

Correlation Between NYC Trees and Property Prices

Introduction

A series of studies have shown that trees increase property prices by between 5% to 18% (CABE Space, 2005 and Morales et al, 1983). A Philadelphia-based study demonstrated that properties close to new tree plantings increased in price by about 10% (Wachter and Gillen, 2006).

In our project, we tried to find a correlation between property prices and number/ health/species of the trees in NYC neighborhoods and boroughs, and then visualize it.

We analyzed the NYC Tree Census Data (2015) and Property Valuation and Assessment Data (2015/2016). For each borough and neighborhood, we calculated several values, such as number of trees per square mile, average trees' health, trees' species distribution, average land/properties price, number of properties per square mile, etc.. Our goal was to show that there is a correlation between trees and properties values in NYC.

Showing this correlation would help people searching for a new property to evaluate prices accordingly. For some people trees around the property are not a priority. So if two places have the same price, but one has many trees around, this person would probably choose the other property, knowing that trees increase the price.

Consolidated datasets, python scripts and Dash code can be found on the GitHub:

https://github.com/MarinaOrzechowski/NYC_Trees_Properties

Data Analysis

We used the NYC Open Data site to retrieve data about trees and properties. Because datasets for different years had different fields, and it was difficult to compare them, we decided to only use information from 2015/2016.

Trees data

Trees data was taken from [NYC Tree Census Data](#). It contained 684 thousands of rows and 45 columns. We decided to leave only 5 columns: 'borough', 'species', 'health', 'longitude', 'latitude'. The rest of the fields described the size of trees, health issues of different parts of the tree.

The next task was to find neighborhood names based on tree coordinates. We used [carto.com](#). This site allows you to upload a geojson file with area borders and a file with longitudes and latitudes of different objects. After uploading, we were able to run a query and retrieve neighborhoods names for each tree

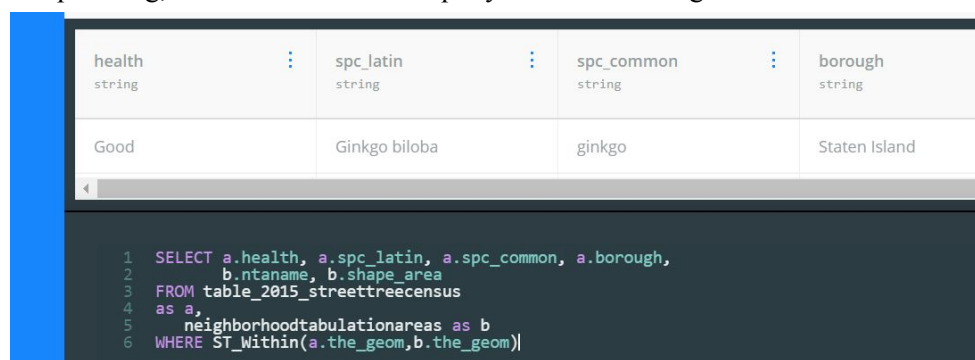


Figure 1. Use of [www.carto.com](#) to retrieve trees neighborhoods based on coordinates

The next step was to convert ordinal categorical health data ('good', 'fair', 'bad', 'dead') into numerical values, where 'good' health got score 15, 'fair' - 10, 'poor' - 5, and 'dead' - 0. After that data was grouped by boroughs and neighborhoods, and four values were calculated for each neighborhood:

	ntaname	borough	count	shape_area	health	trees/sq.mile
0	airport	queens	155	8.185520	14.322581	18.935877
1	allerton-pelham gardens	bronx	3609	1.136832	13.941535	3174.612844
2	annadale-huguenot-prince's bay-eltingville	staten island	12538	5.059752	13.418408	2477.987095
3	arden heights	staten island	6729	1.808239	13.910685	3721.299694
4	astoria	queens	4183	1.410967	14.040163	2964.632633
...
192	windsor terrace	brooklyn	2030	0.503676	14.098522	4030.372441

Figure 2. Trees grouped by neighborhoods

Separately, we found percent of species in each neighborhood:

Out[188]:

			count	area	total	health	trees/sq.mile	speices %
ntaname	borough	spc_common						
airport	queens	american hornbeam	5.0	8.18552	155.0	14.322581	18.935877	3.225806
		amur maple	2.0	8.18552	155.0	14.322581	18.935877	1.290323
		black locust	8.0	8.18552	155.0	14.322581	18.935877	5.161290
		callery pear	9.0	8.18552	155.0	14.322581	18.935877	5.806452
		cherry	3.0	8.18552	155.0	14.322581	18.935877	1.935484
		crab apple	1.0	8.18552	155.0	14.322581	18.935877	0.645161
		dawn redwood	1.0	8.18552	155.0	14.322581	18.935877	0.645161
		english oak	2.0	8.18552	155.0	14.322581	18.935877	1.290323
		ginkgo	1.0	8.18552	155.0	14.322581	18.935877	0.645161
		golden raintree	3.0	8.18552	155.0	14.322581	18.935877	1.935484
		hawthorn	2.0	8.18552	155.0	14.322581	18.935877	1.290323
		honeylocust	50.0	8.18552	155.0	14.322581	18.935877	32.258065
		japanese tree lilac	1.0	8.18552	155.0	14.322581	18.935877	0.645161

Figure 3. Trees species grouped by neighborhoods

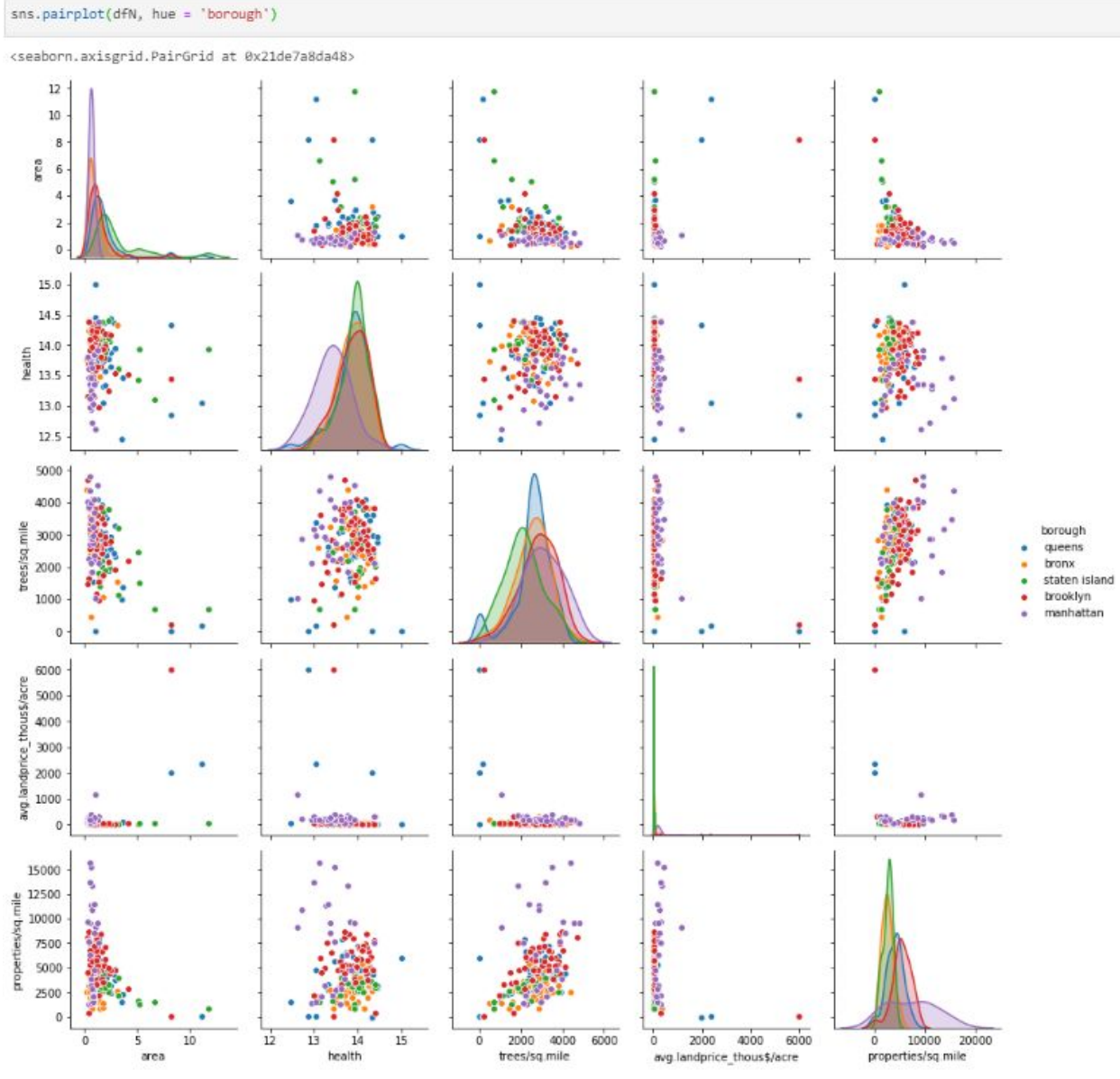
Properties data

Data was taken from [Property Valuation and Assessment Data](#). Initially, it contained over 9 million rows and 39 columns. We only left data for 2015/2016 years, which reduced the number of entries to 1 million.

Same as with trees data, we needed to know neighborhoods for each property. We used carto.com again. This time we had to divide data into batches as the site's platform has a limit on the size of analyzed data. After retrieving necessary information we appended data frames back. We only left information about average land value per acre, average property value per acre, and the number of properties per square mile, grouped by neighborhood.

Merged data

At this point, we had two data frames with trees and property data grouped by neighborhoods. So we merged them and built correlation matrices.



```
dfN.corr(method = 'pearson')
```

	area	health	trees/sq.mile	avg.landprice_thous\$/acre	properties/sq.mile
area	1.000000	-0.109530	-0.538290	0.551459	-0.368280
health	-0.109530	1.000000	0.082988	-0.243112	-0.064489
trees/sq.mile	-0.538290	0.082988	1.000000	-0.376926	0.444192
avg.landprice_thous\$/acre	0.551459	-0.243112	-0.376926	1.000000	-0.172714
properties/sq.mile	-0.368280	-0.064489	0.444192	-0.172714	1.000000

Figure 4. Correlation Matrix for Neighborhoods

```
dfB.corr(method='pearson')
```

	area	health	trees/sq.mile	avg.landprice_thous\$/acre	properties/sq.mile
area	1.000000	0.594033	-0.352201	0.126564	-0.280944
health	0.594033	1.000000	-0.770257	-0.684566	-0.849138
trees/sq.mile	-0.352201	-0.770257	1.000000	0.801241	0.890927
avg.landprice_thous\$/acre	0.126564	-0.684566	0.801241	1.000000	0.847321
properties/sq.mile	-0.280944	-0.849138	0.890927	0.847321	1.000000

Figure 5. Correlation Matrix for Boroughs

Results

- Results show that there is a low positive correlation between number of trees/sq.mile and number of properties/sq.mile (Pearson coefficient is 0.44 for neighborhoods and 0.89 for boroughs).

This is a surprising result, so we calculated the p-value.

```
import statsmodels.api as sm
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
y = scaler.fit_transform(np.array(dfn['trees/sq.mile']).reshape(-1,1))
scaler = StandardScaler()
X = scaler.fit_transform(np.array(dfn['properties/sq.mile']).reshape(-1,1))
X2 = sm.add_constant(X)
est = sm.OLS(y, X2)
|
print(est.fit().f_pvalue)

1.7109296370278895e-10
```

The p-value is very low, therefore there is a statistically significant relationship between the number of trees/sq.mile and number of properties/sq.mile.

- Results for neighborhoods show that there is actually a very low negative correlation between average land price and number of trees per square mile.

We think we got this result because of the method which we used when analyzing data. Instead of considering lands with similar properties (size, size of land, size of buildings, etc.) across all NYC, we considered absolutely different properties in each neighborhood.

However, for the boroughs this correlation is high positive. Data shows that Manhattan has the most expensive properties, and the most trees per square mile, whereas Staten Island has the cheapest land prices and least number of trees per square mile.

#####

NEED TO ADD MORE DISCUSSION OF RESULTS

#####

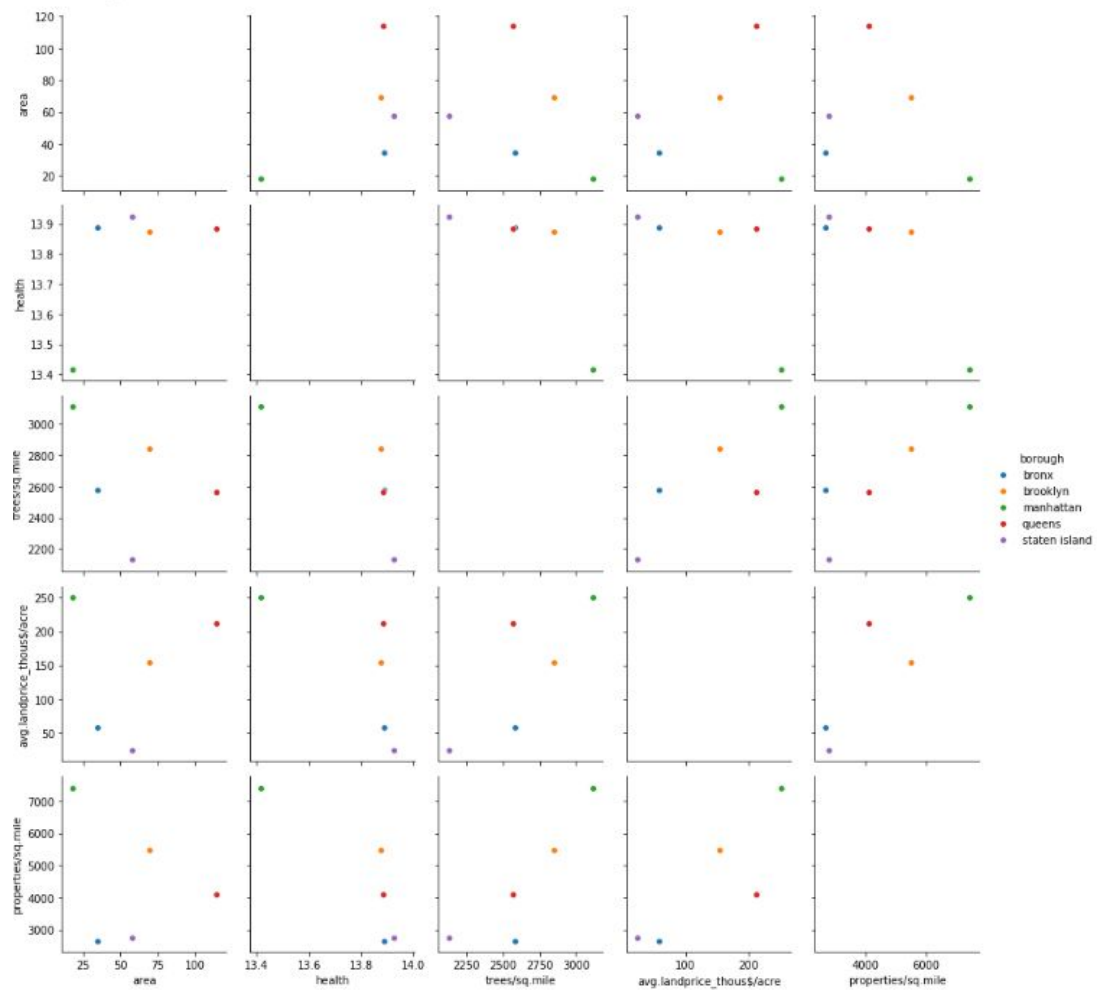
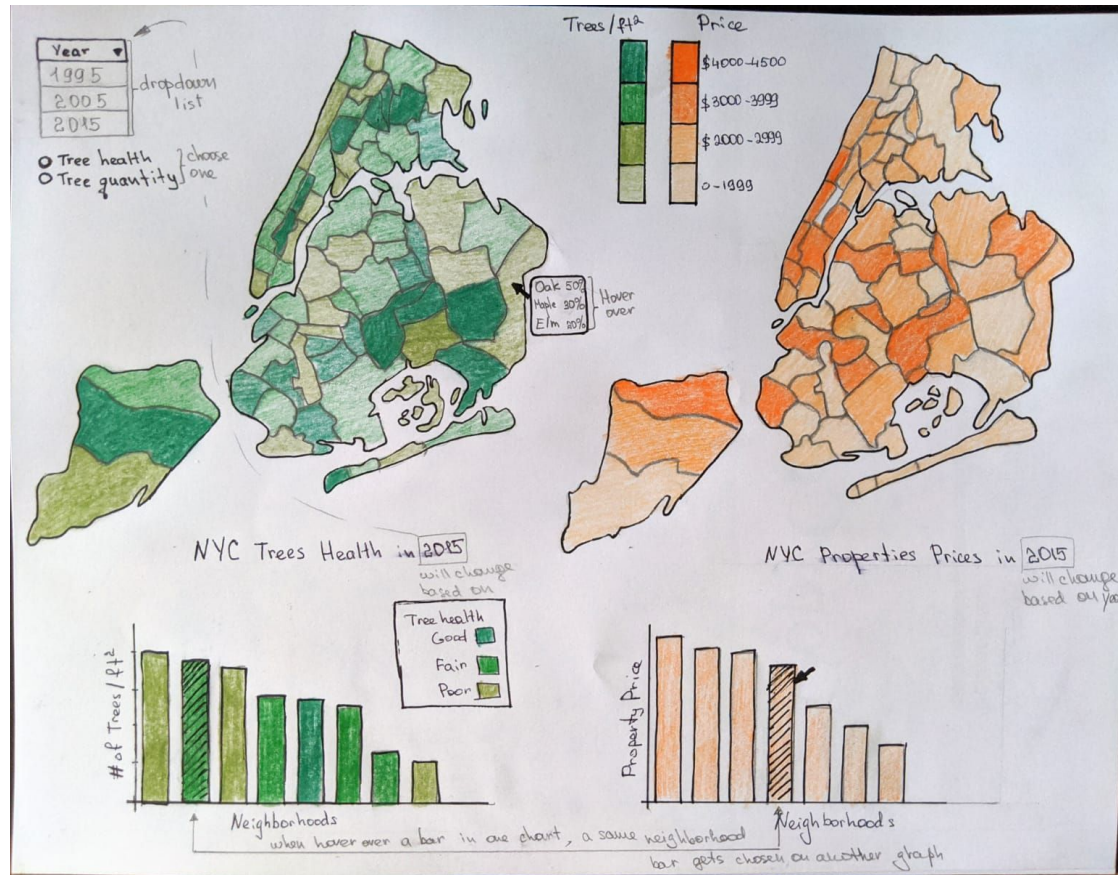


Figure 6. Pairplot for Boroughs

Building the Visualization

We decided to use Python for our visualization, therefore we chose to use Dash.

The first sketch of visualization was the following:



Task 1

The very first task for us was to visualize neighborhoods and boroughs borders on the map. We were using a Mapbox styled map. But Mapbox gives free access to countries and states boundaries, not neighborhoods and boroughs. We couldn't figure out how to color boundaries according to our data without access to Mapbox boundaries.

We even emailed to Mapbox help desk, asking for short access to the NYC neighborhoods and boroughs boundaries:

ALERT - Lead Directed to Community

This person was sent to Sales and they have decided to point them to the Community Team:

Marketo Alert Information

Lead: [Marina Orzechowski \(SFDC Detail\)](#)

Campaign: OPR-sales-dev-emails-community-support.01a-send-email-generic-community-link

Time: Apr 27, 2020 11:46 AM PDT

PERSON INFO

Name: Marina Orzechowski

Title: unknown

Email: mskachk000@citymail.cuny.edu

Last Inbound Message Date: 2020-04-20 19:57:00

Last Inbound Message: Hello! Hope you are staying safe! I am student, working on a visualization project. I would like to get an access to mapbox boundaries (specifically, I need access to NYC neighborhoods). What will be the price for this access?

Current Lead/Contact Owner: Cheri Roohi

COMPANY INFO

Company Name: City University of New York



Mikel Maron (Mapbox)

Apr 27, 10:10 PM PDT

Hello Marina ,

Thank you for contacting Mapbox – our Sales team directed your inquiry to our Community team, which supports non-profits, educational institutions, and other positive impact projects. For more information about the Community Team and the types of projects we support visit:

<https://www.mapbox.com/community/>.

Can you share some more about your visualization project, your studies, and how you'd utilize boundaries?

Best,



Marina Orzechowski

Apr 28, 6:36 AM PDT

Hi Mikel!

Thank you for reaching out to me!
I already fixed the problem and built map using NYC Open Data geojson files.

Thank you again! Stay safe!

Sincerely,
Marina



Mikel Maron (Mapbox)

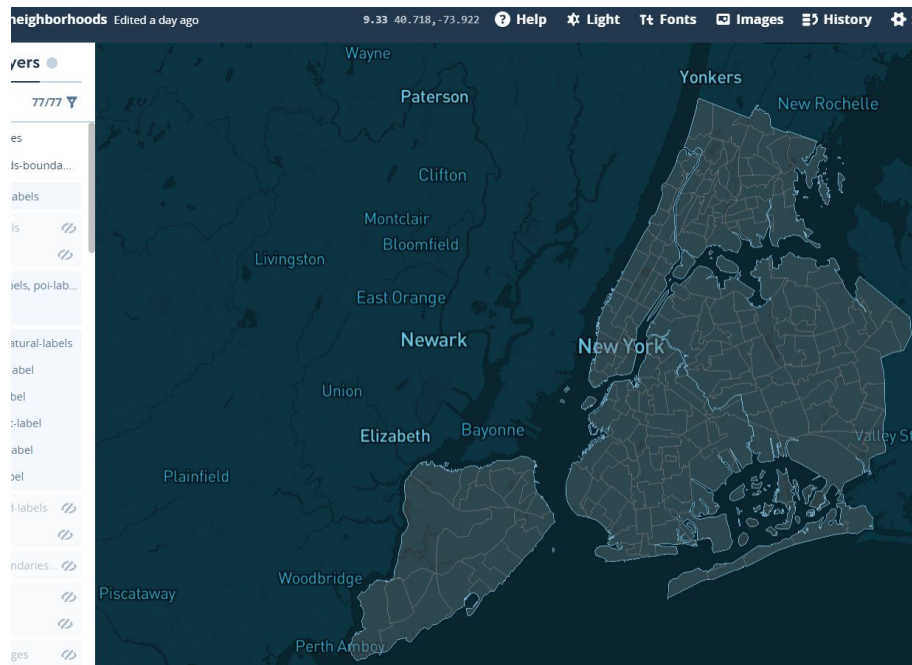
Apr 28, 6:43 AM PDT

Great, was going to recommend that too. keep us posted on how it goes

Figure 7. Conversation with Mapbox Person About the Access to NYC boundaries

We solved this problem without using Mapbox boundaries. We downloaded a [geojson file](#) with boundaries from NYC Open Data.

- First, we added it the Mapbox Studio style:



- Then we found centers of the neighborhoods and boroughs ([FindCentroids.ipynb](#)).
- After that we needed to divide data into different bins, in our case - number of trees per square mile, and average land price. And then, using previously downloaded NYC boundaries data set, we built new geojson files for each bin ([GeojsonFactory.ipynb](#)):



0-500 trees/sq.mile



1001-1500 trees/sq.mile



1501-2000 trees/sq.mile

etc...

Now in code we could associate each of these shapes with a color, so it could represent one bin.

- Finally, in code, we needed to color each area from the previous bullet point, add them up, and add as a new layer to the Mapbox styled map.
- For the information to show when hover, we created a column in data:

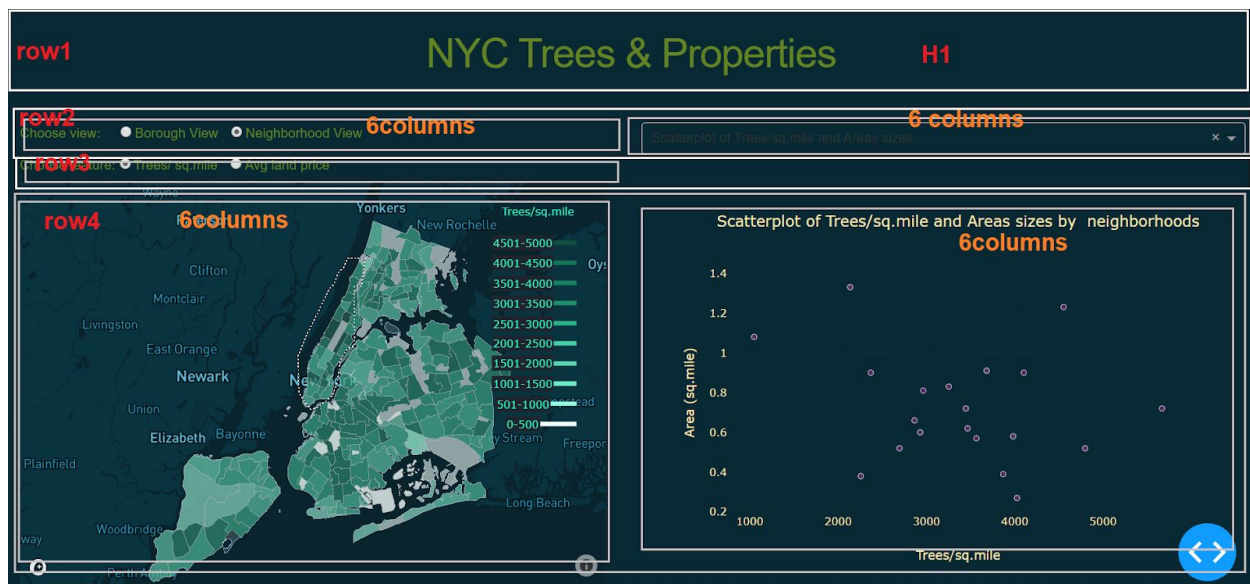
Search this file...

/sq.mile	centerLong	centerLat	hover
-73.74590558624465	40.63648234062998	airport queens trees/sq.mile: 18.94 avg. landprice: 1995.03 trees health: 14.32/15	
-73.84742515249926	40.86447228407376	allerton-pelham gardens bronx trees/sq.mile: 3174.61 avg. landprice: 18.24 trees health: 13.94/15	
-74.18769988091802	40.528667188342695	annadale-huguenot-prince's bay-eltingville staten island trees/sq.mile: 2477.99 avg. landprice: 19.11 trees	

This works just as fine, as Mapbox boundaries.

Task 2

Build a layout in Dash. We decided to change the original layout (as drawn at the picture above) because we didn't find a correlation between land prices and number of trees.



Task 3

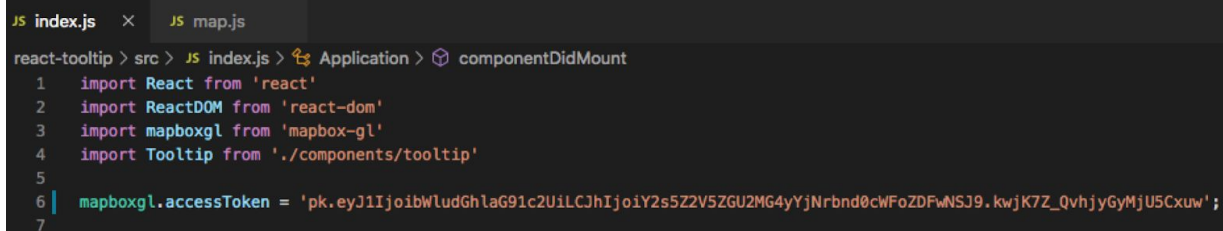
Make visualization interactive. We achieved this by using callbacks in Dash.

Attempt to Use Alternative Frameworks

Along with a successful development for this project using Dash in Python, another programming language which is React had been also considered to develop this visualization for past weeks. For building a React application with mapbox-gl-js, we went over the tons of online tutorials about geocoding and tried to learn the skills of making interactive features based on our data.

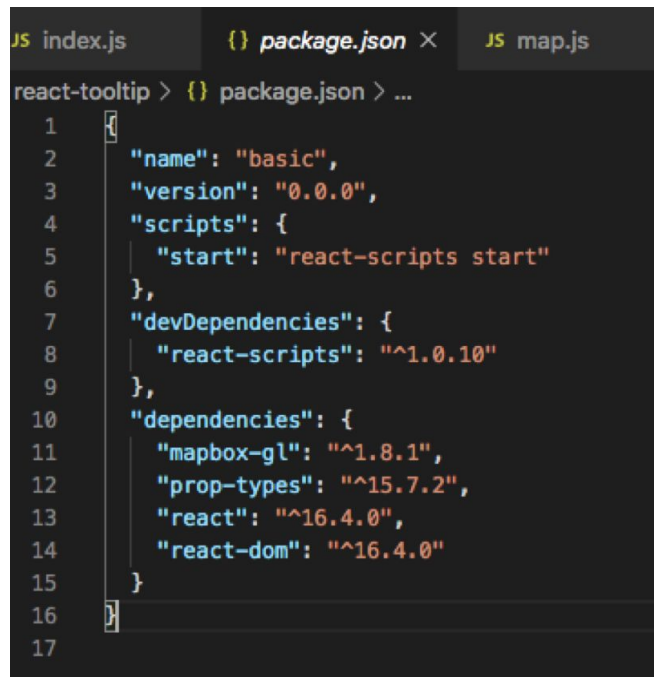
From what we researched and learned about using Mapbox GL JS in React app during the past few weeks, React manipulates the DOM and it can be very confusing to connect React with other libraries that also manipulate the DOM and manage Mapbox GL.JS. However, we persisted to learn how to use Mapbox GL JS to render the map, display the coordinate of the map points and user interaction.

1. First off, we needed to import a few libraries for React, DOM, mapboxgl along with the access token that you are given from Map Box.

A screenshot of a code editor with a dark theme. The top bar shows two tabs: 'JS index.js' and 'JS map.js'. The 'JS index.js' tab is active. The editor shows the following code:

```
react-tooltip > src > JS index.js > Application > componentDidMount
1 import React from 'react'
2 import ReactDOM from 'react-dom'
3 import mapboxgl from 'mapbox-gl'
4 import Tooltip from './components/tooltip'
5
6 mapboxgl.accessToken = 'pk.eyJ1IjoibWludGhlaG91c2UiLCJhIjoieY2s5Z2V5ZGU2MG4yYjNrbd0cWFOZDFwNSJ9.kwjK7Z_QvhjyGyMjU5Cxuw';
7
```

2. And make sure that we have included all the packages downloaded in package.json file.

A screenshot of a code editor with a dark theme. The top bar shows three tabs: 'JS index.js', '{} package.json', and 'JS map.js'. The 'package.json' tab is active. The editor shows the following JSON content:

```
react-tooltip > {} package.json > ...
1 {
2   "name": "basic",
3   "version": "0.0.0",
4   "scripts": {
5     "start": "react-scripts start"
6   },
7   "devDependencies": {
8     "react-scripts": "^1.0.10"
9   },
10  "dependencies": {
11    "mapbox-gl": "^1.8.1",
12    "prop-types": "^15.7.2",
13    "react": "^16.4.0",
14    "react-dom": "^16.4.0"
15  }
16
17
```

3. Back in the main file called index.js, we imported some more libraries and classes for visualization(mapping, toggling, legend) and a Data Base from react-redux as well.

```
JS index.js react-tooltip/... JS map.js JS index.js data-overlay-redux/... X
data-overlay-redux > src > JS index.js > ...
1  import React from 'react'
2  import ReactDOM from 'react-dom'
3  import { Provider } from 'react-redux';
4  import { store } from '../redux/store'
5  import { setActiveOption } from '../redux/action-creators'
6  import Map from '../components/map'
7  import Toggle from '../components/toggle'
8  import Legend from '../components/legend'
9
10 class Application extends React.Component {
11   render() {
12     return (
13       <Provider store={store}>
14         <div>
15           <Map />
16           <Toggle onChange={setActiveOption} />
17           <Legend />
18         </div>
19       </Provider>
20     );
21   }
22 }
23
24 ReactDOM.render(<Application />, document.getElementById('app'));
25
```

4. In legend class, we used propTypes to handle different types of prop in react app. Just like HTML tag deals with many different types of attributes, prop does same thing in react app. It can be any type such as a string, function, or an array, as long as it is a valid javascript. With having propTypes, we got to add boundaries on the map.

```
1  import React from 'react'
2  import PropTypes from 'prop-types'
3  import { connect } from 'react-redux'
4
5  let Legend = class Legend extends React.Component {
6
7    static propTypes = {
8      active: PropTypes.object.isRequired
9    };
10
11    render() {
12      const { name, description, stops } = this.props.active;
13
14      const renderLegendKeys = (stop, i) => {
15        return (
16          <div key={i} className='txt-s'>
17            <span className='mr6 round-full w12 h12 inline-block align-middle' style={{ backgroundColor: stop[i] }} />
18            <span>{'${stop[i].toLocaleString()}'}/>
19          </div>
20        );
21      };
22
23      return [
24        <div className='bg-white absolute bottom right mr12 mb24 py12 px12 shadow-darken10 round z1 wmax180'>
25          <div className='mb6'>
26            <h2 className='txt-bold txt-s block'>{name}</h2>
27            <p className='txt-s color-gray'>{description}</p>
28          </div>
29          {stops.map(renderLegendKeys)}
30        </div>
31      ];
32    }
33  };
```


<<Few challenges and Failures that we had encountered>>

1. We wanted to add the toggle button on top of the map so we can change the view from a property-value-based scatter map to a tree-value-based scatter map.

We could only figure the way how to put the toggle button visually but never made it connect to the dataset that we wanted. Below is the piece of code for the toggle class.

```
data-overlay-redux > src > components > JS toggle.js > ...
1  import React from 'react'
2  import PropTypes from 'prop-types'
3  import { connect } from 'react-redux'
4
5  let Toggle = class Toggle extends React.Component {
6
7      static propTypes = {
8          options: PropTypes.array.isRequired,
9          active: PropTypes.object.isRequired,
10         onChange: PropTypes.func.isRequired
11     };
12
13     render() {
14         const { options, active } = this.props;
15
16         const renderOptions = (option, i) => {
17             return (
18                 <label key={i} className="toggle-container">
19                     <input onChange={() => this.props.onChange(option)} checked={option.property === active.property} name="toggle" type="radio" />
20                     <div className="toggle txt-s py3 toggle--active-white">{option.name}</div>
21                 </label>
22             );
23         };
24
25         return (
26             <div className="toggle-group absolute top left ml12 mt12 border border--2 border--white bg-white shadow-darken10 z1">
27                 {options.map(renderOptions)}
28             </div>
29         );
30     }
31 }
32
33 function mapStateToProps(state) {
34     return {
35         options: state.options,
36         active: state.active
37     };
38 }
```

2. Also, another big challenge that we had was making interactive bar-chart to show data for categorized neighborhoods. We hadn't had enough time to touch upon the bar-chart because we were caught up with the main map for a long time, but still couldn't make it work.

Citation

<https://data.cityofnewyork.us/City-Government/Property-Valuation-and-Assessment-Data/yjxr-fw8i>
<https://data.cityofnewyork.us/Environment/2015-Street-Tree-Census-Tree-Data/uvpi-gqnh>
<https://www.greenblue.com/na/how-trees-increase-property-values/>
<https://github.com/jackparmer/mapbox-counties/blob/master/GeoJSON%20factory.ipynb>

Consolidated datasets, python scripts and Dash code can be found on the GitHub:

https://github.com/MarinaOrzechowski/NYC_Trees_Properties