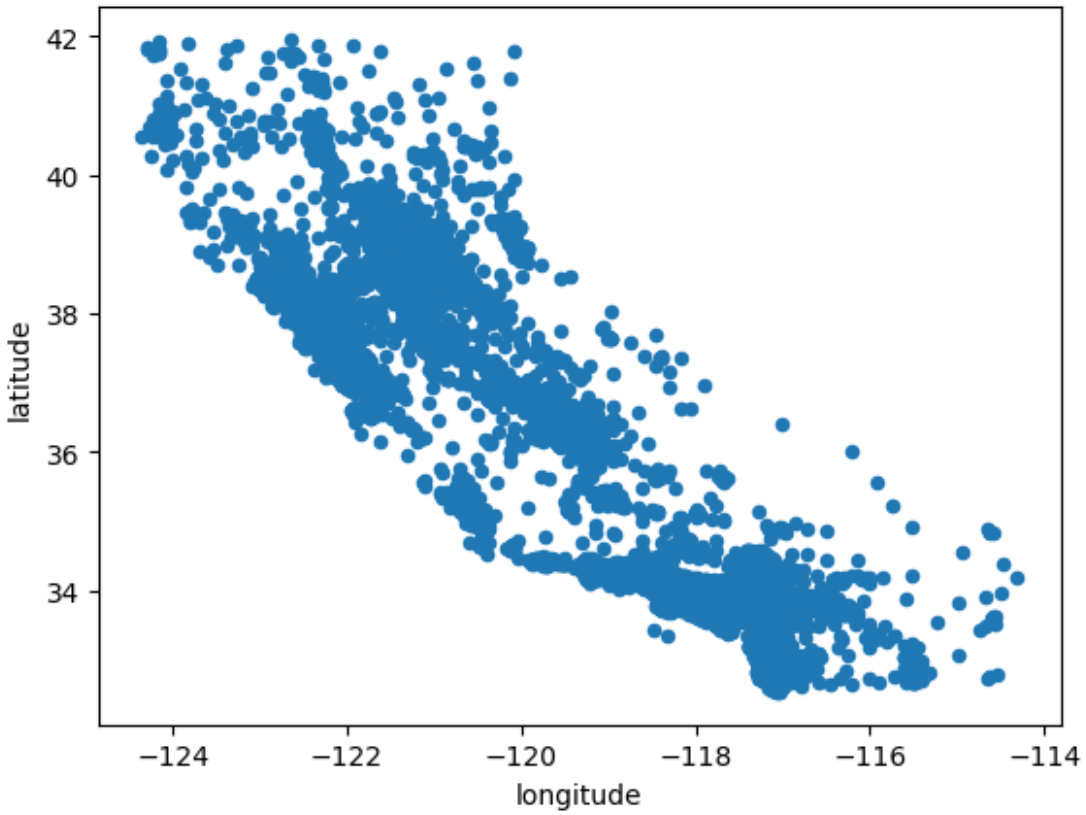
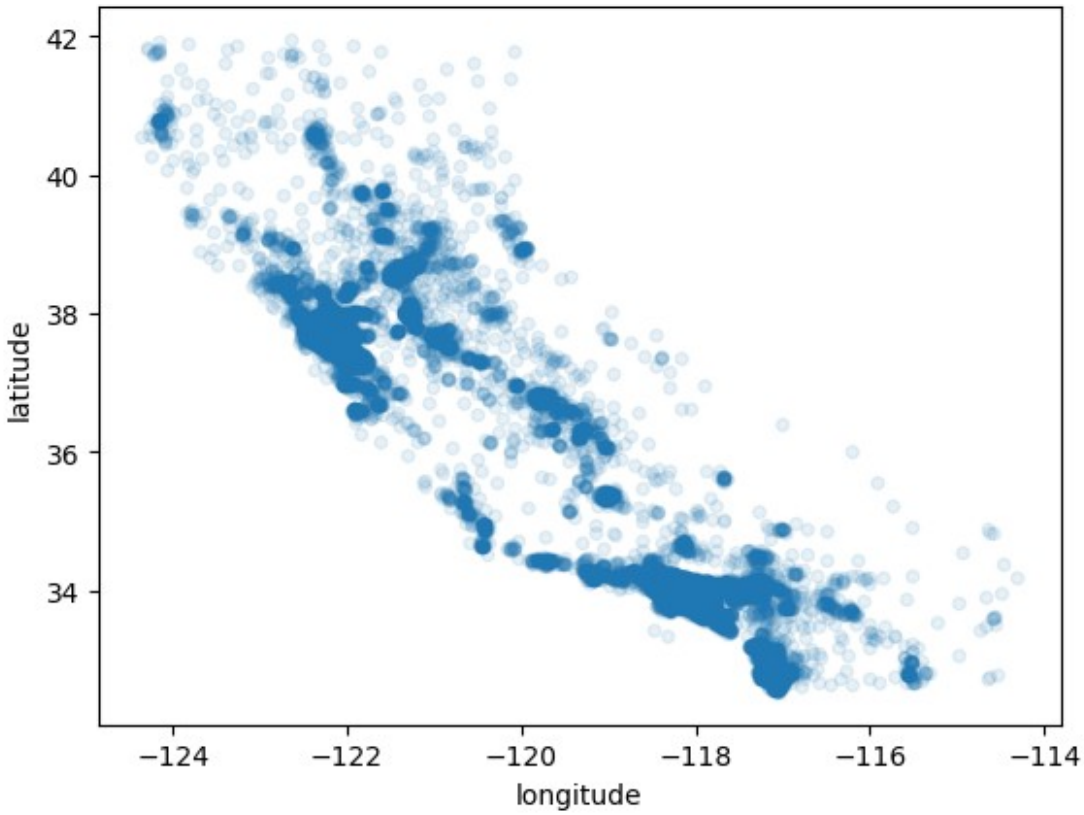
```
<Axes: xlabel='longitude', ylabel='latitude'>
```



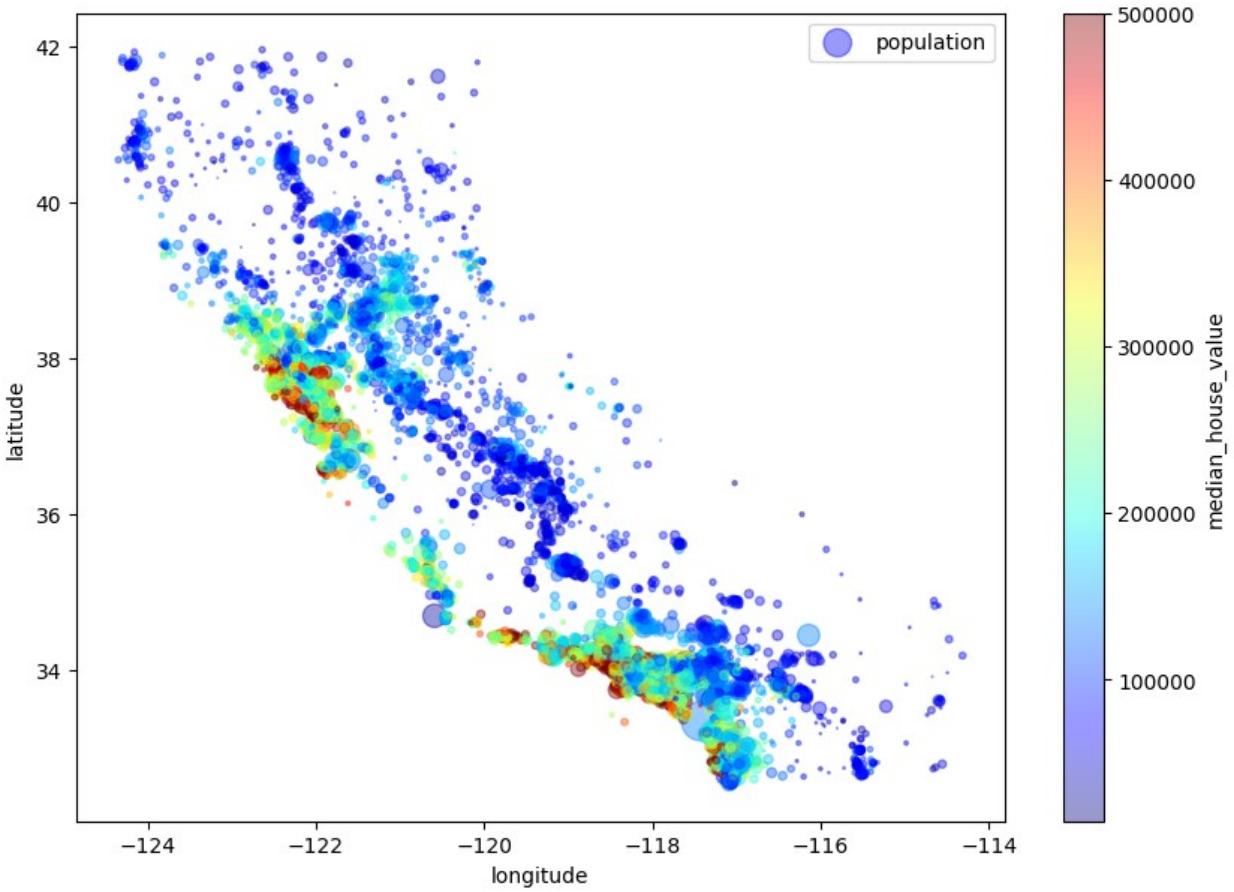

```
housing.plot(kind="scatter", x="longitude", y="latitude", alpha=0.1)
```

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




```
housing.plot(kind="scatter", x="longitude", y="latitude", alpha=0.4,
 s=housing["population"]/100, label="population", figsize=(10,7),
 c="median_house_value", cmap=plt.get_cmap("jet"), colorbar=True,
)
plt.legend()
```

```
<matplotlib.legend.Legend at 0x211680e77c0>
```



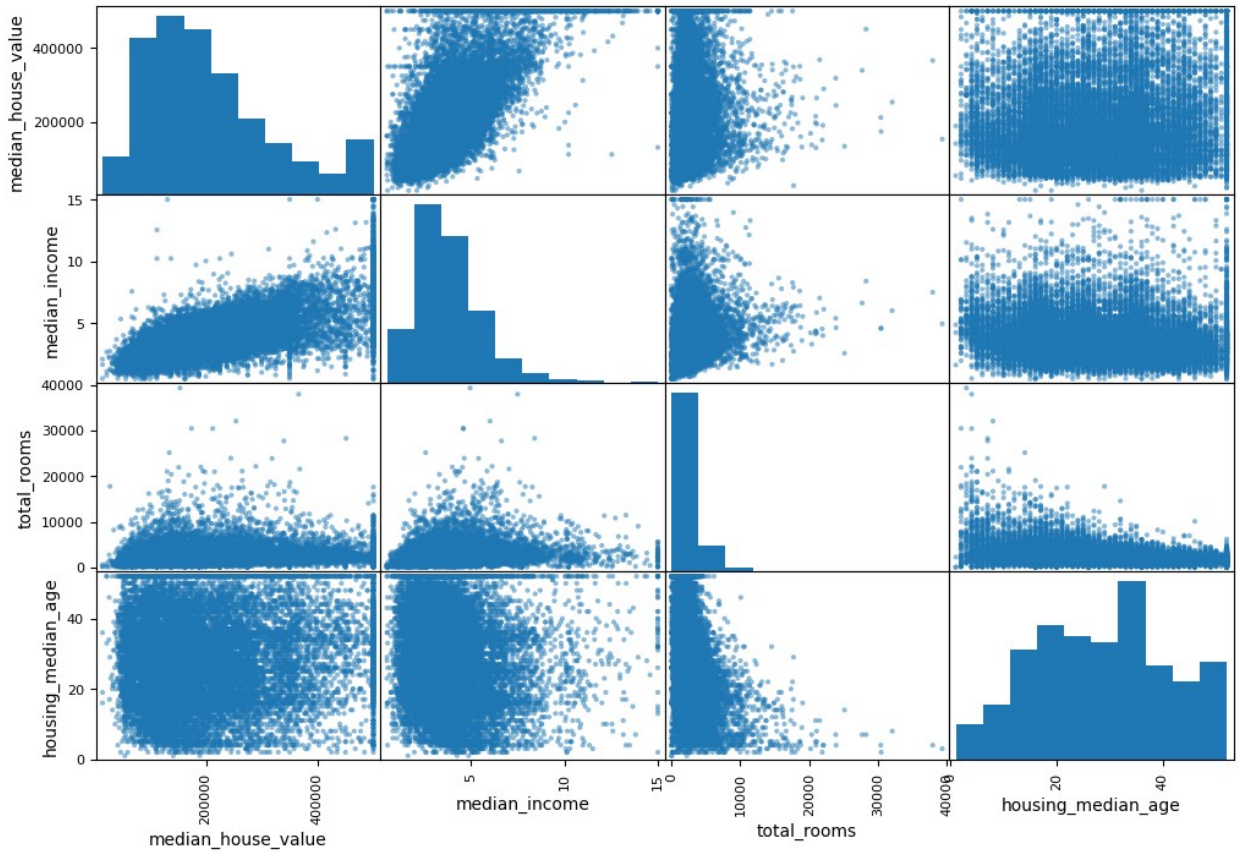

```
# Selecting only the numeric columns
numeric_columns = housing.select_dtypes(include=[np.number])

# Compute the correlation matrix
corr_matrix = numeric_columns.corr()
#instead of:
#corr_matrix = housing.corr()

corr_matrix["median_house_value"].sort_values(ascending=False)
```

```
median_house_value    1.000000
median_income         0.687151
total_rooms           0.135140
housing_median_age    0.114146
households            0.064590
total_bedrooms        0.047781
population           -0.026882
longitude            -0.047466
latitude             -0.142673
Name: median_house_value, dtype: float64
```

```
from pandas.plotting import scatter_matrix
attributes = ["median_house_value", "median_income", "total_rooms",
```

```
 "housing_median_age"]
scatter_matrix(housing[attributes], figsize=(12, 8))
```

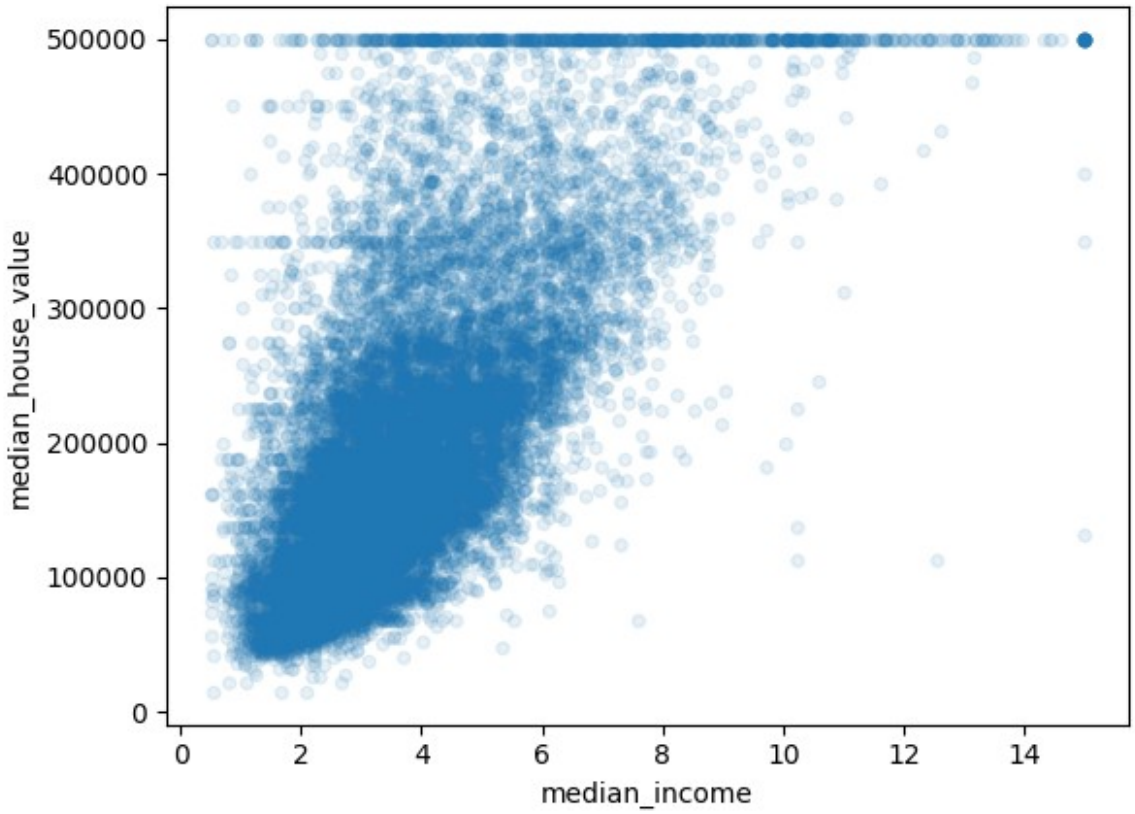
```
array([[<Axes: xlabel='median_house_value',
ylabel='median_house_value'>,
        <Axes: xlabel='median_income', ylabel='median_house_value'>,
        <Axes: xlabel='total_rooms', ylabel='median_house_value'>,
        <Axes: xlabel='housing_median_age',
ylabel='median_house_value'>],
       [<Axes: xlabel='median_house_value', ylabel='median_income'>,
        <Axes: xlabel='median_income', ylabel='median_income'>,
        <Axes: xlabel='total_rooms', ylabel='median_income'>,
        <Axes: xlabel='housing_median_age', ylabel='median_income'>],
       [<Axes: xlabel='median_house_value', ylabel='total_rooms'>,
        <Axes: xlabel='median_income', ylabel='total_rooms'>,
        <Axes: xlabel='total_rooms', ylabel='total_rooms'>,
        <Axes: xlabel='housing_median_age', ylabel='total_rooms'>],
       [<Axes: xlabel='median_house_value',
ylabel='housing_median_age'>,
        <Axes: xlabel='median_income', ylabel='housing_median_age'>,
        <Axes: xlabel='total_rooms', ylabel='housing_median_age'>,
        <Axes: xlabel='housing_median_age',
ylabel='housing_median_age'>]],
      dtype=object)
```



The most promising attribute to predict the median house value is the median income, so let's zoom in on their correlation scatterplot

```
housing.plot(kind="scatter", x="median_income",
y="median_house_value",
 alpha=0.1)
```

```
<Axes: xlabel='median_income', ylabel='median_house_value'>
```



Experimenting with Attribute Combinations

```
housing["rooms_per_household"] =
housing["total_rooms"]/housing["households"]
housing["bedrooms_per_room"] =
housing["total_bedrooms"]/housing["total_rooms"]
housing["population_per_household"]=housing["population"]/housing["hou
seholds"]

# Selecting only the numeric columns
numeric_columns = housing.select_dtypes(include=[np.number])

# Compute the correlation matrix
corr_matrix = numeric_columns.corr()
#instead of:
#corr_matrix = housing.corr()

corr_matrix["median_house_value"].sort_values(ascending=False)
```

```
median_house_value          1.000000
median_income               0.687151
rooms_per_household         0.146255
total_rooms                 0.135140
housing_median_age          0.114146
households                  0.064590
```

```
total_bedrooms               0.047781
population_per_household    -0.021991
population                  -0.026882
longitude                   -0.047466
latitude                    -0.142673
bedrooms_per_room           -0.259952
Name: median_house_value, dtype: float64
```