DTSA5304 Fundementals of Datavisualization

Report prepared by Alec Jeffery

For my final project I have obtained Senatorial voting data for the 116th United States Senate (2019-2021). I will develop a dashboard for comparing voting roll call data by members of the Senate according to specific bills.

Data is obtained from:

source: Lewis, Jeffrey B., Keith Poole, Howard Rosenthal, Adam Boche, Aaron Rudkin, and Luke Sonnet (2022). Voteview: Congressional Roll-Call Votes Database. https://voteview.com/

Elements for grading:

1. A brief recap of your data, goals, and tasks, focusing on those that most directly influence your design
2. Screenshots of and/or a link to your visualization implementation (see below for additional guidance)
3. A summary of the key elements of your design and accompanying justification
4. A discussion of your final evaluation approach, including the procedure, people recruited, and results. Note that, due to the difficulty of recruiting experts, you can use colleagues, friends, classmates, or family to evaluate your designs if experts or others from your target population are unavailable.
5. A synthesis of your findings, including what elements of your approach worked well and what elements you would refine in future iterations.

I will address each of these items throughout this report according to the numbers above.

Background of dataset and goals/tasks (addressing item 1). As mentioned in the opening of this report, the dataset is Senate voting data from the 116th session. This session preceeded the 2020 Presidential election, at a time when the Senate was very divided and bitter partisan fighting was commonplace. My goal is to seek to understand how individuals and parties voted during this session. I will consider it successful if I produce a datatool that helps understand individual and aggregate voting.

Over the next couple of code cells I will deploy various visual inplementations as well as summaries of the cells and justification for the use of the specific visuals. This facilitates items 2 & 3.

```
'''
We will import applicable libraries as well as data for voting roll calls
'''


import numpy as np
import pandas as pd
import altair as alt

# Suppress any warnings:
import warnings
warnings.simplefilter('ignore')

votes = pd.read_csv('S116_votes.csv')
members = pd.read_csv('S116_members.csv')
memberVotes = pd.merge(votes, members, on='icpsr')
memberVotes.head()
```

	congress_x	chamber_x	rollnumber	icpsr	cast_code	prob	congress_y	chamber_y	st
0	116	Senate	1	14226	1	97.6	116	Senate	
1	116	Senate	2	14226	1	100.0	116	Senate	
2	116	Senate	3	14226	1	98.1	116	Senate	
3	116	Senate	4	14226	1	66.1	116	Senate	
4	116	Senate	5	14226	6	66.0	116	Senate	

5 rows × 27 columns

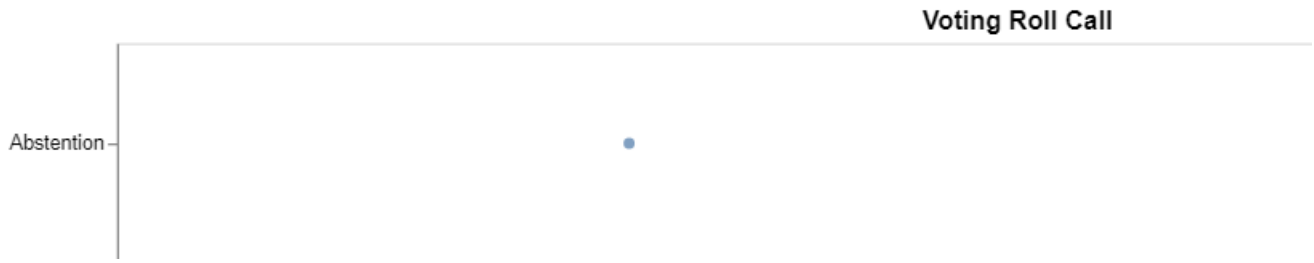
```
list(memberVotes)
```

```
['congress_x',
 'chamber_x',
 'rollnumber',
 'icpsr',
 'cast_code',
 'prob',
 'congress_y',
 'chamber_y',
 'state_icpsr',
 'district_code',
 'state_abbrev',
 'party_code',
 'occupancy',
 'last_means',
 'bioname',
```

```
        'bioguide_id',
        'born',
        'died',
        'nominate_dim1',
        'nominate_dim2',
        'nominate_log_likelihood',
        'nominate_geo_mean_probability',
        'nominate_number_of_votes',
        'nominate_number_of_errors',
        'conditional',
        'nokken_poole_dim1',
        'nokken_poole_dim2']


#%% Data preparation

votes = memberVotes
# Convert Cast Codes to Text
votes.cast_code[votes.cast_code == 1] = 'Yea'
votes.cast_code[votes.cast_code == 6] = 'Nay'
votes.cast_code[votes.cast_code == 7] = 'Present'
votes.cast_code[votes.cast_code == 9] = 'Abstention'
# Convert Party code to Text
votes.party_code[votes.party_code == 100] = 'Dem.'
votes.party_code[votes.party_code == 200] = 'Rep.'
votes.party_code[votes.party_code == 328] = 'Ind.'
```

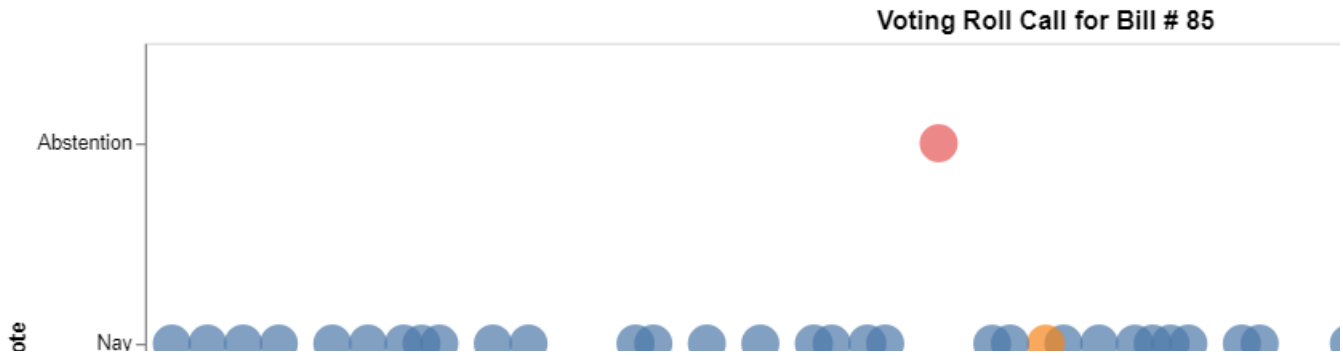
We will begin by simply charting the voting data for one specific bill. This will give us an initial understanding of what the data looks like visually and where we may potentially tweak to garner deeper insight.

```
# Base Chart
alt.Chart(votes[votes.rollnumber == 85]).mark_circle().encode(
    x=alt.X('state_abbrev', axis=alt.Axis(title='Voting State')),
    y=alt.Y('cast_code', axis=alt.Axis(title='Vote')),
).properties(height=300, width=900, title = 'Voting Roll Call')
```



We can see that there is a lot of information to this chart, but not very well structured. We see that votes were cast in three categories (Yea, Nay or Abstention) but we do not know which bill received a specific vote. If we squint our eyes a bit we can see that some of the dots are slightly more dark than others, this being a feature of both senators from a single state voting the same direction. Furthermore, we do not have much differentiation according to individual senators or party in terms of voting patterns. We will therefore seek to differentiate according to these categories.

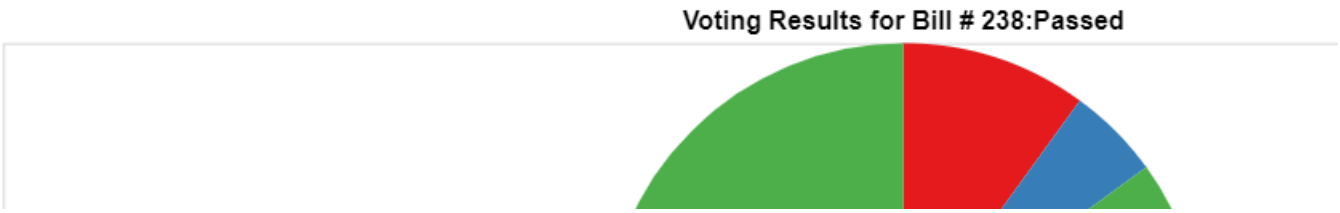
```
# Extended Chart
def makeBillVotePlot(billnumber):
  brush = alt.selection_interval(encodings=['y'])
  return alt.Chart(votes[votes.rollnumber == billnumber]).mark_circle().encode(
      x=alt.X('bioname', axis=alt.Axis(title='Voting State')),
      y=alt.Y('cast_code', axis=alt.Axis(title='Vote')),
      color = alt.Color("party_code", scale = alt.Scale(scheme="tableau10")),
      tooltip = ['bioname', 'party_code'],
      size = 'prob'
  ).properties(height=300, width=900, title = 'Voting Roll Call for Bill # ' + str(billnumber
  ).add_selection(brush)

# Replace the default number in the function to visualize different bills
makeBillVotePlot(85)
```



We now see specific votes per individual senators but we actually do not know if the bill passed or was voted down. The added 'tooltip' function helps us isolate individual voting data as well. We begin seeing how individuals and parties are voting according to a bill.

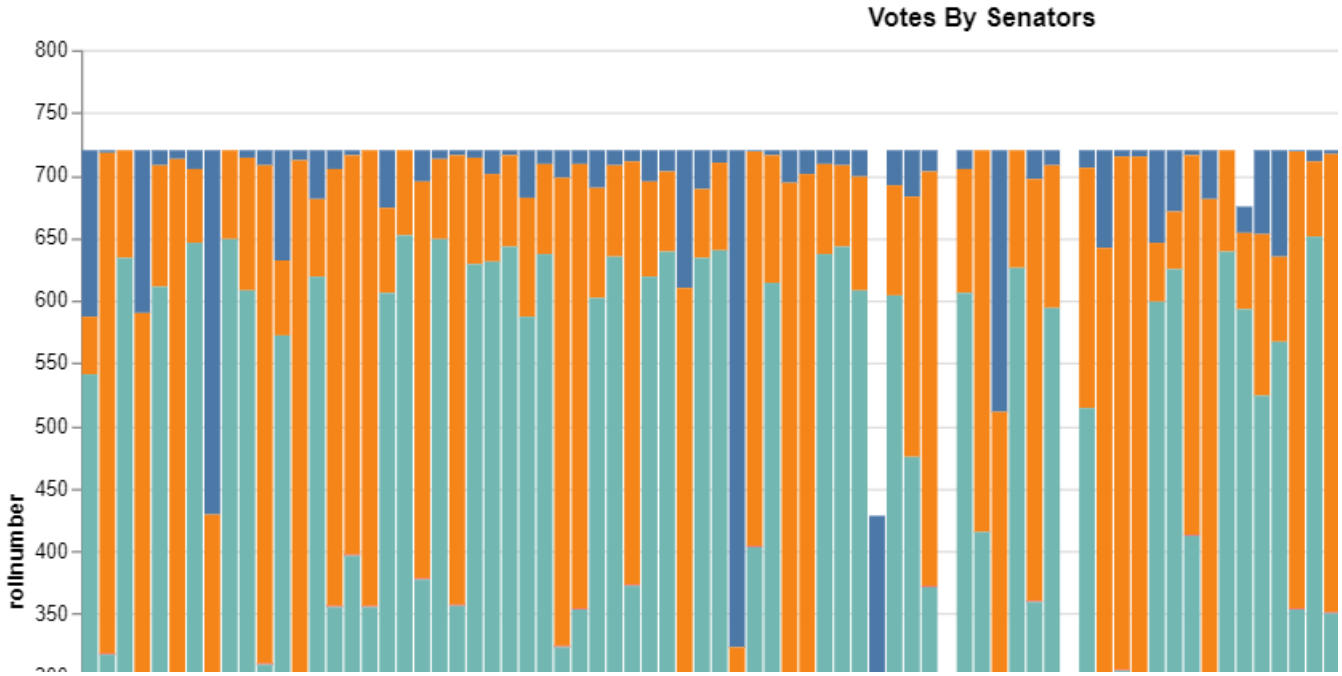
A simple pie chart will be a helpful depiction of the voting tally as well as results. We will compute the voting results and place the outcome in the chart title.

```
def producePie(billnumber):
  tempD = votes[(votes.rollnumber == billnumber)]# & (votes.party_code == 'Rep.')]
  tempD = tempD.groupby(['cast_code']).count()
  tempD= tempD.reset_index()
  # print(tempD[tempD.cast_code == 'Yea'].rollnumber.values[0])
  if tempD[tempD.cast_code == 'Yea'].rollnumber.values[0] >=50:
    result = 'Passed'
  else:
      result = 'Not Passed'
  DemVotes = alt.Chart(tempD).mark_arc().encode(
      theta=alt.Theta(field='rollnumber', type='quantitative'),
      color=alt.Color(field='cast_code', type='nominal', scale=alt.Scale(scheme='set1')),
      tooltip=['cast_code','rollnumber']
  ).properties(height=300, width=900, title = 'Voting Results for Bill # ' + str(billnumber)
  )
  return DemVotes

# You can swap the default number in the function to visualize different bills
producePie(238)
```



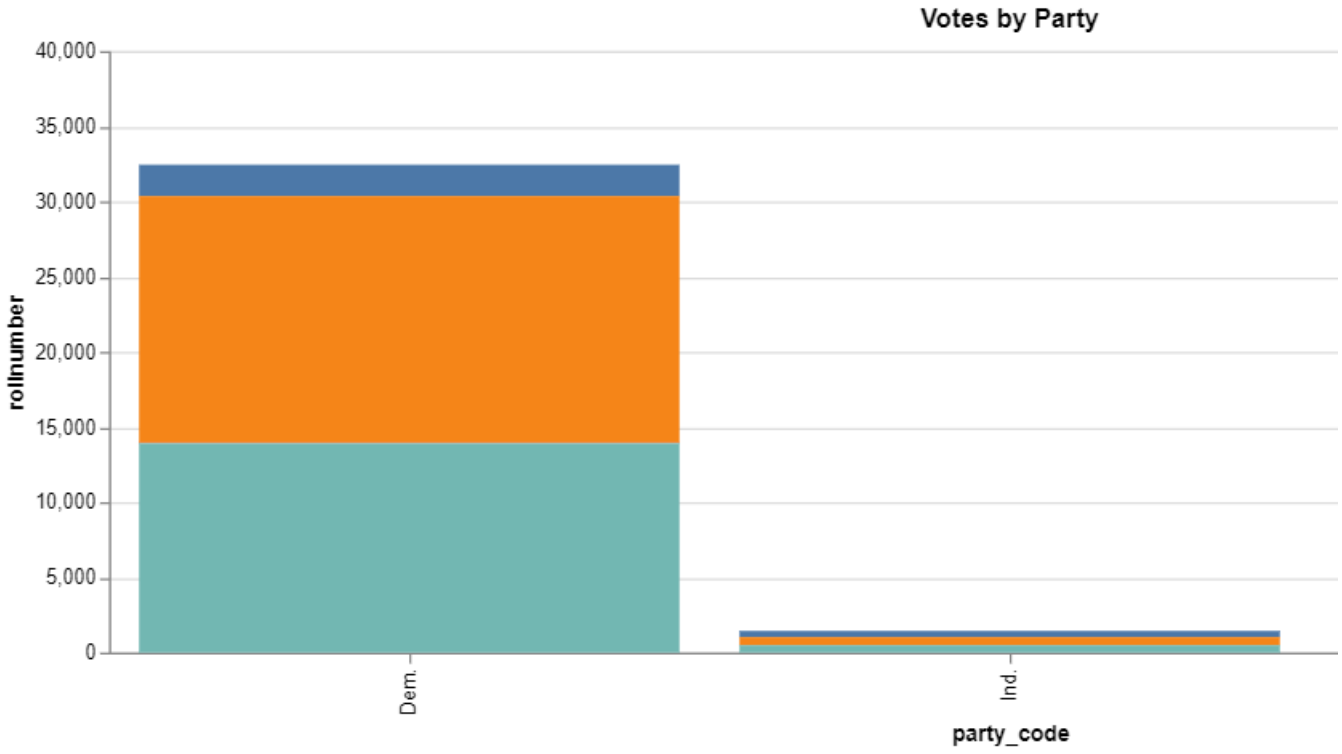
We may now be Curious how Specific parties or Senators have voted on all Bills historically. This information may convey some information about whether the senator is typically in line with the rest of the senate or is more of an obstructionist. The same can be said for the various parties. If one party votes 'Nay' more frequently than 'Yea' the party can be said to be in opposition to the senatorial agenda.

```
# We need to place Voting Data in to tables with tallied results
df = votes.groupby(['bioname','cast_code']).count().reset_index().dropna()
df2 = votes.groupby(['party_code','cast_code']).count().reset_index().dropna()


alt.Chart(df).mark_bar().encode(
    x='bioname',
    y='rollnumber',
    color=alt.Color(field='cast_code',scale=alt.Scale(scheme='tableau10')),
    tooltip=['bioname','cast_code','rollnumber']
).properties(height=500, width=900,title='Votes By Senators')
```



But senators are simply part of the larger party, we wish to know how parties voted historically. Is one party voting in favor of bills more frequently than the other party?

```
alt.Chart(df2).mark_bar().encode(
    x='party_code',
    y='rollnumber',
    color=alt.Color(field='cast_code',scale=alt.Scale(scheme='tableau10')),
    tooltip=['party_code','cast_code','rollnumber']
).properties(height=300, width=900, title = 'Votes by Party')
```



Interestingly, we see that the Republicans voted 'Yea' much more frequently than the Democrats. Compounding this, we see that Democrats voted 'Nay' much more than Republicans. It is save to say that the Democrats are typically in opposition to bills brought forth to the Senate.

Let's take a look at the Senators who votes 'Nay' most frequently in descending order.

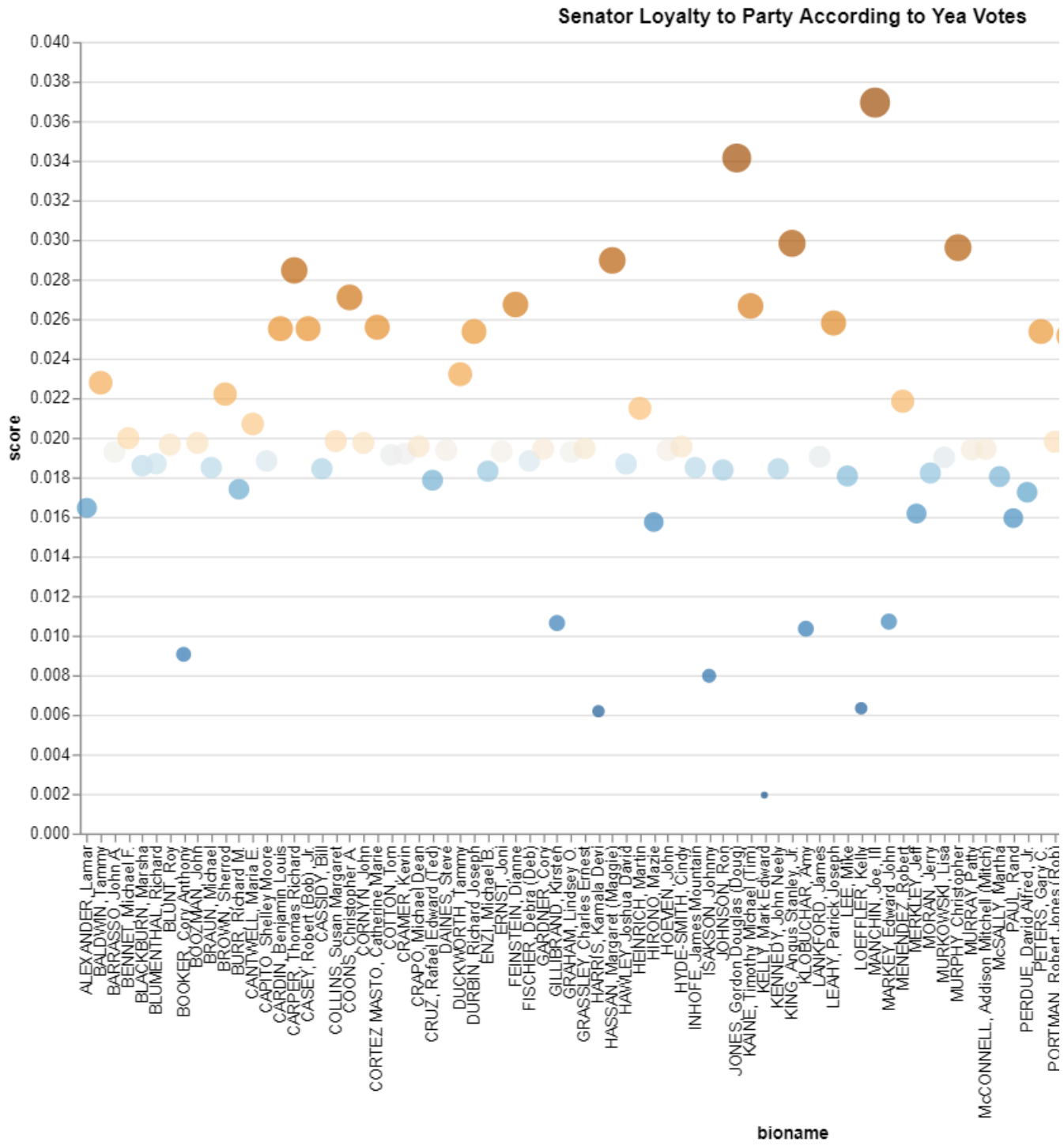
```
ObsSenators = df[df.cast_code == 'Nay'].sort_values(by='rollnumber', ascending=False)
ObsSenators[:15].bioname.values
```

```
array(['MARKEY, Edward John', 'MERKLEY, Jeff', 'HIRONO, Mazie',
       'WYDEN, Ronald Lee', 'SCHUMER, Charles Ellis (Chuck)',
       'GILLIBRAND, Kirsten', 'BLUMENTHAL, Richard',
       'VAN HOLLEN, Christopher', 'CANTWELL, Maria E.',
       'SCHATZ, Brian Emanuel', 'UDALL, Thomas (Tom)', 'SMITH, Tina',
       'MURRAY, Patty', 'MENENDEZ, Robert', 'STABENOW, Deborah Ann'],
      dtype=object)
```

Another way of visualizing party loyalty is to look at how frequently a Senator votes along the party average.

```
RepYes = df2[(df2.cast_code == 'Yea') & (df2.party_code == 'Rep.')].rollnumber.values[0]
DemYes = df2[(df2.cast_code == 'Yea') & (df2.party_code == 'Dem.')].rollnumber.values[0]
IndYes = df2[(df2.cast_code == 'Yea') & (df2.party_code == 'Dem.')].rollnumber.values[0]

SenatorLoyalty = pd.DataFrame()#columns=['bioname','score'])

for i in df.bioname.unique():
  if votes[(votes.bioname == i)].party_code.values[0] == 'Rep.':
    SenatorLoyalty = SenatorLoyalty.append(pd.DataFrame([i, df[(df.bioname == i) & (df.cast_c
  elif votes[(votes.bioname == i)].party_code.values[0] == 'Dem.':
    SenatorLoyalty = SenatorLoyalty.append(pd.DataFrame([i, df[(df.bioname == i) & (df.cast_c
  else:
    SenatorLoyalty = SenatorLoyalty.append(pd.DataFrame([i, df[(df.bioname == i) & (df.cast_c
SenatorLoyalty = SenatorLoyalty.rename(columns={0: "bioname", 1: "score"})
```

```
alt.Chart(SenatorLoyalty).mark_circle().encode(
    x='bioname',
    y='score',
    size='score',
    color=alt.Color(field='score', scale=alt.Scale(scheme='blueorange')),
    tooltip=['bioname','score']
).properties(height=500, width=900,title='Senator Loyalty to Party According to Yea Votes')
```



This paints an interesting vantage of party loyalty. We see that most senators vote according to party lines in terms of 'Yea' votes. We do, however, see some senators vote moreso in line with the party as well as strictly against their party. **Very interestingly, we see 6 of the 10 most 'disloyal' senators actually ran for the Democratic Presidential nominee that year (2020).**

```
SenatorLoyalty = SenatorLoyalty.sort_values(by='score')
print(SenatorLoyalty[:10])
```

```
                 bioname     score
49    KELLY, Mark Edward   0.00194
```

```
78      SANDERS, Bernard  0.005174
37   HARRIS, Kamala Devi   0.00618
56       LOEFFLER, Kelly  0.006326
98     WARREN, Elizabeth  0.006539
94    TRUMP, Donald John  0.007847
45       ISAKSON, Johnny  0.007968
7   BOOKER, Cory Anthony  0.009054
52        KLOBUCHAR, Amy  0.010348
34   GILLIBRAND, Kirsten  0.010635
```

Summary of findings

Over the handful of visualizations pulled together for the Senate voting data we have seen patterns in how senators vote. We have seen specifically that senators typically vote along party lines. We have seen that during the 116th senate session, the democrats were in opposition to the more numerous republicans. When we dig deeper in to the data we see that some senators break from their respective party and gain some political visibility. This feature is observed when we look at the most 'disloyal' senators within the democratic ranks, we can see that the majority of these senators were actually running for the presidential nomination that year. One potential conclusion is that these individuals seek to stand out from their party and go rogue on several votes.

This completes question 3 of the grading rubric.

Notes on approach and synthesis of findings

I leveraged my professional network for 'experts'. MY reasoning is that most people are familiar with politics and would lend insight. I made interviews with people over my lunch break and compiled their suggestions.

The first insight was that grouping votes by individual states did not help visualize the data. Often times, both senators from a state would vote in the same manner which was not visual on the charts. I opted for showing votes according to senators.

The second insight was that voting is drastically different from bill to bill. I chose to show how senators voted for all of the bills brought forth during the 116th session. This allowed us to see that some senators voted "Nah" way more frequently than "Yea" and this division seemed to be based on party lines. This was confirmed when we added a chart that grouped aggregate votes according to party lines.

In future iterations I would like to dig into specific issues within bills and show how parties & individuals vote. This potentially could be done by pulling bill text from an online repo and running vectorized text data through a unsupervised ML engine, specifically a clustering algo, to find how bills are grouped relative to one another. KEy words from these groups could be used as a basis to

infer what issues are on the block. These issues can be added to the dataset and used to make additional charts.

This completes tasks 4 & 5 from the grading rubric.