

<https://www.kaggle.com/datasets/lava18/google-play-store-apps>
(<https://www.kaggle.com/datasets/lava18/google-play-store-apps>)

In [2]:

```
1 import numpy as np
2 import pandas as pd
3 import matplotlib.pyplot as plt
4 import seaborn as sns
5 %matplotlib inline
6 import warnings
7 warnings.filterwarnings('ignore')
```

In [4]:

```
1 df=pd.read_csv("clean_gpaydata.csv")
2 df.head(5)
```

Out[4]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19.0	10000	Free	0.0	Everyone	
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14.0	500000	Free	0.0	Everyone	De
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8.7	5000000	Free	0.0	Everyone	
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25.0	50000000	Free	0.0	Teen	
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2.8	100000	Free	0.0	Everyone	Des

In [5]:

```
1 # check null values
2 df.isnull().sum()
```

Out[5]:

App	0
Category	0
Rating	1474
Reviews	0
Size	1695
Installs	0
Type	1
Price	0
Content Rating	0
Genres	0
Last Updated	0
Current Ver	8
Android Ver	2
day	0
date	0
month	0
year	0

dtype: int64

In [7]:

```
1 # check total null values
2 df.isnull().sum().sum()
```

Out[7]:

3180

In [11]:

```
1 # sort null values in descending order
2 df.isnull().sum().sort_values(ascending=False)
```

Out[11]:

Size	1695
Rating	1474
Current Ver	8
Android Ver	2
Type	1
App	0
Last Updated	0
month	0
date	0
day	0
Content Rating	0
Genres	0
Category	0
Price	0
Installs	0
Reviews	0
year	0

dtype: int64

show null values in count plot

In [14]:

```
1 df.shape
```

Out[14]:

(10840, 17)

In [16]:

```
1 # drop duplicate values
2 df = df.drop_duplicates()
```

In [17]:

```
1 df.shape
```

Out[17]:

(10357, 17)

In [18]:

```
1 len(df)
```

Out[18]:

10357

plot bar plot

In [19]:

```
1 null_counts=df.isnull().sum().sort_values(ascending=False)/len(df)
```

In [20]:

```
1 null_counts
```

Out[20]:

```
Size          0.147340
Rating        0.141450
Current Ver   0.000772
Android Ver   0.000193
Type          0.000097
App           0.000000
Last Updated  0.000000
month         0.000000
date          0.000000
day           0.000000
Content Rating 0.000000
Genres        0.000000
Category      0.000000
Price         0.000000
Installs      0.000000
Reviews       0.000000
yrrar         0.000000
dtype: float64
```

In [24]:

```
1 len(null_counts)
```

Out[24]:

```
17
```

In [115]:

```
1 null_counts.index
```

Out[115]:

```
Index(['Size', 'Rating', 'Current Ver', 'Android Ver', 'Type', 'App',
      'Last Updated', 'month', 'date', 'day', 'Content Rating', 'Genres',
      'Category', 'Price', 'Installs', 'Reviews', 'yrrar'],
      dtype='object')
```

In [116]:

```
1 np.arange(len(null_counts))
```

Out[116]:

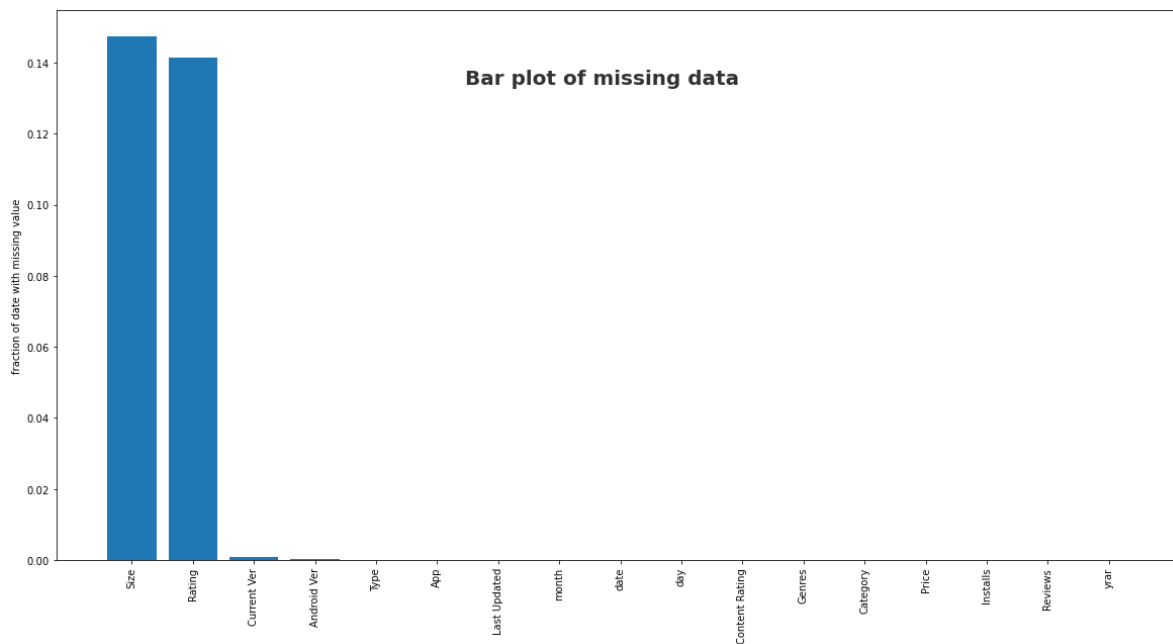
```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16])
```

In [114]:

```

1 plt.figure(figsize=(20,10))
2 plt.suptitle("Bar plot of missing data" ,fontsize= 20,fontweight= "bold",alpha=0.8,y=0.
3
4 # we will get the sum of percentage count by multiplying len(df)
5
6 null_counts=df.isnull().sum().sort_values(ascending= False)/len(df)
7
8 plt.xticks(np.arange(len(null_counts)),null_counts.index,rotation ="vertical")
9 plt.ylabel("fraction of date with missing value")
10
11 plt.bar(np.arange(len(null_counts)),null_counts)
12 plt.show()
13
14 # Ask in doubt clearning session

```



In [31]:

```

1 # create a copy of dataa
2 df_copy= df.copy()

```

In [32]:

```
1 df_copy.head(3)
```

Out[32]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19.0	10000	Free	0.0	Everyone	Art and Design
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14.0	500000	Free	0.0	Everyone	Design
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8.7	5000000	Free	0.0	Everyone	Art and Design

get a columns which has null values

In [38]:

```
1 cols= [fea for fea in df_copy.columns if df_copy[fea].isnull().mean()*100]
```

In [40]:

```
1 cols
```

Out[40]:

```
['Rating', 'Size', 'Type', 'Current Ver', 'Android Ver']
```

In [42]:

```
1 # five columns which has null values
2 len(cols)
```

Out[42]:

5

In [46]:

```
1 df_copy["Rating"].isnull().mean()
```

Out[46]:

0.14145022689968137

In [47]:

```
1 # to get %  
2 df_copy["Rating"].isnull().mean()*100
```

Out[47]:

14.145022689968137

In [49]:

```
1 # cols having no null value  
2 df_copy["Installs"].isnull().mean()*100
```

Out[49]:

0.0

in python any numeric value less than 0 or greater than 0 but not equal to zero gives true value

In [50]:

```
1 if True :  
2     print("shweta")  
3
```

shweta

In [52]:

```
1 if False :  
2     print("shweta")
```

In [45]:

```
1 # meaning of above line no 38
2 # to get all column names
3 for i in df_copy.columns:
4     print(i)
```

App
Category
Rating
Reviews
Size
Installs
Type
Price
Content Rating
Genres
Last Updated
Current Ver
Android Ver
day
date
month
year

In [53]:

```
1 cols= [var for var in df_copy.columns if df_copy[var].isnull().mean()*100]
2 cols
```

Out[53]:

['Rating', 'Size', 'Type', 'Current Ver', 'Android Ver']

In [54]:

```
1 # data frame only contains null values
2 df_copy[cols]
```

Out[54]:

	Rating	Size	Type	Current Ver	Android Ver
0	4.1	19.0	Free	1.0.0	4.0.3 and up
1	3.9	14.0	Free	2.0.0	4.0.3 and up
2	4.7	8.7	Free	1.2.4	4.0.3 and up
3	4.5	25.0	Free	Varies with device	4.2 and up
4	4.3	2.8	Free	1.1	4.4 and up
...
10835	4.5	53.0	Free	1.48	4.1 and up
10836	5.0	3.6	Free	1.0	4.1 and up
10837	NaN	9.5	Free	1.0	2.2 and up
10838	4.5	NaN	Free	Varies with device	Varies with device
10839	4.5	19.0	Free	Varies with device	Varies with device

10357 rows × 5 columns

In [55]:

```
1 df_copy[cols].dropna()
```

Out[55]:

	Rating	Size	Type	Current Ver	Android Ver
0	4.1	19.000	Free	1.0.0	4.0.3 and up
1	3.9	14.000	Free	2.0.0	4.0.3 and up
2	4.7	8.700	Free	1.2.4	4.0.3 and up
3	4.5	25.000	Free	Varies with device	4.2 and up
4	4.3	2.800	Free	1.1	4.4 and up
...
10832	4.8	0.619	Free	0.8	2.2 and up
10833	4.0	2.600	Free	1.0.0	4.1 and up
10835	4.5	53.000	Free	1.48	4.1 and up
10836	5.0	3.600	Free	1.0	4.1 and up
10839	4.5	19.000	Free	Varies with device	Varies with device

7418 rows × 5 columns

In [56]:

```
1 # assign new variable
2 drope_df=df_copy[cols].dropna()
```

In [57]:

```
1 drope_df.shape
```

Out[57]:

(7418, 5)

In [58]:

```
1 df_copy.shape
```

Out[58]:

(10357, 17)

In []:

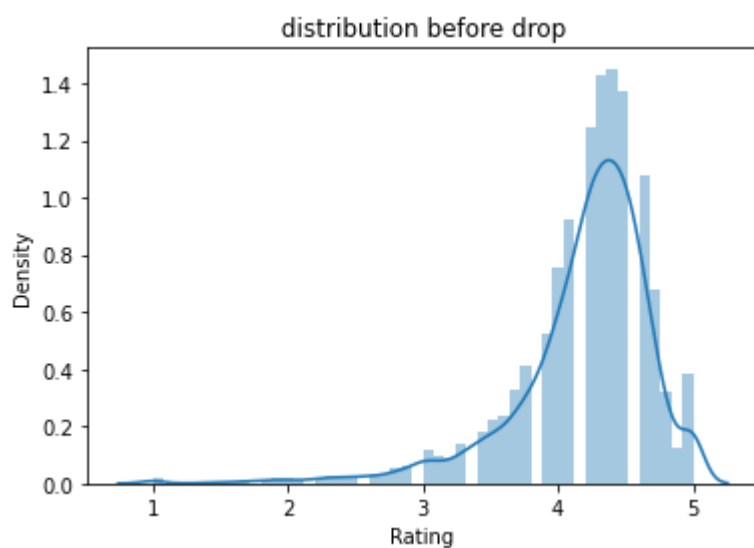
```
1 # after dropping the null value check the distribution of data
```

In [60]:

```
1 sns.distplot(df_copy["Rating"])  
2 plt.title(" distribution before drop ")
```

Out[60]:

Text(0.5, 1.0, ' distribution before drop ')

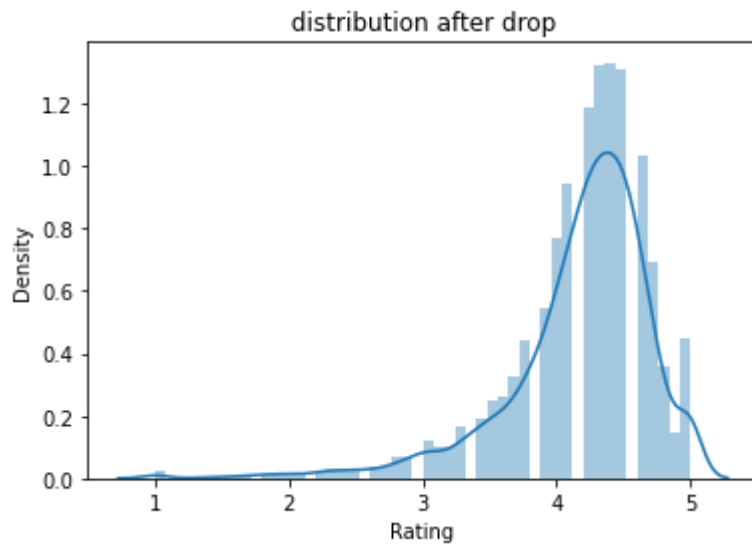


In [62]:

```
1 sns.distplot(drope_df["Rating"])
2 plt.title(" distribution after drop ")
```

Out[62]:

Text(0.5, 1.0, ' distribution after drop ')



Observations

Distribution does not change after drope

Mean Median and Mode methode

In [67]:

```
1 # create a copy of dataframe
2 df_copy_MEANMODE=df.copy()
3
```

In [69]:

```
1 df_copy_MEANMODE
```

Out[69]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19.0	10000	Free	0.0	E
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14.0	500000	Free	0.0	E
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8.7	5000000	Free	0.0	E
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25.0	50000000	Free	0.0	
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2.8	100000	Free	0.0	E
...	
10835	Sya9a Maroc - FR	FAMILY	4.5	38	53.0	5000	Free	0.0	E
10836	Fr. Mike Schmitz Audio Teachings	FAMILY	5.0	4	3.6	100	Free	0.0	E
10837	Parkinson Exercices FR	MEDICAL	NaN	3	9.5	1000	Free	0.0	E
10838	The SCP Foundation DB fr nn5n	BOOKS_AND_REFERENCE	4.5	114	NaN	1000	Free	0.0	
10839	iHoroscope - 2018 Daily Horoscope & Astrology	LIFESTYLE	4.5	398307	19.0	10000000	Free	0.0	E

10357 rows × 17 columns



In [71]:

```

1 # fill the null values in size column with mean
2 #df_copy_MEANMODE['Size'].fillna()-----Pass mean inside()
3 df_copy_MEANMODE['Size'].fillna(df_copy_MEANMODE['Size'].mean())

```

Out[71]:

```

0      19.00000
1      14.00000
2       8.70000
3      25.00000
4       2.80000
...
10835   53.00000
10836    3.60000
10837    9.50000
10838   21.28875
10839   19.00000
Name: Size, Length: 10357, dtype: float64

```

In [75]:

```

1 # check null values
2 df_copy_MEANMODE['Size'].fillna(df_copy_MEANMODE['Size'].mean()).isnull().sum()

```

Out[75]:

0

fill with median

In [76]:

```

1 df_copy_MEANMODE['Size'].fillna(df_copy_MEANMODE['Size'].median())

```

Out[76]:

```

0      19.0
1      14.0
2       8.7
3      25.0
4       2.8
...
10835   53.0
10836    3.6
10837    9.5
10838   13.0
10839   19.0
Name: Size, Length: 10357, dtype: float64

```

In [77]:

```

1 df_copy_MEANMODE['Size'].fillna(df_copy_MEANMODE['Size'].median()).isna().sum()

```

Out[77]:

0

In []:

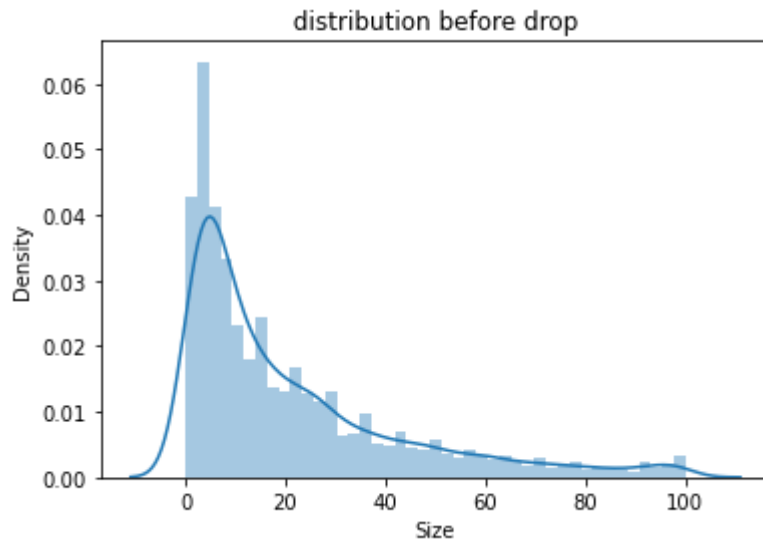
```
1 # check the distribution
2
```

In [78]:

```
1 sns.distplot(df_copy["Size"])
2 plt.title(" Original distribution before fill ")
```

Out[78]:

Text(0.5, 1.0, ' distribution before drop ')

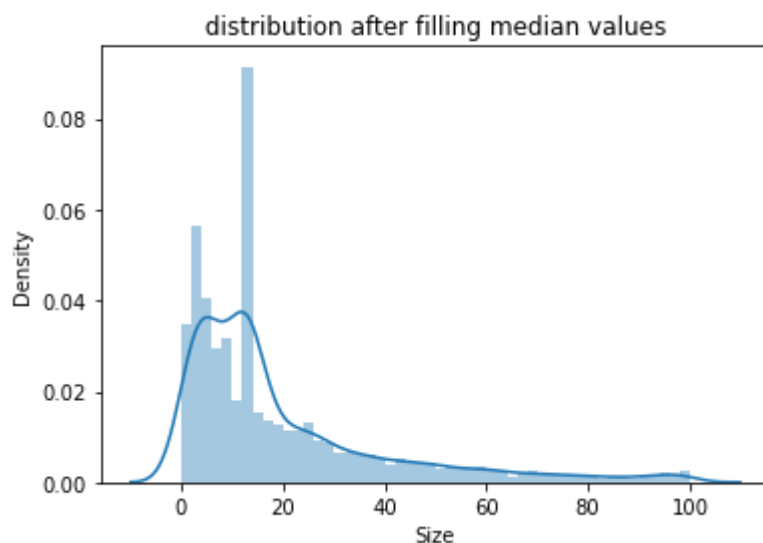


In [80]:

```
1 sns.distplot(df_copy_MEANMODE['Size'].fillna(df_copy_MEANMODE['Size'].median()))
2 plt.title(" distribution after filling median values ")
```

Out[80]:

Text(0.5, 1.0, ' distribution after filling median values ')

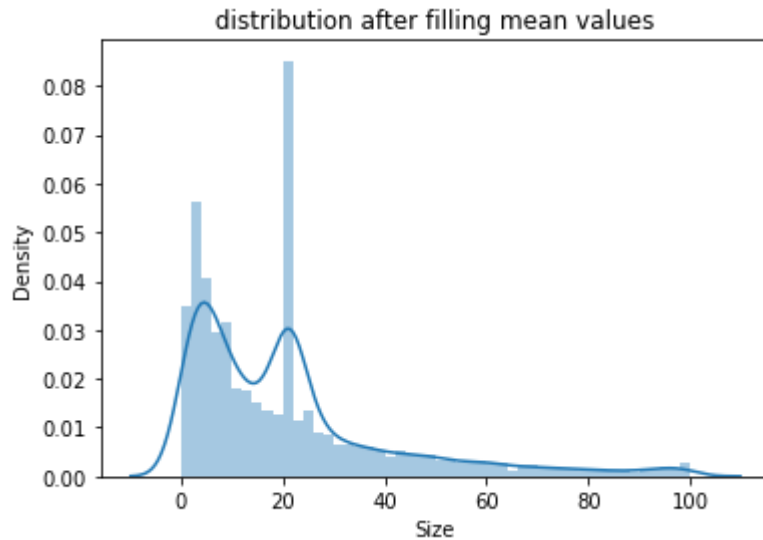


In [81]:

```
1 sns.distplot(df_copy_MEANMODE['Size'].fillna(df_copy_MEANMODE['Size'].mean()))
2 plt.title(" distribution after filling mean values ")
```

Out[81]:

Text(0.5, 1.0, ' distribution after filling mean values ')



Distribution is going to change

Handle missing values by random sample imputation

with respect to size

In [99]:

```

1 # create a copy of dataframe
2 df_RANDOMIPU=df.copy()
3 df_RANDOMIPU.head(3)

```

Out[99]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19.0	10000	Free	0.0	Everyone	Art and Design
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14.0	500000	Free	0.0	Everyone	Design
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8.7	5000000	Free	0.0	Everyone	Art and Design

In [100]:

```

1 #
2 df_RANDOMIPU["Size"].isna().sum()

```

Out[100]:

1526

In [101]:

```

1 # collect the sample from this data but first drop the null values
2 df_RANDOMIPU["Size"].dropna().sample() # its giving one sample

```

Out[101]:

```

10470    4.0
Name: Size, dtype: float64

```


In [102]:

```
1 # we need collect the samples as many as the null values in the data as null values are  
2 df_RANDOMIPU["Size"].dropna().sample(1526)
```

Out[102]:

```
8421    27.0  
6113    80.0  
4057    18.0  
3544    11.0  
7701     3.3  
...  
8288     8.7  
7623    64.0  
6196     4.9  
460     1.6  
4750    16.0
```

Name: Size, Length: 1526, dtype: float64

In [89]:

```
1 random_samplecollect=df_RANDOMIPU["Size"].dropna().sample(1526)
```

In [90]:

```
1 random_samplecollect
```

Out[90]:

```
7421    92.0  
717     21.0  
1337    15.0  
7594    31.0  
8845     2.6  
...  
5520    37.0  
8426    85.0  
9723    47.0  
9169    43.0  
6983    44.0
```

Name: Size, Length: 1526, dtype: float64

In [91]:

```
1 # data frame of null values
2 df_RANDOMPU[df_RANDOMPU["Size"].isnull()]
```

Out[91]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price
37	Floor Plan Creator	ART_AND_DESIGN	4.1	36639	NaN	5000000	Free	0.0
42	Textgram - write on photos	ART_AND_DESIGN	4.4	295221	NaN	10000000	Free	0.0
52	Used Cars and Trucks for Sale	AUTO_AND_VEHICLES	4.6	17057	NaN	1000000	Free	0.0
67	Ulysse Speedometer	AUTO_AND_VEHICLES	4.3	40211	NaN	5000000	Free	0.0
68	REPUVE	AUTO_AND_VEHICLES	3.9	356	NaN	100000	Free	0.0
...
10712	My Earthquake Alerts - US & Worldwide Earthquakes	WEATHER	4.4	3471	NaN	100000	Free	0.0
10724	Posta App	MAPS_AND_NAVIGATION	3.6	8	NaN	1000	Free	0.0
10764	Chat For Strangers - Video Chat	SOCIAL	3.4	622	NaN	100000	Free	0.0
10825	Frim: get new friends on local chat rooms	SOCIAL	4.0	88486	NaN	5000000	Free	0.0
10838	The SCP Foundation DB fr nn5n	BOOKS_AND_REFERENCE	4.5	114	NaN	1000	Free	0.0

1526 rows × 17 columns



In [93]:

```
1 # get the indexes having null values of above frame
2 df_RANDOMPU[df_RANDOMPU["Size"].isnull()].index
```

Out[93]:

```
Int64Index([ 37, 42, 52, 67, 68, 73, 85, 88, 89,
            92,
            ...,
            10646, 10678, 10680, 10706, 10711, 10712, 10724, 10764, 10825,
            10838],
            dtype='int64', length=1526)
```

In [103]:

```
1 # take this inside new variable
2 random_samplecollect.index =df_RANDOMPU[df_RANDOMPU["Size"].isnull()].index
```

In []:

```
1 # with the help of location we can fill the null values
2 # take a location df_RANDOMPU.loc[]
3 # pass the particular feature inside it df_RANDOMPU.loc[df_RANDOMPU["Size"]]
4 #Check wheather it is isnull df_RANDOMPU.loc[df_RANDOMPU["Size"].isnull()]
5 # fill the null values with random samples df_RANDOMPU.loc[df_RANDOMPU["Size"].isnull()]
6
```

In [106]:

```
1 df_RANDOMPU.loc[df_RANDOMPU["Size"].isnull(),"Size"] = random_samplecollect
```

In [109]:

```
1 df_RANDOMPU.loc[df_RANDOMPU["Size"].isnull(),"Size"]
```

Out[109]:

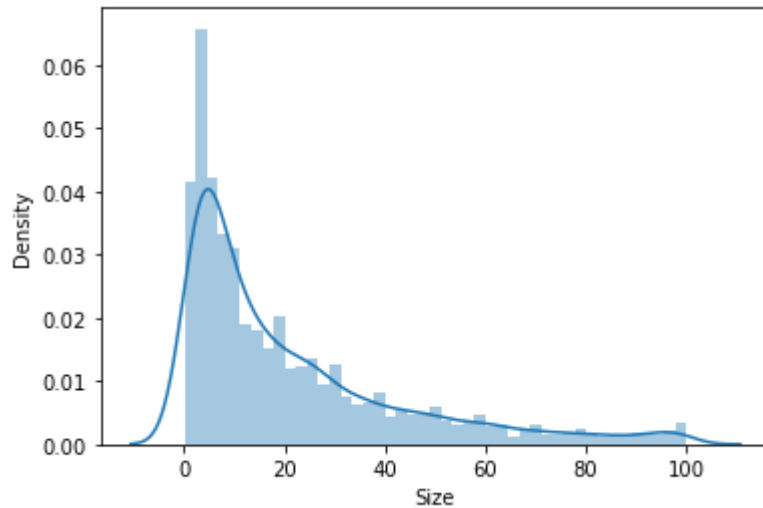
```
Series([], Name: Size, dtype: bool)
```

In [113]:

```
1 # check the distribution after imputation
2 sns.distplot(df_RANDOMIMPU["Size"])
```

Out[113]:

<AxesSubplot:xlabel='Size', ylabel='Density'>



In []:

1

In []:

1

In []:

1

In []:

1

In []:

1

In []:

1

In []:

1

In []:

1

In []:

1

In []:

1

In []:

1

In []:

1

In []:

1

In []:

1

In []:

1