In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import ttest_ind
import numpy as np
from urllib.request import urlopen
import json
import plotly.express as px
```

In [2]:
```
data = pd.read_csv('data/election_train.csv')
demographics_data = pd.read_csv('data/demographics_train.csv')
state_map = pd.read_csv('data/state_map.csv')
```

In [3]:
```
data.head(20)
```

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
5	2018	AZ	Coconino County	US Senator	Republican	19249
6	2018	AZ	Gila County	US Senator	Democratic	7643
7	2018	AZ	Gila County	US Senator	Republican	12180
8	2018	AZ	Graham County	US Senator	Democratic	3368
9	2018	AZ	Graham County	US Senator	Republican	6870
10	2018	AZ	La Paz County	US Senator	Democratic	1609
11	2018	AZ	La Paz County	US Senator	Republican	3265
12	2018	AZ	Maricopa County	US Senator	Democratic	732671
13	2018	AZ	Maricopa County	US Senator	Republican	672505

	Year	State	County	Office	Party	Votes
14	2018	AZ	Mohave County	US Senator	Democratic	19214
15	2018	AZ	Mohave County	US Senator	Republican	50209
16	2018	AZ	Navajo County	US Senator	Democratic	16624
17	2018	AZ	Navajo County	US Senator	Republican	18767
18	2018	AZ	Pima County	US Senator	Democratic	221242
19	2018	AZ	Pima County	US Senator	Republican	160550

In [4]:

```
demographics_data.head(20)
```

Out[4]:

	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	Percent Age 29 and Under	Percent Age 65 and Older	Me House Inc
0	Wisconsin	La Crosse	55063	117538	0	90.537528	1.214075	1.724549	2.976059	51.171536	43.241335	14.702479	5
1	Virginia	Alleghany	51005	15919	12705	91.940449	5.207614	1.432251	1.300333	51.077329	31.660280	23.902255	4
2	Indiana	Fountain	18045	16741	12750	95.705155	0.400215	2.359477	1.547100	49.770026	35.899887	18.941521	4
3	Ohio	Geauga	39055	94020	0	95.837056	1.256116	1.294405	2.578175	50.678579	36.281642	18.028079	7
4	Wisconsin	Jackson	55053	20566	15835	86.662453	1.983857	3.082758	1.376058	46.649810	36.292911	17.587280	4
5	Texas	Baylor	48023	3639	0	86.644683	1.841165	8.353943	2.473207	51.662545	30.090684	24.402308	3
6	Nebraska	Madison	31119	35125	24885	81.249822	1.155872	14.217794	6.784342	50.448399	41.432028	15.404982	4
7	Hawaii	Hawaii	15001	193680	0	30.401694	0.547811	12.405514	11.003717	50.143019	36.008881	17.580545	5
8	Tennessee	Henry	47079	32291	25285	87.662197	8.599919	2.201852	1.560806	51.441578	33.238364	21.476572	3
9	Michigan	Oceana	26127	26152	18930	82.486999	1.131845	14.419547	5.578923	49.395840	36.643469	19.088406	4
10	Nebraska	Pierce	31139	7179	5385	96.893718	0.222872	1.587965	0.780053	49.658727	36.634629	18.540187	5
11	Texas	Jack	48237	8866	6535	78.411911	4.376269	15.880893	5.549289	43.187458	38.732236	15.677871	5
12	Florida	Walton	12131	61528	47490	84.447731	4.950592	5.888376	5.759979	49.349889	33.165388	18.783318	4

	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	Percent Age 29 and Under	Percent Age 65 and Older	Me House Inc
13	Virginia	Washington	51191	54562	0	95.484037	1.268282	1.414904	1.611011	50.500348	32.005792	20.301675	4.
14	Florida	Escambia	12033	309574	0	65.219624	21.532816	5.389988	4.747492	50.299444	41.364585	15.774581	4
15	Texas	Wheeler	48483	5642	3785	68.397731	2.658632	26.462247	9.517901	49.202410	39.188231	17.795108	5
16	Arizona	Yavapai	4025	218586	0	81.159361	0.518331	14.054880	6.456955	51.092476	28.717301	28.272625	4
17	Nebraska	Loup	31115	542	435	97.970480	0.000000	0.000000	0.000000	52.398524	30.996310	24.538745	5
18	Michigan	Antrim	26009	23215	0	95.179841	0.323067	1.955632	2.015938	50.273530	29.450786	25.410295	4
19	Minnesota	Wabasha	27157	21327	16385	94.926619	0.150045	2.874291	1.355090	50.171145	34.594645	18.807146	5

In [5]:

```
state_map.head()
```

Out[5]:

	State	Code
0	Alabama	AL
1	Alaska	AK
2	Arizona	AZ
3	Arkansas	AR
4	California	CA

Task 1: Reshape dataset election_train from long format to wide format.

In [6]:

```
election_data = pd.pivot_table(data, index=['State', 'Year', 'Office','County'], columns='Party', values='Votes').reset_i
```

In [7]:

```
election_data
```

Out[7]:

Party	State	Year	Office	County	Democratic	Republican

Party	State	Year	Office	County	Democratic	Republican
0	AZ	2018	US Senator	Apache County	16298.0	7810.0
1	AZ	2018	US Senator	Cochise County	17383.0	26929.0
2	AZ	2018	US Senator	Coconino County	34240.0	19249.0
3	AZ	2018	US Senator	Gila County	7643.0	12180.0
4	AZ	2018	US Senator	Graham County	3368.0	6870.0
...	...	...	...	...	...	...
1200	WY	2018	US Senator	Platte County	801.0	2850.0
1201	WY	2018	US Senator	Sublette County	668.0	2653.0
1202	WY	2018	US Senator	Sweetwater County	3943.0	8577.0
1203	WY	2018	US Senator	Uinta County	1371.0	4713.0
1204	WY	2018	US Senator	Washakie County	588.0	2423.0

1205 rows × 6 columns

Task 2: Merge reshaped dataset election_train with dataset demographics_train. Address all inconsistencies in the names of the states and the counties before merging

In [8]:

```
dict = {}
for element in state_map.values:
    dict[element[1]] = element[0]

# Replacing State Name with Code
election_data.replace({"State": dict},inplace=True)

#Removing County word and changing string to lower case
election_data['County'] = election_data.County.str.replace(' County','').str.lower()
demographics_data['County'] = demographics_data['County'].str.lower()

#Merging datasets through inner join on State and County
merge_dataframe = pd.merge(election_data, demographics_data, how="inner", on=['County','State'])
merge_dataframe
```

Out[8]:

	State	Year	Office	County	Democratic	Republican	FIPS	Total Population	Citizen Voting-Age Population	Percent White, not Hispanic or Latino	...	Percent Hispanic or Latino	Percent Foreign Born	Perc Fem
0	Arizona	2018	US Senator	apache	16298.0	7810.0	4001	72346	0	18.571863	...	5.947806	1.719515	50.598
1	Arizona	2018	US Senator	cochise	17383.0	26929.0	4003	128177	92915	56.299492	...	34.403208	11.458374	49.069
2	Arizona	2018	US Senator	coconino	34240.0	19249.0	4005	138064	104265	54.619597	...	13.711033	4.825298	50.581
3	Arizona	2018	US Senator	gila	7643.0	12180.0	4007	53179	0	63.222325	...	18.548675	4.249798	50.296
4	Arizona	2018	US Senator	graham	3368.0	6870.0	4009	37529	0	51.461536	...	32.097844	4.385942	46.313
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1195	Wyoming	2018	US Senator	platte	801.0	2850.0	56031	8740	6830	89.359268	...	7.814645	2.780320	47.711
1196	Wyoming	2018	US Senator	sublette	668.0	2653.0	56035	10032	0	91.646730	...	7.814992	2.053429	46.949
1197	Wyoming	2018	US Senator	sweetwater	3943.0	8577.0	56037	44812	30565	79.815674	...	15.859591	5.509685	47.824
1198	Wyoming	2018	US Senator	uinta	1371.0	4713.0	56041	20893	14355	87.718375	...	8.959939	3.986981	49.327
1199	Wyoming	2018	US Senator	washakie	588.0	2423.0	56043	8351	0	82.397318	...	13.962400	3.783978	51.359

1200 rows × 21 columns

Task 3: Explore the merge dataset.

In [9]:
```
merge_dataframe.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1200 entries, 0 to 1199
Data columns (total 21 columns):
 #   Column                               Non-Null Count  Dtype  
---  ------                               --------------  -----  
 0   State                                1200 non-null   object 
 1   Year                                 1200 non-null   int64  
 2   Office                               1200 non-null   object 
 3   County                               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]:

```
merge_dataframe[merge_dataframe['Citizen Voting-Age Population'] == 0]
```

Out[10]:

	State	Year	Office	County	Democratic	Republican	FIPS	Total Population	Citizen Voting-Age Population	Percent White, not Hispanic or Latino	...	Percent Hispanic or Latino	Percent Foreign Born	Perce Fema
0	Arizona	2018	US Senator	apache	16298.0	7810.0	4001	72346	0	18.571863	...	5.947806	1.719515	50.5985
3	Arizona	2018	US Senator	gila	7643.0	12180.0	4007	53179	0	63.222325	...	18.548675	4.249798	50.2961
4	Arizona	2018	US Senator	graham	3368.0	6870.0	4009	37529	0	51.461536	...	32.097844	4.385942	46.3135

	State	Year	Office	County	Democratic	Republican	FIPS	Total Population	Citizen Voting-Age Population	Percent White, not Hispanic or Latino	...	Percent Hispanic or Latino	Percent Foreign Born	Percer Fema
7	Arizona	2018	US Senator	mohave	19214.0	50209.0	4015	203629	0	78.252606	...	15.708470	6.969047	49.6766
9	Arizona	2018	US Senator	pima	221242.0	160550.0	4019	1003338	0	53.271579	...	36.105978	12.903428	50.8074
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1188	Wyoming	2018	US Senator	converse	834.0	3959.0	56009	14223	0	88.849047	...	7.691767	2.706883	49.9332
1190	Wyoming	2018	US Senator	goshen	1020.0	3658.0	56015	13546	0	86.409272	...	10.519711	2.724051	47.0913
1192	Wyoming	2018	US Senator	lincoln	1152.0	5846.0	56023	18543	0	92.600982	...	4.416761	2.151755	48.7731
1196	Wyoming	2018	US Senator	sublette	668.0	2653.0	56035	10032	0	91.646730	...	7.814992	2.053429	46.9497
1199	Wyoming	2018	US Senator	washakie	588.0	2423.0	56043	8351	0	82.397318	...	13.962400	3.783978	51.3591

680 rows × 21 columns

How mant variables does the dataset have?

The dataset has 21 variables.

What is the type of these variables?

The type of these variables are object, int64 and float64.

Are there any irrelevant or redundant variables?

Yes, there are irrelevant or redundant variables in the dataset.
Year has a value of only 2018, and no other value. Hence, this is an irrelevant/redundant variable.

Office has a value of only US Senator, hence Office is an irrelevant/redundant variable as well.
Citizen Voting-Age Population has more than 50% of the rows with a value of 0. With such a lack of data, this variable becomes irrelevant for data analysis as well.

How will you deal with these variables?

We should delete the Year, Office and Citizen Voting-Age Population column and insert the year 2018 and US Senator in the table header.

In [11]:
```
merge_dataframe.drop(columns=['Citizen Voting-Age Population', 'Year', 'Office'],inplace=True)
```

Task 4: Search the merged dataset for missing values.

In [12]:
```
merge_dataframe.isnull().sum()
```

Out[12]:
```
State                                0
County                               0
Democratic                           3
Republican                           2
FIPS                                 0
Total Population                     0
Percent White, not Hispanic or Latino    0
Percent Black, not Hispanic or Latino    0
Percent Hispanic or Latino           0
Percent Foreign Born                 0
Percent Female                       0
Percent Age 29 and Under             0
Percent Age 65 and Older             0
Median Household Income              0
Percent Unemployed                   0
Percent Less than High School Degree    0
Percent Less than Bachelor's Degree     0
Percent Rural                        0
dtype: int64
```

Are there any missing values?

There are missing values in Democratic and Republican columns. Also, Citizen Voting-Age Population had values mentioned as 0.

How will you deal with these values?

We have already removed the Citizen Voting-Age Population since it has over 50% of data with the value 0.

We will remove the 5 entries of Democratic and Republican since a small observation won't impact the data analysis.

In [13]:

```
# Dropping the null observations and storing the rest back in the merge_dataframe
merge_dataframe = merge_dataframe.dropna()
```

Task 5: Create a new variable named 'Party' that labels each county as Democratic or Republican.

Value should be 1 if there were more votes cast for the Democratic party and 0 if more votes were cast for the Republican Party

In [14]:

```
def compare_values(row):
    democratic = row[0]
    republican = row[1]

    # One of the rules
    if democratic > republican:
        return 1
    else:
        return 0

    return None

merge_dataframe["Party"] = merge_dataframe[["Democratic", "Republican"]].apply(compare_values, axis=1)
```

```
<ipython-input-14-43fa2310f346>:13: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-v
iew-versus-a-copy
  merge_dataframe["Party"] = merge_dataframe[["Democratic", "Republican"]].apply(compare_values, axis=1)
```

In [15]:

```
merge_dataframe
```

Out[15]:

	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	Percent Female	Percent Age 29 and Under	Pe A(
0	Arizona	apache	16298.0	7810.0	4001	72346	18.571863	0.486551	5.947806	1.719515	50.598513	45.854643	13.32
1	Arizona	cochise	17383.0	26929.0	4003	128177	56.299492	3.714395	34.403208	11.458374	49.069646	37.902276	19.75
2	Arizona	coconino	34240.0	19249.0	4005	138064	54.619597	1.342855	13.711033	4.825298	50.581614	48.946141	10.87
3	Arizona	gila	7643.0	12180.0	4007	53179	63.222325	0.552850	18.548675	4.249798	50.296170	32.238290	26.39
4	Arizona	graham	3368.0	6870.0	4009	37529	51.461536	1.811932	32.097844	4.385942	46.313518	46.393456	12.31
...	...	...	...	...	...	...	...	...	...	...	...	...	
1195	Wyoming	platte	801.0	2850.0	56031	8740	89.359268	0.057208	7.814645	2.780320	47.711670	32.700229	22.01
1196	Wyoming	sublette	668.0	2653.0	56035	10032	91.646730	0.000000	7.814992	2.053429	46.949761	36.393541	13.33
1197	Wyoming	sweetwater	3943.0	8577.0	56037	44812	79.815674	0.865840	15.859591	5.509685	47.824244	44.153352	9.41
1198	Wyoming	uinta	1371.0	4713.0	56041	20893	87.718375	0.186665	8.959939	3.986981	49.327526	43.205858	10.67
1199	Wyoming	washakie	588.0	2423.0	56043	8351	82.397318	0.790325	13.962400	3.783978	51.359119	34.774279	19.65

1195 rows × 19 columns

Task 6.1: Compute the Median Household Income for Democratic and Republican counties.

In [16]:

```
#mean of median household income from Democratic and Republican County
democratic_household = merge_dataframe[merge_dataframe['Party'] == 1]
democratic_household['Median Household Income'].mean()
```

Out[16]: 53798.732307692306

In [17]:

```
republican_household = merge_dataframe[merge_dataframe['Party'] == 0]
republican_household['Median Household Income'].mean()
```

48746.81954022989

Out[17]:

Compute the mean median household income for Democratic counties and Republican counties. Which one is higher?

Democratic Household Income is higher

Task 6.2: Perform a hypothesis test to determine whether this difference is statistically significant at the $\alpha = 0.05$ significance level.

In [18]:
```
ttest,pval = ttest_ind(democratic_household['Median Household Income'], republican_household['Median Household Income'],
pval=pval/2
print("p-value",pval)
```

```
p-value 3.5747186815913e-08
```

What conclusion do you make from this result?

Since p-value is less than the significance value we have sufficient evidence to reject the null hypothesis.

Task 7.1: Compute the mean population for Democratic and Republican Counties.

In [19]:
```
#mean of population from Democratic and Republican County
democratic_population = merge_dataframe[merge_dataframe['Party'] == 1]
democratic_population['Total Population'].mean()
```

Out[19]: 300998.3169230769

In [20]:
```
#mean of population from Democratic and Republican County
republic_population = merge_dataframe[merge_dataframe['Party'] == 0]
republic_population['Total Population'].mean()
```

Out[20]: 53864.6724137931

Compute the mean population for Democratic counties and Republican counties. Which one is higher?

The population mean is higher for Republican Counties.

Task 7.2: Perform a hypothesis test to determine whether this difference is statistically

significant at the $\alpha = 0.05$ significance level.

In [21]:

```
ttest,pvall = ttest_ind(democratic_population['Total Population'], republic_population['Total Population'], equal_var=Fal
pvall=pvall/2
print("p-value",pvall)
```

```
p-value 1.0239358801486512e-14
```

What conclusion do you make from this result?

Since p-value less than the significance value we have sufficient evidence to reject the null hypothesis.

Task 8: Compare Democratic and Republican counties in terms of age, gender, race and ethnicity, and education by computing descriptive statistics and creating plots to visualize the results. Share conclusions for each variable from the descriptive statistics and the plots.

In [22]:

```
merge_dataframe['Percent age 30 to 64'] = 100 - (merge_dataframe['Percent Age 29 and Under'] + merge_dataframe['Percent A
merge_dataframe.groupby(by=['Party'])['Percent Age 29 and Under','Percent age 30 to 64','Percent Age 65 and Older'].descr
```

```
<ipython-input-22-46de5057d7b1>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-v
iew-versus-a-copy
  merge_dataframe['Percent age 30 to 64'] = 100 - (merge_dataframe['Percent Age 29 and Under'] + merge_dataframe['Percent
Age 65 and Older'])
<ipython-input-22-46de5057d7b1>:2: FutureWarning: Indexing with multiple keys (implicitly converted to a tuple of keys) w
ill be deprecated, use a list instead.
  merge_dataframe.groupby(by=['Party'])['Percent Age 29 and Under','Percent age 30 to 64','Percent Age 65 and Older'].des
cribe().T
```

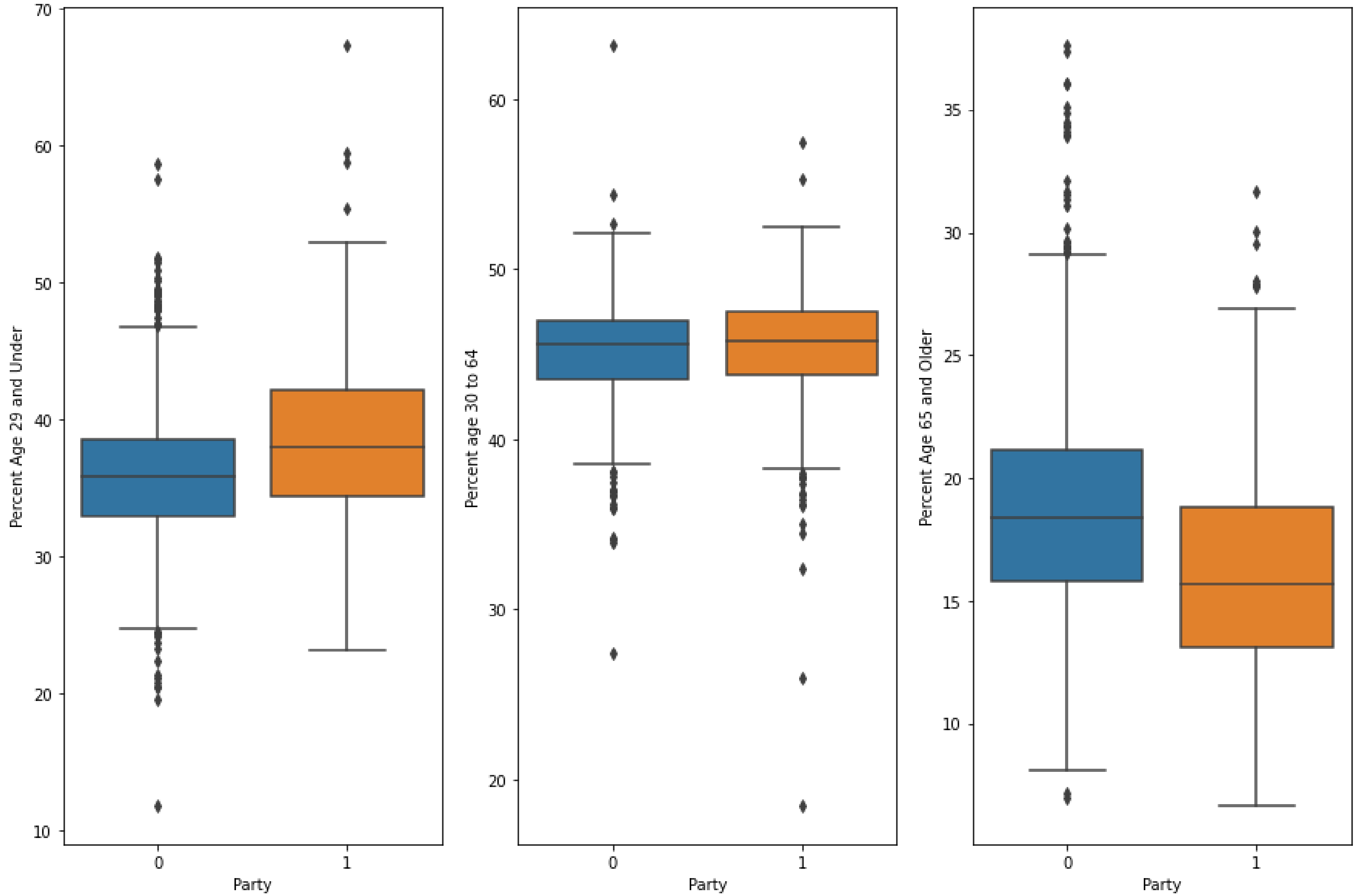
Out[22]:

	Party	0	1
Percent Age 29 and Under	**count**	870.000000	325.000000
	mean	36.005719	38.726959
	std	5.181522	6.252786
	min	11.842105	23.156452
	25%	32.983652	34.488444

	Party	0	1
	50%	35.846532	38.074151
	75%	38.539787	42.161162
	max	58.749116	67.367823
Percent age 30 to 64	**count**	870.000000	325.000000
	mean	45.166015	45.078214
	std	2.910264	3.907598
	min	27.421759	18.433769
	25%	43.522522	43.741937
	50%	45.553295	45.817819
	75%	46.975771	47.448269
	max	63.157895	57.478906
Percent Age 65 and Older	**count**	870.000000	325.000000
	mean	18.828267	16.194826
	std	4.733155	4.282422
	min	6.954387	6.653188
	25%	15.784982	13.106233
	50%	18.377896	15.698087
	75%	21.112847	18.806426
	max	37.622759	31.642106

In [23]:

```
age_groups = ['Percent Age 29 and Under','Percent age 30 to 64','Percent Age 65 and Older']
fig, axes = plt.subplots(1, 3, figsize=(15, 10))
for index,group in enumerate(age_groups):
    sns.boxplot(ax=axes[index], x="Party", y=group, data=merge_dataframe)
```



Conclusion

Although there is not much of a difference. By seeing the descriptive statistics & plots we conclude that counties with 'Percent Age 29 and Under' are democratic and counties with 'Percent Age 65 and Older' are Republicans counties. We also see that counties under 'Percent age 30 to 64' are almost equally divided having the same mean.

In [24]:
```
merge_dataframe['Percent Male'] = 100 - merge_dataframe['Percent Female']
merge_dataframe.groupby(by=['Party'])['Percent Female','Percent Male'].describe().T
```

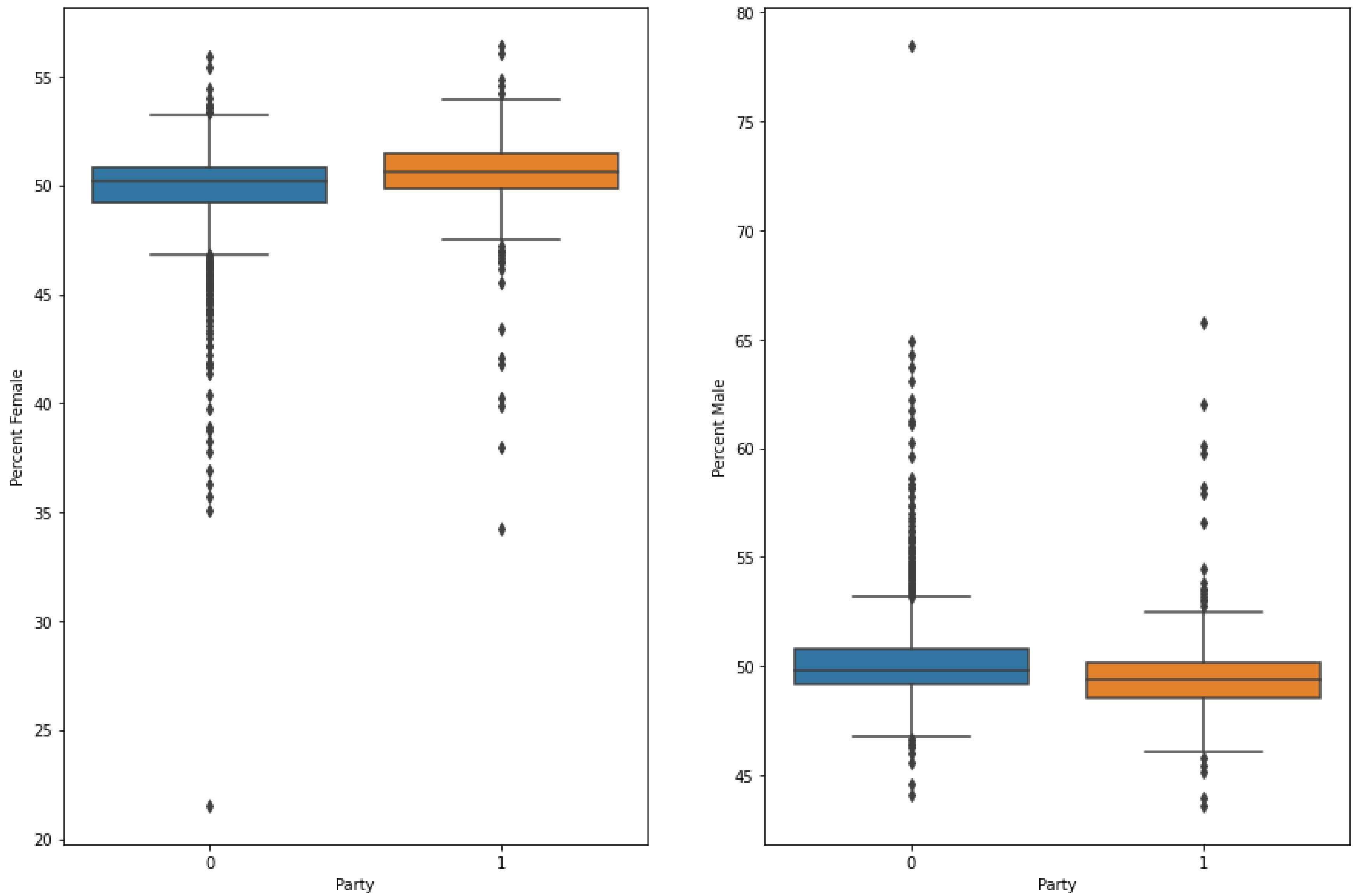
```
<ipython-input-24-f7b2122eaeb9>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-v
iew-versus-a-copy
  merge_dataframe['Percent Male'] = 100 - merge_dataframe['Percent Female']
<ipython-input-24-f7b2122eaeb9>:2: FutureWarning: Indexing with multiple keys (implicitly converted to a tuple of keys) w
ill be deprecated, use a list instead.
  merge_dataframe.groupby(by=['Party'])['Percent Female','Percent Male'].describe().T
```

Out[24]:

	Party	0	1
Percent Female	**count**	870.000000	325.000000
	mean	49.630898	50.385433
	std	2.429013	2.149359
	min	21.513413	34.245291
	25%	49.222905	49.854280
	50%	50.176792	50.653830
	75%	50.829770	51.492075
	max	55.885023	56.418468
Percent Male	**count**	870.000000	325.000000
	mean	50.369102	49.614567
	std	2.429013	2.149359
	min	44.114977	43.581532
	25%	49.170230	48.507925
	50%	49.823208	49.346170
	75%	50.777095	50.145720
	max	78.486587	65.754709

In [25]:
```
gender_groups = ['Percent Female','Percent Male']
```

```
fig, axes = plt.subplots(1, 2, figsize=(15, 10))
for index,group in enumerate(gender_groups):
    sns.boxplot(ax=axes[index], x="Party", y=group, data=merge_dataframe)
```



Conclusion

We can see that the mean of 'Percent female' & 'Percent male' voting are very close, although its more in the case of Democrats in 'Percent female' and republicans in case of 'Percent male'. We cannot conclude any county to be republican or democratic just on the basis of gender.

In [26]:

```
merge_dataframe.groupby(by=['Party'])['Percent White, not Hispanic or Latino','Percent Black, not Hispanic or Latino','Pe
```

```
<ipython-input-26-a1e8bda2b265>:1: FutureWarning: Indexing with multiple keys (implicitly converted to a tuple of keys) w
ill be deprecated, use a list instead.
  merge_dataframe.groupby(by=['Party'])['Percent White, not Hispanic or Latino','Percent Black, not Hispanic or Latin
o','Percent Hispanic or Latino','Percent Foreign Born'].describe().T
```

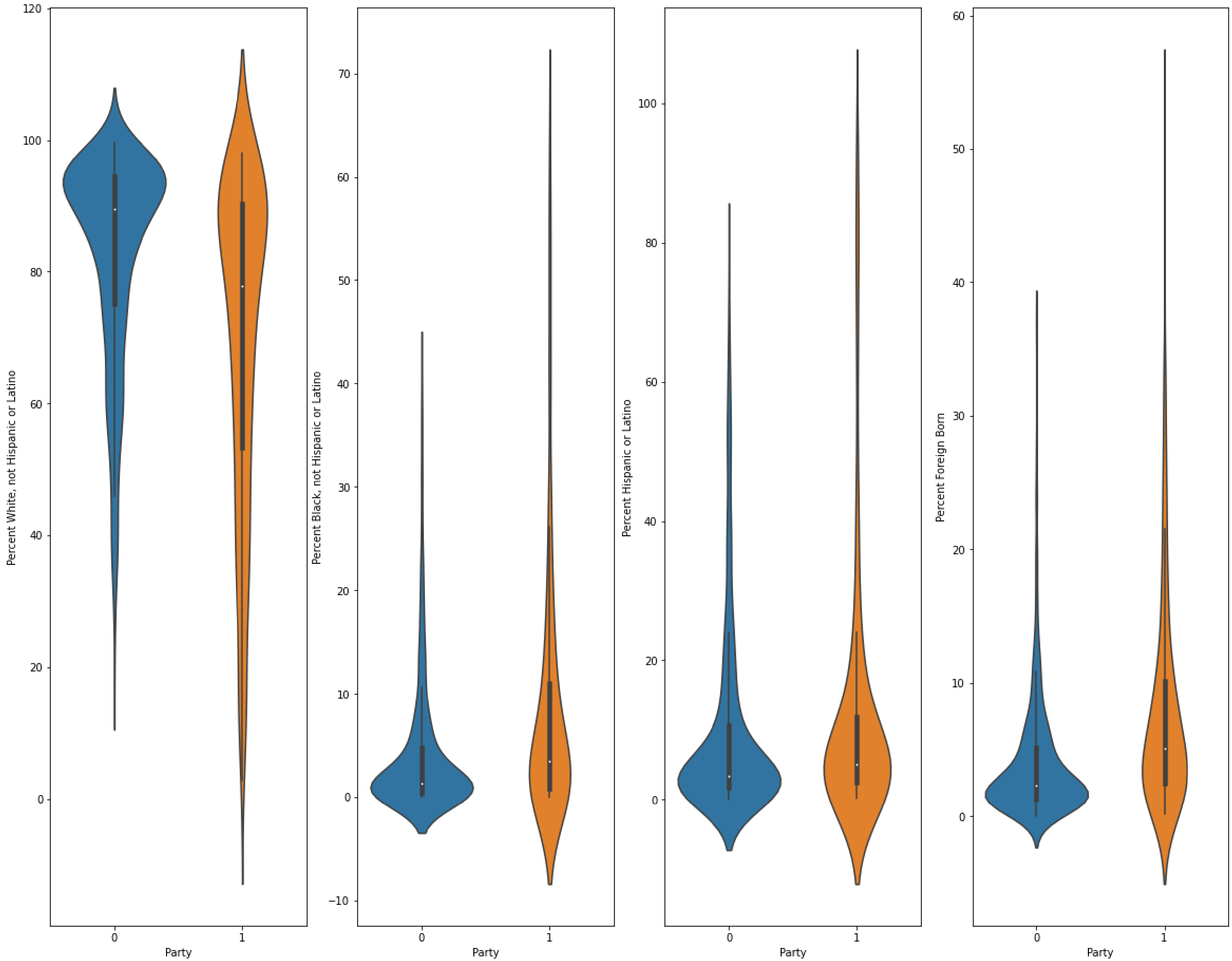
Out[26]:

	Party	0	1
Percent White, not Hispanic or Latino	**count**	870.000000	325.000000
	mean	82.656646	69.683766
	std	16.056122	24.981502
	min	18.758977	2.776702
	25%	75.016397	53.271579
	50%	89.434849	77.786090
	75%	94.466596	90.300749
	max	99.627329	98.063495
Percent Black, not Hispanic or Latino	**count**	870.000000	325.000000
	mean	4.189241	9.242649
	std	6.721695	13.351340
	min	0.000000	0.000000
	25%	0.460419	0.839103
	50%	1.318311	3.485992
	75%	4.753831	11.058843
	max	41.563041	63.953279
Percent Hispanic or Latino	**count**	870.000000	325.000000
	mean	9.733094	12.587391

	Party	0	1
	std	14.049576	19.575030
	min	0.000000	0.193349
	25%	1.704539	2.531017
	50%	3.427435	5.039747
	75%	10.709696	11.857116
	max	78.397012	95.479801
Percent Foreign Born	**count**	870.000000	325.000000
	mean	3.990096	7.986330
	std	4.507786	8.330740
	min	0.000000	0.179769
	25%	1.320101	2.470508
	50%	2.326317	5.105490
	75%	5.149429	10.144555
	max	37.058317	52.229868

In [27]:

```
ethnicity_groups = ['Percent White, not Hispanic or Latino','Percent Black, not Hispanic or Latino','Percent Hispanic or
fig, axes = plt.subplots(1, 4, figsize=(20, 16))
for index,group in enumerate(ethnicity_groups):
    sns.violinplot(ax=axes[index], x="Party", y=group, data=merge_dataframe)
```


Percent White, not Hispanic or Latino
Percent Black, not Hispanic or Latino
Percent Hispanic or Latino
Percent Foreign Born
Party
0
1

Conclusion

In case of Percent white, not Hispanic or Latino we can see that the counties with a greater number of them are republican counties whereas in the case of all other three categories counties with higher number of other ethnicities are more inclined towards Democrats.

In [28]:

```
ethnicity_groups = ['Percent Less than High School Degree', 'Percent Less than Bachelor\'s Degree', 'Bacherlor\'s Degree
merge_dataframe['Bacherlor\'s Degree and higher'] = 100 - merge_dataframe['Percent Less than Bachelor\'s Degree']
merge_dataframe.groupby(by=['Party'])['Percent Less than High School Degree', 'Percent Less than Bachelor\'s Degree', 'Ba
```

```
<ipython-input-28-3938638fd173>:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-v
iew-versus-a-copy
  merge_dataframe['Bacherlor\'s Degree and higher'] = 100 - merge_dataframe['Percent Less than Bachelor\'s Degree']
<ipython-input-28-3938638fd173>:3: FutureWarning: Indexing with multiple keys (implicitly converted to a tuple of keys) w
ill be deprecated, use a list instead.
  merge_dataframe.groupby(by=['Party'])['Percent Less than High School Degree', 'Percent Less than Bachelor\'s Degree',
'Bacherlor\'s Degree and higher'].describe().T
```

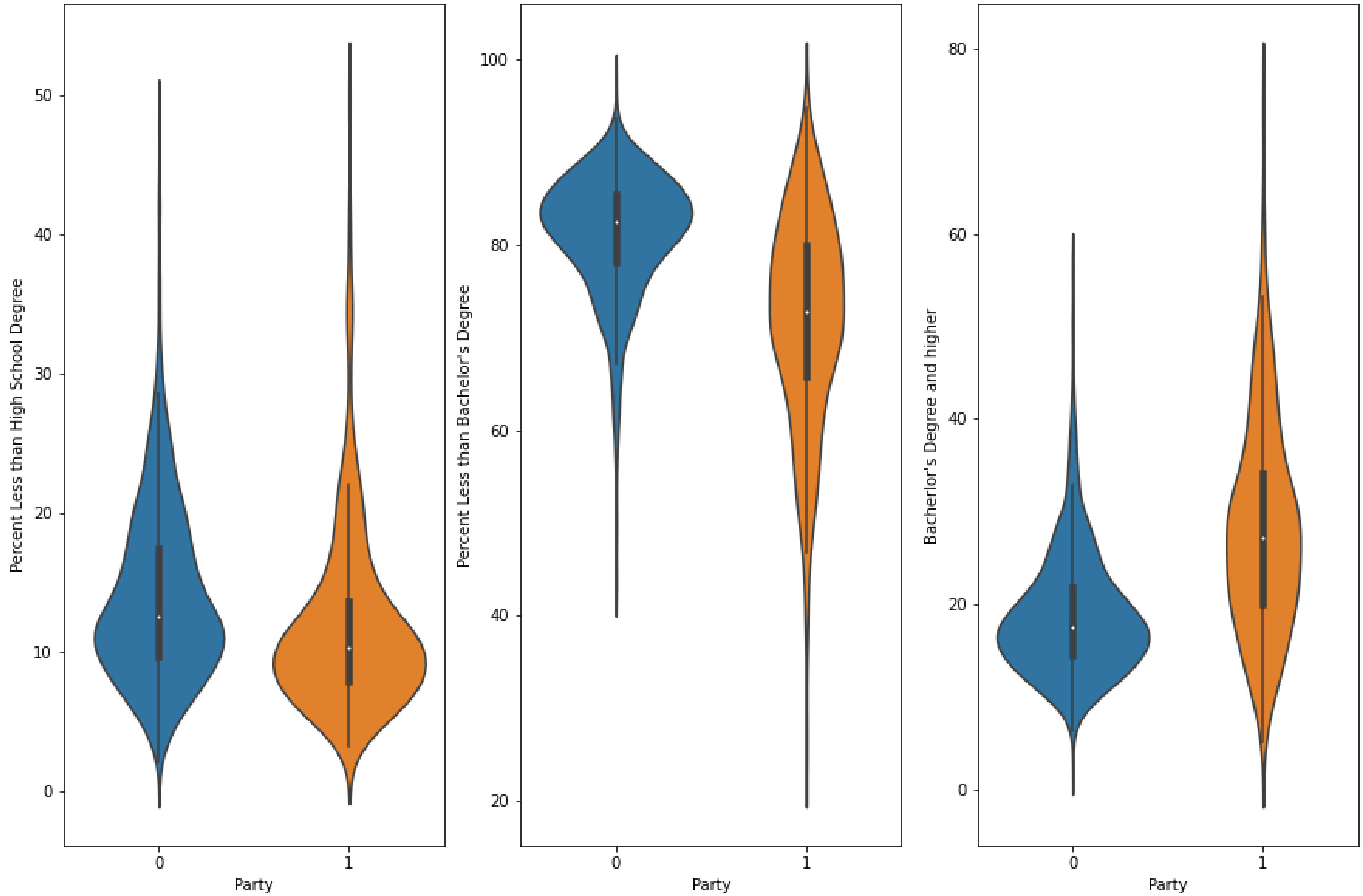
Out[28]:

	Party	0	1
Percent Less than High School Degree	**count**	870.000000	325.000000
	mean	14.009112	11.883760
	std	6.303126	6.505613
	min	2.134454	3.215803
	25%	9.662491	7.893714
	50%	12.572435	10.370080
	75%	17.447168	13.637059
	max	47.812773	49.673777
Percent Less than Bachelor's Degree	**count**	870.000000	325.000000
	mean	81.095427	71.968225
	std	6.815537	11.192404
	min	43.419470	26.335440
	25%	78.108424	65.711800
	50%	82.406700	72.736143

	Party	0	1
	75%	85.546272	79.903653
	max	97.014925	94.849957
Bacherlor's Degree and higher	**count**	870.000000	325.000000
	mean	18.904573	28.031775
	std	6.815537	11.192404
	min	2.985075	5.150043
	25%	14.453728	20.096347
	50%	17.593300	27.263857
	75%	21.891576	34.288200
	max	56.580530	73.664560

In [29]:

```
fig, axes = plt.subplots(1, 3, figsize=(15, 10))
for index,group in enumerate(ethnicity_groups):
    sns.violinplot(ax=axes[index], x="Party", y=group, data=merge_dataframe)
```



Conclusion

Counties with more percent of 'Percent Less than High School Degree' and 'Percent Less than Bachelor's Degree' people are inclined towards Republicans. Counties with more percent of people having 'Bacherlor's Degree and higher' are inclined towards Democrats.

Task 9: 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.

Total population is one of the important variable to determine whether a county is Republican or Democratic because the mean population of democratic counties is a lot higher than the republican counties which means the higher total population counties are inclined towards Democrats.

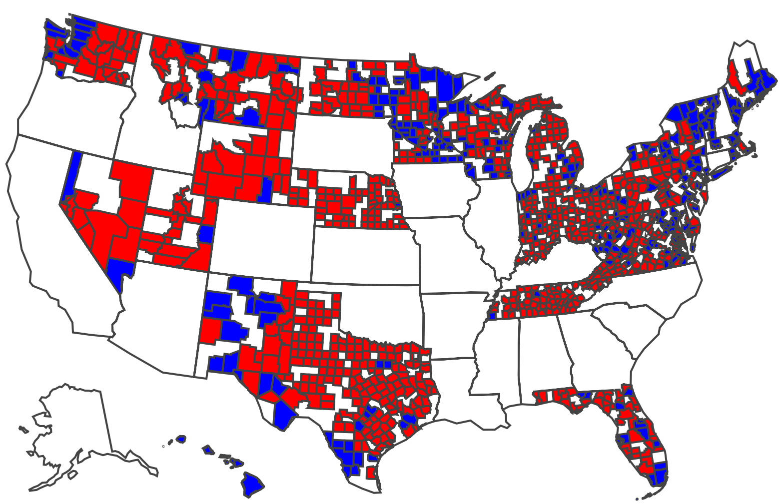
Education level(Percent Less than High School Degree/Bachelors Degree or Bacherlor's Degree and higher) and Ethnicity(Percent White or Percent black, Percent hispanic/latino) are very important variables in determining a county is democratic or republican because according to the descriptive statistics and plots the mean of percent of these different variables vary significantly more in terms of democrats and republicans.

In [30]:

```
with urlopen('https://raw.githubusercontent.com/plotly/datasets/master/geojson-counties-fips.json') as response:
    counties = json.load(response)
```

In [31]:

```
fips = merge_dataframe['FIPS'].to_list()
values = merge_dataframe['Party']

fig = px.choropleth(merge_dataframe,geojson=counties,locations='FIPS',scope='usa',color='Party',color_continuous_scale=px
fig.layout.template = None
fig.show()
```



```
In [ ]:
```