MuscleHub A/B Test

Patrick Tsai

Overview

Current membership funnel:

1. Take a fitness test with a personal trainer
2. Fill out an application for the gym
3. Send in payment to complete 1st month's membership

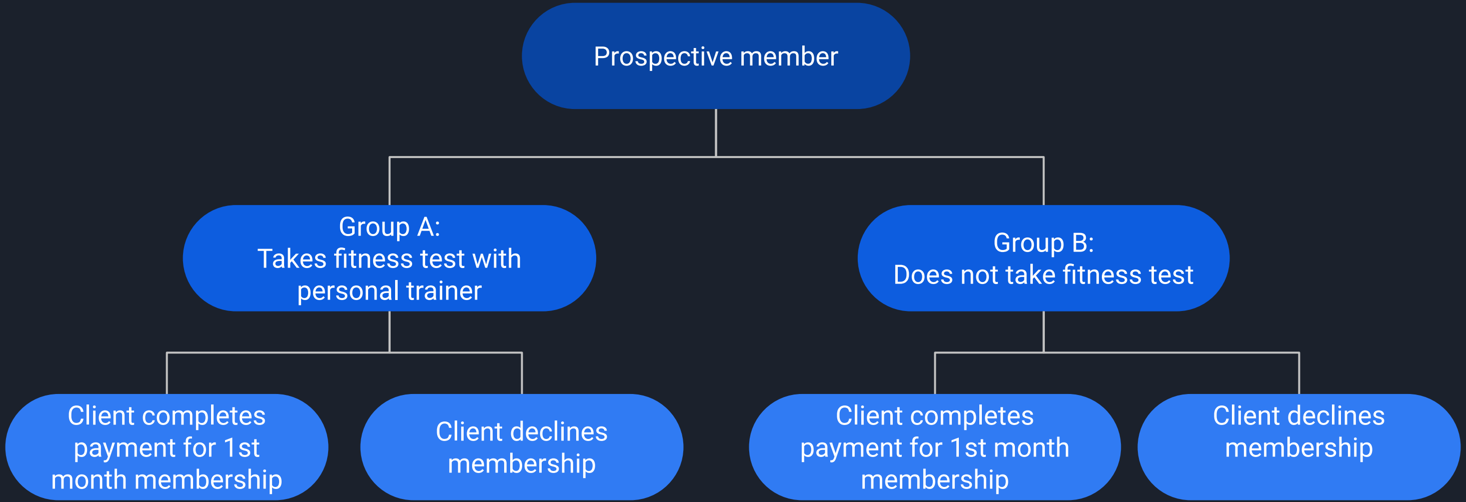
My task:

- MuscleHub's manager feels that the fitness test may be intimidating to prospective clients
- As an analyst, I will run an A/B test to see which group attracts more customers to purchase a membership

Experimental groups:

- Group A will follow the current membership funnel outlined above, and complete the fitness test
- Group B will skip the fitness test, and proceed directly to step 2 (application)

A/B Test: Membership funnel



Prospective member

Group A:
Takes fitness test with personal trainer

Client completes payment for 1st month membership

Client declines membership

Group B:
Does not take fitness test

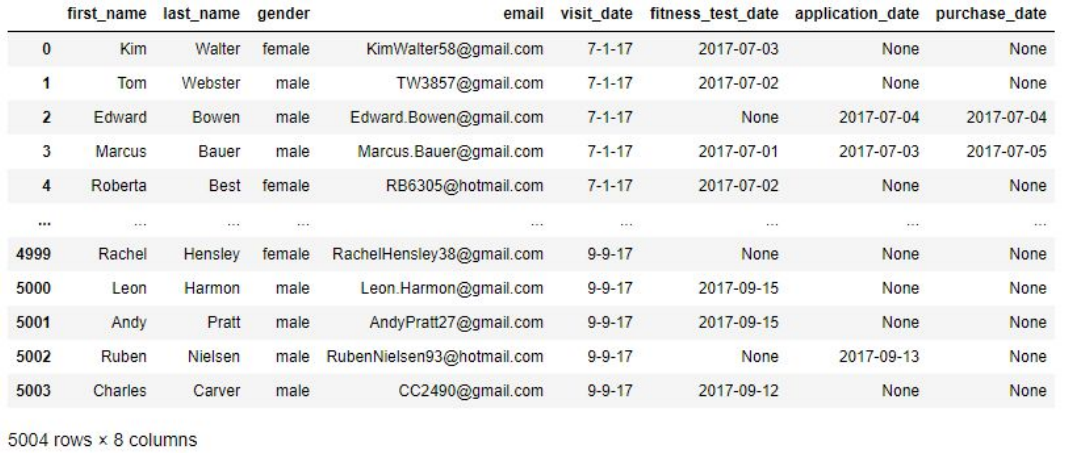
Client completes payment for 1st month membership

Client declines membership

Obtaining data from SQLite database

```
SELECT v.first_name, v.last_name, v.gender,
       v.email, v.visit_date, ft.fitness_test_date,
       a.application_date, p.purchase_date
FROM visits AS 'v'
LEFT JOIN fitness_tests AS 'ft'
    ON v.email = ft.email
    AND v.first_name = ft.first_name
    AND v.last_name = ft.last_name
LEFT JOIN applications AS 'a'
    ON v.email = a.email
    AND v.first_name = a.first_name
    AND v.last_name = a.last_name
LEFT JOIN purchases AS 'p'
    ON v.email = p.email
    AND v.first_name = p.first_name
    AND v.last_name = p.last_name
WHERE v.visit_date >= "7-1-17";
```

- Performed a series of LEFT JOINs to merge the 4 tables of relevant consumer data, and added a WHERE clause to filter client visits that occurred prior to the A/B test start date (7/1/2017)



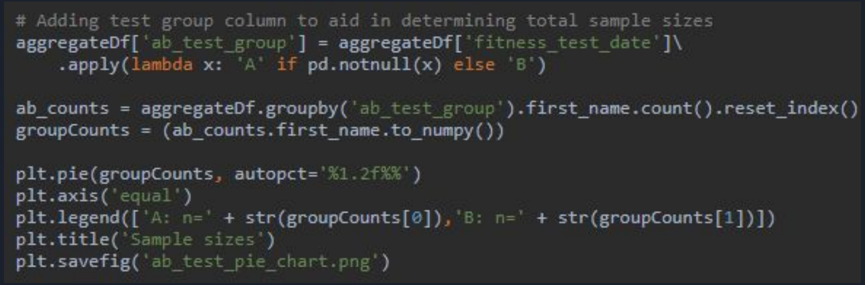
	first_name	last_name	gender	email	visit_date	fitness_test_date	application_date	purchase_date
0	Kim	Walter	female	KimWalter58@gmail.com	7-1-17	2017-07-03	None	None
1	Tom	Webster	male	TW3857@gmail.com	7-1-17	2017-07-02	None	None
2	Edward	Bowen	male	Edward.Bowen@gmail.com	7-1-17	None	2017-07-04	2017-07-04
3	Marcus	Bauer	male	Marcus.Bauer@gmail.com	7-1-17	2017-07-01	2017-07-03	2017-07-05
4	Roberta	Best	female	RB6305@hotmail.com	7-1-17	2017-07-02	None	None
...	...	...	...	...	...	...	...	...
4999	Rachel	Hensley	female	RachelHensley38@gmail.com	9-9-17	None	None	None
5000	Leon	Harmon	male	Leon.Harmon@gmail.com	9-9-17	2017-09-15	None	None
5001	Andy	Pratt	male	AndyPratt27@gmail.com	9-9-17	2017-09-15	None	None
5002	Ruben	Nielsen	male	RubenNielsen93@hotmail.com	9-9-17	None	2017-09-13	None
5003	Charles	Carver	male	CC2490@gmail.com	9-9-17	2017-09-12	None	None

5004 rows × 8 columns

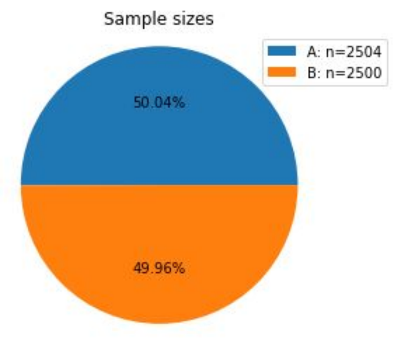
*Datasets used for this project are fictional data provided by Codeacademy

Analyzing the metric choices

Invariant metric used for sanity check: sample sizes of groups A and B



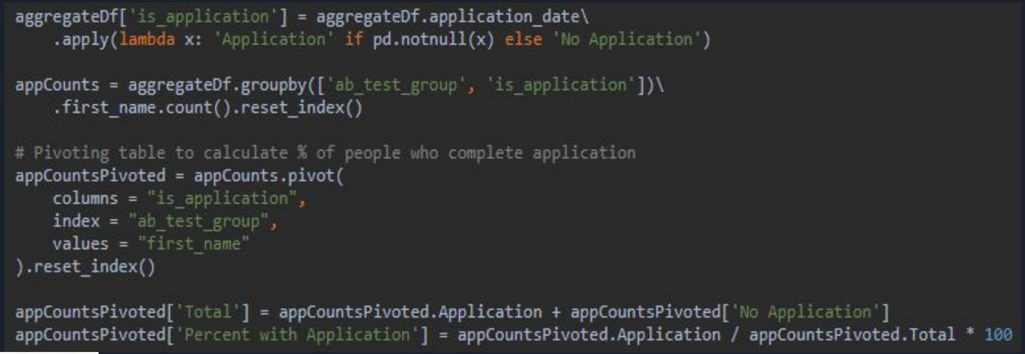
```
# Adding test group column to aid in determining total sample sizes
aggregateDf['ab_test_group'] = aggregateDf['fitness_test_date']\
    .apply(lambda x: 'A' if pd.notnull(x) else 'B')

ab_counts = aggregateDf.groupby('ab_test_group').first_name.count().reset_index()
groupCounts = (ab_counts.first_name.to_numpy())

plt.pie(groupCounts, autopct='%1.2f%%')
plt.axis('equal')
plt.legend(['A: n=' + str(groupCounts[0]),'B: n=' + str(groupCounts[1])])
plt.title('Sample sizes')
plt.savefig('ab_test_pie_chart.png')
```



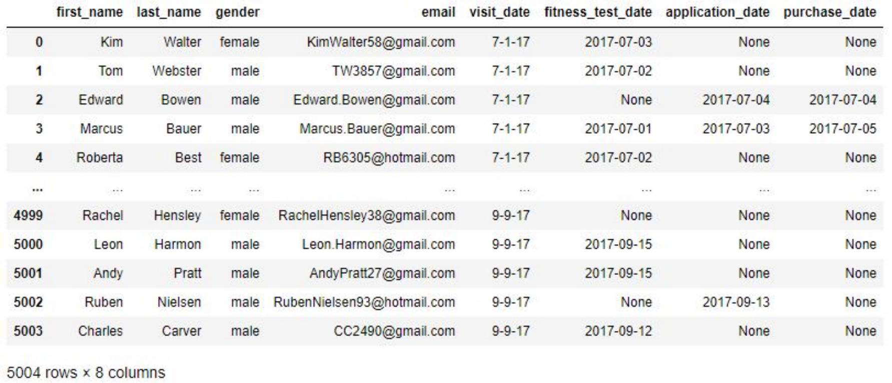
Evaluation metrics used as performance indications:

Metric Name	Metric Formula
Gross Application Conversion	$\frac{\#\ \textit{visitors who complete an application}}{\textit{total}\ \#\ \textit{visitors}}$
Gross Membership Conversion	$\frac{\#\ \textit{visitors who purchase a membership}}{\#\ \textit{visitors who complete an application}}$
Net Conversion	$\frac{\#\ \textit{visitors who purchase a membership}}{\textit{total}\ \#\ \textit{visitors}}$

Determining who completes an application

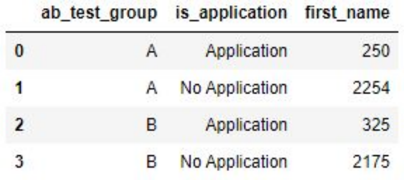


```
aggregateDf['is_application'] = aggregateDf.application_date\
    .apply(lambda x: 'Application' if pd.notnull(x) else 'No Application')

appCounts = aggregateDf.groupby(['ab_test_group', 'is_application'])\
    .first_name.count().reset_index()

# Pivoting table to calculate % of people who complete application
appCountsPivoted = appCounts.pivot(
    columns = "is_application",
    index = "ab_test_group",
    values = "first_name"
).reset_index()

appCountsPivoted['Total'] = appCountsPivoted.Application + appCountsPivoted['No Application']
appCountsPivoted['Percent with Application'] = appCountsPivoted.Application / appCountsPivoted.Total * 100
```

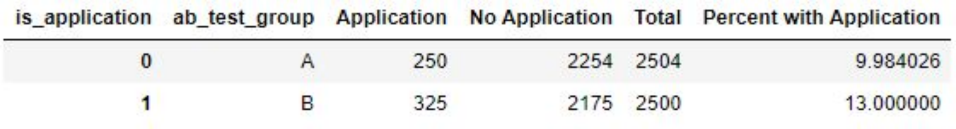


	first_name	last_name	gender	email	visit_date	fitness_test_date	application_date	purchase_date
0	Kim	Walter	female	KimWalter58@gmail.com	7-1-17	2017-07-03	None	None
1	Tom	Webster	male	TW3857@gmail.com	7-1-17	2017-07-02	None	None
2	Edward	Bowen	male	Edward.Bowen@gmail.com	7-1-17	None	2017-07-04	2017-07-04
3	Marcus	Bauer	male	Marcus.Bauer@gmail.com	7-1-17	2017-07-01	2017-07-03	2017-07-05
4	Roberta	Best	female	RB6305@hotmail.com	7-1-17	2017-07-02	None	None
...	...	...	...	...	...	...	...	...
4999	Rachel	Hensley	female	RachelHensley38@gmail.com	9-9-17	None	None	None
5000	Leon	Harmon	male	Leon.Harmon@gmail.com	9-9-17	2017-09-15	None	None
5001	Andy	Pratt	male	AndyPratt27@gmail.com	9-9-17	2017-09-15	None	None
5002	Ruben	Nielsen	male	RubenNielsen93@hotmail.com	9-9-17	None	2017-09-13	None
5003	Charles	Carver	male	CC2490@gmail.com	9-9-17	2017-09-12	None	None

5004 rows × 8 columns



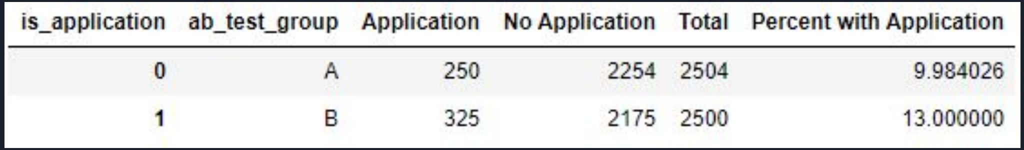
	ab_test_group	is_application	first_name
0	A	Application	250
1	A	No Application	2254
2	B	Application	325
3	B	No Application	2175



is_application	ab_test_group	Application	No Application	Total	Percent with Application
0	A	250	2254	2504	9.984026
1	B	325	2175	2500	13.000000

Determining if our observed difference is statistically significant

- Since we are determining if there is a statistical difference between categorical variables in the same population (whether or not a new client becomes a member or non-member depending on which sample group they belong to), the chi2 contingency test is most appropriate

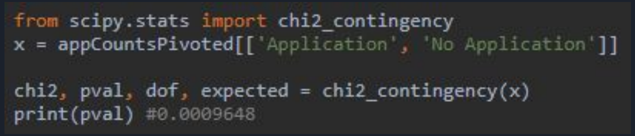


is_application	ab_test_group	Application	No Application	Total	Percent with Application
0	A	250	2254	2504	9.984026
1	B	325	2175	2500	13.000000

Null Hypothesis:

There is no association between the percentage of new applicants and whether or not they underwent a fitness test

Analysis:

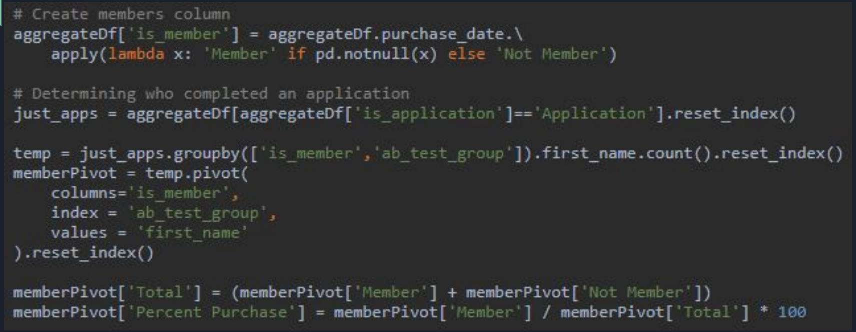


```
from scipy.stats import chi2_contingency
x = appCountsPivoted[['Application', 'No Application']]

chi2, pval, dof, expected = chi2_contingency(x)
print(pval) #0.0009648
```

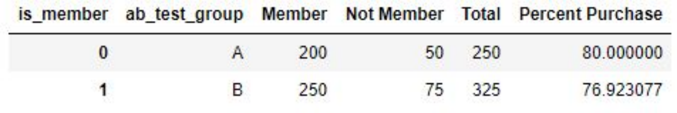
Since our p-value 0.00096 is less than our significance value (alpha=0.05), we conclude that the results are statistically significant, and therefore reject the null hypothesis

Conclusion: The percentage of new applicants is dependent on whether or not they undergo a fitness test prior!

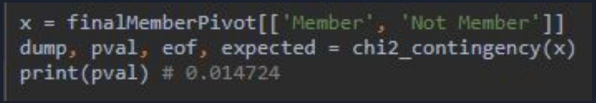
Determining and analyzing percentage of applicants who purchase membership



```
# Create members column
aggregateDf['is_member'] = aggregateDf.purchase_date.\
    apply(lambda x: 'Member' if pd.notnull(x) else 'Not Member')

# Determining who completed an application
just_apps = aggregateDf[aggregateDf['is_application']=='Application'].reset_index()

temp = just_apps.groupby(['is_member','ab_test_group']).first_name.count().reset_index()
memberPivot = temp.pivot(
    columns='is_member',
    index = 'ab_test_group',
    values = 'first_name'
).reset_index()

memberPivot['Total'] = (memberPivot['Member'] + memberPivot['Not Member'])
memberPivot['Percent Purchase'] = memberPivot['Member'] / memberPivot['Total'] * 100
```



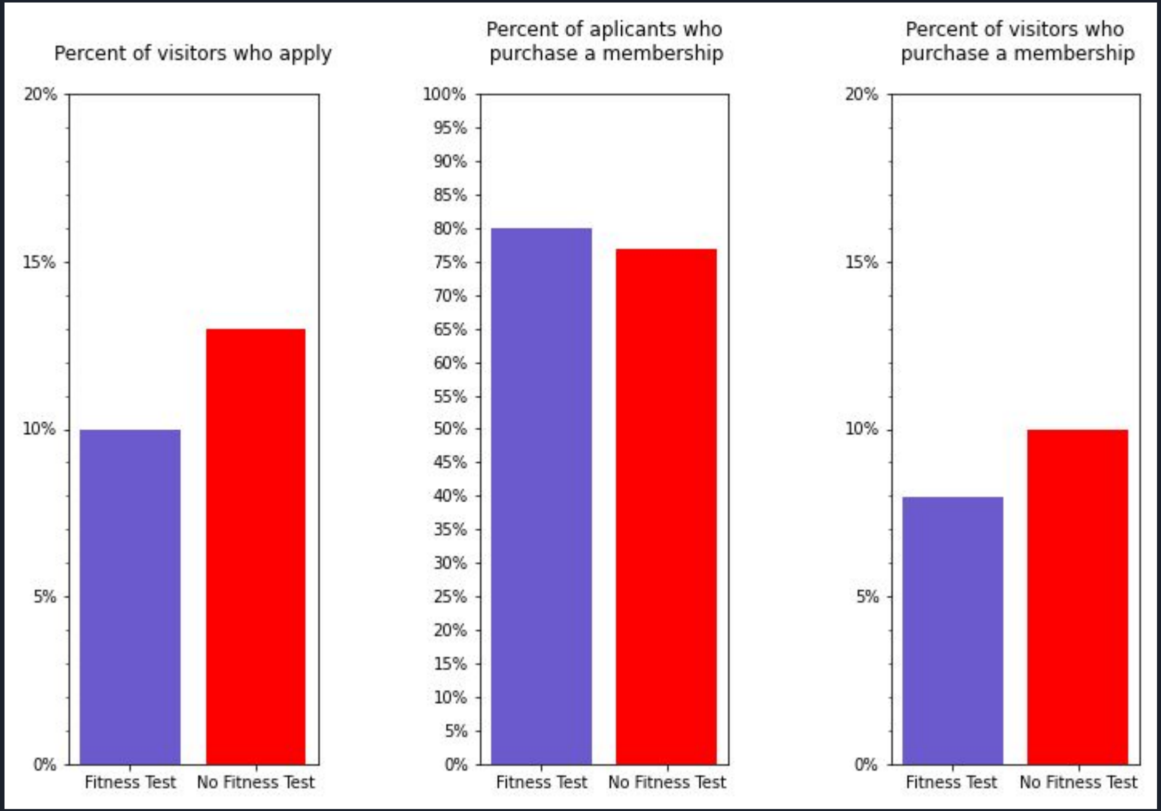
is_member	ab_test_group	Member	Not Member	Total	Percent Purchase
0	A	200	50	250	80.000000
1	B	250	75	325	76.923077



```
x = finalMemberPivot[['Member', 'Not Member']]
dump, pval, eof, expected = chi2_contingency(x)
print(pval) # 0.014724
```

Since our p-value 0.014724 is greater than our significance value (alpha=0.05), we fail to reject the null hypothesis and conclude the results are not statistically significant

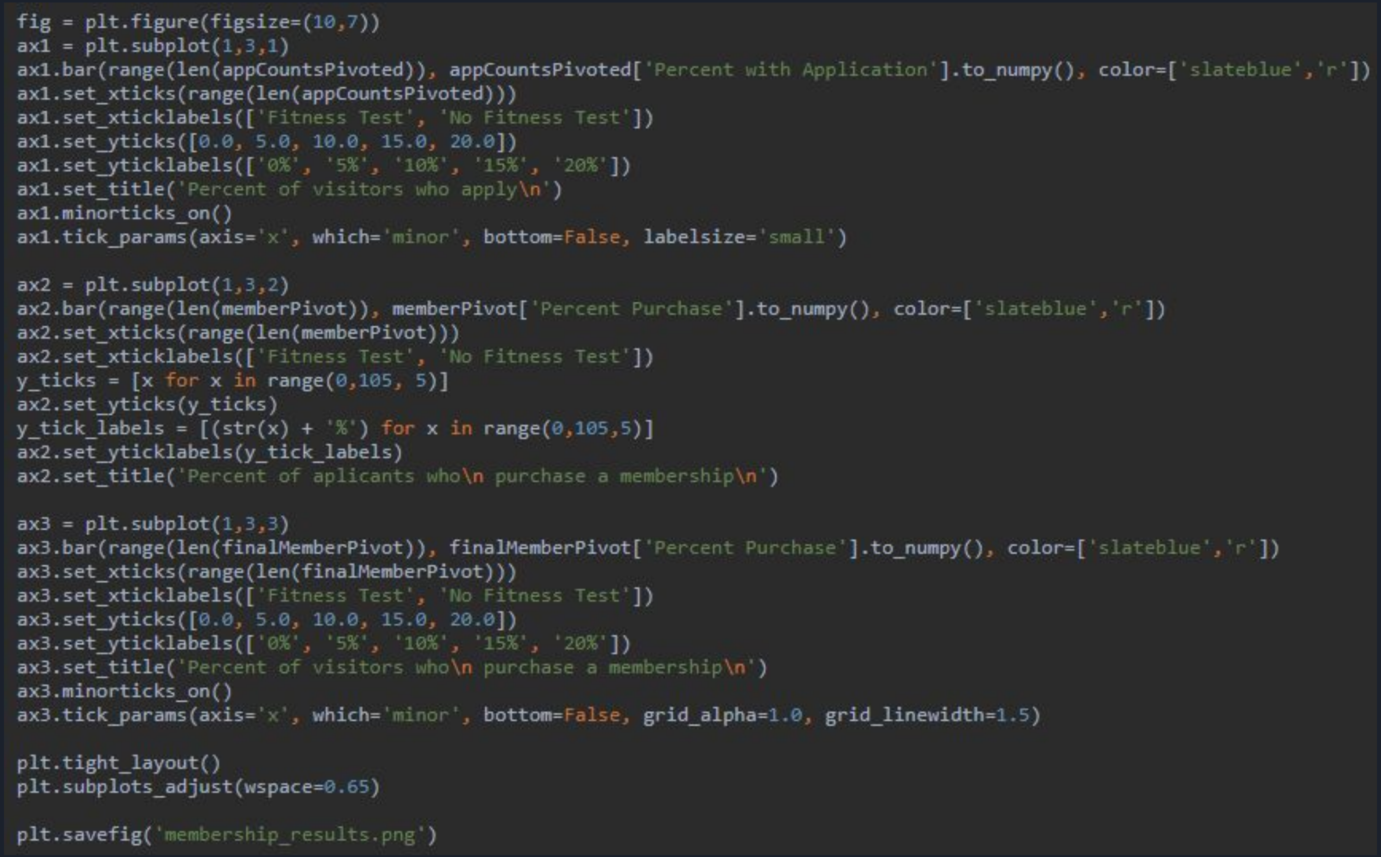
Summary of Acquisition Funnel



Improving the experiment

- Would be useful to have estimated baseline values of our obtained metrics, by analyzing estimator data prior to our A/B test start date on 7-1-2017

Acquisition Funnel source code



```
fig = plt.figure(figsize=(10,7))
ax1 = plt.subplot(1,3,1)
ax1.bar(range(len(appCountsPivoted)), appCountsPivoted['Percent with Application'].to_numpy(), color=['slateblue','r'])
ax1.set_xticks(range(len(appCountsPivoted)))
ax1.set_xticklabels(['Fitness Test', 'No Fitness Test'])
ax1.set_yticks([0.0, 5.0, 10.0, 15.0, 20.0])
ax1.set_yticklabels(['0%', '5%', '10%', '15%', '20%'])
ax1.set_title('Percent of visitors who apply\n')
ax1.minorticks_on()
ax1.tick_params(axis='x', which='minor', bottom=False, labelsize='small')

ax2 = plt.subplot(1,3,2)
ax2.bar(range(len(memberPivot)), memberPivot['Percent Purchase'].to_numpy(), color=['slateblue','r'])
ax2.set_xticks(range(len(memberPivot)))
ax2.set_xticklabels(['Fitness Test', 'No Fitness Test'])
y_ticks = [x for x in range(0,105, 5)]
ax2.set_yticks(y_ticks)
y_tick_labels = [(str(x) + '%') for x in range(0,105,5)]
ax2.set_yticklabels(y_tick_labels)
ax2.set_title('Percent of aplicants who\n purchase a membership\n')

ax3 = plt.subplot(1,3,3)
ax3.bar(range(len(finalMemberPivot)), finalMemberPivot['Percent Purchase'].to_numpy(), color=['slateblue','r'])
ax3.set_xticks(range(len(finalMemberPivot)))
ax3.set_xticklabels(['Fitness Test', 'No Fitness Test'])
ax3.set_yticks([0.0, 5.0, 10.0, 15.0, 20.0])
ax3.set_yticklabels(['0%', '5%', '10%', '15%', '20%'])
ax3.set_title('Percent of visitors who\n purchase a membership\n')
ax3.minorticks_on()
ax3.tick_params(axis='x', which='minor', bottom=False, grid_alpha=1.0, grid_linewidth=1.5)

plt.tight_layout()
plt.subplots_adjust(wspace=0.65)

plt.savefig('membership_results.png')
```