HeroesOfPymoli

October 24, 2020

1 Heroes of Pymopoli Pandas Homework

Author: Matt Davies **Date:** 24/10/20

1.0.1 Conclusions from data:

Men are the largest source of revenue for Heroes of Pymopoli (HoP) by gender, however they tend to spend less per user than women and gender-non-specified users

20-24 year olds represent the largest percentage of players and revenue stream for HoP, while users age 10 and below spend the most per transaction

The 5 most popular and 5 highest revenue generating items are at least >33% more expensive than the mean item price in the game

[135]:
```
# Dependencies and Setup
import pandas as pd

# File to Load (Remember to Change These)
file_to_load = "Resources/purchase_data.csv"

# Read Purchasing File and store into Pandas data frame
purchase_data = pd.read_csv(file_to_load)
purchase_data.head()
```

[135]:
```
   Purchase ID             SN  Age Gender  Item ID  \
0            0        Lisim78   20   Male      108
1            1    Lisovynya38   40   Male      143
2            2     Ithergue48   24   Male       92
3            3  Chamassasya86   24   Male      100
4            4      Iskosia90   23   Male      131

                                    Item Name  Price
0  Extraction, Quickblade Of Trembling Hands   3.53
1                           Frenzied Scimitar   1.56
2                                Final Critic   4.88
3                                 Blindscythe   3.27
4                                        Fury   1.44
```

1.1 Player Count

[136]:
```
# Total Number of Players:
# Count length of unique SN in dataframe
player_count = len(purchase_data.SN.unique())
player_count
```

[136]:
```
576
```

1.2 Purchasing Analysis (Total)

[137]:
```
# number of unique items
unique_items = len(purchase_data['Item ID'].unique())
# average price
avg_price = purchase_data['Price'].mean()
# number of purchases
number_purchases = len(purchase_data['Purchase ID'].unique())
# Revenue
revenue = purchase_data['Price'].sum()

# Assemble Dataframe
summary_df = pd.DataFrame({'unique_items':unique_items, 'avg_price':avg_price,␣
 ↪'number_purchases':number_purchases, 'revenue':revenue}, index = [0])

# Format Variables
summary_df.avg_price = summary_df.avg_price.map('${:.2f}'.format)
summary_df.revenue = summary_df.revenue.map('${:.2f}'.format)
summary_df
```

[137]:
```
   unique_items avg_price  number_purchases   revenue
0           179     $3.05               780  $2379.77
```

1.3 Gender Demographics

[146]:
```
# Slice Dataframe to include ONLY unique users
unique_users_df = purchase_data.drop_duplicates(subset=['SN'])

# Total Counts
# Male
male_users_df= unique_users_df.loc[unique_users_df['Gender'] == 'Male', :]
number_male_users = len(male_users_df)
# Female
female_users_df= unique_users_df.loc[unique_users_df['Gender'] == 'Female', :]
number_female_users = len(female_users_df)
```

```
# Other
other_users_df= unique_users_df.loc[unique_users_df['Gender'] == 'Other /␣
↪Non-Disclosed', :]
number_other_users = len(other_users_df)

# Make lists of Data for array input
male_list = ['Male', number_male_users, number_male_users/player_count*100 ]
female_list = ['Female', number_female_users, number_female_users/
↪player_count*100 ]
other_list = ['Other / Non-Disclosed', number_other_users, number_other_users/
↪player_count*100 ]

# Make DataFrame
gender_df = pd.DataFrame([male_list, female_list, other_list],␣
↪columns=['gender','total_count','pct_players'])
gender_df.pct_players = gender_df.pct_players.map('{:.2f}%'.format)

# Set Gender as index
gender_df = gender_df.set_index('gender')
gender_df
```

[146]:
```
                       total_count pct_players
gender
Male                           484      84.03%
Female                          81      14.06%
Other / Non-Disclosed           11       1.91%
```

1.4 Purchasing Analysis (Gender)

[139]:
```
gender_grouped = purchase_data.groupby('Gender')

# Purchase Count
purchase_count = gender_grouped['Price'].count().round(2)
# Average Purchase Price
avg_purchase_price = gender_grouped['Price'].mean().map('${:.2f}'.format)
# Total Purchase Revenue
tot_purchase_rev = gender_grouped['Price'].sum().map('${:,.2f}'.format)
# Avg total purchased per user
user_gender_grouped = purchase_data.groupby(['Gender','SN'])
user_gender_grouped = user_gender_grouped.sum()
avg_tot_user = user_gender_grouped['Price'].groupby('Gender').mean().map('${:.
↪2f}'.format)

# Assemble DataFrame
## Merge Purcase Count with Avg Purchase Price
```

```
gender_purchase_table = pd.merge(purchace_count, avg_purchace_price, on =␣
 ↪'Gender')
## Rename columns
gender_purchase_table = gender_purchase_table.rename(columns={'Price_x':
 ↪'purchase_count', 'Price_y':'average_purchase_price'})
## Merge tot. purchase rev. and avg. tot. per user
gender_purchase_table = pd.merge(gender_purchase_table, tot_purchase_rev, on =␣
 ↪'Gender')
gender_purchase_table = pd.merge(gender_purchase_table, avg_tot_user, on =␣
 ↪'Gender')
## Rename Columns
gender_purchase_table = gender_purchase_table.rename(columns={'Price_x':
 ↪'total_purchase_revenue', 'Price_y':'average_total_purchased_per_user'})

# DataFrame
gender_purchase_table
```

[139]:
```
                       purchase_count average_purchase_price  \
Gender
Female                            113                  $3.20
Male                              652                  $3.02
Other / Non-Disclosed              15                  $3.35

                      total_purchase_revenue average_total_purchased_per_user
Gender
Female                               $361.94                            $4.47
Male                               $1,967.64                            $4.07
Other / Non-Disclosed                 $50.19                            $4.56
```

1.5 Age Demographics

[140]:
```
# Establish bins for ages
bins = [0,9,14,19,24,29,34,39,100]
group_names = ['<10', '10-14', '15-19', '20-24', '25-29', '30-34', '35-40',␣
 ↪'40+']
# Categorize the existing players using age bins
age_purchase_data = purchase_data
age_purchase_data['Age'] = pd.cut(purchase_data['Age'], bins,␣
 ↪labels=group_names)
```

[141]:
```
# Calculate the unique users by age group
counts_by_age = age_purchase_data.groupby(['Age', 'SN']).count().groupby('Age').
 ↪count()
# Discard erroneous columns
users_by_age = counts_by_age['Purchase ID']
```

```
# Creat variable for percentage of players
pct_by_age = users_by_age/player_count*100
# Format
pct_by_age = pct_by_age.map('{:.2f}%'.format)
# Create a summary data frame to hold the results
age_demos = pd.merge(users_by_age, pct_by_age, on = "Age")
age_demos = age_demos.rename(columns={"Purchase ID_x":'unique_users', ␣
↪'Purchase ID_y':'percentage_of_players'})
age_demos
```

[141]:

```
       unique_users percentage_of_players
Age
<10              17                 2.95%
10-14            22                 3.82%
15-19           107                18.58%
20-24           258                44.79%
25-29            77                13.37%
30-34            52                 9.03%
35-40            31                 5.38%
40+              12                 2.08%
```

1.6 Purchasing Analysis (Age)

[142]:

```
age_grouped = age_purchase_data.groupby('Age')

# Purchase Count
age_purchace_count = age_grouped['Price'].count().round(2)
# Average Purchase Price
age_avg_purchace_price = age_grouped['Price'].mean().map('${:.2f}'.format)
# Total Purchase Revenue
age_tot_purchase_rev = age_grouped['Price'].sum().map('${:,.2f}'.format)
# Avg total purchased per user
age_user_gender_grouped = purchase_data.groupby(['Age','SN'])
age_user_gender_grouped = age_user_gender_grouped.sum()
age_avg_tot_user = age_user_gender_grouped['Price'].groupby('Age').mean().
↪map('${:.2f}'.format)

# Make DataFrame
## Merge Purcase Count with Avg Purchase Price
age_purchase_table = pd.merge(age_purchace_count, age_avg_purchace_price, on =␣
↪'Age')
## Rename columns
age_purchase_table = age_purchase_table.rename(columns={'Price_x':
↪'purchase_count', 'Price_y':'average_purchase_price'})
## Merge tot. purchase rev. and avg. tot. per user
```

```
age_purchase_table = pd.merge(age_purchase_table, age_tot_purchase_rev, on =␣
↪'Age')
age_purchase_table = pd.merge(age_purchase_table, age_avg_tot_user, on = 'Age')
## Rename Columns
age_purchase_table = age_purchase_table.rename(columns={'Price_x':
↪'total_purchase_revenue', 'Price_y':'average_total_purchased_per_user'})

# DataFrame
age_purchase_table
```

[142]:
```
       purchase_count average_purchase_price total_purchase_revenue  \
Age
<10                23                  $3.35                 $77.13
10-14              28                  $2.96                 $82.78
15-19             136                  $3.04                $412.89
20-24             365                  $3.05              $1,114.06
25-29             101                  $2.90                $293.00
30-34              73                  $2.93                $214.00
35-40              41                  $3.60                $147.67
40+                13                  $2.94                 $38.24

      average_total_purchased_per_user
Age
<10                              $4.54
10-14                            $3.76
15-19                            $3.86
20-24                            $4.32
25-29                            $3.81
30-34                            $4.12
35-40                            $4.76
40+                              $3.19
```

1.7 Top Spenders

[143]:
```
users_df = purchase_data
# Create dummy variable to count instances on group
users_df['purchase_count'] = 1
# Sum and sort
whale_df = users_df.groupby('SN').sum().sort_values('Price', ascending = False)
# Rename variables
whale_df = whale_df.rename(columns = {'Price':'total_purchase_value'})
# Calculate and add Average Price Column
whale_df['average_purchase_price'] = whale_df['total_purchase_value']/
↪whale_df['purchase_count']
# Drop vestigal rows
```

```
whale_df = whale_df.drop(['Purchase ID', 'Item ID'], axis = 1)
# Format columns
whale_df.total_purchase_value = whale_df.total_purchase_value.map('${:.2f}'.
 ↪format)
whale_df.average_purchase_price = whale_df.average_purchase_price.map('${:.2f}'.
 ↪format)

whale_df.head()
```

[143]:
```
            total_purchase_value  purchase_count average_purchase_price
SN
Lisosia93                 $18.96               5                  $3.79
Idastidru52               $15.45               4                  $3.86
Chamjask73                $13.83               3                  $4.61
Iral74                    $13.62               4                  $3.40
Iskadarya95               $13.10               3                  $4.37
```

1.8 Most Popular and Profitable Items

[144]:
```
# Subset Data
items_df = purchase_data.loc[:, ['Item ID', 'Item Name' ,'Price']]
# Create dummy variable to count instances on group
items_df['purchase_count'] = 1
# Aggregate to unique iteam level
grouped_items = items_df.groupby(['Item ID', 'Item Name']).sum()
# Sort items by most Purchased
popular_items = grouped_items.sort_values('purchase_count', ascending = False)
# Calculate item price
popular_items['item_price'] = popular_items['Price']/
 ↪popular_items['purchase_count']
# Rename columns
popular_items = popular_items.rename(columns={'Price':'total_purchase_value'})

# Format table
popular_items['item_price'] = popular_items['item_price'].map('${:.2f}'.format)
popular_items['total_purchase_value'] = popular_items['total_purchase_value'].
 ↪map('${:.2f}'.format)

popular_items.head()
```

[144]:
```
                                                         total_purchase_value  \
Item ID Item Name
92      Final Critic                                                   $59.99
178     Oathbreaker, Last Hope of the Breaking Storm                   $50.76
145     Fiery Glass Crusader                                           $41.22
```

```
132     Persuasion                                          $28.99
108     Extraction, Quickblade Of Trembling Hands           $31.77

                                                   purchase_count  \
Item ID Item Name
92      Final Critic                                           13
178     Oathbreaker, Last Hope of the Breaking Storm           12
145     Fiery Glass Crusader                                    9
132     Persuasion                                              9
108     Extraction, Quickblade Of Trembling Hands               9

                                                  item_price
Item ID Item Name
92      Final Critic                                   $4.61
178     Oathbreaker, Last Hope of the Breaking Storm       $4.23
145     Fiery Glass Crusader                           $4.58
132     Persuasion                                     $3.22
108     Extraction, Quickblade Of Trembling Hands      $3.53
```

1.9 Most Profitable Items

[145]:
```
# Sort items by revenue
revenue_sort = grouped_items.sort_values('Price', ascending = False)
# Calculate item price
revenue_sort['item_price'] = revenue_sort['Price']/
 ↪revenue_sort['purchase_count']
revenue_sort = revenue_sort.rename(columns={'Price':'total_purchase_value'})

# Format table
revenue_sort['item_price'] = revenue_sort['item_price'].map('${:.2f}'.format)
revenue_sort['total_purchase_value'] = revenue_sort['total_purchase_value'].
 ↪map('${:.2f}'.format)

revenue_sort.head()
```

[145]:
```
                                                  total_purchase_value  \
Item ID Item Name
92      Final Critic                                           $59.99
178     Oathbreaker, Last Hope of the Breaking Storm           $50.76
82      Nirvana                                                $44.10
145     Fiery Glass Crusader                                   $41.22
103     Singed Scalpel                                         $34.80

                                                   purchase_count  \
Item ID Item Name
92      Final Critic                                           13
```

```
178     Oathbreaker, Last Hope of the Breaking Storm             12
82      Nirvana                                                   9
145     Fiery Glass Crusader                                      9
103     Singed Scalpel                                            8

                                                     item_price
Item ID Item Name
92      Final Critic                                      $4.61
178     Oathbreaker, Last Hope of the Breaking Storm      $4.23
82      Nirvana                                           $4.90
145     Fiery Glass Crusader                              $4.58
103     Singed Scalpel                                    $4.35
```

```
[ ]:
```