PROJE 2

PYBTCMP-10 / 10-Proje-Grubu-2

(**Dataset Linki**: https://www.kaggle.com/datasets/luiscorter/netflix-original-films-imdb-scores)

● Veri setine göre uzun soluklu filmler hangi dilde oluşturulmuştur? Görselleştirme yapınız.

● 2019 Ocak ile 2020 Haziran tarihleri arasında 'Documentary' türünde çekilmiş filmlerin IMDB değerlerini bulup görselleştiriniz.

● İngilizce çekilen filmler içerisinde hangi tür en yüksek IMDB puanına sahiptir?

● 'Hindi' Dilinde çekilmiş olan filmlerin ortalama 'runtime' suresi nedir?

● 'Genre' Sütunu kaç kategoriye sahiptir ve bu kategoriler nelerdir? Görselleştirerek ifade ediniz.

● Veri setinde bulunan filmlerde en çok kullanılan 3 dili bulunuz.

● IMDB puanı en yüksek olan ilk 10 film hangileridir?

● IMDB puanı ile 'Runtime' arasında nasıl bir korelasyon vardır? İnceleyip görselleştiriniz.

● IMDB Puanı en yüksek olan ilk 10 'Genre' hangileridir? Görselleştiriniz.

● 'Runtime' değeri en yüksek olan ilk 10 film hangileridir? Görselleştiriniz.

● Hangi yılda en fazla film yayımlanmıştır? Görselleştiriniz.

● Hangi dilde yayımlanan filmler en düşük ortalama IMBD puanına sahiptir? Görselleştiriniz.

● Hangi yılın toplam "runtime" süresi en fazladır?

● Her bir dilin en fazla kullanıldığı "Genre" nedir?

● Veri setinde outlier veri var mıdır? Açıklayınız.

PROJE 2

- Öncelikle projeye gerekli kütüphaneleri import ederek başladık.

```
In [2]: import pandas as pd
        import matplotlib.pyplot as plt
        import numpy as np
        from datetime import datetime
        import seaborn as sns
```

- Datasetimizi pd.read_csv() metodu ile "dataset" değişkenine atadık ve head() metodu ile veri setimizin kolon ve indekslerini inceledik.

```
In [3]: # Dataset okuma
        dataset = pd.read_csv("NetflixOriginals.csv", encoding="ISO-8859-1")
```

```
In [4]: dataset.head()
```

Out[4]:

	Title	Genre	Premiere	Runtime	IMDB Score	Language
0	Enter the Anime	Documentary	August 5, 2019	58	2.5	English/Japanese
1	Dark Forces	Thriller	August 21, 2020	81	2.6	Spanish
2	The App	Science fiction/Drama	December 26, 2019	79	2.6	Italian
3	The Open House	Horror thriller	January 19, 2018	94	3.2	English
4	Kaali Khuhi	Mystery	October 30, 2020	90	3.4	Hindi

- **Soru1:** Bu noktadan sonra projeye başlamaya hazırız. İlk sorumuz: "Veri setine göre uzun soluklu filmler hangi dilde oluşturulmuştur? Görselleştirme yapınız. "

  Görselleştirme için sns.barplot(x = None, y= None) metodunu kullandık. Burada x dillerin isimlerini, y ise bu isimlere karşılık gelen değerleri vermektedir.

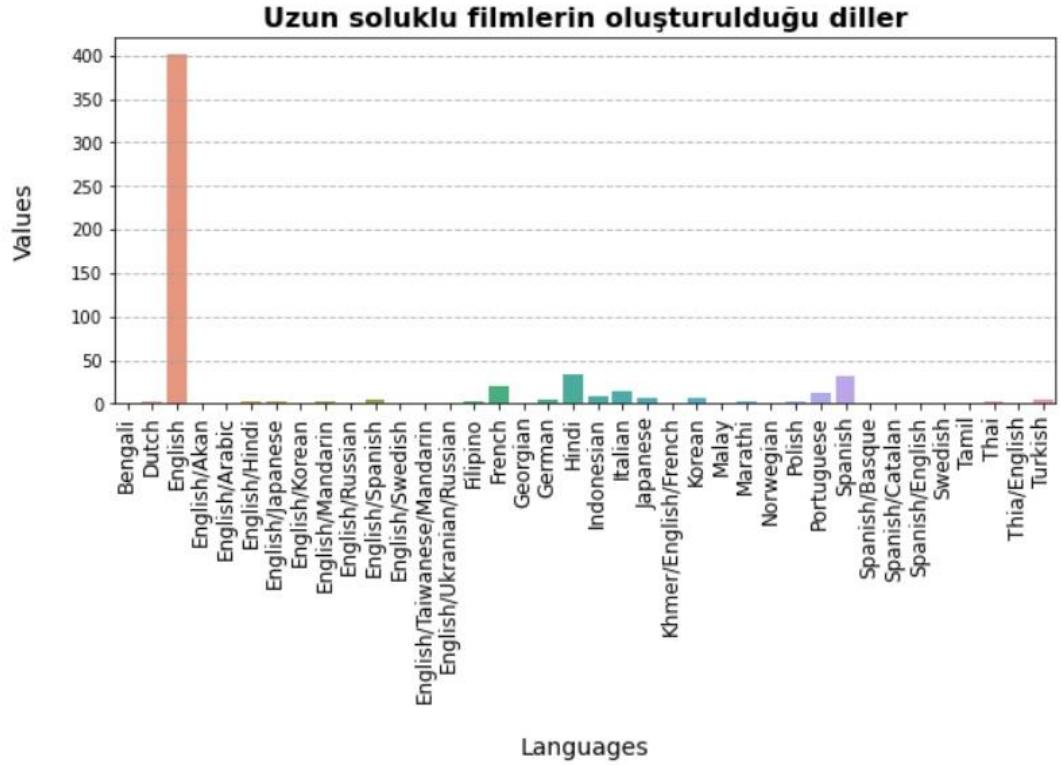
  dataset.groupby("Language").count()["Title"] metodu ile kaç farklı dil olduğuna ve karşılık gelen değerlerine ulaşabiliriz.

```
In [6]: dataset.groupby("Language").count()["Title"]
```

```
Out[6]: Language
        Bengali                      1
        Dutch                        3
        English                    401
        English/Akan                 1
        English/Arabic               1
        English/Hindi                2
        English/Japanese             2
        English/Korean               1
        English/Mandarin             2
        English/Russian              1
        English/Spanish              5
        English/Swedish              1
        English/Taiwanese/Mandarin   1
        English/Ukranian/Russian     1
        Filipino                     2
        French                      20
        Georgian                     1
```

Burada anahtar değerlere yani dil isimlerine ulaşmak için aynı metoda .keys() ekleyebiliriz. Bu anahtar değerler çıktımızdaki x'e denk gelmektedir.
y değerlerimiz için de bir döngü yazarak bu değerleri listeye atayabiliriz.

Sonuç olarak çıktımız aşağıdaki şekilde oluşur;



- **Soru2: "**2019 Ocak ile 2020 Haziran tarihleri arasında 'Documentary' türünde çekilmiş filmlerin IMDB değerlerini bulup görselleştiriniz."

Bu soruda hem 'documentary' hem de belli bir tarih aralığında filtreleme yapmamız gerekmektedir. Tarih için 'Premiere', documentary için 'Genre' kolonlarına erişmeliyiz.

'Premiere' kolonunun data type'ını kontrol ettiğimizde pandas.core.series.Series olduğunu görebiliriz. Bu type'ı datetime'a çevirmeliyiz. İkinci olarak da bizden istenen tarih aralığını pythonın datetime kütüphanesini kullanarak iki ayrı değişkene atayabiliriz.

```
In [11]: type(dataset["Premiere"])
Out[11]: pandas.core.series.Series

In [12]: # pandas.core.series.Series type'ını datetime'a çevirmek
         dataset["Premiere"] = pd.to_datetime(dataset["Premiere"])

In [13]: type(dataset["Premiere"][0])
Out[13]: pandas._libs.tslibs.timestamps.Timestamp

In [14]: # Başlangıç ve bitiş tarihlerini datetime ile değişkene atama

         start_date = datetime.strptime("January 1, 2019","%B %d, %Y" )
         end_date = datetime.strptime("June 1, 2020","%B %d, %Y" )
```

İstenilen veriyi aşağıdaki gibi filtreleyerek çıktısını alabiliriz:

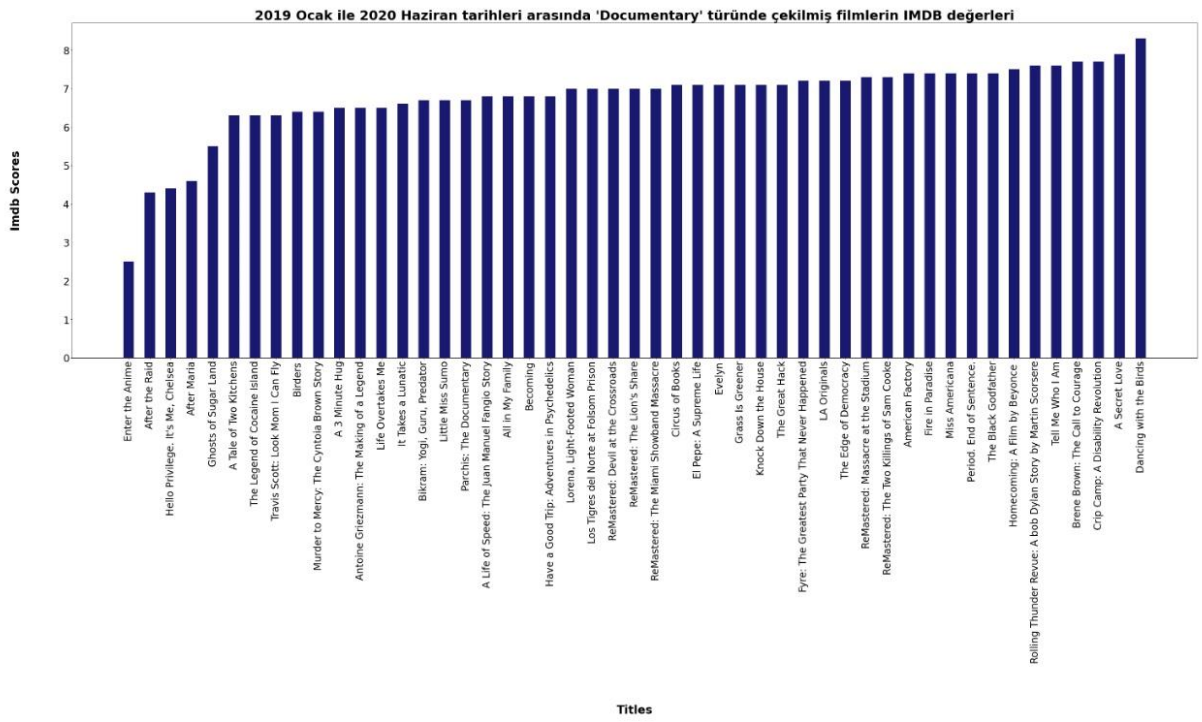
```
In [15]: # Soru 2 İstenilen veri:

soru2 = dataset[(dataset["Genre"] == "Documentary") & (dataset["Premiere"] > start_date) & (dataset["Premiere"] < end_date)]
```

```
In [16]: soru2
```

Out[16]:

	Title	Genre	Premiere	Runtime	IMDB Score	Language
0	Enter the Anime	Documentary	2019-08-05	58	2.5	English/Japanese
15	After the Raid	Documentary	2019-12-19	25	4.3	Spanish
20	Hello Privilege. It's Me, Chelsea	Documentary	2019-09-13	64	4.4	English
30	After Maria	Documentary	2019-05-24	37	4.6	English/Spanish
111	Ghosts of Sugar Land	Documentary	2019-10-16	21	5.5	English
263	A Tale of Two Kitchens	Documentary	2019-05-22	30	6.3	English/Spanish
286	The Legend of Cocaine Island	Documentary	2019-03-29	87	6.3	English
290	Travis Scott: Look Mom I Can Fly	Documentary	2019-08-28	85	6.3	English
295	Birders	Documentary	2019-09-25	37	6.4	English/Spanish
303	Murder to Mercy: The Cyntoia Brown Story	Documentary	2020-04-29	97	6.4	English
320	A 3 Minute Hug	Documentary	2019-10-28	28	6.5	English/Spanish
324	Antoine Griezmann: The Making of a Legend	Documentary	2019-03-21	60	6.5	French

Matplolib çıktısı:



- **Soru3:** **"**İngilizce çekilen filmler içerisinde hangi tür en yüksek IMDB puanına sahiptir? "

"Language" kolonundan 'English' diline sahip filmleri filtreleyerek ve max() metodunu kullanarak istenilen sonuca rahatlıkla ulaşabiliriz.

```
In [18]: ingilizce_filmler = dataset[(dataset["Language"] == "English")]
```

```
In [19]: max_imdb = ingilizce_filmler["IMDB Score"].max()
```

```
In [20]: ingilizce_filmler[(ingilizce_filmler["IMDB Score"] == max_imdb)]["Genre"]
```

```
Out[20]: 583    Documentary
         Name: Genre, dtype: object
```

- **Soru4: “**'Hindi' Dilinde çekilmiş olan filmlerin ortalama 'runtime' suresi nedir? “

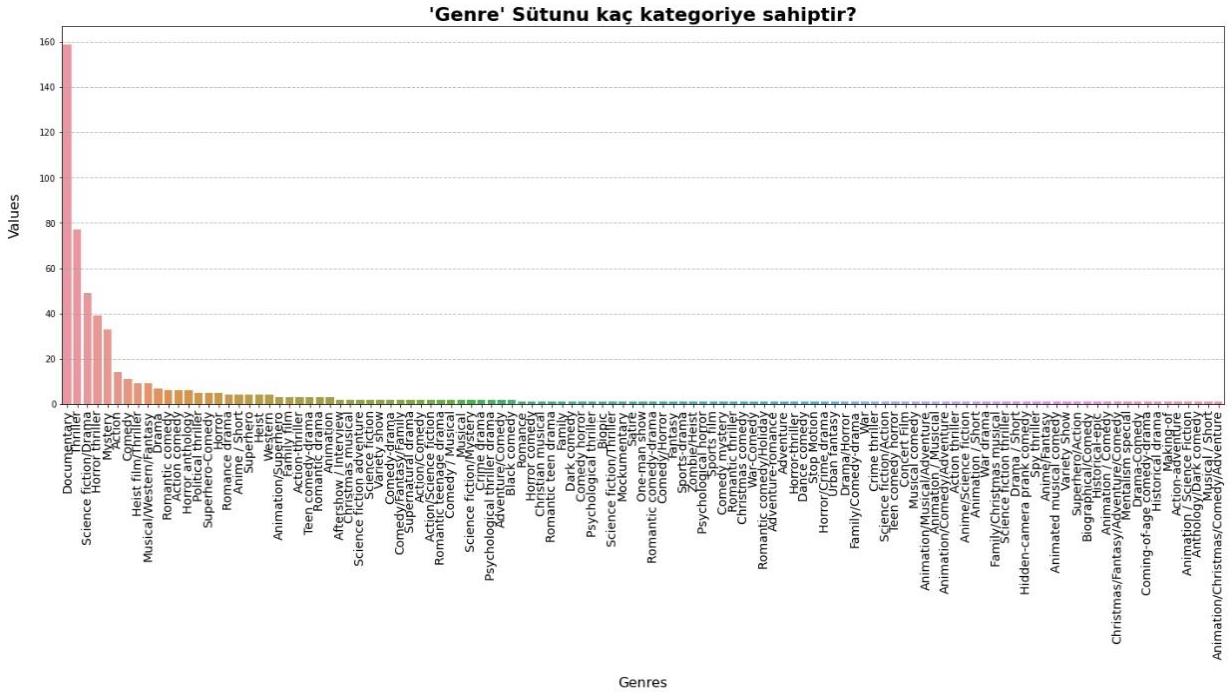
  Soru3’te olduğu gibi ‘Hindi’ dilinde çekilmiş filmleri bir değişkene atayarak sonrasında ‘runtime’ kolonundan mean() metodu ile ortalama ‘runtime’ süresine ulaşabiliriz.

```
In [21]: hindi_films = dataset[(dataset["Language"] == "Hindi")]

In [22]: # SONUÇ SORU 4

         hindi_films["Runtime"].mean()

Out[22]: 115.78787878787878
```

- **Soru5: “** 'Genre' Sütunu kaç kategoriye sahiptir ve bu kategoriler nelerdir? Görselleştirerek ifade ediniz. “

  ‘Genre’ kolonundan value_counts() metodu ile hem ‘Genre’ kategorilerine hem de bunların toplam sayısına ulaşabiliriz. Sonrasında değerleri bir listeye atayabiliriz.

  Grafiğin x değeri yani ‘Genre’ kategorilerine ulaşmak için; dataset["Genre"].unique() kullanılabilir. Y değerlerini ise oluşturduğumuz liste ile tanımlayabiliriz. Sonuç olarak çıktımız aşağıdaki gibi olur;



- **Soru6: “**Veri setinde bulunan filmlerde en çok kullanılan 3 dili bulunuz.”

  Dilleri sort_values() ile küçükten büyüğe sıralayabiliriz. En çok kullanılan üç değer için ise listenin sonuna tail(3) metodu ile erişebiliriz.

  Sonuç olarak en çok kullanılan 3 dil; English (401), Hindi(33) ve Spanish(31)’dir.

- **Soru7:** “IMDB puanı en yüksek olan ilk 10 film hangileridir?”

  Dataseti incelediğimizde zaten imdb puanlarına göre küçükten büyüğe doğru sıralandığını görebiliriz. Burada tek bir satır ile sonuca erişebiliriz;

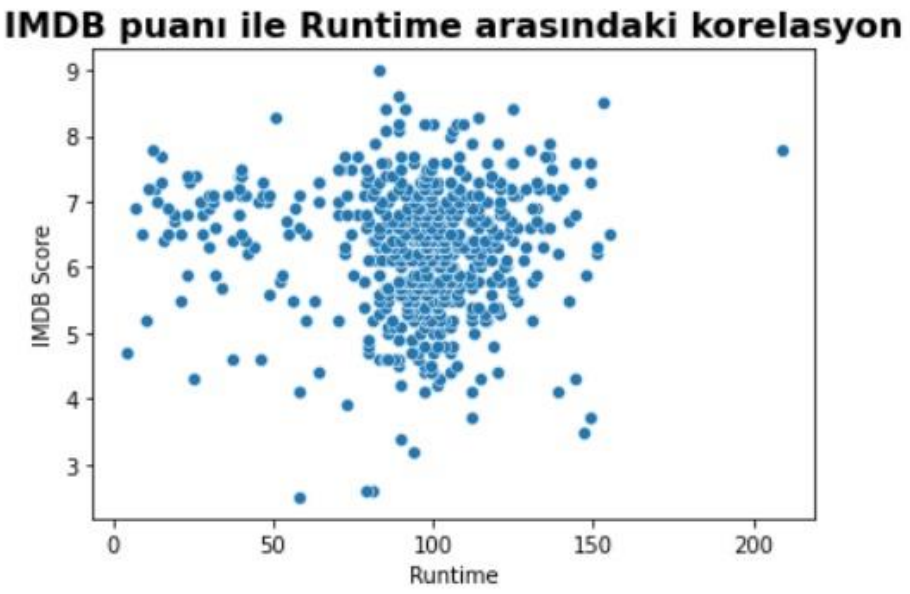
```
In [32]: dataset.sort_index(ascending = False ).head(10)
```

Out[32]:

	Title	Genre	Premiere	Runtime	IMDB Score	Language
583	David Attenborough: A Life on Our Planet	Documentary	2020-10-04	83	9.0	English
582	Emicida: AmarElo - It's All For Yesterday	Documentary	2020-12-08	89	8.6	Portuguese
581	Springsteen on Broadway	One-man show	2018-12-16	153	8.5	English
580	Winter on Fire: Ukraine's Fight for Freedom	Documentary	2015-10-09	91	8.4	English/Ukranian/Russian
579	Taylor Swift: Reputation Stadium Tour	Concert Film	2018-12-31	125	8.4	English
578	Ben Platt: Live from Radio City Music Hall	Concert Film	2020-05-20	85	8.4	English
577	Dancing with the Birds	Documentary	2019-10-23	51	8.3	English
576	Cuba and the Cameraman	Documentary	2017-11-24	114	8.3	English
575	The Three Deaths of Marisela Escobedo	Documentary	2020-10-14	109	8.2	Spanish
574	Seaspiracy	Documentary	2021-03-24	89	8.2	English

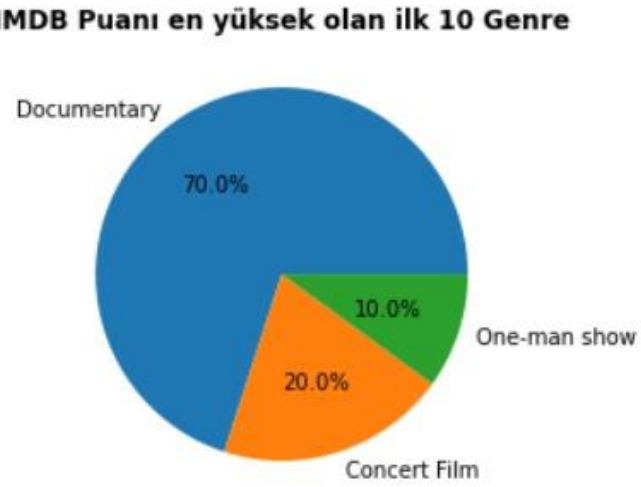
- **Soru8:** “IMDB puanı ile 'Runtime' arasında nasıl bir korelasyon vardır? İnceleyip görselleştiriniz.”

  İki kolon arasındaki korelasyon ilişkisini scatter çıktısı ile inceleyebiliriz;
  sns.scatterplot(data=dataset, x= "Runtime", y= "IMDB Score")



- **Soru9: “**IMDB Puanı en yüksek olan ilk 10 'Genre' hangileridir? Görselleştiriniz. “

  Önceki sorularda datasetin imdb puanlarına göre sıralı olduğundan bahsetmiştik. Bu soruda da tek satır ile istenilen veriye ulaşabiliriz;
  dataset.sort_index(ascending = False).head(10)["Genre"]
  Sonrasında dictionary ya da listeler yardımıyla pie chart çıktısı alınabilir:

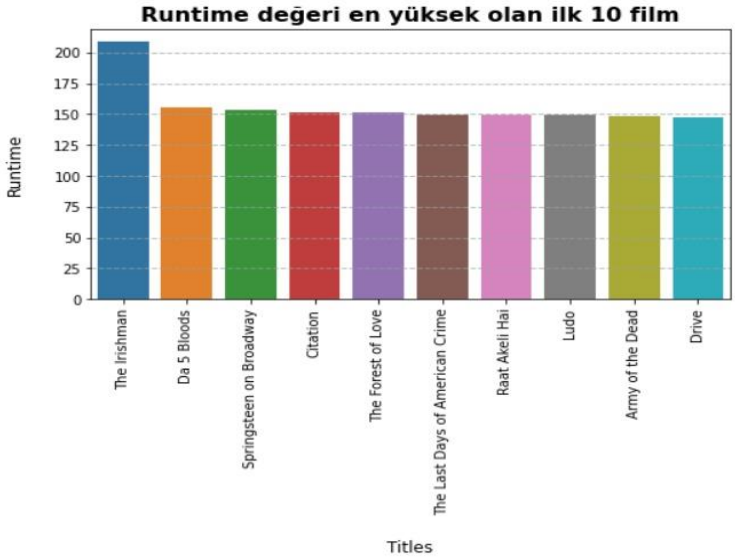


- **Soru10: “**'Runtime' değeri en yüksek olan ilk 10 film hangileridir? Görselleştiriniz.”

  dataset["Runtime"].sort_values(ascending = False).head(10) ile ‘runtime’ değeri en yüksek olan ilk 10 filme erişilebilir.
  ‘Title’ ve ‘Runtime’ iki ayrı listeye atılarak bar plot çıktısı alınabilir;

```
In [53]: top10_runtime = dataset["Runtime"].sort_values(ascending = False).head(10)
         top10_runtime = dict(top10_runtime)
```

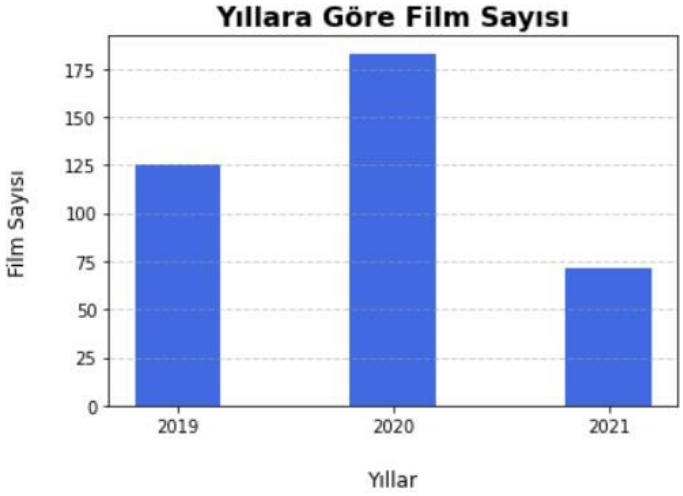
```
In [54]: top10_runtime_titles = list()
         top10_runtime_values = list()
         for value in top10_runtime.values():
             for j in range(len(dataset)):
                 if value == dataset["Runtime"][j]:
                     top10_runtime_titles.append(dataset["Title"][j])
                     top10_runtime_values.append(dataset["Runtime"][j])
```



- **Soru11: “**Hangi yılda en fazla film yayımlanmıştır? Görselleştiriniz. “

  Her yılı datetime kütüphanesini kullanarak değişkenlere atayabiliriz. Sonrasında da dataset[(dataset["Premiere"].dt.year == year2019.year)] ile filtreleyerek yıl üzerinden filmleri görebiliriz.

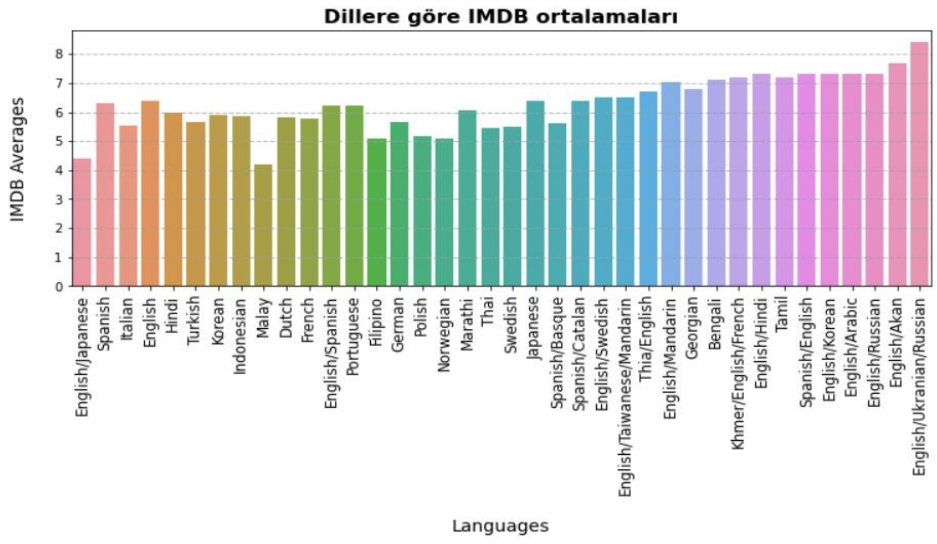
  .count() metodu ile istenilen yılara göre film sayılarına erişerek bunları görselleştirebiliriz.



- **Soru12: “**Hangi dilde yayımlanan filmler en düşük ortalama IMBD puanına sahiptir? Görselleştiriniz”

  Örneğin 0. İndeksteki yani ‘English/Japanese’ dili için imdb puanına şu satırı kullanarak ulaşabiliriz;
  dataset[(dataset["Language"] == dataset["Language"].unique()[0])]["IMDB Score"]
  Bu satıra mean() metodunu ekleyerek imdb ortalamasını rahatlıkla alabiliriz.

  Tek bir dil için imdb ortalamasını nasıl elde edebileceğimizi bulduk, bütün diller için bunu döngü yardımıyla yapabiliriz. Ve elde ettiğimiz verileri dictionary’e atarak görselleştirebiliriz:



- **Soru13:** “Hangi yılın toplam "runtime" süresi en fazladır?”

  11. soruda olduğu gibi datetime’ı kullanarak yılları değişkenlere atayabiliriz.

  Burada sum() metodunu kullanarak yıllara göre filtrelediğimiz datasetimizden toplam runtime değerlerini çekebiliriz;

  total_runtime2019 =
  dataset[(dataset["Premiere"].dt.year == year2019.year)].sum()["Runtime"]

  Buna göre en yüksek runtime süresine sahip yıl 2020’dir. Runtime süresi ise 17384’tür.

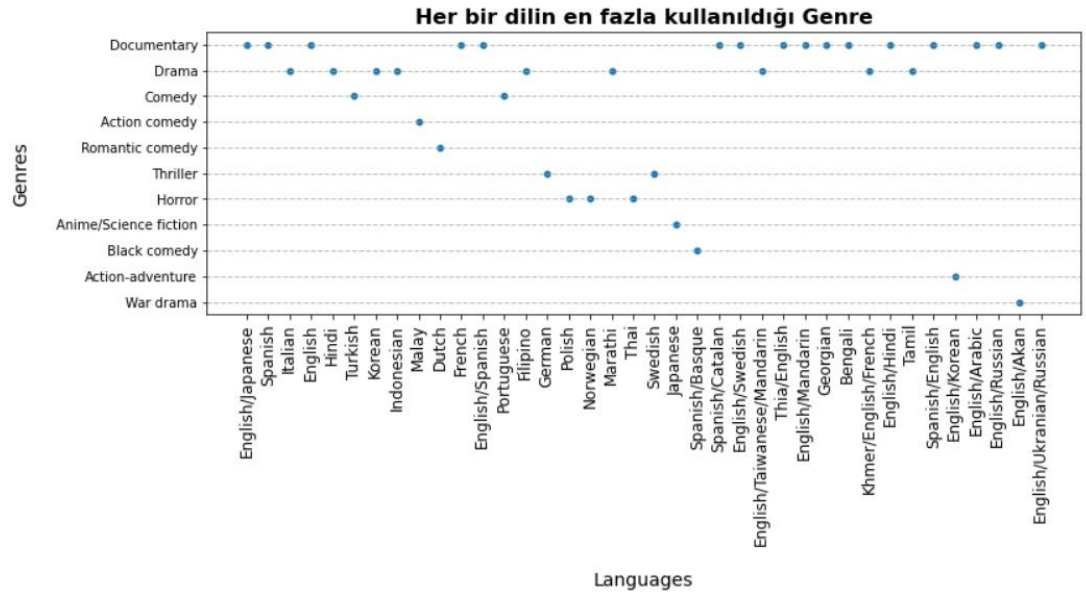
- **Soru14:** “Her bir dilin en fazla kullanıldığı "Genre" nedir? “

  Örneğin 4. İndekste bulunan dilin ‘Genre’ kategorilerine ve değerlerine erişmeye çalışalım;

```
In [79]: # 4. indexte bulunan dilin "Genre" değerleri

         dataset[(dataset["Language"] == dataset["Language"].unique()[4])]["Genre"].value_counts()
```

```
Out[79]: Drama                  13
         Thriller                5
         Comedy                  4
         Romantic comedy         3
         Comedy-drama            2
         Mystery                 1
         Action                  1
         Horror anthology        1
         Horror                  1
         Crime drama             1
         Anthology/Dark comedy   1
         Name: Genre, dtype: int64
```

İstediğimiz değerlere ulaşabildik, buradan en çok kullanılan ‘Genre’ye ulaşabilmek için filtrelemenin sonuna “ [0] “ ifadesini yerleştirebiliriz.
Bir dil için istenilen veriye nasıl ulaşabileceğimizi bulduk. Bütün diller için bunu yine döngü yardımıyla yapabiliriz. Sonuç olarak çıktımız şu şekilde oluşur;



- **Soru15:** "Veri setinde outlier veri var mıdır? Açıklayınız."

Aykırı değerler ortalamadan negatif veya pozitif yönde üç veya daha fazla standart uzaklıktadır. Diğer adıyla 3 sigma ilkesini ele alıp tüm imdb değerlerinin z skorunu hesaplayarak aykırı değer tespiti yapabiliriz.

$$Z = \frac{data - \mu}{\sigma}$$

Burada μ ortalamayı, σ standart sapmayı belirtmektedir.

mean() ve std() metodları ile imdb puanlarının ortalamalarına ve standart sapmalarına rahatlıkla ulaşabiliriz.

Döngü yardımıyla Z skorlarını hesaplayarak bir listeye atayabiliriz.

Daha sonra abs() >= 3 filtrelemesi ile, 3 sigma ilkesini if metoduyla ele alabiliriz.

Burada filtrelemeyi sağlayan değerlerin indekslerini bir listeye atayabiliriz. Ve bu indeksleri datasetimizden çekerek aykırı değerlerin tespitini yapabiliriz.

Bu durumda aykırı imdb değerlerimiz; 2.5, 2.6, 2.6 ve 3.2 olur.

In [97]:
```
# Z score'ları listeye atıyoruz.

Z_score = list()

for i in range(len(dataset["IMDB Score"])):
    Z_score.append((dataset["IMDB Score"][i] - dataset["IMDB Score"].mean()) / (dataset["IMDB Score"].std()))
```

In [98]:
```
# 3 sigma ilkesinden yola çıkarak mutlak değeri 3 değerinden büyük olan verileri outliner olarak kabul edebiliriz.

Z_score_index = list()

for i in range(len(Z_score)):
    if abs(Z_score[i]) >= 3:
        Z_score_index.append(i)
```

In [99]:
```
Z_score_index
```

Out[99]:
```
[0, 1, 2, 3]
```

In [100]:
```
print("Outlier IMDB Değerleri: ")

for j in range(len(Z_score_index)):
    print(dataset["IMDB Score"][Z_score_index[j]])
```

```
Outlier IMDB Değerleri:
2.5
2.6
2.6
3.2
```