In [ ]:

```
%pip install statsmodels
```

```
Looking in indexes: https://pypi.tuna.tsinghua.edu.cn/simple
Collecting statsmodels
  Downloading https://pypi.tuna.tsinghua.edu.cn/packages/e9/33/1e9c80d6c8ce9aac7228e
155d098994536bf518891273638641d584b1a74/statsmodels-0.13.2-cp37-cp37m-win_amd64.whl
(9.0 MB)
     ---------------------------------------- 9.0/9.0 MB 6.3 MB/s eta 0:00:00
Collecting patsy>=0.5.2
  Downloading https://pypi.tuna.tsinghua.edu.cn/packages/87/7f/d37cd027c25145eeba92b
1a756976931c831803d92547c8637a3400c339f/patsy-0.5.2-py2.py3-none-any.whl (233 kB)
     ------------------------------------- 233.7/233.7 kB 14.0 MB/s eta 0:00:00
Requirement already satisfied: numpy>=1.17 in c:\users\xuyic\miniconda3\lib\site-pac
kages (from statsmodels) (1.21.6)
Requirement already satisfied: pandas>=0.25 in c:\users\xuyic\miniconda3\lib\site-pa
ckages (from statsmodels) (1.3.5)
Requirement already satisfied: scipy>=1.3 in c:\users\xuyic\miniconda3\lib\site-pack
ages (from statsmodels) (1.7.3)
Requirement already satisfied: packaging>=21.3 in c:\users\xuyic\miniconda3\lib\site
-packages (from statsmodels) (21.3)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in c:\users\xuyic\miniconda3
\lib\site-packages (from packaging>=21.3->statsmodels) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7.3 in c:\users\xuyic\miniconda3\l
ib\site-packages (from pandas>=0.25->statsmodels) (2.8.2)
Requirement already satisfied: pytz>=2017.3 in c:\users\xuyic\miniconda3\lib\site-pa
ckages (from pandas>=0.25->statsmodels) (2022.1)
Requirement already satisfied: six in c:\users\xuyic\miniconda3\lib\site-packages (f
rom patsy>=0.5.2->statsmodels) (1.16.0)
Installing collected packages: patsy, statsmodels
Successfully installed patsy-0.5.2 statsmodels-0.13.2
Note: you may need to restart the kernel to use updated packages.
```

In [33]:

```
# dataset url https://archive.ics.uci.edu/ml/machine-learning-databases/housing/
dataset_path = './housing/housing.data'
dataset_name = './housing/housing.names'

import os
print(os.path.exists(dataset_path),os.path.exists(dataset_name))
```

```
True True
```

In [34]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import sklearn
from sklearn import datasets
```

In [35]:

```
# turn to dataframe
df = pd.read_csv(dataset_path,header=None,sep='\s+')
df.columns = ['CRIM','ZN','INDUS','CHAS','NOX','RM','AGE','DIS','RAD','TAX','PTRATIC

# show the first 5 rows
df.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	296.0	15.3	396.90	4.98
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242.0	17.8	396.90	9.14
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242.0	17.8	392.83	4.03
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222.0	18.7	394.63	2.94
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222.0	18.7	396.90	5.33

In [36]:

```
# 将房价(变量14)相对于所有其他变量进行回归
Y = df['MEDV'].values
X = df.iloc[:,0:13].values

from sklearn.linear_model import LinearRegression
slr = LinearRegression()
slr.fit(X,Y)
y_pred = slr.predict(X)
```

In [37]:

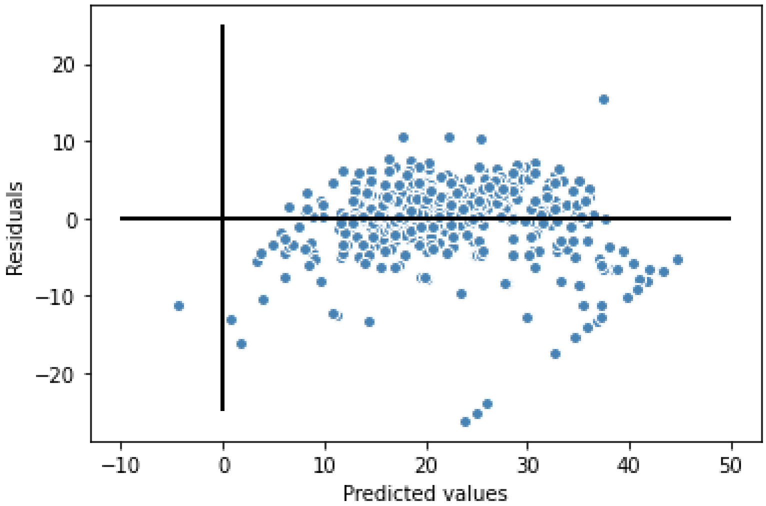
```
accuray = slr.score(X,Y)
print('R^2: %.3f' % accuray)

# 画出残差图
plt.scatter(y_pred, y_pred - Y, c='steelblue', marker='o', edgecolor='white', label=
# 根据库克距离的定义，画出一条水平线
plt.hlines(y=0, xmin=-10, xmax=50, lw=2, color='black')
# 根据杠杆效应的定义，画出一条垂直线
plt.vlines(x=0, ymin=-25, ymax=25, lw=2, color='black')
plt.xlabel('Predicted values')
plt.ylabel('Residuals')
```

```
R^2: 0.741
```

Out[37]:

```
Text(0, 0.5, 'Residuals')
```



In [38]:

```
# 移除所有怀疑为异常的点 残差在-20到10之间为正常值
df = df[(y_pred - Y > -20) & (y_pred - Y < 10)]
```

In [39]:

```
# 重新计算回归
Y = df['MEDV'].values
X = df.iloc[:,0:13].values
new_slr = LinearRegression()
new_slr.fit(X,Y)
new_y_pred = new_slr.predict(X)
```

```
new_accuray = new_slr.score(X,Y)
print('R^2: %.3f' % new_accuray)
```

```
R^2: 0.790
```

In [ ]:

```
# 对因变量应用Box-Cox变换
from scipy import stats
from scipy.stats import boxcox
from scipy.special import inv_boxcox
# 画出原始数据的直方图
plt.hist(Y, bins=50)

transformed_Y = boxcox(Y, 0.1)
```

In [ ]:

```
slr_2 = LinearRegression()
slr_2.fit(X,transformed_Y)
y_pred_2 = slr_2.predict(X)
accuray_2 = slr_2.score(X,transformed_Y)
print('R^2: %.3f' % accuray_2)

# 画出残差图
plt.scatter(y_pred_2, y_pred_2 - transformed_Y, c='steelblue', marker='o', edgecolor
# 根据库克距离的定义, 画出一条水平线
plt.hlines(y=0, xmin=-10, xmax=50, lw=2, color='black')
```

In [ ]:

```
# 绘制房价的拟合值相对于真实值的曲线
plt.scatter(Y, y_pred_2, c='steelblue', marker='o', edgecolor='white', label='Train
```

In [32]:

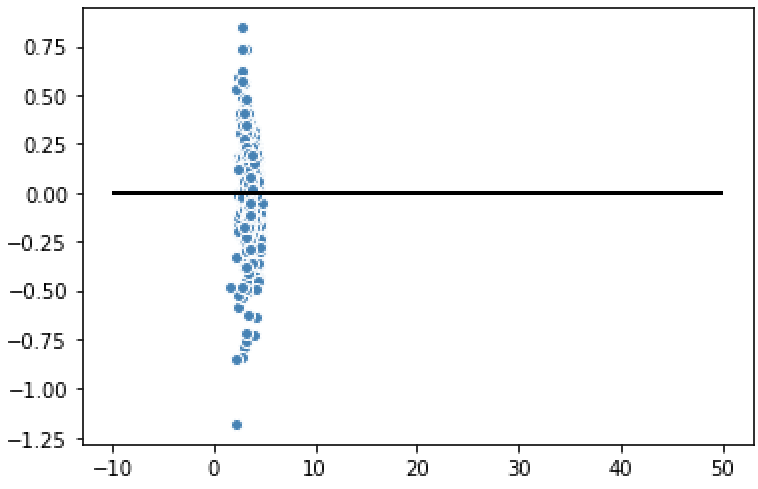
```
slr_2 = LinearRegression()
slr_2.fit(X,transformed_Y)
y_pred_2 = slr_2.predict(X)
accuray_2 = slr_2.score(X,transformed_Y)
print('R^2: %.3f' % accuray_2)

# 画出残差图
plt.scatter(y_pred_2, y_pred_2 - transformed_Y, c='steelblue', marker='o', edgecolor
# 根据库克距离的定义, 画出一条水平线
plt.hlines(y=0, xmin=-10, xmax=50, lw=2, color='black')
```

```
R^2: 0.818
```

Out[32]:

```
<matplotlib.collections.LineCollection at 0x2231643fe88>
```

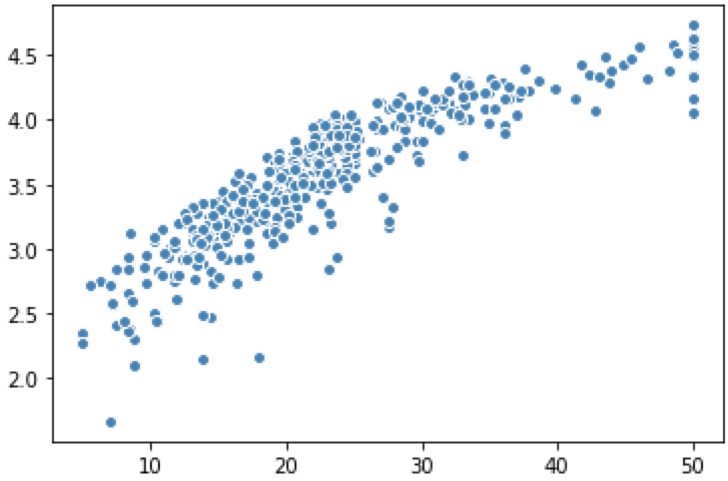


In [41]:

```
# 绘制房价的拟合值相对于真实值的曲线
plt.scatter(Y, y_pred_2, c='steelblue', marker='o', edgecolor='white', label='Train
```

Out[41]:

```
<matplotlib.collections.PathCollection at 0x223165cf5c8>
```



In [ ]: