Introduction:

This data analysis project focuses on exploring and understanding the Google Play Store Apps dataset.

The dataset is loaded into a Pandas DataFrame using Python, and various data analysis tasks are performed to gain insights into the characteristics of the apps available on the Google Play Store.

Dataset Overview:

The dataset consists of 10,841 entries and 13 columns, each representing different attributes of the apps.

These attributes include the app name, category, rating, number of reviews, size, number of installs, type (free or paid), price, content rating, genres, last updated information, current version, and Android version compatibility.

Basic Data Exploration:

The initial steps involve displaying the top 5 and last 3 rows of the dataset to get a glimpse of the data.

Additionally, the shape of the dataset is explored, revealing that it contains 10,841 rows and 13 columns.

In [1]:
```
# Import all required library

import numpy as np
import pandas as pd
```

In [2]:
```
df=pd.read_csv("googleplaystore.csv")
df.head()
```

Out[2]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Cont Rat
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19M	10,000+	Free	0	Every
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14M	500,000+	Free	0	Every
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8.7M	5,000,000+	Free	0	Every
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25M	50,000,000+	Free	0	T
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2.8M	100,000+	Free	0	Every

In [3]:
```
# Shape of dataset

df.shape
```

Out[3]:
```
(10841, 13)
```

Get complete info about the dataset

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10841 entries, 0 to 10840
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   App             10841 non-null  object 
 1   Category        10841 non-null  object 
 2   Rating          9367 non-null   float64
 3   Reviews         10841 non-null  object 
 4   Size            10841 non-null  object 
 5   Installs        10841 non-null  object 
 6   Type            10840 non-null  object 
 7   Price           10841 non-null  object 
 8   Content Rating  10840 non-null  object 
 9   Genres          10841 non-null  object 
 10  Last Updated    10841 non-null  object 
 11  Current Ver     10833 non-null  object 
 12  Android Ver     10838 non-null  object 
dtypes: float64(1), object(12)
memory usage: 1.1+ MB
```

Get overall statistics about the dataframe

In [5]:
```
df.describe()
```

Out[5]:

	Rating
count	9367.000000
mean	4.193338
std	0.537431
min	1.000000
25%	4.000000
50%	4.300000
75%	4.500000
max	19.000000

In [6]:
```
df.describe(include = 'all')
```

Out[6]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Co
count	10841	10841	9367.000000	10841	10841	10841	10840	10841	
unique	9660	34	NaN	6002	462	22	3	93	
top	ROBLOX	FAMILY	NaN	0	Varies with device	1,000,000+	Free	0	Ev
freq	9	1972	NaN	596	1695	1579	10039	10040	
mean	NaN	NaN	4.193338	NaN	NaN	NaN	NaN	NaN	
std	NaN	NaN	0.537431	NaN	NaN	NaN	NaN	NaN	
min	NaN	NaN	1.000000	NaN	NaN	NaN	NaN	NaN	
25%	NaN	NaN	4.000000	NaN	NaN	NaN	NaN	NaN	
50%	NaN	NaN	4.300000	NaN	NaN	NaN	NaN	NaN	
75%	NaN	NaN	4.500000	NaN	NaN	NaN	NaN	NaN	
max	NaN	NaN	19.000000	NaN	NaN	NaN	NaN	NaN	

Total number of app titles contain Astrology

In [7]:
```
df.columns
```

Out[7]:
```
Index(['App', 'Category', 'Rating', 'Reviews', 'Size', 'Installs', 'Type',
       'Price', 'Content Rating', 'Genres', 'Last Updated', 'Current Ver',
       'Android Ver'],
      dtype='object')
```

In [8]:
```
len(df[df['App'].str.contains('Astrology', case=False)])
```

Out[8]:
```
3
```

Find the average App Rating

In [9]:
```
df.columns
```

Out[9]:
```
Index(['App', 'Category', 'Rating', 'Reviews', 'Size', 'Installs', 'Type',
       'Price', 'Content Rating', 'Genres', 'Last Updated', 'Current Ver',
       'Android Ver'],
      dtype='object')
```

In [12]:
```
df['Rating'].mean()
```

Out[12]:
```
np.float64(4.193338315362443)
```

Find total number of unique category

In [13]:
```
df['Category'].nunique()
```

Out[13]:
```
34
```

Which category getting the highest avgerage rating?

In [15]:
```
df.groupby('Category')['Rating'].mean().sort_values(ascending=False)
```

Out[15]:
```
Category
1.9                    19.000000
EVENTS                  4.435556
EDUCATION               4.389032
ART_AND_DESIGN          4.358065
BOOKS_AND_REFERENCE     4.346067
PERSONALIZATION         4.335987
PARENTING               4.300000
GAME                    4.286326
BEAUTY                  4.278571
HEALTH_AND_FITNESS      4.277104
SHOPPING                4.259664
SOCIAL                  4.255598
WEATHER                 4.244000
SPORTS                  4.223511
PRODUCTIVITY            4.211396
HOUSE_AND_HOME          4.197368
FAMILY                  4.192272
PHOTOGRAPHY             4.192114
AUTO_AND_VEHICLES       4.190411
MEDICAL                 4.189143
LIBRARIES_AND_DEMO      4.178462
FOOD_AND_DRINK          4.166972
COMMUNICATION           4.158537
COMICS                  4.155172
NEWS_AND_MAGAZINES      4.132189
FINANCE                 4.131889
ENTERTAINMENT           4.126174
BUSINESS                4.121452
TRAVEL_AND_LOCAL        4.109292
LIFESTYLE               4.094904
VIDEO_PLAYERS           4.063750
MAPS_AND_NAVIGATION     4.051613
TOOLS                   4.047411
DATING                  3.970769
Name: Rating, dtype: float64
```

Find total number of apps having 5 star rating

In [16]:
```
len(df[df['Rating']==5.0])
```

Out[16]:
```
274
```

Find average value of reviews

In [17]:
```
df['Reviews'].dtypes
```

Out[17]:
```
dtype('O')
```

Convert the data type from Object to Int or Float

In [18]:
```
df['Reviews'].astype(float)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[18], line 3
      1 #convert the data type from Object to Int or Float
----> 3 df['Reviews'].astype(float)

File ~\anaconda3\Lib\site-packages\pandas\core\generic.py:6643, in NDFrame.astype
(self, dtype, copy, errors)
   6637     results = [
   6638         ser.astype(dtype, copy=copy, errors=errors) for _, ser in self.it
ems()
   6639     ]
   6641 else:
   6642     # else, only a single dtype is given
-> 6643     new_data = self._mgr.astype(dtype=dtype, copy=copy, errors=errors)
   6644     res = self._constructor_from_mgr(new_data, axes=new_data.axes)
   6645     return res.__finalize__(self, method="astype")

File ~\anaconda3\Lib\site-packages\pandas\core\internals\managers.py:430, in Base
BlockManager.astype(self, dtype, copy, errors)
    427 elif using_copy_on_write():
    428     copy = False
--> 430 return self.apply(
    431     "astype",
    432     dtype=dtype,
    433     copy=copy,
    434     errors=errors,
    435     using_cow=using_copy_on_write(),
    436 )

File ~\anaconda3\Lib\site-packages\pandas\core\internals\managers.py:363, in Base
BlockManager.apply(self, f, align_keys, **kwargs)
    361         applied = b.apply(f, **kwargs)
    362     else:
--> 363         applied = getattr(b, f)(**kwargs)
    364     result_blocks = extend_blocks(applied, result_blocks)
    366 out = type(self).from_blocks(result_blocks, self.axes)

File ~\anaconda3\Lib\site-packages\pandas\core\internals\blocks.py:758, in Block.
astype(self, dtype, copy, errors, using_cow, squeeze)
    755         raise ValueError("Can not squeeze with more than one column.")
    756     values = values[0, :]  # type: ignore[call-overload]
--> 758 new_values = astype_array_safe(values, dtype, copy=copy, errors=errors)
    760 new_values = maybe_coerce_values(new_values)
    762 refs = None

File ~\anaconda3\Lib\site-packages\pandas\core\dtypes\astype.py:237, in astype_ar
ray_safe(values, dtype, copy, errors)
    234     dtype = dtype.numpy_dtype
    236 try:
--> 237     new_values = astype_array(values, dtype, copy=copy)
    238 except (ValueError, TypeError):
    239     # e.g. _astype_nansafe can fail on object-dtype of strings
    240     #  trying to convert to float
    241     if errors == "ignore":

File ~\anaconda3\Lib\site-packages\pandas\core\dtypes\astype.py:182, in astype_ar
ray(values, dtype, copy)
    179     values = values.astype(dtype, copy=copy)
    181 else:
```

```
--> 182     values = _astype_nansafe(values, dtype, copy=copy)
    184 # in pandas we don't store numpy str dtypes, so convert to object
    185 if isinstance(dtype, np.dtype) and issubclass(values.dtype.type, str):

File ~\anaconda3\Lib\site-packages\pandas\core\dtypes\astype.py:133, in _astype_n
ansafe(arr, dtype, copy, skipna)
    129     raise ValueError(msg)
    131 if copy or arr.dtype == object or dtype == object:
    132     # Explicit copy, or required since NumPy can't view from / to object.
--> 133     return arr.astype(dtype, copy=True)
    135 return arr.astype(dtype, copy=copy)

ValueError: could not convert string to float: '3.0M'
```

In [20]:
```
# We got an error of '3.0M'

df[df['Reviews']=='3.0M']
```

Out[20]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Conte Rati
10472	Life Made WI-Fi Touchscreen Photo Frame	1.9	19.0	3.0M	1,000+	Free	0	Everyone	Na

In [21]:
```
df['Reviews'] = df['Reviews'].replace('3.0M', 3.0)
```

In [22]:
```
# Now convert it to float data type

df['Reviews'] = df['Reviews'].astype('float')
```

In [23]:
```
df['Reviews'].dtypes
```

Out[23]:
```
dtype('float64')
```

In [24]:
```
df['Reviews'].mean()
```

Out[24]:
```
np.float64(444111.9265750392)
```

Find total number of Free and Paid apps

In [25]:
```
df['Type'].value_counts()
```

Out[25]:
```
Type
Free    10039
Paid      800
0           1
Name: count, dtype: int64
```

Which app has maximum reviews?

In [26]:
```
df[df['Reviews'].max()==df['Reviews']]['App']
```

Out[26]:
```
2544    Facebook
Name: App, dtype: object
```

Display Top 5 Apps Having Highest Reviews

In [27]:
```
index=df['Reviews'].sort_values(ascending=False).head(5).index
```

In [28]:
```
df.iloc[index]['App']
```

Out[28]:
```
2544              Facebook
3943              Facebook
381     WhatsApp Messenger
336     WhatsApp Messenger
3904    WhatsApp Messenger
Name: App, dtype: object
```

Find Average Rating of Free and Paid Apps

In [29]:
```
df.groupby('Type')['Rating'].mean()
```

Out[29]:
```
Type
0       19.000000
Free     4.186203
Paid     4.266615
Name: Rating, dtype: float64
```

Display Top 5 Apps Having Maximum Installs

In [30]:
```
df.head(1)
```

Out[30]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159.0	19M	10,000+	Free	0	Everyone

In [31]:
```
df['Installs'].dtype
```

Out[31]:
```
dtype('O')
```

In [32]:
```
df['Installs_1']=df['Installs'].str.replace(',','')
```

In [34]:
```
df.tail(1)
```

Out[34]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Conten Ratin
10840	iHoroscope - 2018 Daily Horoscope & Astrology	LIFESTYLE	4.5	398307.0	19M	10,000,000+	Free	0	Everyon

In [35]:
```
df['Installs_1']=df['Installs_1'].str.replace('+','')
```

In [36]:
```
df.tail(1)
```

Out[36]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Conten Ratin
10840	iHoroscope - 2018 Daily Horoscope & Astrology	LIFESTYLE	4.5	398307.0	19M	10,000,000+	Free	0	Everyon

In [38]:
```
df['Installs_1'].astype('int')
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[38], line 1
----> 1 df['Installs_1'].astype('int')

File ~\anaconda3\Lib\site-packages\pandas\core\generic.py:6643, in NDFrame.astype
(self, dtype, copy, errors)
   6637     results = [
   6638         ser.astype(dtype, copy=copy, errors=errors) for _, ser in self.it
ems()
   6639     ]
   6641 else:
   6642     # else, only a single dtype is given
-> 6643     new_data = self._mgr.astype(dtype=dtype, copy=copy, errors=errors)
   6644     res = self._constructor_from_mgr(new_data, axes=new_data.axes)
   6645     return res.__finalize__(self, method="astype")

File ~\anaconda3\Lib\site-packages\pandas\core\internals\managers.py:430, in Base
BlockManager.astype(self, dtype, copy, errors)
    427 elif using_copy_on_write():
    428     copy = False
--> 430 return self.apply(
    431     "astype",
    432     dtype=dtype,
    433     copy=copy,
    434     errors=errors,
    435     using_cow=using_copy_on_write(),
    436 )

File ~\anaconda3\Lib\site-packages\pandas\core\internals\managers.py:363, in Base
BlockManager.apply(self, f, align_keys, **kwargs)
    361         applied = b.apply(f, **kwargs)
    362     else:
--> 363         applied = getattr(b, f)(**kwargs)
    364     result_blocks = extend_blocks(applied, result_blocks)
    366 out = type(self).from_blocks(result_blocks, self.axes)

File ~\anaconda3\Lib\site-packages\pandas\core\internals\blocks.py:758, in Block.
astype(self, dtype, copy, errors, using_cow, squeeze)
    755         raise ValueError("Can not squeeze with more than one column.")
    756     values = values[0, :]  # type: ignore[call-overload]
--> 758 new_values = astype_array_safe(values, dtype, copy=copy, errors=errors)
    760 new_values = maybe_coerce_values(new_values)
    762 refs = None

File ~\anaconda3\Lib\site-packages\pandas\core\dtypes\astype.py:237, in astype_ar
ray_safe(values, dtype, copy, errors)
    234     dtype = dtype.numpy_dtype
    236 try:
--> 237     new_values = astype_array(values, dtype, copy=copy)
    238 except (ValueError, TypeError):
    239     # e.g. _astype_nansafe can fail on object-dtype of strings
    240     #  trying to convert to float
    241     if errors == "ignore":

File ~\anaconda3\Lib\site-packages\pandas\core\dtypes\astype.py:182, in astype_ar
ray(values, dtype, copy)
    179     values = values.astype(dtype, copy=copy)
    181 else:
--> 182     values = _astype_nansafe(values, dtype, copy=copy)
```

```
    184 # in pandas we don't store numpy str dtypes, so convert to object
    185 if isinstance(dtype, np.dtype) and issubclass(values.dtype.type, str):

File ~\anaconda3\Lib\site-packages\pandas\core\dtypes\astype.py:133, in _astype_n
ansafe(arr, dtype, copy, skipna)
    129     raise ValueError(msg)
    131 if copy or arr.dtype == object or dtype == object:
    132     # Explicit copy, or required since NumPy can't view from / to object.
--> 133     return arr.astype(dtype, copy=True)
    135 return arr.astype(dtype, copy=copy)

ValueError: invalid literal for int() with base 10: 'Free'
```

In [39]:
```
df[df['Installs_1']=='Free']
```

Out[39]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Conte Rati
10472	Life Made WI-Fi Touchscreen Photo Frame	1.9	19.0	3.0	1,000+	Free	0	Everyone	Na

In [40]:
```
df['Installs_1']=df['Installs_1'].str.replace('Free','0')
```

In [41]:
```
df['Installs_1']=df['Installs_1'].astype('int')

# now it is successfully converted to int
# assign it back - data['Installs_1']=
```

In [42]:
```
df['Installs_1'].dtype
```

Out[42]:
```
dtype('int64')
```

In [43]:
```
index=df['Installs_1'].sort_values(ascending=False).head(5).index
```

In [44]:
```
df.iloc[index]['App']
```

Out[44]:
```
5856    Google Play Games
5395        Google Photos
2853        Google Photos
2884        Google Photos
4170         Google Drive
Name: App, dtype: object
```

Using the iloc() function in python, we can easily retrieve any particular value from a row or column using index values.

reference -

https://www.youtube.com/watch?v=qBOw_kcTLpU&list=PL_1pt6K-CLoDMEbYy2PcZuITWEjqMfyoA&index=8

In [ ]:

```

```