Objective: Make a model to predict the app rating, with other information about the app provided.

Problem Statement:

Google Play Store team is about to launch a new feature wherein, certain apps that are promising, are boosted in visibility. The boost will manifest in multiple ways including higher priority in recommendations sections (“Similar apps”, “You might also like”, “New and updated games”). These will also get a boost in search results visibility. This feature will help bring more attention to newer apps that have the potential.

Domain:

General

Analysis to be done:

The problem is to identify the apps that are going to be good for Google to promote. App ratings, which are provided by the customers, is always a great indicator of the goodness of the app. The problem reduces to: predict which apps will have high ratings.

Content:

Dataset: Google Play Store data (“googleplaystore.csv”)

Cleaning Google App Store Data

ref: https://www.kaggle.com/datasets/lava18/google-play-store-apps Web scraped data of 10k Play Store apps for analysing the Android market.

Acknowledgements

This information is scraped from the Google Play Store. This app information would not be available without it.

Inspiration

The Play Store apps data has enormous potential to drive app-making businesses to success. Actionable insights can be drawn for developers to work on and capture the Android market!

Setup

In [186…
```
import numpy as np
import pandas as pd
import seaborn as sns
from prettytable import PrettyTable
import matplotlib.pyplot as plt
%matplotlib inline
```

1. Load the data file using pandas.

In [205…
```
df = pd.read_csv('googleplaystore.csv')
df.head(5)
```

Out[205]:

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

In [206…
```
df.shape
```

Out[206]:
```
(10841, 13)
```

In [207…
```
df.dtypes
```

Out[207]:
```
App                object
Category           object
Rating            float64
Reviews            object
Size               object
Installs           object
Type               object
Price              object
Content Rating     object
Genres             object
Last Updated       object
Current Ver        object
Android Ver        object
dtype: object
```

In [208…
```
df.dtypes.groupby(df.dtypes.values).count()
```

Out[208]:
```
float64     1
object     12
dtype: int64
```

In [209…
```
df.columns = df.columns.str.replace(' ','_')
df.columns
```

Out[209]:
```
Index(['App', 'Category', 'Rating', 'Reviews', 'Size', 'Installs', 'Type',
       'Price', 'Content_Rating', 'Genres', 'Last_Updated', 'Current_Ver',
       'Android_Ver'],
      dtype='object')
```

Cleaning data and checking for inconsistency

2. Check for null values in the data. Get the number of null values for each column.

In [210…
```
df.isnull().sum()
```

Out[210]:
```
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
```

We notice that Rating, Type,Content_Rating, Current_Ver and Android_Ver attributes have 1474, 1, 1, 8 and 3 numbers of missing values respectively.

3. Drop records with nulls in any of the columns.

In [211…
```
df.drop(['Current_Ver', 'Android_Ver'], axis=1,inplace=True)
df.columns
```

Out[211]:
```
Index(['App', 'Category', 'Rating', 'Reviews', 'Size', 'Installs', 'Type',
       'Price', 'Content_Rating', 'Genres', 'Last_Updated'],
      dtype='object')
```

In [212…
```
df.dropna(subset="Rating",axis=0, inplace=True)
```

In [213…
```
df.shape
```

Out[213]:
```
(9367, 11)
```

4. Variables seem to have incorrect type and inconsistent formatting. You need to fix them:

4.1 Size column has sizes in Kb as well as Mb. To analyze, you'll need to convert these to numeric.

Extract the numeric value from the column

Multiply the value by 1,000, if size is mentioned in Mb

In [214…
```
def clean_Size(val):
    return val.replace(".","").replace("M","").replace("k","").replace("Varies with de
#type(clean_Size(('8.7M')))
df.Size = df.Size.apply(clean_Size)
df['Size'] = df['Size'].apply(lambda x: x.replace('+', '').replace(',','')).astype(int
```

In [215…
```
df['Size'].dtypes
```

Out[215]:
```
dtype('int32')
```

4.2 Price field is a string and has dollar symbol. Remove dollar sign, and convert it to numeric.

In [216…
```
df["Price"].sort_values()
```

Out[216]:
```
4773        $0.99
8219        $0.99
9060        $0.99
10682       $0.99
9057        $0.99
           ...
3312            0
3313            0
3314            0
3308            0
10472    Everyone
Name: Price, Length: 9367, dtype: object
```

In [217…
```
df.drop(index=10472,inplace=True)
df['Price']=df['Price'].apply(lambda x: str(x).replace('$','')if '$' in str(x) else st
df["Price"]=df.Price.astype(float)
```

4.3 Reviews is a numeric field that is loaded as a string field. Convert it to numeric (int/float).

In [218…
```
df['Reviews'] = df['Reviews'].astype(int)
df.dtypes
```

Out[218]:
```
App                object
Category           object
Rating            float64
Reviews             int32
Size                int32
Installs           object
Type               object
Price             float64
Content_Rating     object
Genres             object
Last_Updated       object
dtype: object
```

4.4 Installs field is currently stored as string and has values like 1,000,000+.

Treat 1,000,000+ as 1,000,000

remove '+', ',' from the field, convert it to integer

In [219…
```
# Remove '+' and ',' characters and convert 'Installs' to integer
df['Installs'] = df['Installs'].apply(lambda x: x.replace(',', '').replace('+', '')).a
df.dtypes
```

Out[219]:
```
App               object
Category          object
Rating           float64
Reviews            int32
Size               int32
Installs           int32
Type              object
Price            float64
Content_Rating    object
Genres            object
Last_Updated      object
dtype: object
```

5. Sanity checks:

5.1 Average rating should be between 1 and 5 as only these values are allowed on the play store. Drop the rows that have a value outside this range.

In [220…
```
df["Rating"].sort_values(ascending=False)
```

Out[220]:
```
9056     5.0
8395     5.0
8493     5.0
6330     5.0
6342     5.0
        ...
7806     1.0
10591    1.0
7427     1.0
7926     1.0
4127     1.0
Name: Rating, Length: 9366, dtype: float64
```

5.2 Reviews should not be more than installs as only those who installed can review the app. If there are any such records, drop them.

In [222…
```
df[df['Reviews']>df['Installs']]
```

Out[222]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content_Rating	Genres	Last
2454	KBA-EZ Health Guide	MEDICAL	5.0	4	25	1	Free	0.00	Everyone	Medical	
4663	Alarmy (Sleep If U Can) - Pro	LIFESTYLE	4.8	10249	0	10000	Paid	2.49	Everyone	Lifestyle	Jul
5917	Ra Ga Ba	GAME	5.0	2	20	1	Paid	1.49	Everyone	Arcade	F
6700	Brick Breaker BR	GAME	5.0	7	19	5	Free	0.00	Everyone	Arcade	Jul
7402	Trovami se ci riesci	GAME	5.0	11	61	10	Free	0.00	Everyone	Arcade	
8591	DN Blog	SOCIAL	5.0	20	42	10	Free	0.00	Teen	Social	Jul
10697	Mu.F.O.	GAME	5.0	2	16	1	Paid	0.99	Everyone	Arcade	Mar

In [223…

```
df.drop(df[df['Reviews']>df['Installs']].index,inplace=True)
df.shape
```

Out[223]:

```
(9359, 11)
```

5.3 For free apps (type = “Free”), the price should not be >0. Drop any such rows.

In [224…

```
#Performing the sanity checks on prices of free apps
df[(df.Type == "Free") & (df.Price > 0)]
```

Out[224]:

App	Category	Rating	Reviews	Size	Installs	Type	Price	Content_Rating	Genres	Last_Updated

Write Clean Data to new file for further exploration

In [225…

```
df.to_csv("googleappstore_cleandata.csv")
```

In [ ]: