San Francisco Housing Rental Analysis

In this assignment, you will perform basic analysis for the San Francisco Housing Market to allow potential real estate investors to choose rental investment properties.

In [73]:

```
# initial imports
import os
import pandas as pd
import matplotlib.pyplot as plt
import hvplot.pandas
import plotly.express as px
from pathlib import Path

%matplotlib inline
```

In [160]:

```
# Read the Mapbox API key
mapbox_token = os.getenv("MAPBOX_API_KEY")
px.set_mapbox_access_token(mapbox_token)
```

Load Data

In [75]:

```
# Read the census data into a Pandas DataFrame
file_path = Path("Data/sfo_neighborhoods_census_data.csv")
sfo_data = pd.read_csv(file_path, index_col="year")
sfo_data.head()
```

Out[75]:

year	neighborhood	sale_price_sqr_foot	housing_units	gross_rent
2010	Alamo Square	291.182945	372560	1239
2010	Anza Vista	267.932583	372560	1239
2010	Bayview	170.098665	372560	1239
2010	Buena Vista Park	347.394919	372560	1239
2010	Central Richmond	319.027623	372560	1239

Housing Units Per Year

In this section, you will calculate the number of housing units per year and visualize the results as a bar chart using the Pandas plot function.

Hint: Use the Pandas groupby function

Optional challenge: Use the min, max, and std to scale the y limits of the chart.

In [66]:

```
# Calculate the mean number of housing units per year (hint: use groupby)
# YOUR CODE HERE!
df_sfo_mean = sfo_data.groupby(['year']).mean()
df_sfo_mean.housing_units
```

Out[66]:

```
year
2010    372560
2011    374507
2012    376454
2013    378401
2014    380348
2015    382295
2016    384242
Name: housing_units, dtype: int64
```

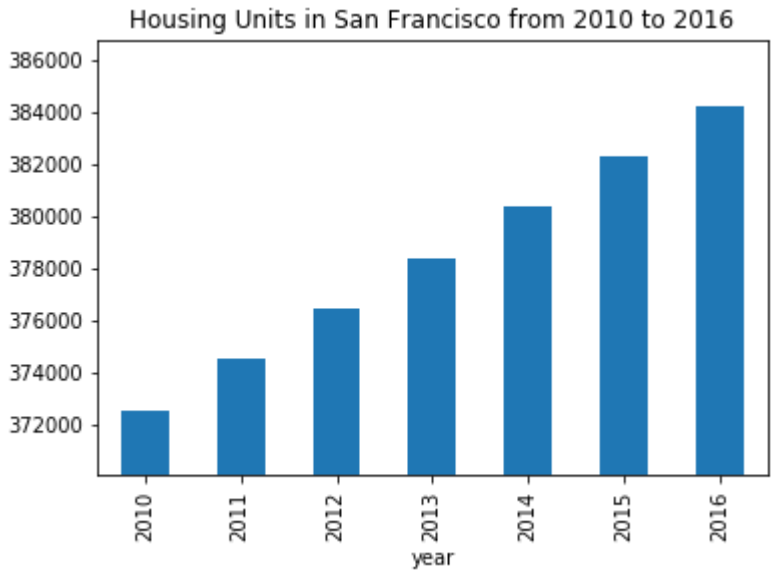
In [67]:

```
# Use the Pandas plot function to plot the average housing units per year.
# Note: You will need to manually adjust the y limit of the chart using the min and max va
lues from above.
# YOUR CODE HERE!
df_sfo_mean_max = df_sfo_mean.max()
df_sfo_mean_min = df_sfo_mean.min()

df_sfo_mean.housing_units.plot.bar(
    x='Year',
    y= 'Housing Units',
    title = 'Housing Units in San Francisco from 2010 to 2016',
    ylim = (df_sfo_mean_min.housing_units - 2500, df_sfo_mean_max.housing_units + 2500)
)
# Optional Challenge: Use the min, max, and std to scale the y limits of the chart
# YOUR CODE HERE!
```

Out[67]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1c9815a5f48>
```



Average Prices per Square Foot

In this section, you will calculate the average gross rent and average sales price for each year. Plot the results as a line chart.

Average Gross Rent in San Francisco Per Year

In [68]:

```
# Calculate the average gross rent and average sale price per square foot
# YOUR CODE HERE!
df_sfo_mean
```

Out[68]:

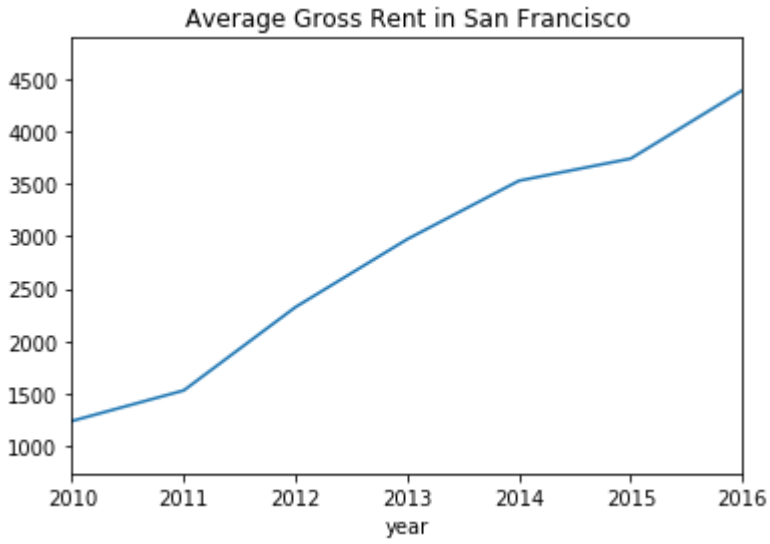
year	sale_price_sqr_foot	housing_units	gross_rent
2010	369.344353	372560	1239
2011	341.903429	374507	1530
2012	399.389968	376454	2324
2013	483.600304	378401	2971
2014	556.277273	380348	3528
2015	632.540352	382295	3739
2016	697.643709	384242	4390

In [57]:

```
# Plot the Average Gross Rent per Year as a Line Chart
# YOUR CODE HERE!
df_sfo_mean.gross_rent.plot.line(
    x='Year',
    y= 'Gross Rent',
    title = 'Average Gross Rent in San Francisco',
    ylim = (df_sfo_mean_min.gross_rent - 500, df_sfo_mean_max.gross_rent + 500)
)
```

Out[57]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1c9814bf608>
```



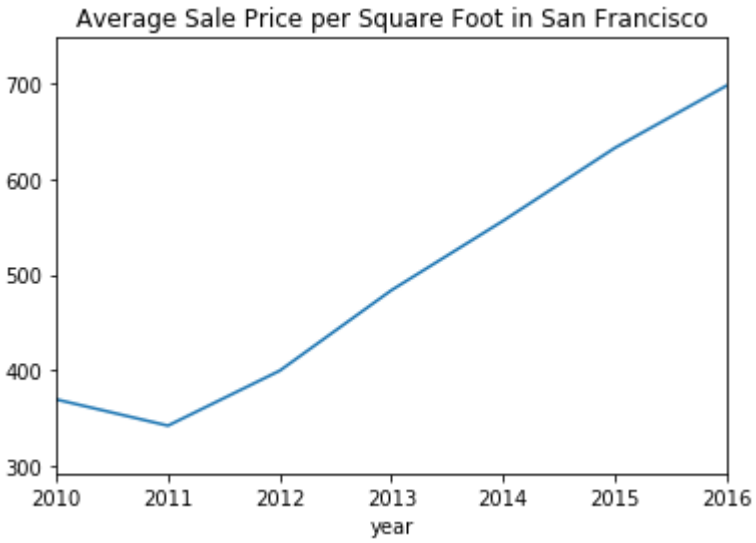
Average Sales Price per Year

In [59]:

```
# Plot the Average Sales Price per Year as a line chart
# YOUR CODE HERE!
df_sfo_mean.sale_price_sqr_foot.plot.line(
    x='Year',
    y= 'Avg. Sale Price',
    title = 'Average Sale Price per Square Foot in San Francisco',
    ylim = (df_sfo_mean_min.sale_price_sqr_foot - 50, df_sfo_mean_max.sale_price_sqr_foot
+ 50)
)
```

Out[59]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1c98024e688>
```



Average Prices by Neighborhood

In this section, you will use hvplot to create an interactive visulization of the Average Prices with a dropdown selector for the neighborhood.

Hint: It will be easier to create a new DataFrame from grouping the data and calculating the mean prices for each year and neighborhood

In [85]:

```
# Group by year and neighborhood and then create a new dataframe of the mean values
# YOUR CODE HERE!
sfo_data_neighborhood = sfo_data.groupby(['year', 'neighborhood']).mean()
sfo_data_neighborhood
```

Out[85]:

		sale_price_sqr_foot	housing_units	gross_rent
year	**neighborhood**			
2010	**Alamo Square**	291.182945	372560	1239
	Anza Vista	267.932583	372560	1239
	Bayview	170.098665	372560	1239
	Buena Vista Park	347.394919	372560	1239
	Central Richmond	319.027623	372560	1239
...	**...**	...	...	...
2016	**Telegraph Hill**	903.049771	384242	4390
	Twin Peaks	970.085470	384242	4390
	Van Ness/ Civic Center	552.602567	384242	4390
	Visitacion Valley	328.319007	384242	4390
	Westwood Park	631.195426	384242	4390

397 rows × 3 columns

In [92]:

```
# Use hvplot to create an interactive line chart of the average price per sq ft.
# The plot should have a dropdown selector for the neighborhood
# YOUR CODE HERE!
sfo_data_neighborhood.sale_price_sqr_foot.hvplot.line(groupby = 'neighborhood')
```

Out[92]:

neighborhood:

The Top 10 Most Expensive Neighborhoods

In this section, you will need to calculate the mean sale price for each neighborhood and then sort the values to obtain the top 10 most expensive neighborhoods on average. Plot the results as a bar chart.

In [103]:

```
# Getting the data from the top 10 expensive neighborhoods
# YOUR CODE HERE!
sfo_group_neighborhoods = sfo_data.groupby(['neighborhood']).mean()
top10_neighborhoods = sfo_group_neighborhoods.sort_values(['sale_price_sqr_foot'], ascendi
ng=False)
df_top10_neighborhoods = top10_neighborhoods.head(10)
df_top10_neighborhoods
```

Out[103]:

neighborhood	sale_price_sqr_foot	housing_units	gross_rent
Union Square District	903.993258	377427.50	2555.166667
Merced Heights	788.844818	380348.00	3414.000000
Miraloma Park	779.810842	375967.25	2155.250000
Pacific Heights	689.555817	378401.00	2817.285714
Westwood Park	687.087575	382295.00	3959.000000
Telegraph Hill	676.506578	378401.00	2817.285714
Presidio Heights	675.350212	378401.00	2817.285714
Cow Hollow	665.964042	378401.00	2817.285714
Potrero Hill	662.013613	378401.00	2817.285714
South Beach	650.124479	375805.00	2099.000000

In [107]:

```
# Plotting the data from the top 10 expensive neighborhoods
# YOUR CODE HERE!
df_top10_neighborhoods.sale_price_sqr_foot.hvplot.bar(rot=90)
```

Out[107]:

Parallel Coordinates and Parallel Categories Analysis

In this section, you will use plotly express to create parallel coordinates and parallel categories visualizations so that investors can interactively filter and explore various factors related to the sales price of the neighborhoods.

Using the DataFrame of Average values per neighborhood (calculated above), create the following visualizations:

1. Create a Parallel Coordinates Plot
2. Create a Parallel Categories Plot

In [115]:

```
# Parallel Coordinates Plot
# YOUR CODE HERE!
px.parallel_coordinates(
    df_top10_neighborhoods,
    color= 'sale_price_sqr_foot',
    color_continuous_scale=px.colors.sequential.Inferno,
)
```

In [117]:

```
# Parallel Categories Plot
# YOUR CODE HERE!
px.parallel_categories(
    df_top10_neighborhoods,
    color= 'sale_price_sqr_foot',
    color_continuous_scale=px.colors.sequential.Inferno,
)
```

Neighborhood Map

In this section, you will read in neighboor location data and build an interactive map with the average prices per neighborhood. Use a scatter_mapbox from plotly express to create the visualization. Remember, you will need your mapbox api key for this.

Load Location Data

In [145]:

```
# Load neighborhoods coordinates data
file_path = Path("Data/neighborhoods_coordinates.csv")
df_neighborhood_locations = pd.read_csv(file_path)
df_neighborhood_locations.rename(columns={'Neighborhood': 'neighborhood'}, inplace=True)
df_neighborhood_locations.tail(5)
```

Out[145]:

	neighborhood	Lat	Lon
68	West Portal	37.74026	-122.463880
69	Western Addition	37.79298	-122.435790
70	Westwood Highlands	37.73470	-122.456854
71	Westwood Park	37.73415	-122.457000
72	Yerba Buena	37.79298	-122.396360

Data Preparation

You will need to join the location data with the mean prices per neighborhood

1. Calculate the mean values for each neighborhood
2. Join the average values with the neighborhood locations

In [139]:

```
# Calculate the mean values for each neighborhood
# YOUR CODE HERE!
df_sfo_all_neighbor = sfo_data.groupby(['neighborhood'], as_index=False).mean()
df_sfo_all_neighbor.tail(5)
```

Out[139]:

	neighborhood	sale_price_sqr_foot	housing_units	gross_rent
68	West Portal	498.488485	376940.75	2515.500000
69	Western Addition	307.562201	377427.50	2555.166667
70	Westwood Highlands	533.703935	376454.00	2250.500000
71	Westwood Park	687.087575	382295.00	3959.000000
72	Yerba Buena	576.709848	377427.50	2555.166667

In [155]:

```
# Join the average values with the neighborhood locations
# YOUR CODE HERE!
df_neighborhood_locations.set_index(['neighborhood'],inplace=True)
df_sfo_all_neighbor.set_index(['neighborhood'], inplace=True)
df_sfo_locations_value = pd.concat([df_neighborhood_locations, df_sfo_all_neighbor], axis=
1, sort=True)
df_sfo_locations_value
```

Out[155]:

	Lat	Lon	sale_price_sqr_foot	housing_units	gross_rent
Alamo Square	37.791012	-122.402100	366.020712	378401.00	2817.285714
Anza Vista	37.779598	-122.443451	373.382198	379050.00	3031.833333
Bayview	37.734670	-122.401060	204.588623	376454.00	2318.400000
Bayview Heights	37.728740	-122.410980	590.792839	382295.00	3739.000000
Bernal Heights	37.728630	-122.443050	NaN	NaN	NaN
...	...	...	...	...	...
West Portal	37.740260	-122.463880	498.488485	376940.75	2515.500000
Western Addition	37.792980	-122.435790	307.562201	377427.50	2555.166667
Westwood Highlands	37.734700	-122.456854	533.703935	376454.00	2250.500000
Westwood Park	37.734150	-122.457000	687.087575	382295.00	3959.000000
Yerba Buena	37.792980	-122.396360	576.709848	377427.50	2555.166667

77 rows × 5 columns

Mapbox Visualization

Plot the aveage values per neighborhood with a plotly express scatter_mapbox visualization.

In [161]:

```
# Create a scatter mapbox to analyze neighborhood info
# YOUR CODE HERE!
px.scatter_mapbox(data_frame=df_sfo_locations_value, lat="Lat", lon="Lon", color="sale_pri
ce_sqr_foot")
```

In [ ]:

```

```