```
from sklearn.datasets import fetch_california_housing
import pandas as pd
housing = fetch_california_housing(as_frame=True)
df = housing.frame
# Now df is your full California Housing dataset!
print(df.head())
```

```
   MedInc  HouseAge  AveRooms  AveBedrms  Population  AveOccup  Latitude  \
0  8.3252      41.0  6.984127   1.023810       322.0  2.555556     37.88
1  8.3014      21.0  6.238137   0.971880      2401.0  2.109842     37.86
2  7.2574      52.0  8.288136   1.073446       496.0  2.802260     37.85
3  5.6431      52.0  5.817352   1.073059       558.0  2.547945     37.85
4  3.8462      52.0  6.281853   1.081081       565.0  2.181467     37.85

   Longitude  MedHouseVal
0    -122.23        4.526
1    -122.22        3.585
2    -122.24        3.521
3    -122.25        3.413
4    -122.25        3.422
```

```
df.head()   #Display the first 5 rows
```

	MedInc	HouseAge	AveRooms	AveBedrms	Population	AveOccup	Latitude	Longitude	MedHouseVal
0	8.3252	41.0	6.984127	1.023810	322.0	2.555556	37.88	-122.23	4.526
1	8.3014	21.0	6.238137	0.971880	2401.0	2.109842	37.86	-122.22	3.585
2	7.2574	52.0	8.288136	1.073446	496.0	2.802260	37.85	-122.24	3.521
3	5.6431	52.0	5.817352	1.073059	558.0	2.547945	37.85	-122.25	3.413
4	3.8462	52.0	6.281853	1.081081	565.0	2.181467	37.85	-122.25	3.422

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
df.info()     #Print the column names and their data types
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20640 entries, 0 to 20639
Data columns (total 9 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   MedInc       20640 non-null  float64
 1   HouseAge     20640 non-null  float64
 2   AveRooms     20640 non-null  float64
 3   AveBedrms    20640 non-null  float64
 4   Population   20640 non-null  float64
 5   AveOccup     20640 non-null  float64
 6   Latitude     20640 non-null  float64
 7   Longitude    20640 non-null  float64
 8   MedHouseVal  20640 non-null  float64
dtypes: float64(9)
memory usage: 1.4 MB
```

```
print(df.isnull().sum())      #Check for missing values in the dataset
```

```
MedInc         0
HouseAge       0
AveRooms       0
AveBedrms      0
Population     0
AveOccup       0
Latitude       0
Longitude      0
MedHouseVal    0
dtype: int64
```

```
df.describe()      #Get basic statistical summaries using .describe()
```

	MedInc	HouseAge	AveRooms	AveBedrms	Population	AveOccup	Latitude	Longitude	MedHouseVal
count	20640.000000	20640.000000	20640.000000	20640.000000	20640.000000	20640.000000	20640.000000	20640.000000	20640.000000
mean	3.870671	28.639486	5.429000	1.096675	1425.476744	3.070655	35.631861	-119.569704	2.068558
std	1.899822	12.585558	2.474173	0.473911	1132.462122	10.386050	2.135952	2.003532	1.153956
min	0.499900	1.000000	0.846154	0.333333	3.000000	0.692308	32.540000	-124.350000	0.149990
25%	2.563400	18.000000	4.440716	1.006079	787.000000	2.429741	33.930000	-121.800000	1.196000
50%	3.534800	29.000000	5.229129	1.048780	1166.000000	2.818116	34.260000	-118.490000	1.797000
75%	4.743250	37.000000	6.052381	1.099526	1725.000000	3.282261	37.710000	-118.010000	2.647250
max	15.000100	52.000000	141.909091	34.066667	35682.000000	1243.333333	41.950000	-114.310000	5.000010

```
df[df.duplicated()]       #Check for and remove any duplicated rows
df.drop_duplicates()
```

	MedInc	HouseAge	AveRooms	AveBedrms	Population	AveOccup	Latitude	Longitude	MedHouseVal
0	8.3252	41.0	6.984127	1.023810	322.0	2.555556	37.88	-122.23	4.526
1	8.3014	21.0	6.238137	0.971880	2401.0	2.109842	37.86	-122.22	3.585
2	7.2574	52.0	8.288136	1.073446	496.0	2.802260	37.85	-122.24	3.521
3	5.6431	52.0	5.817352	1.073059	558.0	2.547945	37.85	-122.25	3.413
4	3.8462	52.0	6.281853	1.081081	565.0	2.181467	37.85	-122.25	3.422
...	...	...	...	...	...	...	...	...	...
20635	1.5603	25.0	5.045455	1.133333	845.0	2.560606	39.48	-121.09	0.781
20636	2.5568	18.0	6.114035	1.315789	356.0	3.122807	39.49	-121.21	0.771
20637	1.7000	17.0	5.205543	1.120092	1007.0	2.325635	39.43	-121.22	0.923
20638	1.8672	18.0	5.329513	1.171920	741.0	2.123209	39.43	-121.32	0.847
20639	2.3886	16.0	5.254717	1.162264	1387.0	2.616981	39.37	-121.24	0.894

20640 rows × 9 columns

```
df = df.fillna(df.mean())       #Fill missing numerical features with the mean value
```

```
df['PricePerRoom'] = df['MedHouseVal'] / df['AveRooms']       #Create a new column PricePerRoom = median_house_value / total_rooms
```

```
print(df.columns)
```

```
Index(['MedInc', 'HouseAge', 'AveRooms', 'AveBedrms', 'Population', 'AveOccup',
       'Latitude', 'Longitude', 'MedHouseVal', 'PricePerRoom'],
      dtype='object')
```

```
df['HighPopulationArea'] = df['Population'].apply(lambda x: 1 if x > 500 else 0)          #Create a column HighPopulationArea:1 if populatio
```

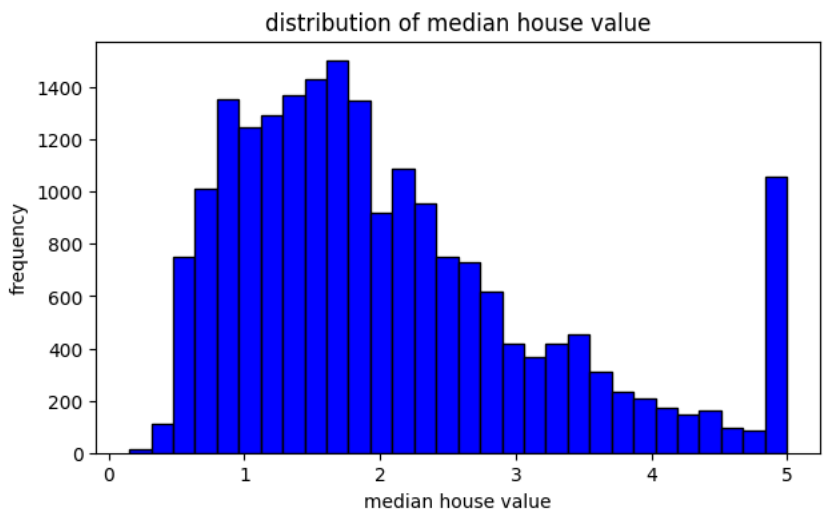
```
print(df.columns)
```

```
Index(['MedInc', 'HouseAge', 'AveRooms', 'AveBedrms', 'Population', 'AveOccup',
       'Latitude', 'Longitude', 'MedHouseVal', 'PricePerRoom',
       'HighPopulationArea'],
      dtype='object')
```

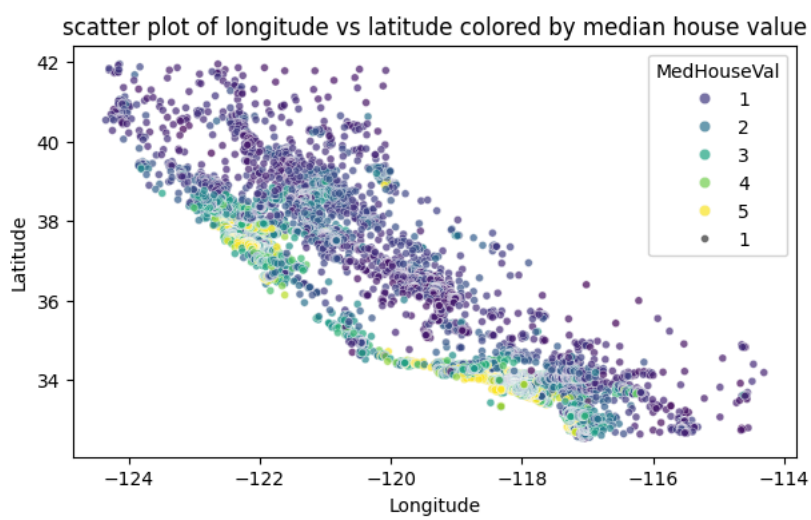
```
df['IncomeCategory'] = pd.cut(                                  # Bin the median_income into 5 equal-sized bins and label them as Very Lo

    df['MedInc'],
    bins=5,
    labels=['Very Low', 'Low', 'Medium', 'High', 'Very High']
)
```

```
df = df.drop(columns=['MedInc', 'AveRooms'])            #Drop columns that seem redundant after feature creation (if any)
print(df.columns)
```

```
import matplotlib.pyplot as plt   #Plot the distribution of median_house_value with a histogram
plt.figure(figsize=(7, 4))
plt.hist(df['MedHouseVal'], bins=30, color='blue', edgecolor='black')
plt.title('distribution of median house value', fontsize=12)
plt.xlabel('median house value', fontsize=10)
plt.ylabel('frequency', fontsize=10)
plt.show()
```



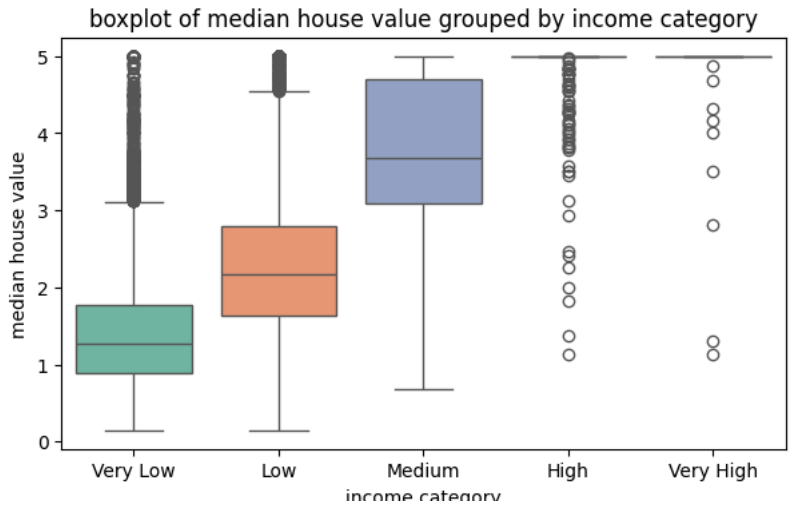

```
import seaborn as sns                        #Create a scatter plot of longitude vs latitude, colored by median_house_value
plt.figure(figsize=(7, 4))
sns.scatterplot(x='Longitude', y='Latitude', hue='MedHouseVal', data=df, palette='viridis', size=1, alpha=0.7)
plt.title('scatter plot of longitude vs latitude colored by median house value', fontsize=12)
plt.xlabel('Longitude', fontsize=10)
plt.ylabel('Latitude', fontsize=10)
plt.show()
```



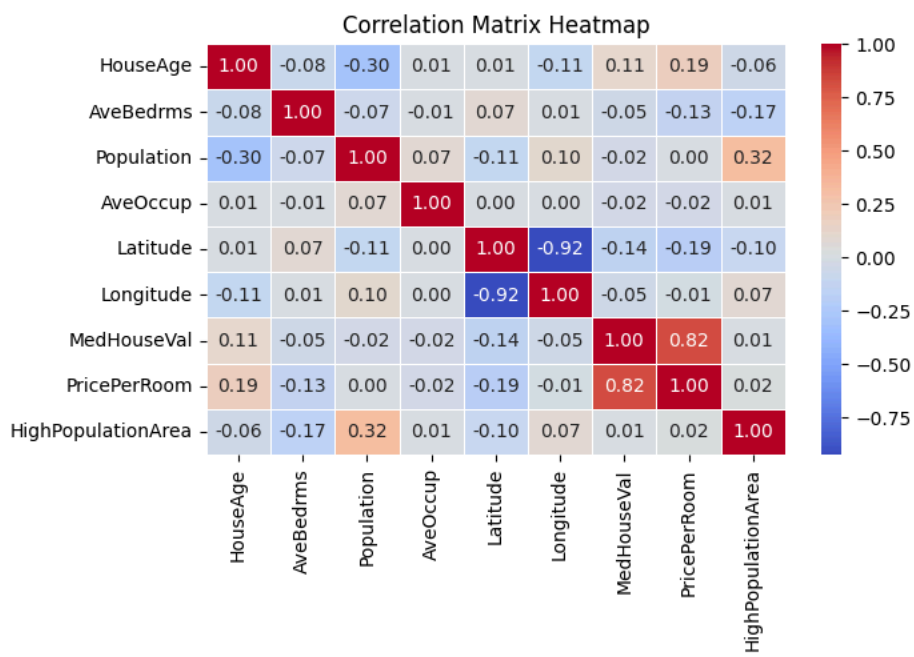
Generate | 10 random numbers using numpy | Close

```
plt.figure(figsize=(7, 4))                                #Plot a boxplot of median_house_value grouped by the new income categories
sns.boxplot(x='IncomeCategory', y='MedHouseVal', data=df, palette='Set2')
plt.title('boxplot of median house value grouped by income category', fontsize=12)
plt.xlabel('income category', fontsize=10)
plt.ylabel('median house value', fontsize=10)
plt.show()
```

```
<ipython-input-47-86420c842226>:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend

  sns.boxplot(x='IncomeCategory', y='MedHouseVal', data=df, palette='Set2')
```

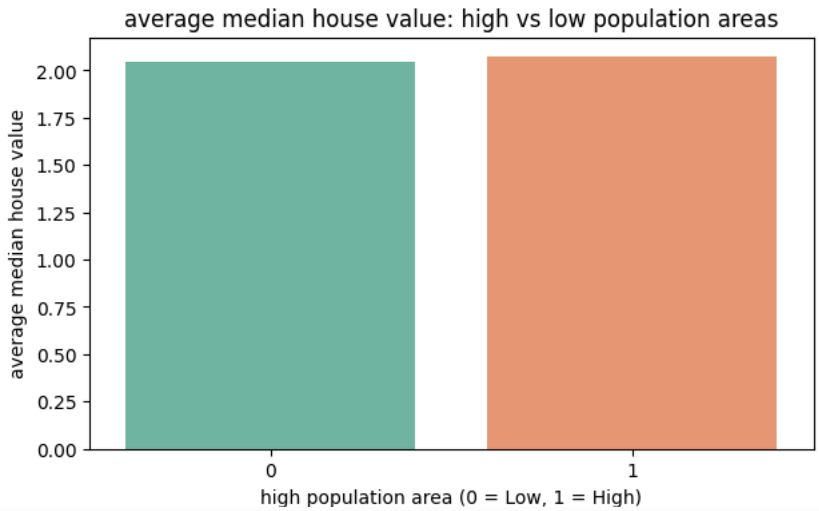



```
numeric_df = df.select_dtypes(include=['float64', 'int64'])          #Plot the correlation matrix heatmap between numerical features
corr_matrix = numeric_df.corr()
plt.figure(figsize=(7, 4))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', fmt=".2f", linewidths=0.5)
plt.title('correlation matrix heatmap', fontsize=12)
plt.show()
```




```
avg_price_by_population = df.groupby('HighPopulationArea')['MedHouseVal'].mean().reset_index()
plt.figure(figsize=(7, 4))
sns.barplot(x='HighPopulationArea', y='MedHouseVal', data=avg_price_by_population, palette='Set2')
plt.title('average median house value: high vs low population areas', fontsize=12)
plt.xlabel('high population area (0 = Low, 1 = High)', fontsize=10)
plt.ylabel('average median house value', fontsize=10)
plt.show()
```

```
<ipython-input-51-28cdb568b93a>:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend

  sns.barplot(x='HighPopulationArea', y='MedHouseVal', data=avg_price_by_population, palette='Set2')
```




```
from sklearn.datasets import fetch_california_housing
import pandas as pd
housing = fetch_california_housing(as_frame=True)
df = housing.frame
print(df.head())
selected_features = df[['MedInc', 'HouseAge', 'MedHouseVal']]
sns.pairplot(selected_features, diag_kind='kde', corner=True)
plt.show()
```

```
   MedInc  HouseAge  AveRooms  AveBedrms  Population  AveOccup  Latitude  \
0  8.3252      41.0  6.984127   1.023810       322.0  2.555556     37.88
1  8.3014      21.0  6.238137   0.971880      2401.0  2.109842     37.86
2  7.2574      52.0  8.288136   1.073446       496.0  2.802260     37.85
3  5.6431      52.0  5.817352   1.073059       558.0  2.547945     37.85
4  3.8462      52.0  6.281853   1.081081       565.0  2.181467     37.85

   Longitude  MedHouseVal
0    -122.23        4.526
1    -122.22        3.585
2    -122.24        3.521
3    -122.25        3.413
4    -122.25        3.422
```