**用指标对决策树和逻辑回归模型进行评价**

def change\_one\_hot\_label(X): # 用one\_hot编码转换

T = np.zeros((X.size, 3))

for idx, row in enumerate(T):

row[X[idx]] = 1

return T

from sklearn import metrics

import numpy as np

print('决策树Accracy：',accuracy\_score(y\_predict, y\_test))

y\_score = tree.predict\_proba(X\_test)

y\_one\_hot = change\_one\_hot\_label(y\_test)

fpr, tpr, thresholds = metrics.roc\_curve(y\_one\_hot.ravel(),y\_score.ravel())

auc = metrics.auc(fpr, tpr)

print('决策树AUC：', auc)

print('逻辑回归Accracy：',accuracy\_score(y\_predict2, y\_test))

y\_score2 = log.predict\_proba(X\_test)

y\_one\_hot2 = change\_one\_hot\_label(y\_test)

fpr2, tpr2, thresholds2 = metrics.roc\_curve(y\_one\_hot2.ravel(),y\_score2.ravel())

auc2 = metrics.auc(fpr2, tpr2)

print('逻辑回归AUC：', auc2)

#画图

import matplotlib.pyplot as plt

plt.plot(fpr, tpr, c = 'r', label = u'tree\_auc=%.3f' % auc)

plt.plot(fpr2, tpr2, c = 'b', label = u'log\_auc=%.3f' % auc2)

plt.plot((0, 1), (0, 1), c = 'k', ls = '--')

plt.xlim((-0.01, 1.01))

plt.ylim((-0.01, 1.01))

plt.xticks(np.arange(0, 1.1, 0.1))

plt.yticks(np.arange(0, 1.1, 0.1))

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.grid(b=True, ls=':')

plt.legend(loc='lower right')

plt.show()

**运行（测试）过程及结果：**



