Alumni Terps Analysis

Professor Adam Lee

BUDT704_0507_09

Eileen Fang,

Sai Vaishnav Vinjamuri

Yishang Huang

Kai Li

Introduction

The purpose of this assignment was to analyze the data set of the past Alumni Association events and use that information to find strong correlation that leads to the highest number of first time attendees and major gift prospect attendees. We then would use the results and to improve the event participation for future events. We looked at the variables provided in the dataset to see how the Alumni Association could get more first time attendees, more major gift prospects, and higher participation in general. We focused on 5 different variables: location, group, activity, age, and time of attendance (year and month).

Mission Statement

We used the variables that resulted in the highest number of first time attendees and major gift prospect attendees. We hoped that we could improve the event participation for future event events by using the results we found.

Mission Objective

We focused on putting on an event that was attractive to the highest number of first time attendees and major gift prospect attendees. We inspected the variables to improve the Alumni Association Event Participation. Higher donations to the universities would be caused by higher participation.

Method

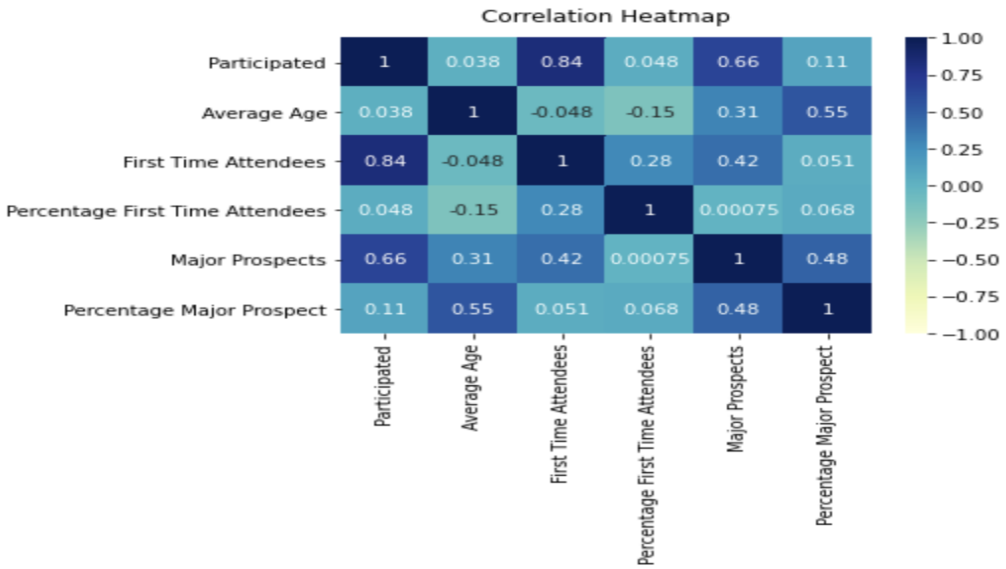
We used the categorical codes for activity, location, and group to determine our findings, grouped the location code: DMV, online, North East, West Coast, South East, Other, looked at

distribution for the numerical values, created the clusters and cluster centers by using the numerical variables, formed correlation matrix and heat map to observe the correlation between different numerical variables and finally performed the regression analysis to figure out the relationship between each numerical variable.

General description

Correlation

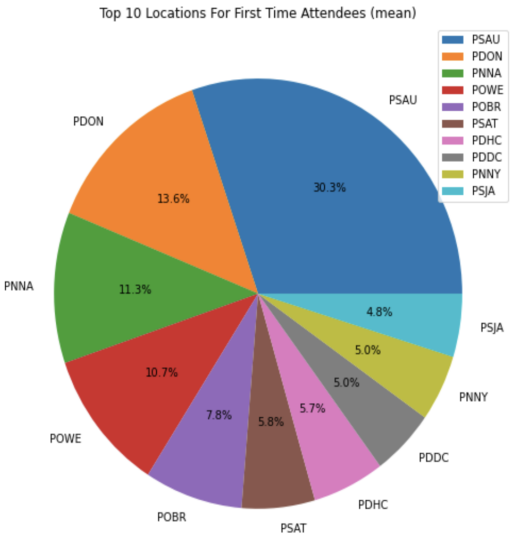
	Participated	Average Age	First Time Attendees	Percentage First Time Attendees	Major Prospects	Percentage Major Prospect
Participated	1.000000	0.037616	0.835996	0.047840	0.658973	0.113415
Average Age	0.037616	1.000000	-0.048204	-0.152633	0.308342	0.549320
First Time Attendees	0.835996	-0.048204	1.000000	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

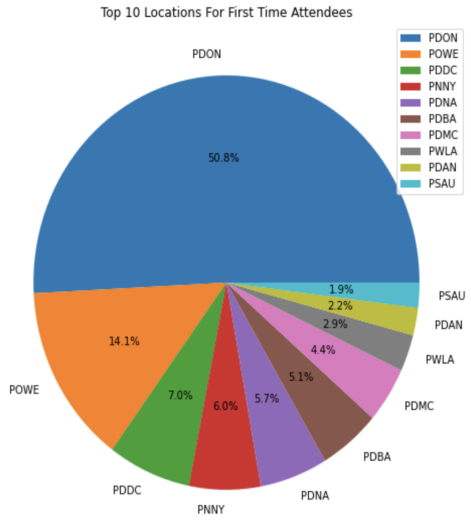


For the general description, we created a correlation matrix and heat map in order to better observe the correlations between 6 numerical variables. We know how highly or poorly these numerical variables were correlated to each other. According to the correlation matrix and heatmap, we can see that the “participated” and “first time attendees” are highly correlated to each other and the same situation goes for “major prospects” and “participated”. Then in order to further discover how the variables affected each other, we did more complicated research of categorical variables and numerical variables.

Location Analysis

We created a pie chart based off of location code and count of participants. We focused on the top 10 locations (both sum and mean) and made the participants the value. Based on the sum of





location results, the highest percentage for location was CP DMV (PDON) with 44.7%, CP online (POWE) with 12.2% and CP Northeast-New York (PNNY) with 11.3%. Then we did the same pie chart for both first time attendees (pictured below) and major prospects. By analyzing the data, the location with the highest first time attendees included CP DMV (PDON) with 50.8%, CP Online- Webinar (POWE) with 14.1%, and CP Southeast- Austin (PSAU) with 6.0%.

When looking into the data set, we can see that the last event in Southeast- Austin was in 2017-2018. The total participants was 260 people, first time attendees was 139 people, and major prospects was 50 people. All three of these were all well above the mean values (pictures to the left). This finding leads us to recommend at least one event in Southeast- Austin every year. For the major prospects, the top three locations were CP DMV- On Campus (PDON) with 49.5%, CP Northeast - New York (PNNY) with 16.7%, CP

```
print(df['Participated'].mean())
print(df['First Time Attendees'].mean())
print(df['Major Prospects'].mean())
```

```
44.80385852090032
13.456591639871382
5.966237942122186
```

```
#Grouping Location Code to be more general
df2=df
lst1=df2["Location Code"]
lst2=[]
for i in range(0,lst1.size):
    if "PS" in df2["Location Code"][i]:
        lst2.append("South East")
    elif "PW" in df2["Location Code"][i]:
        lst2.append("West Coast")
    elif "PD" in df2["Location Code"][i]:
        lst2.append("DMV")
    elif "PN" in df2["Location Code"][i]:
        lst2.append("North East")
    elif "PO" in df2["Location Code"][i]:
        lst2.append("Online")
    else:
        lst2.append("Other")
```

DMV - Baltimore (PDBA) with 5.8%. Based on these results, we can conclude that the DMV area has high counts for all three different types of attendees.

Additionally to looking at specific locations, we generalized the data to be categorized into DMV, online, North East, West Coast, South East, and Other. This emphasized the relationship between higher participation, first time attendees, and major prospects with the location of the DMV area and online events. These variables were higher in these locations. The virtual setting has been advantageous especially since the pandemic. People are able to join these events from all over the world, no matter their location. A huge part of location is convenience. Most people who come to the University of Maryland are in-state or close to the East coast.

Group Analysis

For the group analysis, we grouped the code into more generalized categories: Social, PreDev, Athletics, Cultural, Affinity, Service, Advocacy Group, General, D&I Alumnae, Campaign,

```
#Grouping the Group Code to be more General
df3=df
lst1=df3["Group Code"]
lst2=[]
for i in range(0,lst1.size):
    if "PS" in df3["Group Code"][i]:
        lst2.append("Social")
    elif "PC" in df3["Group Code"][i]:
        lst2.append("ProDev")
    elif "PA" in df3["Group Code"][i]:
        lst2.append("Athletics")
    elif "PU" in df3["Group Code"][i]:
        lst2.append("Cultural")
    elif "PQ" in df3["Group Code"][i]:
        lst2.append("Affinity")
    elif "PO" in df3["Group Code"][i]:
        lst2.append("Service")
    elif "PD" in df3["Group Code"][i]:
        lst2.append("Advocacy")
    elif "P9" in df3["Group Code"][i]:
        lst2.append("General")
    elif "PV" in df3["Group Code"][i]:
        lst2.append("D&I Alumnae")
    elif "PG" in df3["Group Code"][i]:
        lst2.append("Campaign")
    elif "PH" in df3["Group Code"][i]:
        lst2.append("Stewardship")
    elif "PI" in df3["Group Code"][i]:
        lst2.append("Cultivation")
    elif "PM" in df3["Group Code"][i]:
        lst2.append("Membership")
    else:
        lst2.append("Other")
```

Stewardship, Cultivation, and Membership. This bar graph showed that for total attendees, first time attendees, and major prospects, the top three groups were membership, stewardship, advocacy group.

Then we looked at more specific groups and went deeper into the categories. We looked at the top 15 group codes based on first time attendees and major prospects. One bar graph was based on the highest 15 group code with first time attendees with the comparison of major prospects right next to the data and the other bar graph created was with the highest 15 group code with major prospects in comparison with the first time attendees next to it. The top 15 codes were on the x axis and the count of first time attendees or major prospects were on the y axis. Based on these graphs, for first time attendees, the highest count was for CP Social- Students, CP Membership - General, and CP Cultivation - General. For major prospects, the highest count for

groups were CP Stewardship - General, CP Membership - General, CP Stewardship - Membership.

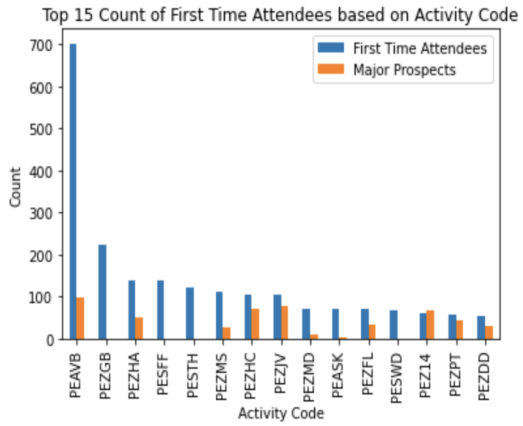
Activity Analysis

For the activity analysis, we first created the new group which included the activity code, first-time attendees and the major prospects. Then we groupby this group by using their mean. Because we want the top 15 count of the first time attendees based on the activity code, we have to make a new group with a descending order. In order to have a considerable number of observations, but not too much information, we chose the head 15. Then what we did is just gave the title and xlabel the appropriate name. And we did the same thing for the dataset of the major prospects.

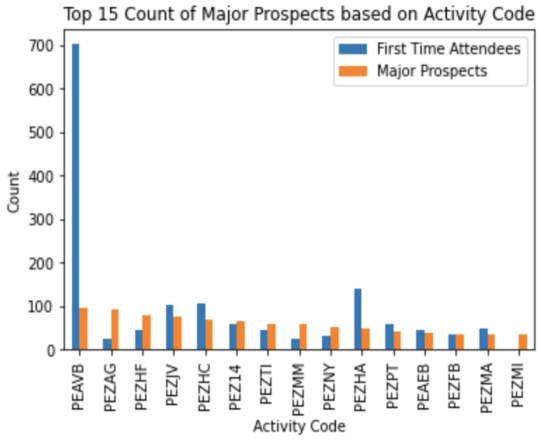
```
# make the diatribution of the Activity according to their First Time Attendees and Major Prospect
group2=df[['Activity Code','First Time Attendees','Major Prospects']]
group2=group2.groupby('Activity Code').mean()
group2SortFT=group2.sort_values(['First Time Attendees'],ascending=False)
group2SortPS=group2.sort_values(['Major Prospects'],ascending=False)
group2SortFT10=group2SortFT.head(15)
group2SortPS10=group2SortPS.head(15)

group2SortFT10.plot.bar()
plt.title("Top 15 Count of First Time Attendees based on Activity Code")
plt.ylabel("Count")
group2SortPS10.plot.bar()
plt.title("Top 15 Count of Major Prospects based on Activity Code")
plt.ylabel("Count")
```

And we got the plots below. It is not hard to find the top 3 of this top 15 list are virtual book club, gradbash and happy hour. And the difference between the amount of the first time attendees of the virtual book club and the gradbash is really huge. So we can focus on these three activity themes and even pay more attention to the virtual book club. One interesting thing is that the last happy hour was hosted in 2017-2018 which maybe is because of the CORVID-19, so maybe we could do happy hour online again during the pandemic.



As for the major prospects. The top 1 was still the virtual book club,which means we should pay a lot of attention to this activity area. Then is following by awards gala, and hall of fame ceremony.



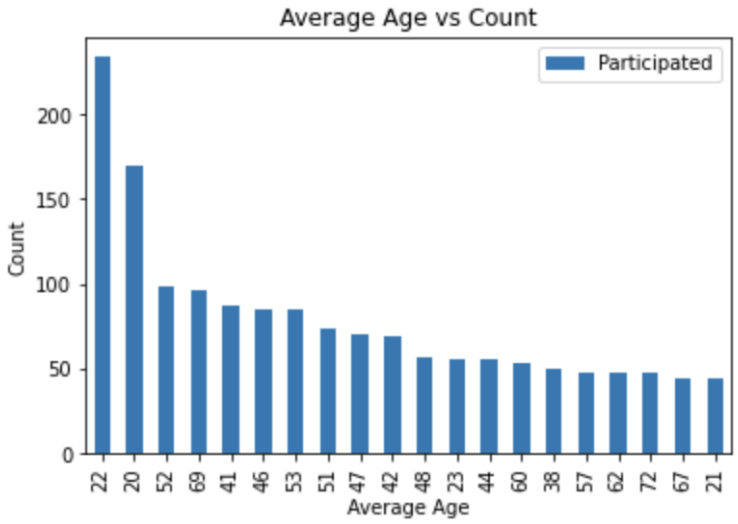
Age

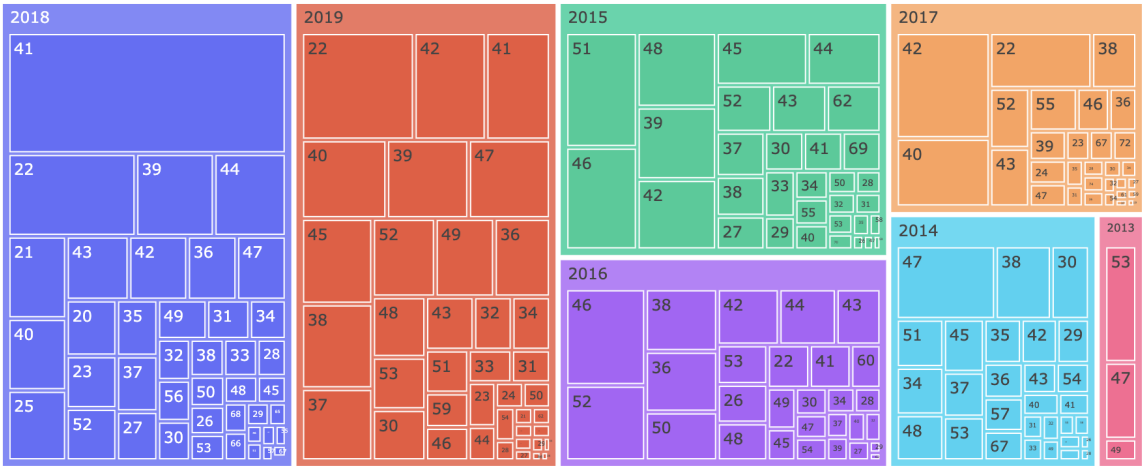
```
# make the distribution of the total participted based on the average age.
group1=df[['Average Age','Participated']]
group1=group1.groupby('Average Age').mean()
group1SortP=group1.sort_values(['Participated'],ascending=False)
group1SortP20=group1SortP.head(20)
group1SortP20.plot.bar()
plt.ylabel('Count')
plt.title("Average Age vs Count")
```

About the age part, we created the group which just included the average age and the total count of the participants. Next we groupby this group by using the mean of the average. Then create a new group with a descending order based on the total count of the participants. And we chose the top 20 of them this time.

```
import plotly.express as px
fig = px.treemap(df, path = ["year","Average Age"], values = "Participated")
fig.show()
```

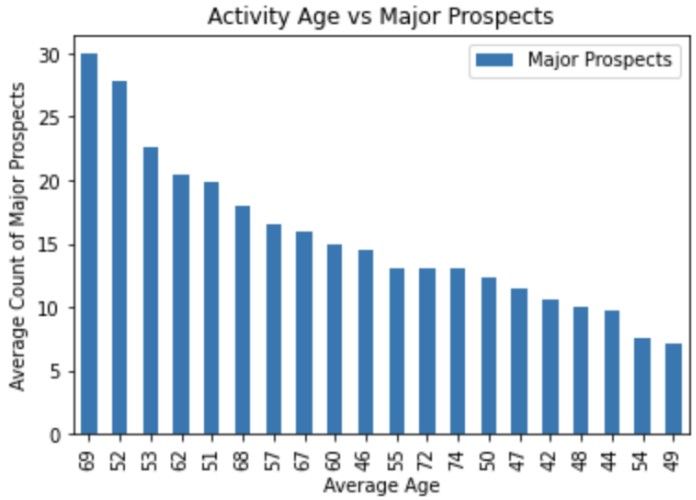
Finally we got the one of the results that we want below. Then we want to find out the specific age distribution of every different year, so we import the ploty.express as px first, then we made the treemap by using the year and the average age of the [df] dataframe, and number of the total attendees the values of this figure. After that, we got the second plot below.





According to the first plot following which showed the age distribution about their participants, we can see that participants were mostly young people, most of them are 20-22 years old. About the heat map we can find out that during the 2013-2015, participants were mostly elderly people who are around 40-50, young people became the main participants which just happened in recent years.

We also did the same thing for the major prospects, only changing the group by using the average age and the major prospects. Then we got the distribution below. we can see most Participants of the major prospects are over 50. So maybe we have more chances to get some donations from elder people through these major events.



Time of Attendees(Year/ Month)

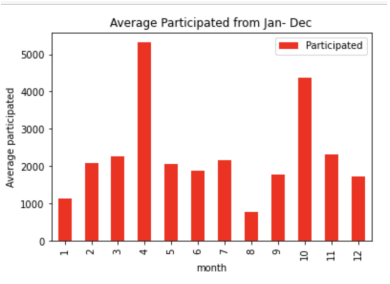
We want to find out the relationship between the event time(date) and the total attendees. So we create the month distribution for the total number of the participants, first time attendees and the major prospects. Here are our codes for creating the plots.

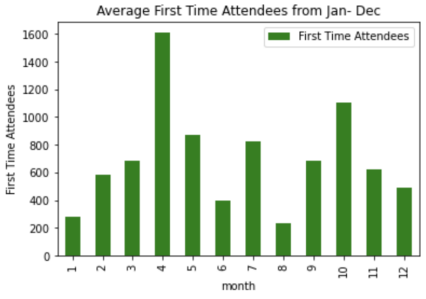
```
#bar chart showing mean participation per month every year
df['month']= pd.DatetimeIndex(df['Event Date']).month
Month = df.groupby('month').sum()
Month.reset_index(inplace = True)
Month.plot(x = 'month', y = 'Participated',kind = "bar",color= "red")
plt.ylabel('Average participated')
plt.title('Average Participated from Jan- Dec')
plt.show()
```

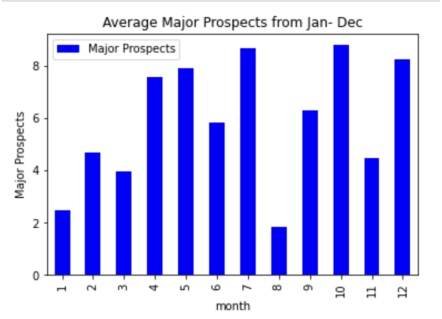
```
#bar chart showing mean first time attendees per month
df['month']= pd.DatetimeIndex(df['Event Date']).month
Month = df.groupby('month').sum()
Month.reset_index(inplace = True)
Month.plot(x = 'month', y = 'First Time Attendees', kind = "bar", color= "green")
plt.ylabel('First Time Attendees')
plt.title('Average First Time Attendees from Jan- Dec')
plt.show()
```

```
#bar chart showing mean major prospects per month
Monthave = df.groupby('month').mean()
Monthave.reset_index(inplace = True)
Monthave.plot(x = 'month', y = 'Major Prospects',kind = "bar", color= "blue")
plt.ylabel('Major Prospects')
plt.title('Average Major Prospects from Jan- Dec')
plt.show()
```

We first created the new data frame which is called month. For the total attendees and the first time attendees, we used the sum of the month when we groupby it, but for the major prospects, we used the mean of the month. And we set our xabel as month, yabel as each goal. Then we got







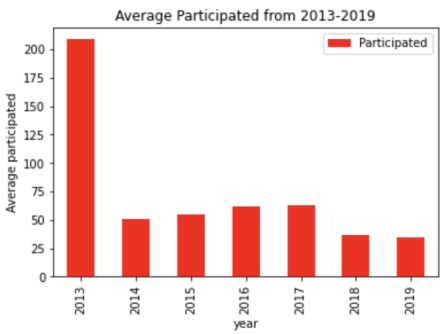
these figures. We can get the same conclusion from three different aspects is that we should host more event on April and(or) October.

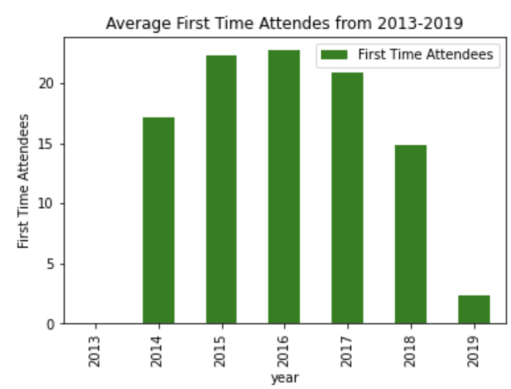
```
#bar chart showing mean first time attendees per year
df['year']= pd.DatetimeIndex(df['Event Date']).year
Year = df.groupby('year').mean()
Year.reset_index(inplace = True)
Year.plot(x = 'year', y = 'First Time Attendees',kind = "bar", color= "green")
plt.ylabel('First Time Attendees')
plt.title('Average First Time Attendes from 2013-2019')
plt.show()
```

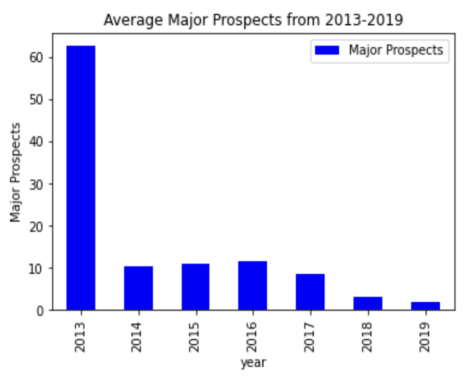
```
#bar chart showing mean major prospects per year
df['year']= pd.DatetimeIndex(df['Event Date']).year
Year = df.groupby('year').mean()
Year.reset_index(inplace = True)
Year.plot(x = 'year', y = 'Major Prospects',kind = "bar", color= "blue")
plt.ylabel('Major Prospects')
plt.title('Average Major Prospects from 2013-2019')
plt.show()
```

```
#bar chart showing mean major prospects per year
df['year']= pd.DatetimeIndex(df['Event Date']).year
Year = df.groupby('year').mean()
Year.reset_index(inplace = True)
Year.plot(x = 'year', y = 'Major Prospects',kind = "bar", color= "blue")
plt.ylabel('Major Prospects')
plt.title('Average Major Prospects from 2013-2019')
plt.show()
```

And we did the same thing for the year to find out the relationship between the year and the total attendees, first time attendees and major prospects.







The logic of the code was the same as what we used for month distribution. After ran the code, we can get the result immediately.

We can see that 2013 had the most Participants and major prospects, 2016 had the highest first-time attendance.

Regression Analysis

As we have stated from Mission Objectives that the main focus is upon Dependent variables namely Major Prospect gift attendees and First time Attendees. Hence , we wanted to find the combination of Independent variables that have the highest Linearity linkage with the dependent variables. After Performing Regression analysis with many different combinations of independent variables , we were able to pick two best models in total , One model for major prospect attendees and one model for first time attendees.

Model-1

Model-1 is considered with Major Prospect attendees as dependent variables and with independent variables as Participated , Average age , Year (dummy variable) , first time attendees. A dummy variable is created for Variable ‘Year’.

```
#REGRESSION MODELS
## reg1 -- Major prospects vs age,participation, year

x = pd.concat((df['Average Age'],df['Participated'],df['year'],df['First Time Attendees']),axis=1)
x = sm.add_constant(x)
Y = df['Major Prospects']

# Note the difference in argument order
model = sm.OLS(Y, x).fit()
predictions = model.predict(x) # make the predictions by the model

# Print out the statistics
model.summary()
```

As above we can see the code for Regression Analysis that we have performed with dependent variables as Major Prospect Attendees and Independent variables as Participated , Average Age , Year and First time Attendees. With this particular set of combinations of Independent variables we have evaluated a best Regression model with Major Prospect Attendees and Dependent Variable.

After Evaluating the model , We have got a Significantly higher R^2 value of 0.611 and which shows the model is pretty good to be considered.

OLS Regression Results

Dep. Variable:	Major Prospects	**R-squared:**	0.611
Model:	OLS	**Adj. R-squared:**	0.609
Method:	Least Squares	**F-statistic:**	242.6
Date:	Wed, 08 Dec 2021	**Prob (F-statistic):**	4.58e-125
Time:	10:14:49	**Log-Likelihood:**	-2235.1
No. Observations:	622	**AIC:**	4480.
Df Residuals:	617	**BIC:**	4502.
Df Model:	4		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	4046.7652	425.193	9.517	0.000	3211.764	4881.767
Average Age	0.3455	0.037	9.365	0.000	0.273	0.418
Participated	0.1465	0.007	20.905	0.000	0.133	0.160
year	-2.0123	0.211	-9.550	0.000	-2.426	-1.598
First Time Attendees	-0.1410	0.016	-8.826	0.000	-0.172	-0.110

Omnibus:	405.791	**Durbin-Watson:**	1.950
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	22273.493
Skew:	2.191	**Prob(JB):**	0.00
Kurtosis:	31.987	**Cond. No.**	2.42e+06

Model-2

Model-2 is considered with First Time attendees as dependent variables and with independent variables as Participated , Average age .

```
## reg2 -- First time attendeee vs age,participation

x = pd.concat((df['Average Age'],df['Participated']),axis=1)
x = sm.add_constant(x)

Y = df['First Time Attendees']
# Note the difference in argument order
model = sm.OLS(Y, x).fit()
predictions = model.predict(x) # make the predictions by the model

# Print out the statistics
model.summary()
```

As above we can see the code for Regression Analysis that we have performed with dependent variables as First Time Attendees and Independent variables as Participated , Average Age . With this particular set of combinations of Independent variables we have evaluated a best Regression model with First time Attendees and Dependent Variables.

OLS Regression Results

Dep. Variable:	First Time Attendees	**R-squared:**	0.705
Model:	OLS	**Adj. R-squared:**	0.704
Method:	Least Squares	**F-statistic:**	740.5
Date:	Wed, 08 Dec 2021	**Prob (F-statistic):**	6.29e-165
Time:	10:15:10	**Log-Likelihood:**	-2813.6
No. Observations:	622	**AIC:**	5633.
Df Residuals:	619	**BIC:**	5646.
Df Model:	2		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	10.3740	3.812	2.721	0.007	2.887	17.861
Average Age	-0.3366	0.092	-3.653	0.000	-0.518	-0.156
Participated	0.3702	0.010	38.421	0.000	0.351	0.389

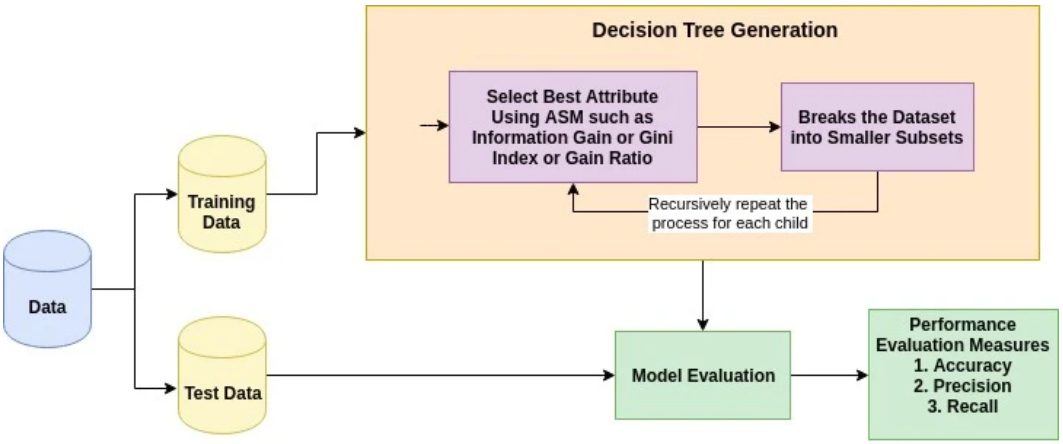
Omnibus:	252.686	**Durbin-Watson:**	1.661
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	95641.173
Skew:	0.309	**Prob(JB):**	0.00
Kurtosis:	63.745	**Cond. No.**	447.

After Evaluating the model , We have got a Significantly higher R^2 value of 0.705 and which shows the model is pretty good to be considered.

Decision Tree Algorithm

The basic idea behind any decision tree algorithm is as follows:

1. Select the best attribute using Attribute Selection Measures(ASM) to split the records.
2. Make that attribute a decision node and break the dataset into smaller subsets.
3. Starts tree building by repeating this process recursively for each child until one of the condition will match:
    - All the tuples belong to the same attribute value.
    - There are no more remaining attributes.
    - There are no more instances.



Model-1

Decision tree Algorithm is the first method that we have chosen to find the accuracy of our Models. We have evaluated the data into two categories by choosing train and test data for which we have chosen 80% data from train data and 20% of data from test data for Prediction.

```
## decision Tree 1 ( For Model-1 )
x = pd.concat((df['Average Age'],df['First Time Attendees'],df['Participated'],df['year']),axis=1)
Y = df['Major Prospects']

# Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(x, Y, test_size=0.3, random_state=1)

# Create Decision Tree classifer object
clf = DecisionTreeClassifier(criterion="entropy",max_depth=4)

# Train Decision Tree Classifer
clf = clf.fit(X_train,y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)

# Model Accuracy, how often is the classifier correct?
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.39037433155080214
```

After Predicting the values and analysing , we have got an accuracy of 0.3903 which shows that model is good

Model-2

Similar to the above model , We have evaluated the data into two categories by choosing train and test data for which we have chosen 80% data from train data and 20% of data from test data for Prediction.

```
## decision tree 2 ( For Model-2)

x = pd.concat((df['Average Age'],df['Participated']),axis=1)
Y = df['First Time Attendees']

# Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(x, Y, test_size=0.3, random_state=1)

# Create Decision Tree classifer object
clf = DecisionTreeClassifier(criterion="entropy",max_depth=4)

# Train Decision Tree Classifer
clf = clf.fit(X_train,y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)

# Model Accuracy, how often is the classifier correct?
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

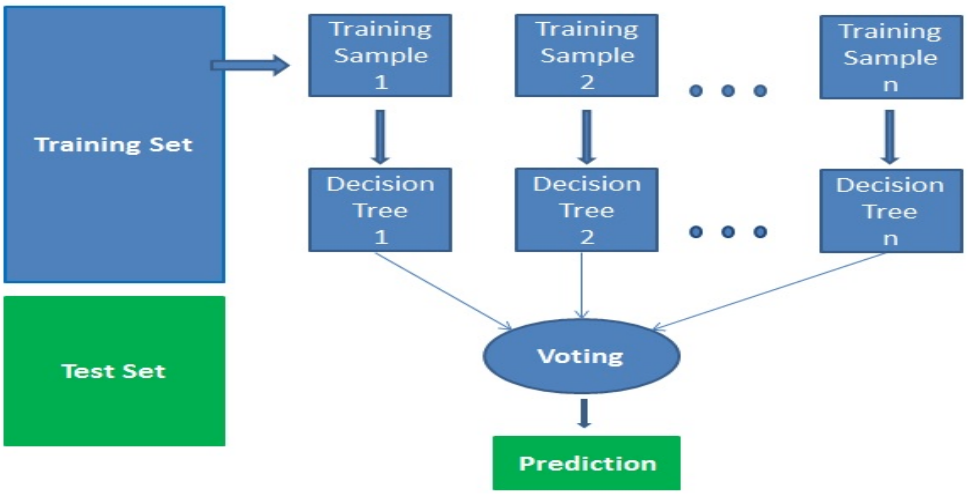
```
Accuracy: 0.1497326203208556
```

After Predicting the values and analysing , we have got an accuracy of 0.149 which shows that model is good.

Random Forests Algorithm

Random forest algorithm works in four steps:

1. Select random samples from a given dataset.
2. Construct a decision tree for each sample and get a prediction result from each decision tree.
3. Perform a vote for each predicted result.
4. Select the prediction result with the most votes as the final prediction.



We have used following libraries to perform Random Forests Algorithm:

```
from sklearn.tree import DecisionTreeClassifier # Import Decision Tree Classifier
from sklearn.model_selection import train_test_split # Import train_test_split function
from sklearn import metrics #Import scikit-learn metrics module for accuracy calculation
```

Model-1

Random Forests Algorithm is the Second method that we have chosen to find the accuracy of our Models. We have evaluated the data into two categories by choosing train and test data for which we have chosen 80% data from train data and 20% of data from test data for Prediction. We have even used a set of classifiers such as Gaussian clarifiers in our model for prediction.

```
## Random Forest 1 ( Model-1)


x = pd.concat((df['Average Age'],df['First Time Attendees'],df['Participated'],df['year']),axis=1)
Y = df['Major Prospects']

# Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(x, Y, test_size=0.3, random_state=1)

#Import Random Forest Model
from sklearn.ensemble import RandomForestClassifier

#Create a Gaussian Classifier
clf=RandomForestClassifier(n_estimators=100)

#Train the model using the training sets y_pred=clf.predict(X_test)
clf.fit(X_train,y_train)

y_pred=clf.predict(X_test)

#Import scikit-learn metrics module for accuracy calculation
from sklearn import metrics
# Model Accuracy, how often is the classifier correct?
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.41711229946524064
```

When we have evaluated the results for Model-1 using Random Forests we have got an accuracy of 0.417

Model-2

Similar to Model-1 , when we have performed random Forests Algorithm and evaluated the accuracy value we have got a value of 0.139.

```
## Random forest 2

x = pd.concat((df['Average Age'],df['Participated']),axis=1)
Y = df['First Time Attendees']

# Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(x, Y, test_size=0.3, random_state=1)

#Import Random Forest Model
from sklearn.ensemble import RandomForestClassifier

#Create a Gaussian Classifier
clf=RandomForestClassifier(n_estimators=100)

#Train the model using the training sets y_pred=clf.predict(X_test)
clf.fit(X_train,y_train)

y_pred=clf.predict(X_test)

#Import scikit-learn metrics module for accuracy calculation
from sklearn import metrics
# Model Accuracy, how often is the classifier correct?
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.13903743315508021
```

After finding and evaluating values for Both models using Decision Trees and Random Forests , the reason we have considered Decision tree values for accuracy is because they are reliable.

Random Forests vs Decision Trees

- Random forests are a set of multiple decision trees.
- Deep decision trees may suffer from overfitting, but random forests prevent overfitting by creating trees on random subsets.
- Decision trees are computationally faster.
- Random forests are difficult to interpret, while a decision tree is easily interpretable and can be converted to rules.

Hence , we have considered the Decision Tree Algorithm to find and produce the Accuracy for two Regression Models that we have built .

Future Work and Recommendations

Based on the results of the findings from the data, we can conclude the following variables to be successful in reaching the objective. To increase the total attendance and first time attendance:

- Location: Focus on having more events in the DMV area and online, however, have at least one event in CP Southeast- Austin per year
- Group: We can have more events about CP Social- Students
- Activity: More events based on CP Happy Hour and consider an online Happy Hour.
- Time: More events in April and October

To increase major prospects:

- Location: Have more events on CP DMV - On campus
- Group: Have more events about CP Stewardship - Groups
- Activity: Have more events about CP AA- Virtual Book Club
- Time: More events in April and October

How to test and evaluate your project?

Use Jupyter Notebook to test the code. And about the project, we can start with the recommendation that we offered, we can see if there is an increase of the total attendees and the first time attendees and the major prospects of the next year's events when the association accepts our suggestion for future work.