In [ ]:

```

```

In [1]:

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

from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score, mean_squared_error, mean_absolute_error
```

In [2]:

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

Out[2]:

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_ir
0	-122.23	37.88	41.0	880.0	129.0	322.0	126.0	
1	-122.22	37.86	21.0	7099.0	1106.0	2401.0	1138.0	
2	-122.24	37.85	52.0	1467.0	190.0	496.0	177.0	
3	-122.25	37.85	52.0	1274.0	235.0	558.0	219.0	
4	-122.25	37.85	52.0	1627.0	280.0	565.0	259.0	
...	...	...	...	...	...	...	...	
20635	-121.09	39.48	25.0	1665.0	374.0	845.0	330.0	
20636	-121.21	39.49	18.0	697.0	150.0	356.0	114.0	
20637	-121.22	39.43	17.0	2254.0	485.0	1007.0	433.0	
20638	-121.32	39.43	18.0	1860.0	409.0	741.0	349.0	
20639	-121.24	39.37	16.0	2785.0	616.0	1387.0	530.0	

20640 rows × 10 columns

In [3]:

```
data.isnull().sum()
```

Out[3]:

```
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
```

In [ ]:

```

```

In [4]:

```
#total rows and columns in dataset
data.shape
```

Out[4]:

```
(20640, 10)
```

In [5]:
```python
data['ocean_proximity'].value_counts()
```

Out[5]:
```
<1H OCEAN     9136
INLAND        6551
NEAR OCEAN    2658
NEAR BAY      2290
ISLAND           5
Name: ocean_proximity, dtype: int64
```

In [6]:
```python
data.describe()
```

Out[6]:

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	household
count	20640.000000	20640.000000	20640.000000	20640.000000	20433.000000	20640.000000	20640.00000
mean	-119.569704	35.631861	28.639486	2635.763081	537.870553	1425.476744	499.53968
std	2.003532	2.135952	12.585558	2181.615252	421.385070	1132.462122	382.32975
min	-124.350000	32.540000	1.000000	2.000000	1.000000	3.000000	1.00000
25%	-121.800000	33.930000	18.000000	1447.750000	296.000000	787.000000	280.00000
50%	-118.490000	34.260000	29.000000	2127.000000	435.000000	1166.000000	409.00000
75%	-118.010000	37.710000	37.000000	3148.000000	647.000000	1725.000000	605.00000
max	-114.310000	41.950000	52.000000	39320.000000	6445.000000	35682.000000	6082.00000

In [7]:
```python
data.corr()
```

Out[7]:

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	househ
longitude	1.000000	-0.924664	-0.108197	0.044568	0.069608	0.099773	0.05
latitude	-0.924664	1.000000	0.011173	-0.036100	-0.066983	-0.108785	-0.07
housing_median_age	-0.108197	0.011173	1.000000	-0.361262	-0.320451	-0.296244	-0.30
total_rooms	0.044568	-0.036100	-0.361262	1.000000	0.930380	0.857126	0.91
total_bedrooms	0.069608	-0.066983	-0.320451	0.930380	1.000000	0.877747	0.97
population	0.099773	-0.108785	-0.296244	0.857126	0.877747	1.000000	0.90
households	0.055310	-0.071035	-0.302916	0.918484	0.979728	0.907222	1.00
median_income	-0.015176	-0.079809	-0.119034	0.198050	-0.007723	0.004834	0.01
median_house_value	-0.045967	-0.144160	0.105623	0.134153	0.049686	-0.024650	0.06

In [8]:
```python
#knowing highest price of house
data.groupby('ocean_proximity')['median_house_value'].max()
```

Out[8]:
```
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
```

In [9]:
```python
#average of house price
data.groupby('ocean_proximity')['households'].mean()
```

Out[9]:
```
ocean_proximity
<1H OCEAN     517.744965
INLAND        477.447565
```

```
ISLAND        276.600000
NEAR BAY      488.616157
NEAR OCEAN    501.244545
Name: households, dtype: float64
```

In [10]:
```python
data.isnull().sum()
```

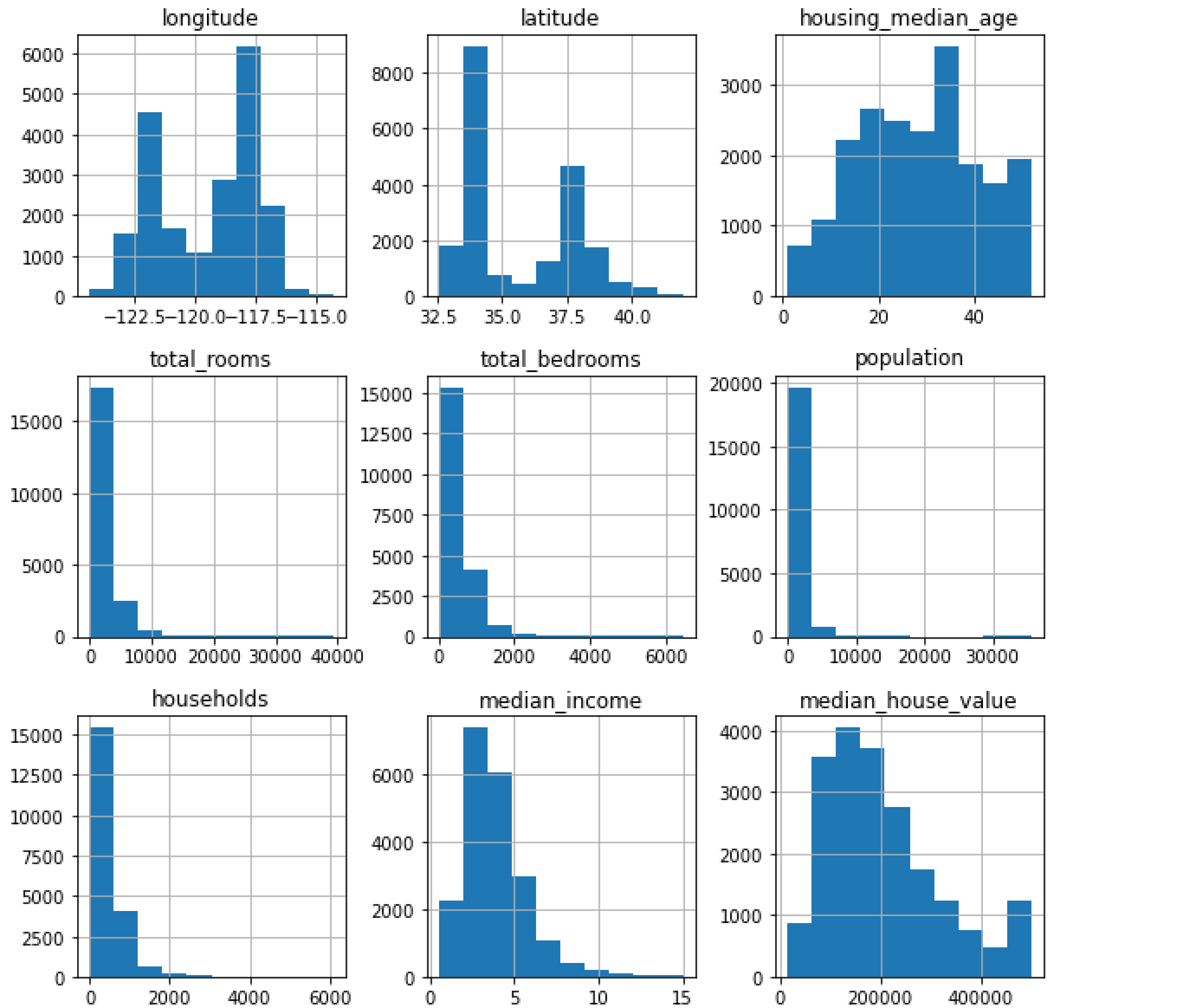
Out[10]:
```
longitude               0
latitude                0
housing_median_age      0
total_rooms             0
total_bedrooms        207
population              0
households              0
median_income           0
median_house_value      0
ocean_proximity         0
dtype: int64
```

In [11]:
```python
data.dropna(axis =0,inplace = True)
```

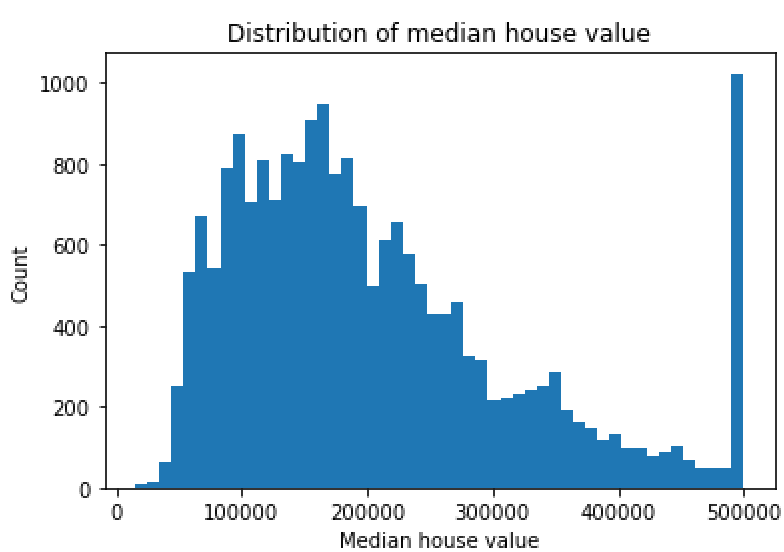
In [12]:
```python
data.isnull().sum()
```

Out[12]:
```
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
```

In [13]:
```python
data.hist(figsize=(10,10))
plt.show()
```



In [14]:
```
#graph of house price
plt.hist(data['median_house_value'], bins=50)
plt.xlabel('Median house value')
plt.ylabel('Count')
plt.title('Distribution of median house value')
plt.show()
```



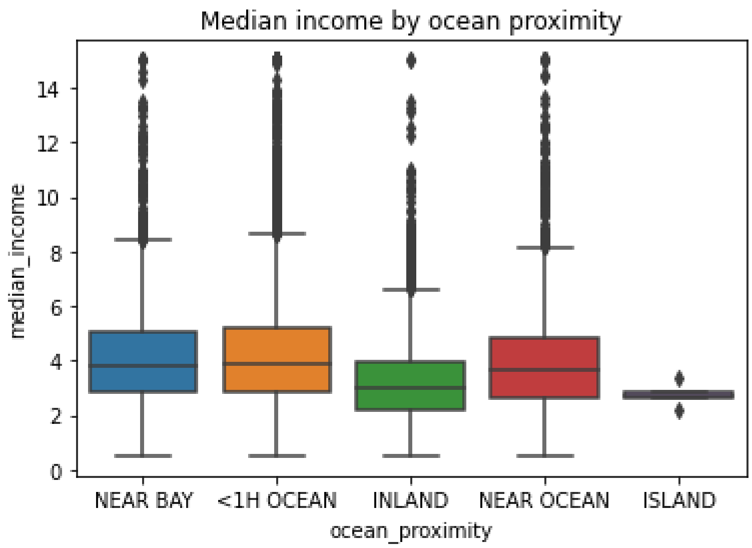
In [15]:
```
# # Calculate the interquartile range (IQR)
Q1 = data['median_house_value'].quantile(0.25)
```

```
Q3 = data['median_house_value'].quantile(0.75)
IQR = Q3 - Q1
```

In [16]:
```
IQR
```

Out[16]:
```
145200.0
```

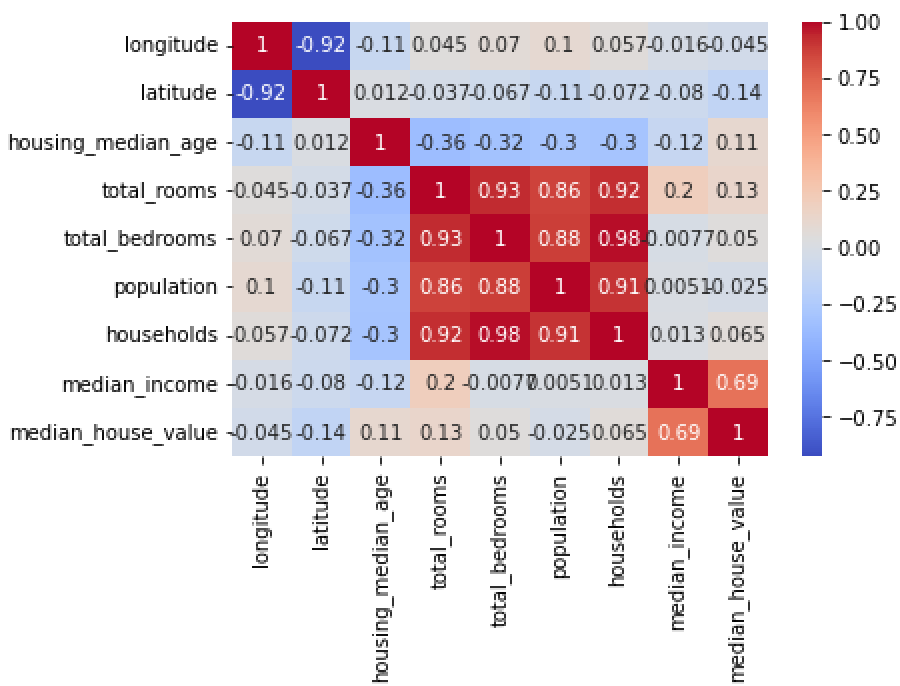
In [18]:
```
sns.boxplot(x='ocean_proximity', y='median_income', data=data)
plt.title('Median income by ocean proximity')
plt.show()
```



In [19]:
```
corr_matrix = data.corr()

# Create a heatmap of the correlation matrix using Seaborn
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')

# Show the plot
plt.show()
```



In [20]:
```
corr_matrix["median_house_value"].sort_values(ascending=False)
```

Out[20]:
```
median_house_value    1.000000
median_income         0.688355
total_rooms           0.133294
```

```
housing_median_age    0.106432
households            0.064894
total_bedrooms        0.049686
population           -0.025300
longitude            -0.045398
latitude             -0.144638
Name: median_house_value, dtype: float64
```

In [ ]:

In [21]:
```
df = pd.get_dummies(data, columns = ["ocean_proximity"], drop_first = True)
```

In [22]:
```
df.isnull().sum()
```

Out[22]:
```
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_INLAND        0
ocean_proximity_ISLAND        0
ocean_proximity_NEAR BAY      0
ocean_proximity_NEAR OCEAN    0
dtype: int64
```

In [23]:
```
X = df.drop(["median_house_value"], axis = 1)
y = df[["median_house_value"]]
```

In [24]:
```
y.dropna(axis = 0)
```

Out[24]:

	median_house_value
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

20433 rows × 1 columns

In [ ]:

In [30]:
```
x_train,x_test,y_train,y_test = train_test_split(X,y,test_size = 0.2, random_state=42)
x_train
```

Out[30]:

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_ir
17727	-121.80	37.32	14.0	4412.0	924.0	2698.0	891.0	
2057	-119.63	36.64	33.0	1036.0	181.0	620.0	174.0	
6453	-118.06	34.12	25.0	3891.0	848.0	1848.0	759.0	
4619	-118.31	34.07	28.0	2362.0	949.0	2759.0	894.0	
15266	-117.27	33.04	27.0	1839.0	392.0	1302.0	404.0	
...	...	...	...	...	...	...	...	
11397	-117.97	33.72	24.0	2991.0	500.0	1437.0	453.0	
12081	-117.54	33.76	5.0	5846.0	1035.0	3258.0	1001.0	
5447	-118.42	34.01	42.0	1594.0	369.0	952.0	362.0	
866	-122.04	37.57	12.0	5719.0	1064.0	3436.0	1057.0	
15948	-122.43	37.73	52.0	3602.0	738.0	2270.0	647.0	

16346 rows × 12 columns

In [26]:

```
linear_model = LinearRegression()
tree_model = DecisionTreeRegressor()
forest_model = RandomForestRegressor()
```

In [27]:

```
import warnings
warnings.filterwarnings('ignore')
# Training and evaluate each model
for model in [linear_model, tree_model, forest_model]:
    model.fit(x_train, y_train)
    y_pred = model.predict(x_test)
    mse = mean_squared_error(y_test, y_pred)
    print(f"{model.__class__.__name__} MSE: {mse:.2f}")
```

```
LinearRegression MSE: 4802173538.60
DecisionTreeRegressor MSE: 4437392130.75
RandomForestRegressor MSE: 2356136302.00
```

In [29]:

```
r2 = r2_score(y_test, y_pred)

print("R2 score:", r2)
```

```
R2 score: 0.8277071185482207
```

In [ ]: