CS 418: Introduction to Data Science

Project 01: Exploratory Data Analysis - Fall 2020

Instructions:

This assignment is due Friday, October 16, at 11:59PM (Central Time). For this assignment, you must work in teams of three students. Each member of the team must be assigned one of three roles (project manager, scribe, or timekeeper) and everyone must switch roles in every project. Deliverables for this assignment (see Deliverables section below) must be submitted on Blackboard by the project manager. Only one submission per team is required. Additionally, every member of the team must submit a self- and peer-evaluation form. Late submissions will be accepted within 0-12 hours after the deadline with a 5-point penalty and within 12-24 hours after the deadline with a 20-point penalty. No late submissions will be accepted more than 24 hours after the deadline. Offering or receiving any kind of unauthorized or unacknowledged assistance in this assignment is a violation of the University's academic integrity policies, will result in a grade of zero for the assignment, and will be subject to disciplinary action.

Project Description

Given the following datasets:

• election_train.csv with results of the 2018 United States Senate elections, including the number of votes received by each party (Democratic or Republican).

• demographics_train.csv with demographic information for United States counties collected from 2012 to 2016 by the United States Census Bureau, including population, age, gender, race and ethnicity, education, income, and other statistics (www.census.gov/quickfacts/table/PST045215/00).

In [1]:
```
# Load libraries
import pandas as pd
import numpy
from scipy.stats import ttest_ind
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn import linear_model
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.figure_factory as ff
```

In [2]:
```
# United States of America Python Dictionary to translate States,
# Districts & Territories to Two-Letter codes and vice versa.
#
# https://gist.github.com/rogerallen/1583593
#
# Dedicated to the public domain.  To the extent possible under law,
# Roger Allen has waived all copyright and related or neighboring
# rights to this code.

us_state_abbrev = {
    'Alabama': 'AL',
    'Alaska': 'AK',
    'American Samoa': 'AS'
```

```
    'Arkansas': 'AR',
    'California': 'CA',
    'Colorado': 'CO',
    'Connecticut': 'CT',
    'Delaware': 'DE',
    'District of Columbia': 'DC',
    'Florida': 'FL',
    'Georgia': 'GA',
    'Guam': 'GU',
    'Hawaii': 'HI',
    'Idaho': 'ID',
    'Illinois': 'IL',
    'Indiana': 'IN',
    'Iowa': 'IA',
    'Kansas': 'KS',
    'Kentucky': 'KY',
    'Louisiana': 'LA',
    'Maine': 'ME',
    'Maryland': 'MD',
    'Massachusetts': 'MA',
    'Michigan': 'MI',
    'Minnesota': 'MN',
    'Mississippi': 'MS',
    'Missouri': 'MO',
    'Montana': 'MT',
    'Nebraska': 'NE',
    'Nevada': 'NV',
    'New Hampshire': 'NH',
    'New Jersey': 'NJ',
    'New Mexico': 'NM',
    'New York': 'NY',
    'North Carolina': 'NC',
    'North Dakota': 'ND',
    'Northern Mariana Islands':'MP',
    'Ohio': 'OH',
    'Oklahoma': 'OK',
    'Oregon': 'OR',
    'Pennsylvania': 'PA',
    'Puerto Rico': 'PR',
    'Rhode Island': 'RI',
    'South Carolina': 'SC',
    'South Dakota': 'SD',
    'Tennessee': 'TN',
    'Texas': 'TX',
    'Utah': 'UT',
    'Vermont': 'VT',
    'Virgin Islands': 'VI',
    'Virginia': 'VA',
    'Washington': 'WA',
    'West Virginia': 'WV',
    'Wisconsin': 'WI',
    'Wyoming': 'WY'
}

# thank you to @kinghelix and @trevormarburger for this idea
abbrev_us_state = dict(map(reversed, us_state_abbrev.items()))
```

In [3]:

```
# Load dataset and display the first five rows
demographics_data = pd.read_csv('demographics_train.csv')
demographics_data.head()
```

	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	Perce Age 2 ar Und
0	Wisconsin	La Crosse	55063	117538	0	90.537528	1.214075	1.724549	2.976059	51.171536	43.24133
1	Virginia	Alleghany	51005	15919	12705	91.940449	5.207614	1.432251	1.300333	51.077329	31.66028
2	Indiana	Fountain	18045	16741	12750	95.705155	0.400215	2.359477	1.547100	49.770026	35.89988
3	Ohio	Geauga	39055	94020	0	95.837056	1.256116	1.294405	2.578175	50.678579	36.28164
4	Wisconsin	Jackson	55053	20566	15835	86.662453	1.983857	3.082758	1.376058	46.649810	36.29291

In [4]:
```
# Load dataset and display the first five rows
election_data = pd.read_csv('election_train.csv')
print(election_data)
```

```
      Year State             County      Office       Party  Votes
0     2018    AZ      Apache County  US Senator  Democratic  16298
1     2018    AZ      Apache County  US Senator  Republican   7810
2     2018    AZ     Cochise County  US Senator  Democratic  17383
3     2018    AZ     Cochise County  US Senator  Republican  26929
4     2018    AZ    Coconino County  US Senator  Democratic  34240
...    ...   ...                ...         ...         ...    ...
2400  2018    WY  Sweetwater County  US Senator  Republican   8577
2401  2018    WY       Uinta County  US Senator  Democratic   1371
2402  2018    WY       Uinta County  US Senator  Republican   4713
2403  2018    WY    Washakie County  US Senator  Democratic    588
2404  2018    WY    Washakie County  US Senator  Republican   2423

[2405 rows x 6 columns]
```

1. (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]:
```
election_data = pd.pivot_table(election_data, values = 'Votes',
                               index =['Year', 'State', 'County', 'Office'], columns = 'Party').rese
print(election_data)
```

```
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
...     ...   ...                ...         ...          ...         ...
1200   2018    WY      Platte County  US Senator        801.0      2850.0
1201   2018    WY    Sublette County  US Senator        668.0      2653.0
1202   2018    WY  Sweetwater County  US Senator       3943.0      8577.0
1203   2018    WY       Uinta County  US Senator       1371.0      4713.0
1204   2018    WY    Washakie County  US Senator        588.0      2423.0

[1205 rows x 6 columns]
```

2. (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]:
```
                                                    ta['State'].map(abbrev_us_state)
```

```
banned = ['County']
f = lambda x: ' '.join([item for item in x.split() if item not in banned])
election_data['County'] = election_data['County'].apply(f)
election_data['County'] = election_data['County'].str.lower()
demographics_data['County'] = demographics_data['County'].str.lower()
merged_data = election_data.merge(demographics_data, how = 'inner', on=['State', 'County'], sort =
merged_data['County']= merged_data['County'].str.title()
merged_data.drop_duplicates()
```

Out[6]:

	Year	State	County	Office	Democratic	Republican	FIPS	Total Population	Citizen Voting-Age Population	Percent White, not Hispanic or Latino	...
0	2018	Arizona	Apache	US Senator	16298.0	7810.0	4001	72346	0	18.571863	...
1	2018	Arizona	Cochise	US Senator	17383.0	26929.0	4003	128177	92915	56.299492	...
2	2018	Arizona	Coconino	US Senator	34240.0	19249.0	4005	138064	104265	54.619597	...
3	2018	Arizona	Gila	US Senator	7643.0	12180.0	4007	53179	0	63.222325	...
4	2018	Arizona	Graham	US Senator	3368.0	6870.0	4009	37529	0	51.461536	...
...	...	...	...	...	...	...	...	...	...	...	...
1195	2018	Wyoming	Platte	US Senator	801.0	2850.0	56031	8740	6830	89.359268	...
1196	2018	Wyoming	Sublette	US Senator	668.0	2653.0	56035	10032	0	91.646730	...
1197	2018	Wyoming	Sweetwater	US Senator	3943.0	8577.0	56037	44812	30565	79.815674	...
1198	2018	Wyoming	Uinta	US Senator	1371.0	4713.0	56041	20893	14355	87.718375	...
1199	2018	Wyoming	Washakie	US Senator	588.0	2423.0	56043	8351	0	82.397318	...

1200 rows × 21 columns

3. (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?

Answer: Since there are 21 columns in our merged dataset, this means that these are our variables, so we have 21 variables in our merged dataset. To check the types of these variables, we used the info() function to determine the types. From this, we figured that there are 13 float64 variables, 5 int64 variables, and 3 object variables. Our Year, FIPS, Total Population, Citizen Voting-Age Population, and Median Household Income columns are all int64 type variables. Our State, County and Office columns are all object type variables. The rest of the columns that weren't mentioned are all float64 type variables. There are a few variables that we feel are

o not end up using them in Tasks 1-10 such as "Year", "Citizen

Voting-Age Population", and "Office". Since we ended up merging our 2018 election data with the 2012-2016 demographics data, our year is unnecessary and it's redundant to have every row say "2018" in the Year column. This is basically the same case for the "Office" column. We don't use this column anywhere, and it's being redundant by having every row say "US Senator". For our "Citizen Voting-Age Population", we don't use this information anywhere for Tasks 1-10, so it's also not needed. Therefore, to deal with these variables, we will remove these columns from our dataset. So our merged dataset will now be 1200 rows x 18 columns since we removed 3 of these variables.

In [7]:
```
del merged_data['Year']
```

In [8]:
```
del merged_data['Office']
```

In [9]:
```
del merged_data['Citizen Voting-Age Population']
merged_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1200 entries, 0 to 1199
Data columns (total 18 columns):
 #   Column                                Non-Null Count  Dtype  
---  ------                                --------------  -----  
 0   State                                 1200 non-null   object 
 1   County                                1200 non-null   object 
 2   Democratic                            1197 non-null   float64
 3   Republican                            1198 non-null   float64
 4   FIPS                                  1200 non-null   int64  
 5   Total Population                      1200 non-null   int64  
 6   Percent White, not Hispanic or Latino 1200 non-null   float64
 7   Percent Black, not Hispanic or Latino 1200 non-null   float64
 8   Percent Hispanic or Latino            1200 non-null   float64
 9   Percent Foreign Born                  1200 non-null   float64
 10  Percent Female                        1200 non-null   float64
 11  Percent Age 29 and Under              1200 non-null   float64
 12  Percent Age 65 and Older              1200 non-null   float64
 13  Median Household Income               1200 non-null   int64  
 14  Percent Unemployed                    1200 non-null   float64
 15  Percent Less than High School Degree  1200 non-null   float64
 16  Percent Less than Bachelor's Degree   1200 non-null   float64
 17  Percent Rural                         1200 non-null   float64
dtypes: float64(13), int64(3), object(2)
memory usage: 178.1+ KB
```

4. (10 pts.) Search the merged dataset for missing values. Are there any missing values? If so, how will you deal with these values?

Answer: After searching the merged dataset for missing values, we noticed that there were a few missing values in our Democratic and Republican columns. To deal with these values, we decided that we will replace the missing values with 0. This is to indicate that the county doesn't have any Democrats or Republicans. So basically, if there is a row that has a value in Democratic column, but it has a 0 in the Republican column, this means that the county is full Democratic. This will be the same case if it was the other way around. So then if a county has a 0 in Democratic, then the county is full Republican.

In [10]:
```
merged_data.replace(numpy.nan, 0)
merged_data.head(1200)
```

Out[10]:

	State	County	Democratic	Republican	FIPS	Total Population	Percent White, not Hispanic or Latino	Percent Black, not Hispanic or Latino	Percent Hispanic or Latino	Percent Foreign Born
0	Arizona	Apache	16298.0	7810.0	4001	72346	18.571863	0.486551	5.947806	1.719515
1	Arizona	Cochise	17383.0	26929.0	4003	128177	56.299492	3.714395	34.403208	11.458374
2	Arizona	Coconino	34240.0	19249.0	4005	138064	54.619597	1.342855	13.711033	4.825298
3	Arizona	Gila	7643.0	12180.0	4007	53179	63.222325	0.552850	18.548675	4.249798
4	Arizona	Graham	3368.0	6870.0	4009	37529	51.461536	1.811932	32.097844	4.385942
...	...	...	...	...	...	...	...	...	...	...
1195	Wyoming	Platte	801.0	2850.0	56031	8740	89.359268	0.057208	7.814645	2.780320
1196	Wyoming	Sublette	668.0	2653.0	56035	10032	91.646730	0.000000	7.814992	2.053429
1197	Wyoming	Sweetwater	3943.0	8577.0	56037	44812	79.815674	0.865840	15.859591	5.509685
1198	Wyoming	Uinta	1371.0	4713.0	56041	20893	87.718375	0.186665	8.959939	3.986981
1199	Wyoming	Washakie	588.0	2423.0	56043	8351	82.397318	0.790325	13.962400	3.783978

1200 rows × 18 columns

5. (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 [11]:
```
merged_data['Party']= numpy.where(merged_data['Democratic'] > merged_data['Republican'], 1, 0)
merged_data
```

Out[11]:

	State	County	Democratic	Republican	FIPS	Total Population	Percent White, not Hispanic or Latino	Percent Black, not Hispanic or Latino	Percent Hispanic or Latino	Percent Foreign Born
0	Arizona	Apache	16298.0	7810.0	4001	72346	18.571863	0.486551	5.947806	1.719515
1	Arizona	Cochise	17383.0	26929.0	4003	128177	56.299492	3.714395	34.403208	11.458374
2	Arizona	Coconino	34240.0	19249.0	4005	138064	54.619597	1.342855	13.711033	4.825298
3	Arizona	Gila	7643.0	12180.0	4007	53179	63.222325	0.552850	18.548675	4.249798
4	Arizona	Graham	3368.0	6870.0	4009	37529	51.461536	1.811932	32.097844	4.385942
...	...	...	...	...	...	...	...	...	...	...
1195	Wyoming	Platte	801.0	2850.0	56031	8740	89.359268	0.057208	7.814645	2.780320
1196	Wyoming	Sublette	668.0	2653.0	56035	10032	91.646730	0.000000	7.814992	2.053429
1197	Wyoming	Sweetwater	3943.0	8577.0	56037	44812	79.815674	0.865840	15.859591	5.509685
1198	Wyoming	Uinta	1371.0	4713.0	56041	20893	87.718375	0.186665	8.959939	3.986981

	State	County	Democratic	Republican	FIPS	Total Population	Percent White, not Hispanic or Latino	Percent Black, not Hispanic or Latino	Percent Hispanic or Latino	Percent Foreign Born
1199	Wyoming	Washakie	588.0	2423.0	56043	8351	82.397318	0.790325	13.962400	3.783978

1200 rows × 19 columns

6. (10 pts.) Compute the mean median household income for Democratic counties and Republican counties.

In [12]:
```
democratic = merged_data[merged_data['Party'] == 1]
mean1 = democratic['Median Household Income'].mean()
mean1
```

Out[12]: 53798.732307692306

In [13]:
```
republican = merged_data[merged_data['Party'] == 0]
mean2 = republican['Median Household Income'].mean()
mean2
```

Out[13]: 48724.15085714286

Which one is higher?

Answer: The Democratic mean median household income is higher than the Republican mean median household income.

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 [14]:
```
[ttestval, pval] = ttest_ind(democratic['Median Household Income'],republican['Median Household Ir

if pval < 0.05:
  print("pval:", pval/2, "\nwe reject null hypothesis")
else:
  print("pval:", pval/2, "\nwe accept null hypothesis")
```

```
pval: 3.0866199456151866e-08
we reject null hypothesis
```

7. (10 pts.) Compute the mean population for Democratic counties and Republican counties.

In [15]:
```
democratic = merged_data.loc[merged_data['Party'] == 1]
democratic['Total Population'].mean()
```

Out[15]: 300998.3169230769

In [16]:
```
republican = merged_data.loc[merged_data['Party'] == 0]
republican['Total Population'].mean()
```

Out[16]: 53974.214857142855

Which one is higher?

Answer: The Democratic total population is higher than the Republican total population.

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 [17]:
```
[ttestval, pval] = ttest_ind(democratic['Total Population'],republican['Total Population'], equal_

if pval < 0.05:
  print("pval:", pval/2, "\nwe reject null hypothesis")
else:
  print("pval:", pval/2, "\nwe accept null hypothesis")
```

```
pval: 1.0482859676754979e-14
we reject null hypothesis
```

8. (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 [18]:
```
merged_data[merged_data['Party'] == 1].describe()
```

Out[18]:

	Democratic	Republican	FIPS	Total Population	Percent White, not Hispanic or Latino	Percent Black, not Hispanic or Latino	Percent Hispanic or Latino	Percent Foreign Born	
count	325.000000	325.000000	325.000000	3.250000e+02	325.000000	325.000000	325.000000	325.000000	3
mean	71193.172308	41322.861538	37130.873846	3.009983e+05	69.683766	9.242649	12.587391	7.986330	
std	125306.803889	74689.108440	13860.571592	5.536000e+05	24.981502	13.351340	19.575030	8.330740	
min	521.000000	220.000000	4001.000000	1.969000e+03	2.776702	0.000000	0.193349	0.179769	
25%	5242.000000	3611.000000	27027.000000	2.364500e+04	53.271579	0.839103	2.531017	2.470508	
50%	18159.000000	12348.000000	36103.000000	8.204900e+04	77.786090	3.485992	5.039747	5.105490	
75%	72677.000000	46403.000000	51095.000000	2.847880e+05	90.300749	11.058843	11.857116	10.144555	
max	881802.000000	672505.000000	56001.000000	4.434257e+06	98.063495	63.953279	95.479801	52.229868	

In [19]:
```
merged_data[merged_data['Party'] == 0].describe()
```

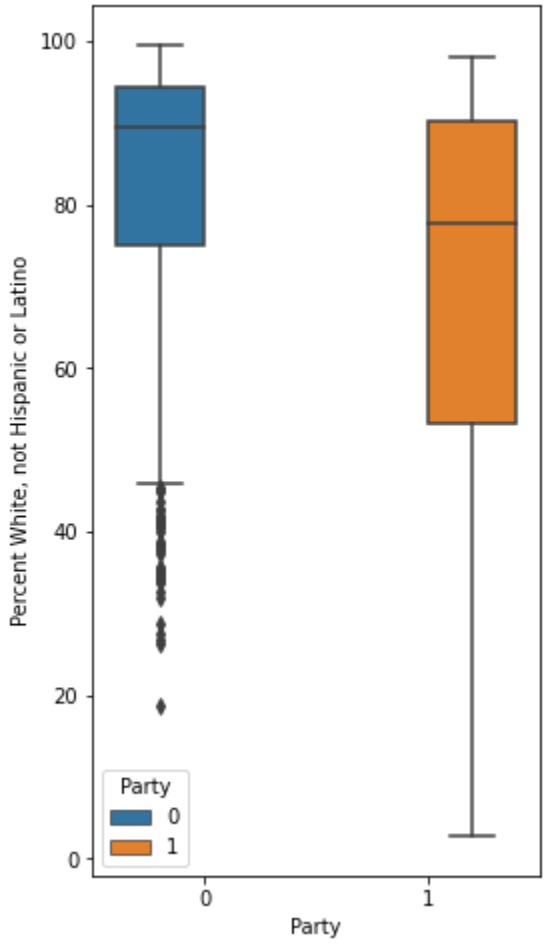
Out[19]:

	Democratic	Republican	FIPS	Total Population	Percent White, not Hispanic or Latino	Percent Black, not Hispanic or Latino	Percent Hispanic or Latino	Percent Foreign Born	
count	872.000000	873.000000	875.000000	8.750000e+02	875.000000	875.000000	875.000000	875.000000	8
mean	7915.712156	12661.404353	38755.305143	5.397421e+04	82.597026	4.182092	9.801825	3.989607	
std	17519.971129	22602.919685	12648.319628	9.433409e+04	16.134097	6.706383	14.144003	4.497946	
min	6.000000	46.000000	4003.000000	7.600000e+01	18.758977	0.000000	0.000000	0.000000	
25%	958.500000	2542.000000	30076.000000	9.565000e+03	74.960538	0.460803	1.704640	1.320845	
50%	2899.500000	5833.000000	42047.000000	2.540300e+04	89.418396	1.318775	3.440794	2.326782	

	Democratic	Republican	FIPS	Total Population	Percent White, not Hispanic or Latino	Percent Black, not Hispanic or Latino	Percent Hispanic or Latino	Percent Foreign Born
75%	7000.250000	12637.000000	48342.000000	5.363400e+04	94.468872	4.750447	10.785963	5.139964
max	215190.000000	219990.000000	56043.000000	1.092518e+06	99.627329	41.563041	78.397012	37.058317

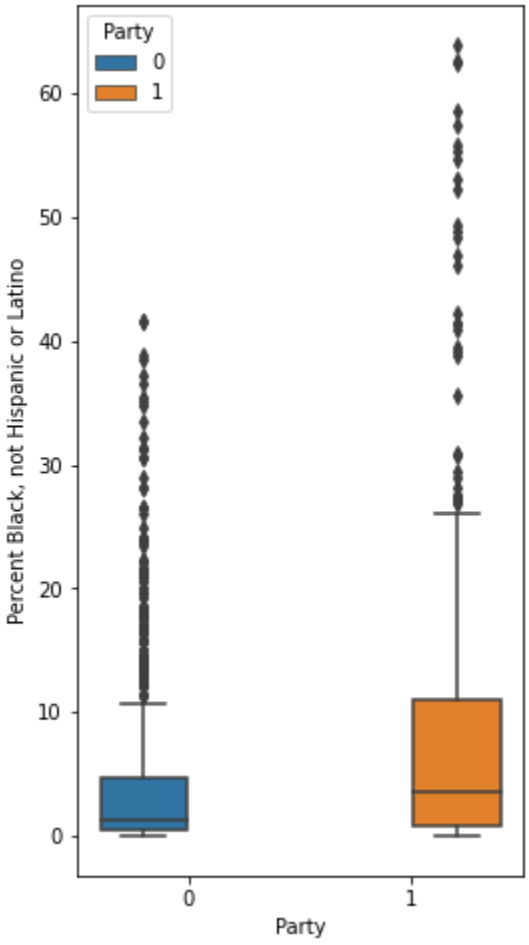
In [20]:
```
plt.figure(figsize=(4,8))
sns.boxplot(x = 'Party', y = 'Percent White, not Hispanic or Latino', hue = 'Party', data = merged
```

Out[20]: <AxesSubplot:xlabel='Party', ylabel='Percent White, not Hispanic or Latino'>



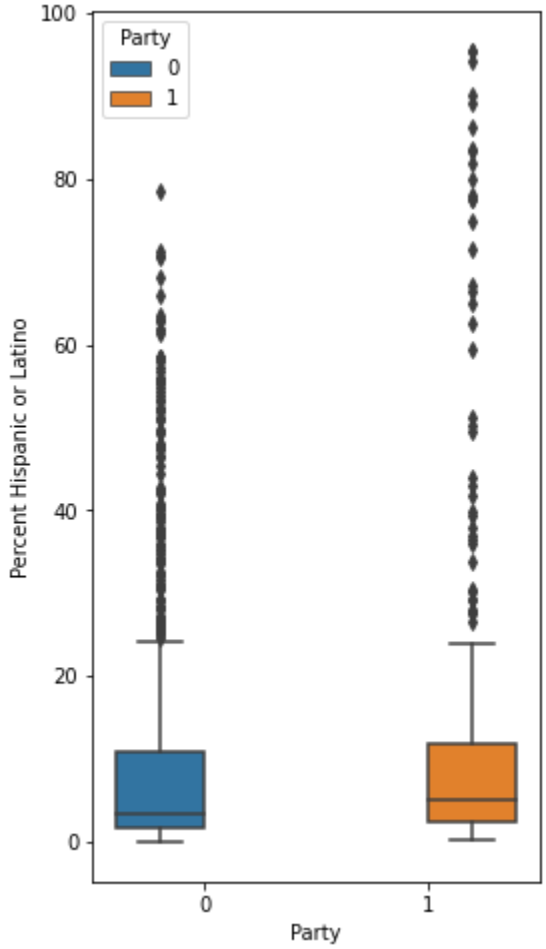
In [21]:
```
plt.figure(figsize=(4,8))
sns.boxplot(x = 'Party', y = 'Percent Black, not Hispanic or Latino', hue = 'Party', data = merged
```

Out[21]: <AxesSubplot:xlabel='Party', ylabel='Percent Black, not Hispanic or Latino'>



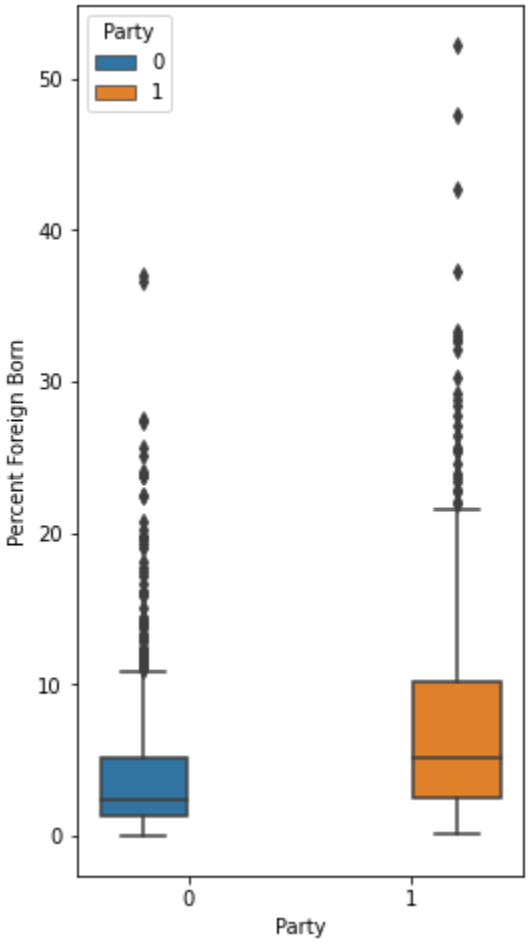
In [22]:
```
plt.figure(figsize=(4,8))
sns.boxplot(x = 'Party', y = 'Percent Hispanic or Latino', hue = 'Party', data = merged_data)
```

Out[22]:
```
<AxesSubplot:xlabel='Party', ylabel='Percent Hispanic or Latino'>
```



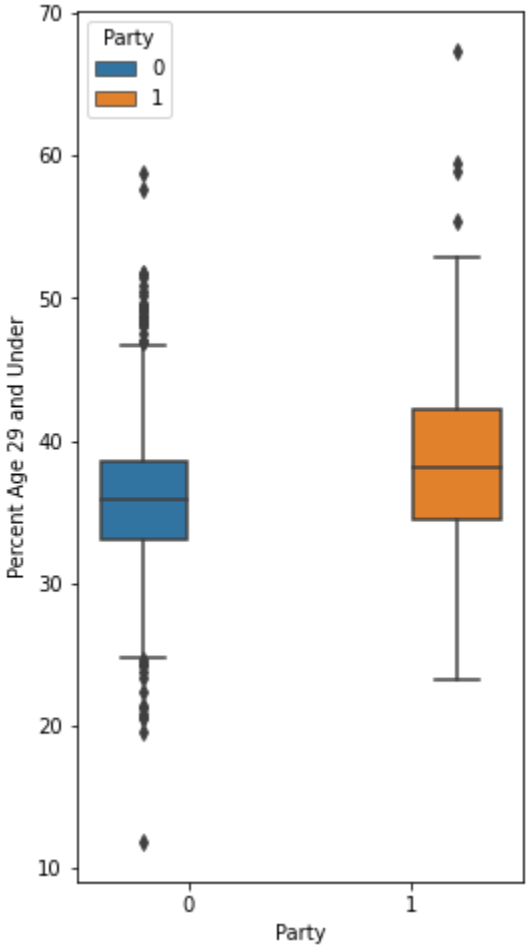
In [23]:
```
plt.figure(figsize=(4,8))
sns.boxplot(x = 'Party', y = 'Percent Foreign Born', hue = 'Party', data = merged_data)
```

Out[23]:
```
<AxesSubplot:xlabel='Party', ylabel='Percent Foreign Born'>
```



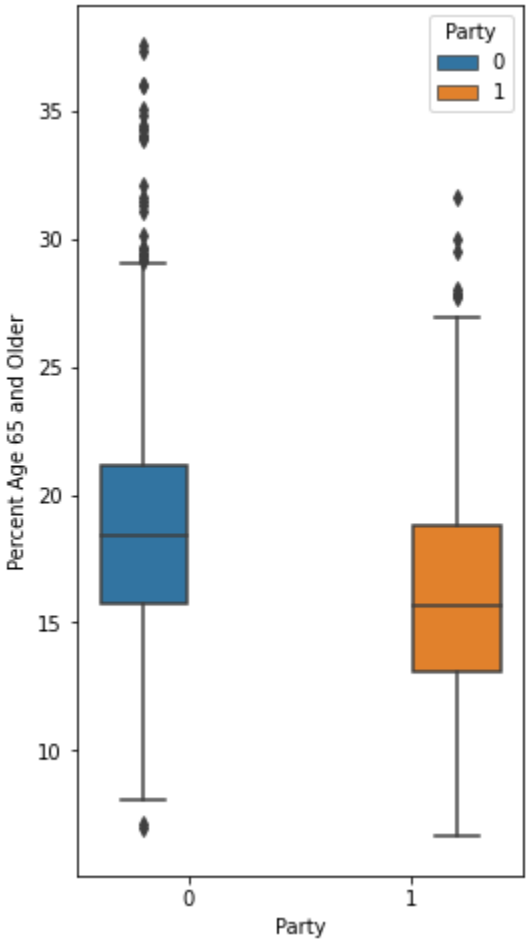
In [24]:
```
plt.figure(figsize=(4,8))
sns.boxplot(x = 'Party', y = 'Percent Age 29 and Under', hue = 'Party', data = merged_data)
```

Out[24]:
```
<AxesSubplot:xlabel='Party', ylabel='Percent Age 29 and Under'>
```



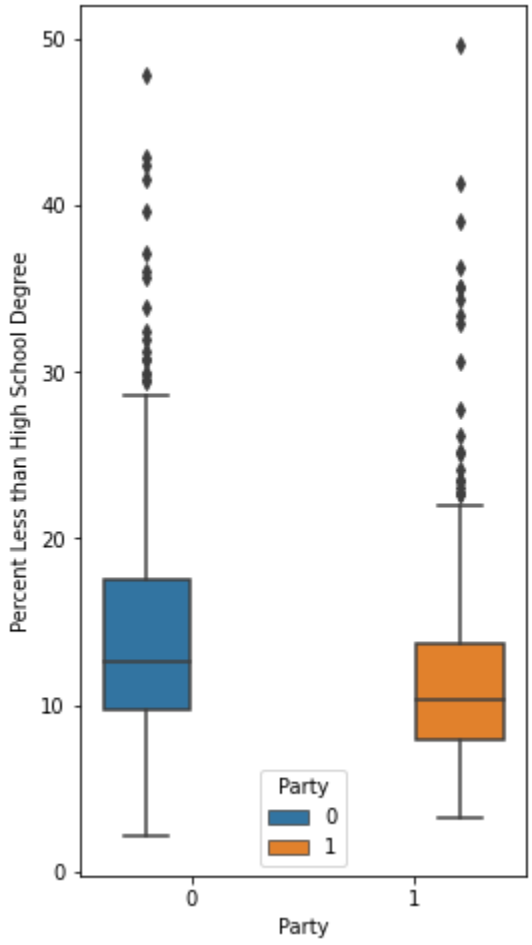
In [25]:
```
plt.figure(figsize=(4,8))
sns.boxplot(x = 'Party', y = 'Percent Age 65 and Older', hue = 'Party', data = merged_data)
```

Out[25]:
```
<AxesSubplot:xlabel='Party', ylabel='Percent Age 65 and Older'>
```



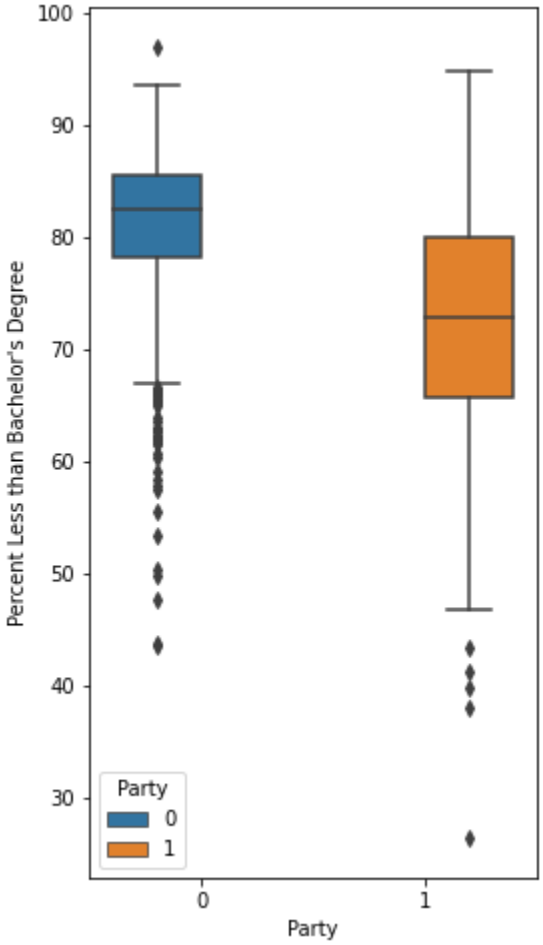
In [26]:
```
plt.figure(figsize=(4,8))
sns.boxplot(x = 'Party', y = 'Percent Less than High School Degree', hue = 'Party', data = merged_
```

Out[26]:
```
<AxesSubplot:xlabel='Party', ylabel='Percent Less than High School Degree'>
```



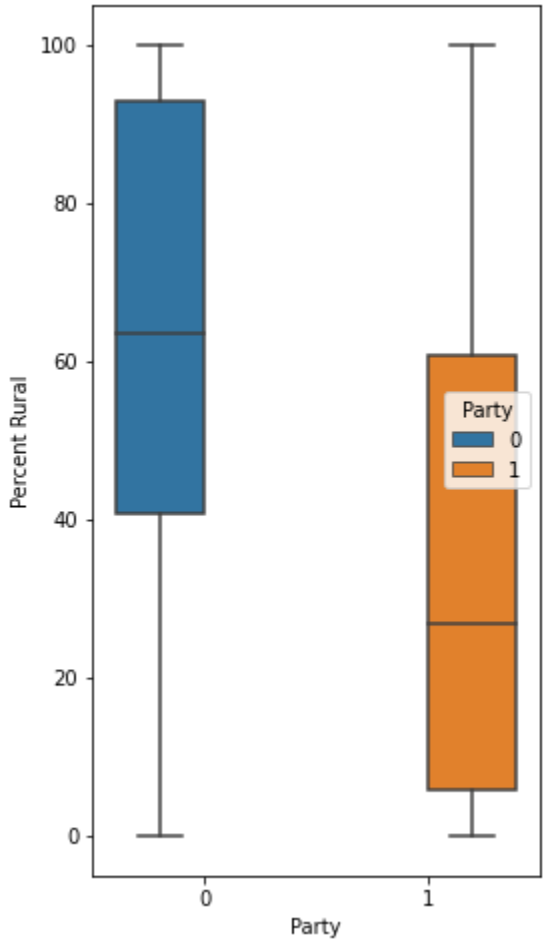
In [27]:
```
plt.figure(figsize=(4,8))
sns.boxplot(x = 'Party', y = 'Percent Less than Bachelor\'s Degree', hue = 'Party', data = merged_
```

Out[27]:
```
<AxesSubplot:xlabel='Party', ylabel="Percent Less than Bachelor's Degree">
```



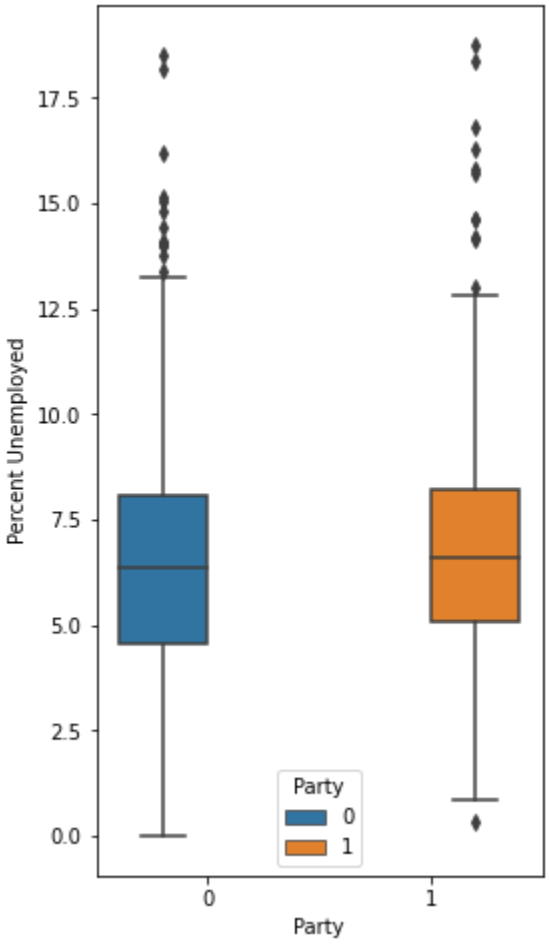
In [28]:
```
plt.figure(figsize=(4,8))
sns.boxplot(x = 'Party', y = 'Percent Rural', hue = 'Party', data = merged_data)
```

Out[28]:
```
<AxesSubplot:xlabel='Party', ylabel='Percent Rural'>
```



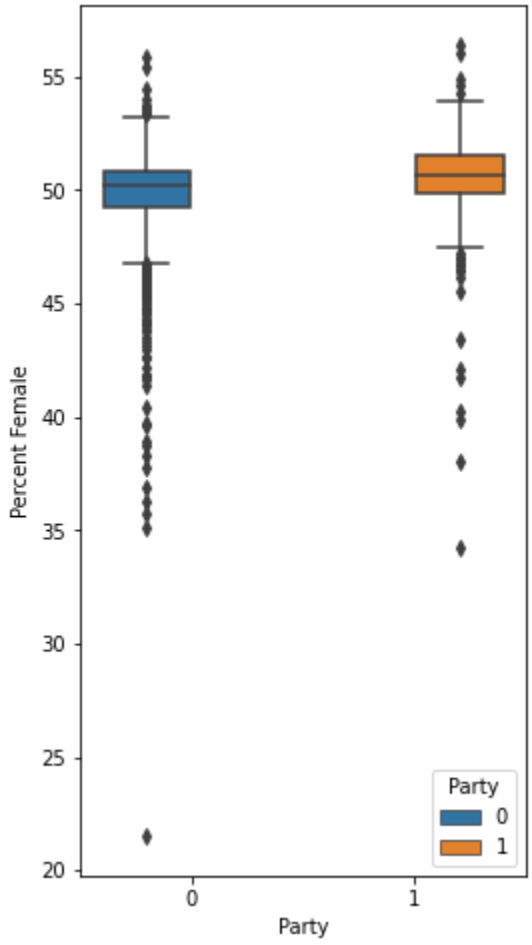
In [29]:
```
plt.figure(figsize=(4,8))
sns.boxplot(x = 'Party', y = 'Percent Unemployed', hue = 'Party', data = merged_data)
```

Out[29]:
```
<AxesSubplot:xlabel='Party', ylabel='Percent Unemployed'>
```



In [30]:
```
plt.figure(figsize=(4,8))
sns.boxplot(x = 'Party', y = 'Percent Female', hue = 'Party', data = merged_data)
```

Out[30]:
```
<AxesSubplot:xlabel='Party', ylabel='Percent Female'>
```



9. (5 pts.) Based on your results for tasks 6-8, 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.

Answer: Based on our results from tasks 6-8, we just need the Party, Democratic, and Republican variables in our merged_data. The Democratic and Republican variables shows how much of the county is Democratic or Republican, while our Party variable determines which counties are mostly Democratic or Republican by checking if the Democratic variable is greater than the Republican variable in each cell. If Democratic is greater than Republican, then it will place a "1" in the Party column in that County row. Otherwise, if Republican is greater than Democratic in the County, then it will place a "0" in the Party column in that County row. This is to distinguish between which county is Democratic or Republican.

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.

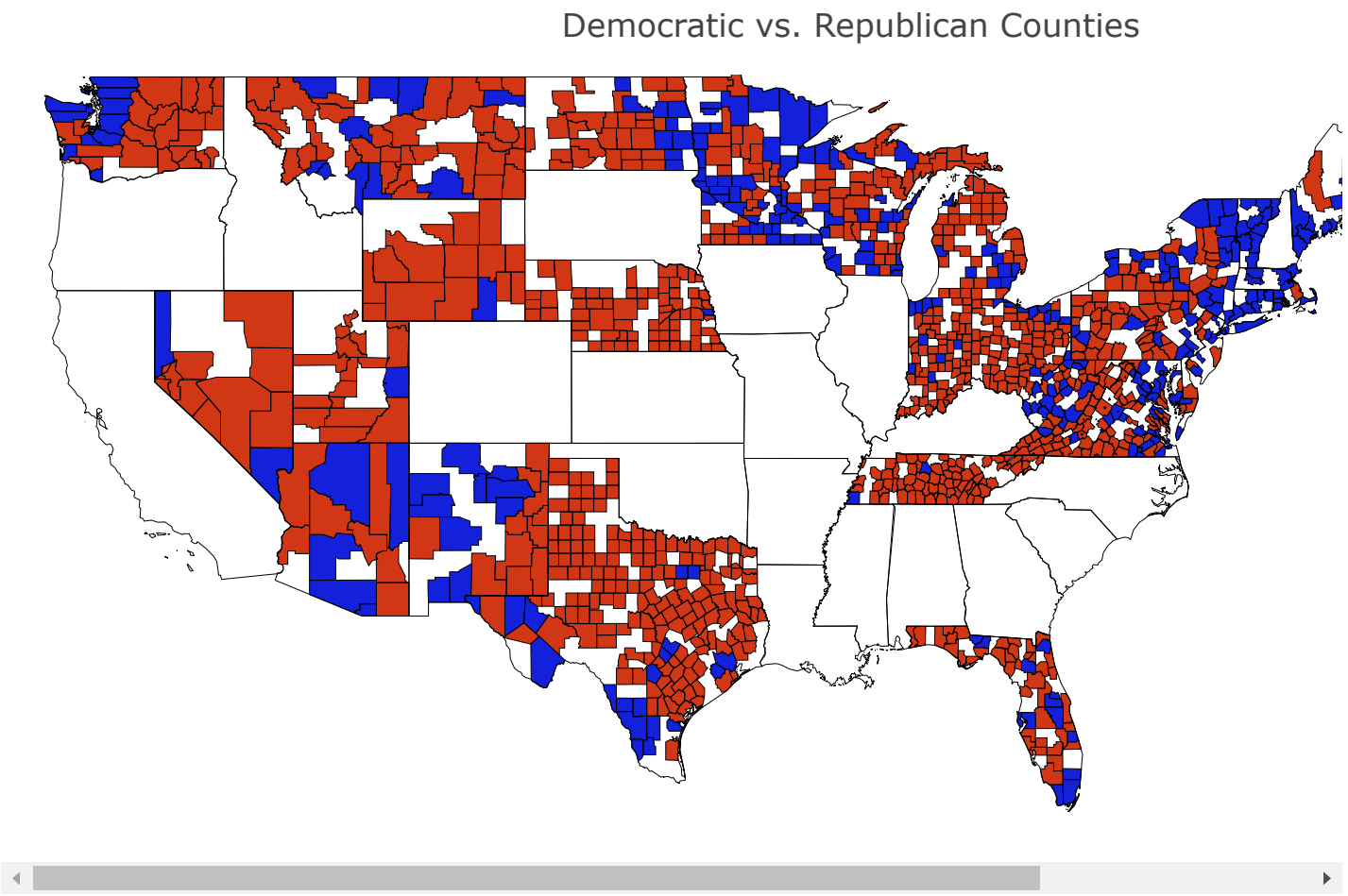
In [31]:
```
merged_data['FIPS'] = merged_data['FIPS'].apply(lambda x: str(x).zfill(3))

colorscale = ["#d13715", "#1421db"]
endpts = list(numpy.linspace(1, 12, len(colorscale) - 1))
fips = merged_data['FIPS'].tolist()
values = merged_data['Party'].tolist()

fig = ff.create_choropleth(
    fips=fips, values=values,
    colorscale=colorscale,
    show_state_data=True,
    show_hover=True, centroid_marker={'opacity': 0},
    'width': 0.5},
    county_outline={'color': 'black', 'width': 0.5},
```

```
    asp=2.9, title='Democratic vs. Republican Counties',
    legend_title='Party'
)

fig.layout.template = None
fig.show()
```



/