机器学习作业二

计算机科学与技术卓越班 蔡嘉轩 20185670

1. 证明：

设x_0在超平面(ω,b)的投影设为x_1。

$\because x_0$在超平面(ω,b)上，

$\therefore w \cdot x_0+b=0\ (*)$

$\because \overrightarrow{x_1x_0}$ // 超平面的法向量ω

$$\therefore |\omega\overrightarrow{x_1x_0}|=|\omega||\overrightarrow{x_1x_0}|=\sqrt{(\omega^1)^2+\ldots+(\omega^N)^2}d=\|\omega\|d$$

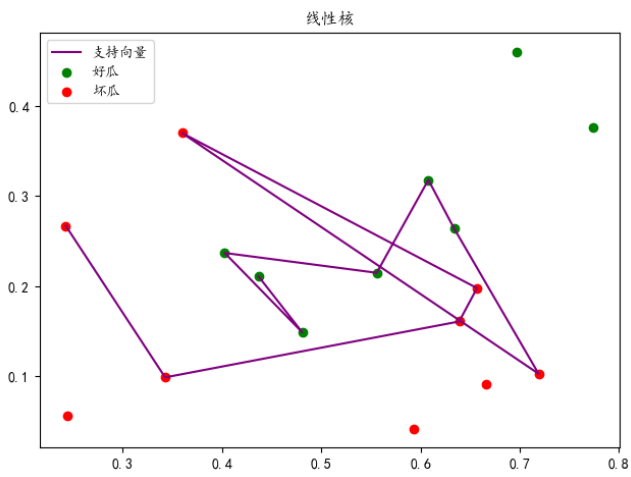
$$\begin{aligned}\omega\overrightarrow{x_1x_0}&=\omega^1(x_0^1-x_1^1)+\omega^2(x_0^2-x_1^2)+\ldots+\omega^N(x_0^N-x_1^N)\\&=\omega^1x_0^1+\omega^2x_0^2+\ldots+\omega^Nx_0^N-(\omega^1x_1^1+\omega^2x_1^2+\ldots+\omega^Nx_N^N)\\&=\omega^1x_0^1+\omega^2x_0^2+\ldots+\omega^Nx_0^N+b\ \text{(由于*)}\end{aligned}$$

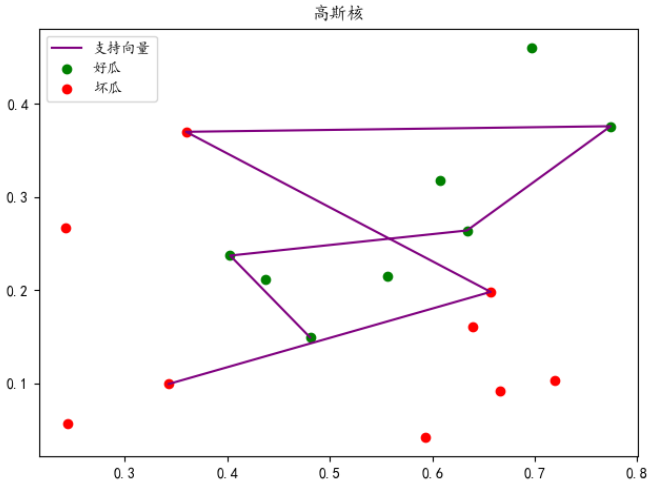
$$\therefore \|\omega\|d=|\omega^1x_0^1+\omega^2x_0^2+\ldots+\omega^Nx_0^N|=|\omega x_0+b|$$

$$\therefore d=\frac{|\omega x_0+b|}{\|\omega\|}\quad QED$$

2. 调用 SVM 库，编写如下程序并运行，查看训练的 SVM 的效果以及各自的支持向量：

```
Linear
Predict:  [ 1  1  1  1  1  1 -1  1 -1  1 -1 -1 -1 -1  1 -1 -1]
Actual:  [ 1  1  1  1  1  1  1  1 -1 -1 -1 -1 -1 -1 -1 -1 -1]
SupportingVerctor:  [ 9 11 12 13 14 16  2  3  4  5  6  7]
Gauss
Predict:  [ 1  1  1  1  1  1  1  1 -1 -1 -1 -1 -1 -1 -1 -1 -1]
Actual:  [ 1  1  1  1  1  1  1  1 -1 -1 -1 -1 -1 -1 -1 -1 -1]
SupportingVerctor:  [11 13 14  1  2  5  6]
```






```python
from sklearn import svm
import pandas as pd
from matplotlib import pyplot as plt

plt.rcParams['font.sans-serif'] = ['KaiTi', 'SimHei', 'FangSong']


def visualize(clf, x, y, kernel):
    p = y == 1
    n = y == -1
    ax = plt.subplot()
    ax.scatter(x[p, 0], x[p, 1], label='好瓜', color='green')
    ax.scatter(x[n, 0], x[n, 1], label='坏瓜', color='red')
    plt.plot(x[clf.support_, 0], x[clf.support_, 1], label='支持向量', color='purple')  # 支持向量图
    ax.legend()  # 标签
    ax.set_title(r'{}'.format(kernel))
    plt.show()


if __name__ == "__main__":
    path = 'watermelon_3a.csv'  # 数据集文件路径
    data = pd.read_table(path, delimiter=',', dtype=float)  # 使用panda读取数据集
    X = data.iloc[:, [1, 2]].values  # 提取第一二列
    labels = data.iloc[:, 3].values  # 提取第三列
    labels[labels == 0] = -1  # 把0变为-1

    C = 500  # 测试后选取SVC的C为500，使得误差尽量小且不易发生过拟合
    # 查看线性核支持向量
    print()
    clf_linear = svm.SVC(C=C, kernel='linear')
    clf_linear.fit(X, labels.astype(int))
    pred_linear = clf_linear.predict(X)
    print('Linear')
    print('Predict: ', pred_linear)
    print('Actual: ', labels.astype(int))
    print('SupportingVerctor: ', clf_linear.support_)

    # 查看高斯核支持向量
    clf_rbf = svm.SVC(C=C)
    clf_rbf.fit(X, labels.astype(int))
    pred_rbf = clf_rbf.predict(X)
    print('Gauss')
    print('Predict: ', pred_rbf)
    print('Actual: ', labels.astype(int))
    print('SupportingVerctor: ', clf_rbf.support_)

    # 可视化
    visualize(clf_rbf, X, labels, '高斯核')
    visualize(clf_linear, X, labels, '线性核')
```

3. 等价条件：

(1)线性判别分析和线性核支持向量机能将数据以同类样例间低方差和不同

样例中心之间以大间隔投射到一条直线上，但是如果样本线性不可分，那么就不能有效进行。

(2)线性判别分析与线性核支持向量机均可用于样本最优划分超平面的求解，即法向量 w,两者相等时有

$$w=S_w^{-1}(\mu_0-\mu_1)=\sum_{i=1}^{m}\alpha_i y_i x_i$$

即两者的超平面相同，此时两者等效。

综上所述可知，线性判别分析与线性核支持向量机的等价条件为：(1)数据是线性可分的；(2)线性判别分析求出的投影面与线性核支持向量机求出的分隔超平面垂直。

4. 完整 KTT 条件：

根据 KKT 条件的定义，每一个拉格朗日函数中的带有拉格朗日乘子的项对应的 KKT 条件有三个式子。

在原式中，带有拉格朗日乘子的共有四项，所以完整的 KKT 条件应该共有 12 个式子。

非等式约束写成拉格朗日乘子式，取最优解要满足两个条件：(1)拉格朗日乘子式对所有非拉格朗日参数的一阶偏导为 0；(2)非等式约束对应的拉格朗日项，要么非等式的等号成立，要么对应的拉格朗日参数为 0。

综上，分别对每一项列出式子，得出完整的 KKT 条件为：

$$\xi_i \geqslant 0$$

$$\hat{\xi}_i \geqslant 0$$

$$f(x_i) - y_i - \epsilon - \xi_i \leqslant 0$$

$$y_i - f(x_i) - y_i - \epsilon - \hat{\xi}_i \leqslant 0$$

$$\mu_i \geqslant 0$$

$$\hat{\mu}_i \geqslant 0$$

$$\alpha_i \geqslant 0$$

$$\hat{\alpha}_i \geqslant 0$$

$$\mu_i \xi_i = 0$$

$$\hat{\mu}_i \hat{\xi}_i = 0$$

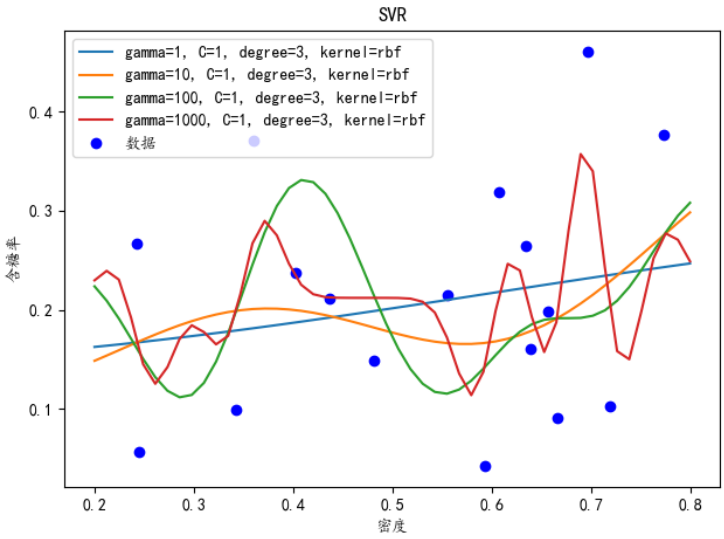
$$\alpha_i (f(x_i) - y_i - \epsilon - \xi_i) = 0$$

$$\hat{\alpha}_i (y_i - f(x_i) - y_i - \epsilon - \hat{\xi}_i) = 0$$

5. 编写程序，以密度为输入，含糖量为输出。调整不同的参数训练 SVR 并查看效果与参数

(1) 固定 C=1，调整 Gamma 的值：

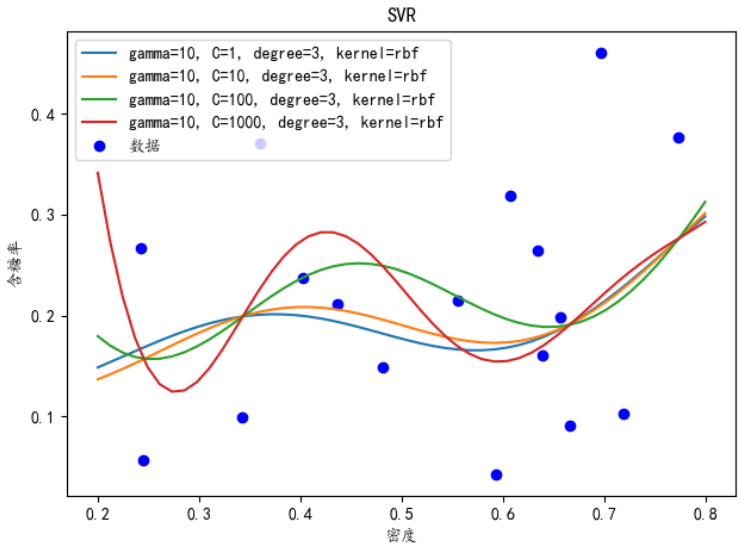
```
accuracy     = 0.035749609530696835
svr param = {'C': 1, 'cache_size': 200, 'coef0': 0.0, 'degree': 3, 'epsilon': 0.1, 'gamma': 1, 'kernel': 'rbf', 'max_iter': -1, 'shrinking': True, 'tol': 0.001, 'verbose': False}
accuracy     = 0.08813971528860708
svr param = {'C': 1, 'cache_size': 200, 'coef0': 0.0, 'degree': 3, 'epsilon': 0.1, 'gamma': 10, 'kernel': 'rbf', 'max_iter': -1, 'shrinking': True, 'tol': 0.001, 'verbose': False}
accuracy     = 0.028274317244996294
svr param = {'C': 1, 'cache_size': 200, 'coef0': 0.0, 'degree': 3, 'epsilon': 0.1, 'gamma': 100, 'kernel': 'rbf', 'max_iter': -1, 'shrinking': True, 'tol': 0.001, 'verbose': False}
accuracy     = 0.5144484413803242
svr param = {'C': 1, 'cache_size': 200, 'coef0': 0.0, 'degree': 3, 'epsilon': 0.1, 'gamma': 1000, 'kernel': 'rbf', 'max_iter': -1, 'shrinking': True, 'tol': 0.001, 'verbose': False}
```




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

plt.rcParams['font.sans-serif'] = ['KaiTi', 'SimHei', 'FangSong']

if __name__ == "__main__":
    path = 'watermelon_3a.csv'  # 数据路径
    data = pd.read_table(path, delimiter=',', dtype=float)  # 使用panda读取数据集
    X = data.iloc[:, [1]].values  # 提取第一列
    y = data.iloc[:, 2].values  # 提取第二列

    gamma = 1  # 默认
    c = 1  # 默认
    for gamma in [1, 10, 100, 1000]:
        svr = svm.SVR(gamma=gamma, C=c, kernel='rbf')  # 训练SVR
        svr.fit(X, y)
        print("score     = {}".format(svr.score(X, y)))
        print("svr param = {}".format(svr.get_params()))
        ax = plt.subplot()  # 可视化

        ax.plot(np.linspace(0.2, 0.8), svr.predict(np.linspace(0.2, 0.8).reshape(-1, 1)),
                label='gamma={}, C={}, degree={}, kernel={}'.format(svr.gamma, svr.C, svr.degree, svr.kernel))
    ax.scatter(X, y, color='blue', label='数据')
    ax.legend()
    ax.set_xlabel('密度')
    ax.set_ylabel('含糖率')
    plt.title("SVR")
    plt.show()
```

(2) 固定 gamma=1，调整 C 的值：

```
accuracy     = 0.09465888876886046
svr param = {'C': 10, 'cache_size': 200, 'coef0': 0.0, 'degree': 3, 'epsilon': 0.1, 'gamma': 10, 'kernel': 'rbf', 'max_iter': -1, 'shrinking': True, 'tol': 0.001, 'verbose': False}
accuracy     = 0.06581533382066762
svr param = {'C': 100, 'cache_size': 200, 'coef0': 0.0, 'degree': 3, 'epsilon': 0.1, 'gamma': 10, 'kernel': 'rbf', 'max_iter': -1, 'shrinking': True, 'tol': 0.001, 'verbose': False}
accuracy     = 0.06289739284393159
svr param = {'C': 1000, 'cache_size': 200, 'coef0': 0.0, 'degree': 3, 'epsilon': 0.1, 'gamma': 10, 'kernel': 'rbf', 'max_iter': -1, 'shrinking': True, 'tol': 0.001, 'verbose': False}
```




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

plt.rcParams['font.sans-serif'] = ['KaiTi', 'SimHei', 'FangSong']

if __name__ == "__main__":
    path = 'watermelon_3a.csv'  # 数据路径
    data = pd.read_table(path, delimiter=',', dtype=float)  # 使用panda读取数据集
    X = data.iloc[:, [1]].values  # 提取第一列
    y = data.iloc[:, 2].values  # 提取第二列

    gamma = 10  # 默认
    c = 1  # 默认
    for c in [1, 10, 100, 1000]:
        svr = svm.SVR(gamma=gamma, C=c, kernel='rbf')  # 训练SVR
        svr.fit(X, y)
        print("score     = {}".format(svr.score(X, y)))
        print("svr param = {}".format(svr.get_params()))
        ax = plt.subplot()  # 可视化

        ax.plot(np.linspace(0.2, 0.8), svr.predict(np.linspace(0.2, 0.8).reshape(-1, 1)),
                label='gamma={}, C={}, degree={}, kernel={}'.format(svr.gamma, svr.C, svr.degree, svr.kernel))
    ax.scatter(X, y, color='blue', label='数据')
    ax.legend()
    ax.set_xlabel('密度')
    ax.set_ylabel('含糖率')
    plt.title("SVR")
    plt.show()
```