

**T.C.**

**BİLECİK ŞEYH EDEBALİ ÜNİVERSİTESİ**

**FEN FAKÜLTESİ**

**İSTATİSTİK VE BİLGİSAYAR BİLİMLERİ BÖLÜMÜ**

**2023-2024 AKADEMİK YILI BAHAR DÖNEMİ**

**BİLGİSAYAR PROGRAMLAMA DERSİ**

**ÖDEV RAPORU**

**SPOTIFY VE YOUTUBE PLATFORMLARINDAKİ**

**TÜRKİYEDE EN ÇOK İZLENEN/DİNLENEN İLK 50 İÇERİĞİ  
PYTHON DİLİNDE GÖRSELLEŞTİRİLMESİ VE ANALİZLERİ**

**Seyfettin Budak**

**BİLECİK 2024**

# İçindekiler

[1 İçindekiler 1](#_Toc169624705)

[KONU 2](#_Toc169624706)

[ANALİZLER İÇİN GEREKLİ ADIMLAR VERİLERİN KULLANIMI 2](#_Toc169624707)

[YILLARA GÖRE SPOTIFY VERİ ANALİZLERİ 4](#_Toc169624708)

[YILLARA GÖRE YOUTUBE VERİ ANALİZLERİ 11](#_Toc169624709)

[KAYNAKÇA VE EKLER 21](#_Toc169624710)

# KONU

Spotify ve Youtube sosyal medya platformlarının Türkiyede yıllara göre (2020-2021-2022-2023) en çok izlenen/dinlenen ilk 50 içeriği ile ilgili Python dilinde görselleştirme işlemi ve analizi.

Kaynak olarak veriler kaynakçada belirttiğim Javascript dili ile yazdığım kodlar yardımı ile Youtube API V3 ve Spotify WEB API kullanılarak elde edilmiştir.

Sonuçlar grafikler ile sunulmuş ve grafikler ile ilgili değerlendirmeler yapılarak tamamlanmıştır.

# ANALİZLER İÇİN GEREKLİ ADIMLAR VERİLERİN KULLANIMI

Gerekli kütüphanelerin yüklenmesi için aşağıdaki kodları kullandım.

import numpy as np

import pandas as pd

!pip install isodate # isodate modülünü indirme

#(colabda dahili olmadığı için indiriyoruz)

import isodate

import matplotlib as mpl

import matplotlib.pyplot as plt

Sonrasında CSV dosyalarımın okunabilmesi için

data1 = pd.read\_csv("https://raw.githubusercontent.com/bymayfe/YoutubeSpotifyStatsVirtualization/main/datasets/spotify2020.csv")

data2 = pd.read\_csv("https://raw.githubusercontent.com/bymayfe/YoutubeSpotifyStatsVirtualization/main/datasets/spotify2021.csv")

data3 = pd.read\_csv("https://raw.githubusercontent.com/bymayfe/YoutubeSpotifyStatsVirtualization/main/datasets/spotify2022.csv")

data4 = pd.read\_csv("https://raw.githubusercontent.com/bymayfe/YoutubeSpotifyStatsVirtualization/main/datasets/spotify2023.csv")

data5 = pd.read\_csv("https://raw.githubusercontent.com/bymayfe/YoutubeSpotifyStatsVirtualization/main/datasets/youtube2020.csv")

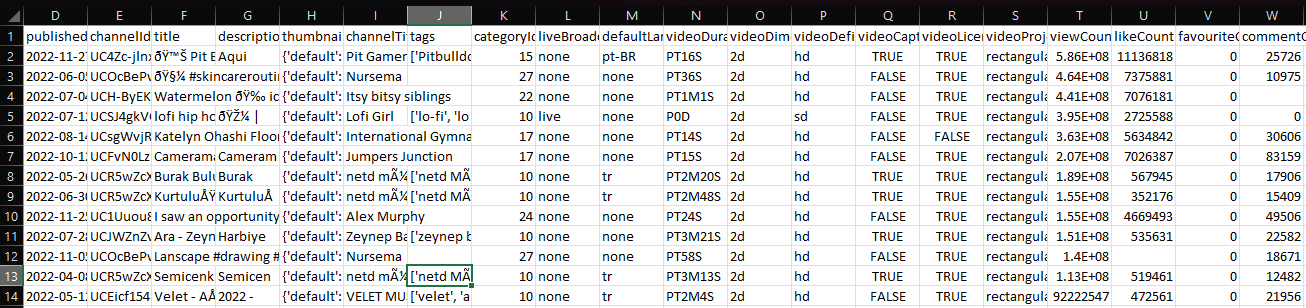
data6 = pd.read\_csv("https://raw.githubusercontent.com/bymayfe/YoutubeSpotifyStatsVirtualization/main/datasets/youtube2021.csv")

data7 = pd.read\_csv("https://raw.githubusercontent.com/bymayfe/YoutubeSpotifyStatsVirtualization/main/datasets/youtube2022.csv")

data8 = pd.read\_csv("https://raw.githubusercontent.com/bymayfe/YoutubeSpotifyStatsVirtualization/main/datasets/youtube2023.csv")

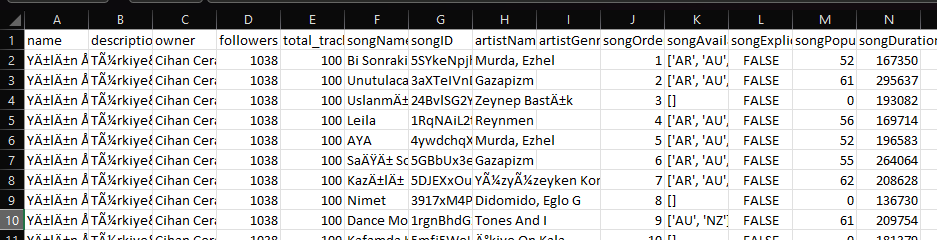
Grafikleri sunmadan önce verilerimin nasıl gözüktüğünü paylaşayım.

data7.head(14)



**Tablo 1** (youtube2022.csv) ayrıca tüm Youtube dataları buna benzemektedir.

data1.head(11)

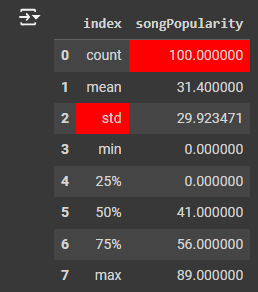
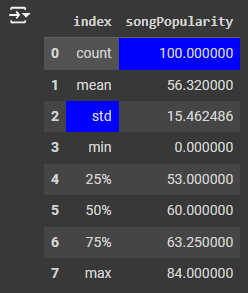


**Tablo 2** (spotify2020.csv) ayrıca tüm Spotify dataları buna benzemektedir.

statistics = data1['songPopularity'].describe().reset\_index()

statistics.style.format({"number": "{:20,.0f}"}).highlight\_max(color='RED') statistics = data2['songPopularity'].describe().reset\_index()

statistics.style.format({"number": "{:20,.0f}"}).highlight\_max(color='BLUE')

**Tablo 3** (Kırmızı): 2020 yılına ait **Tablo 4** (Mavi): 2021 yılına ait

Ortalama, standart sapma, min/max değerleri ve çeyrekliklerini göstermektedir

# YILLARA GÖRE SPOTIFY VERİ ANALİZLERİ

Bu değerleri bulduktan sonra, matplotlib ve pandas kütüphanelerini kullanarak aşağıda belirttiğim 20 grafiği oluşturdum ve altlarında açıklamasıyla beraber sizlere sundum.

Burada 2020 ve 2021 yılına ait Spotify şarkı liste sıralaması ve şarkının toplam popülerliğini matplotlib kütüphanesi kullanarak plot ile görselleştirdim.

plt.plot(data1['songOrder'], data1['songPopularity'] , label = '2020', color="red")

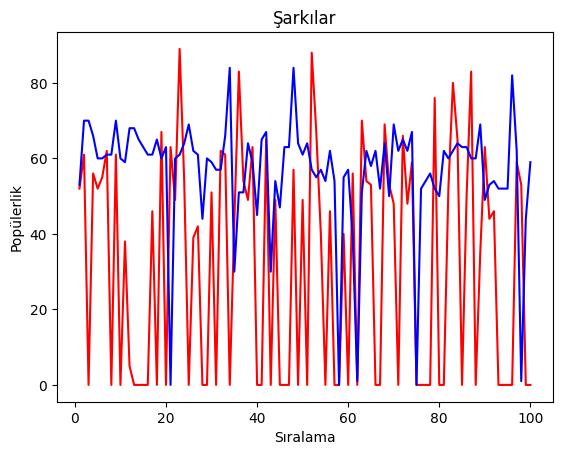
plt.plot(data2['songOrder'], data2['songPopularity'] , label = '2021' , color="blue")

plt.title('Şarkılar')

plt.xlabel('Sıralama')

plt.ylabel('Popülerlik')

plt.show()



**Tablo 5**

Burada kırmızı olan 2020 mavi olan ise 2021 yılına ait veri olmak üzere aşağıdaki çıkarımları yapabiliriz.

Spotify’da 2021 yılındaki en çok ilk 50 şarkı popülerlik konusunda 2020 yılındaki en çok dinlenen ilk 50 şarkıdan daha popülerdir.

Burada 2020 ve 2021 yılına ait Spotify şarkı liste sıralaması ve şarkının toplam popülerliğini matplotlib kütüphanesi kullanarak scratter ile görselleştirdim.

plt.scatter(data1['songOrder'], data1['songPopularity'], label = '2020', color="red")

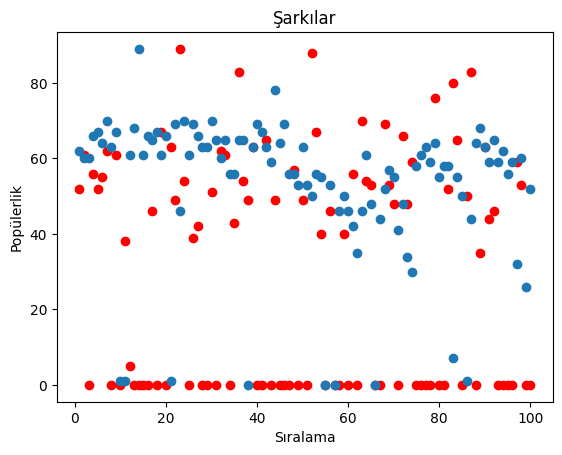
plt.scatter(data3['songOrder'], data3['songPopularity'], label = '2022')

plt.title('Şarkılar')

plt.xlabel('Sıralama')

plt.ylabel('Popülerlik')

plt.show()



**Tablo 6**

Burada kırmızı olan 2020 mavi olan ise 2022 yılına ait veri olmak üzere aşağıdaki çıkarımları yapabiliriz.

Spotify’da 2022 yılındaki en çok ilk 50 şarkı popülerlik konusunda 2020 yılındaki en çok dinlenen ilk 50 şarkıdan daha popülerdir.

Burada 2020 ve 2023 yılına ait Spotify şarkı liste sıralaması ve şarkının toplam popülerliğini matplotlib kütüphanesi kullanarak bar ile görselleştirdim.

plt.bar(data1['songOrder'], data1['songPopularity'], label = '2020', color="red")

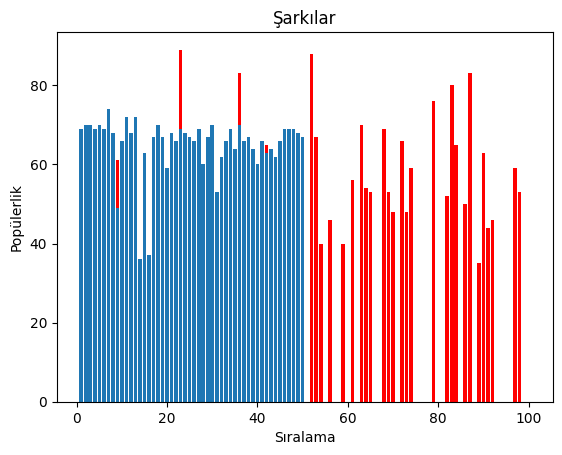
plt.bar(data4['songOrder'], data4['songPopularity'], label = '2023')

plt.title('Şarkılar')

plt.xlabel('Sıralama')

plt.ylabel('Popülerlik')

plt.show()



**Tablo 7**

Burada kırmızı olan 2020 mavi olan ise 2023 yılına ait veri olmak üzere aşağıdaki çıkarımları yapabiliriz.

Spotify’da 2023 yılındaki en çok ilk 50 şarkı popülerlik konusunda 2020 yılındaki en çok dinlenen ilk 50 şarkıdan daha popülerdir.

Burada 2020, 2021 ve 2022 yılına ait Spotify şarkıların süreleri bakımından karşılaştırma ve çıkarım yapmak için matplotlib kütüphanesi kullanarak colorbar ile görselleştirdim.

dataset1 = [data1['songDuration'], data2['songDuration'], data3['songDuration']]

dataset1 = np.array(dataset1).T

plt.imshow(dataset1, aspect='auto', cmap='hot')

plt.colorbar()  # Renk çubuğunu ekleme

plt.title('Şarkı Verileri Isı Haritası')

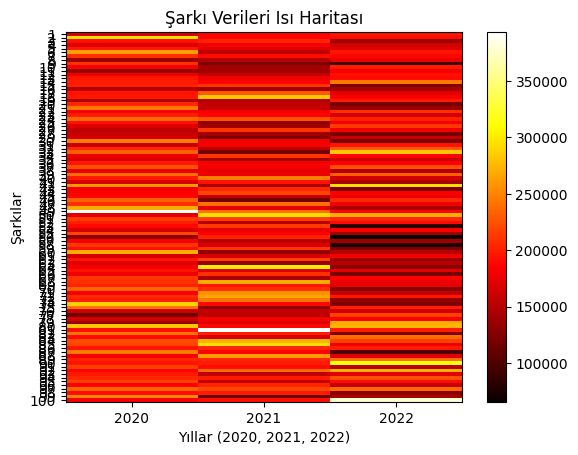
plt.xlabel('Yıllar (2020, 2021, 2022)')

plt.ylabel('Şarkılar')

plt.xticks(ticks=[0, 1, 2], labels=['2020', '2021', '2022'])

plt.yticks(ticks=range(len(data1['songDuration'])), labels=data1['songOrder'])

plt.show()



**Tablo 8**

Burada tüm yıllar için rengin hafif turuncuya gittiği ve kırmızı turuncu arasında olup 200000 milisaniye (3.33 dakika) ve 250000 milisaniye (4.16 dakika) olduğu çıkarımını ve 2020 de çıkan ve popüler olan şarkıların rengine bakarak şarkı uzunluklarının arttığı çıkarımına varıyoruz. Yıllar arttıkça şarkı süreleri kısalmaya başlamıştır.

Burada 2020 ve 2023 yılına ait Spotify şarkıların süreleri bakımından karşılaştırma ve çıkarım yapmak için matplotlib kütüphanesi kullanarak contour ile görselleştirdim.

duration\_2020 = data1['songDuration']

duration\_2023 = data4['songDuration']

# Ortak bir ölçek oluşturmak için verileri normalize edelim

popularity\_2020 = (duration\_2020 - np.min(duration\_2020)) / (np.max(duration\_2020) - np.min(duration\_2020))

popularity\_2021 = (duration\_2023 - np.min(duration\_2023)) / (np.max(duration\_2023) - np.min(duration\_2023))

X, Y = np.meshgrid(duration\_2020, duration\_2023)

Z = np.abs(X - Y)

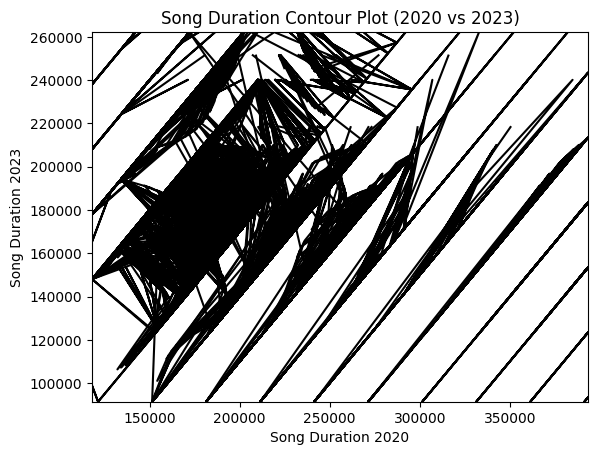
plt.contour(X, Y, Z, levels=10, colors='black')

plt.title('Song Duration Contour Plot (2020 vs 2023)')

plt.xlabel('Song Duration 2020')

plt.ylabel('Song Duration 2023')

plt.show()



**Tablo 9**

Burada Spotify şarkılarının süreleri ile ilgili tekrar bir çıkarım yapmak gerekirse, 2020 ile 2023 yılındaki üstte yaptığım çıkarımı destekleyici bir grafik görüyoruz. Yılların şarkı süreleri ile ilgili kesiştiği yerler ortalamaya yakın ve 2023 yılında 260000 milisaniye max seviye ve oralarda biraz yoğunluk oluşumu var. 2020 yılına bakacak olursak max 350000 milisaniye olup yoğunlukar oldukça az. Yani yıllar arttıkça popüler şarkıların şarkı süreleri azalmıştır.

Burada 2020 ve 2021 yılına ait Spotify şarkıların popülerlik indexleri bakımından karşılaştırma ve çıkarım yapmak için matplotlib kütüphanesi kullanarak pcolormesh ile görselleştirdim.

popularity\_2020 = data1['songPopularity']

popularity\_2021 = data2['songPopularity']

# Ortak bir ölçek oluşturmak için verileri normalize edelim

popularity\_2020 = (popularity\_2020 - np.min(popularity\_2020)) / (np.max(popularity\_2020) - np.min(popularity\_2020))

popularity\_2021 = (popularity\_2021 - np.min(popularity\_2021)) / (np.max(popularity\_2021) - np.min(popularity\_2021))

X, Y = np.meshgrid(popularity\_2020, popularity\_2021)

Z = np.abs(X - Y)

plt.pcolormesh(X, Y, Z, cmap='viridis', vmin=0, vmax=1)

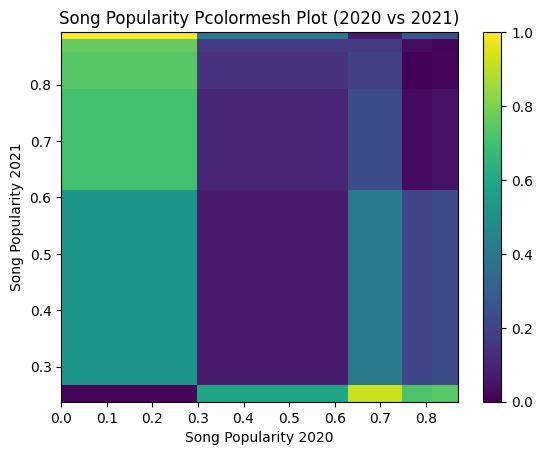
plt.colorbar()

plt.title('Song Popularity Pcolormesh Plot (2020 vs 2021)')

plt.xlabel('Song Popularity 2020')

plt.ylabel('Song Popularity 2021')

plt.show()



**Tablo 10**

Burada 2020 ve 2021 yılana ait verilerin popülerliğinin değer aralığını görüyoruz. Popülerlik indexi 0-100 arasında değer almaktadır. Popülerlik farklılıklarını listenin ilk sıralarında görüyoruz. Ve farklılıklar 0.6 (60) ile 1 (100) index değerleri arasında yer almakta. Listenin ilk sıralarının alan şarkılar yıllara göre popülerlik anlamında farklılık göstermiştir.

Burada 2023 yılına ait Spotify’da en çok dinlenen ilk 5 şarkının şarkı süreleri ile ilgili çıkarım yapmak için matplotlib kütüphanesi kullanarak pie ile görselleştirdim.

sizes = data4['songDuration'][:5]  # İlk dört popülarite değeri

labels = data4['songName'][:5]  # İlk dört şarkı ismi

colors = ['gold', 'yellowgreen', 'lightcoral', 'lightskyblue', "pink"]

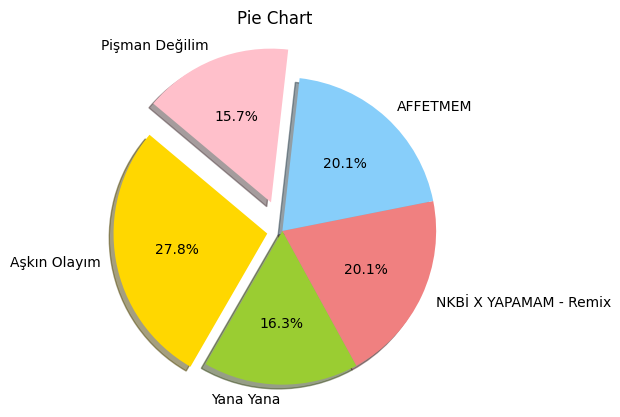
explode = (0.1, 0, 0, 0,0.2)  # İlk dilimi öne çıkar

plt.pie(sizes, explode=explode, labels=labels, colors=colors, autopct='%1.1f%%', shadow=True, startangle=140)

plt.title('Pie Chart')

plt.axis('equal')  # Daire şeklinde çizim

plt.show()



**Tablo 11**

Burada en çok dinlenen şarkının en çok şarkı süresine sahip olduğunu ve burada listede 5. Sırada bulunan şarkının ise en az şarkı süresine sahip olduğunu ilk 5 şarkı için sıralama azaldıkça şarkı süresinin azaldığını görebiliyoruz. Bu ilk 5 şarkı için geçerli olup listenin tamamına bakmamız gerektiği çıkarımını da göz ardı etmememiz gerekiyor.

# YILLARA GÖRE YOUTUBE VERİ ANALİZLERİ

Burada 2020 yılına ait Youtube’da en çok izlenen ilk 5 videonun yorum sayısı ve izlenme sayısı ile ilgili çıkarım yapmak için matplotlib kütüphanesi kullanarak scatter with text ile görselleştirdim.

# x = np.arange(len(data5['viewCount'][:5]))

x = data5['viewCount'][:5]

y = data5['commentCount'][:5]

# Scatter plot with text

plt.scatter(x, y)

for i in range(len(x)):

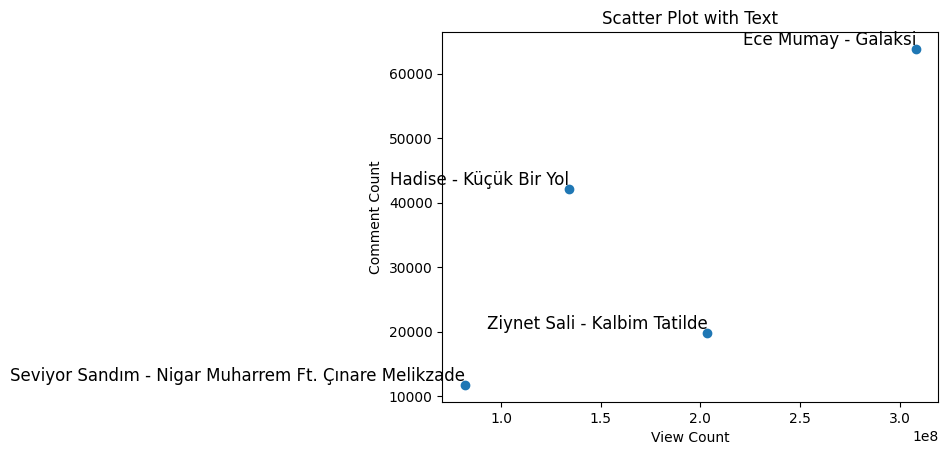
    plt.text(x[i], y[i], data5['title'][i], fontsize=12, ha='right', va='bottom')

plt.title('Scatter Plot with Text')

plt.xlabel('View Count')

plt.ylabel('Comment Count')

plt.show()



**Tablo 12**

Burada en çok izlenen ilk 5 videonun başlıklarıyla beraber izlenme sayılarının (x\*100milyon) yorumlar üzerindeki etkisine bakacak olursak genellikle izlenme sayısı artınca yorum sayıları da artmış bulunmaktadır. Tümüne yorum yapmak için aldığımız değer sayısını arttırmak daha iyi olacaktır. Ben grafiği daha temiz sunmak amacı ile ilk 5 videoyu aldım. Siz kodda değişikliğe giderek sayıyı arttırıp tam sonuca ulaşabilirsiniz.

Burada 2022 ve 2023 yılına ait Youtube’da en çok izlenen ilk 50 videosu ile ilgili izlenme sayısının beğeni sayı üzerine etkisi ile ilgili çıkarım yapmak için matplotlib kütüphanesi kullanarak

Fill between ile görselleştirdim.

x = data7['viewCount']

y1 = data7['likeCount']

y2 = data8['likeCount']

plt.fill\_between(x, y1, y2, color='lightblue', where=(y1 > y2), interpolate=True)

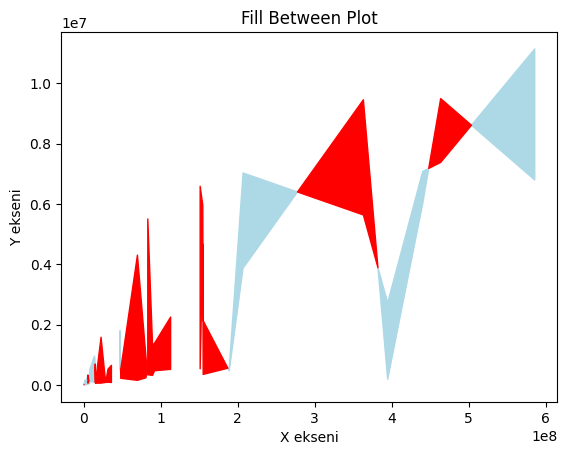
plt.fill\_between(x, y1, y2, color='red', where=(y1 <= y2), interpolate=True)

plt.title('Fill Between Plot')

plt.xlabel('X ekseni')

plt.ylabel('Y ekseni')

plt.show()



**Tablo 13**

Burada 2022 ve 2023 yılına ait Youtube’daki en çok izlenen ilk 50 videonun izlenme sayısı ile beğenme ve yorum sayısını karşılaştırıcı bir değer aralığı grafiği ile yorumluyoruz. Burada y1(2022 beğeni sayısı) y2(2023 beğeni sayısı) değerleri olmak üzere y1 ve y2 aralarındaki değerlerden büyük olanların kullanımıyla oluşturulmuştur. Yorum olarak izlenme sayısı arttıkça yorum sayısının da genel olarak arttığı söylenebiliyor.

Burada 2020, 2021, 2022 ve 2023 yılına ait Youtube’da en çok izlenen ilk 50 videosu ile ilgili izlenme sayısının popülerlik üzerine etkisi ile ilgili çıkarım yapmak için matplotlib kütüphanesi kullanarak

Boxplot ile görselleştirdim.

data = [data5['viewCount'], data6['viewCount'], data7['viewCount'], data8['viewCount']]

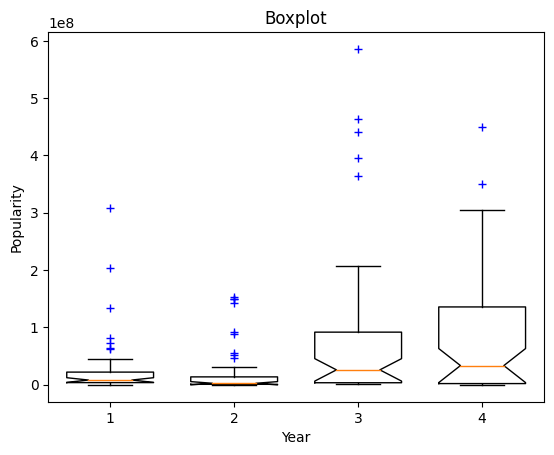
plt.boxplot(data, notch=True, sym='b+', widths=0.7)

plt.title('Boxplot')

plt.xlabel('Year')

plt.ylabel('Popularity')

plt.show()



**Tablo 14**

Burada 2020, 2021, 2022 ve 2023 yıllarına ait Youtube’da en çok izlenen ilk 50 videodaki izlenmelerin popülerliği ve medyanları ile ilgili ve izlenmenin popülerlik üzerindeki etkisini bu grafik ile çıkarım yaparak şu yoruma varabiliriz. 2023 yılı diğer yıllardan daha fazla izleyici sayısına ulaşmış ve daha popüler olmuş aynı yorumu medyan (ortanca değer) için de yapabilirim. Ortanca değerleri de yıllar arttıkça artış gösteriyor. Ayrıca 2020 yılı izlenmeleri 2021 yılı izlenmelerinden daha fazla popülerlik sağlamış 2021 yılı harici diğer yıllar arttıkça izlenme sayılarına bağlı popülerlik artış gösteriyor.

Burada 2020, 2021, 2022 ve 2023 yılına ait Youtube’da en çok izlenen ilk 10 videosunun açıklama kısmındaki karakter sayısının videolardaki izlenme sayısına etkisi ile ilgili çıkarım yapmak için matplotlib kütüphanesi kullanarak errorbar ile görselleştirdim.

datas= [data5, data6, data7, data8]

colors = ['red', 'blue', "green", "yellow"]

count= 0

for data in datas:

  x=np.arange(len(data['description'][:10]))

  y = data['viewCount'][:10]

  yerr = np.random.rand(len(data['title'][:10])) \* 10

  plt.errorbar(x,y, yerr=yerr, fmt='o', ecolor='r', capsize=5, color=colors[count])

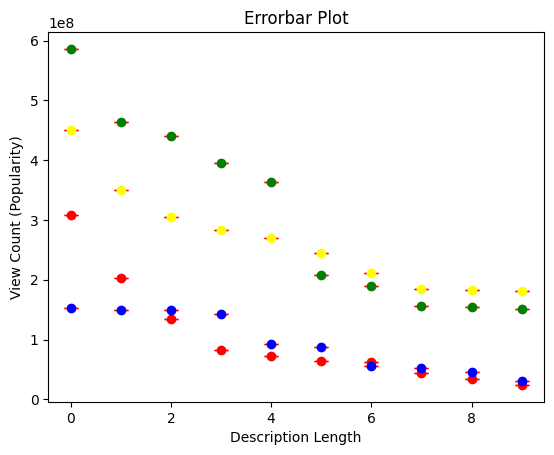
  count+=1

plt.title('Errorbar Plot')

plt.xlabel('Description Length')

plt.ylabel('View Count (Popularity)')

plt.show()



**Tablo 15**

Burada izlenme sayısı ile Açıklama kısmındaki yazı uzunluğu arasında ters orantı var. Çok izlenen videoların açıklama kısmındaki karakter sayısı az olup izlenme sayısı arttıkça açıklama kısmındaki karakter sayısı da azalıyor. Youtube’da çok izlenmek için açıklama kısmını kısa tutmalıyız.

Burada 2020, 2021, 2022 ve 2023 yılına ait Youtube’da en çok izlenen ilk 50 videonun yorum sayısı ile ilgili çıkarım yapmak için matplotlib kütüphanesi kullanarak hist ile görselleştirdim.

data = [data5['commentCount'], data6["commentCount"], data7['commentCount'], data8['commentCount']]

colors = ['red', 'blue', "green", "yellow"]

# Histogram plot

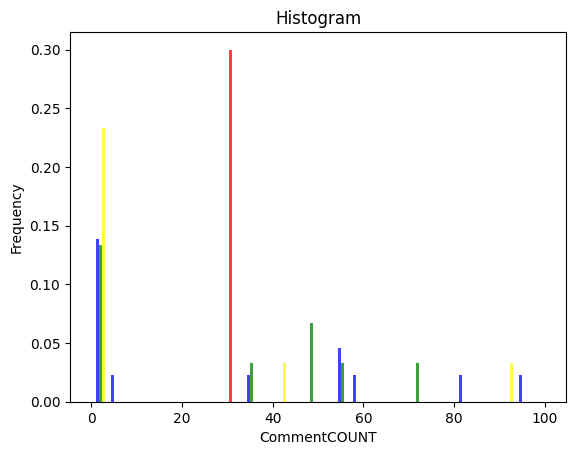
plt.hist(data, bins=30, range=(0, 100), density=True, color=colors, alpha=0.75)

plt.title('Histogram')

plt.xlabel('CommentCOUNT')

plt.ylabel('Frequency')

plt.show()



**Tablo 16**

Burada yıllara göre yorum sayılarının hangi frekanslarda olduğunu görüyoruz. Burada 2019 yılından sonra yorum kapatma özelliği geldiği için anlamlı çok yorum yapamıyoruz. En çok izlenen videolarda yıllar arttıkça yorum sayısı 0 üzerinde yoğunlaşmış yani günümüze geldiğimizde çok izlenen videolarda yorumlar devredışı bırakılıyor. Ayrıca 2020 yılında en çok izlenen videolarda neredeyse hiç kapatılmadığı çıkarımına ulaşıyoruz.

Burada 2020, 2021, 2022 ve 2023 yılına ait Youtube’da en çok izlenen ilk 50 videonun categoryId’leri ile ilgili çıkarım yapmak için matplotlib kütüphanesi kullanarak violinplot ile görselleştirdim.

data = [data5['categoryId'], data6["categoryId"], data7['categoryId'], data8['categoryId']]

# Violin plot

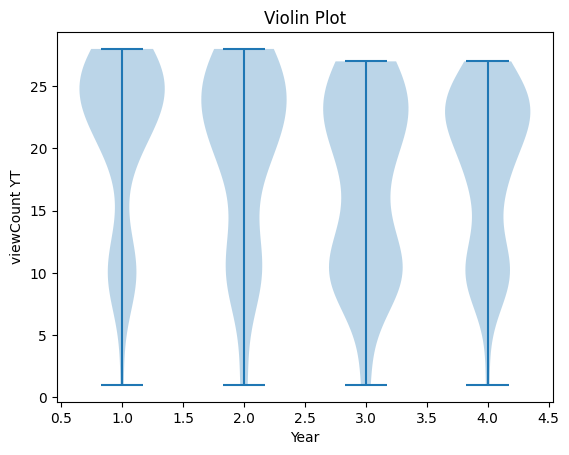
plt.violinplot(data, positions=[1, 2, 3, 4], widths=0.7, vert=True)

plt.title('Violin Plot')

plt.xlabel('Year')

plt.ylabel('viewCount YT')

plt.show()



**Tablo 17**

Burada en çok izlenen videolarla ilgili hangi kategori üzerinde yığılım yaptığı çıkarımına varabiliriz.

2020 ve 2021 yılında 25 - News & Politics kategorisinde yığılım olduğunu görebiliyoruz. Diğer yıllarda ise 25 - News & Politics kategorisinin yanında 20 – Gaming ve 10 – Music kategorilerinde yığılmalar artmış ve o yönlere kaymış durumda. Tüm kategorileri ve IDleri görebilmek için github sayfamda hangi id hangi kategoriye ait paylaştım. Detaylıca inceleyebilirsiniz.

[YoutubeSpotifyStatsVirtualization/datasets/youtubeCategoriesId.txt at main · bymayfe/YoutubeSpotifyStatsVirtualization (github.com)](https://github.com/bymayfe/YoutubeSpotifyStatsVirtualization/blob/main/datasets/youtubeCategoriesId.txt)

Burada tüm Youtube yıl verilerini birleştirdim iso formatındaki videoDuration verimi isodate modülü ile milisaniyeye dönüştürdüm. Anlamlı bir grafik çıkarmak için normalize edip izlenme sayısı verisi ile analiz yapabilmek için matplotlib modülünün barbs grafiği ile görselleştirdim. Fakat yorum yapamadım.

data = pd.concat([data5, data6, data7, data8]) # Verileri birleştirme

def iso\_to\_ms(duration): # ISO 8601 formatından milisaniyelere dönüştürme

    duration\_obj = isodate.parse\_duration(duration)

    return duration\_obj.total\_seconds() \* 1000

data['videoDurationMS'] = data['videoDuration'].apply(iso\_to\_ms)

num\_points = 10  # Daha küçük bir numara kullanarak meshgrid boyutunu belirleme

x = np.linspace(0, 2 \* np.pi, num\_points)

y = np.linspace(0, 2 \* np.pi, num\_points)

X, Y = np.meshgrid(x, y)

# Özellikleri normalize etme

U = data['videoDurationMS'].values[:num\_points] / max(data['videoDurationMS'].values)

V = data['viewCount'].values[:num\_points] / max(data['viewCount'].values)

U = np.resize(U, (num\_points, num\_points))

V = np.resize(V, (num\_points, num\_points))

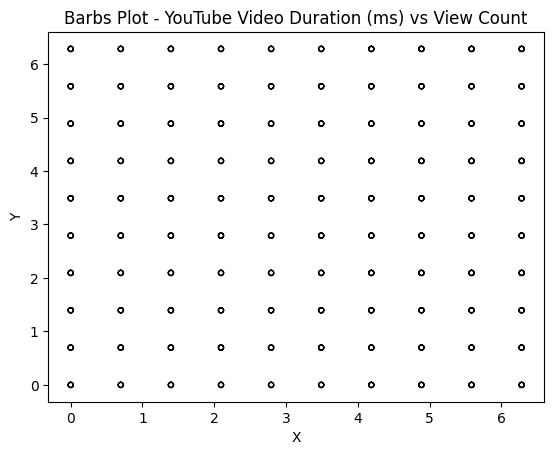
plt.barbs(X, Y, U, V, length=5, pivot='middle')

plt.title('Barbs Plot - YouTube Video Duration (ms) vs View Count')

plt.xlabel('X')

plt.ylabel('Y')

plt.show()



**Tablo 18**

Burada tüm youtube yıl verilerini birleştirdim iso formatındaki videoDuration verimi isodate modülü ile milisaniyeye dönüştürdüm. Video süresi verisi ile analiz yapabilmek için matplotlib modülünün evenplot ile görselleştirdim.

data = pd.concat([data5, data6, data7, data8])

def iso\_to\_ms(duration):

    duration\_obj = isodate.parse\_duration(duration)

    return duration\_obj.total\_seconds() \* 1000

data['videoDurationMS'] = data['videoDuration'].apply(iso\_to\_ms)

sinif = data["videoDurationMS"]

datalar = [sinif + i for i in range(len(sinif))]  # Farklı seriler oluşturma

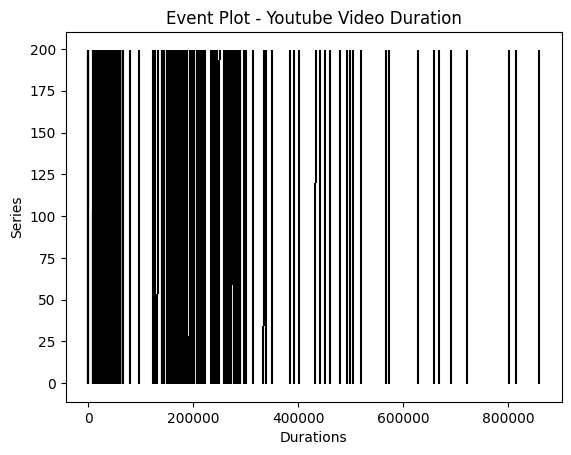
plt.eventplot(datalar, orientation='horizontal', lineoffsets=1, colors='black')

plt.title('Event Plot - Youtube Video Duration')

plt.xlabel('Durations')

plt.ylabel('Series')

plt.show()



**Tablo 19**

Burada tüm yıl değerlerine ait verilerin video süreleri ile ilgili yorum yapacak olursak. Video süreleri 160000 milisaniye (2.66 dakika) ile 300000 milisaniye (5 dakika) arasında yoğunluk göstermektedir. Genel olarak video süreleri bu milisaniye aralığında yer almaktadır. Ayrıca kısa uzunluktaki videolar da en çok izlenen videolar listesine girmeyi başarmıştır.

Kullandığım veri Quiver Plot için uygun olmadığından kendim yapay bir veri seti oluşturarak o tablo kullanımını göstermek istedim.

# Yapay veri oluştur

x = np.linspace(-2, 2, 10)

y = np.linspace(-2, 2, 10)

X, Y = np.meshgrid(x, y)

U = -Y

V = X

plt.figure()

plt.quiver(X, Y, U, V, units='xy', scale=1, color='blue')

plt.title('Quiver Plot Örneği')

plt.xlabel('X ekseni')

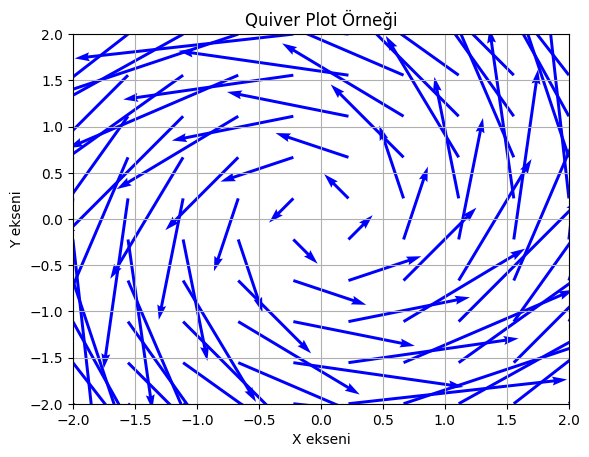
plt.ylabel('Y ekseni')

plt.xlim([-2, 2])

plt.ylim([-2, 2])

plt.grid()

plt.show()



**Tablo 20**

Burada tüm youtube yıl verilerini birleştirdim. İzlenme sayısı ve beğeni sayısı ile ilgili analiz yapabilmek için matplotlib modülünün figüre ve numpy modülünün polyfit fonksiyonu ile görselleştirdim.

data = pd.concat([data5, data6, data7, data8])

# Scatter plot of Views vs. Likes

plt.figure(figsize=(10, 6))

plt.scatter(data['viewCount'], data['likeCount'], color='skyblue', alpha=0.7, edgecolors='black')

plt.title('Relationship between Views and Likes')

plt.xlabel('Views')

plt.ylabel('Likes')

plt.grid(True, linestyle='--', alpha=0.7)

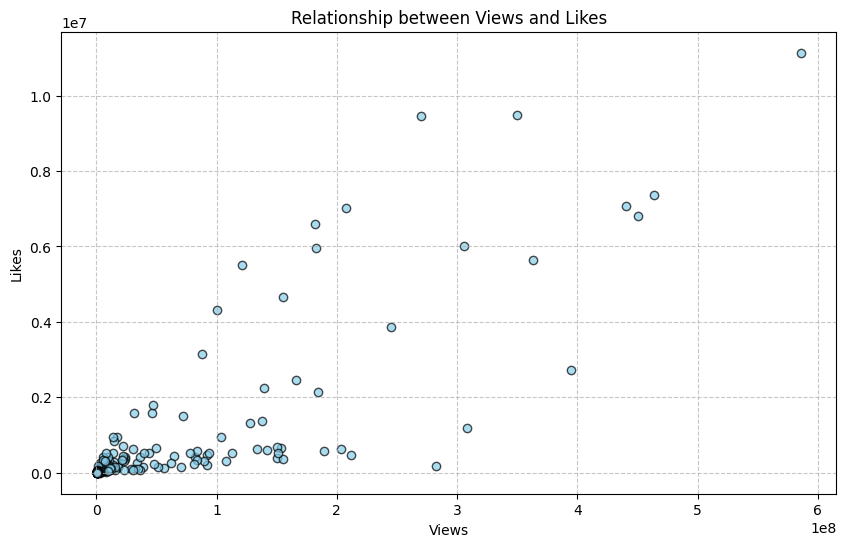
# Adding a trendline

z = np.polyfit(data['viewCount'], data['likeCount'], 1)

p = np.poly1d(z)

plt.plot(data['viewCount'],p(data['likeCount']), color="red")

plt.show()



Tablo 21

Burada Youtubedaki videoların izlenme ve beğeni sayısı ile ilgili bağıntıyı yorumlamak için bir grafik oluşturdum. İzlenme sayısı azaldıkça beğeni sayısı azalmış ve yoğunluk az kesimde. İzlenme arttıkça beğeni sayısı da artmış durumda. Ama yüksek kesimde yoğunluk az.

# KAYNAKÇA VE EKLER

[YouTube Data API  |  Google for Developers](https://developers.google.com/youtube/v3)

[Web API | Spotify for Developers](https://developer.spotify.com/documentation/web-api)

[Colab | Google Notebook Service](https://colab.research.google.com/)

**VERİYİ ELDE ETTİĞİM KODLAR**

Veriyi elde ettiğim kodlar javascript ve python dilinde olup çok fazla kod olduğu için bir github sayfası oluşturdum. Github sayfasında javascript klasörünün içinde veriyi elde edip düzenlediğim kodlar bulunmakta. (Javascript/main.js, Javascript/youtube.js, Javascript/spotify.js)

Ayrıca veriyi json halinde kaydedip sonrasında py koduyla csv dosyasına çevirdim.

(Javascript/main.js, Javascript/savejson.js) (Python/JSONtoCSV/main.py)

Buradaki kodlarla (Javascript/JSON) klasörünün içindeki json dosyalarını csv dosyasına dönüştürdüm.

Sonrasına hem üstte sunumda belirttiğim kodlarla üstteki tabloları hazırlayıp yorumladım hem de aşağıdaki github sayfasına colabdan hazırladığım kodları paylaştım. (Python/Colab/IST204Odev.ipynb)

[bymayfe/YoutubeSpotifyStatsVirtualization (github.com)](https://github.com/bymayfe/YoutubeSpotifyStatsVirtualization)