Overview

The main goal of our project is to provide fellow data visualization students with a brief web-based walkthrough of creating their own Voronoi treemap. To create a successful tutorial, our team needed to learn the fundamentals of Voronoi treemaps and apply our knowledge by implementing Voronoi treemaps using D3.

Motivation

The inspiration for this project's design extends from our results from assignment 3. Our original idea for assignment 3 was to assess individuals' accuracy in comparing ratios within bubble charts featuring various shapes such as circles, squares, and polygons. After discussing our idea with Professor Harrison, we decided to shift our focus towards comparing accuracy in ratio comparison within tree maps and Voronoi treemaps instead.

There are no standard D3 libraries available for Voronoi treemaps which is why Professor Harrison recommended investigating how to construct Voronoi graphs using D3. We encountered several complex examples of how to do this on GitHub, Stack Overflow, and Observable.

For our final project, we wanted to continue in our pursuit of Voronoi treemaps because it was a visualization none of us were familiar with and were interested in investigating further. Because this data visualization is not very popular, Professor Harrison suggested creating a tutorial tailored for data visualization students, aimed at teaching them how to develop Voronoi visualizations for themselves as well as giving examples to explain different use cases.

Related Work:

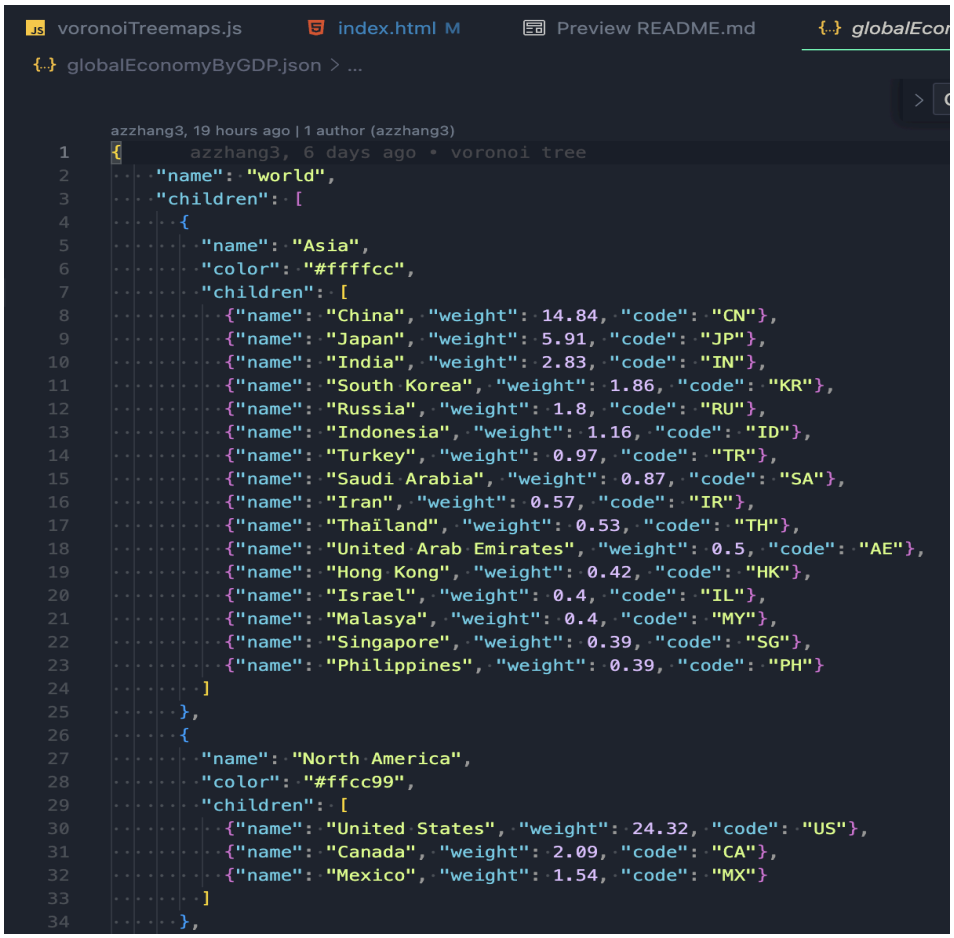
We were inspired by the different visualizations of treemaps. For A3, we experimented specifically with treemaps, so we felt compelled to take the concepts we learned into the final, transitioning from the experimental aspects of Voronoi treemaps to a more holistic approach, helping people understand and realize the capabilities of these amazing visualizations. The examples we encountered while researching for A3 and the final were extremely high-level, and we felt that we wanted to introduce the Voronoi treemap topic more simply. It was more important for us to portray how the graphs were built as opposed to jumping to an overwhelming example which was too difficult to explain.

Questions:

In our exploration of Voronoi treemaps, we were particularly interested in how many hierarchies can be effectively conveyed using this visualization technique, identifying the best use cases for Voronoi treemaps, and assessing their effectiveness in communicating datasets. While completing this project, another question asked was “What information is necessary to teach these concepts to new users?” This became the basis for our web page design.

Data:

The dataset selected for this project was global GDP data categorized by country. This particular dataset was chosen for its simplicity to effectively illustrate the function of Voronoi treemaps. This dataset also allowed us to showcase how hierarchies can be created within Voronoi treemaps. The countries in this dataset are grouped by continent for visualization.



Exploratory Data Analysis:

Our project mainly focuses on Voronoi treemaps, so we started by looking at various treemaps and comparing them. We gained a lot of insight from this primarily about how many different ways we can make treemaps. These insights helped inform our design as we created a website, that is beginner-friendly, but if they want to explore and investigate it more, that is also an option, as we provide them with links, videos, and libraries they can look into.

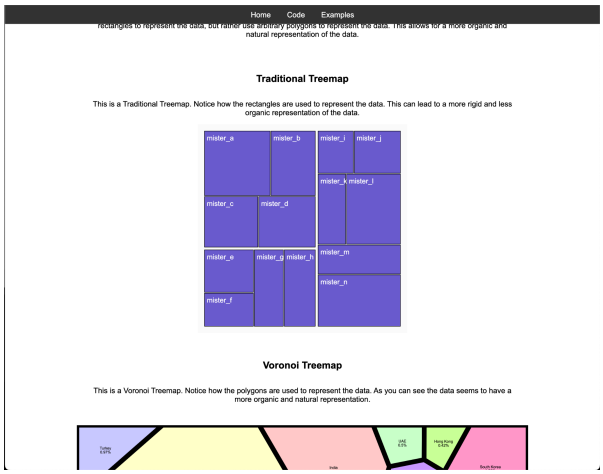
Design Evolution:

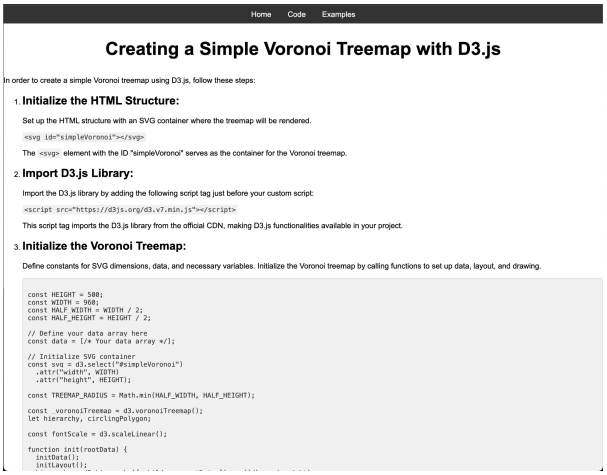
Since our main focus for this project was Voronoi treemaps, we didn't consider any other visualization options. In our design we wanted to make the text labels clear and visually appealing, using categorical concepts and design principles. We did deviate a little from our proposal, as initially, we were thinking more about the possibilities of displaying various data with treemaps. We concluded with Professor Harrison, that a tutorial would be more beneficial and interesting to create.

Implementation:

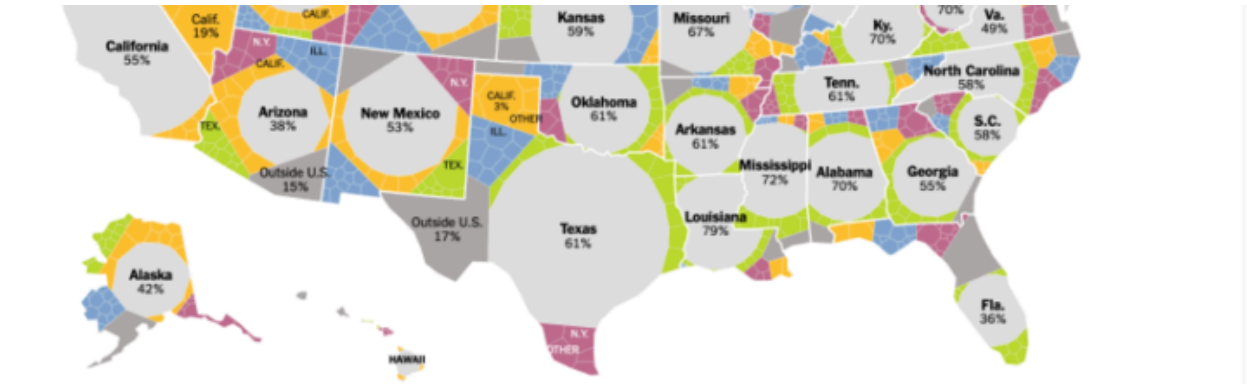
- Create the initial Voronoi graph for the home page to figure out the sizing for the page. Here there is no special design and the data is irrelevant to the final product.

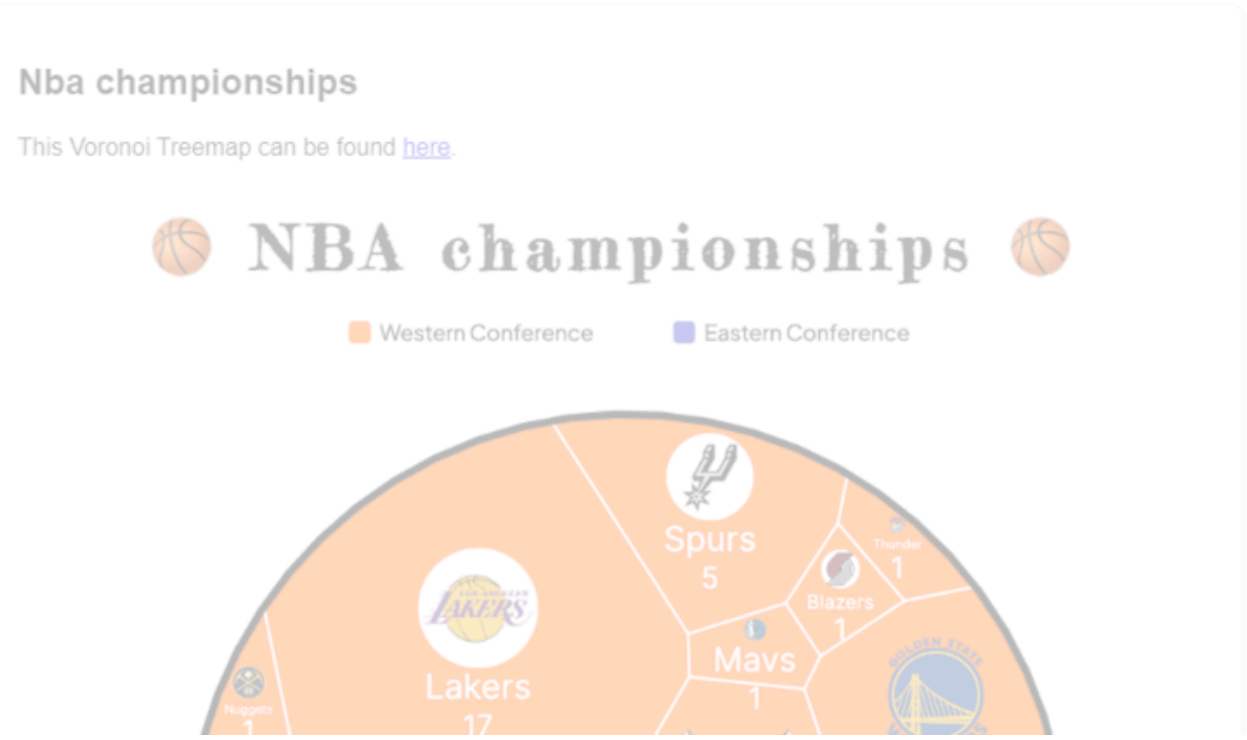
- The initial page structure was different as we experimented with different design layouts (home, code, color, examples)





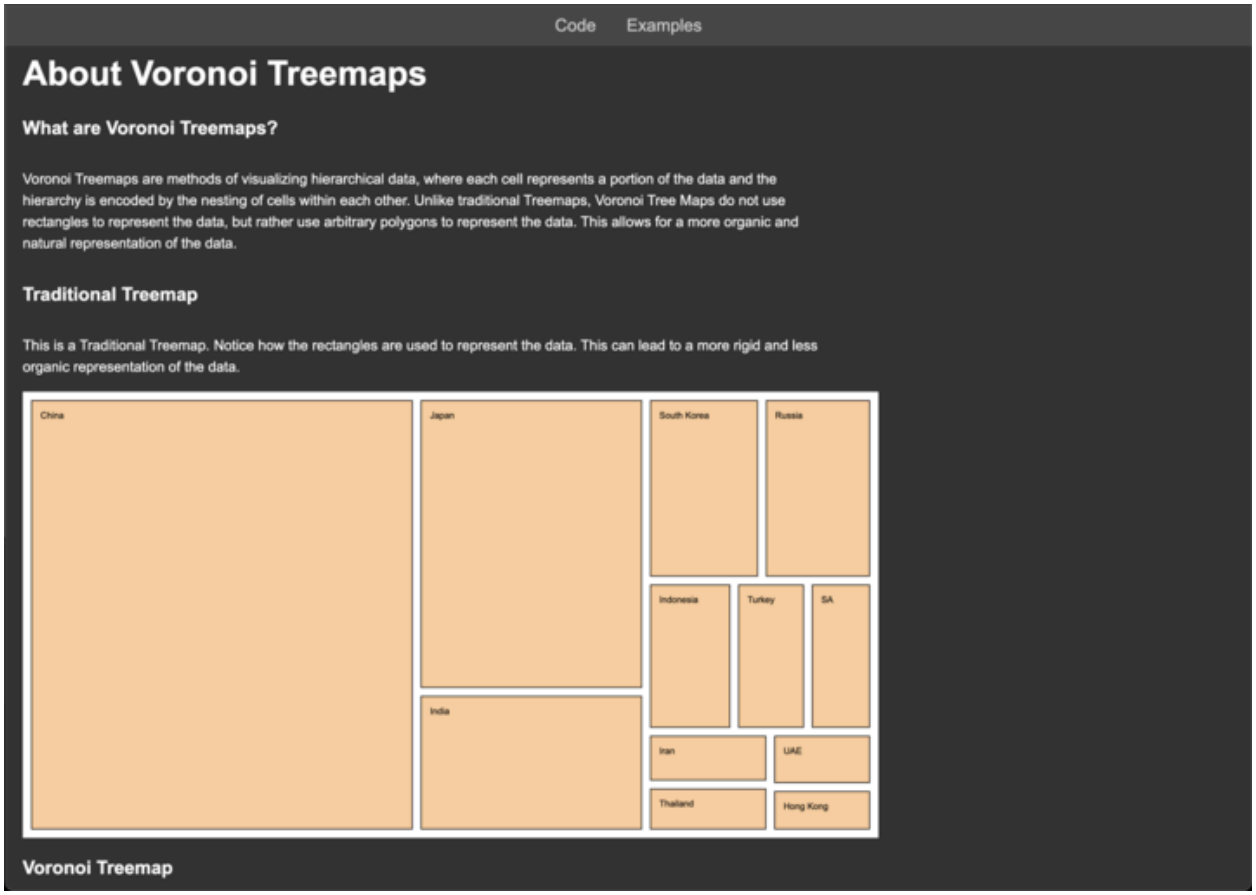
- Implementing fade animations when scrolling





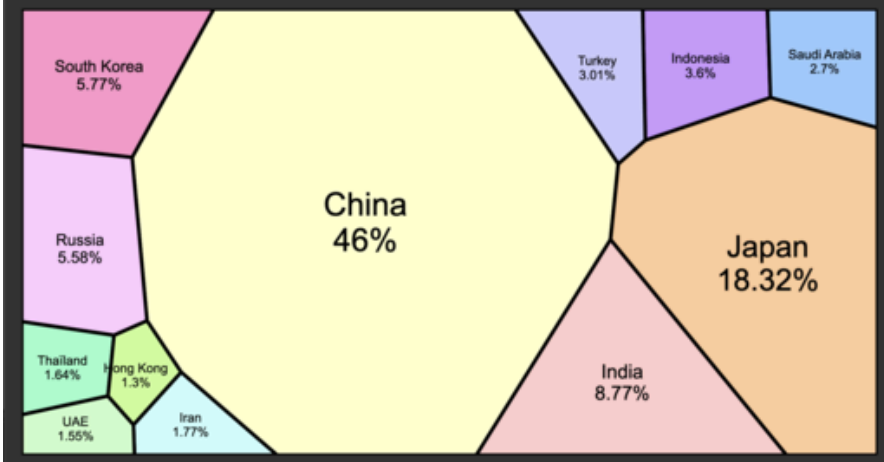
- The final version, complete and adaptable with light and dark mode browsers, properly spaced, code blocks for the tutorial, colorful treemap examples, embedded links, videos, details regarding the imported libraries, the “how and why” for the math behind a Voronoi treemap, and other sources showing the

possibilities of the concepts in the webpage.



Code Examples

This is a Voronoi Treemap representing the GDP of countries in asia. Notice how the polygons are used to represent the data.
As you can see the data seems to have a more organic and natural representation.



Creating a Simple Voronoi Treemap with D3.js

In order to create a simple Voronoi treemap using D3.js, follow these steps:

1. **Initialize the HTML Structure:**

   Set up the HTML structure with an SVG container where the treemap will be rendered.

   ```
   <svg id="simpleVoronoi"></svg>
   ```

   The `<svg>` element with the ID "simpleVoronoi" serves as the container for the Voronoi treemap.

Code Examples

Creating a Simple Voronoi Treemap with D3.js

In order to create a simple Voronoi treemap using D3.js, follow these steps:

1. **Initialize the HTML Structure:**

   Set up the HTML structure with an SVG container where the treemap will be rendered.

   `<svg id="simpleVoronoi"></svg>`

   The `<svg>` element with the ID "simpleVoronoi" serves as the container for the Voronoi treemap.

2. **Import D3.js Library:**

   Import the D3.js library by adding the following script tag just before your custom script:

   `<script src="https://d3js.org/d3.v7.min.js"></script>`

   This script tag imports the D3.js library from the official CDN, making D3.js functionalities available in your project.

3. **Initialize the Voronoi Treemap:**

   Define constants for SVG dimensions, data, and necessary variables. Initialize the Voronoi treemap by calling functions to set up data, layout, and drawing.

```
const HEIGHT = 500;
const WIDTH = 960;
const HALF_WIDTH = WIDTH / 2;
const HALF_HEIGHT = HEIGHT / 2;

// Define your data array here
const data = [
{ name: "China", weight: 46.0, color: "#ffffcc" },
  { name: "Japan", weight: 18.32, color: "#ffcc99" },
  { name: "India", weight: 8.77, color: "#ffcccc" },
  { name: "South Korea", weight: 5.77, color: "#ff99cc" },
  { name: "Russia", weight: 5.58, color: "#ffccff" },
  { name: "Indonesia", weight: 3.6, color: "#cc99ff" },
  { name: "Turkey", weight: 3.01, color: "#ccccff" },
  { name: "Saudi Arabia", weight: 2.7, color: "#99ccff" },
```

Voronoi General Background Processes

1. **Scatter Plot:**

A voronoi diagram starts with a set of 2d coordinate points plotted on a plane.

2. **Delaunay triangulation:**

Run a Delaunay triangulation on the set of two-dimensional points

3. **Circumcenters:**

Code Examples

How Voronoi Treemaps are generated

The d3-voronoi-treemap.js library

This code sets up a module that can be used in both CommonJS and AMD (Asynchronous Module Definition) environments. It handles dependency injection, allowing the module to work with various module systems. The module takes exports and d3VoronoiMap as dependencies. It then passes these dependencies to the factory function.

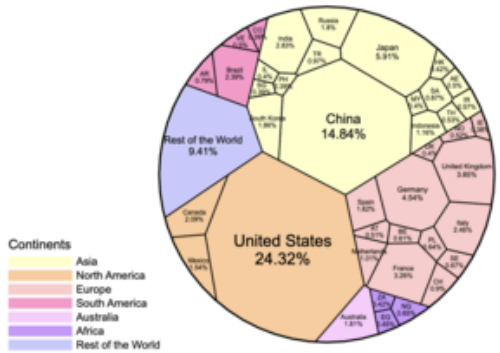
`_voronoiTreemap` is the main function representing the Voronoi treemap algorithm. Additional functions are defined on `_voronoiTreemap` to get or set various parameters such as `convergence ratio`, `max iteration count`, etc.

`recurse` is a private function used internally to generate the Voronoi treemap recursively. It assigns the clipping polygon to the node and computes the Voronoi map for its children recursively.

```
(function (global, factory) {
typeof exports === 'object' && typeof module !== 'undefined' ? factory(exports, require('d3-vo
typeof define === 'function' && define.amd ? define(['exports', 'd3-voronoi-map'], factory) :
(factory((global.d3 = global.d3 || {}),global.d3));
}(this, function (exports,d3VoronoiMap) { 'use strict';
```

Code Examples

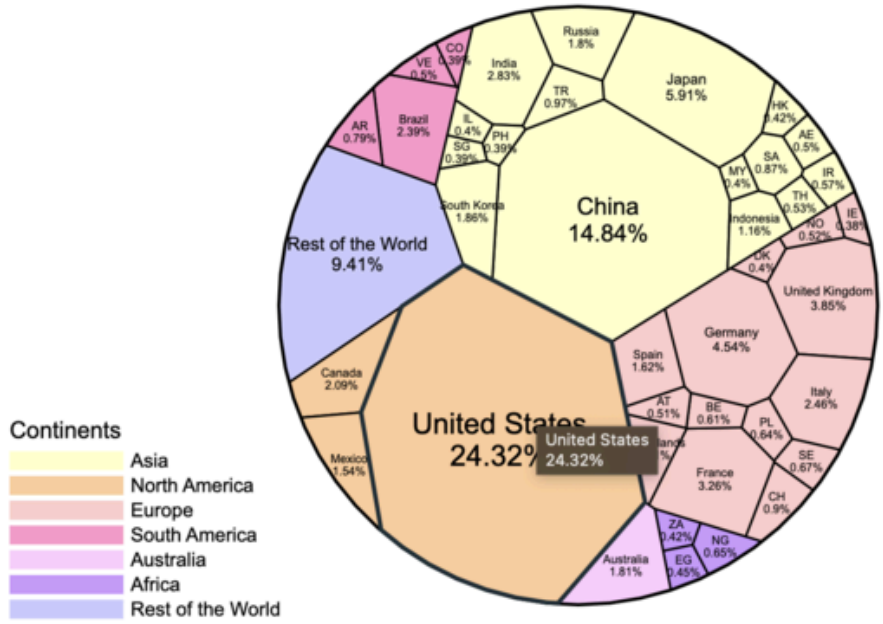
GDP by Country (Interactive)



Vornoi geography map hybrid

This Voronoi Treemap is from the following article.

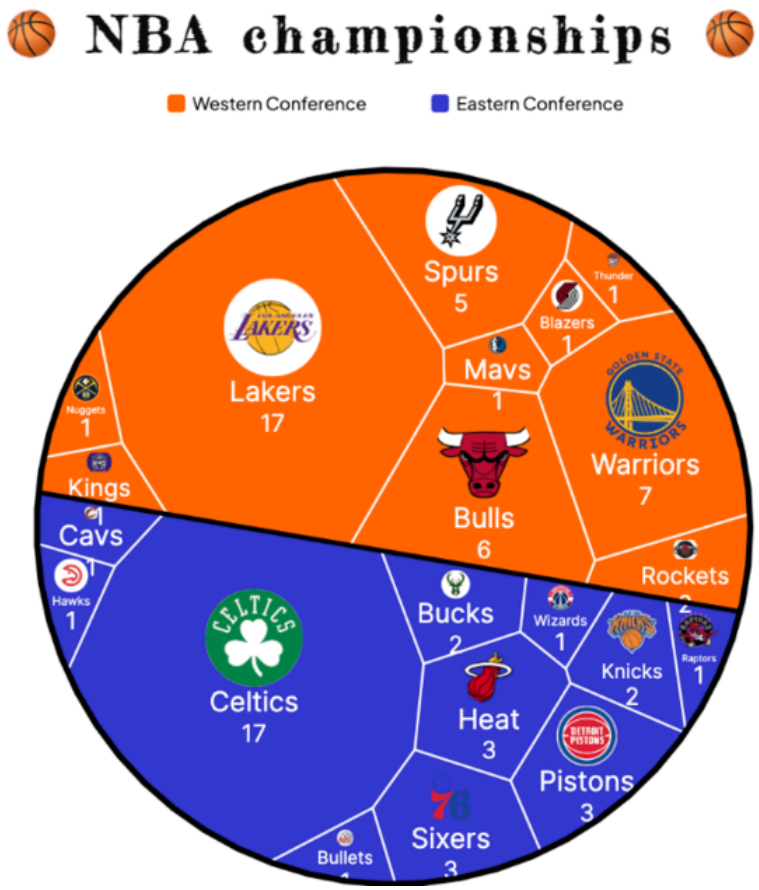
GDP by Country (Interactive)



This Voronoi Treemap is from the following article.

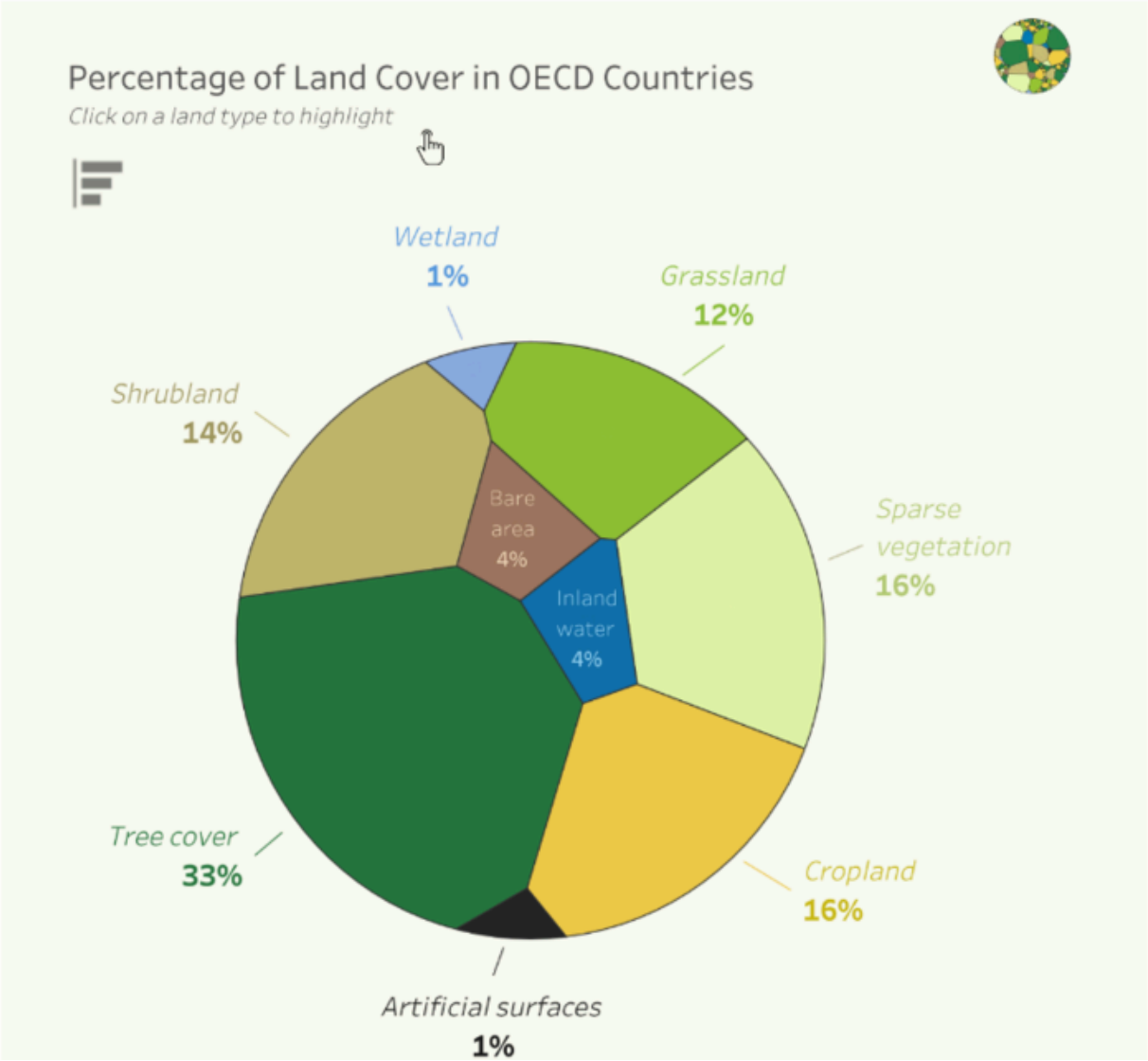
Nba championships

This Voronoi Treemap can be found here.



Land cover by country

This Voronoi Treemap can be found here.



Evaluation:

When working on this project we realized that there are so many different datasets that would be worth experimenting with on a Voronoi treemap. We also learned that advanced implementations are capable of including world map features or images as well as having a clear connection to the anatomy of a cell, and can be easily used for biology maps as shown above. We believe that the webpage tutorial provides helpful insight into the introduction of a Voronoi treemap, and if we had more time to

work on this webpage we and the professor agree that testing its teachability and learnability with peers for feedback would take this final project to the next level. If we were to experiment with students on what they would like to see in a tutorial website (say, make it similar to Codecademy?) we could reverse engineer the experiment into a fully loaded and revamped walkthrough of Voronoi treemaps.