In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
%matplotlib inline
warnings.filterwarnings('ignore')
from keras.models import Sequential
from keras.layers import Dense
from sklearn import linear_model
from sklearn.metrics import mean_squared_error,mean_absolute_error
```

In [2]:
```
from tensorflow.keras.datasets import boston_housing
```

In [3]:
```
(X_train,y_train),(X_test,y_test)=boston_housing.load_data()
```

In [4]:
```
from sklearn.preprocessing import StandardScaler
scaler=StandardScaler()
scaler.fit(X_train)
xts=scaler.transform(X_train)
xtss=scaler.transform(X_test)
```

In [7]:
```
model=Sequential()
model.add(Dense(128,input_dim=13,activation='relu'))
model.add(Dense(64,activation='relu'))
model.add(Dense(1,activation='linear'))
```

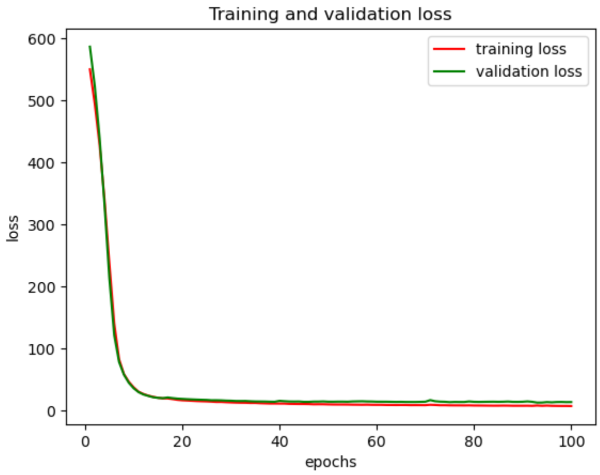
In [9]:
```
model.compile(loss="mean_squared_error",optimizer='adam',metrics=['mae'])
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 128)               1792

 dense_1 (Dense)             (None, 64)                8256

 dense_2 (Dense)             (None, 1)                 65

=================================================================
Total params: 10,113
Trainable params: 10,113
Non-trainable params: 0
_________________________________________________________________
```

In [10]:
```
history=model.fit(xts,y_train,validation_split=0.2,epochs=100)
```

```
Epoch 9/100
11/11 [==============================] - 0s 4ms/step - loss: 46.3007 - mae: 4.9837 - val_loss: 44.6090 - val_mae: 5.0091
Epoch 10/100
11/11 [==============================] - 0s 5ms/step - loss: 37.3325 - mae: 4.3813 - val_loss: 35.7505 - val_mae: 4.4251
Epoch 11/100
11/11 [==============================] - 0s 6ms/step - loss: 30.2102 - mae: 3.8977 - val_loss: 29.5624 - val_mae: 4.0142
Epoch 12/100
11/11 [==============================] - 0s 7ms/step - loss: 26.5790 - mae: 3.5750 - val_loss: 25.7387 - val_mae: 3.7614
Epoch 13/100
11/11 [==============================] - 0s 6ms/step - loss: 23.8737 - mae: 3.3828 - val_loss: 23.4518 - val_mae: 3.5987
Epoch 14/100
11/11 [==============================] - 0s 4ms/step - loss: 21.9894 - mae: 3.2736 - val_loss: 21.2639 - val_mae: 3.5101
Epoch 15/100
11/11 [==============================] - 0s 6ms/step - loss: 20.3797 - mae: 3.2075 - val_loss: 20.2995 - val_mae: 3.4204
Epoch 16/100
11/11 [==============================] - 0s 5ms/step - loss: 19.5081 - mae: 3.0902 - val_loss: 19.7428 - val_mae: 3.3475
Epoch 17/100
11/11 [==============================] - 0s 5ms/step - loss: 19.3336 - mae: 3.1207 - val_loss: 20.9067 - val_mae: 3.4753
Epoch 18/100
```

In [11]:
```
from matplotlib import pyplot as plt
loss=history.history['loss']
val_loss=history.history['val_loss']
e=range(1,len(loss)+1)
plt.plot(e,loss,'r',label='training loss')
plt.plot(e,val_loss,'g',label='validation loss')
plt.title('Training and validation loss')
plt.xlabel('epochs')
plt.ylabel('loss')
plt.legend()
plt.show()
```



In [12]:
```
mse_neural,mae_neural=model.evaluate(xtss,y_test)
print("mean squared error from neural net:",mse_neural)
print("mean absolute error from neural net:",mae_neural)
```

```
4/4 [==============================] - 0s 3ms/step - loss: 24.9424 - mae: 3.0239
mean squared error from neural net: 24.942373275756836
mean absolute error from neural net: 3.023918390274048
```

```
In [13]: p=model.predict(xtss[:5])
         print("prediction values are:",p)
         print("real values are:",y_test[:5])
```

```
1/1 [==============================] - 0s 72ms/step
prediction values are: [[ 8.196472]
 [17.955849]
 [21.273643]
 [32.10984 ]
 [25.273575]]
real values are: [ 7.2 18.8 19.  27.  22.2]
```

```
In [ ]:
```