EDU
CSDN学院 IT实战派

SVM模型

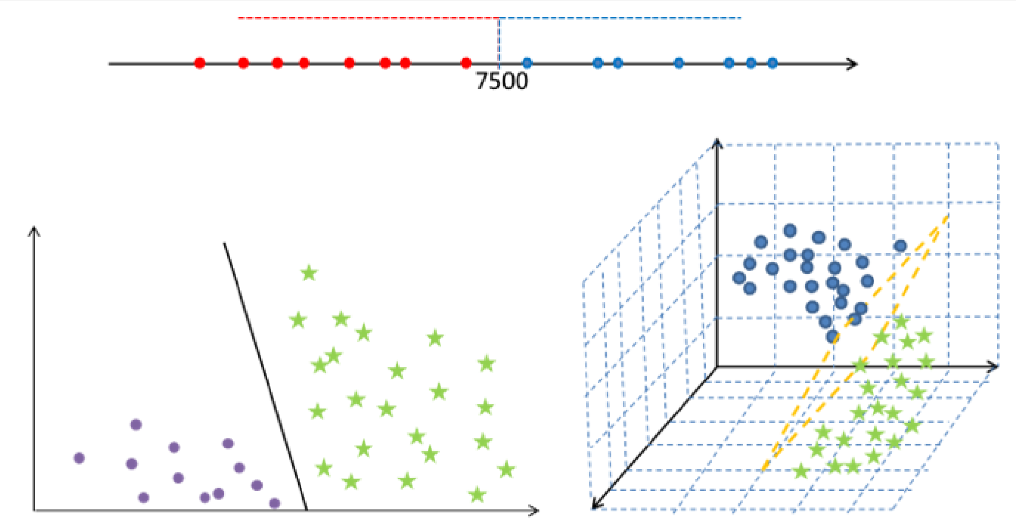
讲师：刘顺祥

课程目标

1. 理解SVM模型的思想和理论
2. 熟悉线性可分的SVM
3. 理解非线性可分的SVM
4. 掌握SVM的实操

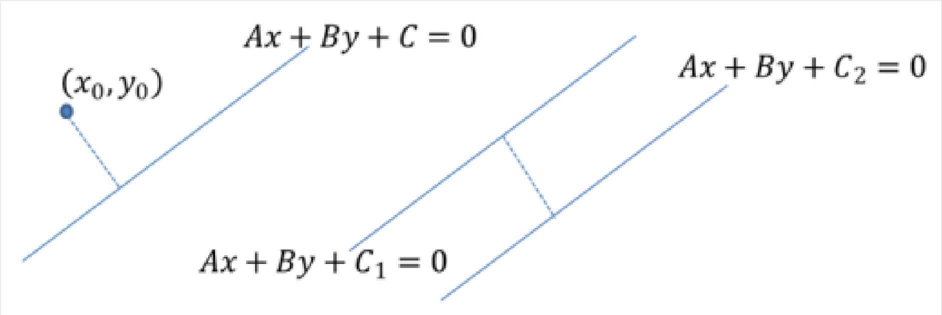
SVM模型

模型介绍



超平面的理解：在一维空间中，如需将数据切分为两段，只需要一个点即可；在二维空间中，对于线性可分的样本点，将其切分为两类，只需一条直线即可；在三维空间中，将样本点切分开来，就需要一个平面。

距离计算



点到线的距离：

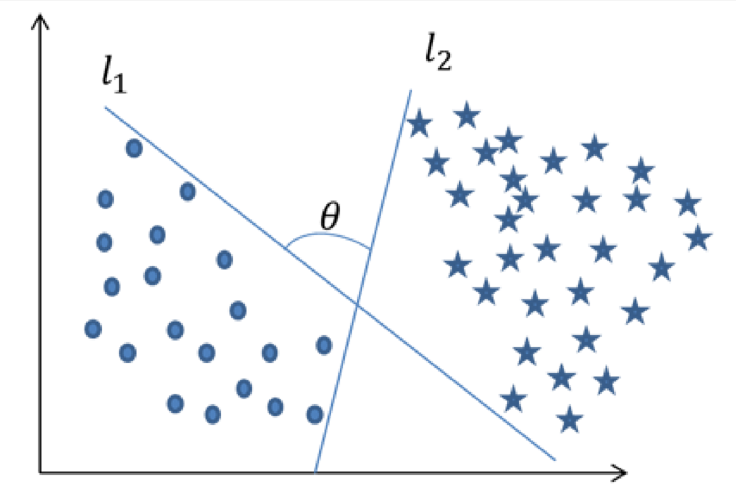
$$d=\frac{|Ax_0+By_0+C|}{\sqrt{A^2+B^2}}$$

平行线间的距离：

$$d=\frac{|C_1-C_2|}{\sqrt{A^2+B^2}}$$

EDU CSDN学院 IT实战派

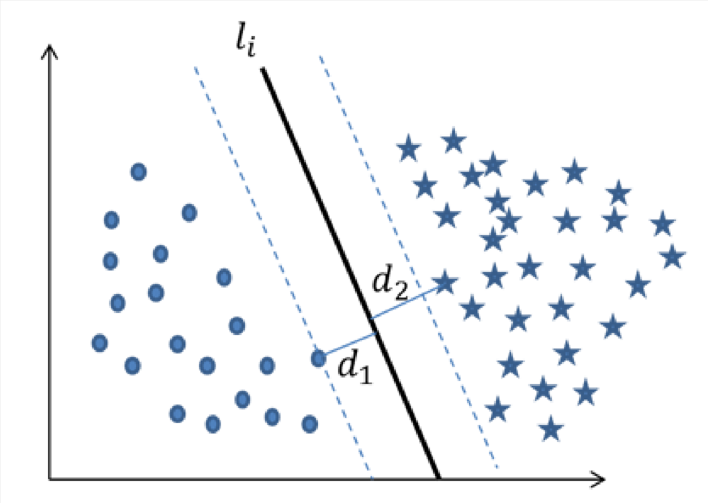
思想介绍



图中绘制了两条分割直线，利用这两条直线，可以方便地将样本点所属的类别判断出来。虽然从直观上来看这两条分割线都没有问题，但是哪一条直线的分类效果更佳呢（训练样本点的分类效果一致，并不代表测试样本点的分类效果也一样）？甚至于在直线l_1和l_2之间还存在无数多个分割直线，那么在这么多的分割线中是否存在一条最优的“超平面”呢?

EDU CSDN学院 IT实战派

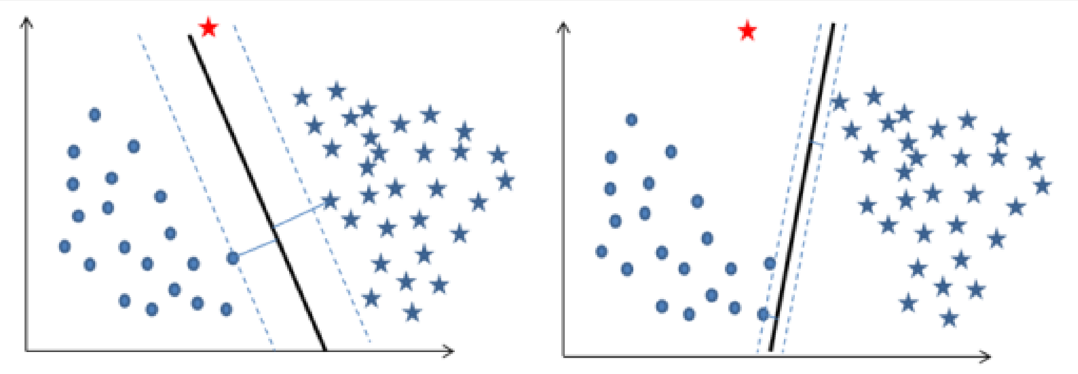
思想介绍



假设直线l_i是l_1和l_2之间的某条直线(分割面)，为了能够寻找到最优的分割面l_i，需要做三件事，首先计算两个类别中的样本点到直线l_i的距离；然后从两组距离中各挑选出一个最短的（如图中所示的距离d_1和d_2），继续比较d_1和d_2，再选出最短的距离（如图中的d_1），并以该距离构造“分割带”（如图中经平移后的两条虚线）；最后利用无穷多个分割直线l_i，构造无穷多个“分割带”，并从这些“分割带”中挑选出带宽最大的l_i。

SVM模型

分隔带



“分割带”代表了模型划分样本点的能力或可信度，“分割带”越宽，说明模型能够将样本点划分得越清晰，进而保证模型泛化能力越强，分类的可信度越高；反之，“分割带”越窄，说明模型的准确率越容易受到异常点的影响，进而理解为模型的预测能力越弱，分类的可信度越低。

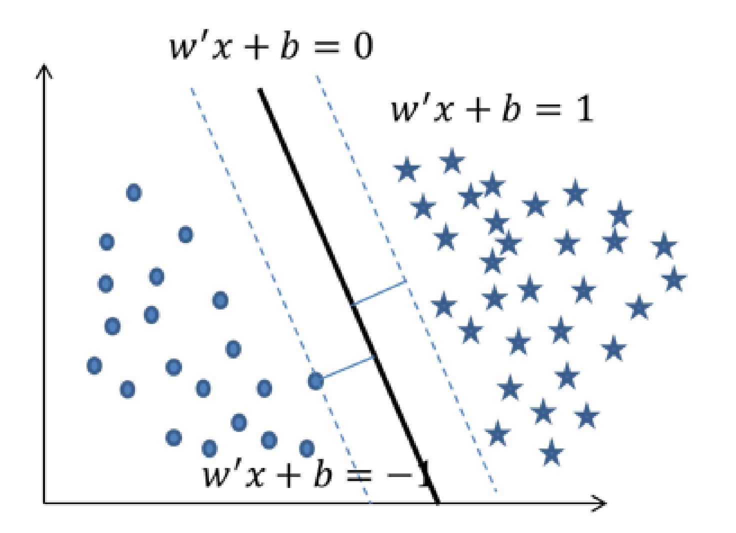
目标函数

$$J(w,b,i) = arg_{w,b}maxmin(d_i)$$

其中，d_i表示样本点i到某条固定分割面的距离；$min(d_i)$表示所有样本点与某个分割面之间距离的最小值；$arg_{w,b}maxmin(d_i)$表示从所有的分割面中寻找“分割带”最宽的“超平面”；其中w和b代表线性分割面的参数。

EDU
CSDN学院 IT实战派

函数间隔



将图中五角星所代表的正例样本用1表示，将实心圆所代表的负例样本用-1表示；实体加粗直线表示某条分割面；两条虚线分别表示因变量y取值为+1和-1时的情况，它们与分割面平行。

不管是五角星代表的样本点，还是实心圆代表的样本点，这些点均落在两条虚线以及虚线之外，则说明这些点带入到方程$w'x+b$所得的绝对值一定大于等于1。

进而可以说明如果点对应的取值越小于-1，该样本为负例的可能性越高；点对应的取值越大于+1，样本为正例的可能性越高。

函数间隔

$$\hat{\gamma}_i = y_i \times (w'x_i + b)$$

其中，y_i表示样本点所属的类别，用+1和-1表示。当$w'x_i + b$计算的值小于等于-1时，根据分割面可以将样本点x_i对应的y_i预测为-1；当$w'x_i + b$计算的值大于等于+1时，分割面会将样本点x_i对应的y_i预测为+1。故利用如上的乘积公式可以得到线性可分的SVM所对应的函数间隔满足$\hat{\gamma}_i \geq 1$的条件。

几何间隔

$$\gamma_i = \frac{\hat{\gamma}_i}{\|w\|} = \frac{y_i \times (w'x_i + b)}{\|w\|} = \frac{|w'x_i + b|}{\|w\|} = d_i$$

当分割面中的参数w和b同比例增加时，所对应的$\hat{\gamma}_i$值也会同比例增加，但这样的增加对分割面$w'x + b = 0$来说却丝毫没有影响。

所以，为了避免这样的问题，需要对函数间隔做约束，常见的约束为单位化处理。

线性可分的SVM模型

目标函数

$$
\begin{aligned}
J(w,b,i) &= arg_{w,b} maxmin(d_i) \\
&= arg_{w,b} maxmin \frac{y_i \times (w' x_i + b)}{\|w\|} \\
&= arg_{w,b} max \frac{1}{\|w\|} min\left(y_i \times (w' x_i + b)\right) \\
&= arg_{w,b} max \frac{1}{\|w\|} min(\hat{y}_i)
\end{aligned}
$$

目标函数的等价转换

线性可分的SVM所对应的函数间隔满足$\hat{\gamma}_i \geq 1$的条件，故$min(\hat{\gamma}_i)$就等于1。所以，可以将目标函数$J(w,b,i)$等价为如下的表达式：

$$\begin{cases} max \dfrac{1}{\|w\|} \\ s.t. \quad y_i \times (w'x_i + b) \geq 1 \end{cases}$$

由于最大化$\frac{1}{\|w\|}$与最小化$\frac{1}{2}\|w\|^2$是等价的，故可以将上面的表达式重新表示为：

$$\begin{cases} min \dfrac{1}{2} \|w\|^2 \\ s.t. \quad y_i \times (w'x_i + b) \geq 1 \end{cases}$$

拉格朗日乘子法

假设存在一个需要最小化的目标函数$f(x)$，并且该目标函数同时受到$g(x) \le 0$的约束。如需得到最优化的解，则需要利用拉格朗日对偶性将原始的最优化问题转换为对偶问题，即：

$$
\begin{aligned}
min(f(x)) &= min_x max_\lambda (L(x,\lambda)) \\
&= min_x max_\lambda \left(f(x) + \sum_{i=1}^{k} \lambda_i g_i(x) \right) \\
&= max_\lambda min_x \left(f(x) + \sum_{i=1}^{k} \lambda_i g_i(x) \right)
\end{aligned}
$$

其中，$f(x) + \sum_{i=1}^{k} \lambda_i g_i(x)$为拉格朗日函数；$\lambda_i$即为拉格朗日乘子，且$\lambda_i > 0$。

线性可分的SVM模型

基于拉格朗日乘子法的目标函数

$$
\begin{aligned}
min\frac{1}{2}\|w\|^2 &= max_{\alpha}min_{w,b}\left(L(w,b,\alpha_i)\right)\\
&= max_{\alpha}min_{w,b}\left(\frac{1}{2}\|w\|^2+\sum_{i=1}^{n}\alpha_i\left(1-y_i\times(w'x_i+b)\right)\right)\\
&= max_{\alpha}min_{w,b}\left(\frac{1}{2}\|w\|^2-\sum_{i=1}^{n}\alpha_i y_i\times(w'x_i+b)+\sum_{i=1}^{n}\alpha_i\right)
\end{aligned}
$$

目标函数的求解

- 求偏导，令导函数为0

$$
\begin{cases}
\dfrac{\partial L(w,b,\alpha)}{\partial w} = w - \sum_{i=1}^{n} \alpha_i y_i x_i = 0 \\
\dfrac{\partial L(w,b,\alpha)}{\partial b} = \sum_{i=1}^{n} \alpha_i y_i = 0
\end{cases}
$$

目标函数的求解

- 将导函数反代之目标函数

$$
\begin{aligned}
min\frac{1}{2}\|w\|^2 &= max_\alpha\left(\frac{1}{2}\|w\|^2+\sum_{i=1}^{n}\alpha_i\left(1-y_i\times(w'x_i+b)\right)\right)\\
&= max_\alpha\left(\frac{1}{2}w'w+\sum_{i=1}^{n}\alpha_i-\sum_{i=1}^{n}\alpha_i y_i w' x_i-\sum_{i=1}^{n}\alpha_i y_i b\right)\\
&= max_\alpha\left(\frac{1}{2}w'\sum_{i=1}^{n}\alpha_i y_i x_i+\sum_{i=1}^{n}\alpha_i-w'\sum_{i=1}^{n}\alpha_i y_i x_i-b\sum_{i=1}^{n}\alpha_i y_i\right)\\
&= max_\alpha\left(-\frac{1}{2}w'\sum_{i=1}^{n}\alpha_i y_i x_i+\sum_{i=1}^{n}\alpha_i-0\right)\\
&= max_\alpha\left(-\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j(x_i\cdot x_j)+\sum_{i=1}^{n}\alpha_i-0\right)
\end{aligned}
$$

目标函数的求解

$$\begin{cases} min_{\alpha}\left(\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j(x_i \cdot x_j) - \sum_{i=1}^{n}\alpha_i\right) \\ s.t.\ \ \sum_{i=1}^{n}\alpha_i y_i = 0 \\ \alpha_i \geq 0 \end{cases}$$

其中，$(x_i \cdot x_j)$表示两个样本点的内积。最终根据已知样本点(x_i, y_i)计算$\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j(x_i \cdot x_j) - \sum_{i=1}^{n}\alpha_i$的极小值，并利用拉格朗日乘子$\alpha_i$的值计算分割面$w'x + b = 0$的参数$w$和$b$：

$$\begin{cases} \hat{w} = \sum_{i=1}^{n}\hat{\alpha}_i y_i x_i \\ \hat{b} = y_j - \sum_{i=1}^{n}\hat{\alpha}_i y_i(x_i \cdot x_j) \end{cases}$$

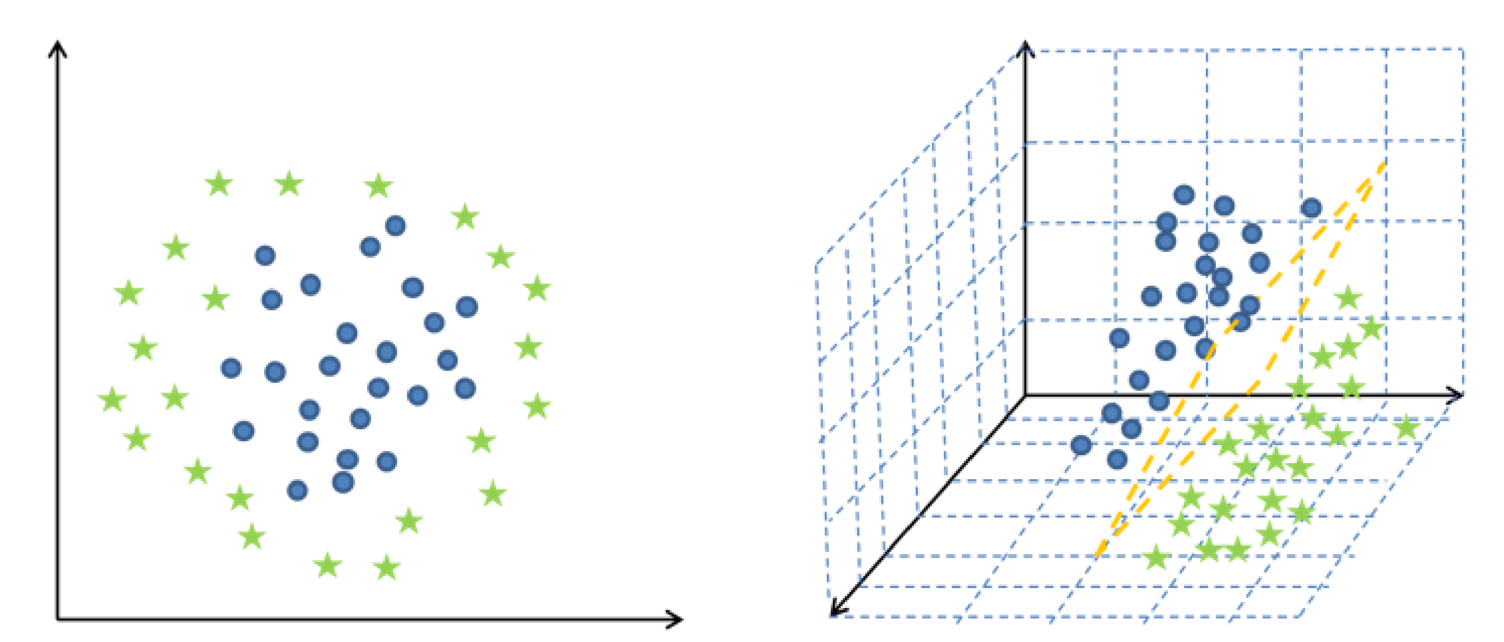
EDU CSDN学院 IT实战派

函数介绍

LinearSVC(tol=0.0001, C=1.0, multi_class='ovr', fit_intercept=True, intercept_scaling=1, class_weight=None, max_iter=1000)

tol：用于指定SVM模型迭代的收敛条件，默认为0.0001
C：用于指定目标函数中松弛因子的惩罚系数值，默认为1
fit_intercept：bool类型参数，是否拟合线性“超平面”的截距项，默认为True
intercept_scaling：当参数fit_intercept为True时，该参数有效，通过给参数传递一个浮点值，就相当于在自变量X矩阵中添加一常数列，默认该参数值为1
class_weight：用于指定因变量类别的权重，如果为字典，则通过字典的形式{class_label:weight}传递每个类别的权重；如果为字符串'balanced'，则每个分类的权重与实际样本中的比例成反比，当各分类存在严重不平衡时，设置为'balanced'会比较好；如果为None，则表示每个分类的权重相等
max_iter：指定模型求解过程中的最大迭代次数，默认为1000

非线性可分的SVM模型



目标函数

对于非线性SVM模型而言，需要经过两个步骤，一个是将原始空间中的样本点映射到高维的新空间中，另一个是在新空间中寻找一个用于识别各类别样本点线性“超平面”。

假设原始空间中的样本点为x，将样本通过某种转换$\phi(x)$映射到高维空间中，则非线性SVM模型的目标函数可以表示为：

$$\begin{cases} min_{\alpha}\left(\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j\left(\phi(x_i)\cdot\phi(x_j)\right)-\sum_{i=1}^{n}\alpha_i\right) \\ s.t.\quad \sum_{i=1}^{n}\alpha_i y_i = 0 \\ 0\leq\alpha_i\leq C \end{cases}$$

目标函数的求解

其中，内积$\phi(x_i)\cdot\phi(x_j)$可以利用核函数替换，即$K(x_i,x_j)=\phi(x_i)\cdot\phi(x_j)$。对于上式而言，同样需要计算最优的拉格朗日乘子$\alpha_i$，进而可以得到线性“超平面”$w$与$b$的值：

$$\begin{cases} \hat{w}=\sum_{i=1}^{n}\hat{\alpha}_i y_i \phi(x_i) \\ \hat{b}=y_j-\sum_{i=1}^{n}\hat{\alpha}_i y_i K(x_i,x_j) \end{cases}$$

非线性可分的SVM模型

目标函数的求解

其中，内积$\phi(x_i)\cdot\phi(x_j)$可以利用核函数替换，即$K(x_i,x_j)=\phi(x_i)\cdot\phi(x_j)$。对于上式而言，同样需要计算最优的拉格朗日乘子$\alpha_i$，进而可以得到线性“超平面”$w$与$b$的值：

$$
\begin{cases}
\hat{w}=\sum_{i=1}^{n}\hat{\alpha}_i y_i \phi(x_i) \\
\hat{b}=y_j-\sum_{i=1}^{n}\hat{\alpha}_i y_i K(x_i,x_j)
\end{cases}
$$

核函数

假设原始空间中的两个样本点为(x_i, x_j)，在其扩展到高维空间后，它们的内积$\phi(x_i)\cdot\phi(x_j)$如果等于样本点(x_i, x_j)在原始空间中某个函数的输出，那么该函数就称为核函数。

- 线性核函数
- 多项式核函数
- 高斯核函数
- Sigmoid核函数

经验之谈：大多数情况下，选择高斯核函数是一种相对偷懒而有效的方法，因为高斯核是一种指数函数，它的泰勒展开式可以是无穷维的，即相当于把原始样本点映射到高维空间中。

非线性可分的SVM模型

函数介绍

SVC(C=1.0, kernel= ‘rbf’ , degree=3, gamma= ‘auto’ , coef0=0.0, tol=0.001,
class_weight=None, verbose=False, max_iter=-1, random_state=None)

C：用于指定目标函数中松弛因子的惩罚系数值，默认为1
kernel：用于指定SVM模型的核函数，该参数如果为'linear'，就表示线性核函数；如果为'poly'，就表示多项式核函数，核函数中的r和p值分别使用degree参数和gamma参数指定；如果为'rbf'，表示径向基核函数，核函数中的r参数值仍然通过gamma参数指定；如果为'sigmoid'，表示Sigmoid核函数，核函数中的r参数值需要通过gamma参数指定；如果为'precomputed'，表示计算一个核矩阵
degree：用于指定多项式核函数中的p参数值
gamma：用于指定多项式核函数或径向基核函数或Sigmoid核函数中的r参数值
coef0：用于指定多项式核函数或Sigmoid核函数中的r参数值
tol：用于指定SVM模型迭代的收敛条件，默认为0.001
class_weight：用于指定因变量类别的权重，如果为字典，则通过字典的形式{class_label:weight}传递每个类别的权重；如果为字符串'balanced'，则每个分类的权重与实际样本中的比例成反比，当各分类存在严重不平衡时，设置为'balanced'会比较好；如果为None，则表示每个分类的权重相等
max_iter：指定模型求解过程中的最大迭代次数，默认为-1，表示不限制迭代次数

手写体字母的识别

```
# 读取外部数据
letters = pd.read_csv(r'C:\Users\Administrator\Desktop\letterdata.csv')
# 将数据拆分为训练集和测试集
predictors = letters.columns[1:]
X_train,X_test,y_train,y_test = model_selection.train_test_split(letters[predictors],
                                                   letters.letter, test_size = 0.25, random_state = 1234)
# 使用网格搜索法，选择线性可分SVM “类”中的最佳C值
C=[0.05,0.1,0.5,1,2,5]
parameters = {'C':C}
grid_linear_svc = model_selection.GridSearchCV(estimator = svm.LinearSVC(),
                                  param_grid =parameters, scoring='accuracy',cv=5,verbose =1)
# 模型在训练数据集上的拟合
grid_linear_svc.fit(X_train,y_train)
# 返回交叉验证后的最佳参数值
grid_linear_svc.best_params_, grid_linear_svc.best_score_
```

out:
({'C': 0.1}, 0.6915333333333333)

手写体字母的识别

```
# 模型在测试集上的预测
pred_linear_svc = grid_linear_svc.predict(X_test)
# 模型的预测准确率
metrics.accuracy_score(y_test, pred_linear_svc)

out:
0.7147999999999999
```

手写体字母的识别

```
# 使用网格搜索法，选择非线性可分SVM “类” 中的最佳C值和核函数
kernel=['rbf','linear','poly','sigmoid']
C=[0.1,0.5,1,2,5]
parameters = {'kernel':kernel,'C':C}
grid_svc = model_selection.GridSearchCV(estimator = svm.SVC(), param_grid =parameters,
                                        scoring='accuracy',cv=5,verbose =1)
# 模型在训练数据集上的拟合
grid_svc.fit(X_train,y_train)
# 返回交叉验证后的最佳参数值
grid_svc.best_params_, grid_svc.best_score_

out:
({'C': 5, 'kernel': 'rbf'}, 0.9734000000000004)
```

手写体字母的识别

```
# 模型在测试集上的预测
pred_svc = grid_svc.predict(X_test)
# 模型的预测准确率
metrics.accuracy_score(y_test,pred_svc)
```

out:
0.9788

经过5重交叉验证后，发现最佳的惩罚系数C为5，最佳的核函数为径向基核函数。相比于线性可分SVM模型来说，基于核技术的SVM表现了极佳的效果，模型在训练数据集上的平均准确率高达97.34%，而且其在测试数据集的预测准确率也接近98%，说明利用非线性可分SVM模型拟合及预测手体字母数据集是非常理想的。

EDU

CSDN学院　IT实战派