Homework 2 (Due: 2021.04.01 11:59 PM)

In [61]:

```
Name:YeoHyuksoo
```

Student ID:2016312761

For this homework, you are provided with an input file. Submit the code that you used for analysis as an ipython notebook and an exported PDF file.

Q1 Analyzing the Election Data (3 x 8 = 24 pts) ¶

Run the following cell to set the filename.

In [62]:

```
filename = 'bush-gore-results-fl_demo.csv'
```

(a) Determine how many counties Bush won. You can assume that each row corresponds to a unique county. Demo file: 'bush-gore-results-fl_demo.csv'

In [63]:

```
#
# YOUR CODE HERE
#
filename = 'bush-gore-results-fl_demo.csv'
f = open(filename, 'r')
data = f.readlines()
data = [s.strip().split(',') for s in data]
won = 0
data = data[1:]
for d in data:
    candidate = list(map(float, d[11:]))
    if float(max(candidate)) == float(d[11]):
        won += 1
print(won)
```

51

(b) Determine who won the largest county. The column named "npop" records the size of each county. Demo file: 'bush-gore-results-fl_demo.csv'

In [64]:

```
#
# YOUR CODE HERE
#
filename = 'bush-gore-results-fl_demo.csv'
f = open(filename, 'r')
data = f.readlines()
data = [s.strip().split(',') for s in data]
header = data[0]
data = data[1:]
max_pop = 0
max_i = 0
for i in range(len(data)):
    if float(data[i][3]) > max_pop:
        max_pop = float(data[i][3])
        max_i = i

cand = list(map(float, data[max_i][11:]))

print(header[11+cand.index(max(cand))])
```

gore

(c) Determine the average number of votes per county that Buchanan obtained. You can assume that each row corresponds to a unique county. Demo file: 'bush-gore-results-fl_demo.csv'

In [65]:

```
#
# YOUR CODE HERE
#
filename = 'bush-gore-results-fl_demo.csv'
f = open(filename, 'r')
data = f.readlines()
data = [s.strip().split(',') for s in data]
data = data[1:]
avg_vote = 0
for d in data:
    avg_vote += float(d[17])

print(avg_vote / len(data))
```

360.67164179104475

(d) Determine, for the number of votes Buchanan obtained in Palm Beach, how many standard deviations it is away from Buchanan's overall mean, in absolute value. The row with coounty number 50 ("co"=50) records the results for Palm Beach County. (You can assumed that such a row will exist in the test case.) Demo file: 'bush-gore-results-fl_demo.csv'

In [2]:

```
#
# YOUR CODE HERE
#
import numpy as np
filename = 'bush-gore-results-fl_demo.csv'
f = open(filename, 'r')
data = f.readlines()
data = [s.strip().split(',') for s in data]
data = data[1:]
vote = 0
buch = []
for d in data:
    vote += float(d[17])
    buch.append(float(d[17]))

std = np.std(buch)
print(abs(float(data[49][17]) - (vote / len(data))) / std)
```

7.045796738007179

(e) Now calculate the above statistic (same as in part f) for all the counties and report them in decreasing order. Demo file: 'bush-gore-results-fl_demo.csv' Example output: county_50 6.993018
county_52 ... ...

In [3]:

```
#
# YOUR CODE HERE
#
import operator
import numpy as np
filename = 'bush-gore-results-fl_demo.csv'
f = open(filename, 'r')
data = f.readlines()
data = [s.strip().split(',') for s in data]
data = data[1:]
vote = 0
buch = []
for d in data:
    vote += float(d[17])
    buch.append(float(d[17]))

std = np.std(buch)
avgvote = vote / len(data)
deviate = {}
std = np.std(buch)

for d in data:
    deviate[int(d[0])] = abs(float(d[17]) - avgvote) / std
deviate = sorted(deviate.items(), key=operator.itemgetter(1), reverse=True)

for key, value in deviate:
    print("county_"+str(key), value)
```

```
county_50 7.045796738007179
county_52 1.6847423691011076
county_28 1.3130067695863175
county_6 1.180883875782868
county_15 0.8763294087105095
county_5 0.6927009800345287
county_51 0.6927009800345287
county_41 0.677025382464628
county_43 0.6703072692203847
county_53 0.6076048789407816
county_21 0.5635861966389492
county_33 0.5613468255575348
county_16 0.5404237464983497
county_25 0.534474372580562
county_23 0.5322350014991476
county_64 0.5269875200098633
county_62 0.52327751717349
county_14 0.5187987750106612
county_20 0.5187987750106612
county_32 0.5187987750106612
county_39 0.5187987750106612
county_24 0.5165594039292468
county_18 0.5098412906850036
county_13 0.5031231774407604
county_63 0.500883806359346
county_19 0.4986444352779316
county_38 0.4964050641965172
county_47 0.4874475798708596
county_65 0.4807294666266164
county_44 0.478490095545202
county_4 0.4381814160797428
county_37 0.433702673916914
county_22 0.42474518959125646
county_2 0.4202664474284276
county_48 0.4150189659391433
county_29 0.41354833418418446
county_17 0.39787273661428363
county_67 0.38667588120721164
county_12 0.38443651012579727
county_7 0.38219713904438285
county_45 0.38219713904438285
county_31 0.3553246860674101
county_30 0.34860657282316687
county_61 0.34188845957892366
county_42 0.3329309752532661
county_60 0.32845223309043725
county_66 0.3150160066019509
county_11 0.31053726443912205
county_59 0.30605852227629327
county_27 0.29934040903205006
county_49 0.2590317295665909
county_54 0.2523136163223477
county_8 0.1761749995542581
county_10 0.16721751522860054
county_57 0.14930254657728534
county_55 0.11270386994819938
county_35 0.09926764345971298
county_56 0.09926764345971298
county_58 0.07092455872778136
county_34 0.0634377061570826
county_36 0.0477621085871818
```

```
county_26 0.04181273466939417
county_3 0.028376508180907773
county_40 0.02312902669162341
county_9 0.02088965561020901
county_46 0.01417154236596581
county_1 0.005214058040308214
```

(f) Assuming that the votes were distributed across the white, black, and hispanic population uniformly, determine which candidate obtained the largest number of votes for each subpopulation. Demo file: 'bush-gore-results-fl_demo.csv' Example output: white: bush black: .. hispanic: ..

In [68]:

```
#
# YOUR CODE HERE
#
filename = 'bush-gore-results-fl_demo.csv'
f = open(filename, 'r')
data = f.readlines()
data = [s.strip().split(',') for s in data]
header = data[0]
data = data[1:]
white = [0 for i in range(10)]
black = [0 for i in range(10)]
hispanic = [0 for i in range(10)]
for d in data:
    for i in range(0, 10, 1):
        white[i] += float(d[i+11]) * float(d[4]) / 100
        black[i] += float(d[i+11]) * float(d[5]) / 100
        hispanic[i] += float(d[i+11]) * float(d[6]) / 100

print("white:", header[11+white.index(max(white))])
print("black:", header[11+black.index(max(black))])
print("hispanic:", header[11+hispanic.index(max(hispanic))])
```

```
white: bush
black: gore
hispanic: gore
```

(g) Calculate the correlation between the difference in votes between Bush and Gore, and the votes obtained by Nader. (FYI: Pearson's correlation coefficient) https://en.wikipedia.org/wiki/Pearson_correlation_coefficient (https://en.wikipedia.org/wiki/Pearson_correlation_coefficient) Demo file: 'bush-gore-results-fl_demo.csv'

In [69]:

```
#
# YOUR CODE HERE
#
import pandas as pd
filename = 'bush-gore-results-fl_demo.csv'
f = open(filename, 'r')
data = f.readlines()
data = [s.strip().split(',') for s in data]
data = data[1:]
bush_gore = []
nader = []
lists = []
for d in data:
    bush_gore.append(abs(float(d[12])-float(d[11])))
    nader.append(float(d[14]))
lists.append(bush_gore)
lists.append(nader)
df = pd.DataFrame(lists).T
corr = df.corr(method = 'pearson')
print(corr[0][1])
```

0.5443059987726585

In [73]:

(h) Find the distance between the county that Bush won by the largest margin and the county that Gore won by the largest margin. (Just use basic Euclidean distance between the latitude (lat) and longitude (lon) values for the counties, no need to compute spherical distance.) (FYI: Euclid en distane is decribed in https://en.wikipedia.org/wiki/Euclidean_distance#:~:text=In%20mathema tics%2C%20the%20Euclidean%20distance,metric%20as%20the%20Pythagorean%20metric.)

Demo file: 'bush-gore-results-fl_demo.csv'

In [74]:

```
#
# YOUR CODE HERE
#
import math
filename = 'bush-gore-results-fl_demo.csv'
f = open(filename, 'r')
data = f.readlines()
data = [s.strip().split(',') for s in data]
data = data[1:]
bush_max = 0
gore_max = 0
bush_i = 0
gore_i = 0
for i in range(len(data)):
    if float(data[i][11]) > float(data[i][12]):
        if float(data[i][11]) - float(data[i][12]) > bush_max:
            bush_max = float(data[i][11]) - float(data[i][12])
            bush_i = i
    elif float(data[i][11]) < float(data[i][12]):
        if float(data[i][12]) - float(data[i][11]) > gore_max:
            gore_max = float(data[i][12]) - float(data[i][11])
            gore_i = i

print(math.sqrt(((float(data[bush_i][1])-float(data[gore_i][1])) ** 2) + ((float(data[bush_i][2
])-float(data[gore_i][2])) ** 2)))
```

4.341658669218476

In [ ]:

```

```