EDA

April 8, 2019

```
In [1]: import pandas as pd
        import numpy as np
        from matplotlib import pyplot as plt
        import seaborn as sns
        plt.style.use('seaborn')
        sns.set(font_scale=1.8)
```

```
In [2]: data=pd.read_csv("C:/Users/lxy/Desktop/Data Mining/semester project/AppleStore.csv")
        data.head()
```

```
Out[2]:    Unnamed: 0         id                                          track_name  \
        0           1  281656475                                     PAC-MAN Premium
        1           2  281796108                           Evernote - stay organized
        2           3  281940292     WeatherBug - Local Weather, Radar, Maps, Alerts
        3           4  282614216   eBay: Best App to Buy, Sell, Save! Online Shop...
        4           5  282935706                                               Bible

           size_bytes currency  price  rating_count_tot  rating_count_ver  \
        0   100788224      USD   3.99             21292                26
        1   158578688      USD   0.00            161065                26
        2   100524032      USD   0.00            188583              2822
        3   128512000      USD   0.00            262241               649
        4    92774400      USD   0.00            985920              5320

           user_rating  user_rating_ver     ver cont_rating   prime_genre  \
        0          4.0              4.5   6.3.5          4+         Games
        1          4.0              3.5   8.2.2          4+  Productivity
        2          3.5              4.5   5.0.0          4+       Weather
        3          4.0              4.5  5.10.0         12+      Shopping
        4          4.5              5.0   7.5.1          4+     Reference

           sup_devices.num  ipadSc_urls.num  lang.num  vpp_lic
        0               38                5        10        1
        1               37                5        23        1
        2               37                5         3        1
        3               37                5         9        1
        4               37                5        45        1
```

"id" : App ID
"track_name": App Name
"size_bytes": Size (in Bytes)
"currency": Currency Type
"price": Price amount
"rating_count_tot": User Rating counts (for all version)
"rating_count_ver": User Rating counts (for current version)
"user_rating" : Average User Rating value (for all version)
"user_rating_ver": Average User Rating value (for current version)
"ver" : Latest version code
"cont_rating": Content Rating
"prime_genre": Primary Genre
"sup_devices.num": Number of supporting devices
"ipadSc_urls.num": Number of screenshots showed for display
"lang.num": Number of supported languages
"vpp_lic": Vpp Device Based Licensing Enabled

```
In [3]: data.isnull().sum()
```

```
Out[3]: Unnamed: 0          0
        id                  0
        track_name          0
        size_bytes          0
        currency            0
        price               0
        rating_count_tot    0
        rating_count_ver    0
        user_rating         0
        user_rating_ver     0
        ver                 0
        cont_rating         0
        prime_genre         0
        sup_devices.num     0
        ipadSc_urls.num     0
        lang.num            0
        vpp_lic             0
        dtype: int64
```

There is no missing value in the data set.

```
In [4]: description=pd.read_csv("C:/Users/lxy/Desktop/Data Mining/semester project/appleStore_c
        description.head()
```

```
Out[4]:           id                                        track_name  size_bytes  \
        0  281656475                                   PAC-MAN Premium   100788224
        1  281796108                         Evernote - stay organized   158578688
        2  281940292    WeatherBug - Local Weather, Radar, Maps, Alerts   100524032
        3  282614216  eBay: Best App to Buy, Sell, Save! Online Shop...   128512000
        4  282935706                                             Bible    92774400
```

```
                                            app_desc
0  SAVE 20%, now only $3.99 for a limited time!\n...
1  Let Evernote change the way you organize your ...
2  Download the most popular free weather app pow...
3  The eBay app is the best way to find anything ...
4  On more than 250 million devices around the wo...
```

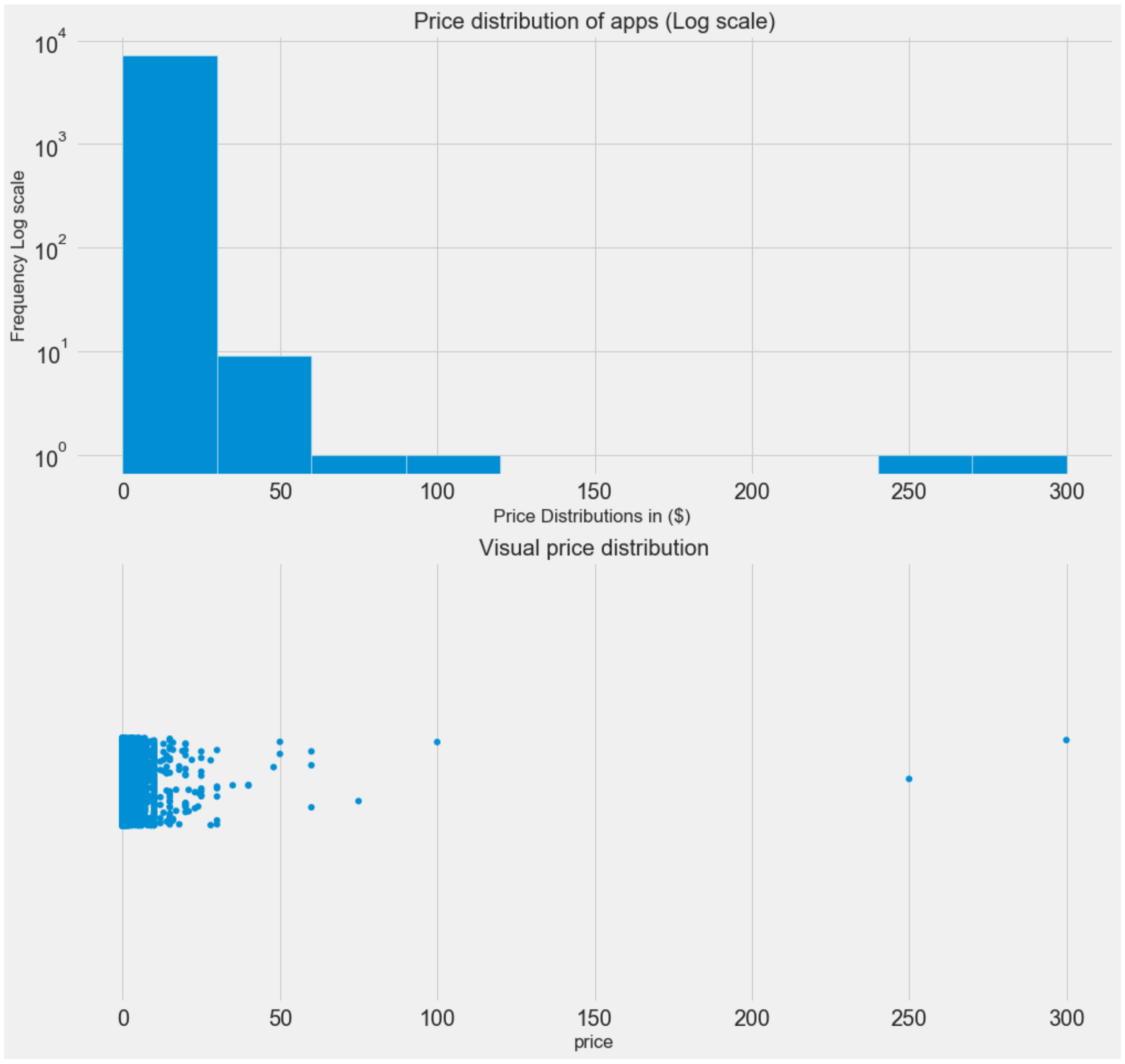
```
In [5]: description.isnull().sum()
```

```
Out[5]: id            0
        track_name    0
        size_bytes    0
        app_desc      0
        dtype: int64
```

1 Price Effect

Let's first explore the distribution of the price of the APP.

```
In [6]: plt.style.use('fivethirtyeight')
        plt.figure(figsize=(15,15))
        plt.subplot(2,1,1)

        plt.hist(data.price,log=True)
        plt.title('Price distribution of apps (Log scale)')
        plt.ylabel("Frequency Log scale")
        plt.xlabel("Price Distributions in ($) ")

        plt.subplot(2,1,2)
        plt.title('Visual price distribution')
        sns.stripplot(data=data,y='price',jitter= True,orient = 'h' ,size=6)
        plt.show()
```

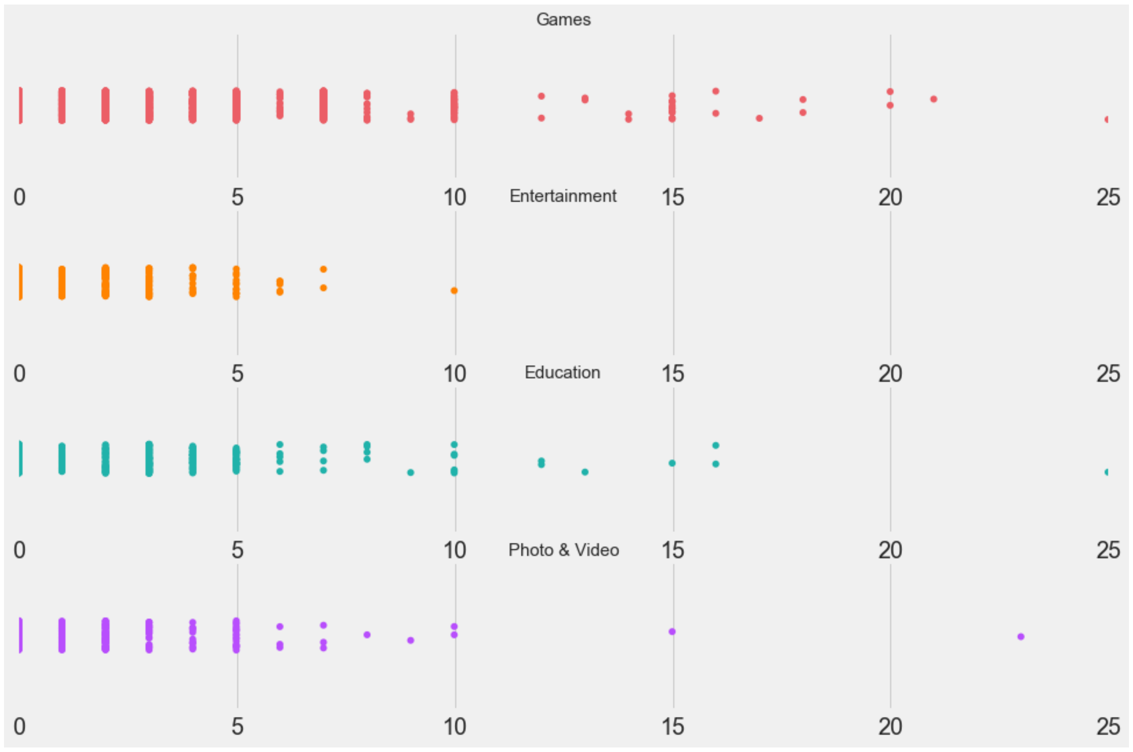


```
In [7]: print ('Number of free apps:' + str(sum(data.price == 0)))
        print ('Counting (outliers) super expensive apps:' + str(sum(data.price > 100)))
        print ('which is around ' + str(sum(data.price > 100)/len(data.price)*100) +
               " % of the total Apps")
        print ('Thus we will dropping the following apps')
        outlier=data[data.price>100][['track_name','price','prime_genre','user_rating']]
        freeapps = data[data.price==0]
        outlier
```

```
Number of free apps:4056
Counting (outliers) super expensive apps 2
which is around 0.027789356676392943 % of the total Apps
 Thus we will dropping the following apps
```

```
Out[7]:                            track_name   price prime_genre  user_rating
        115   Proloquo2Go - Symbol-based AAC  249.99   Education          4.0
        1479             LAMP Words For Life  299.99   Education          4.0
```

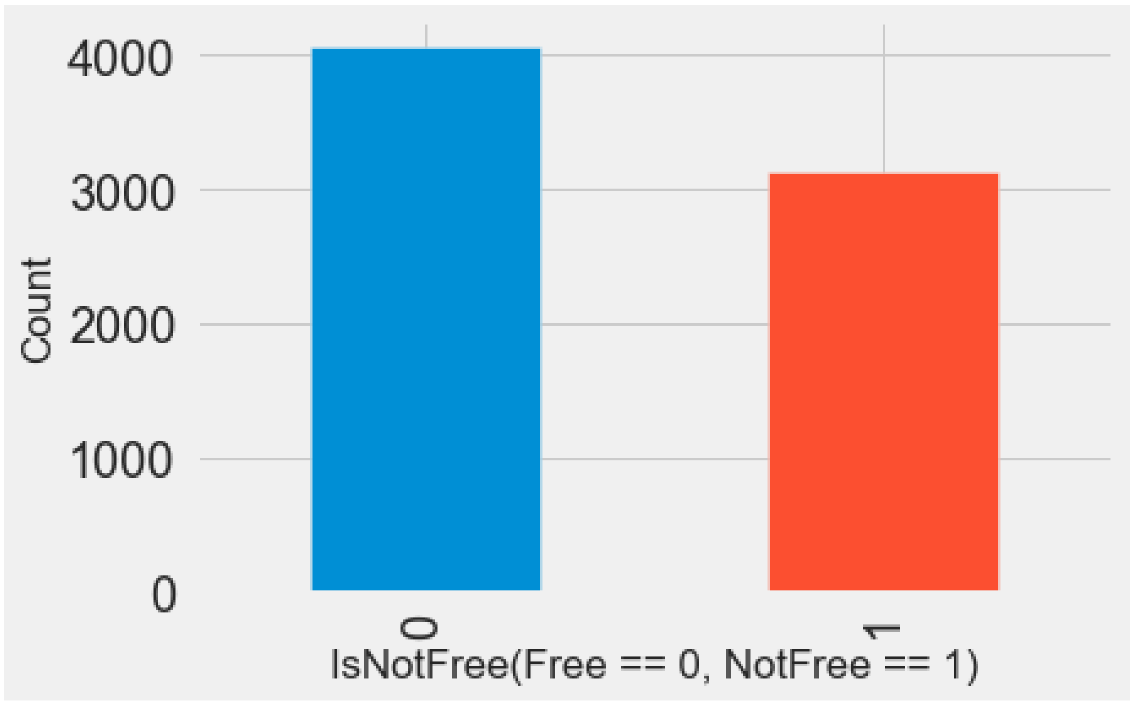
```
In [8]: yrange = [0,25]
        fsize =15

        plt.figure(figsize=(15,10))

        plt.subplot(4,1,1)
        plt.xlim(yrange)
        games = data[data.prime_genre=='Games']
        sns.stripplot(data=games,y='price',jitter= True , orient ='h',size=6,color='#eb5e66')
        plt.title('Games',fontsize=fsize)
        plt.xlabel('')

        plt.subplot(4,1,2)
        plt.xlim(yrange)
        ent = data[data.prime_genre=='Entertainment']
        sns.stripplot(data=ent,y='price',jitter= True ,orient ='h',size=6,color='#ff8300')
        plt.title('Entertainment',fontsize=fsize)
        plt.xlabel('')

        plt.subplot(4,1,3)
        plt.xlim(yrange)
        edu = data[data.prime_genre=='Education']
        sns.stripplot(data=edu,y='price',jitter= True ,orient ='h' ,size=6,color='#20B2AA')
        plt.title('Education',fontsize=fsize)
        plt.xlabel('')

        plt.subplot(4,1,4)
        plt.xlim(yrange)
        pv = data[data.prime_genre=='Photo & Video']
        sns.stripplot(data=pv,y='price',jitter= True  ,orient ='h',size=6,color='#b84efd')
        plt.title('Photo & Video',fontsize=fsize)
        plt.xlabel('')

        plt.show()
```



From the plot, we can conclude that the price are effected by genre of the APP, and people are more willing to spend money on App about games and education.

1.1 Free V.S. Non-free

```
In [9]: data['isNotFree'] = data['price'].apply(lambda x: 1 if x > 0 else 0)
        data['isNotFree'].value_counts().plot.bar()
        plt.xlabel('IsNotFree(Free == 0, NotFree == 1)')
        plt.ylabel('Count')
        plt.show()
```



There are many free Apps, therefore we can analyze the data depending on whether the APP is free or not free. Let's make the two dataframes for free and not-free.

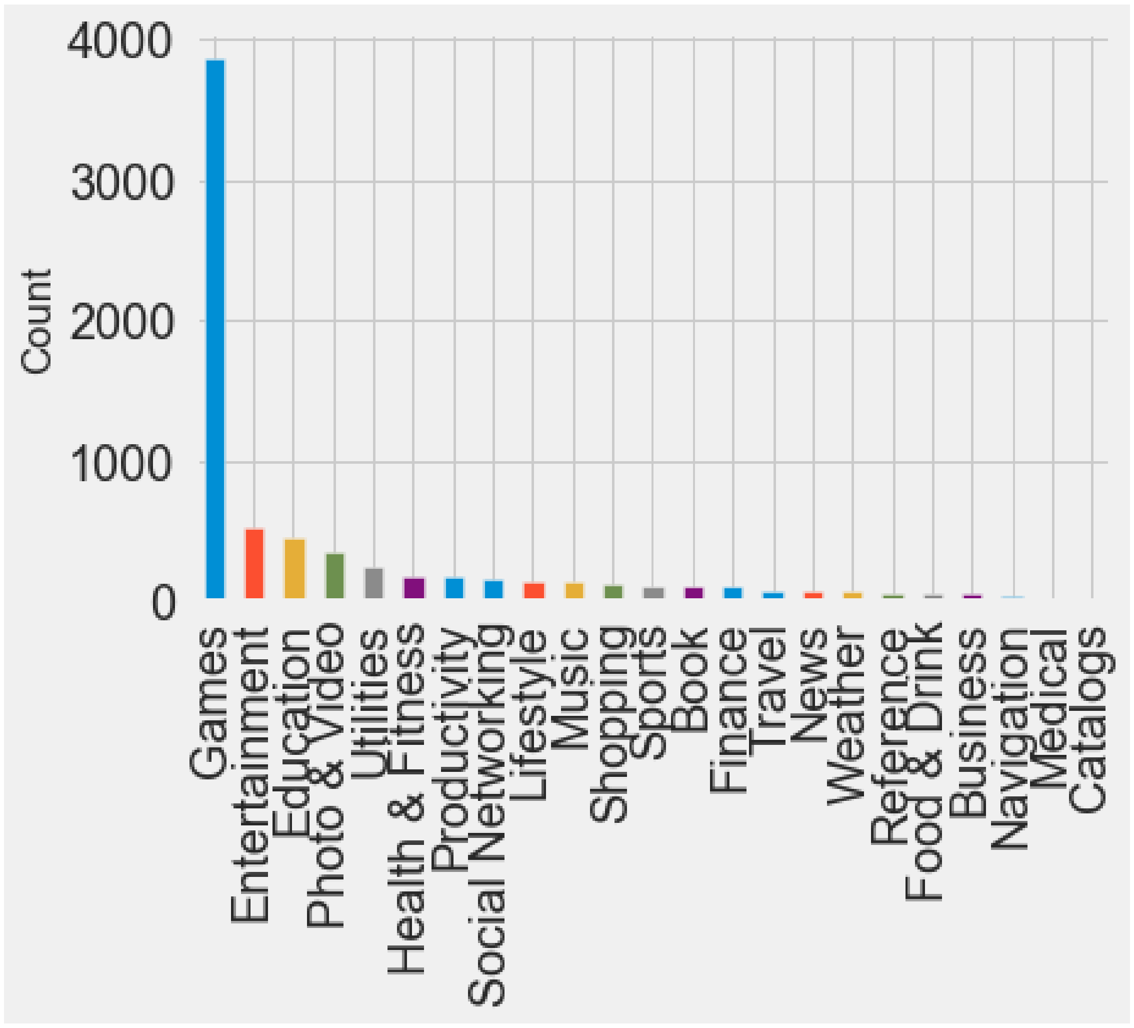
```
In [10]: data_notfree = data[data['isNotFree'] == 1]
         data_free = data[data['isNotFree'] == 0]

         print('There are {} Not-Free Apps in this dataset'.format(data_notfree.shape[0]))
         print('There are {} Free Apps in this dataset'.format(data_free.shape[0]))
```

```
There are 3141 Not-Free Apps in this dataset
There are 4056 Free Apps in this dataset
```

2 Genre

We are going to plot the count for each type of genre.

```
In [11]: data['prime_genre'].value_counts().plot.bar()
         plt.ylabel('Count')
         plt.show()
```

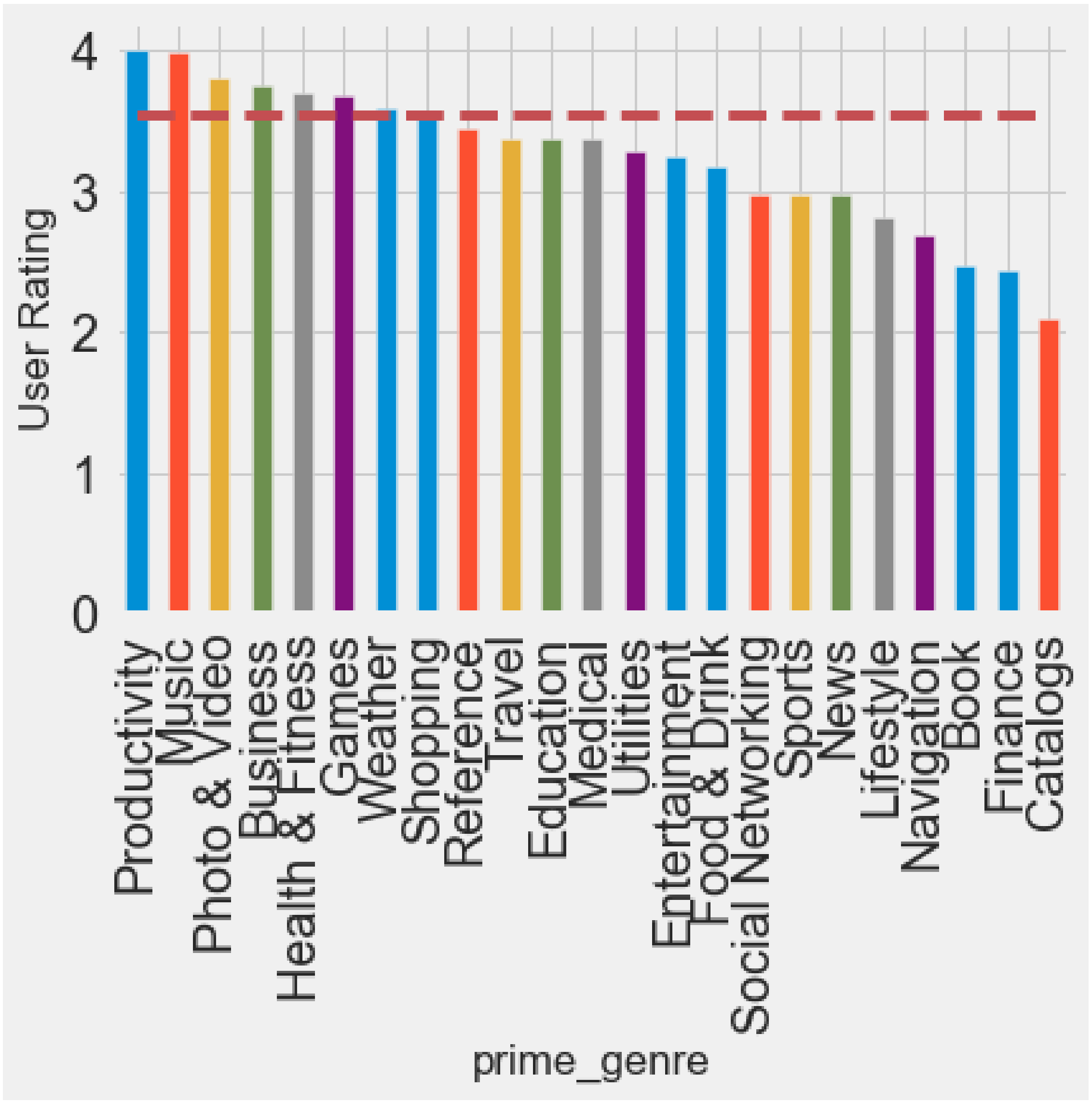


Then we are going to explore the user rating depending on genre.

```
In [12]: rate_genre=data[['prime_genre', 'user_rating']].groupby('prime_genre').mean()['user_ra
         plt.ylabel('User Rating')
         plt.hlines(data['user_rating'].mean(),0,22,colors = "r", linestyles = "dashed")
         data[['prime_genre', 'user_rating']].groupby('prime_genre').mean()['user_rating']
```

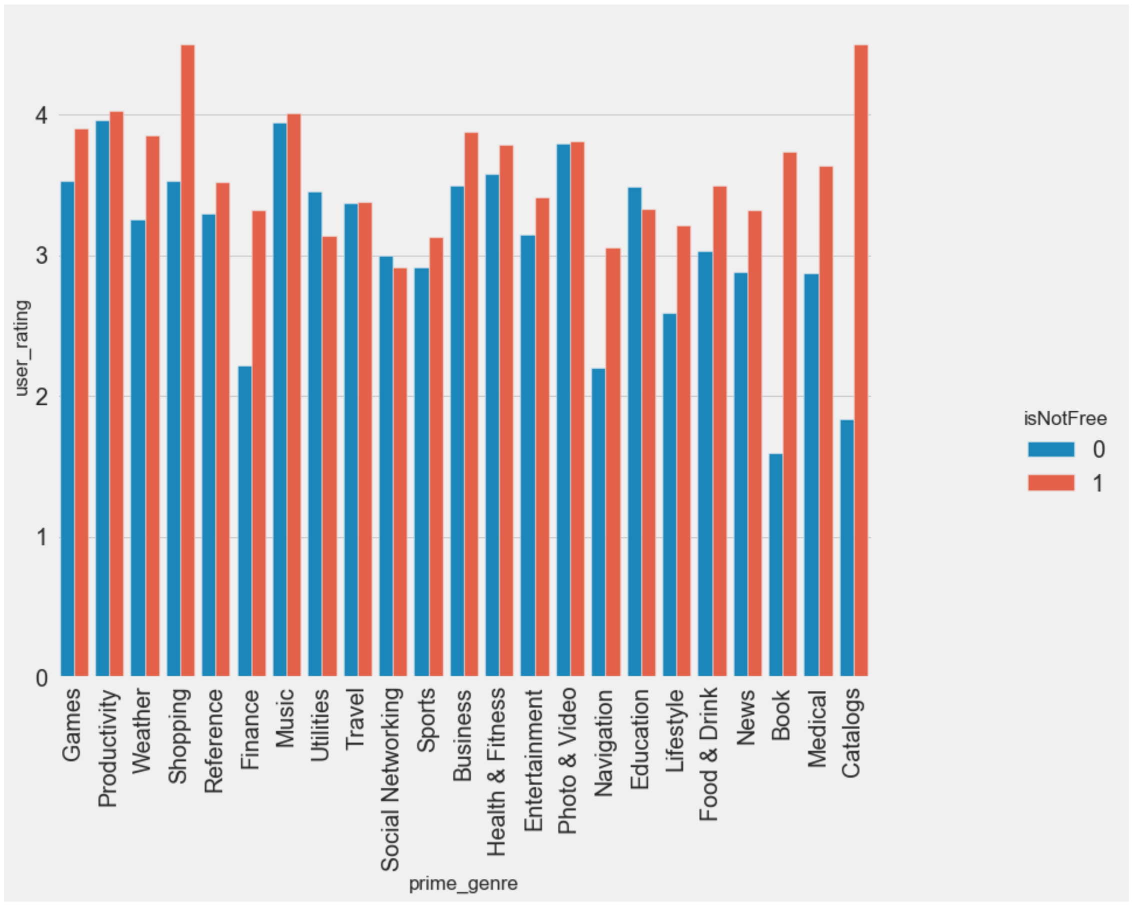
```
Out[12]: prime_genre
         Book                 2.477679
         Business             3.745614
         Catalogs             2.100000
         Education            3.376380
         Entertainment        3.246729
         Finance              2.432692
         Food & Drink         3.182540
         Games                3.685008
         Health & Fitness     3.700000
```

```
Lifestyle            2.805556
Medical              3.369565
Music                3.978261
Navigation           2.684783
News                 2.980000
Photo & Video        3.800860
Productivity         4.005618
Reference            3.453125
Shopping             3.540984
Social Networking    2.985030
Sports               2.982456
Travel               3.376543
Utilities            3.278226
Weather              3.597222
Name: user_rating, dtype: float64
```



The Apps for Productivity, Music, Photo & Vidio, Bussiness, Health & Fitness and Games have higher mean user rating. The mean user rating of Book, Finance and Catalogs are less than 2.5. Then we can contrast the rating of free Apps and Non-free Apps for each genre.

```
In [13]: g=sns.catplot(x="prime_genre", y="user_rating", hue="isNotFree", kind="bar", data=data
         g.fig.set_size_inches(15,10)
         g.set_xticklabels(rotation=90)

Out[13]: <seaborn.axisgrid.FacetGrid at 0x146c1bc85f8>
```

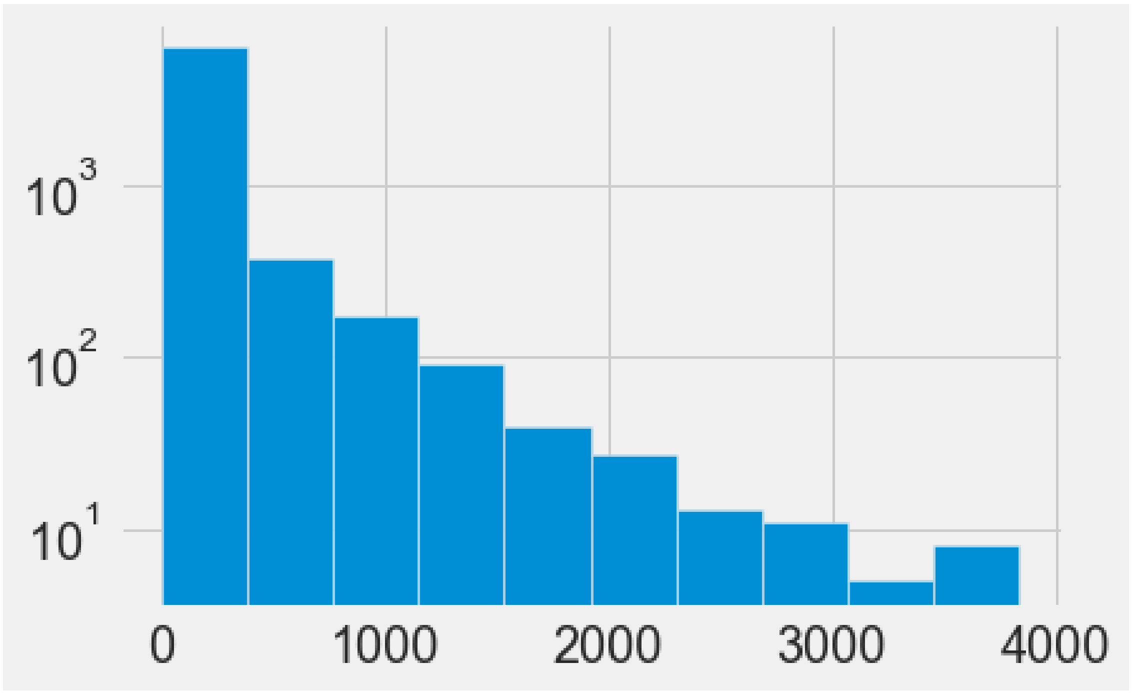


From the plot we can see that Book, Catalogs and Navigation Apps have much higher ratings when they are not free, and non-free Apps will have higher user rating compared to the free one in the same genre.

3 Size

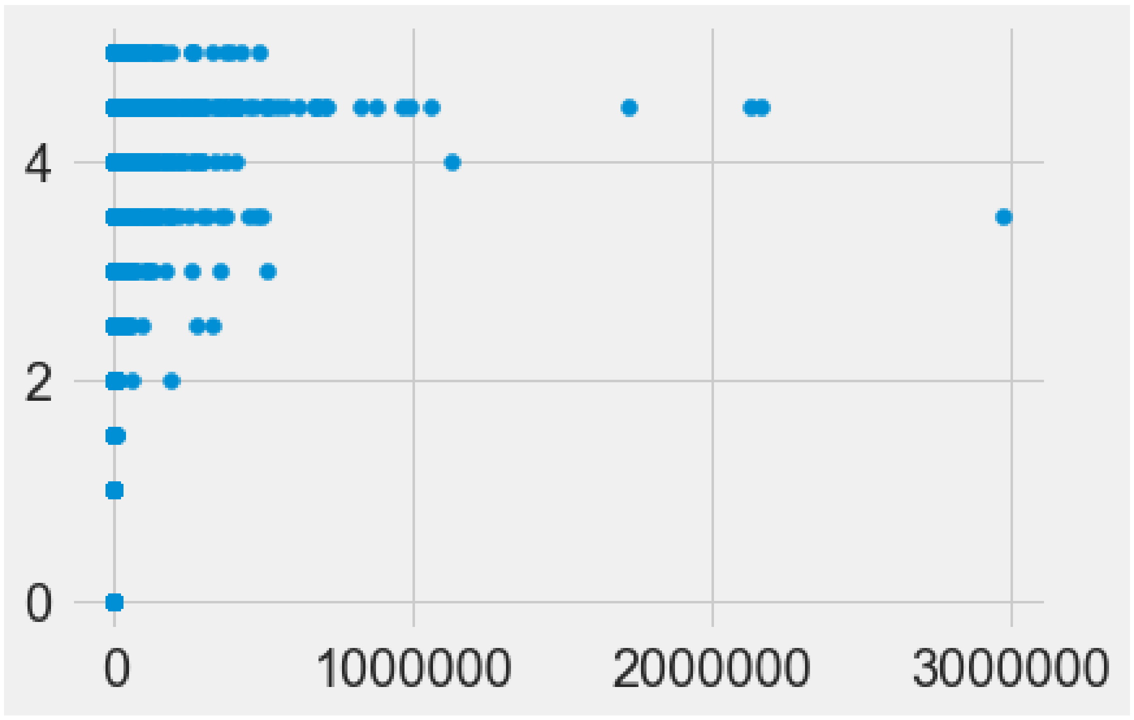
For convenience, change the unit of size_bytes into Megabytes.

```
In [14]: data['size_bytes_MB'] = data['size_bytes'] / (1024 * 1024.0)
         plt.hist(data['size_bytes_MB'],log=True)

Out[14]: (array([6.451e+03, 3.770e+02, 1.740e+02, 9.100e+01, 4.000e+01, 2.700e+01,
                 1.300e+01, 1.100e+01, 5.000e+00, 8.000e+00]),
          array([5.62500000e-01, 3.84452637e+02, 7.68342773e+02, 1.15223291e+03,
                 1.53612305e+03, 1.92001318e+03, 2.30390332e+03, 2.68779346e+03,
                 3.07168359e+03, 3.45557373e+03, 3.83946387e+03]),
          <a list of 10 Patch objects>)
```



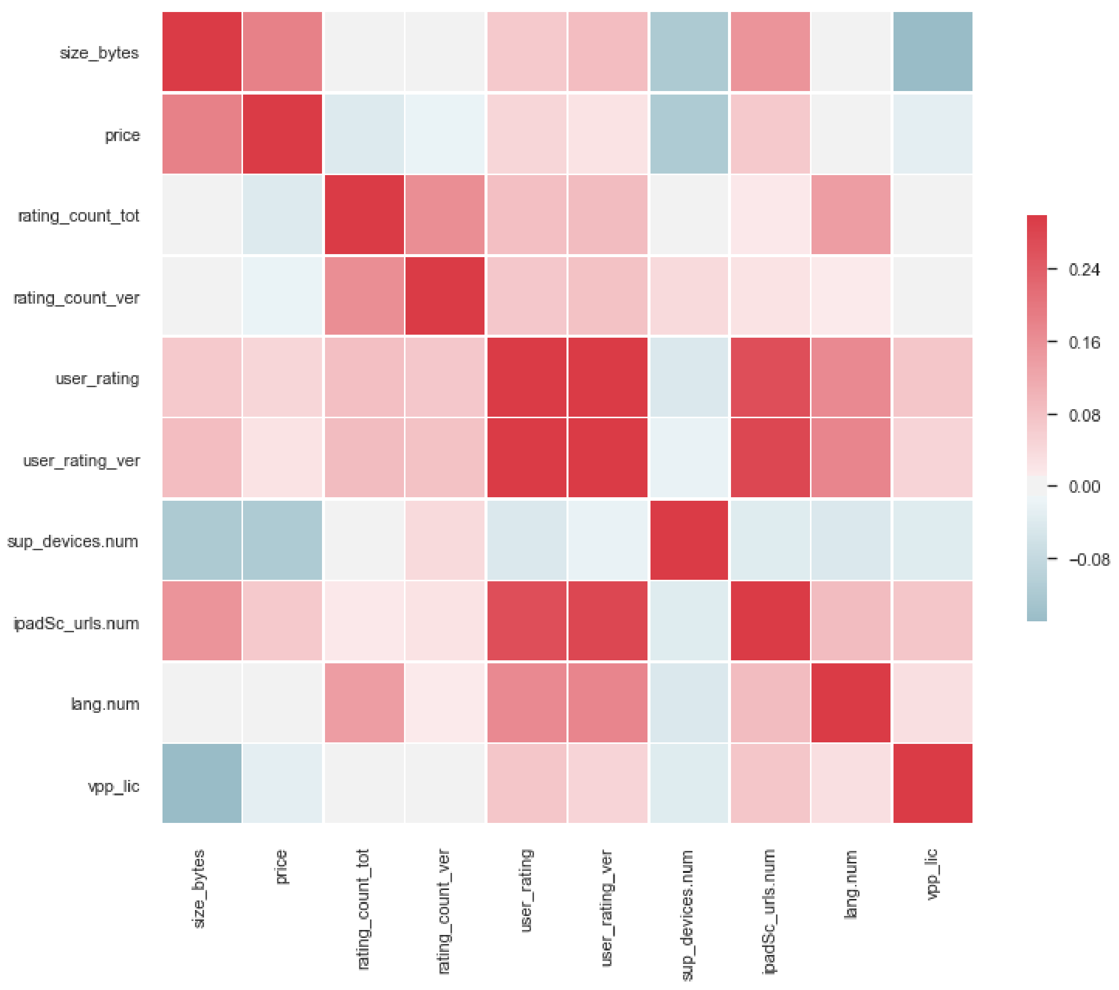
4 Total Number of Rating

```
In [15]: plt.scatter(data['rating_count_tot'],data['user_rating'])

Out[15]: <matplotlib.collections.PathCollection at 0x146c1564ef0>
```



5 Correlation

```
In [36]: sns.set(style="white")

         # Compute the correlation matrix
         corr = corrdata.corr()

         # Set up the matplotlib figure
         f, ax = plt.subplots(figsize=(11, 9))

         # Generate a custom diverging colormap
         cmap = sns.diverging_palette(220, 10, as_cmap=True)

         # Draw the heatmap with the mask and correct aspect ratio
         sns.heatmap(corr, cmap=cmap, vmax=.3, center=0,
                     square=True, linewidths=.5, cbar_kws={"shrink": .5})

Out[36]: <matplotlib.axes._subplots.AxesSubplot at 0x146c1abfcc0>
```



```
In [37]: corr['user_rating'].sort_values(ascending=False)

Out[37]: user_rating          1.000000
         user_rating_ver      0.774140
         ipadSc_urls.num      0.265671
         lang.num             0.170976
         rating_count_tot     0.083310
         vpp_lic              0.069816
         rating_count_ver     0.068754
         size_bytes           0.066256
         price                0.046601
         sup_devices.num     -0.042451
         Name: user_rating, dtype: float64
```