PRESIDENCY UNIVERSITY

Private University Estd. in Karnataka State by Act No. 41 of 2013

GOOGLE PLAYSTORE APPS

A Project Report Submitted By

CHATAKONDU VENKATA KALYAN BABU	*20181CSE0135*
CHENNA LOHITH	*20181CSE0138*
CHILUKALA TEJA VAMSHIDHAR REDDY	*20181CSE0147*
KUSUMANJALI VEGI	*20181CSE0370*

as part of lab-based course Data Visualization, CSE 367 of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

under the supervision of

Mr. Rama Krishna K-Assistant Professor (CSE) Ms. Shruthi U-Assistant Professor (CSE)

PRESIDENCY UNIVERSITY

Private University Estd. in Karnataka State by Act No. 41 of 2013

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that the project report entitled Google play store application is a bonafide record of Mini Project work done as part of CSE 367 Data Visualization Using Python during the academic year 2021-2022 by:

CHATAKONDU VENKATA KALYAN BABU (20181CSE0135)

CHENNA LOHITH (20181CSE0138)

CHILUKALA TEJA VAMSHIDHAR REDDY (20181CSE0147)

KUSUMANJALI VEGI (20181CSE0370)

CONTENTS

INTRODUCTION

The mobile phones have made our lives easy and convenient. The major roles in these mobile phones are the applications people use for their day today life. The installation of these applications are usually done from play store. So we decided to take a data set based on this and visualize the data, to check the requirement of the people using these application.

About dataset:

There are several attributes in the given data such as:

- Application : Name of the given application
- Category: Describes about to which category does the application belong to. There are several categories like Family, Lifestyle, games, communication, travel, music, weather, books, new, photography etc.,
- Rating: Refers to the rating given by the users based on their experience regarding the application.
- Reviews: Refers to the comments given by the users based on their experience.
- Size: Refers to memory allocation of the application in the system.
- Installs: Shows the number of installations done by the users all over.
- Type: Shows the type of applications whether its for free or charged.
- Price: Shows the price of the application to be paid to use that particular application.
- Content Writing: Content writing is about the allowance of the people to use the application based on the user's age.
- Last update: Shows the date of the latest update of the applications.
- Current version: Shows the present version of the application which runs in the system.
- Android version: Shows the version of the system in which the application runs.

PYTHON LIBRARIES USED:

- **Matplotlib:** Cross platform for DV and graphical plotting library in python. Matplotlib supports all the popular charts (lots, histograms, power spectra, bar charts, error charts, scatterplots, etc.) right out of the box. There are also extensions that you can use to create advanced visualizations like 3-Dimensional plots, etc
- **Pandas:** Famous and highly used data manipulation tool and its key data structure is df which is used to store and manipulate tabular data in rows of observations and columns of variables.
- **Seaborn:** Core purpose is making statistical graphics in python and can be customized easily, useful feature of Seaborn is that it supports a plethora of advanced plots like categorical plotting (catplot), distribution plotting using kde (distplot), swarm plot, etc. right out of the box. And of course, we saw one example of relplot above.
- **WordCount:** Wordcount library and stopwords is used to eliminate words that are commonly used which carries very little useful info.

CODE:

```
import matplotlib.pyplot as plt
import pandas as pd
import plotly.express as px
import seaborn as sns
import matplotlib
%matplotlib inline
from wordcloud import WordCloud


data = pd.read_csv('/content/googleplaystore.csv')
```

Using WordCloud to display the columns in the row

```
plt.subplots(figsize=(8,8))     #setting the size of the word cloud
wordcloud = WordCloud(
                           background_color='white',     #setting teh pa
rameters for word cloud
                           width=512,
                           height=384
                          ).generate(" ".join(data))
plt.imshow(wordcloud)      #it will show the word cloud on the screen wi
th the specified parameters
plt.axis('off')            #It will hide the axis and the borders
plt.savefig('graph.png')

plt.show()
```

Pie chart for the Android version supported by the app

```
_1=0
_2=0
_3=0
_4=0
_5=0
_6=0
_7=0
_8=0
And_ver=data["Android_Ver"]
for i in range(0,4000):
  if And_ver[i].startswith('1') == True:
    _1=_1+1
  elif And_ver[i].startswith('2') == True:        #In this for loop we
are countting the apps and their android version
    _2=_2+1
  elif And_ver[i].startswith('3') == True:
```

```
    _3=_3+1
  elif And_ver[i].startswith('4') == True:
    _4=_4+1
  elif And_ver[i].startswith('5') == True:
    _5=_5+1
  elif And_ver[i].startswith('6') == True:
    _6=_6+1
  elif And_ver[i].startswith('7') == True:
    _7=_7+1
  elif And_ver[i].startswith('8') == True:
    _8=_8+1
labels=["1.0","2.0","3.0","4.0","5.0","6.0","7.0","8.0"]
sizes=[_1,_2,_3,_4,_5,_6,_7,_8]
explode = (0.2,0.2,0.1,0.3,0.3,0.3,0.2,0.1)
colors=["#abcae4","yellow","#86b2d8","red","#3d84bf","#316a9a","#255075
","#193750"]
plt.pie(sizes,colors=colors, autopct='%1.1f%%', startangle=90, pctdista
nce=0.85, explode = explode)
centre_circle = plt.Circle((0,0),0.70,fc='white')
fig = plt.gcf()
fig.gca().add_artist(centre_circle)
plt.tight_layout()
plt.legend(labels,loc="best")
plt.show()
```

Showing the top categories in the dataset using bar graph

```
sns.set_style('darkgrid')               #this set the graph style to da
rk grids
matplotlib.rcParams['font.size'] = 14     #setting the font size
matplotlib.rcParams['figure.figsize'] = (9, 5)      #setting the figure
 size
matplotlib.rcParams['figure.facecolor'] = '#00000000'     #setting the
face color as black


y = data['Category'].value_counts().index     #takes the indexes of the
 counted values
x = data['Category'].value_counts()     #count each category and its re
petence
xsis = []
ysis = []
for i in range(len(x)):
    xsis.append(x[i])     #the value of repetence is appended
    ysis.append(y[i])     #the indexes are appended
```

```
plt.figure(figsize=(18,13))     #setting the graph size
plt.xlabel("Count")
plt.ylabel("Category")

graph = sns.barplot(x = xsis, y = ysis, palette= "husl")      #in husl
pallet i will cover all the colors in pallet
graph.set_title("Top categories on Google Playstore", fontsize = 25);
```

Using Pie chart to show number of paid and free apps

```
type = data['Type']
free=0
paid=0
for i in range(10000):
  if type[i]=='Free':
    free=free+1
  elif type[i]=='Paid':     #counting the no.of paid apps and the free
apps
    paid=paid+1
  else:
    continue
data2=[free,paid]
labels=['Free','Paid']
colors = ["yellow", "grey"]
explode = (0.4, 0)
plt.pie(data2, explode=explode, labels=labels, colors=colors,autopct='%
1.1f%%', shadow=True, startangle=140)
plt.title("The No.of of Paid apps and Free apps\n" + "Out of 10,842 app
s", bbox={'facecolor':'0.8', 'pad':5})
plt.show()
```

Using Horizontal bar graph to show top 10 genres

```
gen = data['Genres'].value_counts().head(10)      #counting the genres
and taking the top 10 out of it
data1= ['Tools','Entertainment','Education','Medical','Business','Produ
ctivity','Sports','Personalization','Communication','Lifestyle']
x=data['Genres'].size
y= gen
fig,ax = plt.subplots()
ax.barh(data1,y,color='r')
for i, v in enumerate(y):
    ax.text(v + 3, i + .25, str(v), color='blue' )
plt.title("Top 10 Genres in created apps")
plt.xlabel('Total No.of apps')
plt.ylabel('Top 10 Genres')
print("Total No.Of apps Listes : ",x)
```

```
print("")
plt.show()
```

Using Bar graph to show number of apps in category wise

```
plt.figure(figsize=(12,5))
plt.title("Apps Category Wise")
plt.ylabel('No.of Apps')
plt.xlabel('Category')
plt.xticks(rotation=60,fontsize=10)
data['Category'].value_counts().head(10).plot(kind='bar')
plt.show()
```

Bar graph to show distribution according to android version of the app

```
plt.title('Distruibution according to the "Android Version" of the App'
,fontweight=900)
plt.ylabel('Android Version')
plt.xlabel('No. of Apps')
data['Android_Ver'].value_counts().head(10).plot(kind='barh')
plt.show()
```

HeatMap to show Number of app rated in each age group

```
plt.title("No. of Apps rated in each Age Group")
sns.heatmap(data.groupby('Content Rating')[['App']].count(),fmt="d", an
not=True, cmap='Reds')
```

Using plot to show number of apps in individual category

```
plt.figure(figsize=(30,5))
gen = sns.countplot(data.Category)
gen.set_xticklabels(gen.get_xticklabels(), rotation=90, fontsize=12)
plt.show()
```

Using plot to show paid apps in ascending order

```
plt.figure(figsize=(20, 7))
paid_apps_df = data[data['Type'] == 'Paid'].sort_values(by=['Price'],as
cending=True)
plot_df = sns.countplot(paid_apps_df['Price'])
plot_df.set_xticklabels(plot_df.get_xticklabels(), rotation=90, ha="rig
ht")
```

Using bar graph to show number of apps in different android version

```
data['Android_Ver'].unique()
data['Android_Ver'].replace(to_replace=['4.4W and up','Varies with devi
ce'], value=['4.4','1.0'],inplace=True)
data['Android_Ver'].replace({k: '1.0' for k in ['1.0','1.0 and up','1.5
 and up','1.6 and up']},inplace=True)
data['Android_Ver'].replace({k: '2.0' for k in ['2.0 and up','2.0.1 and
 up','2.1 and up','2.2 and up','2.2 -
 7.1.1','2.3 and up','2.3.3 and up']},inplace=True)
data['Android_Ver'].replace({k: '3.0' for k in ['3.0 and up','3.1 and u
p','3.2 and up']},inplace=True)
data['Android_Ver'].replace({k: '4.0' for k in ['4.0 and up','4.0.3 and
 up','4.0.3 - 7.1.1','4.1 and up','4.1 -
 7.1.1','4.2 and up','4.3 and up','4.4','4.4 and up']},inplace=True)
data['Android_Ver'].replace({k: '5.0' for k in ['5.0 - 6.0','5.0 -
 7.1.1','5.0 - 8.0','5.0 and up','5.1 and up']},inplace=True)
data['Android_Ver'].replace({k: '6.0' for k in ['6.0 and up']},inplace=
True)
data['Android_Ver'].replace({k: '7.0' for k in ['7.0 -
 7.1.1','7.0 and up','7.1 and up']},inplace=True)
data['Android_Ver'].replace({k: '8.0' for k in ['8.0 and up']},inplace=
True)
data['Android_Ver'].fillna('1.0', inplace=True)
print(data.groupby('Category')['Android_Ver'].value_counts())
Type_cat = data.groupby('Category')['Android_Ver'].value_counts().unsta
ck().plot.barh(figsize=(20,40), width=2)
plt.show()
```

OUTPUT:

Google Play Store.ipynb

File Edit View Insert Runtime Tools Help Changes will not be saved

Share

+ Code + Text Copy to Drive

Connect Editing

DATA

```
[ ] import matplotlib.pyplot as plt
    import pandas as pd
    import plotly.express as px
    import seaborn as sns
    import matplotlib
    %matplotlib inline
    from wordcloud import WordCloud
```

```
[ ] data = pd.read_csv('/content/googleplaystore.csv')
```

```
[ ] data.describe()
```

	Rating	Reviews	Size
count	9366.000000	1.084000e+04	10840.000000
mean	4.191757	4.441529e+05	35.773441
std	0.515219	2.927761e+06	90.861795
min	1.000000	0.000000e+00	1.000000
25%	4.000000	3.800000e+01	6.800000
50%	4.300000	2.094000e+03	20.000000
75%	4.500000	5.477550e+04	29.000000

Google Play Store.ipynb

File Edit View Insert Runtime Tools Help Changes will not be saved

Share

+ Code + Text Copy to Drive

Connect Editing

```
[ ] data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10841 entries, 0 to 10840
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   App             10840 non-null  object
 1   Category        10840 non-null  object
 2   Rating          9366 non-null   float64
 3   Reviews         10840 non-null  float64
 4   Size            10840 non-null  float64
 5   Installs        10840 non-null  object
 6   Type            10839 non-null  object
 7   Price           10840 non-null  object
 8   Content Rating  10840 non-null  object
 9   Genres          10840 non-null  object
 10  Last Updated    10840 non-null  object
 11  Current Ver     10832 non-null  object
 12  Android Ver     10840 non-null  object
dtypes: float64(3), object(10)
memory usage: 1.1+ MB
```

```
[ ] data.shape
```

```
(10841, 13)
```

```
[ ] data.isnull().sum()     #sum of null values in each colomn
```

```
App                1
```

Google Play Store.ipynb

File Edit View Insert Runtime Tools Help Changes will not be saved

Share

+ Code + Text Copy to Drive

Connect Editing

Using WordCloud to display the columns in the row

```
[ ] plt.subplots(figsize=(8,8))      #setting the size of the word cloud
    wordcloud = WordCloud(
                          background_color='white',      #setting teh parameters for word cloud
                          width=512,
                          height=384
                         ).generate(" ".join(data))
    plt.imshow(wordcloud)      #it will show the word cloud on the screen with the specified parameters
    plt.axis('off')          #It will hide the axis and the borders
    plt.savefig('graph.png')

    plt.show()
```

Google Play Store.ipynb

File Edit View Insert Runtime Tools Help Changes will not be saved

Share

+ Code + Text Copy to Drive

Connect Editing

Pie chart for the Android version supported by the app

```
[ ] _1=0
    _2=0
    _3=0
    _4=0
    _5=0
    _6=0
    _7=0
    _8=0
    And_ver=data["Android_Ver"]
    for i in range(0,4000):
      if And_ver[i].startswith('1') == True:
        _1=_1+1
      elif And_ver[i].startswith('2') == True:       #In this for loop we are countting the apps and their android version
        _2=_2+1
      elif And_ver[i].startswith('3') == True:
        _3=_3+1
      elif And_ver[i].startswith('4') == True:
        _4=_4+1
      elif And_ver[i].startswith('5') == True:
        _5=_5+1
      elif And_ver[i].startswith('6') == True:
        _6=_6+1
      elif And_ver[i].startswith('7') == True:
        _7=_7+1
      elif And_ver[i].startswith('8') == True:
        _8=_8+1
```

Google Play Store.ipynb

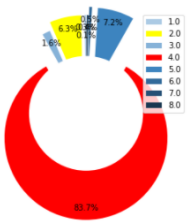
File Edit View Insert Runtime Tools Help Changes will not be saved

Share

+ Code + Text Copy to Drive

Connect Editing

```
[ ]     elif And_ver[i].startswith('8') == True:
        _8=_8+1
    labels=["1.0","2.0","3.0","4.0","5.0","6.0","7.0","8.0"]
    sizes=[_1,_2,_3,_4,_5,_6,_7,_8]
    explode = (0.2,0.2,0.1,0.3,0.3,0.3,0.2,0.1)
    colors=["#abcae4","yellow","#86b2d8","red","#3d84bf","#316a9a","#255075","#193750"]
    plt.pie(sizes,colors=colors, autopct='%1.1f%%', startangle=90, pctdistance=0.85, explode = explode)
    centre_circle = plt.Circle((0,0),0.70,fc='white')
    fig = plt.gcf()
    fig.gca().add_artist(centre_circle)
    plt.tight_layout()
    plt.legend(labels,loc="best")
    plt.show()
```



Google Play Store.ipynb

File Edit View Insert Runtime Tools Help Changes will not be saved

Share

+ Code + Text Copy to Drive

Connect Editing

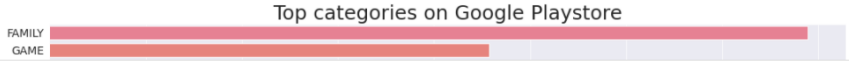
Showing the top categories in the dataset using bar graph

```
[ ] sns.set_style('darkgrid')              #this set the graph style to dark grids
    matplotlib.rcParams['font.size'] = 14     #setting the font size
    matplotlib.rcParams['figure.figsize'] = (9, 5)      #setting the figure size
    matplotlib.rcParams['figure.facecolor'] = '#00000000'     #setting the face color as black
```

```
[ ] y = data['Category'].value_counts().index     #takes the indexes of the counted values
    x = data['Category'].value_counts()     #count each category and its repetence
    xsis = []
    ysis = []
    for i in range(len(x)):
        xsis.append(x[i])     #the value of repetence is appended
        ysis.append(y[i])     #the indexes are appended
```

```
[ ] plt.figure(figsize=(18,13))     #setting the graph size
    plt.xlabel("Count")
    plt.ylabel("Category")

    graph = sns.barplot(x = xsis, y = ysis, palette= "husl")     #in husl pallet i will cover all the colors in pallet
    graph.set_title("Top categories on Google Playstore", fontsize = 25);
```



Google Play Store.ipynb

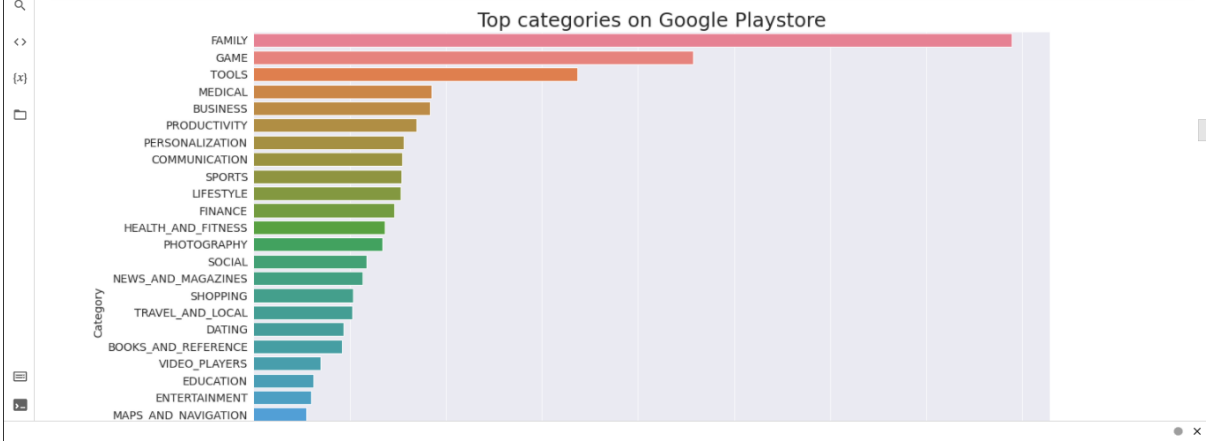
File Edit View Insert Runtime Tools Help Changes will not be saved

Share

+ Code + Text Copy to Drive

Connect Editing

```
[ ] graph.set_title("Top categories on Google Playstore", fontsize = 25);
```



Google Play Store.ipynb

File Edit View Insert Runtime Tools Help Changes will not be saved

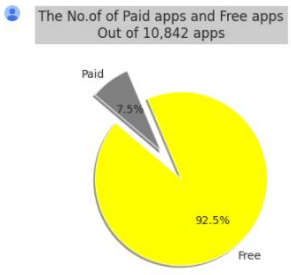
Share

+ Code + Text Copy to Drive

Connect Editing

Using Pie chart to show number of paid and free apps

```
[ ] #pie chart based on the app subsciption
    type = data['Type']
    free=0
    paid=0
    for i in range(10000):
      if type[i]=='Free':
        free=free+1
      elif type[i]=='Paid':      #counting the no.of paid apps and the free apps
        paid=paid+1
      else:
        continue
    data2=[free,paid]
    labels=['Free','Paid']
    colors = ["yellow", "grey"]
    explode = (0.4, 0)
    plt.pie(data2, explode=explode, labels=labels, colors=colors,autopct='%1.1f%%', shadow=True, startangle=140)
    plt.title("The No.of of Paid apps and Free apps\n" + "Out of 10,842 apps", bbox={'facecolor':'0.8', 'pad':5})
    plt.show()
```



Google Play Store.ipynb

File Edit View Insert Runtime Tools Help Changes will not be saved

Share

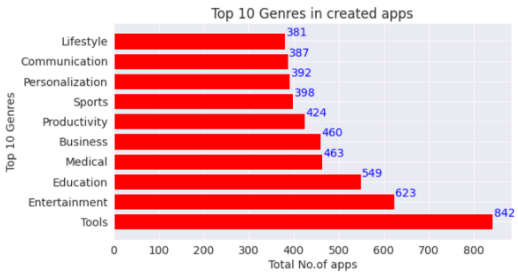
+ Code + Text Copy to Drive

Connect Editing

Using Horizontal bar graph to show top 10 genres

```
[ ] #Bar chart of top 10 Genres
    gen = data['Genres'].value_counts().head(10)      #counting the genres and taking the top 10 out of it
    data1= ['Tools','Entertainment','Education','Medical','Business','Productivity','Sports','Personalization','Communication','Lifestyle']
    x=data['Genres'].size
    y= gen
    fig,ax = plt.subplots()
    ax.barh(data1,y,color='r')
    for i, v in enumerate(y):
        ax.text(v + 3, i + .25, str(v), color='blue' )
    plt.title("Top 10 Genres in created apps")
    plt.xlabel('Total No.of apps')
    plt.ylabel('Top 10 Genres')
    print("Total No.Of apps Listes : ",x)
    print("")
    plt.show()
```

```
Total No.Of apps Listes :  10841
```



Google Play Store.ipynb

File Edit View Insert Runtime Tools Help Changes will not be saved

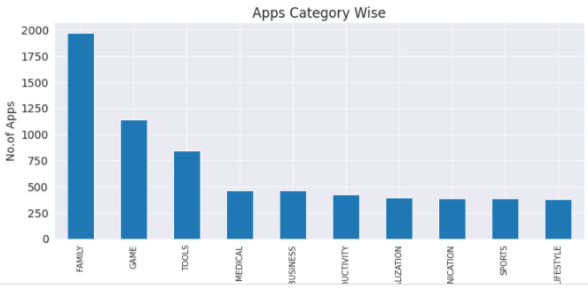
Share

+ Code + Text Copy to Drive

Connect Editing

Using Bar graph to show number of apps in category wise

```
[ ] plt.figure(figsize=(12,5))
    plt.title("Apps Category Wise")
    plt.ylabel('No.of Apps')
    plt.xlabel('Category')
    plt.xticks(rotation=60,fontsize=10)
    data['Category'].value_counts().head(10).plot(kind='bar')
    plt.show()
```



Google Play Store.ipynb

File Edit View Insert Runtime Tools Help Changes will not be saved

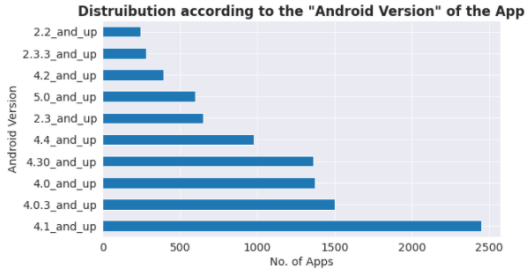
Share

+ Code + Text Copy to Drive

Connect Editing

Bar graph to show distribution according to android version of the app

```
[ ] plt.title('Distruibution according to the "Android Version" of the App',fontweight=900)
    plt.ylabel('Android Version')
    plt.xlabel('No. of Apps')
    data['Android_Ver'].value_counts().head(10).plot(kind='barh')
    plt.show()
```



Google Play Store.ipynb

File Edit View Insert Runtime Tools Help Changes will not be saved

Share

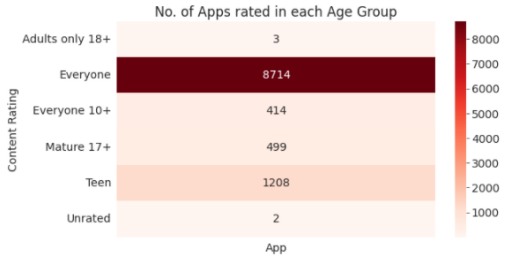
+ Code + Text Copy to Drive

Connect Editing

HeatMap to show Number of app rated in each age group

```
[ ] plt.title("No. of Apps rated in each Age Group")
    sns.heatmap(data.groupby('Content Rating')[['App']].count(),fmt="d", annot=True, cmap='Reds')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f446a78fc50>
```



Google Play Store.ipynb

File Edit View Insert Runtime Tools Help Changes will not be saved

Share

+ Code + Text Copy to Drive

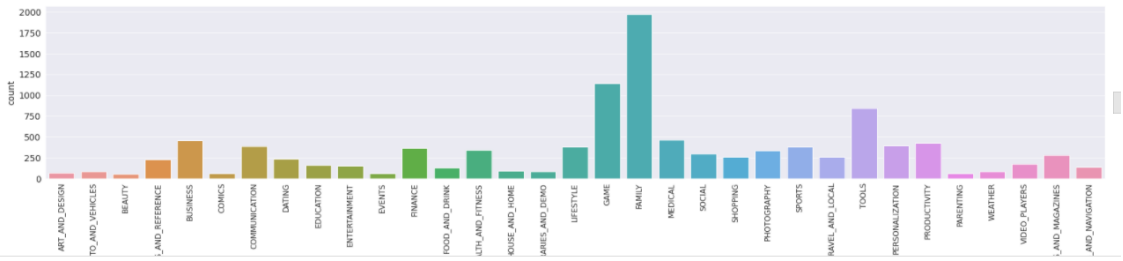
Connect Editing

Using plot to show number of apps in individual category

```
[ ] plt.figure(figsize=(30,5))
    gen = sns.countplot(data.Category)
    gen.set_xticklabels(gen.get_xticklabels(), rotation=90, fontsize=12)
    plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning:

Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will r
```



Google Play Store.ipynb

File Edit View Insert Runtime Tools Help Changes will not be saved

Share

+ Code + Text Copy to Drive

Connect Editing

Using plot to show paid apps in ascending order

```
[ ] plt.figure(figsize=(20, 7))
    paid_apps_df = data[data['Type'] == 'Paid'].sort_values(by=['Price'],ascending=True)
    plot_df = sns.countplot(paid_apps_df['Price'])
    plot_df.set_xticklabels(plot_df.get_xticklabels(), rotation=90, ha="right")
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning:

Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will

[Text(0, 0, '$0.99'),
 Text(0, 0, '$1.00'),
 Text(0, 0, '$1.04'),
 Text(0, 0, '$1.20'),
 Text(0, 0, '$1.26'),
 Text(0, 0, '$1.29'),
 Text(0, 0, '$1.49'),
 Text(0, 0, '$1.50'),
 Text(0, 0, '$1.59'),
 Text(0, 0, '$1.61'),
 Text(0, 0, '$1.70'),
 Text(0, 0, '$1.75'),
 Text(0, 0, '$1.76'),
 Text(0, 0, '$1.96'),
 Text(0, 0, '$1.97'),
 Text(0, 0, '$1.99'),
 Text(0, 0, '$10.00'),
 Text(0, 0, '$10.99'),
```