Logistic 回归案例：Boston 房价

```
from sklearn.linear_model import LogisticRegression

from sklearn.cross_validation import train_test_split

from sklearn.metrics import classification_report

from sklearn.datasets import load_boston

boston = load_boston()

X = boston.data

y = boston.target

avg_price = np.average(y)

# 划分训练集和测试集
X_train,X_test,y_train,y_test = train_test_split(X,y)

# 将目标 y 改成类别型数据
# 训练集中超过均价的房子
high_price_idx = (y_train > avg_price)

# 训练集目标改成 1
y_train[high_price_idx] = 1

# 否则改成 0
y_train[np.logical_not(high_price_idx)] = 0

y_train = y_train.astype(np.int8)      # 1.0/0.0 转换成 1/0

# 对测试集做同样的事情
high_price_idx = (y_test > avg_price)

y_test[high_price_idx] = 1

y_test[np.logical_not(high_price_idx)] = 0

y_test = y_test.astype(np.int8)

# 构造 logistic 回归模型
model = LogisticRegression()
```

```
# 在训练集上拟合
model.fit(X_train,y_train)

# 在测试集上预测
y_predicted = model.predict(X_test)

y_predicted
array([1, 1, 0, 0, 0, 0, 1, 1, ... 0, 0, 1], dtype=int8)

# 评估分类效果
model.score(X_test,y_test)
0.8740157480314961

print classification_report(y_test,y_predicted)

            precision    recall  f1-score   support

          0       0.95      0.90      0.92        81
          1       0.84      0.91      0.87        46

avg / total       0.91      0.91      0.91       127


# 空模型：每次都猜 0 有 59%的准确率
pd.Series(y_train).value_counts(normalize=True)
0    0.593668
1    0.406332


# 利用虚拟变量将类别型特征转换成数值特征

# 先将距离特征离散化成类别型特征
def f(x):
    if x<=3:
        return 'near'
    elif 3<x and x<=6:
        return 'medium'
    else:
        return 'far'

X = pd.DataFrame(boston.data)
X[13] = X[7].apply(f)   # 从第 8 个特征（距离）新建第 14 个特征
X[13]
```

```
0     medium
1     medium
2     medium
3        far
4        far
       ...
502     near
503     near
504     near
505     near
Name: 13, dtype: object

# 用虚拟变量转换新特征
dis_dummy = pd.get_dummies(X[13],prefix='dis')

dis_dummy
    dis_far  dis_medium  dis_near
0       0.0        1.0      0.0
1       0.0        1.0      0.0
2       0.0        1.0      0.0
3       1.0        0.0      0.0
4       1.0        0.0      0.0
...
504     0.0        0.0      1.0
505     0.0        0.0      1.0

[506 rows x 3 columns]

# 利用虚拟变量和师生比来预测房价
# 组合这些特征构成数据集
X1 = pd.concat([X[[10]],dis_dummy],axis=1)
X1
In [343]: X1
Out[343]:
      10  dis_far  dis_medium  dis_near
0   15.3     0.0        1.0      0.0
1   17.8     0.0        1.0      0.0
2   17.8     0.0        1.0      0.0
3   18.7     1.0        0.0      0.0
4   18.7     1.0        0.0      0.0
5   18.7     1.0        0.0      0.0
......
502 21.0     0.0        0.0      1.0
503 21.0     0.0        0.0      1.0
```

```
504  21.0     0.0       0.0      1.0
505  21.0     0.0       0.0      1.0

[506 rows x 4 columns]

# 训练预测
y = boston.target
avg_price = np.average(y)

high_price_idx = (y > avg_price)

y[high_price_idx] = 1

y[np.logical_not(high_price_idx)] = 0

y = y.astype(np.int8)

mod = LogisticRegression()

mod.fit(X1,y)

mod.score(X1,y)
0.7371541501976284
```