# Project 1

In [3]:
```
# Load libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.figure_factory as ff
```

In [4]:
```
data_demographics = pd.read_csv("demographics_train.csv")
data_election = pd.read_csv("election_train.csv")
print(data_election.head(10))
```

```
   Year State           County      Office       Party    Votes
0  2018    AZ    Apache County  US Senator  Democratic  16298.0
1  2018    AZ    Apache County  US Senator  Republican   7810.0
2  2018    AZ   Cochise County  US Senator  Democratic  17383.0
3  2018    AZ   Cochise County  US Senator  Republican  26929.0
4  2018    AZ  Coconino County  US Senator  Democratic  34240.0
5  2018    AZ  Coconino County  US Senator  Republican  19249.0
6  2018    AZ      Gila County  US Senator  Democratic   7643.0
7  2018    AZ      Gila County  US Senator  Republican  12180.0
8  2018    AZ    Graham County  US Senator  Democratic   3368.0
9  2018    AZ    Graham County  US Senator  Republican   6870.0
```

**Task 01 (of 10): (5 pts.) Reshape dataset election_train from long format to wide format. Hint: the reshaped dataset should contain 1205 rows and 6 columns.**

In [5]:
```
# convert from long format to wide format using pivot_table
data_election_tidy = pd.pivot_table(data_election, values='Votes', index
=['Year', 'State', 'County', 'Office'],
                                    columns='Party', aggfunc = np.sum).r
eset_index()

# reset the missing values which were set to 0 by the above pivot table
data_election_tidy.loc[data_election_tidy['Democratic'] == 0,'Democrati
c'] = np.nan
data_election_tidy.loc[data_election_tidy['Republican'] == 0, 'Republica
n'] = np.nan
print(data_election_tidy.head(10))
data_election_tidy.shape
```

```
Party  Year State           County      Office  Democratic  Republican
0      2018    AZ    Apache County  US Senator     16298.0      7810.0
1      2018    AZ   Cochise County  US Senator     17383.0     26929.0
2      2018    AZ  Coconino County  US Senator     34240.0     19249.0
3      2018    AZ      Gila County  US Senator      7643.0     12180.0
4      2018    AZ    Graham County  US Senator      3368.0      6870.0
5      2018    AZ    La Paz County  US Senator      1609.0      3265.0
6      2018    AZ  Maricopa County  US Senator    732671.0    672505.0
7      2018    AZ    Mohave County  US Senator     19214.0     50209.0
8      2018    AZ    Navajo County  US Senator     16624.0     18767.0
9      2018    AZ      Pima County  US Senator    221242.0    160550.0
```

Out[5]: (1205, 6)

**Task 02 (of 10): (20 pts.) Merge reshaped dataset election_train with dataset demographics_train. Make sure that you address all inconsistencies in the names of the states and the counties before merging. Hint: the merged dataset should contain 1200 rows.**

In [6]:
```
# address the inconsistencies in the states and counties
data_election_tidy = data_election_tidy.replace({'AZ': 'Arizona', 'CT':
'Connecticut', 'DE': 'Delaware', 'FL' : 'Florida',
'HI': 'Hawaii', 'IN':'Indiana', 'ME': 'Maine', 'MD': 'Maryland', 'MA':'M
assachusetts', 'MI': 'Michigan', 'MN':'Minnesota',
'MT':'Montana','NE':'Nebraska', 'NV':'Nevada', 'NJ':'New Jersey', 'NM':
'New Mexico', 'NY':'New York', 'ND':'North Dakota',
'OH':'Ohio','PA':'Pennsylvania', 'RI':'Rhode Island', 'TN':'Tennessee',
'TX':'Texas', 'UT':'Utah', 'VT':'Vermont',
'VA':'Virginia','WA':'Washington', 'WV':'West Virginia','WI':'Wisconsin'
, 'WY':'Wyoming'})
data_election_tidy['County'] = data_election_tidy['County'].str.replace(
"County", "").str.strip()
data_election_tidy['State'] = data_election_tidy['State'].str.title()
data_election_tidy['County'] = data_election_tidy['County'].str.title()
data_demographics['State'] = data_demographics['State'].str.title()
data_demographics['County'] = data_demographics['County'].str.title()
data_election_tidy.head(10)

# merge data_demographics and data_election_wide
data_merged = pd.merge(data_demographics, data_election_tidy, how = 'inn
er', on = ['State','County'], sort = True)
print(data_merged.shape)
data_merged.head(3)
```

```
(1200, 21)
```

Out[6]:

| | State | County | FIPS | Total Population | Citizen Voting-Age Population | Percent White, not Hispanic or Latino | Percent Black, not Hispanic or Latino | Percent Hispanic or Latino | Percent Foreign Born | |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | Arizona | Apache | 4001 | 72346 | 0 | 18.571863 | 0.486551 | 5.947806 | 1.719515 | 5 |
| **1** | Arizona | Cochise | 4003 | 128177 | 92915 | 56.299492 | 3.714395 | 34.403208 | 11.458374 | 4 |
| **2** | Arizona | Coconino | 4005 | 138064 | 104265 | 54.619597 | 1.342855 | 13.711033 | 4.825298 | 5 |

3 rows × 21 columns

**Task 03 (of 10): (5 pts.) Explore the merged dataset. How many variables does the dataset have? What is the type of these variables? Are there any irrelevant or redundant variables? If so, how will you deal with these variables?**

In [7]:
```
data_merged.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1200 entries, 0 to 1199
Data columns (total 21 columns):
State                                    1200 non-null object
County                                   1200 non-null object
FIPS                                     1200 non-null int64
Total Population                         1200 non-null int64
Citizen Voting-Age Population            1200 non-null int64
Percent White, not Hispanic or Latino    1200 non-null float64
Percent Black, not Hispanic or Latino    1200 non-null float64
Percent Hispanic or Latino               1200 non-null float64
Percent Foreign Born                     1200 non-null float64
Percent Female                           1200 non-null float64
Percent Age 29 and Under                 1200 non-null float64
Percent Age 65 and Older                 1200 non-null float64
Median Household Income                  1200 non-null int64
Percent Unemployed                       1200 non-null float64
Percent Less than High School Degree     1200 non-null float64
Percent Less than Bachelor's Degree      1200 non-null float64
Percent Rural                            1200 non-null float64
Year                                     1200 non-null int64
Office                                   1200 non-null object
Democratic                               1195 non-null float64
Republican                               1195 non-null float64
dtypes: float64(13), int64(5), object(3)
memory usage: 206.2+ KB
```

In [8]:
```
data_merged.Year.unique()
```

Out[8]:
```
array([2018])
```

In [9]:
```
data_merged.Office.unique()
```

Out[9]:
```
array(['US Senator'], dtype=object)
```

In [10]:
```
data_merged = data_merged.drop(columns=['Year', 'Office'])
data_merged.head()
```

Out[10]:

| | State | County | FIPS | Total Population | Citizen Voting-Age Population | Percent White, not Hispanic or Latino | Percent Black, not Hispanic or Latino | Percent Hispanic or Latino | Percent Foreign Born | |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | Arizona | Apache | 4001 | 72346 | 0 | 18.571863 | 0.486551 | 5.947806 | 1.719515 | 5 |
| **1** | Arizona | Cochise | 4003 | 128177 | 92915 | 56.299492 | 3.714395 | 34.403208 | 11.458374 | 4 |
| **2** | Arizona | Coconino | 4005 | 138064 | 104265 | 54.619597 | 1.342855 | 13.711033 | 4.825298 | 5 |
| **3** | Arizona | Gila | 4007 | 53179 | 0 | 63.222325 | 0.552850 | 18.548675 | 4.249798 | 5 |
| **4** | Arizona | Graham | 4009 | 37529 | 0 | 51.461536 | 1.811932 | 32.097844 | 4.385942 | 4 |

**Task 04 (of 10): (10 pts.) Search the merged dataset for missing values. Are there any missing values? If so, how will you deal with these values?**

In [11]:
```
data_merged[data_merged.isnull().any(axis=1)]
```

Out[11]:

| | State | County | FIPS | Total Population | Citizen Voting-Age Population | Percent White, not Hispanic or Latino | Percent Black, not Hispanic or Latino | Percent Hispanic or Latino | Perc Fore B |
|---|---|---|---|---|---|---|---|---|---|
| **389** | Nebraska | Lancaster | 31109 | 301707 | 0 | 82.659667 | 3.783472 | 6.531834 | 7.523 |
| **714** | Tennessee | Meigs | 47121 | 11804 | 0 | 94.713656 | 1.330058 | 1.465605 | 1.067 |
| **750** | Texas | Bee | 48025 | 32706 | 0 | 32.660674 | 7.989360 | 57.909864 | 4.344 |
| **865** | Texas | Menard | 48327 | 2163 | 0 | 56.310680 | 1.248266 | 39.389736 | 4.299 |
| **1156** | Wisconsin | Lafayette | 55065 | 16793 | 0 | 94.771631 | 0.339427 | 3.507414 | 2.286 |

In [12]:
```
data_merged = data_merged.dropna(how = 'any')
data_merged[data_merged.isnull().any(axis=1)]
```

Out[12]:

| | State | County | FIPS | Total Population | Citizen Voting-Age Population | Percent White, not Hispanic or Latino | Percent Black, not Hispanic or Latino | Percent Hispanic or Latino | Percent Foreign Born | Percent Female | F |
|---|---|---|---|---|---|---|---|---|---|---|---|

**Task 05 (of 10): (5 pts.) Create a new variable named “Party” that labels each county as Democratic or Republican. This new variable should be equal to 1 if there were more votes cast for the Democratic party than the Republican party in that county and it should be equal to 0 otherwise.**

In [13]:
```
data_merged['Party'] = np.where(data_merged['Democratic'] > data_merged[
'Republican'], 1, 0)
data_merged.head(10)
```

Out[13]:

| | State | County | FIPS | Total Population | Citizen Voting-Age Population | Percent White, not Hispanic or Latino | Percent Black, not Hispanic or Latino | Percent Hispanic or Latino | Percent Foreign Born | |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | Arizona | Apache | 4001 | 72346 | 0 | 18.571863 | 0.486551 | 5.947806 | 1.719515 | 5 |
| **1** | Arizona | Cochise | 4003 | 128177 | 92915 | 56.299492 | 3.714395 | 34.403208 | 11.458374 | 4 |
| **2** | Arizona | Coconino | 4005 | 138064 | 104265 | 54.619597 | 1.342855 | 13.711033 | 4.825298 | 5 |
| **3** | Arizona | Gila | 4007 | 53179 | 0 | 63.222325 | 0.552850 | 18.548675 | 4.249798 | 5 |
| **4** | Arizona | Graham | 4009 | 37529 | 0 | 51.461536 | 1.811932 | 32.097844 | 4.385942 | 4 |
| **5** | Arizona | La Paz | 4012 | 20304 | 15245 | 58.884949 | 0.379236 | 26.182033 | 11.372143 | 4 |
| **6** | Arizona | Maricopa | 4013 | 4088549 | 2723565 | 56.918114 | 5.013612 | 30.286833 | 14.729333 | 5 |
| **7** | Arizona | Mohave | 4015 | 203629 | 0 | 78.252606 | 0.951731 | 15.708470 | 6.969047 | 4 |
| **8** | Arizona | Navajo | 4017 | 108209 | 76280 | 41.927196 | 0.672772 | 11.049913 | 2.914730 | 4 |
| **9** | Arizona | Pima | 4019 | 1003338 | 0 | 53.271579 | 3.199719 | 36.105978 | 12.903428 | 5 |

**Task 06 (of 10): (10 pts.) Compute the mean population for Democratic counties and Republican counties. Which one is higher? Perform a hypothesis test to determine whether this difference is statistically significant at the $\alpha = 0.05$ significance level. What is the result of the test? What conclusion do you make from this result?**

In [20]:
```
#get the mean based on the party and then use loc to get the row and col
umn that correspond to the democratic and republican mean
repMean = data_merged.groupby(['Party']).mean().loc[0:1,'Total Populatio
n'][0]
demMean = data_merged.groupby(['Party']).mean().loc[0:1,'Total Populatio
n'][1]

#print('Republican Total Population Mean',repMean)
#print('Demoractic Total Population Mean',demMean)

#isolating data for democratic and republican total population
dem = data_merged[data_merged['Party']==1].loc[:,'Total Population']
rep = data_merged[data_merged['Party']==0].loc[:,'Total Population']

print('Democratic Total Population Mean:',demMean)
print('Republican Total Population Mean:',repMean)

#compute unpaired t test
import scipy.stats as st
[statistic, pvalue] = st.ttest_ind(dem , rep, equal_var = False)
print('T test statistic:',statistic)
print('Pvalue:',pvalue)
```

```
Democratic Total Population Mean: 300998.3169230769
Republican Total Population Mean: 53864.6724137931
T test statistic: 8.004638577960957
Pvalue: 2.0478717602973023e-14
```

**Task 07 (of 10): (10 pts.) Compute the mean median household income for Democratic counties and Republican counties. Which one is higher? Perform a hypothesis test to determine whether this difference is statistically significant at the $\alpha = 0.05$ significance level. What is the result of the test? What conclusion do you make from this result?**

In [21]:
```
repIncome = data_merged.groupby(['Party']).mean().loc[0:1,'Median Household Income'][0]
demIncome = data_merged.groupby(['Party']).mean().loc[0:1,'Median Household Income'][1]
print('Democrats Mean Household Income:' ,demIncome)
print('Republican Mean Household Income:',repIncome)

#isolating data for democratic and republican median household income
demIncome = data_merged[data_merged['Party']==1].loc[:,'Median Household Income']

repIncome = data_merged[data_merged['Party']==0].loc[:,'Median Household Income']

#compute unpaired t test
import scipy.stats as st
[statistic, pvalue] = st.ttest_ind(demIncome , repIncome, equal_var = False)

print('Test stastic:',statistic)
print('Pvalue:',pvalue)
```

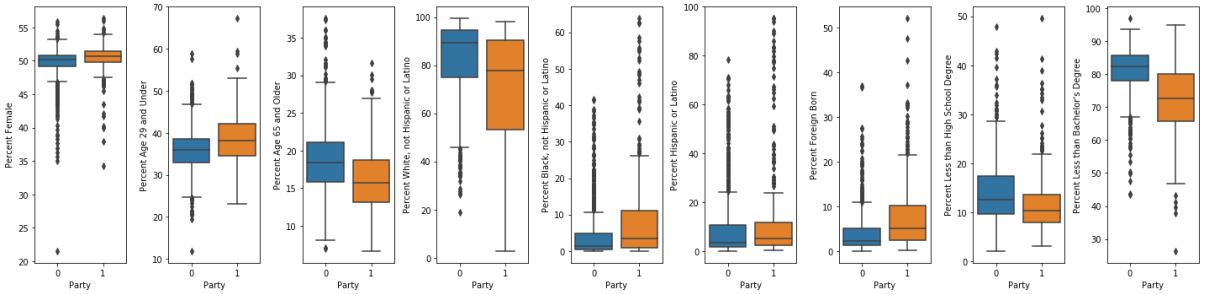
```
Democrats Mean Household Income: 53798.732307692306
Republican Mean Household Income: 48746.81954022989
Test stastic: 5.479141589767387
Pvalue: 7.149437363182572e-08
```

**Task 08 (of 10): (20 pts.) Compare Democratic counties and Republican counties in terms of age, gender, race and ethnicity, and education by computing descriptive statistics and creating plots to visualize the results. What conclusions do you make for each variable from the descriptive statistics and the plots?**

In [16]:
```
#descriptive statistics
pd.set_option('display.max_column',None)
data_merged.groupby(data_merged['Party']).describe()

fix, axes = plt.subplots(1,9,figsize = (20,5))

#gender
sns.boxplot(y = 'Percent Female', x = 'Party' ,data = data_merged, ax =
axes[0])

#age
sns.boxplot(y = 'Percent Age 29 and Under', x = 'Party' ,data = data_mer
ged, ax = axes[1])
sns.boxplot(y = 'Percent Age 65 and Older', x = 'Party' ,data = data_mer
ged, ax = axes[2])

#race
sns.boxplot(y = 'Percent White, not Hispanic or Latino', x = 'Party' ,da
ta = data_merged, ax = axes[3])
sns.boxplot(y = 'Percent Black, not Hispanic or Latino', x = 'Party' ,da
ta = data_merged, ax = axes[4])
sns.boxplot(y = 'Percent Hispanic or Latino', x = 'Party' ,data = data_m
erged, ax = axes[5])
sns.boxplot(y = 'Percent Foreign Born', x = 'Party' ,data = data_merged,
ax = axes[6])

#education

sns.boxplot(y = 'Percent Less than High School Degree', x = 'Party' ,dat
a = data_merged, ax = axes[7])
sns.boxplot(y = 'Percent Less than Bachelor\'s Degree', x = 'Party' ,dat
a = data_merged, ax = axes[8])

plt.tight_layout()

data_merged.groupby(data_merged['Party']).describe()
```

Out[16]:

| | FIPS | | | | | | | | Total Popul | |
|---|---|---|---|---|---|---|---|---|---|---|
| | count | mean | std | min | 25% | 50% | 75% | max | count | me |
| Party | | | | | | | | | | |
| 0 | 870.0 | 38714.074713 | 12658.615292 | 4003.0 | 30073.5 | 42040.0 | 48342.5 | 56043.0 | 870.0 | 53 |
| 1 | 325.0 | 37130.873846 | 13860.571592 | 4001.0 | 27027.0 | 36103.0 | 51095.0 | 56001.0 | 325.0 | 30( |



**Task 09 (of 10): (5 pts.) Based on your previous analysis, which variables in the dataset do you think are more important to determine whether a county is labeled as Democratic or Republican? Justify your answer.**

In [19]:
```
'Answer is explained in the report "Project1_report.pdf"'
```

Out[19]:
```
'Answer is explained in the report "Project1_report.pdf"'
```

**Task 10 (of 10): (10 pts.) Create a map of Democratic counties and Republican counties using the counties' FIPS codes and Python's Plotly library (plot.ly/python/county-choropleth/). Note that this dataset does not include all United States counties.**

In [18]:
```
fips = data_merged['FIPS'].tolist()
values = data_merged['Party'].tolist()
colorscale = ['rgb(244,109,67)', 'rgb(49,54,149)']
fig = ff.create_choropleth(
    colorscale=colorscale,
    fips=fips, values=values,
    title='Counties by Democratic/Republican',
    legend_title='1 = Democratic Counties, 0 = Republican Counties'
)
fig.layout.template = None
fig.show(sort=True)
```

In [ ]: