Our Objective

The confirmation of Brett Kavanaugh to the Supreme Court last year resurrected an old problem about the US constitution. Many observers had noted that the 50 senators who voted to confirm Kavanaugh represented only 45 percent of the population. Many historians pointed out that the Senate represents states and not the people. Even though senators represent the states, they nevertheless make key decisions that impact the entire population. Is the minority making key major decisions?

We realized that this could be a great data analysis problem and decided to study population growth and analysis of two key decisions made by the senate. We first summarized the issues and then came up with hypotheses / questions that could be tested through data analysis.

Summary of the Issue:

1. Each state in the US has 2 senators, this creates a unique situation in terms of people's voice
2. Wyoming with population of 575K has 2 senators and California with 40M has 2 senators
3. Senate also makes several major decisions that in many cases represent the voice of only a minority of population
4. The project will analyze a couple of key decisions and highlight what % of the total population or % of the population that voted was responsible for the decision.
5. Decisions analyzed: President Impeachment, Tax Cuts and Jobs Act

Our hypotheses:

1. Does the minority of the population have notable influence on the senate?
2. For close votes like 53-47, do the minority of voters make the major decisions?

Our datasets and Sanity checks

It is claimed that almost 80% of the time is spent in data preparation. This amount of time can be significantly reduced by conducting some basic sanity checks for the data. Many times due to time pressure or just negligence, we overlook performing basic sanity checks on the dataset we are working on. Overlooking data accuracy at initial stages of the analysis can prove very costly and therefore it pays off to invest some time and conduct some basic sanity checks before embarking on the data preparation step. Data preparation and cleansing is critical but the effort can be significantly reduced by knowing what kind of data that we have to work with.

Our datasets and some of the sanity checks are included below:

File 1: Senate elections data from 1918 through today

https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/PEJ5QU

The senate election data looks reasonably clean. It has 18 columns and 3421 rows. The columns are the year the senate race was conducted, the state code, state name, senators party affiliation, votes won by senators and the total number of votes.

One issue that we found with this dataset is that there is no key to identify each record and also there is no easy way to join this with the voting records dataset.

Following are some of the sanity checks conducted by us:

```
senators_data = pd.read_csv("senators.csv",encoding= 'unicode_escape')
population_2010_dump = pd.read_csv("population_2010.csv",encoding= 'unicode_escape')
```

```
senators_data.head()
```

	year	state	state_po	state_fips	state_cen	state_ic	office	district	stage	special	candidate	party	writein	mode	candidatevotes	totalvotes
0	1976	Arizona	AZ	4	86	61	US Senate	statewide	gen	False	Sam Steiger	republican	False	total	321236	741210
1	1976	Arizona	AZ	4	86	61	US Senate	statewide	gen	False	Wm. Mathews Feighan	independent	False	total	1565	741210
2	1976	Arizona	AZ	4	86	61	US Senate	statewide	gen	False	Dennis DeConcini	democrat	False	total	400334	741210
3	1976	Arizona	AZ	4	86	61	US Senate	statewide	gen	False	Allan Norwitz	libertarian	False	total	7310	741210
4	1976	Arizona	AZ	4	86	61	US Senate	statewide	gen	False	Bob Field	independent	False	total	10765	741210

```
senators_data.shape
```

```
(3421, 18)
```

```
senators_data.columns
```

```
Index(['year', 'state', 'state_po', 'state_fips', 'state_cen', 'state_ic',
       'office', 'district', 'stage', 'special', 'candidate', 'party',
       'writein', 'mode', 'candidatevotes', 'totalvotes', 'unofficial',
       'version'],
      dtype='object')
```

```
senators_data.describe()
```

	year	state_fips	state_cen	state_ic	candidatevotes	totalvotes	version
count	3,421.00	3,421.00	3,421.00	3,421.00	3,421.00	3,421.00	3,420.00
mean	1,998.86	29.04	52.91	38.99	405,230.36	2,179,958.21	20,171,859.84
std	12.67	15.43	26.04	22.74	772,209.82	2,114,723.76	3,936.51
min	1,976.00	1.00	11.00	1.00	1.00	1.00	20,171,011.00
25%	1,988.00	17.00	33.00	21.00	4,745.00	640,702.00	20,171,011.00
50%	2,000.00	29.00	54.00	41.00	57,382.00	1,541,036.00	20,171,011.00
75%	2,010.00	41.00	74.00	56.00	476,604.00	2,802,641.00	20,171,011.00
max	2,018.00	56.00	95.00	82.00	7,864,624.00	12,578,511.00	20,190,110.00

```
senators_data.state.value_counts()

vars = {'state' : 'nunique',
    'candidate' : 'nunique',
    'candidatevotes' : 'sum'}

grp = senators_data.groupby(['year'])
vrb_stats = grp.agg(vars)
vrb_stats
```

year	state	candidate	candidatevotes
1976	33	132	59173007
1978	33	116	30674857
1980	33	120	59169658
1982	33	114	51589995
1984	33	106	45462266
1986	34	103	48511005
1988	33	109	67424244
1990	35	85	35027371
1992	34	144	83636606
1994	34	137	59753773
1996	33	143	50068865

File 2: US Census data 2020

https://www.census.gov/2010census/data/

The population data was taken from the official US census records and was pretty clean. The 2010 dataset has 57 columns and 151 rows. There were columns that were estimates of future years through

2019. This was very helpful for our analysis as we could use these numbers as the next census is during the current (2020) year.

Following are some of the sanity checks conducted by us:

```
population_2010_dump.head()
```

	SUMLEV	REGION	DIVISION	STATE	NAME	CENSUS2010POP	ESTIMATESBASE2010	POPESTIMATE2010	POPESTIMATE2011	POPESTIMATE2012	...	RI
0	10	0	0	0	United States	308745538	308758105	309321666	311556874	313830990	...	
1	20	1	0	0	Northeast Region	55317240	55318443	55380134	55604223	55775216	...	
2	20	2	0	0	Midwest Region	66927001	66929725	66974416	67157800	67336743	...	
3	20	3	0	0	South Region	114555744	114563030	114866680	116006522	117241208	...	
4	20	4	0	0	West Region	71945553	71946907	72100436	72788329	73477823	...	

```
population_2010_dump.shape
```

```
(57, 151)
```

```
population_2010_dump.columns
```

```
Index(['SUMLEV', 'REGION', 'DIVISION', 'STATE', 'NAME', 'CENSUS2010POP',
       'ESTIMATESBASE2010', 'POPESTIMATE2010', 'POPESTIMATE2011',
       'POPESTIMATE2012',
       ...
       'RDOMESTICMIG2019', 'RNETMIG2011', 'RNETMIG2012', 'RNETMIG2013',
       'RNETMIG2014', 'RNETMIG2015', 'RNETMIG2016', 'RNETMIG2017',
       'RNETMIG2018', 'RNETMIG2019'],
      dtype='object', length=151)
```

```
population_2010_dump.describe()
```

	SUMLEV	STATE	CENSUS2010POP	ESTIMATESBASE2010	POPESTIMATE2010	POPESTIMATE2011	POPESTIMATE2012	POPESTIMATE2013	POPESTIMATI
count	57.00	57.00	57.00	57.00	57.00	57.00	57.00	57.00	
mean	38.07	27.18	16,315,129.88	16,315,797.75	16,345,377.60	16,462,269.37	16,581,183.47	16,694,284.60	16,814,6
std	6.39	18.13	44,360,750.28	44,362,600.08	44,445,861.28	44,776,538.31	45,114,843.64	45,435,942.46	45,780,7
min	10.00	0.00	563,626.00	563,775.00	564,487.00	567,299.00	576,305.00	582,122.00	582,5
25%	40.00	12.00	1,852,994.00	1,853,018.00	1,854,239.00	1,856,301.00	1,856,872.00	1,865,279.00	1,879,3
50%	40.00	27.00	4,625,364.00	4,625,366.00	4,635,649.00	4,671,994.00	4,717,354.00	4,764,080.00	4,823,6
75%	40.00	41.00	9,535,483.00	9,535,751.00	9,574,323.00	9,657,592.00	9,749,476.00	9,843,336.00	9,929,8
max	40.00	72.00	308,745,538.00	308,758,105.00	309,321,666.00	311,556,874.00	313,830,990.00	315,993,715.00	318,301,0

```
vars = {'CENSUS2010POP' : 'sum'}

grp = population_2010_dump.groupby(['NAME'])
vrb_stats = grp.agg(vars)
vrb_stats
```

NAME	CENSUS2010POP
Alabama	4779736
Alaska	710231
Arizona	6392017
Arkansas	2915918
California	37253956
Colorado	5029196
Connecticut	3574097
Delaware	897934

File 3: Voting records for senate:

https://www.govtrack.us/congress/votes#session=287&chamber[]=1

The site govtrack.us stores all the bills taken up by the congress. It stores the voting records for both the chambers - house of representative and senate. We analyzed two such decisions:

1. Senate vote #323 - Tax Cuts & Job Act (2017-12-20)
2. Senate vote #34 - Guilty or Not Guilty (2020-02-05)

The data in these datasets was very clean as it contained only 100 records for each vote - 2 senators for 50 states. We focused on just the three columns - state, vote (yea or nay | Guilty or Not Guilty) and the name of the senator.

One key issue with this file was that there was no easy way to join this file with the senators election file except on name. We used the same algorithm to bring the names into a unique identifier. Our logic worked but there were few exceptions that we handled separately.

One real life example was Senator Jeff Sessions. He was in the election file but was not in the impeachment voting file. This happened because Jeff Sessions became the Attorney General and left his seat from Alabama vacant. A special election was held in 2017 that was won by Doug Jones.

Following are some of the sanity checks conducted by us:

```
vote_data_impeach= pd.read_csv("congress_votes_116-2020_s34.csv", skiprows=1).drop(['person','district','party' ],axis
vote_data_impeach
vote_data_impeach.sort_values(['state']).head(50)
```

	state	vote	name
14	AK	Not Guilty	Sen. Lisa Murkowski [R]
81	AK	Not Guilty	Sen. Dan Sullivan [R]
19	AL	Not Guilty	Sen. Richard Shelby [R]
92	AL	Guilty	Sen. Doug Jones [D]
25	AR	Not Guilty	Sen. John Boozman [R]
69	AR	Not Guilty	Sen. Tom Cotton [R]
70	AZ	Guilty	Sen. Kyrsten Sinema [D]
80	AZ	Not Guilty	Sen. Martha McSally [R]
87	CA	Guilty	Sen. Kamala Harris [D]
8	CA	Guilty	Sen. Dianne Feinstein [D]
58	CO	Not Guilty	Sen. Cory Gardner [R]

```
vote_data_impeach.shape
```

```
(100, 3)
```

```
vote_data_impeach.columns
```

```
Index(['state', 'vote', 'name'], dtype='object')
```

```
vote_data_impeach.describe()
```

	state	vote	name
count	100	100	100
unique	50	2	100
top	NY	Not Guilty	Sen. Joni Ernst [R]
freq	2	53	1

```
vote_data_impeach.groupby(['state']).agg(['count'])
```

	vote	name
	count	count
state		
AK	2	2
AL	2	2
AR	2	2
AZ	2	2
CA	2	2
CO	2	2
CT	2	2
DE	2	2

Data cleansing and aggregation

As mentioned above, it is claimed that almost 80% of the time is spent in data preparation. Below is a summary of our approach for the same on this project.

Step 1: Clean population dataset

The census population dataset that was used for this evaluation has 57 rows and 151 columns. The rows represent the states and zones while the columns are population estimates over time and additional data around the change over time. To cleanse this population dataset, appropriate columns were selected, the ones that did not apply towards the evaluation being done were removed and some of the column labels renamed to be more descriptive. Only the population data for the year that applies to the year of voting was kept. This reduced the dataset from a 57 x 151 array to a 52 x 2 array. Some highlights are listed below.

```
pop_data = population_2010_dump.loc[:,['NAME','POPESTIMATE2017']].iloc[5:,:].sort_values('POPESTIMATE2017')
pop_data = pop_data.rename(columns={'NAME':'state','POPESTIMATE2017':'population_2017'})
fig =  plt.figure(figsize=(20,10))
ax = fig.add_subplot(1,1,1)
ax.bar(pop_data['state'], pop_data['population_2017'])
ax.set_xticklabels(pop_data['state'],rotation = 90)
ax.set_ylabel("Population in 2017")
fig.suptitle("Population Estimate by state", fontsize=20)
pass
```

```
population_2010_dump.shape
```

```
(57, 151)
```

```
pop_data.shape
```

```
(52, 2)
```

The population data for the decade was also reshaped and cleaned to represent the data for states over time. This was part of a secondary evaluation done on only the population dataset to understand the relative change in state populations over time.

```
pop_data_trend = population_2010_dump.drop(['SUMLEV','REGION','DIVISION','STATE'],axis=1)
pop_data_trend = pop_data_trend.loc[:,:'POPESTIMATE2019'].iloc[5:,:]
pop_data_trend = pop_data_trend.set_index('NAME')
pop_data_trend = pop_data_trend.div(pop_data_trend.CENSUS2010POP,axis=0)
pop_years = range(2010,2020)
pop_data_trend = pop_data_trend.T.drop(['ESTIMATESBASE2010','POPESTIMATE2010'],axis=0)
pop_data_trend
```

NAME	Alabama	Alaska	Arizona	Arkansas	California	Colorado	Connecticut	Delaware	District of Columbia	Florida	...	Tennessee	Texas	Utah	V
CENSUS2010POP	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	...	1.00	1.00	1.00	
POPESTIMATE2011	1.00	1.02	1.01	1.01	1.01	1.02	1.00	1.01	1.03	1.01	...	1.01	1.02	1.02	
POPESTIMATE2012	1.01	1.03	1.03	1.01	1.02	1.03	1.01	1.02	1.06	1.03	...	1.02	1.04	1.03	
POPESTIMATE2013	1.01	1.04	1.04	1.01	1.03	1.05	1.01	1.03	1.08	1.04	...	1.02	1.05	1.05	
POPESTIMATE2014	1.01	1.04	1.05	1.02	1.04	1.06	1.01	1.04	1.10	1.06	...	1.03	1.07	1.06	
POPESTIMATE2015	1.02	1.04	1.07	1.02	1.04	1.08	1.00	1.05	1.12	1.07	...	1.04	1.09	1.08	
POPESTIMATE2016	1.02	1.04	1.09	1.03	1.05	1.10	1.00	1.06	1.14	1.10	...	1.05	1.11	1.10	
POPESTIMATE2017	1.02	1.04	1.10	1.03	1.06	1.12	1.00	1.07	1.15	1.12	...	1.06	1.13	1.12	
POPESTIMATE2018	1.02	1.04	1.12	1.03	1.06	1.13	1.00	1.08	1.17	1.13	...	1.07	1.14	1.14	
POPESTIMATE2019	1.03	1.03	1.14	1.03	1.06	1.15	1.00	1.08	1.17	1.14	...	1.08	1.15	1.16	

Step 2: Clean senator election dataset and merge with population dataset, by state

The senator election dataset is a 3421 x 9 array that contains data from 1978 to the last senate election in 2018. For this dataset, some columns not relevant to the evaluation were dropped and some columns renamed. This dataset was then merged with the population dataset on the state column that resulted in a dataset with only relevant variables. The critical step was to have the states labeled consistently between the 2 datasets to get a successful merge. Some highlights are listed below.

```
senators_new = senators_data.loc[:,['year','state','state_po','stage','special','candidate','party','candidatevotes','totalvo
senators = pd.merge(senators_new, pop_data, how='left', on='state')
senators = senators.rename(columns={'candidatevotes':'candidate_votes','totalvotes':'total_votes'}).sort_values(['state','yea
senators.tail()
```

	year	state	state_po	stage	special	candidate	party	candidate_votes	total_votes	population_2017
3047	2014	Wyoming	WY	gen	False	Charlie Hardy	democrat	29377	171153	578931
3417	2018	Wyoming	WY	gen	False	John Barrasso	republican	136210	203420	578931
3418	2018	Wyoming	WY	gen	False	Gary Trauner	democrat	61227	203420	578931
3419	2018	Wyoming	WY	gen	False	Joseph Porambo	libertarian	5658	203420	578931
3420	2018	Wyoming	WY	gen	False	NaN	NaN	325	203420	578931

Finally, one of the most critical steps of our data cleansing was to manipulate the senator names and transform them into unique identifiers between the senator election data and the senator voting records. In our case, we had a few different formats for names, e.g. “Michael R. A. Erdey”, “Sen. Lamar Alexander [R]”, “Sen. Maria Cantwell [D, 2001-2024]”, etc. We decided to create a “**name_identifier”** variable that would be of a “last name first-initial” format. This was accomplished by defining a custom function to do this effectively.

```
def last_name(x):
    """This function takes the name of senators as listed in the data and returns in the lastname_initial format"""
    x_list = x.upper().split()
    return x_list[-1]+" "+x_list[0][0]

senators["name_identifier"] = senators['candidate'].map(lambda x:str(x))
senators["name_identifier"] = senators['name_identifier'].map(last_name)
```

```
senators.head()
```

	year	state	state_po	stage	special	candidate	party	candidate_votes	total_votes	population_2017	name_identifier
144	1978	Alabama	AL	gen	False	Jerome B. Couch	prohibition	34951	582005	4874486	COUCH J
145	1978	Alabama	AL	gen	True	James D. Martin	republican	316170	729842	4874486	MARTIN J
146	1978	Alabama	AL	gen	True	Michael R. A. Erdey	libertarian	6006	729842	4874486	ERDEY M
147	1978	Alabama	AL	gen	True	Donald Stewart	democrat	401852	729842	4874486	STEWART D
148	1978	Alabama	AL	gen	False	Howell Heflin	democrat	547054	582005	4874486	HEFLIN H

The senator election data and census population are common datasets to the 2 evaluations that were done. At this point, this combined dataset was used for 2 different evaluations, one for the voting record on the 2017 Tax Reform and the second for the 2020 Impeachment Vote.

Step 3: Clean senator voting record dataset and merge with state population + senator dataset above

Once again, with the voting record dataset, one of the most critical steps of our data cleansing was to manipulate the senator names and transform them into unique identifiers between the senator election data and the senator voting records.

```
def last_name_vote(x):
    """This function takes the name of senators as listed in the data for voting record and returns in the lastname_initial
    x_list = x.upper().split()
    y = x_list[0:-1]
    if x_list[-1] == '[R]'or x_list[-1] == '[D]':
        return y[-1]+" "+y[1][0]
    else:
        return y[-2]+" "+y[1][0]
```

```
vote_data_tax2017["name_identifier"] = vote_data_tax2017['name'].map(last_name_vote)
vote_data_tax2017 = vote_data_tax2017.rename(columns = {'year':'election_year'}).sort_values('state')
vote_data_tax2017.head()
```

	state	vote	name	name_identifier
17	AK	Yea	Sen. Lisa Murkowski [R]	MURKOWSKI L
89	AK	Yea	Sen. Dan Sullivan [R]	SULLIVAN D
23	AL	Yea	Sen. Richard Shelby [R]	SHELBY R
99	AL	Yea	Sen. Luther Strange [R, 2017-2017]	STRANGE L
28	AR	Yea	Sen. John Boozman [R]	BOOZMAN J

Once we had the common “name_identifier” variable defined and calculated in both the datasets to be merged, the datasets were combined. For our evaluation, the election data for within 6 years of the bill being passed was important to consider, since senators have a 6-year term. Therefore, the appropriate rows were selected to consider only the election results in the right time frame.

```
data_sen_vote = pd.merge(vote_data_tax2017, senators, on='name_identifier')
#Looking at data for 2010-2016
data_sen_vote = data_sen_vote[(data_sen_vote.year>2009) & (data_sen_vote.year<2018)]

data_sen_vote.head()
```

	state_x	vote	name	name_identifier	year	state_y	state_po	stage	special	candidate	party	candidate_votes	total_votes	population_201
1	AK	Yea	Sen. Lisa Murkowski [R]	MURKOWSKI L	2010	Alaska	AK	gen	False	Lisa Murkowski	NaN	101091	255503	73970
2	AK	Yea	Sen. Lisa Murkowski [R]	MURKOWSKI L	2016	Alaska	AK	gen	False	Lisa Murkowski	republican	138149	311441	73970
3	AK	Yea	Sen. Dan Sullivan [R]	SULLIVAN D	2014	Alaska	AK	gen	False	Dan Sullivan	republican	135445	282400	73970
8	AL	Yea	Sen. Richard Shelby [R]	SHELBY R	2010	Alabama	AL	gen	False	Richard C. Shelby	republican	968181	1485499	487448
9	AL	Yea	Sen. Richard Shelby [R]	SHELBY R	2016	Alabama	AL	gen	False	Richard C. Shelby	republican	1335104	2087444	487448

Step 4: Cleaning of the combined dataset

The initial cleansing of the combined dataset involved structuring data in a logical order, renaming variables appropriately and sorting values in a logical manner. One of the more interesting steps was to remove entries that had been duplicated during the merging step, i.e., we expected to get 2 data entries for each state (since there are 2 senators per state), but in some cases we got more than 2 entries since we considered a 6 year period selecting data (because senators have a 6-year turn). Effectively, we needed only the most recent entry for each senator. This was achieved by writing a custom function to remove the extra rows.

```
data_sen_vote.shape
```

```
(141, 12)
```

```
data_sen_vote = data_sen_vote.reset_index()
data_sen_vote.drop('index',axis=1).sort_values('state').head()
```

	state	vote	name_identifier	year	state_name	state_po	candidate	party	candidate_votes	total_votes	population_2017
0	AK	Yea	MURKOWSKI L	2010	Alaska	AK	Lisa Murkowski	NaN	101091	255503	739700
1	AK	Yea	MURKOWSKI L	2016	Alaska	AK	Lisa Murkowski	republican	138149	311441	739700
2	AK	Yea	SULLIVAN D	2014	Alaska	AK	Dan Sullivan	republican	135445	282400	739700
3	AL	Yea	SHELBY R	2010	Alabama	AL	Richard C. Shelby	republican	968181	1485499	4874486
4	AL	Yea	SHELBY R	2016	Alabama	AL	Richard C. Shelby	republican	1335104	2087444	4874486

```
def delete_duplicates(df):
    delete_list = []
    for i in range(0,len(list(df.index))-1):
        if df['name_identifier'][i] == df['name_identifier'][i+1] and df['state'][i] == df['state'][i+1]:
            delete_list.append(i)
    new_df = df.drop(delete_list).drop(['index'],axis=1).reset_index()
    return new_df.drop('index',axis=1)
```

```
d1 = delete_duplicates(data_sen_vote)
d1.sort_values('state').head()
```

	state	vote	name_identifier	year	state_name	state_po	candidate	party	candidate_votes	total_votes	population_2017
0	AK	Yea	MURKOWSKI L	2016	Alaska	AK	Lisa Murkowski	republican	138149	311441	739700
1	AK	Yea	SULLIVAN D	2014	Alaska	AK	Dan Sullivan	republican	135445	282400	739700
2	AL	Yea	SHELBY R	2016	Alabama	AL	Richard C. Shelby	republican	1335104	2087444	4874486
3	AR	Yea	BOOZMAN J	2016	Arkansas	AR	John Boozman	republican	661984	1107522	3001345
4	AR	Yea	COTTON T	2014	Arkansas	AR	Tom Cotton	republican	478819	847505	3001345

```
d1.shape
```

```
(99, 11)
```

Step 5: Aggregation of the final dataset to generate insights

The final steps in modifying the data included a few different ‘groupby’ and ‘aggregation’ operations to get the date in a form that would be more insightful and easier to make deductions from. Additionally some new calculated variables had to be included to investigate our hypothesis further. One such example is listed below.

To understand the equivalent population numbers corresponding to the senate votes, the data had to be grouped by state and vote type (Yea vs Nay), aggregate functions were applied to summarize the data, the resulting table unstacked to get a better representation and finally the table transposed to make it easy to follow.

```
d3.head()
```

	candidate	state	state_name	vote	party	candidate_votes	total_votes	year	population_2017	percentage_of_people_voted
0	Lisa Murkowski	AK	Alaska	Yea	republican	138149	311441	2016	739700	42.10
1	Dan Sullivan	AK	Alaska	Yea	republican	135445	282400	2014	739700	38.18
2	Richard C. Shelby	AL	Alabama	Yea	republican	1335104	2087444	2016	4874486	42.82
3	Richard C. Shelby	AL	Alabama	Yea	republican	1335104	2087444	2016	4874486	42.82
4	John Boozman	AR	Arkansas	Yea	republican	661984	1107522	2016	3001345	36.90

```
g1 = d3.groupby(['vote','state'])
d6 = g1['candidate_votes'].agg(max).unstack().T.fillna(0)
d6.head()
```

vote	Nay	Not Voting	Yea
state			
AK	0.00	0.00	138,149.00
AL	0.00	0.00	1,335,104.00
AR	0.00	0.00	661,984.00
AZ	0.00	1,359,267.00	1,104,457.00
CA	7,864,624.00	0.00	0.00

```
g2 = d3.groupby(['state'])
d7 = g2[['population_2017']].agg(max)
d7.head()
```

state	population_2017
AK	739700
AL	4874486
AR	3001345
AZ	7044008
CA	39358497

```
d8 = pd.merge(d6,d7,on='state')
d8['not_heard'] = d8['population_2017']-d8['Yea']-d8['Nay']-d8['Not Voting']
d8 = d8.sort_values('population_2017',ascending=False)
d9 = pd.DataFrame(d8.apply(sum))
```

```
d8.head(20)
```

state	Nay	Not Voting	Yea	population_2017	not_heard
CA	7,884,624.00	0.00	0.00	39358497	31,493,873.00
TX	0.00	0.00	4,440,137.00	28295273	23,855,136.00
FL	4,523,451.00	0.00	4,835,191.00	20963613	11,604,971.00
NY	4,420,043.00	0.00	0.00	19589572	15,169,529.00
PA	3,021,364.00	0.00	2,951,702.00	12787641	6,814,575.00
IL	3,012,940.00	0.00	0.00	12778828	9,765,888.00
OH	2,762,690.00	0.00	3,118,567.00	11659650	5,778,393.00
GA	0.00	0.00	2,135,806.00	10410330	8,274,524.00
NC	0.00	0.00	2,395,376.00	10268233	7,872,857.00
MI	2,735,826.00	0.00	0.00	9973114	7,237,288.00
NJ	1,987,680.00	0.00	0.00	8885525	6,897,845.00
VA	2,010,067.00	0.00	0.00	8463587	6,453,520.00
WA	1,913,979.00	0.00	0.00	7423362	5,509,383.00
AZ	0.00	1,359,267.00	1,104,457.00	7044008	4,580,284.00
MA	1,696,346.00	0.00	0.00	6859789	5,163,443.00
TN	0.00	0.00	1,506,443.00	6708799	5,202,356.00
IN	1,281,181.00	0.00	1,423,991.00	6658078	3,952,906.00
MO	1,494,125.00	0.00	1,378,458.00	6106670	3,234,087.00
MD	1,474,028.00	0.00	0.00	6023868	4,549,840.00
WI	1,547,104.00	0.00	1,479,471.00	5790186	2,763,611.00

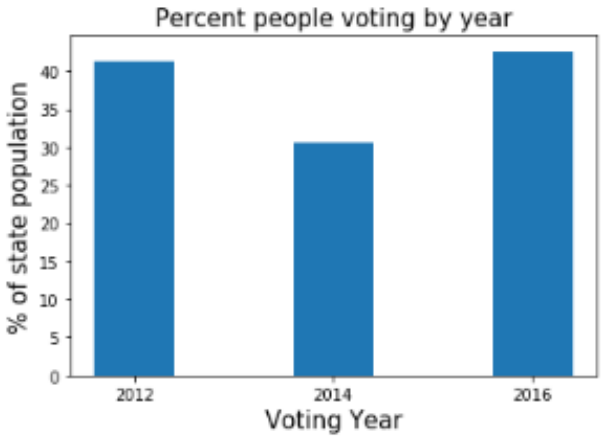
Insights on Data

The results of our analysis were very interesting. Let's consider each of the hypotheses separately and also discuss additional insights we got from the data.

As we began looking at the data and continued to slice and dice it in many different ways, we got some insights in addition to the primary focus of our research. We believe these are interesting and therefore worth mentioning.

Insight: All elections are not equal

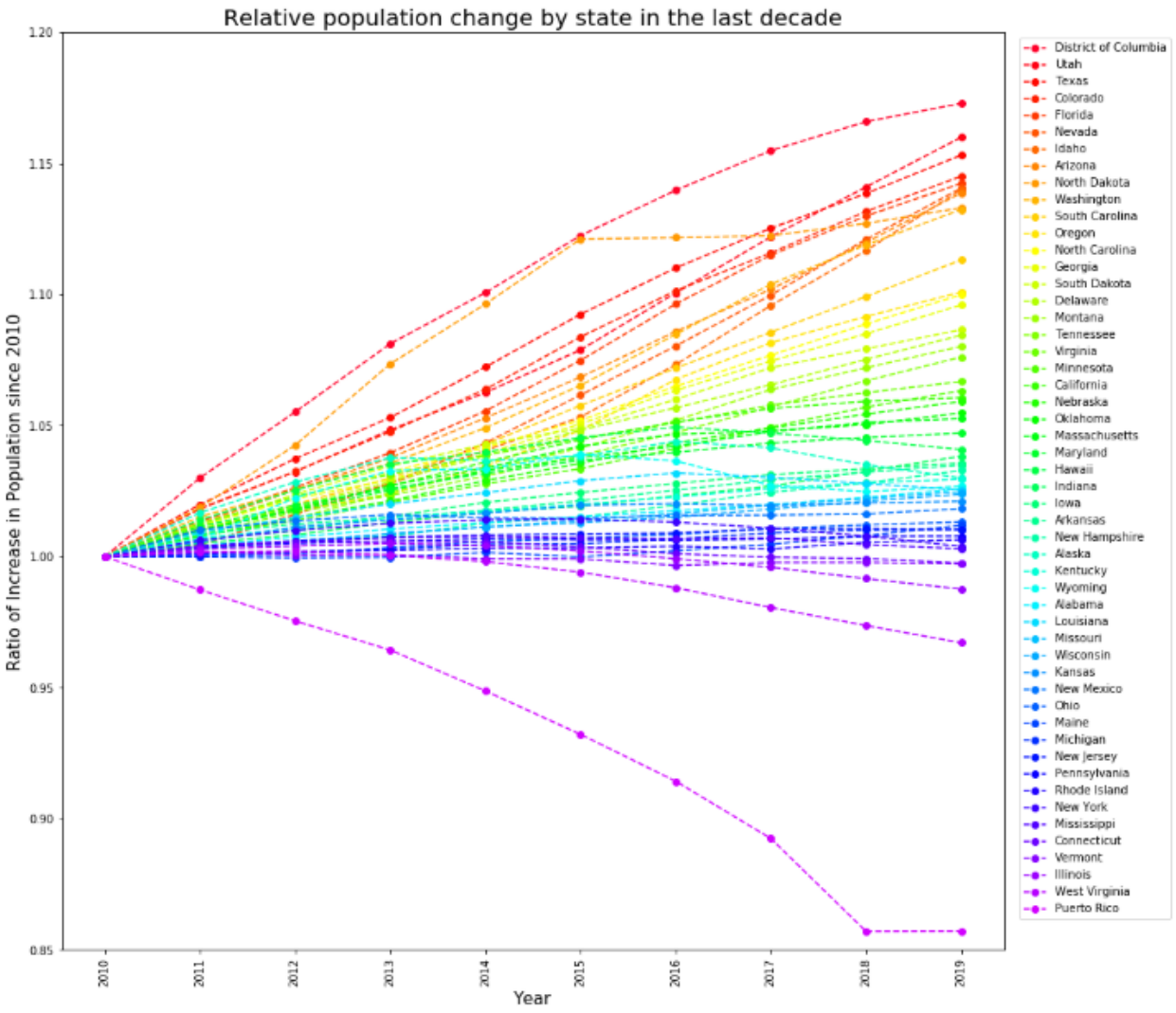
We were curious to see if people treated all the senate elections equally. So we looked at 3 senate elections following up to the senate vote for the Tax Cuts and Jobs Act of 2017. It became clear that the voting rate was much higher for the years that coincided with presidential elections.



Insight: State populations change differently

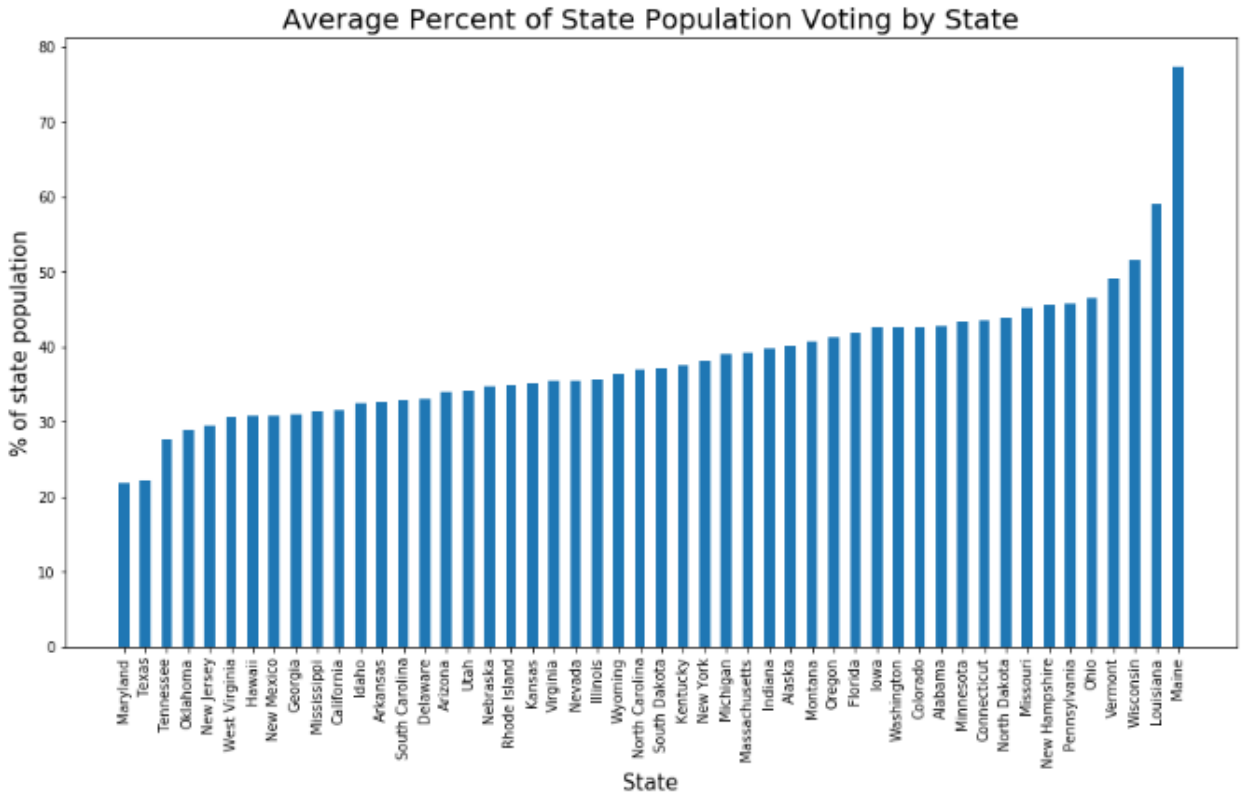
Second, on the population data, it was not enough to simply know that the population of the country increased as a whole over the last decade. We wanted to delve into how each of the states were growing (or maybe contracting?). The graph below plots each of the state's population growth relative to the 2010 census population over time.

While DC, Utah and Texas grew over 15% in population over the last decade, the population of Puerto Rico dropped by ~15%. Note that from the perspective of voting power, since the number of senators per state remains constant, the states with an increase effectively saw a reduction in the voting power per capita while the states with a population decrease saw the power of each vote increase.



Insight: States Vote Differently

Similarly, from the voting data, it became clear that all states don't vote equally. We saw quite a bit of variability in the voting percentages.



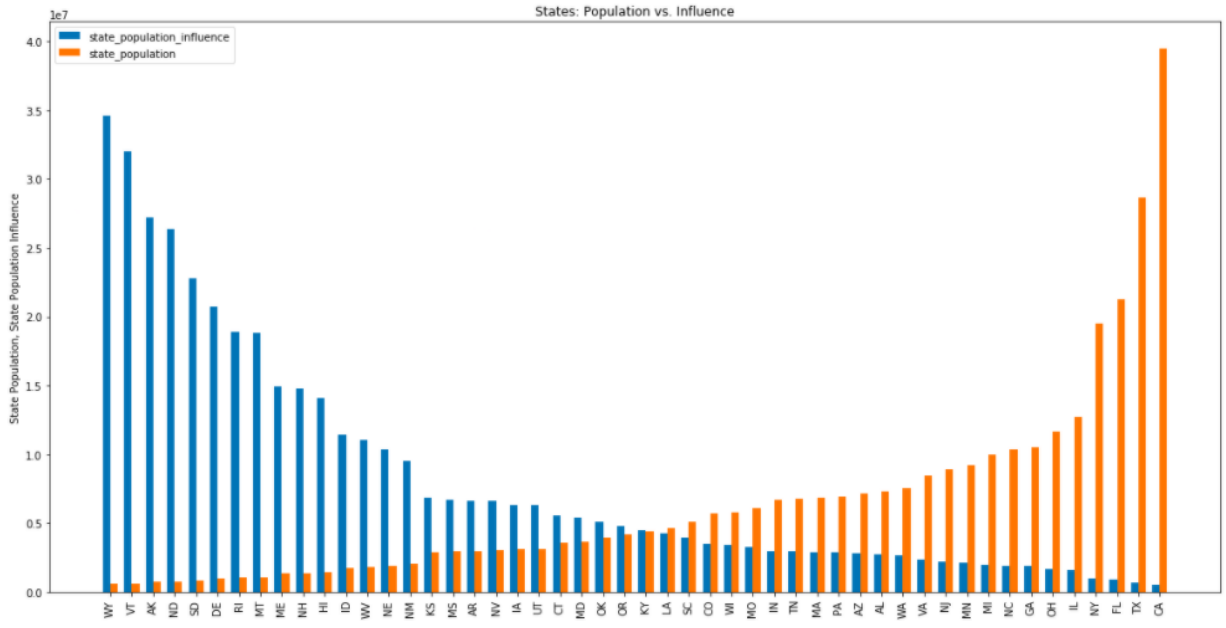
Now we move on to evaluating the research questions that we started off with.

Hypothesis / Question 1: Does the minority of the population have notable influence on the senate?

This hypothesis was not that difficult to prove. We found that the many smaller states have significantly more power over many large states. There are 7 states with only 1 house of representatives but two senators. All of these states have populations of less than or close to 1 million.

State	# of House of Representative	Population
Alaska	1	734002
Delaware	1	982895
Montana	1	1086759
North Dakota	1	761723
South Dakota	1	903027
Vermont	1	628061
Wyoming	1	567025

After several permutations and combinations we came up with a graph to showcase the discrepancy. See the chart below for the Impeachment vote (House Resolution 755). We obtained similar results for the Tax Cuts and Jobs Act of 2017.



The blue bars are showing the influence of each state compared to its population. As you can see in the graph the states with the lowest population have the most influence over the senate and its decision making ability. We used the following algorithm to come up with the formula for states significance:

We wanted to plot both the population and senate numbers on one graph. They are of course ways apart, senators are just 100 while the population of the US is 330M. So to bring the numbers in the same range - below 100, the data is normalized as follows:

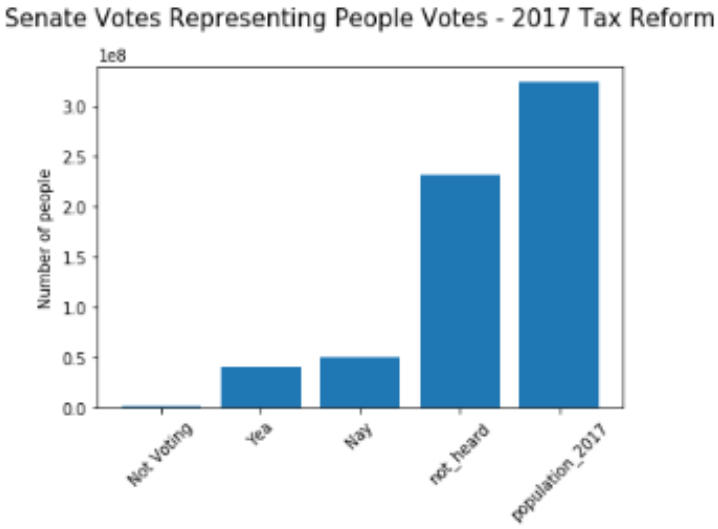
- State Population Influence = *(1/half_of_state_population) * 1000000
- For example Maine has population of ~1.3 M and its influence score is 14.8
- While for California with population of ~39M the influence score is only 0.5

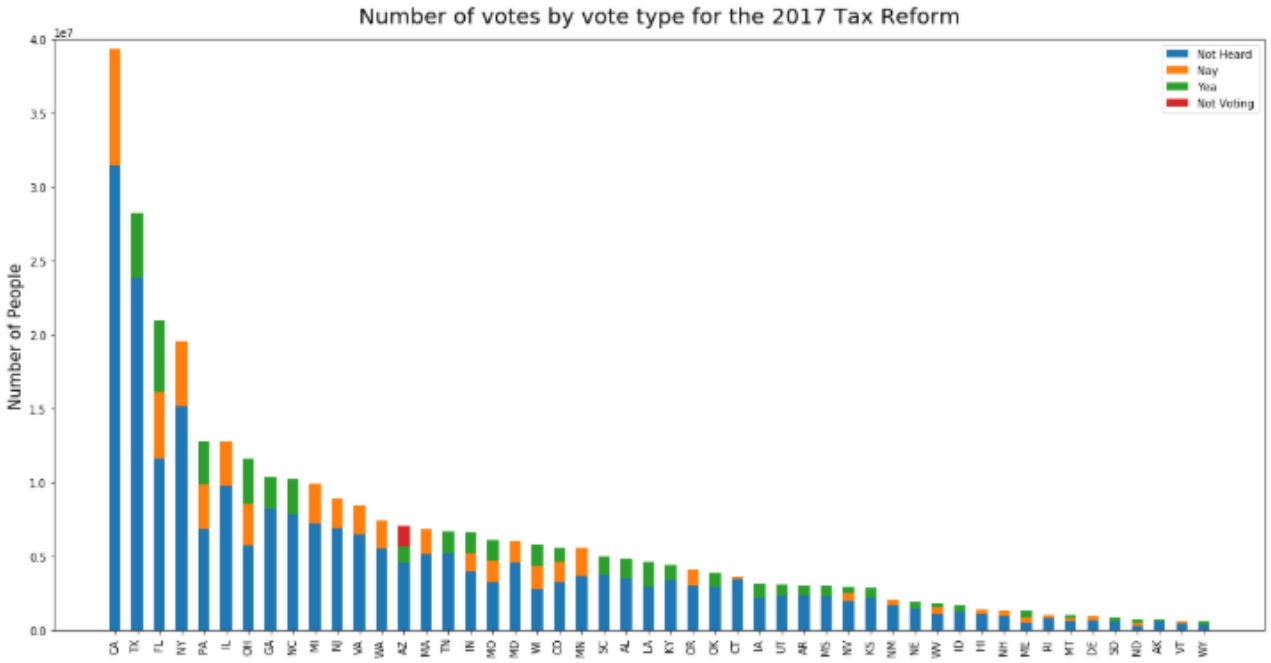
A major insight from the data was that the votes of the senators effectively represented the votes of only a small minority of the state populations as a whole. The rest were either not motivated to vote, or voted for a candidate that did not win.

The first plot shows the combined data for all of the USA. The ‘Yea’ and ‘Nay' populations represent the people who voted for a senator who won and voted ‘Yea’ or ‘Nay’ respectively. The small population corresponding to the ‘Not Voting’ tag represents the folks who voted for Senator McCain who could not vote on this bill due to health issues.All the others, who either voted for a competing candidate or did

not vote at all make up the ‘Not heard’ population. The 2nd plot shows the distribution of these categories by state.

Note that the results below are for the Tax Cuts and Jobs Act of 2017, but we got similar results for the Impeachment vote (House Resolution 755)





The data in the graph above is shown below as a table of numbers. We are showing the first 20 states by population:

state	Nay	Not Voting	Yea	population_2017	not_heard
CA	7,884,624.00	0.00	0.00	39358497	31,493,873.00
TX	0.00	0.00	4,440,137.00	28295273	23,855,136.00
FL	4,523,451.00	0.00	4,835,191.00	20963613	11,604,971.00
NY	4,420,043.00	0.00	0.00	19589572	15,169,529.00
PA	3,021,364.00	0.00	2,951,702.00	12787641	6,814,575.00
IL	3,012,940.00	0.00	0.00	12778828	9,765,888.00
OH	2,762,690.00	0.00	3,118,567.00	11659650	5,778,393.00
GA	0.00	0.00	2,135,806.00	10410330	8,274,524.00
NC	0.00	0.00	2,395,376.00	10268233	7,872,857.00
MI	2,735,826.00	0.00	0.00	9973114	7,237,288.00
NJ	1,987,680.00	0.00	0.00	8885525	6,897,845.00
VA	2,010,067.00	0.00	0.00	8463587	6,453,520.00
WA	1,913,979.00	0.00	0.00	7423362	5,509,383.00
AZ	0.00	1,359,267.00	1,104,457.00	7044008	4,580,284.00
MA	1,696,346.00	0.00	0.00	6859789	5,163,443.00
TN	0.00	0.00	1,506,443.00	6708799	5,202,356.00
IN	1,281,181.00	0.00	1,423,991.00	6658078	3,952,906.00
MO	1,494,125.00	0.00	1,378,458.00	6106870	3,234,087.00
MD	1,474,028.00	0.00	0.00	6023868	4,549,840.00
WI	1,547,104.00	0.00	1,479,471.00	5790186	2,763,611.00

Hypothesis 2: For several close votes like 53-47, the minority of people may be making major decisions

Impeachment vote #34, (House Resolution 755)

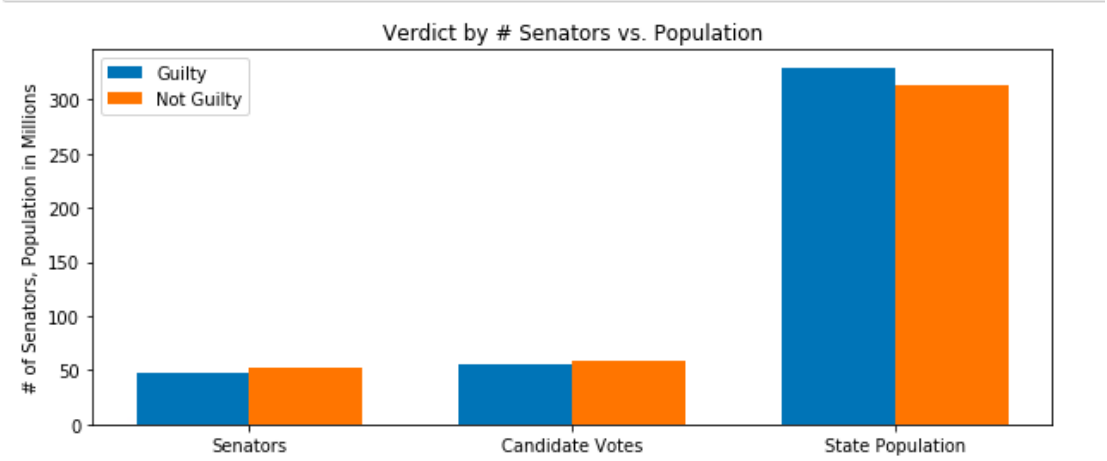
There was a surprise in this decision - given the huge discrepancy between populations of each state and their respective influence, the close votes should have proved the influence of the minority. For example for the the impeachment vote #34 where the President was found not guilty by 53 senators, the 53 states could have represented a minority of voters. The large states like California and New York voted guilty, so we had assumed that the result would be skewed. But the analysis resulted in something surprising.

See the table below:

vote	Guilty	Not Guilty
Senators	47.00	53.00
Candidate Votes	55.22	59.35
State Population	329.84	312.88

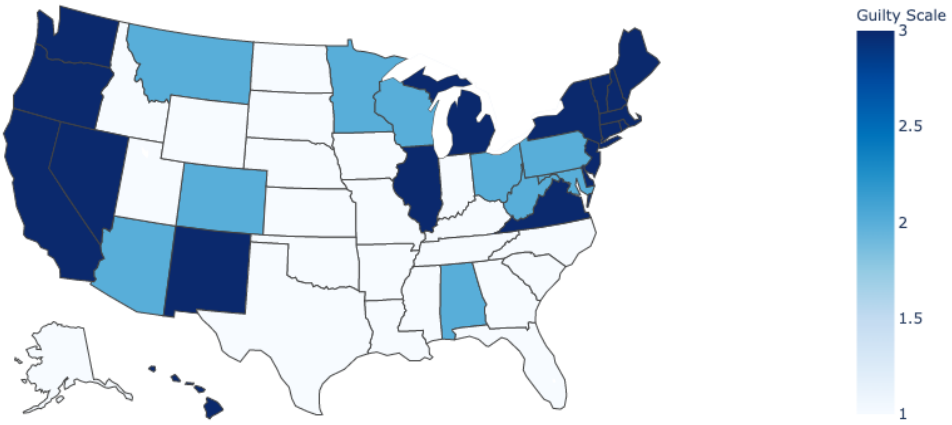
As you can see from the table above the majority of senate votes follow the majority of people who voted for them. The president was found not guilty by 6 votes (53-47) and surprisingly the same number of senators had around 4 million more votes than the ones who found the President guilty. When we investigated this we found that the 53 senators also represent some of the other populous states like Texas and Florida. When we look at the total population the guilty states again flip the decision. (The total population is almost twice the actual population. This is due to the fact that we add votes from the same states twice for the two senators.)

See the same table graphically in the picture below:



For the next analysis we wanted to use choropleth and come up with a guilty scale to analyze the impeachment vote. The graph shows three shades - in the dark states, both the senators voted guilty while in the lightest states both the senators voted not guilty. In the gray states one senator voted guilty while the other voted not guilty.

Guilty or Not: By States



Conclusion

Our first formal data analysis project was a rewarding experience. We picked a topic that is frequently a topic of discussion in bars and parties. Political buffs and armchair constitutional experts have debated on several political results from presidential elections to supreme court nominations. We picked a topic that is recently in the news - the power of 100 senators against a population of 330 million people.

The recent confirmation of Brett Kavanaugh to the Supreme Court last year and the observation by experts that the 50 senators represented only 45 percent of the population, had put many of the recent decisions by the senate on the radar. We had participated in some of the debates. The final project gave us an opportunity to not only debate but also back our claims by data and analysis.

We used a simple and proven methodology and experienced the importance of each step through experience. The following observations summarize our findings during each critical step:

Select a problem and come up with a hypothesis: We found out that before embarking on detailed analysis it helps to come up with a hypothesis. The hypothesis can be based on past results, current observations and availability of datasets. Our hypothesis was based on some of the past senate decisions and the clear discrepancy between the population of various states.

Confirm the feasibility of the effort: Though it is critical to select a problem that needs analysis what we confirmed was that the feasibility of analysis is even more critical. There are thousands of problems that could be analyzed but datasets may not be available for many of them. Our project confirmed that the identification of the datasets ahead of time was critical to successfully complete the project.

Importance of data cleaning: We were excited when we found the two critical datasets for our analysis - the senate election record and senate voting records but when we started using them we realized that just having the necessary dataset is not enough - they must be cleansed and made usable. The two datasets had the names of the senators but we soon realized that we cannot join them by names. One dataset will have Chuck Schumer while the other will have Charle E. Schumer. We have to cleanse and standardize both the datasets and come up with a “key” to join the dataset.

We also found several real life challenges. We found out that senator Kelly Loeffler was not present in the senate election datasets. After investigation we found out that Kelly was an Illinois governor's appointment for Senator Johnny Isakson, who had retired for health reasons!

Analyze the data and validate the hypothesis: A huge blessing with Python was the ability of performing different types of analysis and plotting various kinds of graphs (this can also be a challenge as we were new to so many options, permutations and combinations!)

We were able to prove our hypothesis with multiple types of analysis but at the same time we also found out that sometimes something that seems so obvious is not so. It is easy to be biased about California with its 40 million population and democratic leaning. We were thinking that most of the close senate

decisions will be skewed towards the minority population states like Wyoming but that was not the case. The senate votes were backed up by the corresponding voting. Sure California and New York are big states that are democratic but we cannot forget other large states like Texas and Florida who might be leaning republican!

All in all it was a rewarding experience. There is a saying - “put your money where your mouth is.” We could modify that saying and say - “put your money where your data is!”

Thank you!

Anil Inamdar & Harvi Singh