线性回归(LR)&局部加权线性回归(LWLR)

作者：刘伟杰 日期：2015-12-03
参考：
[1]《机器学习实战》 Peter Harrington
[2] scikit-learn官方手册

1. 理论

1. 概述：

   线性回归(Linear Regression)

   > 寻找一个线性函数的参数k使得与训练集的拟合效果最好。

   局部加权线性回归(Locally Weighted Linear Regression)

   > 输入一个新的样本时，对训练集做如下权重分配（距离新样本近的权重大，远的权重小），按照配置权

2. 线性回归：

   线性回归模型:

   $$y = K = k_1 x^{(1)} + k_2 x^{(2)} + ... + k_n x^{(n)}$$

   惩罚函数(风险函数、目标函数):

   $$e = \sum_{i=1}^{n} (y_i - K \cdot X_i)^2$$

   训练过程：用最小二乘法，可以求得：

   $$K = (X^T X)^{-1} Y$$

3. 局部加权线性回归：

   给训练集的点设置权重，用矩阵W给每个数据点赋予权重，则训练结果为:

   $$K = (X^T W X)^{-1} W^T Y$$

一般选用高斯函数来构造W，这样越靠近输入点x的训练集点xi的权重就越大。

$$w(i,i) = e^{\frac{|x_i - x|}{-2k^2}}$$

k为高斯函数的标准差，k越大，则权重的分配就越平均，越靠近普通线性回归。k越小，权重分配就越集中在输入点周围，容易造成过拟合。一般k在0.01~1之间调整，用来防止欠拟合与过拟合，也可以用交叉验证选取最优的k。

2. 实现

1. 我的实现：

   线性回归：

```
def stand_regress(data_list, label_list):
    X = mat(data_list); Y = mat(label_list)
    XTX = X.T*X
    if linalg.det(XTX) == 0.0:
        print "this matrix is singular, cannot to inverse!"
        return None
    else:
        w_estimate = (X.T*X).I * X.T * Y.T
    return w_estimate.T.tolist()[0]
```

   加权局部线性回归：

```
def locally_weighted_linear_regress(test_point, data_list, label_list, k=1.0):
    X_mat = mat(data_list); Y_mat = mat(label_list); test_point = mat(test_point)
    # create weighted_mat
    m = shape(X_mat)[0]
    weighted_mat = mat(eye(m))
    for i in range(m):
        dis = X_mat[i] - test_point
        diff = math.sqrt((dis * dis.T)[0,0])
        weighted_mat[i,i] = exp(diff/(-2.0*k**2))
    # calculate w
    XTX = X_mat.T * (weighted_mat * X_mat)
    if linalg.det(XTX) == 0.0:
        print "this matrix is singular, cannot to inverse!"
        return None
    else:
        w_estimate = (XTX).I * X_mat.T * weighted_mat * Y_mat.T
    # return regress resultt
    label_estimate = (test_point * w_estimate)[0,0]
    return label_estimate
```

2. scikit-learn:

   线性回归：

```
#Import Library
#Import other necessary libraries like pandas, numpy...
from sklearn import linear_model
#Load Train and Test datasets
#Identify feature and response variable(s) and values must be numeric and numpy ar
x_train=input_variables_values_training_datasets
y_train=target_variables_values_training_datasets
x_test=input_variables_values_test_datasets
# Create linear regression object
linear = linear_model.LinearRegression()
# Train the model using the training sets and check score
linear.fit(x_train, y_train)
linear.score(x_train, y_train)
#Equation coefficient and Intercept
print('Coefficient: n', linear.coef_)
print('Intercept: n', linear.intercept_)
#Predict Output
predicted= linear.predict(x_test)
```

   加权局部线性回归：

```
目前我还没有在scikit-learn库中找到该方法，如果你发现了，请告诉，非常感谢。
```