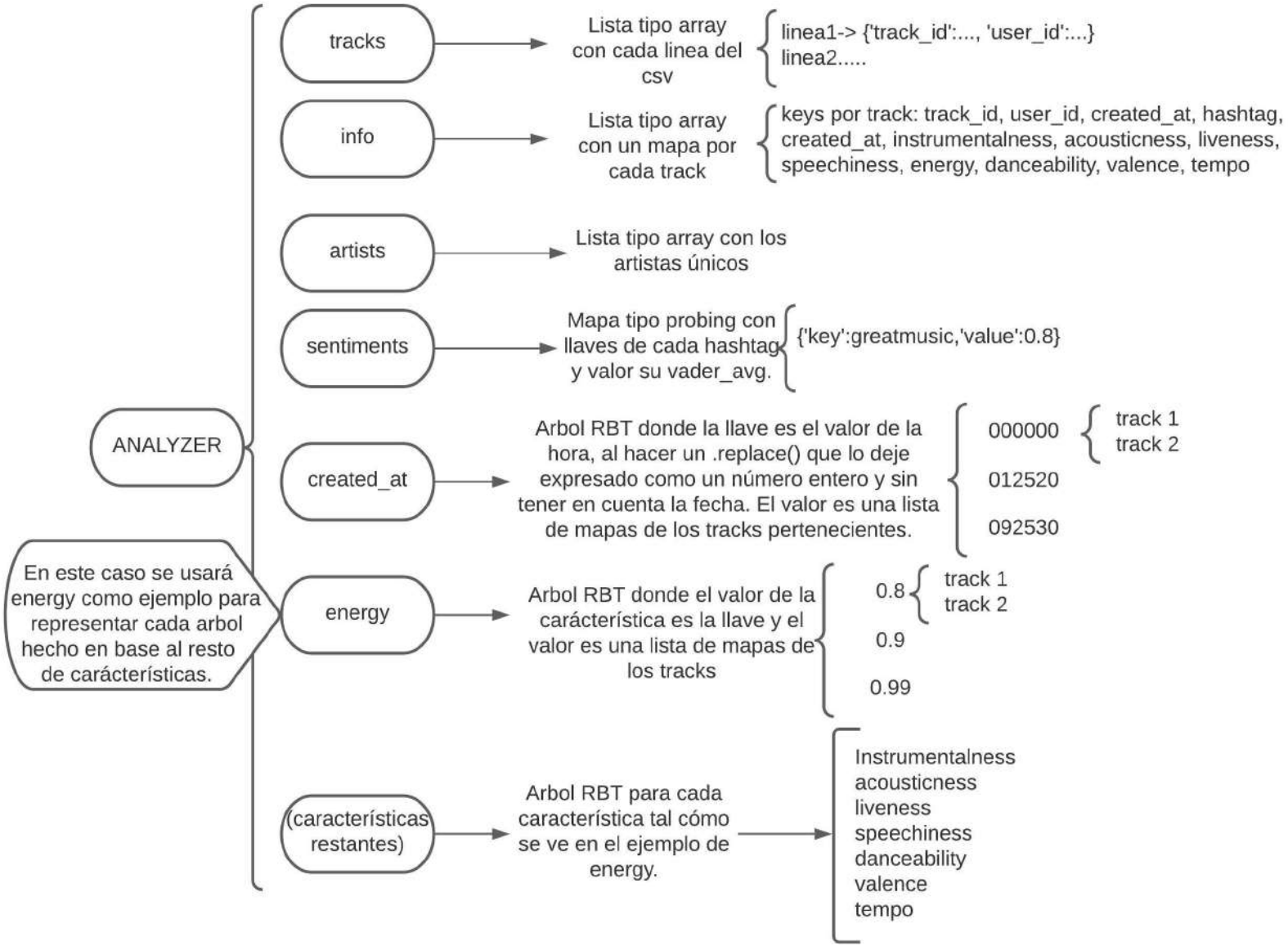
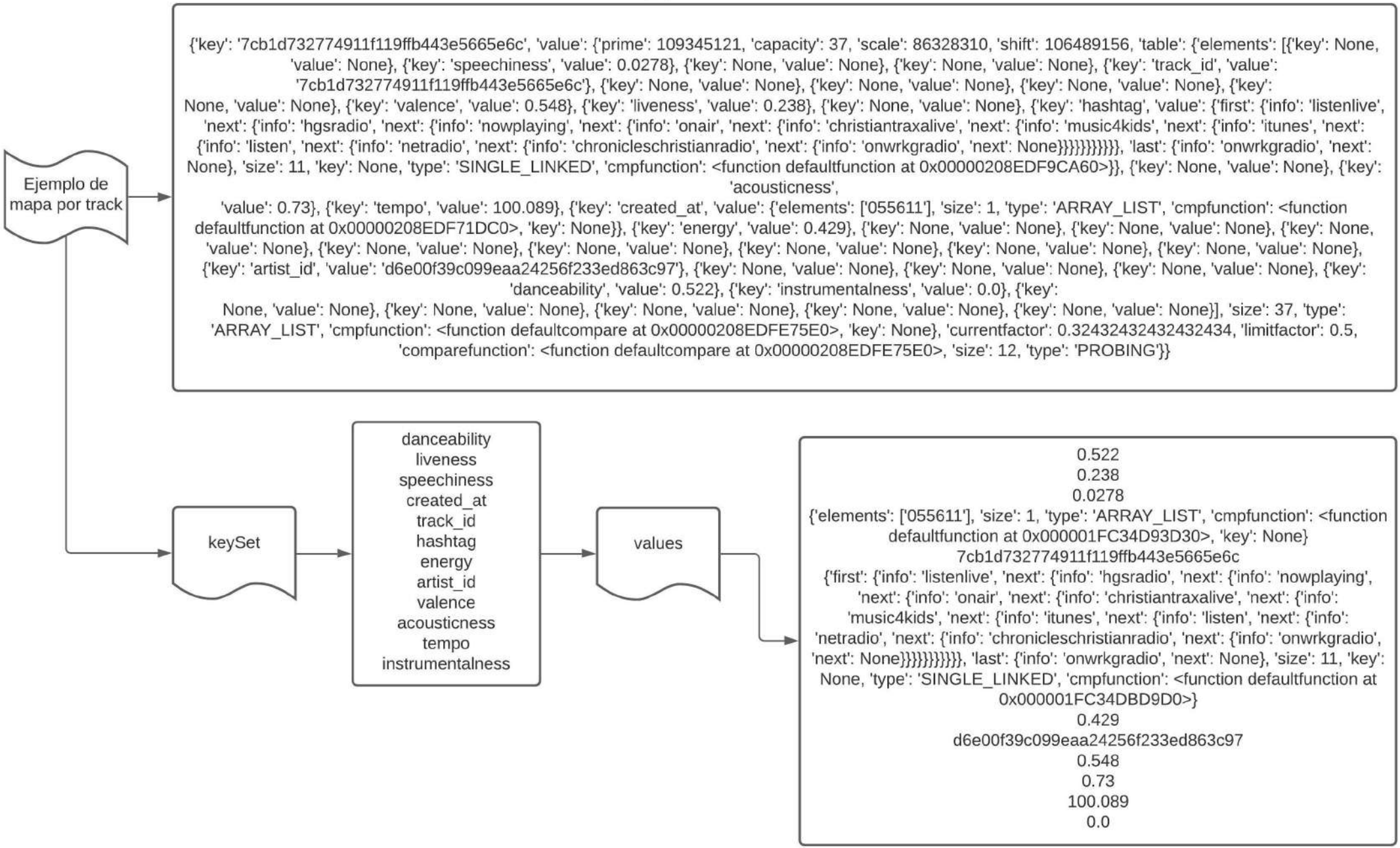
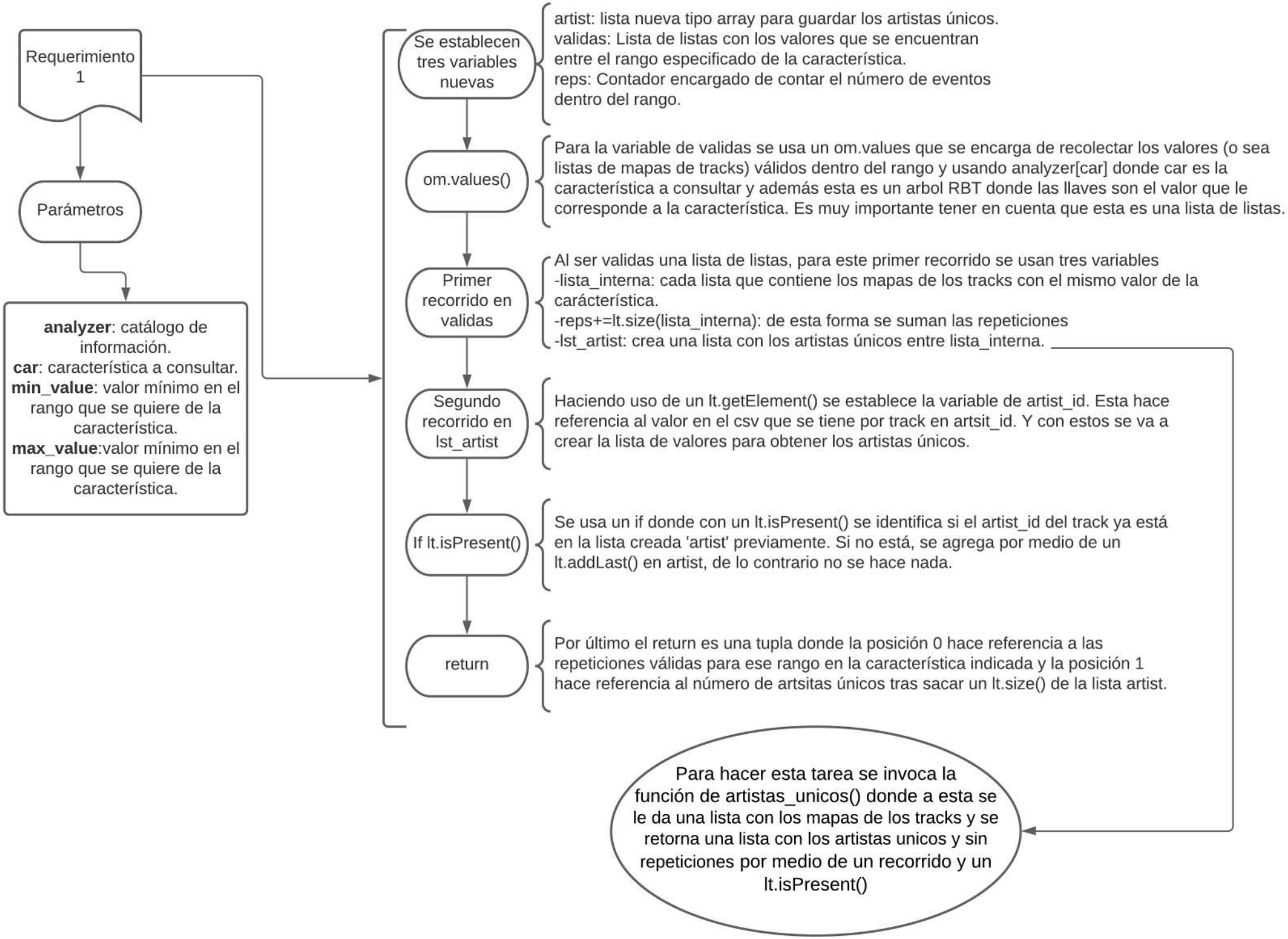

ANALYZER
En este caso se usará energy como ejemplo para representar cada arbol hecho en base al resto de características.
tracks
Lista tipo array con cada linea del csv
linea1-> {'track_id':..., 'user_id':...}
linea2.....
info
Lista tipo array con un mapa por cada track
keys por track: track_id, user_id, created_at, hashtag, created_at, instrumentalness, acousticness, liveness, speechiness, energy, danceability, valence, tempo
artists
Lista tipo array con los artistas únicos
sentiments
Mapa tipo probing con llaves de cada hashtag y valor su vader_avg.
{'key':greatmusic,'value':0.8}
created_at
Arbol RBT donde la llave es el valor de la hora, al hacer un .replace() que lo deje expresado como un número entero y sin tener en cuenta la fecha. El valor es una lista de mapas de los tracks pertenecientes.
000000
track 1
track 2
012520
092530
energy
Arbol RBT donde el valor de la característica es la llave y el valor es una lista de mapas de los tracks
0.8
track 1
track 2
0.9
0.99
(características restantes)
Arbol RBT para cada característica tal cómo se ve en el ejemplo de energy.
Instrumentalness
acousticness
liveness
speechiness
danceability
valence
tempo


Ejemplo de mapa por track
{'key': '7cb1d732774911f119ffb443e5665e6c', 'value': {'prime': 109345121, 'capacity': 37, 'scale': 86328310, 'shift': 106489156, 'table': {'elements': [{'key': None, 'value': None}, {'key': 'speechiness', 'value': 0.0278}, {'key': None, 'value': None}, {'key': None, 'value': None}, {'key': 'track_id', 'value': '7cb1d732774911f119ffb443e5665e6c'}, {'key': None, 'value': None}, {'key': None, 'value': None}, {'key': None, 'value': None}, {'key': None, 'value': None}, {'key': 'valence', 'value': 0.548}, {'key': 'liveness', 'value': 0.238}, {'key': None, 'value': None}, {'key': 'hashtag', 'value': {'first': {'info': 'listenlive', 'next': {'info': 'hgsradio', 'next': {'info': 'nowplaying', 'next': {'info': 'onair', 'next': {'info': 'christiantraxalive', 'next': {'info': 'music4kids', 'next': {'info': 'itunes', 'next': {'info': 'listen', 'next': {'info': 'netradio', 'next': {'info': 'chronicleschristianradio', 'next': {'info': 'onwrkgradio', 'next': None}}}}}}}}}}}, 'last': {'info': 'onwrkgradio', 'next': None}, 'size': 11, 'key': None, 'type': 'SINGLE_LINKED', 'cmpfunction': <function defaultfunction at 0x00000208EDF9CA60>}}, {'key': None, 'value': None}, {'key': 'acousticness', 'value': 0.73}, {'key': 'tempo', 'value': 100.089}, {'key': 'created_at', 'value': {'elements': ['055611'], 'size': 1, 'type': 'ARRAY_LIST', 'cmpfunction': <function defaultfunction at 0x00000208EDF71DC0>, 'key': None}}, {'key': 'energy', 'value': 0.429}, {'key': None, 'value': None}, {'key': None, 'value': None}, {'key': None, 'value': None}, {'key': None, 'value': None}, {'key': None, 'value': None}, {'key': None, 'value': None}, {'key': None, 'value': None}, {'key': None, 'value': None}, {'key': 'artist_id', 'value': 'd6e00f39c099eaa24256f233ed863c97'}, {'key': None, 'value': None}, {'key': None, 'value': None}, {'key': None, 'value': None}, {'key': 'danceability', 'value': 0.522}, {'key': 'instrumentalness', 'value': 0.0}, {'key': None, 'value': None}, {'key': None, 'value': None}, {'key': None, 'value': None}, {'key': None, 'value': None}, {'key': None, 'value': None}], 'size': 37, 'type': 'ARRAY_LIST', 'cmpfunction': <function defaultcompare at 0x00000208EDFE75E0>, 'key': None}, 'currentfactor': 0.32432432432432434, 'limitfactor': 0.5, 'comparefunction': <function defaultcompare at 0x00000208EDFE75E0>, 'size': 12, 'type': 'PROBING'}}
keySet
danceability
liveness
speechiness
created_at
track_id
hashtag
energy
artist_id
valence
acousticness
tempo
instrumentalness
values
0.522
0.238
0.0278
{'elements': ['055611'], 'size': 1, 'type': 'ARRAY_LIST', 'cmpfunction': <function defaultfunction at 0x000001FC34D93D30>, 'key': None}
7cb1d732774911f119ffb443e5665e6c
{'first': {'info': 'listenlive', 'next': {'info': 'hgsradio', 'next': {'info': 'nowplaying', 'next': {'info': 'onair', 'next': {'info': 'christiantraxalive', 'next': {'info': 'music4kids', 'next': {'info': 'itunes', 'next': {'info': 'listen', 'next': {'info': 'netradio', 'next': {'info': 'chronicleschristianradio', 'next': {'info': 'onwrkgradio', 'next': None}}}}}}}}}}}, 'last': {'info': 'onwrkgradio', 'next': None}, 'size': 11, 'key': None, 'type': 'SINGLE_LINKED', 'cmpfunction': <function defaultfunction at 0x000001FC34DBD9D0>}
0.429
d6e00f39c099eaa24256f233ed863c97
0.548
0.73
100.089
0.0


Requerimiento 1
Parámetros
analyzer: catálogo de información.
car: característica a consultar.
min_value: valor mínimo en el rango que se quiere de la característica.
max_value:valor mínimo en el rango que se quiere de la característica.
Se establecen tres variables nuevas
artist: lista nueva tipo array para guardar los artistas únicos.
validas: Lista de listas con los valores que se encuentran entre el rango especificado de la característica.
reps: Contador encargado de contar el número de eventos dentro del rango.
om.values()
Para la variable de validas se usa un om.values que se encarga de recolectar los valores (o sea listas de mapas de tracks) válidos dentro del rango y usando analyzer[car] donde car es la característica a consultar y además esta es un arbol RBT donde las llaves son el valor que le corresponde a la característica. Es muy importante tener en cuenta que esta es una lista de listas.
Primer recorrido en validas
Al ser validas una lista de listas, para este primer recorrido se usan tres variables
-lista_interna: cada lista que contiene los mapas de los tracks con el mismo valor de la carácterística.
-reps+=lt.size(lista_interna): de esta forma se suman las repeticiones
-lst_artist: crea una lista con los artistas únicos entre lista_interna.
Segundo recorrido en lst_artist
Haciendo uso de un lt.getElement() se establece la variable de artist_id. Esta hace referencia al valor en el csv que se tiene por track en artsit_id. Y con estos se va a crear la lista de valores para obtener los artistas únicos.
If lt.isPresent()
Se usa un if donde con un lt.isPresent() se identifica si el artist_id del track ya está en la lista creada 'artist' previamente. Si no está, se agrega por medio de un lt.addLast() en artist, de lo contrario no se hace nada.
return
Por último el return es una tupla donde la posición 0 hace referencia a las repeticiones válidas para ese rango en la característica indicada y la posición 1 hace referencia al número de artsitas únicos tras sacar un lt.size() de la lista artist.
Para hacer esta tarea se invoca la función de artistas_unicos() donde a esta se le da una lista con los mapas de los tracks y se retorna una lista con los artistas unicos y sin repeticiones por medio de un recorrido y un lt.isPresent()