


```
In [1]: import pandas as pd
import plotly.io as pio
import plotly.express as px
```

```
In [2]: pd.options.display.float_format = '{:,.2f}'.format
```

```
In [3]: df_apps = pd.read_csv('apps.csv')
df_apps.head()
```

Out[3]:


	App	Category	Rating	Reviews	Size_MB	Installs	Type	Price	Cor
0	Ak Parti Yardim Toplama	SOCIAL	NaN	0	8.70	0	Paid	\$13.99	
1	Ain Arabic Kids Alif Ba ta	FAMILY	NaN	0	33.00	0	Paid	\$2.99	
2	Popsicle Launcher for Android P 9.0 launcher	PERSONALIZATION	NaN	0	5.50	0	Paid	\$1.49	
3	Command & Conquer: Rivals	FAMILY	NaN	0	19.00	0	NaN	0	E
4	CX Network	BUSINESS	NaN	0	10.00	0	Free	0	



```
In [4]: df_apps.sample(5)
```

Out[4]:

	App	Category	Rating	Reviews	Size_MB	Installs	Type	Price	
8860	MegaN64 (N64 Emulator)	FAMILY	4.60	877576	12.00	10,000,000	Free	0	
4184	Multiple Sclerosis Dx & Mgmt.	MEDICAL	4.50	131	60.00	10,000	Free	0	
8945	Hello Stars	GAME	4.60	101686	31.00	10,000,000	Free	0	
7852	Offroad Car Q	FAMILY	4.10	24668	40.00	1,000,000	Free	0	
989	FO RCBT	COMMUNICATION	NaN	5	15.00	100	Free	0	



```
In [5]: print(df_apps.columns)
```

```
Index(['App', 'Category', 'Rating', 'Reviews', 'Size_MBs', 'Installs', 'Type',
      'Price', 'Content_Rating', 'Genres', 'Last_Updated', 'Android_Ver'],
      dtype='object')
```

```
In [6]: df_apps.drop(['Last_Updated', 'Android_Ver'], axis=1, inplace=True)
df_apps.head()
```

```
Out[6]:
```

	App	Category	Rating	Reviews	Size_MBs	Installs	Type	Price	Cor
0	Ak Parti Yardim Toplama	SOCIAL	NaN	0	8.70	0	Paid	\$13.99	
1	Ain Arabic Kids Alif Ba ta	FAMILY	NaN	0	33.00	0	Paid	\$2.99	
2	Popsicle Launcher for Android P 9.0 launcher	PERSONALIZATION	NaN	0	5.50	0	Paid	\$1.49	
3	Command & Conquer: Rivals	FAMILY	NaN	0	19.00	0	NaN	0	E
4	CX Network	BUSINESS	NaN	0	10.00	0	Free	0	

```
In [7]: nan_rows = df_apps.Rating.isna()
print(nan_rows.shape)
```

```
(10841,)
```

```
In [8]: df_apps_clean = pd.read_csv("apps.csv")
print(df_apps_clean.head())
```

	App	Category	Rating	\
0	Ak Parti Yardım Toplama	SOCIAL	NaN	
1	Ain Arabic Kids Alif Ba ta	FAMILY	NaN	
2	Popsicle Launcher for Android P 9.0 launcher	PERSONALIZATION	NaN	
3	Command & Conquer: Rivals	FAMILY	NaN	
4	CX Network	BUSINESS	NaN	

	Reviews	Size_MBs	Installs	Type	Price	Content_Rating	Genres	\
0	0	8.70	0	Paid	\$13.99	Teen	Social	
1	0	33.00	0	Paid	\$2.99	Everyone	Education	
2	0	5.50	0	Paid	\$1.49	Everyone	Personalization	
3	0	19.00	0	NaN	0	Everyone 10+	Strategy	
4	0	10.00	0	Free	0	Everyone	Business	

	Last_Updated	Android_Ver
0	July 28, 2017	4.1 and up
1	April 15, 2016	3.0 and up
2	July 11, 2018	4.2 and up
3	June 28, 2018	Varies with device
4	August 6, 2018	4.1 and up

```
In [9]: duplicated_rows = df_apps_clean[df_apps_clean.duplicated()]
print(duplicated_rows.shape)
duplicated_rows.head()
```

(483, 12)

Out[9]:

	App	Category	Rating	Reviews	Size_MBs	Installs	Type	Price	Content
--	-----	----------	--------	---------	----------	----------	------	-------	---------

190	RT 516 VET	MEDICAL	NaN	0	29.00	10	Free	0	E
741	Penn State Health OnDemand	MEDICAL	NaN	0	40.00	50	Free	0	E
803	Maricopa AH	MEDICAL	NaN	0	29.00	100	Free	0	E
914	Breastfeeding Tracker Baby Log	MEDICAL	NaN	6	23.00	100	Free	0	E
946	420 BZ Budeze Delivery	MEDICAL	5.00	2	11.00	100	Free	0	Mat

```
In [10]: df_apps_clean = df_apps.dropna()
df_apps_clean.shape
```

Out[10]: (9367, 10)

```
In [11]: df_apps_clean[df_apps_clean.App == 'Instagram']
df_apps_clean = df_apps_clean.drop_duplicates(subset = ['App', 'Type', 'Price'])
df_apps_clean[df_apps_clean.App == 'Instagram']
```

Out[11]:

	App	Category	Rating	Reviews	Size_MBs	Installs	Type	Price	Cc
--	-----	----------	--------	---------	----------	----------	------	-------	----

10806	Instagram	SOCIAL	4.50	66577313	5.30	1,000,000,000	Free	0	
-------	-----------	--------	------	----------	------	---------------	------	---	--


In [12]: `df_apps_clean.shape`

Out[12]: (8199, 10)

In [13]: `df_apps_clean.head()`

Out[13]:

	App	Category	Rating	Reviews	Size_MBs	Installs	Type	Price	Content_Ra
21	KBA-EZ Health Guide	MEDICAL	5.00	4	25.00	1	Free	0	Every
28	Ra Ga Ba	GAME	5.00	2	20.00	1	Paid	\$1.49	Every
47	Mu.F.O.	GAME	5.00	2	16.00	1	Paid	\$0.99	Every
82	Brick Breaker BR	GAME	5.00	7	19.00	5	Free	0	Every
99	Anatomy & Physiology Vocabulary Exam Review App	MEDICAL	5.00	1	4.60	5	Free	0	Every



In [14]: `df_apps_clean.sort_values('Rating', ascending=False).head()`

Out[14]:

	App	Category	Rating	Reviews	Size_MBs	Installs	Type	Price	Cont
21	KBA-EZ Health Guide	MEDICAL	5.00	4	25.00	1	Free	0	
1230	Sway Medical	MEDICAL	5.00	3	22.00	100	Free	0	
1227	AJ Men's Grooming	LIFESTYLE	5.00	2	22.00	100	Free	0	
1224	FK Dedinje BGD	SPORTS	5.00	36	2.60	100	Free	0	
1223	CB VIDEO VISION	PHOTOGRAPHY	5.00	13	2.60	100	Free	0	



In [15]: `df_apps_clean.sort_values('Size_MBs', ascending=False).head()`

Out[15]:

	App	Category	Rating	Reviews	Size_MBs	Installs	Type
9942	Talking Babsy Baby: Baby Games	LIFESTYLE	4.00	140995	100.00	10,000,000	Free
10687	Hungry Shark Evolution	GAME	4.50	6074334	100.00	100,000,000	Free
9943	Miami crime simulator	GAME	4.00	254518	100.00	10,000,000	Free
9944	Gangster Town: Vice District	FAMILY	4.30	65146	100.00	10,000,000	Free
3144	Vi Trainer	HEALTH_AND_FITNESS	3.60	124	100.00	5,000	Free

In [16]: `df_apps_clean.sort_values('Reviews', ascending=False).head(10)`

Out[16]:

	App	Category	Rating	Reviews	Size_MBs	Installs	Type
10805	Facebook	SOCIAL	4.10	78158306	5.30	1,000,000,000	Free
10785	WhatsApp Messenger	COMMUNICATION	4.40	69119316	3.50	1,000,000,000	Free
10806	Instagram	SOCIAL	4.50	66577313	5.30	1,000,000,000	Free
10784	Messenger – Text and Video Chat for Free	COMMUNICATION	4.00	56642847	3.50	1,000,000,000	Free
10650	Clash of Clans	GAME	4.60	44891723	98.00	100,000,000	Free
10744	Clean Master- Space Cleaner & Antivirus	TOOLS	4.70	42916526	3.40	500,000,000	Free
10835	Subway Surfers	GAME	4.50	27722264	76.00	1,000,000,000	Free
10828	YouTube	VIDEO_PLAYERS	4.30	25655305	4.65	1,000,000,000	Free
10746	Security Master - Antivirus, VPN, AppLock, Boo...	TOOLS	4.70	24900999	3.40	500,000,000	Free
10584	Clash Royale	GAME	4.60	23133508	97.00	100,000,000	Free


In [17]: `df_apps_clean.sort_values('Reviews', ascending=False).head(10)`

Out[17]:

	App	Category	Rating	Reviews	Size_MBs	Installs	Type
10805	Facebook	SOCIAL	4.10	78158306	5.30	1,000,000,000	Free
10785	WhatsApp Messenger	COMMUNICATION	4.40	69119316	3.50	1,000,000,000	Free
10806	Instagram	SOCIAL	4.50	66577313	5.30	1,000,000,000	Free
10784	Messenger – Text and Video Chat for Free	COMMUNICATION	4.00	56642847	3.50	1,000,000,000	Free
10650	Clash of Clans	GAME	4.60	44891723	98.00	100,000,000	Free
10744	Clean Master- Space Cleaner & Antivirus	TOOLS	4.70	42916526	3.40	500,000,000	Free
10835	Subway Surfers	GAME	4.50	27722264	76.00	1,000,000,000	Free
10828	YouTube	VIDEO_PLAYERS	4.30	25655305	4.65	1,000,000,000	Free
10746	Security Master - Antivirus, VPN, AppLock, Boo...	TOOLS	4.70	24900999	3.40	500,000,000	Free
10584	Clash Royale	GAME	4.60	23133508	97.00	100,000,000	Free

In [18]: `ratings = df_apps_clean.Content_Rating.value_counts()`
`ratings`

Out[18]:

Content_Rating	
Everyone	6621
Teen	912
Mature 17+	357
Everyone 10+	305
Adults only 18+	3
Unrated	1
Name: count, dtype: int64	

In [19]: `fig = px.pie(labels=ratings.index,`
`values=ratings.values,`
`title="Content Rating",`
`names=ratings.index,`
`)`
`fig.update_traces(textposition='outside', textinfo='percent+label')`

```
fig.show()
```


Content Rating

	Matur
	4.3
Everyone 10	3.72%
Adults only 18+	0.0366%
Unrated	

```
In [40]: df_apps_clean.sort_values('Installs', ascending=False).head()
```


Out[40]:

	App	Category	Rating	Reviews	Size_MBs	Installs	Type	Pri
10731	My Talking Tom	GAME	4.50	14891223	36.00	500,000,000	Free	
10746	Security Master - Antivirus, VPN, AppLock, Boo...	TOOLS	4.70	24900999	3.40	500,000,000	Free	
10711	SHAREit - Transfer & Share	TOOLS	4.60	7790693	17.00	500,000,000	Free	
10713	imo free video calls and chat	COMMUNICATION	4.30	4785892	11.00	500,000,000	Free	
10717	Pou	GAME	4.30	10485308	24.00	500,000,000	Free	



```
In [42]: installs = df_apps_clean.Installs.value_counts()
installs
```

```
Out[42]: Installs
1,000,000      1417
100,000        1096
10,000          988
10,000,000      933
1,000           698
5,000,000       607
500,000         504
50,000          457
5,000           425
100             303
50,000,000      202
500             199
100,000,000     189
10              69
50              56
500,000,000     24
1,000,000,000   20
5                9
1                3
Name: count, dtype: int64
```

```
In [44]: df_apps_clean.Installs.describe()
```

```
Out[44]: count      8199
unique        19
top      1,000,000
freq        1417
Name: Installs, dtype: object
```

```
In [46]: df_apps_clean.Installs= df_apps_clean.Installs.astype(str).str.replace(',','')
df_apps_clean.Installs = pd.to_numeric(df_apps_clean.Installs)
df_apps_clean[['App', 'Installs']].groupby('Installs').count()
```

Out[46]:

App	
Installs	
1	3
5	9
10	69
50	56
100	303
500	199
1000	698
5000	425
10000	988
50000	457
100000	1096
500000	504
1000000	1417
5000000	607
10000000	933
50000000	202
100000000	189
500000000	24
1000000000	20

```
In [48]: df_apps_clean.Price = df_apps_clean.Price.astype(str).str.replace('[$,]', '', re
df_apps_clean.Price = pd.to_numeric(df_apps_clean.Price)
df_apps_clean[['App', 'Price']].groupby('Price').count()
```

Out[48]:

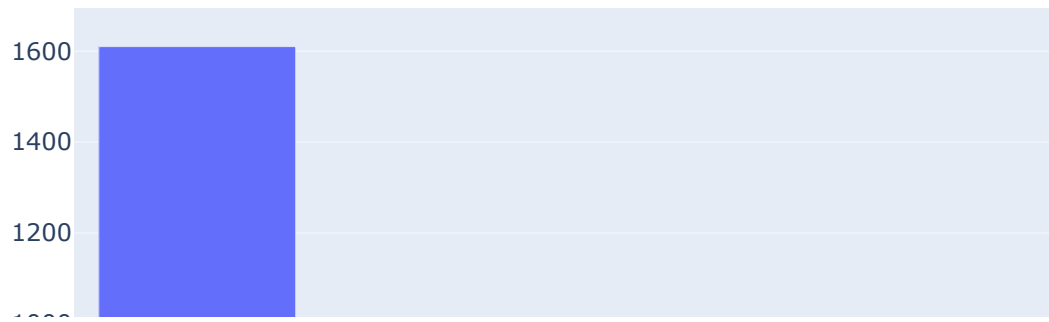
App	Price
0.00	7595
0.99	104
1.00	2
1.20	1
1.29	1
...	...
299.99	1
379.99	1
389.99	1
399.99	11
400.00	1

73 rows × 1 columns

In [50]: `df_apps_clean.Category.nunique()`

Out[50]: 33

In [52]: `top10_category = df_apps_clean.Category.value_counts()[:10]`
`top10_category`Out[52]: Category
FAMILY 1610
GAME 910
TOOLS 719
FINANCE 302
LIFESTYLE 302
PRODUCTIVITY 301
PERSONALIZATION 298
MEDICAL 292
PHOTOGRAPHY 263
BUSINESS 262
Name: count, dtype: int64In [54]: `bar = px.bar(x = top10_category.index, # index = category name`
`y = top10_category.values)`
`bar.show()`

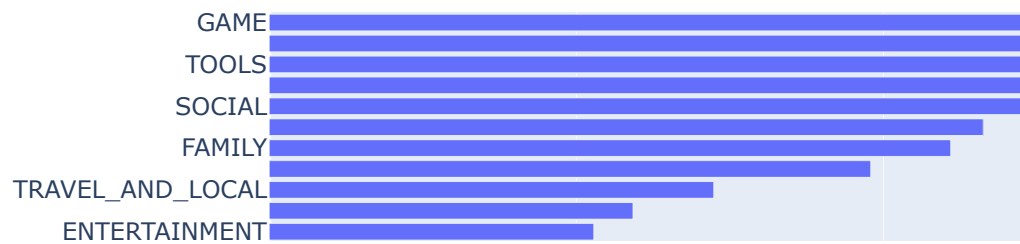


```
In [56]: category_installs = df_apps_clean.groupby('Category').agg({'Installs': pd.Series
category_installs.sort_values('Installs', ascending=True, inplace=True)
```

```
In [60]: h_bar = px.bar(x = category_installs.Installs,
                        y = category_installs.index,
                        orientation='h',
                        title='Category Popularity')

h_bar.update_layout(xaxis_title='Number of Downloads', yaxis_title='Category')
h_bar.show()
```

Category Popularity



```
In [62]: cat_number = df_apps_clean.groupby('Category').agg({'App': pd.Series.count})
```

```
In [64]: cat_merged_df = pd.merge(cat_number, category_installs, on='Category', how='inner')
print(f'The dimensions of the DataFrame are: {cat_merged_df.shape}')
cat_merged_df.sort_values('Installs', ascending=False)
```

The dimensions of the DataFrame are: (33, 2)

Out[64]:

	App	Installs
Category		
GAME	910	13858762717
COMMUNICATION	257	11039241530
TOOLS	719	8099724500
PRODUCTIVITY	301	5788070180
SOCIAL	203	5487841475
PHOTOGRAPHY	263	4649143130
FAMILY	1610	4437579590
VIDEO_PLAYERS	148	3916897200
TRAVEL_AND_LOCAL	187	2894859300
NEWS_AND_MAGAZINES	204	2369110650
ENTERTAINMENT	102	2113660000
BOOKS_AND_REFERENCE	169	1665791655
PERSONALIZATION	298	1532352930
SHOPPING	180	1400331540
HEALTH_AND_FITNESS	243	1134006220
SPORTS	260	1096431465
BUSINESS	262	692018120
LIFESTYLE	302	503742120
MAPS_AND_NAVIGATION	118	503267560
FINANCE	302	455312400
WEATHER	72	361096500
EDUCATION	118	352852000
FOOD_AND_DRINK	94	211677750
DATING	134	140912410
ART_AND_DESIGN	61	114233100
HOUSE_AND_HOME	62	97082000
AUTO_AND_VEHICLES	73	53129800
LIBRARIES_AND_DEMO	64	52083000
COMICS	54	44931100
MEDICAL	292	39162676
PARENTING	50	31116110
BEAUTY	42	26916200

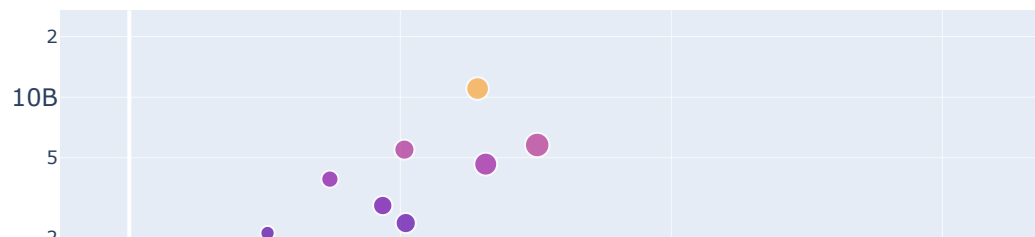
	App	Installs
Category		
EVENTS	45	15949410

```
In [66]: scatter = px.scatter(cat_merged_df, # data
                             x='App', # column name
                             y='Installs',
                             title='Category Concentration',
                             size='App',
                             hover_name=cat_merged_df.index,
                             color='Installs')

scatter.update_layout(xaxis_title="Number of Apps (Lower=More Concentrated)",
                      yaxis_title="Installs",
                      yaxis=dict(type='log'))

scatter.show()
```

Category Concentration



```
In [68]: stack = df_apps_clean.Genres.str.split(';', expand=True).stack()
print(f'We now have a single column with shape: {stack.shape}')
num_genres = stack.value_counts()
print(f'Number of genres: {len(num_genres)}')
```

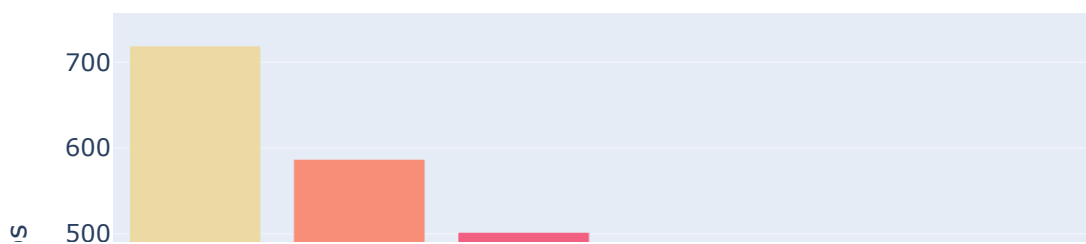
We now have a single column with shape: (8579,)
 Number of genres: 53

```
In [70]: bar = px.bar(x = num_genres.index[:15], # index = category name
                      y = num_genres.values[:15], # count
                      title='Top Genres',
                      hover_name=num_genres.index[:15],
                      color=num_genres.values[:15],
                      color_continuous_scale='Agsunset')

bar.update_layout(xaxis_title='Genre',
                  yaxis_title='Number of Apps',
                  coloraxis_showscale=False)

bar.show()
```

Top Genres



```
In [72]: df_apps_clean.Type.value_counts()
```

```
Out[72]: Type
Free      7595
Paid       604
Name: count, dtype: int64
```

```
In [74]: df_free_vs_paid = df_apps_clean.groupby(["Category", "Type"], as_index=False).agg(
df_free_vs_paid.head()
```


Out[74]:

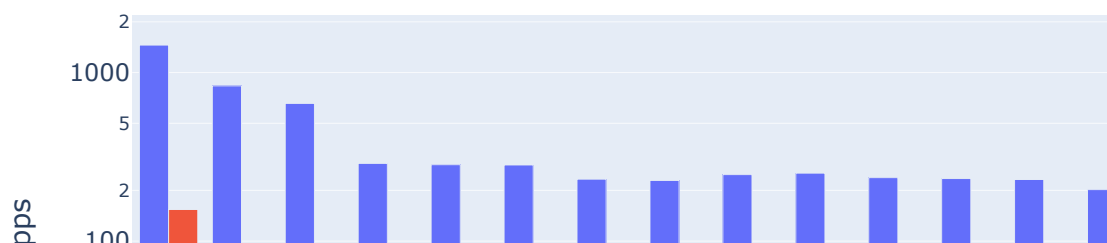
	Category	Type	App
0	ART_AND_DESIGN	Free	58
1	ART_AND_DESIGN	Paid	3
2	AUTO_AND_VEHICLES	Free	72
3	AUTO_AND_VEHICLES	Paid	1
4	BEAUTY	Free	42

```
In [76]: g_bar = px.bar(df_free_vs_paid,
                        x='Category',
                        y='App',
                        title='Free vs Paid Apps by Category',
                        color='Type',
                        barmode='group')

g_bar.update_layout(xaxis_title='Category',
                    yaxis_title='Number of Apps',
                    xaxis={'categoryorder':'total descending'},
                    yaxis=dict(type='log'))

g_bar.show()
```

Free vs Paid Apps by Category



```
In [84]: box = px.box(df_apps_clean,
                     y='Installs',
```

```

x='Type',
color='Type',
notched=True,
points='all',
title='How Many Downloads are Paid Apps Giving Up?')

box.update_layout(yaxis=dict(type='log'))

box.show()

```



How Many Downloads are Paid Apps Giving Up?



```
In [86]: df_paid_apps.Price.median()
```

```
Out[86]: 2.99
```

```

In [88]: box = px.box(df_paid_apps,
                      x='Category',
                      y="Price",
                      title='Price per Category')

box.update_layout(xaxis_title='Category',
                  yaxis_title='Paid App Price',
                  xaxis={'categoryorder': 'max descending'},
                  yaxis=dict(type='log'))

box.show()

```

Price per Category



In []: