Chandana Narla **G01211387**

Google Play Store- Data Analytics Project

Introduction:

Android is today's dominant mobile operating system with about 85% of all mo bile devices running Google's OS. Google Play Store is the largest and most pop ular Android app store.
The information from the Play Store applications has immense potential to drive the growth of app-making companies. The chosen dataset contains information about the Google Play Store apps. It is the web scraped data of 10k Play Store apps for analysing the Android market. It consists of in total of 10842 rows and 13 columns. This data set gives the detailed overview about the number of Apps hosted or supported by the Google play store along with the various other attributes like the Category of the App, Rating, Reviews, Number of Installations, Price and other attributes.

Data-Set Description:

This dataset is prepared and analysed by Lavanya Gupta who is a Machine Learning Engineer and Data Analyst at HSBC in India. The size of the chosen CSV file is **1.35 MB.** This data set consists of **13 columns** and a total of **10842 rows**. This dataset includes columns like

App-This column gives the name of the applications (Apps) present on the Google store. This comes under the Nominal Data Type

Category- This column gives the information related to the type or category under which the respective app belongs to. This is also a Nominal Data Type

Rating- Overall user rating for each application is given under this column. This also gives details related to the app popularity. This comes under the Ratio Data Type

Reviews- Number of reviews for each app. This comes under the Integer Data Type

Size- Details related to the memory used by the specific application is given. This comes under the Nominal Data Type

Installs- Number of users who have already downloaded the App is given. This comes under the Nominal Data Type

Type- Information about whether the app is paid or free. This comes under the Nominal Data Type

Price- Price of the respective App is given. This comes under the Ratio Data Type

Content Rating- Gives the age groups to which the particular app is targeted to. This comes under the Nominal Data Type

Genres- Any app can belong to multiple genres apart for the main category. This comes under the Nominal Data Type

Last Updated- Date when the application was last updated by Google in it play store. This comes under the Ordinal Data Type

Current Ver- The current version available on the play store related details are given in this column. This comes under the Nominal Data Type

Android Ver- What is the minimum android version required to download that particular app is given in this column. This comes under the Nominal Data Type.[1]

Languages Used:

I have used **Python** for data cleaning, visualizations and for regression analysis. For certain other visualizations, **Tableau** software is used. Created a schema, imported the cleaned csv file on the **SQL** and performed basic queries to analyse the data-set

Data-Set Cleaning:

After loading the data-set and checking for each data type of the variables, we could see that the size, installs and price are of object data types. Hence in-order to use these variables and produce some visualizations and further calculations, we need to normalise the values and clean the data. I have used the Map lambda functions for data cleaning

```
In [11]: googlePlayStore= pd.read_csv('C:/Users/chand/Documents/ChandanaNarla/chandana/CourseWork/Sem_1/Assignments/AIT580/AITFinal/gc

googlePlayStore
```

Out[11]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19M	10,000+	Free	0	Everyone	Art & Design	January 7, 2018	1.0.0	4.0.3 and up
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14M	500,000+	Free	0	Everyone	Art & Design;Pretend Play	January 15, 2018	2.0.0	4.0.3 and up
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8.7M	5,000,000+	Free	0	Everyone	Art & Design	August 1, 2018	1.2.4	4.0.3 and up
	Sketch -										June 8	Varies	4.2 and

[1] (Gupta, 2019)

```
In [53]: googlePlayStore.dtypes

Out[53]: 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
```

- For the column **Size,** we see that the App size related information is given in this column. So, the app size varies from KB to MB. Here, I have decided to convert all the values into MB and remove the unit MB so that we could make the column uniform and convert it into a numeric data type for further visualizations. Here, I have used the **rstrip()** and **strip()** functions to remove the units and also converted the KB into MB by dividing the value with 1024. Some columns contain 'Varies with device' as the size value which is treated as Null value.

```
In [44]: #Data cleaning for columns like size and installs and make them Ratio/ Integer from nominal data types for more accurate resu
googlePlayStore['Size']=googlePlayStore['Size'].map(lambda x: x.rstrip('M')) #rstrip removes the right arguments

#here we are converting the KB into MB
#here we use strip() which removes the given arguments in the data
googlePlayStore['Size']=googlePlayStore['Size'].map(lambda x: str(round((float(x.strip('k'))/1024),1)) if x[-1]=='k' else x)

googlePlayStore['Size']=googlePlayStore['Size'].map(lambda x: np.nan if x.startswith('Varies') else x)

googlePlayStore['Size']    #All the App sizes are now commonly measured in MB

Out[44]: 0                    19
         1                    14
         2                   8.7
         3                    25
         4                   2.8
                      ...
         10836                53
         10837               3.6
         10838               9.5
         10839    Varies with device
         10840                19
         Name: Size, Length: 10841, dtype: object
```

- For column **Installs**, we see that the **+** symbol does not allow the column to be a numeric datatype thus making it an object datatype. So we use the **rstrip()** function to remove the '+' symbol and also using the **split()** function to remove the ','

```
In [27]: googlePlayStore['Installs']=googlePlayStore['Installs'].map(lambda x: x.rstrip('+'))
         googlePlayStore['Installs']=googlePlayStore['Installs'].map(lambda x: ''.join(x.split(',')))
         googlePlayStore['Installs']

Out[27]: 0          10000
         1         500000
         2        5000000
         3       50000000
         4         100000
                   ...
         10836       5000
         10837        100
         10838       1000
         10839       1000
         10840    10000000
         Name: Installs, Length: 10841, dtype: object
```

- For column **Price**, we need to remove the **'$'** symbol so that we could use the price values for various further predictions. For this we use the **lstrip()** function as we have the $ symbol towards the left of the value.

```
In [47]: googlePlayStore['Price']=googlePlayStore['Price'].map(lambda x: x.lstrip('$').rstrip())
         googlePlayStore['Price']

Out[47]: 0        0
         1        0
         2        0
         3        0
         4        0
                  ..
         10836    0
         10837    0
         10838    0
         10839    0
         10840    0
         Name: Price, Length: 10841, dtype: object
```

- Here using the **isnull()** function, we could find that there NULL value is the dataset. Hence by dropping the Null values using the **dropna()** function we remove the null values and clean the data set.

```
In [20]: googlePlayStore.isnull().sum()
```

```
Out[20]: App                0
         Category           0
         Rating          1474
         Reviews            0
         Size            1695
         Installs           0
         Type               1
         Price              0
         Content Rating     1
         Genres             0
         Last Updated       0
         Current Ver        8
         Android Ver        3
         dtype: int64
```

```
In [21]: googlePlayStore=googlePlayStore.dropna()
         googlePlayStore
```

		Book											
...	...	...	...	...	...	...	...	...	...	...	...	...	...
10833	Chemin (fr)	BOOKS_AND_REFERENCE	4.8	44	619k	1000	Free	0	Everyone	Books & Reference	March 23, 2014	0.8	2.2 and up
10834	FR Calculator	FAMILY	4.0	7	2.6	500	Free	0	Everyone	Education	June 18, 2017	1.0.0	4.1 and up
10836	Sya9a Maroc - FR	FAMILY	4.5	38	53	5000	Free	0	Everyone	Education	July 25, 2017	1.48	4.1 and up
10837	Fr. Mike Schmitz Audio Teachings	FAMILY	5.0	4	3.6	100	Free	0	Everyone	Education	July 6, 2018	1	4.1 and up
10840	iHoroscope - 2018 Daily Horoscope & Astrology	LIFESTYLE	4.5	398307	19	10000000	Free	0	Everyone	Lifestyle	July 25, 2018	Varies with device	Varies with device

7723 rows × 13 columns

- Hence, we have a total of **7723** records after removing the null values.
- Data types after conversion into integer/float data types using the **to_numeric()** function.
- The cleaned data set is exported into another csv file and this file is used as the input file for **Tableau** and **SQL.**

```
In [ ]: #to convert the required columns into numeric / integers for better data visualizations and regression
        googlePlayStore['Size']=pd.to_numeric(googlePlayStore['Size'])
        googlePlayStore['Installs']=pd.to_numeric(googlePlayStore['Installs'])
        googlePlayStore['Price']=pd.to_numeric(googlePlayStore['Price'])
        googlePlayStore['Reviews']=pd.to_numeric(googlePlayStore['Reviews'])
```

In [61]:
```
googlePlayStore.dtypes
```

Out[61]:
```
App                object
Category           object
Rating            float64
Reviews             int64
Size              float64
Installs            int64
Type               object
Price             float64
Content Rating     object
Genres             object
Last Updated       object
Current Ver        object
Android Ver        object
dtype: object
```

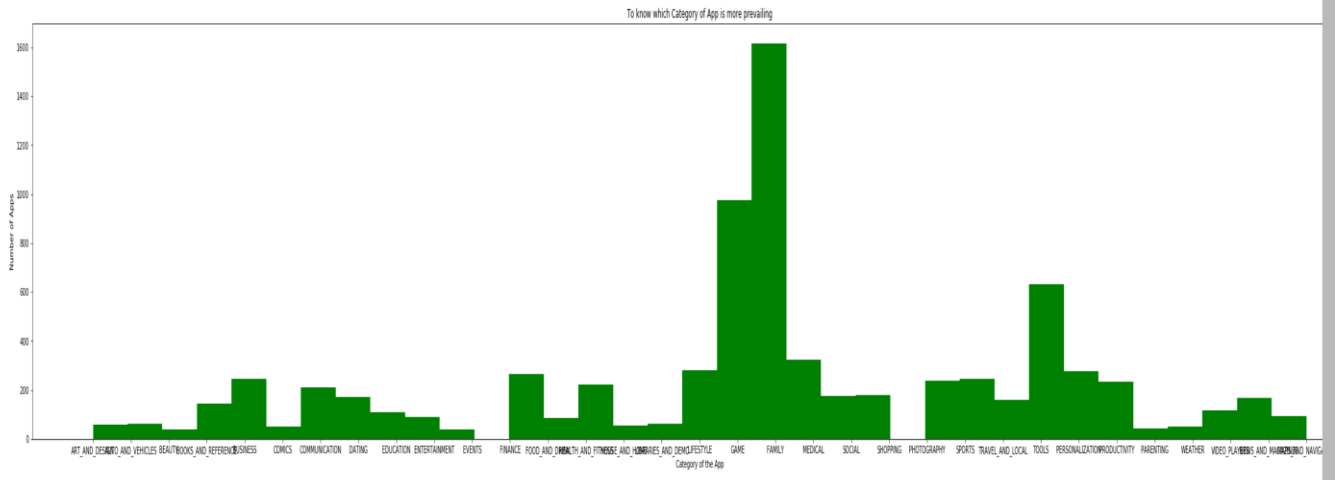
Visualizations:

For better understanding of the data-set, we go for visualisations in-order to understand the relation between few attributes.

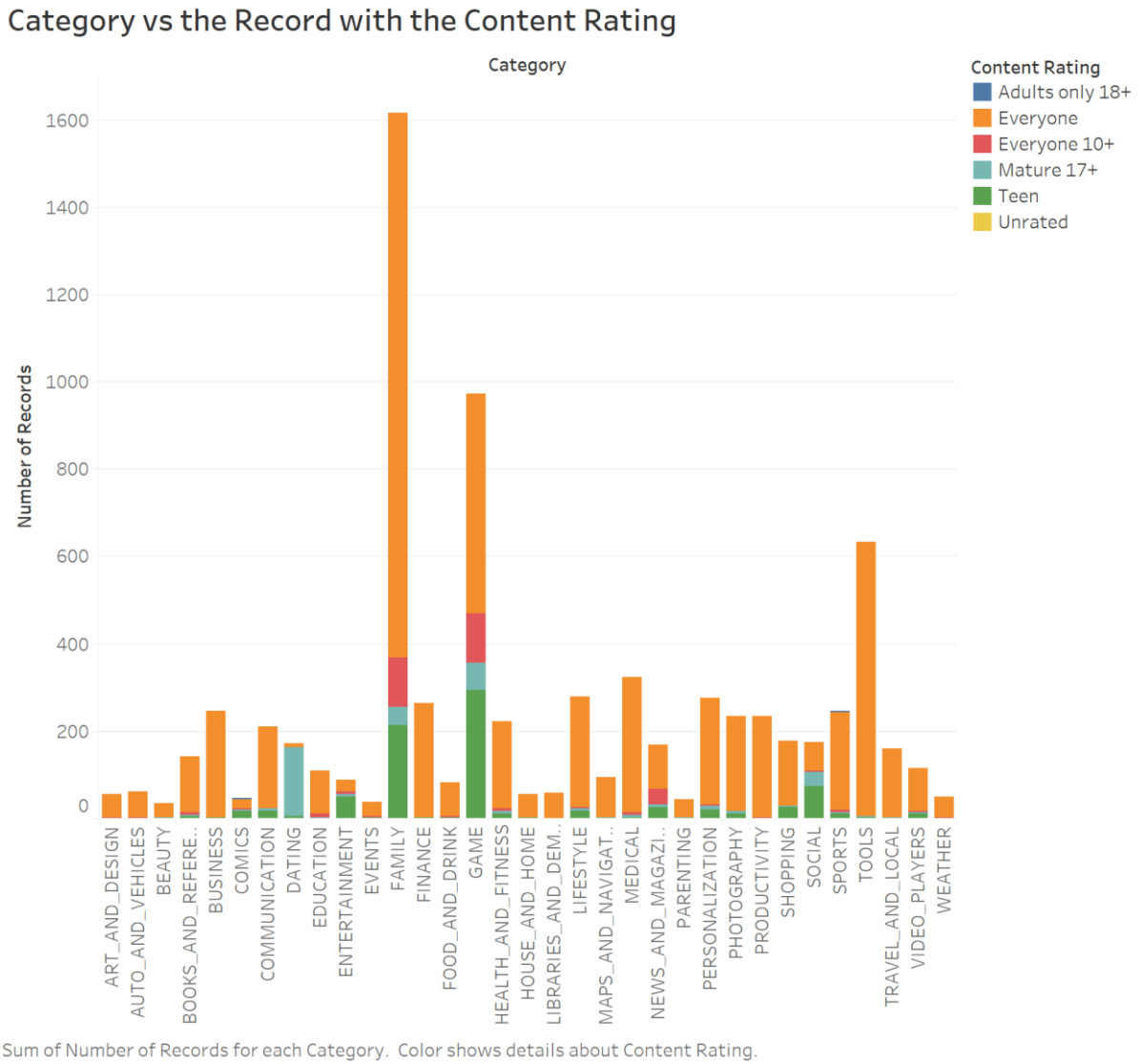
- **Figure1:** From the below histogram figure, we could understand that the most prevailing category among the apps is **Family.** And the next highest category of Apps is **Games.** Hence, the most popular category among the Apps is **Family**.

```
plt.figure(figsize=(50,8))

plt.hist(googlePlayStore['Category'],color='Green',bins=35)
plt.xlabel('Category of the App')
plt.ylabel('Number of Apps')
plt.title('To know which Category of App is more prevailing')
```

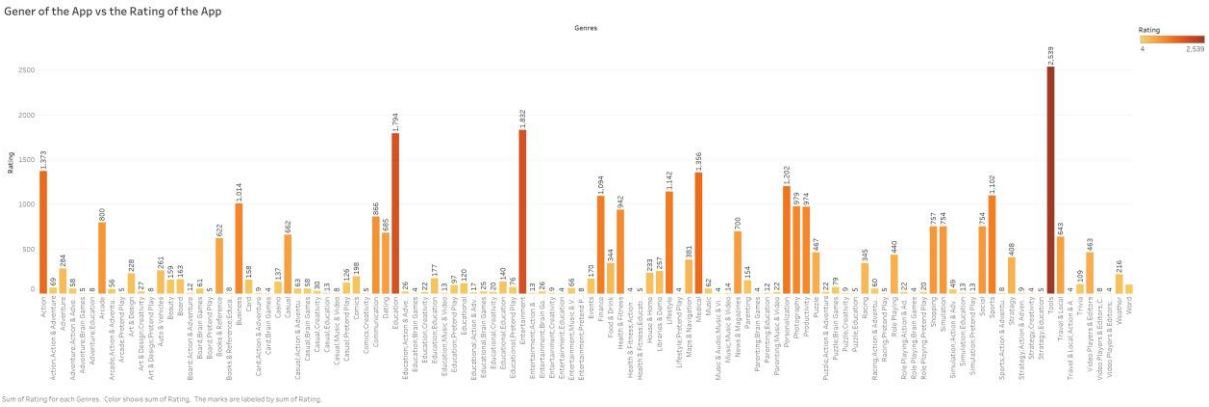
```
Text(0.5, 1.0, 'To know which Category of App is more prevailing')
```



- **Figure2:** The below bar graph shows the category vs the number of records with respect to Content Rating. From the below we could interpret that **Family** type of app is accessed and downloaded by **Everyone**. And the options for kids with 10+ years and Teens are comparatively next most prevailing apps. Whereas the Games are distributed among all the age groups and specially Teens. The legend gives the color coding for each type of age termed as content Rating.

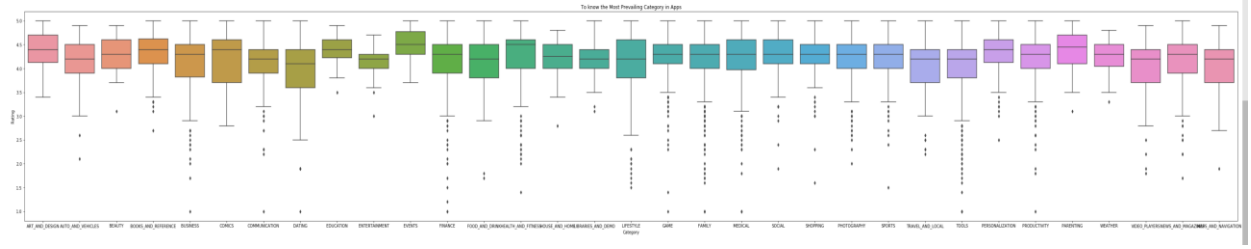


- **Figure3:** The below graph gives the colour variation from the least to the highest rating for the App in a genre. Hence the App which has the highest rating is tools to the least in various other genres.



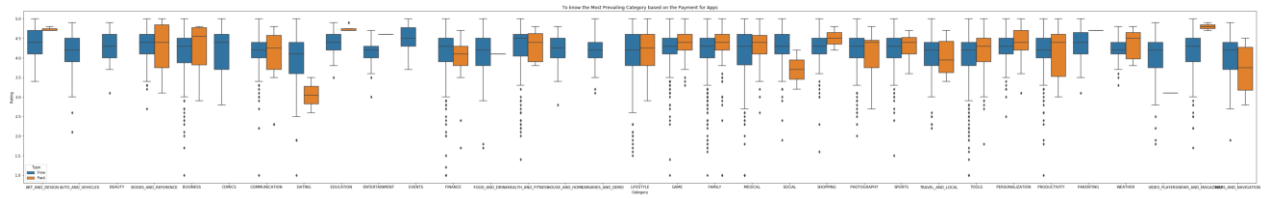
- **Figure4:** The below graph shows the specific category and its respective rating for that App to indicate the most prevailing app. This shows shape of the distribution, its central value, and its variability.

```
plt.figure(figsize=(60,9))
sn.boxplot(x='Category', y='Rating',data=googlePlayStore)
plt.title('To know the Most Prevailing Category in Apps')
plt.show()
```



- **Figure5:** The below graph shows the specific category and its respective rating for that App to indicate the most prevailing app and how is it depended on the Type i.e., whether the app is paid or not. This shows shape of the distribution, its central value, and its variability with respect to the app being paid or not.

```
plt.figure(figsize=(60,9))
sn.boxplot(x='Category', y='Rating',hue='Type',data=googlePlayStore)
plt.title('To know the Most Prevailing Category based on the Payment for Apps')
plt.show()
```



- **Figure6: Correlation Analysis:**

The below figure gives the correlation between each of the numeric variables present in the data-set. This tables gives the relation between each of the variables. Also how much the variables are dependent on each other. Positive values close to 1 indicates the positive correlation and values close to -1 indicates the negative correlation. From the below we can interpret that the **Installation**/ Download of the App is dependent on the **Reviews** for that App with **0.62** and the correlation coefficient.

```
In [90]: googlePlayStore.corr()
```

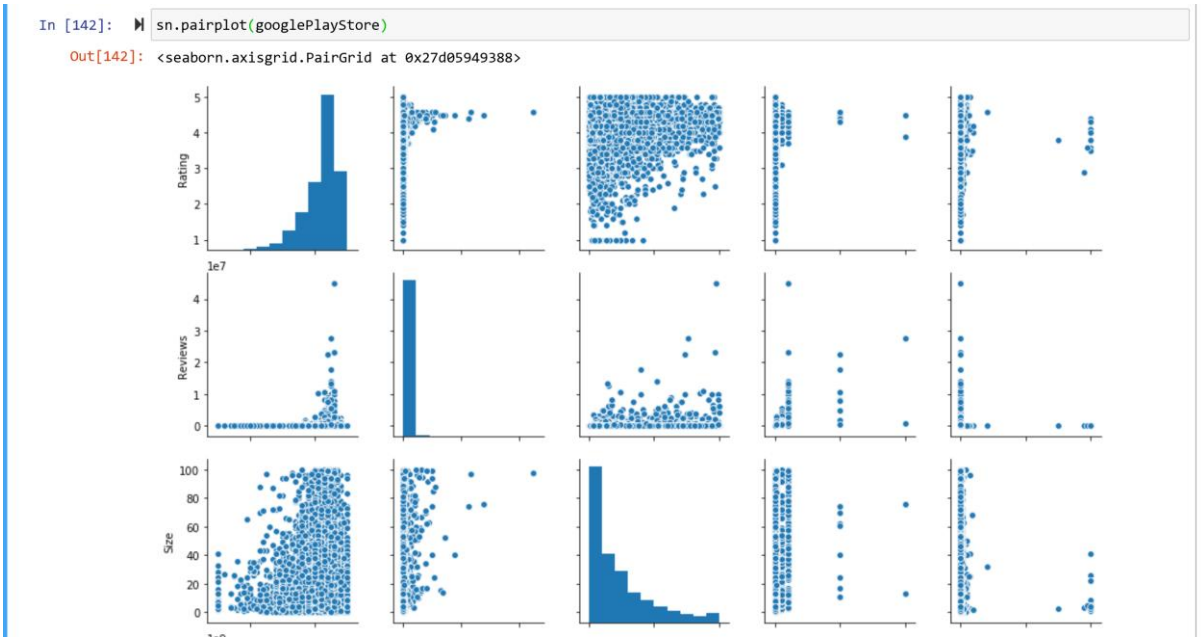
Out[90]:

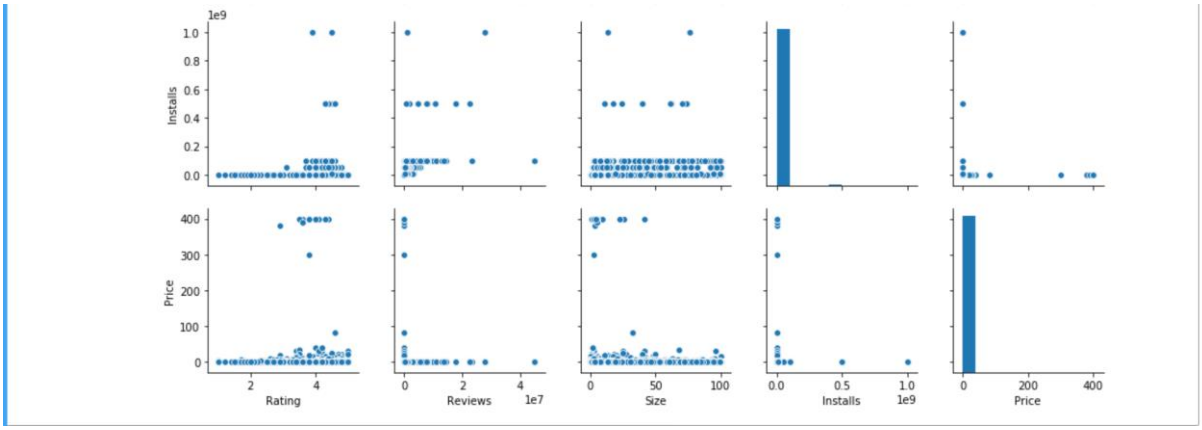
	Rating	Reviews	Size	Installs	Price
Rating	1.000000	0.079819	0.083645	0.052693	-0.021320
Reviews	0.079819	1.000000	0.240381	0.626187	-0.010184
Size	0.083645	0.240381	1.000000	0.162707	-0.026279
Installs	0.052693	0.626187	0.162707	1.000000	-0.010852
Price	-0.021320	-0.010184	-0.026279	-0.010852	1.000000

- **Figure7:** Below pair plots show the relation between single variables and relationship between two variables. This visually represents the above correlation between the variables with respect to each other.

```
In [142]: sn.pairplot(googlePlayStore)
```

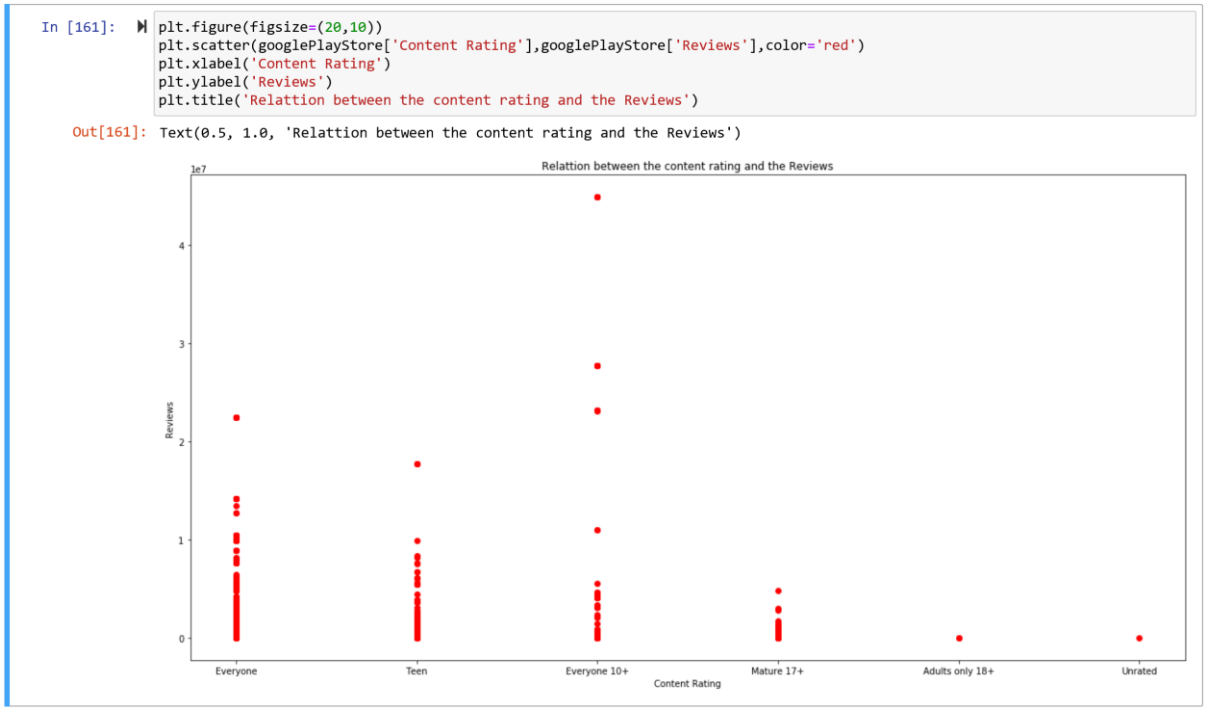
Out[142]: <seaborn.axisgrid.PairGrid at 0x27d05949388>





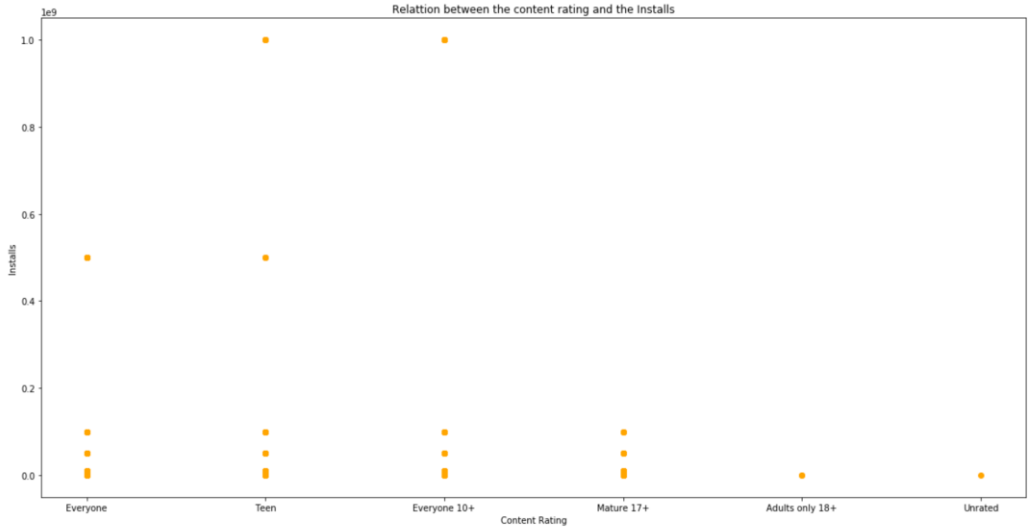
- **Figure8- Scatter Plot with respect to Content Rating:**

The below two scatter plots are plotted with respect to the Content Rating by comparing its relationship with the Reviews and Installs. From the below graph we can interpret that the Age group and reviews are related but has few outliers. But the there is weak relation between the Age and the Installations.



In [162]:
```
plt.figure(figsize=(20,10))
plt.scatter(googlePlayStore['Content Rating'],googlePlayStore['Installs'],color='Orange')
plt.xlabel('Content Rating')
plt.ylabel('Installs')
plt.title('Relattion between the content rating and the Installs')
```

Out[162]: Text(0.5, 1.0, 'Relattion between the content rating and the Installs')



Linear Regression-Regression Analysis:

For the regression analysis, Linear regression is the best method. Hence for the chosen data-set to apply Linear Regression, I have imported the **statsmodels** package. Here we are going to Target the Rating of the App and how is it depending on the price of the App, Size and Content Rating (Age) for the Apps. The below regression is conducted on the 97.5% of confidence level. After conducting the regression, we could see that the R square value is 0.72. Also, from the results, we can deduce that increase in 1 unit of Content rating will affect the Rating by **1.47 times**. And the P value is below 0.05, hence we reject the Null hypothesis.

In [302]:
```
label_encoder=preprocessing.LabelEncoder()
googlePlayStore['Content Rating']=label_encoder.fit_transform(googlePlayStore['Content Rating'])

targetcord=pd.DataFrame(googlePlayStore['Rating'])
X1=googlePlayStore[['Price','Size','Content Rating']]
Y=targetcord['Rating']
model_regression1=sm.OLS(Y,X1).fit()
model_predictions=model_regression1.predict(X1)
model_regression1.summary()
```

Out[302]:

OLS Regression Results

Dep. Variable:	Rating	**R-squared (uncentered):**	0.720
Model:	OLS	**Adj. R-squared (uncentered):**	0.720
Method:	Least Squares	**F-statistic:**	6632.
Date:	Sat, 23 Nov 2019	**Prob (F-statistic):**	0.00
Time:	19:37:04	**Log-Likelihood:**	-17137.
No. Observations:	7723	**AIC:**	3.428e+04
Df Residuals:	7720	**BIC:**	3.430e+04
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Price	0.0061	0.001	4.178	0.000	0.003	0.009
Size	0.0371	0.001	36.387	0.000	0.035	0.039
Content Rating	1.4822	0.019	79.055	0.000	1.445	1.519

Omnibus:	1467.339	**Durbin-Watson:**	1.112
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	2446.561
Skew:	-1.300	**Prob(JB):**	0.00
Kurtosis:	3.917	**Cond. No.**	24.3

By considering Rating tagret and chosing the Genre as one of the additional predictor, Then the R square value goes to 0.87 strong which decreases the dependency on content Rating to 0.77. Greater R value means stronger the dependancy of the target variable by the independent variable.

```
googlePlayStore['Genres']=label_encoder.fit_transform(googlePlayStore['Genres'])


targetcord=pd.DataFrame(googlePlayStore['Rating'])
X2=googlePlayStore[['Price','Size','Content Rating','Genres']]
Y=targetcord['Rating']
model_regression2=sm.OLS(Y,X2).fit()
model_predictions=model_regression.predict(X2)
model_regression2.summary()
```

Out[304]:

OLS Regression Results

Dep. Variable:	Rating	**R-squared (uncentered):**	0.870
Model:	OLS	**Adj. R-squared (uncentered):**	0.870
Method:	Least Squares	**F-statistic:**	1.293e+04
Date:	Sat, 23 Nov 2019	**Prob (F-statistic):**	0.00
Time:	19:38:30	**Log-Likelihood:**	-14177.
No. Observations:	7723	**AIC:**	2.836e+04
Df Residuals:	7719	**BIC:**	2.839e+04
Df Model:	4		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Price	0.0027	0.001	2.733	0.006	0.001	0.005
Size	0.0222	0.001	31.146	0.000	0.021	0.024
Content Rating	0.7773	0.015	52.496	0.000	0.748	0.806
Genres	0.0338	0.000	94.310	0.000	0.033	0.035

Omnibus:	179.778	**Durbin-Watson:**	1.401
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	191.957
Skew:	-0.382	**Prob(JB):**	2.08e-42
Kurtosis:	3.107	**Cond. No.**	60.9

Hypothesis Testing:

Hypothesis Testing gives the rate at which the variables are dependent on each other. We use the **stats** package inorder to do the hypothesis testing

Test1: The Pvalue for Content Rating and Price after conducting the hypothesis testing is 0.08 which is greater 0.05 hence, the null hypothesis testing is accepted which means that the chosen two variables are not dependent on each other.

Test2: The Pvalue for Category and Price after conducting the hypothesis testing is 0.0 which is less than 0.05 hence, the null hypothesis testing is rejected which means that the chosen two variables are dependent on each other.

Test3: The Pvalue for Category and Installs after conducting the hypothesis testing is almost 0.0 which is less than 0.05 hence, the null hypothesis testing is rejected which means that the chosen two variables are dependent on each other.

In [319]:
```
data_hypothesis1=googlePlayStore[['Content Rating','Price']]

ttest1,pval1=stats.ttest_rel(data_hypothesis1['Content Rating'],data_hypothesis1['Price'])
print("Pvalue is "+ pval1.astype(str))
if pval1<0.05:
    print("We reject null hypothesis")
else:
    print("We accept null hypothesis")
```

```
Pvalue is 0.08171811338522678
We accept null hypothesis
```

In [320]:
```
data_hypothesis2=googlePlayStore[['Category','Price']]

ttest2,pval2=stats.ttest_rel(data_hypothesis2['Category'],data_hypothesis2['Price'])
print("Pvalue is "+ pval2.astype(str))
if pval2<0.05:
    print("We reject null hypothesis")
else:
    print("We accept null hypothesis")
```

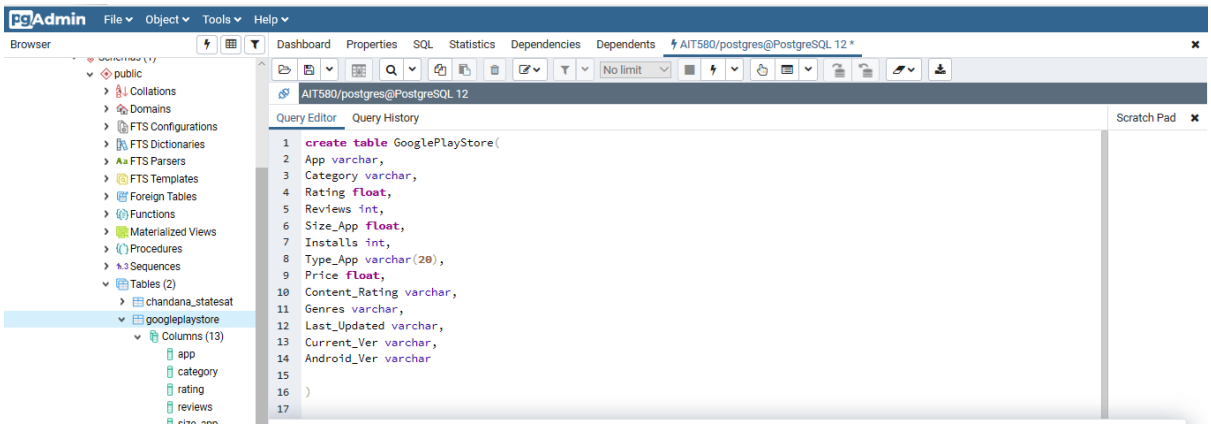
```
Pvalue is 0.0
We reject null hypothesis
```

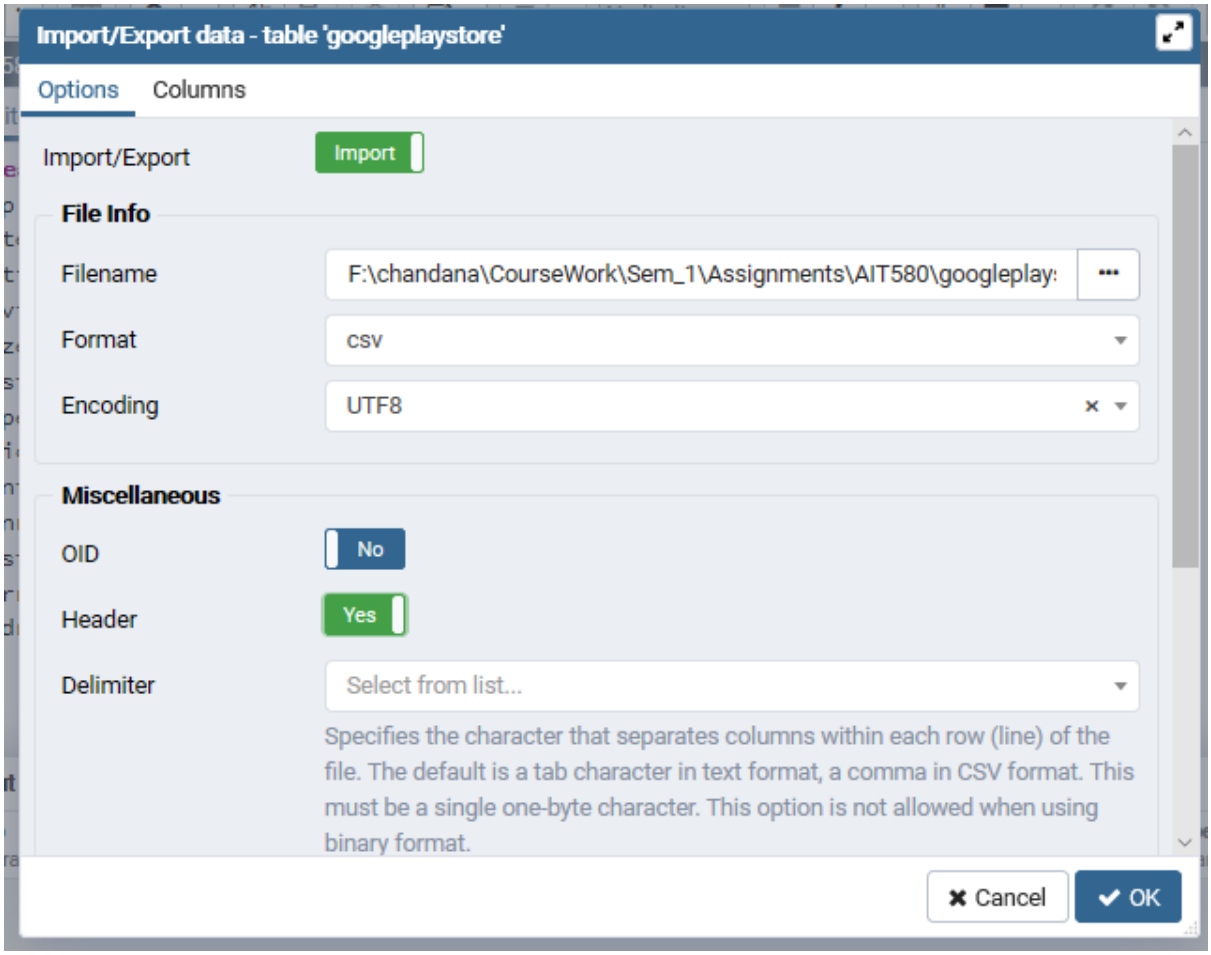
In [322]:
```
data_hypothesis3=googlePlayStore[['Category','Installs']]

ttest3,pval3=stats.ttest_rel(data_hypothesis3['Category'],data_hypothesis3['Installs'])
print("Pvalue is "+ pval3.astype(str))
if pval3<0.05:
    print("We reject null hypothesis")
else:
    print("We accept null hypothesis")
```

```
Pvalue is 1.2199330833054274e-48
We reject null hypothesis
```

SQL Schema and Analysis

Initially a table is created in the AIT580 database. The cleaned data set is imported into the table created with all the attributes. Below screenshots depict the process of creating schema, table and importing the values.





- Below figure shows that by executing the select query, all the 7723 records are loaded into the table **GooglePlayStore**

```
17
18  select * from GooglePlayStore
19
```

Data Output | Explain | Notifications | Messages | Read-only column

	app character varying	category character varying	rating double precision	reviews integer	size_app double precision	installs integer	type_app character varying (20)	price double precision	content_rating character varying
7708	Fr Agnel Pune	FAMILY	4.1	80	13	1000	Free	0	Everyone
7709	FR: My Secret Pets!	FAMILY	4	785	31	50000	Free	0	Teen
7710	Golden Dictionary (FR-...	BOOKS_AND_REFERE...	4.2	5775	4.9	500000	Free	0	Everyone
7711	HTC Sense Input - FR	TOOLS	4	885	8	100000	Free	0	Everyone
7712	Fanfic-FR	BOOKS_AND_REFERE...	3.3	52	3.6	5000	Free	0	Teen
7713	Fr. Daoud Lamei	FAMILY	5	22	8.6	1000	Free	0	Teen
7714	Fr Agnel Ambarnath	FAMILY	4.2	117	13	5000	Free	0	Everyone
7715	Manga-FR - Anime Vos...	COMICS	3.4	291	13	10000	Free	0	Everyone
7716	Bulgarian French Dicti...	BOOKS_AND_REFERE...	4.6	603	7.4	10000	Free	0	Everyone
7717	News Minecraft.fr	NEWS_AND_MAGAZIN...	3.8	881	2.3	100000	Free	0	Everyone
7718	FR Tides	WEATHER	3.8	1195	0.6	100000	Free	0	Everyone
7719	Chemin (fr)	BOOKS_AND_REFERE...	4.8	44	0.6	1000	Free	0	Everyone
7720	FR Calculator	FAMILY	4	7	2.6	500	Free	0	Everyone
7721	Sya9a Maroc - FR	FAMILY	4.5	38	53	5000	Free	0	Everyone
7722	Fr. Mike Schmitz Audio...	FAMILY	5	4	3.6	100	Free	0	Everyone
7723	iHoroscope - 2018 Dail...	LIFESTYLE	4.5	398307	19	10000000	Free	0	Everyone

- Below select query represents the list of Apps which belongs to the **Games** category with a rating above **4.5** and the count of all the list is **331**

```
20  select * from GooglePlayStore where category like '%GAME%' and rating >= 4.5
```

Data Output | Explain | Notifications | Messages

	app character varying	category character varying	rating double precision	reviews integer	size_app double precision	installs integer	type_app character varying (20)	price double precision	content_rating character varying
4	Bubble Shooter	GAME	4.5	148897	46	10000000	Free	0	Everyone
5	Clash Royale	GAME	4.6	23133508	97	100000000	Free	0	Everyone 10+
6	Block Puzzle	GAME	4.6	59800	7.8	5000000	Free	0	Everyone
7	Sonic Dash	GAME	4.5	3778921	75	100000000	Free	0	Everyone
8	Clash of Clans	GAME	4.6	44891723	98	100000000	Free	0	Everyone 10+
9	Super Jim Jump - pixel...	GAME	4.5	10393	18	1000000	Free	0	Everyone
10	8 Ball Pool	GAME	4.5	14198297	52	100000000	Free	0	Everyone
11	Bubble Witch 3 Saga	GAME	4.7	1732263	78	50000000	Free	0	Everyone
12	Word Search	GAME	4.7	295241	3.9	10000000	Free	0	Everyone
13	Granny	GAME	4.5	1135631	59	50000000	Free	0	Teen
14	Zombie Catchers	GAME	4.7	990491	75	10000000	Free	0	Everyone
15	Geometry Dash World	GAME	4.6	760628	63	10000000	Free	0	Everyone
16	My Talking Angela	GAME	4.5	9881829	99	100000000	Free	0	Everyone
17	Subway Surfers	GAME	4.5	27723193	76	1000000000	Free	0	Everyone 10+

```
select count(*) from GooglePlayStore where category like '%GAME%' and rating >= 4.5
```

Data Output | Explain | Notifications | Messages

	count bigint
1	331

- Below select query gives the Apps which belong to the Family category along with the Installs (Downloads) above 100000 which are ordered by the app_size

```
select * from GooglePlayStore where category like '%FAMILY%' and installs >=100000 order by size_app
```

Data Output | Explain | Notifications | Messages

	app character varying	category character varying	rating double precision	reviews integer	size_app double precision	installs integer	type_app character varying (20)	price double precision	content_rating character varying
1	U-Disco	FAMILY	3.6	906	0.2	100000	Free	0	Everyone
2	FML F*ck my life + wid...	FAMILY	4.2	1415	0.2	100000	Free	0	Everyone
3	EXO-L	FAMILY	4.6	67410	0.8	1000000	Free	0	Everyone
4	Denis Brogniart - AH !	FAMILY	4.5	2931	0.8	100000	Free	0	Everyone
5	Citation du Jour - Moti...	FAMILY	4.1	321	0.9	100000	Free	0	Everyone
6	German Vocabulary Tr...	FAMILY	3.3	1218	1	100000	Free	0	Everyone
7	FINAL FANTASY DIME...	FAMILY	4.3	8450	1.1	100000	Paid	13.99	Everyone
8	ChatBolo - AI Chatbot ...	FAMILY	3.3	1407	1.4	100000	Free	0	Everyone
9	Crazy Colors: Bubbles ...	FAMILY	4.6	3063	1.6	100000	Free	0	Everyone
10	TAMAGO	FAMILY	3	42515	1.6	5000000	Free	0	Everyone
11	AC REMOTE UNIVERS...	FAMILY	1.6	402	1.7	100000	Free	0	Everyone
12	DW Learn German - A1,...	FAMILY	4.3	902	1.8	100000	Free	0	Everyone
13	C Programming	FAMILY	4.3	22248	1.8	1000000	Free	0	Everyone
14	Smart-AC Universal Re...	FAMILY	1.8	3270	1.8	500000	Free	0	Everyone
15	Korean English Transla...	FAMILY	4.4	634	1.8	100000	Free	0	Everyone

- Below select query gives the Apps details which come under the medical category where the installs between 100000 and 5000000 with the decreasing order of price greater than 0.

```
select * from GooglePlayStore where category like 'MEDICAL%' and installs between 100000 and 5000000 and
price>0  order by price desc
```

Data Output | Explain | Notifications | Messages

	app character varying	category character varying	rating double precision	reviews integer	size_app double precision	installs integer	type_app character varying (20)	price double precision	content_rating character varying
1	Human Anatomy Atlas 2018...	MEDICAL	4.5	2921	25	100000	Paid	24.99	Everyone
2	Human Anatomy Atlas 2018...	MEDICAL	4.5	2921	25	100000	Paid	24.99	Everyone
3	Human Anatomy Atlas 2018...	MEDICAL	4.5	2923	25	100000	Paid	24.99	Everyone
4	Monash Uni Low FODMAP ...	MEDICAL	4.2	1135	12	100000	Paid	9	Everyone
5	Monash Uni Low FODMAP ...	MEDICAL	4.2	1135	12	100000	Paid	9	Everyone

- Below query gives the App for which the price is the maximum which 400 dollars for the **I'm Rich- Trump Edition.** Hence the most expensive Android App from the given data is I'm Rich-Trump Edition

```
select * from GooglePlayStore order by price desc
```

Data Output | Explain | Notifications | Messages

	app character varying	category character varying	rating double pre	reviews integer	size_app double precision	installs integer	type_app character varying (	price double precision	content_rating character varying
1	I'm Rich - Trump Edition	LIFESTYLE	3.6	275	7.3	10000	Paid	400	Everyone
2	I am rich(premium)	FINANCE	3.5	472	0.9	5000	Paid	399.99	Everyone
3	I Am Rich Pro	FAMILY	4.4	201	2.7	5000	Paid	399.99	Everyone
4	I am Rich Plus	FAMILY	4	856	8.7	10000	Paid	399.99	Everyone
5	I AM RICH PRO PLUS	FINANCE	4	36	41	1000	Paid	399.99	Everyone
6	I am Rich	FINANCE	4.3	180	3.8	5000	Paid	399.99	Everyone
7	I Am Rich Premium	FINANCE	4.1	1867	4.7	50000	Paid	399.99	Everyone
8	💎 I'm rich	LIFESTYLE	3.8	718	26	10000	Paid	399.99	Everyone
9	I am rich	LIFESTYLE	3.8	3547	1.8	100000	Paid	399.99	Everyone
10	I am rich (Most expensive app)	FINANCE	4.1	129	2.7	1000	Paid	399.99	Teen
11	I am Rich!	FINANCE	3.8	93	22	1000	Paid	399.99	Everyone

Comparison

Comparing the Android Apps with Apple Apps is possible, but the results would be inappropriate because 80% of the population use Android phones and 20% are IOS users. Also the Apple Apps are mostly paid and restricted to its users hence, the number of downloads and the rating of the all will also vary.

Conclusion

- Hence from the analysis we can conclude that the Most expensive App is the **I'm Rich- Trump Edition.**
- The Category is dependent on Price and Installs which we could find from the Hypothesis Testing.
- From the Regression Analysis, we could understand that the Content Rating (Age) affects the Rating for the particular App
- More number of Reviews will lead to more Installations of an App which is an understanding from the Correlation analysis

References

Data Set is taken from the below

(Gupta, 2019)