Heroes of Pymoli Data Analysis

- Observation 1: The majority of the players of Heroes of Pymoli are within the age range 15-25. There are some younger players and some older players, but of the 573 players, 373 (65.0%) players fall within this range.
- Observations 2: The proportion of players by gender and the proportion of money spent by gender are approximately equal. For males, 81.15% of the players are male, while 81.69% of the purchases come from males. For females, 17.45% of the players are female, while 16.75% of the purchases come from females. For other, 1.40% of the players are female, while 1.56% of the purchases come from this group.
- Observation 3: The most profitable items are items that are generally more expensive and have decent sales volume. The top 5 listed show that all of the items are well above the average price of an item which is 2.93. Alternatively, the best selling items are much cheaper and all fall below the average price of 2.93.

In [1]:

```
import pandas as pd
```

In [2]:

```
# read json file

filepath = "purchase_data.json"

pymoli_raw = pd.read_json(filepath, orient= 'columns')
# pymoli_raw.head()
```

In [3]:

```
# total number of players

totalPlayersList = pymoli_raw["SN"].unique()
totalPlayers = len(totalPlayersList)
totalPlayer_df  = pd.DataFrame({"Total Number of Players":[totalPlayers]})
totalPlayer_df
```

Out[3]:

	Total Number of Players
0	573

In [4]:

```
# Purchasing Analysis (Total)
# Number of Unique Items
itemsList = pymoli_raw["Item ID"].unique()
items = len(itemsList)
# items
```

In [5]:

```
# Average Purchase Price
avgPP = pymoli_raw["Price"].mean()
# avgPP
```

In [6]:

```
# Total Number of Purchases
pymoli_sort = pymoli_raw.sort_values(by=["Price"])
purchases = len(pymoli_sort)
# purchases
```

In [7]:

```
# Total Revenue
revenue = pymoli_raw["Price"].sum()
# revenue
```

In [8]:

```
# move to dataframe
items_df = pd.DataFrame({"Number of Unique Items":[items],
                        "Average Price": [avgPP],
                        "Number of Purchases": [purchases],
                        "Total Revenue": [revenue]})
```

In [9]:

```
items_df["Average Price"] = items_df["Average Price"].map("${0:,.2f}".format)
items_df["Total Revenue"] = items_df["Total Revenue"].map("${0:,.2f}".format)
items_df = items_df[["Number of Unique Items","Average Price", "Number of Purc
hases","Total Revenue"]]
items_df
```

Out[9]:

	Number of Unique Items	Average Price	Number of Purchases	Total Revenue
0	183	$2.93	780	$2,286.33

In [10]:

```
# Gender Demographics
# Total unique players in list
player_pymoli = pymoli_sort.drop_duplicates(['SN'])
# player_pymoli.head()
```

In [11]:

```
# Percentage and Count of Male Players
maleCount = player_pymoli["Gender"].value_counts()['Male']
# print ("# of Male Players: "+ str(maleCount))
malePercent = 100 * maleCount/totalPlayers
# print ("% of Male Players: " + str(malePercent) + "%")
```

In [12]:

```
# Percentage and Count of Female Players
femaleCount = player_pymoli["Gender"].value_counts()["Female"]
# print ("# of Female Players: " + str(femaleCount))
femalePercent = 100 * femaleCount/totalPlayers
# print ("% of Female Players: " + str(femalePercent) + "%")
```

In [13]:

```
# Percentage and Count of Other / Non-Disclosed
otherCount = player_pymoli["Gender"].value_counts()["Other / Non-Disclosed"]
# print ("# of Other / Non-Disclosed Players: " + str(otherCount))
otherPercent = 100 * otherCount/totalPlayers
# print ("% of Other / Non-Disclosed Players: " + str(otherPercent) + "%")
```

In [14]:

```
genderBreakdown = pd.DataFrame({"Gender": ["Male","Female","Other"],
                                "Percentage of Players": [malePercent, femaleP
ercent, otherPercent],
                                "Total Count": [maleCount, femaleCount, otherC
ount]})

genderBreakdown["Percentage of Players"] = genderBreakdown["Percentage of Play
ers"].map("{0:,.2f}%".format)
genderBreakdown.set_index("Gender", inplace= True)
del genderBreakdown.index.name
genderBreakdown
```

Out[14]:

	Percentage of Players	Total Count
Male	81.15%	465
Female	17.45%	100
Other	1.40%	8

In [15]:

```
# The below each broken by gender
# Purchase Count

malePurchase = pymoli_raw["Gender"].value_counts()["Male"]
# print ("Purchases by males: " + str(malePurchase))
femalePurchase = pymoli_raw["Gender"].value_counts()["Female"]
# print ("Purchases by females: " + str(femalePurchase))
otherPurchase = pymoli_raw["Gender"].value_counts()["Other / Non-Disclosed"]
# print ("Purchases by Other / Non-Disclosed: " + str(otherPurchase))
```

In [16]:

```
# Average Purchase Price
totalMalePurch = pymoli_raw[pymoli_raw["Gender"]=="Male"].sum()["Price"]
avgMalePurch = totalMalePurch / malePurchase
# print ("Average Purchase Price by Males: $" + str(avgMalePurch))

totalFemalePurch = pymoli_raw[pymoli_raw["Gender"]=="Female"].sum()["Price"]
avgFemalePurch = totalFemalePurch / femalePurchase
# print ("Average Purchase Price by Females: $" + str(avgFemalePurch))

totalOtherPurch = pymoli_raw[pymoli_raw["Gender"]=="Other / Non-Disclosed"].su
m()["Price"]
avgOtherPurch = totalOtherPurch / otherPurchase
# print ("Average Purchase Price by Other / Non-Disclosed: $" + str(avgOtherPu
rch))
```

In [17]:

```
# Total Purchase Value
# print ("Total Purchased by Males: $" + str(totalMalePurch))
# print ("Total Purchased by Females: $" + str(totalFemalePurch))
# print ("Total Purchased by Other: $" + str(totalOtherPurch))
```

In [18]:

```
#Standard Deviation of Purchase Price
standardDeviationPrice = pymoli_raw["Price"].std()
# standardDeviationPrice
```

In [19]:

```
# Normalized Totals
normalizedMale = (avgMalePurch - avgPP) / standardDeviationPrice
normalizedFemale = (avgFemalePurch - avgPP) / standardDeviationPrice
normalizedOther = (avgOtherPurch - avgPP) / standardDeviationPrice

# print(normalizedMale)
# print(normalizedFemale)
# print(normalizedOther)
```

In [20]:

```
# gender purchase breakdown
genderPurchaseBreakdown = pd.DataFrame({"Gender": ["Male","Female","Other"],
                                        "Purchase Count": [malePurchase, femal
ePurchase, otherPurchase],
                                        "Average Purchase Price": [avgMalePurc
h, avgFemalePurch, avgOtherPurch],
                                        "Total Purchase Value": [totalMalePurc
h, totalFemalePurch, totalOtherPurch],
                                        "Normalized Totals": [normalizedMale,
normalizedFemale, normalizedOther],})

genderPurchaseBreakdown.set_index("Gender", inplace= True)

genderPurchaseBreakdown["Average Purchase Price"] = genderPurchaseBreakdown["A
verage Purchase Price"].map("${0:,.2f}".format)
genderPurchaseBreakdown["Total Purchase Value"] = genderPurchaseBreakdown["Tot
al Purchase Value"].map("${0:,.2f}".format)
genderPurchaseBreakdown["Normalized Totals"] = genderPurchaseBreakdown["Normal
ized Totals"].map("{0:,.4f}".format)
genderPurchaseBreakdown = genderPurchaseBreakdown[["Purchase Count", "Average
 Purchase Price", "Total Purchase Value", "Normalized Totals"]]
genderPurchaseBreakdown
```

Out[20]:

	Purchase Count	Average Purchase Price	Total Purchase Value	Normalized Totals
Gender				
Male	633	$2.95	$1,867.68	0.0173
Female	136	$2.82	$382.91	-0.1037
Other	11	$3.25	$35.74	0.2849

In [21]:

```
# Age Demographics

# The below each broken into bins of 4 years (i.e. <10, 10-14, 15-19, etc.)
bins = [0, 10, 15, 20, 25, 30, 35, 40, 60]
ageGroup = ["0-10","10-15","15-20","20-25","25-30","30-35","35-40","40+"]
pymoli_raw["Age Group"] = pd.cut(pymoli_raw["Age"], bins, labels = ageGroup)
# pymoli_raw.head()

# Normalized Totals
```

In [22]:

```
# Count of Players in Age Range
ageRange = pymoli_raw.drop_duplicates(['SN'])
# ageRange.head()
# 0-10
players0010 = ageRange["Age Group"].value_counts()["0-10"]
percent0010 = 100 * players0010/totalPlayers
# 10-15
players1015 = ageRange["Age Group"].value_counts()["10-15"]
percent1015 = 100 * players1015/totalPlayers
# 15-20
players1520 = ageRange["Age Group"].value_counts()["15-20"]
percent1520 = 100 * players1520/totalPlayers
# 20-25
players2025 = ageRange["Age Group"].value_counts()["20-25"]
percent2025 = 100 * players2025/totalPlayers
# 25-30
players2530 = ageRange["Age Group"].value_counts()["25-30"]
percent2530 = 100 * players2530/totalPlayers
# 30-35
players3035 = ageRange["Age Group"].value_counts()["30-35"]
percent3035 = 100 * players3035/totalPlayers
# 35-40
players3540 = ageRange["Age Group"].value_counts()["35-40"]
percent3540 = 100 * players3540/totalPlayers
# 40 +
players4060 = ageRange["Age Group"].value_counts()["40+"]
percent4060 = 100 * players4060/totalPlayers

playerDemographic = pd.DataFrame({"Age Range": ["0-10","10-15","15-20","20-25"
,"25-30","30-35","35-40","40+"],
                                  "Percentage of Players": [percent0010, perce
nt1015, percent1520, percent2025, percent2530, percent3035, percent3540, perce
nt4060],
                                 "Total Count": [players0010, players1015, pla
yers1520, players2025, players2530, players3035, players3540, players4060]})

playerDemographic["Percentage of Players"] = playerDemographic["Percentage of
 Players"].map("{0:,.2f}%".format)
playerDemographic.set_index("Age Range", inplace=True)

playerDemographic
```

Out[22]:

	Percentage of Players	Total Count
Age Range		
0-10	3.84%	22
10-15	9.42%	54
15-20	24.26%	139
20-25	40.84%	234
25-30	9.08%	52
30-35	7.68%	44
35-40	4.36%	25
40+	0.52%	3

In [23]:

```
# Purchase Count
# 0-10
purchase0010 = pymoli_raw["Age Group"].value_counts()["0-10"]
# print ("Amount Purchased by 0-10: " + str(purchase0010))
# 10-15
purchase1015 = pymoli_raw["Age Group"].value_counts()["10-15"]
# print ("Amount Purchased by 10-15: " + str(purchase1015))
# 15-20
purchase1520 = pymoli_raw["Age Group"].value_counts()["15-20"]
# print ("Amount Purchased by 15-20: " + str(purchase1520))
# 20-25
purchase2025 = pymoli_raw["Age Group"].value_counts()["20-25"]
# print ("Amount Purchased by 20-25: " + str(purchase2025))
# 25-30
purchase2530 = pymoli_raw["Age Group"].value_counts()["25-30"]
# print ("Amount Purchased by 25-30: " + str(purchase2530))
# 30-35
purchase3035 = pymoli_raw["Age Group"].value_counts()["30-35"]
# print ("Amount Purchased by 30-35: " + str(purchase3035))
# 35-40
purchase3540 = pymoli_raw["Age Group"].value_counts()["35-40"]
# print ("Amount Purchased by 35:40: " + str(purchase3540))
# 40 +
purchase4060 = pymoli_raw["Age Group"].value_counts()["40+"]
# print ("Amount Purchased by 40+: " + str(purchase4060))
```

In [24]:

```
# Average Purchase Price
# Total Purchase Value
# 0-10
totalPurch0010 = pymoli_raw[pymoli_raw["Age Group"]=="0-10"].sum()["Price"]
avgPurch0010 = totalPurch0010 / purchase0010
# 10-15
totalPurch1015 = pymoli_raw[pymoli_raw["Age Group"]=="10-15"].sum()["Price"]
avgPurch1015 = totalPurch1015 / purchase1015
# 15-20
totalPurch1520 = pymoli_raw[pymoli_raw["Age Group"]=="15-20"].sum()["Price"]
avgPurch1520 = totalPurch1520 / purchase1520
# 20-25
totalPurch2025 = pymoli_raw[pymoli_raw["Age Group"]=="20-25"].sum()["Price"]
avgPurch2025 = totalPurch2025 / purchase2025
# 25-30
totalPurch2530 = pymoli_raw[pymoli_raw["Age Group"]=="25-30"].sum()["Price"]
avgPurch2530 = totalPurch2530 / purchase2530
# 30-35
totalPurch3035 = pymoli_raw[pymoli_raw["Age Group"]=="30-35"].sum()["Price"]
avgPurch3035 = totalPurch3035 / purchase3035
# 35-40
totalPurch3540 = pymoli_raw[pymoli_raw["Age Group"]=="35-40"].sum()["Price"]
avgPurch3540 = totalPurch3540 / purchase3540
# 40 +
totalPurch4060 = pymoli_raw[pymoli_raw["Age Group"]=="40+"].sum()["Price"]
avgPurch4060 = totalPurch4060 / purchase4060
```

In [25]:

```
# Normalized Totals

# 0-10
normalized0010 = (avgPurch0010-avgPP) / standardDeviationPrice
# 10-15
normalized1015 = (avgPurch1015-avgPP) / standardDeviationPrice
# 15-20
normalized1520 = (avgPurch1520-avgPP) / standardDeviationPrice
# 20-25
normalized2025 = (avgPurch2025-avgPP) / standardDeviationPrice
# 25-30
normalized2530 = (avgPurch2530-avgPP) / standardDeviationPrice
# 30-35
normalized3035 = (avgPurch3035-avgPP) / standardDeviationPrice
# 35-40
normalized3540 = (avgPurch3540-avgPP) / standardDeviationPrice
# 40 +
normalized4060 = (avgPurch4060-avgPP) / standardDeviationPrice
```

In [26]:

```
agePurchaseBreakdown = pd.DataFrame({"Age Group":["0-10","10-15","15-20","20-2
5","25-30","30-35","35-40","40+"],
                                    "Purchase Count":[purchase0010, purchase10
15, purchase1520, purchase2025, purchase2530, purchase3035, purchase3540, purc
hase4060],
                                    "Average Purchase Price":[avgPurch0010, av
gPurch1015, avgPurch1520, avgPurch2025, avgPurch2530, avgPurch3035, avgPurch35
40, avgPurch4060],
                                    "Total Purchase Value":[totalPurch0010, to
talPurch1015, totalPurch1520, totalPurch2025, totalPurch2530, totalPurch3035,
totalPurch3540, totalPurch4060],
                                    "Normalized Cost":[normalized0010, normali
zed1015, normalized1520, normalized2025, normalized2530, normalized3035, norma
lized3540, normalized4060]})
agePurchaseBreakdown.set_index("Age Group", inplace= True)
del agePurchaseBreakdown.index.name
agePurchaseBreakdown["Average Purchase Price"] = agePurchaseBreakdown["Average
 Purchase Price"].map("${0:,.2f}".format)
agePurchaseBreakdown["Normalized Cost"] = agePurchaseBreakdown["Normalized Cos
t"].map("{0:,.4f}".format)
agePurchaseBreakdown["Total Purchase Value"] = agePurchaseBreakdown["Total Pur
chase Value"].map("${0:,.2f}".format)

agePurchaseBreakdown = agePurchaseBreakdown[["Purchase Count", "Average Purcha
se Price", "Total Purchase Value", "Normalized Cost"]]
agePurchaseBreakdown
```

Out[26]:

	Purchase Count	Average Purchase Price	Total Purchase Value	Normalized Cost
0-10	32	$3.02	$96.62	0.0790
10-15	78	$2.87	$224.15	-0.0515
15-20	184	$2.87	$528.74	-0.0516
20-25	305	$2.96	$902.61	0.0253
25-30	76	$2.89	$219.82	-0.0348
30-35	58	$3.07	$178.26	0.1275
35-40	44	$2.90	$127.49	-0.0302
40+	3	$2.88	$8.64	-0.0459

In [27]:

```
# **Top Spenders**

# * Identify the the top 5 spenders in the game by total purchase value, then
 list (in a table):
#   * SN
#   * Purchase Count
snPurchaseCount = pymoli_raw['SN'].value_counts()
```

In [28]:

```
#   * Total Purchase Value
groupedPymoli = pymoli_raw.groupby(['SN'])
# groupedPymoli.count().head(10)
```

In [29]:

```
snTotalPurch = groupedPymoli['Price'].sum()
# snTotalPurch
```

In [30]:

```
snAvgPurch = groupedPymoli['Price'].mean()
# snAvgPurch
```

In [31]:

```
snPurchaseDF = pd.DataFrame({"Purchase Count": snPurchaseCount,
                            "Average Purchase Price": snAvgPurch,
                            "Total Purchase Value": snTotalPurch})
snPurchaseDF["Average Purchase Price"] = snPurchaseDF["Average Purchase Price"
].map("${0:,.2f}".format)

snPurchaseDF = snPurchaseDF[["Purchase Count","Average Purchase Price","Total
 Purchase Value"]]
snPurchaseDF = snPurchaseDF.sort_values(["Total Purchase Value"], ascending =
False)
snPurchaseDF["Total Purchase Value"] = snPurchaseDF["Total Purchase Value"].ma
p("${0:,.2f}".format)
snPurchaseDF.index.name = "SN"
snPurchaseDF.head(5)
```

Out[31]:

	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 [32]:

```
itemPurchaseCount = pymoli_raw['Item ID'].value_counts()
# itemPurchaseCount
```

In [33]:

```
groupedItem = pymoli_raw.groupby(['Item ID'])
# groupedItem.count().head(10)
```

In [34]:

```
itemPrice = groupedItem['Price'].mean()
# itemPrice
```

In [35]:

```
itemTotalPurch = groupedItem['Price'].sum()
# itemTotalPurch
```

In [36]:

```
# get itemID for item
itemCatalog = pymoli_raw.drop_duplicates(['Item ID'])
itemCatalog.set_index(['Item ID'], inplace=True)
itemCatalog = itemCatalog[["Item Name"]]
itemCatalog = itemCatalog.sort_index()
# itemCatalog.head(5)
```

In [37]:

```
itemDF = pd.DataFrame({"Purchase Count": itemPurchaseCount,
                      "Item Price": itemPrice,
                      "Total Purchase Value": itemTotalPurch})
itemDF.head()

merge_itemDF = pd.merge(itemDF, itemCatalog, left_index=True, right_index=True
, how="left")

# itemDF["Item ID"] = itemDF["Item ID"].map("{0:,.0f}".format)
merge_itemDF = merge_itemDF.reset_index()
merge_itemDF = merge_itemDF.rename(columns={"index":"Item Name"})
merge_itemDF = merge_itemDF.set_index(['Item ID', 'Item Name'])
# merge_itemDF.head()
```

In [38]:

```
# Most Popular Items by Purchase Count
popularItems = merge_itemDF.sort_values(["Purchase Count"], ascending = False)
popularItems["Item Price"] = popularItems["Item Price"].map("${0:,.2f}".format
)
popularItems["Total Purchase Value"] = popularItems["Total Purchase Value"].ma
p("${0:,.2f}".format)
popularItems.head(5)
```

Out[38]:

		Item Price	Purchase Count	Total Purchase Value
Item ID	**Item Name**			
39	**Betrayal, Whisper of Grieving Widows**	$2.35	11	$25.85
84	**Arcane Gem**	$2.23	11	$24.53
31	**Trickster**	$2.07	9	$18.63
175	**Woeful Adamantite Claymore**	$1.24	9	$11.16
13	**Serenity**	$1.49	9	$13.41

In [39]:

```
# Most Profitable
profitItems = merge_itemDF.sort_values(["Total Purchase Value"], ascending = False)
profitItems["Item Price"] = profitItems["Item Price"].map("${0:,.2f}".format)
profitItems["Total Purchase Value"] = profitItems["Total Purchase Value"].map("${0:,.2f}".format)
profitItems.head(5)
```

Out[39]:

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