

Bilgisayar Mühendisliği Programında Hazırlanan  
Bitirme Projesi

# MÜZİK TÜRLERİ SINIFLANDIRILMASI VE ÖNERİ SİSTEMİ

**TEZ DANIŞMANI : DR. ARZU KAKIŞIM**

**HAZIRLAYAN : ŞEVVAL YOĞURTCUOĞLU**

# İÇİNDEKİLER

## ► Giriş

1. Müzik nedir, türleri nelerdir ?
2. Neden müzik sınıflandırılması yapıp öneri sunulmalıdır ?
3. Problemin gerçek hayatta görüldüğü alanlar
4. Problemin çözümü adımları

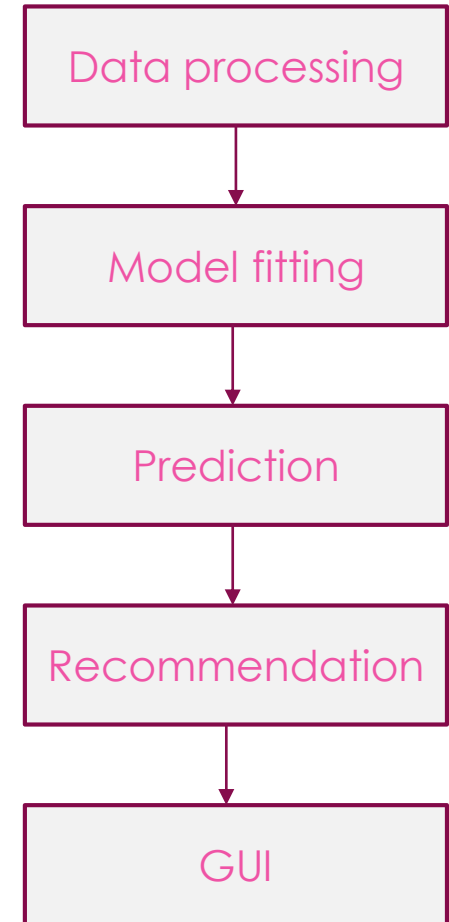
## ► Deneysel kurulum

1. Veri seti tanıtımı, veri ön işleme
2. Sınıflandırma için kullanılan algoritmalar
3. Öneri sistemi
4. Kodun çalışma mantığının açıklanması

## ► Sonuç

1. Karşılaşılan problemler
2. Geliştirilebilir yanlar

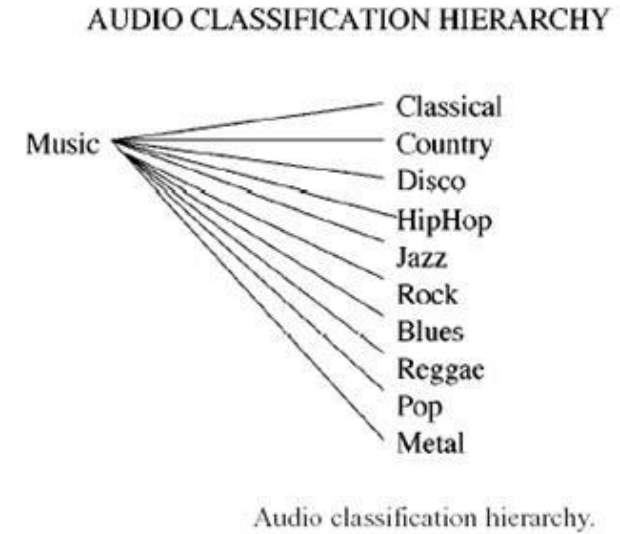
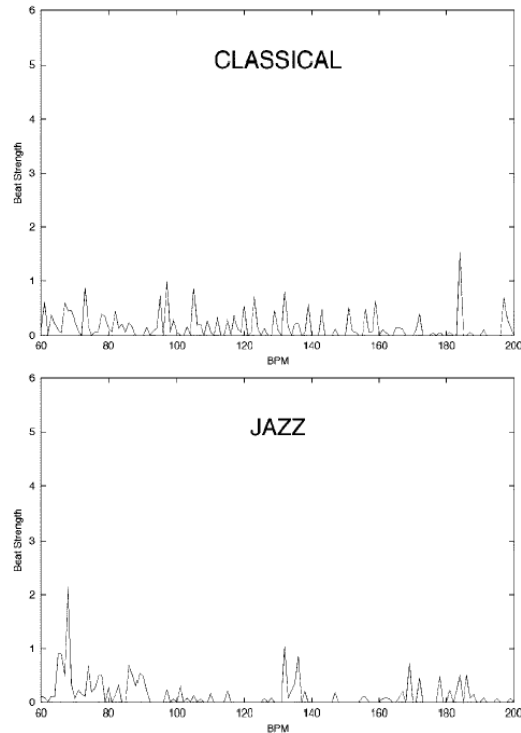
## ► Kullanıcı ara yüzü



# GİRİŞ (INTRODUCTION)

# Müzik nedir? , Müzik türleri nelerdir?

- Müzik, sesin biçim ve anlamlı titreşimler kazanmış halidir.



# Neden Müzik Sınıflandırılması Yapılmalıdır ?

## Neden Öneri Sunulmalıdır ?

- Müzik, çoğu insanın günlük hayatında sıkça kullandığı bir sanattır. Bu nedenle müzik hizmetleri insanların sevdikleri müzikleri dinleyebilmeleri için birincil araç haline gelmiştir. Müzik hizmetleri kullanıcıların hareketlerine bakarak onlara kişiselleştirmiş bir deneyim sunarlar. Kullanıcının dinlediği müziklere benzer parçalar önerildiğinde kullanıcı sevdiği tarzda müziklere rahatça erişebilmiş olur.



# Problemin Gerçek Hayatta Görüldüğü Alanlar

► Günümüz en popüler müzik hizmetleri ;

○ Spotify



○ Shazam



○ Soundcloud



○ Fizy



Müşterilerine müzik önerisinde bulunmak için müzik sınıflandırması kullanmaktadır. Müzik türlerinin belirlenmesi bu yöndeki ilk adımdır.

# Problemin Çözümü Adımları

- Bu projede makine öğrenmesi ile müzik türlerinin sınıflandırılması ve seçilen müziğe benzer önerilerin sunulması hedeflenmiştir.

Müzik verileri ile ;

- 1) Pre-processing ( Ön işleme) , Feature extraction ( öznitelik çıkarımı), feature selection ( öznitelik seçimi) uygulanarak veri seti oluşturuldu
- 2) Makine öğrenmesi algoritmaları kullanılarak müzikler sınıflandırıldı.
- 3) Performans değerlendirmesi yaparak en iyi algoritma seçildi.
- 4) Seçilen algoritma ile öneri sistemi geliştirildi.

# DENEYSEL KURULUM ( EXPERIMENTAL SETUP)



# Kullanılan Veri

← → ↻ freemusicarchive.org/genre/Blues

Music Community Tribe of Noise Acquired Free Music Archive. Follow [@freemusicarchiv](#) for updates.

## FMA Free Music Archive

Curators Genres Charts About For Musicians For Filmmakers

**Blues**  
Genres > Blues

Gospel

Artist

Blues

Classical

Country

Electronic

Experimental

Folk

Hip-Hop

Instrumental

International

Jazz

Novelty

Old-Time / Historic

Pop

Rock

Soul-RnB

Spoken

Date Added

▶	Checkie Brown	Mar			↓
▶	Lobo Loco	Brain (ID 1270)	Not my Brain	Blues, Country	↓
▶	Lobo Loco	Brain - Instrumental Retro (ID 1271)	Not my Brain	Blues, Old-Time / Historic	↓
▶	Lobo Loco	Madness is Everywhere (ID 1228)	Salad Mixed	Blues, Country & Western	↓
▶	Lobo Loco	Allright in Louisiana (ID 1234)	Salad Mixed	Blues, Old-Time / Historic, Countr	↓
▶	Lobo Loco	Peaceful Morning (ID 1229)	Salad Mixed	Blues, Ambient, Instrumental	↓

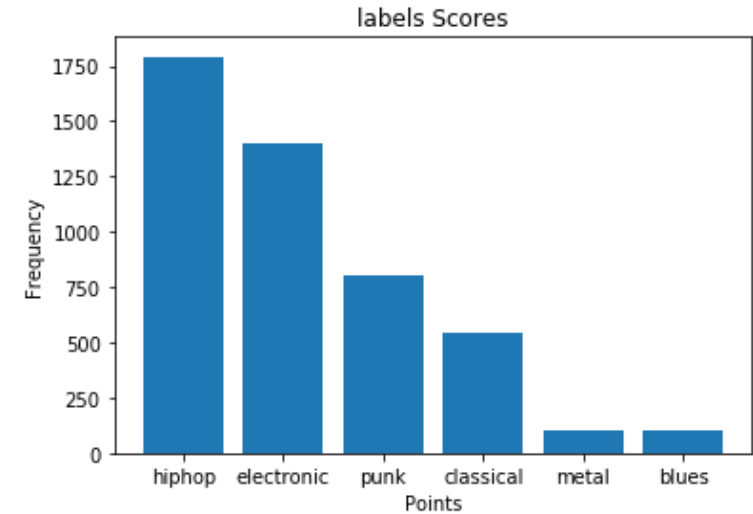
# Dataset

- Veri seti her biri 30 saniye uzunluğunda 25.000 parça mp3 ve 16 türden oluşmaktadır. Tez çalışması kapsamında kullanılması için 6 farklı tür (**blues**, **classical**, **electronic**, **hip-hop**, **metal**, **punk** ) seçilip veri seti oluşturulmuştur.

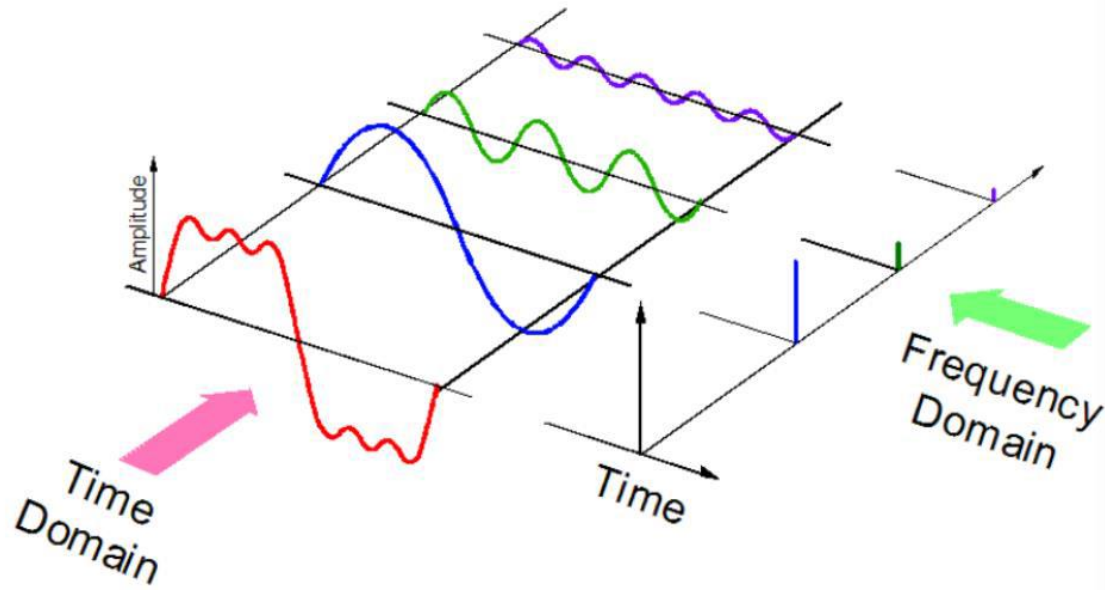
- 0\_blues
- 1\_classical
- 3\_disco
- 4\_electronic
- 5\_folk
- 6\_hiphop
- 8\_metal
- 9\_pop
- 10\_punk
- 11\_reggae
- 12\_rock

004850.mp3	6	Preludes, Book 2 - "Ge
004851.mp3	4	Preludes, Book 2 - La ç
010026.mp3	1	Sonata No. 1 in F Minc
010027.mp3	2	Sonata No. 1 in F Minc
010028.mp3	3	Sonata No. 1 in F Minc
010029.mp3	4	Sonata No. 1 in F Minc
010030.mp3	1	Sonata No. 32 in C Mir
010032.mp3	1	Promenade Allegro gi
010033.mp3	2	I. Gnomus Vivo
010035.mp3	4	II. Il vecchio castello A
010037.mp3	6	III. Tuileries Allegretto
010038.mp3	7	IV. Bydlo Sempre mod
010040.mp3	9	V. Ballet des poussins i
010041.mp3	10	VI. Samuel Goldenberç

- 004850.wav
- 004851.wav
- 010026.wav
- 010027.wav
- 010028.wav
- 010029.wav
- 010030.wav
- 010032.wav
- 010033.wav
- 010035.wav
- 010037.wav
- 010038.wav
- 010040.wav
- 010041.wav
- 010042.wav



# Audio Processing



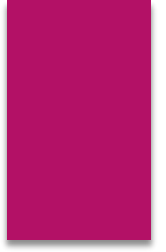
- ▶ Audio, frekans, bant genişliği desibel gibi parametrelere sahip olan bir ses sinyali şeklinde temsil edilebilir.
- ▶ Tipik bir ses sinyali, genlik ve zamanın bir fonksiyonu olarak ifade edilir.
- ▶ Python'ın librosa kütüphanesi ile ses işleme yapılmıştır.

# Ses Dosyasının Yükleneceği

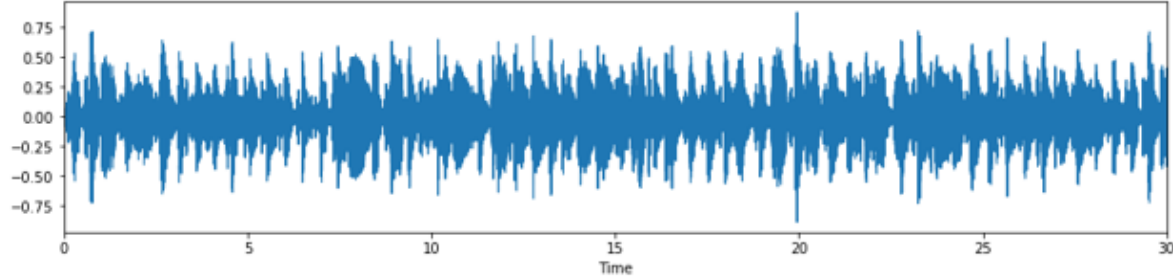
Name	Type	Size	Value
ses	str	1	classical.00000.wav
sr	int	1	22050
y	float32	(661794,)	[-0.02008057 -0.01748657 0.00418091 ... 0.01934814 0.027771 0.031 ...

y: ses zaman serisi  
sr: ses frekansı

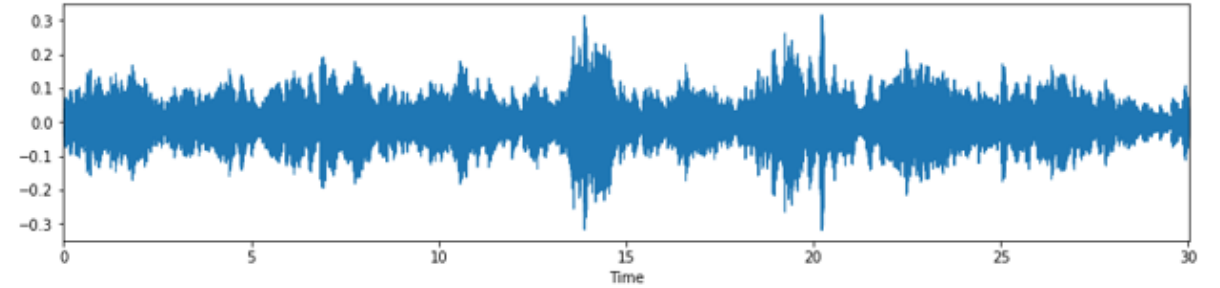
# Sesin Görüntülenmesi



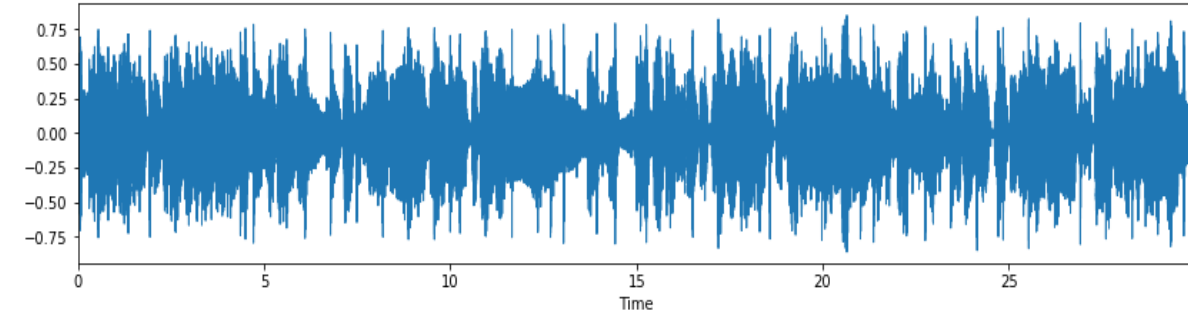
BLUES



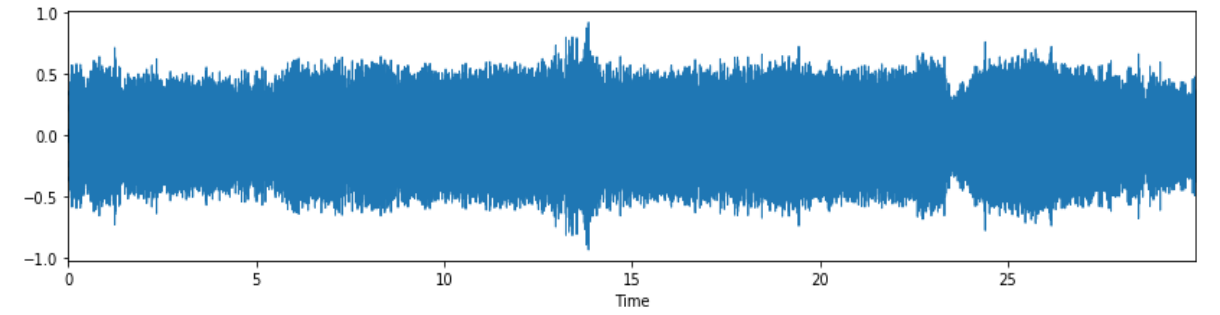
CLASSICAL



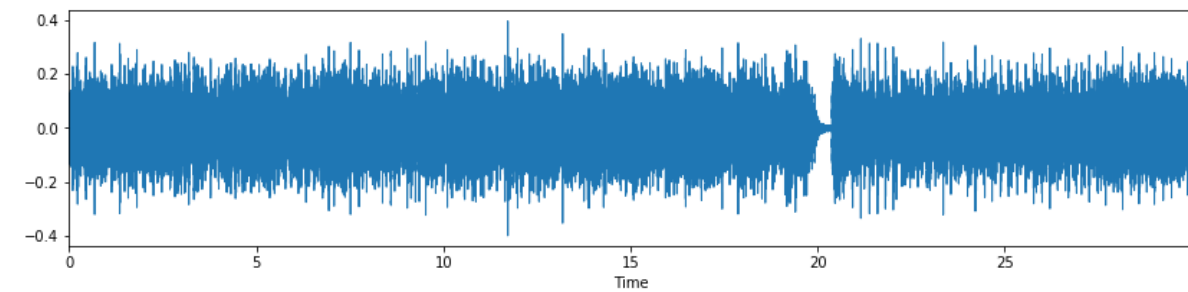
HIP-HOP



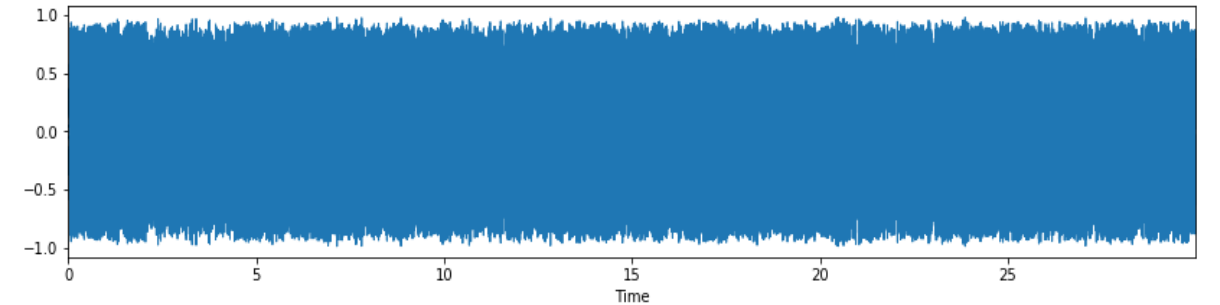
ELECTRONIC



METAL



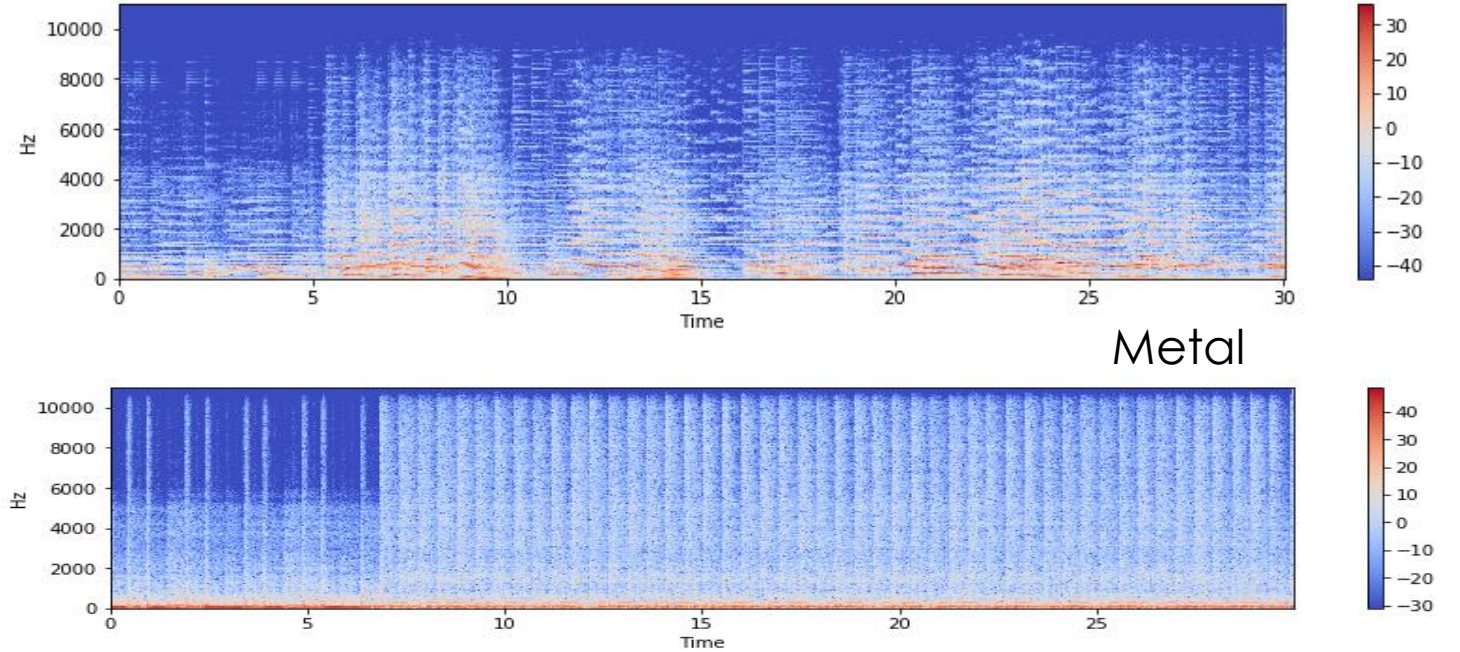
PUNK



# Spektogram

- Spektogram, belirli bir dalga formunda bulunan çeşitli frekanslarda bir sinyalin gücünü veya yüksekliğini temsil eden görsele verilen addır.

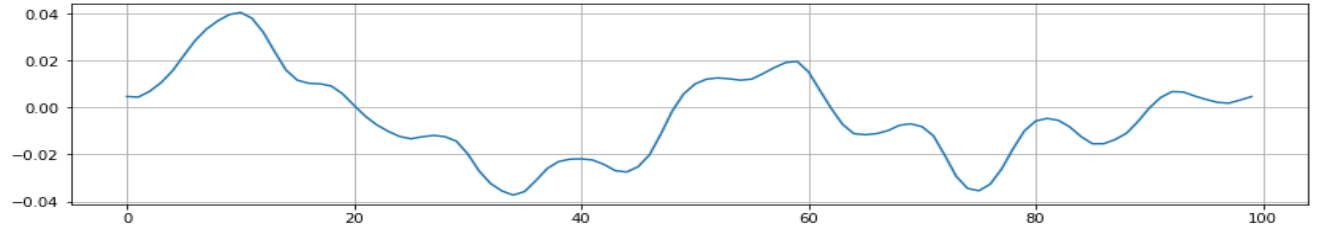
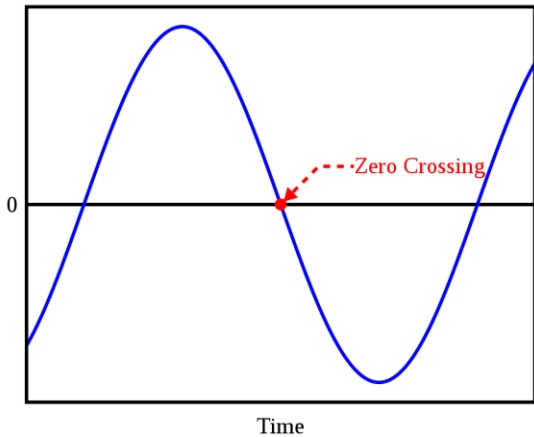
Dikey eksen frekansları (0-10 KHz)  
Yatay eksen audionun süresini  
gösterir.



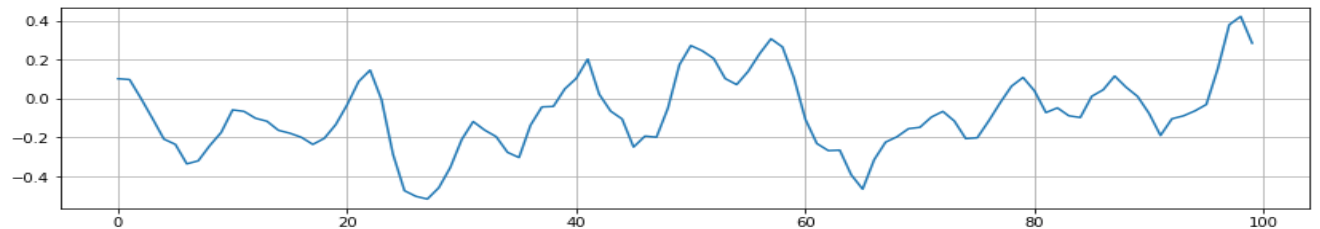
# Öznitelik Çıkarımı ( Feature Extraction)

## ► Zero Crossing Rate

Zero crossing rate, bir sinyal boyunca işaret değiştirme oranıdır, yani sinyalin pozitiften negatife geçişi veya geri dönme hızıdır. Bu özellik müzik bilgisi alımında yoğun bir şekilde kullanılmıştır.



Classical  
(4)



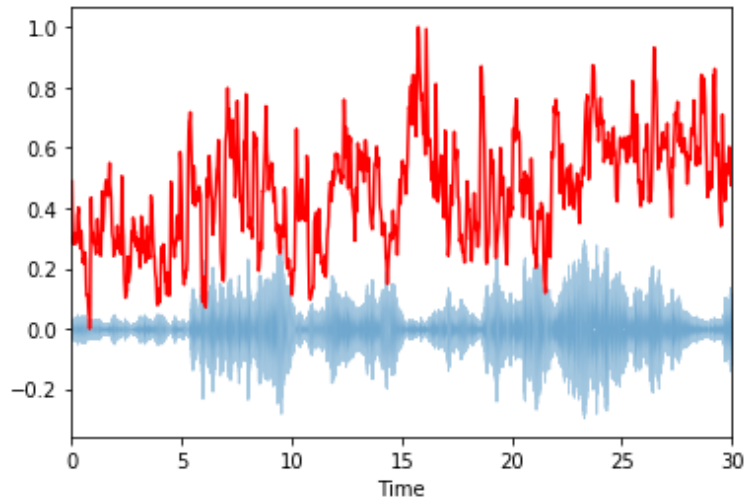
Punk  
(12)



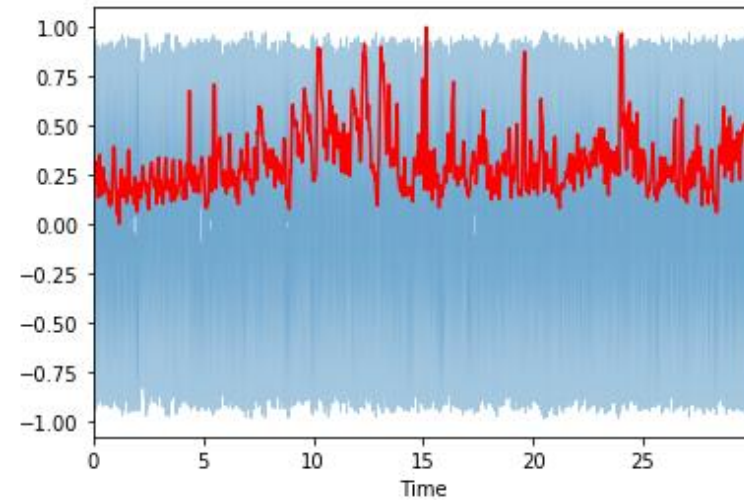
### ► Spectral Centroid

Spektrumun kütle merkezinin nerede olduğunu gösterir. Bir ses için kütle merkezinin nerede bulunduğu ve seste bulunan frekansların ağırlıklı ortalaması olarak hesaplandığını gösterir.

Classical



Punk

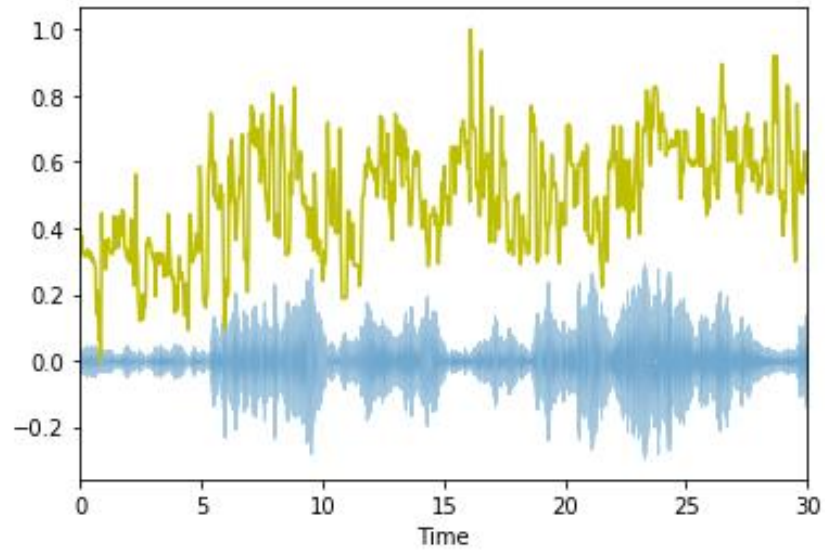




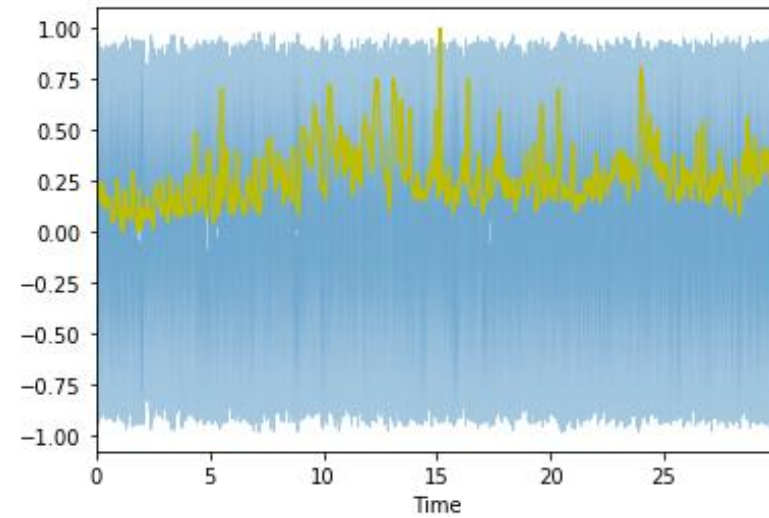
## ► Spectral Rolloff

Sinyal şeklinin ölçüsüdür. Toplam spektral enerjisinin belli bir yüzdesini temsil eder.

Classical



Punk

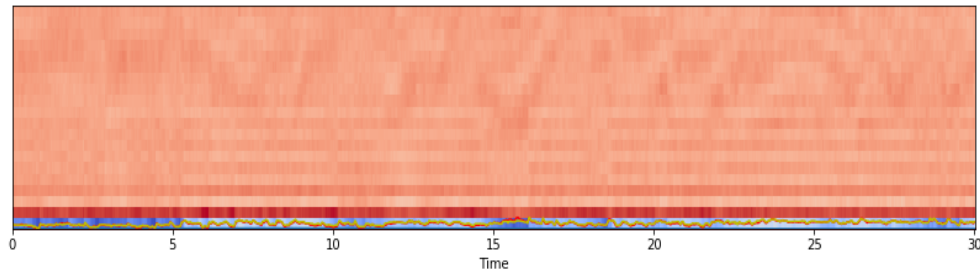


### ► Mel-Frekans Kepstral Katsayıları (Mel-Frequency Cepstral Coefficients)

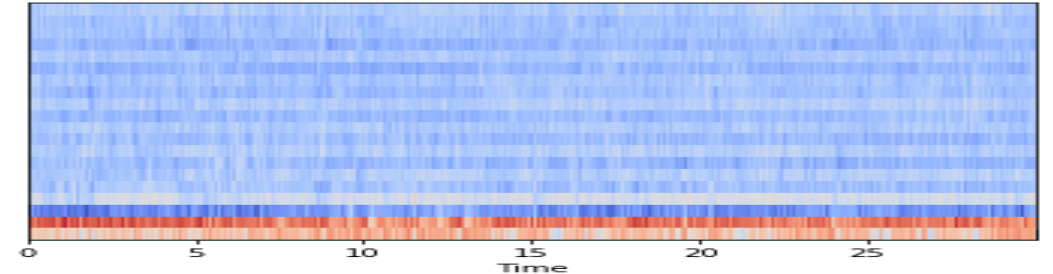
Ses sinyalinin kısa zamanlı güç spektrumunun Mel ölçeği üzerindeki ifadesidir. Frekanstan Mel ölçeğine dönüşüm formülü aşağıda ki gibidir. M, mel ölçeğini f ise Frekansı belirtir.

$$M = 1125 \times \ln(1 + (f \div 700))$$

Classical

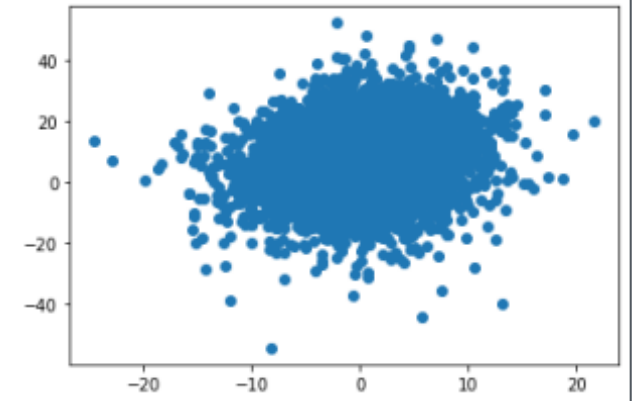


Punk



# Veri inceleme adımları ve yapılan işlemler

- ▶ Eksik veri olmadığı gözlemlenmiştir.
- ▶ Veri değişken tipleri incelendiğinde filename ve class (label/sınıf) değerlerinin kategorik verileri olduğu gözlemlenmiştir. Label\_Encoder işlemi uygulandı.
- ▶ Özellikler ( sütunlar ) arası uyuma bakmak için korelasyon katsayısı incelenip görselleştirilmiştir.
- ▶ Kullanılmayacak öznitelikler temizlendi



# Feature Selection (Öznitelik Seçimi)

Dataframe, öznitelik isimleri ve dönüşüm uygulandıktan sonraki durumları verilmiştir.

```
In [73]: data
Out[73]:
```

	filename	chroma_stft	rmse	...	mfcc19	mfcc20	label
0	blues.00000.wav	0.349943	0.130225	...	-2.300208	1.219928	blues
1	blues.00001.wav	0.340983	0.095918	...	-0.287431	0.531573	blues
2	blues.00002.wav	0.363603	0.175573	...	-3.433434	-2.226821	blues
3	blues.00003.wav	0.404779	0.141191	...	-0.619690	-3.408233	blues
4	blues.00004.wav	0.308590	0.091563	...	-4.409333	-11.703781	blues
5	blues.00005.wav	0.302346	0.103468	...	-3.046866	-8.115809	blues
6	blues.00006.wav	0.291308	0.141796	...	-3.449033	-6.495511	blues
7	blues.00007.wav	0.307921	0.131785	...	-4.693749	-8.638613	blues
8	blues.00008.wav	0.409037	0.142438	...	3.409809	-2.698353	blues
9	blues.00009.wav	0.274009	0.081352	...	2.891266	-4.233204	blues
10	blues.00010.wav	0.303954	0.142939	...	-0.743599	-4.986112	blues
11	blues.00011.wav	0.367137	0.065713	...	1.836151	-4.897420	blues
12	blues.00012.wav	0.269320	0.119072	...	-1.418707	-5.932607	blues
...	...	...	...	...	...	...	...
4725	149940.wav	0.532339	0.176460	...	-1.704041	1.504423	punk
4726	150479.wav	0.476088	0.361779	...	-4.396061	0.628166	punk
4727	150481.wav	0.402240	0.374054	...	-5.500439	2.192954	punk
4728	150482.wav	0.430244	0.387793	...	-10.172757	6.198986	punk
4729	151964.wav	0.473416	0.230945	...	-4.207032	0.603890	punk
4730	151965.wav	0.495777	0.227619	...	-6.927705	-3.049185	punk
4731	155314.wav	0.421965	0.141610	...	-8.734600	-5.078706	punk

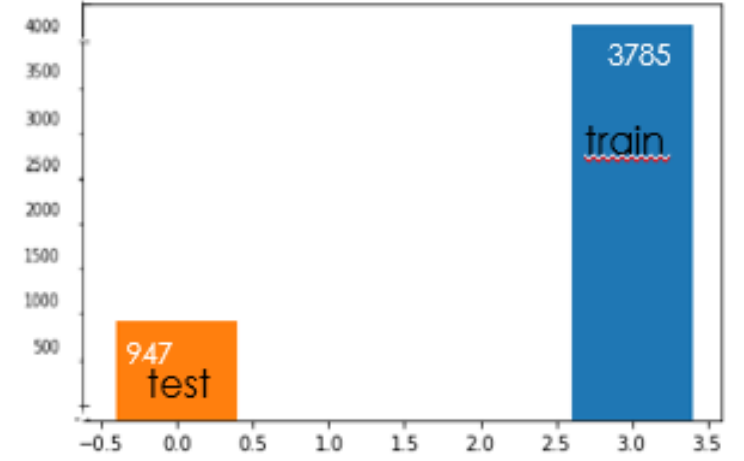
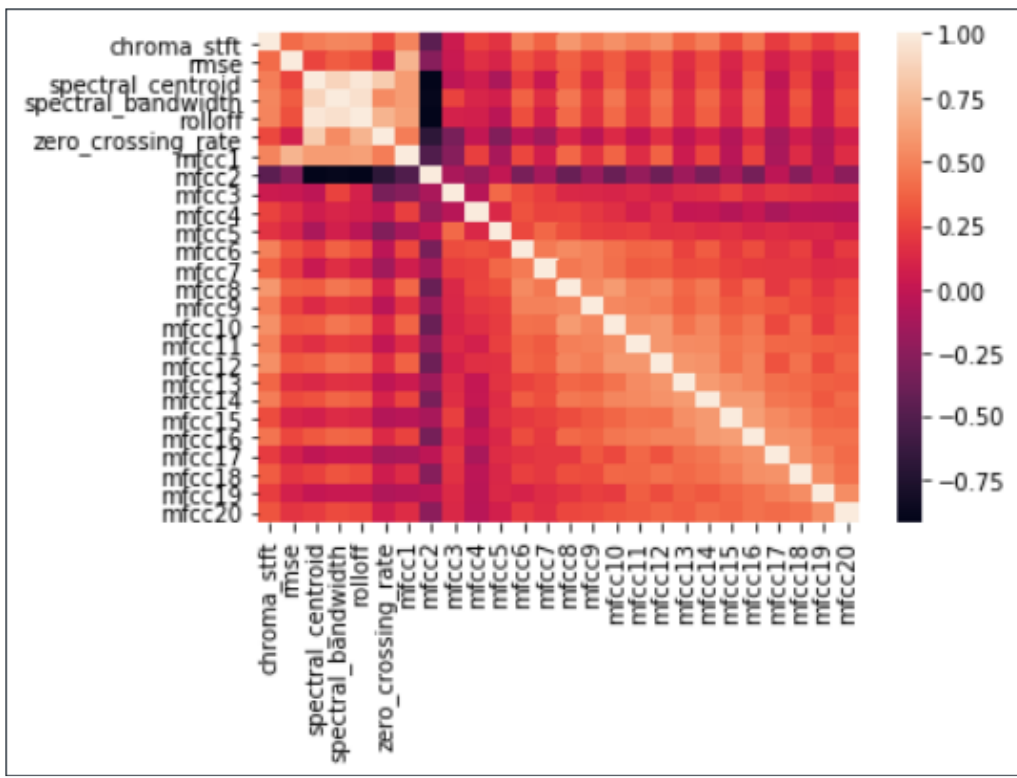
[4732 rows x 28 columns]

```
filename      object
chroma_stft   float64
rmse          float64
spectral_centroid float64
spectral_bandwidth float64
rolloff       float64
zero_crossing_rate float64
mfcc1         float64
mfcc2         float64
mfcc3         float64
mfcc4         float64
mfcc5         float64
mfcc6         float64
mfcc7         float64
mfcc8         float64
mfcc9         float64
mfcc10        float64
mfcc11        float64
mfcc12        float64
mfcc13        float64
mfcc14        float64
mfcc15        float64
mfcc16        float64
mfcc17        float64
mfcc18        float64
mfcc19        float64
mfcc20        float64
label         object
```

```
In [83]: data.dtypes
Out[83]:
```

chroma_stft	float64
rmse	float64
spectral_centroid	float64
spectral_bandwidth	float64
rolloff	float64
zero_crossing_rate	float64
mfcc1	float64
mfcc3	float64
mfcc4	float64
mfcc5	float64
mfcc6	float64
mfcc7	float64
mfcc8	float64
mfcc9	float64
mfcc10	float64
mfcc11	float64
mfcc12	float64
mfcc13	float64
mfcc14	float64
mfcc15	float64
mfcc16	float64
mfcc17	float64
mfcc18	float64
mfcc19	float64
mfcc20	float64
label	int32

```
In [91]: data.isna().sum().sum()
Out[91]: 0
```



## Öznitelikler arası korelasyon matrisi

data - DataFrame

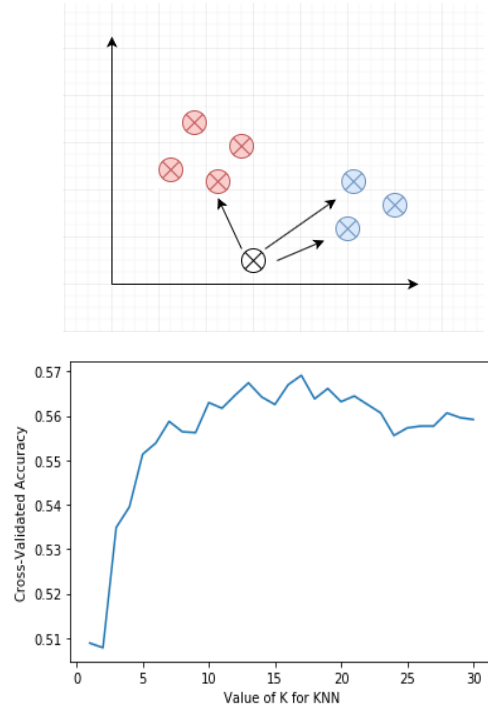
Index	chroma_stft	rmse	spectral_centroid	spectral_bandwidth	rolloff	zero_crossing_rate	mfcc1	mfcc2	mfcc3	mfcc4	mfcc5	mfcc6	mfcc7	mfcc8	mfcc9	mfcc10	mfcc11	mfcc12	mfcc13
0	0.349943	0.130225	1784.42	2002.65	3806.49	0.0830664	-113.597	121.557	-19.1588	42.351	-6.37646	18.6189	-13.6979	15.3446	-12.2853	10.9805	-8.32432	8.81067	-3.66737
1	0.340983	0.0959184	1529.84	2038.62	3548.82	0.0560443	-207.557	124.007	8.93056	35.8747	2.91604	21.5237	-8.5547	23.3587	-10.1036	11.9037	-5.56039	5.3768	-2.23912
2	0.363603	0.175573	1552.48	1747.17	3040.51	0.0763007	-90.7544	140.46	-29.11	31.689	-13.987	25.7548	-13.6496	11.6293	-11.7806	9.70644	-13.1231	5.78926	-8.90522
3	0.404779	0.141191	1070.12	1596.33	2185.03	0.0333089	-199.431	150.099	5.64759	26.8719	1.75446	14.2383	-4.83088	9.29797	-0.757741	8.14901	-3.19631	6.08768	-2.47642
4	0.30859	0.0915632	1835.49	1748.36	3580.95	0.1015	-160.266	126.199	-35.6054	22.1533	-32.4893	10.8645	-23.3579	0.503117	-11.8058	1.2068	-13.0838	-2.80638	-6.93412
5	0.302346	0.103468	1831.94	1729.48	3480.94	0.0940399	-177.869	118.197	-17.5507	30.7586	-21.7427	11.9038	-20.7342	3.1806	-8.58348	-0.936488	-11.7763	-2.42061	-9.33936

Modelleme yapılmadan önce veri setinin son hali

# Sınıflandırma ve Performans Değerlendirmesi

## ► KNN ( en yakın komşu) ile modelleme

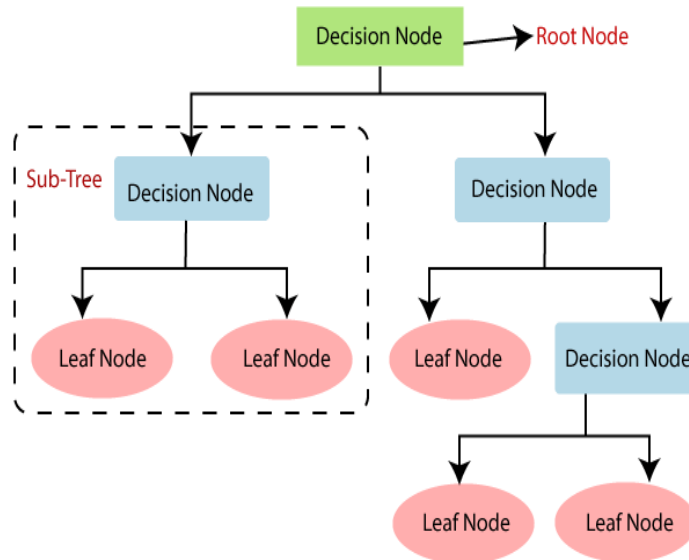
## ► Decision Tree ile modelleme



```
In [12]: runcell('KNN', 'C:/Users/TOSHIBA/Desktop/tk/train.p
Accuracy: 0.5755015839493136
knn:
precision    recall  f1-score   support

0           0.43      0.21      0.29         14
1           0.81      0.88      0.84        108
2           0.50      0.44      0.46        287
3           0.57      0.63      0.60        367
4           0.63      0.63      0.63          19
5           0.53      0.51      0.52         152

accuracy                0.58         947
macro avg              0.58      0.55      0.56         947
weighted avg           0.57      0.58      0.57         947
```

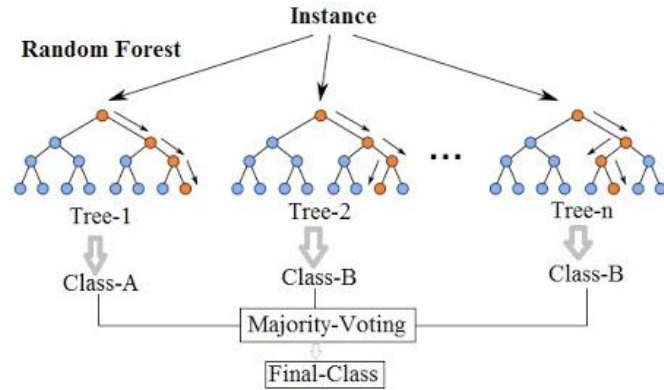


```
In [13]: runcell('Decision Tree', 'C:/Users/TOSHIBA/Desktop
Accuracy: 0.6187961985216474
Decision Tree
precision    recall  f1-score   support

0           0.35      0.43      0.39         14
1           0.83      0.83      0.83        108
2           0.53      0.54      0.54        287
3           0.62      0.57      0.60        367
4           0.78      0.74      0.76          19
5           0.64      0.72      0.67         152

accuracy                0.62         947
macro avg              0.62      0.64      0.63         947
weighted avg           0.62      0.62      0.62         947
```

## ► Random Forest ile modelleme



```
In [40]: runfile('C:/Users/TOSHIBA/Desktop/tk/train.py', w
0.7624076029567054
Random Forest:
```

	precision	recall	f1-score	support
0	0.85	0.79	0.81	14
1	0.93	0.92	0.93	103
2	0.72	0.64	0.68	286
3	0.73	0.81	0.77	359
4	0.94	0.76	0.84	21
5	0.79	0.76	0.77	164
accuracy			0.76	947
macro avg	0.83	0.78	0.80	947
weighted avg	0.76	0.76	0.76	947

## ► Naive Bayes ile modelleme

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)}$$

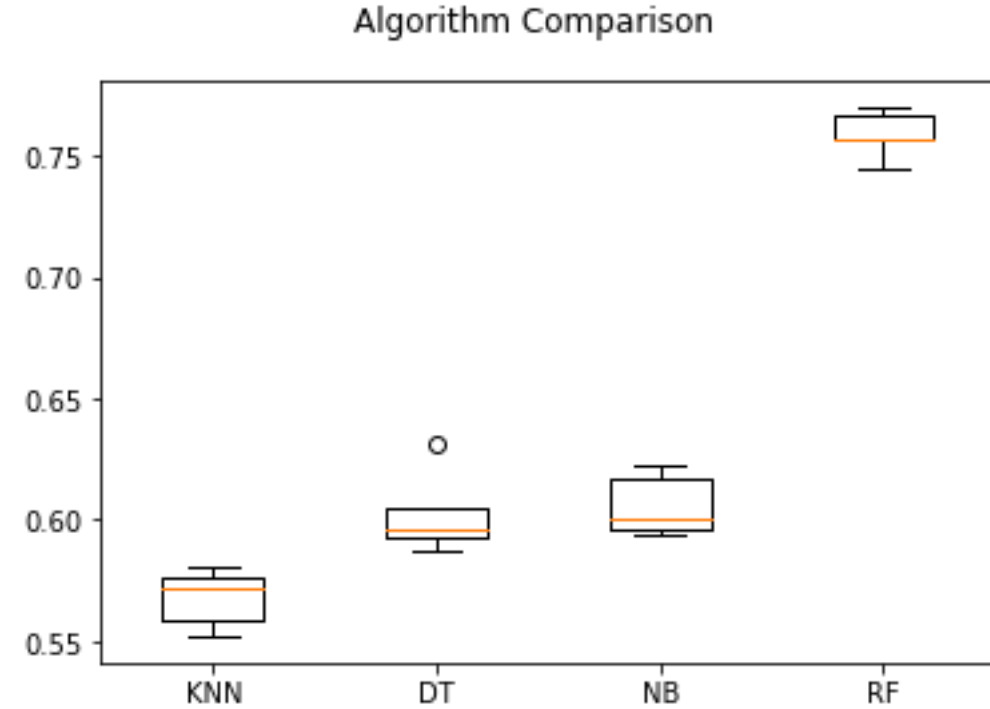
Labels: Likelihood, Class Prior Probability, Posterior Probability, Predictor Prior Probability

$$P(c|X) = P(x_1|c) \times P(x_2|c) \times \dots \times P(x_n|c) \times P(c)$$

```
In [15]: runfile('Naive Bayes', 'C:/Users/TOSHIBA/Desktop/t
Accuracy: 0.5966209081309398
GaussianNB
```

	precision	recall	f1-score	support
0	0.23	0.36	0.28	14
1	0.74	0.92	0.82	108
2	0.58	0.24	0.34	287
3	0.59	0.69	0.63	367
4	0.57	0.89	0.69	19
5	0.58	0.80	0.67	152
accuracy			0.60	947
macro avg	0.55	0.65	0.57	947
weighted avg	0.59	0.60	0.57	947

Sonuçların  
değerlendirilmesi ve  
öneri sistemi için en  
iyi model seçimi

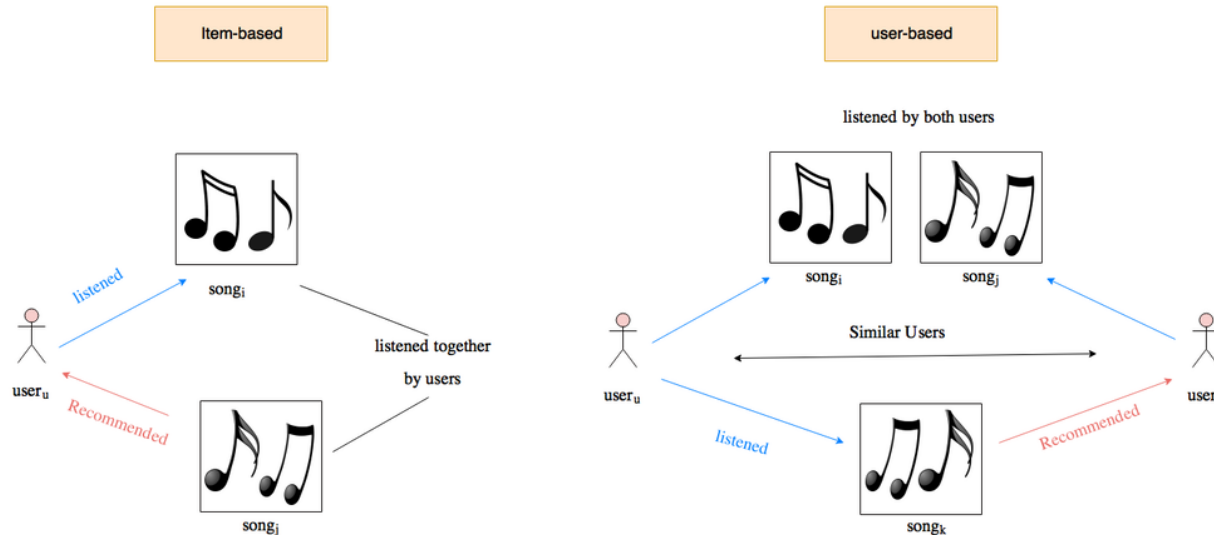




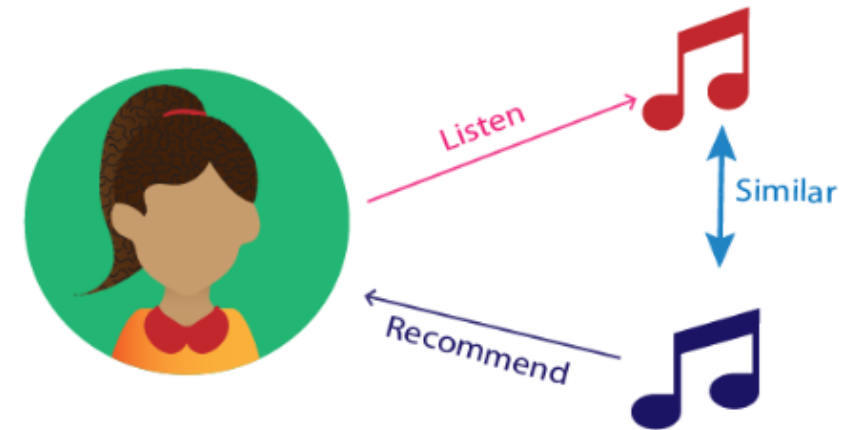
# Recommendation (Öneri) Sistemi

- Genellikle Tavsiye Sistemleri olarak adlandırılan, büyük bir bilgi tabanından faydalı bilgileri filtreleyerek kullanıcıya en alakalı ve doğru öğeleri sunmayı amaçlayan bir algoritmalar. Öneri motorları, tüketicilerin seçimlerini öğrenerek veri setindeki veri modellerini keşfeder ve onların ihtiyaçları ve ilgileriyle ilişkili sonuçlar üretir.

## Collaborative (İşbirlikçi) Filtreleme



## Content Based (İçerik Tabanlı) Filtreleme

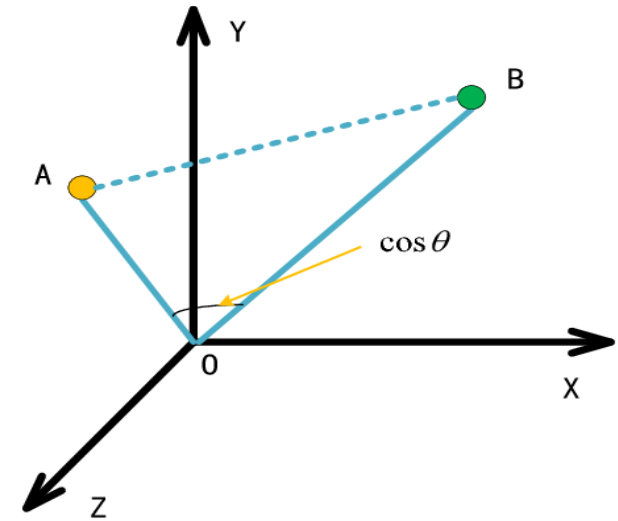


Seilen müziğın diğeri müziklerin öznitelik değeri ile arasında ki ilişkinin anlaşılabilmesi için benzerlik fonksiyonu kullanılmıştır.

## ► Kosinüs Benzerliğı ( Cosine Similarity)

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}},$$

Kosinüs benzerliğı, bir ürün uzayının iki vektörü arasındaki benzerliğı ölçer. İki vektör arasındaki açının kosinüsü ile ölçölür ve iki vektörün aynı yönü gösterip göstermediğini belirler



# Önerilen Müzikler

Benzerlik oranları

cos\_sim - NumPy array

	0
0	0.998027
1	0.997303
2	0.998289
3	0.991826
4	0.998096
5	0.998417
6	0.99886
7	0.9994
8	0.997788
9	0.998376
10	0.999584
11	0.998261
12	0.999519

Format Resize Background

Önerilen ilk 5 benzer müziğin listesi

oneri - DataFrame

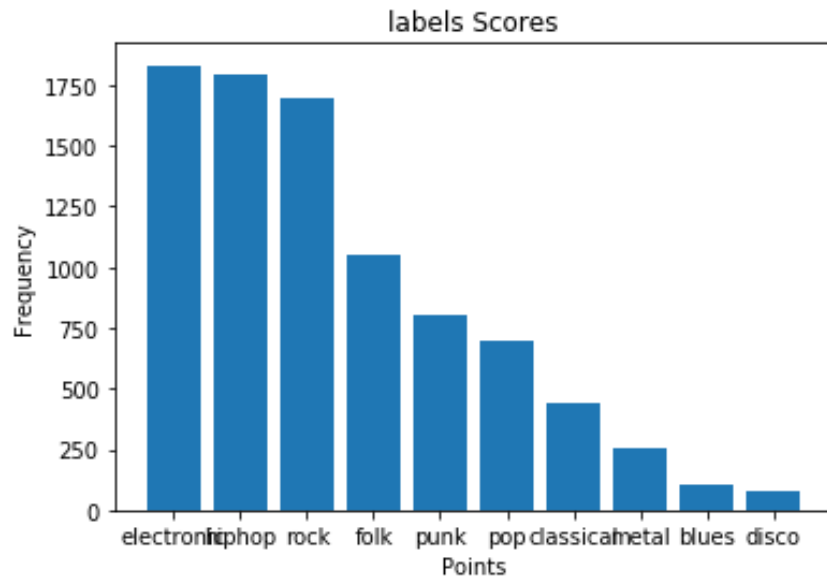
Index	simila	adi	label
105	0.999977	classical.00005.wav	1
383	0.999941	055953.wav	1
388	0.999936	055958.wav	1
4657	0.999916	131166.wav	5
110	0.999916	classical.00010.wav	1

# SONUÇ (CONCLUSION)

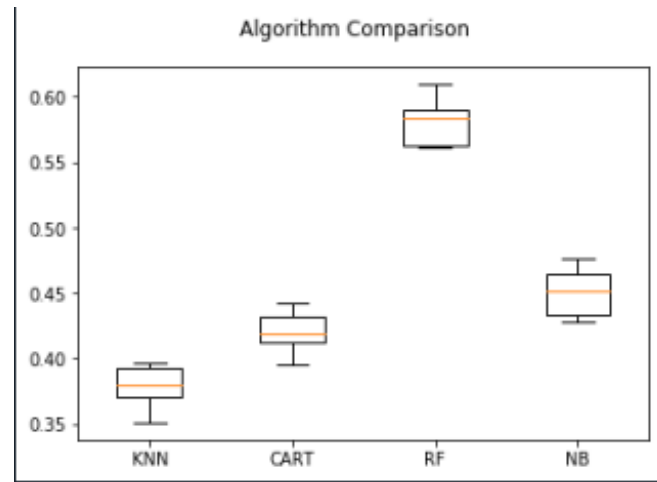
# Karşılaşılan Problemler

10 farklı tür ile çalışıldığında ;

Veri setinde ki türler arası boyut farklı



Düşük accuracy değerleri

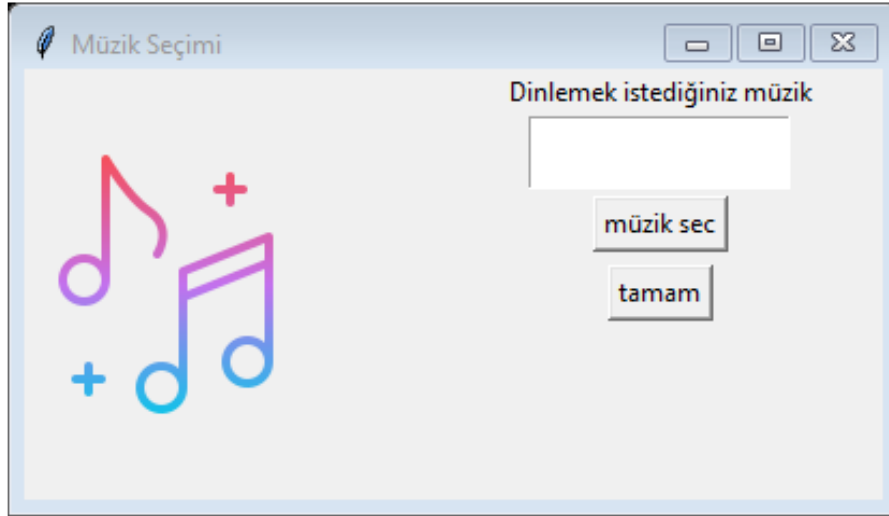


Underfitting durumu

Random Forest:

	precision	recall	f1-score	support
0	1.00	0.67	0.80	15
1	0.78	0.86	0.82	94
2	0.00	0.00	0.00	15
3	0.48	0.59	0.53	336
4	0.64	0.77	0.70	211
5	0.54	0.60	0.57	360
6	0.64	0.16	0.26	55
7	0.19	0.03	0.06	151
8	0.62	0.37	0.46	171
9	0.53	0.67	0.59	338
accuracy			0.55	1746
macro avg	0.54	0.47	0.48	1746
weighted avg	0.53	0.55	0.53	1746

# Kullanıcı Ara Yüzü



# Kullanıcı ara yüzü videolu gösterim

The image shows the Spyder Python IDE interface. The main window is titled "Spyder (Python 3.7)". The menu bar includes File, Edit, Search, Source, Run, Debug, Consoles, Projects, Tools, View, and Help. The toolbar contains various icons for file operations, running, and debugging. The file explorer shows the current project location as "C:\Users\TOSHIBA\Desktop\tk\app.py". The code editor displays the following Python code:

```
7
8 import os
9 import threading
10 from tkinter import *
11 from pygame import mixer
12 from tkinter import filedialog
13
14 # DOSYADAN MÜZİK SEÇİMİ
15 sec = Tk()
16 sec.geometry('400x200')
17 sec.title("Müzik Seçimi")
18
19 foto_nota=PhotoImage(file="music-note.png")
20 labelfoto2=Label( image=foto_nota )
21 labelfoto2.pack(side=LEFT)
22
23
24 leftframe = Frame(sec)
25 leftframe.pack(side=LEFT, padx=30)
26
27
28 rightframe = Frame(sec)
29 rightframe.pack()
30
31 topframe = Frame(rightframe)
32 topframe.pack()
33
34 r=[]
35
36 def file_al():
37     global filename_path
38     filename_path = filedialog.askopenfilename()
39     add_to_playlist(filename_path)
40
41
42 def add_to_playlist(filename):
```

The right sidebar contains a "Usage" window with the following text:

Usage

Here you can get help of any object by pressing front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in [Preferences > Help](#).

New to Spyder? Read our [tutorial](#)

The bottom right corner shows the "Console 1/A" window with the text "In [12]:". The status bar at the bottom indicates the current file is "app.py", line 18, column 1, with a UTF-8 encoding, CRLF line endings, and a memory usage of 71%.

Teşekkür Ederim..