* 乳腺癌数据集的文件名： wdbc.data， wdbc.names， 网址：<http://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/>
* 将数据集60%作为训练集，40%作为测试集，使用scikit-learn中SVM分类器对训练集进行训练，并利用测试数据集测试结果
* 调节svm中的参数，做出测试报告

例子1:

#导入svm和数据集

from sklearn import svm,datasets

#调用SVC(),SVC(C=1.0, cache\_size=200, class\_weight=None,

decision\_function\_shape='ovr', degree=3, gamma='auto', kernel='rbf')

clf = svm.SVC()

#载入鸢尾花数据集

iris = datasets.load\_iris()

X = iris.data

y = iris.target

#fit()训练

clf.fit(X,y)

#predict()预测

pre\_y = clf.predict(X[5:10])

print(pre\_y)

print(y[5:10])

#导入numpy

import numpy as np

test = np.array([[5.1,2.9,1.8,3.6]])

#对test进行预测

test\_y = clf.predict(test)

print(test\_y)

例子2:

import numpy as np

import matplotlib.pyplot as plt

from sklearn import svm, datasets

# 导入数据集

iris = datasets.load\_iris()

X = iris.data[:, :2] # 只取前两维特征

y = iris.target

h = .02 # 网格中的步长

# 创建支持向量机实例，并拟合出数据

C = 1.0 # SVM正则化参数

svc = svm.SVC(kernel='linear', C=C).fit(X, y) # 线性核

rbf\_svc = svm.SVC(kernel='rbf', gamma=0.7, C=C).fit(X, y) # 径向基核

poly\_svc = svm.SVC(kernel='poly', degree=3, C=C).fit(X, y) # 多项式核

lin\_svc = svm.LinearSVC(C=C).fit(X, y) #线性核

# 创建网格，以绘制图像

x\_min, x\_max = X[:, 0].min() - 1, X[:, 0].max() + 1

y\_min, y\_max = X[:, 1].min() - 1, X[:, 1].max() + 1

xx, yy = np.meshgrid(np.arange(x\_min, x\_max, h),

np.arange(y\_min, y\_max, h))

# 图的标题

titles = ['SVC with linear kernel',

'LinearSVC (linear kernel)',

'SVC with RBF kernel',

'SVC with polynomial (degree 3) kernel']

for i, clf in enumerate((svc, lin\_svc, rbf\_svc, poly\_svc)):

# 绘出决策边界，不同的区域分配不同的颜色

plt.subplot(2, 2, i + 1) # 创建一个2行2列的图，并以第i个图为当前图

plt.subplots\_adjust(wspace=0.4, hspace=0.4) # 设置子图间隔

Z = clf.predict(np.c\_[xx.ravel(), yy.ravel()]) #将xx和yy中的元素组成一对对坐标，作为支持向量机的输入，返回一个array

# 把分类结果绘制出来

Z = Z.reshape(xx.shape) #(220, 280)

plt.contourf(xx, yy, Z, cmap=plt.cm.Paired, alpha=0.8) #使用等高线的函数将不同的区域绘制出来

# 将训练数据以离散点的形式绘制出来

plt.scatter(X[:, 0], X[:, 1], c=y, cmap=plt.cm.Paired)

plt.xlabel('Sepal length')

plt.ylabel('Sepal width')

plt.xlim(xx.min(), xx.max())

plt.ylim(yy.min(), yy.max())

plt.xticks(())

plt.yticks(())

plt.title(titles[i])

plt.show()