```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

data = pd.read_csv("housing.csv")

data
```

```
       longitude  latitude  housing_median_age  total_rooms
total_bedrooms  \
0        -122.23     37.88                41.0        880.0
129.0
1        -122.22     37.86                21.0       7099.0
1106.0
2        -122.24     37.85                52.0       1467.0
190.0
3        -122.25     37.85                52.0       1274.0
235.0
4        -122.25     37.85                52.0       1627.0
280.0
...          ...       ...                 ...          ...
...
20635    -121.09     39.48                25.0       1665.0
374.0
20636    -121.21     39.49                18.0        697.0
150.0
20637    -121.22     39.43                17.0       2254.0
485.0
20638    -121.32     39.43                18.0       1860.0
409.0
20639    -121.24     39.37                16.0       2785.0
616.0

       population  households  median_income  median_house_value  \
0           322.0       126.0         8.3252            452600.0
1          2401.0      1138.0         8.3014            358500.0
2           496.0       177.0         7.2574            352100.0
3           558.0       219.0         5.6431            341300.0
4           565.0       259.0         3.8462            342200.0
...           ...         ...            ...                 ...
20635       845.0       330.0         1.5603             78100.0
20636       356.0       114.0         2.5568             77100.0
20637      1007.0       433.0         1.7000             92300.0
20638       741.0       349.0         1.8672             84700.0
20639      1387.0       530.0         2.3886             89400.0

      ocean_proximity
0            NEAR BAY
1            NEAR BAY
```

```
2            NEAR BAY
3            NEAR BAY
4            NEAR BAY
...               ...
20635          INLAND
20636          INLAND
20637          INLAND
20638          INLAND
20639          INLAND

[20640 rows x 10 columns]
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20640 entries, 0 to 20639
Data columns (total 10 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   longitude           20640 non-null  float64
 1   latitude            20640 non-null  float64
 2   housing_median_age  20640 non-null  float64
 3   total_rooms         20640 non-null  float64
 4   total_bedrooms      20433 non-null  float64
 5   population          20640 non-null  float64
 6   households          20640 non-null  float64
 7   median_income       20640 non-null  float64
 8   median_house_value  20640 non-null  float64
 9   ocean_proximity     20640 non-null  object 
dtypes: float64(9), object(1)
memory usage: 1.6+ MB
```

```
data.dropna(inplace=True)
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 20433 entries, 0 to 20639
Data columns (total 10 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   longitude           20433 non-null  float64
 1   latitude            20433 non-null  float64
 2   housing_median_age  20433 non-null  float64
 3   total_rooms         20433 non-null  float64
 4   total_bedrooms      20433 non-null  float64
 5   population          20433 non-null  float64
 6   households          20433 non-null  float64
 7   median_income       20433 non-null  float64
 8   median_house_value  20433 non-null  float64
```

```
 9   ocean_proximity     20433 non-null  object
dtypes: float64(9), object(1)
memory usage: 1.7+ MB
```

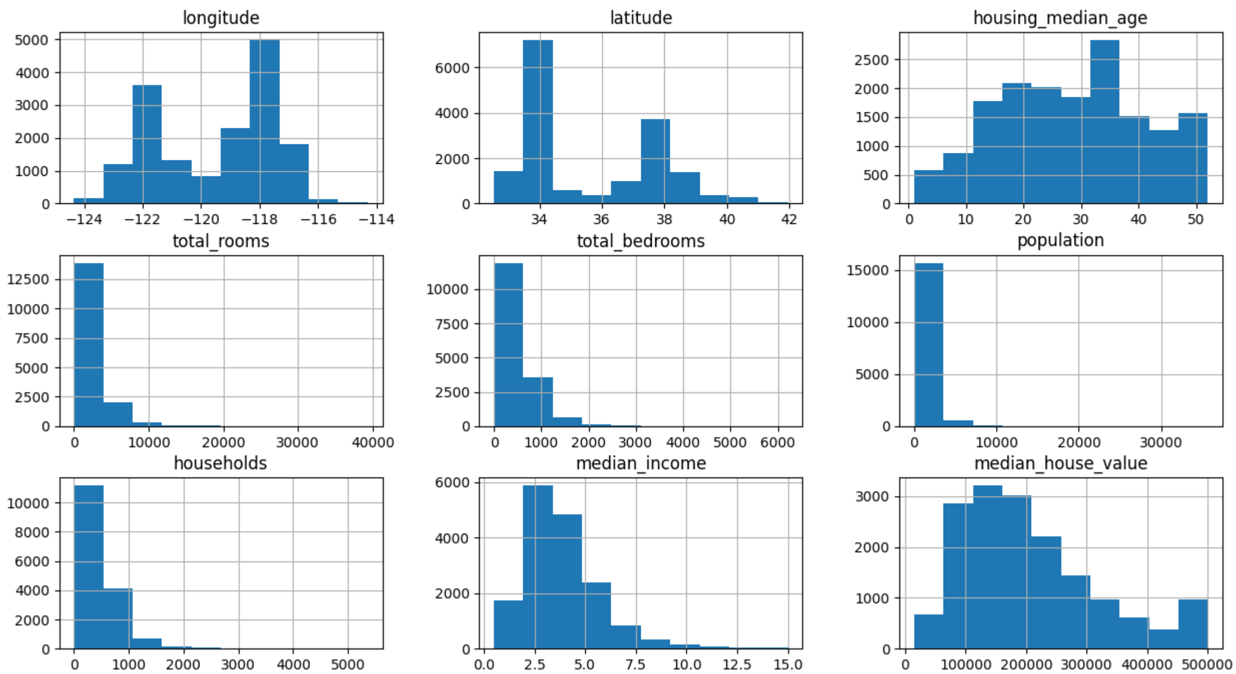
```
from sklearn.model_selection import train_test_split

x = data.drop(['median_house_value'], axis=1)
y = data['median_house_value']

x_train, x_test,y_train, y_test = train_test_split(x, y, 
test_size=0.2)

train_data = x_train.join(y_train)

train_data
```

```
       longitude  latitude  housing_median_age  total_rooms  
total_bedrooms  \
13042    -121.13     38.55                 8.0        530.0  
109.0   
3091     -118.45     35.62                18.0       2304.0  
527.0   
14486    -117.25     32.86                30.0       1670.0  
219.0   
16293    -121.23     37.96                37.0       2351.0  
564.0   
8620     -118.38     33.86                24.0       3124.0  
560.0   
...          ...       ...                 ...          ...  
...   
8002     -118.14     33.86                36.0       1774.0  
348.0   
9983     -122.52     38.67                35.0       1705.0  
321.0   
12766    -121.42     38.62                33.0       3171.0  
832.0   
9334     -122.68     38.01                41.0       1865.0  
392.0   
14731    -117.02     32.81                27.0       1950.0  
317.0   

       population  households  median_income ocean_proximity  \
13042       398.0        96.0         4.2031          INLAND   
3091        782.0       390.0         1.4141          INLAND   
14486       606.0       202.0        12.4429      NEAR OCEAN   
16293      1591.0       549.0         1.6563          INLAND   
8620       1312.0       542.0         6.3021       <1H OCEAN   
...           ...         ...            ...             ...   
8002        934.0       333.0         4.8571       <1H OCEAN   
9983        708.0       253.0         3.4539          INLAND   
12766      1591.0       695.0         2.0786          INLAND   
```

```
9334         825.0       369.0         4.2011     NEAR OCEAN
14731        950.0       320.0         4.0656      <1H OCEAN

       median_house_value
13042            212500.0
3091              75800.0
14486            500001.0
16293             57200.0
8620             333800.0
...                   ...
8002             203300.0
9983             300000.0
12766             88600.0
9334             255400.0
14731            164000.0

[16346 rows x 10 columns]
```

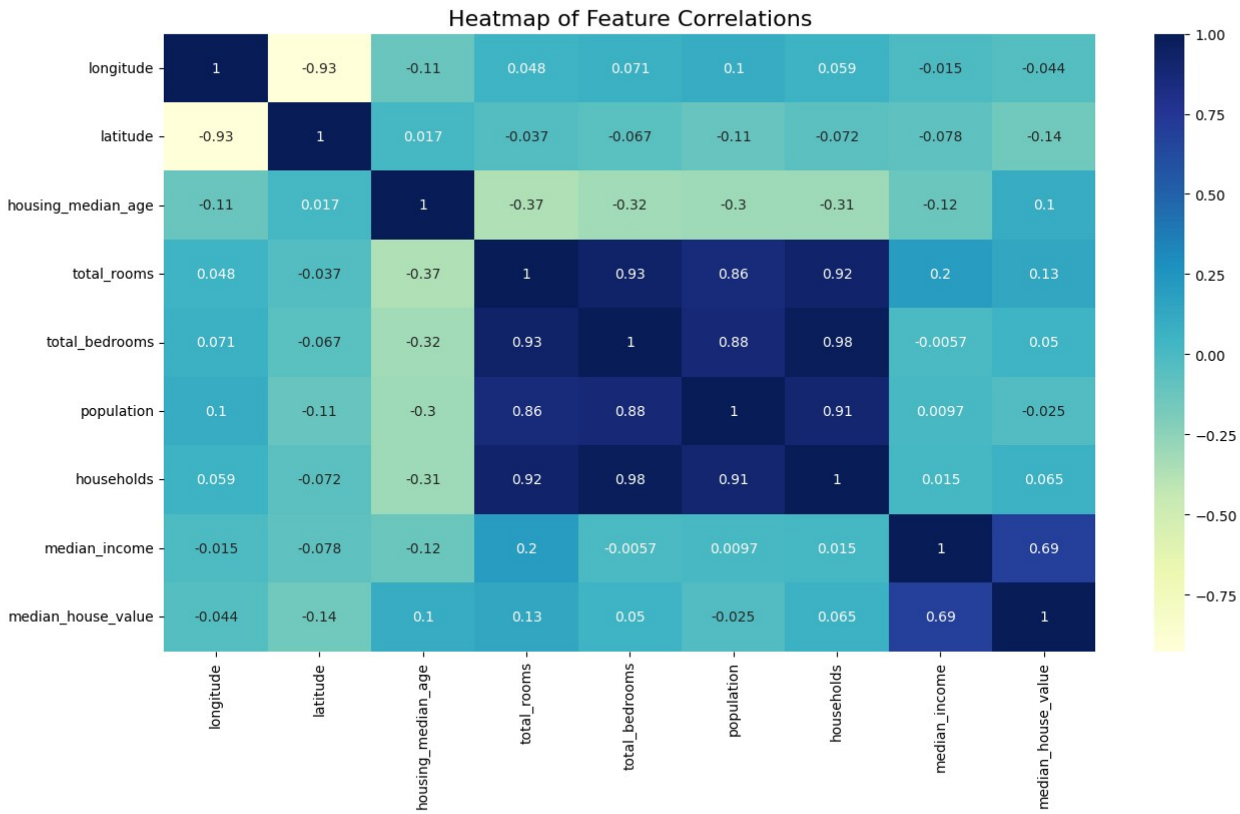
```
train_data.hist(figsize=(15,8))
```

```
array([[<Axes: title={'center': 'longitude'}>,
        <Axes: title={'center': 'latitude'}>,
        <Axes: title={'center': 'housing_median_age'}>],
       [<Axes: title={'center': 'total_rooms'}>,
        <Axes: title={'center': 'total_bedrooms'}>,
        <Axes: title={'center': 'population'}>],
       [<Axes: title={'center': 'households'}>,
        <Axes: title={'center': 'median_income'}>,
        <Axes: title={'center': 'median_house_value'}>]],
dtype=object)
```



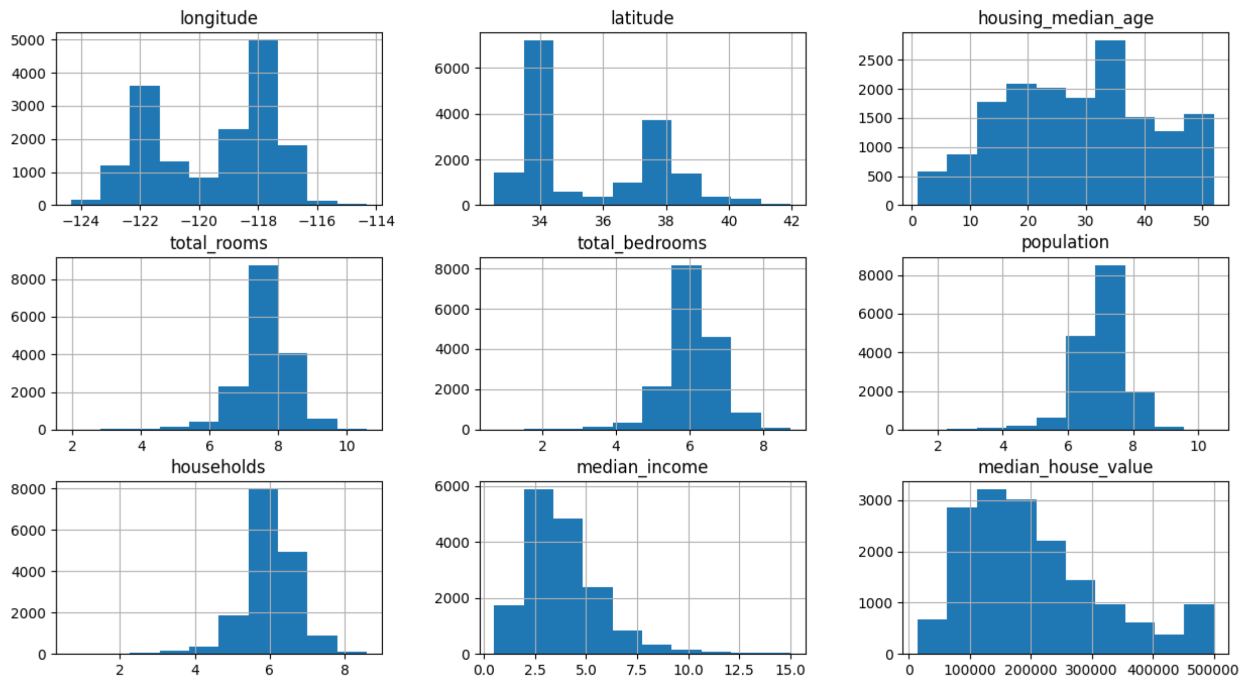

```
train_data.corr(numeric_only=True)

                    longitude  latitude  housing_median_age
total_rooms  \
longitude            1.000000 -0.925551           -0.114506
0.047664
latitude            -0.925551  1.000000            0.017210    -
0.036572
housing_median_age  -0.114506  0.017210            1.000000    -
0.365221
total_rooms          0.047664 -0.036572           -0.365221
1.000000
total_bedrooms       0.071255 -0.066807           -0.323527
0.929137
population           0.104915 -0.110697           -0.301750
0.857913
households           0.058500 -0.071591           -0.306438
0.918270
median_income       -0.014569 -0.078059           -0.123771
0.202048
median_house_value  -0.043753 -0.143297            0.099769
0.134498

                    total_bedrooms  population  households
median_income  \
longitude                 0.071255    0.104915    0.058500      -
0.014569
latitude                 -0.066807   -0.110697   -0.071591      -
0.078059
```

```
housing_median_age        -0.323527   -0.301750   -0.306438        -
0.123771
total_rooms                0.929137    0.857913    0.918270
0.202048
total_bedrooms             1.000000    0.877862    0.980288        -
0.005714
population                 0.877862    1.000000    0.906536
0.009677
households                 0.980288    0.906536    1.000000
0.015084
median_income             -0.005714    0.009677    0.015084
1.000000
median_house_value         0.050333   -0.025071    0.064554
0.688152

                    median_house_value
longitude                    -0.043753
latitude                     -0.143297
housing_median_age            0.099769
total_rooms                   0.134498
total_bedrooms                0.050333
population                   -0.025071
households                    0.064554
median_income                 0.688152
median_house_value            1.000000

plt.figure(figsize=(15,8))
sns.heatmap(train_data.corr(numeric_only=True), annot=True,
cmap="YlGnBu")
plt.title("Heatmap of Feature Correlations", fontsize=16)
plt.show()
```




```
train_data['total_rooms'] = np.log(train_data['total_rooms'] + 1)
train_data['total_bedrooms'] = np.log(train_data['total_bedrooms'] +
1)
train_data['population'] = np.log(train_data['population'] + 1)
train_data['households'] = np.log(train_data['households'] + 1)

train_data.hist(figsize=(15,8))
```

```
array([[<Axes: title={'center': 'longitude'}>,
        <Axes: title={'center': 'latitude'}>,
        <Axes: title={'center': 'housing_median_age'}>],
       [<Axes: title={'center': 'total_rooms'}>,
        <Axes: title={'center': 'total_bedrooms'}>,
        <Axes: title={'center': 'population'}>],
       [<Axes: title={'center': 'households'}>,
        <Axes: title={'center': 'median_income'}>,
        <Axes: title={'center': 'median_house_value'}>]],
dtype=object)
```

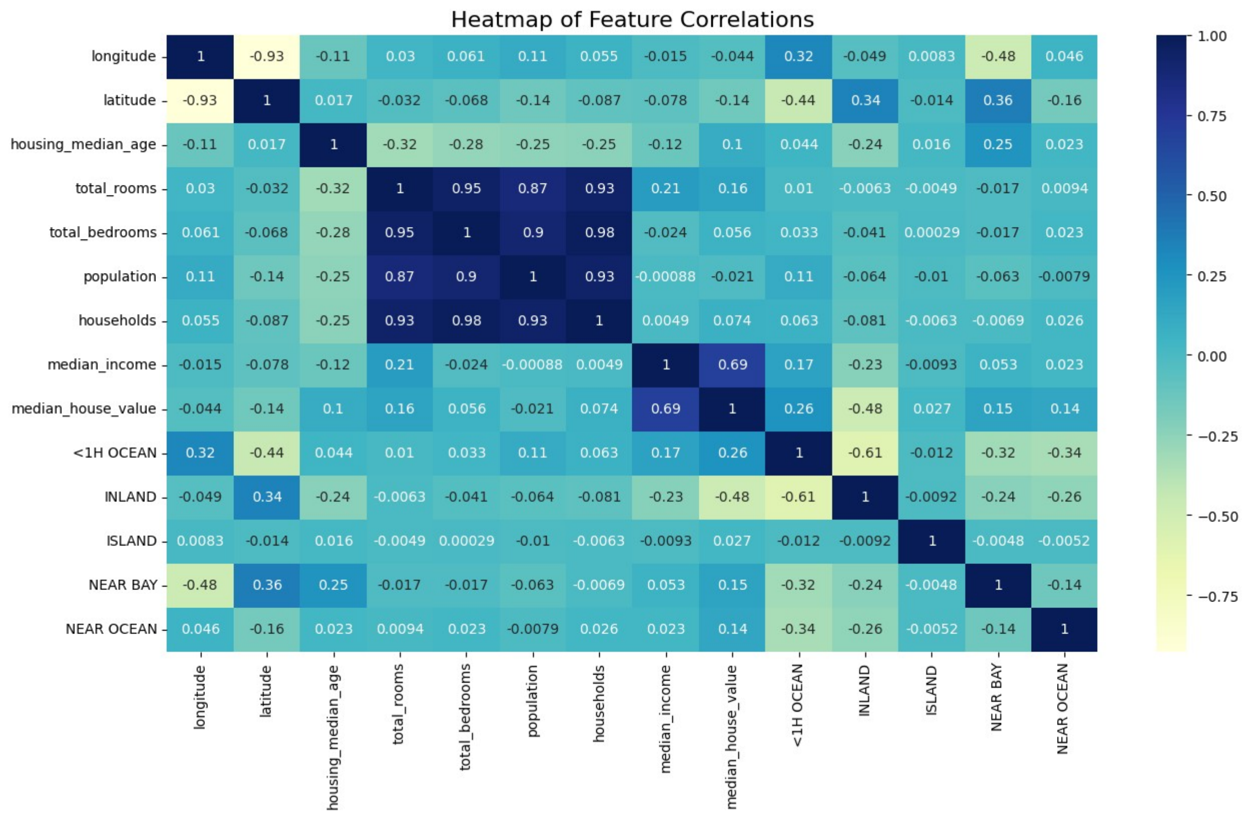



```
train_data.ocean_proximity.value_counts()

ocean_proximity
<1H OCEAN     7284
INLAND        5137
NEAR OCEAN    2094
NEAR BAY      1828
ISLAND           3
Name: count, dtype: int64

train_data =
train_data.join(pd.get_dummies(train_data.ocean_proximity)).drop(['oce
an_proximity'],axis=1)

train_data

       longitude  latitude  housing_median_age  total_rooms
total_bedrooms  \
13042    -121.13     38.55                 8.0     6.274762
4.700480
3091     -118.45     35.62                18.0     7.742836
6.269096
14486    -117.25     32.86                30.0     7.421178
5.393628
16293    -121.23     37.96                37.0     7.763021
6.336826
8620     -118.38     33.86                24.0     8.047190
6.329721
...          ...       ...                 ...          ...
```

```
...
8002     -118.14     33.86                36.0     7.481556
5.855072
9983     -122.52     38.67                35.0     7.441907
5.774552
12766    -121.42     38.62                33.0     8.062118
6.725034
9334     -122.68     38.01                41.0     7.531552
5.973810
14731    -117.02     32.81                27.0     7.576097
5.762051

       population  households  median_income  median_house_value  <1H
OCEAN  \
13042    5.988961    4.574711         4.2031            212500.0
False
3091     6.663133    5.968708         1.4141             75800.0
False
14486    6.408529    5.313206        12.4429            500001.0
False
16293    7.372746    6.309918         1.6563             57200.0
False
8620     7.180070    6.297109         6.3021            333800.0
True
...           ...         ...            ...                 ...
...
8002     6.840547    5.811141         4.8571            203300.0
True
9983     6.563856    5.537334         3.4539            300000.0
False
12766    7.372746    6.545350         2.0786             88600.0
False
9334     6.716595    5.913503         4.2011            255400.0
False
14731    6.857514    5.771441         4.0656            164000.0
True

       INLAND  ISLAND  NEAR BAY  NEAR OCEAN
13042    True   False     False       False
3091     True   False     False       False
14486   False   False     False        True
16293    True   False     False       False
8620    False   False     False       False
...       ...     ...       ...         ...
8002    False   False     False       False
9983     True   False     False       False
12766    True   False     False       False
9334    False   False     False        True
14731   False   False     False       False
```

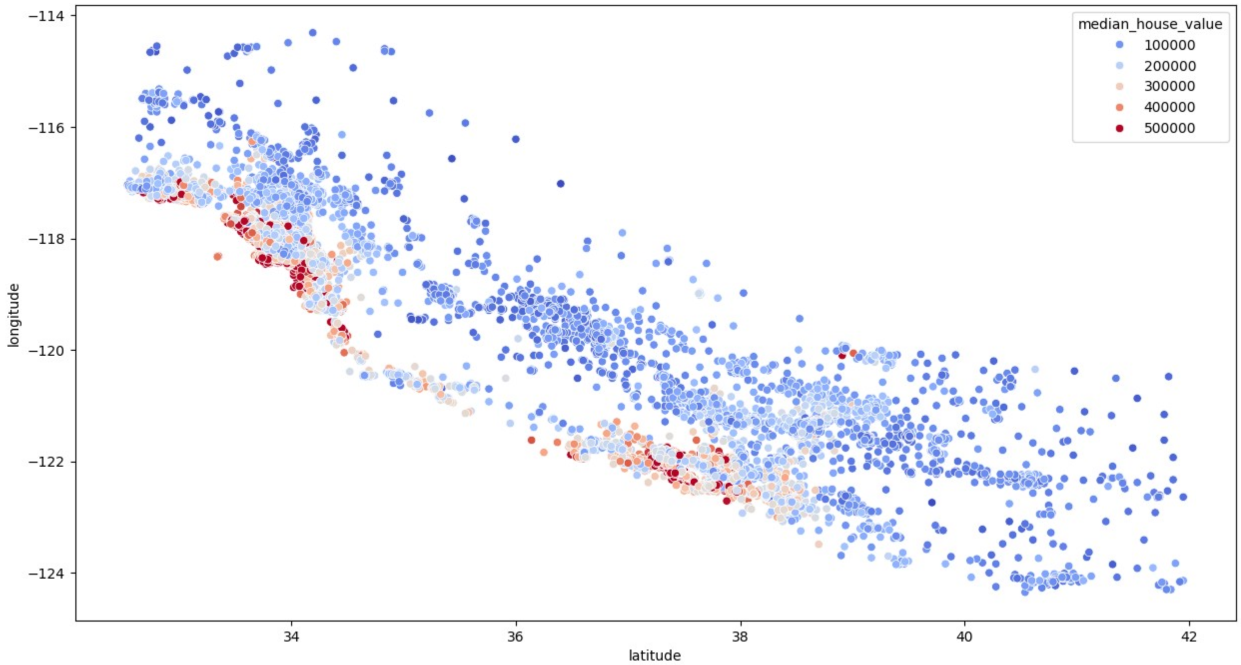
```
[16346 rows x 14 columns]
```

```python
plt.figure(figsize=(15,8))
sns.heatmap(train_data.corr(numeric_only=True), annot=True,
cmap="YlGnBu")
plt.title("Heatmap of Feature Correlations", fontsize=16)
plt.show()
```

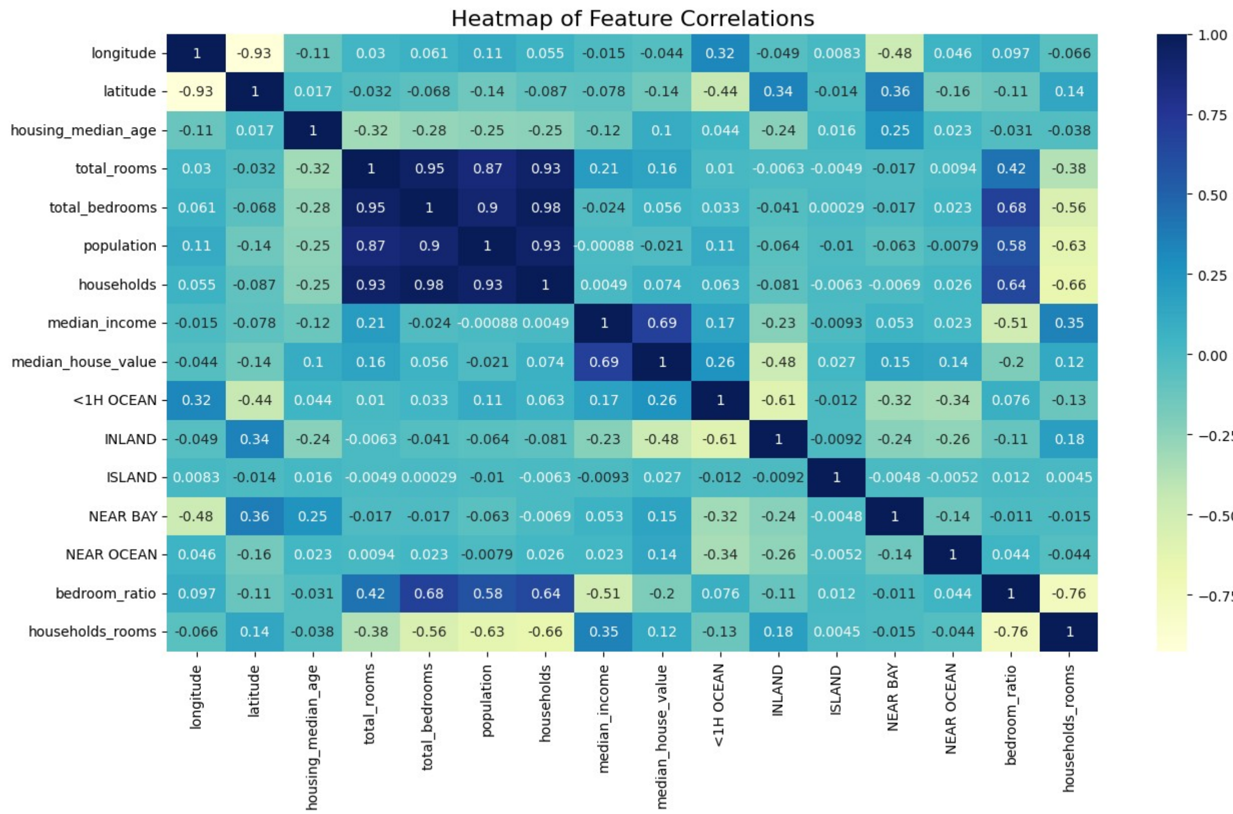



```python
plt.figure(figsize=(15,8))
sns.scatterplot(x="latitude", y="longitude", data=train_data,
hue="median_house_value", palette="coolwarm")
```

```
<Axes: xlabel='latitude', ylabel='longitude'>
```




```
train_data['bedroom_ratio'] =
train_data['total_bedrooms']/train_data['total_rooms']
train_data['households_rooms'] =
train_data['total_rooms']/train_data['households']

plt.figure(figsize=(15,8))
sns.heatmap(train_data.corr(numeric_only=True), annot=True,
cmap="YlGnBu")
plt.title("Heatmap of Feature Correlations", fontsize=16)
plt.show()
```




```python
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline

# Separate features and target
x_train = train_data.drop(['median_house_value'], axis=1)
y_train = train_data['median_house_value']

# Identify categorical and numeric columns
cat_cols = x_train.select_dtypes(include=['object']).columns
num_cols = x_train.select_dtypes(exclude=['object']).columns


numeric_transformer = StandardScaler()
categorical_transformer = OneHotEncoder(handle_unknown='ignore')

# Combine transformations
preprocessor = ColumnTransformer(
    transformers=[
        ('num', numeric_transformer, num_cols),
        ('cat', categorical_transformer, cat_cols)
    ]
)
```

```
# Create a pipeline with preprocessing + regression
model = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('regressor', LinearRegression())
])


model.fit(x_train, y_train)
```

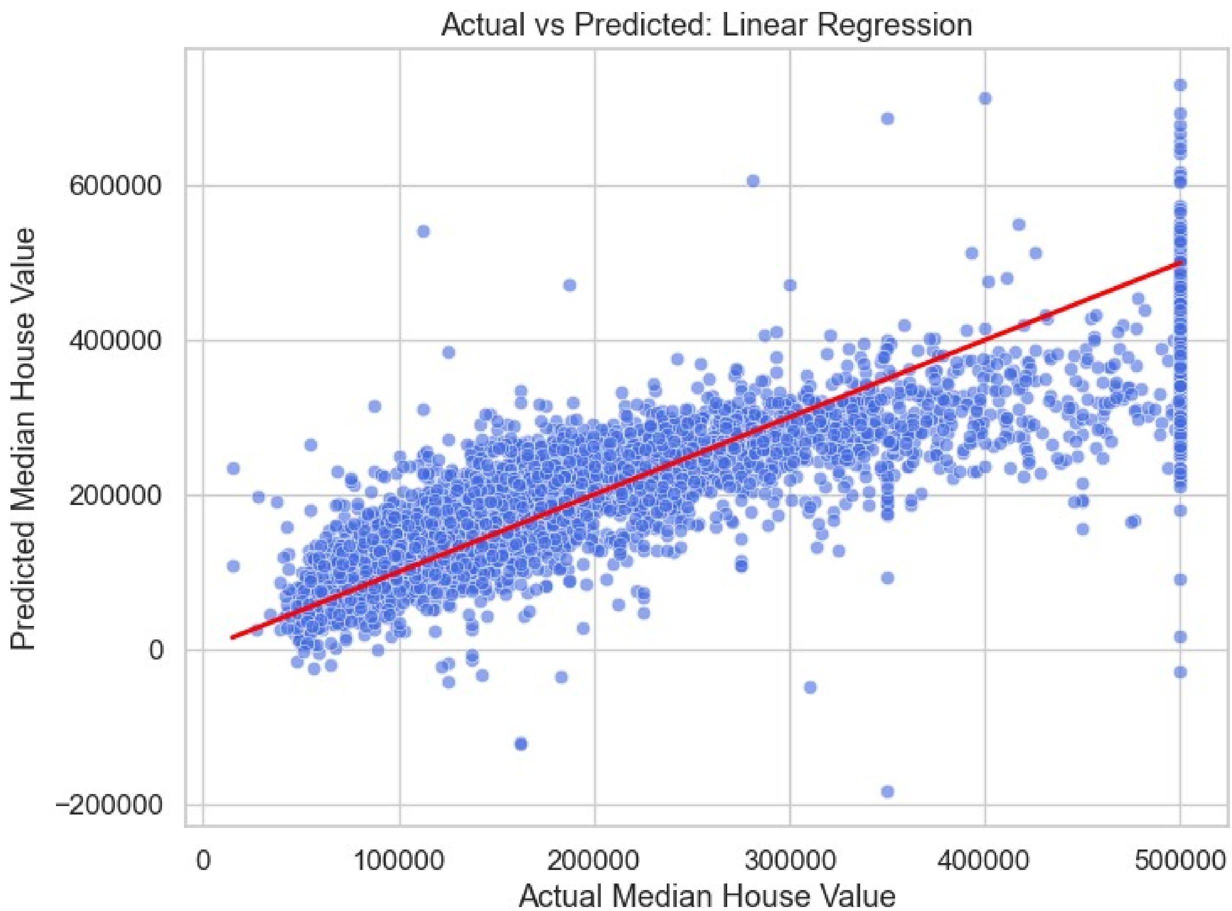
```
Pipeline(steps=[('preprocessor',
                 ColumnTransformer(transformers=[('num',
StandardScaler(),
                                                  Index(['longitude',
'latitude', 'housing_median_age', 'total_rooms',
       'total_bedrooms', 'population', 'households', 'median_income',
       '<1H OCEAN', 'INLAND', 'ISLAND', 'NEAR BAY', 'NEAR OCEAN',
       'bedroom_ratio', 'households_rooms'],
      dtype='object')),
                                                 ('cat',

OneHotEncoder(handle_unknown='ignore'),
                                                  Index([],
dtype='object'))])),
                ('regressor', LinearRegression())])
```

```
import numpy as np
import pandas as pd


test_data = x_test.copy()


for col in ['total_rooms', 'total_bedrooms', 'population',
'households']:
    test_data[col] = np.log(test_data[col] + 1)

#  Add engineered features
test_data['bedroom_ratio'] = test_data['total_bedrooms'] /
test_data['total_rooms']
test_data['households_rooms'] = test_data['total_rooms'] /
test_data['households']

test_data = test_data.join(pd.get_dummies(test_data.ocean_proximity))


test_data = test_data.drop(['ocean_proximity'], axis=1)

expected_cols = ['ISLAND', '<1H OCEAN', 'NEAR OCEAN', 'INLAND', 'NEAR
BAY']

for col in expected_cols:
```

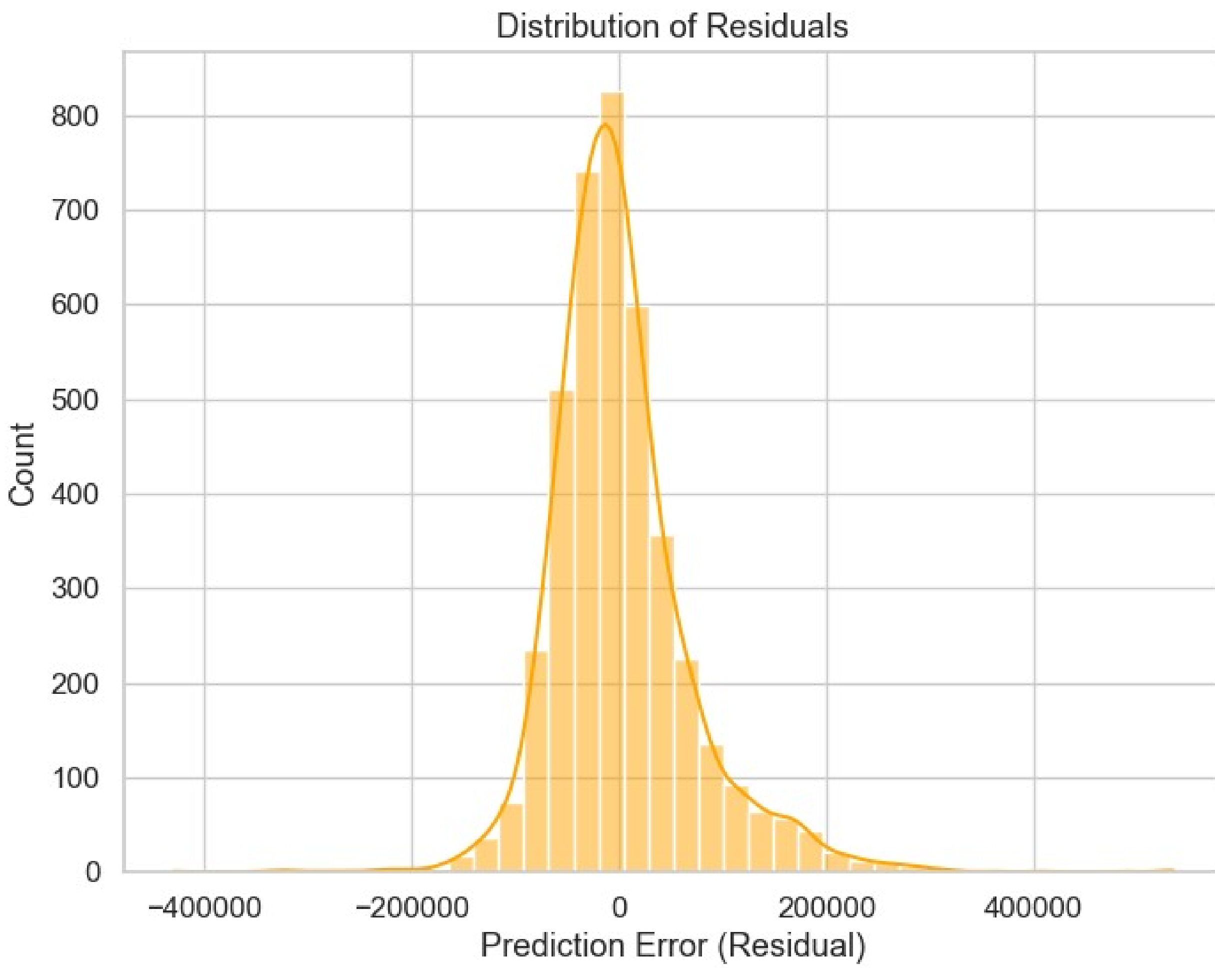
```python
    if col not in test_data.columns:
        test_data[col] = 0  # add missing columns with 0s


test_data = test_data.reindex(columns=model.feature_names_in_,
fill_value=0)


y_pred = model.predict(test_data)

print("Predictions generated successfully!")
print(y_pred[:5])
```

```
Predictions generated successfully!
[192363.1902475  144359.6370845  267832.54592692 327284.99875232
 224316.36393459]
```

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


sns.set(style="whitegrid", font_scale=1.1)

# Scatter plot — Actual vs Predicted
plt.figure(figsize=(8, 6))
sns.scatterplot(x=y_test, y=y_pred, alpha=0.6, color="royalblue")
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()],
color='red', lw=2)  # perfect prediction line
plt.xlabel("Actual Median House Value")
plt.ylabel("Predicted Median House Value")
plt.title("Actual vs Predicted: Linear Regression")
plt.show()
```




```
2 2 Residuals plot – shows errors
residuals = y_test - y_pred
plt.figure(figsize=(8, 6))
sns.histplot(residuals, bins=40, kde=True, color='orange')
plt.xlabel("Prediction Error (Residual)")
plt.title("Distribution of Residuals")
plt.show()
```




```
import joblib

joblib.dump(model, "linear_regression_model.pkl")
print("Model saved successfully as 'linear_regression_model.pkl'")


loaded_model = joblib.load("linear_regression_model.pkl")
print("Model loaded successfully!")

# Test
sample_pred = loaded_model.predict(test_data)
print("Sample predictions:", sample_pred[:5])

Model saved successfully as 'linear_regression_model.pkl'
Model loaded successfully!
Sample predictions: [191196.04061322 147073.28426518 270872.48908267
327020.34119031
 223866.7442592 ]


import joblib
```

```python
# Load trained model
model = joblib.load("linear_regression_model.pkl")

#  new data
new_data = pd.DataFrame({
    'longitude': [-122.23],
    'latitude': [37.88],
    'housing_median_age': [41.0],
    'total_rooms': [880.0],
    'total_bedrooms': [129.0],
    'population': [322.0],
    'households': [126.0],
    'median_income': [8.3252],
    'ocean_proximity': ['NEAR BAY']
})

for col in ['total_rooms', 'total_bedrooms', 'population',
'households']:
    new_data[col] = np.log(new_data[col] + 1)


new_data['bedroom_ratio'] = new_data['total_bedrooms'] /
new_data['total_rooms']
new_data['households_rooms'] = new_data['total_rooms'] /
new_data['households']


new_data =
new_data.join(pd.get_dummies(new_data.ocean_proximity)).drop(['ocean_p
roximity'], axis=1)


expected_cols = ['ISLAND', '<1H OCEAN', 'NEAR OCEAN', 'INLAND', 'NEAR
BAY']
for col in expected_cols:
    if col not in new_data.columns:
        new_data[col] = 0  # add missing dummies with 0s


new_data = new_data.reindex(columns=model.feature_names_in_,
fill_value=0)

predicted_value = model.predict(new_data)

print("Predicted Median House Value:", round(predicted_value[0], 2))
```

```
Predicted Median House Value: 411523.12
```