What Are The Features of a Successful Android App?

This program contains code for extracting, cleaning, and visualising data about 2.3 million+ google play store applications, as part of a research project to identify and evaluate features that make an android app successful. The dataset is from (https://www.kaggle.com/gauthamp10/google-playstore-apps (https://www.kaggle.com/gauthamp10/google-playstore-apps)) and it contains 23 attributes of 2.3 million+ android apps that were released in the google playstore from the year 2010 to 2021.

In [28]:
```
# Importing libraries: pandas, numpy, matplotlib, and plotly.express.
import pandas as pd
import numpy as np
import matplotlib.pyplot as pt
import plotly.express as px

# Reading the dataframe
# Dataset source (extracted on 8 August 2021): https://www.kaggle.com/gauthamp10/google-playstore-apps
df1 = pd.read_csv("Google-Playstore.csv")
```

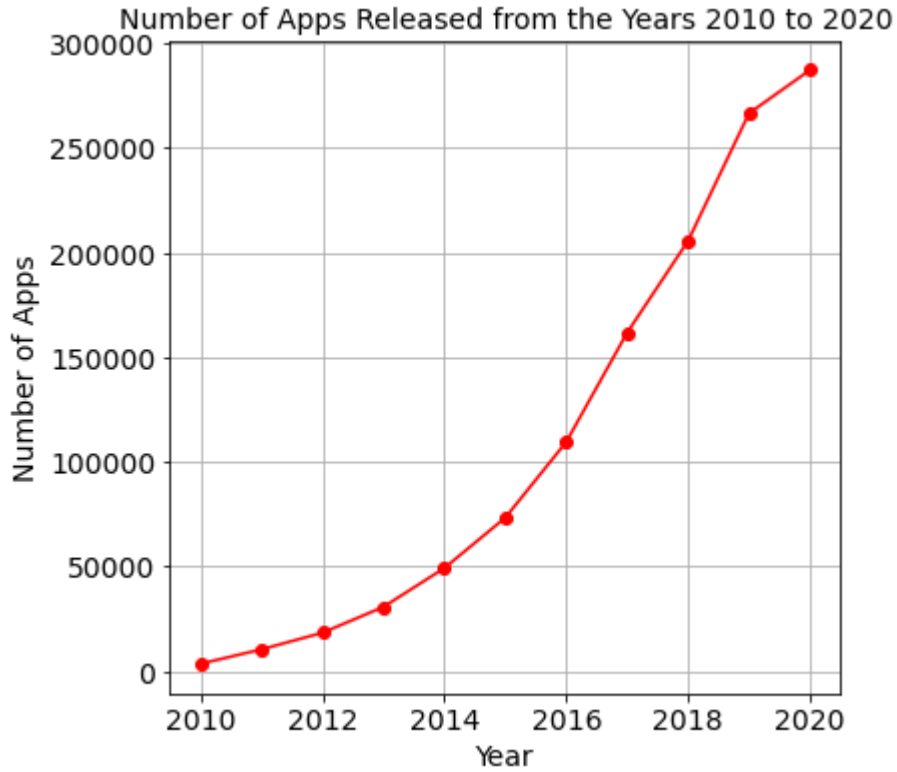
In [29]:
```
# Cleaning dataset, by removing the null values
df1.dropna(inplace=True)
```

In [30]:
```
# Date format of "Released" date of apps were changed to a format that is recognizabe by python
df1['Released'] = pd.to_datetime(df1['Released'], format='%b %d, %Y',
infer_datetime_format = True, errors='coerce')
```

In [31]:
```
# New column created to contain years of the app release date only
df1['Year released'] = pd.DatetimeIndex(df1['Released']).year
years = df1['Year released'].value_counts()

# Printing the years and the number of apps released each year
# This is done to use the information to create a line graph
print(years)
```

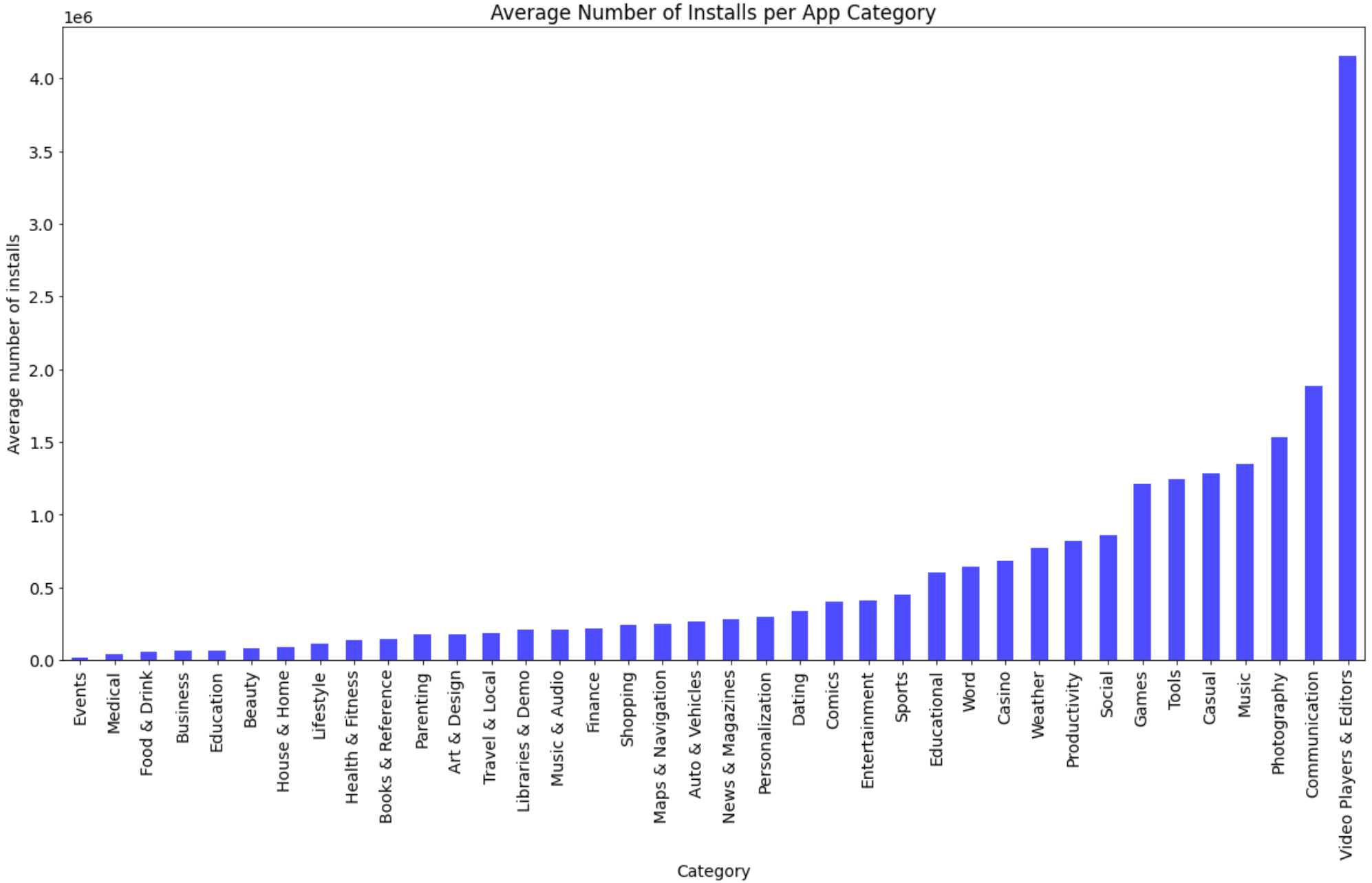
```
2020    287217
2019    266522
2018    205549
2017    161665
2016    109432
2015     73230
2021     71825
2014     49163
2013     30601
2012     18242
2011     10319
2010      3426
Name: Year released, dtype: int64
```

In [32]:
```
# Using the information, above array of years and count are created
yearR = [2020, 2019, 2018, 2017, 2016, 2015, 2014, 2013, 2012, 2011, 2010]
count = [287217, 266522, 205549, 161665, 109432, 73230, 49163, 30601, 18242, 10319, 3426]

# Using matplotlib for plotting a line graph to visualise the number of Apps Released from the Years 2010 to 2020
# where the x-axis is the array "yearR" containg the years
# and the y-axis is the array "count" which conatins the number of apps released in each year
pt.plot(yearR, count, color='red', marker='o')
pt.title('Number of Apps Released from the Years 2010 to 2020', fontsize=14)
pt.xlabel('Year', fontsize=14)
pt.ylabel('Number of Apps', fontsize=14)

#Displaying the line graph with grids
pt.grid(True)
pt.show()
```

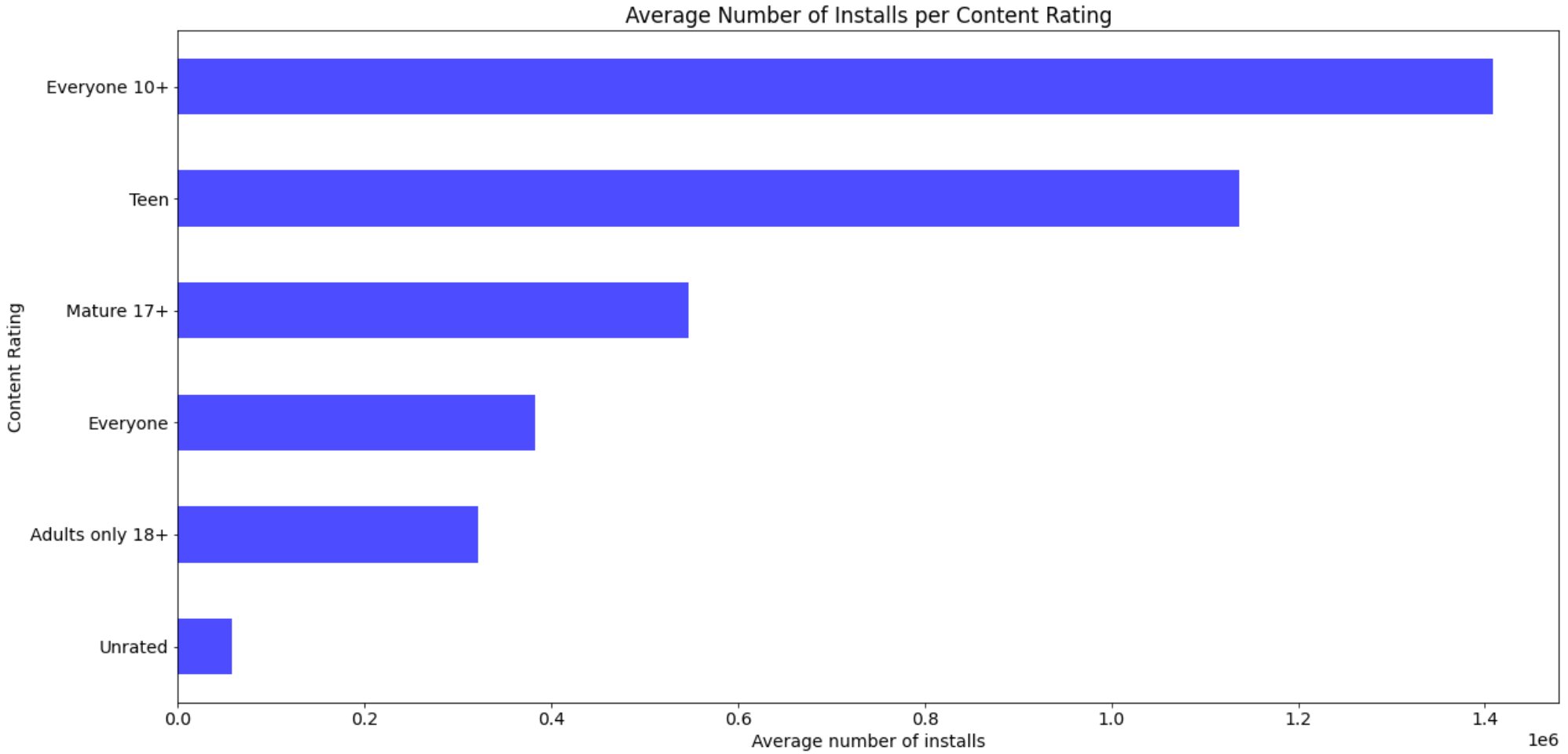


In [33]:
```
# To further organise the dataset, a new app category was created called "Games"
# and all the different gaming categories were combined into this one category
df1['Category'] = df1['Category'].replace({'Racing': 'Games', 'Puzzle':'Games', 'Arcade':'Games',
                                           'Simulation':'Games', 'Action':'Games', 'Adventure':'Games',
                                           'Trivia':'Games','Role Playing':'Games', 'Board':'Games', 'Card':'Games',
                                           'Strategy':'Games'})
```

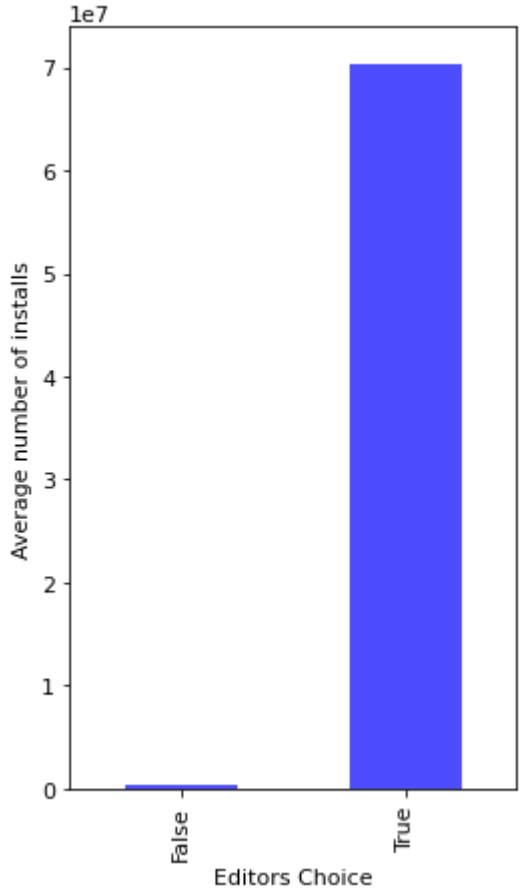
In [46]:
```
# Creating a variable "avg_installs" to store the average number of installs for each app category
avg_install = df1.groupby('Category')['Maximum Installs'].mean()

# Using matplotlib to create a bar plot comparing the average number of installs for each app category
pt.axes().set_facecolor("white")
pt.rcParams.update({'font.size': 14, 'figure.figsize': (20,10)})
pt.xlabel('Category')
pt.ylabel('Average number of installs')
avg_install.sort_values().plot(kind = "bar", color=(0.3,0.3,1,1), title = 'Average Number of Installs per App Category')
```


Average Number of Installs per App Category
1e6
Average number of installs
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Events
Medical
Food & Drink
Business
Education
Beauty
House & Home
Lifestyle
Health & Fitness
Books & Reference
Parenting
Art & Design
Travel & Local
Libraries & Demo
Music & Audio
Finance
Shopping
Maps & Navigation
Auto & Vehicles
News & Magazines
Personalization
Dating
Comics
Entertainment
Sports
Educational
Word
Casino
Weather
Productivity
Social
Games
Tools
Casual
Music
Photography
Communication
Video Players & Editors
Category

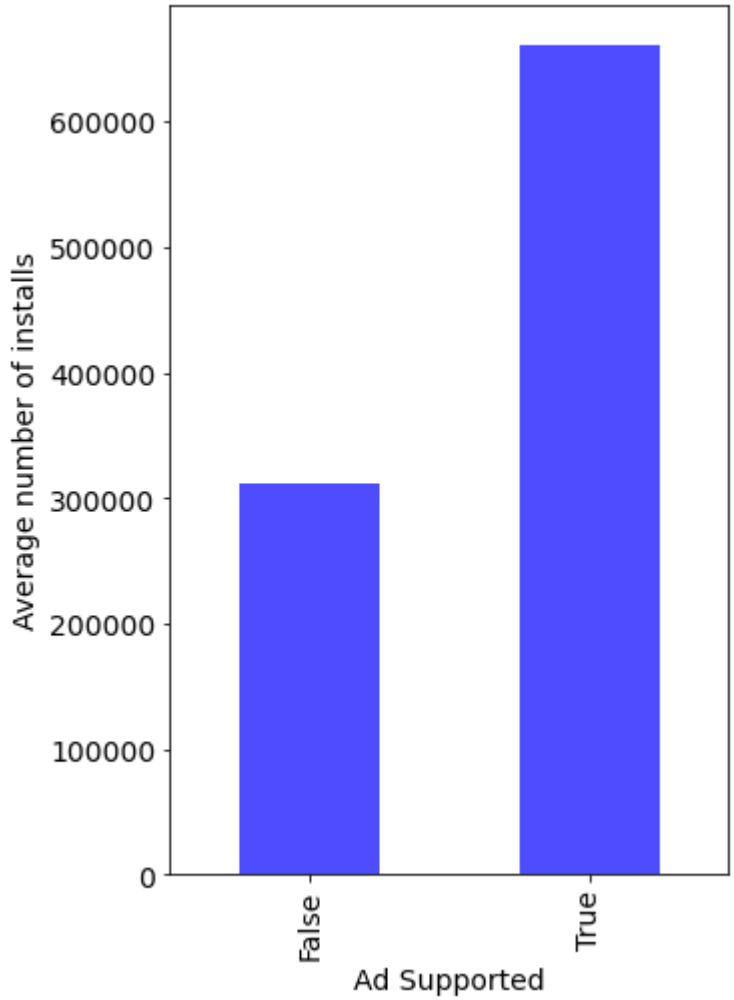
In [35]:
```
# Creating a variable to store the average number of installs for each type of app content rating
contentR_install = df1.groupby('Content Rating')['Maximum Installs'].mean()

# Using matplotlib, to create a barplot to show the Average number of installs per content rating of app
# where the x-axis is the average number of installs and y-axis is the app content rating
pt.axes().set_facecolor("white")
pt.rcParams.update({'font.size': 14, 'figure.figsize': (6, 6)})
pt.ylabel('Content Rating')
pt.xlabel('Average number of installs')
contentR_install.sort_values().plot(kind="barh",
                                    title = 'Average Number of Installs per Content Rating',color=(0.3,0.3,1,1));
```



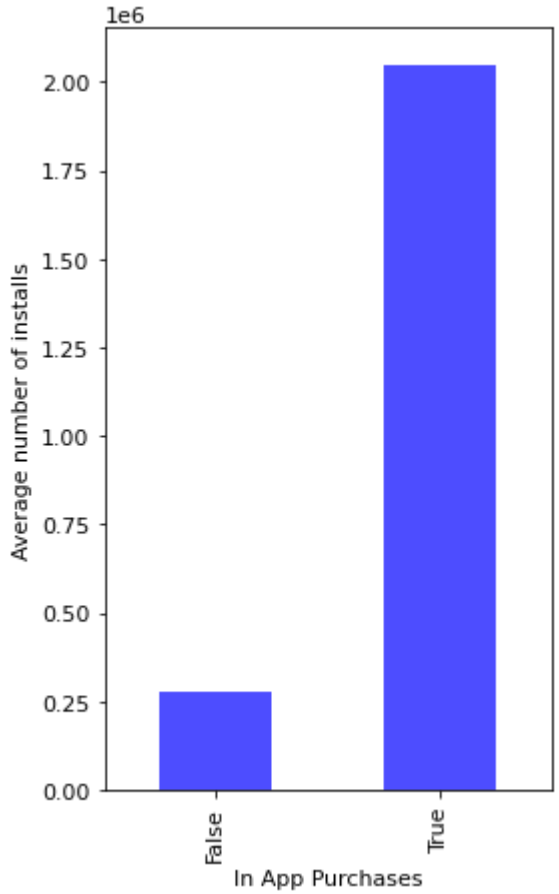
In [44]:
```
# Creating a variable to store the average number of installs for apps that are editor's choice and apps that are not
editorC_install = df1.groupby('Editors Choice')['Maximum Installs'].mean()

# Using matplotlib, to create a barplot to show the Average number of installs for apps that are editor's choice
# apps that are not
# The x-axis is true for apps that are editor's choice and false for apps that are not
# and the y-axis is the average number of installs
pt.axes().set_facecolor("white")
pt.rcParams.update({'font.size': 14, 'figure.figsize': (5, 9)})
pt.ylabel('Average number of installs')
pt.xlabel('Editors Choice')
editorC_install.sort_index().plot(kind = "bar",color=(0.3,0.3,1,1));
```



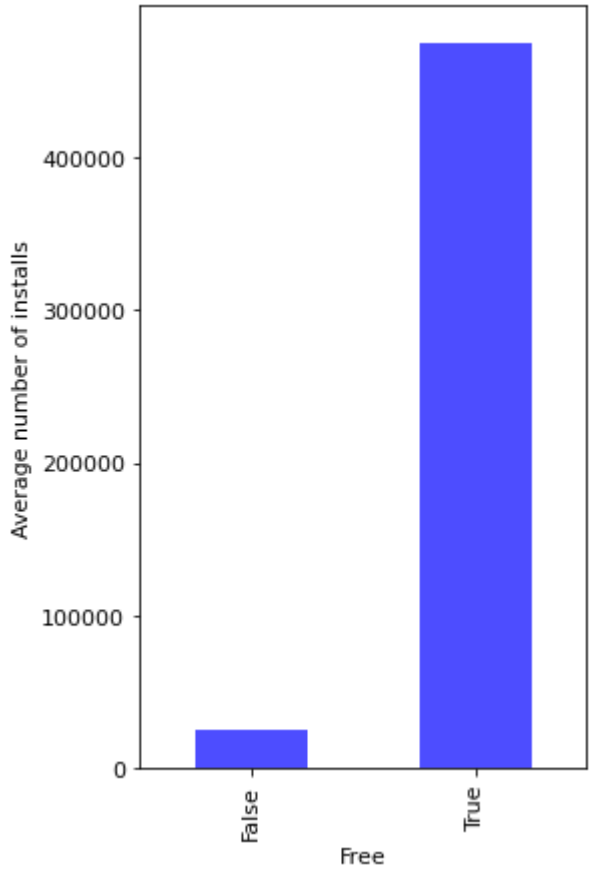
In [37]:
```
# Creating a variable to store the average number of installs for apps with and without ads
ads_install = df1.groupby('Ad Supported')['Maximum Installs'].mean()

# Using matplotlib, to create a barplot to show the Average number of installs for apps with and without adds
# where the x-axis is true for apps with ads and false for apps without ads
# and the y-axis is the average number of installs
pt.rcParams.update({'font.size': 14, 'figure.figsize': (5, 8)})
pt.xlabel('Ad Supported')
pt.ylabel('Average number of installs')
ads_install.sort_index().plot(kind = "bar",color=(0.3,0.3,1,1));
```



In [38]:
```
# Creating a variable to store the average number of installs for apps that have in-app purhases and apps that do not
in_app_purchase = df1.groupby('In App Purchases')['Maximum Installs'].mean()

# Using matplotlib, to create a barplot to show the Average number of installs for apps with and without in-app purchase
# where the x-axis is true for apps with in-app purchases and false for apps without in-app purchases
# and the y-axis is the average number of installs
pt.rcParams.update({'font.size': 11, 'figure.figsize': (4, 7)})
pt.ylabel('Average number of installs')
pt.xlabel('In App Purchases')
in_app_purchase.sort_index().plot(kind="bar",color=(0.3,0.3,1,1));
```



In [39]:
```
# Creating a variable to store the average number of installs for apps that are free to install and apps that are not
free = df1.groupby('Free')['Maximum Installs'].mean()

# Using matplotlib, to create a barplot to show the Average number of installs for free apps and paid apps
# where the x-axis is true for apps that are free to install and false for apps that are not free to install
# and the y-axis is the average number of installs
pt.rcParams.update({'font.size': 11, 'figure.figsize': (4, 7)})
pt.ylabel('Average number of installs')
pt.xlabel('Free')
free.sort_index().plot(kind="bar",color=(0.3,0.3,1,1));
```

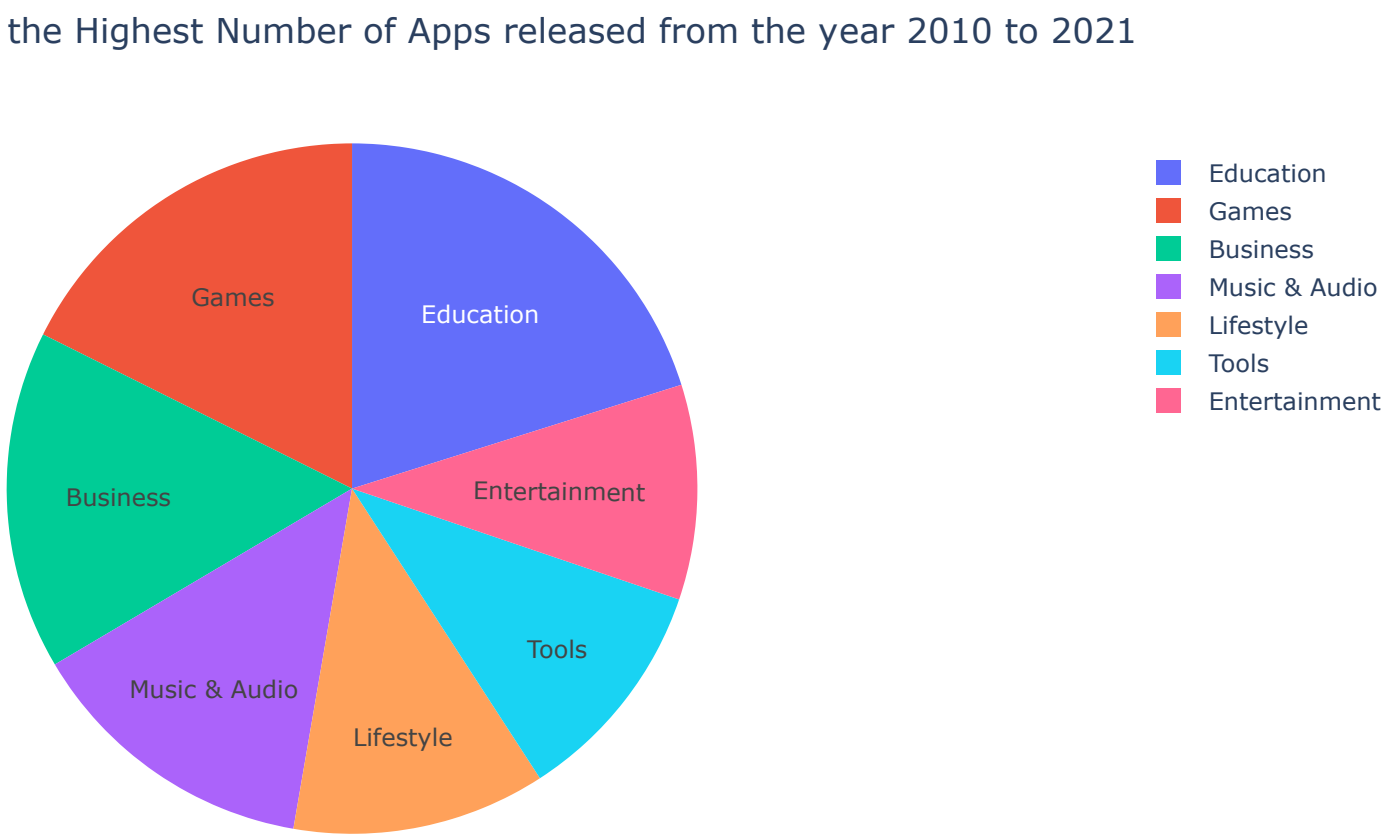


In [40]:
```
# Finding the number of apps that belong to each category
categories = df1['Category'].value_counts()

# Displaying the categories and total number of apps that belong to those categories
# This is done to use the information in a piechart
print(categories)
```

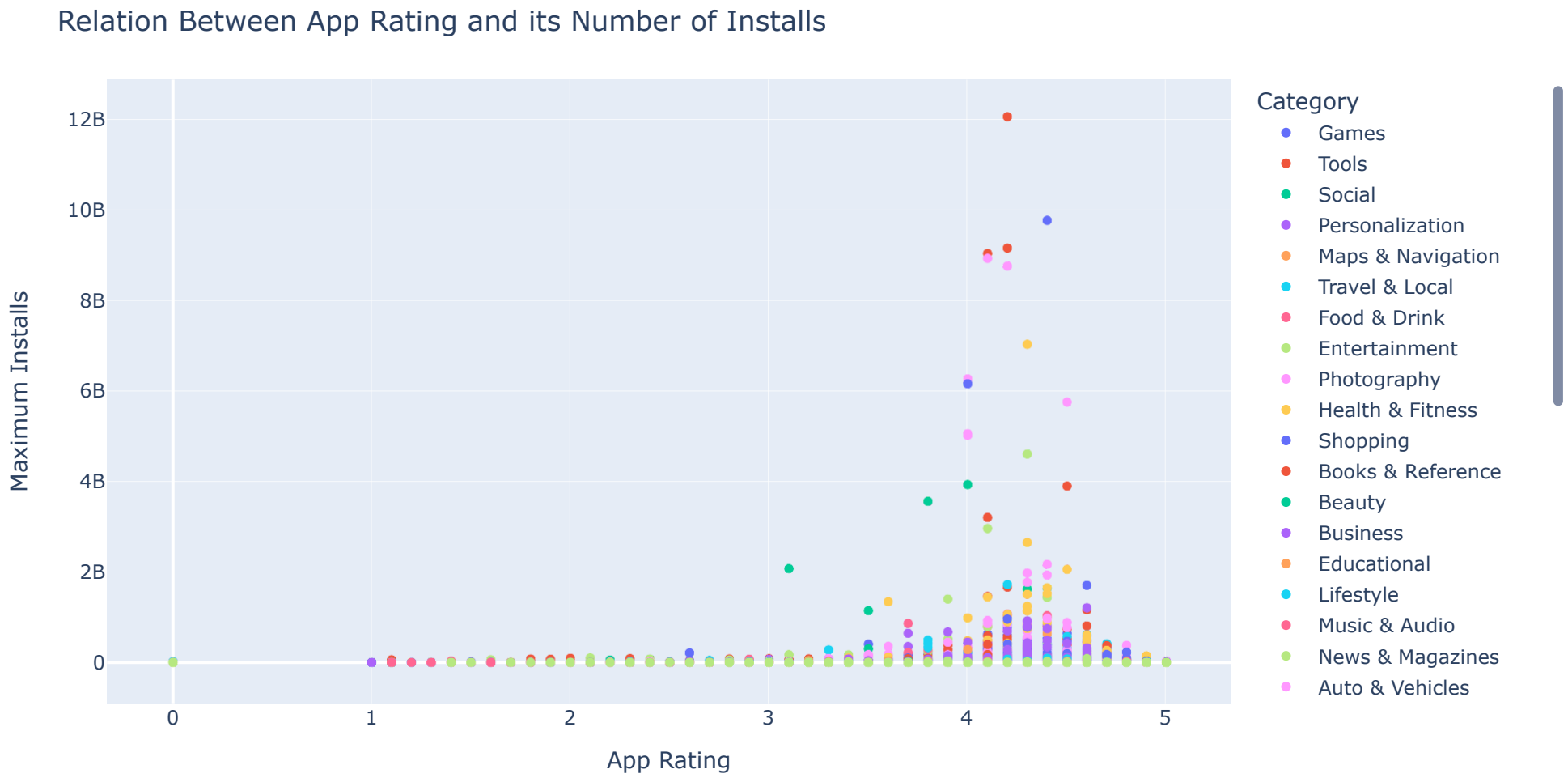
```
Education                 127219
Games                     111372
Business                  100311
Music & Audio              87008
Lifestyle                  75017
Tools                      67215
Entertainment              63581
Books & Reference          56515
Health & Fitness           51770
Shopping                   48981
Productivity               46960
Travel & Local             46613
Food & Drink               45773
Finance                    44111
Personalization            37819
Communication              30703
News & Magazines           29320
Sports                     29180
Social                     27164
Casual                     21958
Medical                    20260
Photography                16827
Maps & Navigation          15821
Educational                13048
Auto & Vehicles            10744
House & Home                8707
Events                      8497
Art & Design                8007
Video Players & Editors     6883
Beauty                      6179
Word                        4989
Weather                     4103
Dating                      3429
Casino                      2810
Parenting                   2381
Libraries & Demo            2371
Music                       2200
```

```
Comics                          1345
Name: Category, dtype: int64
```

In [41]:
```
# Using the information above, an interactive piechart is created
# that shows the top six category with the highest number of apps released from the year 2010 to 2021
genre = ['Education','Games','Business', 'Music & Audio','Lifestyle','Tools', 'Entertainment']
count = [127219,111372, 100311, 87008,75017, 67215, 63581]

# plotly.express is used to create the interactive piechart
# The piechart is interactive so upon hovering on each category sections, information about the exact number of apps
# belonging to the category is shown
fig = px.pie( values=count, names = genre, title='Top Six App Category with the Highest Number of Apps released from the
fig.update_traces(textposition='inside', textinfo='label')
fig.show()
```



In [42]:
```
# Interactive scatter plot created using plotly.express with x values as the app rating a
# nd y values as the number of installs, each data point on the scatter plot represents one app,
# and its colour is based on what category it belongs to

# Upon hovering to each data point, more information about the app can be seen
# such as whether its free, its content rating, if it has ads, if it includes in-app purchases
# and whether it's an editor's choice app
fig = px.scatter(df1, x="Rating", y="Maximum Installs", color="Category",
                 labels=dict(Rating="App Rating"),
                 hover_data =["Free","Content Rating", "Ad Supported", "In App Purchases", "Editors Choice"])
fig.update_layout(title='Relation Between App Rating and its Number of Installs')
fig.show()
```



In [43]:
```
# Interactive scatter plot created using plotly.express with x values as the year the app was released
# and y values as the number of installs, each data point on the scatter plot represents one app,
# and its colour is based on what category it belongs to

# Upon hovering to each data point, more information about the app can be seen such as whether its free,
# its content rating, if it has ads, if it includes in-app purchases and whether it's an editor's choice app
fig = px.scatter(df1, x="Released", y="Maximum Installs", color="Category",
                 labels=dict(Released="Release Year"),
                 hover_data =["Free","Content Rating", "Ad Supported", "In App Purchases", "Editors Choice"])
fig.update_layout(title='Number of App Installs and App Release Year')
fig.show()
```