In [1]:
```
import pandas as pd
```

In [2]:
```
import numpy as np
```

In [3]:
```
import matplotlib.pyplot as plt
```

In [15]:
```
df = pd.read_excel("Downloads/bostonhousing (1).xlsx")
```

In [18]:
```
df.shape
```

Out[18]:
```
(506, 14)
```

In [19]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 506 entries, 0 to 505
Data columns (total 14 columns):
 #   Column   Non-Null Count  Dtype  
---  ------   --------------  -----  
 0   crim     506 non-null    float64
 1   zn       506 non-null    float64
 2   indus    506 non-null    float64
 3   chas     506 non-null    int64  
 4   nox      506 non-null    float64
 5   rm       506 non-null    float64
 6   age      506 non-null    float64
 7   dis      506 non-null    float64
 8   rad      506 non-null    int64  
 9   tax      506 non-null    int64  
 10  ptratio  506 non-null    float64
 11  b        506 non-null    float64
 12  lstat    506 non-null    float64
 13  medv     506 non-null    float64
dtypes: float64(11), int64(3)
memory usage: 55.5 KB
```

In [24]:
```
boston = {
    "data": X.values,
    "target": y.values,
    "feature_names": X.columns
}

print(boston["feature_names"])
```

```
Index(['crim', 'zn', 'indus', 'chas', 'nox', 'rm', 'age', 'dis', 'rad', 'tax',
       'ptratio', 'b', 'lstat'],
      dtype='object')
```

In [27]:
```
data = pd.DataFrame(df)
data.head()
```

Out[27]:

	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	b	lstat	r
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.98	
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.14	
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	5.33	

In [31]:
```
data.columns
```

Out[31]:
```
Index(['crim', 'zn', 'indus', 'chas', 'nox', 'rm', 'age', 'dis', 'rad', 'tax',
       'ptratio', 'b', 'lstat', 'medv'],
      dtype='object')
```

In [32]:
```
X = df.drop("medv", axis=1)
```

In [35]:
```
data = X.copy()
data.columns = X.columns
data.head()
```

Out[35]:

	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	b	lstat
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.98
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.14
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	5.33

In [34]:
```
data.rename(columns={'medv': 'PRICE'}, inplace=True)
data.head()
```

Out[34]:

	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	b	lstat
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.98
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.14
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	5.33

In [36]:
```
data.isnull().sum()[data.isnull().sum() > 0]
```

Out[36]:
```
Series([], dtype: int64)
```

In [40]:
```
data.columns = data.columns.str.strip()
```

In [42]:
```
target_column = data.columns[-1]
```

In [43]:
```
x = data.drop(target_column, axis=1)
y = data[target_column]
```

In [44]:
```
print("Target column is:", target_column)
print(x.shape, y.shape)
```

```
Target column is: lstat
(506, 12) (506,)
```

In [45]:
```
from sklearn.model_selection import train_test_split

xtrain, xtest, ytrain, ytest = train_test_split(
    x, y,
    test_size=0.2,
    random_state=0
)
```

In [46]:
```
print(xtrain.shape)
print(xtest.shape)
print(ytrain.shape)
print(ytest.shape)
```

```
(404, 12)
(102, 12)
(404,)
(102,)
```

In [47]:
```
from sklearn.linear_model import LinearRegression
```

In [48]:
```
lm = LinearRegression()
```

In [49]:
```
model = lm.fit(xtrain, ytrain)

print("Model training completed ✅")
```

```
Model training completed ✅
```

In [53]:
```
from sklearn.linear_model import LinearRegression

lm = LinearRegression()
lm.fit(xtrain, ytrain)
```

Out[53]:

▾ LinearRegression

```
LinearRegression()
```

In [54]:
```
ytrain_pred = lm.predict(xtrain)
ytest_pred = lm.predict(xtest)
```

In [55]:
```
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score

print("Train R2:", r2_score(ytrain, ytrain_pred))
print("Test R2:", r2_score(ytest, ytest_pred))
```

```
Train R2: 0.6830386021775365
Test R2: 0.4970790902412068
```

In [56]:
```
from sklearn.metrics import mean_squared_error, r2_score
```

In [57]:
```
mse_test = mean_squared_error(ytest, ytest_pred)
print("Test MSE:", mse_test)
```

```
Test MSE: 21.085848548855292
```

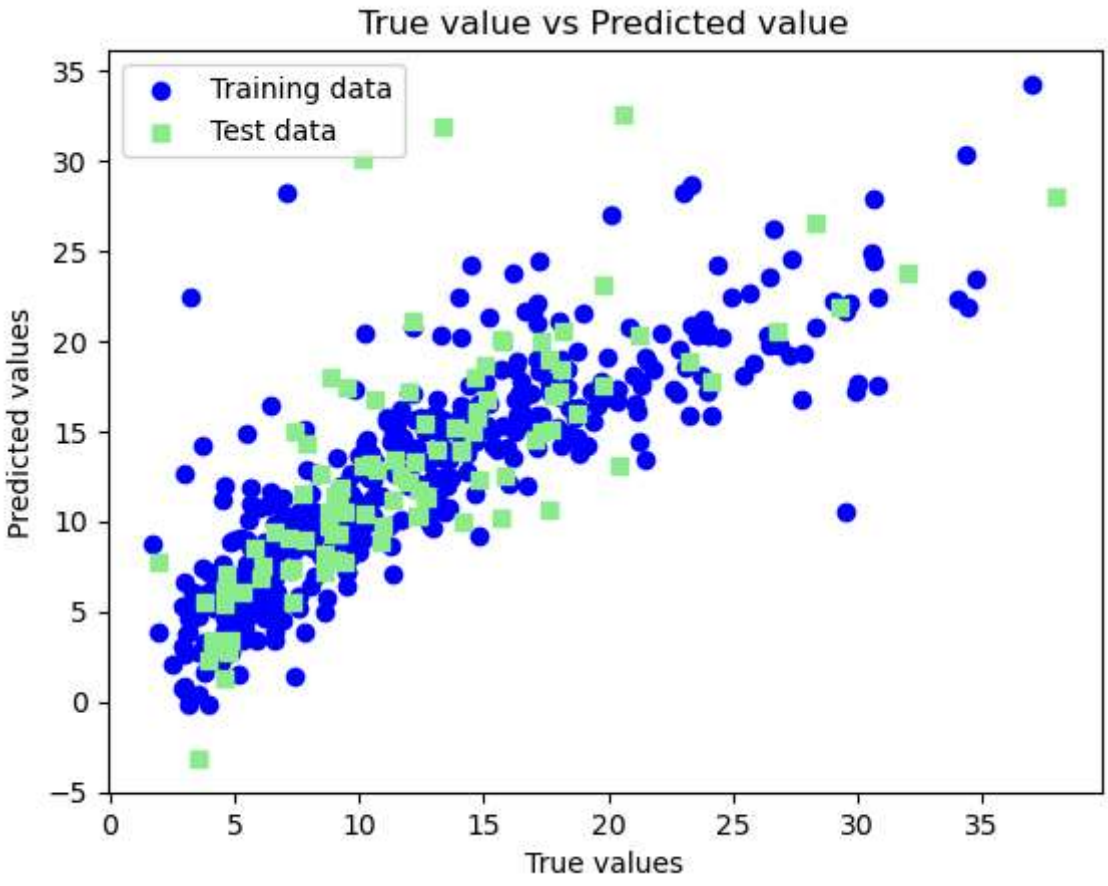
In [58]:
```
mse_train = mean_squared_error(ytrain, ytrain_pred)
print("Train MSE:", mse_train)
```

```
Train MSE: 16.84458940970367
```

In [61]:
```
plt.scatter(ytrain, ytrain_pred, c='blue', marker='o', label='Training data')
plt.scatter(ytest, ytest_pred, c='lightgreen', marker='s', label='Test data')

plt.xlabel('True values')
plt.ylabel('Predicted values')
plt.title("True value vs Predicted value")

plt.legend(loc='upper left')
plt.show()
```


True value vs Predicted value
Training data
Test data
Predicted values
True values
35
30
25
20
15
10
5
0
−5
0
5
10
15
20
25
30
35

In [ ]: