CMSC 394 survey data analysis NEW

Christian Roncal

Dependencies

In [1]:
```
import pandas as pd
import numpy as np
```

Load and clean up the data

In [2]:
```
df = pd.read_csv('surveyresponses.csv')
```

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

Out[3]:

	Timestamp	Did you graduate from a highschool under the Prince George's County Public Schools system?	What year do you expect to receive your bachelors (if in combined BS/MS programs when did you finish your undergrad requirements)?	What year did you graduate from high school?	How many summer semesters have you taken?	What was your high school GPA?	What is your first semester college GPA?	How many "pre-requisite classes" did you have to take? e.g. (Math115 - Precalculus to take Math140 - Calculus 1)
0	4/24/2019 23:28:18	Yes	Spring 2019	2015	3	3.500	3.800	3
1	4/24/2019 23:33:04	No	Spring 2020	2017	1	3.760	3.947	0
2	4/24/2019 23:34:46	No	Spring 2020	2016	4	4.778	2.987	0
3	4/24/2019 23:35:23	No	Spring 2020	2015	0	3.700	3.900	3
4	4/24/2019 23:37:05	No	Spring 2020	2016	0	3.750	3.750	2

Rename columns

1. pg - graduated from pgcps 1/0 (yes/no)
2. cgrad -college grad year semester
3. hsgrad - hs grad year
4. nsummer - number of summer semesters
5. nremedial - number of "prereq classes"
6. hsgpa - hs gpa
7. cgpa - first semester college gpa
8. nswitch - number of major switches
9. unsure - unsure of college plans
10. uninterested - unintersted in hs schoolwork
11. mil - military path known
12. voc - vocational path known
13. it - it path known

In [4]:
```
df.columns = ['timestamp', 'pg', 'cgrad', 'hsgrad', 'nsummer', 'hsgpa'
, 'cgpa', 'nremedial', 'nswitch', 'unsure', 'uninterested', 'mil', 'vo
c', 'it']
```

In [5]:
```
df = df.drop(columns='timestamp')
df.head()
```

Out[5]:

	pg	cgrad	hsgrad	nsummer	hsgpa	cgpa	nremedial	nswitch	unsure	uninterested	
0	Yes	Spring 2019	2015	3	3.500	3.800	3	0	Disagree	Agree	Dis
1	No	Spring 2020	2017	1	3.760	3.947	0	0	Agree	Agree	Dis
2	No	Spring 2020	2016	4	4.778	2.987	0	1	Agree	Agree	
3	No	Spring 2020	2015	0	3.700	3.900	3	0	Agree	Agree	Dis
4	No	Spring 2020	2016	0	3.750	3.750	2	1	Agree	Agree	Dis

Fill in NaNs and numericalize

In [6]:
```
df = df.fillna('') # replaces all NaNs with empty string
```

Numericalize

It's easier to deal with numbers when computationally analyzing data so let's only deal with numbers

In [7]:
```
agree_map = {'Yes': 1, 'No': 0, 'Agree':1, 'Disagree':0}
def numericalize(x):
    try:
        return agree_map[x]
    except:
        print('errorrrr')
        print(x)
```

In [8]:
```
col_names = ['pg', 'unsure', 'uninterested', 'mil', 'voc', 'it']
df[col_names].head()
```

Out[8]:

	pg	unsure	uninterested	mil	voc	it
0	Yes	Disagree	Agree	Disagree	Disagree	Disagree
1	No	Agree	Agree	Disagree	Disagree	Disagree
2	No	Agree	Agree	Agree	Agree	Agree
3	No	Agree	Agree	Disagree	Agree	Agree
4	No	Agree	Agree	Disagree	Agree	Disagree

In [9]:
```
for col in col_names:
    df[col] = df[col].apply(numericalize)
```

In [10]:
```
df.head()
```

Out[10]:

	pg	cgrad	hsgrad	nsummer	hsgpa	cgpa	nremedial	nswitch	unsure	uninterested	mil
0	1	Spring 2019	2015	3	3.500	3.800	3	0	0	1	0
1	0	Spring 2020	2017	1	3.760	3.947	0	0	1	1	0
2	0	Spring 2020	2016	4	4.778	2.987	0	1	1	1	1
3	0	Spring 2020	2015	0	3.700	3.900	3	0	1	1	0
4	0	Spring 2020	2016	0	3.750	3.750	2	1	1	1	0

Creating new features

nsems = grad year - hs grad year + 1 (if fall) + nsummer

In [11]:
```
def nsems(x):
    szn, year = x['cgrad'].split()
    year = int(year)
    fallszn = False if szn == 'Spring' else True

    res = year - x['hsgrad'] + x['nsummer']
    res += 1 if fallszn else 0
    return res
```

In [12]:
```
df['nsems'] = df.apply(nsems, axis=1)
```

In [13]:
```
df.head()
```

Out[13]:

	pg	cgrad	hsgrad	nsummer	hsgpa	cgpa	nremedial	nswitch	unsure	uninterested	mil
0	1	Spring 2019	2015	3	3.500	3.800	3	0	0	1	0
1	0	Spring 2020	2017	1	3.760	3.947	0	0	1	1	0
2	0	Spring 2020	2016	4	4.778	2.987	0	1	1	1	1
3	0	Spring 2020	2015	0	3.700	3.900	3	0	1	1	0
4	0	Spring 2020	2016	0	3.750	3.750	2	1	1	1	0

alt score = "degree of college alternative knowledge (ie number of alternative paths shown)"
$\sum[\text{unintersted, mil, voc}]$

In [14]:
```
def altscore(x):
    return x['mil'] + x['voc'] + x['it']
```

In [15]:
```
df['altscore'] = df.apply(altscore, axis=1)
```

In [16]:
```
df.head()
```

Out[16]:

	pg	cgrad	hsgrad	nsummer	hsgpa	cgpa	nremedial	nswitch	unsure	uninterested	mil
0	1	Spring 2019	2015	3	3.500	3.800	3	0	0	1	0
1	0	Spring 2020	2017	1	3.760	3.947	0	0	1	1	0
2	0	Spring 2020	2016	4	4.778	2.987	0	1	1	1	1
3	0	Spring 2020	2015	0	3.700	3.900	3	0	1	1	0
4	0	Spring 2020	2016	0	3.750	3.750	2	1	1	1	0

In [17]:
```
df.to_csv('num_394.csv', index=False)
```

In [18]:
```
t = pd.read_csv('num_394.csv')
len(t)
```

Out[18]: 50

In [19]:
```
df_clean = df.drop(columns=['cgrad', 'hsgrad', 'nsummer'])
df_clean.head()
```

Out[19]:

	pg	hsgpa	cgpa	nremedial	nswitch	unsure	uninterested	mil	voc	it	nsems	altscore
0	1	3.500	3.800	3	0	0	1	0	0	0	7	0
1	0	3.760	3.947	0	0	1	1	0	0	0	4	0
2	0	4.778	2.987	0	1	1	1	1	1	1	8	3
3	0	3.700	3.900	3	0	1	1	0	1	1	5	2
4	0	3.750	3.750	2	1	1	1	0	1	0	4	1

In [20]:
```
df_clean.to_csv('394clean.csv', index=False)
```

In [23]:
```
pg_df = df.loc[df['pg'] == 1]
```

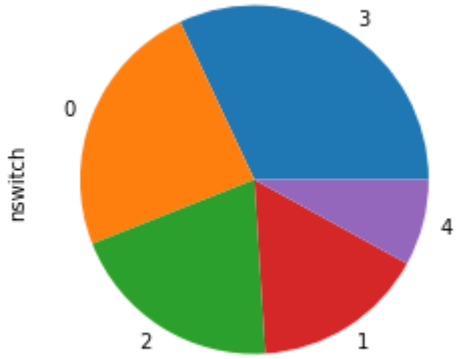
In [25]:
```
len(pg_df.loc[pg_df['nswitch'] > 0])
```

Out[25]: 19

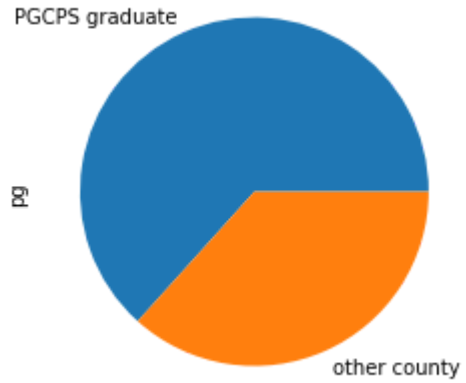
In [37]:
```
len(pg_df)
```

Out[37]: 25

In [31]:
```
nswitch_pg = pg_df['nswitch'].value_counts().plot.pie().get_figure()
nswitch_pg.savefig('switchpgpie.png')
```



In [41]:
```
pgswitchpie = df.loc[df['nswitch'] > 0]['pg'].apply(lambda x: "PGCPS g
raduate" if x == 1 else "other county").value_counts().plot.pie().get_
figure()
```



In [42]:
```
pgswitchpie.savefig('pgvsothersswitches.png')
```

In [39]:
```
pgswitchpie
```

Out[39]:
```
PGCPS graduate    19
other county      11
Name: pg, dtype: int64
```

In [40]:
```
19/30
```

Out[40]:
```
0.6333333333333333
```

In [ ]:
```

```