西北大学

智能信息系统综合实践
实验报告

题　　目：　SVM 算法

年　　级：　2020

专　　业：　软件工程

姓　　名：　何铖俊

一、题目（原题目）

- 1. 利用Iris Dataset（选前两类），交叉验证，测试SVM分类器性能，改变SVM超参数，对比不同超参数对结果的影响。
- 2. 对比SVM和第一章中的逻辑回归分类器，自定衡量指标。

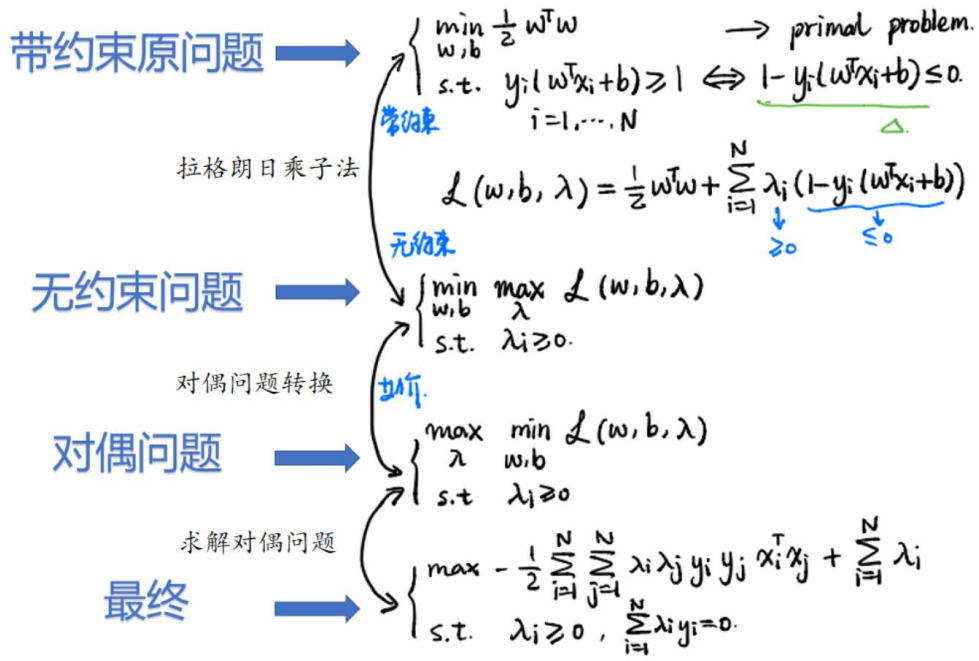
二、解题步骤（思路+代码）

支持向量机（Support Vector Machine，SVM）是一种常用的监督学习算法，它可以用于分类和回归问题。SVM 的基本思想是找到一个超平面来对数据进行分割，使得分割后的两个类别间的间隔最大化，从而达到最优分类效果。

具体来说，SVM 试图在数据集中找到一个超平面，使得不同类别之间的间隔最大化。这个超平面可以被视为一个线性决策边界，将不同类别的数据分隔开来。为了最大化这个间隔，SVM 选择距离超平面最近的一些样本点作为支持向量（Support Vector），并根据它们的位置计算出超平面的位置和形状。

在实际应用中，SVM 可以通过不同的核函数对数据进行非线性变换，将低维的数据映射到高维空间中，从而使得不同类别之间的间隔更加明显。

总而言之，根据 SVM 的思想，可以将一系列约束问题转化为如下所示的最终求解带偶问题



而 iris 数据集它包含了 3 类，每类 50 个样本，每个样本有 4 个特征。iris 数据集是一个多元分类问题，旨在通过测量花萼长度、花萼宽度、花瓣长度和花瓣宽度这 4 个特征来对 3 类鸢尾花进行分类。但由于该数据集是一个四维数据集，因此得对其进行核函数降维，然后再进行 SVM 分类。

$$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle = \phi(\boldsymbol{x}_i)^{\mathrm{T}} \phi(\boldsymbol{x}_j)$$

名称	表达式	参数
线性核	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{x}_j$	
多项式核	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{x}_j)^d$	$d \geqslant 1$ 为多项式的次数
高斯核	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{2\sigma^2}\right)$	$\sigma > 0$ 为高斯核的带宽(width)
拉普拉斯核	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|}{\sigma}\right)$	$\sigma > 0$
Sigmoid 核	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \tanh(\beta \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{x}_j + \theta)$	tanh 为双曲正切函数, $\beta > 0, \theta < 0$

先进行 Iris 数据集的分析和处理

对于特征进行一些统计描述

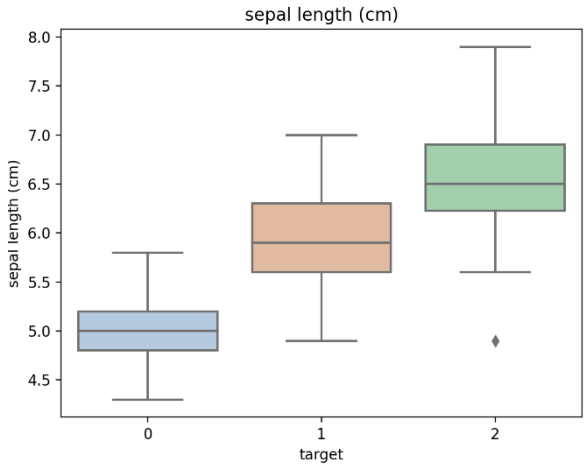
```
import pandas as pd
from sklearn.datasets import load_iris
data = load_iris() #得到数据特征
```

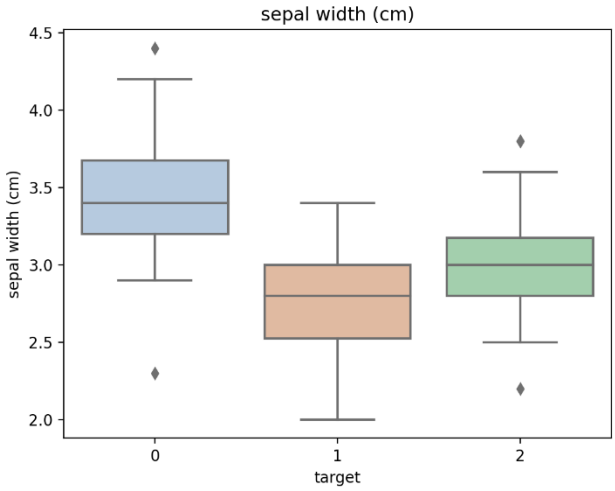
```
iris_target = data.target #得到数据对应的标签
iris_features = pd.DataFrame(data=data.data,
columns=data.feature_names) #利用 Pandas 转化为 DataFrame 格式码片
print(iris_features.describe())
```

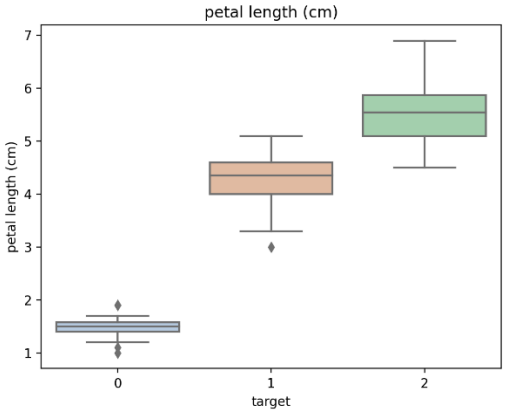
	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.057333	3.758000	1.199333
std	0.828066	0.435866	1.765298	0.762238
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

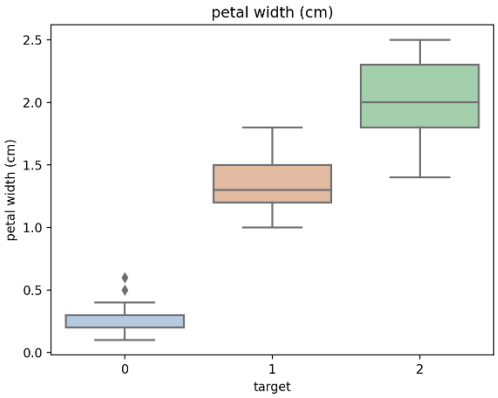
画箱线图看不同类别下各个特征的分布情况，可以根据箱线图知道该特征的 最大值、最小值、上 1/4 分位点和下 1/4 分位点以及中位数。

```
## 合并标签和特征信息
iris_all = iris_features.copy() ##进行浅拷贝，防止对于原始数据的修改
iris_all['target'] = iris_target
for col in iris_features.columns:
    sns.boxplot(x='target', y=col, saturation=0.5, palette='pastel',
data=iris_all)
    plt.title(col)
    plt.show()
```









第一题

- 1. 利用Iris Dataset（选前两类），交叉验证，测试SVM分类器性能，改变SVM超参数，对比不同超参数对结果的影响。

Step1：导入所需要的包

```
# 导入需要得库
import numpy as np
from sklearn import datasets
from sklearn.svm import SVC
from sklearn.model_selection import KFold
from sklearn.metrics import
f1_score,accuracy_score,recall_score,precision_score
```

step2：导入 iris 数据集,并选取前两类

```
# 导入 iris 数据集
iris = datasets.load_iris()
X = iris.data[:100,]
y = iris.target[:100]
```

Step3:先以默认值定义 SVM 模型，同时以 10 折交叉验证来进行验证

```
# 定义 SVM 模型
svm = SVC(kernel='rbf')

# 定义 10 折交叉验证参数
kf = KFold(n_splits=10, shuffle=True, random_state=42)
```

Step4: 创建一个空列表保存每次交叉验证得F1、average、precision、

recall 值

```
# 创建一个空列表保存每次交叉验证得 F1、average、precision、recall 值
f1_list = []
a_list = []
p_list = []
r_list = []
```

Step5:对每个划分进行交叉验证,同时将所有划分的各个评判数值进行记录

```
# 使用 for 循环遍历每次划分索引
for train_index, test_index in kf.split(X):
    # 用训练集索引获取对应数据和标签，并训练模型
    X_train, y_train = X[train_index], y[train_index]
    svm.fit(X_train, y_train)
    # 用测试集索引获取对应数据和标签，并预测结果
    X_test, y_test = X[test_index], y[test_index]
    y_pred = svm.predict(X_test)
    # 计算预测结果和真实标签之间得 F1 值，并保存在列表中
    f1 = f1_score(y_test, y_pred, average='macro')
    accuracy = accuracy_score(y_test, y_pred)
    precision = precision_score(y_test, y_pred,average='macro')
    recall = recall_score(y_test, y_pred,average='macro')
    f1_list.append(f1)
    r_list.append(recall)
    p_list.append(precision)
    a_list.append(accuracy)
```

Step6：对其进行平均，获得该模型的平均 F1、average、precision、recall 值

```
# 计算列表中 F1 值得平均值，作为最终结果
f1_mean = np.mean(f1_list)
a_mean = np.mean(a_list)
r_mean = np.mean(r_list)
p_mean = np.mean(p_list)
print("The mean F1 score of 10-fold cross validation on iris dataset
using SVM is:", f1_mean)
print("The mean accuary score of 10-fold cross validation on iris
dataset using SVM is:", a_mean)
print("The mean precision score of 10-fold cross validation on iris
dataset using SVM is:", p_mean)
```

```
print("The mean recall score of 10-fold cross validation on iris dataset using SVM is:", r_mean)
```

SVM 模型默认参数最终结果为

```
The mean F1 score of 10-fold cross validation on iris dataset using SVM is: 1.0
The mean accuary score of 10-fold cross validation on iris dataset using SVM is: 1.0
The mean precision score of 10-fold cross validation on iris dataset using SVM is: 1.0
The mean recall score of 10-fold cross validation on iris dataset using SVM is: 1.0
```

从 sklearn 包注释中我们可以得知，所调用的 svm 模型有以下几种参数

（1）C：目标函数的惩罚系数 C，用来平衡分类间隔 margin 和错分样本的，default C = 1.0；
（2）kernel：参数选择有 RBF，Linear，Poly，Sigmoid，默认的是"RBF"；
（3）degree：if you choose 'Poly' in param 2，this is effective，degree 决定了多项式的最高次幂；
（4）gamma：核函数的系数('Poly'，'RBF' and 'Sigmoid')，默认是 gamma = 1 / n_features；
（5）coef0：核函数中的独立项，'RBF' and 'Poly'有效；
（6）probablity：可能性估计是否使用(true or false)；
（7）shrinking：是否进行启发式；
（8）tol（default = 1e - 3）：svm 结束标准的精度；
（9）cache_size：制定训练所需要的内存（以 MB 为单位）；
（10）class_weight：每个类所占据的权重，不同的类设置不同的惩罚参数 C，缺省的话自适应；
（11）verbose：跟多线程有关，不大明白啥意思具体；
（12）max_iter：最大迭代次数，default = 1，if max_iter = -1，no limited；
（13）decision_function_shape：'ovo' 一对一，'ovr' 多对多 or None 无，default=None
（14）random_state：用于概率估计的数据重排时的伪随机数生成器的种子。
其中 7、8、9 一般不进行考虑

而对于我们的鸢尾花数据集是一个多维的二分类任务，因此我们需要考虑参数 C、kernel。

对于这两个参数，将采用单一变量法来衡量该参数对分类精度的影响

（1）只考虑参数 C

```
svm = SVC(C=pow(10,2))
```

```
The mean F1 score of 10-fold cross validation on iris dataset using SVM is: 1.0
The mean accuary score of 10-fold cross validation on iris dataset using SVM is: 1.0
The mean precision score of 10-fold cross validation on iris dataset using SVM is: 1.0
The mean recall score of 10-fold cross validation on iris dataset using SVM is: 1.0
```

```
svm = SVC(C=pow(10,1))
```

```
The mean F1 score of 10-fold cross validation on iris dataset using SVM is: 1.0
The mean accuary score of 10-fold cross validation on iris dataset using SVM is: 1.0
The mean precision score of 10-fold cross validation on iris dataset using SVM is: 1.0
The mean recall score of 10-fold cross validation on iris dataset using SVM is: 1.0
```

(default=1.0)

```
# 定义 SVM 模型
svm = SVC()
```

```
The mean F1 score of 10-fold cross validation on iris dataset using SVM is: 1.0
The mean accuary score of 10-fold cross validation on iris dataset using SVM is: 1.0
The mean precision score of 10-fold cross validation on iris dataset using SVM is: 1.0
The mean recall score of 10-fold cross validation on iris dataset using SVM is: 1.0
```

C=0.1

```
# 定义 SVM 模型
svm = SVC(C=pow(10,-1))
```

```
The mean F1 score of 10-fold cross validation on iris dataset using SVM is: 1.0
The mean accuary score of 10-fold cross validation on iris dataset using SVM is: 1.0
The mean precision score of 10-fold cross validation on iris dataset using SVM is: 1.0
The mean recall score of 10-fold cross validation on iris dataset using SVM is: 1.0
```

C=0.01

```
# 定义 SVM 模型
svm = SVC(C=pow(10,-2))
```

```
The mean F1 score of 10-fold cross validation on iris dataset using SVM is: 0.31684981684981683
The mean accuary score of 10-fold cross validation on iris dataset using SVM is: 0.39
The mean precision score of 10-fold cross validation on iris dataset using SVM is: 0.24500000000000005
The mean recall score of 10-fold cross validation on iris dataset using SVM is: 0.55
```

C=0.0001

```
# 定义 SVM 模型
svm = SVC(C=pow(10,-4))
```

```
The mean F1 score of 10-fold cross validation on iris dataset using SVM is: 0.31684981684981683
The mean accuary score of 10-fold cross validation on iris dataset using SVM is: 0.39
The mean precision score of 10-fold cross validation on iris dataset using SVM is: 0.24500000000000005
The mean recall score of 10-fold cross validation on iris dataset using SVM is: 0.55
```

根据以上数据，可以粗略得到一个结论，当其他参数一致，只有 C 值

变化时，C 值越小，所得到的分类精确程度越低

（2）只考虑参数 kernel

(default=’rbf’) 径向基核函数

```
svm = SVC()
```

```
The mean F1 score of 10-fold cross validation on iris dataset using SVM is: 1.0
The mean accuary score of 10-fold cross validation on iris dataset using SVM is: 1.0
The mean precision score of 10-fold cross validation on iris dataset using SVM is: 1.0
The mean recall score of 10-fold cross validation on iris dataset using SVM is: 1.0
```

kernel=’poly’ 多项式核函数

```
svm = SVC(kernel='poly')
```

```
The mean F1 score of 10-fold cross validation on iris dataset using SVM is: 1.0
The mean accuary score of 10-fold cross validation on iris dataset using SVM is: 1.0
The mean precision score of 10-fold cross validation on iris dataset using SVM is: 1.0
The mean recall score of 10-fold cross validation on iris dataset using SVM is: 1.0
```

kernel=’sigmoid’ sigmod 核函数

```
svm = SVC(kernel='sigmoid')
```

```
The mean F1 score of 10-fold cross validation on iris dataset using SVM is: 0.2399267399267399
The mean accuary score of 10-fold cross validation on iris dataset using SVM is: 0.32
The mean precision score of 10-fold cross validation on iris dataset using SVM is: 0.16375
The mean recall score of 10-fold cross validation on iris dataset using SVM is: 0.48
```

kernel=’linear’ 线性核函数

```
svm = SVC(kernel='linear')
```

```
The mean F1 score of 10-fold cross validation on iris dataset using SVM is: 1.0
The mean accuary score of 10-fold cross validation on iris dataset using SVM is: 1.0
The mean precision score of 10-fold cross validation on iris dataset using SVM is: 1.0
The mean recall score of 10-fold cross validation on iris dataset using SVM is: 1.0
```

根据上述结论，可以粗略得到的结论是，当其他参数一致，只有 kernel 参数变化时，除了 sigmod 核函数以外，其他三种的分类精确程度都很高。

（3）结合两个参数 C 和 kernel。

根据先前的数据，惩罚因子在 0.01 之后开始对精确程度产生影响，而 kernel 核函数除 sigmoid 对精确度影响很大以外，其他模型都比

较适合。

因此在本轮中采用 poly 模型和 0.01、0.0001 的 C 值来进行模型精确度预测。

以及采用 sigmoid 模型和 0.01、0.0001 的 C 值来进行模型精确度预测。

poly 模型

```
svm = SVC(kernel='poly',C=pow(10,-2))
```

```
The mean F1 score of 10-fold cross validation on iris dataset using SVM is: 0.989010989010989
The mean accuary score of 10-fold cross validation on iris dataset using SVM is: 0.99
The mean precision score of 10-fold cross validation on iris dataset using SVM is: 0.9875
The mean recall score of 10-fold cross validation on iris dataset using SVM is: 0.9928571428571429
```

```
svm = SVC(kernel='poly',C=pow(10,-4))
```

```
The mean F1 score of 10-fold cross validation on iris dataset using SVM is: 0.3067488067488068
The mean accuary score of 10-fold cross validation on iris dataset using SVM is: 0.38
The mean precision score of 10-fold cross validation on iris dataset using SVM is: 0.23666666666666666
The mean recall score of 10-fold cross validation on iris dataset using SVM is: 0.54
```

sigmoid 模型

```
svm = SVC(kernel='sigmoid',C=pow(10,-2))
```

```
The mean F1 score of 10-fold cross validation on iris dataset using SVM is: 0.2399267399267399
The mean accuary score of 10-fold cross validation on iris dataset using SVM is: 0.32
The mean precision score of 10-fold cross validation on iris dataset using SVM is: 0.16375
The mean recall score of 10-fold cross validation on iris dataset using SVM is: 0.48
```

```
svm = SVC(kernel='sigmoid',C=pow(10,-4))
```

```
The mean F1 score of 10-fold cross validation on iris dataset using SVM is: 0.2399267399267399
The mean accuary score of 10-fold cross validation on iris dataset using SVM is: 0.32
The mean precision score of 10-fold cross validation on iris dataset using SVM is: 0.16375
The mean recall score of 10-fold cross validation on iris dataset using SVM is: 0.48
```

分析和总结：

从惩罚因子 C 值的角度

poly 模型的结论可以推导至其他表现优异的核函数模型（如 rbf、

linear）具有类似的相同结论：C 值（惩罚因子）越小，分类精确度越低。

根据惩罚因子的原理：惩罚因子 C 的取值权衡了经验风险和结构风险：C 越大，经验风险越小，结构风险越大，容易出现过拟合；C 越小，模型复杂度越低，容易出现欠拟合。而该实验结果是符合理论预期的。

从核函数的角度

1、线性核函数：线性核函数是最简单的核函数之一，它的形式为 K(x, y) = x^T y，其中 x 和 y 是两个样本向量。线性核函数适用于数据线性可分的情况，它的计算速度比其他核函数快，并且不容易过拟合。
2、多项式核函数：多项式核函数的形式为 K(x, y) = (x^T y + r)^d，其中 r 和 d 分别为多项式核函数的参数。多项式核函数可以用于处理非线性可分数据，但是它的计算复杂度比线性核函数高，而且需要调整参数 d 和 r，对于不同的数据集可能需要不同的参数。
3、径向基核函数：径向基核函数是最常用的核函数之一，它的形式为 K(x, y) = exp(-γ||x-y||^2)，其中 γ 是径向基函数的参数。径向基核函数可以处理非线性可分数据，它的计算复杂度较高，但是对于数据的依赖性较小，参数 γ 的选择对结果有很大的影响。
4、Sigmoid 核函数：Sigmoid 核函数的形式为 K(x, y) = tanh(α x^T y + c)，其中 α 和 c 是 Sigmoid 核函数的参数。Sigmoid 核函数也可以处理非线性可分数据，但是它的表现不如径向基核函数好，而且容易出现过拟合现象。

对于 iris 数据集，由于该数据集的特征较为简单，样本也相对较少，建议选择计算速度快、泛化性能较好的线性核函数或径向基核函数。如果数据集中存在一些非线性关系，可以选择径向基核函数，并对参数进行适当调整，以达到最佳的分类效果。如果数据集中存在多个类别之间的非线性关系，则可以考虑使用多项式核函数。

鸢尾花数据集是一个多分类问题，常用的 SVM 多分类方法是基于一对多策略，即将每个类别与其他类别分别作为正负样本训练一个 SVM 分类器，最终将这些分类器组合起来形成一个多分类模型。在这种情况

下，使用 sigmoid 核函数可能会出现以下问题：

梯度消失或梯度爆炸问题。Sigmoid 函数的导数在它的自变量很大或很小的时候会趋于 0 或 1，这可能导致在模型训练过程中出现梯度消失或梯度爆炸的问题，使得模型无法收敛或者收敛缓慢。

分类效果较差。在多分类问题中，sigmoid 核函数的分类效果往往不如其他核函数，如 RBF 核函数或多项式核函数，这可能是因为 sigmoid 核函数的决策边界是 S 形的，很难适应复杂的非线性边界。

因此，除了 sigmoid 核函数，其他三种（poly,linear,rbf）都是比较适合 iris 数据集的核函数。

第二题

- 2. 对比SVM和第一章中的逻辑回归分类器，自定衡量指标。

逻辑回归是一种机器学习算法，用于解决分类问题，例如判断一个人是否患有某种疾病 1。逻辑回归的原理是利用逻辑函数(也叫 Sigmoid 函数）将线性回归的结果映射到 0 到 1 之间的一个概率值 234，然后根据概率值和阈值（通常为 0.5）来判断数据属于哪个类别。

Sigmoid函数 $g(z)=\frac{1}{1+e^{-z}}$

利用逻辑回归模型在 Iris 数据集二分类（与 SVM 模型都取前两个分类）上进行训练和预测

在代码上，其余部分和 SVM 对鸢尾花数据集前两类分类一致，不同处如下

```
for train_index, test_index in kf.split(X):
    # 用训练集索引获取对应数据和标签，并训练模型
    X_train, y_train = X[train_index], y[train_index]
    clf = LogisticRegression(random_state=0, solver='lbfgs')
    clf.fit(X_train, y_train)
```

```
    # 用测试集索引获取对应数据和标签，并预测结果
    X_test, y_test = X[test_index], y[test_index]
    y_pred = clf.predict(X_test)
    # 计算预测结果和真实标签之间得 F1 值，并保存在列表中
    f1 = f1_score(y_test, y_pred, average='macro')
    accuracy = accuracy_score(y_test, y_pred)
    precision = precision_score(y_test, y_pred,average='macro')
    recall = recall_score(y_test, y_pred,average='macro')
    f1_list.append(f1)
    r_list.append(recall)
    p_list.append(precision)
    a_list.append(accuracy)
```

```
The  F1 score of Logistic Regression on iris dataset is: 1.0
The  accuary score of Logistic Regression on iris dataset is: 1.0
The  precision score of Logistic Regression on iris dataset is: 1.0
The  recall score of Logistic Regression on iris dataset is: 1.0
```

从四个值可以得出，逻辑回归和 SVM 对小数据集的分类效果都是比较好的。

接着我们使用 barest-cancer 数据集对两个模型分别进行分类精确程度的监测。

加载 barest-cancer 数据集（有 30 个特征，569 个数据点，目标是恶性（癌性）或良性（非癌性）。）

```
from sklearn.datasets import load_breast_cancer
cancer_data = load_breast_cancer()
X = cancer_data.data
y = cancer_data.target
```

逻辑回归：

```
The mean F1 score of 10-fold cross validation on iris dataset using SVM is: 0.9404925184680026
The mean accuary score of 10-fold cross validation on iris dataset using SVM is: 0.9454573934837093
The mean precision score of 10-fold cross validation on iris dataset using SVM is: 0.9407140727392538
The mean recall score of 10-fold cross validation on iris dataset using SVM is: 0.9419816379544546
```

SVM：

```
The mean F1 score of 10-fold cross validation on iris dataset using SVM is: 0.9077559066901009
The mean accuary score of 10-fold cross validation on iris dataset using SVM is: 0.9173245614035087
The mean precision score of 10-fold cross validation on iris dataset using SVM is: 0.9290168697862384
The mean recall score of 10-fold cross validation on iris dataset using SVM is: 0.8987702869869116
```

根据上述数据我们可以得出结论：在大数据集的分类处理上，逻辑回归的精确程度会高于 SVM。

总结

（1）都是监督分类算法，判别模型；

（2）逻辑回归和 SVM 都可以处理分类问题，且一般都用于处理线性二分类问题（在改进的情况下可以处理多分类问题）；

（3）对于小规模数据集，SVM 的效果要好于逻辑回归，但是大数据中，SVM 的计算复杂度受到限制，而 LR 因为训练简单，可以在线训练，所以经常会被大量采用。

三、总结（心得体会）

1 、应当在实验过程中保持一个刨根问底的求学态度，才可以在学习过程中将收获最大化。

2、在进行训练之前，应该先对数据集的总体分布、特征有个大概的印象和了解，方便后续使用恰当的方法进行分类模型和预测。

3、当一种方法行不通时，应当敢于尝试其他的方法，而不是死脑经盯着一处。

4、实验过程中不能心急，急躁的情绪只会让结果越来越不符合自己的心理预期，导致恶性循环。