ROBERT H. SMITH
SCHOOL OF BUSINESS

THE UNIVERSITY OF MARYLAND

****

Final Python Report

Recommendations for UMD Alumni Association
on First Time Attendees and Major Prospects

Team BUDT704_0507_04

Maryland, December 10th, 2021

Prepared by Yun-Jung Fan, Haley O'Reagan, Hsin-Yuan Yen, and Phuong Huynh

Advised by Professor Adam Lee

Table of Contents

Team Work

Overall, our team worked well together and we were able to accomplish the necessary tasks on time. We met once before Thanksgiving break in person to go over the assignment, brainstorm potential solutions, understand which variables we thought would be most useful in answering our mission objective, and begin some preliminary code. We then decided it was best to divide the variables one per person and test different methods over Thanksgiving break to see which solutions worked best. When we got back from break, we were able to meet many times in-person and on Zoom to go over our findings, compare results, ask questions, and bring up any concerns we may have.

Using a shared drive, we were able to check each person's code and collaborate on our doc/presentation. In the end, we were able to create a cohesive and comprehensive presentation that displayed our findings thoroughly. Looking back, we should have cut the material on our slides down slightly in order to meet the 8 minute time frame, but overall we felt it went well.

Excel Transformation

Our team challenged ourselves to not modify the Excel data in any way unless it was through python. For everyone in our group, this semester was the first time using Python and the first time getting to learn a coding language. While we were tempted to originally transform the data in Excel since we were more familiar with the platform and the data wasn't too extensive, we wanted to put our coding knowledge to the test. Although it did take some additional steps and outside research, we were able to categorize, group by, and plot all the necessary data in python itself. The only exception to this rule was putting each year's data into the same sheet after exporting the dummified 'group code' variables from python. This excel file has been titled "FullData".

Initial Ideas

During our first meeting we came up with a few ideas on how we wanted to approach this project. We knew activity code had the most unique data points, and it may be harder to group rows together using this variable. We decided to instead focus on other variables that could be categorized into meaningful sub-groups that would be easier to understand and analyze. Therefore, we turned our attention to four variables: location, group code, age, and date.
We also considered creating additional mission objectives, such as seeing which events had the most recurring attendees, as these may have been more popular events and something the Alumni Association may want to market to potential first time attendees to build CLV. However, after some basic testing in Python, we realized this analysis may not be as important to the Alumni Association and we should focus the majority of our time and efforts on first time attendees and major prospects.

Methods

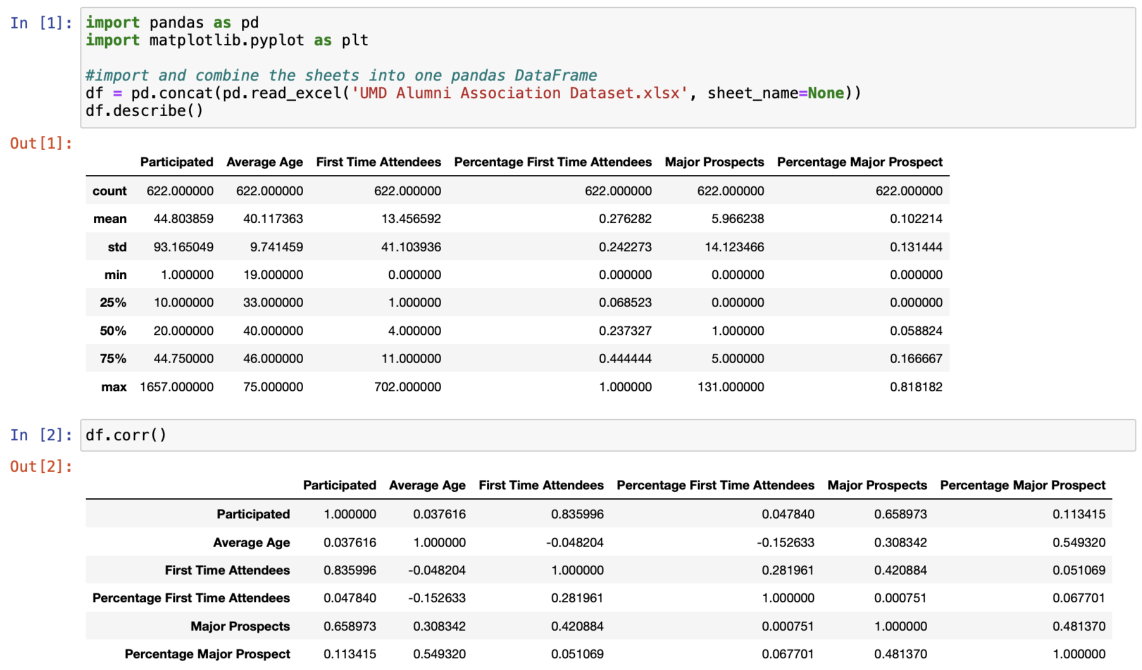
To start our project, our group wanted to combine all the data sheets into one data frame to make our analysis easier. We knew that environmental factors and unforeseen circumstances can affect event data annually (i.e. Covid-19), therefore we thought it was best to concatenate the data sheets to get a more representative data frame.

In order to understand which variables lead to greater attendance for both first time attendees and major prospects, our team utilized three methods mainly. The first method we used was a correlation analysis so we could get a general understanding of which variables had a strong statistical relationship. The correlation matrix was most useful for age, as it was the only numerical variable, and we were able to see its relationship with our two variables of interest.

The next method we utilized was k-means clustering, specifically for group code. We also tried clustering on average age and location, but found that no clusters were significant in explaining the dependent variable (low p-value) or that we couldn't find any obvious trends among the data points. We first separated the data using a 2-means cluster analysis but quickly realized we needed to increase the number of clusters in order to get a more even distribution. After some trial and error, we came to the conclusion of using 5 clusters. Once plotted, we were able to move on to our third method, linear regression.

We ran a linear regression on each cluster and returned the p-value. Using an alpha of 0.05, we wanted to see if any p-values were below this level, which would indicate that the data in the cluster is significant in explaining our independent variables, percentage of major prospects and percentage of first time attendees. We also plotted the linear regression of each cluster to see how the data points lie around the line of best fit. After printing the index, we were able to see which group codes belonged to which clusters, and then filtered the excel file to see if there were any trends in group code type/description.

Creating the Code



```
In [1]: import pandas as pd
        import matplotlib.pyplot as plt

        #import and combine the sheets into one pandas DataFrame
        df = pd.concat(pd.read_excel('UMD Alumni Association Dataset.xlsx', sheet_name=None))
        df.describe()
```

Out[1]:

	Participated	Average Age	First Time Attendees	Percentage First Time Attendees	Major Prospects	Percentage Major Prospect
count	622.000000	622.000000	622.000000	622.000000	622.000000	622.000000
mean	44.803859	40.117363	13.456592	0.276282	5.966238	0.102214
std	93.165049	9.741459	41.103936	0.242273	14.123466	0.131444
min	1.000000	19.000000	0.000000	0.000000	0.000000	0.000000
25%	10.000000	33.000000	1.000000	0.068523	0.000000	0.000000
50%	20.000000	40.000000	4.000000	0.237327	1.000000	0.058824
75%	44.750000	46.000000	11.000000	0.444444	5.000000	0.166667
max	1657.000000	75.000000	702.000000	1.000000	131.000000	0.818182

```
In [2]: df.corr()
```

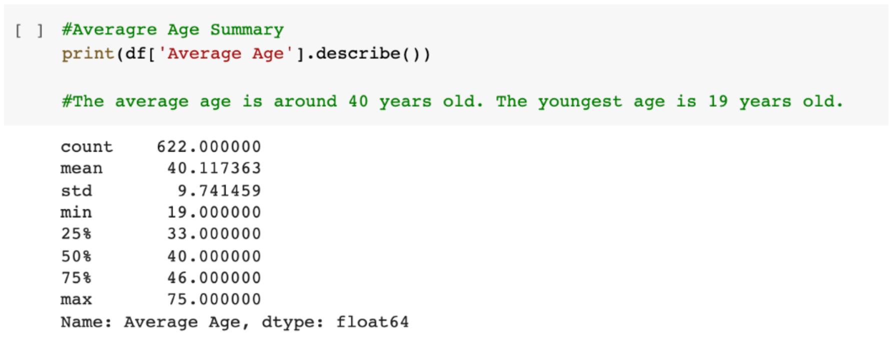
Out[2]:

	Participated	Average Age	First Time Attendees	Percentage First Time Attendees	Major Prospects	Percentage Major Prospect
Participated	1.000000	0.037616	0.835996	0.047840	0.658973	0.113415
Average Age	0.037616	1.000000	-0.048204	-0.152633	0.308342	0.549320
First Time Attendees	0.835996	-0.048204	1.000000	0.281961	0.420884	0.051069
Percentage First Time Attendees	0.047840	-0.152633	0.281961	1.000000	0.000751	0.067701
Major Prospects	0.658973	0.308342	0.420884	0.000751	1.000000	0.481370
Percentage Major Prospect	0.113415	0.549320	0.051069	0.067701	0.481370	1.000000

At the beginning, we combine all the sheets into one pandas DataFrame so that we can analyze the whole data from many years at one time. We also run the correlation code to have a quick look into the relationship between variables.

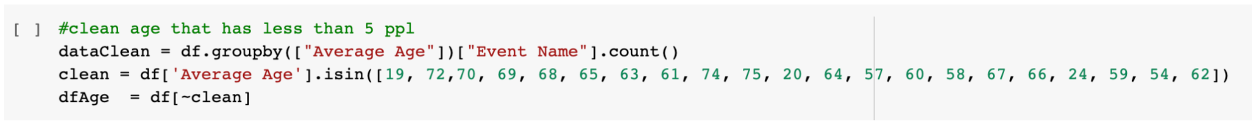
1. Average Age:

To analyze the relationship between average age and our mission objectives, we first use the ".describe()" to get the summary that the mean average age for these events is around 40 years old. The youngest is 19 years old.



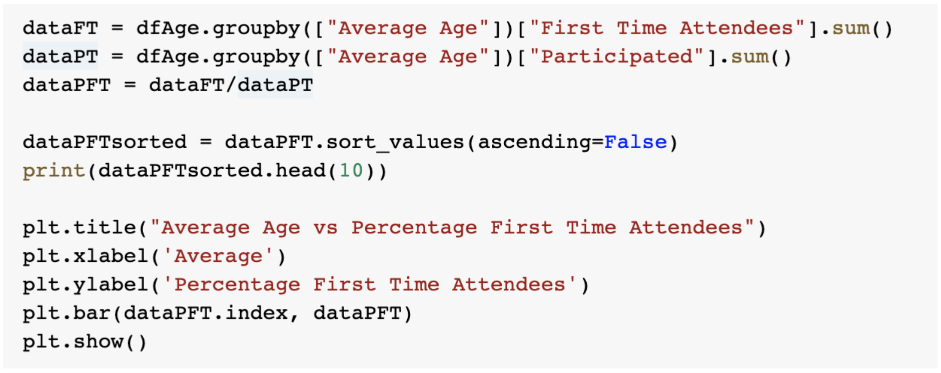
```
[ ]  #Averagre Age Summary
     print(df['Average Age'].describe())

     #The average age is around 40 years old. The youngest age is 19 years old.

     count    622.000000
     mean      40.117363
     std        9.741459
     min       19.000000
     25%       33.000000
     50%       40.000000
     75%       46.000000
     max       75.000000
     Name: Average Age, dtype: float64
```

We clean the data and get a new dataframe called "dfAge" which excludes all outliers of average ages that only have less than 5 events. we use "GroupBy" and ".sum()" to recalculate the percentage of first time attendees to analyze.

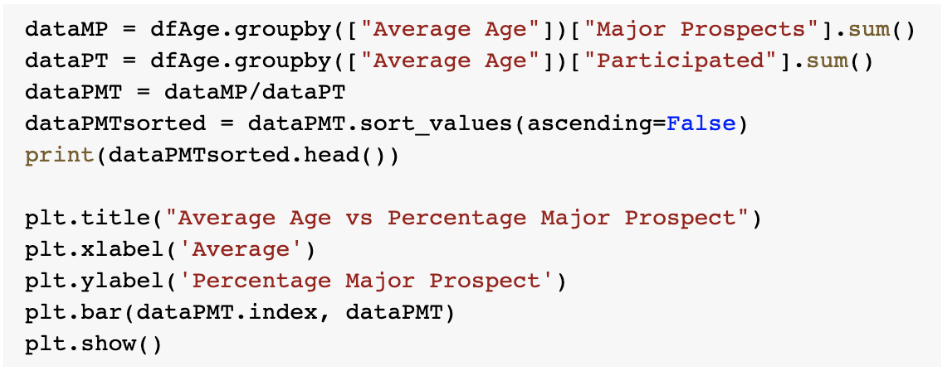


```
[ ]  #clean age that has less than 5 ppl
     dataClean = df.groupby(["Average Age"])["Event Name"].count()
     clean = df['Average Age'].isin([19, 72,70, 69, 68, 65, 63, 61, 74, 75, 20, 64, 57, 60, 58, 67, 66, 24, 59, 54, 62])
     dfAge  = df[~clean]
```

Then, we use “GroupBy” and “.sum()” to recalculate the percentage of first time attendees and major prospects to analyze. At the beginning, we used“Group By AvgAge” to get the top 10 ages with high percentages of first time attendees or major prospects, so that we can focus on events that have average ages between these ranges to do further research. We also do a histogram plot to show the distribution of Average Age.

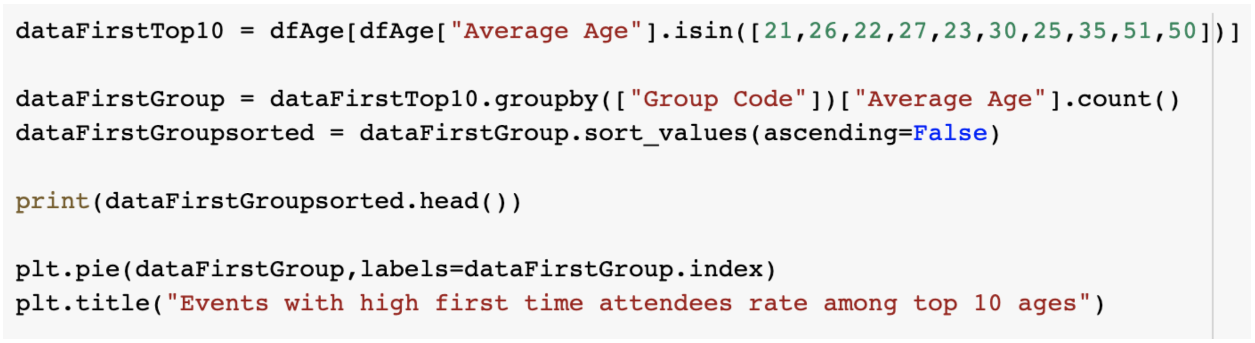


```
dataFT = dfAge.groupby(["Average Age"])["First Time Attendees"].sum()
dataPT = dfAge.groupby(["Average Age"])["Participated"].sum()
dataPFT = dataFT/dataPT

dataPFTsorted = dataPFT.sort_values(ascending=False)
print(dataPFTsorted.head(10))

plt.title("Average Age vs Percentage First Time Attendees")
plt.xlabel('Average')
plt.ylabel('Percentage First Time Attendees')
plt.bar(dataPFT.index, dataPFT)
plt.show()
```



```
dataMP = dfAge.groupby(["Average Age"])["Major Prospects"].sum()
dataPT = dfAge.groupby(["Average Age"])["Participated"].sum()
dataPMT = dataMP/dataPT
dataPMTsorted = dataPMT.sort_values(ascending=False)
print(dataPMTsorted.head())

plt.title("Average Age vs Percentage Major Prospect")
plt.xlabel('Average')
plt.ylabel('Percentage Major Prospect')
plt.bar(dataPMT.index, dataPMT)
plt.show()
```

After targeting the audience, we group the new data “dataFirstTop10” and “dataProspectTop” by Group Code as “dataFirstGroup” and “dataProspectGroup” to get the top 5 Group Codes within our target audiences. We also do the pie chart to show the distribution.



```
dataFirstTop10 = dfAge[dfAge["Average Age"].isin([21,26,22,27,23,30,25,35,51,50])]

dataFirstGroup = dataFirstTop10.groupby(["Group Code"])["Average Age"].count()
dataFirstGroupsorted = dataFirstGroup.sort_values(ascending=False)

print(dataFirstGroupsorted.head())

plt.pie(dataFirstGroup,labels=dataFirstGroup.index)
plt.title("Events with high first time attendees rate among top 10 ages")
```

```
dataProspectTop = dfAge[(dfAge["Average Age"] >= 50) & (dfAge["Average Age"] <= 55)]

dataProspectGroup = dataProspectTop.groupby(["Group Code"])["Average Age"].count()
dataProspectGroupsorted = dataProspectGroup.sort_values(ascending=False)

print(dataProspectGroupsorted.head(3))

plt.pie(dataProspectGroup,labels=dataProspectGroup.index)
plt.title("Events with high first major prospect rate with ages >= 50")
```

As with similar methods for Group Code, we also group the datas by "Location Code" as "dataFirstLoc" and "dataProspectLoc" to get top 5 Location Codes within our target audiences. We also do the pie chart to show the distribution.

```
dataFirstLoc = dataFirstTop10.groupby(["Location Code"])["Average Age"].count()
dataFirstLocsorted = dataFirstLoc.sort_values(ascending=False)

print(dataFirstLocsorted.head())

plt.pie(dataFirstLoc,labels=dataFirstLoc.index)
plt.title("Locations with high first time attendees rate among top 10 ages")
```

```
dataProspectLoc = dataProspectTop.groupby(["Location Code"])["Average Age"].count()
dataProspectLocsorted = dataProspectLoc.sort_values(ascending=False)

print(dataProspectLocsorted.head())

plt.pie(dataProspectLoc,labels=dataProspectLoc.index)
plt.title("Locations with high Percentage Major Prospect with ages >= 50")
```

To get further insight on Average Age, we divide people into two group. One is young people less than 30 years old and old people more than 50 years old.

```
[24] dataYoung = dfAge[dfAge["Average Age"] <= 30]
     dataElder = dfAge[dfAge["Average Age"] >= 50]
```

We group "dataYoung" and "dataElder" by Group Code to get the top three Group Code that young and old people first attending to.

```
dataYoungGroup = dataYoung.groupby(["Group Code"])["Average Age"].count()
dataYoungGroupsorted = dataYoungGroup.sort_values(ascending=False)
dataYoungGroupsorted.head(3)
```

```
dataElderGroup = dataElder.groupby(["Group Code"])["Average Age"].count()
dataElderGroupsorted = dataElderGroup.sort_values(ascending=False)
dataElderGroupsorted.head(3)
```

We use the same method as what we did on Group Code earlier to Location Code, so that we can get the top three locations of events that young and old people first attending to.

```python
dataYoungLoc = dataYoung.groupby(["Location Code"])["Average Age"].count()
dataYoungLocsorted = dataYoungLoc.sort_values(ascending=False)
dataYoungLocsorted.head(3)
```

```python
dataElderLoc = dataElder.groupby(["Location Code"])["Average Age"].count()
dataElderLocsorted = dataElderLoc.sort_values(ascending=False)
dataElderLocsorted.head(3)
```

At the end, we use pit chart to show the distribution of four results.

```python
#Using pie chart to show the distribution
plt.figure(figsize=(15, 3))

plt.subplot(141)
plt.pie(dataYoungGroup,labels=dataYoungGroup.index)
plt.title("Young People Group Code")

plt.subplot(142)
plt.pie(dataElderGroup,labels=dataElderGroup.index)
plt.title("Elder Group Code")

plt.subplot(143)
plt.pie(dataYoungLoc,labels=dataYoungLoc.index)
plt.title("Young People Loc Code")

plt.subplot(144)
plt.pie(dataElderLoc,labels=dataElderLoc.index)
plt.title("Elder Loc Code")


plt.show()
```

2. Date:

To analyze the trend of our objectives, we use ".split()" to split the event date into the year, month, and day.

```python
df = df.join(df['Event Date'].astype(str).str.split('-',expand= True).rename(columns={0:'Year', 1:'Month', 2:'Day'}))
df
```

ation ption	Group Code	Group Description	Event Date	Participated	Average Age	First Time Attendees	Percentage First Time Attendees	Major Prospects	Percentage Major Prospect	LocationCode_dum	ActivityCode_dum	Year	Month	Day
MV- stern hore	PS9	CP Social-General	2019-07-06	63	52	5	0.079365	12	0.190476	328	328	2019	07	06
line- binar	PC9	CP ProDev-General	2019-07-10	7	56	1	0.142857	0	0.000000	177	177	2019	07	10
CP east- NNJ	PC9	CP ProDev-General	2019-07-11	28	45	2	0.071429	2	0.071429	159	159	2019	07	11

Then, we selected the columns that we needed to focus on, such as "Percentage First Time Attendees", "First Time Attendees", "Event Date", and so on. We also added a new column, "Year & Month," to make it easier to analyze.

```
attendees = df[['Group Code','Percentage First Time Attendees','First Time Attendees','Event Date','Year','Month',
                'Major Prospects', 'Percentage Major Prospect']]
attendees['Year&Month'] = attendees['Year'] +"/"+ attendees['Month']
attendees
```

Further, we use "GroupBy" and ".mean()" to recalculate each value in each group. And then, selected specific years(recent 4 years) for further analysis.

```
attendees_date = attendees.groupby(['Year','Month','Year&Month']).mean().reset_index()
attendees_date
```

```
analysis_2019 = attendees_date.query("Year == '2019'")
analysis_2018 = attendees_date.query("Year == '2018'")
analysis_2017 = attendees_date.query("Year == '2017'")
analysis_2016 = attendees_date.query("Year == '2016'")
```

Last but not least, we drew time series plots to analyze the trend of percentage of first-time attendees, number of first-time attendees, percentage of major prospects, and number of major prospects.

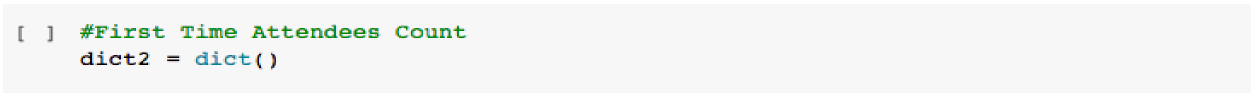
```
fig = plt.figure(figsize = (60,30),dpi = 300)


plt.plot(attendees_date['Year&Month'], attendees_date['Percentage First Time Attendees'])
plt.legend()
plt.xticks(fontsize=30, rotation=90)
plt.yticks(fontsize=30, rotation=90)
plt.xlabel('Year/Month', fontsize=50, color='red')
plt.ylabel('Percentage First Time Attendees', fontsize=40, color='red')
plt.title('Time Series', fontsize=70)
```

```
fig = plt.figure(figsize = (30,30),dpi = 200)

plt.plot(analysis_2019['Month'], analysis_2019['First Time Attendees'], 'ro-', label = '2019')
plt.plot(analysis_2018['Month'], analysis_2018['First Time Attendees'], 'go-', label = '2018')
plt.legend()
plt.xlabel('Month', fontsize=20, color='red')
plt.ylabel('First Time Attendees', fontsize=20, color='red')
plt.xticks(fontsize=20)
plt.yticks(fontsize=20)
plt.plot(analysis_2017['Month'], analysis_2017['First Time Attendees'], 'bo-', label = '2017')
plt.legend()

plt.plot(analysis_2016['Month'], analysis_2016['First Time Attendees'], 'co-', label = '2016')
plt.legend()
plt.title('Number of First Time Attendees vs Month', fontsize=20)
```

```
fig = plt.figure(figsize = (30,30),dpi = 200)
plt.subplot(411)
plt.plot(analysis_2019['Month'], analysis_2019['Percentage First Time Attendees'], 'ro-', label = '2019')
plt.plot(analysis_2018['Month'], analysis_2018['Percentage First Time Attendees'], 'go-', label = '2018')
plt.legend()
plt.xlabel('Month', fontsize=20, color='red')
plt.ylabel('Percentage First Time Attendees', fontsize=20, color='red')
plt.xticks(fontsize=20)
plt.yticks(fontsize=20)
plt.plot(analysis_2017['Month'], analysis_2017['Percentage First Time Attendees'], 'bo-', label = '2017')
plt.legend()

plt.plot(analysis_2016['Month'], analysis_2016['Percentage First Time Attendees'], 'co-', label = '2016')
plt.legend()

plt.title('Percentage First Time Attendees vs Month', fontsize=20)
```

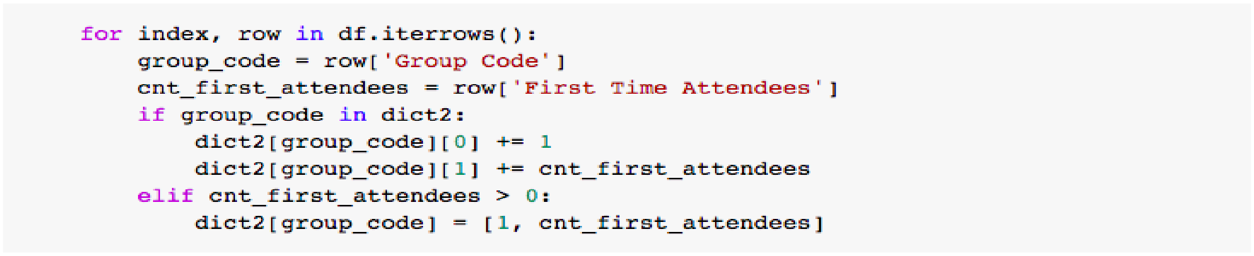
3. **Group Code:**

To analyze which event codes had the highest average number of first time attendees, we created a dictionary using a for loop.



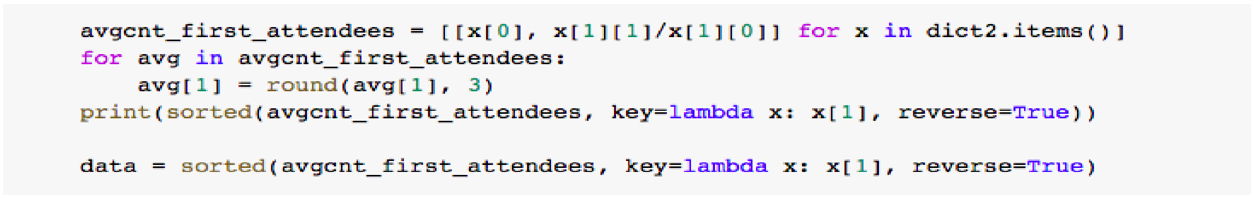
```
[ ]  #First Time Attendees Count
     dict2 = dict()
```

This for loop ran an iteration on each row, and, if the event with a certain group code had at least one first time attendee, the value would be summed together with others belonging to the same group code (a). Simultaneously, the loop was keeping a count for the number of times this group code had at least one first time attendee (b). If the group code was not yet in the dict, using a elif statement, it would then be added and it's number of first time attendees would be counted.



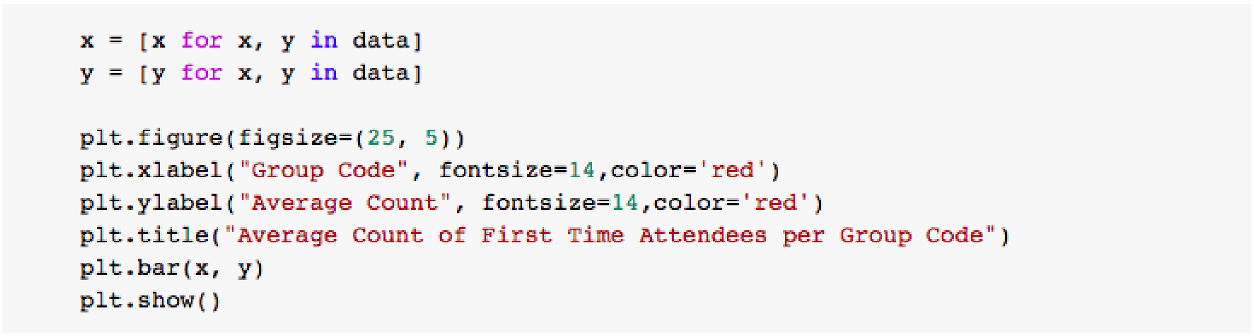
```
for index, row in df.iterrows():
    group_code = row['Group Code']
    cnt_first_attendees = row['First Time Attendees']
    if group_code in dict2:
        dict2[group_code][0] += 1
        dict2[group_code][1] += cnt_first_attendees
    elif cnt_first_attendees > 0:
        dict2[group_code] = [1, cnt_first_attendees]
```

Once the loop was finished and the dictionary complete, we calculated an average by dividing (a) over (b). Since we used a dictionary of dictionaries to return these values, we had to use x[0] and x[1] to compute the average.



```
avgcnt_first_attendees = [[x[0], x[1][1]/x[1][0]] for x in dict2.items()]
for avg in avgcnt_first_attendees:
    avg[1] = round(avg[1], 3)
print(sorted(avgcnt_first_attendees, key=lambda x: x[1], reverse=True))

data = sorted(avgcnt_first_attendees, key=lambda x: x[1], reverse=True)
```

We rounded the data to three data places to make it easier to read and then sorted the data in descending order using "reverse=True."
The same process was repeated for percentage of first time attendees, number of major prospects, and percentage of major prospects.
Using a bar graph, we were able to visualize these results to see which group codes attracted the highest number/percentage of attendees for these two groups using matplotlib. By creating two arrays for the x and y values, we were then able to plot the bar graph.



```
x = [x for x, y in data]
y = [y for x, y in data]

plt.figure(figsize=(25, 5))
plt.xlabel("Group Code", fontsize=14,color='red')
plt.ylabel("Average Count", fontsize=14,color='red')
plt.title("Average Count of First Time Attendees per Group Code")
plt.bar(x, y)
plt.show()
```

Using group code, we also wanted to group the codes into meaningful clusters, and see if we could identify which group codes were significant in relationship to percentage of first time attendees and percentage of major prospects. To do this, we clustered the data using k-means. To start, each group code needed to be categorized.

```
#5-Means Clustering - Percentage First Time Attendees

df['GroupCode_dum'] = df['Group Code'].astype('category').cat.codes
```

Next, using numpy, we created an array using the categorized group codes and percentage of first time attendees. We used percentage instead of count since each group code didn't have the same number of attendees and just using a count wouldn't be meaningful when comparing clusters. After trial and error, we decided on 5 clusters to get an even distribution.

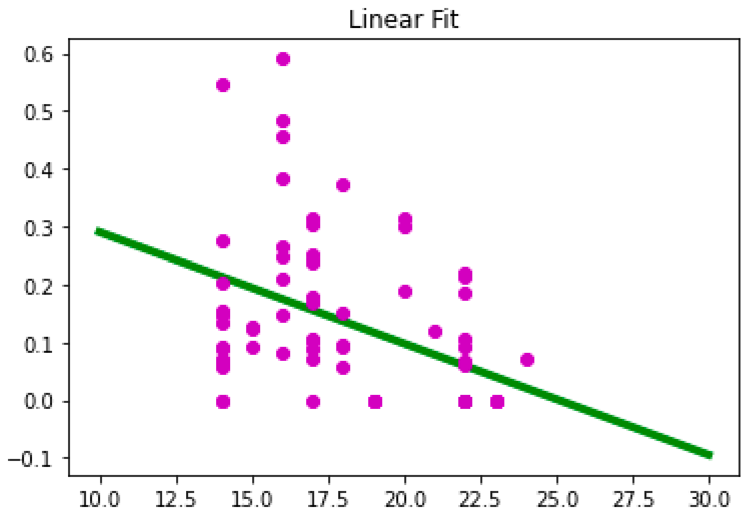
```
data = np.array(df[["GroupCode_dum", "Percentage First Time Attendees"]])

# compute k-means with k = 5 (5 clusters)
centroids,_ = kmeans(data, 5)
```

We then assigned each sample/category of group codes to a cluster and plotted the five clusters using different colors.

```
data = np.array(df[["GroupCode_dum", "Percentage First Time Attendees"]])

# compute k-means with k = 5 (5 clusters)
centroids,_ = kmeans(data, 5)

# assign each sample to a cluster
index,_ = vq(data,centroids)
from matplotlib.pyplot import plot, show

# plot different color for each cluster by its index
plot(data[index==0,0],data[index==0,1],'or')
plot(data[index==1,0],data[index==1,1],'ob')
plot(data[index==2,0],data[index==2,1],'oc')
plot(data[index==3,0],data[index==3,1],'oy')
plot(data[index==4,0],data[index==4,1],'ok')
plot(centroids[:,0],centroids[:,1],'sg',markersize=8)
show()
```

Once clustered, we wanted to run a linear regression analysis on all 5 clusters to see if any had significantly low (<0.05) p-values. We also plotted the data to see if any clustered data points followed the line of best fit.

```
# linear fit cluster 1
linSlope, linIntercept, linR, linP, linSE = linregress (data[index==0,0], data[index==0,1])
print(linIntercept, linSlope)
att, var = curve_fit(lambda dem, itc, slp: itc + slp * dem, data[index==0,0], data[index==0,1])
print(att)

dem = np.linspace(0, 50)
# plot linear fit
plt.title("Linear Fit")
plot(dem, linIntercept + linSlope * dem, '-g', linewidth=4)
plot(data[index==0,0], data[index==0,1], 'om')
# display figure
figure= plt.figure()
figure.tight_layout()
plt.show()
```



Following this, we returned the p-value for each cluster.

```
linSlope0, linIntercept0, linR0, linP0, linSE0 = linregress (data[index==0,0], data[index==0,1])
print(linP0)
linSlope1, linIntercept1, linR1, linP1, linSE1 = linregress (data[index==1,0], data[index==1,1])
print(linP1)
linSlope2, linIntercept2, linR2, linP2, linSE2 = linregress (data[index==2,0], data[index==2,1])
print(linP2)
linSlope3, linIntercept3, linR3, linP3, linSE3 = linregress (data[index==3,0], data[index==3,1])
print(linP3)
linSlope4, linIntercept4, linR4, linP4, linSE4 = linregress (data[index==4,0], data[index==4,1])
print(linP4)
```

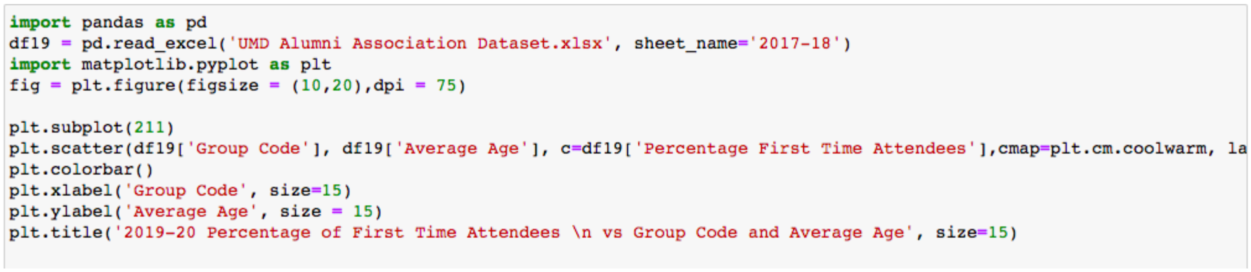
After figuring out which cluster had a p-value below 0.05, we returned all data points in that cluster and exported the dummies into a new excel sheet. This allowed us to filter the codes based on our results, and we could analyze the group codes to see if there were any similarities between the descriptions.

```
print(data[index==1,:])
```

```
df.to_excel('FullData.xlsx')
```

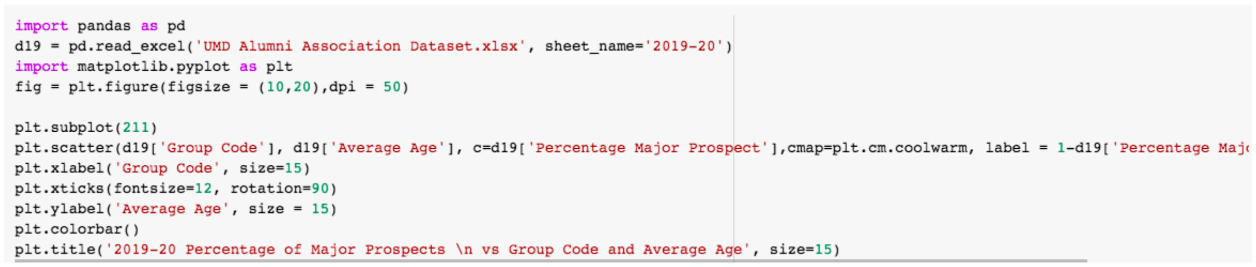
The same steps were repeated for the percentages of major prospects.

We also ran a scatterplot on multiple variables to see if the findings were consistent. For this analysis, we split the excel sheets into their individual years so that the group codes could be easily read and ran it on the three most recent years. We wanted to see if there were any interactions between multiple various, and/or if any conclusions remained the same. We chose average age, group code, and percentage of first time attendees to see if there was any trend. From our results, we found that some group codes had a wide range of ages for first time attendees, while others seemed to have a more clustered or homogeneous age range. This could be useful to the alumni association to figure out which demographics, specifically age ranges, are attending these events as first time attendees.



```
import pandas as pd
df19 = pd.read_excel('UMD Alumni Association Dataset.xlsx', sheet_name='2017-18')
import matplotlib.pyplot as plt
fig = plt.figure(figsize = (10,20),dpi = 75)

plt.subplot(211)
plt.scatter(df19['Group Code'], df19['Average Age'], c=df19['Percentage First Time Attendees'],cmap=plt.cm.coolwarm, la
plt.colorbar()
plt.xlabel('Group Code', size=15)
plt.ylabel('Average Age', size = 15)
plt.title('2019-20 Percentage of First Time Attendees \n vs Group Code and Average Age', size=15)
```

We repeated this same code for the percentage of major prospects across the same two variables.



```
import pandas as pd
d19 = pd.read_excel('UMD Alumni Association Dataset.xlsx', sheet_name='2019-20')
import matplotlib.pyplot as plt
fig = plt.figure(figsize = (10,20),dpi = 50)

plt.subplot(211)
plt.scatter(d19['Group Code'], d19['Average Age'], c=d19['Percentage Major Prospect'],cmap=plt.cm.coolwarm, label = 1-d19['Percentage Maj
plt.xlabel('Group Code', size=15)
plt.xticks(fontsize=12, rotation=90)
plt.ylabel('Average Age', size = 15)
plt.colorbar()
plt.title('2019-20 Percentage of Major Prospects \n vs Group Code and Average Age', size=15)
```

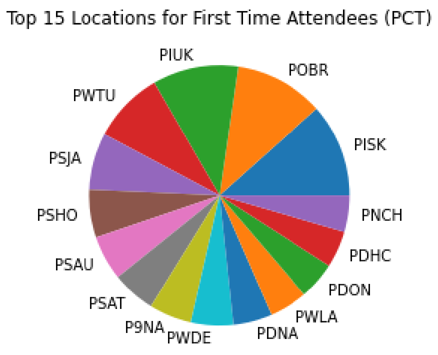
4. Location:

<u>For First Time Attendees</u>

First, we take the average of Percentage First Time Attendees (FTA) to see the Events with highest Percentage (FTA) as we thought using the percentage would be the best option to see the impact of Locations for us.

```
#Top 15 Locations for First Time Attendees (percentage)
df2 = df.groupby('Location Code')['Percentage First Time Attendees'].mean().sort_values(ascending = False)
top_15_FT_pct = df2[:20].copy()
labels_FT_pct = top_15_FT_pct.sort_values(ascending = False).index

plt.pie(top_15_FT_pct, labels = labels_FT_pct)
plt.title("Top 15 Locations for First Time Attendees (PCT)")

top_15_FT_pct
```

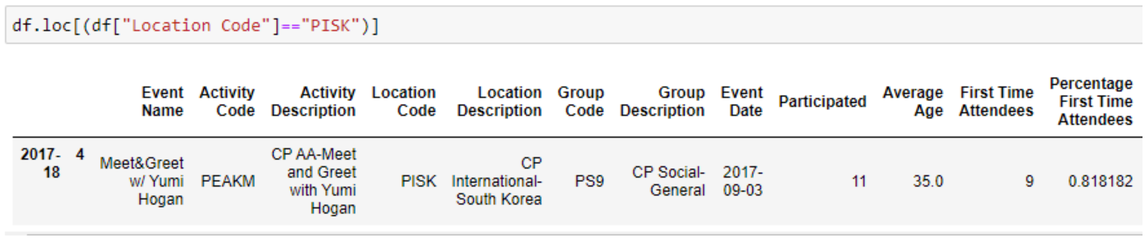
```
Location Code
PISK    0.818182
POBR    0.782609
PIUK    0.750000
PWTU    0.625000
PSJA    0.500000
PSHO    0.416667
PSAU    0.392308
PSAT    0.381905
P9NA    0.371032
PWDE    0.370318
PDNA    0.340746
PWLA    0.333915
PDON    0.328679
PDHC    0.320576
PNCH    0.318306
Name: Percentage First Time Attendees, dtype: float64
```



However, we then realized that in the dataframe, there can be many factors that can skew the insights we draw if we take them into account. For example: Events with high Percentage of First Time Attendees but were only hosted once in the past 6 years and the total number of Attendees for those events is not significant (e.g.: PISK, PIUK, POBR,...)

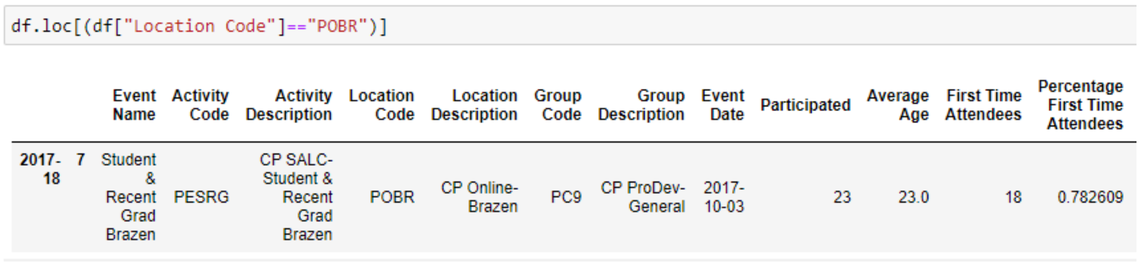
```
df.loc[(df["Location Code"]=="PIUK")]
```

		Event Name	Activity Code	Activity Description	Location Code	Location Description	Group Code	Group Description	Event Date	Participated	Average Age	First Time Attendees	Percentage First Time Attendees
2017-18	**17**	London Happy Hour	PEALN	CP AA-London Happy Hour	PIUK	CP International-UK	PS9	CP Social-General	2017-11-29	8	27.0	6	0.75



```
df.loc[(df["Location Code"]=="PISK")]
```

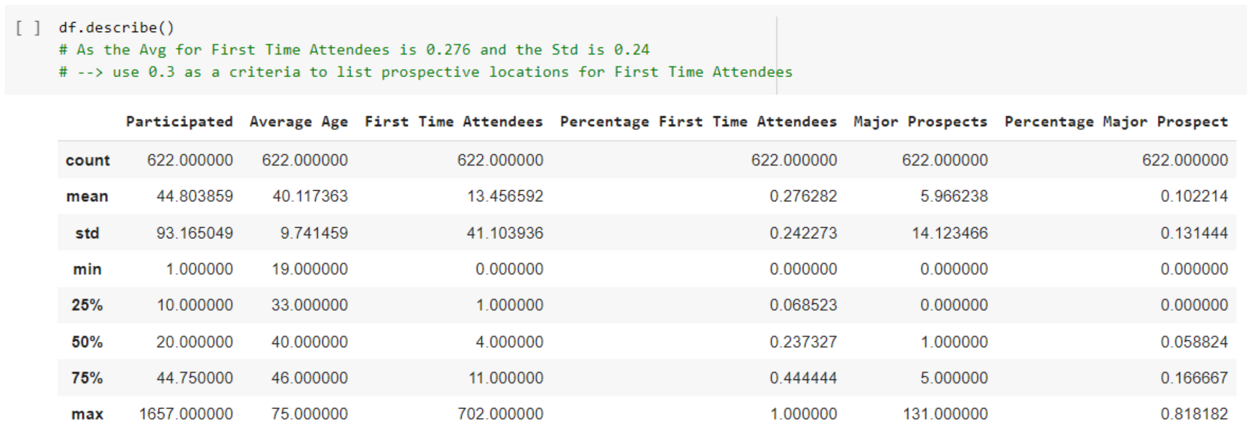
		Event Name	Activity Code	Activity Description	Location Code	Location Description	Group Code	Group Description	Event Date	Participated	Average Age	First Time Attendees	Percentage First Time Attendees
2017-18	4	Meet&Greet w/ Yumi Hogan	PEAKM	CP AA-Meet and Greet with Yumi Hogan	PISK	CP International-South Korea	PS9	CP Social-General	2017-09-03	11	35.0	9	0.818182



```
df.loc[(df["Location Code"]=="POBR")]
```

		Event Name	Activity Code	Activity Description	Location Code	Location Description	Group Code	Group Description	Event Date	Participated	Average Age	First Time Attendees	Percentage First Time Attendees
2017-18	7	Student & Recent Grad Brazen	PESRG	CP SALC-Student & Recent Grad Brazen	POBR	CP Online-Brazen	PC9	CP ProDev-General	2017-10-03	23	23.0	18	0.782609

But then if we use the number of FTA to draw insights, we will have to face situations in which those locations have events with high numbers of FTA, but low in Percentage of FTA. Therefore, in order to solve these aforementioned problems and effectively use the Location variable to see how events hosted in different locations affect the number of First Time Attendees and Major Prospects, we first looked at the description for the whole dataframe.

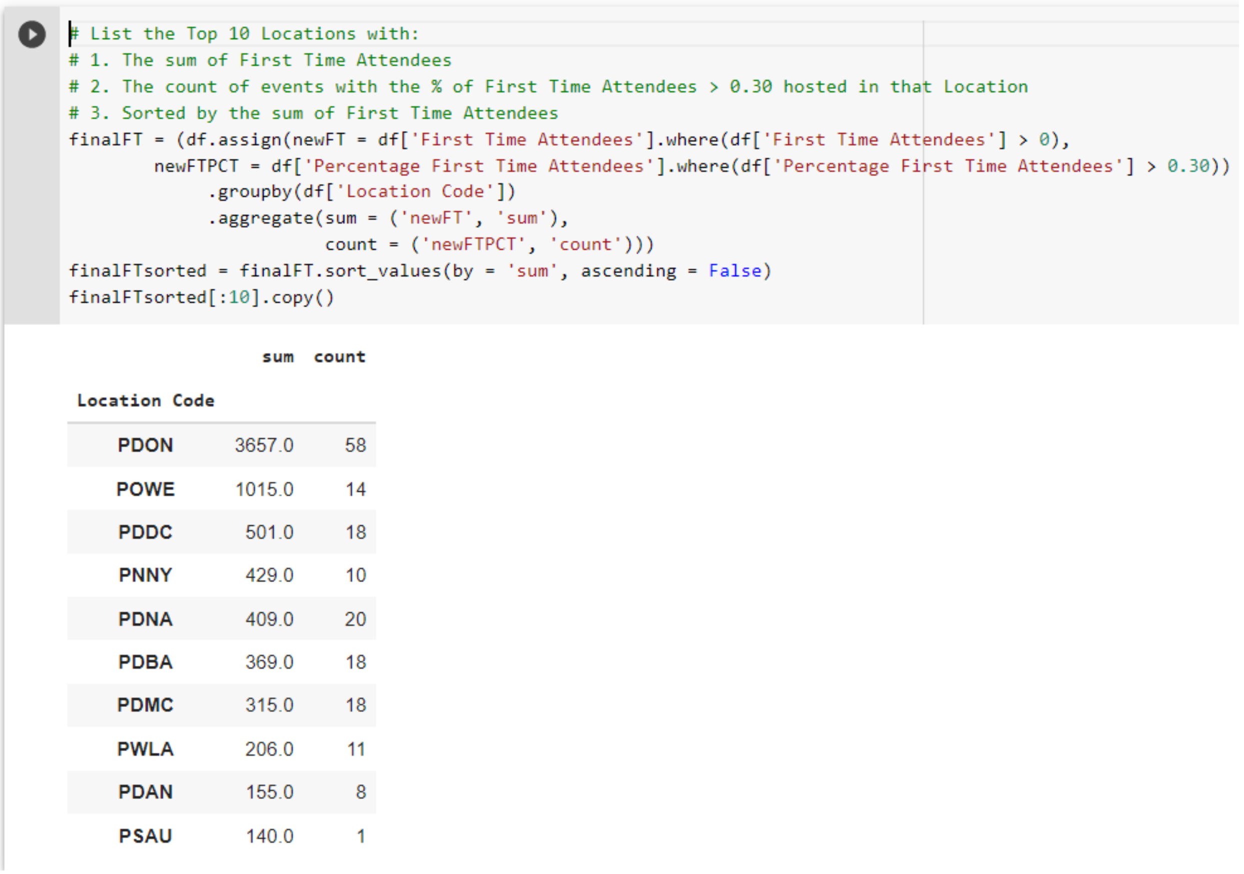


```
[ ]  df.describe()
     # As the Avg for First Time Attendees is 0.276 and the Std is 0.24
     # --> use 0.3 as a criteria to list prospective locations for First Time Attendees
```

	Participated	Average Age	First Time Attendees	Percentage First Time Attendees	Major Prospects	Percentage Major Prospect
count	622.000000	622.000000	622.000000	622.000000	622.000000	622.000000
mean	44.803859	40.117363	13.456592	0.276282	5.966238	0.102214
std	93.165049	9.741459	41.103936	0.242273	14.123466	0.131444
min	1.000000	19.000000	0.000000	0.000000	0.000000	0.000000
25%	10.000000	33.000000	1.000000	0.068523	0.000000	0.000000
50%	20.000000	40.000000	4.000000	0.237327	1.000000	0.058824
75%	44.750000	46.000000	11.000000	0.444444	5.000000	0.166667
max	1657.000000	75.000000	702.000000	1.000000	131.000000	0.818182

Then we identified that the Avg for Percentage First Time Attendees is 0.276, meaning for all the events in the dataframe, on average, 27.6% of the attendees were First Time Attendees with the standard deviation of 0.24. With the information, we decided to use 30% as our criteria to filter out events in order to list prospective locations for First Time Attendees.
We then used df.assign() and df.groupby().aggregate() to apply these filters and came down to a list of Locations with the Sum of the FTA and Count of the Events hosted in those Locations with % of FTA larger than 30%.
Using the list, we sorted by Sum instead of Count in order to filter out the Locations that might host many events but have a low number of total attendees. We picked the Top 10 from the list

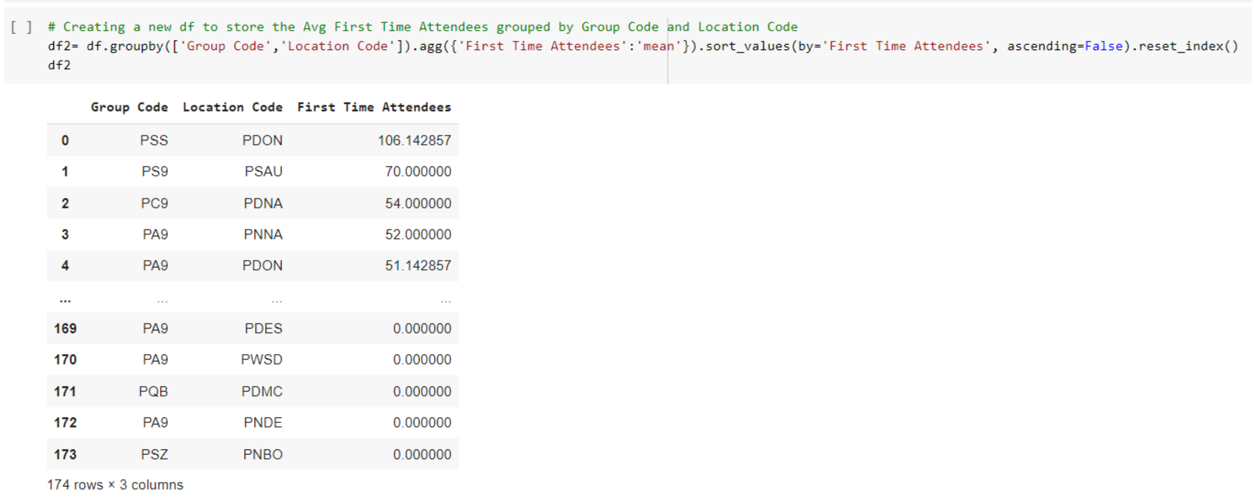
and below is the result:



```
# List the Top 10 Locations with:
# 1. The sum of First Time Attendees
# 2. The count of events with the % of First Time Attendees > 0.30 hosted in that Location
# 3. Sorted by the sum of First Time Attendees
finalFT = (df.assign(newFT = df['First Time Attendees'].where(df['First Time Attendees'] > 0),
        newFTPCT = df['Percentage First Time Attendees'].where(df['Percentage First Time Attendees'] > 0.30))
             .groupby(df['Location Code'])
             .aggregate(sum = ('newFT', 'sum'),
                        count = ('newFTPCT', 'count')))
finalFTsorted = finalFT.sort_values(by = 'sum', ascending = False)
finalFTsorted[:10].copy()
```

	sum	count
Location Code		
PDON	3657.0	58
POWE	1015.0	14
PDDC	501.0	18
PNNY	429.0	10
PDNA	409.0	20
PDBA	369.0	18
PDMC	315.0	18
PWLA	206.0	11
PDAN	155.0	8
PSAU	140.0	1

The next steps are what we used to prepare the data for our plot to show the data we have in a more interactive way.
First, we created a new dataframe to store the information of Group Code, Location Code, and Average of First Time Attendees.



```
[ ] # Creating a new df to store the Avg First Time Attendees grouped by Group Code and Location Code
    df2= df.groupby(['Group Code','Location Code']).agg({'First Time Attendees':'mean'}).sort_values(by='First Time Attendees', ascending=False).reset_index()
    df2
```

	Group Code	Location Code	First Time Attendees
0	PSS	PDON	106.142857
1	PS9	PSAU	70.000000
2	PC9	PDNA	54.000000
3	PA9	PNNA	52.000000
4	PA9	PDON	51.142857
...	...	...	...
169	PA9	PDES	0.000000
170	PA9	PWSD	0.000000
171	PQB	PDMC	0.000000
172	PA9	PNDE	0.000000
173	PSZ	PNBO	0.000000

174 rows × 3 columns

Then, we assigned data associated with each Location above into a new df and combined (concat) those df into another df to show all the events hosted in that location along with their Avg First Time Attendees. We also filtered out the events hosted in that location with no First

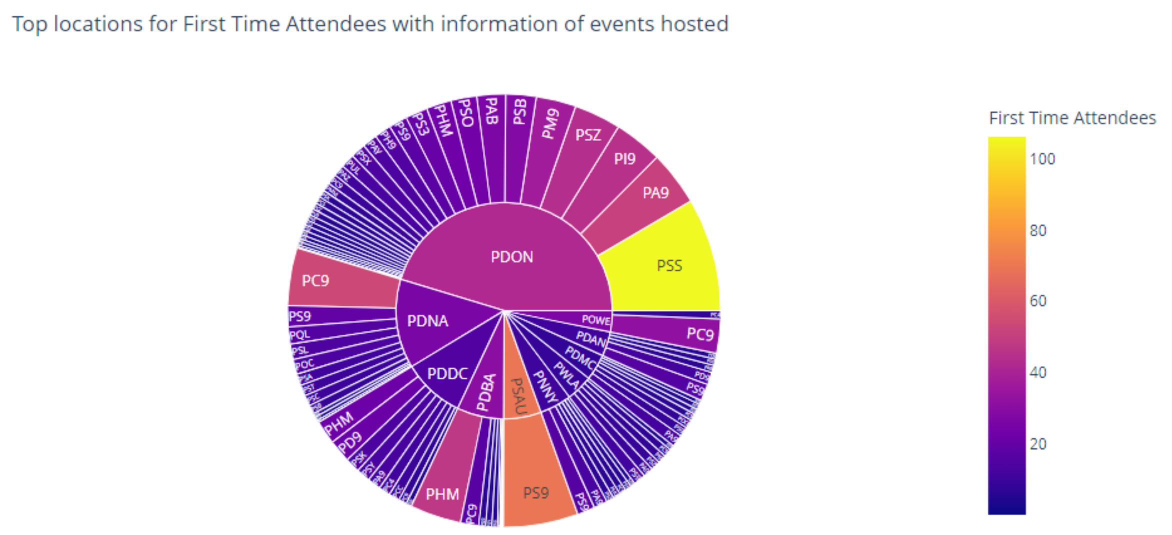
Time Attendees.

```
# Assigning data associated with each Location above into new df and combine (concat) those df into another df
# to show all the events hosted in that location along with their Avg First Time Attendees
# Filter out the events hosted in that location with no First Time Attendees
df_11 = df2.loc[(df2["Location Code"]=="PDON")]
df_12 = df2.loc[(df2["Location Code"]=="POWE")]
df_13 = df2.loc[(df2["Location Code"]=="PDDC")]
df_14 = df2.loc[(df2["Location Code"]=="PNNY")]
df_15 = df2.loc[(df2["Location Code"]=="PDNA")]
df_16 = df2.loc[(df2["Location Code"]=="PDBA")]
df_17 = df2.loc[(df2["Location Code"]=="PDMC")]
df_18 = df2.loc[(df2["Location Code"]=="PWLA")]
df_19 = df2.loc[(df2["Location Code"]=="PDAN")]
df_20 = df2.loc[(df2["Location Code"]=="PSAU")]
df_row_FT = pd.concat([df_11, df_12, df_13, df_14, df_15, df_16, df_17, df_18, df_19, df_20], axis = 0)
print(df_row_FT)
df_plot_FT = df_row_FT[df_row_FT['First Time Attendees']!=0]
```

```
   Group Code Location Code  First Time Attendees
0         PSS          PDON            106.142857
4         PA9          PDON             51.142857
6         PI9          PDON             45.333333
7         PSZ          PDON             44.000000
8         PM9          PDON             37.333333
..        ...           ...                   ...
45        PD9          PDAN             11.666667
80        PA9          PDAN              6.666667
90        PSY          PDAN              6.000000
95        PO9          PDAN              5.500000
1         PS9          PSAU             70.000000

[91 rows x 3 columns]
```

Lastly, we imported plotly.express as px and plotted our data using sunburst format.

```
import plotly.express as px
fig = px.sunburst(df_plot_FT, path=['Location Code', 'Group Code'], values='First Time Attendees',
                  title = "Top locations for First Time Attendees with information of events hosted",
                  color='First Time Attendees')
fig.show()
```



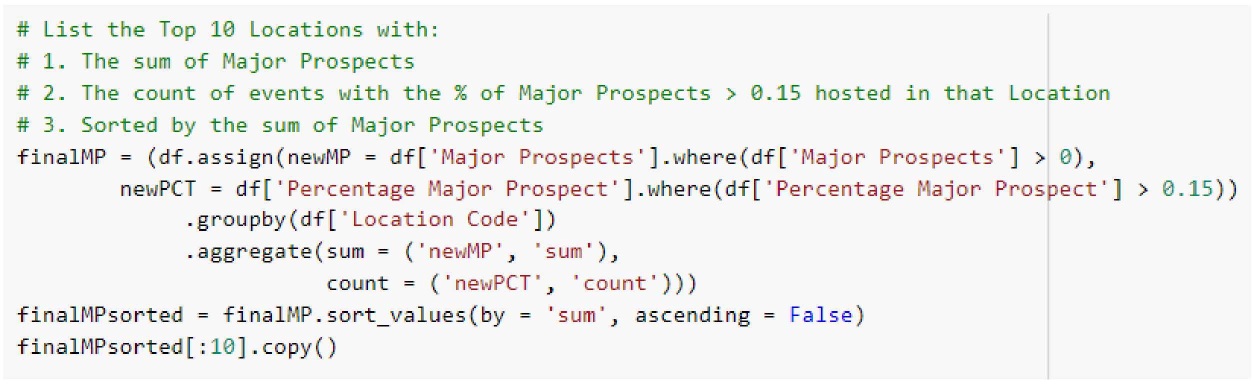
*(*Notes: The visualization above is screenshot from Jupyter, not colab. We realized that with the same code, Jupyter used the average FTA for Group Code as the bench mark and provided a much clearer visualization.)*

<u>For Major Prospects</u>

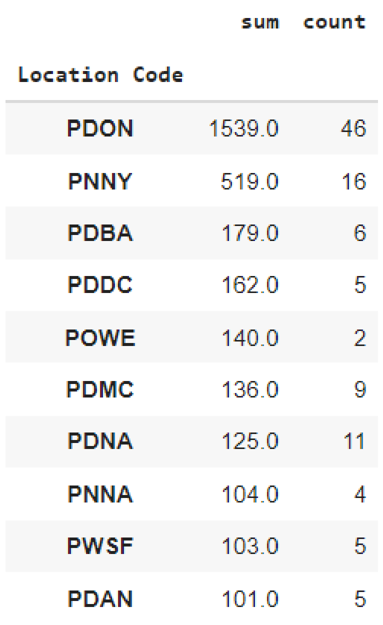
With Major Prospects, we went through the same process and also encountered a similar pattern in data.

Therefore, after identifying that the average Percentage of Major Prospects for each event is 10.22% with a standard deviation of 13.14%, we decided to use 15% as a criteria to list prospective locations for Major Prospects.

We then did the same steps we did when analyzing the relationship between Locations and FTA, and we got the following code and result:



```
# List the Top 10 Locations with:
# 1. The sum of Major Prospects
# 2. The count of events with the % of Major Prospects > 0.15 hosted in that Location
# 3. Sorted by the sum of Major Prospects
finalMP = (df.assign(newMP = df['Major Prospects'].where(df['Major Prospects'] > 0),
        newPCT = df['Percentage Major Prospect'].where(df['Percentage Major Prospect'] > 0.15))
             .groupby(df['Location Code'])
             .aggregate(sum = ('newMP', 'sum'),
                        count = ('newPCT', 'count')))
finalMPsorted = finalMP.sort_values(by = 'sum', ascending = False)
finalMPsorted[:10].copy()
```



Location Code	sum	count
PDON	1539.0	46
PNNY	519.0	16
PDBA	179.0	6
PDDC	162.0	5
POWE	140.0	2
PDMC	136.0	9
PDNA	125.0	11
PNNA	104.0	4
PWSF	103.0	5
PDAN	101.0	5

For this time, we got the column Count as the number of events that have more than 10% of Major Prospects for each location, after that we sorted the data by Sum and picked the Top 10 from the list.

With the list of 10 prospective locations above, we also used the Location Code to locate and get information about events hosted for each location.

```python
# Creating a new df to store the Avg Major Prospects grouped by Group Code and Location Code
df3= df.groupby(['Group Code','Location Code']).agg({'Major Prospects':'mean'}).sort_values(by='Major Prospects', ascending=False).reset_index()
df3
```

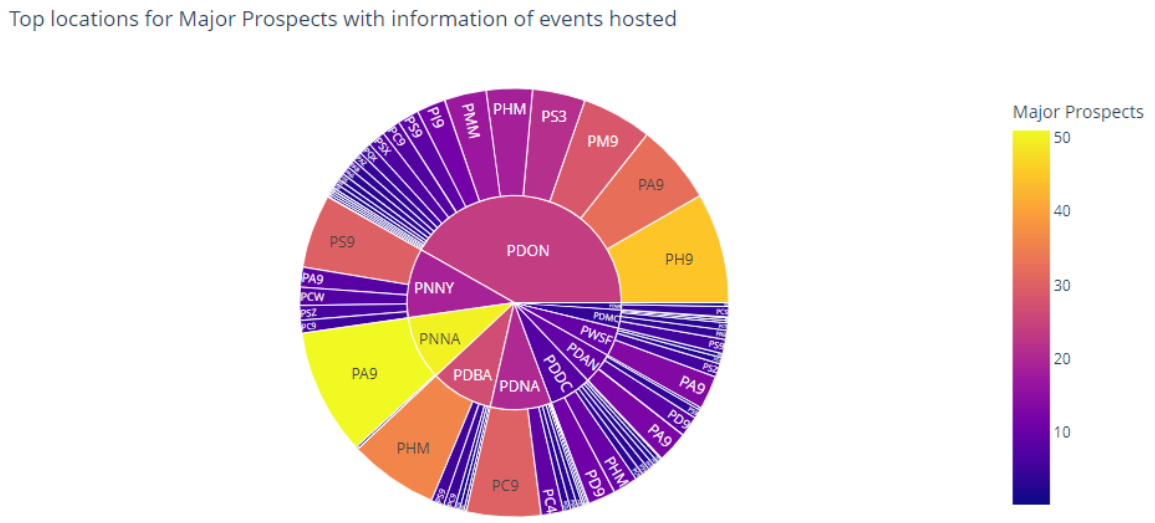
	Group Code	Location Code	Major Prospects
0	PA9	PNNA	51.000000
1	PH9	PDON	44.666667
2	PHM	PDBA	36.000000
3	PA9	PDON	32.428571
4	PC9	PDNA	30.000000
...	...	...	...
169	PSC	PDON	0.000000
170	PS9	PDWE	0.000000
171	PSK	PDON	0.000000
172	PSL	PDNA	0.000000
173	PO9	PDBA	0.000000

174 rows × 3 columns

```python
# Assigning data associated with each Location above into new df and combine (concat) those df into another df
# to show all the events hosted in that location along with their Avg Major Prospects
# Filter out the events hosted in that location with no Major Prospects
df_1 = df3.loc[(df3["Location Code"]=="PDON")]
df_2 = df3.loc[(df3["Location Code"]=="PNNY")]
df_3 = df3.loc[(df3["Location Code"]=="PDBA")]
df_4 = df3.loc[(df3["Location Code"]=="PDDC")]
df_5 = df3.loc[(df3["Location Code"]=="POWE")]
df_6 = df3.loc[(df3["Location Code"]=="PDMC")]
df_7 = df3.loc[(df3["Location Code"]=="PDNA")]
df_8 = df3.loc[(df3["Location Code"]=="PNNA")]
df_9 = df3.loc[(df3["Location Code"]=="PWSF")]
df_10 = df3.loc[(df3["Location Code"]=="PDAN")]
df_row = pd.concat([df_1, df_2, df_3, df_4, df_5, df_6, df_7, df_8, df_9, df_10], axis = 0)
print(df_row)
df_plot = df_row[df_row['Major Prospects']!=0]
```

```
    Group Code Location Code  Major Prospects
1          PH9          PDON        44.666667
3          PA9          PDON        32.428571
6          PM9          PDON        28.333333
8          PS3          PDON        21.400000
10         PHM          PDON        18.500000
..         ...           ...              ...
14         PA9          PDAN        12.666667
25         PD9          PDAN         8.333333
59         PS9          PDAN         3.666667
109        PO9          PDAN         1.000000
132        PSY          PDAN         0.000000

[91 rows x 3 columns]
```

```
# Plot Top locations for Major Prospects with information of events hosted
import plotly.express as px
fig = px.sunburst(df_plot, path=['Location Code', 'Group Code'], values='Major Prospects',
                  title = "Top locations for Major Prospects with information of events hosted", color='Major Prospects')
fig.show()
```



*(*Notes: The visualization above is screenshot from Jupyter, not colab. We realized that with the same code, Jupyter used the average FTA for Group Code as the bench mark and provided a much clearer visualization.)*

Findings

1. **Average Age:**

   a. <u>First Time Attendees</u>:

By dividing event attendees into groups by age, we found that younger people (aged 30 years old and below) have the highest percentage of first time attendees. Running this age group against group code and location code, we found that these individuals usually attend social and athletics events, primarily on campus (Appendix A). To get further insight by diving people into young and old people gorup based on age, we found out that both young peopl less than 30 years old and elders more than 50 years old usually first attend social and athletics events on campus.

   b. <u>Major Prospects</u>:

By dividing the data into different groups by age, we found older people (around 50 years old) have the highest percentage of major prospects. Running this age group against group code as well as location code, we see that these people usually attend social or athletics events on campus (Appendix B).

2. **Date:**

   a. <u>First Time Attendees</u>:

Looking at the data broken up into month/year, we can see a large variation in the percentage of first time attendees in relation to the date. Comparing various years on the same plot, we don't see any trend among percentage of first time attendees and month. This leads us to believe the time of year doesn't have a strong effect on the number of first time attendees (Appendix C).

b. Major Prospects:

After splitting the data by months and year, the line graph shows that the number of major prospects slightly declines over time. While there was a large spike in attendance from August-October in 2016, there does not appear to be any trend between the variables (Appendix D).

3. **Group Code:**

a. First Time Attendees:

Looking at the average number of first time attendees, we found that individuals are overwhelmingly interested in attending event code PSS (CP Social- Students). While this was a bit confusing at first, considering Alumni are no longer students, we found two activities (Grad Bash and Ring Ceremony) take place annually and attract a high number of first time attendees. Looking at which event codes had the highest rates of first time attendees, POO (CP Service-Muslim), P99(CP Social-General), and PSO(CP Service- MAM) were the top three. This data was very interesting and not expected, as Muslim services appeard in two of the top three and are not what most people think of immediately when they hear 'Almumni event'. However, it makes sense that Alumni that belong to these subgroups were probably involved in similar groups or clubs during their time at the University, and are likely to return if they have built friendships and community by attending these events. If the Alumni Association would like to put on a specific event that will bring in a high percentage of first time attendees, more niche events targeted toward specific sub-groups of individuals (i.e. Muslim-oriented events) would be ideal (Appendix E).

b. Major Prospects:

Focusing our attention on major prospects, we found that these individuals are more likely to attend event codes PSZ (CP Social-StuAlum), PH9 (CP Stewardship-General), and P99 (CP Social-General). Specifically, a high number of major prospects enjoy attending award ceremonies/galas and social events focused on giving back to the community (Appendix F). This finding was logical, as major prospects donate their money to helping out students, faculty, and other UMD members and are often very involved with the Terrapin network. Similarly, because these individuals are more affluent and contribute to the donor fund, they most likely attend fancier events like galas and ceremonies to be recognized by the University.

4. **Location:**

a. First Time Attendees:

In general, events hosted **on campus in DMV** (PDON) attracted the most FTA, with a relatively high percentage of FTA of 32.9%. The events that are popular for this specific location are

Socials (PS-) and Athletic Events (PA-), with an average number of FTA for the Social Events for Students (PSS) is significantly high with more than 100 FTA for each event on average. Events hosted **around the DMV area** (PDNA - General, PDDC - Washington DC, PDBA - Baltimore) were also popular among FTA, with the % of FTA are 34%, 27.7%, 24.8% respectively. Aside from PSS, events for professional development in general (PC9) and events coded PHM (Stewardship-Membership) are also popular with attendees can be around 40-80 FTA for each event.
In the most recent year, due to Covid, total attendance and first time attendees decrease and only one fourth prefer going to in-person events on campus in CP DMV (PDON), while the percentage of first time attendees prefer joining online webinars increases. Events hosted in Montgomery County also attract a larger percentage of first time attendees in the most recent year.
(Appendix G)

b. Major Prospects:

In general, major prospects tend to join events hosted **on campus** in the CP DMV area (PDON). Events hosted in **Northeast** (PNNY-New York, PNNA-General) and events in **other area in DMV** (PDBA-Baltimore, PDNA-General, PDDC-Washington DC) as well as in the **West** are the next in terms of preference by Major Prospects.
The events that attract Major Prospects the most are Stewardship-Membership (PHM), Socials (PS-), and Athletic Events (PA9-General).
(Appendix H)

Accomplishments and Recommendations

From our analysis, we would like the Alumni Association to take into account the following items when planning their next events. It is our hope that by focusing on the recommended age groups, group codes, and locations, they will be able to increase their number of first time attendees and major prospects.

In order to attract more first time attendees, the Alumni Association should host professional development, social, and athletic events that take place on campus and virtual. First time attendees are younger, aged less than 30 years old, and are likely recent graduates of UMD.

- Professional development events help progress these individual's careers and/or allow them to network.
- Social and athletic events provide them with an opportunity to relive their college days and rekindle with old friends.

To increase major prospect attendance, the Alumni Association should host service/stewardship, athletic, and social events. Major prospects are older, mainly 50-55 years old, and attend events that take place on campus, in the DMV area, and states in the Northeast region, such as New York.

- Major prospects prefer events that can positively impact the community. Seeing as they are donating large sums of money to the University, these types of events align with their motives to support Terrapin students, staff, and faculty.
- Like first time attendees, major prospects enjoy athletic events on campus. They also attend social events in the DMV (DC, Baltimore), Northeast (New York) and Southeast (Florida) regions.

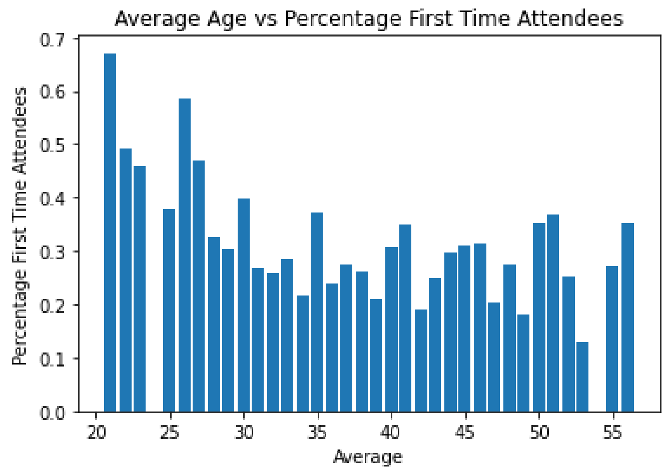
Future Work

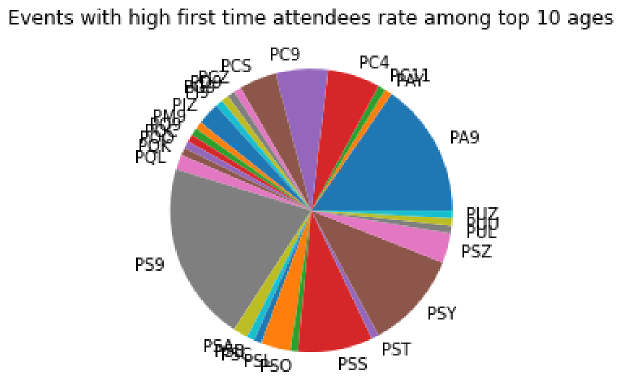
For the future, we believe the Alumni Association could increase their number of first time attendees and major prospects by tracking and including marketing strategies in their analysis. I, Haley, am a current alumnus of the University of Maryland having graduated with my Bachelors degree in May of 2021. I often receive emails from the Alumni Association and use their email newsletters to stay updated about what is happening on campus. I have also attended service events on campus because of their email marketing. This got our group questioning how these major prospects and first time attendees are hearing about alumni events. Is it through emails? Social media? Word of mouth? Obviously, alumni are most likely no longer on campus regularly and need to be contacted about these events through other methods.

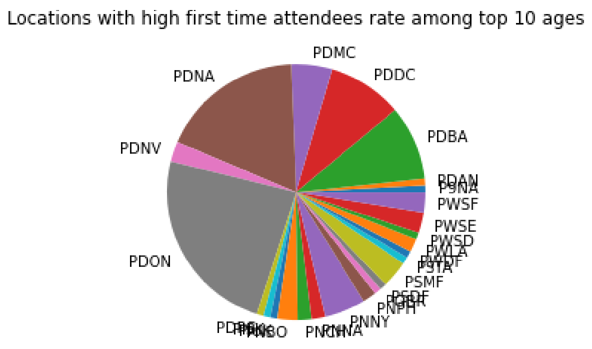
If we had known how individuals were being contacted, we could have recommended strategies to increase outreach and engagement. Additionally, because major prospects are important individuals at the University of Maryland, we could have tracked whether, say, a personalized invitation made the major prospects more inclined to attend these events. Advertising is a large part of event attendance so it would be nice to have this data in the future.

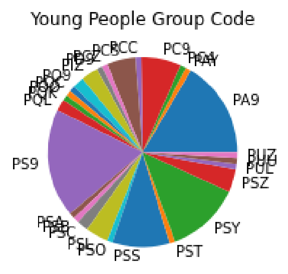
APPENDICES

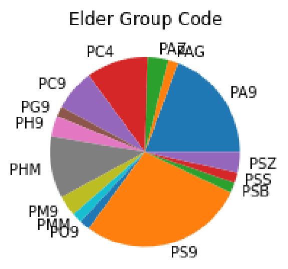
Appendix A

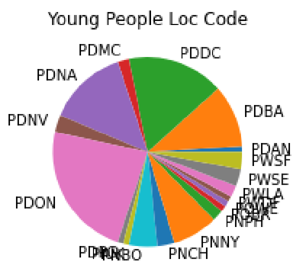


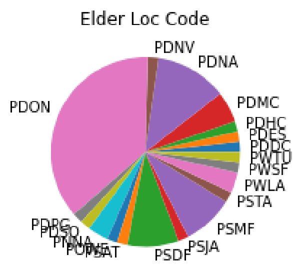




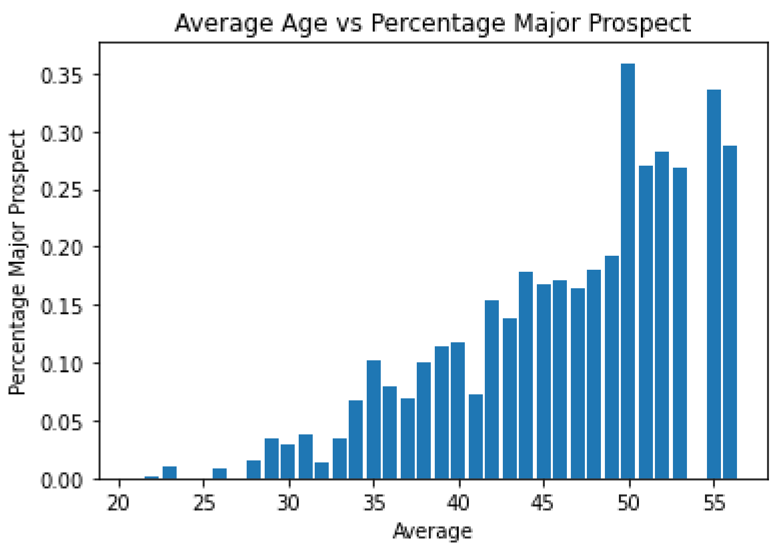


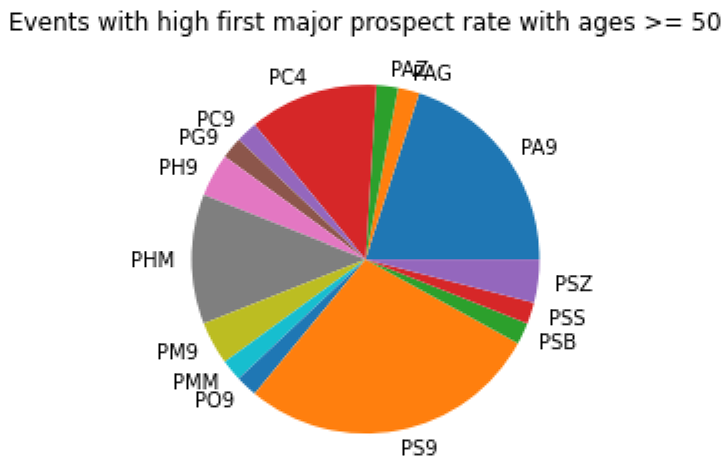


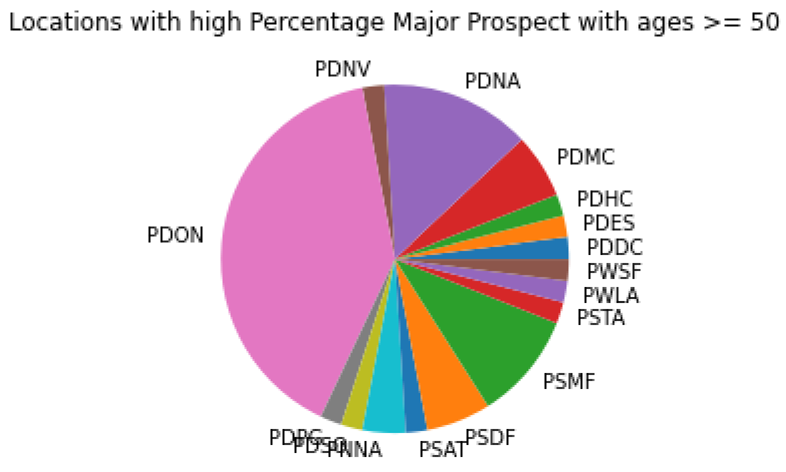




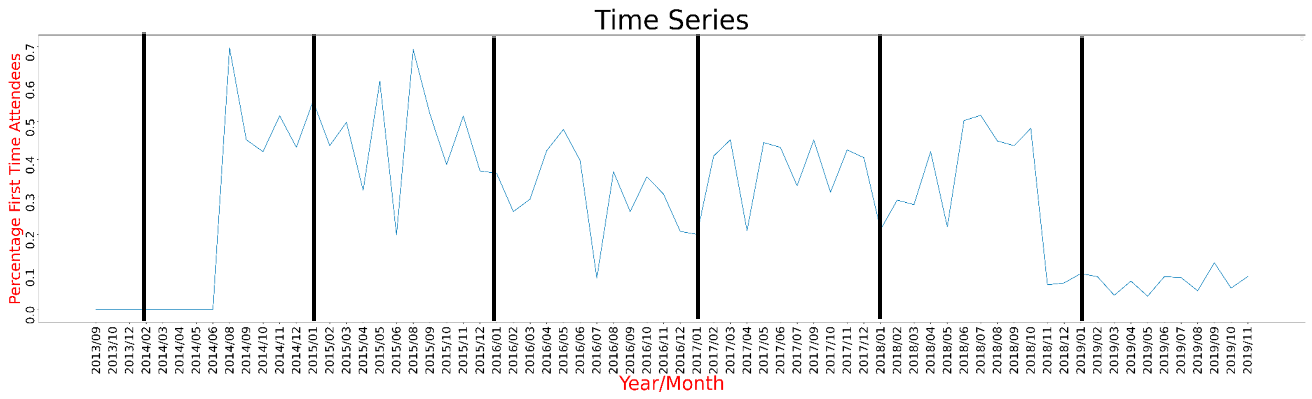
Appendix B


Average Age vs Percentage Major Prospect
Percentage Major Prospect
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
20
25
30
35
40
45
50
55
Average

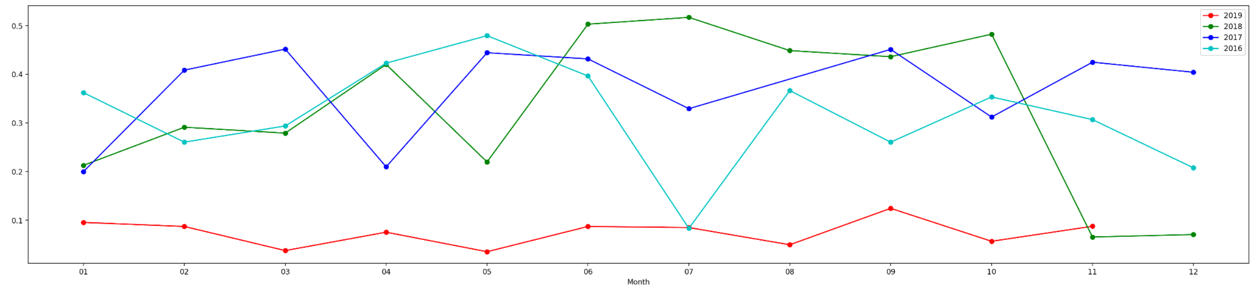

Events with high first major prospect rate with ages >= 50
PAZ
PAG
PC4
PC9
PG9
PH9
PA9
PHM
PSZ
PSS
PSB
PM9
PMM
PO9
PS9


Locations with high Percentage Major Prospect with ages >= 50
PDNV
PDNA
PDMC
PDHC
PDES
PDDC
PDON
PWSF
PWLA
PSTA
PSMF
PDPG
PDSO
PNNA
PSAT
PSDF

Appendix C

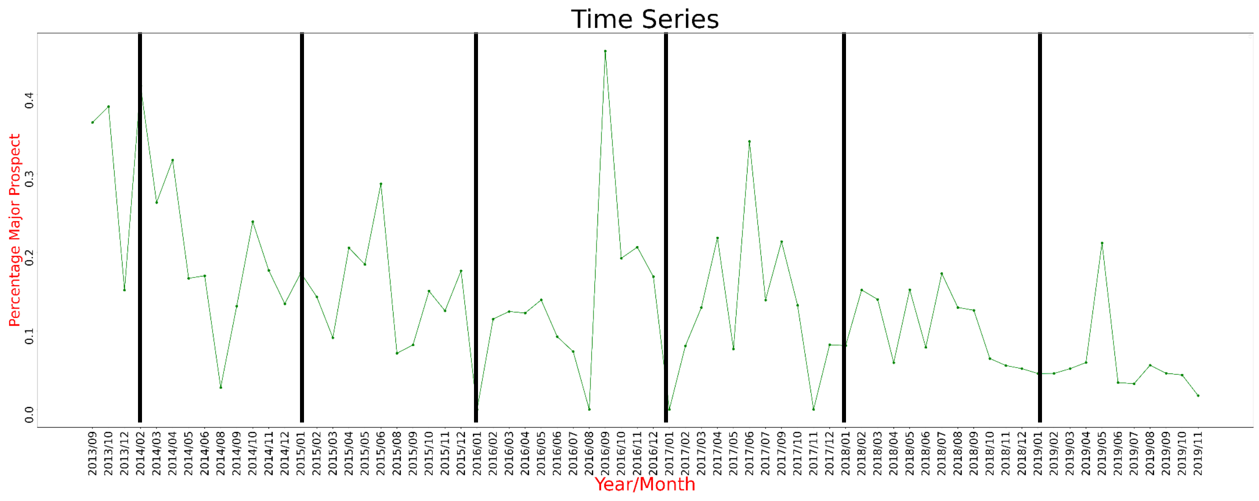


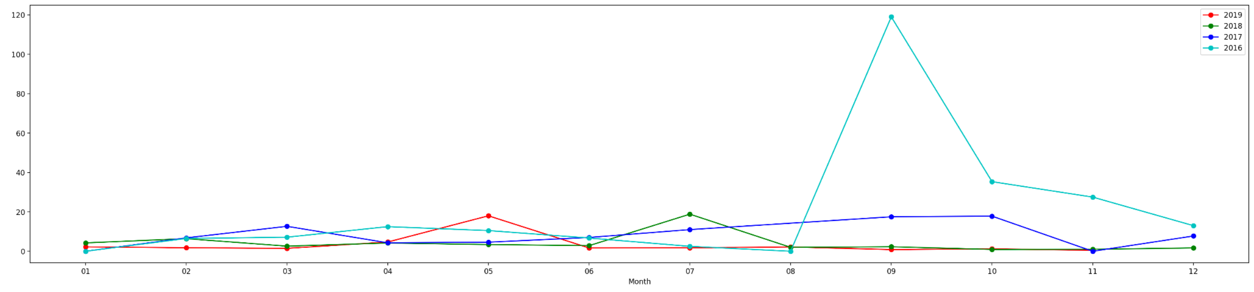
Time Series
Percentage First Time Attendees
Year/Month



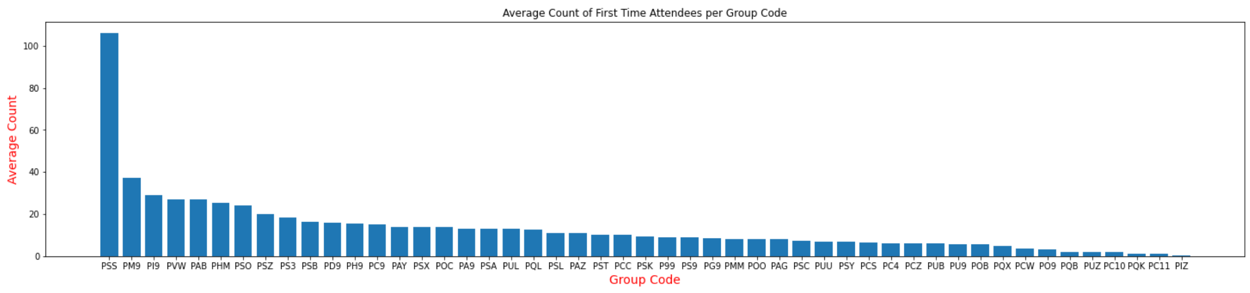
2019
2018
2017
2016
Month

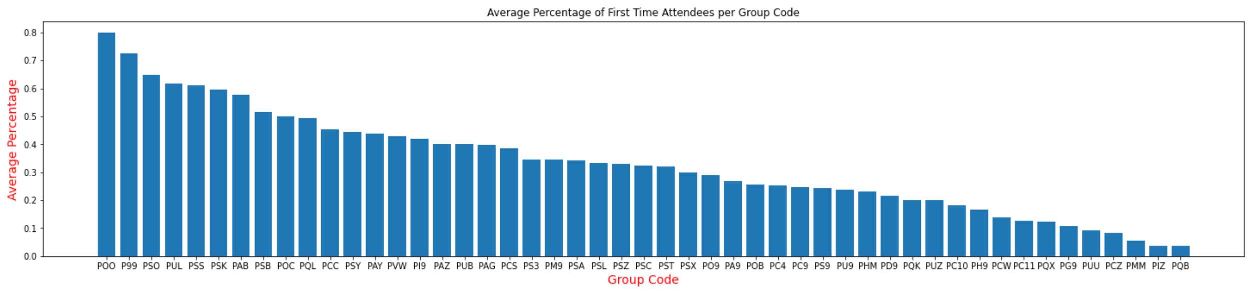
Appendix D

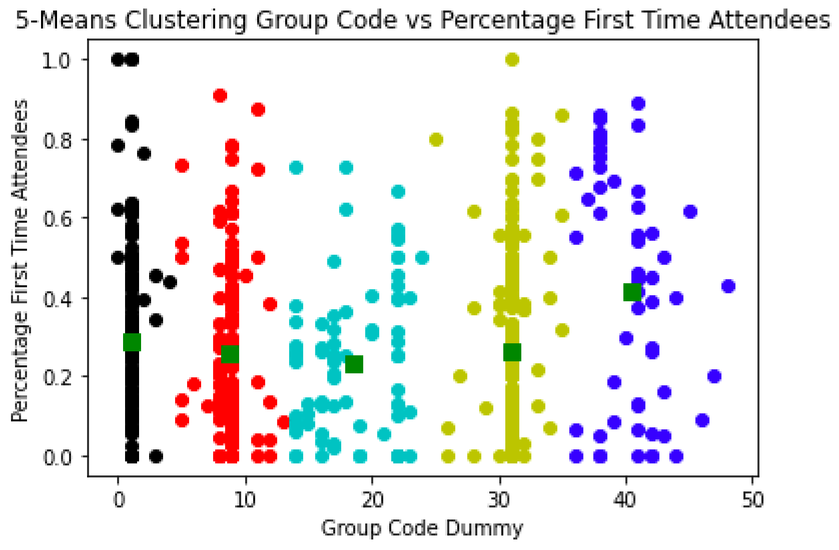

Time Series
Percentage Major Prospect
Year/Month
0.0
0.1
0.2
0.3
0.4

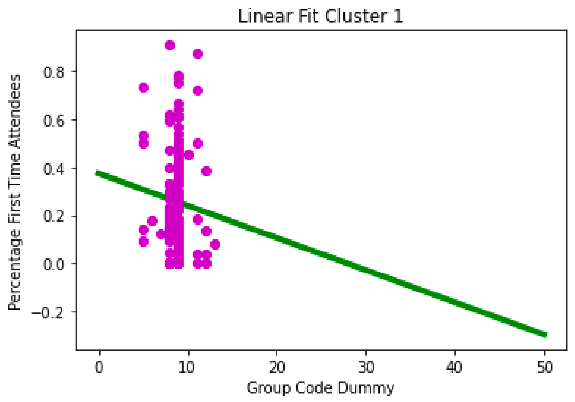

2019
2018
2017
2016
Month
0
20
40
60
80
100
120

Appendix E

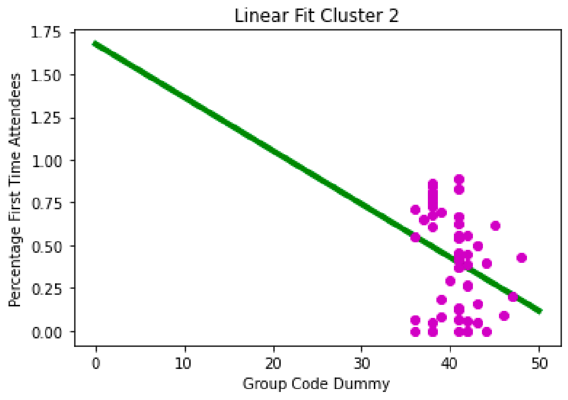

Average Count of First Time Attendees per Group Code
Average Count
0
20
40
60
80
100
PSS PM9 PI9 PVW PAB PHM PSO PSZ PS3 PSB PD9 PH9 PC9 PAY PSX POC PA9 PSA PUL PQL PSL PAZ PST PCC PSK P99 PS9 PG9 PMM POO PAG PSC PUU PSY PCS PC4 PCZ PUB PU9 POB PQX PCW PO9 PQB PUZ PC10 PQK PC11 PIZ
Group Code


Average Percentage of First Time Attendees per Group Code
Average Percentage
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
POO P99 PSO PUL PSS PSK PAB PSB POC PQL PCC PSY PAY PVW PI9 PAZ PUB PAG PCS PS3 PM9 PSA PSL PSZ PSC PST PSX PO9 PA9 POB PC4 PC9 PS9 PU9 PHM PD9 PQK PUZ PC10 PH9 PCW PC11 PQX PG9 PUU PCZ PMM PIZ PQB
Group Code

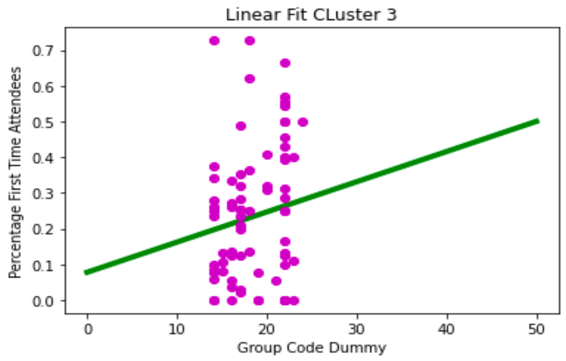

5-Means Clustering Group Code vs Percentage First Time Attendees
Percentage First Time Attendees
0.0
0.2
0.4
0.6
0.8
1.0
0
10
20
30
40
50
Group Code Dummy



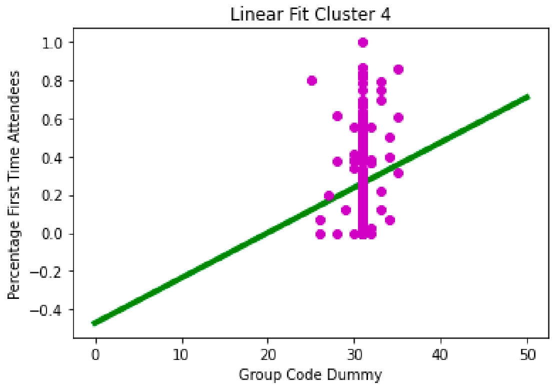
Linear Fit Cluster 1
Percentage First Time Attendees
Group Code Dummy



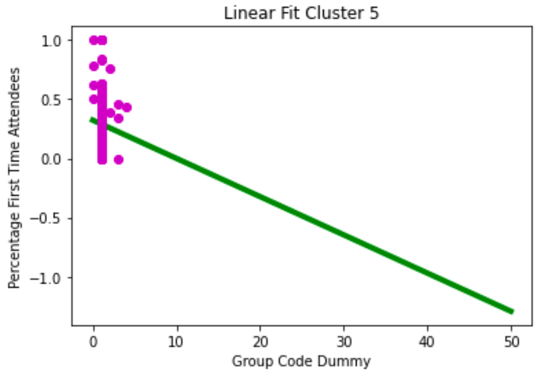
Linear Fit Cluster 2
Percentage First Time Attendees
Group Code Dummy



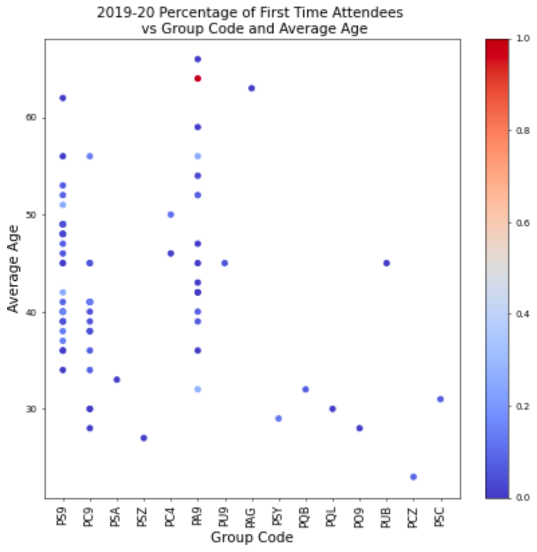
Linear Fit CLuster 3
Percentage First Time Attendees
Group Code Dummy



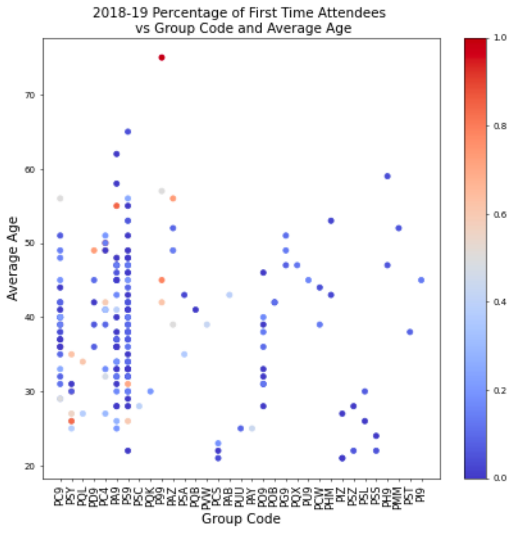
Linear Fit Cluster 4
Percentage First Time Attendees
Group Code Dummy

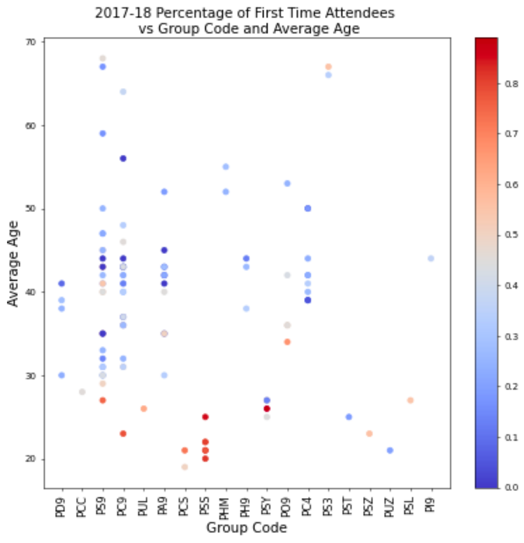


Linear Fit Cluster 5
Percentage First Time Attendees
Group Code Dummy

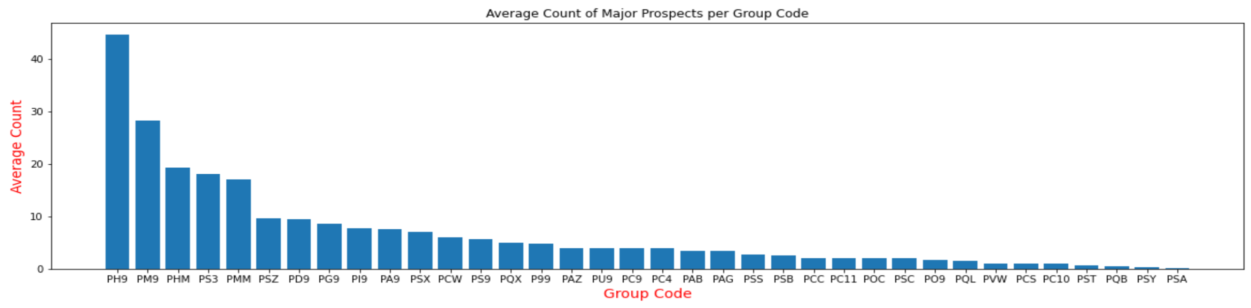


2019-20 Percentage of First Time Attendees
vs Group Code and Average Age
Average Age
Group Code
PS9
PC9
PSA
PSZ
PC4
PA9
PU9
PAG
PSY
PQB
PQL
PO9
PUB
PCZ
PSC

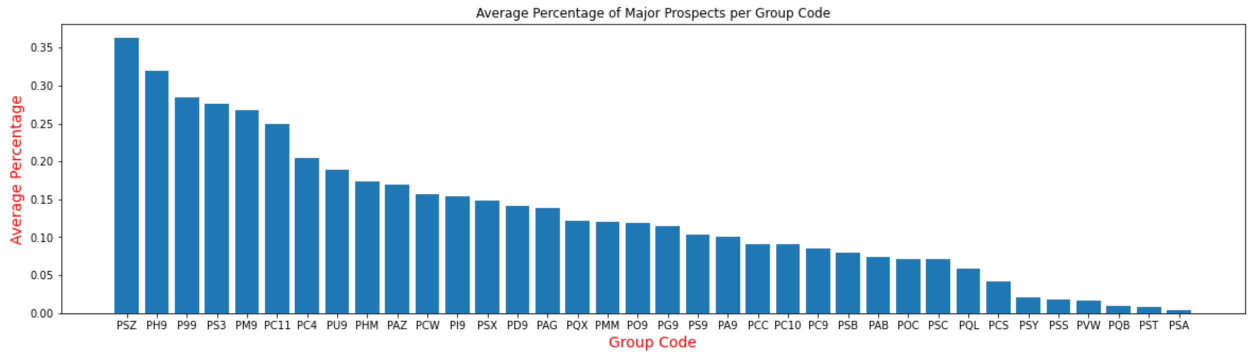

2018-19 Percentage of First Time Attendees
vs Group Code and Average Age
Average Age
Group Code
PC9
PSY
PQL
PD9
PC4
PA9
PS9
PSC
PQK
P99
PAZ
PSA
PQ8
PAW
PCS
PAB
PUU
PAY
PO9
PQB
PG9
PQX
PU9
PCW
PHM
PIZ
PSZ
PSL
PSS
PH9
PMM
PST
PI9


2017-18 Percentage of First Time Attendees
vs Group Code and Average Age
Average Age
Group Code
PD9
PCC
PS9
PC9
PUL
PA9
PCS
PSS
PHM
PH9
PSY
PO9
PC4
PS3
PST
PSZ
PUZ
PSL
PI9

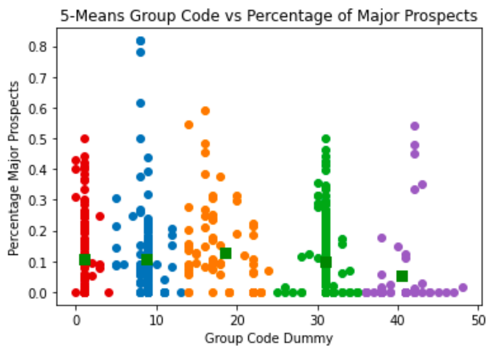
Appendix F



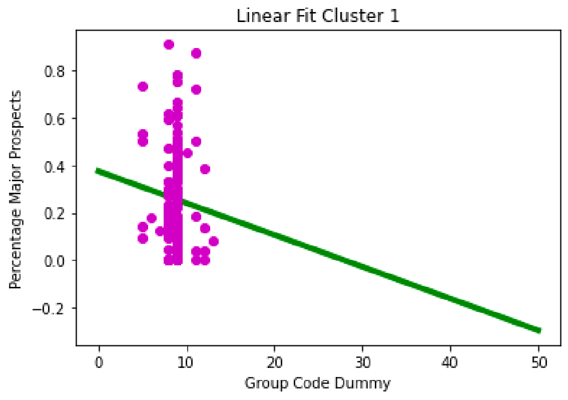
Average Count of Major Prospects per Group Code
Average Count
0
10
20
30
40
PH9
PM9
PHM
PS3
PMM
PSZ
PD9
PG9
PI9
PA9
PSX
PCW
PS9
PQX
P99
PAZ
PU9
PC9
PC4
PAB
PAG
PSS
PSB
PCC
PC11
POC
PSC
PO9
PQL
PVW
PCS
PC10
PST
PQB
PSY
PSA
Group Code



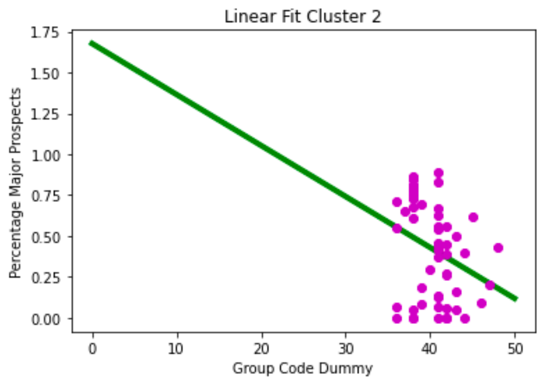
Average Percentage of Major Prospects per Group Code
Average Percentage
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
PSZ
PH9
P99
PS3
PM9
PC11
PC4
PU9
PHM
PAZ
PCW
PI9
PSX
PD9
PAG
PQX
PMM
PO9
PG9
PS9
PA9
PCC
PC10
PC9
PSB
PAB
POC
PSC
PQL
PCS
PSY
PSS
PVW
PQB
PST
PSA
Group Code



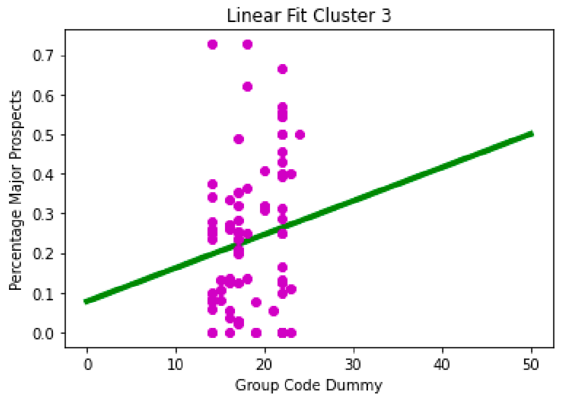
5-Means Group Code vs Percentage of Major Prospects
Percentage Major Prospects
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0
10
20
30
40
50
Group Code Dummy



Linear Fit Cluster 1
Percentage Major Prospects
−0.2
0.0
0.2
0.4
0.6
0.8
0
10
20
30
40
50
Group Code Dummy

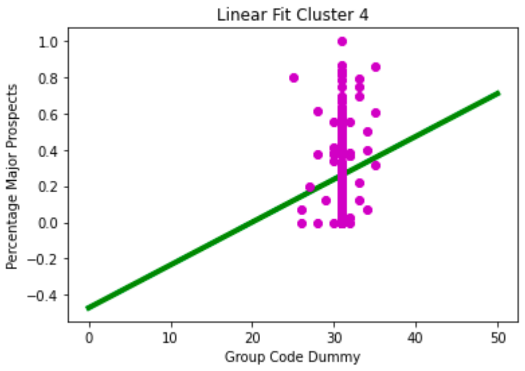


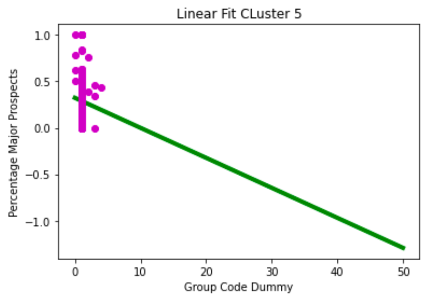
Linear Fit Cluster 2
Percentage Major Prospects
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
0
10
20
30
40
50
Group Code Dummy

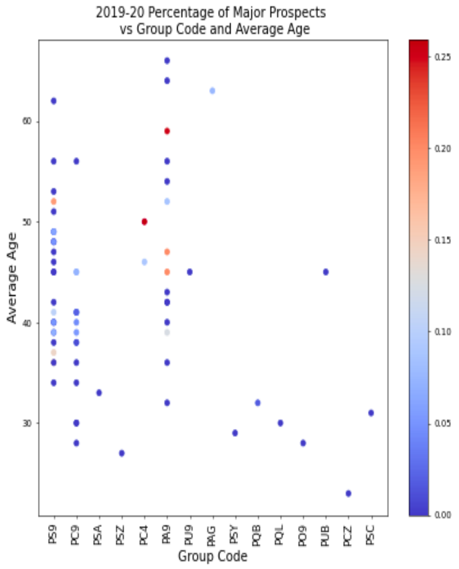


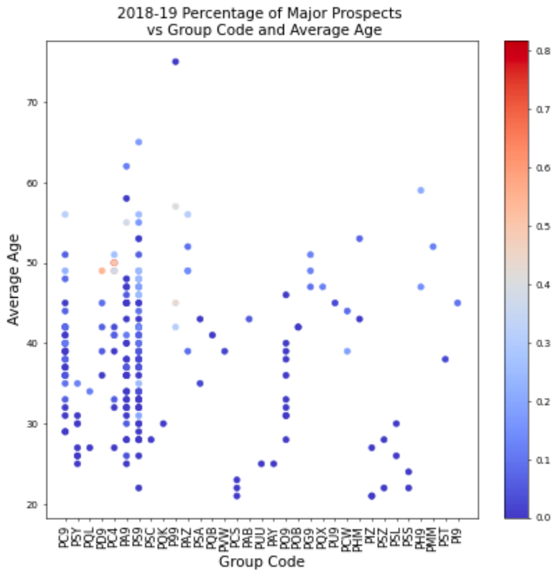
Linear Fit Cluster 3
Percentage Major Prospects
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0
10
20
30
40
50
Group Code Dummy

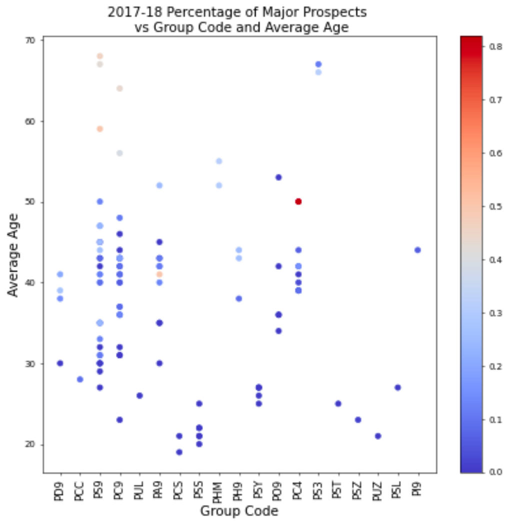
[illegible]


Linear Fit Cluster 4
Percentage Major Prospects
Group Code Dummy
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
0
10
20
30
40
50


Linear Fit CLuster 5
Percentage Major Prospects
Group Code Dummy
1.0
0.5
0.0
−0.5
−1.0
0
10
20
30
40
50

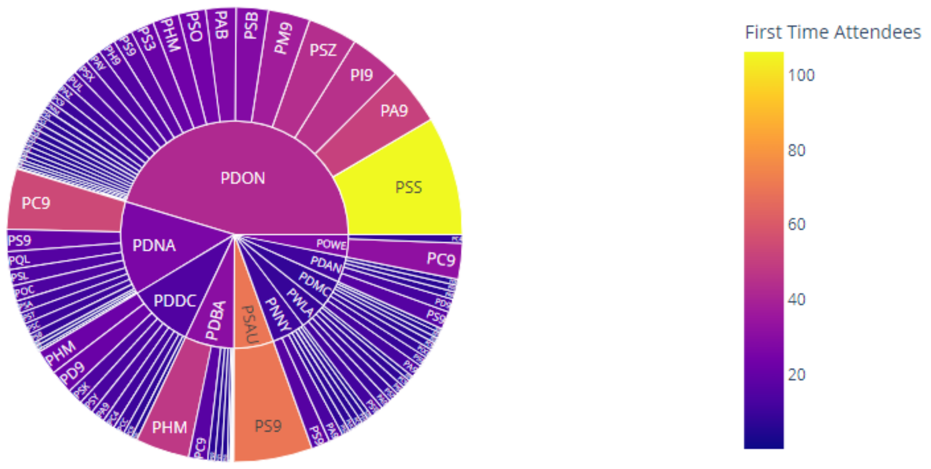

2019-20 Percentage of Major Prospects
vs Group Code and Average Age
Average Age
Group Code
PS9
PC9
PSA
PSZ
PC4
PA9
PU9
PAG
PSY
PQB
PQL
PO9
PUB
PCZ
PSC
0.25
0.20
0.15
0.10
0.05
0.00


2018-19 Percentage of Major Prospects
vs Group Code and Average Age
Average Age
Group Code
PC9
PSY
PQL
PD9
PC4
PA9
PS9
PSC
PQK
P99
PAZ
PSA
PQB
PVW
PCS
PAB
PUU
PAY
PO9
POB
PG9
PQX
PU9
PCW
PHM
PIZ
PSZ
PSL
PSS
PH9
PMM
PST
PI9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0

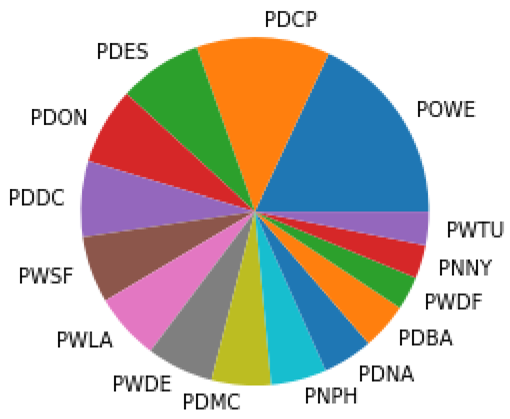

2017-18 Percentage of Major Prospects
vs Group Code and Average Age
Average Age
Group Code
PD9
PCC
PS9
PC9
PUL
PA9
PCS
PSS
PHM
PH9
PSY
PO9
PC4
PS3
PST
PSZ
PUZ
PSL
PI9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0

Appendix G

Top locations for First Time Attendees with information of events hosted

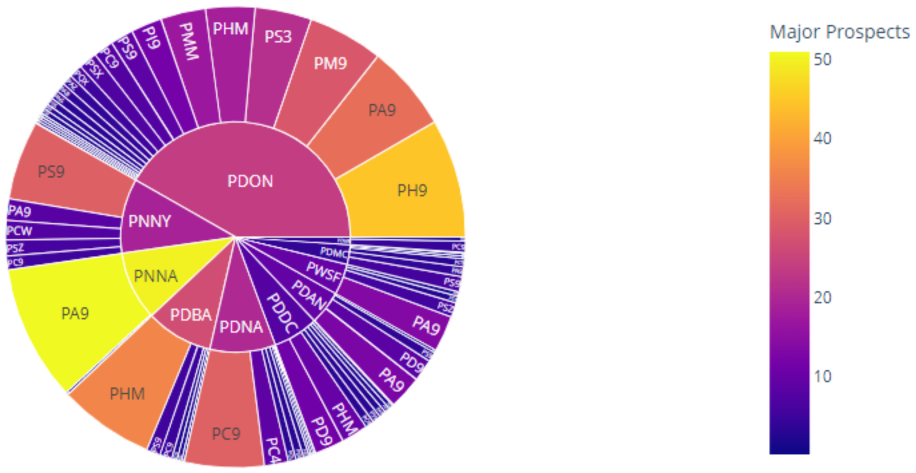


Top 15 Locations for First Time Attendees 2019-20 (AVG)



Appendix H

Top locations for Major Prospects with information of events hosted


PDON
PNNY
PNNA
PDBA
PDNA
PDDC
PDAN
PWSF
PDMC
PH9
PA9
PM9
PS3
PHM
PMM
PI9
PS9
PC9
PS9
PA9
PCW
PSZ
PC9
PA9
PHM
PC9
PC4
PD9
PHM
PA9
PD9
PA9
Major Prospects