ctresc Update README.md c88b3c7 a minute ago

3 contributors

148 lines (84 sloc) 13.4 KB

Data Visualization: Real Estate Map Project

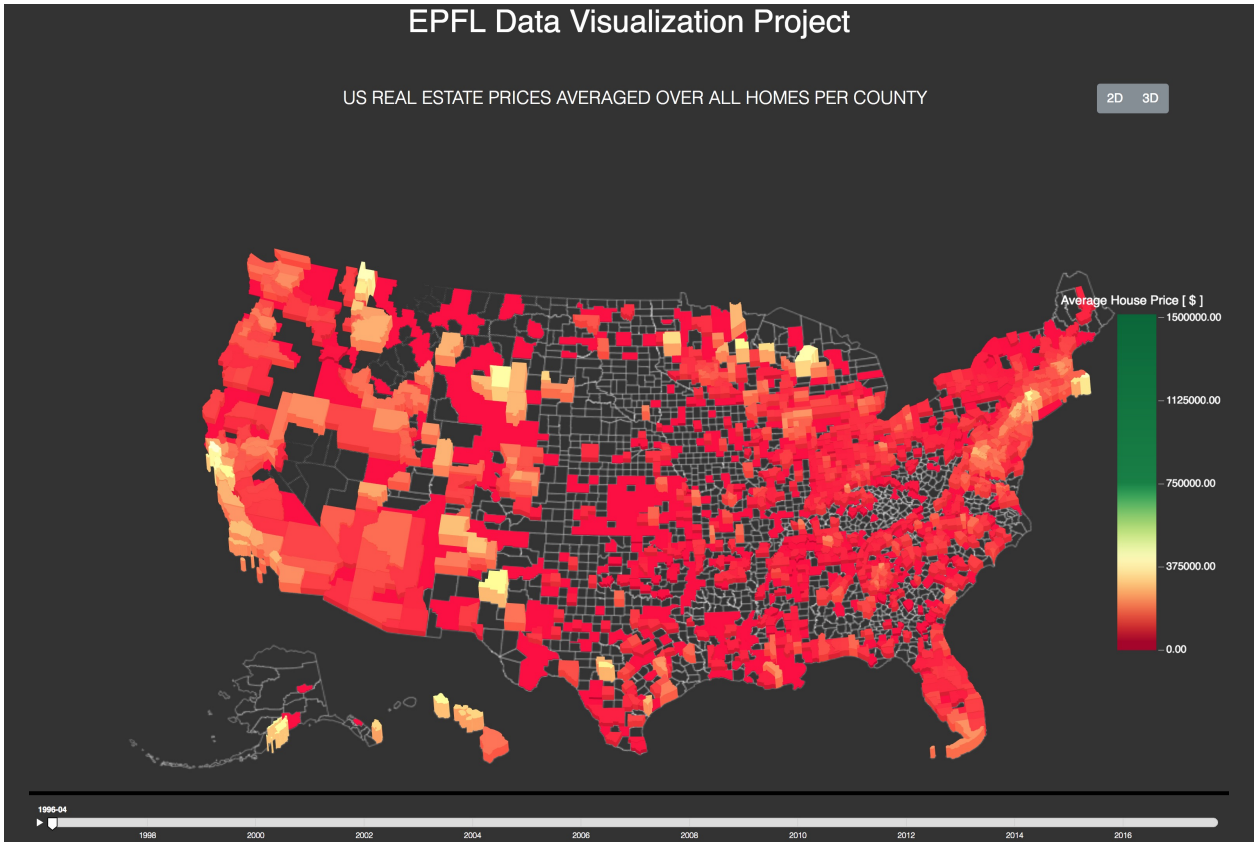
PROCESS BOOK

Process Book of Data visualization project at EPFL involving US housing market data.

Team:

- Juraj Korcek
- Mateusz Paluchowski
- Christian Tresch

Visualization: Website - Real Estate Map Project



Screencast: Screencast - Real Estate Map Project

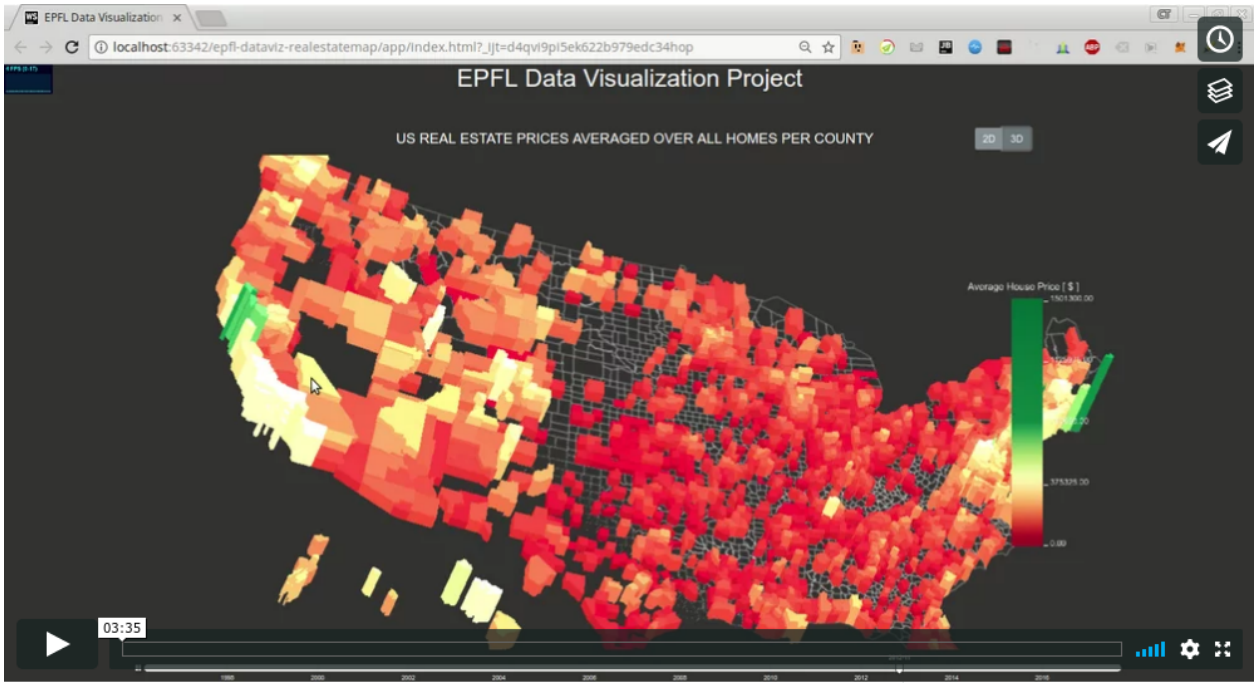


Table of Contents

Overview

The goal of the visualization project was to present data extracted from the US real estate market to visualize the average house price over all different house sizes within a corresponding county from a dataset with time series data, spanning over the past 20 years.

Motivation

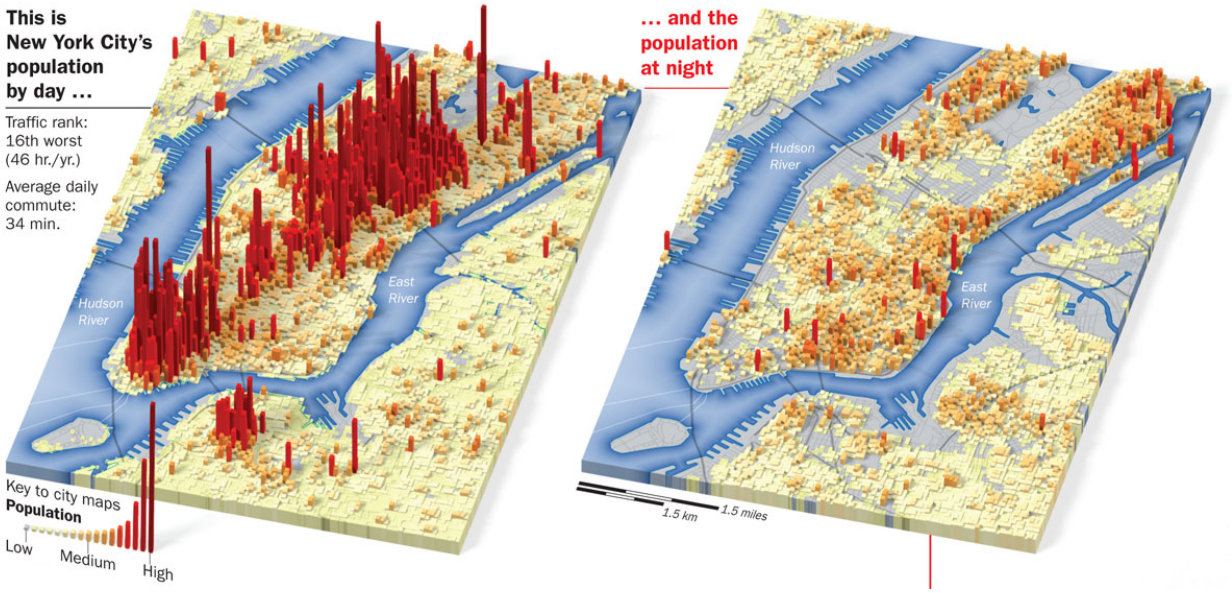
The motivation behind the project was to get a broad overview as well as insights on changes in the housing market in order to help potential investment decisions in the future based on historical data. As a side goal our motivation was to be able to see the crash of the 2008 housing market which can clearly be seen dropping until about 2010 and recovering from then on.

Target Audience

As a target audience we have identified real estate investors and market analysts.

Related Work and Inspiration

We have been inspired by web platforms such as www.trulia.com and www.zillow.com and have a general interest in real estate market data. Moreover, the follwing visualization was particularly inspirational:



Questions To Answer

What am I trying to show in the data viz? Changes in the housing prices of the US between 1996-04 and 2017-09.

Dataset

Time series representing prices of household in various US-located zip codes/counties, spanning across aforementioned dates with one month resolution. https://www.zillow.com/research/data/

Exploratory Data Analysis

Initial Data Analysis consisted of quick and rudimentary data check performed in python, with use of Jupyter Notebooks. Basic histogram plotting amount of data points for each state was performed, as well sanity check for amount of NaN values though the spanning dates for each household. Last but not least, we plotted data in time on simple 2D line plot (for each state) in order to make sure there is 'some story to tell', which we verified is the case, however we want the reader to see it for him/herself. We have also checked the distribtuion of the values. Thanks to this we have found several outliers that would skew a default colormap. Based on this, we have decided to create our own colormap that is robust to outliers.

Designs & Deviations

Initial design

Initial design of the visualisation considered using D3.js + Leaflet.js as it is the most common tool for map based visualizations. In order to achieve the 3D bars effect, OSMBuildings JavaScript library was used, however with partial success. As the name of the library suggests, it is designed to visualise 3D buildings on a Leaflet map, using provided TopoJson of the buildings. We tried expanding this approach to '3D-fing' entire states, however we stumbled upon the issue where 3D bars were only visible up to certain zoom levels of the map and disappeared in full country perspective. This issue would require a deep dive into both Leaflet.js and OSMBuildings frameworks, in order to fix it should that be possible, thus we decided to find some other approach.

Second Design

In our second design we considered using pure D3.js and provided topology in TopoJson format. It would require from us to write a custom TopoJson modifier such that each state/county could be '3D-fied' in a way such that it would be 'lifted' as a bar above all the rest, given the current projection on the map. Such solution would be labour intensive and potentially visually unappealing if done incorrectly, thus we quickly decided to use another approach.

Third Design

In our third - final - design we settled on using D3.js alongside a dedicated 3D framework Three.js.

We have decided to go with county granularity, as zip granularity was too fine-grained (information overload) while state granularity was too coarse.

Implementation

Describe the intent and functionality of the interactive visualizations you implemented. Provide clear and well-referenced images showing the key design and interaction elements.

In this part we will describe the route that lead us to the final solution - step-by-step.

We have started with US county outline in the TopoJson format. We needed to convert this to SVG for visualization purposes. For that we decided to use topojson-svg tool that is part of topojson library. However, as topojson-svg hasn't been ported yet to the newest version of the topojson library we had to use the old one - 1.6.27 in particular.

Once we had the svg we have used three.js library to extrude it into columns. We have drawn inspiration from example found at https://threejs.org/examples/webgl_geometry_extrude_shapes2.html. However, there were problems with triangulation resulting in holes in the columns. After quite some investigation, the problem was solved by using not yet officially released version of three.js - technically, a 5-days-old nightly build.

Afterwards, we have added legend based on the example found at https://threejs.org/examples/webgl_geometry_colors_lookuptable.html. This was not without problems either - the legend title and tick labels were missing. After some investigaion, we have found out that unlike ticks, the tick labels and legend title do not change their size automatically according to the legend automatically. Thus, we had to make up for this missing library feature manually. Also, even though the library allows choosing font size and font type it does not allow choosing font color. Therefore, we had to dig into the library and set the font color there. As there are only 4 colormaps in three.js library by default we have devised a method that transforms a d3.js colormaps into three.js colormaps. In the process we have used https://github.com/d3/d3-scale-chromatic library to get continous d3.js colormaps. In the end, we had to create our own colormap anyway in order to account for outliers.

We have used OrbitControls (part of three.js library) to be able to rotate, pan and zoom the map. Then, we have followed the example https://threejs.org/examples/webgl_sprites.html to find out how to make part of scene movable and part of scene static. We needed this as we did not want the legend to rotate with the map nor to be affected by zoom or pan.

Thanks to the choice to use orthographic camera, creating 2D <-> 3D switch was pretty trivial. Switch to 2D just moves the camera on top of the map, such that camera's angle is aligned with map's normal.

For animations we have used the tween.js library. With animations the FPS becomes of interest as well. In order to quantify smoothness of our animations we have used stats.js library. This also allowed us to quantify benefits of various optimizations. And optimization was indeed needed. Initially, we depicted change in average price of the house between months by animating change in color and animating the column "growth". However, we have found out that three.js is not optimized for object growing, as we had to animate the change in a vertex-by-vertex fashion and by doing so the FPS dropped to 0, which was useless. Therefore, we had to find some workaround. And we indeed found one. At initialization, we have found the highest value in our data and initialized height of every column to this value and during animation we were changing just the z coordinate of whole column - three.js is well optimized for this (no more need to animate every vertex separately). By this we were able to get up to 9 fps. The fact that the columns are bigger then they should be is not a problem, as we hid the excessive height below a ground plane and thus this is not visible to the user. In other words - the outcome is the same, just we got 9 fps instead of 0.

In order to give user an option to animate the change in prices between months we have used chroniton library. It offers a play button for an automatic animation. However, it also allows user to drag the slider and choose precisely which month he desires to see. This library also did not fit our needs exactly and thus we had to dig in and make some changes. The problem was that it was using d3.timer which was firing as fast as possible - this was not compatible with our usecase as our animation was quite computationaly intensive and lasted longer then one tick of this timer. Therefore we have replaced d3-timer in this library with d3-interval which allowed us to define the interval between ticks precisely. Also, this library was, by default, firing update method on brush event. This was again not desired due to our animation being too computationally intensive. Therefore we have changed the library such that the update event is called only on brushend event, i.e. when the user releases the slider. Also, we had slightly changed he functionality play button - when animation ends and user clicks the playbutton again, the animations restarts from start. At last, we have changed the text color to white in order to fit our color scheme.

We have used bootstrap for basic UI styling.

Evaluation

What did you learn about the data by using your visualizations? How did you answer your questions? How well does your visualization work, and how could you further improve it?

Being inspired by the provided NYC population map, initially we wished to visualize each zip code, however due to performance issues we settled on using counties instead. Zip code TopoJson simply consists of way too many arcs and any transformations are CPU intensive resulting in chopped animations.

Peer Assessment

Mateusz Paluchowski

- Preparation – were they prepared during team meetings? Both Juraj and Christian were always ready for the team meetings and have prepared beforehand such that any issues which were needed to be discussed, were resolved at hand without the need to explain anything.
- Contribution – did they contribute productively to the team discussion and work? Christian was extremely helpful and contributed a great deal with both dataset knowledge and D3 expertise. Juraj proved to be irreplacable particularly when it comes to Three.js code and general development know-how.
- Respect for others' ideas – did they encourage others to contribute their ideas? We had many different visions and initial ideas, yet we still managed to convarge tow a mutual goal. Noone was ever disrespectful or hostile towards others and their ideas.
- Flexibility – were they flexible when disagreements occurred? Without flexibility we would have not manage to finish this project at all simple because how different initial visions everyone had.

Christian Tresch

- Preparation – were they prepared during team meetings? Both of my team mates were very active and always available to discuss ideas, problems and milestones along the project. They have prepared for each meeting so we were able to keep the meetings efficient and productive.
- Contribution – did they contribute productively to the team discussion and work? Mateusz was very important to the team, which he has enriched with his knowledge in web development, JavaScript and different D3.js approaches. He has consistently pushed the project forward and has quickly adapted to team decisions. He has contributed very important parts of the code and ideas in the project and helped to reach and finish everything we had envisioned. Juraj was very important to the team, for which he has shared valuable development experience, has come up with great ideas and has solved critical performance issues. He has implemented major parts of the platform and contributed invaluable efforts in creating the 3D environment and data connection to the framework and has consistently solved critical problems in the domain to make the project a success.
- Respect for others' ideas – did they encourage others to contribute their ideas? With initially many different ideas and approaches we were able to reduce the space of ideas in a fair and respectful way and by friendly arguments where everybody has contributed to equal parts and felt that their ideas where respected and also implemented after mutual agreement.
- Flexibility – were they flexible when disagreements occurred? Flexibility very important throughout the hole project since many ideas and approaches where born and discarded along the way. Everybody has been kind and respectful in treating and evaluating the others ideas.