# 《数据挖掘》课程实验报告

### 姓名：常远

### 学号：42023017

### 日期：2023年4月17日星期一

### 实验名称：建立朴素贝叶斯和KNN分类模型

# 实验目的：

1、在iris数据集上，按照8:2随机划分训练集和测试集合。

2、用朴素贝叶斯和KNN建立两个分类模型

# 实验内容：

朴素贝叶斯分类器基于贝叶斯理论。

朴素贝叶斯分类器的决策函数为：



公式上的每一个概率值都能在训练数据上计算出来。由于概率值都是小于1的数，累乘会导致数据过小。因此，在实际运算中，我们将累乘改成累加。得最终函数为：



在本次实验中，我们使用高斯朴素贝叶斯来计算鸢尾花数据集。首先，我们通过.data命令获取数据，得到前4个特征值。再根据x\_train, x\_test, y\_train, y\_test = train\_test\_split(iris\_x, iris\_y, test\_size=0.2,random\_state=1)，对数据进行分类，按8：2随机划分训练集和测试集合。再得到训练集和测试集后，使用 GaussianNB 类在创建了高斯朴素贝叶斯模型的对象 gnb 后，调用其 fit() 函数进行训练，最后在测试集上进行测试和性能评价。

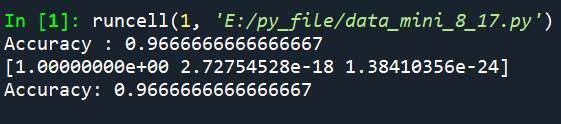
可以看到在数据集上，模型的准确度高达96.67%，

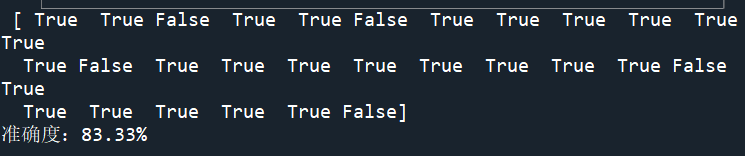
在第二种方法中，我们同样先加载鸢尾花数据，对数据进行分类，按8:2随机划分训练集和测试集。使用 GaussianNB 类在创建了高斯朴素贝叶斯模型的对象 gnb 后，调用其 fit() 函数进行训练拟合。之后我们将其可视化，分别去0列特征的最大最小值，取1列特征得最大最小值；生成500个测试点，进行预测。但是在该方法中，我们只采取前两个特征值，得到准确度为83.33%，相比较于取四个特征值进行预测的准确度稍差。

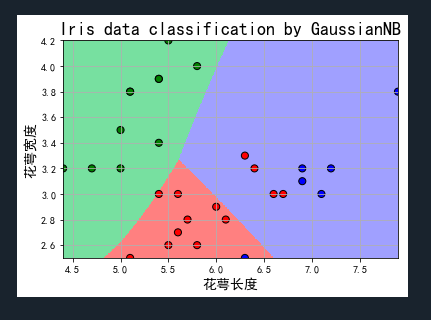
KNN算法也是一种分类模型，对于1条测试数据对象x，KNN模型从训练集中发现和x最相似的k个数据对象作为近邻。根据k个近邻对象的类别标签进行决策，按照少数服从多数的原则将x分配近邻中数量最多的类。

和朴素贝叶斯代码实现一样，先导入鸢尾花数据，取前4个特征值，对数据进行分类，按8：2随机划分训练集和测试集。设最近邻的数量是5，同时将距离设为数据点的权重，用fit()函数进行训练拟合。当我们取前4个特征值计算时，不管是uniform和distance准确率都100%，当我们取第一个特征值时，uniform的准确率为63.33%，distance的准确率为66.67%，准确率有一定提高，说明数据对象按与测试对象的距离的倒数作为权重的准确率更高，但是鸢尾花数据比较少，不具有代表性。

# 结论：





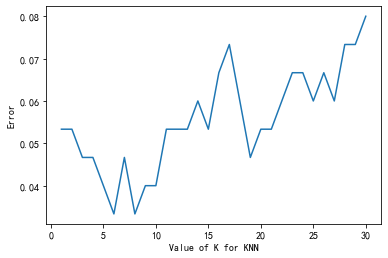


朴素贝叶斯对小规模的数据表现很好，能处理多分类任务；但是只能用于分类问题需要计算先验概率。









KNN属于消极学习，不需要建立模型，但是如果不采用适当的邻近性度量和数据预处理，KNN可能会做出错误的预测。如上图可知，当取30个训练对象作为测试对象的近邻，KNN的误差会接近到8%。

我们看到，在朴素贝叶斯和KNN算法中，采取适当的特征值和训练对象准确率可以达到95%以上。

# 附录：

#%%

# hw 第八周

# gaussian NB

from sklearn import datasets

import pandas as pd

import numpy as np

from sklearn.model\_selection import train\_test\_split

from sklearn.naive\_bayes import GaussianNB

# 读入数据

# df=pd.read\_csv('iris.data')

# # y = df['Personal Loan'] #目标列

# # X = df.drop(['ID', 'ZIP Code','Personal Loan'], axis=1)

# y=df['5.1']

# X=df.drop(['5.1','3.5','1.4'],axis=1)

# X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=0)

iris = datasets.load\_iris() #加载鸢尾花数据

iris\_x = iris.data #获取数据

iris\_x = iris\_x[:, :4] #取前4个特征值

iris\_y = iris.target

x\_train, x\_test, y\_train, y\_test = train\_test\_split(iris\_x, iris\_y, test\_size=0.2,random\_state=1)

#对数据进行分类，按8:2随机划分训练集和测试集合

# 得到训练集和测试集

#训练高斯朴素贝叶斯模型

gnb = GaussianNB()

gnb.fit(x\_train, y\_train)

# 评估模型

y\_pred = gnb.predict(x\_test)

acc =sum (y\_test == y\_pred)/x\_test.shape[0]

print("Accuracy : %s" % (acc))

y\_pred = gnb.predict\_proba(x\_test)

print(y\_pred[0])

acc = gnb.score(x\_test, y\_test)

print('Accuracy: %s'%acc)

#%%

from sklearn import datasets

from sklearn.model\_selection import train\_test\_split

from sklearn.naive\_bayes import MultinomialNB, GaussianNB

import matplotlib.pyplot as plt

import numpy as np

import matplotlib as mpl

from sklearn.preprocessing import StandardScaler

from sklearn.pipeline import Pipeline

iris = datasets.load\_iris() #加载鸢尾花数据

iris\_x = iris.data #获取数据

iris\_x = iris\_x[:, :2] #取前2个特征值

iris\_y = iris.target

x\_train, x\_test, y\_train, y\_test = train\_test\_split(iris\_x, iris\_y, test\_size=0.2,random\_state=1)

#对数据进行分类，按8:2随机划分训练集和测试集合

clf = GaussianNB()

ir = clf.fit(x\_train, y\_train) #利用训练数据进行拟合

#plot

x1\_max, x1\_min = max(x\_test[:, 0]), min(x\_test[:, 0])

#取0列特征的最大最小值

x2\_max, x2\_min = max(x\_test[:, 1]), min(x\_test[:, 1])

#取1列特征的最大最小值

t1 = np.linspace(x1\_min, x1\_max, 500) #生成500个测试点

t2 = np.linspace(x2\_min, x2\_max, 500)

x1, x2 = np.meshgrid(t1, t2) #生成网格采样点

x\_test1 = np.stack((x1.flat, x2.flat),axis=1)

y\_hat = ir.predict(x\_test1) #预测

mpl.rcParams['font.sans-serif'] = [u'simHei'] #识别中文保证不乱码

mpl.rcParams['axes.unicode\_minus'] = False

cm\_light = mpl.colors.ListedColormap(['#77E0A0', '#FF8080', '#A0A0FF'])

# 测试分类的颜色

cm\_dark = mpl.colors.ListedColormap(['g', 'r', 'b'])

# 样本点的颜色

plt.figure(facecolor='w')

plt.pcolormesh(x1, x2, y\_hat.reshape(x1.shape),cmap=cm\_light)

plt.scatter(x\_test[:, 0], x\_test[:, 1], edgecolors='k',s=50,c=y\_test, cmap=cm\_dark)

plt.xlabel(u'花萼长度', fontsize=14)

plt.ylabel(u'花萼宽度', fontsize=14)

plt.title(u'Iris data classification by GaussianNB', fontsize=18)

plt.grid(True)

plt.xlim(x1\_min, x1\_max)

plt.ylim(x2\_min, x2\_max)

plt.show()

y\_hat1 = ir.predict(x\_test)

result = y\_hat1 == y\_test

print(result)

acc = np.mean(result)

print('准确度：%.2f%%' % (100 \* acc))

#%%

from sklearn.datasets import load\_iris

from sklearn.model\_selection import cross\_val\_score

import matplotlib.pyplot as plt

from sklearn.neighbors import KNeighborsClassifier

#读取鸢尾花数据集

iris = load\_iris()

x = iris.data

y = iris.target

#归一化数据

from sklearn.preprocessing import StandardScaler

sc = StandardScaler()

x\_scaled = sc.fit\_transform(x)

k\_range = range(1,51)

k\_error = []

#循环，取k=1到k=30,查看误差效果

for k in k\_range:

knn = KNeighborsClassifier(n\_neighbors=k)

#cv参数决定数据集划分比例，这里是按照8:2划分训练集和测试集

scores = cross\_val\_score(knn, x\_scaled, y, cv=5, scoring='accuracy')

k\_error.append(1-scores.mean())

#画图，x轴为k值，y值为误差值

plt.plot(k\_range, k\_error)

plt.xlabel('Value of K for KNN')

plt.ylabel('Error')

plt.show()

#%%

from sklearn import datasets

import pandas as pd

import numpy as np

from sklearn.model\_selection import train\_test\_split

from sklearn.neighbors import KNeighborsClassifier

# df=pd.read\_csv('UniversalBank.csv')

# y = df['Personal Loan']

# X = df.drop(['ID', 'ZIP Code','Personal Loan'], axis=1)

# X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=0)

iris = datasets.load\_iris() #加载鸢尾花数据

iris\_x = iris.data #获取数据

iris\_x = iris\_x[:, :4] #取前4个特征值

iris\_y = iris.target

x\_train, x\_test, y\_train, y\_test = train\_test\_split(iris\_x, iris\_y, test\_size=0.2,random\_state=1)

#对数据进行分类，按8:2随机划分训练集和测试集合

n\_neighbors=5

for weights in ['uniform', 'distance']:

knn = KNeighborsClassifier(n\_neighbors, weights=weights)

knn.fit(x\_train, y\_train)

acc = knn.score(x\_test,y\_test)

print('%s accuracy: %s'%(weights, acc))