

Political Tendency Analysis and U.S Election Prediction

Zhexu Li, A14532514. Derrick Liu, A13987040.

1. Question: How income, population and geospatial features are related to the political tendency across US counties?

In this project, we will first analyze how income, population and geospatial features of counties correlated to their political affiliation, then use features derived from them to predict election results in those counties. Polls have been the usual tool to predict election outcomes despite having a marginal rate of success. By extracting features from geospatial data, census data and county vote data, it would not only provide insight into what features of counties motivate electorates insides, but also potentially create a model with higher election prediction accuracy than traditional models.

The intended audience would be any individual of the general public interested in the political demographics of US counties and its forming factors, as well as political leaders. The results of the analysis could help campaigning teams to understand the relationship between income and political parties in order to better address problems within each community. The findings might also be valuable for academic purposes, like for political scientists to better understand the hidden factors that influence political affiliation for certain counties.

Understanding this question could help us understand trends between income (and other features of the county) and proportion of people who identify themselves as either Democrats or Republicans in the county. We can also understand how this relationship progresses over years by using machine learning techniques like random forest classification. We expect for the overall geospatial features, income and population for counties that are primarily republican to be different from that of counties that are primarily democratic.

2. Background and Literature

Household Income and Political Affiliation

https://rstudio-pubs-static.s3.amazonaws.com/123239_d0c42d112e934abea0c1c250960a5cf5.html
(https://rstudio-pubs-static.s3.amazonaws.com/123239_d0c42d112e934abea0c1c250960a5cf5.html)

- This project analyzes the relationship between a person's family income and his/her's political affiliation and confirms the relationship between the two variables
- The conclusion mentions outliers in the data and cautions the interpretation of the results
- This project can be improved by cleaning the data in order to remove outliers

Partisan Politics and the U.S. Income Distribution

https://www.russellsage.org/sites/all/files/u4/Bartels_Partisan%20Politics_0.pdf
(https://www.russellsage.org/sites/all/files/u4/Bartels_Partisan%20Politics_0.pdf)

- This project analyzes and projects the income inequality through patterns of income growth observed during democrat administration as well as during republican administrations
- This is key to understanding the relationship between income and political party, as well as their trends during years of republican/democratic administrations
- The conclusion suggests that democratic controls produces a decrease in inequality while republican control produces higher polarization between different classes.

Income distribution and political participation: a multilevel analysis

https://www.researchgate.net/publication/276168635_Income_distribution_and_political_participation_a_multilevel
https://www.researchgate.net/publication/276168635_Income_distribution_and_political_participation_a_multilevel

- This project investigates the impact of income inequality on citizens' involvement in voting, membership of political groups, participation in political to broaden the theoretical perspective on the connection between income inequality and citizens' political participation
- Hypothesizes that people of richer contexts are more likely to be involved in conventional political activities.
- This could lead us to ask, is there a larger voter turnout in higher-income counties?

Political Polarization and Income Inequality

<https://www.princeton.edu/~nmccarty/ineqpold.pdf> (<https://www.princeton.edu/~nmccarty/ineqpold.pdf>)

- Argues that partisanship over time has become more stratified by income and that the trend is a consequence both of polarization of the parties on economic issues and increased economic inequality.

These articles help us understand that many of the analysis tackling the relationship between income and political affiliation tend to include the idea of income inequality. Other variables are also studied or mentioned in these articles, which inspired us to discover interesting features other than income, like relationship of party votes and population, relationship between votes and distance to closest population center, and relationship between votes and distance to the closest coastline.

Import Necessary Packages.

- Pandas: A common and powerful tool for fast and clean data processing. It's essential for data cleaning and data processing parts in data science pipeline. We also used it for parts of feature extraction and feature analysis.
- Numpy: A package useful for data manipulation, it allows fast computation for large amounts of data whatever it's stored in dataset or not.
- Geopandas: Powerful package for processing geospatial information. It's used for data analysis because it's simple yet quite powerful. Maps are generated using Geopandas since the quality of maps created by it is improved and it loads faster and actually looks great on a pdf.
- ArcGIS: Also powerful for processing geospatial information, and holds a large collection of high quality layers. We used ArcGIS.content to search for useful layers. There are three layers imported from ArcGIS.

This list is very similar to that of our proposal; most libraries such as pandas, GeoPandas, and ArcGIS were used during analysis as well as the resulting map.

```
In [1]: import pandas as pd
import arcgis
from arcgis.gis import GIS

import geopandas as gpd
import numpy as np
```

```
In [3]: gis = GIS(username='zh1411_dsc170fa20')
```

Enter password: ••••••••

Data Sources.

U.S Counties feature layer. Updated by USDA Forest Service on ArcGIS, includes geospatial information of all counties in the U.S in 3233 spatial objects.

Source: <https://ucsdonline.maps.arcgis.com/home/item.html?id=d8d3db28fe72445aa0449cfbcd6d0da3>
(<https://ucsdonline.maps.arcgis.com/home/item.html?id=d8d3db28fe72445aa0449cfbcd6d0da3>)

```
In [3]: county = gis.content.get("d8d3db28fe72445aa0449cfbcd6d0da3")
county
```

Out [3]:



US Counties and Equivalent Governmental Units

(<https://UCSDOnline.maps.arcgis.com/home/item.html?id=d8d3db28fe72445aa0449cfbcd6d0da3>)

US Counties and Equivalent Governmental Units



Feature Layer Collection by USFSMapsandApps

Last Modified: May 03, 2019

(<https://UCSDOnline.maps.arcgis.com/home/item.html?id=d8d3db28fe72445aa0449cfbcd6d0da3>)
Comments: 1, 282, 156 views

```
In [5]: co = county.layers[0].query(out_sr = 3857).sdf
co.head()
```

Out[5]:

	OBJECTID	STATECODE	COUNTYCODE	GNIS_ID	ST_CNTY_CODE	COUNTYNAME	LEGAL_N
0	6755	20	103	00485016	20103	Leavenworth	Leaver C
1	6756	48	281	01383927	48281	Lampasas	Lam C
2	6757	30	071	01720023	30071	Phillips	Phillips C
3	6758	53	045	01529221	53045	Mason	Mason C
4	6759	30	063	01719596	30063	Missoula	Mis C

```
In [10]: co.spatial.sr
```

Out[10]: {'wkid': 102100, 'latestWkid': 3857}

U.S Major Cities. Updated by Esri on ArcGIS, includes the location of U.S cities which population is >= 10000, contains 3886 records.

Source: <https://ucsdonline.maps.arcgis.com/home/item.html?id=85d0ca4ea1ca4b9abf0c51b9bd34de2e>
(<https://ucsdonline.maps.arcgis.com/home/item.html?id=85d0ca4ea1ca4b9abf0c51b9bd34de2e>)

```
In [11]: cities = gis.content.get("85d0ca4ea1ca4b9abf0c51b9bd34de2e")
cities
```

Out[11]:



USA Major Cities

(<https://UCSDOnline.maps.arcgis.com/home/item.html?id=85d0ca4ea1ca4b9abf0c51b9bd34de2e>).

This layer presents the locations of cities within the United States with populations of approximately 10,000 or greater, all state capitals, and the national capital.



(<https://UCSDOnline.maps.arcgis.com/home/item.html?id=85d0ca4ea1ca4b9abf0c51b9bd34de2e>)
Feature Layer Collection by esri_dm
Last Modified: May 19, 2020
1 comments, 31,357,459 views

```
In [12]: cs = cities.layers[0].query().sdf
cs = cs[["NAME", "POPULATION", "SHAPE"]]
cs.head()
```

Out[12]:

	NAME	POPULATION	SHAPE
0	Ammon	15181	{"x": -12462673.723706165, "y": 5384674.994080...
1	Blackfoot	11946	{"x": -12506251.313993266, "y": 5341537.793529...
2	Boise City	225405	{"x": -12938676.6836459, "y": 5403597.04949123...
3	Burley	10727	{"x": -12667411.402393516, "y": 5241722.820606...
4	Caldwell	53942	{"x": -12989383.674504515, "y": 5413226.487333...

```
In [13]: cs.spatial.sr
```


Out[13]: {'wkid': 102100, 'latestWkid': 3857}

U.S Coast Lines. Updated on ArcGIS by sjones_ALHub, original source is Geography division in U.S Census Bureau. It contains coastlines in the U.S.


Source: <https://ucsdonline.maps.arcgis.com/home/item.html?id=a65b254f5e1e4a1989b2bc7ccdbbec48>
(<https://ucsdonline.maps.arcgis.com/home/item.html?id=a65b254f5e1e4a1989b2bc7ccdbbec48>)

```
In [22]: coastline = gis.content.get("a65b254f5e1e4a1989b2bc7ccdbbec48")
coastline
```

Out[22]:



US Coastline
(<https://UCSDOnline.maps.arcgis.com/home/item.html?id=a65b254f5e1e4a1989b2bc7ccdbbec48>)
Source: US Census Bureau, Geography Division

 Feature Layer Collection by sjones_ALHub
Last Modified: February 09, 2018

(<https://UCSDOnline.maps.arcgis.com/home/item.html?id=a65b254f5e1e4a1989b2bc7ccdbbec48>) **0 comments** **430 views**

```
In [25]: coast = coastline.layers[0].query().sdf
coast.head(5)
```

Out[25]:

	FID	NAME	MTFCC	Shape_Length	SHAPE
0	1	Gulf	L4150	442.489268	{"paths": [[[-10057826.7178824, 3400824.573682...
1	2	Gulf	L4150	23134.699335	{"paths": [[[-10057229.5997463, 3394326.253773...
2	3	Gulf	L4150	625.628241	{"paths": [[[-10057640.1457142, 3382018.612427...
3	4	Gulf	L4150	705.515457	{"paths": [[[-10060205.0595829, 3404730.652197...
4	5	Gulf	L4150	559.061286	{"paths": [[[-10052175.805181, 3404804.6376785...

```
In [27]: coast.spatial.sr
```

```
Out[27]: {'wkid': 102100, 'latestWkid': 3857}
```

U.S Presidential Election results in 2012 and 2016. Uploaded by Joel Wilson on Kaggle, contains electoral votes in U.S counties in 2012 and 2016 election.

Source: https://www.kaggle.com/joelwilson/2012-2016-presidential-elections?select=US_County_Level_Presidential_Results_12-16.csv (https://www.kaggle.com/joelwilson/2012-2016-presidential-elections?select=US_County_Level_Presidential_Results_12-16.csv)

```
In [28]: former = pd.read_csv("ele_train.csv")
former.head()
```

```
Out[28]:
```

	Unnamed: 0	combined_fips	votes_dem_2016	votes_gop_2016	total_votes_2016	per_dem_2016
0	0	2013	93003.0	130413.0	246588.0	0.377159
1	1	2016	93003.0	130413.0	246588.0	0.377159
2	2	2020	93003.0	130413.0	246588.0	0.377159
3	3	2050	93003.0	130413.0	246588.0	0.377159
4	4	2060	93003.0	130413.0	246588.0	0.377159

5 rows × 21 columns

U.S Presidential Election results in 2020. Uploaded by Raphael Fontes on Kaggle, contains election results in U.S counties in 2020 presidential election.

Source: https://www.kaggle.com/unanimad/us-election-2020?select=president_county_candidate.csv (https://www.kaggle.com/unanimad/us-election-2020?select=president_county_candidate.csv)

```
In [29]: ele = pd.read_csv("ele.csv")
ele.head()
```

```
Out[29]:
```

	state	county	candidate	party	total_votes	won
0	Delaware	Kent County	Joe Biden	DEM	44552	True
1	Delaware	Kent County	Donald Trump	REP	41009	False
2	Delaware	Kent County	Jo Jorgensen	LIB	1044	False
3	Delaware	Kent County	Howie Hawkins	GRN	420	False
4	Delaware	New Castle County	Joe Biden	DEM	195034	True

Annual Personal Income and Total Population by County, retrieved from public resources in U.S Bureau of Economic Analysis.

<https://apps.bea.gov/regional/downloadzip.cfm> (<https://apps.bea.gov/regional/downloadzip.cfm>).

```
In [30]: incomes = pd.read_csv("incomes.csv", encoding = "ISO-8859-1")
incomes.head()
```

Out[30]:

	GeoFIPS	GeoName	Region	TableName	LineCode	IndustryClassification	Description	
0	"00000"	United States		CAINC1	1.0	...	Personal income (thousands of dollars)	Thousi of dc
1	"00000"	United States		CAINC1	2.0	...	Population (persons) 1/	Numb per:
2	"00000"	United States		CAINC1	3.0	...	Per capita personal income (dollars) 2/	Dc
3	"01000"	Alabama	5	CAINC1	1.0	...	Personal income (thousands of dollars)	Thousi of dc
4	"01000"	Alabama	5	CAINC1	2.0	...	Population (persons) 1/	Numb per:

5 rows × 59 columns

At first, we decided only use the U.S counties feature layer, the income by counties dataset and the 2020 presidential election result dataset. After we read through related resources mentioned in "Background and Literature" above, we realized the necessity of additional features in order to capture more underlying factors related to election, and achieve higher prediction accuracy. We then search through ArcGIS and Kaggle and added three more dataset: U.S Major Cities feature layer, U.S Coastline feature layer and 2012, 2016 presidential election dataset. These dataset are useful for our analysis and model developing.

The U.S counties feature layer, U.S Major Cities feature layer, and census dataset are retrieved from credible sources (USDA Forest Service, Esri, and U.S Bureau of Economic Analysis) so the quality of those data is relatively high. The election dataset are retrieved from Kaggle, but they are both highly rated by other Kaggle users so we believe the quality of those dataset should not be a problem.

The ideal dataset for our study is a dataset containing income, population, geospatial features and election results for all U.S counties in different years. It allows us to directly analyze their relationships without worrying about the quality of spatial joins, and it will greatly simplify our analysis process. But we searched through the Internet and it seems such ideal dataset doesn't exist yet. Note there are several dataset suggested by professor and experts during the presentation, those dataset might be great for our analysis but we need to change the entire structure of our analysis in order to switch to other dataset. We include it in future works because it is too time-consuming for now.

Data Cleaning.

According to source documentation of those dataset, they are already cleaned by its uploaders. Considering they are both retrieved from trustworthy sources, we believe we can directly use those dataset. One issue is that we found the election result is lacking for Alaska in 2012 and 2016, so we excluded it from the dataset in analysis steps because we currently don't see better datasets on Kaggle. And we expect a lot of data preparation to be done, since there are six different dataset and we need to combine them together. The data preparation is quite a long process since it includes feature extraction and other steps, we believe it's better to include it in the following section instead of just listing it there without any explanation.

Descriptive Statistics.

We combine Data Preparation, or more specifically, Feature Extraction into descriptive statistics section because the process of Descriptive Statistics is encouraging us to generate new features from existing features after we observe their relation or underlying pattern. It's also easier for reader to understand the features we created if we include visualizations.

Visualize counties, major cities, and coastline in the US

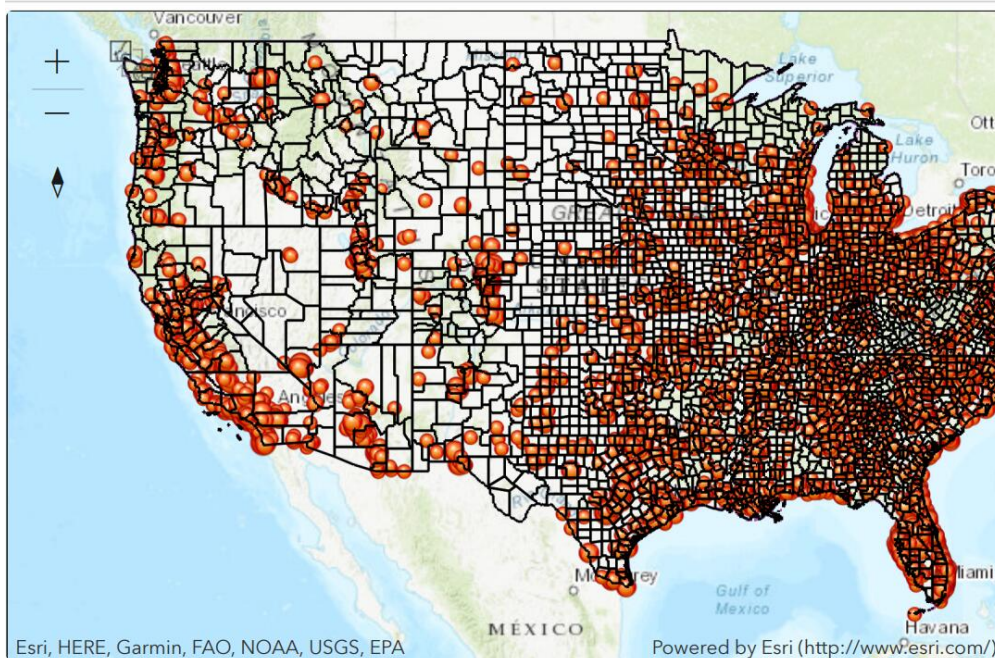
```
In [151]: mp = gis.map("United States")
mp.add_layer(county)
mp.add_layer(cities)

mp.add_layer(coastline)
mp
```

In case you don't want to run the codes, the result is:

```
In [171]: from IPython.display import Image
Image(filename = "map.jpg", width = 500, height = 200)
```

Out[171]:



The map is a little bit hard to interpret. But it seems the sizes of counties varies a lot, and major cities tends to cluster together. These pattern might be helpful for our further analysis of spatial relationships.

Discovering income

We extract per capita personal income for every county, combine it to the counties spatial dataset and visualize it on a choropleth map.

```
In [31]: income = incomes.drop(["GeoFIPS", "Region", "TableName", "IndustryClassification", "Description", "Unit"], axis = 1)
income = income[income["LineCode"] == 3] # LineCode = 3 indicates income
income = income[income["GeoName"].str.contains(",")]

income.head()
```

Out[31]:

	GeoName	LineCode	1969	1970	1971	1972	1973	1974	1975	1976	...	2010	2011	2012
8	Autauga, AL	3.0	2780	3158	3454	3687	4039	4246	4431	5035	...	33348	34337	35326
11	Baldwin, AL	3.0	2760	2905	3270	3584	4176	4628	5088	5781	...	36143	37881	38850
14	Barbour, AL	3.0	2147	2545	2686	3068	3401	3847	3962	4428	...	27770	28163	28556
17	Bibb, AL	3.0	1988	2359	2630	2897	3180	3530	3811	4394	...	25057	25993	26930
20	Blount, AL	3.0	2625	2595	2790	3108	3618	3669	4312	4679	...	27701	28400	29100

5 rows × 53 columns

```
In [33]: st_lst = income["GeoName"].str.split(",") # Reformat the GeoName
income["st"] = st_lst.str[1]
income["county"] = st_lst.str[0]

income["st"] = income["st"].str.replace(" ", "")
income["st"] = income["st"].str.replace(";", "")
income.head()
```

Out[33]:

	GeoName	LineCode	1969	1970	1971	1972	1973	1974	1975	1976	...	2012	2013	2014
8	Autauga, AL	3.0	2780	3158	3454	3687	4039	4246	4431	5035	...	35067	35538	36010
11	Baldwin, AL	3.0	2760	2905	3270	3584	4176	4628	5088	5781	...	38259	38222	38185
14	Barbour, AL	3.0	2147	2545	2686	3068	3401	3847	3962	4428	...	28206	30092	29975
17	Bibb, AL	3.0	1988	2359	2630	2897	3180	3530	3811	4394	...	27042	27417	27792
20	Blount, AL	3.0	2625	2595	2790	3108	3618	3669	4312	4679	...	29647	30320	31000

5 rows × 55 columns

Now we have income extracted for every county. Let's Join spatial dataframe county to incomes.

```
In [35]: co["joinname"] = co["COUNTYNAME"] + ", " + co["STATE_POSTAL_ABBR"]
co_income = co[["joinname", "SHAPE"]].merge(income, left_on = "joinname", right_on = "GeoName", how = "inner")
co_income.head()
```

Out[35]:

	joinname	SHAPE	GeoName	LineCode	1969	1970	1971	1972	1973
0	Leavenworth, KS	{'rings': [[[-10571898.558177462, 4781852.2564...	Leavenworth, KS	3.0	3259	3586	3760	4348	4800
1	Lampasas, TX	{'rings': [[[-10898969.37498506, 3641762.65628...	Lampasas, TX	3.0	3389	3458	3502	3632	3645
2	Phillips, MT	{'rings': [[[-11931203.449280191, 6274880.1111...	Phillips, MT	3.0	2867	3521	3392	4537	6175
3	Mason, WA	{'rings': [[[-13706521.885616744, 6041746.2540...	Mason, WA	3.0	3774	3817	4146	4578	5062
4	Missoula, MT	{'rings': [[[-12649689.290937627, 6040570.7455...	Missoula, MT	3.0	3382	3574	3896	4253	4509

5 rows × 57 columns

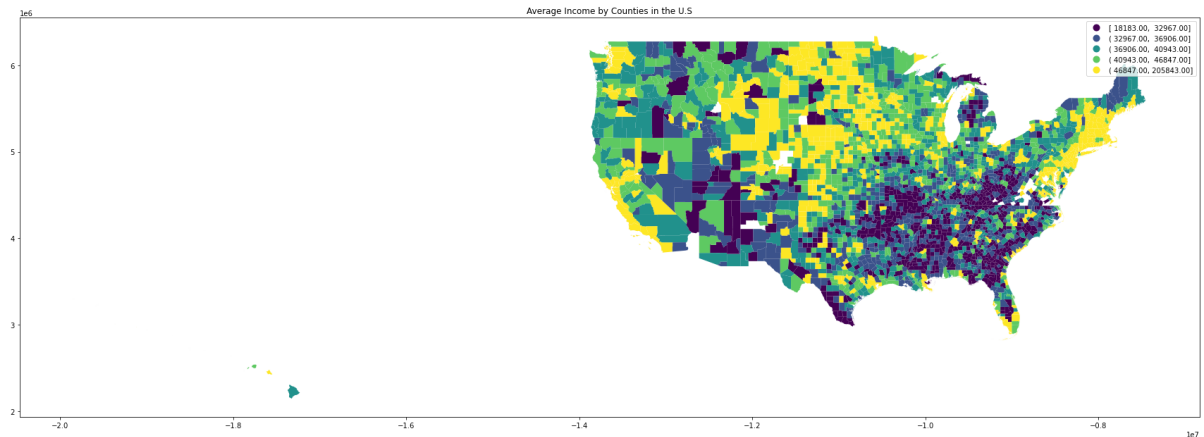
Visualize Average Income by Counties

Now we can Visualize income level by county on a choropleth map. Note we don't add labels to axis because it's unnecessary for map.

```
In [50]: import geopandas as gpd
co_gpd = gpd.GeoDataFrame(co_income)
co_gpd = co_gpd.set_geometry("SHAPE")

co_gpd["2016"] = co_gpd["2016"].astype("int")
mp = co_gpd.plot(column = "2016", legend = True, scheme = 'quantiles', figsize = (30, 30))
mp.set(title = "Average Income by Counties in the U.S")
```

```
Out[50]: [Text(0.5, 1.0, 'Average Income by Counties in the U.S')]
```



It seems average income is higher in southwest and northeast counties. We will compare the map to the political map later to see if we can observe any relationship. Notice we don't have Alaska and several counties are missing in our map, because certain information about those places are missing we had to removed it from the analysis. It might be unfair for people there but we really do not hav other resources available right now.

Visualize Total Population by Counties.

```

In [51]: income = incomes.drop(["GeoFIPS", "Region", "TableName", "IndustryClassification", "Description", "Unit"], axis = 1)
pop = income[income["LineCode"] == 2] # This indicates population
pop = pop[pop["GeoName"].str.contains(",")]

st_lst = pop["GeoName"].str.split(",")
pop["st"] = st_lst.str[1]
pop["county"] = st_lst.str[0]

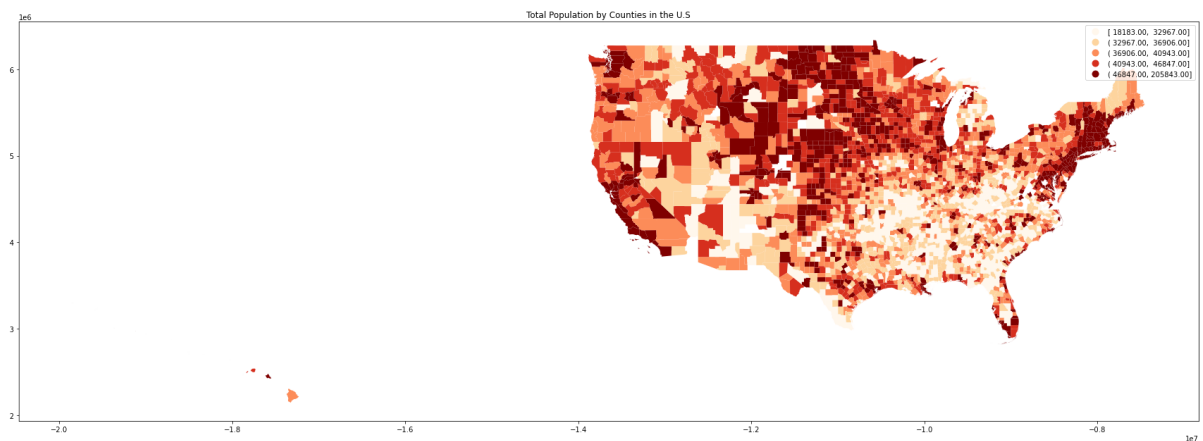
pop["st"] = pop["st"].str.replace(" ", "")
pop["st"] = pop["st"].str.replace("*", "")
co["joinname"] = co["COUNTYNAME"] + ", " + co["STATE_POSTAL_ABBR"]

cipop = co[["joinname", "SHAPE"]].merge(pop, left_on = "joinname", right_on = "GeoName",
, how = "inner")
cpgpd = gpd.GeoDataFrame(co_income)
cpgpd = cpgpd.set_geometry("SHAPE")

cpgpd["2016"] = cpgpd["2016"].astype("int")
mpl = cpgpd.plot(column = "2016", legend = True, cmap = "OrRd", scheme = 'quantiles', figsize = (30, 30))
mpl.set(title = "Total Population by Counties in the U.S")

```

```
Out[51]: [Text(0.5, 1.0, 'Total Population by Counties in the U.S')]
```



It's pretty similar to the income map, population clustered in southwest and northeast. We will compare this map to political map later to see if there is any relationship.

Join income, population and county.

```
In [52]: copop = copop[["SHAPE", "GeoName", "st", "county", "2012", "2016", "2019"]]
co_income = co_income[["SHAPE", "GeoName", "st", "county", "2012", "2016", "2019"]]
in_po_co = copop.merge(co_income, on = ["GeoName", "SHAPE", "st", "county"], suffixes =
("_pop", "_income"))

in_po_co.head()
```

Out[52]:

	SHAPE	GeoName	st	county	2012_pop	2016_pop	2019_pop	2012_
0	{'rings': [[[-10571898.558177462, 4781852.2564...]]}	Leavenworth, KS	KS	Leavenworth	77644	80244	81758	
1	{'rings': [[[-10898969.37498506, 3641762.65628...]]}	Lampasas, TX	TX	Lampasas	20054	20524	21428	
2	{'rings': [[[-11931203.449280191, 6274880.1111...]]}	Phillips, MT	MT	Phillips	4121	4113	3954	
3	{'rings': [[[-13706521.885616744, 6041746.2540...]]}	Mason, WA	WA	Mason	60681	62142	66768	
4	{'rings': [[[-12649689.290937627, 6040570.7455...]]}	Missoula, MT	MT	Missoula	111042	116349	119600	

Join income, population and spatial information to presidential dataframes.

```
In [53]: former = pd.read_csv("ele_train.csv")
former = former[["state_abbr", "county_name", "votes_dem_2016", "votes_gop_2016", "votes_dem_2012", "votes_gop_2012"]]
former = former.dropna()

former = former.rename(columns = {"state_abbr": "st", "county_name": "county"})
former["county"] = former["county"].str.replace(" County", "")
former = former.merge(in_po_co, on = ["st", "county"], how = "left")
former
```

Out[53]:

	st	county	votes_dem_2016	votes_gop_2016	votes_dem_2012	votes_gop_2012	
0	AL	Autauga	5908.0	18110.0	6354.0	17366.0	[[[-1:
1	AL	Baldwin	18409.0	72780.0	18329.0	65772.0	[[[-1:
2	AL	Barbour	4848.0	5431.0	5873.0	5539.0	[[[-1:
3	AL	Bibb	1874.0	6733.0	2200.0	6131.0	[[[-1:
4	AL	Blount	2150.0	22808.0	2961.0	20741.0	[[[-1:
...
3111	WY	Sweetwater	3233.0	12153.0	4773.0	11427.0	[[[-1:
3112	WY	Teton	7313.0	3920.0	6211.0	4858.0	[[[-1:
3113	WY	Uinta	1202.0	6154.0	1628.0	6613.0	[[[-1:
3114	WY	Washakie	532.0	2911.0	794.0	3013.0	[[[-1:
3115	WY	Weston	294.0	2898.0	422.0	2821.0	[[[-1:

3116 rows × 14 columns

Derive "Political Tendency" in dataframes.

As Professor mentioned in feedbacks, We define Political Tendency to be the ratio of votes between Democrats and Republicans candidates ($\text{votes_dem} / \text{votes_gop}$). Values higher than 1 indicates more votes for democrats candidates, and vice versa. We don't use the Esri market potential dataset suggested by experts in presentation because we want to focus on elections.

```
In [55]: former["pt_2012"] = former["votes_dem_2012"] / former["votes_gop_2012"]
former["pt_2016"] = former["votes_dem_2016"] / former["votes_gop_2016"]
former[["GeoName", "pt_2012", "pt_2016"]].head()
```

Out[55]:

	GeoName	pt_2012	pt_2016
0	Autauga, AL	0.365887	0.326229
1	Baldwin, AL	0.278675	0.252940
2	Barbour, AL	1.060300	0.892653
3	Bibb, AL	0.358832	0.278331
4	Blount, AL	0.142761	0.094265

Visualize political tendency for counties.

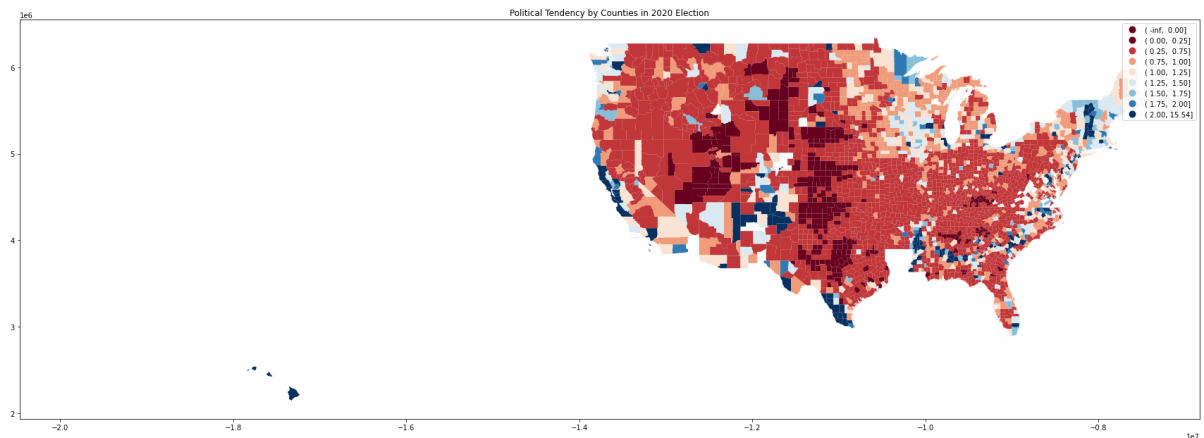
```
In [57]: fs_gpd = gpd.GeoDataFrame(former)
fs_gpd = fs_gpd.set_geometry("SHAPE")
```

Political Tendency by Counties in 2012 Election

```
In [59]: # We fill 1 (tie) for missing values, this is just for mapping, it's not going to affect training.
fs_gpd["pt_2012"] = fs_gpd["pt_2012"].fillna(1)
mpt = fs_gpd.plot(column = "pt_2012", legend = True, cmap = "RdBu",

                 scheme = 'user_defined', classification_kwds = {'bins': [0, 0.25, 0.75, 1, 1.25, 1.5, 1.75, 2]}),
                 figsize = (30, 30))
mpt.set(title = "Political Tendency by Counties in 2020 Election")
```

```
Out[59]: [Text(0.5, 1.0, 'Political Tendency by Counties in 2020 Election')]
```



It seems most counties in central America support Republicans, while most counties along coastlines support Democrat.

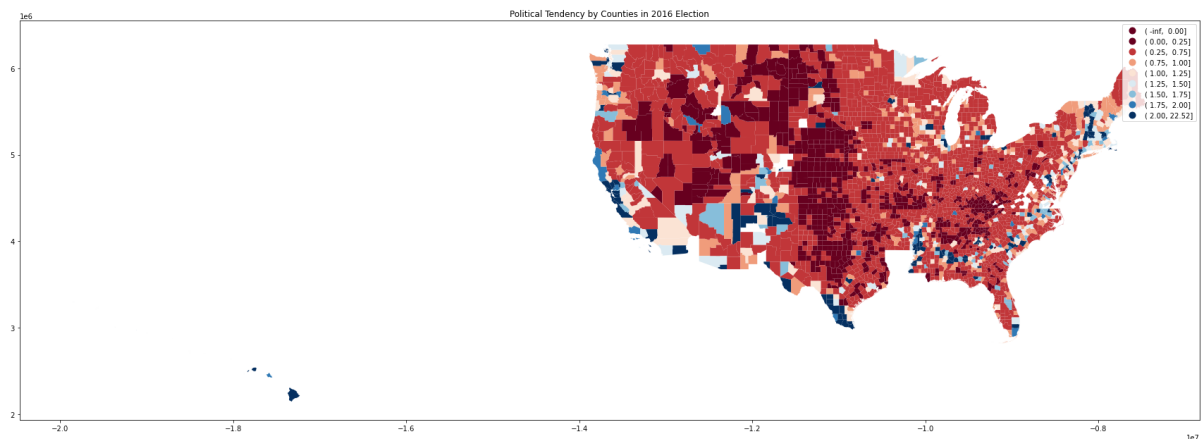
Political Tendency by Counties in 2016 Election

```
In [61]: # Do the same for 2016 elections
fs_gpd["pt_2016"] = fs_gpd["pt_2016"].fillna(1)
fs_gpd.crs = "EPSG:3857"

mp = fs_gpd.plot(column = "pt_2016", legend = True, cmap = "RdBu",

                 scheme = 'user_defined', classification_kwds = {'bins': [0, 0.25, 0.75, 1,
1.25, 1.5, 1.75, 2]}),
                 figsize = (30, 30))
mp.set(title = "Political Tendency by Counties in 2016 Election")
```

```
Out[61]: [Text(0.5, 1.0, 'Political Tendency by Counties in 2016 Election')]
```



The result doesn't seem to vary a lot from 2012 election, while the notable difference is Wisconsin turned red in 2016.

It doesn't look good for Democrats right?

Don't worry, let's extract feature from Major Cities Layer and see how is that related to the political tendency.

As suggested by Professor, We define feature "dis_pop_center" to be the county's distance to closest population center. The "population center" is defined by major cities whose population is bigger than or equal to 500000 people. We will see how those population centers related to the political tendency in different counties.

```
In [67]: cs = cs[cs["POPULATION"] >= 500000]
cs
```

Out[67]:

	NAME	POPULATION	SHAPE
62	Chicago	2781116	{"x": -9756835.705284344, "y": 5124572.0445172...
279	Indianapolis	864712	{"x": -9589766.807367168, "y": 4833603.6083573...
414	Los Angeles	3986442	{"x": -13165820.99345245, "y": 4035892.5853389...
549	San Diego	1397856	{"x": -13040584.546974456, "y": 3858240.841215...
553	San Francisco	871042	{"x": -13630231.137502154, "y": 4546552.050080...
557	San Jose	1042940	{"x": -13568333.643846016, "y": 4486522.906773...
673	Denver	699521	{"x": -11686600.752084276, "y": 4826692.811135...
763	Washington	674875	{"x": -8574678.18779218, "y": 4705822.67561191...
909	Phoenix	1601381	{"x": -12476005.927345268, "y": 3954668.990210...
930	Tucson	539162	{"x": -12353066.832517056, "y": 3792182.081440...
1127	Fresno	525594	{"x": -13335384.259593802, "y": 4403266.246591...
1314	Baltimore	620488	{"x": -8527908.59092266, "y": 4763321.88630549...
1436	Boston	661977	{"x": -7909997.096692591, "y": 5214997.9015214...
1532	Detroit	656087	{"x": -9246816.84030526, "y": 5210840.97090118...
1677	Jacksonville	886969	{"x": -9089865.86652973, "y": 3546319.41030348...
2036	Albuquerque	567516	{"x": -11872160.492018322, "y": 4176562.219010...
2173	New York	8691599	{"x": -8238770.183515309, "y": 4969744.1655956...
2263	Charlotte	838742	{"x": -8997982.833413022, "y": 4192166.4273739...
2386	Columbus	871273	{"x": -9238285.460804678, "y": 4864257.4003026...
2551	Seattle	687870	{"x": -13616617.94888544, "y": 6039155.5722018...
2636	Milwaukee	591865	{"x": -9785753.490007848, "y": 5318127.2846952...
2858	Oklahoma City	665635	{"x": -10855782.83326024, "y": 4227619.7782277...
2912	Portland	637683	{"x": -13653315.021931803, "y": 5703126.712440...
2986	Philadelphia	1587761	{"x": -8366882.6254769275, "y": 4858876.345453...
3363	Las Vegas	642798	{"x": -12816785.82245754, "y": 4324931.5075030...
3513	Memphis	668228	{"x": -10023451.237307431, "y": 4183064.782282...
3519	Nashville	672371	{"x": -9660005.729731545, "y": 4323551.7816427...
3548	Austin	935806	{"x": -10880543.896767916, "y": 3537909.436095...
3591	Dallas	1323651	{"x": -10775254.59703981, "y": 3865959.6760583...
3604	El Paso	693738	{"x": -11854098.222693913, "y": 3731534.170835...
3612	Fort Worth	851362	{"x": -10832839.340519603, "y": 3861747.999752...
3640	Houston	2333285	{"x": -10616262.315905476, "y": 3472578.411494...
3725	San Antonio	1442472	{"x": -10964134.924024679, "y": 3429663.909911...

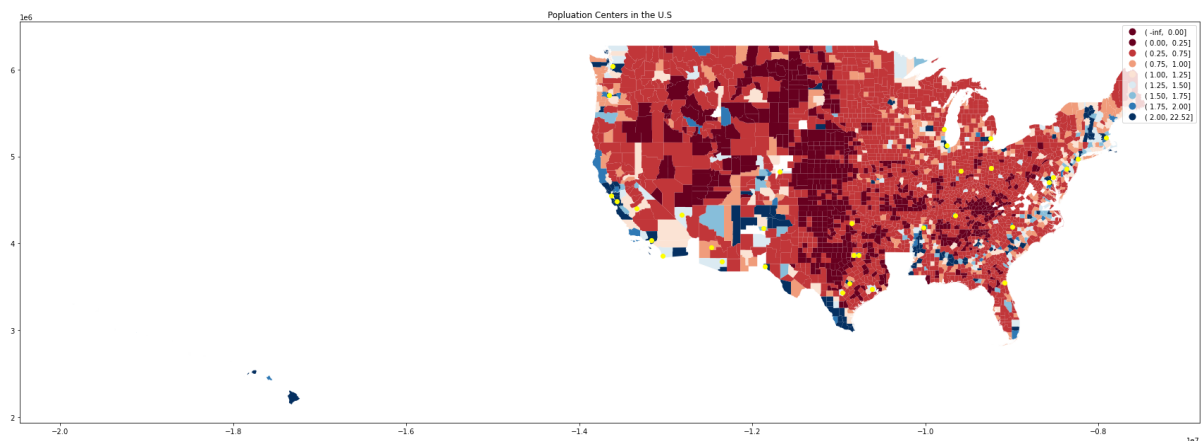
Now we have population centers, let's plot it on the map for 2016 election results.

```
In [68]: # Let's plot those population centers on the map
cs_gpd = gpd.GeoDataFrame(cs)
cs_gpd = cs_gpd.set_geometry("SHAPE")

cs_gpd.crs = "EPSG:3857"
incounty = gpd.sjoin(cs_gpd, fs_gpd, op = "within")
mp = fs_gpd.plot(column = "pt_2016", legend = True, cmap = "RdBu",
                 scheme = 'user_defined', classification_kwds = {'bins': [0, 0.25, 0.75, 1,
1.25, 1.5, 1.75, 2]}),

                 figsize = (30, 30))
mp = incounty.plot(ax = mp, color = "yellow", figsize = (30, 30))
mp.set(title = "Popluation Centers in the U.S")
```

Out[68]: [Text(0.5, 1.0, 'Popluation Centers in the U.S')]



It seems most population centers are in "blue" counties, but it's a little bit hard to see on the map because the yellow dot actually covers the county. Let's do one more spatial join to see the actual number.

```
In [69]: # Calculate the actual number of population centers located in "blue" counties.
dem_co = fs_gpd[fs_gpd["pt_2016"] > 1] # Counties whose votes for Democrats is more th
an for Republicans.
dem_cs = gpd.sjoin(cs_gpd, dem_co, op = "within")

print("Out of " + str(len(cs)) + " population centers, there are "

      + str(len(dem_cs)) + " population centers ("
      + str(len(dem_cs) / len(cs) * 100)[0:5] + "%) located in blue counties.")
```

Out of 33 population centers, there are 29 population centers (87.87%) located in blue counties.

Yes, most population centers are located in counties that's more democratic.

Which brings us a question, how could the political tendency of those population centers affect nearby counties? Or in other words, is there a relationship between the county's distance to nearest population center and its political tendency?

Analyze Relationship

Outline In the Analysis section, we analyze how the income, population, and geospatial information features are related to the county's political tendency. We use scatter, correlation and coefficient of best fitting line generated by linear regression to analyze the relationship. Specifically:

- We first discover the relationship between the county's distance to closest population center and the county's political tendency.
- We then analyze how the county's population related to its political tendency.
- We analyze how the county's average income and its political tendency are related.
- We discover the relationship between the county's distance to closest coastline and it's political tendency.

After we finish analyzing those features, we can do predictions on election results!

The analysis steps are different to plans we stated in the proposal. Mostly because we add three dataset outside of the sources we planned in the proposal. The definition of political tendency changed to the ratio between Democratic votes and Republican votes according to the suggestion by Professor. The original definition is the difference between the number of Democratic votes and the number of Republican votes and divided by the total number of votes in the county, which is a little bit too complex. We no longer use the linear regression model for prediction, instead we use it for generating beest fitting lines to better interpret the correlations.

Discover the relationship between the county's distance to closest population center and the county's political tendency

Let's calculate the "dis_pop_center", which is the county's distance to closest population center.

```
In [71]: def closest_dis(row):
          # Return na if no shape, so it will not interfere training process
          if row["SHAPE"] is None:

              return np.nan
          else:
              dis = cs_gpd["SHAPE"].distance(row["SHAPE"])
              return min(dis)

          fs_gpd["dis_pop_center"] = fs_gpd.apply(closest_dis, axis = 1)
```

```
In [72]: fs_gpd[["GeoName", "dis_pop_center"]]
```

```
Out[72]:
```

	GeoName	dis_pop_center
0	Autauga, AL	467299.137768
1	Baldwin, AL	566512.048465
2	Barbour, AL	428837.226782
3	Bibb, AL	396462.197238
4	Blount, AL	261723.239062
...
3111	Sweetwater, WY	376020.556426
3112	Teton, WY	783545.085329
3113	Uinta, WY	593589.074642
3114	Washakie, WY	609656.529892
3115	Weston, WY	561534.528674

3116 rows × 2 columns

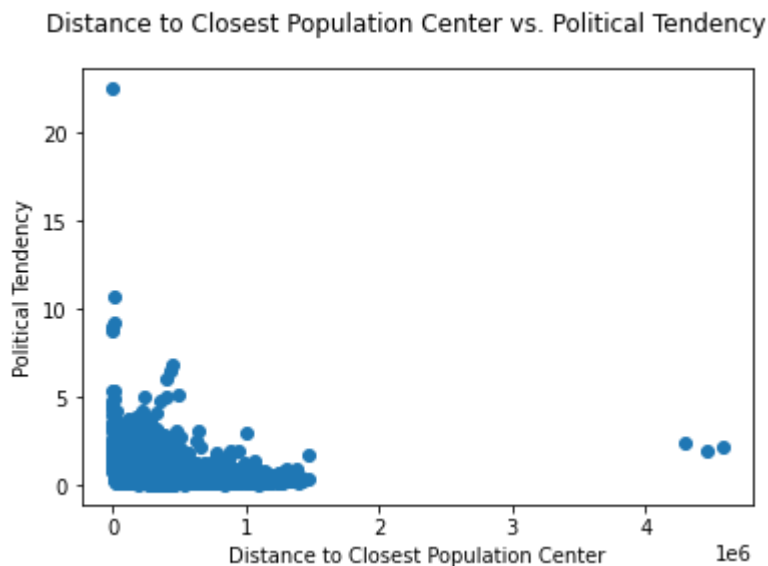
Let's create a scatter to see their relationship.

```
In [127]: import matplotlib.pyplot as plt
fs_lr = fs_gpd.dropna(subset = ["dis_pop_center"])
st = plt.figure()

plt.scatter(x = fs_lr["dis_pop_center"], y = fs_lr["pt_2016"])

st.suptitle("Distance to Closest Population Center vs. Political Tendency")
plt.xlabel("Distance to Closest Population Center")
plt.ylabel("Political Tendency")

plt.show()
```



It seems like there is a negative relationship between distance to closest population centers and the county's support for Democrats. But the results is difficult to interpret. Let's train a simple linear regression model to generate the best fitting line, whose coefficient tells us the correlation between two variables.

```
In [128]: from sklearn.linear_model import LinearRegression
lr = LinearRegression()
lr.fit(X = fs_lr[["dis_pop_center"]], y = fs_lr[["pt_2012"]])

print("Coefficient in 2012 is: " + str(lr.coef_[0][0]))

lr = LinearRegression()
lr.fit(X = fs_lr[["dis_pop_center"]], y = fs_lr[["pt_2016"]])
print("Coefficient in 2016 is: " + str(lr.coef_[0][0]))
```

```
Coefficient in 2012 is: -1.5786130582166985e-07
Coefficient in 2016 is: -3.1936314034989134e-07
```

Or, We can directly calculate the correlation between two columns.


```
In [129]: # Calculate correlation
cor2012 = fs_lr["dis_pop_center"].corr(fs_lr["pt_2012"])
cor2016 = fs_lr["dis_pop_center"].corr(fs_lr["pt_2016"])

print("Correlation between distance and ratio in 2012 is: " + str(cor2012))

print("Correlation between distance and ratio in 2016 is: " + str(cor2016))
```

```
Correlation between distance and ratio in 2012 is: -0.06425005722872416
Correlation between distance and ratio in 2016 is: -0.11766671728735201
```

The results shows the distance to closest population centers is negatively related to the ratio between Democratic and Republican votes in the county.

We believe it makes sense because the population near population centers are tend to be more diverse, and more liberal.

How the population in the county correlated to its political tendency?

We do the same pipeline for population.

```
In [130]: # Original dataset stores population string, we change it to integer
fs_po = fs_gpd.dropna(subset = ["2012_pop", "2016_pop"])
fs_po["2012_pop"] = fs_po["2012_pop"].astype("int")

fs_po["2016_pop"] = fs_po["2016_pop"].astype("int")
```

```
/opt/conda/lib/python3.7/site-packages/geopandas/geodataframe.py:853: SettingWithCopy
Warning:
```

```
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/us
er_guide/indexing.html#returning-a-view-versus-a-copy
super(GeoDataFrame, self).__setitem__(key, value)
```

```

In [131]: # Population
st = plt.figure()
plt.scatter(x = fs_po["2016_pop"], y = fs_po["pt_2016"])

st.suptitle("Population in County vs. Political Tendency")
plt.xlabel("Total Population in the County")
plt.ylabel("Political Tendency")

plt.show()

# Calculate correlation
cor2012 = fs_po["2012_pop"].corr(fs_po["pt_2012"])
cor2016 = fs_po["2016_pop"].corr(fs_po["pt_2016"])

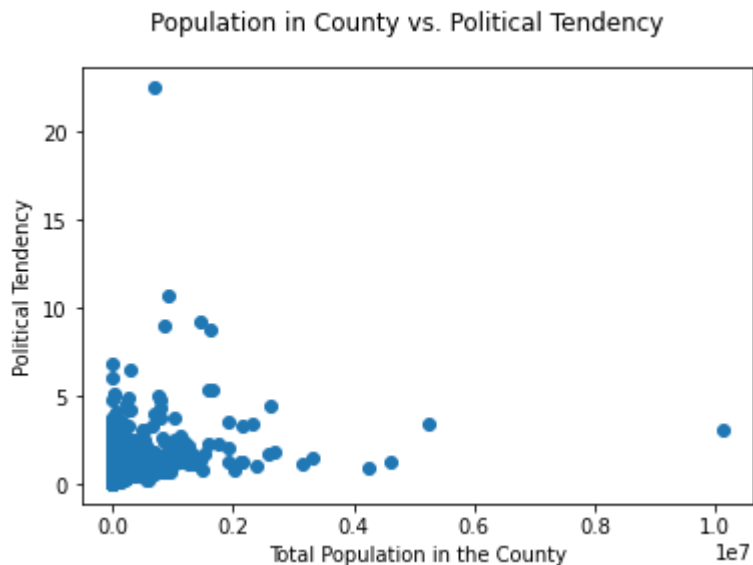
print("Correlation between population and ratio in 2012 is: " + str(cor2012))
print("Correlation between population and ratio in 2016 is: " + str(cor2016))

lr = LinearRegression()
lr.fit(X = fs_po[["2012_pop"]], y = fs_po[["pt_2012"]])

print("Coefficient in 2012 is: " + str(lr.coef_[0][0]))

lr = LinearRegression()
lr.fit(X = fs_po[["2016_pop"]], y = fs_po[["pt_2016"]])
print("Coefficient in 2016 is: " + str(lr.coef_[0][0]))

```



```

Correlation between population and ratio in 2012 is: 0.27342882375759503
Correlation between population and ratio in 2016 is: 0.3377467078654044
Coefficient in 2012 is: 6.214311138018681e-07
Coefficient in 2016 is: 8.202274481062764e-07

```

The relation between population and the support for Democrat candidates is positive.

It makes sense because counties with large population tends to be more diverse, which makes it more liberal. It basically confirms our explanation above.

How the average income in the county correlated to its political tendency?

We use the exact analyze process above.

```
In [133]: # Original dataset stores income string, we change it to integer
fs_in = fs_gpd.dropna(subset = ["2012_income", "2016_income"])
fs_in["2012_income"] = fs_in["2012_income"].astype("int")

fs_in["2016_income"] = fs_in["2016_income"].astype("int")
```

/opt/conda/lib/python3.7/site-packages/geopandas/geodataframe.py:853: 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-view-versus-a-copy
`super(GeoDataFrame, self).__setitem__(key, value)`

```

In [140]: # Income
st = plt.figure()
plt.scatter(x = fs_in["2016_income"], y = fs_in["pt_2016"])

st.suptitle("Average Income in County vs. Political Tendency")
plt.xlabel("Average Income in the County")
plt.ylabel("Political Tendency")

plt.show()

# Calculate correlation
cor2012 = fs_in["2012_income"].corr(fs_in["pt_2012"])
cor2016 = fs_in["2016_income"].corr(fs_in["pt_2016"])

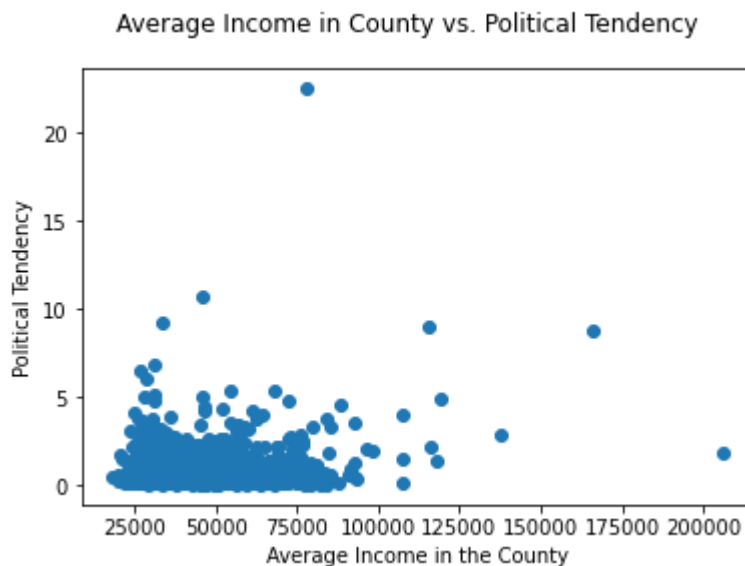
print("Correlation between income and ratio in 2012 is: " + str(cor2012))
print("Correlation between income and ratio in 2016 is: " + str(cor2016))

lr = LinearRegression()
lr.fit(X = fs_in[["2012_income"]], y = fs_in[["pt_2012"]])

print("Coefficient in 2012 is: " + str(lr.coef_[0][0]))

lr = LinearRegression()
lr.fit(X = fs_in[["2016_income"]], y = fs_in[["pt_2016"]])
print("Coefficient in 2016 is: " + str(lr.coef_[0][0]))

```



```

Correlation between income and ratio in 2012 is: 0.07773080400177981
Correlation between income and ratio in 2016 is: 0.24496969777584196
Coefficient in 2012 is: 5.342799917095575e-06
Coefficient in 2016 is: 1.7762164463803172e-05

```

It seems the income per capita in the county is positively related to the ratio between Democrat votes and Republicans votes. It's reasonable because Democrat voters tend to live in urban regions where income per capita is higher than rural regions.

In the map above, counties on coastline are mostly Democrat? Are they related?

As Professor suggested, let's see how is distance to the coastline affects the political tendency of counties.

We define "dis_coast" to be the distance from county to the closest coastline.

```
In [141]: # convert for calculation the distance
coast_gpd = gpd.GeoDataFrame(coast)
coast_gpd = coast_gpd.set_geometry("SHAPE")
coast_gpd.crs = "EPSG:3857"

def coast_dis(row):
    # Return na if no shape, so it will not interfere training process
    if row["SHAPE"] is None:

        return np.nan
    else:
        dis = coast["SHAPE"].distance(row["SHAPE"])
        return min(dis)

# create feature "dis_coast"
fs_gpd["dis_coast"] = fs_gpd.apply(closest_dis, axis = 1)
```

```
In [142]: # Resulting dataset
fs_gpd[["GeoName", "dis_coast"]].head()
```

Out[142]:

	GeoName	dis_coast
0	Autauga, AL	467299.137768
1	Baldwin, AL	566512.048465
2	Barbour, AL	428837.226782
3	Bibb, AL	396462.197238
4	Blount, AL	261723.239062

Analyze their relationship using the process above.

```

In [143]: # Population
fs_coast = fs_gpd.dropna(subset = ["dis_coast"])
st = plt.figure()

plt.scatter(x = fs_coast["dis_coast"], y = fs_coast["pt_2016"])

st.suptitle("Distance to Coastline vs. Political Tendency")
plt.xlabel("Distance to the Closest Coastline")
plt.ylabel("Political Tendency")

plt.show()

# Calculate correlation
cor2012 = fs_coast["dis_coast"].corr(fs_coast["pt_2012"])
cor2016 = fs_coast["dis_coast"].corr(fs_coast["pt_2016"])

print("Correlation between distance to closest coastline and ratio in 2012 is: " + str(
cor2012))
print("Correlation between distance to closest coastline and ratio in 2016 is: " + str(
cor2016))

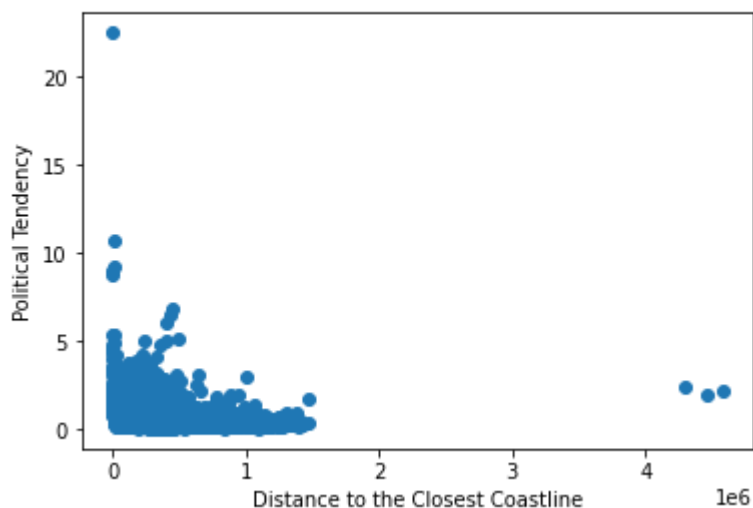
lr = LinearRegression()
lr.fit(X = fs_coast[["dis_coast"]], y = fs_coast[["pt_2012"]])

print("Coefficient in 2012 is: " + str(lr.coef_[0][0]))

lr = LinearRegression()
lr.fit(X = fs_coast[["dis_coast"]], y = fs_coast[["pt_2016"]])
print("Coefficient in 2016 is: " + str(lr.coef_[0][0]))

```

Distance to Coastline vs. Political Tendency



Correlation between distance to closest coastline and ratio in 2012 is: -0.0642500572
2872416

Correlation between distance to closest coastline and ratio in 2016 is: -0.1176667172
8735201

Coefficient in 2012 is: -1.5786130582166985e-07

Coefficient in 2016 is: -3.1936314034989134e-07

The results shows there is a negative relationship between county's distance to the closest coastline and its ratio between votes for Democrats and Republicans. Which confirms our guess above (counties close to coastlines are mostly pro democratic).

We extracted interesting features and analyzed their relationships, now it's time to do prediction!

Predication

We first use 2012 election results to predict 2016 presidential election. We predict the political tendency of county, not which party could win in the election!

We encodes political tendency to be True and False in "label" columns. True indicates more voters votes for Democrats, False indicates more voters votes for Republican.

```
In [419]: # Split training set
train = fs_gpd.dropna()[["GeoName", "2012_pop", "2012_income", "dis_pop_center", "dis_coast", "pt_2012"]]
train["label"] = train["pt_2012"] > 1

test = fs_gpd.dropna()[["GeoName", "2016_pop", "2016_income", "dis_pop_center", "dis_coast", "pt_2016"]]
test["label"] = test["pt_2016"] > 1

train = train.rename(columns = {"2012_pop": "pop", "2012_income": "income"})
test = test.rename(columns = {"2016_pop": "pop", "2016_income": "income"})
```

We have training and testing data. We encode categorical features and use Random Forest Classifier to predict.

```
In [441]: from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier
from sklearn import preprocessing

fs = RandomForestClassifier(n_estimators = 500)

# Encodes features
le = preprocessing.LabelEncoder()
le.fit(train["GeoName"])
train["GeoName"] = le.transform(train["GeoName"])

le.fit(test["GeoName"])
test["GeoName"] = le.transform(test["GeoName"])
fs.fit(X = train.drop(["pt_2012", "label"], axis = 1), y = train["label"])

# Predict
pre = fs.predict(test.drop(["pt_2016", "label"], axis = 1))
```

```
In [457]: # The predictions
pre
```

```
Out[457]: array([False, False, False, ..., False, False, False])
```

We use accuracy to evaluate our prediction, accuracy is suitable for evaluating voting predictions.

```
In [455]: print("Accuracy of prediction: " + str(sum(pre == test["label"]) / len(pre) * 100) +
"%.")
```

```
Accuracy of prediction: 90.30629417704476%.
```

The prediction is quite good! In 90% of the time our prediction for political tendency in county is correct!

Then, we make prediction for 2020 U.S presidential election. But before we do that, we needs to process the 2020 election dataset because it comes from different sources in different formats.


```
In [589]: # Create "GeoName" feature
us_state_abbrev = {
    'Alabama': 'AL',

    'Alaska': 'AK',
    'American Samoa': 'AS',
    'Arizona': 'AZ',
    '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'
}

ele["st"] = ele["state"].apply(lambda x: us_state_abbrev[x])
ele["county"] = ele["county"].str.replace(" County", "")
ele["GeoName"] = ele["county"] + ", " + ele["st"]

ele.sample(5)

```

Out[589]:

	state	county	candidate	party	total_votes	won	st	GeoName
24667	Vermont	Burke	Brooke Paige	GOP	2	False	VT	Burke, VT
29257	Arkansas	Sevier	Don Blankenship	CST	50	False	AR	Sevier, AR
2609	Indiana	Tippecanoe	Jo Jorgensen	LIB	1832	False	IN	Tippecanoe, IN
885	Georgia	Heard	Donald Trump	REP	4519	True	GA	Heard, GA
5334	Louisiana	St. Landry Parish	Joe Biden	DEM	17372	False	LA	St. Landry Parish, LA

In [611]:

```

# split ele based on party and merge it
ele_dem = ele[ele["party"] == "DEM"]
ele_rep = ele[ele["party"] == "REP"]

ele_clean = ele_dem.merge(ele_rep, on = "GeoName", how = "inner")[["GeoName", "total_votes_x", "total_votes_y"]]

ele_clean["label"] = ele_clean["total_votes_x"] > ele_clean["total_votes_y"] # label column, Democrats votes > Republican votes.

# resulting election results
ele_clean.sample(1)

```

Out[611]:

	GeoName	total_votes_x	total_votes_y	label
1651	Bridgewater, MA	7688	6572	True

```
In [615]: # merge election results with features
ele_all = ele_clean.merge(fs_gpd, on = "GeoName", how = "right")
ele_all.head()
```

Out[615]:

	GeoName	total_votes_x	total_votes_y	label	st	county	votes_dem_2016	votes_gop_2016
0	Autauga, AL	7503.0	19838.0	False	AL	Autauga	5908.0	18110.0
1	Baldwin, AL	24578.0	83544.0	False	AL	Baldwin	18409.0	72780.0
2	Barbour, AL	4816.0	5622.0	False	AL	Barbour	4848.0	5431.0
3	Bibb, AL	1986.0	7525.0	False	AL	Bibb	1874.0	6733.0
4	Blount, AL	2640.0	24711.0	False	AL	Blount	2150.0	22808.0

5 rows × 21 columns

Training set: clean Geospatial features and 2012, 2016 results.

```
In [617]: train0 = ele_all.dropna()[["GeoName", "2012_pop", "2012_income", "dis_pop_center", "dis_coast", "pt_2012"]]
train1 = ele_all.dropna()[["GeoName", "2016_pop", "2016_income", "dis_pop_center", "dis_coast", "pt_2016"]]
train = train0.append(train1)

def label(row):
    if pd.isna(row["pt_2012"]):
        return row["pt_2016"] > 1
    else:
        return row["pt_2012"] > 1
train["label"] = train.apply(label, axis = 1)
train = train.drop(["pt_2012", "pt_2016"], axis = 1)

def get_pop(row):
    if pd.isna(row["2016_pop"]):
        return row["2012_pop"]
    else:
        return row["2016_pop"]

def get_income(row):
    if pd.isna(row["2016_income"]):
        return row["2012_income"]
    else:
        return row["2016_income"]

train["income"] = train.apply(get_income, axis = 1)
train["pop"] = train.apply(get_pop, axis = 1)
train = train.drop(["2012_pop", "2012_income", "2016_pop", "2016_income"], axis = 1)

train = train.dropna()
train
```

Out[617]:

	GeoName	dis_pop_center	dis_coast	label	income	pop
0	Autauga, AL	467299.137768	467299.137768	False	35067	54954
1	Baldwin, AL	566512.048465	566512.048465	False	38259	190145
2	Barbour, AL	428837.226782	428837.226782	True	28206	27169
3	Bibb, AL	396462.197238	396462.197238	False	27042	22667
4	Blount, AL	261723.239062	261723.239062	False	29647	57580
...
3111	Sweetwater, WY	376020.556426	376020.556426	False	47486	44222
3112	Teton, WY	783545.085329	783545.085329	True	205843	23234
3113	Uinta, WY	593589.074642	593589.074642	False	37731	20682
3114	Washakie, WY	609656.529892	609656.529892	False	43615	8165
3115	Weston, WY	561534.528674	561534.528674	False	41990	7220

5898 rows × 6 columns

Test set.

```
In [618]: test = ele_all.dropna()[["GeoName", "label", "2019_income", "2019_pop", "dis_pop_center", "dis_coast"]]
test = test.rename(columns = {"2019_income": "income", "2019_pop": "pop"})
test.head()
```

Out[618]:

	GeoName	label	income	pop	dis_pop_center	dis_coast
0	Autauga, AL	False	43917	55869	467299.137768	467299.137768
1	Baldwin, AL	False	47485	223234	566512.048465	566512.048465
2	Barbour, AL	False	35763	24686	428837.226782	428837.226782
3	Bibb, AL	False	31725	22394	396462.197238	396462.197238
4	Blount, AL	False	36412	57826	261723.239062	261723.239062

Predict.

```
In [621]: fs = RandomForestClassifier(n_estimators = 500)

# Encodes features
le = preprocessing.LabelEncoder()
le.fit(train["GeoName"])
train["GeoName"] = le.transform(train["GeoName"])

le.fit(test["GeoName"])
test["GeoName"] = le.transform(test["GeoName"])
fs.fit(X = train.drop(["label"], axis = 1), y = train["label"])

# Predict
pre = fs.predict(test.drop(["label"], axis = 1))
pre
```

Out[621]: array([True, True, False, ..., True, True, True])

```
In [622]: print("Accuracy of prediction: " + str(sum(pre == test["label"]) / len(pre) * 100) +
"%.")
```

Accuracy of prediction: 40.6578501186843%.

Summary .

Those maps and charts for relationship analysis are already shown during the analysis process, we believe it's inappropriate to use the same codes to show those again.

In the relationship analysis section, results are:

- Distance to closest population centers is negatively related to the county's preference to Democratic candidates (measured by ratio of Democratic votes to Republican votes). It implies if a county is further away from a population center, there might be more people in the county support Republican candidate in presidential election. This finding might be valuable for political campaign teams to adjust their strategies accordingly.

```
In [152]: # Results
cor2012 = fs_lr["dis_pop_center"].corr(fs_lr["pt_2012"])
cor2016 = fs_lr["dis_pop_center"].corr(fs_lr["pt_2016"])

print("Correlation between distance to closest population center and preference to Democratic candidates in 2012 is: " + str(cor2012))

print("Correlation between distance to closest population center and preference to Democratic candidates in 2016 is: " + str(cor2016))
```

```
Correlation between distance to closest population center and preference to Democratic candidates in 2012 is: -0.06425005722872416
Correlation between distance to closest population center and preference to Democratic candidates in 2016 is: -0.11766671728735201
```

- Population in the county is positively related to its preference to Democratic candidates. It implies counties with higher population tends to support Democratic candidate more. This information might be useful for political campaign teams to adjust their strategies accordingly.

```
In [153]: # Calculate correlation
cor2012 = fs_po["2012_pop"].corr(fs_po["pt_2012"])
cor2016 = fs_po["2016_pop"].corr(fs_po["pt_2016"])

print("Correlation between population and preference to Democratic candidates in 2012 is: " + str(cor2012))
print("Correlation between population and preference to Democratic candidates in 2016 is: " + str(cor2016))
```

```
Correlation between population and preference to Democratic candidates in 2012 is: 0.27342882375759503
Correlation between population and preference to Democratic candidates in 2016 is: 0.3377467078654044
```

- Average income in the county is positively related to its preference to Democratic candidates. It implies county with higher average income might tend to support Democratic candidates more. This information might be useful for political campaign teams to adjust their strategies accordingly.

```
In [157]: # Calculate correlation
cor2012 = fs_in["2012_income"].corr(fs_in["pt_2012"])
cor2016 = fs_in["2016_income"].corr(fs_in["pt_2016"])

print("Correlation between income and preference to Democratic candidates in 2012 is: "
      + str(cor2012))
print("Correlation between income and preference to Democratic candidates in 2016 is: "
      + str(cor2016))
```

Correlation between income and preference to Democratic candidates in 2012 is: 0.07773080400177981

Correlation between income and preference to Democratic candidates in 2016 is: 0.24496969777584196

- Distance to closest coastline is negatively related to the county's preference to Democratic candidates. It implies county further away from coastlines tends to support Republican candidates more. This finding might be valuable for political campaign teams to adjust their strategies accordingly.

```
In [155]: # Calculate correlation
cor2012 = fs_coast["dis_coast"].corr(fs_coast["pt_2012"])
cor2016 = fs_coast["dis_coast"].corr(fs_coast["pt_2016"])

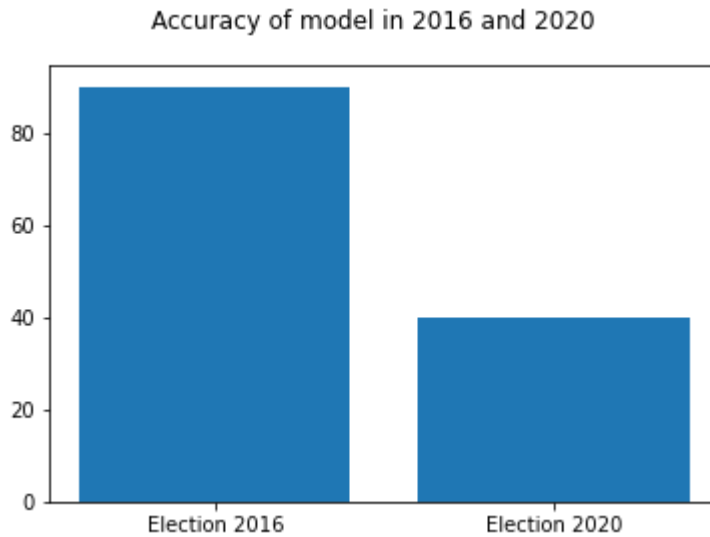
print("Correlation between distance to closest coastline and preference to Democratic c
andidates in 2012 is: " + str(cor2012))
print("Correlation between distance to closest coastline and preference to Democratic c
andidates in 2016 is: " + str(cor2016))
```

Correlation between distance to closest coastline and preference to Democratic candidates in 2012 is: -0.06425005722872416

Correlation between distance to closest coastline and preference to Democratic candidates in 2016 is: -0.11766671728735201

The accuracy of our model for predicting presidential election results in 2016 and 2020:

```
In [167]: plt.bar(x = ["Election 2016", "Election 2020"], height = [90, 40])  
plt.suptitle("Accuracy of model in 2016 and 2020")  
plt.show()
```



- The accuracy of our model for predicting 2016 presidential election result using 2012 election result is 90.3%, which means our election predicting model is quite effective at predicting 2016 election results.
- But the accuracy of our model for predicting 2020 presidential election results is quite low (around 40%). This implies our model requires further development in order to be actually practical.

Discussion

Discussion of results in analysis

Our analysis proved average income and population are positively correlated to the county's support for Democratic candidates. It validates what we found in most articles in our literature sources. But those sources usually only focus on the income while ignoring geospatial features.

Since "Distance to closest population center" and "Distance to closest coastline" are features we extracted during our own analysis process, there is no related research in our literature sources. These are new findings which improves our understand of factors related to county's political affiliation.

Discussion of Inconsistency in Prediction

The inconsistency in prediction accuracy might be caused by inconsistency in dataset we are using. Let's review the election dataset we are using.


```
In [670]: print("There are " + str(ele["GeoName"].nunique()) + " unique counties in 2020 U.S presidential election dataset")

print("There are " + str(former["GeoName"].nunique()) + " unique counties in 2012 and 2016 U.S presidential election dataset")
```

There are 4633 unique counties in 2020 U.S presidential election dataset
 There are 2967 unique counties in 2012 and 2016 U.S presidential election dataset

There is a big difference between number of "counties" in those dataset! Who is correct? It turns out there are 3007 counties in the United States, which means the 2012 and 2016 election dataset is more accurate.

Why the number of counties in 2020 presidential election dataset is more than the number actually exists? It turns out the definition of "county" is different in these two dataset! An example is there are 8 different "counties" for District of Columbia in 2020 election dataset, while there is only one in 2012 and 2016 election dataset.

```
In [168]: # One record for District of Columbia in 2012 and 2016 election dataset.
former.dropna()[former["GeoName"].dropna().str.contains("DC")]
```

Out[168]:

	st	county	votes_dem_2016	votes_gop_2016	votes_dem_2012	votes_gop_2012	
287	DC	District of Columbia	260223.0	11553.0	222332.0	17337.0	[[[-8574 4719

```
In [680]: # Multiple records for District of Columbia in 2020 election dataset.
ele_clean[ele_clean["GeoName"].str.contains("DC")]
```

Out[680]:

	GeoName	total_votes_x	total_votes_y	label
3	District of Columbia, DC	39041	1725	True
4	Ward 2, DC	29078	2918	True
5	Ward 3, DC	39397	3705	True
6	Ward 4, DC	42489	1913	True
7	Ward 5, DC	43320	1769	True
8	Ward 6, DC	56719	4337	True
9	Ward 7, DC	36382	1134	True
10	Ward 8, DC	30897	1085	True

Notice the "District of Columbia, DC", which I believe it means "Ward 1, DC". If we try to GeoEncode using those GeoName, the "District of Columbia, DC" will be encoded to include all regions of all wards, even though it's actually referring to Ward1. If this is common in the dataset, it's not possible to create accurate predictions. The solution to the problem is simply switch to other dataset with consistent records, but at the time we finished the project, we did not find any dataset which contains all information we need and meets the requirement.

Conclusions, and Future Works

In this project, we successfully analyzed how a county's income, population and geospatial features are related to its political tendency. We trained a Random Forest Classifier to predict election results in 2016 and 2020. The accuracy of prediction for 2016 election is quite high (90%) but the accuracy of prediction in 2020 election is really low (40%). We believe our initial questions about relationships are mostly answered, but there is one issue that Alaska is missing in our dataset which is unfair for people in Alaska. The prediction model still required further development. To improve our model, we need dataset with consistent definition of "county". We can switch to other dataset suggested by professor and experts, but we are not sure if those are sufficient for our analysis, and it requires a lot of works to use new dataset in our analysis.

Our analysis approach is highly generalizable. One can easily change features we analyzed to other features they are interested in. The prediction model can also be easily modified, for example, one can even use population, political affiliation and geospatial features to predict average income in the given county. It's also possible to do similar analysis using completely different dataset, but the feature extraction process needs to be modified accordingly. Additional dataset might be helpful to improve our current analysis and prediction model, but it needs to be consistent to those we are currently using.

The results might be valuable for political campaign teams to evaluate and adjust strategies in different counties. Academic readers, including political scientists, might get inspirations from our analysis.