COMP40610 Visual Exploration Tool Design Document

Student Name (s): Ye Xing

Student Number: 22200374

Title:

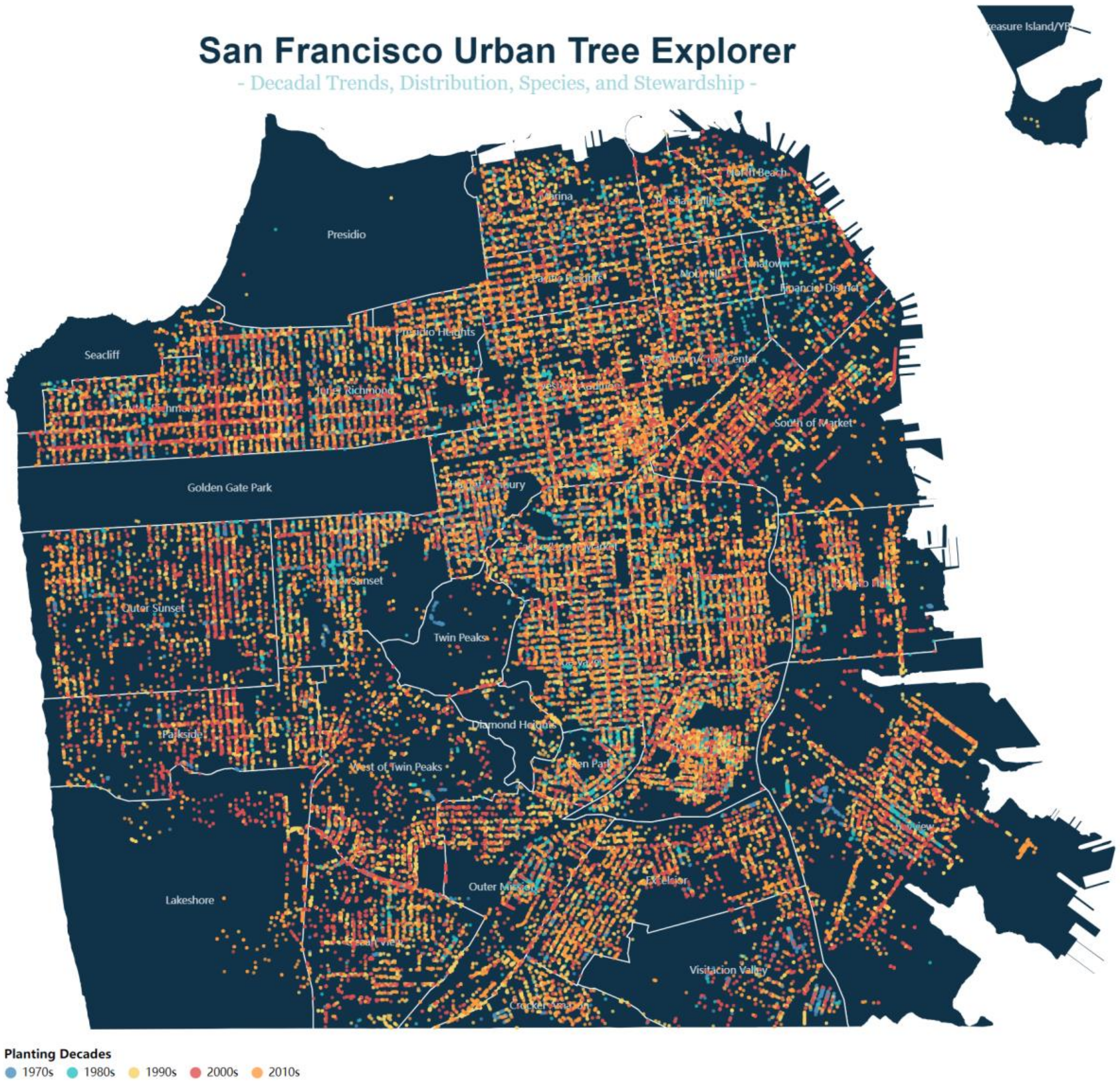
San Francisco Urban Tree Explorer

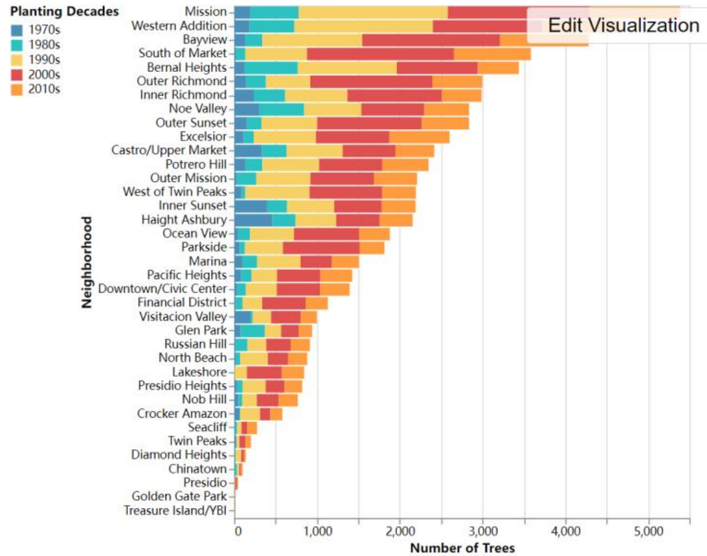
-Decadal Trends, Distribution, Species, and Stewardship-

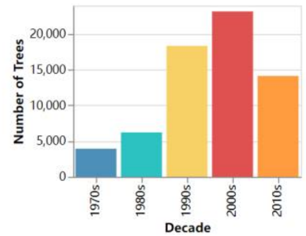
Screenshot:

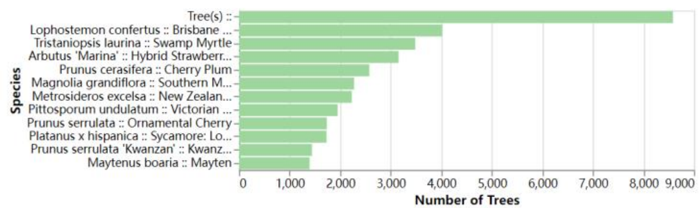
San Francisco Urban Tree Explorer

- Decadal Trends, Distribution, Species, and Stewardship -









Dataset overview:

****Requirement: This section should detail the dataset you used, where it came from, and any manipulation you have performed on the dataset.****

The dataset I am using is divided into two parts. The first section, "San Francisco Trees," is a CSV file provided by the San Francisco Department of Public Works. It includes valuable information on all trees in San Francisco. The second section, "san-francisco-ca," is a GeoJSON format file, also supplied by the San Francisco Department of Public Works. This file delineates the community divisions within San Francisco. I intend to use this to create an analytical map of San Francisco, featuring a clear depiction of the city's major residential areas along with their corresponding names. Both sections of the dataset have been thoroughly cleaned and prepared to facilitate my upcoming analysis. This meticulous preparation is a vital component of my research and report on San Francisco's urban tree explorer.

Methodology:

Before delving into the segmented analysis, let me outline the methods I employed for data manipulation and cleaning. I utilized Jupyter Notebook as the platform, integrating tools from GeoPandas and methods from Shapely.geometry to process the 'sf_trees.csv' and 'san-francisco-ca_.geojson' files. To ensure the efficacy of the cleaning process, I also used Pandas to validate the accuracy and integrity of the consolidated data. This multi-faceted approach not only enhanced the precision of the data but also facilitated a more robust analysis by ensuring that the datasets were thoroughly refined and reliable for the subsequent analytical phases.

sf_trees.csv Overview:

The primary content of the dataset includes fields like *tree_id*, *legal_status*, *species*, *address*, *site_order*, *address order*, *site_info*, *caretaker*, *date*, *dbh*, *plot_size*, *latitude*, and *longitude*. For my analysis, I focused particularly on certain key attributes: *tree_id*, to calculate the total number of trees in each area; *species*, to understand the types of trees preferred for planting in specific areas and eras in San Francisco; *address*, which is useful for pinpointing the exact location of each tree; *caretaker*, to identify and tally the primary caretakers of the trees planted in specific areas during certain periods; and *date*, for categorization, specifically dividing the trees by the decades they were planted in, such as the 1970s, 1980s, 1990s, 2000s, and 2010s. This targeted approach allows for a detailed and nuanced understanding of the urban forestry trends in San Francisco over time.

Manipulation and Cleaning Process of sf_trees.csv:

1. Preliminary Steps: This dataset contains dozens of tree species and numerous other interesting features! I did remove some columns that were either more than 75% missing or redundant. You are welcome to refer to the original source for the complete dataset.
2. Step 1. Cleaning by Removing Entries with Null in Key Features: Creating a scatter plot of trees on a map of San Francisco is a critical component of my analysis, making the coordinate information of the trees essential. Therefore, it's imperative that there are no null or NA values in the *latitude* and *longitude* data. Similarly, the *date* information of the trees is a vital aspect of the scatter plot's color coding. Trees will be categorized

into different decades based on their planting dates—1970s, 1980s, 1990s, 2000s, and 2010s—and assigned different colors accordingly. Thus, these fields also cannot contain null or NA values. After this cleaning process, the number of elements in the fields *tree_id*, *species*, *caretaker*, *date*, *latitude*, and *longitude* are identical, indicating that the essential data for points to be displayed on the scatter plot is complete and free of missing values. Following this process, we successfully obtained 66,448 useful tree data points, ready for detailed scatter plot analysis.

In [3]:
```
unique_values_per_column = sf_trees_df.nunique()
print(unique_values_per_column)

# Remove rows with NA values in 'latitude' or 'longitude' or 'date' columns
cleaned_sf_trees_df = sf_trees_df.dropna(subset=['latitude', 'longitude','date'])
```
executed in 296ms, finished 17:55:30 2023-11-21

In [6]:
```
columns_and_non_null_count = cleaned_sf_trees_df.info()
```
executed in 61ms, finished 17:55:30 2023-11-21

```
<class 'pandas.core.frame.DataFrame'>
Index: 66448 entries, 0 to 68376
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   tree_id       66448 non-null  int64
 1   legal_status  66410 non-null  object
 2   species       66448 non-null  object
 3   address       66372 non-null  object
 4   site_order    65346 non-null  float64
 5   site_info     66448 non-null  object
 6   caretaker     66448 non-null  object
 7   date          66448 non-null  object
 8   dbh           37655 non-null  float64
 9   plot_size     29429 non-null  object
 10  latitude      66448 non-null  float64
 11  longitude     66448 non-null  float64
dtypes: float64(4), int64(1), object(7)
memory usage: 6.6+ MB
```

3. Step 2: Eliminating Trees Planted Outside the Range of January 1, 1970, to December 31, 2019: For the scatter plot and subsequent bar graphs, it's crucial that trees are categorized by decades of planting—1970s, 1980s, 1990s, 2000s, and 2010s—with corresponding colors for each period. Therefore, my study focuses exclusively on trees planted within the timeframe of January 1, 1970, to December 31, 2019. Trees planted outside

this range are not relevant to my analysis and have been removed. After this cleaning step, the number of entries in the fields of *tree_id*, *species*, *caretaker*, *date*, *latitude*, and *longitude* remain consistent, ensuring that the essential data for the scatter plot's points is complete, accurate, and devoid of missing values. Following this process, we successfully obtained 65,830 useful tree data points, ready for detailed scatter plot analysis.

In [10]:

```
# change 'date' column as datetime type
cleaned_sf_trees_df['date'] = pd.to_datetime(cleaned_sf_trees_df['date'])

# filter the data after 1970
cleaned_sf_trees_df = cleaned_sf_trees_df[cleaned_sf_trees_df['date'] >= pd.to_datetime('1970-01-01')]

# filter the data before 2020
cleaned_sf_trees_df = cleaned_sf_trees_df[cleaned_sf_trees_df['date'] <= pd.to_datetime('2019-12-31')]
```

executed in 91ms, finished 17:55:32 2023-11-21

In [11]:

```python
unique_values_per_column = cleaned_sf_trees_df.nunique()
print(unique_values_per_column)
columns_and_non_null_count = cleaned_sf_trees_df.info()
```

executed in 106ms, finished 17:55:33 2023-11-21

```
tree_id          65830
legal_status         9
species            426
address          43140
site_order          93
site_info           26
caretaker           20
date              7282
dbh                 58
plot_size          311
latitude         52828
longitude        52821
Neighborhood        37
dtype: int64
<class 'pandas.core.frame.DataFrame'>
Index: 65830 entries, 503 to 66332
Data columns (total 13 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   tree_id       65830 non-null  int64
 1   legal_status  65792 non-null  object
 2   species       65830 non-null  object
 3   address       65754 non-null  object
 4   site_order    64728 non-null  float64
 5   site_info     65830 non-null  object
 6   caretaker     65830 non-null  object
 7   date          65830 non-null  datetime64[ns]
 8   dbh           37406 non-null  float64
 9   plot_size     29300 non-null  object
 10  latitude      65830 non-null  float64
 11  longitude     65830 non-null  float64
 12  Neighborhood  65826 non-null  object
dtypes: datetime64[ns](1), float64(4), int64(1), object(7)
memory usage: 7.0+ MB
```

4. <u>Step 3: Integrating Data with 'san-francisco-ca.geojson' , Adding the 'Neighbourhood' Column and Removing Null Values from 'Neighbourhood':</u> To display tree scatter plots by each major neighborhood in the subsequent analysis, it was essential to integrate data from 'san-francisco-ca.geojson'. This involved using its boundary data to determine which neighborhood each tree's coordinates fell into and assigning the corresponding neighborhood to each tree, thereby creating a new field: *'Neighbourhood'*. The 'Neighbourhood' data is critical; any tree entries with a null value in the *'Neighbourhood'* field were removed. Following this process, we successfully obtained 65,826 useful tree data points, each now equipped with a *'Neighbourhood'* field, ready for detailed scatter plot analysis.

In [9]:

```python
# create Trees'GeoDataFrame
gdf_trees = gpd.GeoDataFrame(
    cleaned_sf_trees_df,
    geometry=[Point(xy) for xy in zip(cleaned_sf_trees_df.longitude, cleaned_sf_trees_df.latitude)],
    crs="EPSG:4326"
)

# add neighbourhood boundary' data
gdf_neighborhoods = gpd.read_file('san-francisco-ca_.geojson')

# Spatial linking, matching tree data with community names.
gdf_trees = gpd.sjoin(gdf_trees, gdf_neighborhoods[['name', 'geometry']], how="left", op='within')



# Reset the index to ensure index consistency.
gdf_trees.reset_index(drop=True, inplace=True)
cleaned_sf_trees_df.reset_index(drop=True, inplace=True)


# Add the community names from the join results to the original sf_trees DataFrame.
cleaned_sf_trees_df['Neighborhood'] = gdf_trees['name']

# At this point, the sf_trees DataFrame includes a new 'Neighborhood' column.
```

executed in 2.04s, finished 17:55:32 2023-11-21

In [11]:

```python
unique_values_per_column = cleaned_sf_trees_df.nunique()
print(unique_values_per_column)
columns_and_non_null_count = cleaned_sf_trees_df.info()
```

executed in 106ms, finished 17:55:33 2023-11-21

```
tree_id         65830
legal_status        9
species           426
address         43140
site_order         93
site_info          26
caretaker          20
date             7282
dbh                58
plot_size         311
latitude        52828
longitude       52821
Neighborhood       37
dtype: int64
<class 'pandas.core.frame.DataFrame'>
Index: 65830 entries, 503 to 66332
Data columns (total 13 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   tree_id       65830 non-null  int64
 1   legal_status  65792 non-null  object
 2   species       65830 non-null  object
 3   address       65754 non-null  object
 4   site_order    64728 non-null  float64
 5   site_info     65830 non-null  object
 6   caretaker     65830 non-null  object
 7   date          65830 non-null  datetime64[ns]
 8   dbh           37406 non-null  float64
 9   plot_size     29300 non-null  object
 10  latitude      65830 non-null  float64
 11  longitude     65830 non-null  float64
 12  Neighborhood  65826 non-null  object
dtypes: datetime64[ns](1), float64(4), int64(1), object(7)
memory usage: 7.0+ MB
```

In [12]:

```
cleaned_sf_trees_df = cleaned_sf_trees_df.dropna(subset=['Neighborhood'])
```

executed in 30ms, finished 17:55:33 2023-11-21

In [13]:

```
unique_values_per_column = cleaned_sf_trees_df.nunique()
print(unique_values_per_column)
columns_and_non_null_count = cleaned_sf_trees_df.info()
```

executed in 122ms, finished 17:55:33 2023-11-21

```
tree_id         65826
legal_status        9
species           426
address         43138
site_order         93
site_info          26
caretaker          20
date             7282
dbh                58
plot_size         311
latitude        52825
longitude       52818
Neighborhood       37
dtype: int64
<class 'pandas.core.frame.DataFrame'>
Index: 65826 entries, 503 to 66332
Data columns (total 13 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   tree_id       65826 non-null  int64
 1   legal_status  65788 non-null  object
 2   species       65826 non-null  object
 3   address       65750 non-null  object
 4   site_order    64724 non-null  float64
 5   site_info     65826 non-null  object
 6   caretaker     65826 non-null  object
 7   date          65826 non-null  datetime64[ns]
 8   dbh           37405 non-null  float64
 9   plot_size     29300 non-null  object
 10  latitude      65826 non-null  float64
 11  longitude     65826 non-null  float64
 12  Neighborhood  65826 non-null  object
dtypes: datetime64[ns](1), float64(4), int64(1), object(7)
memory usage: 7.0+ MB
```

5. Step 4: Saving and Exporting the Newly Cleaned 'sf_trees_cleaned.csv': After completing the cleaning process, the final step involves saving and exporting the thoroughly cleaned and updated dataset as 'sf_trees_cleaned.csv'. This file now contains all the refined data, ready for further analysis and use.

In [16]:
```
# Save the cleaned DataFrame back to a CSV file, if needed
cleaned_sf_trees_df.to_csv('sf_trees_cleaned.csv', index=False)
```
executed in 854ms, finished 17:57:35 2023-11-21

san-francisco-ca_.geojson Overview:

The main contents of this file include *name*, *cartodb_id*, *created_at*, and *updated_at*, along with geometric data. The geometry is of the type MultiPolygon, consisting of a detailed list of coordinates that outline the shapes of various large neighborhood areas. These coordinates are crucial as they define the boundaries and distinct regions within San Francisco, providing essential information for analyses requiring a clear understanding of the city's neighborhood divisions.

Manipulation and Cleaning Process of san-francisco-ca_.geojson:

1. Step 1: Generating *'centroid_longitude'* and *'centroid_latitude'* for Each Neighborhood Based on Existing Boundary Coordinates: The reason for needing *'centroid_longitude'* and *'centroid_latitude'* is to display the names of each neighborhood on the map within the scatter plot created in Vega-Lite, enhancing user readability. The location for each neighborhood name is determined by the center of its respective polygon. I used the .centroid method to obtain *'centroid_longitude'* and *'centroid_latitude'* for this purpose. This step is crucial for accurately positioning neighborhood names on the map, thereby making the data visualization more informative and user-friendly.

In [18]:

```python
# Load GeoJSON data.
gdf = gpd.read_file('san-francisco-ca_.geojson')

# Calculate the geometric center of each plot.
gdf['centroid'] = gdf.geometry.centroid

# Add the latitude and longitude of the center points as new attributes to the GeoJSON.
gdf['centroid_longitude'] = gdf.centroid.x
gdf['centroid_latitude'] = gdf.centroid.y

# Remove the non-serializable 'centroid' column.
gdf = gdf.drop(columns=['centroid'])

# Convert any Timestamp data to strings.
gdf = gdf.applymap(lambda x: x.isoformat() if isinstance(x, pd.Timestamp) else x)

# Convert the GeoDataFrame to GeoJSON.
updated_geojson = json.loads(gdf.to_json())

# Save the updated GeoJSON to a new file.
with open('updated_san-francisco-ca.geojson', 'w') as f:
    json.dump(updated_geojson, f)
```

executed in 407ms, finished 17:59:34 2023-11-21

2. Step 2: Saving and Exporting the Newly Updated and Cleaned 'updated_san-francisco-ca.geojson': After completing the data manipulation and adding key geographic details, the final step involves saving and exporting the refined dataset as 'updated_san-francisco-ca.geojson'. This file now contains all the enhanced and updated geographic information, ready for integration into further spatial analyses and mapping projects.

Design considerations:

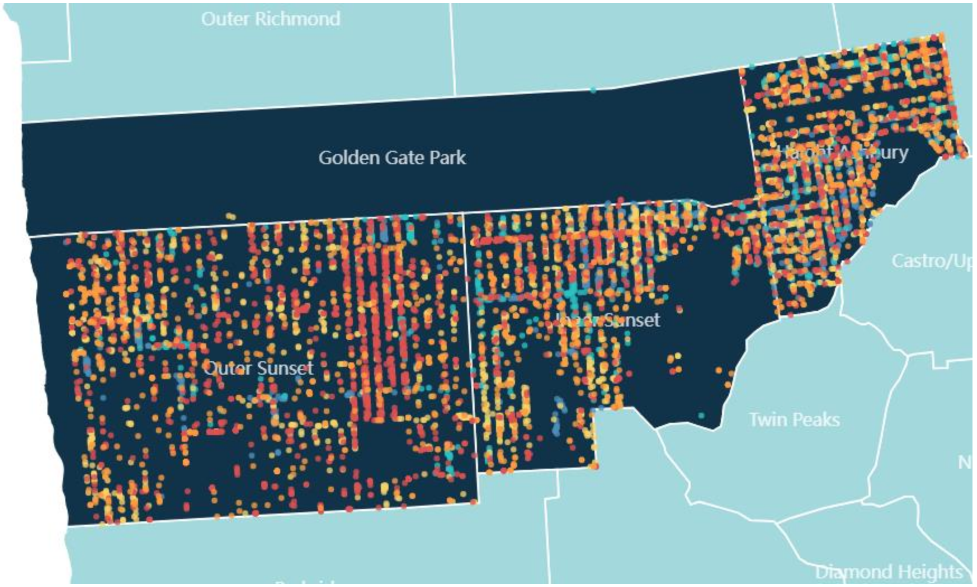
****This should provide an overview of your visualisation, a discussion of why you used specific encoding / interaction options, and the pros/cons of your visualisation vs alternatives.****

Overall goal:

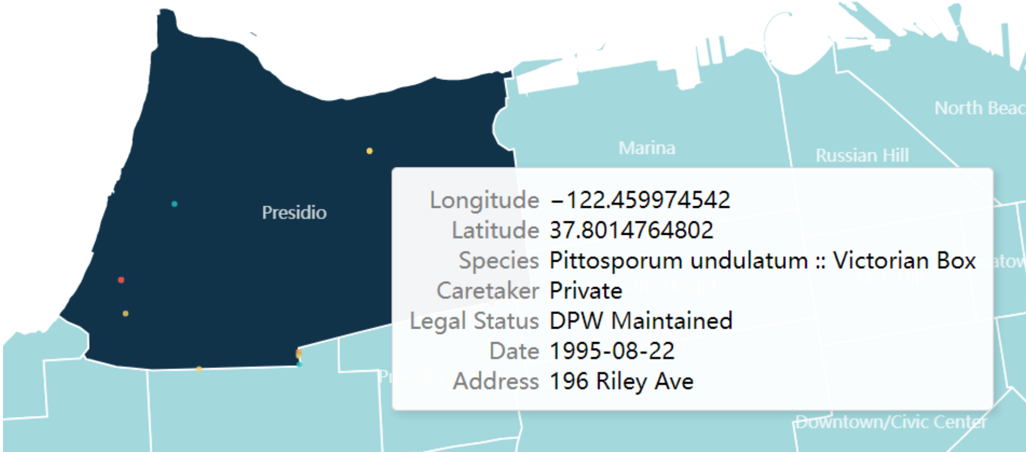
My goal with this tool is to explore the overall urban tree planting trends and fluctuations in numbers within San Francisco from the 1970s to the 2010s, both city-wide and within individual neighborhoods. I aim to observe the proportion of tree ownership (caretakers) and identify the most popular tree species overall, by decade, by area, and within each category of ownership (caretaker), focusing on the top 12 varieties.

San Francisco Urban Tree Planting Decades Scatter Plot:

I used a scatter plot to display the distribution of trees across various areas of San Francisco. Each point represents a tree, with color coding to distinguish between planting decades. The 1970s are represented by a deep blue, the 1980s by teal, the 1990s by yellow, the 2000s by red, and the 2010s by orange. To enhance visual comfort, I deliberately avoided overly saturated or harsh colors that could cause discomfort to viewers. Considering accessibility for colorblind users, my color scheme is friendly and does not rely on color combinations difficult for colorblind users to distinguish, such as red and green. The color differences and gradients, along with my choice of a deep shade of navy for the map's background, follow common data visualization practices to ensure that data points are clear and readable. Dark backgrounds offer high contrast against brightly colored data points, highlighting the information presented.



This visual encoding method allows users to intuitively see the overall distribution of trees and easily identify the planting density in specific neighborhoods, as well as each decade's preference for planting in specific neighborhoods. I considered using a heatmap as an alternative but found that scatter plots provide clearer information on precise locations and tree counts. Although there is overlap due to the density of the data, the transparency of individual points means that areas with high overlap will appear darker, allowing users to perceive density. I also provided a hover window feature; if you're curious about a specific point or want detailed information, simply hover your mouse over it to display a window with its exact coordinates, caretaker, species, legal status, planting date, and address for closer study.

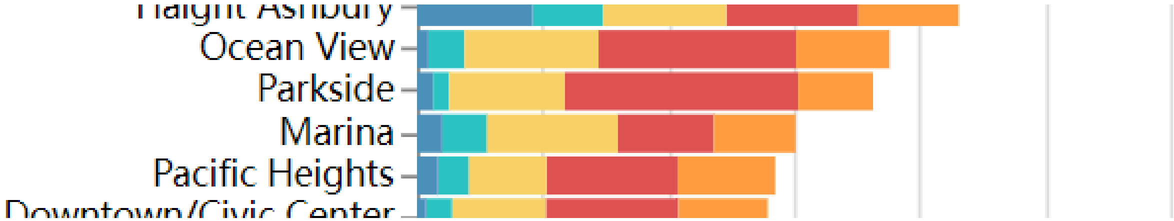


Additionally, the areas of neighborhoods not selected will lighten in color, drawing the user's focus to the selected area. The name of each neighborhood is strategically positioned at the polygonal centroid of the respective neighborhood, reducing the cognitive load for users trying to identify them. This careful use of color variation and strategic placement of neighborhood names enhances user engagement by simplifying the process of geographical identification and comparison within the visualization.

San Francisco Tree Distribution by Decade and Neighborhood Stacked Bar Chart:

This stacked bar chart showcases the number of trees planted in various neighborhoods of San Francisco across different decades, arranged in descending order to highlight the areas with the highest number of trees. Each color represents a decade, consistent with the encoding used in my scatter plot. The color gradients reflect the passage of time and changes in planting trends, which can inform analyses of environmental policies or urban development plans. The chart effectively displays the historical context of tree planting in each neighborhood, helping users identify which areas have been more proactive in environmental greening.

However, this design has its trade-offs. The stacked segments of different decades within the same neighborhood can make it challenging to discern specific differences in tree planting between decades. Shorter bars, especially where multiple colors are stacked, may be difficult to read for exact values. I considered replacing this with a Grouped Bar Chart, which would display the number of trees planted in different decades side by side for each neighborhood, rather than stacked. This would facilitate easier comparisons within neighborhoods across different decades. Yet, this alternative sacrifices the compactness of the chart, as it requires more horizontal or vertical space to display the same data. Given the limited space, the stacked bar chart is highly efficient in showing the relationship between the parts and the whole.



San Francisco Tree Planting Trends by Decade Bar Chart:

This bar chart employs length and color encoding to demonstrate the change in the number of trees planted in San Francisco across various decades. The length of each bar intuitively represents the total number of trees planted in each decade, while the use of distinct colors allows viewers to quickly identify the time period. This straightforward approach is particularly effective for displaying clear trends and comparisons within time series data. Although line charts have their advantages in highlighting trends over time, the bar chart is superior in this context due to its directness and simplicity in presenting aggregate numbers for each decade. It offers an at-a-glance understanding of the planting quantities per decade, ideal for rapidly grasping fundamental data trends. Here, the bar chart's unambiguous and concise presentation trumps the line chart, especially when it comes to independently comparing data points.

San Francisco Tree Caretaker Distribution Donut Chart:

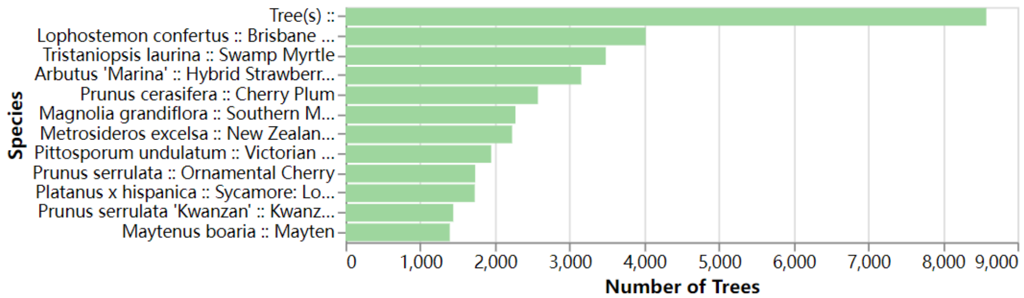
The donut chart uses color coding to clearly display the distribution of tree caretakers in San Francisco. The main categories highlighted are private and the Department of Public Works (DPW), which take up the majority of the chart, while the less numerous categories are consolidated into "Others" to simplify the visual presentation while maintaining the integrity of the information. The color scheme—deep blue, teal, and yellow—maintains visual consistency with previous charts, enhancing intuitiveness and the overall visual effect. The even distribution of colors and the design that avoids center-focused clutter make the information highly efficient and easy to understand for the audience.

The design of the donut chart avoids complexity, especially since the number of categories is limited, preventing a crowded appearance. While pie charts could serve as an alternative, the donut chart, with its modern feel and high ink ratio, is more effective in emphasizing the main categories and displaying data proportions. These design advantages make the donut chart the ideal choice for presenting this type of data, hence its selection as the final display method. This design strikes a balance between visual clarity, understandability, and aesthetic appeal, making it an effective way to convey information.

San Francisco Tree Species Distribution Bar Chart:

This bar chart, in a soft green hue, displays the planting numbers for the 12 most common tree species in specific decades and neighborhoods in San Francisco, with a descending order that highlights the most frequently planted species. The color choice not only avoids confusion with other charts but also provides a visually soothing effect that clearly accentuates the data. The bar chart's straightforward format allows viewers to quickly identify the most prevalent species, thereby understanding the data hierarchy and trends at a glance. Its conciseness ensures direct and comprehensible information transmission.

While pie charts or donut charts could also depict the proportion of each tree species, they fall short in showing precise quantities as clearly as a bar chart does. Bar charts offer clear visual cues for comparing numerical values across different categories, making them an excellent choice for presenting ranking data. They are particularly adept at accurately representing the specific number of each category, which is ideal for rapidly identifying the most common tree species within certain caretaker categories or neighborhoods. Therefore, despite the advantages other chart types might offer in certain scenarios, the bar chart was retained for its clarity and efficacy in displaying rankings and precise values.

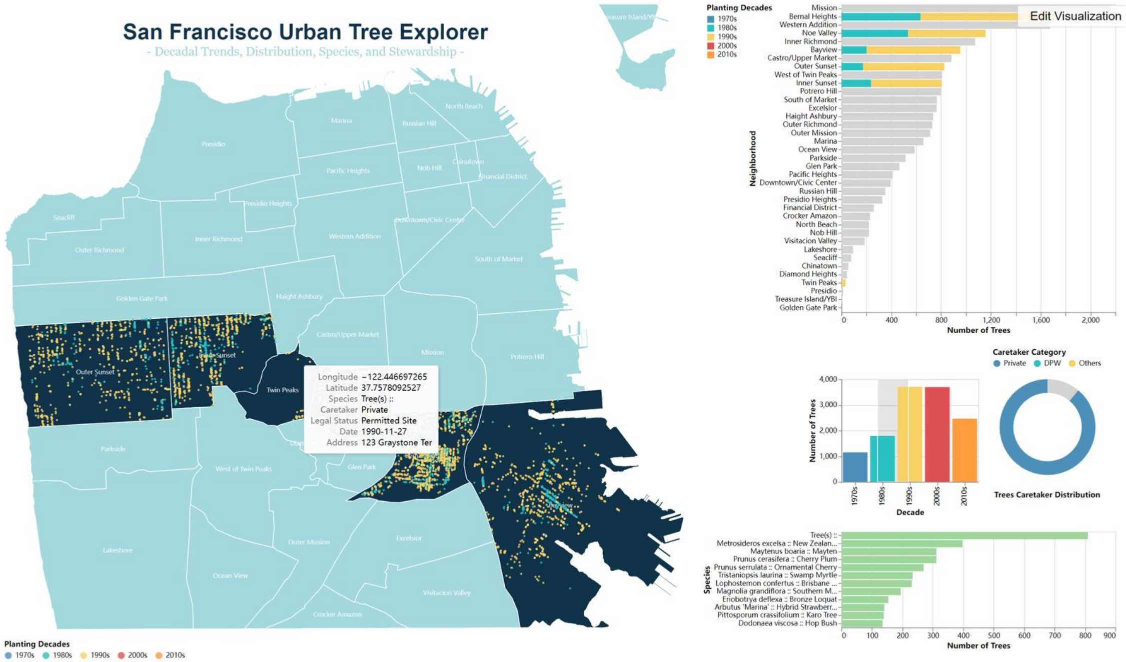


Interaction consideration:

In this suite of charts, the primary interaction mechanism I implemented is cross-filtering. For the "San Francisco Tree Distribution by Decade and Neighborhood Stacked Bar Chart," I enabled multi-selection clicking, allowing users to click on one or multiple bars. Selected bars are highlighted while the rest are dimmed. For the "San Francisco Tree Planting Trends by Decade Bar Chart," I utilized a box-selection tool, where the filtering criteria are based on the parameters drawn by the user's selection box. The same multi-selection clicking approach is applied to the "San Francisco Tree Caretaker Distribution Donut Chart," with selected areas highlighted and others dimmed.

Furthermore, any interaction within the "San Francisco Tree Distribution by Decade and Neighborhood Stacked Bar Chart," "San Francisco Tree Planting Trends by Decade Bar Chart," or "San Francisco Tree Caretaker Distribution Donut Chart" affects the data filtering and display across all charts. This interactivity is visibly evident in the "San Francisco Urban Tree Planting Decades Scatter Plot," where the data is filtered and displayed based on interactions from the other charts. A hover-over feature is present; hovering over a point displays a tooltip with detailed information about the selected tree.

For instance, if a user selects Bernal Heights, Noe Valley, Bayview, Twin Peaks and Inner Sunset neighborhoods in the stacked bar chart, 1980s and 1990s in the trends bar chart, and the "Private" category in the caretaker donut chart, the changes will be reflected across all charts. The scatter plot will highlight the selected neighborhoods, filter out trees that don't meet the criteria, and the "San Francisco Tree Species Distribution Bar Chart" will display the top 12 most popular tree species planted under these filtered conditions. This interactive design not only allows for dynamic exploration of the data but also enhances the user's ability to understand complex datasets through visually intuitive filters.



Insight:

My analysis offers an insightful overview of the tree-planting trends in San Francisco. There is a notable fluctuation in the number of trees planted from the 1970s to the 2010s, with the 2000s marking the peak of planting, possibly reflecting an increased environmental consciousness or intensified urban greening policies during that era. Neighborhoods such as Mission, Western Addition, Bay View, South of Market, and Bernal Heights lead in planting efforts, indicating their pivotal role in urban greening and environmental improvements.

The predominance of private caretakers in tree planting suggests a strong personal commitment and environmental awareness among residents to enhance living conditions. The Department of Public Works (DPW) follows, underscoring the role of public-private collaboration in urban greening. The selection of tree species, with a significant number being unspecified, followed by Lophostemon and Tristaniopsis laurina, may be attributed to their adaptability to the environment and aesthetic value.

The planting boom over the last three decades highlights the substantial efforts made by San Francisco to improve living environments, with tree planting activities evenly spread across the city rather than concentrated in a specific area, suggesting a multifaceted approach involving city planning, community development projects, and increased resident participation. This widespread distribution of tree planting also reflects the city's commitment to ecological diversity and urban aesthetics.

In summary, the history of tree planting in San Francisco over the past fifty years not only reveals the ecological trends in urban development but also exemplifies the active participation of citizens and the effectiveness of environmental policies. These data pave the way for anticipating future potentials and areas for improvement in environmental sustainability and urban greening.