In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
data=pd.read_csv("housing.csv")
```

In [3]:
```
data
```

Out[3]:

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	popu
0	-122.23	37.88	41	880	129.0	
1	-122.22	37.86	21	7099	1106.0	
2	-122.24	37.85	52	1467	190.0	
3	-122.25	37.85	52	1274	235.0	
4	-122.25	37.85	52	1627	280.0	
...	...	...	...	...	...	
20635	-121.09	39.48	25	1665	374.0	
20636	-121.21	39.49	18	697	150.0	
20637	-121.22	39.43	17	2254	485.0	
20638	-121.32	39.43	18	1860	409.0	
20639	-121.24	39.37	16	2785	616.0	

20640 rows × 10 columns

In [4]:
```
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  int64  
 3   total_rooms         20640 non-null  int64  
 4   total_bedrooms      20433 non-null  float64
 5   population          20640 non-null  int64  
 6   households          20640 non-null  int64  
 7   median_income       20640 non-null  float64
 8   ocean_proximity     20640 non-null  object 
 9   median_house_value  20640 non-null  int64  
dtypes: float64(4), int64(5), object(1)
memory usage: 1.6+ MB
```

In [5]:
```
data.dropna(inplace=True)
```

In [6]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 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  int64  
 3   total_rooms         20433 non-null  int64  
 4   total_bedrooms      20433 non-null  float64
 5   population          20433 non-null  int64  
 6   households          20433 non-null  int64  
 7   median_income       20433 non-null  float64
 8   ocean_proximity     20433 non-null  object 
 9   median_house_value  20433 non-null  int64  
dtypes: float64(4), int64(5), object(1)
memory usage: 1.7+ MB
```

In [7]:
```
from sklearn.model_selection import train_test_split

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

In [8]:
```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3)
```

In [9]:
```
train_data=X_train.join(y_train)
train_data
```

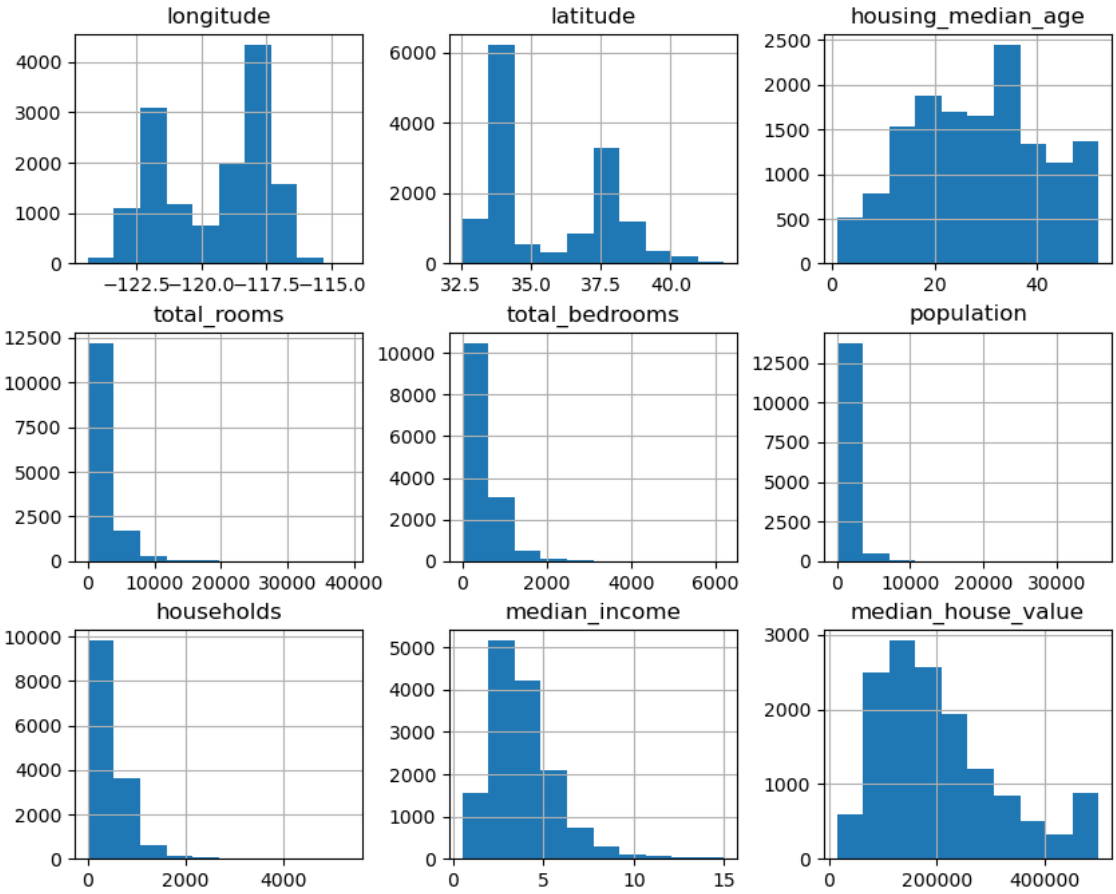
Out[9]:

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	popu
2130	-119.71	36.80	17	2056	366.0	
5868	-118.35	34.18	46	2711	491.0	
2815	-119.05	35.42	35	2353	483.0	
5704	-118.26	34.23	38	1107	194.0	
486	-122.26	37.86	52	3497	832.0	
...	...	...	...	...	...	
7145	-118.13	34.02	43	396	91.0	
4713	-118.36	34.06	52	2130	455.0	
5780	-118.26	34.15	18	2481	756.0	
15957	-122.46	37.71	52	1580	337.0	
12849	-121.39	38.69	38	300	47.0	

14303 rows × 10 columns

In [10]:
```
train_data.hist(figsize=(10,8))
```

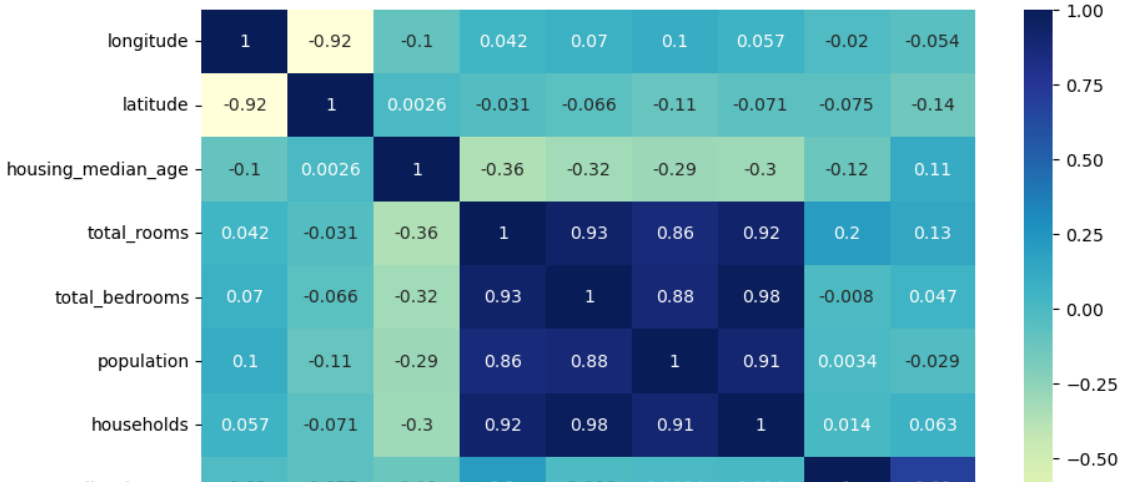
Out[10]:
```
array([[<Axes: title={'center': 'longitude'}>,
        <Axes: title={'center': 'latitude'}>,
        <Axes: title={'center': 'housing_median_age'}>],
       [<Axes: title={'center': 'total_rooms'}>,
        <Axes: title={'center': 'total_bedrooms'}>,
        <Axes: title={'center': 'population'}>],
       [<Axes: title={'center': 'households'}>,
        <Axes: title={'center': 'median_income'}>,
        <Axes: title={'center': 'median_house_value'}>]], dtype=object)
```

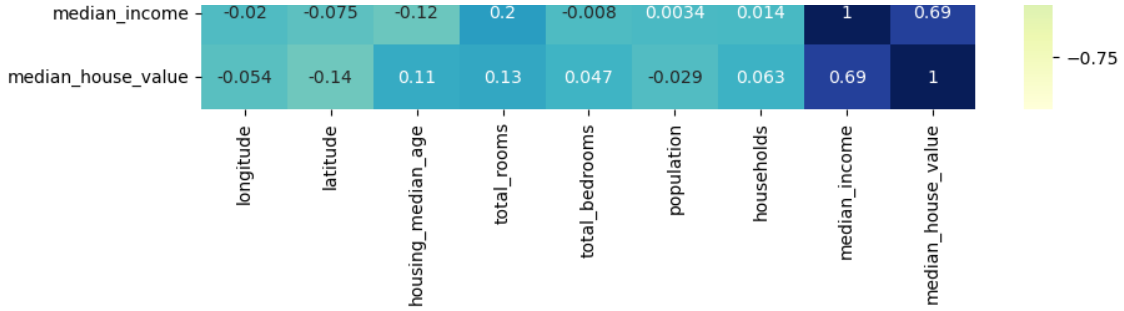


In [11]:
```
plt.figure(figsize=(10,6))
sns.heatmap(train_data.corr(),annot=True, cmap="YlGnBu")
```

```
C:\Users\Sruti Dey\AppData\Local\Temp\ipykernel_8560\3354081449.py:2: FutureWa
rning: The default value of numeric_only in DataFrame.corr is deprecated. In a
future version, it will default to False. Select only valid columns or specify
the value of numeric_only to silence this warning.
  sns.heatmap(train_data.corr(),annot=True, cmap="YlGnBu")
```

Out[11]: <Axes: >





In [12]:

```
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
```

Out[12]:

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	popu
2130	-119.71	36.80	17	7.629004	5.905362	7.1
5868	-118.35	34.18	46	7.905442	6.198479	7.1
2815	-119.05	35.42	35	7.763871	6.182085	7.2
5704	-118.26	34.23	38	7.010312	5.273000	6.2
486	-122.26	37.86	52	8.159947	6.725034	7.3
...	...	...	...	...	...	
7145	-118.13	34.02	43	5.983936	4.521789	5.5
4713	-118.36	34.06	52	7.664347	6.122493	6.8
5780	-118.26	34.15	18	7.816820	6.629363	7.4
15957	-122.46	37.71	52	7.365813	5.823046	7.2
12849	-121.39	38.69	38	5.707110	3.871201	5.0

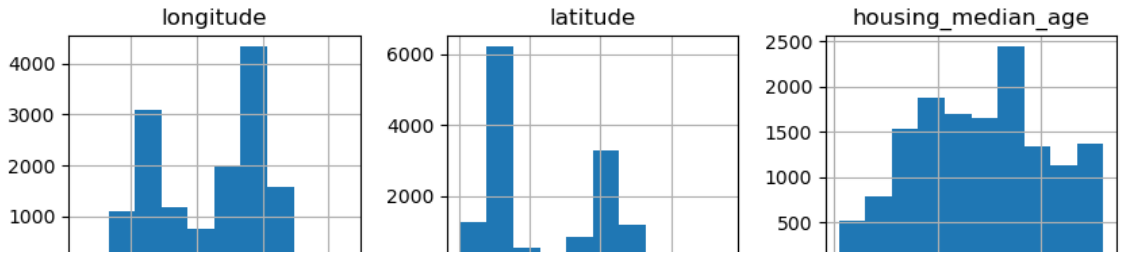
14303 rows × 10 columns

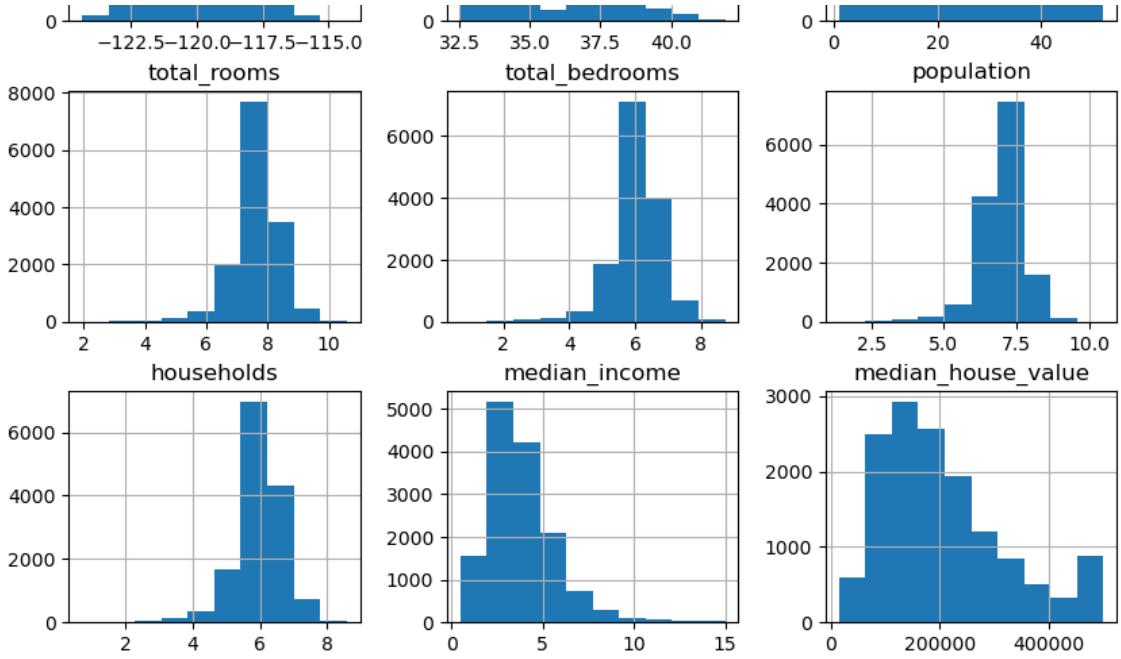
In [13]:

```
train_data.hist(figsize=(10,8))
```

Out[13]:

```
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)
```

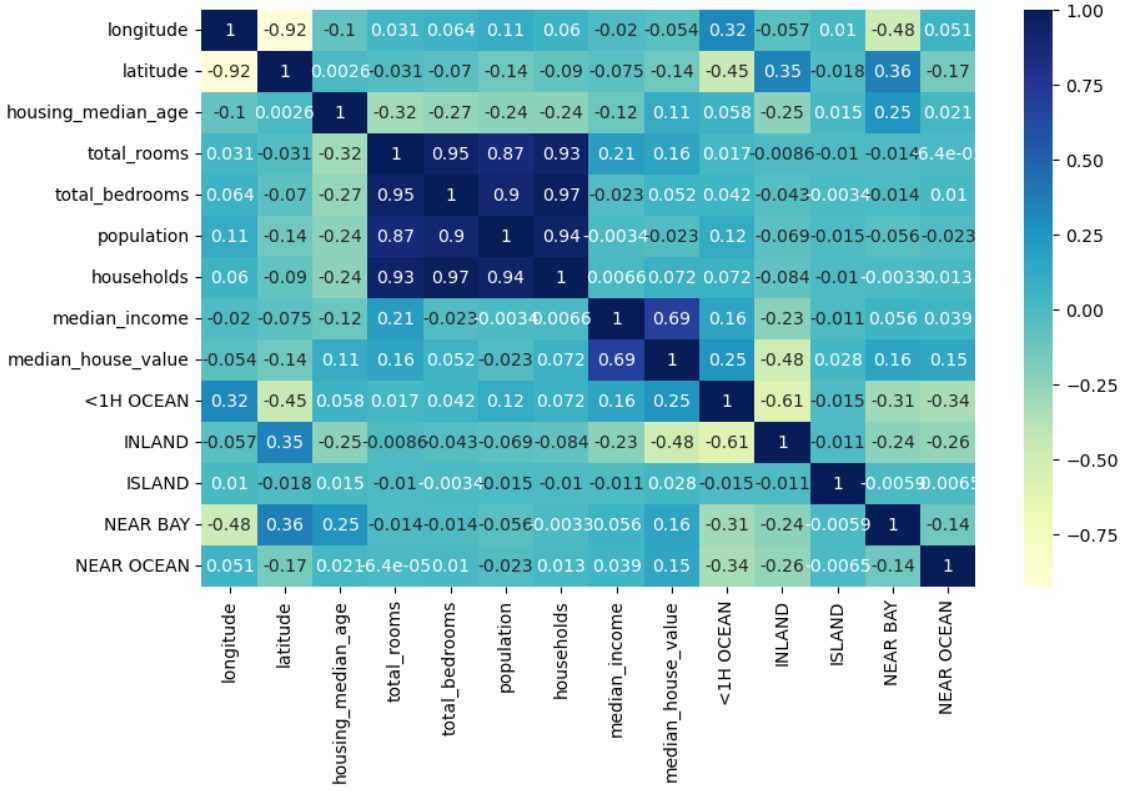




In [14]:
```
train_data.ocean_proximity.value_counts()
pd.get_dummies(train_data.ocean_proximity)
train_data.join(pd.get_dummies(train_data.ocean_proximity))
train_data=train_data.join(pd.get_dummies(train_data.ocean_proximity)).drop(
```

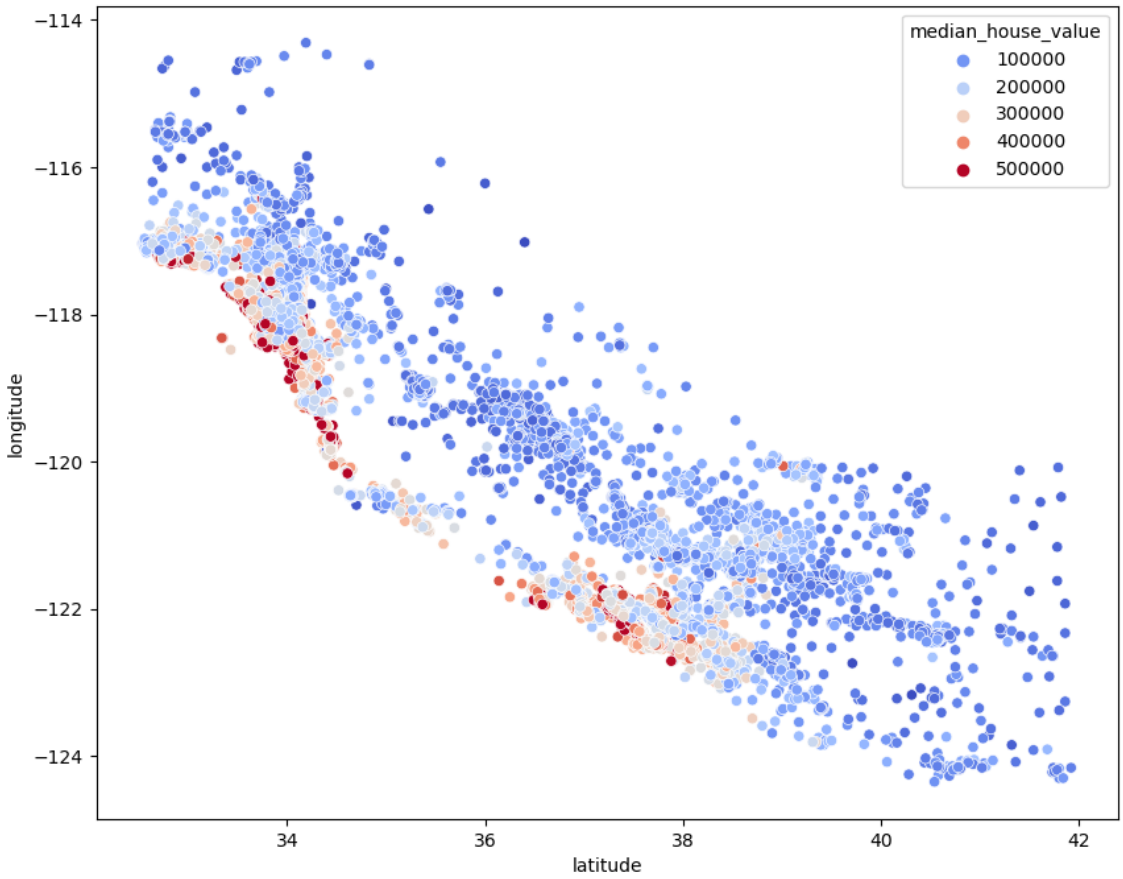
In [15]:
```
plt.figure(figsize=(10,6))
sns.heatmap(train_data.corr(),annot=True, cmap="YlGnBu")
```

Out[15]: <Axes: >



In [16]:
```
plt.figure(figsize=(10,8))
sns.scatterplot(x="latitude",y="longitude",data=train_data,hue="median_house
```

Out[16]: <Axes: xlabel='latitude', ylabel='longitude'>



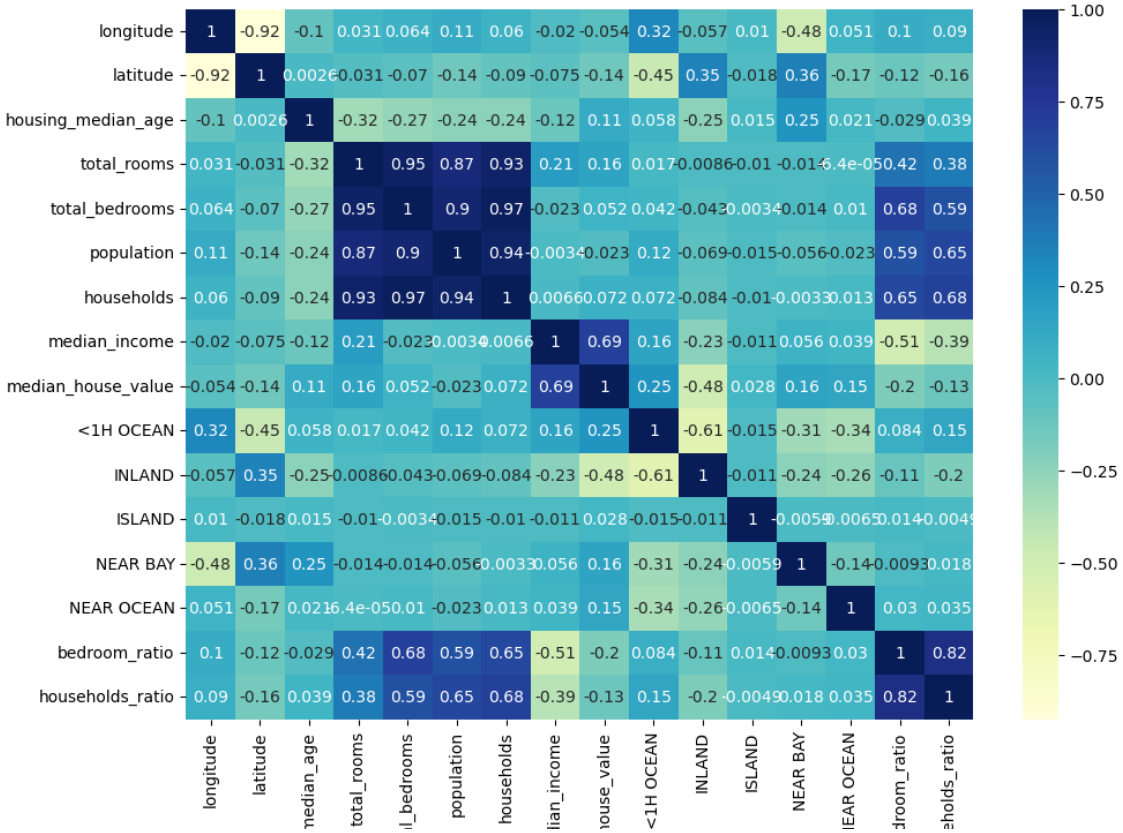
In [17]:

```
train_data['bedroom_ratio']=train_data['total_bedrooms']/train_data['total_r
train_data['households_ratio']=train_data['households']/train_data['total_ro
```

In [18]:

```
plt.figure(figsize=(11,8))
sns.heatmap(train_data.corr(),annot=True, cmap="YlGnBu")
```

Out[18]: <Axes: >



Linear regression

In [19]:
```
from sklearn.linear_model import LinearRegression
train_data=train_data.drop(['ISLAND'],axis=1)
X_train, y_train= train_data.drop(['median_house_value'],axis=1), train_data

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

Out[19]:
```
LinearRegression()
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [20]:
```
train_data
```

Out[20]:

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	popu
2130	-119.71	36.80	17	7.629004	5.905362	7.1
5868	-118.35	34.18	46	7.905442	6.198479	7.1
2815	-119.05	35.42	35	7.763871	6.182085	7.2
5704	-118.26	34.23	38	7.010312	5.273000	6.2
486	-122.26	37.86	52	8.159947	6.725034	7.3
...	...	...	...	...	...	
7145	-118.13	34.02	43	5.983936	4.521789	5.5
4713	-118.36	34.06	52	7.664347	6.122493	6.8
5780	-118.26	34.15	18	7.816820	6.629363	7.4
15957	-122.46	37.71	52	7.365813	5.823046	7.2
12849	-121.39	38.69	38	5.707110	3.871201	5.0

14303 rows × 15 columns

In [21]:
```
test_data=X_test.join(y_test)
test_data
```

Out[21]:

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	popu
7495	-118.25	33.93	38	180	43.0	
18847	-122.38	41.43	45	2245	448.0	
13646	-117.31	34.08	43	1697	387.0	

3594	-118.49	34.25	30	2871	470.0
18827	-122.26	41.66	17	1885	350.0
...	...	...	...	...	...
9655	-120.63	36.98	20	2380	489.0
11367	-117.95	33.74	21	3576	554.0
18752	-122.42	40.63	23	2248	489.0
16622	-120.85	35.37	21	1033	195.0
5135	-118.26	33.97	46	1521	352.0

6130 rows × 10 columns

In [22]:
```
test_data['total_rooms']=np.log(test_data['total_rooms']+1)
test_data['total_bedrooms']=np.log(test_data['total_bedrooms']+1)
test_data['population']=np.log(test_data['population']+1)
test_data['households']=np.log(test_data['households']+1)
```

In [23]:
```
test_data=test_data.join(pd.get_dummies(test_data.ocean_proximity)).drop(['o
test_data=test_data.drop(['ISLAND'],axis=1)
```

In [24]:
```
test_data['bedroom_ratio']=test_data['total_bedrooms']/test_data['total_room
test_data['households_ratio']=test_data['households']/test_data['total_rooms
```

In [25]:
```
X_test, y_test= test_data.drop(['median_house_value'],axis=1), test_data['me
```

In [26]:
```
test_data
```

Out[26]:

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	popu
7495	-118.25	33.93	38	5.198497	3.784190	5.5
18847	-122.38	41.43	45	7.716906	6.107023	7.0
13646	-117.31	34.08	43	7.437206	5.961005	7.0
3594	-118.49	34.25	30	7.962764	6.154858	7.1
18827	-122.26	41.66	17	7.542213	5.860786	6.8
...	...	...	...	...	...	
9655	-120.63	36.98	20	7.775276	6.194405	7.3
11367	-117.95	33.74	21	8.182280	6.318968	7.5
18752	-122.42	40.63	23	7.718241	6.194405	7.0
16622	-120.85	35.37	21	6.941190	5.278115	6.3
5135	-118.26	33.97	46	7.327781	5.866468	7.0

6130 rows × 15 columns

In [27]:
```
reg.score(X_test,y_test)
```

Out[27]: 0.6675915831011885

random-forest (Extra)

In [32]:
```
from sklearn.ensemble import RandomForestRegressor
forest=RandomForestRegressor()
forest.fit(X_train,y_train)
```

Out[32]: RandomForestRegressor()

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [33]:
```
forest.score(X_test,y_test)
```

Out[33]: 0.8188478263561374

In [ ]: