# What does innovation look like in a given state?

Once you've lived in a state for a few years, you may already have some sense of what the innovation in that state looks like. For California, it's technology. For Alaska, you might (correctly) guess it has to do with drilling on the North Slope. But if you live in a state that isn't so dominated by a major corporation or industry, it might be much less obvious what kind of innovation is happening around you.

Among the data the United States Patent and Trademark Office (USPTO) provides, the Patent Trademark Monitoring Team (PTMT) produces aggregations of the underlying patent data, along many different dimensions and partitions. These aggregations answer all kinds of questions just like this, as well as offer opportunities for deeper analysis using data visualizations.

# Technology classes by state

[Viz EMBED HTML: **<script type='text/javascript' src='http://public.tableau.com/javascripts/api/viz\_v1.js'></script><div class='tableauPlaceholder' style='width: 1004px; height: 869px;'><noscript><a href='#'><img alt='main ' src='http:&#47;&#47;public.tableau.com&#47;static&#47;images&#47;st&#47;state-tech-class-treemap&#47;main&#47;1\_rss.png' style='border: none' /></a></noscript><object class='tableauViz' width='1004' height='869' style='display:none;'><param name='host\_url' value='http%3A%2F%2Fpublic.tableau.com%2F' /> <param name='site\_root' value='' /><param name='name' value='state-tech-class-treemap&#47;main' /><param name='tabs' value='no' /><param name='toolbar' value='yes' /><param name='static\_image' value='http:&#47;&#47;public.tableau.com&#47;static&#47;images&#47;st&#47;state-tech-class-treemap&#47;main&#47;1.png' /> <param name='animate\_transition' value='yes' /><param name='display\_static\_image' value='yes' /><param name='display\_spinner' value='yes' /><param name='display\_overlay' value='yes' /><param name='display\_count' value='yes' /><param name='showVizHome' value='no' /><param name='showTabs' value='y' /><param name='bootstrapWhenNotified' value='true' /></object></div>**]

This visualization serves as a detail view of the technology classes that make up a state's utility patents. The treemap above organizes all of a state's technology classes for a given timeframe, with a rectangle's size representing its share of the patent total, and the line graph below depicts the per year values for a given technology class if a rectangle is hovered over, and all classes for the state otherwise. We can pick which state we're looking at using the dropdownbox on the right.

Let’s take a look at Idaho. As some background, Idaho is a big patent producer, driven primarily by Micron Technology, Inc. So, we see that Idaho's largest technology classes by far are 'Semiconductor Device Manufacturing: Process' (technology class 438, with 21.99% of the total patent share from 1995-2014) and 'Active Solid-State Devices (e.g., Transistors, Solid-State Diodes)' (technology class 257, with 13.95% from 1995-2014). Unsurprisingly, given that the large share of Idaho's patent production is through Micron (a technology company), the largest technology classes involve computers. Vermont has a similar breakdown of technology classes, by way of IBM.

Other states are similarly dominated by a particular technology class (or kind of technology classes). Michigan's largest technology class since 2008, for example, is 'DP: Vehicles, Navigation, and Relative Location (Data Processing)' (technology class 701). Given Michigan's traditional status as the locus of the USA's auto-making industry, looking at the last 20 years (1995-2014), we can see that the second largest technology class is technically Internal-Combustion Engines (technology class 123), but this class stays relatively constant year-over-year, while technology class 701 increases significantly, starting in 2008, corresponding with the automotive industry bailout. This may point to a new, deliberate R&D focus in the industry following the injection of funds. Gas prices drove many consumers to purchase more fuel-efficient vehicles, such as Japanese-made models. Perhaps the US automotive industry decided to differentiate their competitive offerings using services based on GPS, rather than trying to play catch-up with fuel efficiency.

These detail views not only give us a good sense of what industry in a particular state looks like (Oklahoma's largest technology class, for example, is 'Wells (shafts or deep borings in the earth, e.g., for oil and gas)'), but can also help us understand trends within an industry if that industry is very specifically associated with a particular state. The automotive industry in Michigan is perhaps the clearest example, but perhaps we could do a similar (though necessarily more subtle) analysis of Silicon Valley. California is not dominated by a single technology class, but many of their top technology classes are computer-oriented. Which technology classes are rising, and which are falling?

# Getting started

In this tutorial, we'll go over how to collate the required data from the PTMT web site using Python web scraping, then walk through creating the visualization in Tableau Public. If you want a jumpstart, the artifacts for this tutorial can be found at the following GitHub repo: [TODO: repo here]

## SEC 1: Collating data

The data for this visualization lives in 51 different HTML tables (50 states + Washington DC), each on a separate page. We could copy and paste each table into an Excel or LibreOffice spreadsheet, unpivot the tables and export to a \*.csv format, but that doesn’t scale well, and worse, it’s boring. Let's write a script!

**"""This script scrapes and collates state and territory data**

**from the USPTO PTMT data site"""**

**from collections import namedtuple**

**import csv**

**import requests**

**from bs4 import BeautifulSoup**

**# US state codes; add territory codes here if desired**

**REGION\_CODES = {**

**'Alabama' : 'AL',**

**'Alaska' : 'AK',**

**'Arizona' : 'AZ',**

**'Arkansas' : 'AR',**

**'California' : 'CA',**

**'Colorado' : 'CO',**

**'Connecticut' : 'CT',**

**'Delaware' : 'DE',**

**'District of Columbia' : 'DC',**

**'Florida' : 'FL',**

**'Georgia' : 'GA',**

**'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',**

**'Ohio' : 'OH',**

**'Oklahoma' : 'OK',**

**'Oregon' : 'OR',**

**'Pennsylvania' : 'PA',**

**'Rhode Island' : 'RI',**

**'South Carolina' : 'SC',**

**'South Dakota' : 'SD',**

**'Tennessee' : 'TN',**

**'Texas' : 'TX',**

**'Utah' : 'UT',**

**'Vermont' : 'VT',**

**'Virginia' : 'VA',**

**'Washington' : 'WA',**

**'West Virginia' : 'WV',**

**'Wisconsin' : 'WI',**

**'Wyoming' : 'WY'**

**}**

**BASE\_URL\_PREFIX = 'http://www.uspto.gov/web/offices/ac/ido/oeip/taf/stcteca/'**

**BASE\_URL\_SUFFIX = 'stcl\_gd.htm'**

**MASTER\_LIST = []**

**StateRow = namedtuple('StateRow', 'state\_name tech\_code year value')**

**# for each state code, generate the target URL and pull the data**

**for state in sorted(REGION\_CODES):**

**print('Processing data for ' + state)**

**path = BASE\_URL\_PREFIX + REGION\_CODES[state].lower() + BASE\_URL\_SUFFIX**

**r = requests.get(path)**

**soup = BeautifulSoup(r.text, "html.parser")**

**# skip first and last rows, which are headers and totals respectively**

**for table\_row in soup.find\_all('tr')[1:-1]:**

**tech\_code = table\_row.find('td', style=' text-align: left; ').string.strip()**

**year = 1963**

**# skip the last element, which is a total; we can aggregate the data ourselves**

**for value in table\_row.find\_all('td', {'style': None})[:-1]:**

**row = StateRow(state, tech\_code, year, value.string.strip())**

**MASTER\_LIST.append(row)**

**year = year + 1**

**# write out to csv**

**with open('./state\_tech.csv', 'w', newline='') as out:**

**print('Writing data to ' + out.name)**

**CSV\_FILE = csv.writer(out, delimiter=',')**

**CSV\_FILE.writerow(['Region', 'Tech Class Code', 'Year', 'Utility Patent Count'])**

**CSV\_FILE.writerows(MASTER\_LIST)**

The above script uses a dictionary of region codes to generate the URLs of the pages we’re targeting, then scrapes the HTML from those pages (using requests and BeautifulSoup) to collect the data we want. Along the way, we wanted to normalize the data, so we defined a namedtuple that corresponded to the form of the row we wanted to output: [state\_name], [tech\_code], [year], [value], and write the HTML data out to a list of these tuples. Finally, we write out this list to a csv file.

Next, we need to pull the codes and names for technology classes. This script is simpler:

**"""This script scrapes tech codes and names from USPTO site"""**

**from collections import namedtuple**

**import csv**

**import requests**

**from bs4 import BeautifulSoup**

**URL = 'http://www.uspto.gov/web/patents/classification/selectnumwithtitle.htm'**

**REQUEST = requests.get(URL)**

**SOUP = BeautifulSoup(REQUEST.text, "html.parser")**

**TECH\_CODES = []**

**ClassRow = namedtuple('ClassRow', 'class\_code class\_name')**

**print('Scraping data')**

**for table\_row in SOUP.find\_all('tr'):**

**class\_code\_tag = table\_row.find('td', width='27')**

**# not a class\_code + name row. skip**

**if class\_code\_tag is None:**

**continue**

**class\_code = class\_code\_tag.string**

**class\_name = table\_row.find('td', width='532').string**

**TECH\_CODES.append(ClassRow(class\_code, class\_name))**

**with open('./tech-code.csv', 'w', newline='') as out:**

**print('Writing data to ' + out.name)**

**CSV\_FILE = csv.writer(out, delimiter=',')**

**CSV\_FILE.writerow(['Class Code', 'Class Name'])**

**CSV\_FILE.writerows(TECH\_CODES)**

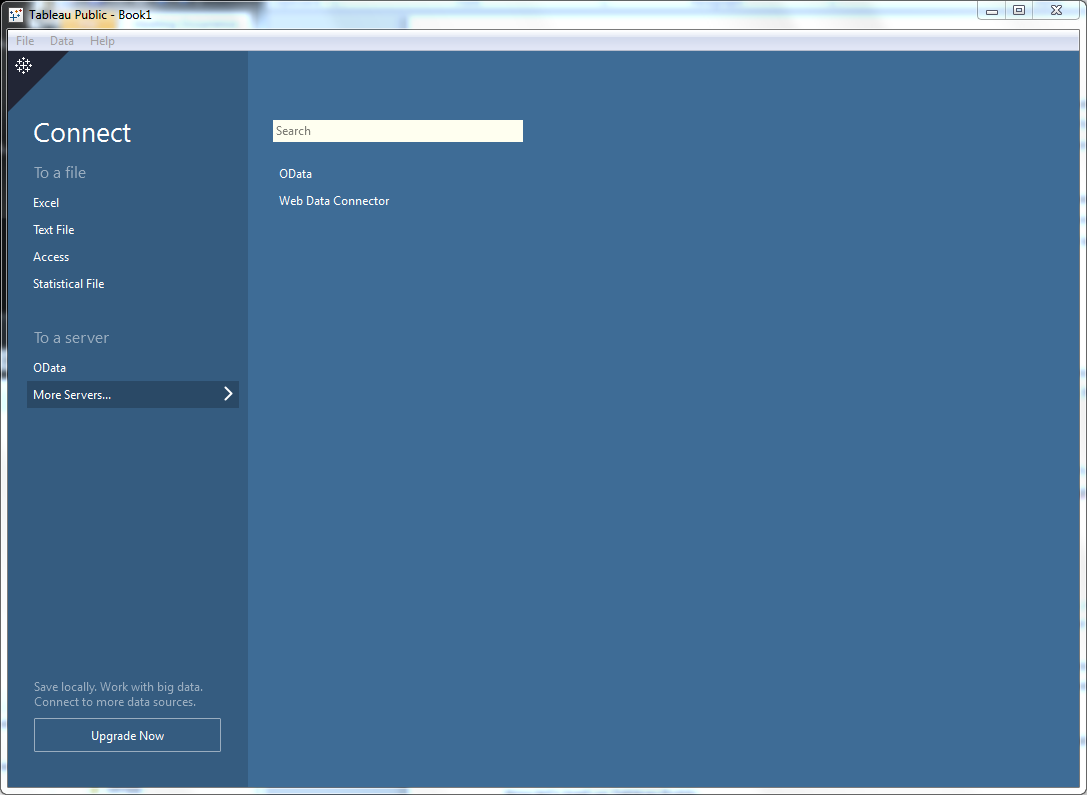
Data collation accomplished!

## SEC 2: Creating the visualization

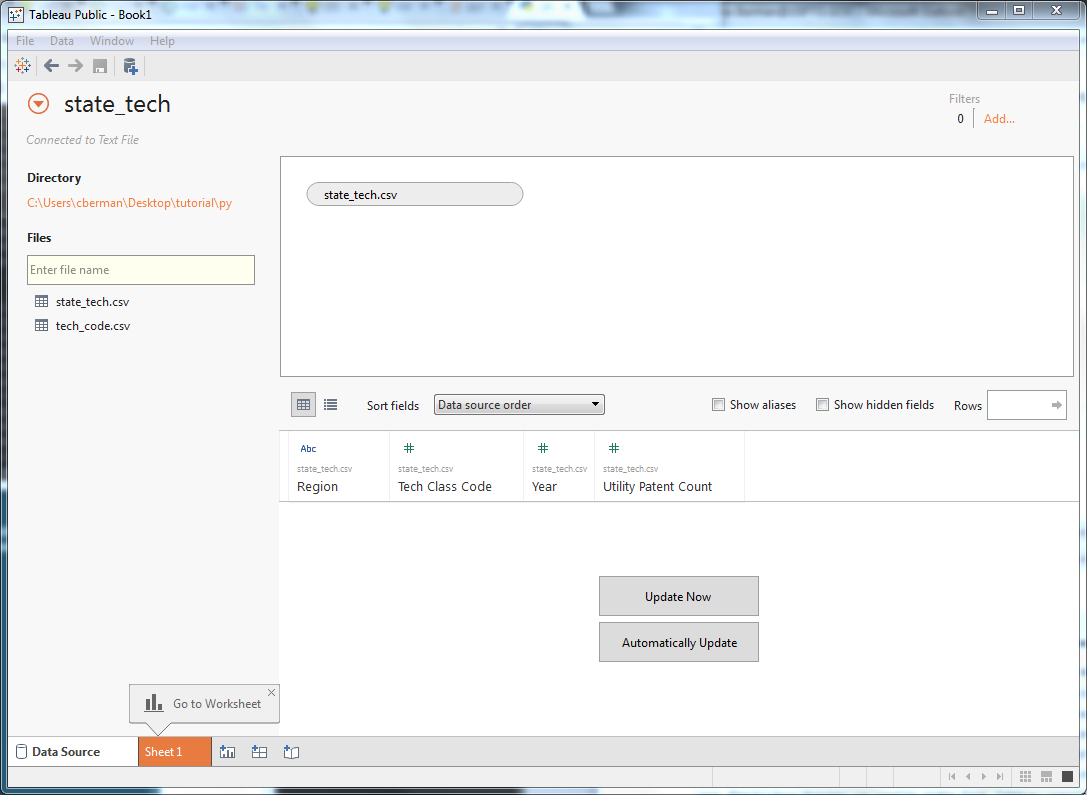
Now that we have our data in a clean, normalized format, let's make the visualization! You'll need to install Tableau Public, a free version of the Tableau visualization software, as well as sign up for a free Tableau Public account. You'll use the Tableau Public software on your computer to create the visualization, then upload the visualization to your Tableau Public account. Once it's uploaded, you'll be able to browse to and interact with your visualization on your personal Tableau Public account web site, as well as embed that visualization in other web pages.

First, install Tableau Public [LINK: https://public.tableau.com/s/]. Once that's done, sign up for a free Tableau Public account.

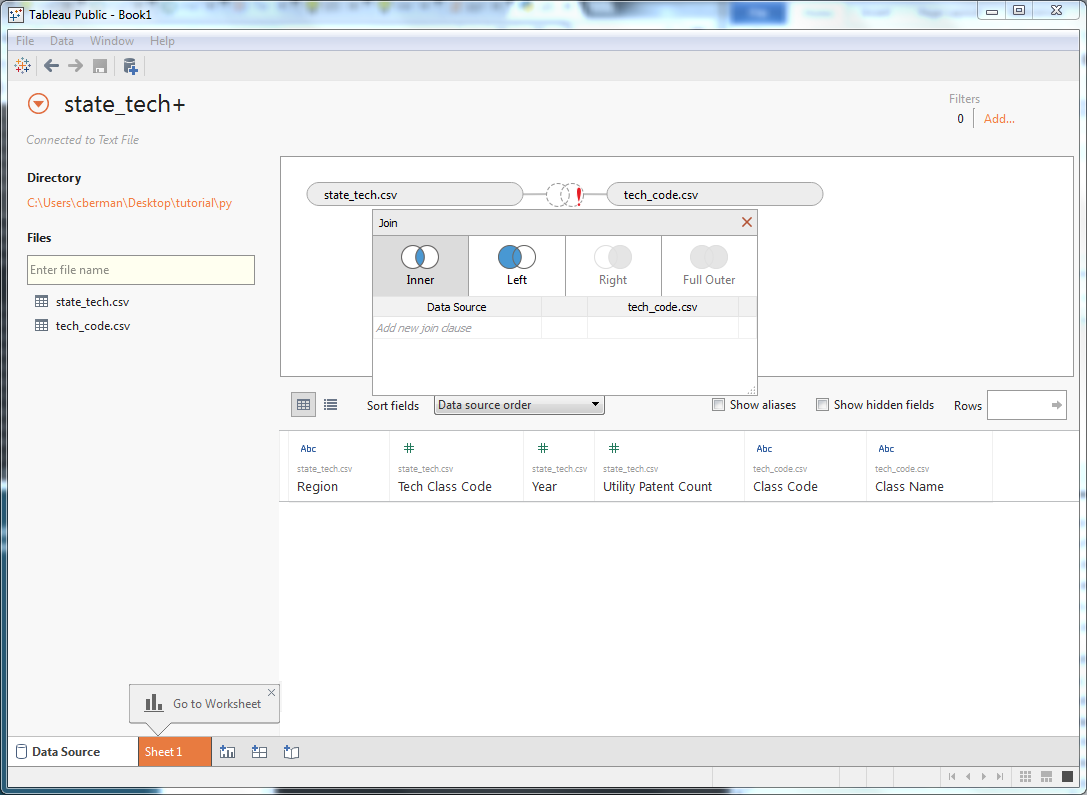
Now let's load up Tableau Public.



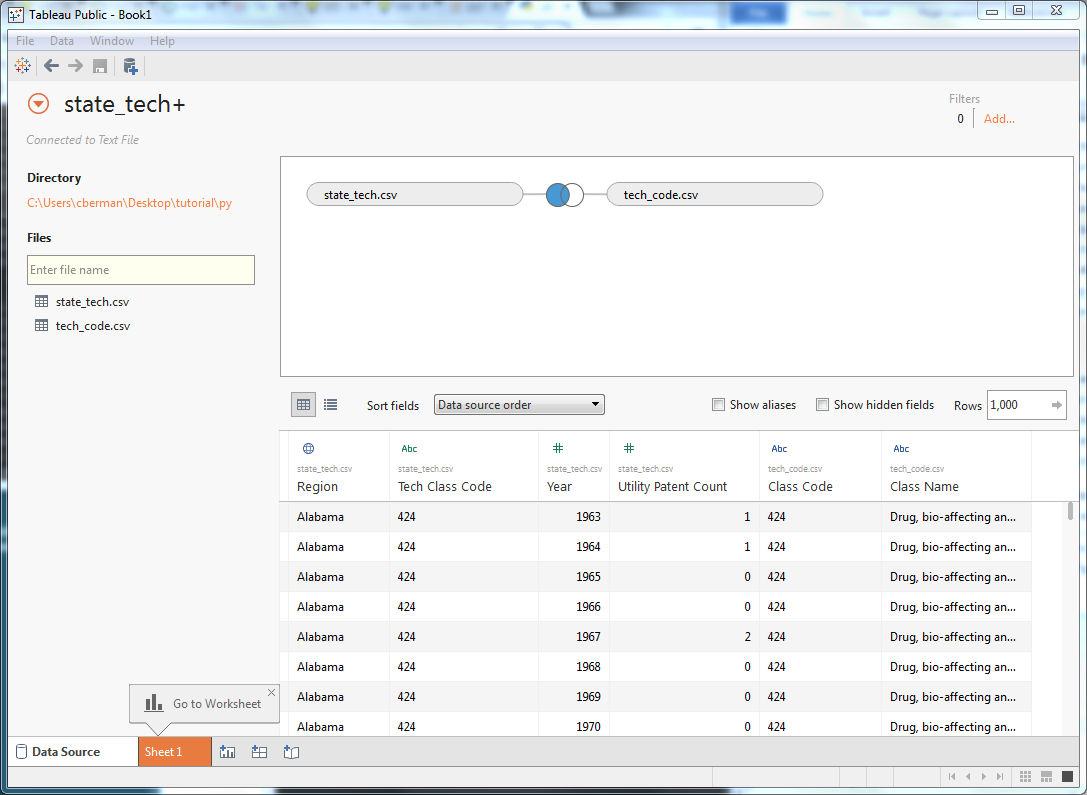
First, we need to connect to our data. Select ‘Text File’ on the left side of the screen, under ‘Connect.’ In the dialog box that opens, navigate to state\_tech.csv file we just created and click Open.



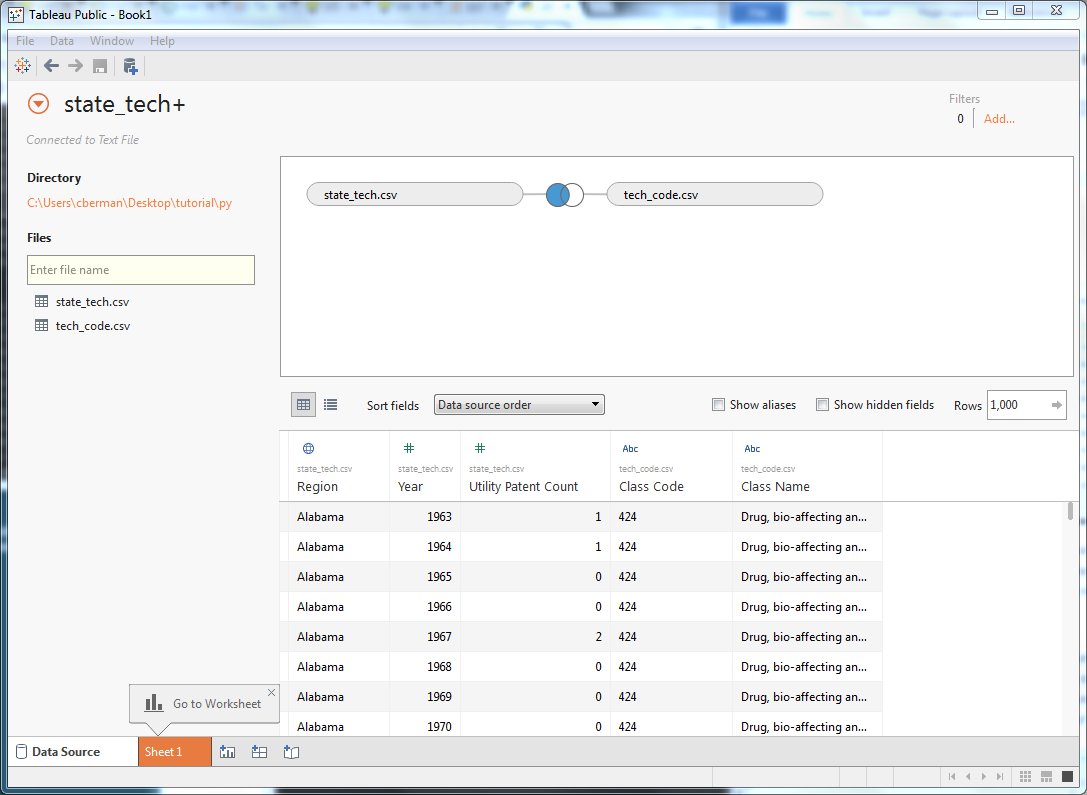
Here, we see that Tableau has linked to the state\_tech.csv file we generated. Our data sources for this visualization will be our state\_tech.csv file joined to our tech\_code.csv file (to supply the name for the technology class). On the left, Tableau lists available text files in the current working directory, and on the top, Tableau shows which sources are currently driving the visualization. So first, we need to join our state\_tech.csv file onto our tech\_code.csv file. Drag the tech\_code.csv entry from the left section (under ‘Files’) onto the top section. Tableau’s Join dialog pops up:



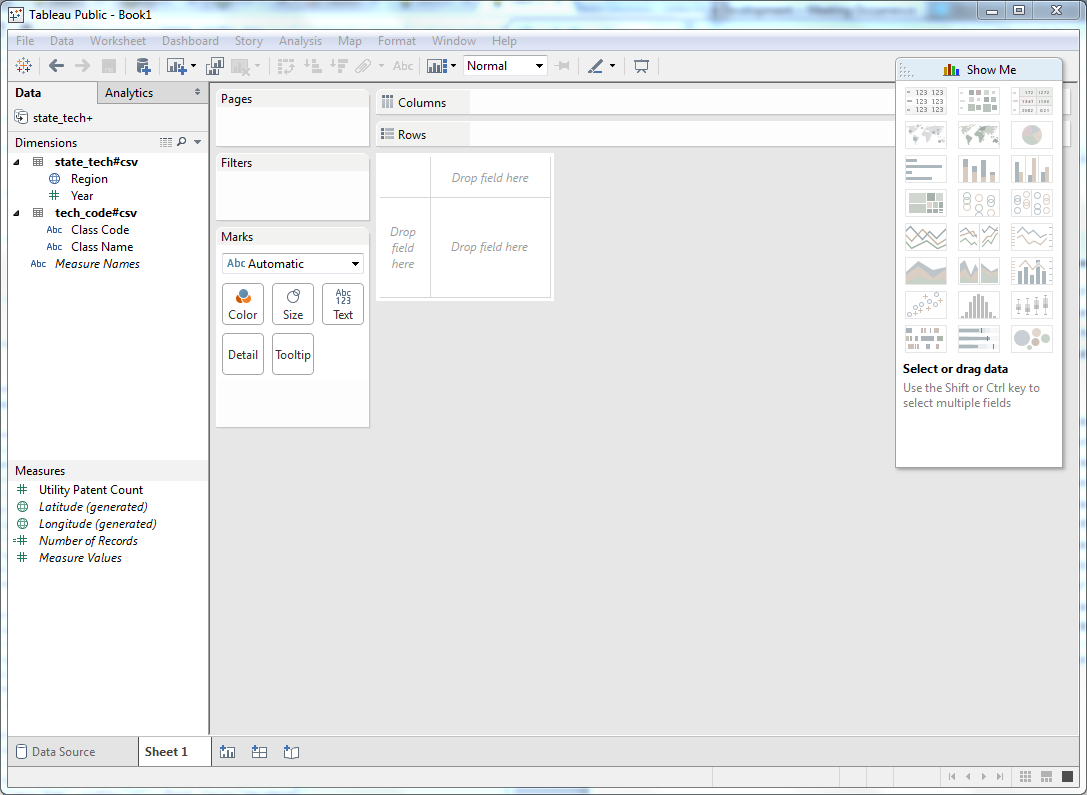
In this case, we want a Left Join, using the ‘Tech Class Code’ field in state\_tech.csv and the ‘Class Code’ field in tech\_code.csv. At first, this won’t work. Tableau supports some extract and load functionality, and will try to infer the data types of the loaded columns. In this case, Tableau infers that the ‘Class Code’ field in tech\_code.csv is a text field, whereas the ‘Tech Class Code’ is a numeric field, which prevents our Join. In fact, there’s multiple changes we need to make. First, Tableau supports many Geographical data types, so we’ll want to change Region to Geographic Role -> State/Province. Next, change the data type of ‘Tech Class Code’ to String. (NOTE: To change data types, click the symbol above a given column header (such as ‘Abc’ or ‘#’)). Once the data types are consistent, the join should work. Click the ‘Automatically Update’ button to see the joined data.



Looking good! Just one more fix: We currently have two columns for class code, but we only need one. Let’s hide the first one (the second column): mouseover the column header and click the arrow in the upper right corner of the header, then select Hide.

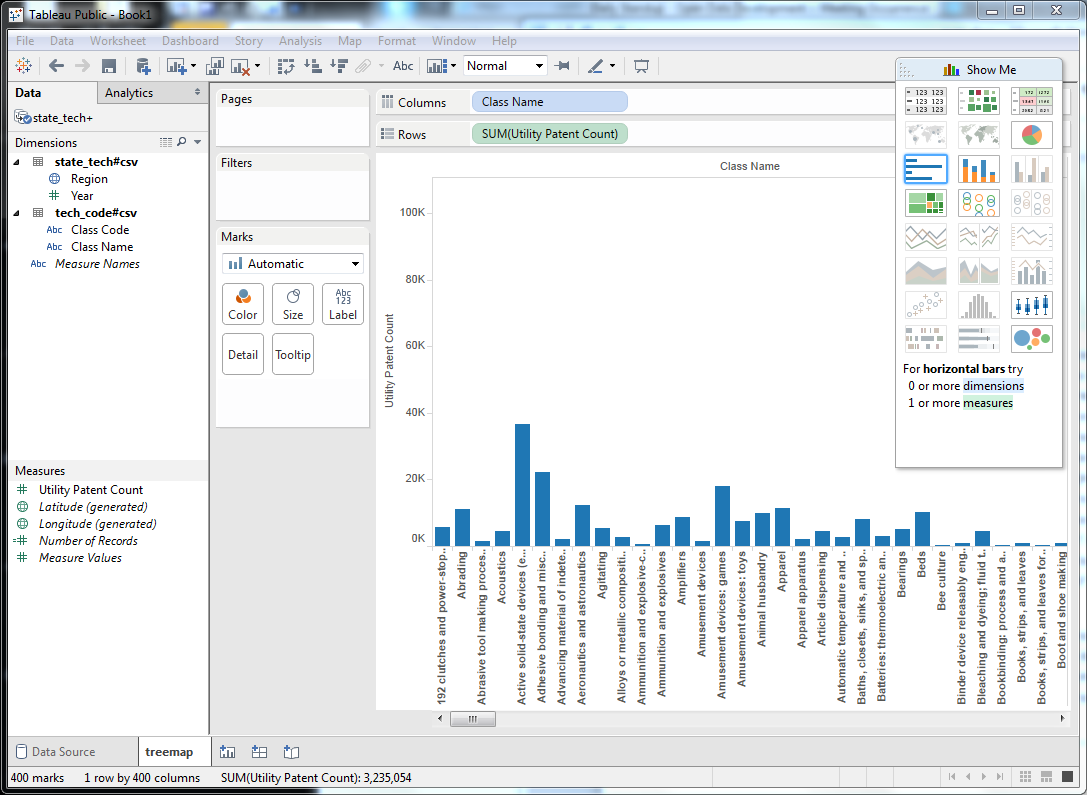


With that done, let’s start putting together our visualization. Select the orange button at the bottom of the page for Sheet 1. This brings us to the worksheet view for Sheet 1. Tableau uses worksheets to represent a single component of a dashboard, which may include many worksheets. For the purposes of this tutorial, we’re treating ‘dashboard’ as synonymous with ‘visualization.’

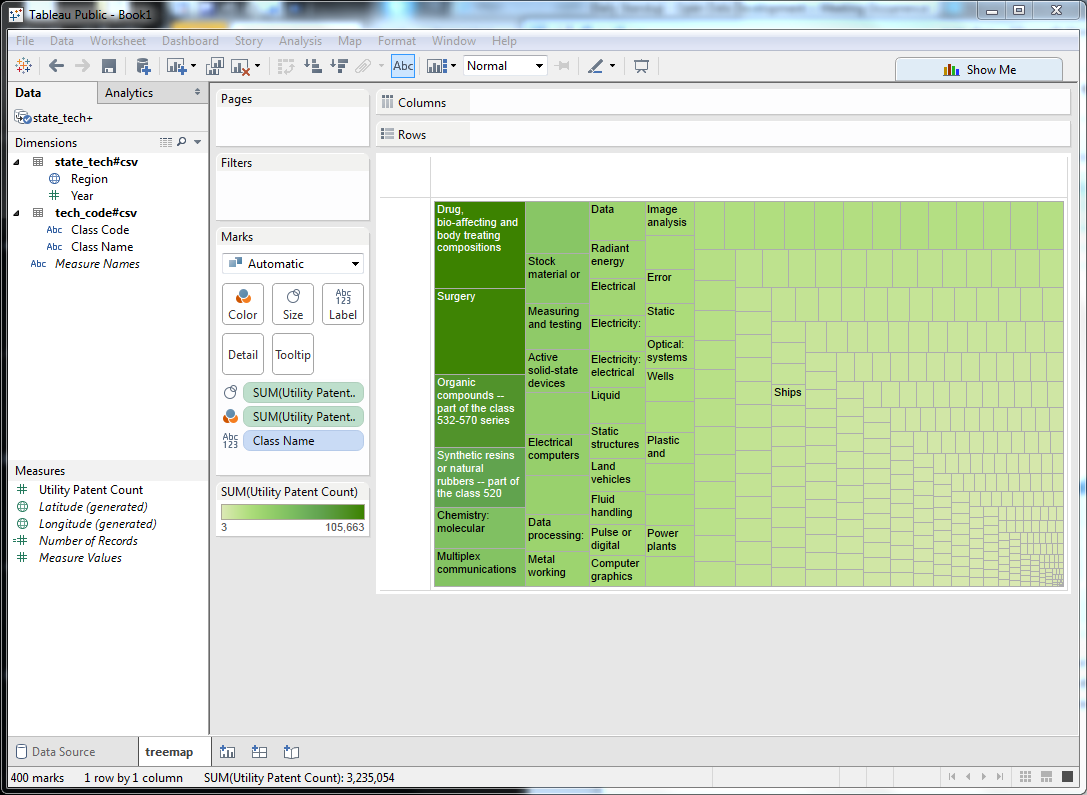


First things first, let’s rename ‘Sheet 1’ to ‘treemap.’ Our visualization will consist of this worksheet as well as an additional worksheet for the bar graph. Now let’s take a look at the screen. On the left side, we find our column names (as well as some extra fields in italics, which for the purposes of this tutorial we can ignore) organized into two different categories: Dimensions and Measures. Put simply, a dimension is a field that identifies a piece of data, whereas a measure is a field that tells us, quantitatively, something about that data. So, ‘Region Name’ identifies a row of data as referring to a given region (such as Alabama), whereas ‘Utility Patent Count’ gives us a quantitative value for that row of data. Tableau will attempt to infer which fields are dimensions and which are measures, but sometimes it requires some adjusting. In this case, Tableau correctly identifies the ‘Region’, ‘Year’, ‘Class Code’ and ‘Class Name’ as dimensions, and ‘Utility Patent Count’ as our only measure. That is, ‘Region’, ‘Year’, ‘Class Code’ and ‘Class Name’ identify the scope of a particular row of data (or, in the case of Class Name, provide additional data on another dimension), and ‘Utility Patent Count’ represents a value for that scope.

Let’s make our treemap. First, drag ‘Class Code’ from the Dimensions section to the ‘Columns’ row at the top of the screen. Then, drag ‘Utility Patent Count’ from the Measures section to the ‘Rows’ row at the top of the screen. This will generate a bar graph, with a column for each class name that has at least one utility patent, and the corresponding bars are aggregated from the Utility Patent Count value for a given class across all states and years.

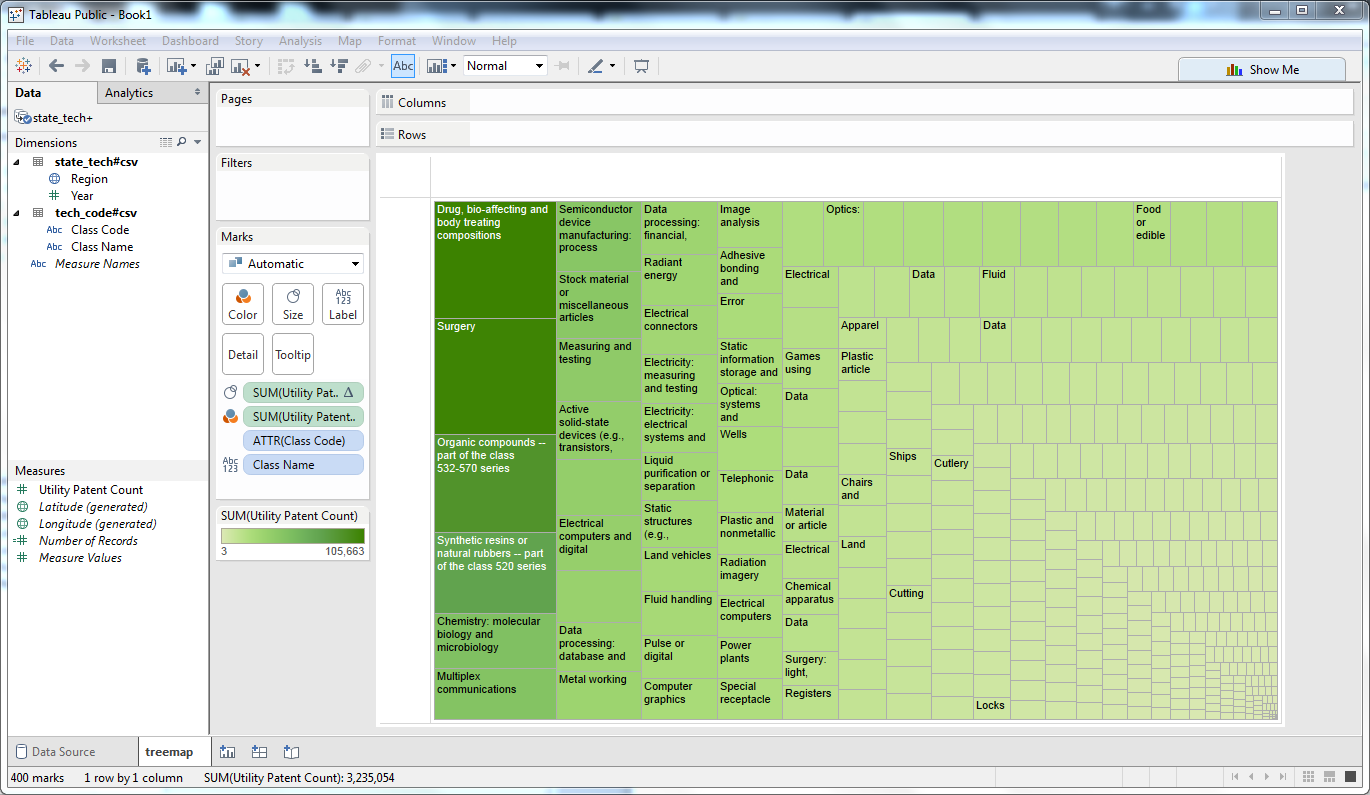


But we want a treemap, not a bar graph. The answer is the Show Me menu on the right. Tableau supports switching between multiple different styles of graph with a single click, as long as the number of dimensions and measures involved are compatible. In the Show Me menu, click the TreeMap square (fourth from the top, on the left).

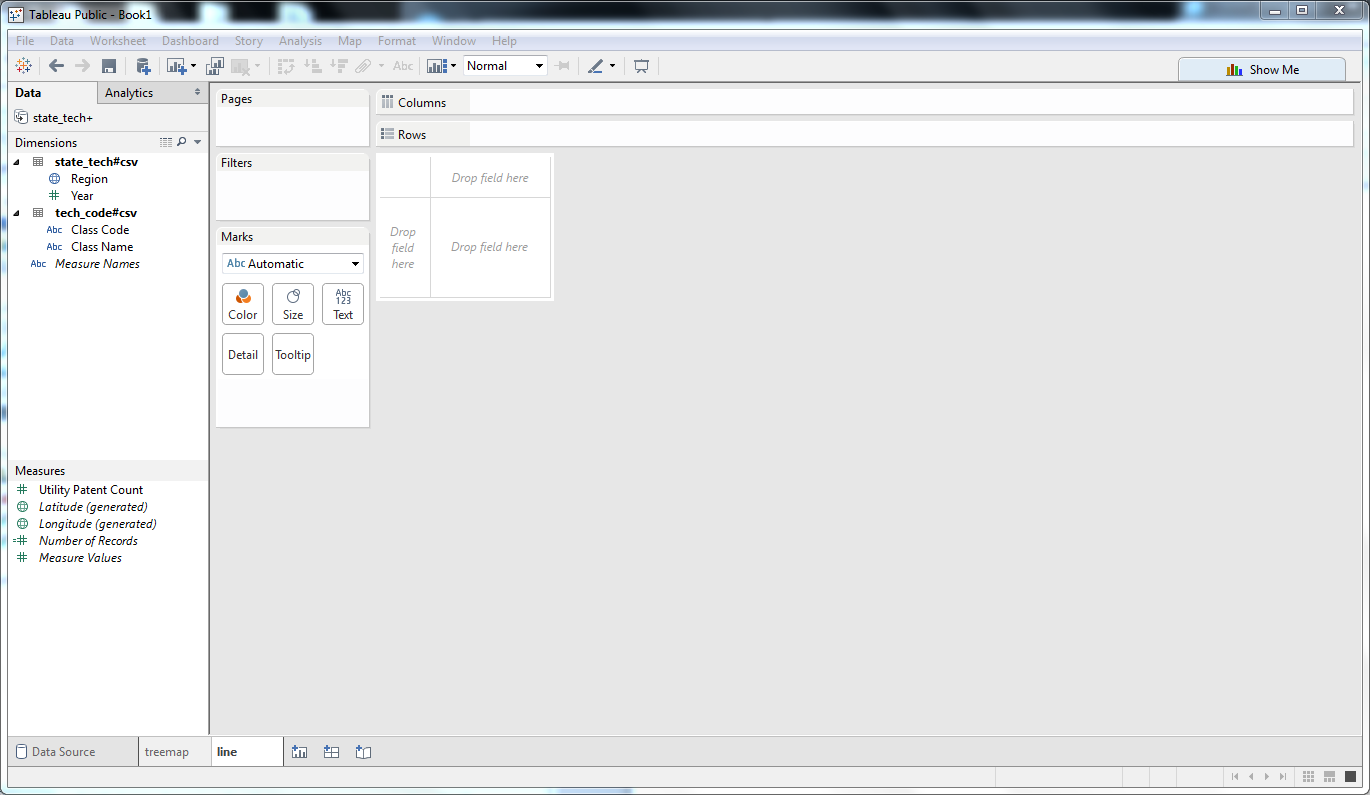


Presto! Tableau moved the Utility Patent Count and Class Name pills from the Columns and Rows rows at the top to the left under Marks. In this case, we could have built the treemap by configuring these items ourselves, but the Show Me menu offers a convenient shortcut.

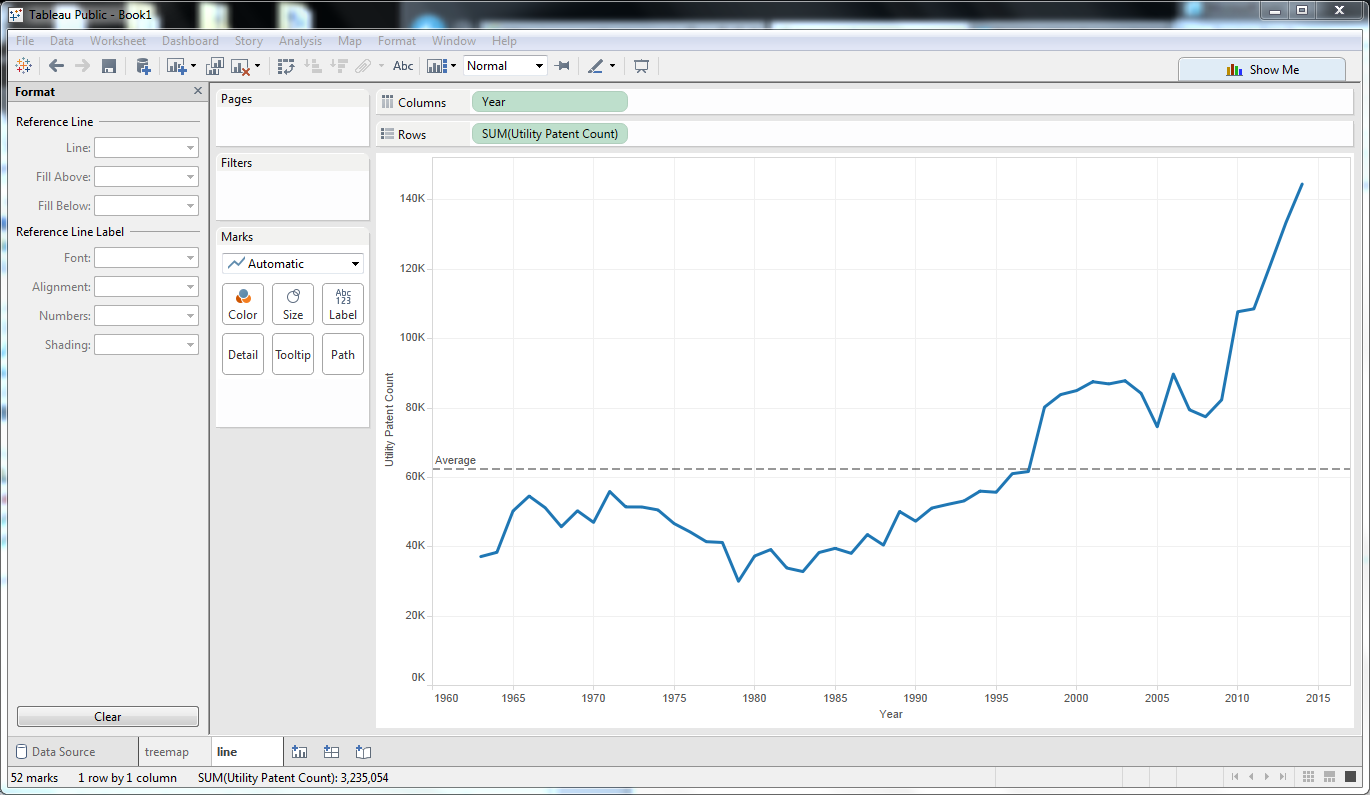
One last thing: Tableau creates a default tooltip using the measures and dimensions involved in the worksheet, but we want to show two additional data items: The class code and the ratio of the patents for a technology class for the total. First, drag the Class Code pill onto the Tooltip square (in the Marks section). Mouseover a square in the treemap and observe that the class code was added. Next, right-click the first Utility Patent Counts pill in the Marks section (with two colorless squares to its right) and select Quick Table Calculation -> Percent of Total. The tooltip now includes the class code and percent of total for a given square. To re-order the tooltip rows to match the original visualization, click the Tooltip square in the Marks section and adjust manually.



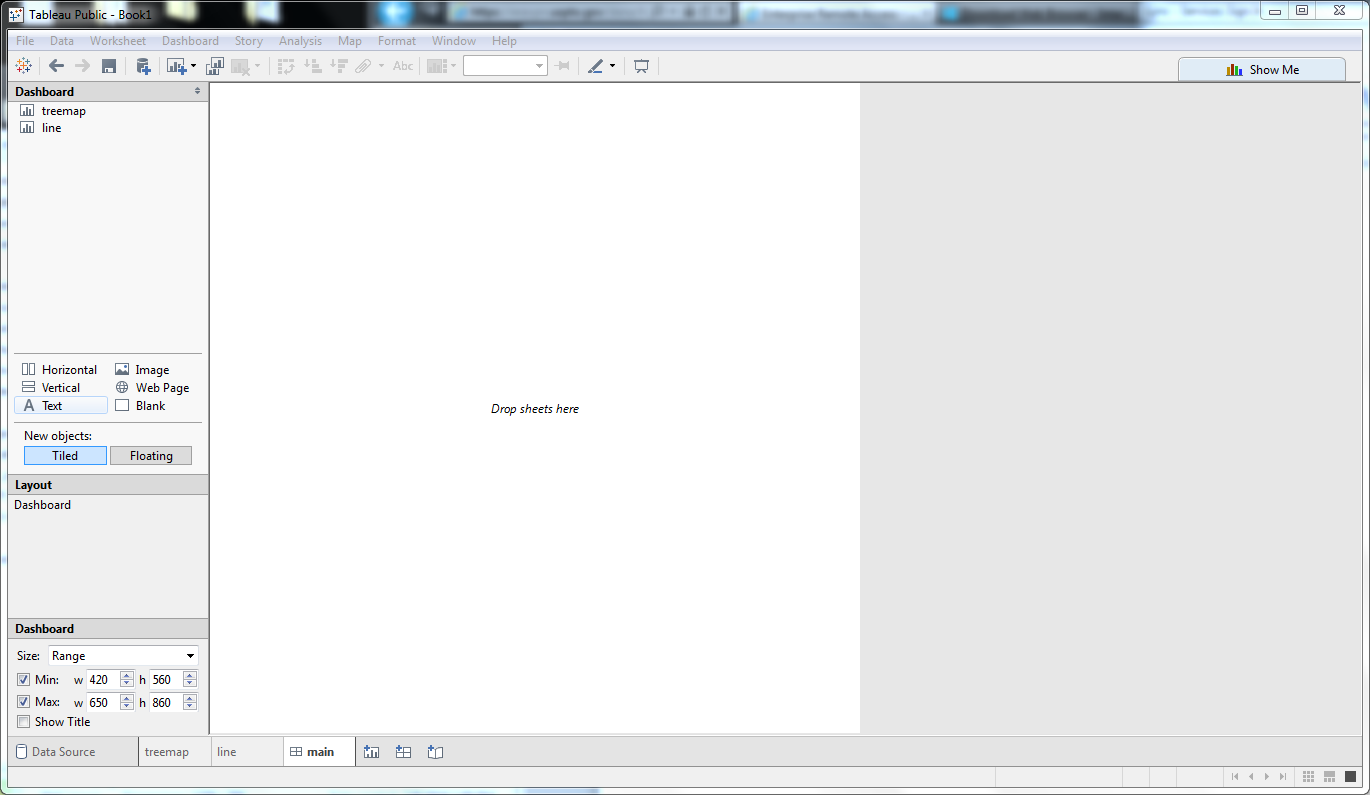
Now let’s get started on the line graph. Click the button directly to the right of the treemap tab on the bottom of the page to create a new worksheet and rename it ‘line.’



For this graph, drag Year to Columns and Utility Patent Count to Rows. Finally, right-click the row header on the right of the graph (labeled ‘Utility Patent Count’), select ‘Add Reference Line’ and press OK. By default, this creates a reference line based on the Average of the Utility Patent Count measure, aggregated by sum for a given year. To have the line appear dashed instead of solid, we can right-click the reference line itself and select ‘Format.’ This opens the Format menu for the reference line on the left side of the screen. Open the ‘Line’ dropdown and select the dashed line. Finally, deselect the reference line by clicking an empty space within the line graph. Your graph should look like this:



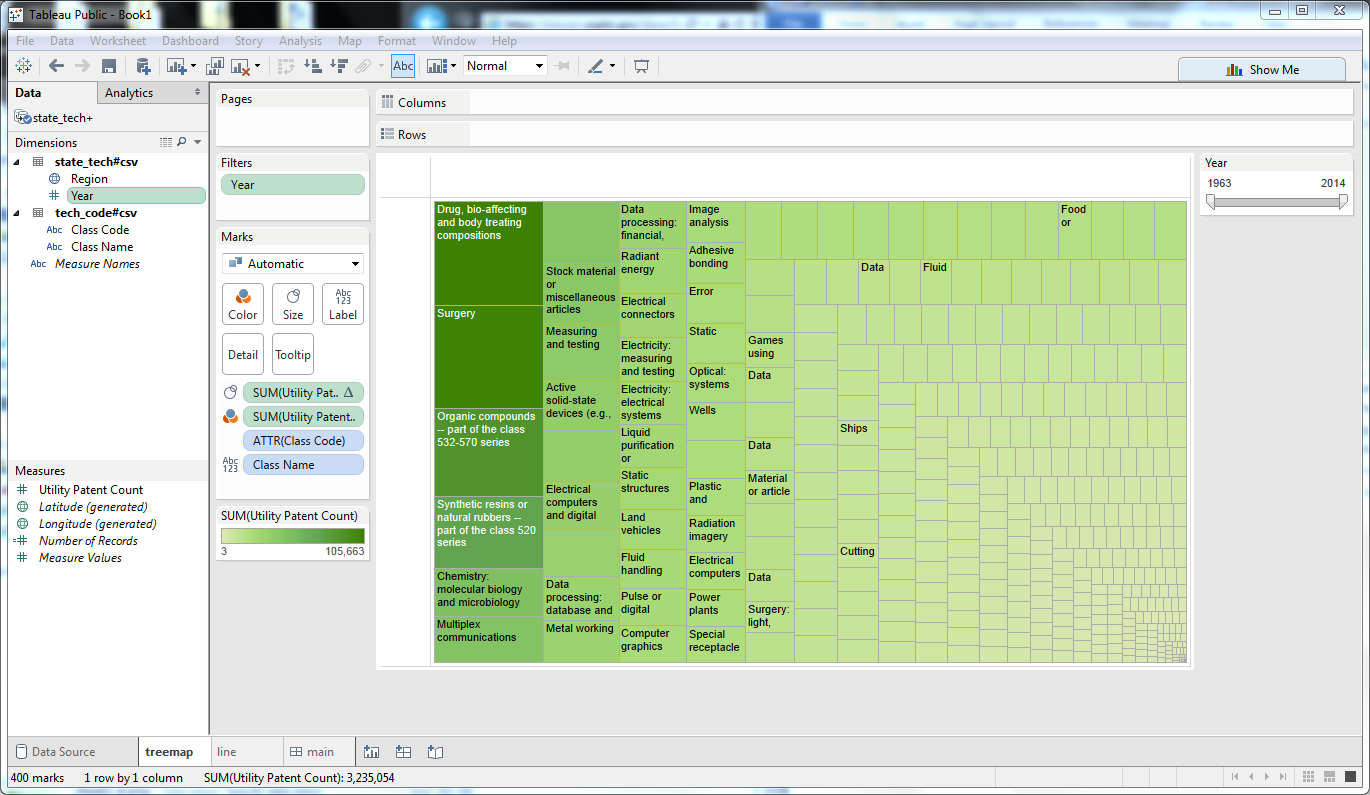
Now we can start putting these components together into a complete visualization. Click the second button to the right of the line tab at the bottom of the page to create a new dashboard, and rename it ‘main.’



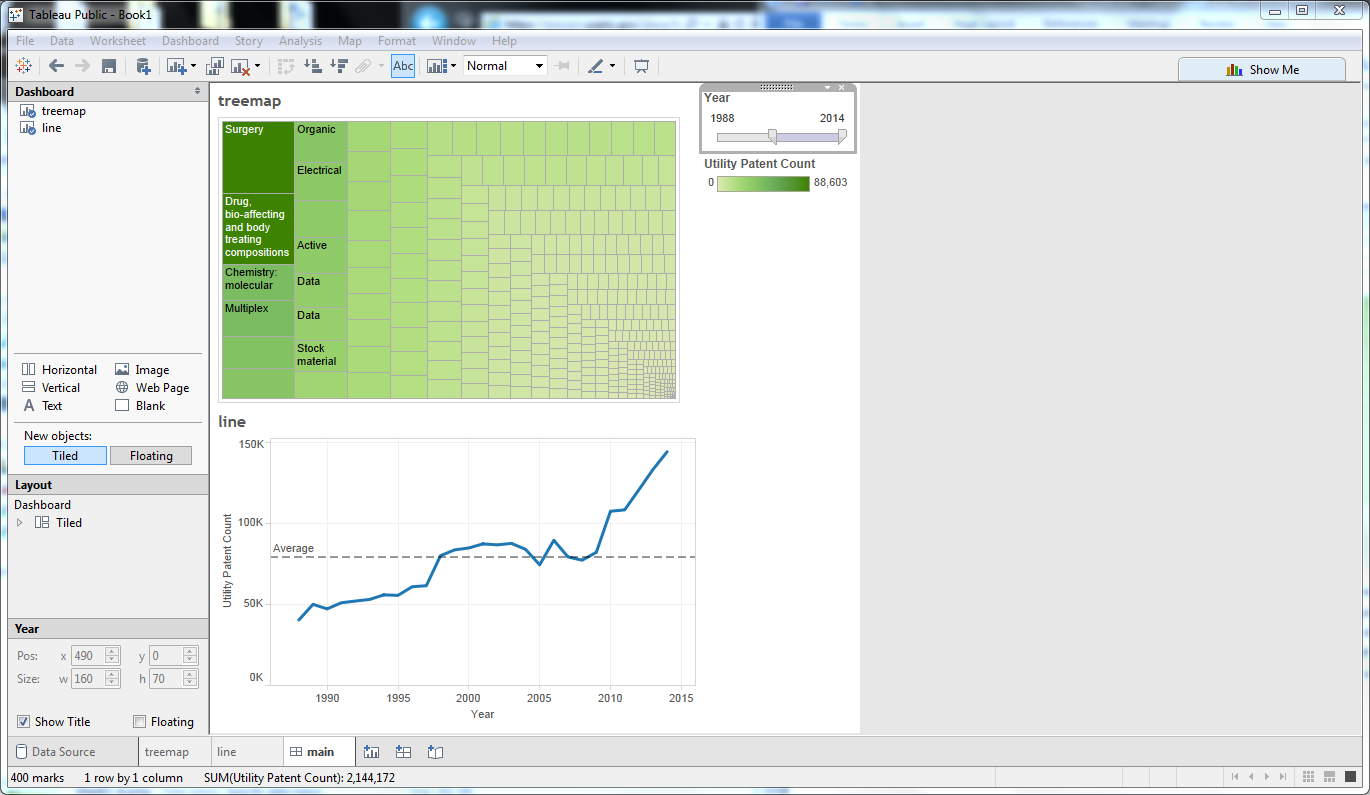
On the top left, we see a Dashboard section, with a row for each worksheet we created. In the middle is the dashboard pane. Drag the treemap row onto the pane, then drag the line row below the treemap sheet. As you drag a component onto the pane, a gray box will show you how it will be oriented. You want the gray box to be below the treemap component, not to either side. Your dashboard should look like this:



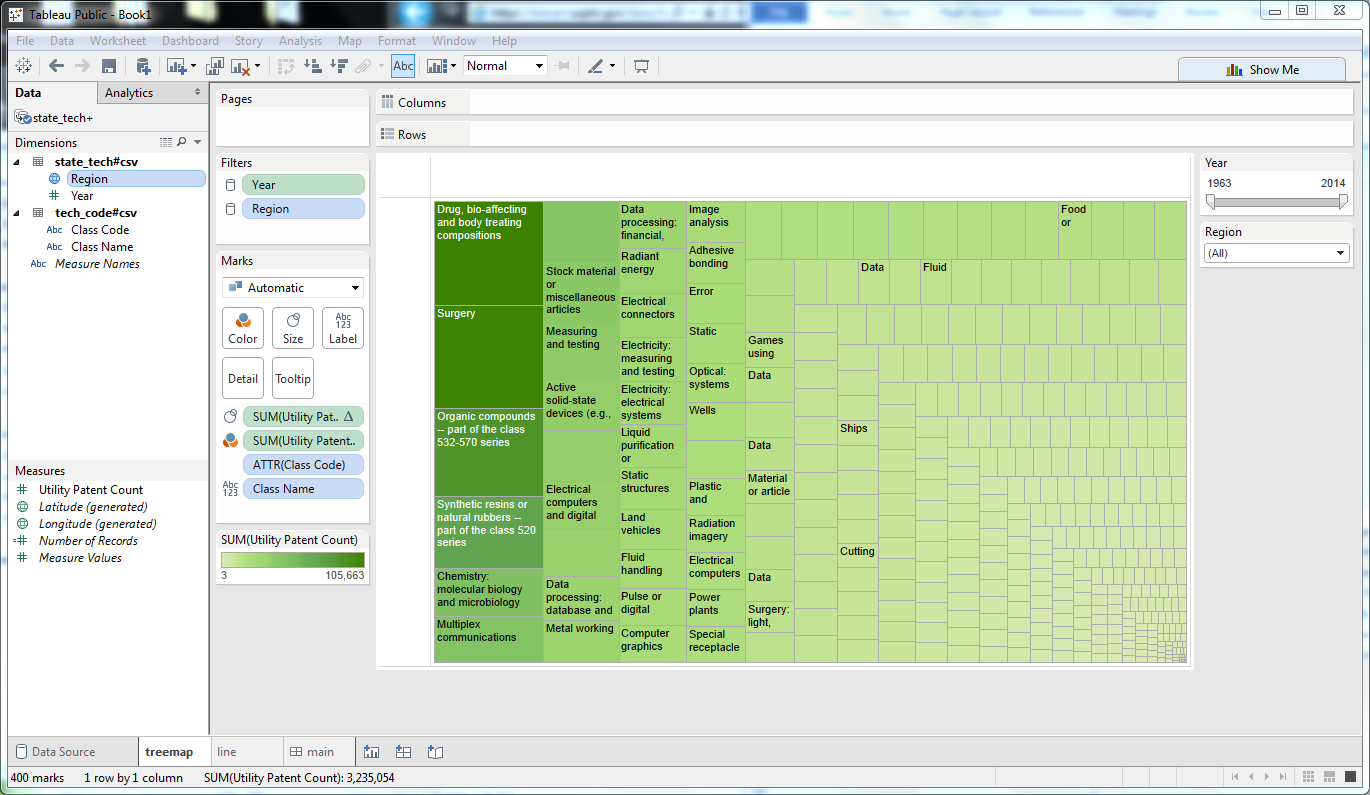
Looks good, but we’re not done yet! We want to allow the user to filter based both on State and Year, but we’re currently showing aggregations for all values, regardless of State and Year. To accomplish this, let’s return to the treemap worksheet. First, let’s create the Year filter. Right-click the Year row in the Dimensions section and select ‘Show Filter.’ Tableau adds a filter control that defaults to the full range of years. If we wanted our dashboard to default to a particular range of years on load, or to exclude certain values in the final product, we can set these bounds to anything we like. Let’s leave it at the default.



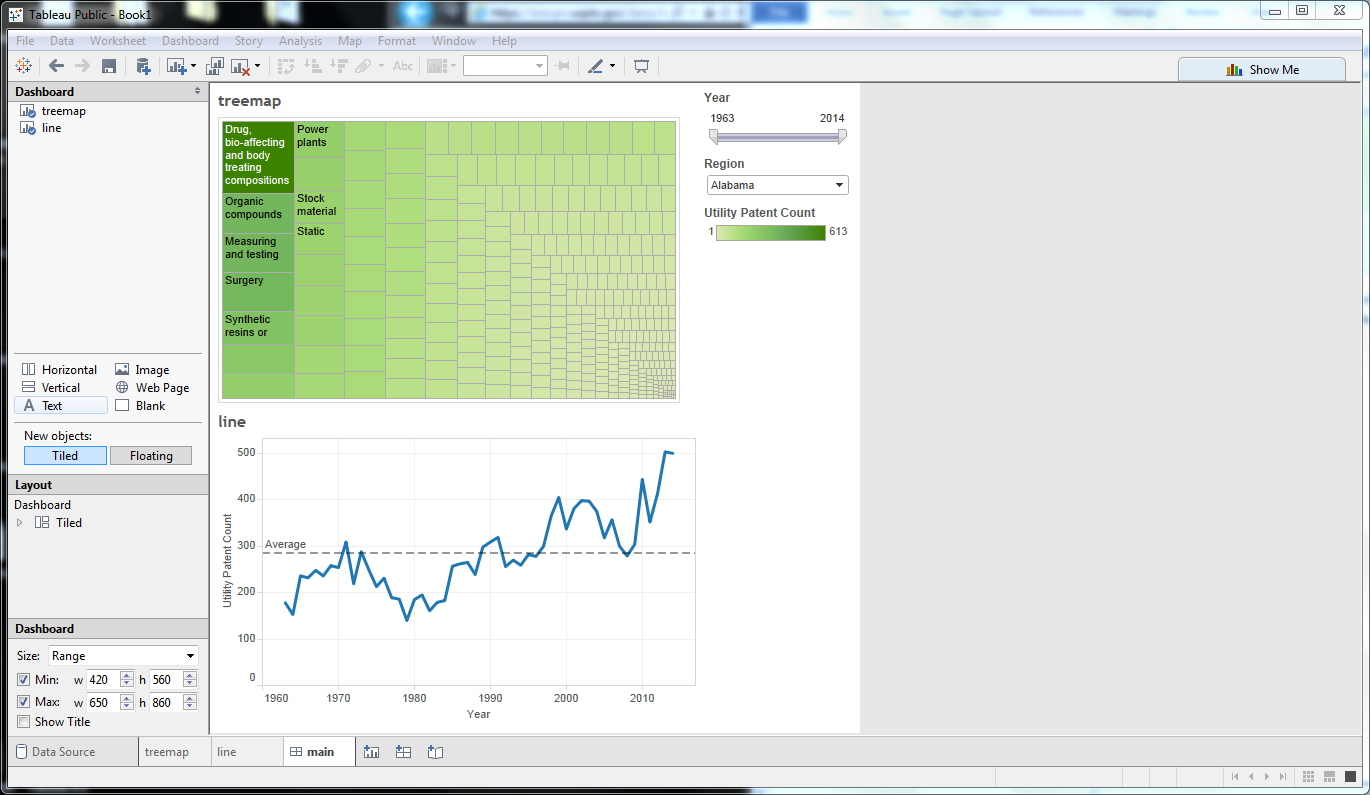
Right now, this filter only applies to the treemap worksheet. To apply it to the line graph as well, click the arrow in the upper-right section of the new Year filter control and select Apply to Worksheets -> All Using This Data Source. If you return to the dashboard, you’ll find that the filter doesn’t show up yet. The simplest way to refresh the dashboard is to drag the treemap component off the pane (removing it from the dashboard), then re-add it to the top of the dashboard. The Year Filter will then pop up. Drag it around a bit and confirm that it affects both the treemap and the line graph.



Now for the Region filter. Return to the treemap worksheet, right-click on Region and select Show Filter. By default, Tableau creates a multi-checkbox filter. Let’s adjust the Region filter component: click the arrow in the upper-right of the Region filter and select ‘Single Value Dropdown.’ Now we can select a region and see the technology class breakdown for that particular region, or select the ‘(All)’ value in the dropdown to clear out the Region filter. For now, let’s set the default to ‘(All).’ And, as before, we need to tell Tableau that we want this filter to apply to other worksheets using the same data source: click the arrow in the upper right of the Region filter and select Apply to Worksheets -> All Using This Data Source.



Now let’s go back to the main dashboard and refresh the treemap component by dragging it off the dashboard pane then dragging it back on. Drag the year slider around and set the Region filter to other values to confirm that it all works.



Looking great! There are some additional tweaks we could make, such as resizing the dashboard using the Dashboard section in the lower-left of the page, and either hiding or renaming the worksheet titles to something more meaningful than ‘treemap’ and ‘line,’ but making these changes is straightforward, . When you’re ready, select File -> Select to Tableau Public to start the publication process to the Tableau Public profile you created earlier. You’ll receive a log-in prompt for Tableau Public, then an opportunity to rename your workbook. Once saved, Tableau Public will open in your browser, and you’ll find the result of your hard work! Your visualization will be publicly available at this point. If you want to share your visualization with other people, either with a direct link or embedded into your own web page, check out the Share button on the bottom-right of the visualization.

If you liked this tutorial and would like to see more visualizations using USPTO data and Tableau Public, please visit https://developer.uspto.gov.  We'd love to hear from you and see what you’ve created!