HouseVis M4 Report

Anastasios Mangelis Apostolos Papadopoulos

Contents

Motivation

What is HouseVis

HouseVis aims to analyze and understand the U.S. House of Representatives Roll Call Data. Therefore the project belongs to the design study category. We aimed to create a visualization system in order to help solve this visualization problem. The data set contains roll call data (i.e., voting history) from the 108th House of Representatives — data about 1218 bills introduced in the House and how each of its 439 members voted on it.

Bills vary in importance, for example some bills reform the U.S medicare system and others names post offices. There are eight different types of bills. House Simple Resolutions (H.RES. in database) and Senate Simple Resolutions (S.RES.) The resolutions are not presented to the President and do not have the force of law. The resolution is used for matters such as establishing the rules under which each body will operate. House Joint Resolutions and Senate Joint Resolutions (H.J.RES., S.J.RES.) require the signature of the President and the approval of both chambers. For every visualization the bill type and subject will be taken in mind.

The data covers the years 2003 and 2004. Potential users of HouseVis will be better positioned to understand the voting record of their Representatives, see never explored before relationships between the bills, the Representatives such as best performing bills in each policy area, congressional delegations, and more. Existing Congressional tools are either too expensive or too outdated and, usually, not open to the public in a form of a reliable API or a user-facing application. As such, users who are interested in Congress and its proceedings have a hard time not only understanding the data but also accessing it. HouseVis aims to solve both problems.

The HouseVis solution

HouseVis proposes to make a Tableau dashboard implementing the aforementioned ideas. The dashboard will break down the data by Representative, policy area, bill-to-law ratios, and Congressional delegations.

For the design approach toward building HouseVis please consult with the "Approach" section. For the implementation details of HouseVis please consult with section 3, "Implementation", and its subsections.

Target group

HouseVis' target group is primarily folks who either employed in or related to the politics, public policy, journalism, civic tech, and lobbying industries.

These are people of of various backgrounds, however the most common are: public policy, economics, law, and statistics. HouseVis' target group is mostly employed in government and non-government organizations (such as think tanks) or media organizations. In government we usually find strategists, political staffers of all levels (from chief of staff to legislative correspondent), and advisors, from two of the three branches (executive and legislative) who are tasked with crafting legislation and the relevant strategy to pursue it. In non-government organizations we usually find political scientists, policy analysts, and data scientists who are tasked with research, writing policy briefs and papers around specific their policy areas.

Both groups value historical hard data because it enhances their institutional knowledge (knowledge that is about other Representatives and Committees; how they've voted and which strategies they've often used to advance legislation.) Admittedly, HouseVis can't help with the second task – only full time Congressional employees are able to do so after years of experience. On the other hand, HouseVis is a great companion for the first task.

Secondly, both groups are eager for actionable data-driven analysis of the Congressional landscape so that they can better understand and navigate it for the present and future. Thus they are able to know when (and to whom) they should "lobby" for a particular interest of theirs.

Use cases (scenarios)

As such, and with that information in mind, we've created two use cases (scenarios) that we believe they highlight the key components of the HouseVis analysis and showcase how HouseVis can assist its users. Both use cases were presented in the `M2` milestone presentation and are listed below.

1. John Appleseed

- Name: John Appleseed
- Age: 21 years old
- Occupation: Political science (major) and Computer Science (minor) student;
  Future campaign & Congressional operative; currently intern for a Massachussetts Democratic Representative

As a political science and computer science student, John is very much into analyzing current affairs with an analytical approach. With HouseVis he can study how Representatives from different parties & ideologies vote over time.

Furthermore, as an intern for a Massachussetts Representative, John is tasked with low level data management, legislative correspondance, basic research, and other ad-hoc tasks issued by senior staff. He is often called to research

other Representatives (from both parties) and their voting records from past Congresses in order to establish potential cooperation on specific policy areas between his boss and the other Representative. With HouseVis he can do so.

With this quantitative information on hand, John is able to infer a qualitative comprehension of the Congressional landscape so that he can assist the senior staff and his Representative.

As a future campaign operative, John will also be able to understand his opponents' voting records in order to further calibrate and customize his ad spending in order to effectively challenge them. By using this congressional data, he can better position his candidate during the campaign and highlight the opponent's flaws and faults.

2. Nate Silver

- Name: Nate Silver
- Age: 35 years old
- Occupation: Journalist

As a data-driven journalist, Nate always strive to write pieces that are based on statistical and data analysis so that they're more objective. He also uses data not only for a historical look back, rather also for prediction modeling future scenarios.

Moreover, Nate is thie chief political editor and correspondent for his organization. (Both for Congressional and campaign coverage.) It is safe to infer that he wants access to high-quality data and analysis which can assist him in writing and understanding the background of each his stories.

Based around these two constraints (historical look back and future modeling,) Nate can use HouseVis to help him in his work. Nate is more focused on bigger-picture thinking. As such he's not particularly interested in how *one* Representative has voted over time but how his delegation or ideologically similar colleagues have. That said, he can use HouseVis to see how party factions have voted and evolved over time and in specific Congressional Sessions. He can understand how legislative subjects have been voted on in specific sessions. He can also see how different state delegations of one party are from a delegation of another state (e.g.: California Democrats vs. Kansas Democrats.) He can also see how productive each session has been.

Using this historical data Nate can also better train his models by feeding them qualitative (his own deductions) and quantitative results as additional input. Thus, he can potentially, try to predict with even bigger certainty specific outcomes because of the enhanced understanding of past Congressional behavior.

Tasks

As elaborated in in the class lectures, a task is design-independent. It's something a user wants and can do regardless of what design is chosen. Thus, in this section, we list system-/design-independent tasks that can be performed generally and not only by the two aforementioned use cases.

- A user will be able to see how often a certain representative has voted.
- A user will be able to see when is the most apt time each year for introducing legislation about a specific topic.
- A user will be able to see how similar bills have been voted in the past.
- A user will be able to see how important specific policy areas are relative to the rest of the legislative output by size and by year.
- A user will be able to see how big and how influential are other congressional state delegations broken down by party.

Hence, tasks are scenario-agnostic and essentially used by all HouseVis users, whether they're closely related to the John and Nate user archetypes or not. Moreoever, they're closely associated with various HouseVis views as described later in this report.

Related work

A variety of interesting visualizations has been implemented over the years, pointing out major differences in The United States House of Representatives. Newspapers are often releasing infographics from researchers in order to show something in a more appealing way for the reader and prove a point in a more unbiased way. [3]

The main issue with the most approaches is that the researches are trying to visualise in the most effective way, without taking in mind that some readers either aren't familiar with the topic or can't understand the complicated visualisations. Furthermore most of the tools aren't free to use and the interactivity offered is not satisfying.

Legislative Explorer, [1] a very interesting visualisation tool implemented by the center for American Politics and Public Policy offers not only great interactivity for the users, but also useful information for every citizen. Legislative Explorer offers a variety of filters, like topics or individual bills that makes the search easier. [2] While it servers as a free demonstration of the data and as a good concept for the everyday user, Legislative Explorer doesn't show deep enough information and can't serve every target group.

Journalists for example are interested in how different state delegations of one party are from a delegation of another state, or want to see how productive a Congressional session has been (especially around election periods.) In our

case, we tried to implement a free tool that is primarily useful for journalists and public policy analysts with the potential to go beyond these two groups to everyday users. Legislative Explorer worked as a fundamental tool to help us understand the most effective way of using filters in order to provide a tool deep and easy enough to use.

Approach

Chief aspect of design and visualization approach was "functional simplicity." We aimed to create a tool powerful and sophisticated enough that can visualize the necessary data in a way that is easy and quickly to comprehend for the end-user without being excessive in visual styles (colors, charts, bars, scales, etc) or functionality (more – or less – interactivity than needed, cumbersome navigation, bad user experience.)

Regarding the visual guideline of HouseVis we based our research (and thought process) on Edward Tufte's rules and "best practices." [4] Namely:

1. **Above all, show data**
   In our opinion, HouseVis is successful at completing this goal. The data representation is clear, understandable, and efficient.
2. **Maximize data/ink ratio**
   Overall, ink is mostly used for data and not for extraneous elements that clutter the graphs. There's also ink used for text and chart legends. We think it's appropriately done so.
3. **Erase non-data ink**
   This was mostly covered in the previous point but we want to note that given the complexity of the visualized data, we opted for slightly more verbose text in order to explain easier a given chart. (In one case, however, we decided to use a background shade in order to better highlight the chart contents due to contrast.)
4. **Erase redundant data ink**
   One aspect of data ink reduction could be the simplification of some chart data. Other than that, the charts used are fairly standard, and while in fact they could be slightly more to the minimal side of things, we think they don't add any visual clutter and cognitive load.
5. **Revise and edit**
   One of our first priorities for subsequent versions of HouseVis. We could debate whether some text perhaps could get cutt off.

Some subjective dimensions of the visualization we would like to cover include aesthetics, style, playfulness, and vividness. The visualization's aesthetics are in our opinion well implemented. It may be true that they could have been improved but together with the visualization's overall style they fit together nicely. HouseVis looks modern and fresh, not difficult for the eyes, and allows

for a very easy quick grasp and view of the data. By playfulness and vividness we would like to elaborate that they mean that they don't detract but enhance the data. The vividness factor ties well with the color encoding that was used.

Furthermore, ater our M2 class presentation and receiving feedback from the course teachers as well as discussing HouseVis with fellow students, and further iterating with our dataset, we decided it was best to amend and deviate from certain aspects of the initially proposed version.

The reasons behind our rationale was both because of design thinking as well as technical limitations of the dataset. We saw that some charts were not possible to be made because either Tableau wasn't able to produce them (such as the Github-like calendar) or the dataset wasn't complete enough (e.g.: plotting subject per district level).

In that case, and based on the initial design approach we improvised and extended it. We decided to develop new ways to derive meaning from the datasets. So we created similar or completely different charts.

Having said that, we did customize the dataset further in order to produce other plots. Using Python as a key functional aspect of our approach and process, we implemented basic exploratory data techniques in order to generate new csv files that we either combined or used them standalone. The Python file is also attached in the website for inspection.

Implementation

Implementation Details

We used Tableau to implement the HouseVis dashboards. We opted for Tableau because it allows for efficient experimentation with one's dataset. After experimenting with our dataset we settled with views that appropriately showcase between discrete data, dimensions, geographical data, and measures.

Of course, we also aimed for views that have high semantic meaning in order to establish the sufficient fullfilment of the general and scenario-specific user tasks. In this process, we heavily employed the aforementioned Tufte rules.

Charts in our Tableau dashboards are fairly standard. They're bar plots/histograms, cartograms, and stacked bars. We selected these as a more "conservative" approach because of their universality. We wanted to make the dashboard for the user as streamlined as possible and it's effectivelly understood by the users even without them having extensive knowledge of different visualization techniques. We hypothesized that more complicated charts would perhaps increase the cognitive load of a user, thus hindering the effectiveness and speed of transfering the contents of the chart.

Initially we did opt for some bubble charts as well as treemaps. However, as discussed later in the "Discussion" section, we reavaluated based on the feedback we received from users and the teaching staff. Thus, we redrew the charts into more traditional — and better "performing" — charts. It was the better choice.

The front-end simplicty, of course, implies heavier (and more targeted) data analysis in the backend. That we covered with the below described Python program.

Another reason for selecting a more conservative array of charts is because of the dataset. As mentioned earlier, due to dataset architecture constraints we had to use Python in order to restructure and exctract additional information from it, that Tableau itself couldn't derive. Thus, by creating individual customized csv's we weren't able to properly integreate them in a more granular scale.

Python

We created `wrangler.py` (available on the website as a standalone file) using Python. Wrangler is a small tool that parses the Stanford csv dataset and cleans and restructures some columns and rows in order to create specific sub-datasets that Tableau will be able to read in order to create the proper charts.

For instance, with `wrangler.py` we're creating the "passed bills" custom data set required for a few charts.

This happens as follows:

```
passed_bills = []
for line in bill_by_votes:
  if line[1] > line[2]:
    line = line[3] + "," + line[0] + "," + str(line[4])
    passed_bills.append(line)
```

And then we save a generated Python `list` or `dict` on disk with:

```
save_csv(transportation, "transportation", "category,date,bill_name")

def save_csv(csv, file_name, header_line):
  with open (file_name+".csv", "wb") as f:
  f.write(header_line+"\n")
  for line in csv:
    f.write(line+"\n")
```

In the above example, we save in a CSV file the frequency of the transporation-related bills data set.

Another task of `wrangler.py` is to rename some some states. The dataset included them neither in full writing nor in standard postal abbreviated form

thus Tableau wasn't able to properly read all of them. Thus we parse and rename. This, for example, could've been in an `awk` script as well.

Another task of `wrangler.py` is to create the csv's for the bill by votes chart (showing the voting record for each bill by Yeas or Nays) and the passed bills charts (i.e., bills that were successfully advanced to the Senate.)

Implemented chart types

We organized our charts in four dashboards. The dashboards are uploaded on our website & git repository for interactive & live viewing as well as local experimenting with the `.twbx` files themselves.

For dashboard one we decided to use stacked bar charts to show the bills that passed. Easily by hovering over the bars you can get the name of each bill that passed, the number of total votes and the month/year it was voted. The main purpose of this dashboard was to give the user a good overview of the bills that passed the House and mainly bills that was voted the most (most important). Stacked bar chart offers a good and clear overview without any need of interaction in this case. It works good for this task because we only have a few number of groups (Year Quarters) and also is ideal for showing data for more than one graph in single graph only (number of votes, bill name, year). Finally for even more detailed display of the solution, we used horizontal bars with the amount of yes/no votes per bill name. Because every graph is using different csv's we weren't able to implement one filter for every graph of the dashboard. That is something we improved on however, and now have 2 filters working together. Using the 3 separate graphs filters for now, a user can filter a specific bill name (by choosing the same bill name in the 3 separate filters) and see the amount of total votes, and the amount of yes/no votes.

In the second dashboard, we can filter it, selecting a specific party. People who are interested in the distinct count of congressional districts depending on the state and party can find that out with the help of the specific chart. For those who are looking for more specific information, the Map can provide the exact locations (other visualization options were unlikely) and Representative names from the respective congressional districts. In addition, the current Tableau visualization implementation allows us to quickly search for a specific state. We implemented a bubble chart in order to provide a clear and quick overview of the most important data represented on a map of the continental US (Alaska and Hawaii are included in the dashboard, just not in the produced screenshots.) The bubble chart looked like the most simplified variant to do this task compared, for example, to a bar graph.

The goal of dashboard three is to visualise in which category do the most voted bills exist. Bar chart are pervasive for this type of charting and can be read and analysed quickly. For the crowd who is more interested in details we displayed the categories of bills voted depending on the year using another bar chart.

The fourth dashboard implements a dashboard specifying voting interactions of the representatives. For example stacked bar charts showing the amount of votes for each representative, how many of the bills they voted passed and finally a visualization that allows users to see for which category/subject did every representative voted the most (medicare, education, etc). The navigation will be easy giving the users the options to choose between representatives.

Implemented filters and interactivity

Dashboard 1

2 filters: Name, Passed Bill name. A user can select with the help of the custom dropdown list a specific bill and then see how many times it got voted over the months/years and the amount of Yeas Nays it got.

Dashboard 2

We implemented 3 filters: Representative party, representative name, state. All filters work globally and they are implemented for a smooth search. Because it's not really easy to manually look on the cartogram and search for a specific district or representative, we thought these filters would really help navigate on the dashboard.

Dashboard 3

A user can easily select the category in which a bill belongs in order to see the overall votes of this category either through the years or totally.

Dashboard 4

Since this dashboard returns information about the representatives, we thought about implementing a drop down list so we can easily select a specific representative. Furthermore the users can select a specific party and search representatives depending on the amount of bills that passed.

Views $\times$ use cases and user stories $\times$ views interaction

Different views of HouseVis interact and implement different user tasks. However, in total, the sum of HouseVis views implements the sum of user tasks. Each chart answers one user task, essentially – and by combining different ones, a user can go "further" in his answer.

User stories (scenarios) complement the dashboard views. The way the views are structured and the way a user interacts with a chart effectivelly answers all general tasks (mentioned earlier) and specific tasks (mentioned in use cases, too.) As elaborated below, this kind of interactivity isn't well complete because of the structural problem of having multiple csv data sources in one dashboard

view. Currently implemented interactivity, though, includes ad-hoc including and excluding of different data points and filtering.

However, the way we've worked around the "structural problem" doesn't hinder the tasks of neither John nor Nate. We consider this of pivotal importance. Hence, any user can that falls within the spectrum of our target group and use cases can reliably use HouseVis to infer meaning (within the predisposed tasks) out of Congressional data.

Color coding

The color templates we selected strive to represent distinctions between parties (red and blue,) voting records (Yea, Nay, Present, Not voted, Not member) and years (2003, 2004.)

We opted for palettes that have enough contrast to represent the differences between each sub-category and cognitive familiarity (for instance, blue party is the Democratic party, red the Republican.) Moreover, all colors are interactively explained upon mouse hover of a specific value in a dashboard within the context of interactivity.

Design problems and challenges

Because of the aforementioned issues along our way, we faced some problems which we solved but took some time to properly experiment with what is appropriate. The key problems were on relying on more than one produced dataset (post-Python exploratory work;) thus we couldn't implement interactivity that easily always. However, we managed to do so by using Tableau to its full extent.

We explored how Tableau can work with multiple datasources in one dashboard and with multiple sheets in order to solve this issue. A totally clean solution hasn't been found yet but what we implemented now works very well. Ergo interactivity is not used always to its full ideal potential but overall it's nicely done. Workaround solutions were found and were implemented. They didn't consist of anything really "hacky" or tricky — merely familiarizing more and in-depth with Tableau's features.

Furthermore, another slightly irritating issue was the fact that in spite of the state name correction using the Python script, Tableau was still not able to parse all states. For example in the Representatives/district map, Wyoming, Montana and parts of Nevada and Vermont as well as the Dakotas were not included on the map despite them being in the produced csv files. However, Wyoming wasn't a problem of Tableau since it completely absent even from the original Stanford dataset in the first place.

Other than this hiccup that we've worked around it by employing our Python program, we didn't face any major issues or other design problems. The charts we've experimented with before settling with the current ones were not problematic per se, but didn't perform as good as the current ones against Tufte's rules as benchmarks for instance. We also couldn't implement a calendar chart (akin to Github's) because Tableau doesn't support one.

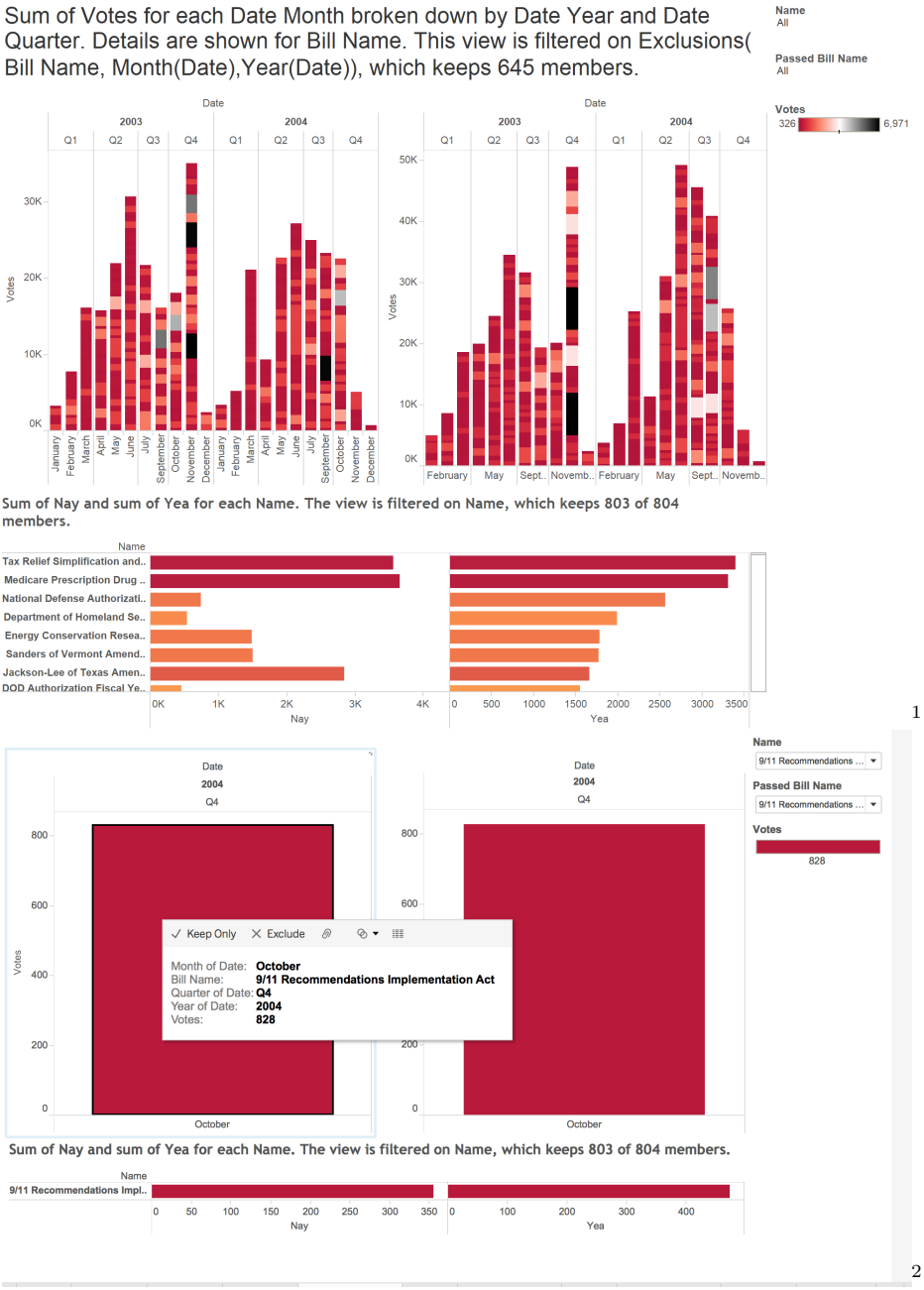
Thus, the changes from the proposed project in `M1` and `M2` to the first major implementation of `M3` and after were focused mainly around the type of charts proposed and actually used, because of the elaborated issues. (As expressed before for instance, we removed the bubble charts after the `M3` review.) We didn't face any problems or any other kind of issue relating to our use cases and scenarios and tasks or technology used.

Results

Demonstration based on scenarios

Scenario 1

Nate wants to see when a certain bill was voted and how it got voted, i.e. its vote tally. Using the first dashboard Nate can search for and select the specific bill he's interested in from the right panels and then identify it in a filtered and refreshed dashboard. Thus, we can see when (month, year, and quarter) the bill underwent a vote and the vote's result on Yeas and Nays. More specific: He decides that he wants to get the total amount of votes and votes result on Yeas and Nays about the 9/11 Recommendations Implementation Act. After selecting the bill name in the filter he gets the result 355 Nay and 473 Yea.

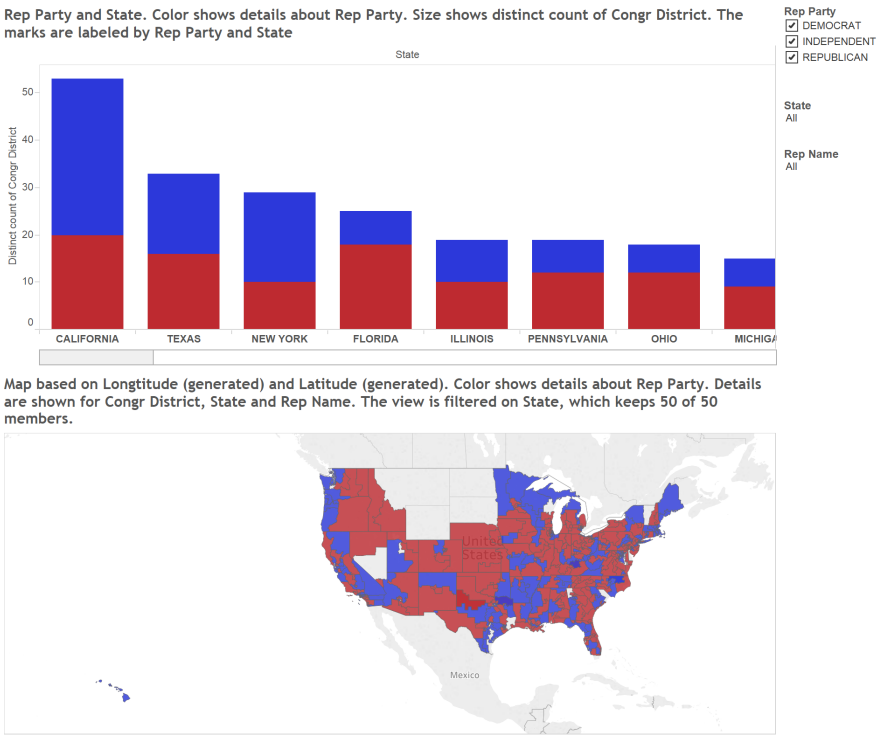


Scenario 2

For his research, John wants to identify representatives from a certain congres-

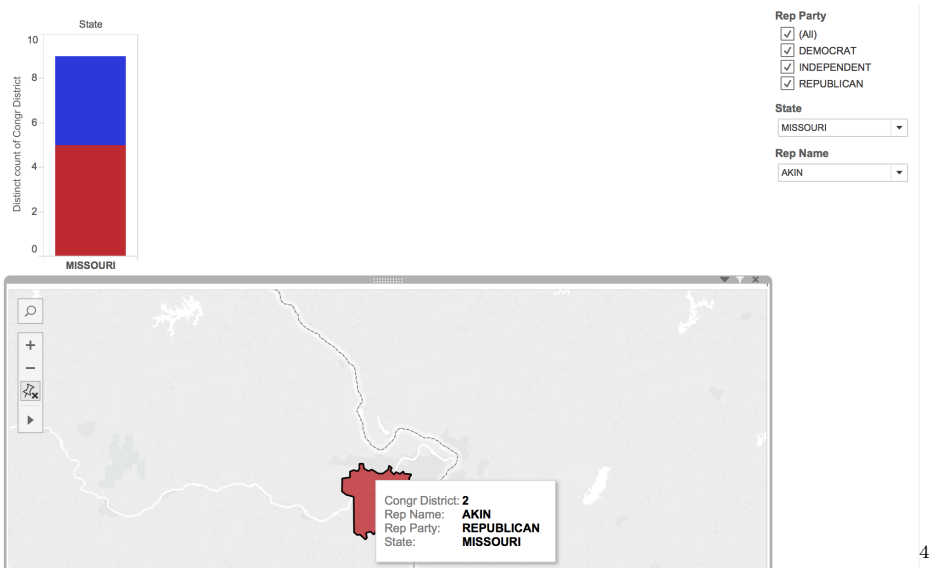
[1]The first dashboard - before.
[2]The first dashboard - after.

sional state delegation, its size, the Representatives' names and districts. Using the second dashboard, John can search, zoom in, find, and hover on the the congressional delegation he's looking for, obtain its size and also filter the specific representative. More specific: John filters the state Missouri to identify how many representatives are republicans and how many democrats. Moreover he can search for a specific representative.



[3]

[3]The second dashboard - before.

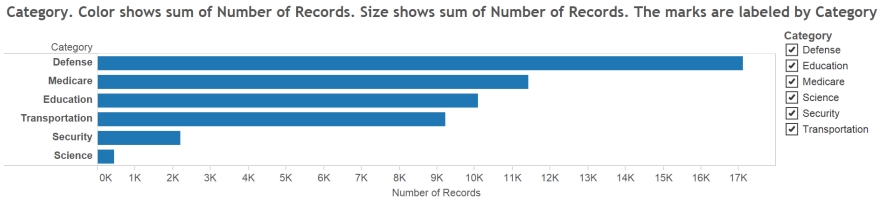


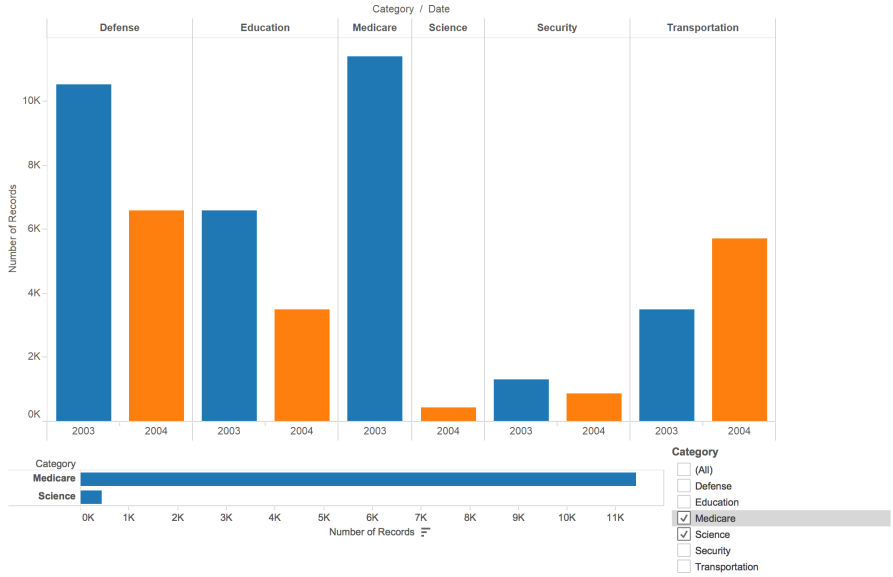
[4]

Scenario 3

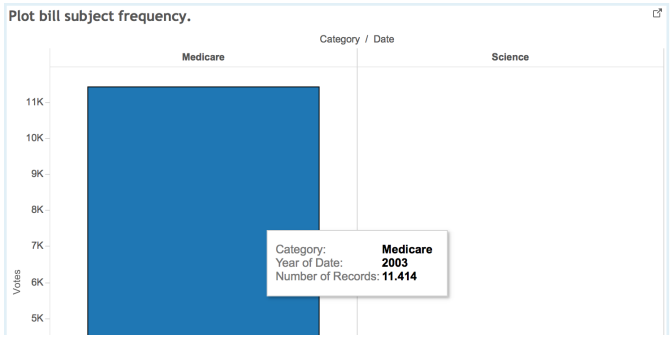
Nate wants to find how often certain legislative subjects get voted on by size and by year. Using the third dashboard, Nate can easily see the breakdown of certain legislative subjects by size in the first chart and when these subjects are voted the most on a year-by-year comparison. More specific Nate wants to find out how many bills voted belong to medicare category and how many to Defense. He selects under Category filter : “Defense and Medicare”

[4] The second dashboard - after.



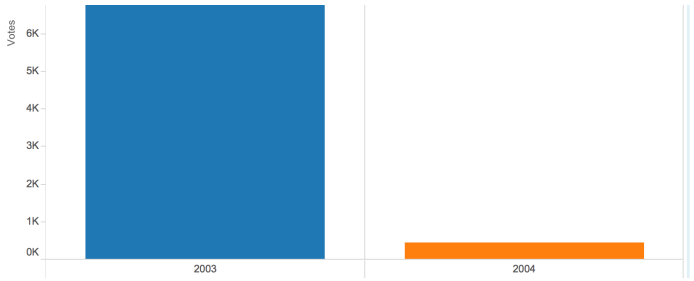


5



6

[5]The third dashboard - before.
[6]The third dashboard - after; part 1.

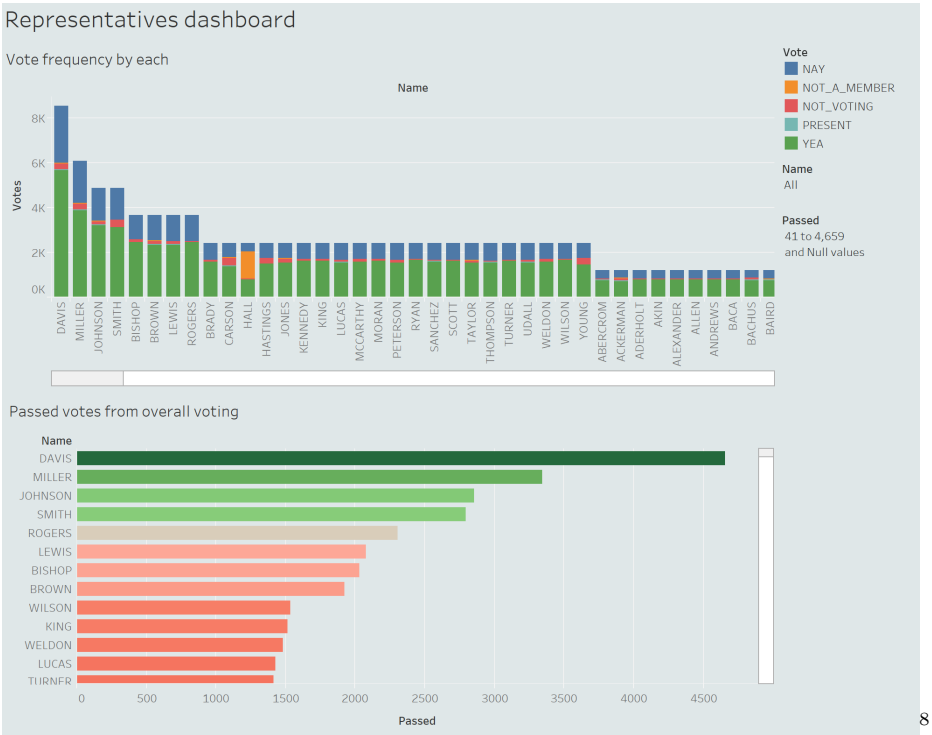


[7]

Scenario 4

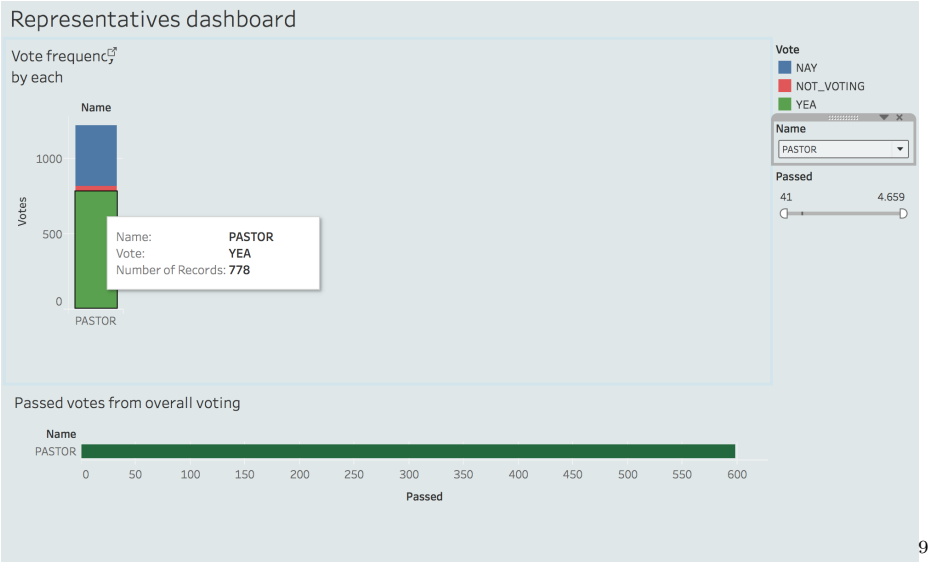
Nate wants publish an article about the Democratic representative from the 4th congressional district of Arizona and their vote record (Alternatively, John wants to do research about the same Representative.)

Filtering his name (the name can be found with the help of the 2nd dashboard,) Nate can find how many times Mr. Pastor didn't vote, what was his vote on specific bills was and how many bills of the total amount he voted for passed.



[8]

[7] The third dashboard - after; part 2.
[8] The fourth dashboard - before.



[9]

Performance and feedback

With regards to performance we're very happy because of the Tableau implementation. It doesn't hinder the browser (nor the desktop, in case one's browsing the individual work sheets) like, potentially, D3. The interactivity is very solid and fast. A user can swiftly change between different dashboards very efficiently. We assume the more Tableau gets developed, the faster our tool(s) will get as well. All in all, the web embeds are exceptionally well developed. Perhaps, in the future HouseVis will incorporate some D3 visualisations as well — for charts that are not easily created through Tableau.

For the best evaluation of the tool we tried to find 3 users, two from the first target group and one from the second and we divided the evaluation phase in two parts. First, two political science students had to complete a simple task we assigned them. They had to find which is Representative Pastor's district and the amount of bills that he voted Yea.

This task was assigned in order to test the usability of the visualisations. Both users found the information required without any struggle. For the second part we gave them a list to fill so we can get the overall interactivity rating, color to data rating, if the dashboards are useful, and if they would use the existing tool in the future under the condition that it will keep being updated, in a scale from 1 to 5 (top to low.)

The results were 2 in interactivity, 1 in color to data. The most successful dashboard was the second one because the combination between the cartogram

[9] The fourth dashboard - after.

and bar chart was really useful. On the other hand, both users hoped that the dashboard 4 would be able to provide even more information about representatives. Both users would use the tool in future and look forward for updates and new visualizations. Finally we asked a journalist's opinion about the tool, and he mentioned that he would really like to see and use a similar visualization with Austrian data.

Discussion

While implementing and evaluating HouseVis with own, user, and reviewer feedback we reached several conclusions pertaining both to our design and visualization approach as well as the implementation itself. In this chapter we will elaborate on the most important ones we want to highlight because we believe they capture the essence of the HouseVis project.

With regards to our design and visualizationa approach we found out that we had to change some of our initial assumptions (as discussed with the teaching staff at the informal M1 meeting and the ones that we later) presented in class in the M2 review. Our assumptions were off for two reasons: i) the initial dataset gathered by Stanford was missing some extra information that we needed in order to implement several charts, and ii) after experimenting with specific chart types in combination with the review feedback, it was clear they were lacking usability – contrary to our initial thought process.

Hence, we had to chiefly iterate and experiment with different chart types. In retrospect, the most obvious was the bubble chart as it is lacking a complete and easy-to-digest visualization of the data. Moreover, another shortcoming was due to the structure of the given dataset. However of its (close to 100%) completeness, the way it was constructed made it cumbersome to extract the required information. In spite of this development, we managed to turn this weakness in a strength by extensively using Python as a data preparatory and manipulation environment. We used this framework in order to create the more detailed and advanced charts in our tool, as well as a data-pipeline. Namely, a specific process from raw data to per-chart extracted and processed subsets of raw data to an easy-to-use consumer tool that visualizes said data. This not only streamlines the process but also allows for future reproducability and scalability.

We are also of the opinion that opting for Tableau was, overall, an additional strength of and for our tool. For two reasons. First, Tableau allows for easy, fast, and high-quality experimentation and data investigation. So iterating between charts was very effective and to the point. Second, Tableau allows for, at the end of the day, creating high quality charts — with great typography, colors, scales, et cetera. The generated charts sport numerous options for user interactivity and filtering thus making them way more engaging than a simple cartesian table in a plain image and are web-embeddable. That way, we can easily create an

interactive online tool that caters to our already described audience.

Pertaining to lessons learned, we want to highlight the importance of reiterating and improving existing aspects of our tool. We incorporated feedback from the teaching staff and other users — and we highly valued that process for it allowed HouseVis to be better. We also want to stress out the importance of Tufte's rules [4] as to what constitutes a good visualization and chart. We found Tufte's work invaluable in our thought and implementation process.

Task seperation

As per usual, we tried to split the work almost equally. Here's the rundown with the help of a simple table.

Apostolos	Anastasios
Report	Website
Charts	Charts
Website	Tableau customization
Python	Report

References

[1] American Politics, C. of and Policy, P. 2014. Legislative explorer. http://www.legex.org.

[2] American Politics, C. of and Policy, P. 2014. Legislative explorer. http://www.legex.org/timeline/index.html#legislation=bills&chamber=0&party=all&committee=all&majority=

[3] Ingraham, C. 2015. A stunning visualization of our divided congress. https://www.washingtonpost.com/news/wonk/wp/2015/04/23/a-stunning-visualization-of-our-divided-congress.

[4] Tufte, E. 2001. *The visual display of quantitative information paperback: Second edition paperback*. Graphics Press.