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

```
df=pd.read_csv('housing.csv')
```

```
df
```

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	median_house_value	o
0	-122.23	37.88	41.0	880.0	129.0	322.0	126.0	8.3252	452600.0	
1	-122.22	37.86	21.0	7099.0	1106.0	2401.0	1138.0	8.3014	358500.0	
2	-122.24	37.85	52.0	1467.0	190.0	496.0	177.0	7.2574	352100.0	
3	-122.25	37.85	52.0	1274.0	235.0	558.0	219.0	5.6431	341300.0	
4	-122.25	37.85	52.0	1627.0	280.0	565.0	259.0	3.8462	342200.0	
...	...	...	...	...	...	...	...	...	...	
20635	-121.09	39.48	25.0	1665.0	374.0	845.0	330.0	1.5603	78100.0	
20636	-121.21	39.49	18.0	697.0	150.0	356.0	114.0	2.5568	77100.0	
20637	-121.22	39.43	17.0	2254.0	485.0	1007.0	433.0	1.7000	92300.0	
20638	-121.32	39.43	18.0	1860.0	409.0	741.0	349.0	1.8672	84700.0	
20639	-121.24	39.37	16.0	2785.0	616.0	1387.0	530.0	2.3886	89400.0	

20640 rows × 10 columns

```
df.shape
```

```
(20640, 10)
```

```
df.isnull().sum()
```

```
longitude               0
latitude                0
housing_median_age      0
total_rooms             0
total_bedrooms        207
population              0
households              0
median_income           0
median_house_value      0
ocean_proximity         0
dtype: int64
```

```
df=df.dropna()
```

```
df.isnull().sum()
```

```
longitude             0
latitude              0
housing_median_age    0
total_rooms           0
total_bedrooms        0
population            0
households            0
median_income         0
median_house_value    0
ocean_proximity       0
dtype: int64
```

```
df.isna().sum()
```

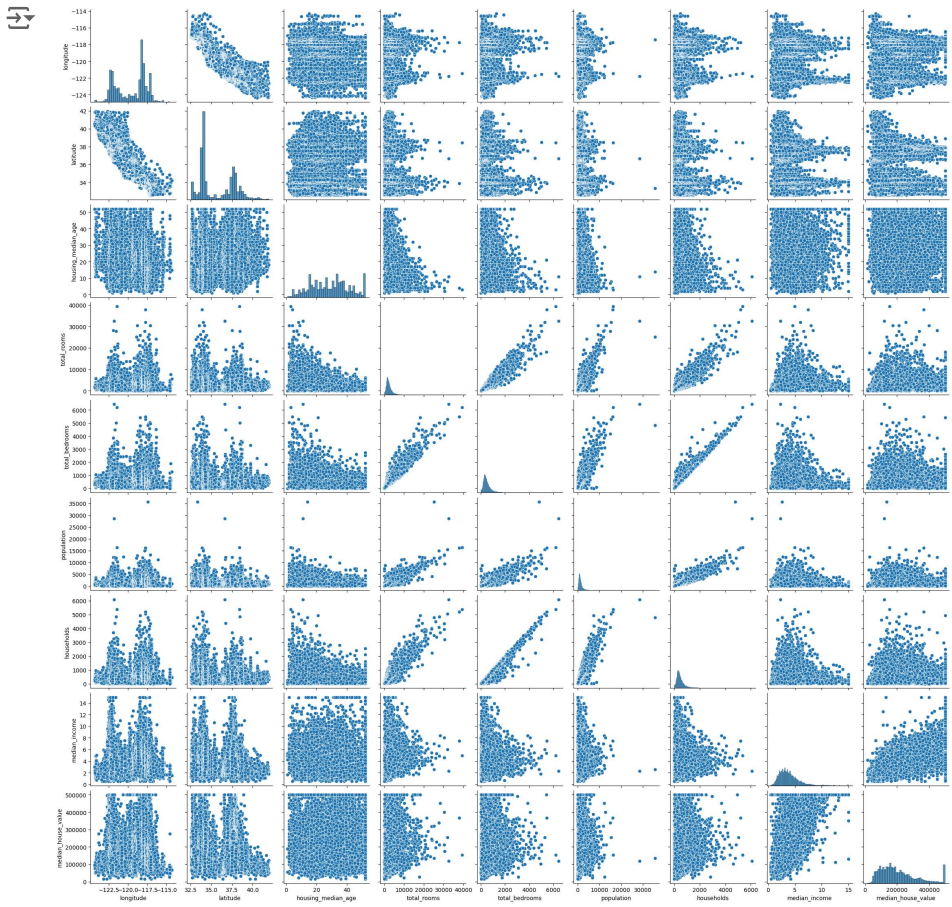
```
longitude             0
latitude              0
housing_median_age    0
total_rooms           0
```

```
total_bedrooms        0
population            0
households            0
median_income         0
median_house_value    0
ocean_proximity       0
dtype: int64
```

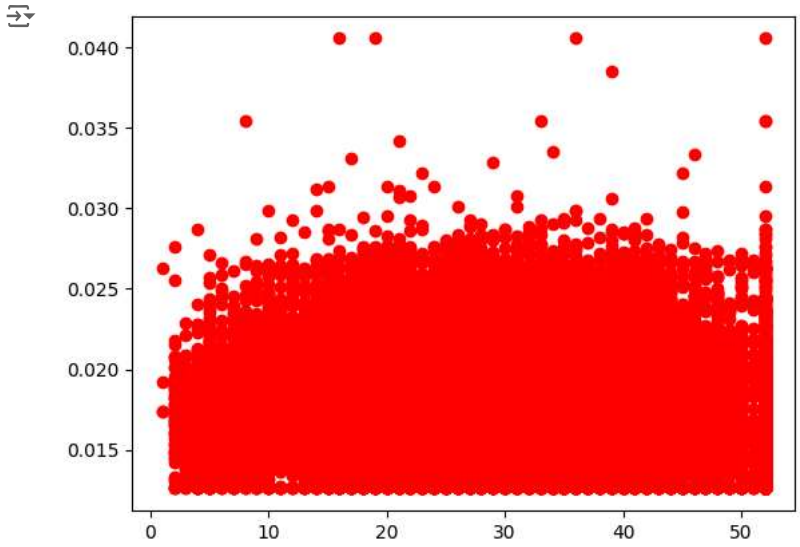
```
df.dtypes
```

```
longitude             float64
latitude              float64
housing_median_age    float64
total_rooms           float64
total_bedrooms        float64
population            float64
households            float64
median_income         float64
median_house_value    float64
ocean_proximity        object
dtype: object
```

```
plot=sns.pairplot(df)
```

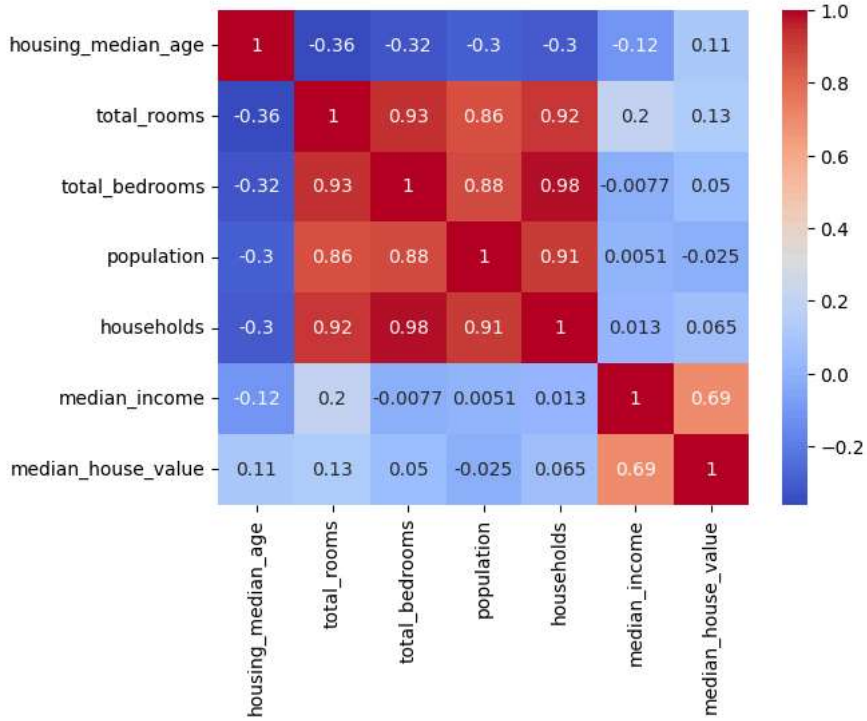


```
import math


x_plot=df['housing_median_age']
y_plot=df['median_house_value']
x = pow(x_plot,1)
y = pow(y_plot,-1/3)
plot=plt.scatter((x),(y),color='red')
```

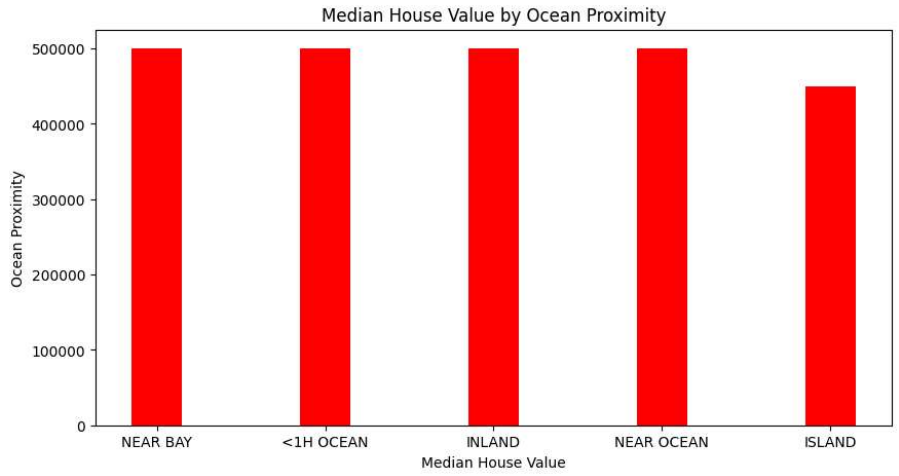



```
df1=df.drop(columns=['ocean_proximity','longitude','latitude'])


correlation_matrix=df1.corr()
sns.heatmap(correlation_matrix, cmap='coolwarm',annot=True)
```

```
<Axes: >
```



```
x=df['median_house_value']
y=df['ocean_proximity']
fig = plt.figure(figsize=(10, 5))

plt.bar(y, x, color ='red', width = 0.3)
plt.xlabel('Median House Value')
plt.ylabel('Ocean Proximity')
plt.title('Median House Value by Ocean Proximity')
plt.show()
```




```
df2=pd.DataFrame(df, columns=['ocean_proximity','median_house_value'])
df2["rank"] = df2.groupby(['ocean_proximity'])["median_house_value"].rank("dense", ascending=False)

df2[df2["rank"]==1][['ocean_proximity','median_house_value']]
```

	ocean_proximity	median_house_value
89	NEAR BAY	500001.0
459	NEAR BAY	500001.0
493	NEAR BAY	500001.0
494	NEAR BAY	500001.0
509	NEAR BAY	500001.0
...	...	...
20422	<1H OCEAN	500001.0
20426	<1H OCEAN	500001.0
20427	<1H OCEAN	500001.0
20436	<1H OCEAN	500001.0
20443	<1H OCEAN	500001.0

960 rows × 2 columns

```
df2.groupby(['ocean_proximity'])["median_house_value"].max()
```

```
ocean_proximity
<1H OCEAN     500001.0
INLAND        500001.0
ISLAND        450000.0
NEAR BAY      500001.0
NEAR OCEAN    500001.0
Name: median_house_value, dtype: float64
```

```
x = df1.drop('median_house_value',axis= 1)
y = df1['median_house_value']
print(x)
print(y)
```

```
       housing_median_age  total_rooms  total_bedrooms  population  \
0                    41.0        880.0           129.0       322.0
1                    21.0       7099.0          1106.0      2401.0
2                    52.0       1467.0           190.0       496.0
3                    52.0       1274.0           235.0       558.0
4                    52.0       1627.0           280.0       565.0
...                   ...          ...             ...         ...
20635                25.0       1665.0           374.0       845.0
20636                18.0        697.0           150.0       356.0
20637                17.0       2254.0           485.0      1007.0
20638                18.0       1860.0           409.0       741.0
20639                16.0       2785.0           616.0      1387.0

       households  median_income
0           126.0         8.3252
1          1138.0         8.3014
2           177.0         7.2574
3           219.0         5.6431
4           259.0         3.8462
...           ...            ...
20635       330.0         1.5603
20636       114.0         2.5568
20637       433.0         1.7000
20638       349.0         1.8672
20639       530.0         2.3886

[20433 rows x 6 columns]
0        452600.0
1        358500.0
2        352100.0
3        341300.0
4        342200.0
           ...
20635     78100.0
20636     77100.0
20637     92300.0
20638     84700.0
20639     89400.0
Name: median_house_value, Length: 20433, dtype: float64
```

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score, mean_absolute_error
from sklearn.metrics import accuracy_score
from sklearn.model_selection import cross_val_score


X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=42)


model = LinearRegression()
model.fit(X_train, y_train)
```

```
▾ LinearRegression
LinearRegression()
```

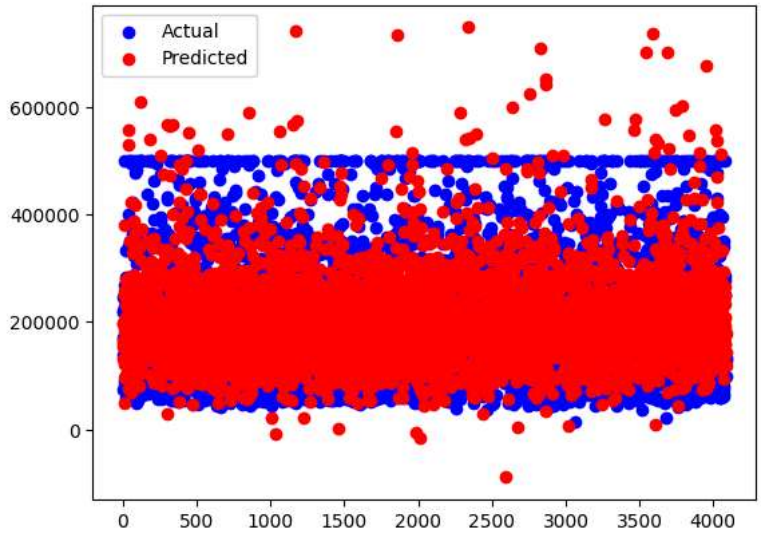
```
predictions = model.predict(X_test)
predictions
```

```
array([165480.22752968, 161093.72645382, 196854.06878554, ...,
       119439.91093011, 178676.94621081, 142692.55421869])
```

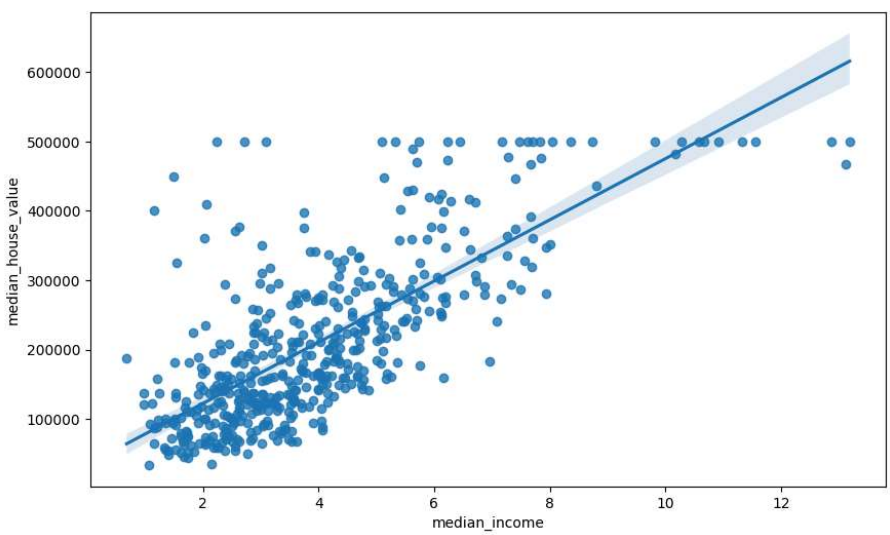
```
print('Mean Squared Error:', mean_squared_error(y_test, predictions))
print('R-squared:', r2_score(y_test, predictions))
print('Mean Absolute Error:', mean_absolute_error(y_test, predictions))
```

```
Mean Squared Error: 5865619646.959267
R-squared: 0.5710755317440979
Mean Absolute Error: 56642.49929908636
```

```
plt.scatter(range(len(y_test)), y_test, color='blue', label='Actual')
plt.scatter(range(len(y_test)), predictions, color='red', label='Predicted')
plt.legend()
plt.show()
```




```
X1 = x.sample(500, random_state=1)
y1 = y.sample(500, random_state=1)

plt.figure(figsize=(10,6))
sns.regplot(x=X1['median_income'],y=y1, data=df1)
plt.show()
```




```
train_score = model.score(X_train, y_train)
test_score = model.score(X_test, y_test)

print('Linear regression score: \n')
print("Training Score:",round(train_score*100),'%')
print("Testing Score:",round(test_score*100),'%')
```

```
Linear regression score:

Training Score: 57 %
Testing Score: 57 %
```

```
max_pred=predictions.max()
min_pred=predictions.min()
print(max_pred)
print(min_pred)
```

```
747530.8747665383
-88934.66004201063
```

```
r2=r2_score(y_test, predictions)
```

```
r2
```

```
0.5710755317440979
```