Name :- Vishwas Mishra

Reg No :- 21BCE0959

VIT Vellore

```python
import pandas as pd

# Load the dataset
df = pd.read_csv('/content/House Price India.csv')

df.head()
```

```
           id   Date  number of bedrooms  number of bathrooms  living
area  \
0  6762810145  42491                   5                 2.50
3650
1  6762810635  42491                   4                 2.50
2920
2  6762810998  42491                   5                 2.75
2910
3  6762812605  42491                   4                 2.50
3310
4  6762812919  42491                   3                 2.00
2710

   lot area  number of floors  waterfront present   number of views  \
0      9050               2.0                   0                 4
1      4000               1.5                   0                 0
2      9480               1.5                   0                 0
3     42998               2.0                   0                 0
4      4500               1.5                   0                 0
```

```
   condition of the house   ...   Built Year   Renovation Year   Postal
Code \
0                        5  ...        1921                 0
122003
1                        5  ...        1909                 0
122004
2                        3  ...        1939                 0
122004
3                        3  ...        2001                 0
122005
4                        4  ...        1929                 0
122006

   Lattitude  Longitude  living_area_renov     lot_area_renov  \
0    52.8645   -114.557                2880               5400
1    52.8878   -114.470                2470               4000
2    52.8852   -114.468                2940               6600
3    52.9532   -114.321                3350              42847
4    52.9047   -114.485                2060               4500

   Number of schools nearby  Distance from the airport       Price
0                         2                         58     2380000
1                         2                         51     1400000
2                         1                         53     1200000
3                         3                         76      838000
4                         1                         51      805000

[5 rows x 23 columns]
```
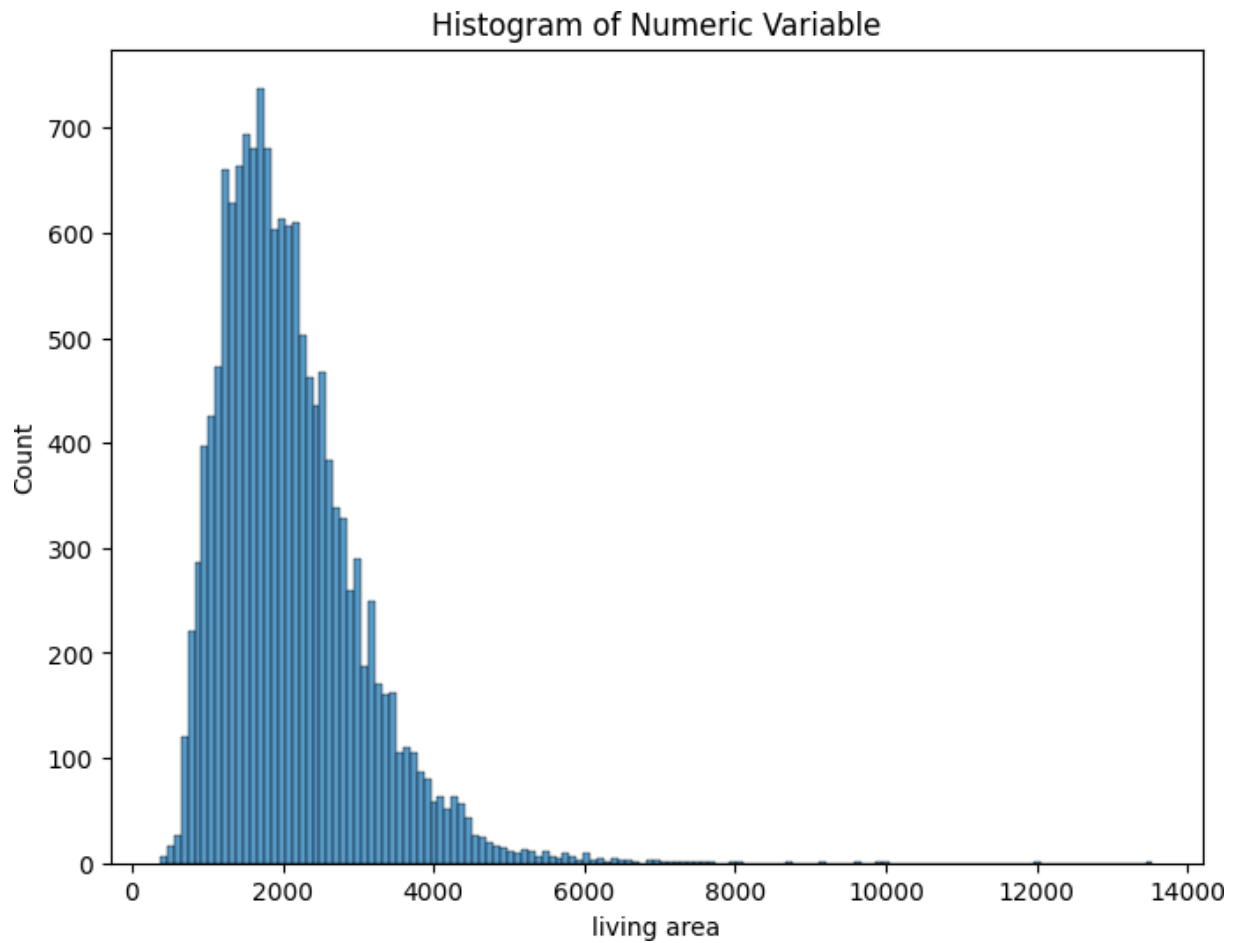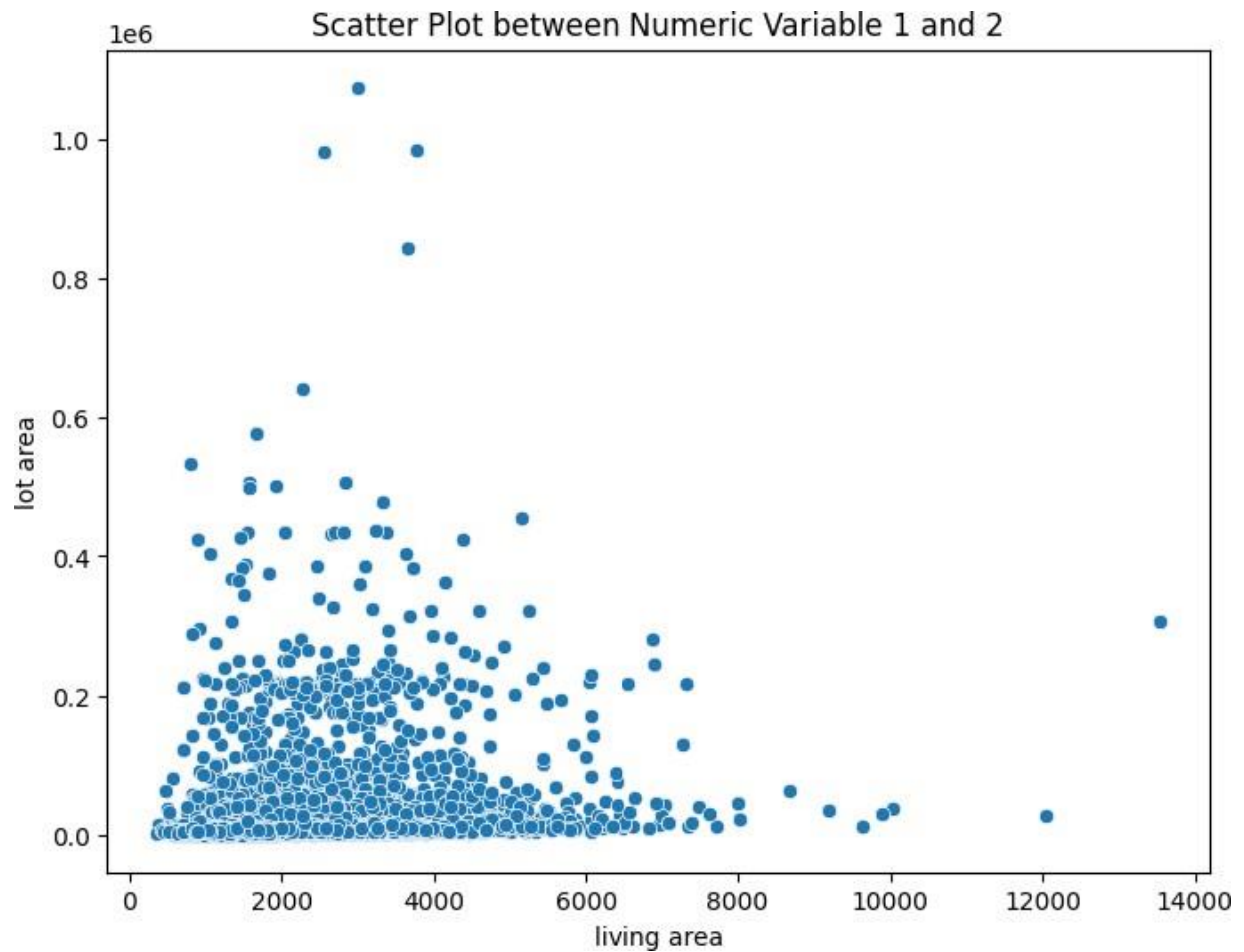
Univariate Analysis

```python
import matplotlib.pyplot as plt
import seaborn as sns

# Example: Histogram for a numeric variable
plt.figure(figsize=(8, 6))
sns.histplot(data=df, x='living area')
plt.title('Histogram of Numeric Variable')
plt.show()
```

Histogram of Numeric Variable

Bivariate Analysis

```python
# Example: Scatter plot between two numeric variables
plt.figure(figsize=(8, 6))
sns.scatterplot(data=df, x='living area', y='lot area')
plt.title('Scatter Plot between Numeric Variable 1 and 2')
plt.show()
```

Scatter Plot between Numeric Variable 1 and 2

Multivariate Analysis

```python
# Example: Correlation heatmap
corr_matrix = df.corr()
plt.figure(figsize=(20,16))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```

Correlation Heatmap
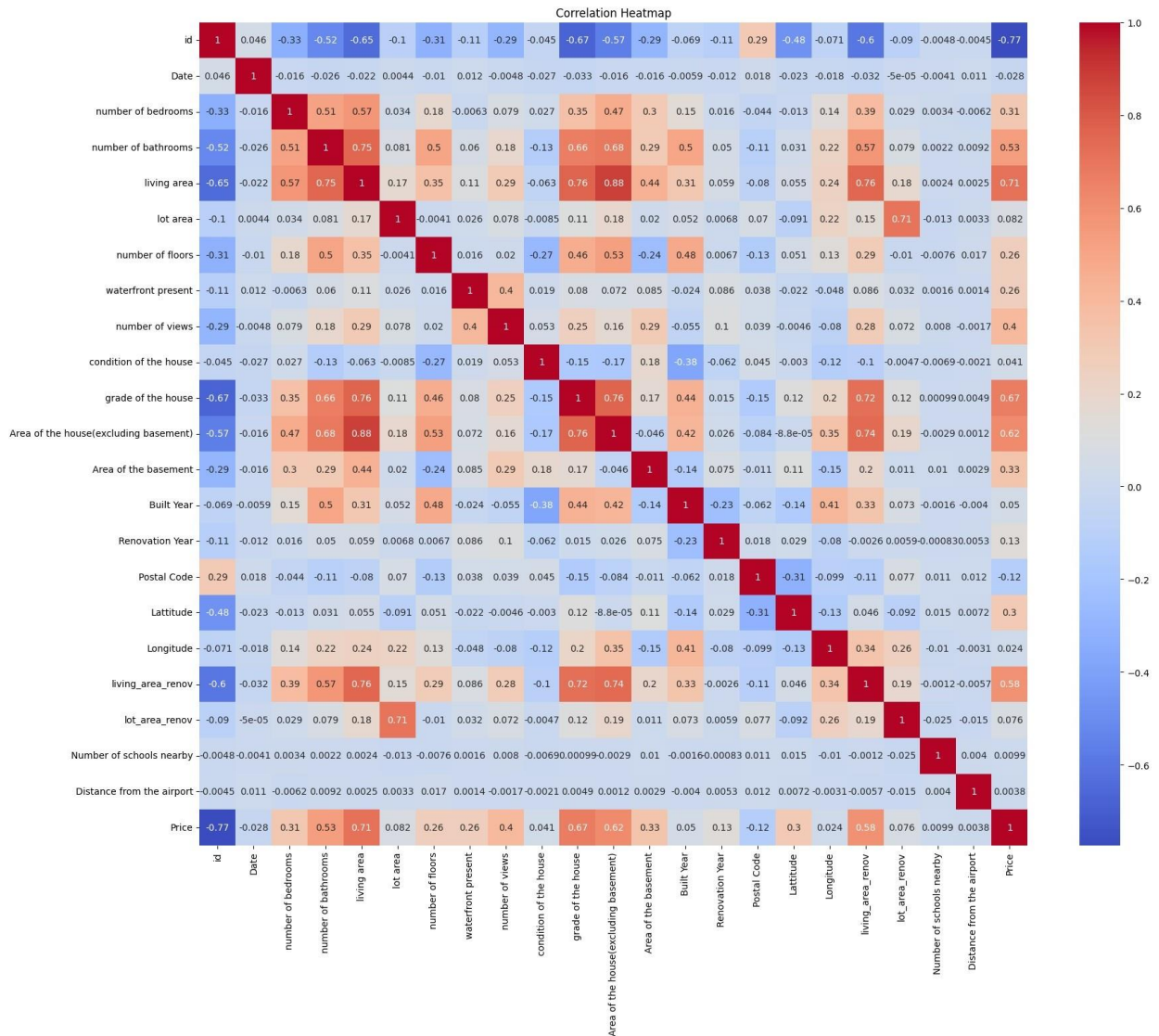
Descriptive Statistics

```
# Example: Descriptive statistics for numeric variables
descriptive_stats = df.describe()
descriptive_stats
```

```
                id          Date    number of bedrooms   number of
bathrooms   \
count   1.462000e+04   14620.000000            14620.000000
14620.000000
mean    6.762821e+09   42604.538646                3.379343
2.129583
std     6.237575e+03      67.347991                0.938719
0.769934
min     6.762810e+09   42491.000000                1.000000
0.500000
```

```
25%     6.762815e+09   42546.000000                    3.000000
1.750000
50%     6.762821e+09   42600.000000                    3.000000
2.250000
75%     6.762826e+09   42662.000000                    4.000000
2.500000
max     6.762832e+09   42734.000000                   33.000000
8.000000

        living area        lot area   number of floors   waterfront
present  \
count    14620.000000    1.462000e+04    14620.000000
14620.000000
mean      2098.262996    1.509328e+04        1.502360
0.007661
std        928.275721    3.791962e+04        0.540239
0.087193
min        370.000000    5.200000e+02        1.000000
0.000000
25%       1440.000000    5.010750e+03        1.000000
0.000000
50%       1930.000000    7.620000e+03        1.500000
0.000000
75%       2570.000000    1.080000e+04        2.000000
0.000000
max      13540.000000    1.074218e+06        3.500000
1.000000

        number of views    condition of the house   ...    Built Year  \
count     14620.000000                14620.000000   ...  14620.000000
mean          0.233105                    3.430506   ...   1970.926402
std           0.766259                    0.664151   ...     29.493625
min           0.000000                    1.000000   ...   1900.000000
25%           0.000000                    3.000000   ...   1951.000000
50%           0.000000                    3.000000   ...   1975.000000
75%           0.000000                    4.000000   ...   1997.000000
max           4.000000                    5.000000   ...   2015.000000

        Renovation Year    Postal Code     Lattitude      Longitude  \
count     14620.000000    14620.000000   14620.000000   14620.000000
mean         90.924008   122033.062244      52.792848    -114.404007
std         416.216661       19.082418       0.137522       0.141326
min           0.000000   122003.000000      52.385900    -114.709000
25%           0.000000   122017.000000      52.707600    -114.519000
50%           0.000000   122032.000000      52.806400    -114.421000
75%           0.000000   122048.000000      52.908900    -114.315000
max        2015.000000   122072.000000      53.007600    -113.505000

        living_area_renov   lot_area_renov   Number of schools nearby  \
count        14620.000000     14620.000000               14620.000000
```

```
mean            1996.702257      12753.500068                         2.012244
std              691.093366      26058.414467                         0.817284
min              460.000000        651.000000                         1.000000
25%             1490.000000       5097.750000                         1.000000
50%             1850.000000       7620.000000                         2.000000
75%             2380.000000      10125.000000                         3.000000
max             6110.000000     560617.000000                         3.000000

        Di;tance from the airport              Price
count                   14620.000000     1.462000e+04
mean                       64.950958     5.389322e+05
std                         8.936008     3.675324e+05
min                        50.000000     7.800000e+04
25%                        57.000000     3.200000e+05
50%                        65.000000     4.500000e+05
75%                        73.000000     6.450000e+05
max                        80.000000     7.700000e+06

[8 rows x 23 columns]

<google.colab._quickchart_helpers.SectionTitle at 0x79cdd6db8190>

import numpy as np
from google.colab import autoviz

def value_plot(df, y, figscale=1):
  from matplotlib import pyplot as plt
  df[y].plot(kind='line', figsize=(8 * figscale, 4 * figscale),
title=y)
  plt.gca().spines[['top', 'right']].set_visible(False)
  plt.tight_layout()
  return autoviz.MplChart.from_current_mpl_state()

chart = value_plot(descriptive_stats, *['id'], **{})
chart

import numpy as np
from google.colab import autoviz

def value_plot(df, y, figscale=1):
  from matplotlib import pyplot as plt
  df[y].plot(kind='line', figsize=(8 * figscale, 4 * figscale),
title=y)
  plt.gca().spines[['top', 'right']].set_visible(False)
  plt.tight_layout()
  return autoviz.MplChart.from_current_mpl_state()

chart = value_plot(descriptive_stats, *['Date'], **{})
chart
```

```python
import numpy as np
from google.colab import autoviz

def value_plot(df, y, figscale=1):
  from matplotlib import pyplot as plt
  df[y].plot(kind='line', figsize=(8 * figscale, 4 * figscale),
title=y)
  plt.gca().spines[['top', 'right']].set_visible(False)
  plt.tight_layout()
  return autoviz.MplChart.from_current_mpl_state()

chart = value_plot(descriptive_stats, *['number of bedrooms'], **{})
chart

import numpy as np
from google.colab import autoviz

def value_plot(df, y, figscale=1):
  from matplotlib import pyplot as plt
  df[y].plot(kind='line', figsize=(8 * figscale, 4 * figscale),
title=y)
  plt.gca().spines[['top', 'right']].set_visible(False)
  plt.tight_layout()
  return autoviz.MplChart.from_current_mpl_state()

chart = value_plot(descriptive_stats, *['number of bathrooms'], **{})
chart

<google.colab._quickchart_helpers.SectionTitle at 0x79cdd670be80>

import numpy as np
from google.colab import autoviz

def histogram(df, colname, num_bins=20, figscale=1):
  from matplotlib import pyplot as plt
  df[colname].plot(kind='hist', bins=num_bins, title=colname,
figsize=(8*figscale, 4*figscale))
  plt.gca().spines[['top', 'right',]].set_visible(False)
  plt.tight_layout()
  return autoviz.MplChart.from_current_mpl_state()

chart = histogram(descriptive_stats, *['id'], **{})
chart

import numpy as np
from google.colab import autoviz

def histogram(df, colname, num_bins=20, figscale=1):
  from matplotlib import pyplot as plt
  df[colname].plot(kind='hist', bins=num_bins, title=colname,
figsize=(8*figscale, 4*figscale))
```

```
  plt.gca().spines[['top', 'right',]].set_visible(False)
  plt.tight_layout()
  return autoviz.MplChart.from_current_mpl_state()

chart = histogram(descriptive_stats, *['Date'], **{})
chart

import numpy as np
from google.colab import autoviz

def histogram(df, colname, num_bins=20, figscale=1):
  from matplotlib import pyplot as plt
  df[colname].plot(kind='hist', bins=num_bins, title=colname,
figsize=(8*figscale, 4*figscale))
  plt.gca().spines[['top', 'right',]].set_visible(False)
  plt.tight_layout()
  return autoviz.MplChart.from_current_mpl_state()

chart = histogram(descriptive_stats, *['number of bedrooms'], **{})
chart

import numpy as np
from google.colab import autoviz

def histogram(df, colname, num_bins=20, figscale=1):
  from matplotlib import pyplot as plt
  df[colname].plot(kind='hist', bins=num_bins, title=colname,
figsize=(8*figscale, 4*figscale))
  plt.gca().spines[['top', 'right',]].set_visible(False)
  plt.tight_layout()
  return autoviz.MplChart.from_current_mpl_state()

chart = histogram(descriptive_stats, *['number of bathrooms'], **{})
chart

<google.colab._quickchart_helpers.SectionTitle at 0x79cdd8fa6ec0>

import numpy as np
from google.colab import autoviz

def scatter_plots(df, colname_pairs, figscale=1, alpha=.8):
  from matplotlib import pyplot as plt
  plt.figure(figsize=(len(colname_pairs) * 6 * figscale, 6 *
figscale))
  for plot_i, (x_colname, y_colname) in enumerate(colname_pairs,
start=1):
    ax = plt.subplot(1, len(colname pairs), plot i)
    df.plot(kind='scatter', x=x_colname, y=y_colname, s=(32 *
figscale), alpha=alpha, ax=ax)
    ax.spines[['top', 'right',]].set_visible(False)
  plt.tight_layout()
```

```
  return autoviz.MplChart.from_current_mpl_state()

chart = scatter_plots(descriptive_stats, *[[['id', 'Date'], ['Date',
'number of bedrooms'], ['number of bedrooms', 'number of bathrooms'],
['number of bathrooms', 'living area']]], **{})
chart

<google.colab._quickchart_helpers.SectionTitle at 0x79cddb5c8610>

import numpy as np
from google.colab import autoviz

def time_series_multiline(df, timelike_colname, value_colname,
series_colname, figscale=1, mpl_palette_name='Dark2'):
  from matplotlib import pyplot as plt
  import seaborn as sns
  figsize = (10 * figscale, 5.2 * figscale)
  palette = list(sns.palettes.mpl_palette(mpl_palette_name))
  def _plot_series(series, series_name, series_index=0):
    if value_colname == 'count()':
      counted = (series[timelike_colname]
                  .value_counts()
                  .reset_index(name='counts')
                  .rename({'index': timelike_colname}, axis=1)
                  .sort_values(timelike_colname, ascending=True))
      xs = counted[timelike_colname]
      ys = counted['counts']
    else:
      xs = series[timelike_colname]
      ys = series[value_colname]
    plt.plot(xs, ys, label=series_name, color=palette[series_index %
len(palette)])

  fig, ax = plt.subplots(figsize=figsize, layout='constrained')
  df = df.sort_values(timelike_colname, ascending=True)
  if series_colname:
    for i, (series_name, series) in
enumerate(df.groupby(series_colname)):
      _plot_series(series, series_name, i)
    fig.legend(title=series_colname, bbox_to_anchor=(1, 1), loc='upper
left')
  else:
    _plot_series(df, '')
  sns.despine(fig=fig, ax=ax)
  plt.xlabel(timelike_colname)
  plt.ylabel(value_colname)
  return autoviz.MplChart.from_current_mpl_state()

chart = time_series_multiline(descriptive_stats, *['Date', 'id',
```

```
None], **{})
chart

import numpy as np
from google.colab import autoviz

def time_series_multiline(df, timelike_colname, value_colname,
series_colname, figscale=1, mpl_palette_name='Dark2'):
  from matplotlib import pyplot as plt
  import seaborn as sns
  figsize = (10 * figscale, 5.2 * figscale)
  palette = list(sns.palettes.mpl_palette(mpl_palette_name))
  def _plot_series(series, series_name, series_index=0):
    if value_colname == 'count()':
      counted = (series[timelike_colname]
                 .value_counts()
                 .reset_index(name='counts')
                 .rename({'index': timelike_colname}, axis=1)
                 .sort_values(timelike_colname, ascending=True))
      xs = counted[timelike_colname]
      ys = counted['counts']
    else:
      xs = series[timelike_colname]
      ys = series[value_colname]
    plt.plot(xs, ys, label=series_name, color=palette[series_index %
len(palette)])

  fig, ax = plt.subplots(figsize=figsize, layout='constrained')
  df = df.sort_values(timelike_colname, ascending=True)
  if series_colname:
    for i, (series_name, series) in
enumerate(df.groupby(series_colname)):
      _plot_series(series, series_name, i)
    fig.legend(title=series_colname, bbox_to_anchor=(1, 1), loc='upper
left')
  else:
    _plot_series(df, '')
  sns.despine(fig=fig, ax=ax)
  plt.xlabel(timelike_colname)
  plt.ylabel(value_colname)
  return autoviz.MplChart.from_current_mpl_state()

chart = time_series_multiline(descriptive_stats, *['Date', 'number of
bedrooms', None], **{})
chart

import numpy as np
from google.colab import autoviz

def time_series_multiline(df, timelike_colname, value_colname,
```

```
series_colname, figscale=1, mpl_palette_name='Dark2'):
  from matplotlib import pyplot as plt
  import seaborn as sns
  figsize = (10 * figscale, 5.2 * figscale)
  palette = list(sns.palettes.mpl_palette(mpl_palette_name))
  def _plot_series(series, series_name, series_index=0):
    if value_colname == 'count()':
      counted = (series[timelike_colname]
                  .value_counts()
                  .reset_index(name='counts')
                  .rename({'index': timelike_colname}, axis=1)
                  .sort_values(timelike_colname, ascending=True))
      xs = counted[timelike_colname]
      ys = counted['counts']
    else:
      xs = series[timelike_colname]
      ys = series[value_colname]
    plt.plot(xs, ys, label=series_name, color=palette[series_index %
len(palette)])

  fig, ax = plt.subplots(figsize=figsize, layout='constrained')
  df = df.sort_values(timelike_colname, ascending=True)
  if series_colname:
    for i, (series_name, series) in
enumerate(df.groupby(series_colname)):
      _plot_series(series, series_name, i)
    fig.legend(title=series_colname, bbox_to_anchor=(1, 1), loc='upper
left')
  else:
    _plot_series(df, '')
  sns.despine(fig=fig, ax=ax)
  plt.xlabel(timelike_colname)
  plt.ylabel(value_colname)
  return autoviz.MplChart.from_current_mpl_state()

chart = time_series_multiline(descriptive_stats, *['Date', 'number of
bathrooms', None], **{})
chart

import numpy as np
from google.colab import autoviz

def time_series_multiline(df, timelike_colname, value_colname,
series_colname, figscale=1, mpl_palette_name='Dark2'):
  from matplotlib import pyplot as plt
  import seaborn as sns
  figsize = (10 * figscale, 5.2 * figscale)
  palette = list(sns.palettes.mpl_palette(mpl_palette_name))
  def _plot_series(series, series_name, series_index=0):
    if value_colname == 'count()':
```

```
        counted = (series[timelike_colname]
                    .value_counts()
                    .reset_index(name='counts')
                    .rename({'index': timelike_colname}, axis=1)
                    .sort_values(timelike_colname, ascending=True))
        xs = counted[timelike_colname]
        ys = counted['counts']
    else:
        xs = series[timelike_colname]
        ys = series[value_colname]
    plt.plot(xs, ys, label=series_name, color=palette[series_index %
len(palette)])

  fig, ax = plt.subplots(figsize=figsize, layout='constrained')
  df = df.sort_values(timelike_colname, ascending=True)
  if series_colname:
    for i, (series_name, series) in
enumerate(df.groupby(series_colname)):
        _plot_series(series, series_name, i)
    fig.legend(title=series_colname, bbox_to_anchor=(1, 1), loc='upper
left')
  else:
    _plot_series(df, '')
  sns.despine(fig=fig, ax=ax)
  plt.xlabel(timelike_colname)
  plt.ylabel(value_colname)
  return autoviz.MplChart.from_current_mpl_state()

chart = time_series_multiline(descriptive_stats, *['Date', 'living
area', None], **{})
chart
```

Handle Missing Values

```
# Example: Handling missing values
df.fillna(df.mean(), inplace=True)
df.fillna

<bound method DataFrame.fillna of                       id    Date number of
bedrooms number of bathrooms \
0      6762810145  42491                      5                      2.50
1      6762810635  42491                      4                      2.50
2      6762810998  42491                      5                      2.75
3      6762812605  42491                      4                      2.50
4      6762812919  42491                      3                      2.00
...            ...    ...                    ...                      ...
14615  6762830250  42734                      2                      1.50
14616  6762830339  42734                      3                      2.00
14617  6762830618  42734                      2                      1.00
```

```
14618  6762830709  42734                          4                        1.00
14619  6762831463  42734                          3                        1.00

       living area  lot area  number of floors  waterfront present  \
0             3650      9050               2.0                   0
1             2920      4000               1.5                   0
2             2910      9480               1.5                   0
3             3310     42998               2.0                   0
4             2710      4500               1.5                   0
...            ...       ...               ...                 ...
14615         1556     20000               1.0                   0
14616         1680      7000               1.5                   0
14617         1070      6120               1.0                   0
14618         1030      6621               1.0                   0
14619          900      4770               1.0                   0

       number of views  condition of the house  ...  Built Year
0                    4                       5  ...        1921
1                    0                       5  ...        1909
2                    0                       3  ...        1939
3                    0                       3  ...        2001
4                    0                       4  ...        1929
...                ...                     ...  ...         ...
14615                0                       4  ...        1957
14616                0                       4  ...        1968
14617                0                       3  ...        1962
14618                0                       4  ...        1955
14619                0                       3  ...        1969

       Renovation Year  Postal Code  Lattitude  Longitude
living_area_renov  \
0                    0       122003    52.8645   -114.557
2880
1                    0       122004    52.8878   -114.470
2470
2                    0       122004    52.8852   -114.468
2940
3                    0       122005    52.9532   -114.321
3350
4                    0       122006    52.9047   -114.485
2060
...                ...          ...        ...        ...
...
14615                0       122066    52.6191   -114.472
2250
14616                0       122072    52.5075   -114.393
1540
14617                0       122056    52.7289   -114.507
1130
14618                0       122042    52.7157   -114.411
```

```
1420
14619                 2009        122018      52.5338    -114.552
900

              _    _                              _

                                                  Price
0         2380000
1         1400000
2         1200000
3          838000
4          805000
...           ...
14615     221700
14616     219200
14617     209000
14618     205000
14619     146000

[14620 rows x 23 columns]>
```

Thank you!