STAT 4911 Project 2
Donor Giving Insights Report
Recommendations

Instructor: Thomas Metzger
Team Number: Team 4
Team Members: Xidan Kou, Songyuan Wu, Caroline Pier, Kat Husar, Troy Stein, Carli Werner

I. Introduction

We analyzed variables to recommend donor designations based on donor history, donation combinations, location trends, and donor demographics. In overview, we used association rules, logistic regression, and user-based collaborative recommendation systems. Our analysis was mainly focused on common fund designations.

II. Designation Overlap Overview

A. Frequent Designation Combinations

Below are the breakdowns for the twenty combinations of designations. The system reads as follows the selected designation plus the other designations donated scaled by count.

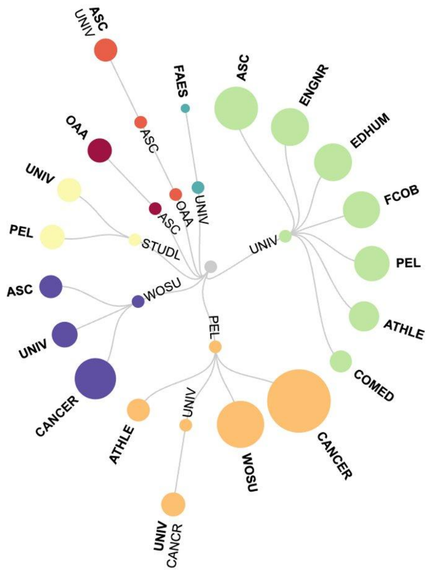


Figure 1: Top 20 Giving Combinations

The data frame was adjusted to only include donation combinations with two or more different designations. Notable giving combinations include those with Pelotonia, WOSU, and the university.

Python Code:

```python
df = df.rename(columns = {'Unnamed: 0':'sets', 'DESIGNATIONFUND_UNIT_CDOCODEalt':'frequency'})
#if set has a comma by definition 2 or more
df_revised = df.drop(df[df.sets.map(lambda x: "," not in x)].index)
```

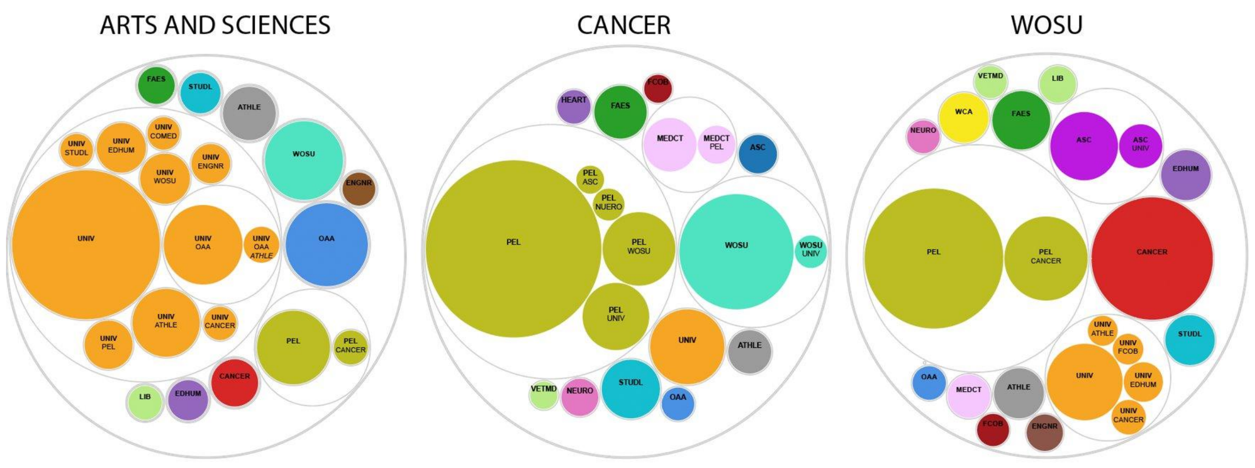


Figure 2: Top Designation Combinations Part 1

In this circle packing visualization we can see the top 20 designations combinations with Arts and Sciences or Cancer or WOSU. There are a notable number of multiple designations for those having donated to Arts and Sciences. Donations to the Arts and Sciences tend to overlap with donations to the University and individual university colleges. Donations for cancer tend to overlap with Pelotonia, Neurology, the Medical Center, and WOSU. Donations to WOSU notably overlap with Pelotonia, cancer, the Arts and Sciences, the university, and humanities.

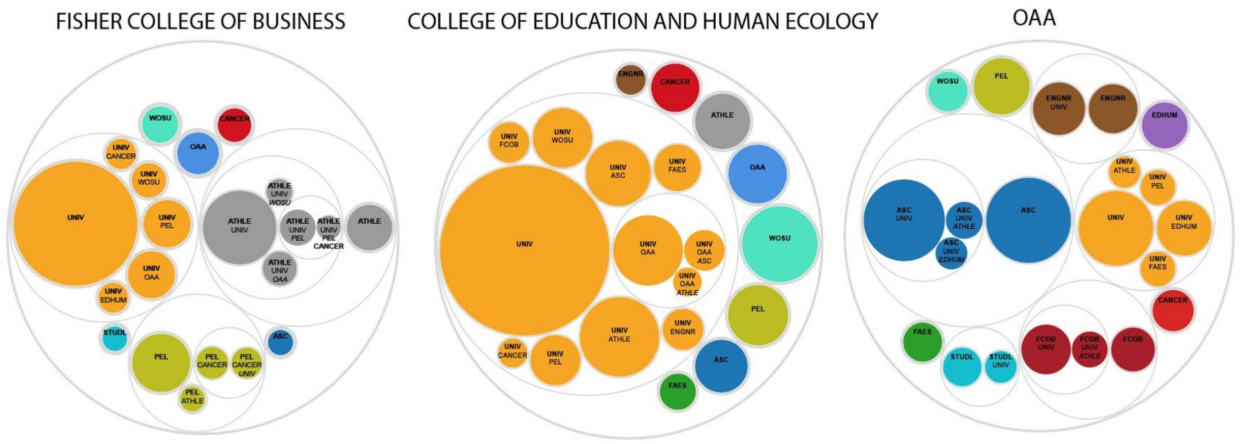


Figure 3: Top Designation Combinations Part 2

In this circle packing visualization we can see the top 20 designations combinations with the Fisher College of Business, the College of Education and Human Ecology, and OAA. The Fisher College of Business tends to have more donations toward Athletics and Pelotonia. Those donating to FCOB also tend to donate to a variety of designations. The College of Education and Human Ecology tends to have more donations toward the university and the university in combination with other designations. The Office of Academic Affairs tends to have more donations, not heavily weighted toward another designation in particular, but among ASC, FCOB, and UNIV then College of Engineering and Student Life.

Python Code:
```
#part 1
df_revised = df.drop(df[df.sets.map(lambda x: "ASC" not in x)].index)
df_revised = df.drop(df[df.sets.map(lambda x: "CANCR" not in x)].index)
df_revised = df.drop(df[df.sets.map(lambda x: "WOSU" not in x)].index)
#part 2
```

```
df_revised = df.drop(df[df.sets.map(lambda x: "WOSU" not in x)].index)
df_revised = df.drop(df[df.sets.map(lambda x: "WOSU" not in x)].index)
df_revised = df.drop(df[df.sets.map(lambda x: "WOSU" not in x)].index)
```

B. Top Giving Unit by Location

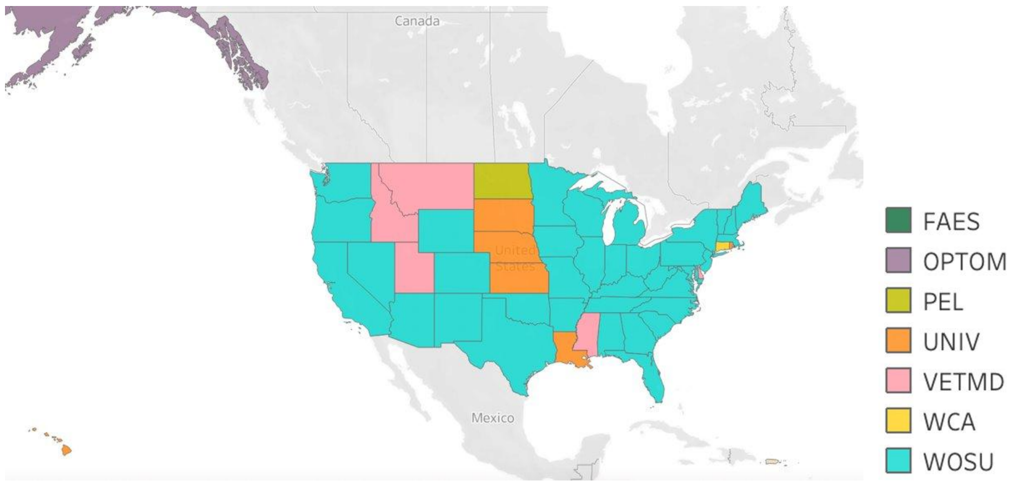


Figure 4: Top Giving Unit by State

By State there is an overall trend of donating to WOSU. In the West there are more donations to Veterinary Medicine.

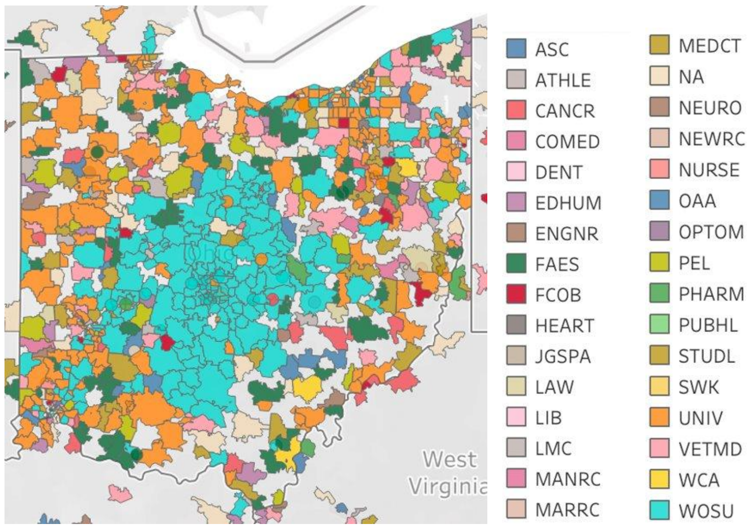


Figure 5: Top Giving Unit by Zip-Code

The top giving unit by zip-code in central Ohio trends toward WOSU. There are notable numbers of donations to the University and the College of Food, Agriculture, and Environmental Sciences.

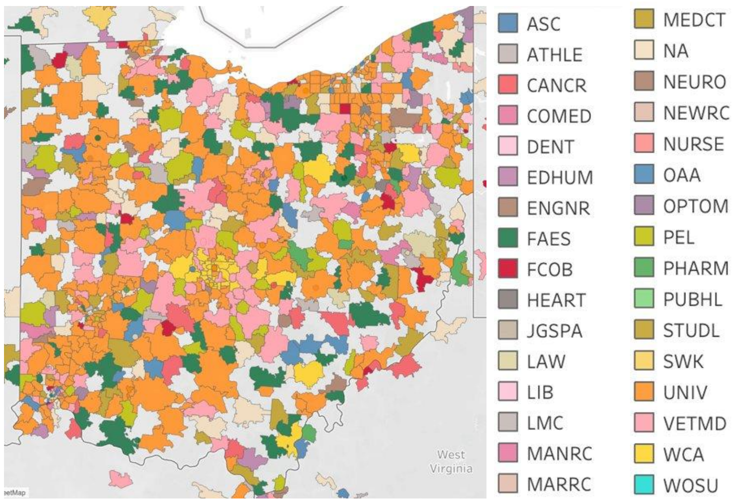


Figure 6: Top Giving Unit by Zip-Code ***excluding WOSU***

When excluding donations to WOSU, there is a notable trend to donate to Wexner Center of the arts in central Ohio. There are also many donations to the University throughout Ohio.

R Code:

```r
# merging datasets and selecting columns
giving <- read_csv("giving.csv")
load("/Users/carolinepier/Downloads/DataConstituent.RData")
library(readxl)
SP2022_Capstone_Project_2_Data_Designation_Info <-
read_excel("/Users/carolinepier/Downloads/SP2022 Capstone - Project 2 - Data - Designation
Info.xlsx")
designation = SP2022_Capstone_Project_2_Data_Designation_Info
# select columns
giving_sub = giving %>% select(CONSTITUENTLOOKUPID, DESIGNATIONLOOKUPID, AMT_DONOR_LT_TOTAL,
N_DONOR_GIFT, AMT_DONOR_GIFT_FIRST)
des_giving = full_join(giving_sub, designation, by = "DESIGNATIONLOOKUPID")
constituent_info3 = DataConstituent %>% select(ID_CONSTITUENTLOOKUPID,
IS_OSUALUMNI,AMT_WEALTH_DEMO_INCOME_RAW_TA, CAT_DONOR_TOPGIVINGUNIT, CAT_CONST_GENDER_CLEAN,
CAT_ADDRESS_ZIP5, CAT_ADDRESS_STATE, CAT_CONSTITUENCY)
# joining dataset
allu_more = full_join(constituent_info3, des_giving, by =
c("ID_CONSTITUENTLOOKUPID"="CONSTITUENTLOOKUPID"))
# random sample to use in Tableau
allu_more_sample = sample_n(allu_more, 25000)
write.csv(allu_more_sample, "allu_more_samp.csv")
```

C. Composition of Different Designation Donors

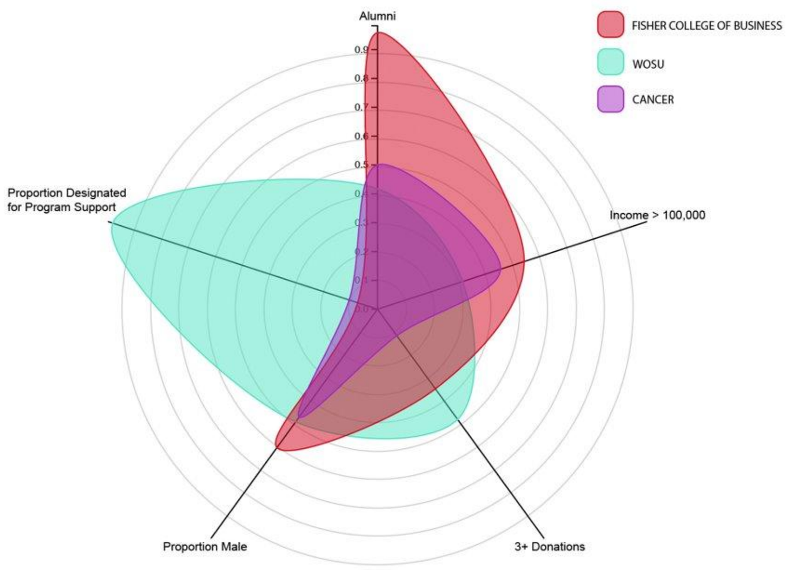


Figure 7: Composition of Different Designation Donors

Fisher College of Business has significantly more alumni donors than the other two categories. The college also has higher income donors and the highest proportion of male donors compared to the other two. WOSU donors have more lifetime donations followed by FCOB followed by Cancer. WOSU donations are often designated for Program Support.

R Code:

```r
#merging datasets and selecting columns
giving <- read_csv("giving.csv")
load("/Users/carolinepier/Downloads/DataConstituent.RData")
library(readxl)
SP2022_Capstone_Project_2_Data_Designation_Info <-
read_excel("/Users/carolinepier/Downloads/SP2022 Capstone - Project 2 - Data - Designation
Info.xlsx")
designation = SP2022_Capstone_Project_2_Data_Designation_Info
#select columns
giving_sub = giving %>% select(CONSTITUENTLOOKUPID, DESIGNATIONLOOKUPID, AMT_DONOR_LT_TOTAL,
N_DONOR_GIFT, AMT_DONOR_GIFT_FIRST)
des_giving = full_join(giving_sub, designation, by = "DESIGNATIONLOOKUPID")
constituent_info3 = DataConstituent %>% select(ID_CONSTITUENTLOOKUPID,
IS_OSUALUMNI,AMT_WEALTH_DEMO_INCOME_RAW_TA, CAT_DONOR_TOPGIVINGUNIT, CAT_CONST_GENDER_CLEAN,
CAT_ADDRESS_ZIP5, CAT_ADDRESS_STATE, CAT_CONSTITUENCY)
#joining dataset
allu = full_join(constituent_info3, des_giving, by =
c("ID_CONSTITUENTLOOKUPID"="CONSTITUENTLOOKUPID"))
wosu = allu %>% filter(cdo == "WOSU")
fcob = allu %>% filter(cdo == "FCOB")
cancer = allu %>% filter(cdo == "CANCR")
#alumni status
table(wosu$IS_OSUALUMNI)
table(fcob$IS_OSUALUMNI)
table(cancer$IS_OSUALUMNI)
#income
table(wosu$AMT_WEALTH_DEMO_INCOME_RAW_TA >100000)
table(fcob$AMT_WEALTH_DEMO_INCOME_RAW_TA >100000)
table(cancer$AMT_WEALTH_DEMO_INCOME_RAW_TA>100000)
```

```
#number of gifts greater than 3
table(wosu$N_DONOR_GIFT >3)
table(fcob$N_DONOR_GIFT >3)
table(cancer$N_DONOR_GIFT >3)
#designated to program support
table(wosu$DESIGNATIONFUND_FUNDPURPOSE == "Program Support")
table(fcob$DESIGNATIONFUND_FUNDPURPOSE == "Program Support")
table(cancer$DESIGNATIONFUND_FUNDPURPOSE == "Program Support")
#gender - proportion male
table(wosu$CAT_CONST_GENDER_CLEAN == "Male")
table(fcob$CAT_CONST_GENDER_CLEAN == "Male")
table(cancer$CAT_CONST_GENDER_CLEAN == "Male")
```

III. Association Rules

Association analysis rules were used to uncover patterns and create recommendations for targeting based on current donation designations and constituents' traits. In order to evaluate the rule strength, a *lift* metric was used. Unlike the *confidence* metric, *lift* accounts for the frequency of the consequence of the rule in the whole dataset. For example, *confidence* could classify rules with donations to cancer funds as significant due to a large proportion of overall donations being made to cancer and Pelotonia funds. However, the *lift* would ensure to rescale the significance based on the frequency of cancer donations among all constituents. A lift value greater than 1 means that the antecedent (left entity) and consequent (right entity) are associated.
To identify "good" rules, first, frequent itemsets are found. Frequent itemsets are the set of characteristics that often appear together (above the specified support threshold). For efficient search, the apriori algorithm was used.

Python Code:
```
constGivingSum = pd.read_csv('SP2022 Capstone - Project 2 - Data - Constituent Designation
Giving Summary.csv', low_memory=False) #designation Giving Summary table
constGivingSum['DESIGNATIONLOOKUPID'] = constGivingSum['DESIGNATIONLOOKUPID'].astype(str)
#ensure ids are of the same type

#remove giving designations with less than 50 donations
freq = constGivingSum['DESIGNATIONLOOKUPID'].value_counts()
freq = freq.loc[freq>=50]
freqGiving =
constGivingSum.loc[constGivingSum['DESIGNATIONLOOKUPID'].isin(freq.index.values.tolist())]

designInfo = pd.read_excel('Data_Designation_Info.xlsx')
#ensure ids are of the same type
designInfo['DESIGNATIONLOOKUPID'] = designInfo['DESIGNATIONLOOKUPID'].astype(str)

designInfoSub = designInfo[['DESIGNATIONLOOKUPID','DESIGNATIONISACTIVE',
'DESIGNATIONFUND_UNIT_CDOCODEalt', 'DESIGNATIONFUND_FUNDPURPOSE']]

df = pd.merge(freqGiving, designInfoSub, left_on="DESIGNATIONLOOKUPID",
right_on="DESIGNATIONLOOKUPID")  #add designations info
df = df.loc[df['DESIGNATIONISACTIVE'] == True] #keep active designations
```

A. Unit code rules

Do people who donate to a specific designation donation unit tend to donate to another designation unit? To explore this, the original data was modified. The Constituent Designation Giving Summary Table was joined with the Designation Information table to identify the designation units constituents donated to. Then a new data of transactions (each constituent corresponded to a list of designation units they donated to) was created. The transactions with one item were removed, as no rule can be created with one item.

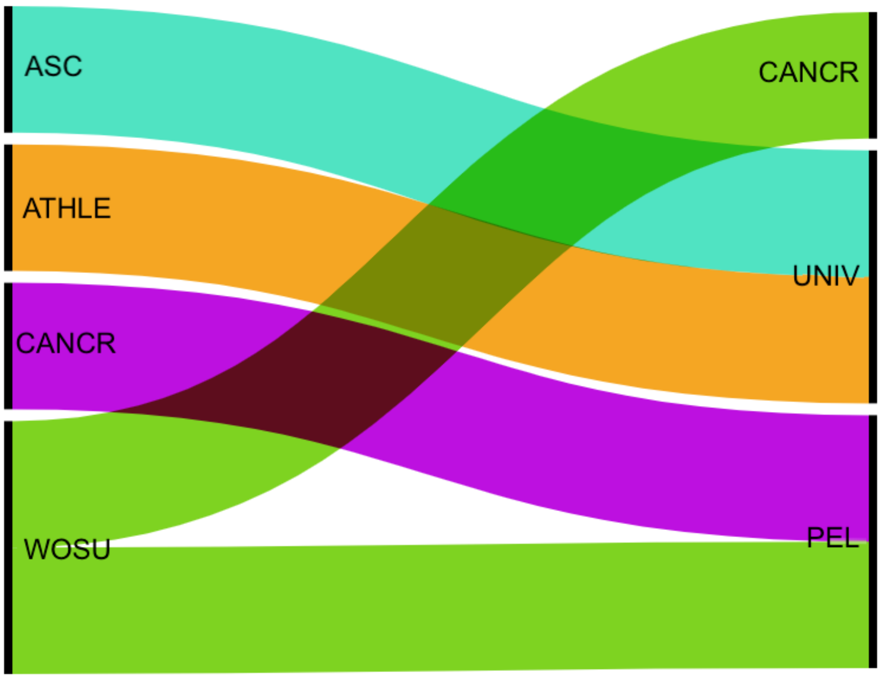


Figure 8: Association Analysis Rules for Designations Unit

The figure above shows that constituents who donate to Arts and Sciences and Athletics funds tend to donate to University funds. People who donate to Cancer funds were found to often donate to Pelotonia. Finally, WOSU donors often contributed to cancer-related funds, including Pelotonia.

Python Code:

```
df_sub = df[['CONSTITUENTLOOKUPID',
       'DESIGNATIONFUND_UNIT_CDOCODEalt']].sort_values(by=['CONSTITUENTLOOKUPID'])
df_new = df_sub.groupby(['CONSTITUENTLOOKUPID'],
                  as_index=False)['DESIGNATIONFUND_UNIT_CDOCODEalt'].agg(lambda x: set(x))
#const and list of designations

# boolean table
df_updated =
df_new['DESIGNATIONFUND_UNIT_CDOCODEalt'].str.join(',').str.get_dummies(sep=',').astype(bool)*
1

#remove rows if constituent only gives to one category
df_updated1 = df_updated[df_updated.sum(axis=1) > 1]
frequent_itemsets = apriori(df_updated1,  min_support=0.1,use_colnames=True) # frequent
itemsets
rules = association_rules(frequent_itemsets, metric="lift", min_threshold = 1)
```

B. Fund purpose rules

Similarly, rules created using the purpose of the fund instead of the designation unit were created.

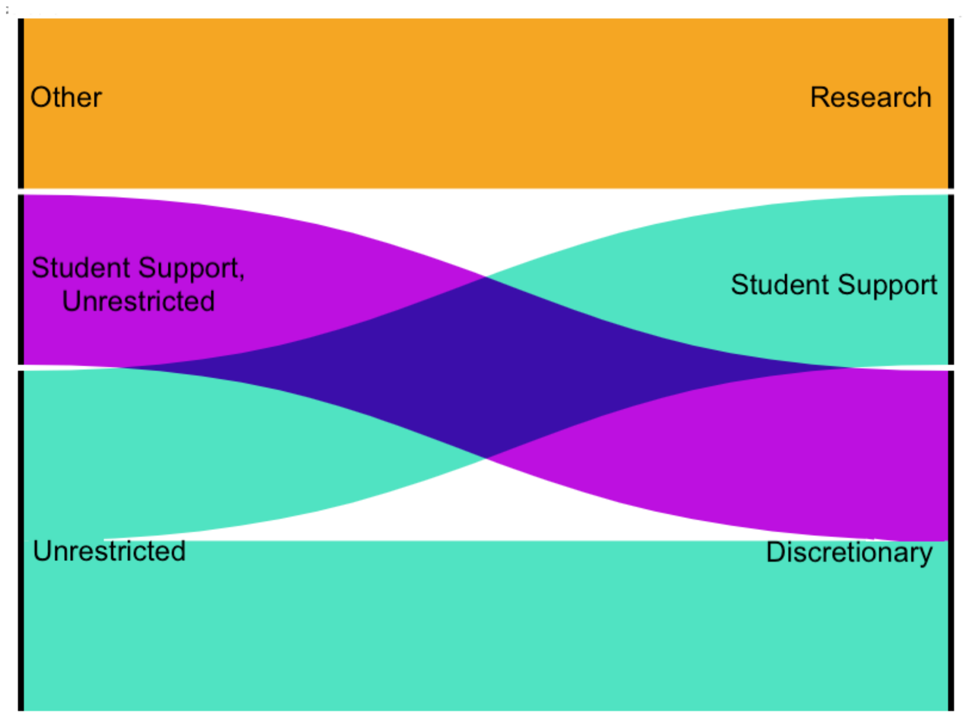


Figure 9: Association Analysis Rules for Fund Purpose

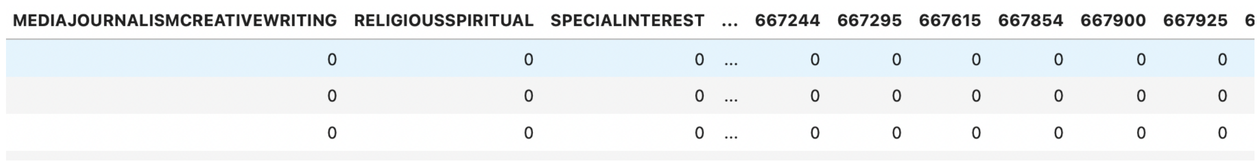
The plot above demonstrated that people who tend to give to funds for purposes other than the ones specified in the data (discretionary, facilities & maintenance, faculty & staff support, program support, research, student support, and unrestricted) tend to also donate to funds that support research. Constituents donating to funds designated for student support and to unrestricted funds (extended at the discretion of the president) also tend to donate to discretionary funds (extended at the discretion of the dean). Lastly, donors for unrestricted funds often donate to student support and discretionary funds.

Python Code:

```
df3 = df[['CONSTITUENTLOOKUPID',
       'DESIGNATIONFUND_FUNDPURPOSE']].sort_values(by=['CONSTITUENTLOOKUPID'])
df3_new = df3.groupby(['CONSTITUENTLOOKUPID'],
                  as_index=False)['DESIGNATIONFUND_FUNDPURPOSE'].agg(lambda x: set(x))
df_updated3 =
df3_new['DESIGNATIONFUND_FUNDPURPOSE'].str.join(',').str.get_dummies(sep=',').astype(bool)*1
df_updated3 = df_updated3[df_updated3.sum(axis=1) > 1]
frequent_itemsets3 = apriori(df_updated3,  min_support=0.1,use_colnames=True)
rules3 = association_rules(frequent_itemsets3, metric="lift", min_threshold = 1.5)
```

C. Student Life Involvement

In addition to fund purposes and unit codes, student life involvement and designation IDs were analyzed to determine if donors who were involved in any activities frequently donated to any funds. Student life involvement refers to the set of variables in the 'Designations' data set labeled 'IS_OSUEDUINVL_STUDENTLIFE_XXX", where 'XXX' is an identifier. This analysis requires that the data set be entirely one-hot encoded, meaning each row is a constituent and each column is either a student life indicator or a designation ID indicator. A new data set was created for this titled “market_basket_data.csv”, and a snapshot of it is included below.



MEDIAJOURNALISMCREATIVEWRITING	RELIGIOUSSPIRITUAL	SPECIALINTEREST	...	667244	667295	667615	667854	667900	667925	6
0	0	0	...	0	0	0	0	0	0	
0	0	0	...	0	0	0	0	0	0	
0	0	0	...	0	0	0	0	0	0	

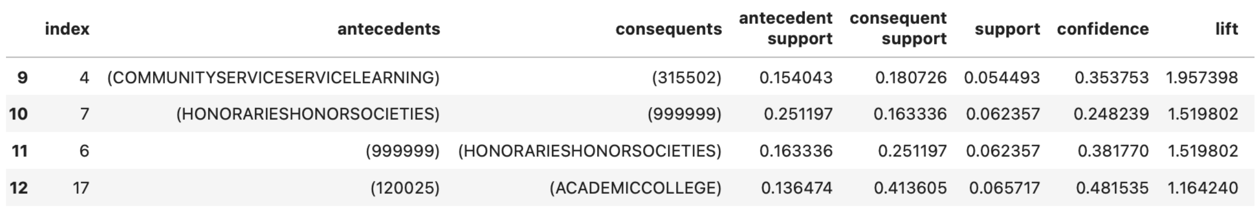
Figure 10: Snapshot of market_basket_data.csv.

The above data was used to create association rules using the apriori algorithm. Each student life variable and designation ID were considered to be items. A support threshold of 5% was used, meaning that a fund and a variable had to occur 5% of the time in order to be considered (their columns needed 1s in them 5% of the time). This produced an intermediate set of association rules. A snapshot is shown in Figure 11.

Python Code:
```
# load required libraries
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules
import pandas as pd
import numpy as np

designations = pd.read_csv('market_basket_data.csv')
designations.drop(columns = ['constituent', 'Unnamed: 0'], inplace = True)
# get frequent itemsets
frequent = apriori(designations, min_support = 0.05, use_colnames =
True).sort_values('support').reset_index()

# Get first set of association rules
full_rules = association_rules(frequent, metric = 'lift').sort_values('lift', ascending =
False).reset_index()
```



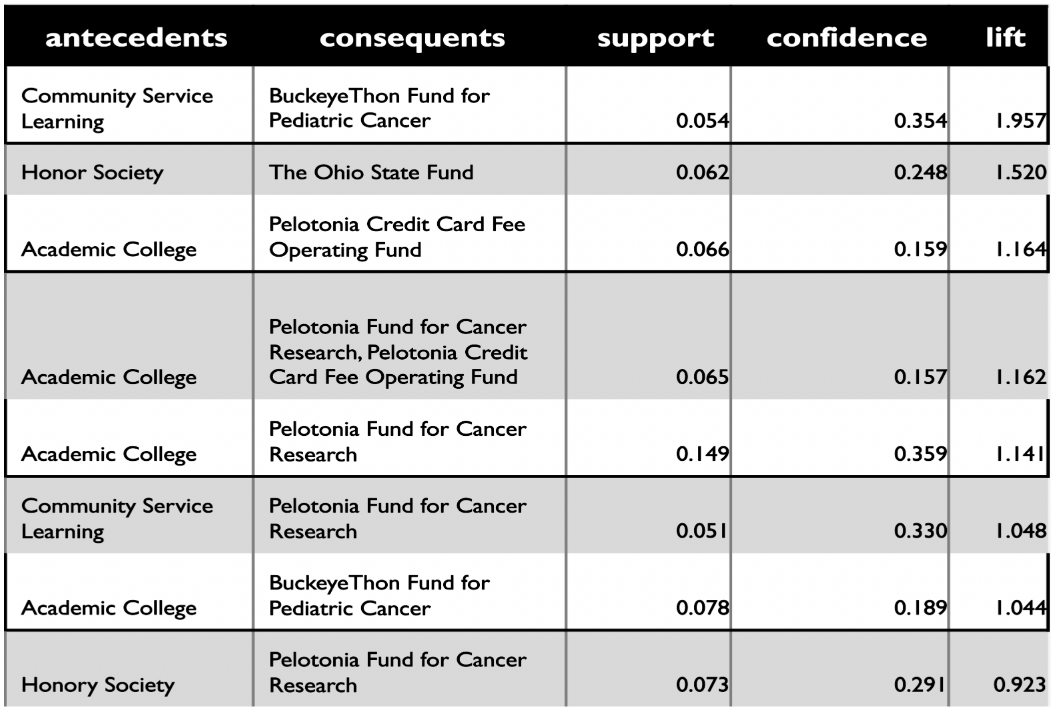
	index	antecedents	consequents	antecedent support	consequent support	support	confidence	lift
9	4	(COMMUNITYSERVICESERVICELEARNING)	(315502)	0.154043	0.180726	0.054493	0.353753	1.957398
10	7	(HONORARIESHONORSOCIETIES)	(999999)	0.251197	0.163336	0.062357	0.248239	1.519802
11	6	(999999)	(HONORARIESHONORSOCIETIES)	0.163336	0.251197	0.062357	0.381770	1.519802
12	17	(120025)	(ACADEMICCOLLEGE)	0.136474	0.413605	0.065717	0.481535	1.164240

Figure 11: Snapshot of intermediate student life association rules.

In this context, it does not make sense for designations to be antecedents or for student life indicators to be consequents, because most donors likely attended OSU *before* they donated to a fund. Therefore, the first set of rules was filtered, allowing us to only consider rules student life indicators are antecedents and designations are consequents.

Python Code:
```
variables = ['ACADEMICCOLLEGE', 'AWARENESSACTIVISM',
       'COMMUNITYSERVICESERVICELEARNING', 'CREATIVEANDPERFORMINGARTS',
       'ETHNICCULTURAL', 'GOVERNANCEORGANIZATIONS', 'HONORARIESHONORSOCIETIES',
        'MEDIAJOURNALISMCREATIVEWRITING', 'RELIGIOUSSPIRITUAL',
        'SPECIALINTEREST', 'SPORTSANDRECREATION', 'TECHNOLOGY']
idx_set = list()
for i in range(len(full_rules)):
    row = full_rules.loc[i]
    all_letters = True
    for a in row['antecedents']:
        if a not in variables:
            all_letters = False
    all_numbers = True
    for c in row['consequents']:
        if c in variables:
            all_numbers = False
    if all_letters and all_numbers:
        idx_set.append(i)

# Rules where the student life variable is the antecedent,
# the designation ID is the consequent,
# and the lift is greater than one
valid_rules = full_rules.loc[idx_set]
```



antecedents	consequents	support	confidence	lift
Community Service Learning	BuckeyeThon Fund for Pediatric Cancer	0.054	0.354	1.957
Honor Society	The Ohio State Fund	0.062	0.248	1.520
Academic College	Pelotonia Credit Card Fee Operating Fund	0.066	0.159	1.164
Academic College	Pelotonia Fund for Cancer Research, Pelotonia Credit Card Fee Operating Fund	0.065	0.157	1.162
Academic College	Pelotonia Fund for Cancer Research	0.149	0.359	1.141
Community Service Learning	Pelotonia Fund for Cancer Research	0.051	0.330	1.048
Academic College	BuckeyeThon Fund for Pediatric Cancer	0.078	0.189	1.044
Honory Society	Pelotonia Fund for Cancer Research	0.073	0.291	0.923

Figure 12: Final set of student life association rules. Note that designation IDs were replaced with fund names.

Based on the associations in Figure 12, the first 7 rows are useful as their lift values are greater than 1.

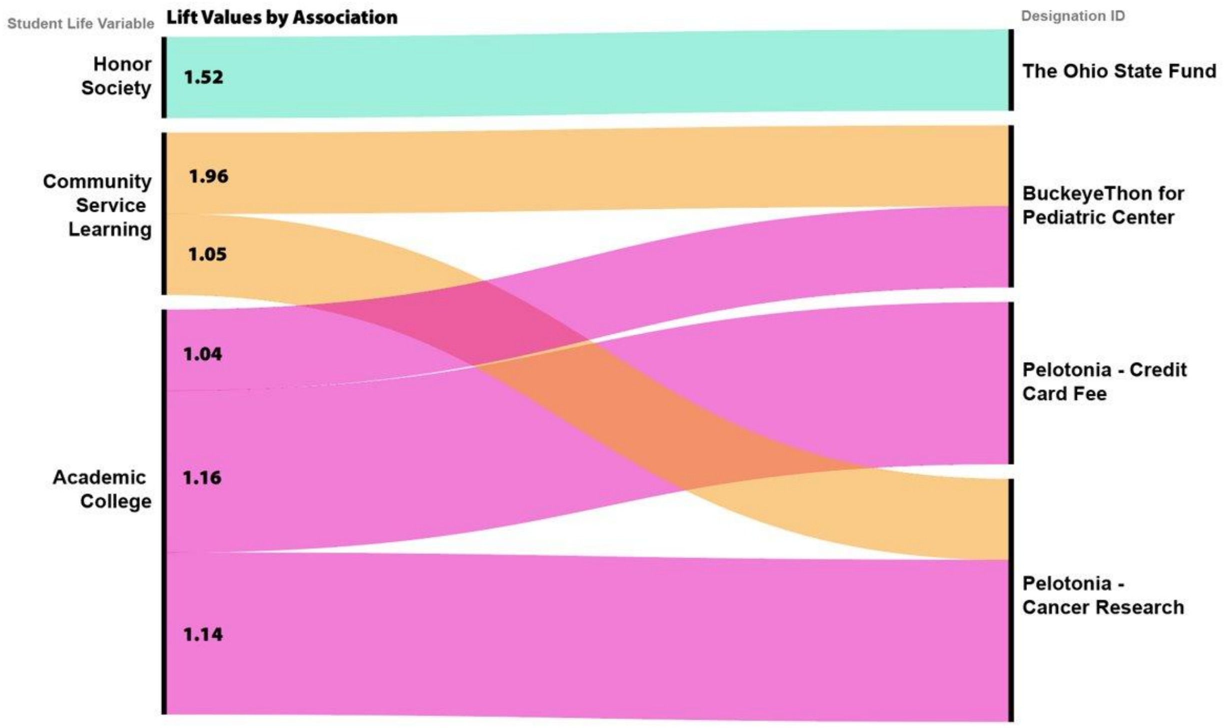


Figure 13: Student life association rules having lift values greater than 1. The antecedents are on the left, with paths leading to the consequent(s) they are associated with on the right.

The above figure describes our recommendations regarding student life involvement and designations. If someone was involved in the honor society, we suggest advertising The Ohio State Fund. In addition, we suggest marketing the BuckeyeThon and the Pelotonia to individuals who participated in community service learning. Finally, persuade people who went to an academic college to donate to the BuckeyeThon, or the Pelotonia.

IV. Pelotonia

Pelotonia was focused on because it had the most donors. From this we decided to choose this because it would maximize the potential amount being donated to the cause, in addition we had a good amount of data to draw conclusions from. Overall the model had a 57% accuracy at successfully determining whether someone has donated and someone who hasn't donated and will never donate.

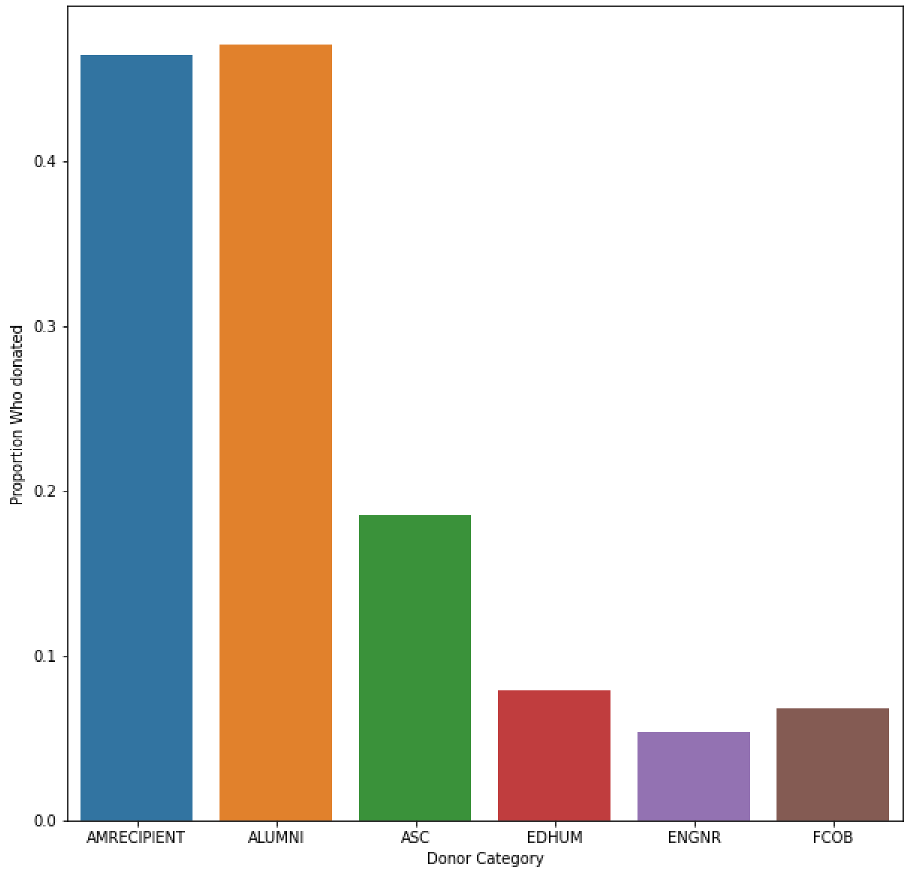


Figure 14: Demographic breakdown for all people who have donated to Pelotonia. Of those people, what demographic groups were represented by at least 5% of donors.

The purpose of this visualization is to show the breakdown of the top 6 most popular demographic categories for Pelotonia donors. This shows us that among the donors most of them shared the following characteristics (either together or separately): being an Alumni Magazine recipient, being an Alumni, Arts and Sciences Graduate, Education and Human Ecology Graduate, Fisher Graduate.

Python Code:

```
constituent_bools = pd.read_csv("designBoolean.csv")


consts  = dt.fread("~\\Downloads\\SP2022 Capstone - Project 2 - Data -
Constituents.csv").to_pandas()

merged_df_of_consts_bools_and_analysis_consts2 =
consts[["ID_CONSTITUENTLOOKUPID","CAT_ADDRESS_STATE","IS_OSAMRECIPIENT_EVER","IS_MEMBER_PC","I
S_OSUALUMNI","IS_ON_OSUBOARD_EVER","IS_OSUATTENDEE","IS_VETMDCLIENT_EVER","N_EVENTS","IS_DEGRE
E_COLLEGECDO_ASC","IS_DEGREE_COLLEGECDO_COMED","IS_DEGREE_COLLEGECDO_DENT","IS_DEGREE_COLLEGEC
DO_EDHUM","IS_DEGREE_COLLEGECDO_ENGNR","IS_DEGREE_COLLEGECDO_FAES","IS_DEGREE_COLLEGECDO_FCOB"
,"IS_DEGREE_COLLEGECDO_JGSPA","IS_DEGREE_COLLEGECDO_LAW","IS_DEGREE_COLLEGECDO_NURSE","IS_DEGR
EE_COLLEGECDO_OAA","IS_DEGREE_COLLEGECDO_OPTOM","IS_DEGREE_COLLEGECDO_PHARM","IS_DEGREE_COLLEG
ECDO_PUBHL","IS_DEGREE_COLLEGECDO_SWK","IS_DEGREE_COLLEGECDO_UNIV","IS_DEGREE_COLLEGECDO_VETMD
","IS_OSUEDUINVL_STUDENTLIFE_ACADEMICCOLLEGE","IS_OSUEDUINVL_STUDENTLIFE_AWARENESSACTIVISM","I
S_OSUEDUINVL_STUDENTLIFE_COMMUNITYSERVICESERVICELEARNING","IS_OSUEDUINVL_STUDENTLIFE_CREATIVEA
NDPERFORMINGARTS","IS_OSUEDUINVL_STUDENTLIFE_ETHNICCULTURAL","IS_OSUEDUINVL_STUDENTLIFE_GOVERN
ANCEORGANIZATIONS","IS_OSUEDUINVL_STUDENTLIFE_HONORARIESHONORSOCIETIES","IS_OSUEDUINVL_STUDENT
LIFE_MEDIAJOURNALISMCREATIVEWRITING","IS_OSUEDUINVL_STUDENTLIFE_RELIGIOUSSPIRITUAL","IS_OSUEDU
INVL_STUDENTLIFE_SPECIALINTEREST","IS_OSUEDUINVL_STUDENTLIFE_SPORTSANDRECREATION","IS_OSUEDUIN
VL_STUDENTLIFE_TECHNOLOGY"]].merge(constituent_bools,how="inner",left_on="ID_CONSTITUENTLOOKUP
ID",right_index=True)

pel_df =
merged_df_of_consts_bools_and_analysis_consts2[merged_df_of_consts_bools_and_analysis_consts2.
PEL == 1]

inds = [i for i,v in enumerate(ends_to_means) if v > .05]
temp_3 =
np.array(["IS_OSAMRECIPIENT_EVER","IS_MEMBER_PC","IS_OSUALUMNI","IS_ON_OSUBOARD_EVER","IS_OSUA
TTENDEE","IS_VETMDCLIENT_EVER","IS_DEGREE_COLLEGECDO_ASC","IS_DEGREE_COLLEGECDO_COMED","IS_DEG
REE_COLLEGECDO_DENT","IS_DEGREE_COLLEGECDO_EDHUM","IS_DEGREE_COLLEGECDO_ENGNR","IS_DEGREE_COLL
EGECDO_FAES","IS_DEGREE_COLLEGECDO_FCOB","IS_DEGREE_COLLEGECDO_JGSPA","IS_DEGREE_COLLEGECDO_LA
W","IS_DEGREE_COLLEGECDO_NURSE","IS_DEGREE_COLLEGECDO_OAA","IS_DEGREE_COLLEGECDO_OPTOM","IS_DE
GREE_COLLEGECDO_PHARM","IS_DEGREE_COLLEGECDO_PUBHL","IS_DEGREE_COLLEGECDO_SWK","IS_DEGREE_COLL
EGECDO_UNIV","IS_DEGREE_COLLEGECDO_VETMD","IS_OSUEDUINVL_STUDENTLIFE_ACADEMICCOLLEGE","IS_OSUE
DUINVL_STUDENTLIFE_AWARENESSACTIVISM","IS_OSUEDUINVL_STUDENTLIFE_COMMUNITYSERVICESERVICELEARNI
NG","IS_OSUEDUINVL_STUDENTLIFE_CREATIVEANDPERFORMINGARTS","IS_OSUEDUINVL_STUDENTLIFE_ETHNICCUL
TURAL","IS_OSUEDUINVL_STUDENTLIFE_GOVERNANCEORGANIZATIONS","IS_OSUEDUINVL_STUDENTLIFE_HONORARI
ESHONORSOCIETIES","IS_OSUEDUINVL_STUDENTLIFE_MEDIAJOURNALISMCREATIVEWRITING","IS_OSUEDUINVL_ST
UDENTLIFE_RELIGIOUSSPIRITUAL","IS_OSUEDUINVL_STUDENTLIFE_SPECIALINTEREST","IS_OSUEDUINVL_STUDE
NTLIFE_SPORTSANDRECREATION","IS_OSUEDUINVL_STUDENTLIFE_TECHNOLOGY"])
inds_necc = temp_3[inds]

final_ends_to_means = []
for i in inds_necc:
    final_ends_to_means.append(pel_df[i].mean())

sns.set_theme(style="darkgrid")
plt.figure(figsize=(10, 10))
plt.xlabel("Donor Category")
plt.ylabel("Proportion Who Donated")
sns.barplot(x=["AMRECIPIENT","ALUMNI","ASC","EDHUM","ENGNR","FCOB"],y= final_ends_to_means)
sns.set(font_scale = 2)
help(sns.set)
```

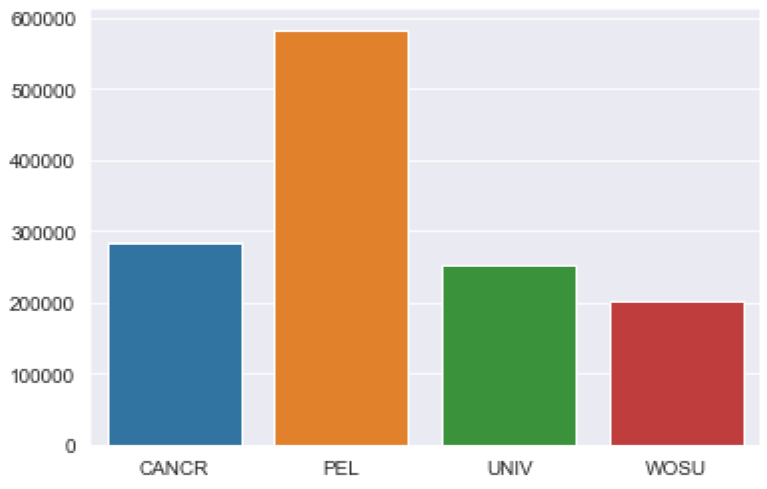


Figure 15: Top 4 Fund Designations by Amount of Donors

This shows the reason why I chose to focus on Pelotonia because of how many people donated and the amount of data there is.
Overall for Pelotonia, the top designations that should be focused on are: Fisher, Arts & Sciences, Education and Human Ecology, and FCOB. In addition to whether they are alumni and Alumni recipients.

Python Code:
```
fund_ser = pd.read_csv("designBoolean.csv").sum(axis=0)
fund_ser = fund_ser[fund_ser>200000]

sns.set_style("darkgrid")
sns.barplot(x=fund_ser.keys(),y=fund_ser.values)

from sklearn.model_selection import train_test_split
X,y =
merged_df_of_consts_bools_and_analysis_consts2[["IS_OSUALUMNI","IS_OSAMRECIPIENT_EVER","IS_DEG
REE_COLLEGECDO_ASC","IS_DEGREE_COLLEGECDO_EDHUM","IS_DEGREE_COLLEGECDO_ENGNR","IS_DEGREE_COLLE
GECDO_FCOB"]],merged_df_of_consts_bools_and_analysis_consts2["PEL"]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=42)

from sklearn.linear_model import LogisticRegression
clf = LogisticRegression(random_state=0).fit(X_train, y_train)
y_pred=clf.predict(X_test)
clf.predict_proba(X_test)

clf.score(X_test,y_test)
```

V. Logistic regression

A. Data Preparation

1. Response and Predictors

Response variable is the fund unit, ‘cdo’ code in the dataset. There are plenty of funds, this report only selects the top five funds for analysis. The following graph was created in Tableau, showing the number of appearances of each fund in the dataset. Predictors are all the donor’s degree, and their relationship to OSU, which predictors will be selected based on boosting and lasso later in the passage.

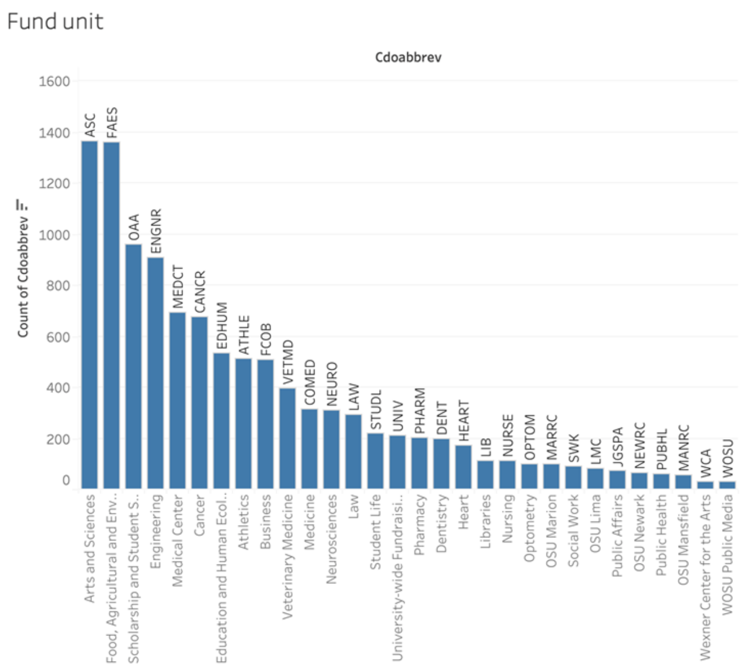


Figure 16: Top fund unit

R Code:
```
Designation_Info_merge = Designation_Info[Designation_Info$IS_DESIGNATION_FOR_ANALYSIS == 1,]
Designation_Info_merge = Designation_Info[,c(1,7:12)]
Designation_Info_merge = Designation_Info[Designation_Info_merge$cdo ==
"ASC"|Designation_Info_merge$cdo == "FAES"|Designation_Info_merge$cdo ==
"OAA"|Designation_Info_merge$cdo == "ENGNR"|Designation_Info_merge$cdo == "MEDCT",]
DataConstituents_merge = DataConstituents[,c(1,109:122,130:134,143,321:338)]
DC_Giving_Summary_merge = DC_Giving_Summary[,c(1:2)]
```

2. Merging Data

Predictors are in the ‘Data Constituens’ dataset, and response is in the ‘Donor Giving Summary’ dataset. There is no common key to combine these two datasets. So, the ‘Designation Info’ was used to merge ‘Data Constituens’ and ‘Donor Giving Summary’. The common key between ‘Designation Info’ and ‘Donor Giving Summary’ is *‘DESIGNATIONLOOKUPID’*. The common key between ‘Designation Info’ and ‘Data Constituens’ is *‘CONSTITUENTLOOKUPID’* and ‘*ID_CONSTITUENTLOOKUPID*’.

R Code:
```
ij = Designation_Info %>% inner_join(DC_Giving_Summary_merge, by = "DESIGNATIONLOOKUPID")
ij_new = ij %>% distinct(CONSTITUENTLOOKUPID, .keep_all = TRUE)
new_data_combined = merge(ij_new,DataConstituents_merge,by.x = "CONSTITUENTLOOKUPID", by.y =
"ID_CONSTITUENTLOOKUPID",all.x = TRUE)
new_data_combined = new_data_combined[new_data_combined$cdo == "ASC"|new_data_combined$cdo ==
"FAES"|new_data_combined$cdo == "OAA"|new_data_combined$cdo == "ENGNR"|new_data_combined$cdo
== "MEDCT",]
new_data_combined = new_data_combined %>% distinct(CONSTITUENTLOOKUPID, .keep_all = TRUE)
```

3. Data Transformation

This is the final step before building the model. Ture and false values will be converted to 1 and 0. Missing values will be deleted, and subsets will be created.

R Code:

```
new_data_combined$ASC = ifelse(new_data_combined$cdo == "ASC", 1,0)
new_data_combined$FAES = ifelse(new_data_combined$cdo == "FAES", 1,0)
new_data_combined$OAA = ifelse(new_data_combined$cdo == "OAA", 1,0)
new_data_combined$ENGNR = ifelse(new_data_combined$cdo == "ENGNR", 1,0)
new_data_combined$MEDCT = ifelse(new_data_combined$cdo == "MEDCT", 1,0)
new_data_combined2 =na.omit(new_data_combined2)
sum(is.na(new_data_combined2))
cols = c("N_OSUBOARD_EVER","IS_DEGREE_COLLEGECDO_ASC","IS_DEGREE_COLLEGECDO_COMED" ,
"IS_DEGREE_COLLEGECDO_DENT" , "IS_DEGREE_COLLEGECDO_EDHUM", "IS_DEGREE_COLLEGECDO_ENGNR"
,"IS_DEGREE_COLLEGECDO_FAES" , "IS_DEGREE_COLLEGECDO_FCOB" , "IS_DEGREE_COLLEGECDO_JGSPA"
,"IS_DEGREE_COLLEGECDO_LAW" ,  "IS_DEGREE_COLLEGECDO_NURSE" ,"IS_DEGREE_COLLEGECDO_OAA"
,"IS_DEGREE_COLLEGECDO_OPTOM", "IS_DEGREE_COLLEGECDO_PHARM" ,"IS_DEGREE_COLLEGECDO_PUBHL"
,"IS_DEGREE_COLLEGECDO_SWK"  , "IS_DEGREE_COLLEGECDO_UNIV" , "IS_DEGREE_COLLEGECDO_VETMD"
,"IS_OSUEDUINVL_ATHLETICS" ,"ASC" ,"FAES" ,"OAA" ,"ENGNR" ,"MEDCT")

new_data_combined2[cols] = lapply(new_data_combined2[cols],factor)
new_data_combined2 = new_data_combined2[,-14]
new_data_combined2 = new_data_combined2[,-c(10,14,15,19,20,23:25)]

new_data_combined2$IS_OSUEMPLOYEE_EVER[new_data_combined2$IS_OSUEMPLOYEE_EVER == "TRUE"] <- 1
new_data_combined2$IS_OSUEMPLOYEE_EVER[new_data_combined2$IS_OSUEMPLOYEE_EVER == "FALSE"] <- 0

new_data_combined2$IS_OSUSTAFF_EVER[new_data_combined2$IS_OSUSTAFF_EVER == "TRUE"] <- 1
new_data_combined2$IS_OSUSTAFF_EVER[new_data_combined2$IS_OSUSTAFF_EVER == "FALSE"] <- 0

new_data_combined2$IS_OSUFACULTY_EVER[new_data_combined2$IS_OSUFACULTY_EVER == "TRUE"] <- 1
new_data_combined2$IS_OSUFACULTY_EVER[new_data_combined2$IS_OSUFACULTY_EVER == "FALSE"] <- 0

new_data_combined2$IS_ON_OSUBOARD_EVER[new_data_combined2$IS_ON_OSUBOARD_EVER == "TRUE"] <- 1
new_data_combined2$IS_ON_OSUBOARD_EVER[new_data_combined2$IS_ON_OSUBOARD_EVER == "FALSE"] <- 0

new_data_combined2$IS_OSUALUMSPOUSE[new_data_combined2$IS_OSUALUMSPOUSE == "TRUE"] <- 1
new_data_combined2$IS_OSUALUMSPOUSE[new_data_combined2$IS_OSUALUMSPOUSE == "FALSE"] <- 0

new_data_combined2$IS_REACHABLE[new_data_combined2$IS_REACHABLE == "TRUE"] <- 1
new_data_combined2$IS_REACHABLE[new_data_combined2$IS_REACHABLE == "FALSE"] <- 0

new_data_combined2$IS_BUCKEYEROOMMEMBER[new_data_combined2$IS_BUCKEYEROOMMEMBER == "TRUE"] <-
1
new_data_combined2$IS_BUCKEYEROOMMEMBER[new_data_combined2$IS_BUCKEYEROOMMEMBER == "FALSE"] <-
0

new_data_combined2$IS_OSUVOLUNTEER_EVER[new_data_combined2$IS_OSUVOLUNTEER_EVER == "TRUE"] <-
1
new_data_combined2$IS_OSUVOLUNTEER_EVER[new_data_combined2$IS_OSUVOLUNTEER_EVER == "FALSE"] <-
0

new_data_combined2$IS_OSUALUMNI[new_data_combined2$IS_OSUALUMNI == "TRUE"] <- 1
new_data_combined2$IS_OSUALUMNI[new_data_combined2$IS_OSUALUMNI == "FALSE"] <- 0

new_data_combined2$IS_OSUATTENDEE[new_data_combined2$IS_OSUATTENDEE == "TRUE"] <- 1
new_data_combined2$IS_OSUATTENDEE[new_data_combined2$IS_OSUATTENDEE == "FALSE"] <- 0

new_data_combined2$IS_OSUPARENT_EVER[new_data_combined2$IS_OSUPARENT_EVER == "TRUE"] <- 1
```

```
new_data_combined2$IS_OSUPARENT_EVER[new_data_combined2$IS_OSUPARENT_EVER == "FALSE"] <- 0

ASC = new_data_combined2[,-c(1:8,39:42)]
FAES = new_data_combined2[,-c(1:8,38,40:42)]
OAA = new_data_combined2[,-c(1:8,38,39,41:42)]
ENGNR = new_data_combined2[,-c(1:8,38:40,42)]
MEDCT = new_data_combined2[,-c(1:8,38:41)]
```

B. Variable Selection

Before building the logistic regression, this project will first select variables with more predictive power. Two variable selection model was considered, boosting and lasso regression. Boosting was selected because it can provides clear visualization with bars showing the magnitude of the importance. Lasso regression was selected because it can purn the coefficients of in-significant variables to 0. The two models were build to on the same scale to make comparisons and decision on the variables to use for the logistic model. Five booting and lasso regression was build on the five fund unit. Figure.17 shows the relative influence of variables for each of the model. Specific Coefficents from lasso regression was not provided here, but can be found by running the R code. After comparisons, results from lasso matches with the results from booting, where the variables with positive lasso coefficients showed the high relative influence in the boosting. In this case, we will use the results from lasso regression to make decisions on what variables to select from. To be more specific, variables with positive lasso coefficients will be selected and used fo the logistic regression. Positive lasso coefficients indicates the variables are positively contributing to the response. Thus, the reason for using positive lasso coefficients for logistic regression is that the model only want variables that will cause this person to donate the fund, but don't want variables that will cause the person not donate to the fund.

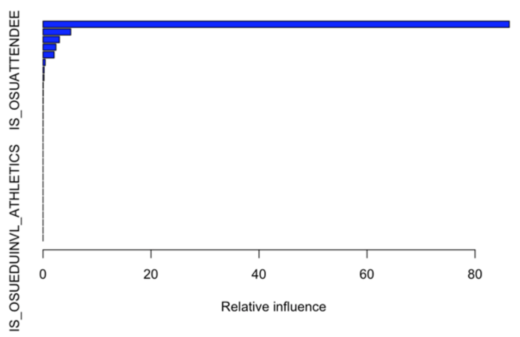


Figure 17.1: Relative Influence of giving to ASC

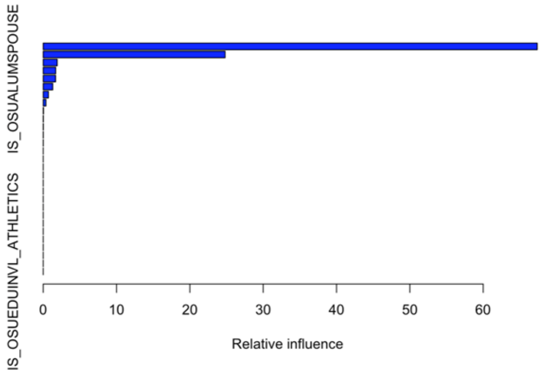
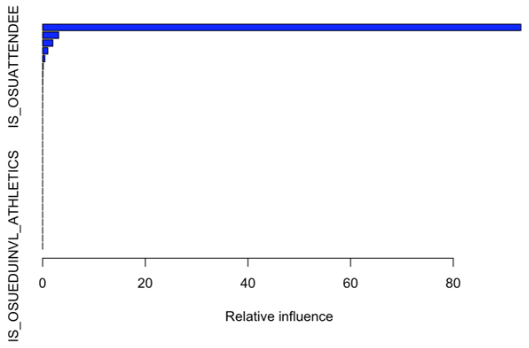


Figure 17.2: Relative Influence of giving to ASC



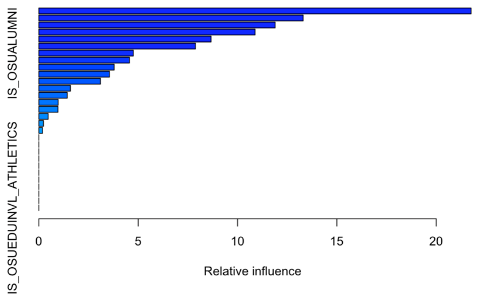


Figure 17.2: Relative Influence of giving to ENGR ***Figure 17.3:*** Relative Influence of giving to OAA

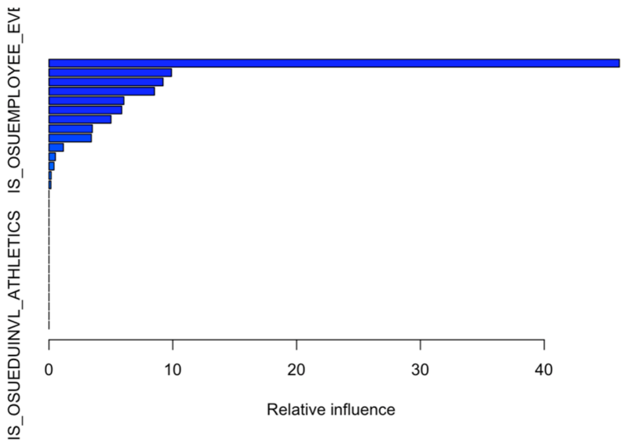


Figure 18: Relative Influence of giving to MEDCT

R Code:

```
boot.ASC = gbm(ASC == 1~., data = ASC, distribution = "bernoulli", n.trees = 50,
interaction.depth = 3)
summary(boot.ASC)

boot.FAES = gbm(FAES == 1~., data = FAES, distribution = "bernoulli", n.trees = 50,
interaction.depth = 3)
summary(boot.FAES)

boot.ENGNR = gbm(ENGNR == 1~., data = ENGNR, distribution = "bernoulli", n.trees = 50,
interaction.depth = 3)
summary(boot.ENGNR)

boot.OAA = gbm(OAA == 1~., data = OAA, distribution = "bernoulli", n.trees = 50,
interaction.depth = 3)
summary(boot.OAA)

boot.MEDCT = gbm(MEDCT == 1~., data = MEDCT, distribution = "bernoulli", n.trees = 50,
interaction.depth = 3)
summary(boot.MEDCT)
lasso.mod.cv.ASC= cv.glmnet(x = data.matrix(ASC[dt,-30]), y = factor(ASC$ASC),alpha=1, family
= "binomial")
lasso.mod.FAES= cv.glmnet(x = data.matrix(FAES[,-30]), y = factor(FAES$FAES),alpha=1, family =
"binomial")
coef(lasso.mod.FAES)
lasso.mod.ENGNR= cv.glmnet(x = data.matrix(ENGNR[,-30]), y = factor(ENGNR$ENGNR),alpha=1,
family = "binomial")
coef(lasso.mod.ENGNR)
lasso.mod.OAA= cv.glmnet(x = data.matrix(OAA[,-30]), y = factor(OAA$OAA),alpha=1, family =
"binomial")
coef(lasso.mod.OAA)
lasso.mod.MEDCT= cv.glmnet(x = data.matrix(MEDCT[,-30]), y = factor(MEDCT$MEDCT),alpha=1,
family = "binomial")
coef(lasso.mod.MEDCT)
```

C. Insights from Variable Selection

An interesting finding during the variable selection analysis is that only few, two or three, variables have the high relative influence and predictive power to the ASC, FEAS, and ENGNR, but there are many important predictors for funds to OAA and MEDCT. Insights we can from here is that funds to OAA and MEDCT are the two popular funds among the five, where people from different backgrounds donates to these funds. On the contrary, for the other three funds, only people who graduated from that college tend to donate to the fund. The following visualizations were built in Adobe Illustrator, which shows the predictors with positive lasso coefficients in each model, and the size of each circle is the magnitude of the coefficient. So, the larger the circle indicates the higher predictive power of the variable.

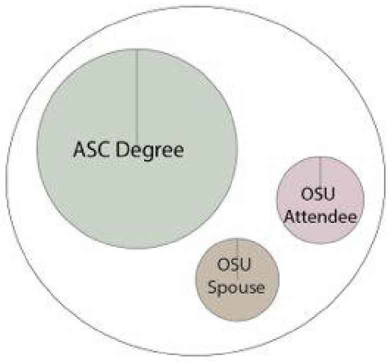


Figure 19.1 Variables with Positive Lasso Coefficients of Giving to ASC

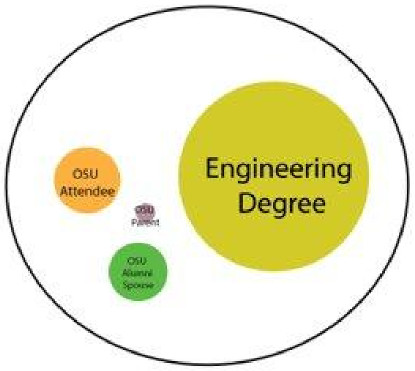


Figure 19.2. Variables with Positive Lasso Coefficients of Giving to ENGNR

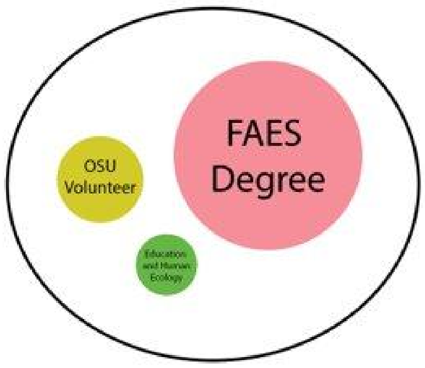


Figure 19.3. Variables with Positive Lasso Coefficients of Giving to FAES

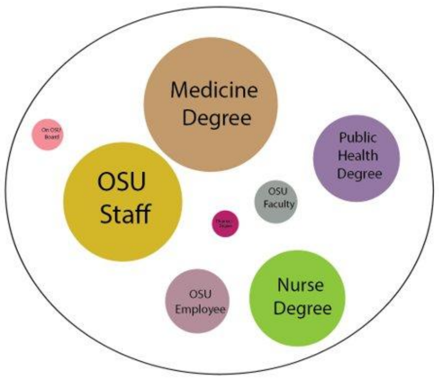


Figure 19.4. Variables with Positive Lasso Coefficients of Giving to MEDCT

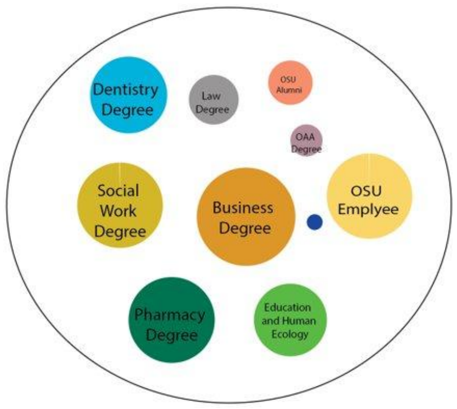


Figure 19.5. Variables with Positive Coefficients of Giving to OAA

D. Prediction From Logistic Regression

Five logistic regression was built based on the variables with positive lasso coefficients. This project further predicts the conditional probability based on the logistic regression to get insights from the result. Taking giving to OAA as an example, if a person has a social worker degree, then he/she has 0.329 more odds of donating to the OAA, then it is reasonable to recommend OAA funds to people who have a social worker degree. If a person has a social worker degree and is an OSU employee, then he/she has 0.688 more odds of donating to the OAA, then it is strongly recommended to introduce OAA funds to such groups of people. The following table shows one of the insights from the logistic regression.

R Code:

```
glm.ASC1 = glm(ASC ~ IS_DEGREE_COLLEGECDO_ASC + IS_OSUATTENDEE + IS_OSUALUMSPOUSE +
IS_OSUEMPLOYEE_EVER + IS_OSUSTAFF_EVER , family = "binomial", data = ASC[dt,])

exp(glm.ASC1$coefficients)/(1+exp(glm.ASC1$coefficients))
exp(glm.ASC$coefficients)/(1+exp(glm.ASC$coefficients))

glm.FAES = glm(FAES ~ IS_DEGREE_COLLEGECDO_FAES + IS_DEGREE_COLLEGECDO_EDHUM +
IS_OSUVOLUNTEER_EVER, family = "binomial", data = FAES)
exp(glm.FAES$coefficients)

glm.ENGNR = glm(ENGNR ~ IS_OSUATTENDEE + IS_OSUALUMSPOUSE + IS_OSUPARENT_EVER +
IS_DEGREE_COLLEGECDO_ENGNR, family = "binomial", data = ENGNR)
exp(glm.ENGNR$coefficients)

glm.OAA = glm(OAA ~ as.factor(IS_OSUEMPLOYEE_EVER) + as.factor(IS_OSUALUMNI)
+IS_DEGREE_COLLEGECDO_DENT + IS_DEGREE_COLLEGECDO_EDHUM + IS_DEGREE_COLLEGECDO_FCOB +
IS_DEGREE_COLLEGECDO_LAW + IS_DEGREE_COLLEGECDO_OAA + IS_DEGREE_COLLEGECDO_PHARM +
IS_DEGREE_COLLEGECDO_SWK+IS_DEGREE_COLLEGECDO_VETMD, family = "binomial", data = OAA)

log_odds = predict(glm.OAA, data.frame("IS_OSUEMPLOYEE_EVER"= as.factor(as.character(0)),
"IS_OSUALUMNI"=as.factor(as.character(1)),
"IS_DEGREE_COLLEGECDO_DENT"=as.factor(as.character(0)),"IS_DEGREE_COLLEGECDO_EDHUM"=
as.factor(as.character(0)), "IS_DEGREE_COLLEGECDO_FCOB"=as.factor(as.character(0)),
"IS_DEGREE_COLLEGECDO_LAW"=as.factor(as.character(0)),"IS_DEGREE_COLLEGECDO_OAA"=
as.factor(as.character(0)), "IS_DEGREE_COLLEGECDO_PHARM"=as.factor(as.character(0)),
"IS_DEGREE_COLLEGECDO_SWK"=as.factor(as.character(0)),"IS_DEGREE_COLLEGECDO_VETMD"=
as.factor(as.character(0))))

exp(log_odds)/(1+exp(log_odds))

glm.MEDCT = glm(MEDCT ~ IS_OSUEMPLOYEE_EVER + IS_OSUALUMNI +IS_DEGREE_COLLEGECDO_DENT +
IS_DEGREE_COLLEGECDO_EDHUM + IS_DEGREE_COLLEGECDO_FCOB + IS_DEGREE_COLLEGECDO_LAW +
IS_DEGREE_COLLEGECDO_OAA + IS_DEGREE_COLLEGECDO_PHARM +
IS_DEGREE_COLLEGECDO_SWK+IS_DEGREE_COLLEGECDO_VETMD, family = "binomial", data = MEDCT)
exp(glm.OAA$coefficients)
```

VI. User-based Collaborative recommendation system:

Introduction

The basic idea behind the recommendation system is that similar people would always have similar tastes and incentives for the same thing. This filtering algorithm was commonly used by internet companies to recommend their specific products and services to the appointed group of people based on their historical preferences and similarities among each group. For example, the algorithm would explore the personal information of their users and their interactive scores with companies' products like ratings, comments or likes. Based on these kinds of data, the algorithm would be able to divide the whole user pool into different groups with similar personal information, preferences and consuming histories. Then, the recommendation system would pick up items interacted by most similar users with our input user in the same group to be the guide list for recommending. One of goals for our project is to give our constituents further donation advice, namely advised designations. Thus, the procedure to achieve this would be nearly the same with that of the recommendation system. Here, constituents are just users, and the interactive scores with products like rating, comments etc could be substituted by the constituents' donation amount to the specific designation.

Data preparation

In the r code part, Constituent_Designation, Designation and Constituent dataset are loaded into the r environment. Then, desired variables of each dataset are filtered. The final_df is the outcome of merging these 3 dataframes with selected variables, which are also used as the input dataset for building the recommendation system.

R code:

```
library(dplyr)
library(tidyverse)

#Load data
load("/Users/wusongyuan/Downloads/constituent_new (4).RData")

#Read 3 data sets
inter_df = data.frame(Constituent_Designation)
desig_df = data.frame(Designation_dataset)
const = data.frame(DataConstituent)
```

```
#Data filtering for each dataset
desig_df = select(desig_df,c("DESIGNATIONLOOKUPID","cdo"))
inter_df =
select(inter_df,c("CONSTITUENTLOOKUPID","DESIGNATIONLOOKUPID","AMT_DONOR_LT_TOTAL"))
const_df = select(const, c(1,20,21,22,33:35,49:61,103:105,109:122,123:127,190:191,223))

df2 <- merge(x=df_const, y=df_3k99k, by.x="ID_CONSTITUENTLOOKUPID",
by.y="CONSTITUENTLOOKUPID",all=FALSE)

merged_df = merge(x=const_df, y=inter_df, by.x="ID_CONSTITUENTLOOKUPID",
by.y="CONSTITUENTLOOKUPID",all=FALSE)
head(merged_df)

final_df = merge(x=merged_df, y=desig_df, by.x="DESIGNATIONLOOKUPID",
by.y="DESIGNATIONLOOKUPID", all=FALSE)
head(final_df)
write.csv(final_df,"final_df02.csv")
```

Python code:

```
import pandas as pd
import numpy as np
from sklearn.metrics.pairwise import cosine_similarity
import operator

const_file_path = '/content/drive/MyDrive/final_df02.csv'
const_df = pd.read_csv(const_file_path, encoding='cp1252')
```

Donation matrix contains information about the detailed donation amount for each constituent to the specific designation. Just as emphasized before, this information could be treated as the interactive scores which would be used by the recommendation system to rank the order of similar constituents in the same group at the end.

```
donation_matrix = const_df.iloc[:,np.r_[1:3,47:48]]
```

Hybrid recommendation engine was used here for better performance of the recsys. The whole dataset was divided into 4 small sub datasets, which were treated as one individual engine to calculate the similarity scores among constituents respectively. Attributes incorporating same kind of information of our constituents would be integrated into one small sub dataset. 4 datasets and their selected variables are shown below:

Events_donation_dataset: 'N_EVENTS', 'N_EVENTS_DATES', 'N_EVENTS_FYS', 'FY_EVENTS_FIRST', 'FY_EVENTS_LAST'
Consistency dataset: 'VAL_DONOR_LT_CONSISTENCY', 'N_DONOR_LT_CONSECFYS',
'N_DONOR_LT_GIVINGFYS', 'VAL_DONOR_AGE_ATFIRSTGIFT',
'N_DONOR_INCREASEDGIVINGYEARS_TIMEPERIOD',
'N_DONOR_DECREASEDGIVINGYEARS_TIMEPERIOD'
Payment method dataset: 'IS_DONOR_PAY_STOCK', 'IS_DONOR_PAY_CASH_OLD', 'IS_DONOR_PAY_CASH_TAS',
'IS_DONOR_PAY_CHECK', 'IS_DONOR_PAY_CHECK_BUSINESS',
'IS_DONOR_PAY_CREDITCARD', 'IS_DONOR_PAY_CREDITCARD_RECUR',
'IS_DONOR_PAY_TRANSFER', 'IS_DONOR_PAY_TRANSFER_RECUR',
'IS_DONOR_PAY_MATCHINGGIFT', 'IS_DONOR_PAY_GIK', 'IS_DONOR_PAY_PAYROLL',
'IS_DONOR_PAY_IRA'
Relationship w osu dataset: 'IS_MEMBER_PC', 'IS_MEMBER_BC', 'IS_OSUEMPLOYEE_EVER',
'IS_OSUFACSTAFF_EVER', 'IS_OSUSTAFF_EVER', 'IS_OSUFACULTY_EVER',

'IS_ON_OSUBOARD_EVER', 'N_OSUBOARD_EVER', 'IS_ON_OSUBOARD_ALUM_EVER', 'IS_ON_OSUBOARD_NONALUM_EVER', 'IS_OSUALUMNI', 'IS_OSUATTENDEE', 'IS_OSUALUMSPOUSE', 'IS_TWOALUMHH', 'IS_VETMDCLIENT_EVER', 'IS_OSUPARENT_EVER', 'IS_DEGREE_ANY', 'VAL_DEGREE_YOG_FIRST', 'VAL_SCORE_ENGAGEMENT'

Python code:

```
"This dataframe incoporating the events_donation information"
events_donation = const_df.iloc[:,np.r_[1:3,39:44]]
events_donation = events_donation.replace(np.nan,0)
events_donation = events_donation.set_index("ID_CONSTITUENTLOOKUPID")
events_donation = events_donation.drop(['DESIGNATIONLOOKUPID'], axis = 1)
events_donation = events_donation.sort_index(axis = 0)
events_donation.head()

"This dataframe incoporating the consistency information"
consistency = const_df.iloc[:,np.r_[1:3,6:9,22:25]]
consistency = consistency.replace(np.nan,0)
consistency = consistency.set_index("ID_CONSTITUENTLOOKUPID")
consistency = consistency.drop(['DESIGNATIONLOOKUPID'], axis = 1)
consistency = consistency.sort_index(axis = 0)
consistency.head()

"This dataframe incoporating the payment approach"
payment_method = const_df.iloc[:,np.r_[1:3,9:22]]
payment_method = payment_method.replace(np.nan,0)
payment_method = payment_method.set_index("ID_CONSTITUENTLOOKUPID")
payment_method = payment_method.drop(['DESIGNATIONLOOKUPID'], axis = 1)
payment_method = payment_method.sort_index(axis = 0)
payment_method = payment_method.replace({True:1, False:0})
payment_method.head()

"This dataframe incoporating the relationship with osu"
rela = const_df.iloc[:,np.r_[1:3,3:5,25:39,44:47]]
rela = rela.replace(np.nan,0)
rela = rela.replace({True:1, False:0})
rela = rela.set_index("ID_CONSTITUENTLOOKUPID")
rela = rela.drop(['DESIGNATIONLOOKUPID'], axis = 1)
rela = rela.sort_index(axis = 0)
rela.head()

matrixlist = []
similarity_score_list = []
matrixlist.append(events_donation)
matrixlist.append(consistency)
matrixlist.append(payment_method)
matrixlist.append(rela)
len(matrixlist)
```

The input parameters of find_similar_users are our input constituent unique id, similarity matrices generated from 4 datasets shown above and desired amount of similar constituents what developer would find in our database. 4 similarity matrices would be combined in the naive weights to generate one final similarity matrix containing the similarity scores for all pairs of constituents. Then, the function would only return a list of the top constituents id by ranking order of their similarity scores with the input constituent.

```
def find_similar_users(user_id, matrixlist, similarity_score_list, k=3):
 indices = list()
 for i in range(4):
   matrix = matrixlist[i]
   user = matrix[matrix.index == user_id]
   other_users = matrix[matrix.index != user_id]
   similarity_score = cosine_similarity(user,other_users)[0].tolist()
   similarity_score_list.append(similarity_score)
   indices = other_users.index.tolist()
 similarity_score_list[0] = [element * 0.3 for element in similarity_score_list[0]]
 similarity_score_list[1] = [element * 0.1 for element in similarity_score_list[1]]
 similarity_score_list[2] = [element * 0.2 for element in similarity_score_list[2]]
 similarity_score_list[3] = [element * 0.4 for element in similarity_score_list[3]]
 similarities = [a + b + c + d for a, b, c, d in
zip(similarity_score_list[0],similarity_score_list[1],

similarity_score_list[2],similarity_score_list[3])]
 print("similarities list looks like:")
 print(similarities)
 index_similarity = dict(zip(indices, similarities))
 index_similarity
 index_similarity_sorted = sorted(index_similarity.items(), key=operator.itemgetter(1))
 index_similarity_sorted.reverse()
 index_similarity_sorted
 top_users_similarities = index_similarity_sorted[:k]
 users = [u[0] for u in top_users_similarities]
 return users
```

The id of the input constituent in the experiment is 51556, one list of 3 most similar constituents [14730, 20332, 68621] were found by the system.

```
current_user = 51556
# try it out
similar_user_indices = find_similar_users(current_user, matrixlist, similarity_score_list)
print("3 most similar users are:")
print(similar_user_indices)

donation_matrix = donation_matrix.set_index("ID_CONSTITUENTLOOKUPID")
donation_matrix.head()
similar_users = donation_matrix[donation_matrix.index.isin(similar_user_indices)]

similar_users_matrix = similar_users.pivot_table(index='ID_CONSTITUENTLOOKUPID',
columns='DESIGNATIONLOOKUPID', values='DonationAmount')
similar_users_matrix = similar_users_matrix.replace(np.nan,0)
similar_users_matrix.head()

user_matrix = donation_matrix[donation_matrix.index == current_user]
user_matrix.head()

user_matrix = user_matrix.pivot_table(index='ID_CONSTITUENTLOOKUPID',
columns='DESIGNATIONLOOKUPID', values='DonationAmount')
user_matrix.head()

similar_users = similar_users_matrix.mean(axis=0)
similar_users
similar_users_df = pd.DataFrame(similar_users, columns=['mean'])
similar_users_df.head()
```

```
def recommend_item(user_index, similar_user_indices, user_matrix, similar_user_matrix,
items=5):

   similar_users = similar_users_matrix.mean(axis=0)
   similar_users_df = pd.DataFrame(similar_users, columns=['mean'])

   user_df_transposed = user_matrix.transpose()
   # rename the column as 'Donation amount'
   user_df_transposed.columns = ['donationAmount']
   # remove any rows without a 0 value. Designations not donated yet
   user_df_transposed = user_df_transposed[user_df_transposed['donationAmount']==0]
   # generate a list of designations the constituent has not donated
   designation_undonated = user_df_transposed.index.tolist()

   # filter avg ratings of similar users for only anime the current user has not seen
   similar_users_df_filtered =
similar_users_df[similar_users_df.index.isin(designation_undonated)]
   # order the data frame
   similar_users_df_ordered = similar_users_df.sort_values(by=['mean'], ascending=False)
   # grab the top n designations
   top_n_designations = similar_users_df_ordered.head(items)
   top_n_designation_indices = top_n_designations.index.tolist()

   designation_information =
const_df[const_df['DESIGNATIONLOOKUPID'].isin(top_n_designation_indices)]
   designation_information = designation_information.reset_index(drop=True)
   designation_information =
designation_information.drop_duplicates("cdo",keep='last').reset_index(drop=True)
   return designation_information["cdo"]#items

recommend_item(current_user, similar_user_indices, user_matrix, similar_users_matrix)
```

By inputting one constituent id, the recommendation system could find the most k similar constituents with the input constituent in the database. Then, donation amounts to the specific designation for these k similar constituents would be used as weights to help give top recommendations for the input constituent. For instance, 3 similar constituents would possibly interact with 10 more designations. However, the system should remove designations which the input constituent has donated to before, and pick up top k designations based on the donation amount level of each designation. In the experiment, our recommendation system would recommend COMED, NEURO and MEDCT when the input user id is 51556.

VII. Results and Conclusions

In conclusion, our recommendation systems help to identify designation types to recommend to donors. We analyzed variables to recommend donor designations based on donor history, finding variables such as giving amount and giving history valuable in distinguishing different designations. Different donation combinations were more popular than others. Certain locations had distinguishing popular designations. We also analyzed donor demographics. Our models are able to output suggested designation types.