İTÜ Büyük Veri ve İş Analitiği Sertifika Programı Final Projesi

GOOGLE PLAY STORE DATA ANALYSIS

Gamze GEDİK

31.03.2024

ABOUT ME

https://www.linkedin.com/in/gamze-gedik-38a223254/

mailto:gmzgdk.g@gmail.com

CONTENTS

GOOGLE PLAY STORE APPS DATA ANALYSIS

OBJECTIVE	PROBLEM
A data analysis will be conducted on the dataset of Google Play Store Apps. You can access the application via cable. The purpose of the application is to perform data analysis on the Google Play Store dataset. Through this analysis, information will be obtained about various features and performances of applications, as well as their relationships with each other. Subsequently, these insights will be analyzed and evaluated.	We know that application developers strive to gain more views and downloads on the Play Store by focusing on popular categories. This analysis will shed light on the necessity for application developers to develop strategies to attract more attention and expand their user base in a competitive environment.

DATA

10841-OBSERVATIONS
13-FEATURES

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19M	10,000+	Free	0	Everyone	Art & Design	January 7, 2018	1.0.0	4.0.3 and up
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14M	500,000+	Free	0	Everyone	Art & Design;Pretend Play	January 15, 2018	2.0.0	4.0.3 and up
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8.7M	5,000,000+	Free	0	Everyone	Art & Design	August 1, 2018	1.2.4	4.0.3 and up
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25M	50,000,000+	Free	0	Teen	Art & Design	June 8, 2018	Varies with device	4.2 and up
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2.8M	100,000+	Free	0	Everyone	Art & Design;Creativity	June 20, 2018	1.1	4.4 and up

GOOGLE PLAY STORE DATA ANALYSIS

Veriye Kaggle üzerinden ulaşılmıştır.10841 satır ve 13 sütundan oluşmaktadır.1.sütün App uygulamalarının adı, 2.sutun kategorisi, 3.sütün reytingi(puanlaması) ,4.sütun yorumlar, 5.sütün boyutu, 6.sütün indirme sayısı, 7.sütün veri tipi (ücretsiz ya da ücretli), 8.sütün fiyat bilgisi, 9 sütün derecelendirme-hangi yaş grubuna hitap ettiğini belirtiyor,10.sütun türü11.sütun son indirme tarihleri,12.sütun en güncel veriyonları,13.sütun Android versiyonlarına uygunluğu nu içermektedir. Öncelikle verimizin içeriğini anlıyoruz. Tüm sütunların karşılaştırmasını nasıl yapacağımızı göreceğiz ve birbirleri ile ilişkisi var mı, analiz edeceğiz.

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import plotly
import plotly.graph_objects as go
import warnings
warnings.filterwarnings("ignore", category=FutureWarning)
pd.options.mode.chained_assignment = None
import numpy as np
import matplotlib.pyplot as plt
```

Bu kodlar bir veri analizi ve görselleştirme kütüphaneleri olan Pandas, NumPy, Seaborn, Matplotlib ve Plotly'yi içe aktarıyor. Satır satır açıklayalım:

import pandas as pd: Pandas kütüphanesini içe aktarır ve kısaltma olarak pd kullanır.
import numpy as np: NumPy kütüphanesini içe aktarır ve kısaltma olarak np kullanır.
import seaborn as sns: Seaborn kütüphanesini içe aktarır ve kısaltma olarak sns kullanır.
import matplotlib.pyplot as plt: Matplotlib kütüphanesinden pyplot modülünü içe aktarır ve kısaltma olarak plt kullanır.
%matplotlib inline: Jupyter Notebook veya Jupyter Lab gibi ortamlarda Matplotlib grafiklerinin doğrudan görüntülenmesini sağlayan bir sihirli komuttur.
import plotly: Plotly kütüphanesini içe aktarır.
import plotly.graph_objects as go: Plotly'nin grafik nesnelerini içe aktarır ve kısaltma olarak go kullanır.
import warnings: Python'ın uyarıları yönetmesine izin veren warnings kütüphanesini içe aktarır.
warnings.filterwarnings("ignore", category=FutureWarning): Gelecekteki uyarıları görmezden gelmek için bir filtre tanımlar. Bu, gelecekteki uyarıları ekranda göstermemek için kullanılır.
pd.options.mode.chained_assignment = None: Pandas'ın "chained assignment" uyarılarını devre dışı bırakır. Bu, bazı atama operasyonlarında uyarı mesajlarının gösterilmesini engeller.
import numpy as np: NumPy kütüphanesini tekrar içe aktarır (muhtemelen gereksiz bir tekrardır).
import matplotlib.pyplot as plt: Matplotlib kütüphanesini tekrar içe aktarır (muhtemelen gereksiz bir tekrardır).

```
In [6]: total=data.isnull().sum()

In [7]: total

Out[7]: App                0
        Category           0
        Rating          1474
        Reviews            0
        Size               0
        Installs           0
        Type               1
        Price              0
        Content Rating     1
        Genres             0
        Last Updated       0
        Current Ver        8
        Android Ver        3
        dtype: int64

In [8]: total=data.isnull().sum().sort_values(ascending=False)
        percent=(data.isnull().sum()/data.isnull().count()).sort_values(ascending=False)
        missing_data=pd.concat([total,percent],axis=1,keys=['Total','Percent'])
        missing_data.head(6)
```

Out[8]:

	Total	Percent
Rating	1474	0.135965
Current Ver	8	0.000738
Android Ver	3	0.000277
Type	1	0.000092
Content Rating	1	0.000092
App	0	0.000000

```
In [9]: data.dropna(how='any',inplace=True)

In [10]: data.shape
Out[10]: (9360, 13)
```

Bu kod, veri setindeki eksik değerleri hesaplar ve bu eksik değerlerin toplam sayısını ve yüzdesini hesaplar. Ayrıca, eksik değerlerin sütunlara göre dağılımını incelemek için bir özet oluşturur. İşte satır satır açıklaması:

total=data.isnull().sum().sort_values(ascending=False): data adlı veri çerçevesindeki eksik değerlerin toplam sayısını hesaplar ve her sütun için bu sayıları azalan sırada sıralar. Yani, her sütundaki eksik değer sayısını içeren bir Seri nesnesi olan total oluşturulur.

percent=(data.isnull().sum()/data.isnull().count()).sort_values(ascending=False): Her sütundaki eksik değerlerin yüzdesini hesaplar. Yani, her sütundaki eksik değerlerin toplam sayısının, o sütundaki toplam veri sayısına oranını alır ve bunları azalan sırada sıralar. Bu bilgileri içeren bir Seri nesnesi olan percent oluşturulur.

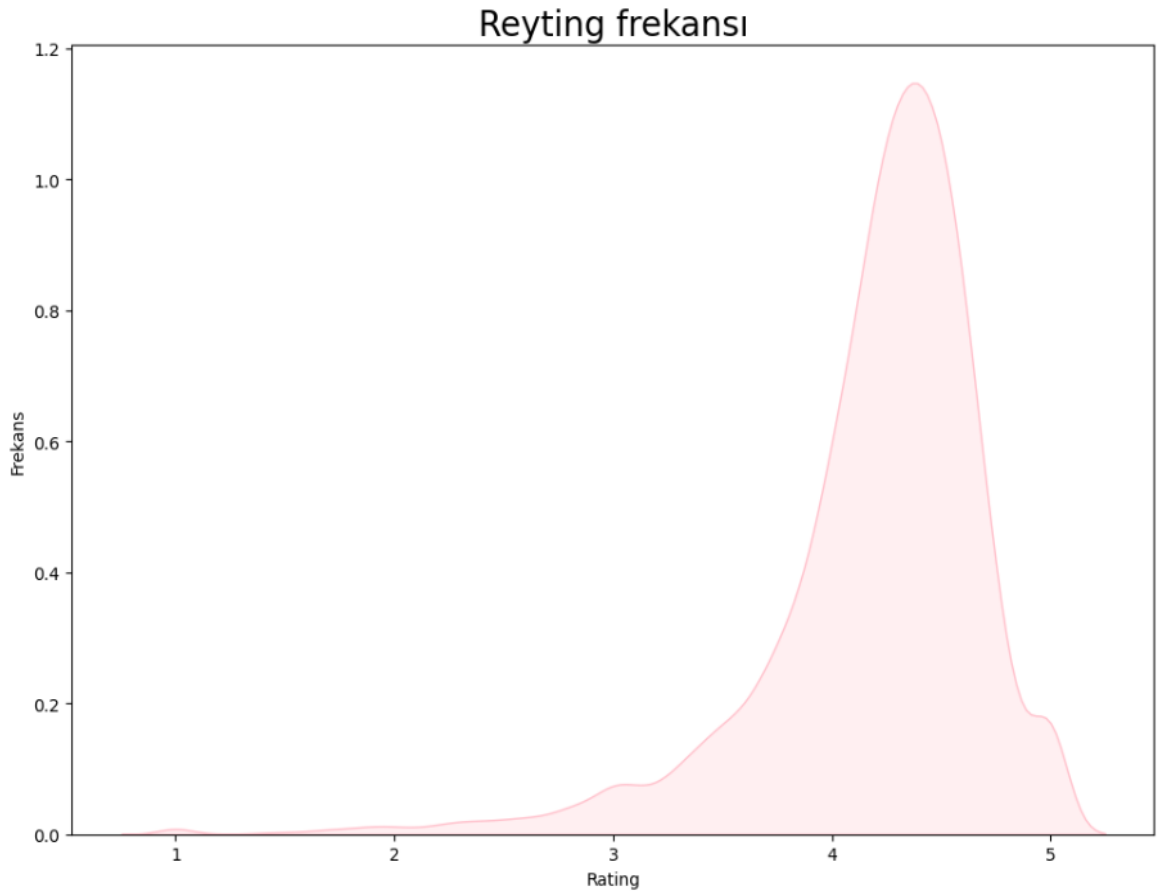
missing_data=pd.concat([total,percent],axis=1,keys=['Total','Percent']): total ve percent Seri nesnelerini yan yana birleştirir ve Total ve Percent başlıklarını kullanarak birleştirilmiş veri çerçevesi oluşturur. Bu çerçevede, her sütunun başlığı Total ve Percent olarak belirtilmiştir.

missing_data.head(6): Oluşturulan eksik veri özetinin ilk altı satırını görüntüler. Bu, en yüksek eksik değerlere sahip altı sütunu görmenizi sağlar.

Örneğin diyebiliriz ki reytingler kısmında 1474 veri eksikmiş Bu da toplamın yüzde13'üne yaklaşık karşılık geliyormuş 8 veri Android versiyonunda 3B taytta ve content reytingde birer veri eksikmiş Bu da çok çok düşük bir Orana karşılık geliyormuş. Bu eksik verilerin olduğu satırları dropna ile atacağız. Ve sonuç Data 9360 satıra düştü.

```
In [11]: import matplotlib.pyplot as plt
         plt.rcParams['figure.figsize']=11.6,8.25
         g=sns.kdeplot(data.Rating,color='Pink',shade=True)
         g.set_xlabel('Rating')
         g.set_ylabel('Frekans')
         plt.title('Reyting frekansı',size =20)

Out[11]: Text(0.5, 1.0, 'Reyting frekansı')
```

Out[11]: Text(0.5, 1.0, 'Reyting frekansı')



Kod Yorumları: Bu kod, seaborn ve matplotlib kütüphanelerini kullanarak bir yoğunluk grafiği (kdeplot) çizmek için kullanılır. İşte satır satır açıklamaları:

import matplotlib.pyplot as plt: matplotlib.pyplot modülünü plt adı altında içe aktarır. Bu modül, grafik çizmek ve özelleştirmek için kullanılır.

plt.rcParams['figure.figsize']=11.6,8.25: Bu satır, çizilen grafiklerin varsayılan boyutunu ayarlar. Burada, (genişlik, yükseklik) değerleri 11.6 ve 8.25 olarak ayarlanmıştır.

g=sns.kdeplot(data.Rating,color='Pink',shade=True): Seaborn kütüphanesinin kdeplot() işlevini kullanarak bir yoğunluk grafiği oluşturur. Grafikte kullanılacak veri, data adlı bir veri çerçevesindeki Rating sütunundan alınır. color='Pink' argümanı, grafiğin rengini pembe olarak ayarlar. shade=True argümanı, grafiğin altındaki alanı boyar, böylece yoğunluk grafiği daha belirgin hale gelir.

g.set_xlabel('Rating') ve g.set_ylabel('Frekans'): X ve Y eksenlerinin etiketlerini ayarlar. X ekseni etiketi 'Rating' olarak ayarlanırken, Y ekseni etiketi 'Frekans' olarak ayarlanır.

plt.title('Reyting frekansı',size =20): Grafik başlığını 'Reyting frekansı' olarak ayarlar. size=20 argümanı, başlık yazısının boyutunu 20 piksel olarak ayarlar.

Bu kod, veri setindeki 'Rating' sütununun yoğunluk dağılımını pembe renkte gösteren bir grafik oluşturur.

Grafik Yorumu: Grafikte bir basıklığın olduğunu görüyoruz. Genel olarak 4 ile 5 arasındaki frekanslarda yoğunluk vardır. Yani oylamanın genellikle 4 ile 5'e yakın aralıkta puanlandığını yorumlayabiliriz.

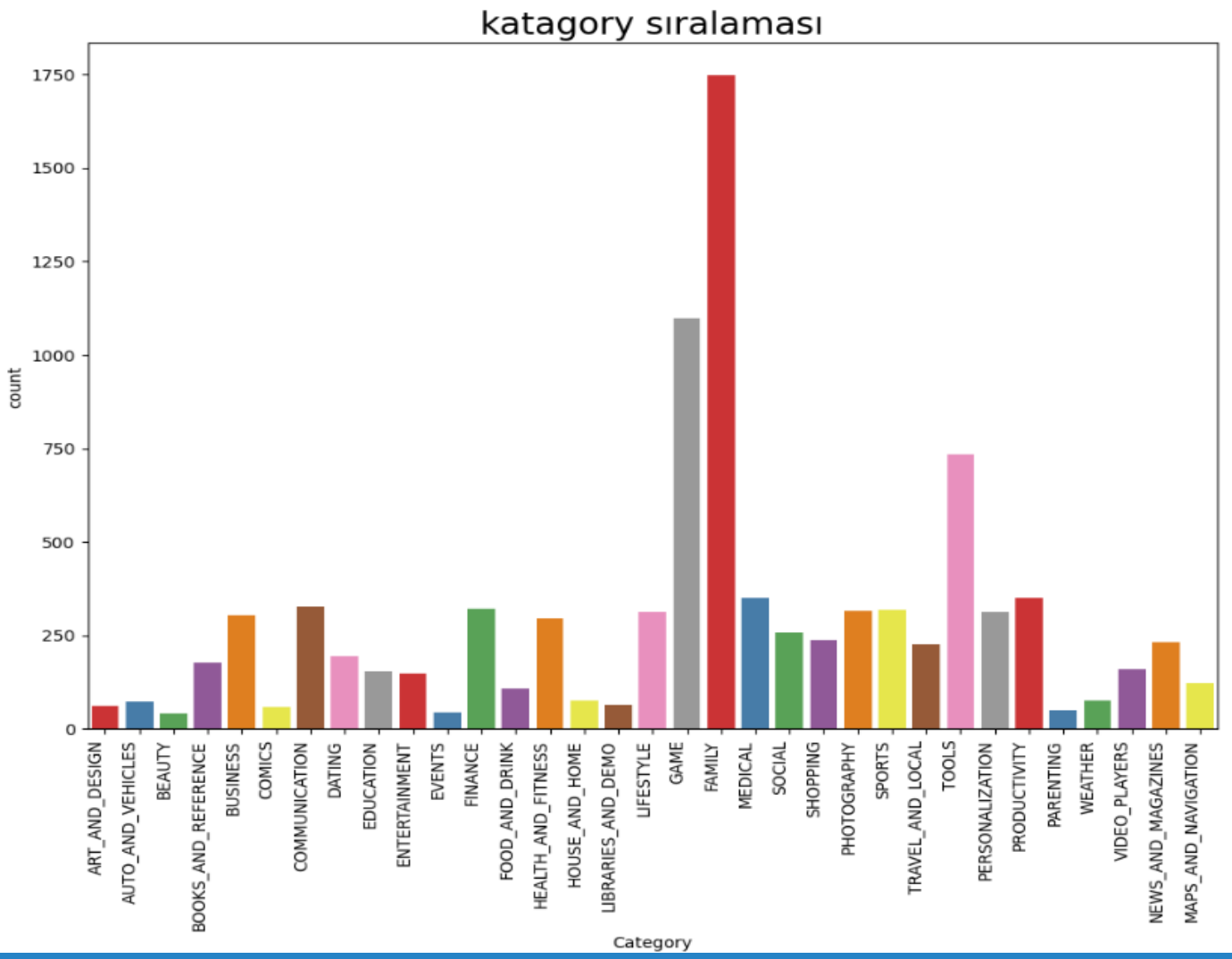
```
In [12]: len(data['Category'].unique()), 'categories'

Out[12]: (33, 'categories')

In [13]: data['Category'].unique()

Out[13]: array(['ART_AND_DESIGN', 'AUTO_AND_VEHICLES', 'BEAUTY',
                'BOOKS_AND_REFERENCE', 'BUSINESS', 'COMICS', 'COMMUNICATION',
                'DATING', 'EDUCATION', 'ENTERTAINMENT', 'EVENTS', 'FINANCE',
                'FOOD_AND_DRINK', 'HEALTH_AND_FITNESS', 'HOUSE_AND_HOME',
                'LIBRARIES_AND_DEMO', 'LIFESTYLE', 'GAME', 'FAMILY', 'MEDICAL',
                'SOCIAL', 'SHOPPING', 'PHOTOGRAPHY', 'SPORTS', 'TRAVEL_AND_LOCAL',
                'TOOLS', 'PERSONALIZATION', 'PRODUCTIVITY', 'PARENTING', 'WEATHER',
                'VIDEO_PLAYERS', 'NEWS_AND_MAGAZINES', 'MAPS_AND_NAVIGATION'],
               dtype=object)
```

Bu kod, bir veri çerçevesindeki 'Category' sütunundaki benzersiz kategorilerin sayısını hesaplar ve bir metin dizesi ile birlikte yazdırır. 33 adet kategori çıktısıdır.

Bu kod veri çerçevesindeki 'Category' sütunundaki her bir farklı kategori adını içeren bir dizi döndürür. Bu dizi, bir veri çerçevesinde bulunan tüm benzersiz kategori adlarını içerir.

```
In [14]: g=sns.countplot(x='Category',data=data,palette='Set1')
         g.set_xticklabels(g.get_xticklabels(),rotation=90,ha='right')
         g
         plt.title('katagory sıralaması',size=20)
```

```
Out[14]: Text(0.5, 1.0, 'katagory sıralaması')
```



Bu kod, seaborn kütüphanesini kullanarak bir kategorik sütunun (Category) görselleştirmesini oluşturur. İşte satır satır açıklamaları:

g=sns.countplot(x='Category',data=data,palette='Set1'): Bu satır, seaborn kütüphanesinin countplot() fonksiyonunu kullanarak 'Category' sütununun (x ekseni) sayısını gösteren bir çubuk grafiği oluşturur. Veri data veri çerçevesinden alınır ve renk paleti olarak 'Set1' kullanılır. Oluşturulan grafiği g adlı değişkene atar.

g.set_xticklabels(g.get_xticklabels(),rotation=90,ha='right'): Bu satır, x eksenindeki (kategori etiketlerindeki) metin etiketlerinin dönüşünü ve hizalamasını ayarlar. get_xticklabels() metodu mevcut x eksenindeki etiketleri alır. rotation=90 argümanı, etiketlerin 90 derece saat yönünde döndürülmesini sağlar, böylece dikey hale gelir. ha='right' argümanı, etiketlerin sağa hizalanmasını sağlar.

g: Bu ifade, oluşturulan çubuk grafiğini ekrana getirir.

plt.title('katagory sıralaması',size=20): Bu satır, grafik başlığını 'katagory sıralaması' olarak ayarlar ve başlığın boyutunu 20 piksel olarak belirler. Başlık, matplotlib kütüphanesinin title() fonksiyonuyla eklenir.
Bu kod, 'Category' sütununun (kategori sıralamasının) görselleştirmesini oluşturur ve kategori etiketlerini dikey olarak döndürür.

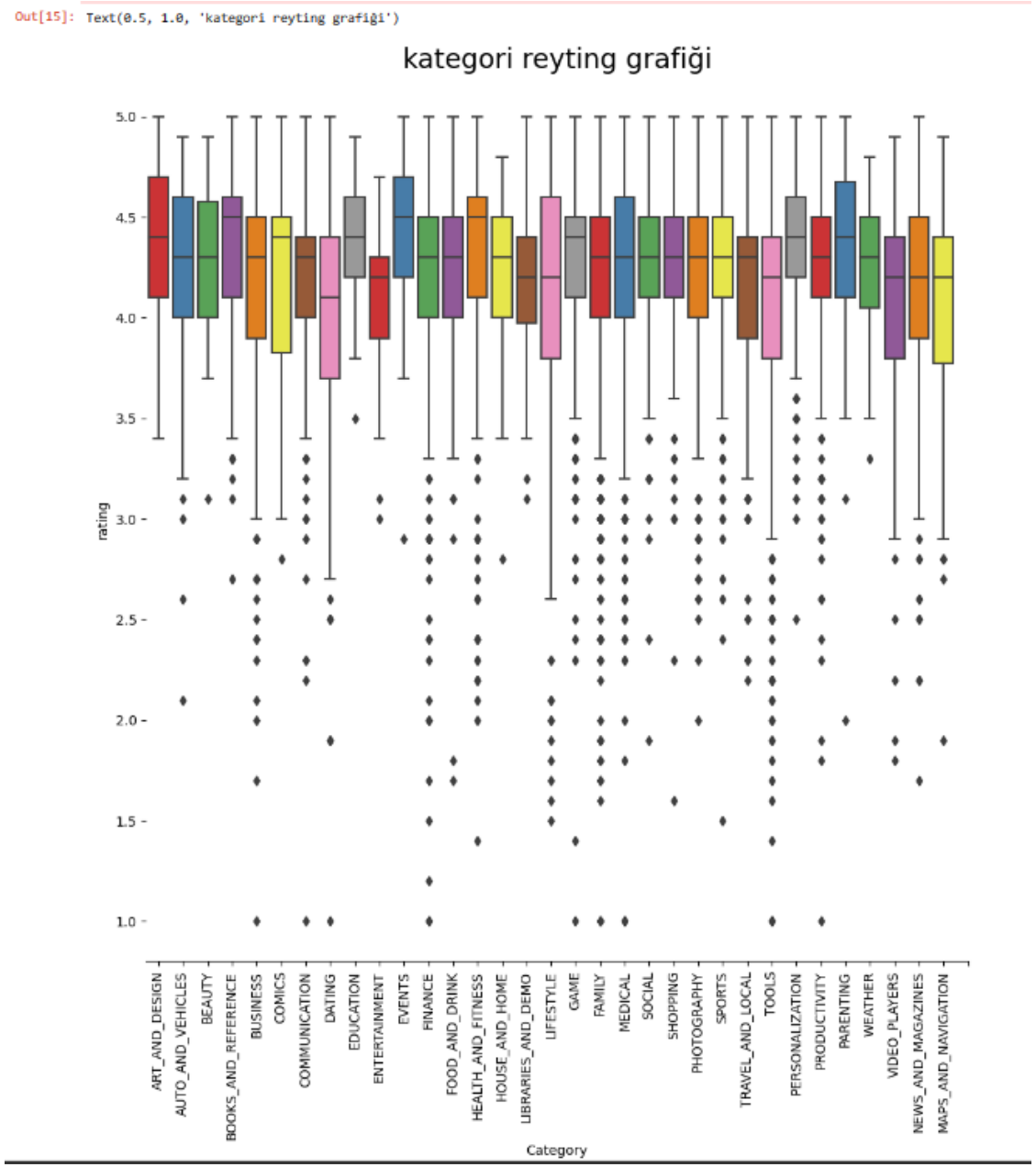
Grafik Yorumu: Grafikte toplam 33 farklı kategori olduğunu görüyoruz. Google Play Store'da çeşitli kategorilere ait frekansları görebilmekteyiz. Peki, bu bilgi işimize yarayacak mı? Görselleştirdiğimiz bu veriler, uygulama geliştiricilerinin Play Store'daki popüler kategorilere odaklanarak daha fazla görüntülenme ve indirme elde etmeye çalışabilecekleri anlamına gelmektedir. Yani, bu verilerin üzerinden giderek popüler kategorileri belirleyebilir ve bu kategorilere yönelik uygulamalar geliştirilebilir. Dolayısıyla, bu kategorilerin popüler olduğunu söyleyebiliriz

```
In [15]: g=sns.catplot(x='Category',y='Rating',data=data,kind='box',height=10,palette='Set1')
         g.despine(left=True)
         g.set_xticklabels(rotation=90)
         g.set(xticks=range(0,34))
         g=g.set_xlabels('Category')
         g=g.set_ylabels('rating')
         plt.title('kategori reyting grafiği',size=20)
```

Silikon Kütüphanesi kullanarak kategorilere göre uygulama derecelerini kutu grafiği ile görselleştirmesini sağladık.

Grafik Yorumu: Her kategorideki uygulama dereceleri arasında belirgin farklılıklar olmadığını söyleyebiliriz. Kutu grafik grafiğindeki kutuların boyutlarına ve yerleşimlerine bakarak bu tür bir çıkarım yapabiliriz. Kutuların çoğu benzer boyutta ve ortalamaları birbirine yakın görünüyor, ancak bazı kategorilerde birkaç uygulama derecesinin daha yüksek veya düşük olduğunu gözlemleyebiliriz. Bu sonuç, uygulama geliştiricilerinin bir uygulama kategorisi seçerken derecelerin çok farklı olup olmadığına daha fazla odaklanmaları gerekmeyebileceğini gösterebilir. Yani, reyting değerlerinin kategoriler arasında genel olarak benzer olduğunu söyleyebiliriz. Bu da, reyting değerlendirmelerinin (oylama) genellikle birbirine yakın olduğunu ve belirli bir kategoride büyük çapta değişmediğini gösterir.

```
Out[15]: Text(0.5, 1.0, 'kategori reyting grafiği')
```



```
In [16]: data['Reviews']. head()

Out[16]: 0       159
         1       967
         2     87510
         3    215644
         4       967
         Name: Reviews, dtype: object
```

```
In [17]: data['Reviews']=data['Reviews'].apply(lambda x: int(x))
```

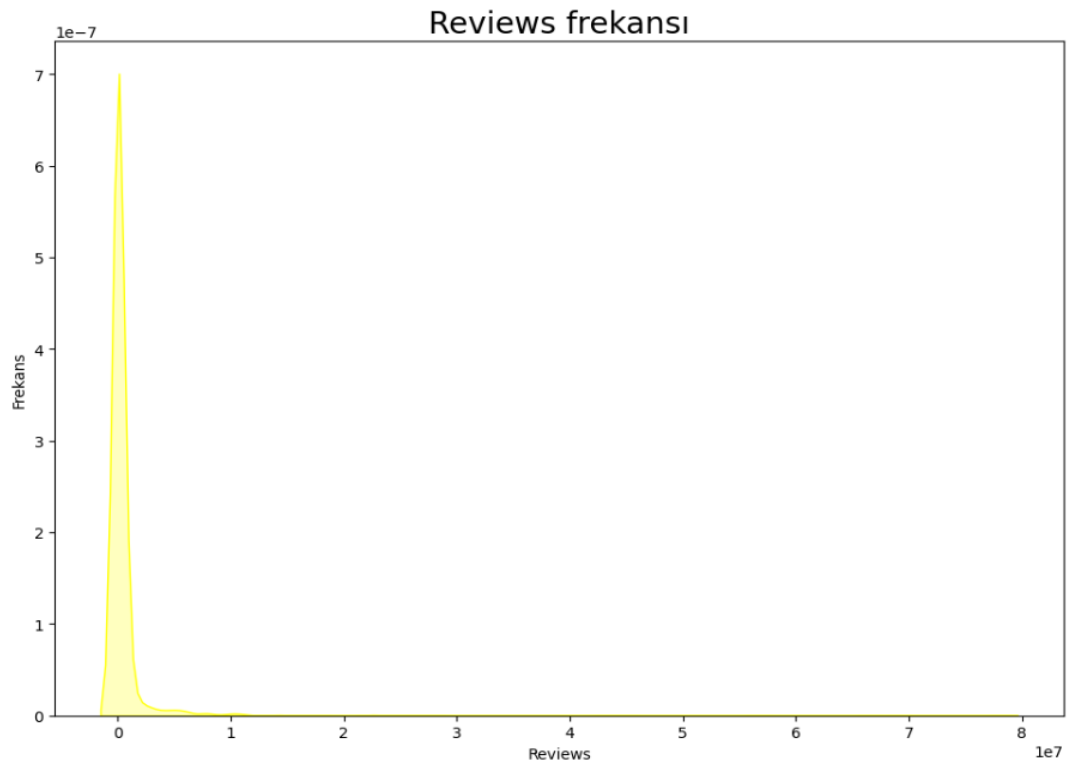
```
In [18]: data['Reviews'].head()

Out[18]: 0       159
         1       967
         2     87510
         3    215644
         4       967
         Name: Reviews, dtype: int64
```

Bu kod 'Reviews' sütunundaki ilk beş satırı gösterecektir. İlk 5 değeri görüntüledik buradaki veri tipine baktığımız zaman bunun aslında sayı olduğunu buradaki sayısal ifadeler gibi durduğunu ancak sayısal bir formatta olmadığını görüyoruz Bu nedenle bu sütunu sayısal bir formata dönüştürmemiz gereklidir. lambda fonksiyon ile object lerin hepsini int 64 formatına dönüştürdüğümüzü görüyoruz.

```
In [19]: plt.rcParams['figure.figsize']=11.6,8.25
         g=sns.kdeplot(data.Reviews,color='Yellow',shade=True)
         g.set_xlabel('Reviews')
         g.set_ylabel('Frekans')
         plt.title('Reviews frekansı',size =20)
```

Bu kod, seaborn ve matplotlib kütüphanelerini kullanarak bir yoğunluk grafiği (kdeplot) çizmek için kullanılmıştır.

```
Out[19]: Text(0.5, 1.0, 'Reviews frekansı')
```



Grafik Yorumu: 5 milyondan fazla yoruma sahip uygulamalara yapılan yorumun frekansını göstermiştir. bizim için bir şey ifade etmemektedir.

```
In [20]: data[data.Reviews>5000000].head(10)
```

Out[20]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver
335	Messenger – Text and Video Chat for Free	COMMUNICATION	4.0	56642847	Varies with device	1,000,000,000+	Free	0	Everyone	Communication	August 1, 2018	Varies with device	Varies with device
336	WhatsApp Messenger	COMMUNICATION	4.4	69119316	Varies with device	1,000,000,000+	Free	0	Everyone	Communication	August 3, 2018	Varies with device	Varies with device
338	Google Chrome: Fast & Secure	COMMUNICATION	4.3	9642995	Varies with device	1,000,000,000+	Free	0	Everyone	Communication	August 1, 2018	Varies with device	Varies with device
342	Viber Messenger	COMMUNICATION	4.3	11334799	Varies with device	500,000,000+	Free	0	Everyone	Communication	July 18, 2018	Varies with device	Varies with device
351	Opera Mini - fast web browser	COMMUNICATION	4.5	5149854	Varies with device	100,000,000+	Free	0	Everyone	Communication	July 19, 2018	Varies with device	Varies with device
365	WeChat	COMMUNICATION	4.2	5387333	Varies with device	100,000,000+	Free	0	Everyone	Communication	July 31, 2018	Varies with device	Varies with device
378	UC Browser - Fast Download Private & Secure	COMMUNICATION	4.5	17712922	40M	500,000,000+	Free	0	Teen	Communication	August 2, 2018	12.8.5.1121	4.0 and up
381	WhatsApp Messenger	COMMUNICATION	4.4	69119316	Varies with device	1,000,000,000+	Free	0	Everyone	Communication	August 3, 2018	Varies with device	Varies with device
382	Messenger – Text and Video Chat for Free	COMMUNICATION	4.0	56646578	Varies with device	1,000,000,000+	Free	0	Everyone	Communication	August 1, 2018	Varies with device	Varies with device
385	Viber Messenger	COMMUNICATION	4.3	11334973	Varies with device	500,000,000+	Free	0	Everyone	Communication	July 18, 2018	Varies with device	Varies with device

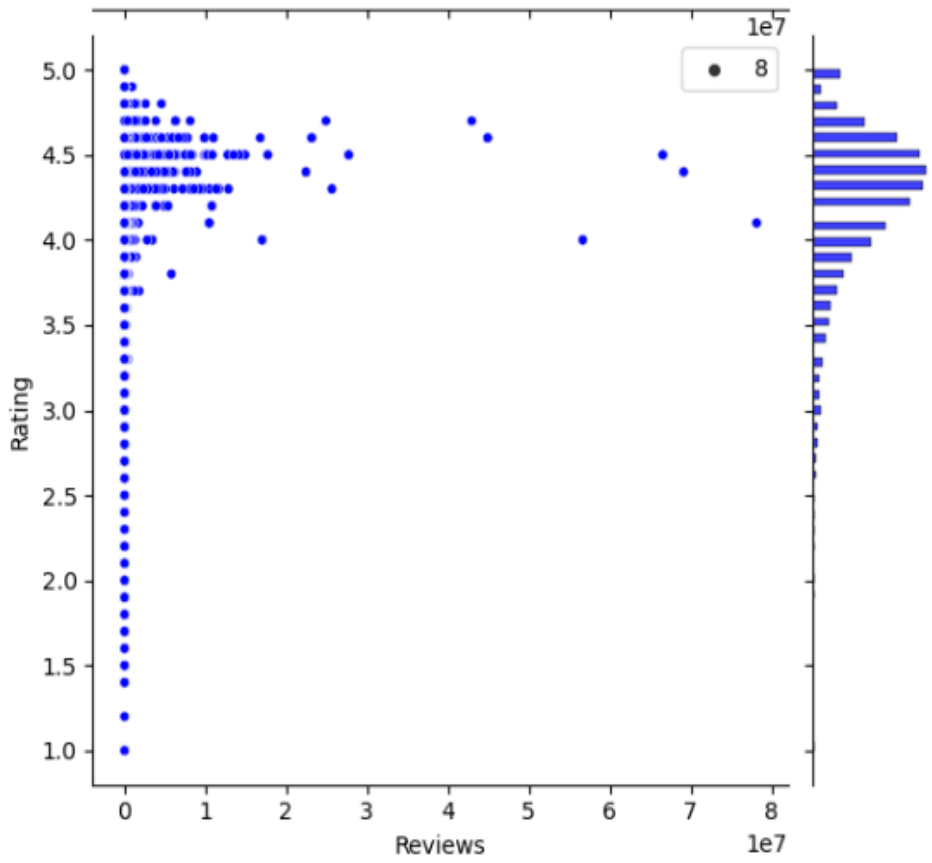
```
In [21]: len(data[data.Reviews>5000000])
Out[21]: 230
```

Bu kod, 'Reviews' sütununda 5.000.000'den büyük olan değerlere sahip olan ilk 10 satırı döndürür. Yani, veri çerçevesinde 'Reviews' sütununda 5.000.000'den büyük değerlere sahip olan gözlemleri filtreler ve bu gözlemleri döndürür.

'Reviews' sütununda 5.000.000'den büyük bir değere sahip olan gözlemlerin sayısını döndürür. Burada 230 tane uygulamanın 5 milyondan fazla verisi vardır diyebiliriz.

```
In [22]: plt.figure(figsize=(10,10))
         g=sns.jointplot(x='Reviews',y='Rating',color='Blue',data=data,size=8)
```

```
<Figure size 1000x1000 with 0 Axes>
```



Bu kod, seaborn kütüphanesini kullanarak 'Reviews' ve 'Rating' sütunları arasındaki ilişkiyi görselleştirmek için bir ekleme grafik (jointplot) oluşturur.

Grafik Yorumu: Burada oylama sayıları arttıkça reyting sayısının arttığını görüyoruz. Her bir veri noktasının nokta pilotu ile gösterdiği ve aynı zamanda her bir eksenin yoğunluğunu gösteren bir yoğunluk plotudur bu şekilde reviews ve rating sütunları arasındaki ilişkinin dağılımı ve yoğunluğu hakkında bilgi alabiliriz. Uygulamaların reytingleri arttıkça gördüğünüz gibi burada yorumlamaların da arttığını görüyoruz çok düşük puana sahip oyunlara nadiren çok düşük oylamalar yapılmıştır. Mesela 4 buçuk reyting bir uygulamaya ya da oyuna çok fazla yorum yapıldığını söyleyebiliriz.

```
In [23]: plt.figure(figsize=(10,10))
         sns.regplot(x='Reviews',y='Rating',color='red',data=data[data['Reviews']<1000000]);
         plt.title('rating reviews',size=20)
```

```
Out[23]: Text(0.5, 1.0, 'rating reviews')
```

Bu kod, seaborn kütüphanesini kullanarak bir regresyon çizimini oluşturur. 'Reviews' ve 'Rating' sütunları arasındaki ilişkiyi gösteren bir regresyon çizimi oluşturur. Bu çizimde, inceleme sayısı 1.000.000'den az olan uygulamaların verileri kullanılır.

Grafik Yorumu: Arasındaki ilişkinin doğrusal bir regresyon olabileceği ve daha yüksek yorum sayısı olan uygulamaların genellikle daha yüksek bir derecelendirme aldığı söylenebilir. Gerçekten de, reyting arttıkça yapılan oylamaların veya yorumların doğrusal bir şekilde arttığını gözlemleyebiliriz. Az önce yaptığımız açıklamaya istinaden, regresyon eğrisinin yukarı doğru hareket ettiğini görüyoruz. Pozitif bir eğilim olarak, daha yüksek yorum sayısına sahip uygulamalar genellikle daha yüksek bir derecelendirme alır. Ancak, bu ilişkinin ve uygulamalarla ilgisi olmayabilir. Yani, yüksek yorum sayısına sahip bir uygulama sadece belirli bir kitle tarafından sıkça kullanılıyor olabilir. Ancak, her zaman bu grafikte gösterilen regresyon eğrisi veya çok sayıda yoruma sahip olması popüler bir uygulama olduğu anlamına gelmez

```
In [24]: data['Size'].head ()
```

```
Out[24]: 0     19M
         1     14M
         2    8.7M
         3     25M
         4    2.8M
         Name: Size, dtype: object
```

```
In [25]: data['Size'].unique()
```

```
Out[25]: array(['19M', '14M', '8.7M', '25M', '2.8M', '5.6M', '29M', '33M', '3.1M',
                '28M', '12M', '20M', '21M', '37M', '5.5M', '17M', '39M', '31M',
                '4.2M', '23M', '6.0M', '6.1M', '4.6M', '9.2M', '5.2M', '11M',
                '24M', 'Varies with device', '9.4M', '15M', '10M', '1.2M', '26M',
                '8.0M', '7.9M', '56M', '57M', '35M', '54M', '201k', '3.6M', '5.7M',
                '8.6M', '2.4M', '27M', '2.7M', '2.5M', '7.0M', '16M', '3.4M',
                '8.9M', '3.9M', '2.9M', '38M', '32M', '5.4M', '18M', '1.1M',
                '2.2M', '4.5M', '9.8M', '52M', '9.0M', '6.7M', '30M', '2.6M',
                '7.1M', '22M', '6.4M', '3.2M', '8.2M', '4.9M', '9.5M', '5.0M',
                '5.9M', '13M', '73M', '6.8M', '3.5M', '4.0M', '2.3M', '2.1M',
                '42M', '9.1M', '55M', '23k', '7.3M', '6.5M', '1.5M', '7.5M', '51M',
                '41M', '48M', '8.5M', '46M', '8.3M', '4.3M', '4.7M', '3.3M', '40M',
                '7.8M', '8.8M', '6.6M', '5.1M', '61M', '66M', '79k', '8.4M',
                '3.7M', '118k', '44M', '695k', '1.6M', '6.2M', '53M', '1.4M',
                '3.0M', '7.2M', '5.8M', '3.8M', '9.6M', '45M', '63M', '49M', '77M',
                '4.4M', '70M', '9.3M', '8.1M', '36M', '6.9M', '7.4M', '84M', '97M',
                '2.0M', '1.9M', '1.8M', '5.3M', '47M', '556k', '526k', '76M',
                '7.6M', '59M', '9.7M', '78M', '72M', '43M', '7.7M', '6.3M', '334k',
                '93M', '65M', '79M', '100M', '58M', '50M', '68M', '64M', '34M',
                '67M', '60M', '94M', '9.9M', '232k', '99M', '624k', '95M', '8.5k',
                '41k', '292k', '80M', '1.7M', '10.0M', '74M', '62M', '69M', '75M',
                '98M', '85M', '82M', '96M', '87M', '71M', '86M', '91M', '81M',
                '92M', '83M', '88M', '704k', '862k', '899k', '378k', '4.8M',
                '266k', '375k', '1.3M', '975k', '980k', '4.1M', '89M', '696k',
                '544k', '525k', '920k', '779k', '853k', '720k', '713k', '772k',
                '318k', '58k', '241k', '196k', '857k', '51k', '953k', '865k',
                '251k', '930k', '540k', '313k', '746k', '203k', '26k', '314k',
                '239k', '371k', '220k', '730k', '756k', '91k', '293k', '17k',
                '74k', '14k', '317k', '78k', '924k', '818k', '81k', '939k', '169k',
                '45k', '965k', '90M', '545k', '61k', '283k', '655k', '714k', '93k',
                '872k', '121k', '322k', '976k', '206k', '954k', '444k', '717k',
                '210k', '609k', '308k', '306k', '175k', '350k', '383k', '454k',
                '1.0M', '70k', '812k', '442k', '842k', '417k', '412k', '459k',
                '478k', '335k', '782k', '721k', '430k', '429k', '192k', '460k',
                '728k', '496k', '816k', '414k', '506k', '887k', '613k', '778k',
                '683k', '592k', '186k', '840k', '647k', '373k', '437k', '598k',
                '716k', '585k', '982k', '219k', '55k', '323k', '691k', '511k',
                '951k', '963k', '25k', '554k', '351k', '27k', '82k', '208k',
                '551k', '29k', '103k', '116k', '153k', '209k', '499k', '173k',
                '597k', '809k', '122k', '411k', '400k', '801k', '787k', '50k',
                '643k', '986k', '516k', '837k', '780k', '20k', '498k', '600k',
                '656k', '221k', '228k', '176k', '34k', '259k', '164k', '458k',
                '629k', '28k', '288k', '775k', '785k', '636k', '916k', '994k',
                '309k', '485k', '914k', '903k', '608k', '500k', '54k', '562k',
                '847k', '948k', '811k', '270k', '48k', '523k', '784k', '280k',
                '24k', '892k', '154k', '18k', '33k', '860k', '364k', '387k',
                '626k', '161k', '879k', '39k', '170k', '141k', '160k', '144k',
                '143k', '190k', '376k', '193k', '473k', '246k', '73k', '253k',
                '957k', '420k', '72k', '404k', '470k', '226k', '240k', '89k',
                '234k', '257k', '861k', '467k', '676k', '552k', '582k', '619k'],
               dtype=object)
```

```
In [26]: len(data[data.Size=='Varies with device'])
```

```
Out[26]: 1637
```

İlk 5 veride farklı boyutlarda uygulamalar olduğunu görüyoruz. Veri tipi burada 'object'miş ve bu sütun örnek için metrik sembol içeriyor. Ayrıca, boyutla ilgili bazı veriler var. Bu verileri kaldırabiliriz. Verilerin hepsi kategorik türde. Veriler sayısal değil, string olarak atanmışlar. Çünkü önlerinde 'kilobayt' veya 'megabayt' gibi ifadeler var. Ayrıca 'veries with device' isminde özel boyutların olduğunu belirtiyor. Bu durum verileri doğru şekilde yorumlamamızı zorlaştırır. Bu nedenle, bu değerleri önce uygun bir şekilde işlememiz gerekecek. Buradaki ifade işletim sistemine bağlı olarak değişebileceği anlamına gelir. Bu, uygulamanın farklı cihazlarda ve işletim sistemlerinde çalışabilmesini sağlar. Ancak, bu durum boyut bilgisinin eksik veya yanıltıcı olmasına neden olabilir. Belirli bir formata uydurmamız lazım. 1637 tane verimiz varmış. Bu verileri silebiliriz ama o zaman veri kaybı olacaktır. Buradaki bu değerlerin Üzerinde işlem yapabilmemiz için 'number'a çevirmemiz gereklidir.

```
In [27]: data['Size'].replace('Varies with device',np.nan,inplace=True)
```

```
In [28]: data['Size'].unique()
```

```
Out[28]: array(['19M', '14M', '8.7M', '25M', '2.8M', '5.6M', '29M', '33M', '3.1M',
                '28M', '12M', '20M', '21M', '37M', '5.5M', '17M', '39M', '31M',
                '4.2M', '23M', '6.0M', '6.1M', '4.6M', '9.2M', '5.2M', '11M',
                '24M', nan, '9.4M', '15M', '10M', '1.2M', '26M', '8.0M', '7.9M',
                '56M', '57M', '35M', '54M', '201k', '3.6M', '5.7M', '8.6M', '2.4M',
                '27M', '2.7M', '2.5M', '7.0M', '16M', '3.4M', '8.9M', '3.9M',
                '2.9M', '38M', '32M', '5.4M', '18M', '1.1M', '2.2M', '4.5M',
                '9.8M', '52M', '9.0M', '6.7M', '30M', '2.6M', '7.1M', '22M',
                '6.4M', '3.2M', '8.2M', '4.9M', '9.5M', '5.0M', '5.9M', '13M',
                '73M', '6.8M', '3.5M', '4.0M', '2.3M', '2.1M', '42M', '9.1M',
                '55M', '23k', '7.3M', '6.5M', '1.5M', '7.5M', '51M', '41M', '48M',
                '8.5M', '46M', '8.3M', '4.3M', '4.7M', '3.3M', '40M', '7.8M',
                '8.8M', '6.6M', '5.1M', '61M', '66M', '79k', '8.4M', '3.7M',
                '118k', '44M', '695k', '1.6M', '6.2M', '53M', '1.4M', '3.0M',
                '7.2M', '5.8M', '3.8M', '9.6M', '45M', '63M', '49M', '77M', '4.4M',
                '70M', '9.3M', '8.1M', '36M', '6.9M', '7.4M', '84M', '97M', '2.0M',
                '1.9M', '1.8M', '5.3M', '47M', '556k', '526k', '76M', '7.6M',
                '59M', '9.7M', '78M', '72M', '43M', '7.7M', '6.3M', '334k', '93M',
                '65M', '79M', '100M', '58M', '50M', '68M', '64M', '34M', '67M',
                '60M', '94M', '9.9M', '232k', '99M', '624k', '95M', '8.5k', '41k',
                '292k', '80M', '1.7M', '10.0M', '74M', '62M', '69M', '75M', '98M',
```

```
In [29]: data.Size=(data.Size.replace(r'[kM]+$', '',regex=True).astype(float) * \
         data.Size.str.extract(r'[\d.]+([KM]+)',expand=False)
         .fillna(1)
         .replace(['k','M'],[10**3,10**6]).astype(int))

In [30]: data['Size'].unique()

Out[30]: array([1.90e+07, 1.40e+07, 8.70e+06, 2.50e+07, 2.80e+06, 5.60e+06,
                2.90e+07, 3.30e+07, 3.10e+06, 2.80e+07, 1.20e+07, 2.00e+07,
                2.10e+07, 3.70e+07, 5.50e+06, 1.70e+07, 3.90e+07, 3.10e+07,
                4.20e+06, 2.30e+07, 6.00e+06, 6.10e+06, 4.60e+06, 9.20e+06,
                5.20e+06, 1.10e+07, 2.40e+07,      nan, 9.40e+06, 1.50e+07,
                1.00e+07, 1.20e+06, 2.60e+07, 8.00e+06, 7.90e+06, 5.60e+07,
                5.70e+07, 3.50e+07, 5.40e+07, 2.01e+02, 3.60e+06, 5.70e+06,
                8.60e+06, 2.40e+06, 2.70e+07, 2.70e+06, 2.50e+06, 7.00e+06,
                1.60e+07, 3.40e+06, 8.90e+06, 3.90e+06, 2.90e+06, 3.80e+07,
                3.20e+07, 5.40e+06, 1.80e+07, 1.10e+06, 2.20e+06, 4.50e+06,
                9.80e+06, 5.20e+07, 9.00e+06, 6.70e+06, 3.00e+07, 2.60e+06,
                7.10e+06, 2.20e+07, 6.40e+06, 3.20e+06, 8.20e+06, 4.90e+06,
                9.50e+06, 5.00e+06, 5.90e+06, 1.30e+07, 7.30e+07, 6.80e+06,
                3.50e+06, 4.00e+06, 2.30e+06, 2.10e+06, 4.20e+07, 9.10e+06,
                5.50e+07, 2.30e+01, 7.30e+06, 6.50e+06, 1.50e+06, 7.50e+06,
                5.10e+07, 4.10e+07, 4.80e+07, 8.50e+06, 4.60e+07, 8.30e+06,
                4.30e+06, 4.70e+06, 3.30e+06, 4.00e+07, 7.80e+06, 8.80e+06,
                6.60e+06, 5.10e+06, 6.10e+07, 6.60e+07, 7.90e+01, 8.40e+06,
                3.70e+06, 1.18e+02, 4.40e+07, 6.95e+02, 1.60e+06, 6.20e+06,
                5.30e+07, 1.40e+06, 3.00e+06, 7.20e+06, 5.80e+06, 3.80e+06,
                9.60e+06, 4.50e+07, 6.30e+07, 4.90e+07, 7.70e+07, 4.40e+06,
                7.00e+07, 9.30e+06, 8.10e+06, 3.60e+07, 6.90e+06, 7.40e+06,
                8.40e+07, 9.70e+07, 2.00e+06, 1.90e+06, 1.80e+06, 5.30e+06,
                4.70e+07, 5.56e+02, 5.26e+02, 7.60e+07, 7.60e+06, 5.90e+07,
                9.70e+06, 7.80e+07, 7.20e+07, 4.30e+07, 7.70e+06, 6.30e+06,
                3.34e+02, 9.30e+07, 6.50e+07, 7.90e+07, 1.00e+08, 5.80e+07,
                5.00e+07, 6.80e+07, 6.40e+07, 3.40e+07, 6.70e+07, 6.00e+07
```

data.Size.replace(r'[kM]+$', '', regex=True): Bu kısım, 'Size' sütunundaki 'k' veya 'M' karakterlerini regex (düzenli ifadeler) kullanarak boş bir dizeyle değiştirir. Yani, 'k' veya 'M' karakterlerini kaldırır.

data.Size.str.extract(r'[\d.]+([KM]+)', expand=False): Bu ifade, 'Size' sütunundaki her bir değer için bir regex deseni uygular. Bu desen, bir sayı veya ondalık nokta ile başlayan ve 'k' veya 'M' ile biten bir dizedir. expand=False argümanı, çıktının bir dizi değil, bir Seri olarak döndürülmesini sağlar.

.fillna(1): Eksik değerleri 1 ile doldurur. Eğer 'k' veya 'M' olmayan bir değer varsa, bu değer 'k' veya 'M' olmayan bir değerdir, yani 1 olmalıdır.

.replace(['k','M'],[10**3,10**6]): 'k' değerlerini 10^3 (1000) ile, 'M' değerlerini 10^6 (1000000) ile değiştirir. Böylece, 'k' olanlar kilobayt cinsinden, 'M' olanlar megabayt cinsinden ifade edilir.

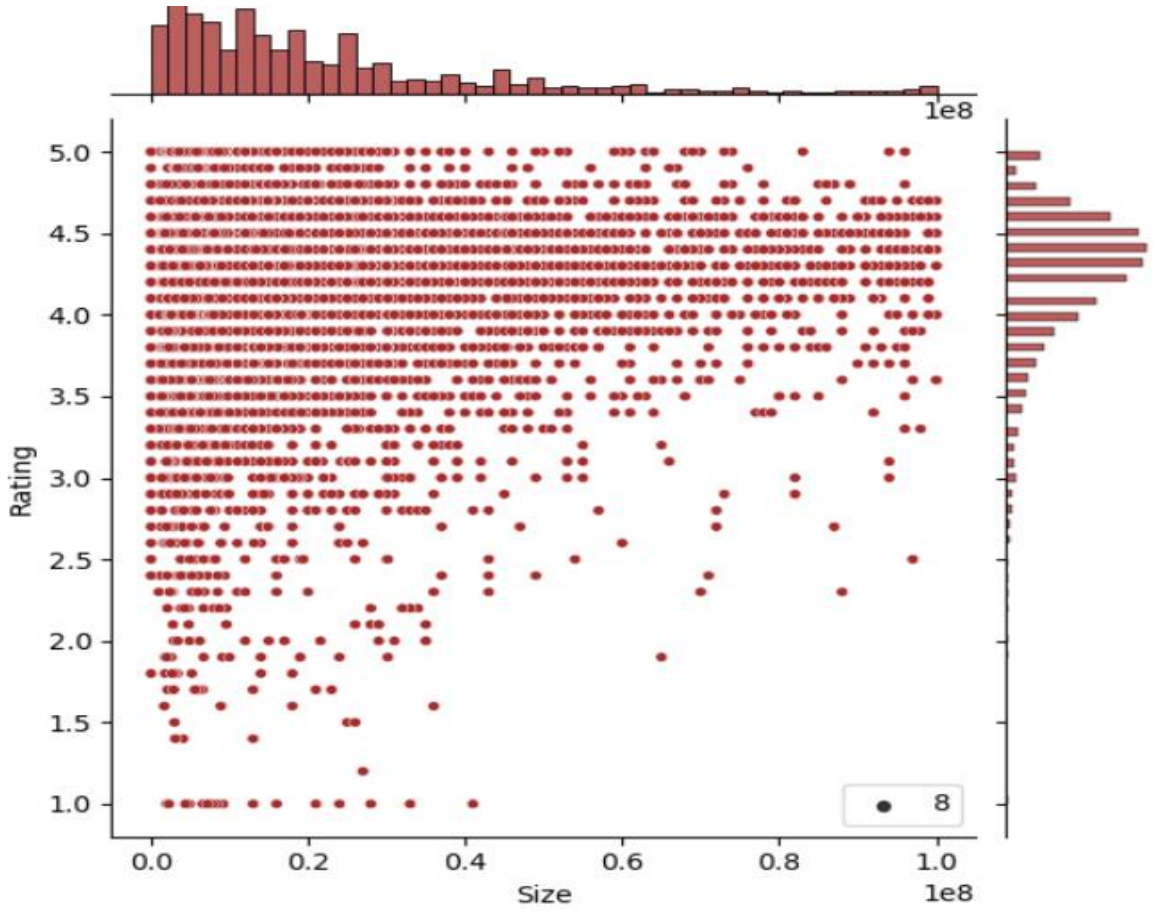
.astype(int): Son olarak, tüm değerleri tamsayıya dönüştürür.

Bu kod, 'Size' sütunundaki değerleri uygun bir formata dönüştürmek için kullanılır.

```
In [31]: data['Size'].fillna(data.groupby('Category')['Size'].transform('mean'),inplace=True)
```

```
In [32]: plt.figure(figsize=(10,10))
         g=sns.jointplot(x='Size',y='Rating',color='brown',data=data,size=8)
```

```
<Figure size 1000x1000 with 0 Axes>
```



Bu kod, 'Size' sütunundaki eksik değerleri her bir kategori için o kategoriye ait diğer değerlerin ortalaması ile doldurur.

data.groupby('Category')['Size'].transform('mean'): Bu ifade, 'Category' sütununa göre gruplandırılan 'Size' sütunundaki değerlerin ortalamasını hesaplar. Bu, her kategori için bir ortalama değer elde eder.

.fillna(): Bu metot, eksik değerleri belirtilen değerlerle doldurur. Burada, eksik değerler, her bir kategori için hesaplanan ortalama değerlerle doldurulur.

inplace=True: Bu argüman, değişikliğin doğrudan orijinal veri çerçevesine uygulanmasını sağlar, yani 'Size' sütunundaki eksik değerler yerinde doldurulur.

Bu kod, 'Size' sütunundaki eksik değerleri her bir kategori için o kategoriye ait diğer değerlerin ortalaması ile doldurarak veri setini tamamlar.

Grafik Yorumu: Dosyanın boyutuyla reytingi arasındaki ilişkiye bakarsak,genellikle Küçük boyutlu uygulamaların daha çok olduğu söylenebilir ancak bununla ilgili herhangi bir yargıda bulunmak doğru olmaz.

```
In [33]: data.head()
```

Out[33]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000000.0	10,000+	Free	0	Everyone	Art & Design	January 7, 2018	1.0.0	4.0.3 and up
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000000.0	500,000+	Free	0	Everyone	Art & Design;Pretend Play	January 15, 2018	2.0.0	4.0.3 and up
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8700000.0	5,000,000+	Free	0	Everyone	Art & Design	August 1, 2018	1.2.4	4.0.3 and up
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25000000.0	50,000,000+	Free	0	Teen	Art & Design	June 8, 2018	Varies with device	4.2 and up
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2800000.0	100,000+	Free	0	Everyone	Art & Design;Creativity	June 20, 2018	1.1	4.4 and up

```
In [34]: data['Installs'].head()
```

```
Out[34]: 0      10,000+
         1     500,000+
         2    5,000,000+
         3   50,000,000+
         4     100,000+
         Name: Installs, dtype: object
```

```
In [35]: data['Installs'].unique()
```

```
Out[35]: array(['10,000+', '500,000+', '5,000,000+', '50,000,000+', '100,000+',
                '50,000+', '1,000,000+', '10,000,000+', '5,000+', '100,000,000+',
                '1,000,000,000+', '1,000+', '500,000,000+', '100+', '500+', '10+',
                '5+', '50+', '1+'], dtype=object)
```

Datayı görüntüledik. İndirme sayıları ile reyting arasındaki ilişkiyi inceleyeceğiz. Installs verilerinin object türünden olduğunu görüyoruz. bunları sayısal verilere çevirmemiz gereklidir. Değerlerdeki artı işaretlerini virgülden kaldırıp sayısal ifadelere çevirmemiz gereklidir.

```
In [36]: data.Installs = data.Installs.apply(lambda x: x.replace(',',''))
         data.Installs = data.Installs.apply(lambda x: x.replace('+',''))
         data.Installs = data.Installs.apply(lambda x: int(x))

In [37]: data['Installs'].unique()

Out[37]: array([     10000,     500000,    5000000,   50000000,     100000,
                    50000,    1000000,   10000000,       5000,  100000000,
           1000000000,       1000,  500000000,        100,        500,
                    10,          5,         50,          1], dtype=int64)

In [38]: data['Installs'].head()

Out[38]: 0       10000
         1      500000
         2     5000000
         3    50000000
         4      100000
         Name: Installs, dtype: int64

In [39]: sorted_value=sorted(list(data['Installs'].unique()))

In [40]: sorted_value

Out[40]: [1,
          5,
          10,
          50,
          100,
          500,
          1000,
          5000,
          10000,
          50000,
          100000,
          500000,
          1000000,
          5000000,
```

'Installs' sütunundaki her değerdeki virgül karakterlerini boş bir dizeyle değiştirir.

'Installs' sütunundaki her değerdeki artı karakterlerini boş bir dizeyle değiştirir.

'Installs' sütunundaki her değeri tamsayıya dönüştürür. Bu işlemler sonucunda 'Installs' sütunundaki değerler virgül ve artı karakterlerinden arındırılmış ve tamsayı formatına dönüştürülmüş olur.

Şimdi burada yükleme sayılarını da küçükten büyüğe olacak şekilde sıralayabildik.

data['Installs']: Bu ifade, 'Installs' sütunundaki tüm değerleri seçer.

list(): Bu işlev, 'Installs' sütunundaki değerleri bir listede toplar.

sorted(): Bu işlev, bir liste veya başka bir sıralanabilir veri yapısı alır ve bu veriyi artan sırada sıralar.

Sonuç olarak, 'Installs' sütunundaki değerler küçükten büyüğe doğru sıralanır ve bu sıralı değerler 'sorted_value' değişkenine atanır.

In [41]:
```
data.head()
```

Out[41]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000000.0	10000	Free	0	Everyone	Art & Design	January 7, 2018	1.0.0	4.0.3 and up
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000000.0	500000	Free	0	Everyone	Art & Design;Pretend Play	January 15, 2018	2.0.0	4.0.3 and up
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8700000.0	5000000	Free	0	Everyone	Art & Design	August 1, 2018	1.2.4	4.0.3 and up
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25000000.0	50000000	Free	0	Teen	Art & Design	June 8, 2018	Varies with device	4.2 and up
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2800000.0	100000	Free	0	Everyone	Art & Design;Creativity	June 20, 2018	1.1	4.4 and up

In [42]:
```
data['Installs'].replace(sorted_value,range(0,len(sorted_value),1),inplace=True)
```

In [43]:
```
data.head()
```

Out[43]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000000.0	8	Free	0	Everyone	Art & Design	January 7, 2018	1.0.0	4.0.3 and up
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000000.0	11	Free	0	Everyone	Art & Design;Pretend Play	January 15, 2018	2.0.0	4.0.3 and up
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8700000.0	13	Free	0	Everyone	Art & Design	August 1, 2018	1.2.4	4.0.3 and up
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25000000.0	15	Free	0	Teen	Art & Design	June 8, 2018	Varies with device	4.2 and up
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2800000.0	10	Free	0	Everyone	Art & Design;Creativity	June 20, 2018	1.1	4.4 and up

sorted_value: Bu, 'Installs' sütunundaki değerlerin sıralanmış halidir.

range(0, len(sorted_value),1): Bu ifade, sıralanmış değerlerin indekslerini temsil eder. 0'dan başlayarak, sıralı değerlerin uzunluğuna kadar olan sayı dizisini oluşturur.

.replace(): Bu metot, bir değeri başka bir değerle değiştirir.

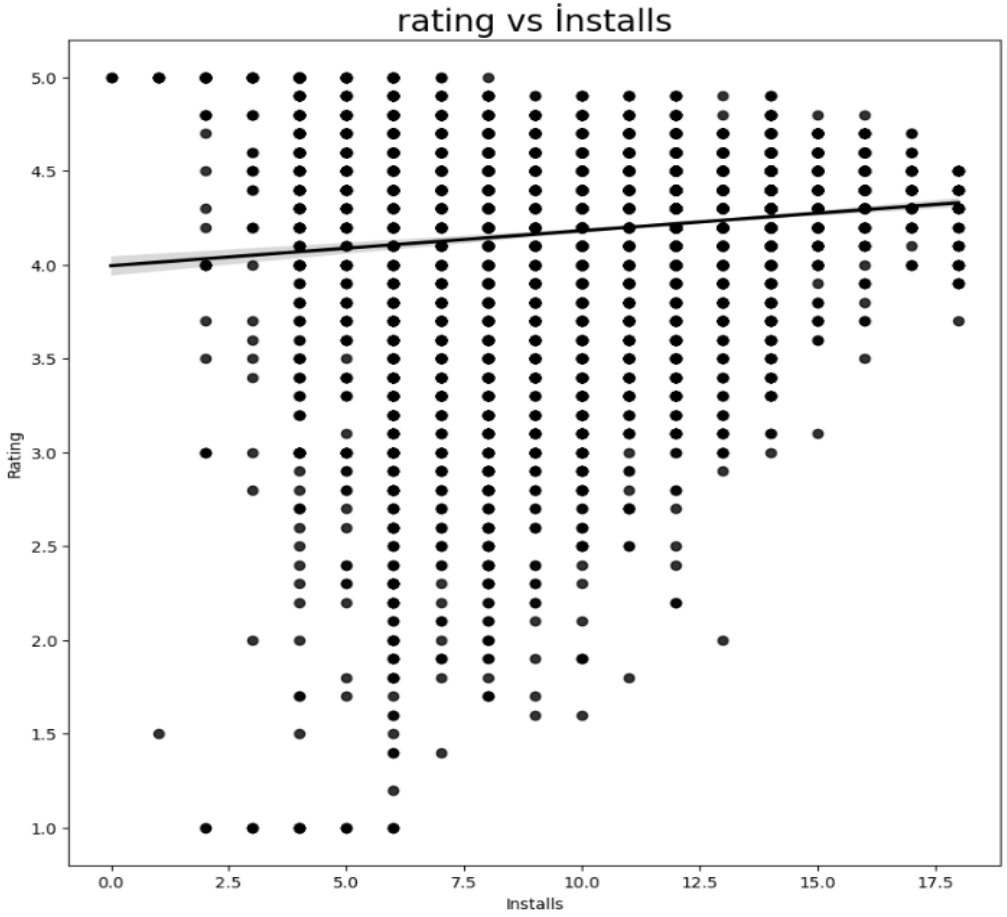
inplace=True: Bu argüman, değişikliğin doğrudan orijinal veri çerçevesine uygulanmasını sağlar.

Sonuç olarak, 'Installs' sütunundaki değerler, sıralanmış bir listeye göre sıralı bir şekilde değiştirilir ve bu değişiklikler orijinal veri çerçevesine uygulanır.

Baktığımız zaman burada indirme sayısına göre bir sıralama yapıldığını görüyoruz.

```
In [45]: plt.figure(figsize=(10,10))
         sns.regplot(x='Installs' ,y='Rating',color='black',data=data);
         plt.title('rating vs İnstalls',size=20)
```

```
Out[45]: Text(0.5, 1.0, 'rating vs İnstalls')
```



Bu kod, 'Installs' ve 'Rating' arasındaki ilişkiyi gösteren bir regresyon çizimini oluşturur. İşte açıklamaları:

plt.figure(figsize=(10,10)): Bu ifade, yeni bir şekil oluşturarak çizimin boyutunu ayarlar. Burada, genişlik ve yükseklik değerleri 10 birim olarak belirlenmiştir.

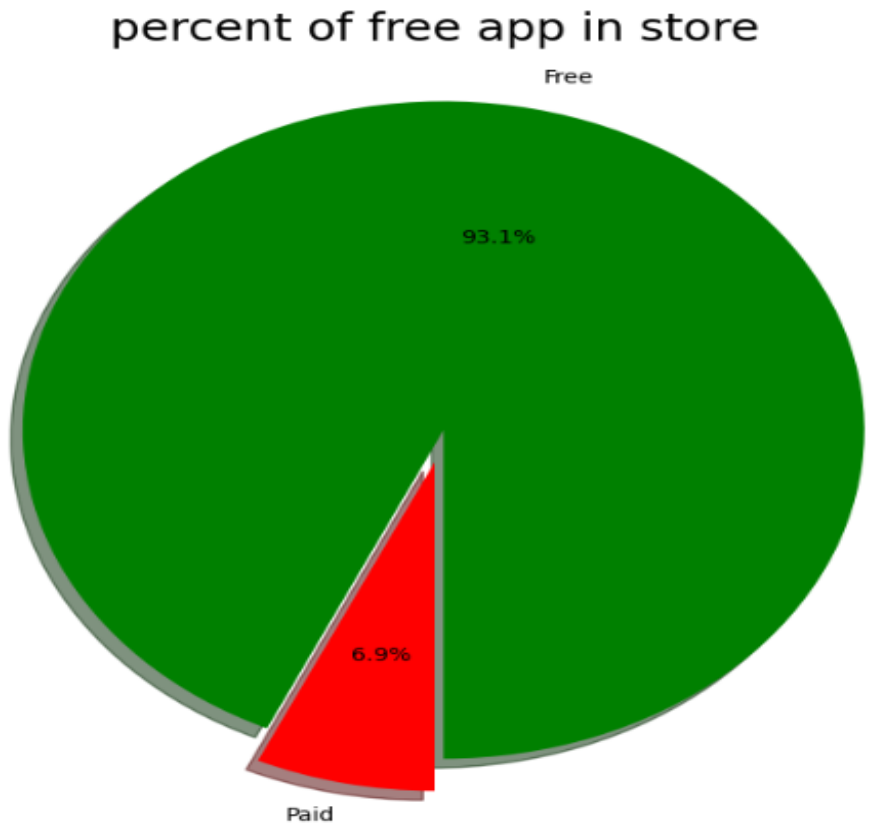
sns.regplot(x='Installs' ,y='Rating',color='black',data=data): Bu satır, seaborn kütüphanesinin regplot() fonksiyonunu kullanarak bir regresyon çizimi oluşturur. X ekseninde 'Installs', Y ekseninde 'Rating' kullanılır. color='black' argümanı, çizimin rengini siyah olarak ayarlar. data=data ifadesi, verinin çizimde kullanılacağını belirtir.

plt.title('rating vs İnstalls',size=20): Bu satır, çizimin başlığını 'rating vs İnstalls' olarak ayarlar ve başlığın boyutunu 20 piksel olarak belirler.

Bu kod, 'Installs' ve 'Rating' arasındaki ilişkiyi gösteren bir regresyon çizimi oluşturur. Bu çizimde, yükleme sayısı ile derecelendirme arasındaki ilişki gösterilir

Grafik Yorumu: Yükleme sayıları arttıkça kısmi olarak Reyting artışı olduğunu görüyoruz.

```
In [47]: data['Type'].unique()
Out[47]: array(['Free', 'Paid'], dtype=object)

In [48]: labels =data['Type'].value_counts(sort = True).index
         sizes = data['Type'].value_counts(sort = True)
         colors = ['Green','Red']
         explode = (0.1,0)  #explode 1st slice
         plt.rcParams['figure.figsize'] =8,8
         #plot
         plt.pie(sizes, explode=explode, labels=labels, colors=colors,
                 autopct='%1.1f%%',shadow=True, startangle=270,)
         plt.title('percent of free app in store',size= 20)
         plt.show()
```



Bu kod, veri setindeki ücretsiz ve ücretli uygulamaların yüzdelik dağılımını gösteren bir pasta grafiği oluşturur.

labels: Bu, 'Type' sütunundaki değerlerin benzersiz indekslerini içerir.

sizes: Bu, 'Type' sütunundaki değerlerin sıklığını içerir.

colors: Bu, pasta dilimlerinin renklerini belirler.

explode: Bu, pasta dilimlerinin ne kadar patlatılacağını belirler. Burada, ilk dilim (%1.1f) patlatılır.

autopct: Bu, pasta dilimlerinin yüzde değerlerini göstermek için formatlamayı belirler.

shadow: Bu, pasta grafiğinin gölgeli olmasını sağlar.

startangle: Bu, pasta grafiğinin başlangıç açısını belirler.

Grafik Yorumu: Şekilde Google Play'deki bu veri seti (ilgili tarih için) o zamana ait uygulamaların %93'ünün ücretsiz,%7'si ücretli olarak karşımıza çıktığını görüyoruz.

```
In [49]: data['Free'] = data['Type'].map(lambda s:1 if s=='Free' else 0)
         data.drop(['Type'], axis=1, inplace=True)
```

```
In [50]: data.head()
```

Out[50]:

	App	Category	Rating	Reviews	Size	Installs	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver	Free
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000000.0	8	0	Everyone	Art & Design	January 7, 2018	1.0.0	4.0.3 and up	1
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000000.0	11	0	Everyone	Art & Design;Pretend Play	January 15, 2018	2.0.0	4.0.3 and up	1
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8700000.0	13	0	Everyone	Art & Design	August 1, 2018	1.2.4	4.0.3 and up	1
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25000000.0	15	0	Teen	Art & Design	June 8, 2018	Varies with device	4.2 and up	1
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2800000.0	10	0	Everyone	Art & Design;Creativity	June 20, 2018	1.1	4.4 and up	1

```
In [51]: data['Free'].unique()
```

```
Out[51]: array([1, 0], dtype=int64)
```

```
In [52]: data['Price'].head()
```

```
Out[52]: 0    0
         1    0
         2    0
         3    0
         4    0
         Name: Price, dtype: object
```

```
In [53]: data['Price'].value_counts().head(15)
```

```
Out[53]: Price
         0          8715
         $2.99       114
         $0.99       106
         $4.99        70
         $1.99        59
         $3.99        58
         $1.49        30
         $2.49        21
         $5.99        18
         $9.99        16
         $6.99        13
```

*data['Free'] = data['Type'].map(lambda s: 1 if s == 'Free' else 0): Bu satır, data DataFrame'inde 'Free' adında yeni bir sütun oluşturur. 'Type' sütunundaki değer 'Free' ise her bir satıra 1 değeri atar, değilse 0 değeri atar. map() fonksiyonu, 'Type' sütunundaki her bir öğeye belirli bir fonksiyonu (bu durumda bir lambda fonksiyonu) uygulamak için kullanılır.data.drop(['Type'], axis=1, inplace=True): Bu satır, DataFrame data'dan 'Type' sütununu kaldırır. drop() fonksiyonu, DataFrame'den sütunları (veya satırları, axis parametresine bağlı olarak) kaldırmak için kullanılır. inplace=True parametresi, işlemin DataFrame üzerinde gerçekleştirildiği ve yeni bir DataFrame döndürmediği anlamına gelir.
Özetlemek gerekirse, bu iki kod satırı, her bir öğenin ücretsiz olup olmadığını gösteren 'Free' adında yeni bir sütun ekler ve ardından DataFrame'den 'Type' sütununu kaldırır.

*Bu kod satırı, 'Free' adlı sütundaki benzersiz değerleri görüntüler. unique() fonksiyonu, belirtilen sütunda bulunan benzersiz değerleri döndürür.
Sonuç, [1, 0] şeklinde bir numpy array olarak döndü. Bu, 'Free' sütununda yalnızca iki farklı değer olduğunu (1 ve 0) gösterir. Bu durumda, 1 değeri ücretsiz uygulamaları temsil ederken, 0 değeri ücretli uygulamaları temsil eder. dtype=int64 ise bu değerlerin tamsayı (integer) türünde olduğunu belirtir.

*İlk satır, 'Price' sütununun ilk beş satırını yazdıracaktır.

*Bu kod, 'Price' sütunundaki farklı değerlerin sayısını hesaplar ve bu değerleri en çok tekrar edenlerden başlayarak sıralar. Sonuç olarak, her bir fiyatın kaç kez tekrarlandığını içeren bir seri döndürülür ve bu seri içinde en çok tekrar eden 15 fiyatı görüntüler.

```
In [53]: data['Price'].value_counts().head(15)
```

```
Out[53]: Price
         0          8715
         $2.99       114
         $0.99       106
         $4.99        70
         $1.99        59
         $3.99        58
         $1.49        30
         $2.49        21
         $5.99        18
         $9.99        16
         $6.99        13
         $399.99      11
         $14.99       10
         $4.49         9
         $3.49         7
         Name: count, dtype: int64
```

*En çok tekrar eden 15 fiyat listelenmiştir. Her bir fiyatın kaç kez tekrarlandığını belirtilen sayıyla birlikte görebiliyoruz.

```
In [54]: data.Price = data.Price.apply(lambda x: x.replace('$',''))
         data['Price'] = data['Price'].apply(lambda x: float(x))
```

*İlk satır, 'Price' sütunundaki her bir değere bir lambda fonksiyonunu uygular. Lambda fonksiyonu, her bir değerdeki "$" işaretini boş bir dizeyle değiştirir, yani "$" işaretini kaldırır. Sonuç olarak, 'Price' sütunundaki her bir değer artık "$" işareti içermeyecek.
*İkinci satır, 'Price' sütunundaki her bir değere başka bir lambda fonksiyonunu uygular. Bu lambda fonksiyonu, her bir değeri ondalık sayıya dönüştürür. Bu dönüşüm, 'Price' sütunundaki değerlerin tamsayı olmadığı durumlarda gereklidir.

```
In [55]: data.head()
```

Out[55]:

	App	Category	Rating	Reviews	Size	Installs	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver	Free
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000000.0	8	0.0	Everyone	Art & Design	January 7, 2018	1.0.0	4.0.3 and up	1
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000000.0	11	0.0	Everyone	Art & Design;Pretend Play	January 15, 2018	2.0.0	4.0.3 and up	1
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8700000.0	13	0.0	Everyone	Art & Design	August 1, 2018	1.2.4	4.0.3 and up	1
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25000000.0	15	0.0	Teen	Art & Design	June 8, 2018	Varies with device	4.2 and up	1
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2800000.0	10	0.0	Everyone	Art & Design;Creativity	June 20, 2018	1.1	4.4 and up	1

```
In [56]: data['Price'].describe()
```

```
Out[56]: count    9360.000000
         mean        0.961279
         std        15.821640
         min         0.000000
         25%         0.000000
         50%         0.000000
         75%         0.000000
         max       400.000000
         Name: Price, dtype: float64
```

Bu kod, 'Price' sütununun istatistiksel özetini hesaplar. Bu özet genellikle bir serideki sayısal değerlerin dağılımı hakkında fikir verir. İşte çıktıda genellikle bulunan istatistikler:
•Count: Sütundaki toplam giriş sayısı.
•Mean: Sütundaki değerlerin ortalama değeri.
•Std: Sütundaki değerlerin standart sapması.
•Min: Sütundaki en küçük değer.
•25%: Sütundaki değerlerin %25'ini içeren ilk çeyrek.
•50%: Sütundaki değerlerin %50'sini içeren medyan (ortanca değer).
•75%: Sütundaki değerlerin %75'ini içeren üçüncü çeyrek.
•Max: Sütundaki en büyük değer.

```
In [57]: data[data['Price']==400]
```

Out[57]:

	App	Category	Rating	Reviews	Size	Installs	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver	Free
4367	I'm Rich - Trump Edition	LIFESTYLE	3.6	275	7300000.0	8	400.0	Everyone	Lifestyle	May 3, 2018	1.0.1	4.1 and up	0

```
In [58]: plt.figure(figsize=(10,10))
sns.regplot(x='Price' ,y='Rating',color='purple',data=data);
plt.title('rating vs Price',size=20)
```

```
Out[58]: Text(0.5, 1.0, 'rating vs Price')
```

Bu kod, 'Price' sütununda değeri 400 olan satırları içeren DataFrame'i döndürecektir

Bu kod, 'Price' ve 'Rating' sütunları arasındaki ilişkiyi görselleştirmek için bir regresyon plotu oluşturur. 'Price' sütunu x-ekseninde, 'Rating' sütunu ise y-ekseninde olacak şekilde noktaları gösterir ve bu noktalar arasında bir regresyon çizgisi çizer. Ayrıca grafiğin başlığını 'rating vs Price' olarak ayarlar.

Grafik yorum:Fiyat arttıkça raytinglerde bir miktar düşüş yaşannmış diyebiliriz.Ürün pahalandıkça insanların beklentisini daha az karşılıyor diyebiliriz.

```
In [59]: data.head()
```

Out[59]:

	App	Category	Rating	Reviews	Size	Installs	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver	Free
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000000.0	8	0.0	Everyone	Art & Design	January 7, 2018	1.0.0	4.0.3 and up	1
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000000.0	11	0.0	Everyone	Art & Design;Pretend Play	January 15, 2018	2.0.0	4.0.3 and up	1
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8700000.0	13	0.0	Everyone	Art & Design	August 1, 2018	1.2.4	4.0.3 and up	1
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25000000.0	15	0.0	Teen	Art & Design	June 8, 2018	Varies with device	4.2 and up	1
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2800000.0	10	0.0	Everyone	Art & Design;Creativity	June 20, 2018	1.1	4.4 and up	1

#Bu kod, veri çerçevesinin ilk beş satırını gösterecektir.

```
In [60]: data[data['Content Rating']=='Unrated']
```

Out[60]:

	App	Category	Rating	Reviews	Size	Installs	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver	Free
8266	DC Universe Online Map	TOOLS	4.1	1186	6400000.0	9	0.0	Unrated	Tools	February 27, 2012	1.3	2.3.3 and up	1

#Bu kod, 'Content Rating' sütunu içinde değeri 'Unrated' olan tüm satırları içeren bir DataFrame döndürecektir. Bu şekilde veri çerçevesindeki 'Content Rating' sütununda 'Unrated' değerine sahip olan girişleri filtreleyebilirsiniz. Eğer 'Unrated(derecelndirlmemiş) içeren herhangi bir satır yoksa, boş bir DataFrame dönecektir.

```
In [61]: data=data[data['Content Rating']!='Unrated']
```

#'Content Rating' sütunu 'Unrated' olmayan satırları içeren bir veri çerçevesi elde edilir ve bu yeni veri çerçevesi data değişkenine atanır.

```
In [62]: data[data['Content Rating']=='Unrated']
```

Out[62]:

App	Category	Rating	Reviews	Size	Installs	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver	Free

#’Unrated’ değere sahip veri yoktur.Boş bir DF dönmüştür

```
In [63]: data['Content Rating'].unique()
```

Out[63]:

```
array(['Everyone', 'Teen', 'Everyone 10+', 'Mature 17+',
       'Adults only 18+'], dtype=object)
```

#Bu kod, 'Content Rating' sütununda bulunan benzersiz değerleri döndürecektir. Yani, veri çerçevesinde 'Content Rating' sütununda hangi değerlerin bulunduğunu görmemizi sağlar.

```
In [64]: data=pd.get_dummies(data,columns=['Content Rating'])
```

#Bu kod, 'Content Rating' sütunundaki kategorik değerleri ikili (binary) özelliklere dönüştürmek için kullanılır. get_dummies() fonksiyonu, belirtilen sütundaki her bir kategorik değeri ayrı bir sütun olarak ele alır ve her bir değerin varlığını veya yokluğunu temsil eden ikili (0 veya 1) değerlerle doldurur.

```
In [65]: data.head()
```

#Data.head ile datayı görüyoruz.

Out[65]:

	App	Category	Rating	Reviews	Size	Installs	Price	Genres	Last Updated	Current Ver	Android Ver	Free	Content Rating_Adults only 18+	Conte Rating_Everyo
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000000.0	8	0.0	Art & Design	January 7, 2018	1.0.0	4.0.3 and up	1	False	Tr
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000000.0	11	0.0	Art & Design;Pretend Play	January 15, 2018	2.0.0	4.0.3 and up	1	False	Tr
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8700000.0	13	0.0	Art & Design	August 1, 2018	1.2.4	4.0.3 and up	1	False	Tr
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25000000.0	15	0.0	Art & Design	June 8, 2018	Varies with device	4.2 and up	1	False	Fal
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2800000.0	10	0.0	Art & Design;Creativity	June 20, 2018	1.1	4.4 and up	1	False	Tr

#'Content Rating_Adults only 18+', 'Content Rating_Everyone', 'Content Rating_Everyone 10+', 'Content Rating_Mature 17+' ve 'Content Rating_Teen'

Bu sütunlar, her bir değerin varlığını (1) veya yokluğunu (0) gösterir. Örneğin, bir uygulamanın 'Content Rating' sütununda 'Teen' değeri varsa, 'Content Rating_Teen' sütunu 1 değerini alırken, diğer sütunlar 0 değerini alır.

Bu şekilde, kategorik verilerin makine öğrenimi modellerine giriş olarak kullanılabilmesi sağlanır. Bu tür bir dönüşüm, özellikle sınıflandırma problemlerinde sıkça kullanılır.

```
In [66]: data.head()
```

Out[66]:

	App	Category	Rating	Reviews	Size	Installs	Price	Genres	Last Updated	Current Ver	Android Ver	Free	Content Rating_Adults only 18+	Conte Rating_Everyo
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000000.0	8	0.0	Art & Design	January 7, 2018	1.0.0	4.0.3 and up	1	False	Tr
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000000.0	11	0.0	Art & Design;Pretend Play	January 15, 2018	2.0.0	4.0.3 and up	1	False	Tr
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8700000.0	13	0.0	Art & Design	August 1, 2018	1.2.4	4.0.3 and up	1	False	Tr
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25000000.0	15	0.0	Art & Design	June 8, 2018	Varies with device	4.2 and up	1	False	Fal
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2800000.0	10	0.0	Art & Design;Creativity	June 20, 2018	1.1	4.4 and up	1	False	Tr

#Bu kod, veri çerçevesinin ilk beş satırını gösterecektir. ‘Genres’ sütununa bakalım.Uygulamanın Google Play'de ya da Google Play Store'da hangi kategoriye ait olduğunu belirtir. Örneğin aksiyon,puzzle ya da bulmaca diyebileceğimiz olarak belirtilmiştir.

```
In [67]: len(data['Genres'].unique()),'genres'
```

```
Out[67]: (115, 'genres')
```

#Bu kod, 'Genres' sütunundaki benzersiz değerlerin sayısını hesaplar ve bu sayıyı ekrana yazdırır. Çıktı 115 benzersiz tür var şeklindedir.

```
In [68]: data['Genres'].unique()
```

```
Out[68]: array(['Art & Design', 'Art & Design;Pretend Play',
       'Art & Design;Creativity', 'Auto & Vehicles', 'Beauty',
       'Books & Reference', 'Business', 'Comics', 'Comics;Creativity',
       'Communication', 'Dating', 'Education;Education', 'Education',
       'Education;Creativity', 'Education;Music & Video',
       'Education;Action & Adventure', 'Education;Pretend Play',
       'Education;Brain Games', 'Entertainment',
       'Entertainment;Music & Video', 'Entertainment;Brain Games',
       'Entertainment;Creativity', 'Events', 'Finance', 'Food & Drink',
       'Health & Fitness', 'House & Home', 'Libraries & Demo',
       'Lifestyle', 'Lifestyle;Pretend Play',
       'Adventure;Action & Adventure', 'Arcade', 'Casual', 'Card',
       'Casual;Pretend Play', 'Action', 'Strategy', 'Puzzle', 'Sports',
       'Music', 'Word', 'Racing', 'Casual;Creativity',
       'Casual;Action & Adventure', 'Simulation', 'Adventure', 'Board',
       'Trivia', 'Role Playing', 'Simulation;Education',
       'Action;Action & Adventure', 'Casual;Brain Games',
       'Simulation;Action & Adventure', 'Educational;Creativity',
       'Puzzle;Brain Games', 'Educational;Education', 'Card;Brain Games',
       'Educational;Brain Games', 'Educational;Pretend Play',
       'Entertainment;Education', 'Casual;Education',
       'Music;Music & Video', 'Racing;Action & Adventure',
       'Arcade;Pretend Play', 'Role Playing;Action & Adventure',
       'Simulation;Pretend Play', 'Puzzle;Creativity',
       'Sports;Action & Adventure', 'Educational;Action & Adventure',
       'Arcade;Action & Adventure', 'Entertainment;Action & Adventure',
       'Puzzle;Action & Adventure', 'Strategy;Action & Adventure',
       'Music & Audio;Music & Video', 'Health & Fitness;Education',
       'Adventure;Education', 'Board;Brain Games',
       'Board;Action & Adventure', 'Board;Pretend Play',
       'Casual;Music & Video', 'Role Playing;Pretend Play',
       'Entertainment;Pretend Play', 'Video Players & Editors;Creativity',
       'Card;Action & Adventure', 'Medical', 'Social', 'Shopping',
       'Photography', 'Travel & Local',
       'Travel & Local;Action & Adventure', 'Tools', 'Tools;Education',
       'Personalization', 'Productivity', 'Parenting',
       'Parenting;Music & Video', 'Parenting;Brain Games',
       'Parenting;Education', 'Weather', 'Video Players & Editors',
       'Video Players & Editors;Music & Video', 'News & Magazines',
       'Maps & Navigation', 'Health & Fitness;Action & Adventure',
```

#Bu kod, 'Genres' sütunundaki benzersiz türleri liste olarak döndürecektir. Örneğin, 'Puzzle', 'Action', 'Adventure' gibi farklı oyun türleri olabilir. Bu şekilde, veri çerçevesindeki benzersiz türleri görebilirsiniz.

```
In [69]: data.Genres.value_counts().head(10)
```

```
Out[69]: Genres
         Tools            731
         Entertainment    533
         Education        468
         Action           358
         Productivity     351
         Medical          350
         Sports           333
         Communication    328
         Finance          323
         Photography      317
         Name: count, dtype: int64
```

#Bu kod, 'Genres' sütunundaki farklı türlerin sayısını hesaplar ve en çok tekrar eden 10 türü gösterir. Yani, her bir türün veri çerçevesinde kaç kez tekrarlandığını sayar ve en çok tekrar eden ilk 10 türü gösterir.

```
In [70]: data.Genres.value_counts().tail(10)
```

```
Out[70]: Genres
         Health & Fitness;Action & Adventure    1
         Health & Fitness;Education             1
         Travel & Local;Action & Adventure      1
         Board;Pretend Play                     1
         Lifestyle;Pretend Play                 1
         Parenting;Brain Games                  1
         Card;Brain Games                       1
         Tools;Education                        1
         Entertainment;Education                1
         Strategy;Creativity                    1
         Name: count, dtype: int64
```

#Bu kod, 'Genres' sütununda bulunan farklı türlerin sayısını hesaplar ve en az tekrar eden son 10 türü gösterir. Yani, her

bir türün veri çerçevesinde kaç kez tekrarlandığını sayar ve en az tekrar eden son 10 türü gösterir

```
In [71]: data['Genres']=data['Genres'].str.split(';').str[0]
```

#Bu kod, 'Genres' sütunundaki değerleri ";" karakterine göre böler ve her bir satır için sadece ilk türü (ilk elemanı) alır. Bunu yapmak için str.split(';').str[0] ifadesi kullanılır.Örneğin, eğer 'Genres' sütununda "Action;Adventure" gibi bir değer varsa, bu kod bu değeri ";" karakterine göre böler vesadece "Action" kısmını alır.

```
In [72]: len(data['Genres'].unique()),'genres'
```

```
Out[72]: (48, 'genres')
```

#Bu kod, 'Genres' sütunundaki benzersiz değerlerin sayısını hesaplar ve bu sayıyı ekrana yazdırır, "genres" kelimesi, bu sayının "tür" olarak adlandırıldığını belirtmek içindir48 farklı tür kalmıştır,veri sadeleştrirlmiştir.

```
In [73]: data['Genres'].unique()
```

```
Out[73]: array(['Art & Design', 'Auto & Vehicles', 'Beauty', 'Books & Reference',
                'Business', 'Comics', 'Communication', 'Dating', 'Education',
                'Entertainment', 'Events', 'Finance', 'Food & Drink',
                'Health & Fitness', 'House & Home', 'Libraries & Demo',
                'Lifestyle', 'Adventure', 'Arcade', 'Casual', 'Card', 'Action',
                'Strategy', 'Puzzle', 'Sports', 'Music', 'Word', 'Racing',
                'Simulation', 'Board', 'Trivia', 'Role Playing', 'Educational',
                'Music & Audio', 'Video Players & Editors', 'Medical', 'Social',
                'Shopping', 'Photography', 'Travel & Local', 'Tools',
                'Personalization', 'Productivity', 'Parenting', 'Weather',
                'News & Magazines', 'Maps & Navigation', 'Casino'], dtype=object)
```

Bu kod, 'Genres' sütunundaki benzersiz türleri bir dizi olarak döndürecektir. Örneğin, 'Puzzle', 'Action', 'Adventure' gibi farklı oyun türleri olabilir. Bu şekilde, veri çerçevesindeki benzersiz türleri görelim

```
In [75]: data['Genres'].replace('Music & Audio','Music',inplace=True)
```

#Bu kod, 'Genres' sütununda 'Music & Audio' değerlerini 'Music' olarak değiştirir. Bu işlem, 'Music & Audio' değerlerinin 'Music' olarak güncellenmesini sağlar.

```
In [78]: data[['Genres','Rating']].groupby(['Genres'],as_index=False).mean().describe()
```

```
Out[78]:
```

	Rating
count	47.000000
mean	4.210662
std	0.104405
min	3.970769
25%	4.132039
50%	4.198246
75%	4.282529
max	4.435556

Bu kodun çıktısı, gruplanmış 'Rating' sütununun istatistiksel özetini verecektir. Bu, her türün ortalama derecesini, standart sapmasını ve diğer istatistikleri görmemizi sağlar.

In [79]:
```
data[['Genres','Rating']].groupby(['Genres'],as_index=False).mean().sort_values('Rating').head()
```

Out[79]:

	Genres	Rating
14	Dating	3.970769
43	Trivia	4.039286
41	Tools	4.047131
25	Maps & Navigation	4.051613
44	Video Players & Editors	4.063190

#Bu kod, 'Genres' sütununa göre gruplanmış ve her bir tür için 'Rating' sütununun ortalamasını hesaplayarak en düşük ortalama dereceye sahip olan ilk beş türü gösterir.

In [80]:
```
data[['Genres','Rating']].groupby(['Genres'],as_index=False).mean().sort_values('Rating').head(1)
```

Out[80]:

	Genres	Rating
14	Dating	3.970769

#Bu kod, 'Genres' sütununa göre gruplanmış ve her bir tür için 'Rating' sütununun ortalamasını hesaplayarak en düşük ortalama dereceye sahip olan ilk türü gösterir.

In [81]:
```
data[['Genres','Rating']].groupby(['Genres'],as_index=False).mean().sort_values('Rating').tail(5)
```

Out[81]:

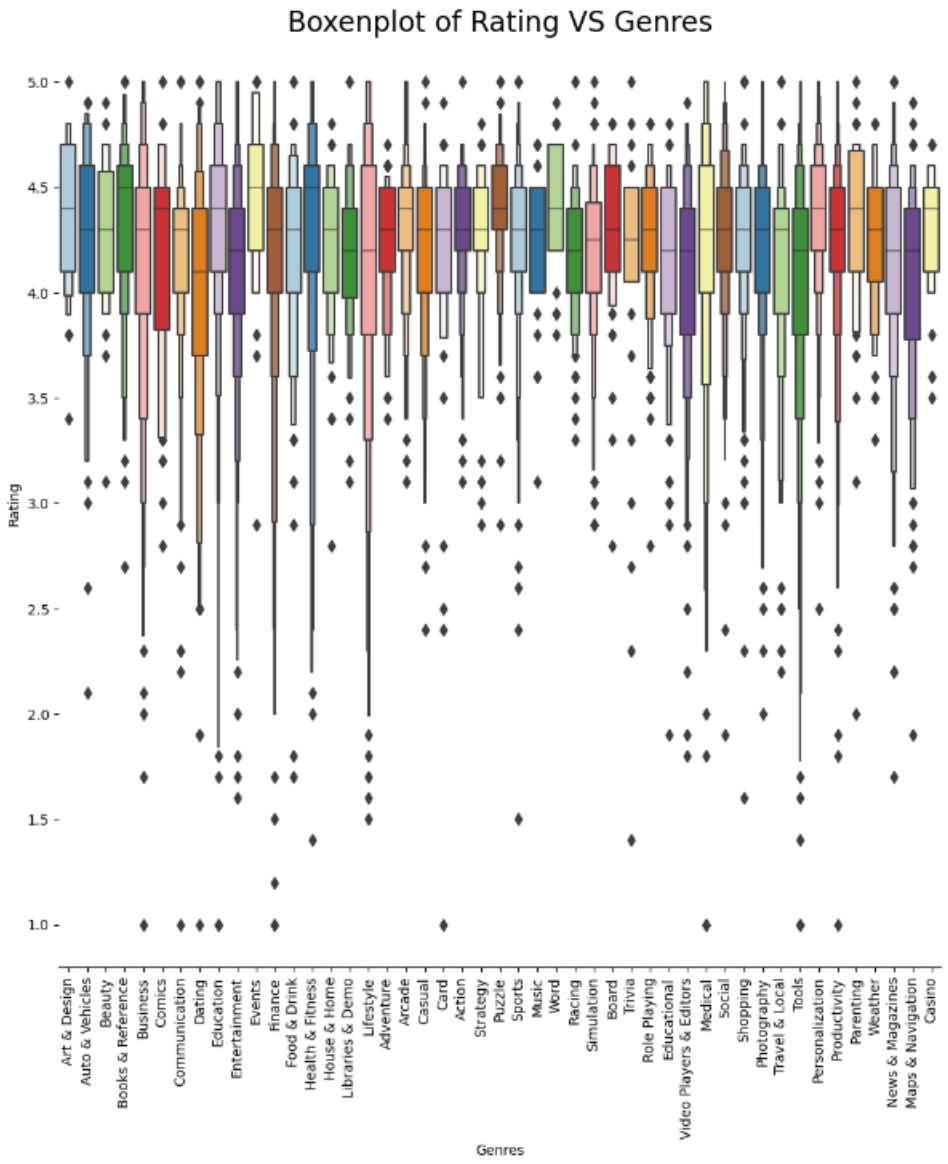
	Genres	Rating
7	Books & Reference	4.344444
3	Art & Design	4.367188
33	Puzzle	4.389116
46	Word	4.410714
18	Events	4.435556

#Bu kod, 'Genres' sütununa göre gruplanmış ve her bir tür için 'Rating' sütununun ortalamasını hesaplayarak en yüksek ortalama dereceye sahip olan son beş türü gösterir.

In [82]:
```
g=sns.catplot(x='Genres',y='Rating',data=data, kind='boxen',height =10 ,palette = 'Paired')
g.despine(left=True)
g.set_xticklabels(rotation=90)
g = g.set_ylabels('Rating')
plt.title('Boxenplot of Rating VS Genres',size = 20)
```

#Bu kod, 'Genres' sütununa göre verilerin 'Rating' sütununa göre dağılımını kutu grafik (boxen plot) türünde görselleştirir.

Out[82]: Text(0.5, 1.0, 'Boxenplot of Rating VS Genres')



#en yüksek ortalama puanının burada Events olduğunu görmüştük , standart sapmaların incelenmesi ile türlerin reytinglerinde çok fazla etkisinin olmadığı da söylenebilir.Çok büyük sapmalar Yoktur

```
In [83]: data['Last Updated'].head()
```

```
Out[83]: 0     January 7, 2018
         1    January 15, 2018
         2      August 1, 2018
         3        June 8, 2018
         4       June 20, 2018
         Name: Last Updated, dtype: object
```

```
In [84]: data['new']=pd.to_datetime(data['Last Updated'])
         data['new'].describe()
```

```
Out[84]: count                            9359
         mean     2017-11-29 18:24:25.541190400
         min                2010-05-21 00:00:00
         25%                2017-10-09 00:00:00
         50%                2018-06-01 00:00:00
         75%                2018-07-24 00:00:00
         max                2018-08-08 00:00:00
         Name: new, dtype: object
```

```
In [85]: data['new'].max()
```

```
Out[85]: Timestamp('2018-08-08 00:00:00')
```

```
In [86]: data.head()
```

Out[86]:

	App	Category	Rating	Reviews	Size	Installs	Price	Genres	Last Updated	Current Ver	Android Ver	Free	Content Rating_Adults only 18+	Content Rating_Everyone	Rati
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000000.0	8	0.0	Art & Design	January 7, 2018	1.0.0	4.0.3 and up	1	False	True	
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000000.0	11	0.0	Art & Design	January 15, 2018	2.0.0	4.0.3 and up	1	False	True	
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8700000.0	13	0.0	Art & Design	August 1, 2018	1.2.4	4.0.3 and up	1	False	True	
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25000000.0	15	0.0	Art & Design	June 8, 2018	Varies with device	4.2 and up	1	False	False	
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2800000.0	10	0.0	Art & Design	June 20, 2018	1.1	4.4 and up	1	False	True	

```
In [87]: data['new'][0]-data['new'].max()
```

```
Out[87]: Timedelta('-213 days +00:00:00')
```

```
In [88]: data['lastupdate']=(data['new']-data['new'].max()).dt.days
         data['lastupdate'].head()
```

```
Out[88]: 0   -213
         1   -205
         2     -7
         3    -61
         4    -49
         Name: lastupdate, dtype: int64
```

#Veri çerçevesindeki 'Last Updated' sütunu, her bir uygulamanın son güncelleme tarihini içerir. Bu tarihler genellikle metin formatında tutulur. String formatında olduğunu ve veri görselleştirme veya modelleme işlemlerinde kullanılamayacağını söyleyebiliriz bunun için bu özelliğin formatına dönüştürülmesi gerekiyor

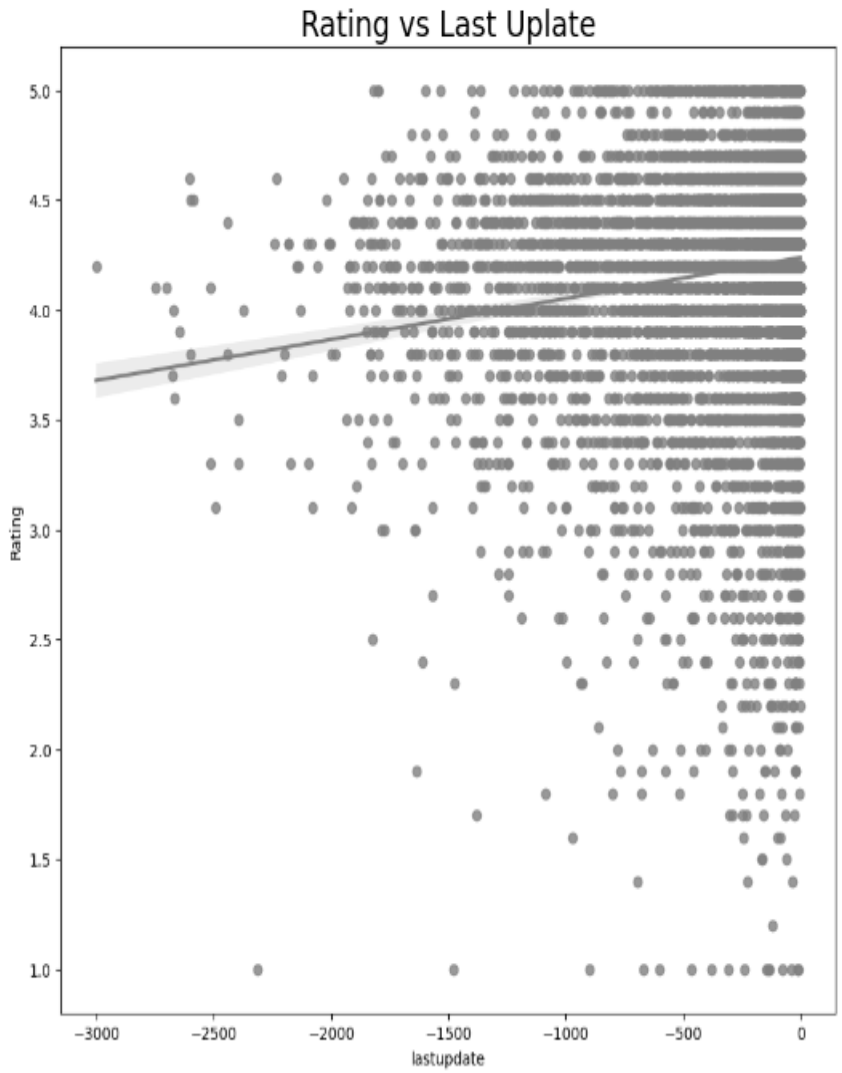
#Bu kod, 'Last Updated' sütunundaki tarih verilerini pd.to_datetime() fonksiyonuyla tarih-zaman nesnelerine dönüştürür ve 'new' adında yeni bir sütuna ekler. Daha sonra, 'new' sütununun istatistiksel özetini hesaplar.

#Bu kod, 'new' sütunundaki en büyük (maximum) tarih-zaman değerini döndürecektir. Yani, veri çerçevesindeki 'Last Updated' sütunundaki en son güncelleme tarihini verecektir.

#Bu kod, 'new' sütunundaki ilk satırdaki tarih-zaman değerini ('new' sütunundaki ilk öğe) ve 'new' sütunundaki en büyük (maximum) tarih-zaman değerini (en son güncelleme tarihi) çıkararak bir zaman farkı hesaplar.(En son 213 gün önce güncellenmiş.

#Bu kodun çalışması sonucunda 'lastupdate' sütununda, her uygulamanın en son güncelleme tarihinden itibaren geçen gün sayısı bulunacaktır. Bu yeni sütun, 'new' sütunundaki tarihler arasındaki farkı günlere dönüştürerek elde edilmiştir.

```
In [89]: plt.figure(figsize=(10,10))
         sns.regplot(x='lastupdate',y='Rating',color='Gray',data=data);
         plt.title('Rating vs Last Uplate',size=20)

Out[89]: Text(0.5, 1.0, 'Rating vs Last Uplate')
```



#Bu kod, her bir uygulamanın en son güncellemeden itibaren geçen gün sayısını ('lastupdate') ve derecesini ('Rating') içeren bir regresyon grafiği oluşturur.

#Eski diyebileceğimiz uygulamaların güncelleme almaması durumunda genellikle reytinglerinin düşük olduğunu söyleyebiliriz. Çünkü, güncellemelerin olmaması uygulamanın güncel ve kullanıcıların taleplerine uygun olmadığı anlamına gelebilir. Bu durumda, kullanıcılar genellikle uygulamayı tercih etmeyebilir ve bu da reytinglerin düşük olmasına neden olabilir. Eğer eğri çok basık ve eğimi fazlaysa, yani güncellemelerle reyting arasında belirgin bir ilişki görülüyorsa, bu durumda reytinglerin güncellemelerle yakından ilişkili olduğunu söyleyebiliriz. Ancak, eğer eğri neredeyse yataysa, yani güncellemelerin reytingler üzerinde belirgin bir etkisi yoksa, bu durumda reytinglerin güncelleme durumuyla ilişkisinin olmadığını düşünebiliriz.

In [192…
```
from warnings import filterwarnings
filterwarnings("ignore")
```

#Bu kod, Python'da bir uyarı filtresi etkinleştirir. "ignore" parametresi, tüm uyarıları görmezden gelmek için filtreleme yapar. Bu, program çalışırken uyarıların konsola yazdırılmasını engeller.

In [193…
```
data.head()
```

Out[193]:

	App	Category	Rating	Reviews	Size	Installs	Price	Genres	Last Updated	Current Ver	Android Ver	Free	Content Rating_Adults only 18+	Content Rating_Everyone	Content Rating_Everyone 10+	Content Rating_Mature 17+	Content Rating_Teen	new	lastupdate
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000000.0	2798	0.0	Art & Design	January 7, 2018	1.0.0	4.0.3 and up	1	False	True	False	False	False	2018-01-07	-213
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000000.0	4951	0.0	Art & Design	January 15, 2018	2.0.0	4.0.3 and up	1	False	True	False	False	False	2018-01-15	-205
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8700000.0	7279	0.0	Art & Design	August 1, 2018	1.2.4	4.0.3 and up	1	False	True	False	False	False	2018-08-01	-7
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25000000.0	8820	0.0	Art & Design	June 8, 2018	Varies with device	4.2 and up	1	False	False	False	False	True	2018-06-08	-61
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2800000.0	4414	0.0	Art & Design	June 20, 2018	1.1	4.4 and up	1	False	True	False	False	False	2018-06-20	-49

In [194…
```
data.info()
```

#Bu kod, veri çerçevesinin bilgi özetini sağlar. Bu özet, her bir sütunun adını, toplamda kaç satır olduğunu ve her bir sütunda kaç değerin olduğunu gösterir.

```
<class 'pandas.core.frame.DataFrame'>
Index: 9359 entries, 0 to 10840
Data columns (total 19 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   App                             9359 non-null   object
 1   Category                        9359 non-null   object
 2   Rating                          9359 non-null   float64
 3   Reviews                         9359 non-null   int64
 4   Size                            9359 non-null   float64
 5   Installs                        9359 non-null   int64
 6   Price                           9359 non-null   float64
 7   Genres                          9359 non-null   object
 8   Last Updated                    9359 non-null   object
 9   Current Ver                     9359 non-null   object
 10  Android Ver                     9359 non-null   object
 11  Free                            9359 non-null   int64
 12  Content Rating_Adults only 18+  9359 non-null   bool
 13  Content Rating_Everyone         9359 non-null   bool
 14  Content Rating_Everyone 10+     9359 non-null   bool
 15  Content Rating_Mature 17+       9359 non-null   bool
 16  Content Rating_Teen             9359 non-null   bool
 17  new                             9359 non-null   datetime64[ns]
 18  lastupdate                      9359 non-null   int64
dtypes: bool(5), datetime64[ns](1), float64(3), int64(4), object(6)
memory usage: 1.4+ MB
```

In [195…
```
from sklearn.model_selection import train_test_split
```

In [196…
```
X_train, X_test, y_train, y_test = train_test_split(data.iloc[:, :-1], data.iloc[:, -1], test_size=0.2)
```

In [197…
```
y_train
```

Out[197]:
```
4442     -90
7006    -173
182      -20
3197     -21
9247   -1170
        ...
743      -24
253      -23
9810     -83
8442      -6
6731    -746
Name: lastupdate, Length: 7487, dtype: int64
```

In [198…
```
X_train
```

Out[198]:

	App	Category	Rating	Reviews	Size	Installs	Price	Genres	Last Updated	Current Ver	Android Ver	Free	Content Rating_Adults only 18+	Content Rating_Everyone	Content Rating_Everyone 10+	Content Rating_Mature 17+	Content Rating_Teen	new
4442	Verdad o Reto	SOCIAL	3.8	826	5.100000e+06	4951	0.0	Social	May 10, 2018	2.0	4.1 and up	1	False	False	False	False	True	2018-05-10
7006	3D Color by Number: Voxel, Unicorn, Pixel Art 3D	FAMILY	4.3	2017	3.015551e+07	4414	0.0	Entertainment	February 16, 2018	2.4	4.1 and up	1	False	True	False	False	False	2018-02-16
182	Golden Dictionary (EN-AR)	BOOKS_AND_REFERENCE	4.4	51269	6.100000e+06	6527	0.0	Books & Reference	July 19, 2018	7.0.4.6	4.2 and up	1	False	True	False	False	False	2018-07-19
3197	Choice Hotels	TRAVEL_AND_LOCAL	4.4	17915	2.524500e+07	6527	0.0	Travel & Local	July 18, 2018	Varies with device	Varies with device	1	False	True	False	False	False	2018-07-18
9247	EC SPORTS	SPORTS	5.0	1	6.300000e+06	80	0.0	Sports	May 26, 2015	4.1.1	2.3.3 and up	1	False	True	False	False	False	2015-05-26
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
743	edX - Online Courses by Harvard, MIT & more	EDUCATION	4.6	32381	1.000000e+07	6527	0.0	Education	July 15, 2018	2.15.1	4.1 and up	1	False	False	True	False	False	2018-07-15
253	join.me - Simple Meetings	BUSINESS	4.0	6989	1.490042e+07	6527	0.0	Business	July 16, 2018	4.3.0.508	4.4 and up	1	False	True	False	False	False	2018-07-16
9810	ES Billing System (Offline App)	PRODUCTIVITY	5.0	1	4.200000e+06	445	0.0	Productivity	May 17, 2018	1.0	4.1 and up	1	False	True	False	False	False	2018-05-17
8442	Phone Tracker : Family Locator	SOCIAL	4.3	231446	5.400000e+06	8531	0.0	Social	August 2, 2018	4.81	4.0 and up	1	False	True	False	False	False	2018-08-02
6731	BS Generator	FAMILY	4.2	5	3.300000e+06	445	0.0	Entertainment	July 23, 2016	1.1	4.4 and up	1	False	True	False	False	False	2016-07-23

#Bu kod, Scikit-learn kütüphanesinden train_test_split fonksiyonunu içe aktarır. Bu fonksiyon, veri kümesini eğitim ve test alt kümelerine ayırmak için kullanılır.

#Bu kod, train_test_split fonksiyonunu kullanarak veri çerçevesini (data) girdi özellikleri (X) ve hedef değişkeni (y) olarak ayırır. Özellikler X_train ve X_test alt kümelerine atanırken, hedef değişkeni y_train ve y_test alt kümelerine atanır.

#Bu kod, y_train alt kümesindeki hedef değişkeni değerlerini ekrana yazdıracaktır. Bu değerler, eğitim veri kümesinin hedef değişkeni değerlerini içerir.

#X_train alt kümesi, eğitim veri kümesinin özelliklerini içerir. Yani, bu alt küme eğitim veri kümesinin girdi özelliklerini temsil eder. Bu özellikler, makine öğrenimi modelinin eğitimini yaparken kullanılır.

In [199]:
```
y_train.value_counts()
```

Out[199]:
```
lastupdate
-5       246
-8       230
-6       228
-7       227
-9       147
        ...
-1332      1
-298       1
-1378      1
-339       1
-495       1
Name: count, Length: 1206, dtype: int64
```

#y_train.value_counts() ifadesi, y_train alt kümesindeki hedef değişkeninin benzersiz değerlerinin sayısını hesaplar. Bu, her bir hedef değişken değerinin kaç kez eğitim veri kümesinde bulunduğunu gösterir.

In [200]:
```
y_test.value_counts()
```

Out[200]:
```
lastupdate
-5       73
-6       56
-9       52
-8       49
-7       48
         ..
-1477     1
-941      1
-415      1
-362      1
-1737     1
Name: count, Length: 626, dtype: int64
```

#y_test.value_counts() ifadesi, y_test alt kümesindeki hedef değişkeninin benzersiz değerlerinin sayısını hesaplar. Bu, her bir hedef değişken değerinin test veri kümesinde kaç kez bulunduğunu gösterir.

In [201]:
```
import xgboost as xgb
import numpy as np
from sklearn.metrics import accuracy_score, confusion_matrix
```

#Bu kod, XGBoost kütüphanesini (xgboost olarak içe aktarılır), Numpy kütüphanesini (numpy olarak içe aktarılır) ve doğruluk puanı (accuracy_score) ile karışıklık matrisi (confusion_matrix) gibi metrikleri hesaplamak için Scikit-learn kütüphanesinden gerekli fonksiyonları içe aktarır.Bu kod, XGBoost algoritması ile model eğitimi yapmak ve bu modelin performansını değerlendirmek için kullanılır.

In [210]:
```
xgb_cls = xgb.XGBClassifier(objective="multiclass:softmax", num_class=3)
```

#xgb.XGBClassifier() fonksiyonu, bir XGBoost sınıflandırıcı nesnesi oluşturur. Bu fonksiyon, sınıflandırma problemleri için XGBoost algoritmasını kullanır. Sınıflandırma problemlerinde, sınıflandırıcı, girdi özelliklerine dayanarak verileri farklı sınıflara ayırmak için kullanılır.

In [211]:
```
data.isna().sum()
```

Out[211]:
```
App                                0
Category                           0
Rating                             0
Reviews                            0
Size                               0
Installs                           0
Price                              0
Genres                             0
Last Updated                       0
Current Ver                        0
Android Ver                        0
Free                               0
Content Rating_Adults only 18+     0
Content Rating_Everyone            0
Content Rating_Everyone 10+        0
Content Rating_Mature 17+          0
Content Rating_Teen                0
new                                0
lastupdate                         0
dtype: int64
```

#data.isna().sum() ifadesi her bir sütundaki eksik değerlerin toplam sayısını verir. Bu sayılar, her bir sütunda bulunan eksik değerlerin sayısını gösterir. Bu bilgi, veri setinin eksik değerlerinin durumunu anlamak için kullanılır.

In [212…
```
data.dtypes
```

Out[212]:
```
App                                object
Category                           object
Rating                            float64
Reviews                             int64
Size                              float64
Installs                            int64
Price                             float64
Genres                             object
Last Updated                       object
Current Ver                        object
Android Ver                        object
Free                                int64
Content Rating_Adults only 18+       bool
Content Rating_Everyone              bool
Content Rating_Everyone 10+          bool
Content Rating_Mature 17+            bool
Content Rating_Teen                  bool
new                        datetime64[ns]
lastupdate                          int64
dtype: object
```

In [213…
```
data.head()
```

Out[213]:

	App	Category	Rating	Reviews	Size	Installs	Price	Genres	Last Updated	Current Ver	Android Ver	Free	Content Rating_Adults only 18+	Content Rating_Everyone	Content Rating_Everyone 10+	Content Rating_Mature 17+	Content Rating_Teen	new	lastupdate
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000000.0	2798	0.0	Art & Design	January 7, 2018	1.0.0	4.0.3 and up	1	False	True	False	False	False	2018-01-07	-213
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000000.0	4951	0.0	Art & Design	January 15, 2018	2.0.0	4.0.3 and up	1	False	True	False	False	False	2018-01-15	-205
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8700000.0	7279	0.0	Art & Design	August 1, 2018	1.2.4	4.0.3 and up	1	False	True	False	False	False	2018-08-01	-7
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25000000.0	8820	0.0	Art & Design	June 8, 2018	Varies with device	4.2 and up	1	False	False	False	False	True	2018-06-08	-61
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2800000.0	4414	0.0	Art & Design	June 20, 2018	1.1	4.4 and up	1	False	True	False	False	False	2018-06-20	-49

In [214…
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

#Bu kod, veri çerçevesindeki her bir sütunun adını ve o sütundaki değerlerin veri türünü (int, float, object, vb.) içeren bir çıktı döndürecektir. Bu bilgi, veri çerçevesindeki her bir sütunun veri türünü anlamak için kullanılır ve bu, veri çerçevesinin analiz edilmesi ve işlenmesi sırasında önemlidir.

#train_test_split fonksiyonu ile veri kümesi eğitim ve test alt kümelerine ayrılır. Daha sonra, LinearRegression sınıfından bir model oluşturulur, eğitim veri kümesiyle (X_train, y_train) model eğitilir ve son olarak test veri kümesiyle (X_test) tahminler yapılır.

In [216…
```
# Bağımsız ve bağımlı değişkenleri ayırma
X = data[['Rating']]
y = data['Rating']
```

In [217…
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

#Bu örnekte, X ve y veri kümeleri %70 eğitim ve %30 test alt kümelerine ayrılır. Ayrıca, random_state=42 parametresi, rastgele bölünme işleminin tekrarlanabilirliğini sağlar.

In [218…
```
model = LinearRegression()
model.fit(X_train, y_train)
```

Out[218]:
```
▾ LinearRegression

LinearRegression()
```

LinearRegression() sınıfından bir doğrusal regresyon modeli oluşturulur ve ardından X_train ve y_train veri kümeleriyle bu model eğitilir. Model, eğitildikten sonra, girdi özelliklerini kullanarak hedef değişkeni tahmin etmek için kullanılabilir.

In [219…
```
from sklearn.metrics import mean_squared_error
```

#mean_squared_error fonksiyonu, gerçek hedef değerleri (y_test) ve modelin tahminlerini (y_pred) alarak MSE'yi hesaplar. Sonuç olarak, ortalama karesel hata (MSE) değeri elde edilir ve ekrana yazdırılır. MSE, modelin performansını değerlendirmek için kullanılan bir metrik olup, ne kadar küçük olursa modelin tahminlerinin gerçek değerlere o kadar yakın olduğunu gösterir.

In [220…
```
y_pred = model.predict(X_test)
```

#y_pred değişkeni, modelin test veri kümesindeki girdi özelliklerine dayalı olarak yaptığı tahminleri içerir. Bu tahminler, modelin test veri kümesindeki girdi özelliklerine dayalı olarak hedef değişkenin tahmin edilen değerleridir.

In [221…
```
mse = mean_squared_error(y_test, y_pred)
print('Mean Squared Error:', mse)
```

```
Mean Squared Error: 7.392059319311921e-33
```

In [224…
```
#Ide ettiğimiz MSE değeri 7.392059319311921e-33, yani çok küçük bir değerdir.
#Bu, modelinizin tahminlerinin gerçek değerlerden oldukça yakın olduğunu gösterir
#Bu sonuç, modelin oldukça iyi performans gösterdiğini ve veri setinizdeki özelliklerin uygulama derecelendirmelerini
#tahmin etmede oldukça etkili olduğunu gösterir.
#Ancak, bu sonuca dayanarak modeli mutlaka başarılı olarak kabul etmemeliyiz.
#Diğer performans metriklerini de dikkate almalıyız.
```

#Bu kod, modelin test veri kümesindeki tahminlerini (y_pred) ve gerçek hedef değerlerini (y_test) alarak MSE'yi hesaplar ve hesaplanan MSE değerini ekrana yazdırır. Bu şekilde, modelin performansını değerlendirmek için MSE değerini görebiliriz.

In [42]:
```
import pandas as pd
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
```

In [43]:
```
df=pd.read_csv('Audi_A1_listings (2).csv')
```

In [44]:
```
df.head(5)
```

Out[44]:

	index	Year	Type	Mileage(miles)	Engine	PS	Transmission	Fuel	Number_of_Owners	Price(£)	href	PPY	MileageRank	PriceRank	PPYRank	Score
0	0	2018.0	Hatchback	44000.0	1.6L	114.398422	Manual	Diesel	1	14995.0	https://www.autotrader.co.uk/car-details/20221...	2499.166667	215	163	340	718
1	4	2016.0	Hatchback	42596.0	1.0L	93.688363	Manual	Petrol	3	10755.0	https://www.autotrader.co.uk/car-details/20221...	2688.750000	222	330	276	828
2	7	2015.0	Hatchback	42700.0	1.4L	123.274162	Manual	Petrol	2	10799.0	https://www.autotrader.co.uk/car-details/20221...	3599.666667	221	327	94	642
3	11	2014.0	Hatchback	86000.0	1.6L	103.550296	Manual	Diesel	3	7490.0	https://www.autotrader.co.uk/car-details/20221...	3745.000000	41	449	83	573
4	12	2014.0	Hatchback	104310.0	1.6L	103.550296	Manual	Diesel	3	7400.0	https://www.autotrader.co.uk/car-details/20220...	3700.000000	12	452	85	549

In [45]:
```
df=df.drop(columns=['index','href','PPY','MileageRank','PriceRank','PPYRank','Score'])
```

In [46]:
```
df.head(3)
```

Out[46]:

	Year	Type	Mileage(miles)	Engine	PS	Transmission	Fuel	Number_of_Owners	Price(£)
0	2018.0	Hatchback	44000.0	1.6L	114.398422	Manual	Diesel	1	14995.0
1	2016.0	Hatchback	42596.0	1.0L	93.688363	Manual	Petrol	3	10755.0
2	2015.0	Hatchback	42700.0	1.4L	123.274162	Manual	Petrol	2	10799.0

#veri setini yükleme, özellikler ve hedef değişken olarak ayırma, eğitim ve test veri kümelerine ayırma, doğrusal regresyon modeli oluşturma, modeli eğitim veri kümesiyle eğitme, test veri kümesi üzerinde tahminler yapma ve model performansını değerlendirme adımlarını içerir.

#Audi_A1 veri setin yükledik,Train test yapılmadan önceki tahmini ve train test yapıldıktan sonraki tahmini karşılaştıracağız.

#belirtilen sütunları Fiyat tahminine etkisinin olduğunu düşünmedeğimiz için veri setimizden çıkardık.

In [47]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 471 entries, 0 to 470
Data columns (total 9 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Year              471 non-null    float64
 1   Type              471 non-null    object 
 2   Mileage(miles)    471 non-null    float64
 3   Engine            471 non-null    object 
 4   PS                471 non-null    float64
 5   Transmission      471 non-null    object 
 6   Fuel              471 non-null    object 
 7   Number_of_Owners  471 non-null    int64  
 8   Price(£)          471 non-null    float64
dtypes: float64(4), int64(1), object(4)
memory usage: 33.2+ KB
```

In [48]:
```
df.head(3)
```

Out[48]:

	Year	Type	Mileage(miles)	Engine	PS	Transmission	Fuel	Number_of_Owners	Price(£)
0	2018.0	Hatchback	44000.0	1.6L	114.398422	Manual	Diesel	1	14995.0
1	2016.0	Hatchback	42596.0	1.0L	93.688363	Manual	Petrol	3	10755.0
2	2015.0	Hatchback	42700.0	1.4L	123.274162	Manual	Petrol	2	10799.0

In [49]:
```
df.columns=['yil','kasa','mil','motor','ps','vites','yakit','sahip','fiyat']
```

In [50]:
```
df['motor']=df['motor'].str.replace('L','')
```

In [51]:
```
df['motor']
```

Out[51]:
```
0      1.6
1      1.0
2      1.4
3      1.6
4      1.6
      ...
466    1.4
467    1.0
468    1.4
469    1.0
470    1.0
Name: motor, Length: 471, dtype: object
```

veri setinizin sütunlarını yeniden adlandırdınız. Yeni sütun isimleri aşağıdaki gibi oldu:

'yil': Araç Yılı
'kasa': Araç Kasası
'mil': Kilometre
'motor': Motor Hacmi
'ps': Beygir Gücü
'vites': Vites Türü
'yakit': Yakıt Türü
'sahip': Sahip Sayısı
'fiyat': Fiyat

motor sütunundaki değerlerden 'L' karakterini kaldırdık. 'L' karakterini kaldırarak, veri setinizdeki 'motor' sütununun sayısal bir formatta kalmasını sağladık. Bu, veri setinizdeki motor hacmi değerlerini daha kolay işleyebilir ve analiz edebilmemizi sağlar.

In [52]:
```
df['motor']=pd.to_numeric(df['motor'])
```

#motor hacmi değerlerinin artık sayısal bir formatta saklanacağı anlamına gelir

In [53]:
```
df.head(3)
```

Out[53]:

	yil	kasa	mil	motor	ps	vites	yakit	sahip	fiyat
0	2018.0	Hatchback	44000.0	1.6	114.398422	Manual	Diesel	1	14995.0
1	2016.0	Hatchback	42596.0	1.0	93.688363	Manual	Petrol	3	10755.0
2	2015.0	Hatchback	42700.0	1.4	123.274162	Manual	Petrol	2	10799.0

In [54]:
```
df=pd.get_dummies(df,columns=['kasa','vites','yakit'],drop_first=True)
```

'kasa', 'vites' ve 'yakit' sütunlarındaki kategorik değerleri ikili sınıflandırıcıya dönüştürdük ve ardından bu kategorik sütunları ikili sınıflandırıcıya dönüştürmek için one-hot encoding uyguladık. drop_first=True parametresini kullanarak, her bir kategorik değişken için ilk sınıfı düşürdük

In [55]:
```
y=df[('fiyat')]
x=df.drop('fiyat',axis=1)
```

veri setinizdeki 'fiyat' sütununu hedef değişken olarak belirlediniz ve diğer sütunları bağımsız değişkenler olarak belirlediniz

In [57]:
```
x_eğitim, x_test, y_eğitim, y_test = train_test_split(x, y, train_size=0.70, random_state=20)
```

veri setinizi eğitim ve test alt kümelerine ayırdık. Bu kod, veri setinin %70'ini eğitim veri kümesi olarak ve %30'unu test veri kümesi olarak kullanacak şekilde böler. Ayrıca, veri setinin rastgele bölünmesi sırasında kullanılan rastgele durumu (random_state) belirledik.%91 Ttain test öncesindeki güvenilirliktir.

In [34]:
```
#lm=LinearRegression()
#model=lm.fit(x,y)
#model.predict([[2017,30000,1.6,110,1,2600,1]])
#model.score(x,y)
#0.9110559159967577 train test önceki skor
```

In [81]:
```
lm=LinearRegression()
model=lm.fit(x_eğitim,y_eğitim)
model.score(x_test,y_test)
#testle modelin skoru düştü ama daha güvenilir bir sonuç oldu
```

Out[81]:
```
0.8940133433550822
```

doğrusal regresyon modelinizi eğittik ve eğitim veri kümenizdeki bağımsız değişkenlerle hedef değişken arasındaki ilişkiyi öğrendik. Ardından, modeli test veri kümesi ile değerlendirdik ve belirli bir skor elde ettik.%89 civarında.skor düştü ama güvenirlik arttı. Skor ne kadar yüksek olursa, modelin test veri kümesindeki performansı o kadar iyidir.Fakat güvenilir olması da gerekir.

In [82]:
```
df.head(3)
```

Out[82]:

	yil	mil	motor	ps	sahip	fiyat	vites_Manual	yakit_Petrol
0	2018.0	44000.0	1.6	114.398422	1	14995.0	True	False
1	2016.0	42596.0	1.0	93.688363	3	10755.0	True	True
2	2015.0	42700.0	1.4	123.274162	2	10799.0	True	True

In [83]:
```
y=df[('fiyat')]
x=df.drop('fiyat',axis=1)
```

veri setinde 'fiyat' sütununu hedef değişken olarak belirledik ve diğer sütunları bağımsız değişkenler olarak belirledik.

In [84]:
```
model.predict([[2016,30000,1,30,1,1,1]])
```

Out[84]:
```
C:\Users\Gamze\anaconda3\Lib\site-packages\sklearn\base.py:464: UserWarning: X does not have valid feature names, but LinearRegression was fitted with feature names
  warnings.warn(
array([8035.02732457])
```

Bu kod, eğitilmiş modeli kullanarak belirli bir örneği tahmin etmemizi sağlar. Örneğin, veri setinizdeki özelliklere (yıl, mil, motor, vb.) dayalı olarak bir fiyat tahmini yapmak istedik.Sonuç8035.027 $ çıktı.