San Francisco Rental Prices Dashboard

In this notebook, you will compile the visualizations from the previous analysis into functions that can be used for a Panel dashboard.

In [62]:
```
# %%bash
# ipython nbconvert --to=latex 'Index.ipynb' --post=pdf
```

In [3]:
```
# imports
import panel as pn
pn.extension('plotly')
import plotly.express as px
import pandas as pd
import hvplot.pandas
import matplotlib.pyplot as plt
import os
from pathlib import Path
from dotenv import load_dotenv
```

```
Bad key savefig.frameon in file /Users/dallasgold/opt/anaconda3/envs/pyvizenv/
lib/python3.7/site-packages/matplotlib/mpl-data/stylelib/_classic_test.mplstyl
e, line 421 ('savefig.frameon : True')
You probably need to get an updated matplotlibrc file from
https://github.com/matplotlib/matplotlib/blob/v3.3.4/matplotlibrc.template
or from the matplotlib source distribution

Bad key verbose.level in file /Users/dallasgold/opt/anaconda3/envs/pyvizenv/li
b/python3.7/site-packages/matplotlib/mpl-data/stylelib/_classic_test.mplstyle,
line 472 ('verbose.level  : silent      # one of silent, helpful, debug, debug
-annoying')
You probably need to get an updated matplotlibrc file from
https://github.com/matplotlib/matplotlib/blob/v3.3.4/matplotlibrc.template
or from the matplotlib source distribution

Bad key verbose.fileo in file /Users/dallasgold/opt/anaconda3/envs/pyvizenv/li
b/python3.7/site-packages/matplotlib/mpl-data/stylelib/_classic_test.mplstyle,
line 473 ('verbose.fileo  : sys.stdout  # a log filename, sys.stdout or sys.st
derr')
You probably need to get an updated matplotlibrc file from
https://github.com/matplotlib/matplotlib/blob/v3.3.4/matplotlibrc.template
or from the matplotlib source distribution
In /Users/dallasgold/opt/anaconda3/envs/pyvizenv/lib/python3.7/site-packages/m
atplotlib/mpl-data/stylelib/_classic_test.mplstyle:
The text.latex.preview rcparam was deprecated in Matplotlib 3.3 and will be re
moved two minor releases later.
In /Users/dallasgold/opt/anaconda3/envs/pyvizenv/lib/python3.7/site-packages/m
atplotlib/mpl-data/stylelib/_classic_test.mplstyle:
The mathtext.fallback_to_cm rcparam was deprecated in Matplotlib 3.3 and will
be removed two minor releases later.
In /Users/dallasgold/opt/anaconda3/envs/pyvizenv/lib/python3.7/site-packages/m
atplotlib/mpl-data/stylelib/_classic_test.mplstyle: Support for setting the 'm
athtext.fallback_to_cm' rcParam is deprecated since 3.3 and will be removed tw
o minor releases later; use 'mathtext.fallback : 'cm' instead.
In /Users/dallasgold/opt/anaconda3/envs/pyvizenv/lib/python3.7/site-packages/m
atplotlib/mpl-data/stylelib/_classic_test.mplstyle:
The validate_bool_maybe_none function was deprecated in Matplotlib 3.3 and wil
l be removed two minor releases later.
In /Users/dallasgold/opt/anaconda3/envs/pyvizenv/lib/python3.7/site-packages/m
atplotlib/mpl-data/stylelib/_classic_test.mplstyle:
The savefig.jpeg_quality rcparam was deprecated in Matplotlib 3.3 and will be
removed two minor releases later.
In /Users/dallasgold/opt/anaconda3/envs/pyvizenv/lib/python3.7/site-packages/m
atplotlib/mpl-data/stylelib/_classic_test.mplstyle:
The keymap.all_axes rcparam was deprecated in Matplotlib 3.3 and will be remov
ed two minor releases later.
In /Users/dallasgold/opt/anaconda3/envs/pyvizenv/lib/python3.7/site-packages/m
atplotlib/mpl-data/stylelib/_classic_test.mplstyle:
The animation.avconv_path rcparam was deprecated in Matplotlib 3.3 and will be
removed two minor releases later.
In /Users/dallasgold/opt/anaconda3/envs/pyvizenv/lib/python3.7/site-packages/m
atplotlib/mpl-data/stylelib/_classic_test.mplstyle:
The animation.avconv_args rcparam was deprecated in Matplotlib 3.3 and will be
removed two minor releases later.
```

In [54]:
```
# Read the Mapbox API key
load_dotenv()
map_box_api = os.getenv("mapbox")
px.set_mapbox_access_token('map_box_api')
```

Import Data

In [7]:
```
# Import the necessary CSVs to Pandas DataFrames
file_path = Path("Data/sfo_neighborhoods_census_data.csv")
sfo_data_df = pd.read_csv(file_path, index_col="year")

file_path = Path("Data/neighborhoods_coordinates.csv")
neighborhood_location_df = pd.read_csv(file_path)
```

Panel Visualizations

In this section, you will copy the code for each plot type from your analysis notebook and place it into separate functions that Panel can use to create panes for the dashboard.

These functions will convert the plot object to a Panel pane.

Be sure to include any DataFrame transformation/manipulation code required along with the plotting code.

Return a Panel pane object from each function that can be used to build the dashboard.

Note: Remove any `.show()` lines from the code. We want to return the plots instead of showing them. The Panel dashboard will then display the plots.

In [56]:
```
# Define Panel Visualization Functions
def housing_units_per_year():
    """Housing Units Per Year."""

    houses = sfo_data_df[['housing_units']].groupby('year').mean()

    title = 'Housing Units Sold in San Francisco from 2010 to 2016'
    annual_homes_sold = houses.hvplot.bar(
        x='year',
        y='housing_units',
        title=title,
        xlabel='Year',
        ylabel='Houses Sold',
        ylim = (370000,388000)
    ).opts(yformatter="%.0f")

    return annual_homes_sold

def average_gross_rent():
    """Average Gross Rent in San Francisco Per Year."""

    gross_rent = sfo_data_df[['gross_rent']].groupby('year').mean()

    rent_title = 'Average Gross Rent in San Francisco by Year'
    avg_gross_rent = gross_rent.hvplot.line(
        x='year',
        y='gross_rent',
        title=rent_title,
        xlabel='Year',
        ylabel='Gross Rent'
    )

    return avg_gross_rent

def average_sales_price():
    """Average Sales Price Per Year."""

    sale_df = sfo_data_df[['sale_price_sqr_foot']].groupby('year').mean()

    sqft_sales_title = 'Average Sale Price per SqFt in San Francisco'
    avg_sale_price = sale_df.hvplot.line(
        x='year',
        y='sale_price_sqr_foot',
        title=sqft_sales_title,
        xlabel='Year',
        ylabel='Avg Sale Price')

    return avg_sale_price

def average_price_by_neighborhood():
    """Average Prices by Neighborhood."""

    hood_df = sfo_data_df.groupby(['year','neighborhood']).mean().reset_index

    hood_plot = hood_df.hvplot(label='Choose Your Adventure: Average Price/Sq
       x='year',
       y='sale_price_sqr_foot',
       xlabel='Year',
       ylabel='Avg Sale Price/SqFt',
       groupby='neighborhood')

    return hood_plot

def top_most_expensive_neighborhoods():
    """Top 10 Most Expensive Neighborhoods."""
    hood_df = sfo_data_df.groupby(['year','neighborhood']).mean().reset_index
    most_expensive = hood_df.nlargest(10,'sale_price_sqr_foot').reset_index()
    top_10_title = 'Top 10 Most Expensive SFO Neighborhoods'
    top_10_sales = most_expensive.hvplot.bar(
        rot=90,
        x='neighborhood',
        y='sale_price_sqr_foot',
        title=top_10_title
    )

    return top_10_sales

def most_expensive_neighborhoods_rent_sales():
    """Comparison of Rent and Sales Prices of Most Expensive Neighborhoods.""

    hood_df = sfo_data_df.groupby(['year','neighborhood']).mean().reset_index
    most_expensive = hood_df.nlargest(10,'gross_rent').reset_index().drop(col

    rent_sales_title = 'Top 10 Most Expensive SFO Neighborhoods'
    top_ten_rent = hood_df.hvplot.bar(
       x='year',
       y=['gross_rent','sale_price_sqr_foot'],
       xlabel='Neighborhood',
       ylabel='Num Housing Units',
       groupby='neighborhood',
       width=800)

    return top_ten_rent


# def parallel_coordinates():
#     """Parallel Coordinates Plot."""

#     return parallel_coordinates_plot

# def parallel_categories():
#     """Parallel Categories Plot."""

#     return parallel_categories_plot

def neighborhood_map():
#     """Neighborhood Map."""

    location_data = Path("Data/neighborhoods_coordinates.csv")
    location_data_df = pd.read_csv(location_data)
    neighborhood_avg = sfo_data_df.groupby('neighborhood',as_index=False).mea

    map_title = 'Sale vs Rent Values in SFO Neighborhoods'
    neighborhood_combined = pd.concat(
        [location_data_df, neighborhood_avg],
        axis="columns",
        join="inner"
        ).drop(columns='neighborhood')
    mapbox_plot = px.scatter_mapbox(
        neighborhood_combined,
        lat='Lat',
        lon='Lon',
        size='sale_price_sqr_foot',
        color='gross_rent',
        color_continuous_scale=px.colors.cyclical.IceFire,
        zoom=11,
        title= map_title,
        width=900)

    return mapbox_plot


# def sunburst():
#     """Sunburst Plot."""

    # YOUR CODE HERE!
```

Panel Dashboard

In this section, you will combine all of the plots into a single dashboard view using Panel. Be creative with your dashboard design!

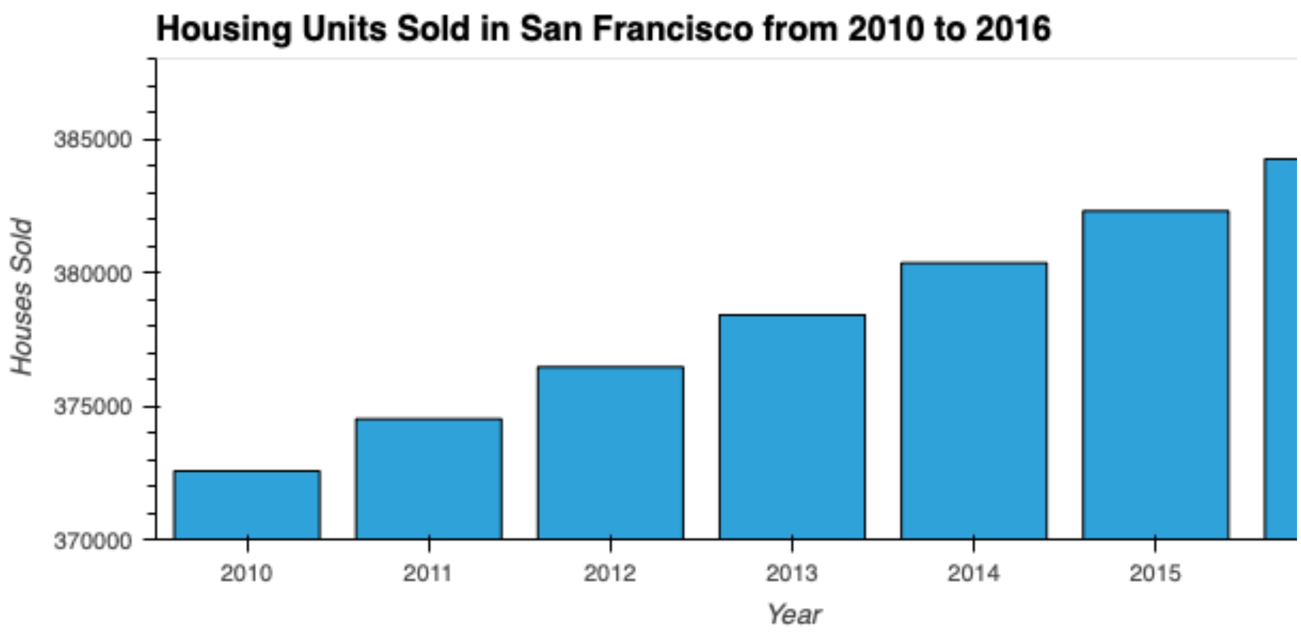
In [ ]:
```
# Create a Title for the Dashboard
# YOUR CODE HERE!


housing_units_column = pn.Column(
    '## Housing Units Sold per Year',
    housing_units_per_year()
)

averages_column = pn.Column(
    'Avg. Gross Rent, Sales Price, and Price by Neighborhood',
    average_gross_rent(),
    average_sales_price(),
    average_price_by_neighborhood()
)

top_ten_column = pn.Column(
    'Top Ten Most Expensive Neighborhoods',
    top_most_expensive_neighborhoods(),
)

parallel_column = pn.Column(
    '## Parallel Coordinates and Categories',
    parallel_coordinates(),
    parallel_categories(),
)

map_column = pn.Column(
    '## Average Values per Neighborhood Maps',
    neighborhood_map()
)

# Create tabs
sfo_dashboard = pn.Tabs(
    ('Housing Units Sold', housing_units_column),
    ('Averages', averages_column),
    ('Most Expensive', top_ten_column),
    ('Parallel Graphs', parallel_column),
    ('Map', map_column)
)
```

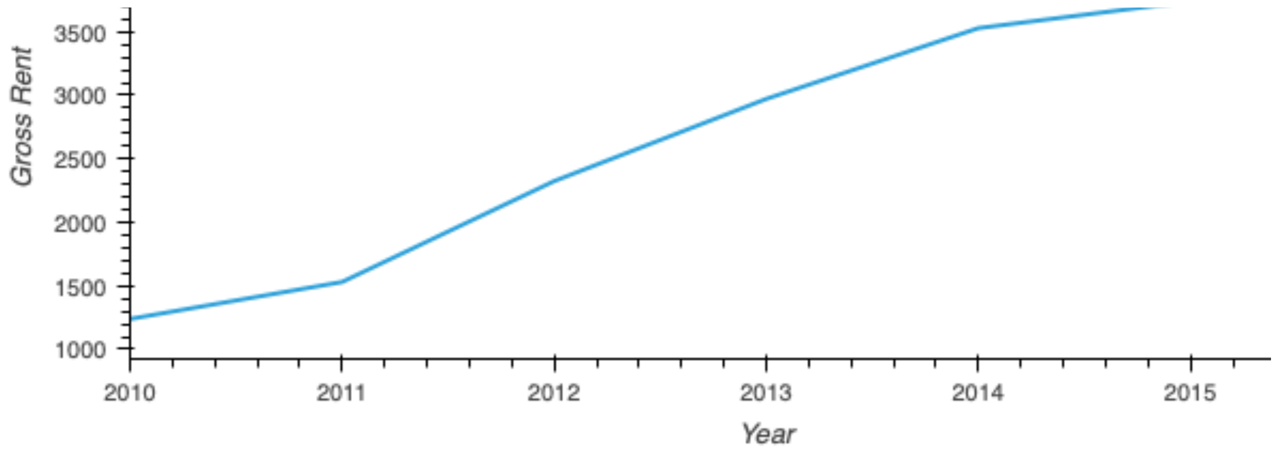
Serve the Panel Dashboard

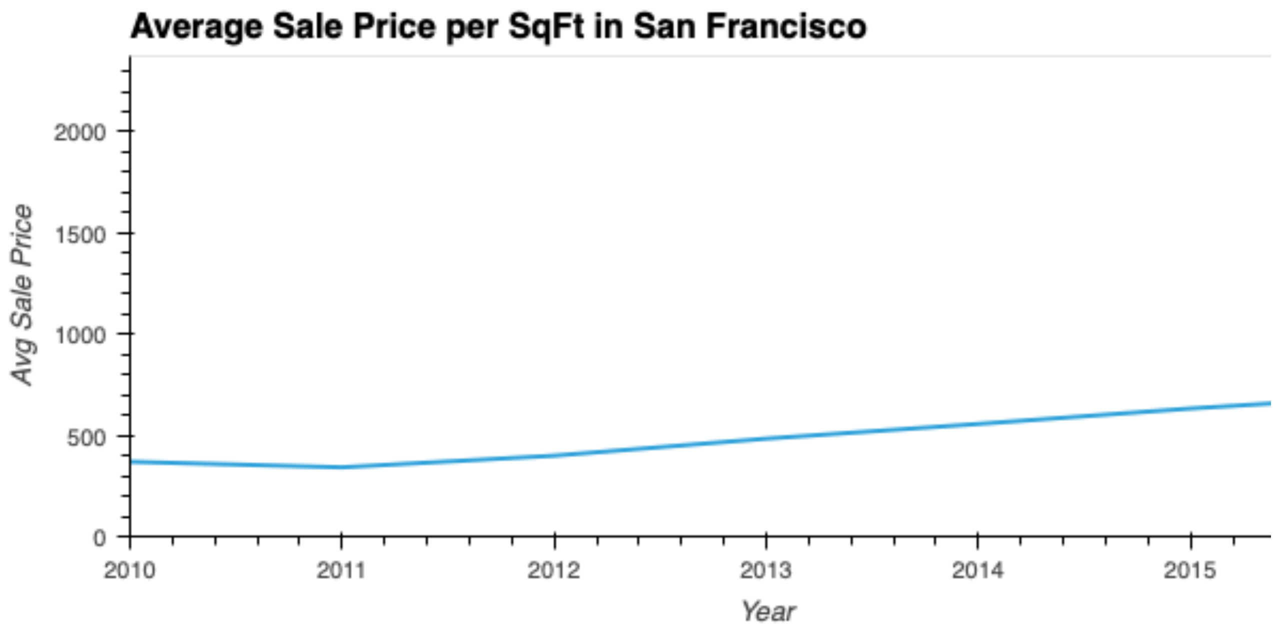
In [58]:
```python
# Serve the# dashboard
left_column = pn.Column(
    "## Static Plots",
    housing_units_per_year(),
    average_gross_rent(),
    average_sales_price(),
    top_most_expensive_neighborhoods()

)

right_column = pn.Column(
    "## Interactive Plots",
    average_price_by_neighborhood(),
    neighborhood_map()
)

dashboard = pn.Tabs(("Static Plots", left_column), ("Interactive Plots", righ

dashboard.servable()
```

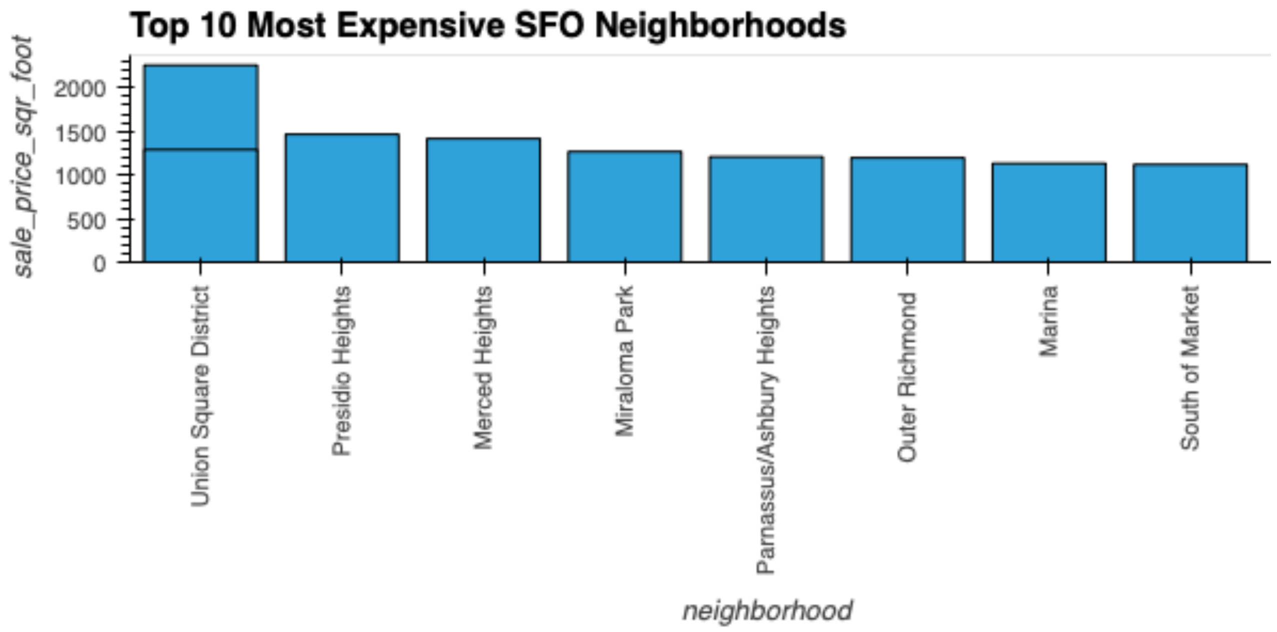
Out[58]: Static Plots | Interactive Plots

Static Plots




Gross Rent
3500
3000
2500
2000
1500
1000
2010
2011
2012
2013
2014
2015
Year


Average Sale Price per SqFt in San Francisco
Avg Sale Price
2000
1500
1000
500
0
2010
2011
2012
2013
2014
2015
Year


Top 10 Most Expensive SFO Neighborhoods
sale_price_sqr_foot
2000
1500
1000
500
0
Union Square District
Presidio Heights
Merced Heights
Miraloma Park
Parnassus/Ashbury Heights
Outer Richmond
Marina
South of Market
neighborhood

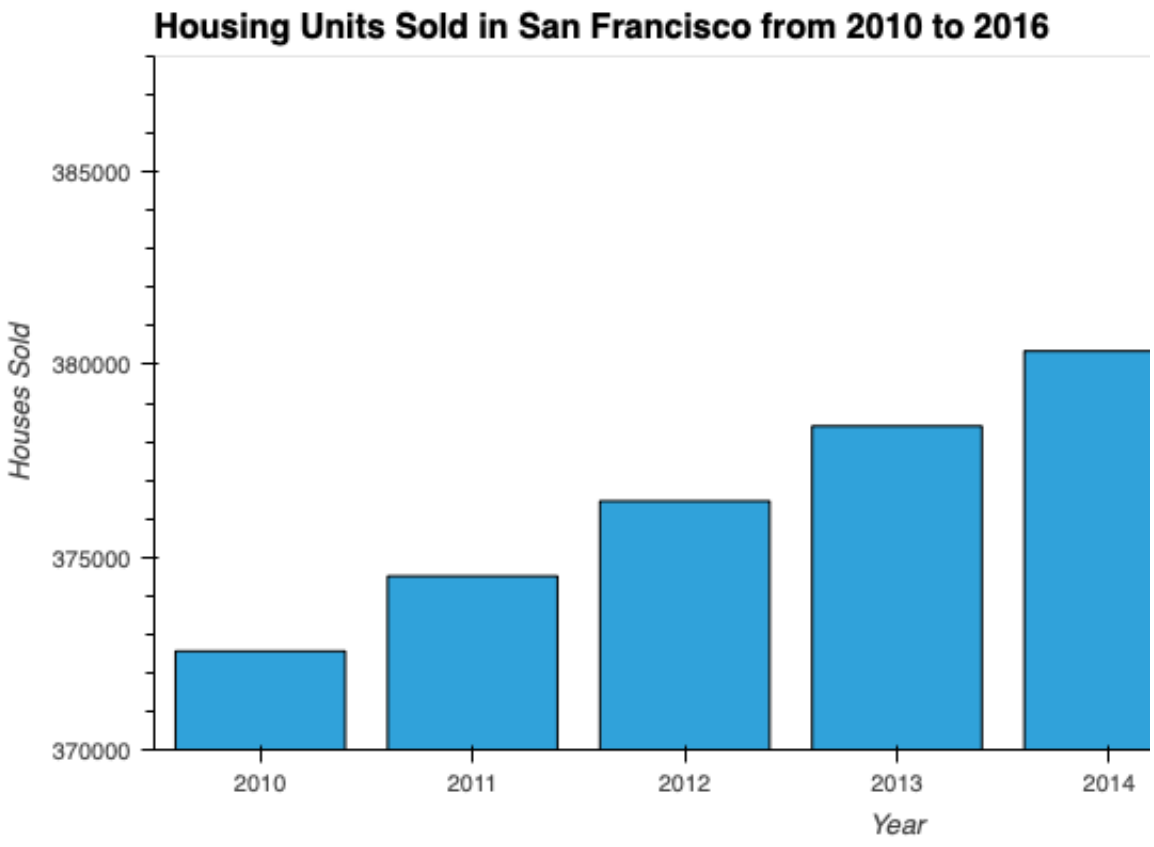
Debugging

Note: Some of the Plotly express plots may not render in the notebook through the panel functions.

However, you can test each plot by uncommenting the following code

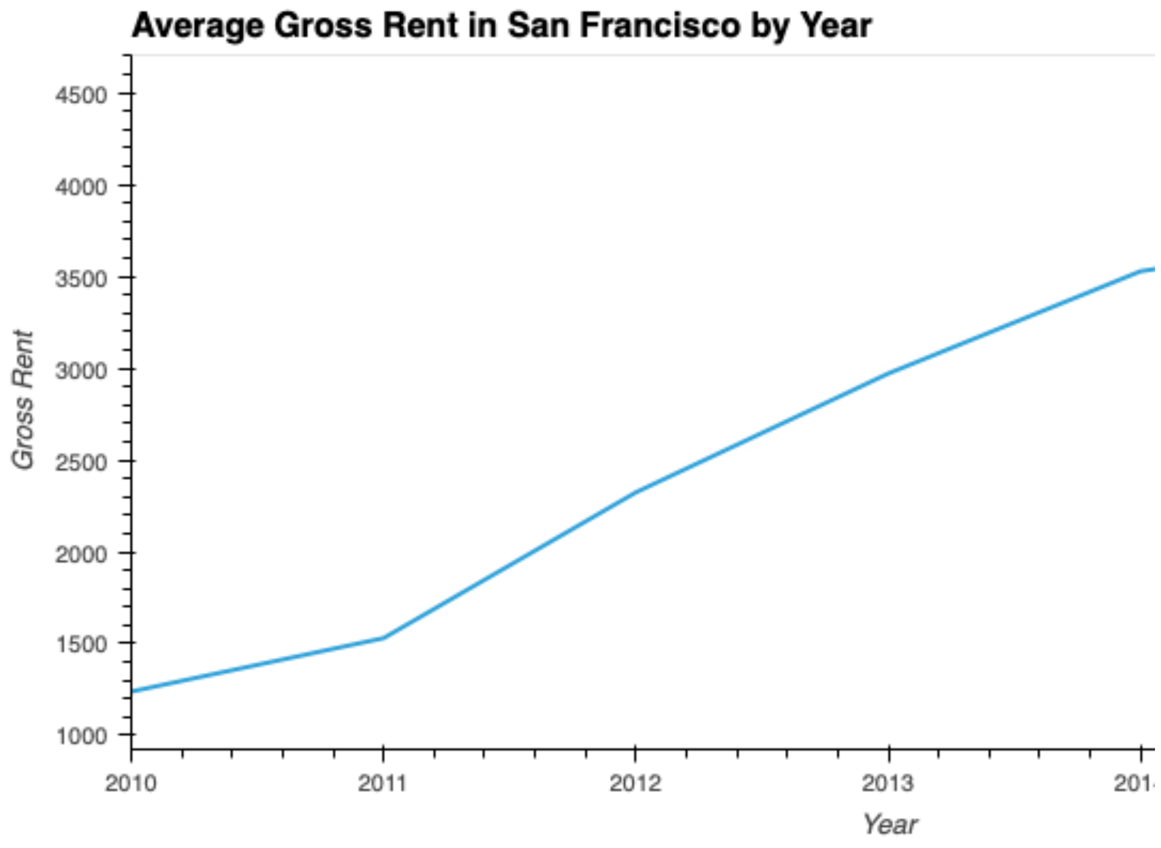
In [11]:
```
housing_units_per_year()
```

Out[11]:



In [13]:
```
average_gross_rent()
```

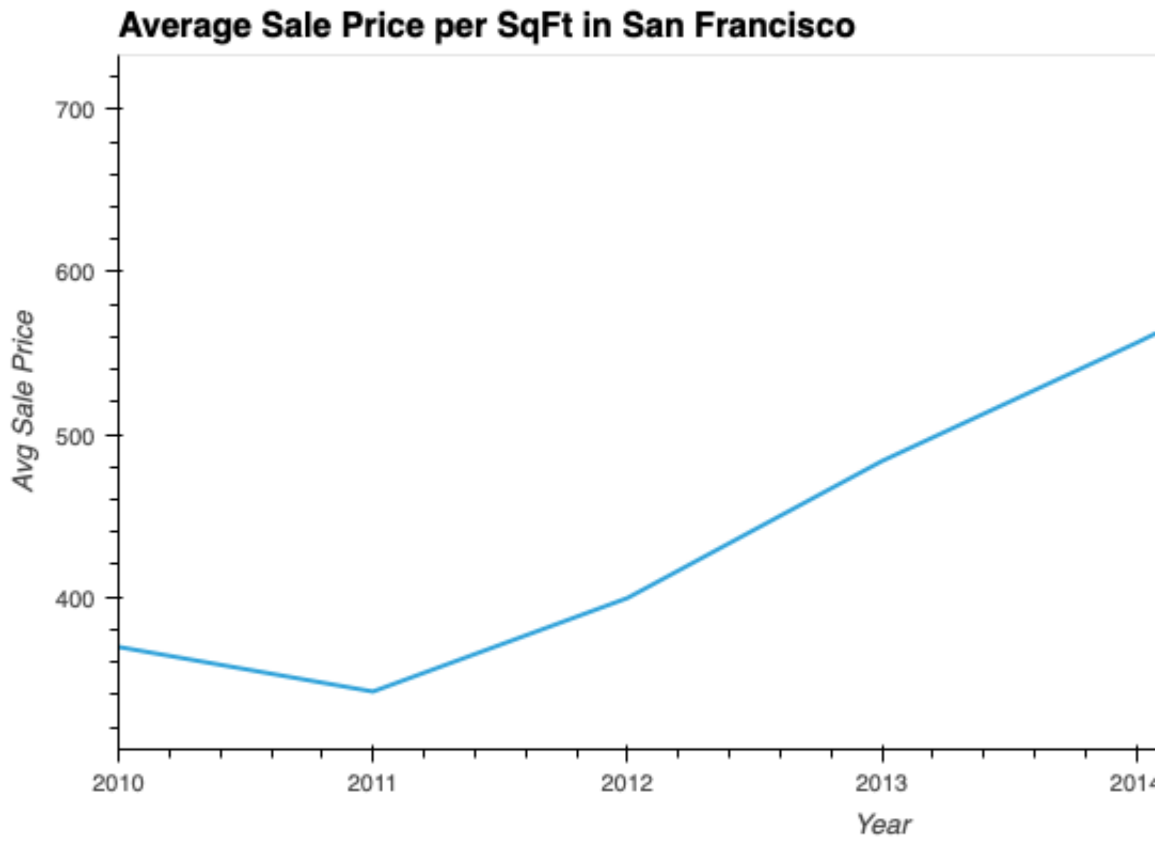
Out[13]:



In [17]:

```
average_sales_price()
```

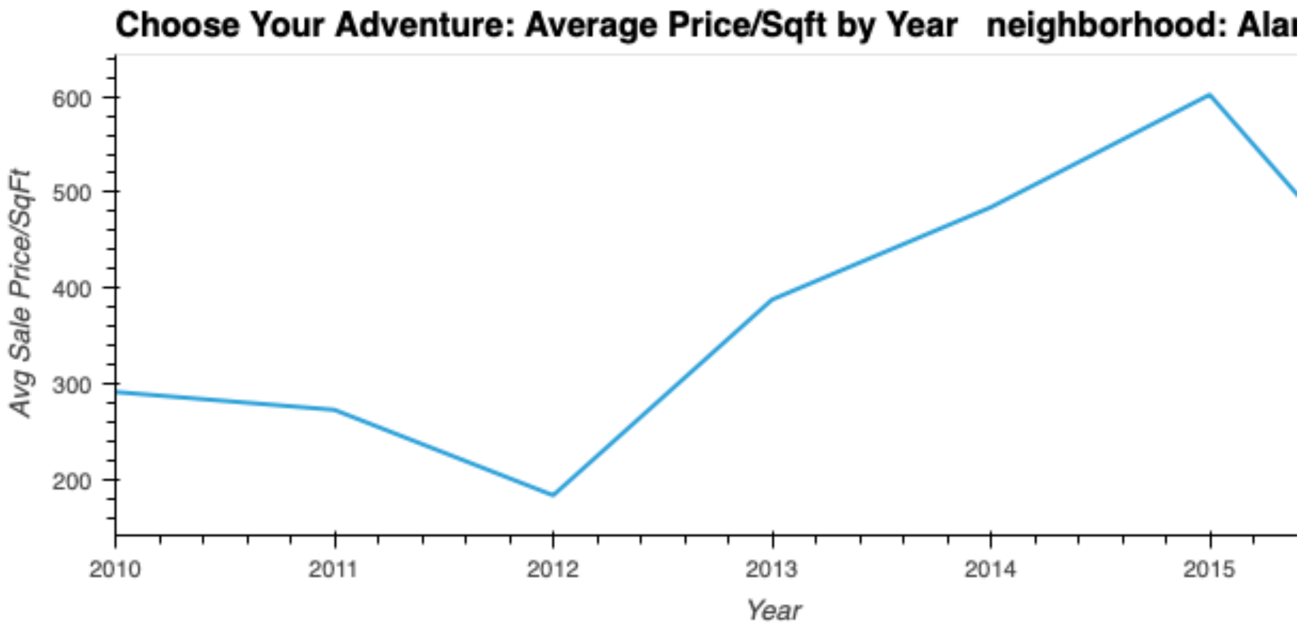
Out[17]:



In [26]:

```
average_price_by_neighborhood()
```

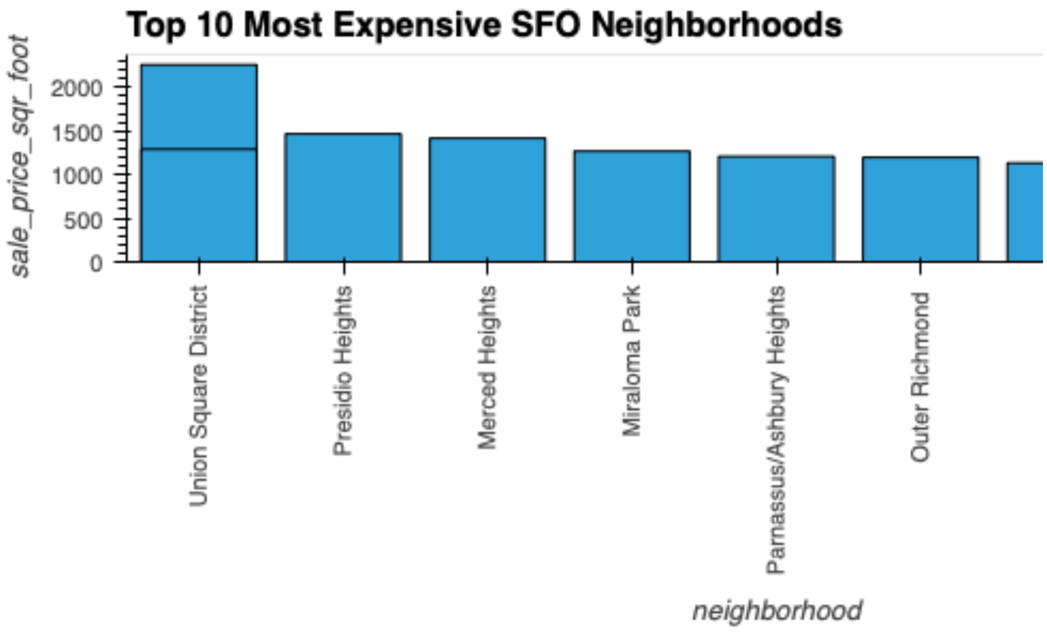
Out[26]:



In [32]:

```
top_most_expensive_neighborhoods()
```

Out[32]:



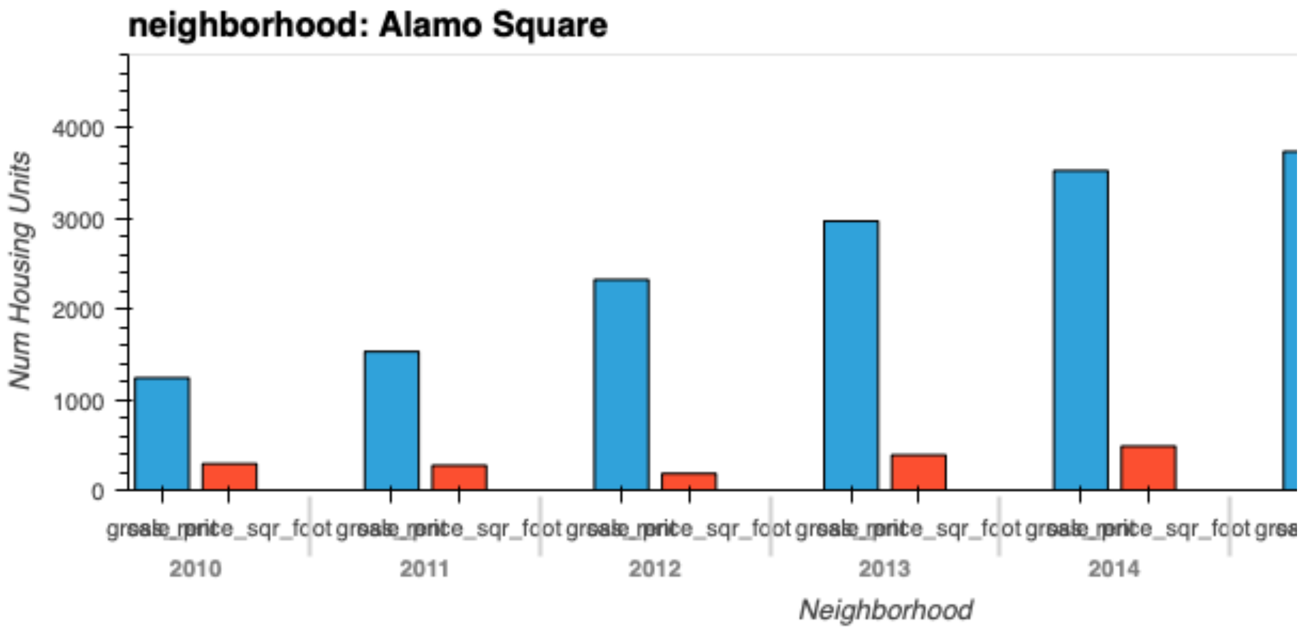
In [ ]:

```

```

In [46]:

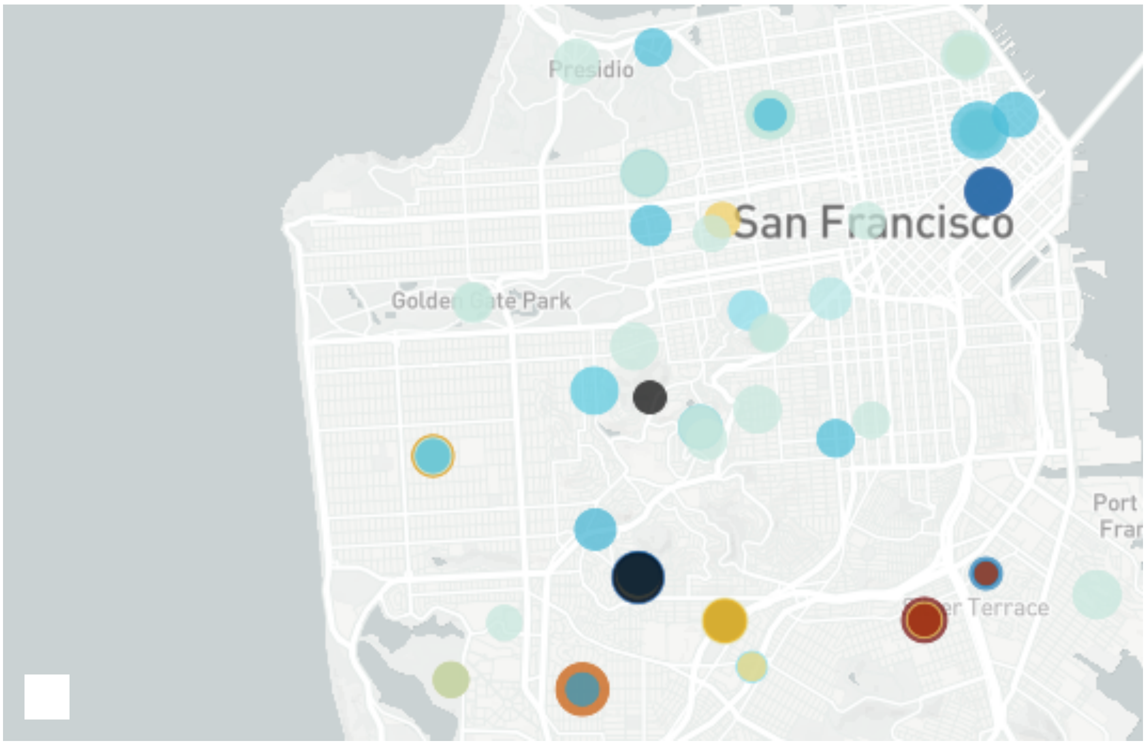
```
most_expensive_neighborhoods_rent_sales()
```

Out[46]:



In [57]:
```
neighborhood_map().show()
```

Sale vs Rent Values in SFO Neighborhoods



In [ ]:
```
# parallel_categories()
```

In [ ]:
```
# parallel_coordinates()
```

In [ ]:
```
# sunburst()
```

In [ ]:
```

```