Univelcity Virtual Intership -- Week 1

Instruction :There are two documents attached; one contains the list of people who registered for a program while the other contains the list of those that eventually got enrolled.

1. Applying Data Analysis and Visualization, glean out insights from both datasets with respect to the Gender of the applicants.
2. What is the probability that a randomly selected registered female will be enrolled in the program?
3. What is the probability that a randomly selected registered male will be enrolled in the program?
4. What is the probability that a registered applicant will eventually enroll in the program?

In [50]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

In [2]:
```
enrolled = pd.read_csv('enrolled applicants.csv',index_col= 'Unnamed: 0')
registered = pd.read_csv('registered applicants.csv',index_col= 'Unnamed: 0')
```

Data Wrangling

Enrolled

In [3]:
```
enrolled.head()
```

Out[3]:

Unnamed: 0	FIRST NAME	LAST NAME	SOURCE	COURSE	TRACK	GENDER
1120	Robert	Mccain	WEBSITE	Full Stack	Weekday	Male
2557	Russel	Reagin	Mr Simps	Full Stack	Weekday	Male
1682	Barry	Dickson	WEBSITE	Full Stack	Weekday	Male
1961	Allen	Williams	Mr Simps	NaN	Weekday	Male
295	Brian	Bryon	Social Media	Python For Data Science	Weekend	Male

In [4]:
```
enrolled['GENDER'].value_counts()
```

Out[4]:
```
Male      1611
Female     142
male        12
female       1
Name: GENDER, dtype: int64
```

In [5]:
```
enrolled['GENDER'] = enrolled['GENDER'].apply(lambda x: x.capitalize())
enrolled['GENDER'].unique()
```

Out[5]:
```
array(['Male', 'Female'], dtype=object)
```

In [6]:
```
enrolled.describe(include='O').T
```

Out[6]:

	count	unique	top	freq
FIRST NAME	1766	550	James	63
LAST NAME	1766	1323	Williams	16
SOURCE	1754	3	Social Media	1096
COURSE	1741	13	Full Stack	310
TRACK	1766	3	Weekday	1131
GENDER	1766	2	Male	1623

In [7]:
```
enrolled['status'] = 'Enrolled'
```

Registered

In [8]:
```
registered.head()
```

Out[8]:

Unnamed: 0	FIRST NAME	LAST NAME	SOURCE	COURSE	TRACK	GENDER
2373	Brian	May	Social Media	Python For DataScience	Weekend	Male
758	Leon	Melvin	Social Media	Product Design(UI/UX)	Weekend	Male
2287	Juan	Harris	Mr Simps	Product Design(UI/UX)	Weekday	Male
2480	Gilbert	Denson	Social Media	Figma Design to Webflow	Weekend	Male
1359	Larry	Williams	WEBSITE	Python For DataScience	Weekday	Male

In [9]:
```
# Missing values
registered.isnull().mean().round(3)
```

Out[9]:
```
FIRST NAME    0.000
LAST NAME     0.000
SOURCE        0.007
COURSE        0.007
TRACK         0.000
GENDER        0.015
dtype: float64
```

In [10]:
```
missing_features = [x for x in registered.columns if registered[x].isnull().mean(

for missing in missing_features:
    registered[missing] = registered[missing].replace(np.NaN, registered[missing]
```

In [11]:
```
registered['GENDER'].unique()
```

Out[11]:
```
array(['Male', 'Female', 'male', 'female'], dtype=object)
```

In [12]:
```
registered['GENDER'] = registered['GENDER'].apply(lambda x: str(x).capitalize())
registered['GENDER'].unique()
```

Out[12]:
```
array(['Male', 'Female'], dtype=object)
```

In [13]:
```
registered.describe(include='O').T
```

Out[13]:

	count	unique	top	freq
FIRST NAME	3676	921	James	121
LAST NAME	3676	2453	Williams	33
SOURCE	3676	3	Social Media	2251
COURSE	3676	13	Full Stack	693
TRACK	3676	3	Weekday	2312
GENDER	3676	2	Male	2959

In [169]:
```
registered['status'] = 'Not Enrolled'    # Added a flag
```

Merge

In [168]:
```
#  Checking if the data overlap

list(enrolled.index) in list(registered.index)
```

Out[168]:
```
False
```

In [17]:
```
data = pd.concat([enrolled, registered])
data.describe(include='O')
```

Out[17]:

	FIRST NAME	LAST NAME	SOURCE	COURSE	TRACK	GENDER	status
count	5442	5442	5430	5417	5442	5442	5442
unique	921	2453	3	13	3	2	2
top	James	Williams	Social Media	Full Stack	Weekday	Male	Not Enrolled
freq	184	49	3347	1003	3443	4582	3676

In [65]:
```
data['COURSE'].unique()
```

Out[65]:
```
array(['Full Stack', nan, 'Python For Data Science', 'FullStack',
       'Product Design(UI/UX)', 'Product Management', 'Frontend',
       'Frontend Web Development', 'Figma Design to Webflow',
       'Product Design(UI/UX) ONLINE', 'Product Design',
       'Backend With Python Django', 'CyberSecurity',
       'Python For DataScience'], dtype=object)
```

In [80]:
```
#  Cleaning Courses column

data['COURSE'] = data['COURSE'].apply(lambda x:str(x).replace(' ',''))
data['COURSE']= data['COURSE'].replace(['ProductDesign(UI/UX)','ProductDesign(UI/
data['COURSE'] =data['COURSE'].replace(['FrontendWebDevelopment'], 'Frontend')
data['COURSE'] =data['COURSE'].replace(['nan'], data['COURSE'].mode()[0])

data['COURSE'].unique()
```

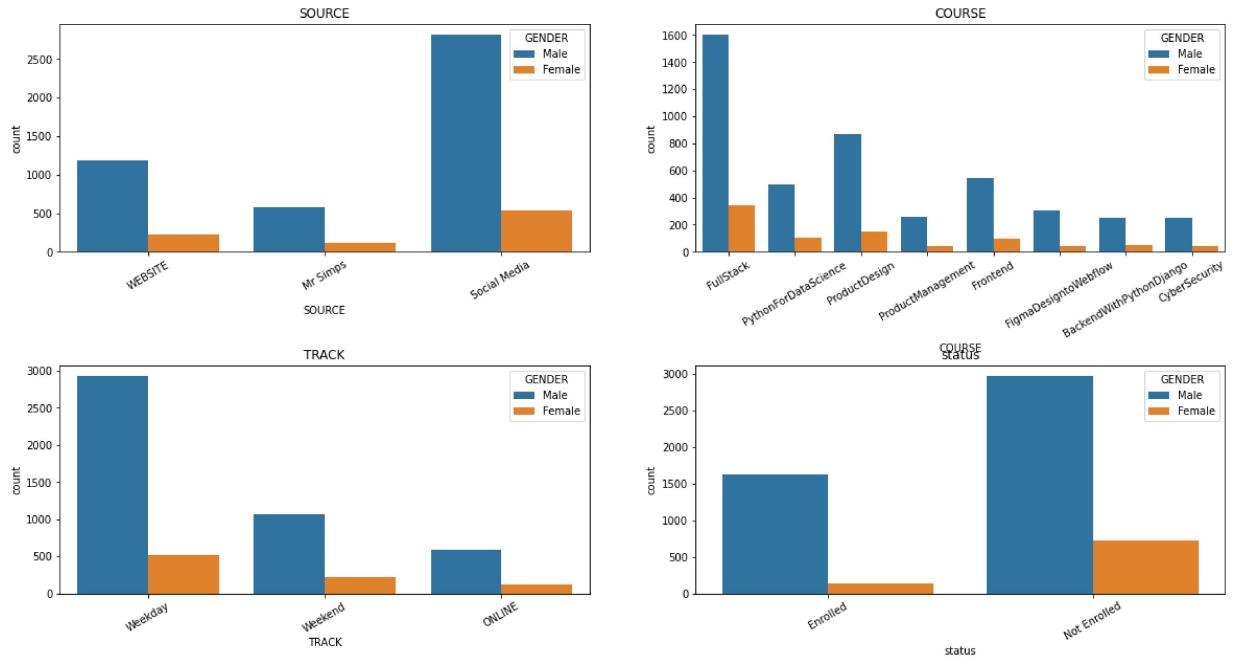
Out[80]:
```
array(['FullStack', 'PythonForDataScience', 'ProductDesign',
       'ProductManagement', 'Frontend', 'FigmaDesigntoWebflow',
       'BackendWithPythonDjango', 'CyberSecurity'], dtype=object)
```

In [103]:
```
features = [x for x in data.columns if data[x].dtypes and data[x].nunique() < 10]
features.pop(-2)
features

fig, axs = plt.subplots(nrows=2, ncols=2, figsize=(18,10))
fig.set_size_inches(20, 10)
fig.subplots_adjust(wspace=0.2)
fig.subplots_adjust(hspace=0.5)

for feature, ax in zip(features, axs.flatten()):
    sns.countplot(x=data[feature], ax=ax, hue='GENDER', data=data)
    ax.set_title(feature)
    ax.tick_params(axis='x', labelrotation=30)

plt.show()
```



Question 2 & 3: What is the probability that a randomly selected registered Female / Male will be enrolled in the program?

In [160]:
```
data['gender_status'] = data['GENDER'] +'_'+ data['status']
```

Female

In [163]:
```
num_female_enrolled = len(data[(data['GENDER']=='Female') & (data['status'] == 'E
total_number_enrolled = len(data)

print('Probability of randomly selecting a Female who enrolled is {}%'.format(rou
```

```
Probability of randomly selecting a female who enrolled is 2.63%
```

Male

In [166]:
```
num_male_enrolled = len(data[data['gender_status']=='Male_Enrolled'])
total_number_enrolled = len(data)

print('Probability of randomly selecting a Male who enrolled is {}%'.format(round
```

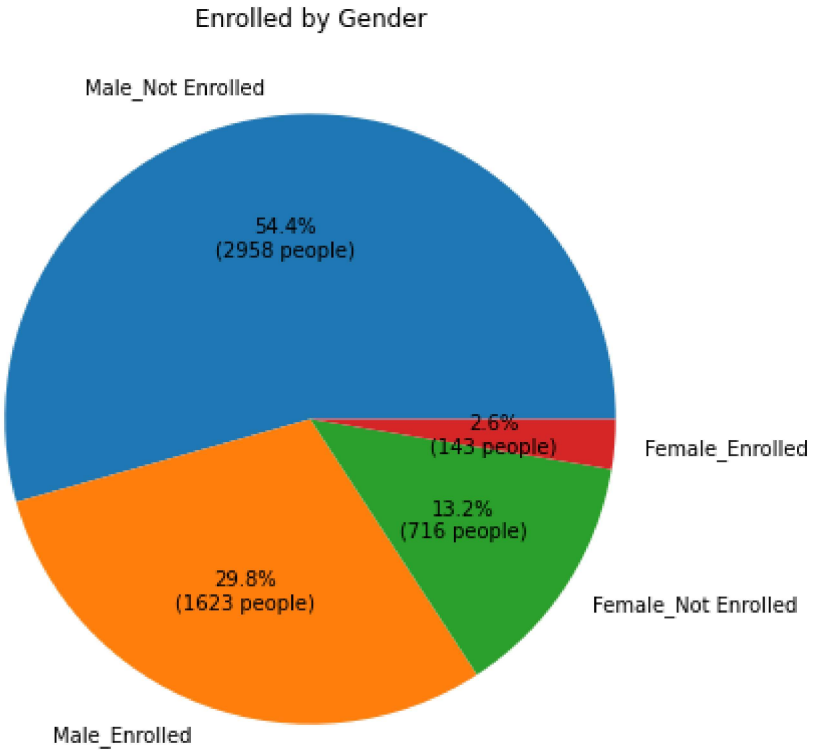
```
Probability of randomly selecting a Male who enrolled is 29.82%
```

In [153]:
```
# Creating autocpt arguments
def func(pct, allvalues):
    absolute = int(pct / 100.*np.sum(allvalues))
    return "{:.1f}%\n({:d} people)".format(pct, absolute)
```

In [158]:
```
new_df = pd.DataFrame(data['gender_status'].value_counts())
new_df.reset_index(inplace=True)
label = list(new_df['index'])
```

In [159]:
```
plt.figure(figsize=(10, 7))

plt.pie(new_df['gender_status'], labels=label ,autopct=lambda pct: func(pct, gen_
plt.title('Enrolled by Gender')
plt.show()
```

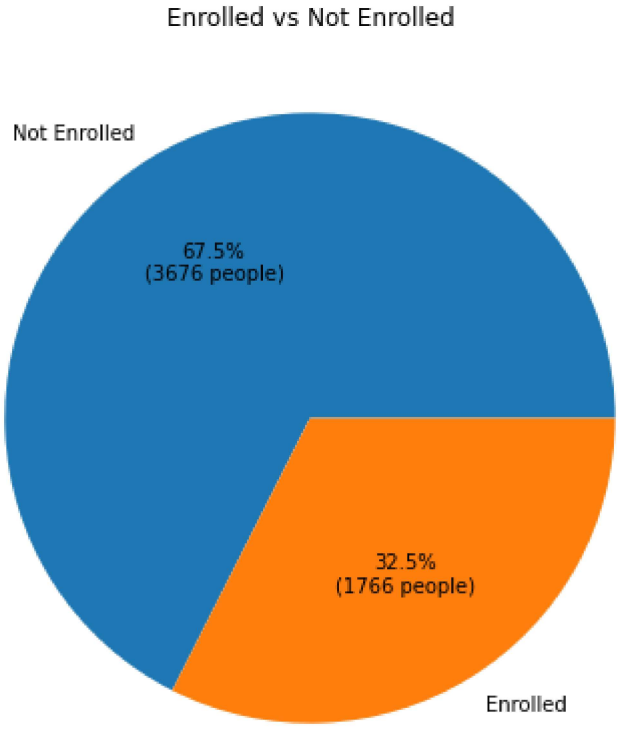


Question 4: What is the probability that a registered applicant will eventually enroll in the program?

In [155]:
```
status_data = list(data['status'].value_counts())
label=['Not Enrolled', 'Enrolled']

plt.figure(figsize=(10, 7))

plt.pie(status_data, labels=label,autopct=lambda pct: func(pct, status_data))
plt.title('Enrolled vs Not Enrolled')
plt.show()
```



In [167]:
```
num_enrolled = len(data[(data['status']=='Enrolled')])
total_number_enrolled = len(data)

print('Probability of randomly selecting a person who enrolled is {}%'.format(rou
```

```
Probability of randomly selecting a person who enrolled is 32.45%
```

In [ ]:
```

```