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

In [2]:
```
# Load dataset from demographics csv and display first five rows
demo_data = pd.read_csv('demographics_train.csv')
demo_data.head()
```

Out[2]:

	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	Wisconsin	La Crosse	55063	117538	0	90.537528	1.214075	1.724549	2.976059
1	Virginia	Alleghany	51005	15919	12705	91.940449	5.207614	1.432251	1.300333
2	Indiana	Fountain	18045	16741	12750	95.705155	0.400215	2.359477	1.547100
3	Ohio	Geauga	39055	94020	0	95.837056	1.256116	1.294405	2.578175
4	Wisconsin	Jackson	55053	20566	15835	86.662453	1.983857	3.082758	1.376058

In [3]:
```
# Load dataset from election csv and display first five rows
elec_data = pd.read_csv('election_train.csv')
elec_data.head()
```

Out[3]:

	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

In [4]:
```
# 1. Reshape dataset election_train from long format to wide format. Hin
t: the
#    reshaped dataset should contain 1205 rows and 6 columns
elec_data_tidy = pd.pivot_table(elec_data, index=['Year', 'State', 'Coun
ty', 'Office'], values='Votes', columns='Party').reset_index()
elec_data_tidy.head()
```

Out[4]:

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

In [5]:
```
# 2. Remove substring 'County' if it exists in any of the 'County' data
 from election table
elec_data_tidy['County'] = elec_data_tidy['County'].apply(lambda x: x.re
place(' County', ''))
elec_data_tidy.head()
```

Out[5]:

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

In [6]:
```
# 2. Replace the abbreviated 'State' data from demographics table w/ its
full abbreviation
change_states = {
    "AL": "Alabama",
    "AK": "Alaska",
    "AS": "American Samoa",
    "AZ": "Arizona",
    "AR": "Arkansas",
    "CA": "California",
    "CO": "Colorado",
    "CT": "Connecticut",
    "DE": "Delaware",
    "DC": "District Of Columbia",
    "FM": "Federated States Of Micronesia",
    "FL": "Florida",
    "GA": "Georgia",
    "GU": "Guam",
    "HI": "Hawaii",
    "ID": "Idaho",
    "IL": "Illinois",
    "IN": "Indiana",
    "IA": "Iowa",
    "KS": "Kansas",
    "KY": "Kentucky",
    "LA": "Louisiana",
    "ME": "Maine",
    "MH": "Marshall Islands",
    "MD": "Maryland",
    "MA": "Massachusetts",
    "MI": "Michigan",
    "MN": "Minnesota",
    "MS": "Mississippi",
    "MO": "Missouri",
    "MT": "Montana",
    "NE": "Nebraska",
    "NV": "Nevada",
    "NH": "New Hampshire",
    "NJ": "New Jersey",
    "NM": "New Mexico",
    "NY": "New York",
    "NC": "North Carolina",
    "ND": "North Dakota",
    "MP": "Northern Mariana Islands",
    "OH": "Ohio",
    "OK": "Oklahoma",
    "OR": "Oregon",
    "PW": "Palau",
    "PA": "Pennsylvania",
    "PR": "Puerto Rico",
    "RI": "Rhode Island",
    "SC": "South Carolina",
    "SD": "South Dakota",
    "TN": "Tennessee",
    "TX": "Texas",
    "UT": "Utah",
    "VT": "Vermont",
```

```
    "VI": "Virgin Islands",
    "VA": "Virginia",
    "WA": "Washington",
    "WV": "West Virginia",
    "WI": "Wisconsin",
    "WY": "Wyoming"
}
elec_data_tidy['State'] = elec_data_tidy['State'].map(change_states)
elec_data_tidy.head()
```

Out[6]:

	Party	Year	State	County	Office	Democratic	Republican
0		2018	Arizona	Apache	US Senator	16298.0	7810.0
1		2018	Arizona	Cochise	US Senator	17383.0	26929.0
2		2018	Arizona	Coconino	US Senator	34240.0	19249.0
3		2018	Arizona	Gila	US Senator	7643.0	12180.0
4		2018	Arizona	Graham	US Senator	3368.0	6870.0

In [7]:
```
# 2. Lowercase all 'State' data from both tables
elec_data_tidy['County'] = elec_data_tidy['County'].str.lower()
demo_data['County'] = demo_data['County'].str.lower()
```

In [8]:
```
# 2. Merge reshaped dataset election_train with dataset demographics_tra
in
data = pd.merge(elec_data_tidy, demo_data, how='inner', on=['State', 'Co
unty'])
data.head()
```

Out[8]:

	Year	State	County	Office	Democratic	Republican	FIPS	Total Population	Citizen Voting-Age Population	Per W Hisp or La
0	2018	Arizona	apache	US Senator	16298.0	7810.0	4001	72346	0	18.57
1	2018	Arizona	cochise	US Senator	17383.0	26929.0	4003	128177	92915	56.29
2	2018	Arizona	coconino	US Senator	34240.0	19249.0	4005	138064	104265	54.61
3	2018	Arizona	gila	US Senator	7643.0	12180.0	4007	53179	0	63.22
4	2018	Arizona	graham	US Senator	3368.0	6870.0	4009	37529	0	51.46

5 rows × 21 columns

In [9]:
```
# 3. Explore the merged dataset. How many variables does the dataset have? What is the type of these variables?
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1200 entries, 0 to 1199
Data columns (total 21 columns):
 #   Column                                 Non-Null Count  Dtype  
---  ------                                 --------------  -----  
 0   Year                                   1200 non-null   int64  
 1   State                                  1200 non-null   object 
 2   County                                 1200 non-null   object 
 3   Office                                 1200 non-null   object 
 4   Democratic                             1197 non-null   float64
 5   Republican                             1198 non-null   float64
 6   FIPS                                   1200 non-null   int64  
 7   Total Population                       1200 non-null   int64  
 8   Citizen Voting-Age Population          1200 non-null   int64  
 9   Percent White, not Hispanic or Latino  1200 non-null   float64
 10  Percent Black, not Hispanic or Latino  1200 non-null   float64
 11  Percent Hispanic or Latino             1200 non-null   float64
 12  Percent Foreign Born                   1200 non-null   float64
 13  Percent Female                         1200 non-null   float64
 14  Percent Age 29 and Under               1200 non-null   float64
 15  Percent Age 65 and Older               1200 non-null   float64
 16  Median Household Income                1200 non-null   int64  
 17  Percent Unemployed                     1200 non-null   float64
 18  Percent Less than High School Degree   1200 non-null   float64
 19  Percent Less than Bachelor's Degree    1200 non-null   float64
 20  Percent Rural                          1200 non-null   float64
dtypes: float64(13), int64(5), object(3)
memory usage: 206.2+ KB
```

In [10]:
```
# 3. Are there any any irrelevant or redundant variables? If so, how will you deal with these variables?
# Drop 'Year' and 'Office' features
data = data.drop(columns=['Year', 'Office'])
data.head()
```

Out[10]:

	State	County	Democratic	Republican	FIPS	Total Population	Citizen Voting-Age Population	Percent White, not Hispanic or Latino	Percent Black, not Hispanic or Latino
0	Arizona	apache	16298.0	7810.0	4001	72346	0	18.571863	0.486551
1	Arizona	cochise	17383.0	26929.0	4003	128177	92915	56.299492	3.714395
2	Arizona	coconino	34240.0	19249.0	4005	138064	104265	54.619597	1.342855
3	Arizona	gila	7643.0	12180.0	4007	53179	0	63.222325	0.552850
4	Arizona	graham	3368.0	6870.0	4009	37529	0	51.461536	1.811932

In [11]:
```
# 4. Search the merged dataset for missing values.
data[data.isna().any(axis=1)]
```

Out[11]:

	State	County	Democratic	Republican	FIPS	Total Population	Citizen Voting-Age Population	Percent White, not Hispanic or Latino	P Hi
425	Nebraska	lancaster	NaN	49449.0	31109	301707	0	82.659667	3.7
714	Tennessee	meigs	NaN	2694.0	47121	11804	0	94.713656	1.3
750	Texas	bee	2811.0	NaN	48025	32706	0	32.660674	7.9
865	Texas	menard	NaN	632.0	48327	2163	0	56.310680	1.2
1114	Wisconsin	lafayette	3592.0	NaN	55065	16793	0	94.771631	0.3

In [12]:
```
# 4. Are there any missing values? If so, how will you deal with these values?
# Fill forward on missing values
data = data.fillna(method = 'ffill')
data[data.isna().any(axis=1)]
```

Out[12]:

	State	County	Democratic	Republican	FIPS	Total Population	Citizen Voting-Age Population	Percent White, not Hispanic or Latino	Percent Black, not Hispanic or Latino	Per Hisp La

In [13]:
```
# 5. 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.
# data['Party'] = np.where(data['Democratic'] >= data['Republican'], 1, 0) // ALTERNATIVE METHOD
data.loc[data['Democratic'] > data['Republican'], 'Party'] = 1
data.loc[data['Democratic'] < data['Republican'], 'Party'] = 0
data.head()
```

Out[13]:

	State	County	Democratic	Republican	FIPS	Total Population	Citizen Voting-Age Population	Percent White, not Hispanic or Latino	Percent Black, not Hispanic or Latino
0	Arizona	apache	16298.0	7810.0	4001	72346	0	18.571863	0.486551
1	Arizona	cochise	17383.0	26929.0	4003	128177	92915	56.299492	3.714395
2	Arizona	coconino	34240.0	19249.0	4005	138064	104265	54.619597	1.342855
3	Arizona	gila	7643.0	12180.0	4007	53179	0	63.222325	0.552850
4	Arizona	graham	3368.0	6870.0	4009	37529	0	51.461536	1.811932

In [14]:
```
# 6. Compute the mean median household income for Democratic counties an
d Republican counties. Which one is higher?

# Democratic Mean Median Household Income
demMean = data.loc[data['Party'].isin(["1.0"]), 'Median Household Incom
e'].mean()
print("Democratic Mean Median Household Income:",demMean)

# Republican Mean Median Household Income
repMean = data.loc[data['Party'].isin(["0.0"]), 'Median Household Incom
e'].mean()
print("Republican Mean Median Household Income:",repMean)

# Perform a hypothesis test to determine whether this difference is stat
istically significant at the α = 0.05 significance level.
dems = data.loc[data['Party'].isin(["1.0"]), 'Median Household Income']
reps = data.loc[data['Party'].isin(["0.0"]), 'Median Household Income']

# What is the result of the test? What conclusion do you make from this
 result?
stats, p = ttest_ind(dems, reps, equal_var = False)

print('Test Statistic = %.3f, p-value=%.10f' % (stats, p))

if p > 0.05:
    print('Same distributions....fail to reject null hypothesis)')

else:
    print('Different distributions.....reject null hypothesis')
```

```
Democratic Mean Median Household Income: 53720.214067278284
Republican Mean Median Household Income: 48741.93585337915
Test Statistic = 5.419, p-value=0.0000000979
Different distributions.....reject null hypothesis
```

In [15]:
```
# 7. Compute the mean population for Democratic counties and Republican
 counties. Which one is higher?

# Democratic Mean Total Population
demPopulation = data.loc[data['Party'].isin(["1.0"]), 'Total Population'
].mean()
print("Democratic Mean of Total Population:",demPopulation)

# Republican Mean Total Population
repPopulation = data.loc[data['Party'].isin(["0.0"]), 'Total Population'
].mean()
print("Republican Mean of Total Population:",repPopulation)

# Perform a hypothesis test to determine whether this difference is stat
istically significant at the α = 0. 05 significance level.
demPop = data.loc[data['Party'].isin(["1.0"]), 'Total Population']
repPop = data.loc[data['Party'].isin(["0.0"]), 'Total Population']

# What is the result of the test? What conclusion do you make from this
 result?
statsPop, pPop = ttest_ind(demPop, repPop, equal_var = False)

print('Test Statistic = %.3f, p-value=%.16f' % (stats, pPop))

if pPop > 0.05:
    print('Same distributions....fail to reject null hypothesis)')

else:
    print('Different distributions.....reject null hypothesis')
```

```
Democratic Mean of Total Population: 299263.9816513761
Republican Mean of Total Population: 54057.9255441008
Test Statistic = 5.419, p-value=0.0000000000000232
Different distributions.....reject null hypothesis
```

In [16]:
```
# 8. Compare Democratic counties and Republican counties in terms of age

# Compute descriptive statistics
demAge = data.loc[data['Party'].isin(["1.0"]), 'Percent Age 29 and Unde
r'].describe()
repAge = data.loc[data['Party'].isin(["0.0"]), 'Percent Age 29 and Unde
r'].describe()
demAge2 = data.loc[data['Party'].isin(["1.0"]), 'Percent Age 65 and Olde
r'].describe()
repAge2 = data.loc[data['Party'].isin(["0.0"]), 'Percent Age 65 and Olde
r'].describe()

# Print descriptive statistics
print("Democratic Summary Statistics Age 29 and Under")
print(demAge)
print("\n")
print("Republican Summary Statistics Age 29 and Under")
print(repAge)
print("\n")
print("Democratic Summary Statistics Age 65 and Older")
print(demAge2)
print("\n")
print("Republican Summary Statistics Age 65 and Older")
print(repAge2)
```

```
Democratic Summary Statistics Age 29 and Under
count    327.000000
mean      38.725202
std        6.235602
min       23.156452
25%       34.507689
50%       38.074151
75%       42.153182
max       67.367823
Name: Percent Age 29 and Under, dtype: float64


Republican Summary Statistics Age 29 and Under
count    873.000000
mean      36.015444
std        5.183572
min       11.842105
25%       32.998088
50%       35.847515
75%       38.543228
max       58.749116
Name: Percent Age 29 and Under, dtype: float64


Democratic Summary Statistics Age 65 and Older
count    327.000000
mean      16.199701
std        4.291217
min        6.653188
25%       13.096022
50%       15.698087
75%       18.806786
max       31.642106
Name: Percent Age 65 and Older, dtype: float64


Republican Summary Statistics Age 65 and Older
count    873.000000
mean      18.819174
std        4.730995
min        6.954387
25%       15.781645
50%       18.377039
75%       21.102195
max       37.622759
Name: Percent Age 65 and Older, dtype: float64
```

In [17]:
```
# 8. Compare Democratic counties and Republican counties in terms of gen
der

# Compute descriptive statistics
demGender = data.loc[data['Party'].isin(["1.0"]), 'Percent Female'].desc
ribe()
repGender = data.loc[data['Party'].isin(["0.0"]), 'Percent Female'].desc
ribe()

# Print descriptive statistics
print("Democratic Summary Statistics by Gender")
print(demGender)
print("\n")
print("Republican Summary Statistics by Gender")
print(repGender)
```

```
Democratic Summary Statistics by Gender
count    327.000000
mean      50.341525
std        2.233010
min       34.245291
25%       49.847663
50%       50.648337
75%       51.488184
max       56.418468
Name: Percent Female, dtype: float64


Republican Summary Statistics by Gender
count    873.000000
mean      49.631843
std        2.425126
min       21.513413
25%       49.220284
50%       50.174893
75%       50.827209
max       55.885023
Name: Percent Female, dtype: float64
```

In [18]:
```
# 8. Compare Democratic counties and Republican counties in terms of rac
e and ethnicity

# Compute descriptive statistics
demPercentWhite = data.loc[data['Party'].isin(["1.0"]), 'Percent White,
 not Hispanic or Latino'].describe()
demPercentBlack = data.loc[data['Party'].isin(["1.0"]), 'Percent Black,
 not Hispanic or Latino'].describe()
demPercentHispanic = data.loc[data['Party'].isin(["1.0"]), 'Percent Hisp
anic or Latino'].describe()
repPercentWhite = data.loc[data['Party'].isin(["0.0"]), 'Percent White,
 not Hispanic or Latino'].describe()
repPercentBlack = data.loc[data['Party'].isin(["0.0"]), 'Percent Black,
 not Hispanic or Latino'].describe()
repPercentHispanic = data.loc[data['Party'].isin(["0.0"]), 'Percent Hisp
anic or Latino'].describe()

# Print descriptive statistics
print("Democratic Summary Statistics Percent White")
print(demPercentWhite)
print("\n")
print("Democratic Summary Statistics Percent Black")
print(demPercentBlack)
print("\n")
print("Democratic Summary Statistics Percent Hispanic")
print(demPercentHispanic)
print("\n")
print("Republican Summary Statistics Percent White")
print(repPercentWhite)
print("\n")
print("Republican Summary Statistics Percent Black")
print(repPercentBlack)
print("\n")
print("Republican Summary Statistics Percent Hispanic")
print(repPercentHispanic)
print("\n")
```

```
Democratic Summary Statistics Percent White
count    327.000000
mean      69.529649
std       24.999523
min        2.776702
25%       52.964474
50%       77.761359
75%       90.257657
max       98.063495
Name: Percent White, not Hispanic or Latino, dtype: float64


Democratic Summary Statistics Percent Black
count    327.000000
mean       9.214369
std       13.317835
min        0.000000
25%        0.839426
50%        3.485992
75%       11.045084
max       63.953279
Name: Percent Black, not Hispanic or Latino, dtype: float64


Democratic Summary Statistics Percent Hispanic
count    327.000000
mean      12.807956
std       19.730351
min        0.193349
25%        2.537143
50%        5.070006
75%       12.203920
max       95.479801
Name: Percent Hispanic or Latino, dtype: float64


Republican Summary Statistics Percent White
count    873.000000
mean      82.684338
std       16.038903
min       18.758977
25%       75.073605
50%       89.451303
75%       94.471136
max       99.627329
Name: Percent White, not Hispanic or Latino, dtype: float64


Republican Summary Statistics Percent Black
count    873.000000
mean       4.181092
std        6.712096
min        0.000000
25%        0.460036
50%        1.318775
75%        4.743677
max       41.563041
```

```
Name: Percent Black, not Hispanic or Latino, dtype: float64


Republican Summary Statistics Percent Hispanic
count    873.000000
mean       9.712826
std       14.030170
min        0.000000
25%        1.704437
50%        3.439315
75%       10.560477
max       78.397012
Name: Percent Hispanic or Latino, dtype: float64
```

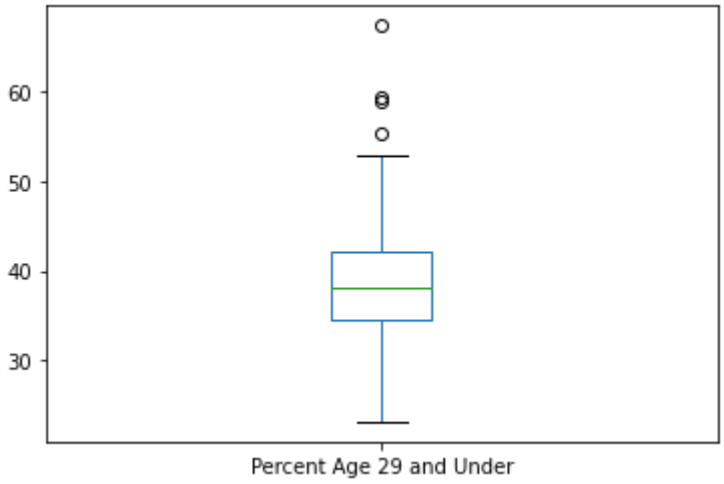
In [19]:
```
# 8. Compare Democratic counties and Republican counties in terms of edu
cation

# Compute descriptive statistics
demHighSchool = data.loc[data['Party'].isin(["1.0"]), 'Percent Less than
High School Degree'].describe()
demBachelors = data.loc[data['Party'].isin(["1.0"]), "Percent Less than
 Bachelor's Degree"].describe()
repHighSchool = data.loc[data['Party'].isin(["0.0"]), 'Percent Less than
High School Degree'].describe()
repBachelors = data.loc[data['Party'].isin(["0.0"]), "Percent Less than
 Bachelor's Degree"].describe()

# Print descriptive statistics
print("Democratic Summary Statistics With High School Degree")
print(demHighSchool)
print("\n")
print("Democratic Summary Statistics With Bachelors Degree")
print(demBachelors)
print("\n")
print("Republican Summary Statistics With High School Degree")
print(repHighSchool)
print("\n")
print("Republican Summary Statistics With Bachelors Degree")
print(repBachelors)
```

```
Democratic Summary Statistics With High School Degree
count    327.000000
mean      11.963116
std        6.570832
min        3.215803
25%        7.906708
50%       10.407118
75%       13.766288
max       49.673777
Name: Percent Less than High School Degree, dtype: float64


Democratic Summary Statistics With Bachelors Degree
count    327.000000
mean      72.066290
std       11.230653
min       26.335440
25%       65.751620
50%       72.759680
75%       80.046043
max       94.849957
Name: Percent Less than Bachelor's Degree, dtype: float64


Republican Summary Statistics With High School Degree
count    873.000000
mean      14.004385
std        6.303227
min        2.134454
25%        9.658025
50%       12.567763
75%       17.449694
max       47.812773
Name: Percent Less than High School Degree, dtype: float64


Republican Summary Statistics With Bachelors Degree
count    873.000000
mean      81.087323
std        6.840917
min       43.419470
25%       78.108081
50%       82.403944
75%       85.547240
max       97.014925
Name: Percent Less than Bachelor's Degree, dtype: float64
```

In [20]:
```
# 8. Democratic Age 29 and Under Plot (AGE)
# Create plots to visualize the results
print("Visualizing Democratic Percent Age 29 and Under")
demAgeVisual = data.loc[data['Party'].isin(["1.0"]), 'Percent Age 29 and Under']
demAgeVisual.plot(kind="box")
```

```
Visualizing Democratic Percent Age 29 and Under
```

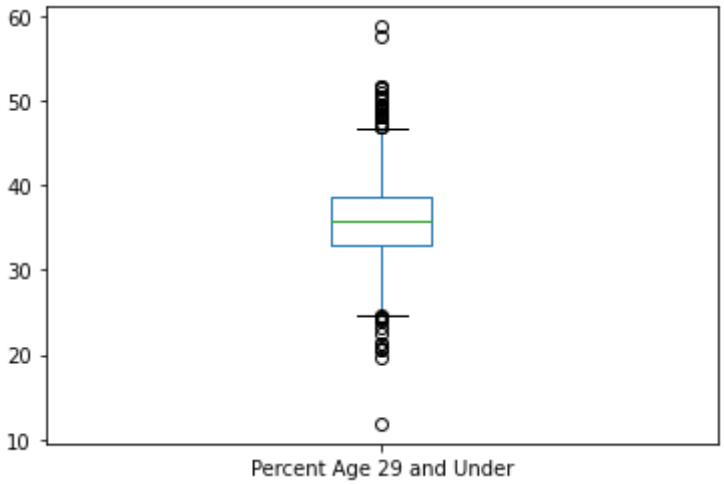
Out[20]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x11f99c9d0>
```



In [21]:
```
# 8. Republican Age 29 and Under Plot (AGE)
# Create plots to visualize the results
print("Visualizing Republican Percent Age 29 and Under")
repAgeVisual = data.loc[data['Party'].isin(["0.0"]), 'Percent Age 29 and Under']
repAgeVisual.plot(kind="box")
```

```
Visualizing Republican Percent Age 29 and Under
```

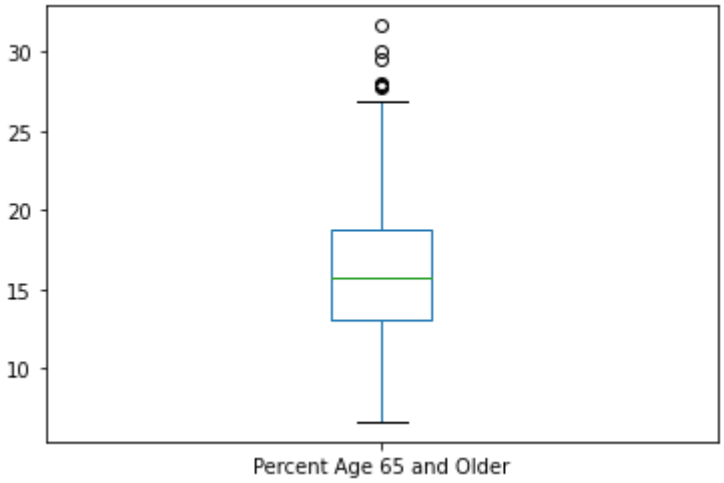
Out[21]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x109ca2970>
```



In [22]:
```
# 8. Democratic Age 65 and Older (AGE)
# Create plots to visualize the results
print("Visualizing Democratic Percent Age 65 and Older")
demAgeVisual2 = data.loc[data['Party'].isin(["1.0"]), 'Percent Age 65 and Older']
demAgeVisual2.plot(kind="box")
```

```
Visualizing Democratic Percent Age 65 and Older
```

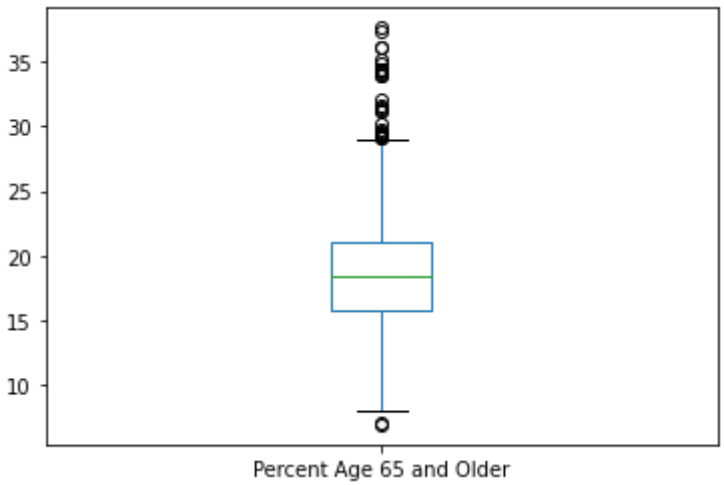
Out[22]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x109d94430>
```



In [23]:
```
# 8. Republican Age 65 and Older (AGE)
# Create plots to visualize the results
print("Visualizing Republican Percent Age 65 and Older")
repAgeVisual2 = data.loc[data['Party'].isin(["0.0"]), 'Percent Age 65 and Older']
repAgeVisual2.plot(kind='box')
```

```
Visualizing Republican Percent Age 65 and Older
```

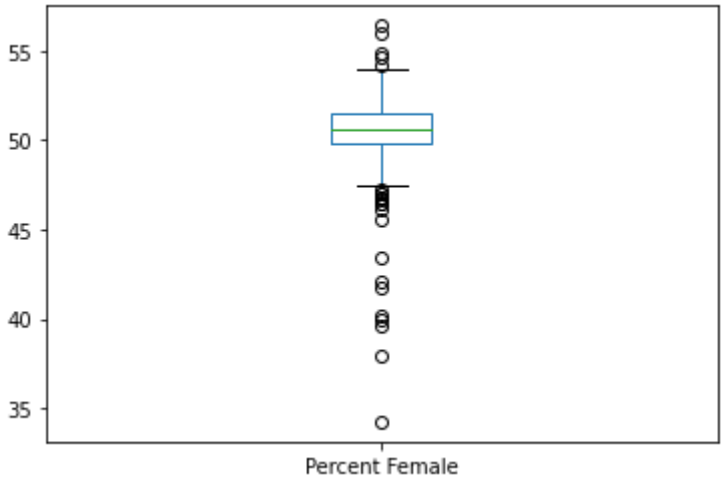
Out[23]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x11e631d60>
```



In [24]:
```
# 8. Democratic Female (GENDER)
# Create plots to visualize the results
print("Visualizing Demoractic Gender")
demGenderVisual = data.loc[data['Party'].isin(["1.0"]), 'Percent Female'
]
demGenderVisual.plot(kind='box')
```

```
Visualizing Demoractic Gender
```

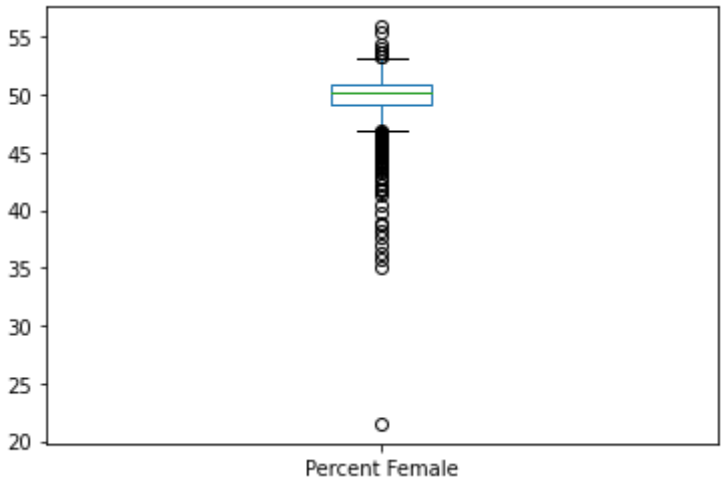
Out[24]: <matplotlib.axes._subplots.AxesSubplot at 0x11fb6a460>



In [25]:
```
# 8. Republican Female (GENDER)
# Create plots to visualize the results
print("Visualizing Republican Gender")
repGenderVisual = data.loc[data['Party'].isin(["0.0"]), 'Percent Female'
]
repGenderVisual.plot(kind='box')
```

```
Visualizing Republican Gender
```

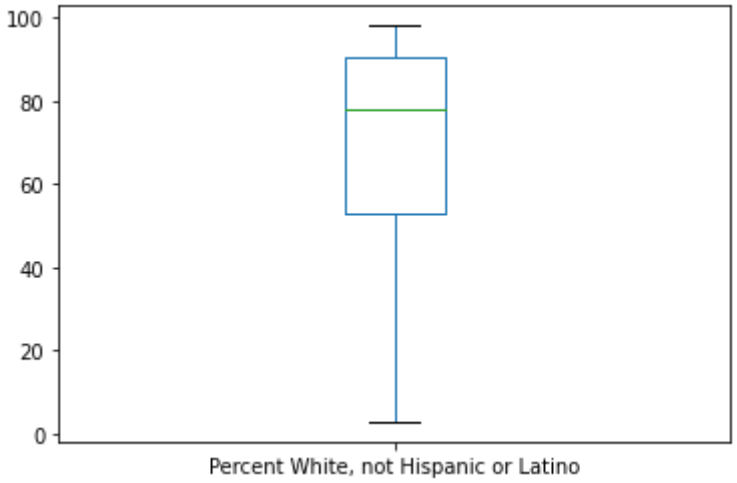
Out[25]: <matplotlib.axes._subplots.AxesSubplot at 0x11fc2a910>



In [26]:
```
# 8. Democratic White (RACE)
# Create plots to visualize the results
print("Visualizing Democratic Percent White")
demPercentWhiteVisual = data.loc[data['Party'].isin(["1.0"]), 'Percent White, not Hispanic or Latino']
demPercentWhiteVisual.plot(kind='box')
```

```
Visualizing Democratic Percent White
```

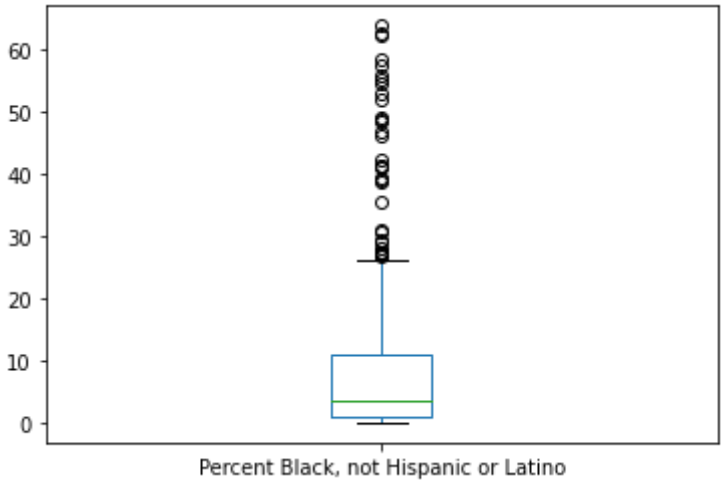
Out[26]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x11fce8df0>
```



In [27]:
```
# 8. Democratic Black (RACE)
# Create plots to visualize the results
print("Visualizing Democratic Percent Black")
demPercentBlackVisual = data.loc[data['Party'].isin(["1.0"]), 'Percent Black, not Hispanic or Latino']
demPercentBlackVisual.plot(kind='box')
```

```
Visualizing Democratic Percent Black
```

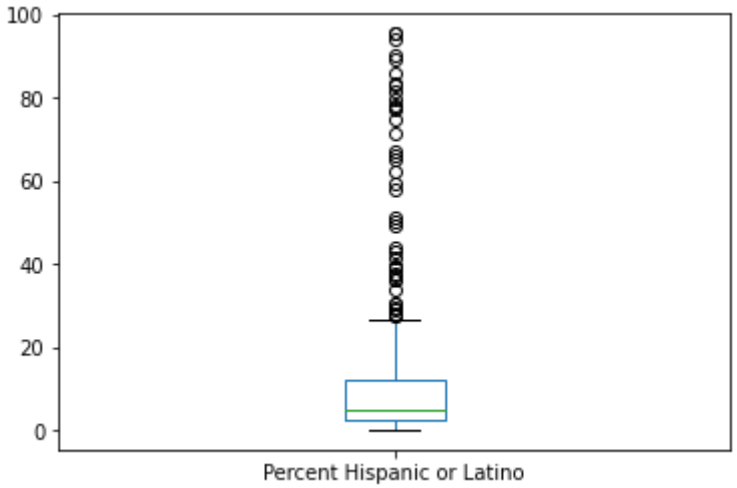
Out[27]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x11fdb7c70>
```



In [28]:
```
# 8. Democratic Hispanic or Latino (RACE)
# Create plots to visualize the results
print("Visualizing Democratic Percent Hispanic")
demPercentHispanicVisual = data.loc[data['Party'].isin(["1.0"]), 'Percen
t Hispanic or Latino']
demPercentHispanicVisual.plot(kind='box')
```

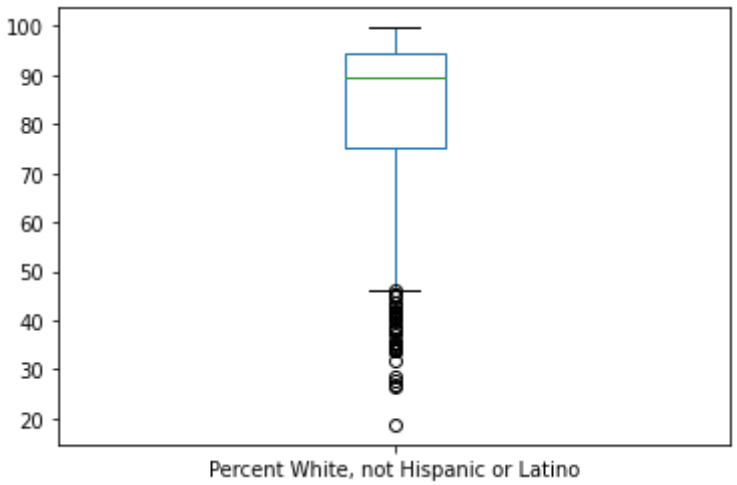
```
Visualizing Democratic Percent Hispanic
```

Out[28]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x11fe7b520>
```



In [29]:
```
# 8. Republican White (RACE)
# Create plots to visualize the results
print("Visualizing Republican Percent White")
repPercentWhiteVisual = data.loc[data['Party'].isin(["0.0"]), 'Percent W
hite, not Hispanic or Latino']
repPercentWhiteVisual.plot(kind='box')
repPercentHispanic = data.loc[data['Party'].isin(["0.0"]), 'Percent Hisp
anic or Latino']
```

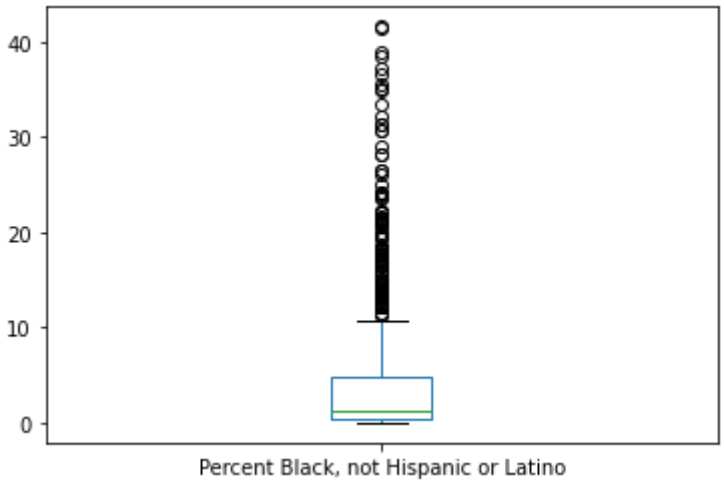
```
Visualizing Republican Percent White
```



In [30]:
```
# 8. Republican Black (RACE)
# Create plots to visualize the results
print("Visualizing Republican Percent Black")
repPercentBlackVisual = data.loc[data['Party'].isin(["0.0"]), 'Percent Black, not Hispanic or Latino']
repPercentBlackVisual.plot(kind='box')
```

```
Visualizing Republican Percent Black
```

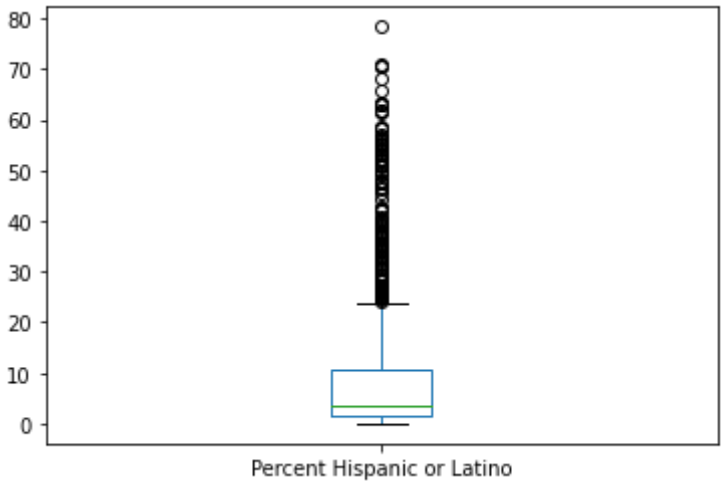
Out[30]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x11fa99af0>
```



In [31]:
```
# 8. Republican Hispanic or Latino (RACE)
# Create plots to visualize the results
print("Visualizing Republican Percent Hispanic or Latino")
repPercentHispanicVisual = data.loc[data['Party'].isin(["0.0"]), 'Percent Hispanic or Latino']
repPercentHispanicVisual.plot(kind='box')
```

```
Visualizing Republican Percent Hispanic or Latino
```

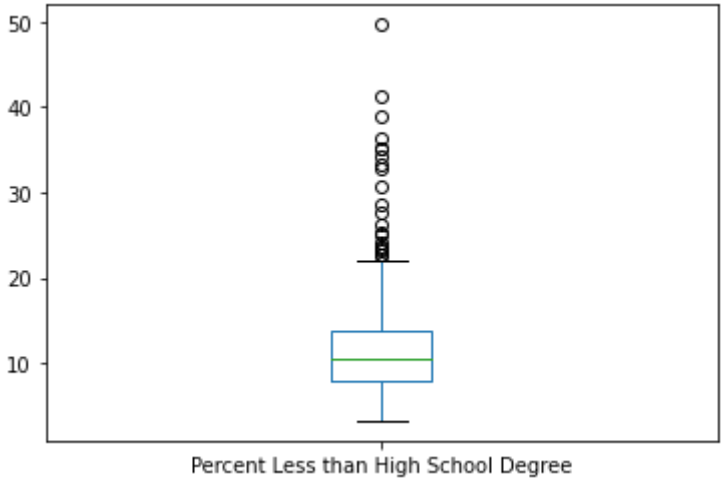
Out[31]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1200ae3a0>
```



In [32]:
```
# 8. Democratic Less than High School Degree (EDUCATION)
# Create plots to visualize the results
print("Visualizing Democratic Percent Less than High School Degree")
demHighSchoolVisual = data.loc[data['Party'].isin(["1.0"]), 'Percent Les
s than High School Degree']
demHighSchoolVisual.plot(kind='box')
```

```
Visualizing Democratic Percent Less than High School Degree
```

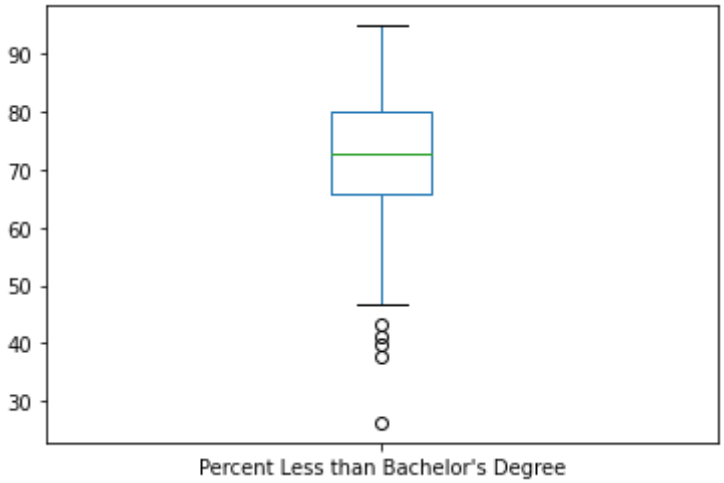
Out[32]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x120179f10>
```



In [33]:
```
# 8. Democratic Less than Bachelor's Degree (EDUCATION)
# Create plots to visualize the results
print("Visualizing Democratic Percent Less than Bachelor's Degree")
demBachelorsVisual = data.loc[data['Party'].isin(["1.0"]), "Percent Less
than Bachelor's Degree"]
demBachelorsVisual.plot(kind='box')
```

```
Visualizing Democratic Percent Less than Bachelor's Degree
```

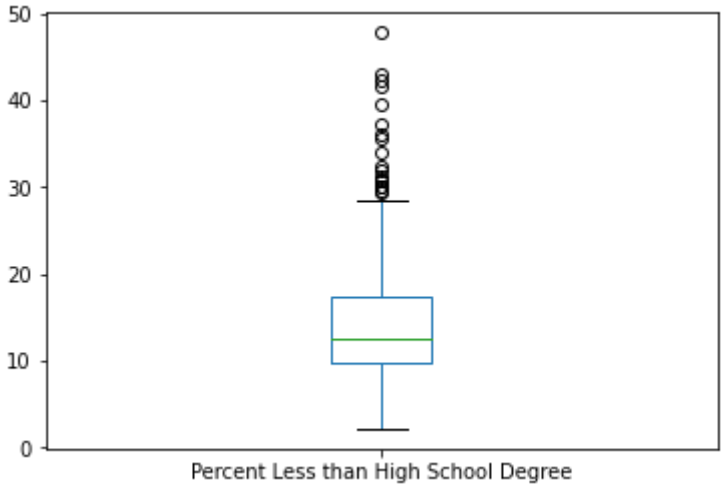
Out[33]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x12023c700>
```



In [34]:
```
# 8. Republican Less than High School Degree (EDUCATION)
# Create plots to visualize the results
print("Visualizing Republican Percent Less than High School Degree")
repHighSchoolVisual = data.loc[data['Party'].isin(["0.0"]), 'Percent Les
s than High School Degree']
repHighSchoolVisual.plot(kind='box')
```

```
Visualizing Republican Percent Less than High School Degree
```

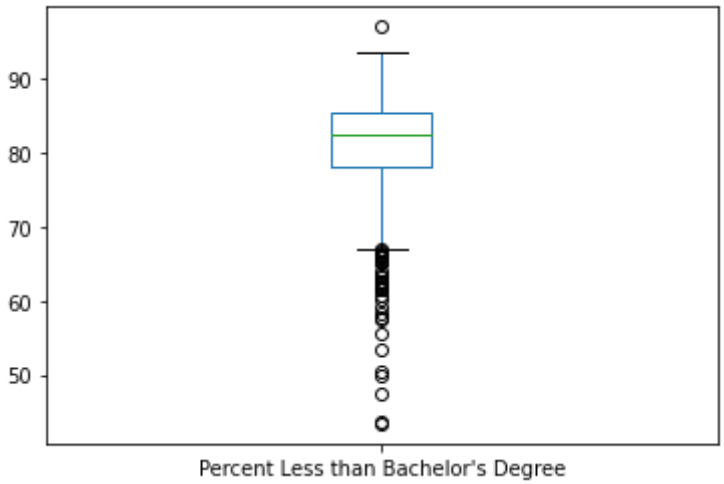
Out[34]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x120312ac0>
```



In [35]:
```
# 8. Republican Less than Bachelor's Degree (EDUCATION)
# Create plots to visualize the results
print("Visualizing Republican Percent Less than Bachelor's Degree")
repBachelorsVisual = data.loc[data['Party'].isin(["0.0"]), "Percent Less
than Bachelor's Degree"]
repBachelorsVisual.plot(kind='box')
```

```
Visualizing Republican Percent Less than Bachelor's Degree
```

Out[35]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1203d4bb0>
```



In [36]:
```
# 9. Extract predictor features from data (referenced from #8)
datax = data[['Total Population', 'Percent White, not Hispanic or Latin
o', 'Percent Black, not Hispanic or Latino', 'Percent Hispanic or Latin
o', 'Percent Female', 'Percent Age 29 and Under', 'Percent Age 65 and Ol
der', 'Median Household Income', 'Percent Less than High School Degree',
"Percent Less than Bachelor's Degree"]]
datax.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1200 entries, 0 to 1199
Data columns (total 10 columns):
 #   Column                                Non-Null Count  Dtype
---  ------                                --------------  -----
 0   Total Population                      1200 non-null   int64
 1   Percent White, not Hispanic or Latino 1200 non-null   float64
 2   Percent Black, not Hispanic or Latino 1200 non-null   float64
 3   Percent Hispanic or Latino            1200 non-null   float64
 4   Percent Female                        1200 non-null   float64
 5   Percent Age 29 and Under              1200 non-null   float64
 6   Percent Age 65 and Older              1200 non-null   float64
 7   Median Household Income               1200 non-null   int64
 8   Percent Less than High School Degree  1200 non-null   float64
 9   Percent Less than Bachelor's Degree   1200 non-null   float64
dtypes: float64(8), int64(2)
memory usage: 103.1 KB
```

In [37]:
```
# 9. Partition the dataset into training set and test set using the trai
n_test_split method
x_train, x_test, y_train, y_test = train_test_split(datax, data['Party'
], test_size=0.25, train_size=0.75, random_state=0)
```

In [38]:
```
# 9. Standardize the training set and test set
scaler = StandardScaler()
scaler.fit(x_train)
x_train_scaled = scaler.transform(x_train)
x_test_scaled = scaler.transform(x_test)
```

In [39]:
```
# 9. Build a simple linear model to predict 'Party' with X as the predic
tor and print the coefficient of the model
model = linear_model.LinearRegression()
fitted_model = model.fit(X = x_train_scaled[:, 9].reshape(-1, 1), y = y_
train)
print(fitted_model.coef_)
```

```
[-0.19727913]
```

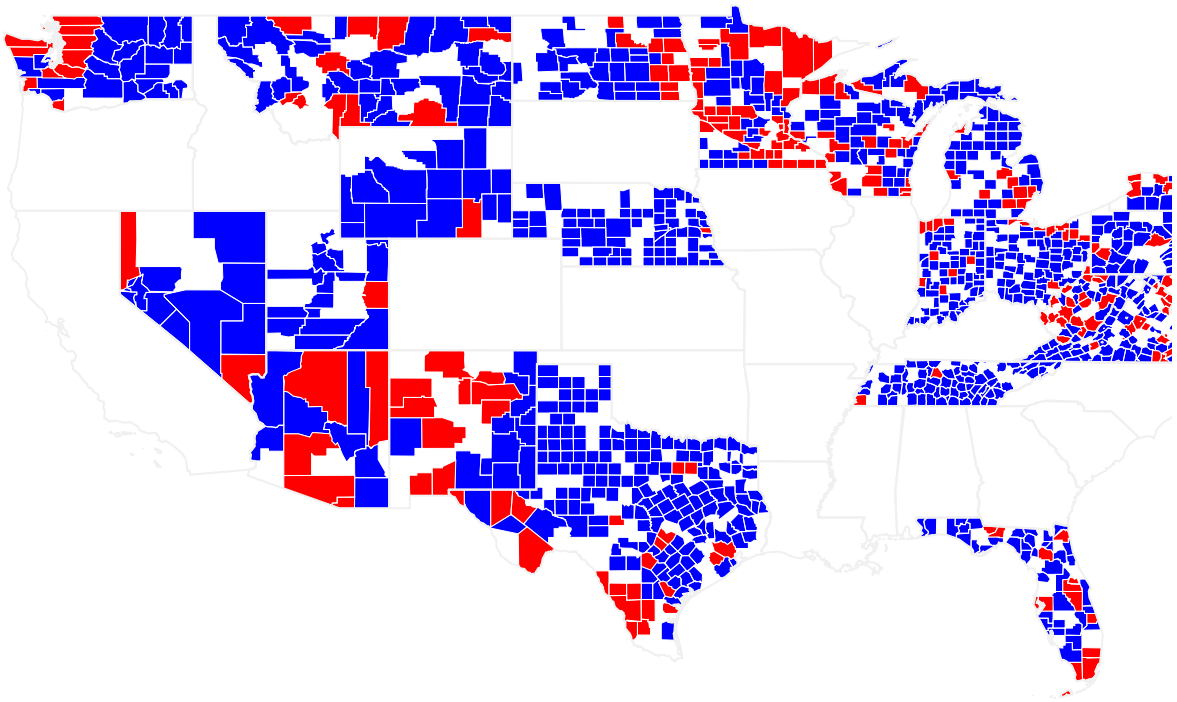
In [40]:
```
# 9. Use the model to predict 'Party' for the test set
predicted = fitted_model.predict(x_test_scaled[:, 9].reshape(-1, 1))
```

In [41]:
```
# 9. Compute the coefficient of determination (R squared) of the model o
ver the test set
import numpy
corr_coef = numpy.corrcoef(predicted, y_test.values)[1, 0]
R_squared = corr_coef ** 2
print(R_squared)
```

```
0.1724302743984145
```

In [42]:
```
# 10. Create a map of Democratic counties and Republican counties using
 the counties' FIPS codes and Python's Plotly library
import plotly.figure_factory as ff
fips = data['FIPS'].tolist()
values = data['Party'].tolist()
colorscale = [
    'rgb(0, 0, 255)',
    'rgb(255, 0, 0)',
]
fig = ff.create_choropleth(fips=fips, values=values, colorscale=colorsca
le,
      county_outline={'color': 'rgb(255,255,255)', 'width': 0.5}, legend
_title='Party by County',
      title='Democratic counties v Republican counties')
fig.layout.template = None
```

In [43]:
```
# 10. Q.E.D
fig.show()
```



In [ ]:
```

```