Game_Analysis

March 15, 2018

Observed Trends: 1. Males not only the make up over 80% of the players of this game, the are also responsible for over 80% of the revenue. 2. Out of the 573 players, each player has spend under 20 dollars on items. 3. The 20-24 age bracket more revenue than any other bracket, but the 25-29 age group, on average, purchases more expensive items. 4. The Retribution Axe not only appears in the most popular item list, it is priced almost 2 dollars more than other popular items on that list and is the item that has generated the most revenue. 5. Other than the Retribution Axe, the most popular items list generates items that are priced below the average purcahse price.

NOTE: 1. The age bins are divided in 10 instead of 4, so that data trends can be analyzed more accurately. 2. To show the normalized total, i have used 0-1 scale (feature scaling method) because when it comes to real time data visualization and to visualize the data where the difference of the value could be 1000 to 20000 then 0-1 scale would be more scalable than the regular normalization.Also prepared supporting documentation from google search.

```
In [1]: import pandas as pd
        import numpy as np
        import csv
```

```
In [2]: # Reading input file to dataframe
        filepath = ("purchase_data.json")
        df = pd.read_json(filepath)
        #df.to_csv("mainjson output.csv", sep =',', encoding='utf-8') - added this step to ana
        #df.head()
        game_df = pd.DataFrame(df)
        game_df.head()
```

```
Out[2]:    Age Gender  Item ID                                 Item Name  Price  \
        0   38   Male      165                 Bone Crushing Silver Skewer   3.37
        1   21   Male      119  Stormbringer, Dark Blade of Ending Misery   2.32
        2   34   Male      174                             Primitive Blade   2.46
        3   21   Male       92                                Final Critic   1.36
        4   23   Male       63                              Stormfury Mace   1.27

                     SN
        0     Aelalis34
        1        Eolo46
        2   Assastnya25
        3  Pheusrical25
        4        Aela59
```

```
In [3]: #Total Players Count
        players_count = len(game_df["SN"].unique())
        total_players_pd = pd.DataFrame({"Total Players Count": [players_count]})
        total_players_pd
```

```
Out[3]:    Total Players Count
        0                  573
```

```
In [4]: #Purchasing Analysis
        purchases_unique = len(game_df["Item ID"].unique())
        purchases_total = game_df["Price"].count()
        purchases_revenue = game_df["Price"].sum()
        puchases_average = round(game_df["Price"].sum()/purchases_total,2)

        purc_analysis_pd = pd.DataFrame({"Number of Unique Items":purchases_unique,"Average Pu
                                 "Total Number of Purchases":purchases_total, "Total Revenue":
                                    columns =["Number of Unique Items","Average Purchase Price"

        purc_analysis_pd["Average Purchase Price"] = purc_analysis_pd["Average Purchase Price"]
        purc_analysis_pd["Total Revenue"] = purc_analysis_pd["Total Revenue"].map("${:.2f}".fo
        purc_analysis_pd
```

```
Out[4]:    Number of Unique Items Average Purchase Price  Total Number of Purchases  \
        0                     183                  $2.93                        780

          Total Revenue
        0      $2286.33
```

```
In [5]: #Gender Demographics
        gender_df = game_df.drop_duplicates(subset=['SN'],keep='first')
        gender_df = gender_df.groupby("Gender")
        gender_count = gender_df["Age"].count()
        gender_percentage = round((gender_count/players_count)*100,2)
        gender_analysis = pd.DataFrame({"Players count":gender_count,"Percentage of Players":ge
        gender_analysis.sort_values(["Players count"],ascending = False)
        gender_analysis["Percentage of Players"] = gender_analysis["Percentage of Players"].map
        gender_analysis
```

```
Out[5]:                       Percentage of Players  Players count
        Gender
        Female                               17.45%            100
        Male                                 81.15%            465
        Other / Non-Disclosed                 1.40%              8
```

```
In [6]: #Purchasing Analysis - Gender based
        purc_gender_df = game_df.groupby("Gender")
        gender_purchases_count = purc_gender_df["Price"].count()
        gender_purchases_average = round(purc_gender_df["Price"].mean(),2)
        gender_purchases_total = purc_gender_df["Price"].sum()
        gender_purchases_max = gender_purchases_total.max()
        gender_purchases_min = gender_purchases_total.min()
        gender_normalize_total = (gender_purchases_total - gender_purchases_min)/(gender_purcha
        gender_purchase_analysis = pd.DataFrame({"Total Gender Purchases":gender_purchases_cou
                                               "Total Purchase Value": gender_purchases_total,
                                                columns =["Total Gender Purchases","Average Ge
        gender_purchase_analysis
```

```
Out[6]:                        Total Gender Purchases  Average Gender Purchases  \
        Gender
        Female                                    136                      2.82
        Male                                      633                      2.95
        Other / Non-Disclosed                      11                      3.25

                               Total Purchase Value  Normalized Totals
        Gender
        Female                               382.91           0.189509
        Male                                1867.68           1.000000
        Other / Non-Disclosed                 35.74           0.000000
```

```
In [7]: #Age Analysis
        bins = [0, 9, 14, 19, 24, 29, 34, 39, 100]
        range_names = ['< 10', '10 - 14', '15 -19', '20 - 24', '25 - 29', '30 - 34', '35 - 39'
        game_new_df = game_df
        game_new_df["Age Range"] = pd.cut(game_new_df["Age"], bins, labels=range_names)
        age_df = game_new_df.drop_duplicates(subset=['SN'],keep='first')
        age_analysis_df = age_df.groupby("Age Range")
        age_players_count = age_analysis_df["Age"].count()
        age_percentage_players = round((age_analysis_df["Age"].count()/players_count)*100,2)
        age_analysis_pd = pd.DataFrame({"Players Count":age_players_count,
                                     "Percentage of Players": age_percentage_players})
        age_analysis_pd["Percentage of Players"] = age_analysis_pd["Percentage of Players"].ma
        age_analysis_pd
```

```
Out[7]:           Percentage of Players  Players Count
        Age Range
        < 10                      3.32%             19
        10 - 14                   4.01%             23
        15 -19                   17.45%            100
        20 - 24                  45.20%            259
        25 - 29                  15.18%             87
        30 - 34                   8.20%             47
        35 - 39                   4.71%             27
        40+                       1.92%             11
```

```
In [8]: #Age Purchase Analysis
        age_purc_df = game_new_df.groupby("Age Range")
        age_purchases_count = age_purc_df["Price"].count()
        age_purchases_average = round(age_purc_df["Price"].mean(),2)
```

```
age_purchases_total = age_purc_df["Price"].sum()
age_purchase_max = age_purchases_total.max()
age_purchase_min = age_purchases_total.min()
age_normalize_total = (age_purchases_total-age_purchase_min)/(age_purchase_max - age_pu
age_purchase_analysis_pd = pd.DataFrame({"Purchase Count":age_purchases_count,"Average
                                "Total Purchase Value": age_purchases_total, "Normalize
                                columns =["Purchase Count","Average Purchase Price","To

age_purchase_analysis_pd["Average Purchase Price"] = age_purchase_analysis_pd["Average
age_purchase_analysis_pd["Total Purchase Value"] = age_purchase_analysis_pd["Total Purc
age_purchase_analysis_pd
```

Out[8]:

```
           Purchase Count Average Purchase Price Total Purchase Value  \
Age Range
< 10                   28                  $2.98               $83.46
10 - 14                35                  $2.77               $96.95
15 -19                133                  $2.91              $386.42
20 - 24               336                  $2.91              $978.77
25 - 29               125                  $2.96              $370.33
30 - 34                64                  $3.08              $197.25
35 - 39                42                  $2.84              $119.40
40+                    17                  $3.16               $53.75

           Normalize Total
Age Range
< 10              0.032118
10 - 14           0.046702
15 -19            0.359635
20 - 24           1.000000
25 - 29           0.342241
30 - 34           0.155132
35 - 39           0.070971
40+               0.000000
```

In [9]:

```
#Top 5 Spenders
spenders_df = game_df.groupby("SN")
spenders_count = spenders_df["Price"].count()
spenders_average_price = spenders_df["Price"].mean()
spenders_total = spenders_df["Price"].sum()

spender_analysis_pd = pd.DataFrame({"Purchase Count":spenders_count,"Average Purchase P
                                "Total Purchase Value": spenders_total},
                                columns =["Purchase Count","Average Purchase Price","To

total_spender_analysis_pd = spender_analysis_pd.sort_values("Total Purchase Value", asc
top_spender_analysis_pd = total_spender_analysis_pd.head(5)
top_spender_analysis_pd["Average Purchase Price"] = top_spender_analysis_pd["Average Pu
top_spender_analysis_pd["Total Purchase Value"] = top_spender_analysis_pd["Total Purcha
```

```
        top_spender_analysis_pd
```

```
C:\Users\santo\Anaconda3\lib\site-packages\ipykernel_launcher.py:13: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html
  del sys.path[0]
C:\Users\santo\Anaconda3\lib\site-packages\ipykernel_launcher.py:14: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html
```

```
Out[9]:              Purchase Count Average Purchase Price Total Purchase Value
        SN
        Undirrala66               5                  $3.41               $17.06
        Saedue76                  4                  $3.39               $13.56
        Mindimnya67               4                  $3.18               $12.74
        Haellysu29                3                  $4.24               $12.73
        Eoda93                    3                  $3.86               $11.58
```

```
In [10]: #Most Popular Items
         item_df = game_df.groupby(["Item ID","Item Name"])
         item_count = item_df["Price"].count()
         item_total = item_df["Price"].sum()
         item_price = item_df["Price"].unique()
         item_analysis_pd = pd.DataFrame({"Purchase Count":item_count,"Purchase Price": item_p
         item_price_pd = item_analysis_pd["Purchase Price"].values.astype(float)
         item_analysis_pd["Price"] = item_price_pd
         new_item_analysis_pd = item_analysis_pd[["Purchase Count","Price","Total Purchase Valu
         total_item_analysis_pd = new_item_analysis_pd.sort_values("Purchase Count",ascending =
         top_item_analysis_pd = total_item_analysis_pd.head(6)

         top_item_analysis_pd["Price"] = top_item_analysis_pd["Price"].map("${:.2f}".format)
         top_item_analysis_pd["Total Purchase Value"] = top_item_analysis_pd["Total Purchase Va

         top_item_analysis_pd
```

```
C:\Users\santo\Anaconda3\lib\site-packages\ipykernel_launcher.py:13: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html
  del sys.path[0]
C:\Users\santo\Anaconda3\lib\site-packages\ipykernel_launcher.py:14: SettingWithCopyWarning:
```

```
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html
```

```
Out[10]:                                                Purchase Count  Price  \
         Item ID Item Name
         39      Betrayal, Whisper of Grieving Widows               11  $2.35
         84      Arcane Gem                                         11  $2.23
         31      Trickster                                           9  $2.07
         175     Woeful Adamantite Claymore                          9  $1.24
         13      Serenity                                            9  $1.49
         34      Retribution Axe                                     9  $4.14

                                                       Total Purchase Value
         Item ID Item Name
         39      Betrayal, Whisper of Grieving Widows                $25.85
         84      Arcane Gem                                          $24.53
         31      Trickster                                           $18.63
         175     Woeful Adamantite Claymore                          $11.16
         13      Serenity                                            $13.41
         34      Retribution Axe                                     $37.26
```

```
In [11]: #Most Profitable Items
         total_item_analysis_purc_pd = total_item_analysis_pd.sort_values("Total Purchase Valu
         top_item_analysis_purc_pd = total_item_analysis_purc_pd.head(5)
         top_item_analysis_purc_pd["Price"] = top_item_analysis_purc_pd["Price"].map("${:.2f}"
         top_item_analysis_purc_pd["Total Purchase Value"] = top_item_analysis_purc_pd["Total P
         top_item_analysis_purc_pd
```

```
C:\Users\santo\Anaconda3\lib\site-packages\ipykernel_launcher.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html
  after removing the cwd from sys.path.
C:\Users\santo\Anaconda3\lib\site-packages\ipykernel_launcher.py:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html
  """
```

```
Out[11]:                                    Purchase Count  Price Total Purchase Value
         Item ID Item Name
         34      Retribution Axe                         9  $4.14               $37.26
         115     Spectral Diamond Doomblade               7  $4.25               $29.75
         32      Orenmir                                  6  $4.95               $29.70
         103     Singed Scalpel                           6  $4.87               $29.22
         107     Splitter, Foe Of Subtlety                8  $3.61               $28.88
```

```
In [ ]:
```