朴素贝叶斯

分类问题

一个训练数据集，数据集中有一些样本，每个样本有一个或若干特征xx(这里xx的是向量)，每个样本都属于一个类别wiwi,通过这个训练集得到一个分类器，这个分类器具有这样的功能:

给定一个新的样本(也就是给定一个特征xx)，你的分类器能判断它属于哪个类别

可以把分类器看成是一个函数f(x)f(x),函数的返回值就是对应特征所属的类别，所有的分类器的目的都是去寻找一个比较好的函数能更好的描述特征与类别之间的关系。

​

贝叶斯定理：

P(wi|x)=P(x|wi)P(wi)∑kj=1P(x|wj)P(wj)

P(wi|x)=P(x|wi)P(wi)∑j=1kP(x|wj)P(wj)

​

P(wi)P(wi)是，表示在不知道样本特征的情况下，某个样本属于wiwi类的概率。

P(x|wi)P(x|wi)是类条件概率密度，可以看成是在某一类别的情况下特征的分布(概率密度函数)

P(wi|x)P(wi|x)是后验概率，就是在知道样本的特征的情况下该样本属于某一个特征的概率。

​

贝叶斯分类器的设计思路

一句话概括贝叶斯分类器:

在知道先验概率和类条件概率密度的情况下算后验概率，后验概率最大的类别作为最终类别

我们的问题是”给定特征判断这个特征所对应的类别”,一个容易想到的思路是算出这个特征属于每个类别的概率，然后取最大的那个类别作为最终的分类。其中”属于某个类别的概率”可以表示成条件概率的形式P(wi|x)P(wi|x)，也就是贝叶斯定理中的后验概率,接下来就是去求得P(x|wi)P(x|wi)和P(wi)P(wi)就行了.

代码实现

#include<iostream>

#include<set>

#include<string.h>

#include<vector>

using namespace std;

/\*

为了清晰表达算法的过程,没有写成通用的算法，很多参数直接给出

如需通用只需动态计算这些参数即可

\*/

int x1[]={1,1,1,1,1,2,2,2,2,2,3,3,3,3,3}; //第一个特征向量

int x1D=3; //就是取值只有三种（1,2,3）

//第二个特征向量

char x2[]={'S','M','M','S','S','S','M','M','L','L','L','M','M','L','L'};

int x2D=3; //同上

//对应的类别

int Y[]={-1,-1,1,1,-1,-1,-1,1,1,1,1,1,1,1,-1};

int y1=6; //6个-1

int y2=9; //9个1

int yD=2;

int NUM=15; //一共15个实例

int getx1Num(int userx,int y)

{

int n=0;

for(int i=0;i<NUM;i++) //在1类中找x1=2的实例个数

{

if(userx==x1[i]&&y==Y[i])

n++;

}

return n;

}

int getx2Num(int userx,int y)

{

int n=0;

for(int i=0;i<NUM;i++) //在1类中找x1=2的实例个数

{

if(userx==x2[i]&&y==Y[i])

n++;

}

return n;

}

int main()

{

int userx1=2; //待分类实例第一个特征

char userx2='S'; //待分类实例第二个特征

int r=1; //就是那个nameda

double py1,py2; //p(Y=1),p(Y=-1)的先验概率

py1= ((double)(y1+r))/(NUM+yD\*r); //先验概率公式:(第一个类的个数+r)/(实例个数+类别个数\*r)

py2=((double)(y2+r))/(NUM+yD\*r);

cout<<"p(Y=-1)= "<<(y1+r)<<"/"<<NUM+yD\*r<<endl;

cout<<"p(Y=1)= "<<(y2+r)<<"/"<<NUM+yD\*r<<endl;

//计算条件概率

int x1Num=getx1Num(userx1,-1); //计算-1类中有多少个x1=2

double px1y1=((double)(x1Num+r))/(y1+x1D\*r); //条件概率：(1类中2的个数+r)/(1类的个数+1类中元素的个数\*r)

cout<<"p(x1=2|Y=-1)= "<<x1Num+r<<"/"<<y1+x1D\*r<<endl;

int x2Num=getx1Num(userx1,1);

double px1y2=((double)(x2Num+r))/(y2+x2D\*r);

cout<<"p(x1=2|Y=1)= "<<x2Num+r<<"/"<<y2+x2D\*r<<endl;

int x21Num=getx2Num(userx2,-1);

double px2y1=((double)(x21Num+r))/(y1+x1D\*r);

cout<<"p(x2='S'|Y=-1)= "<<(x21Num+r)<<"/"<<(y1+x1D\*r)<<endl;

int x22Num=getx2Num(userx2,1);

double px2y2=((double)(x22Num+r))/(y2+x2D\*r);

cout<<"p(x2='S'|Y=1)= "<<(x22Num+r)<<"/"<<(y2+x2D\*r)<<endl;

double re1=py1\*px1y1\*px2y1;

double re2=py2\*px1y2\*px2y2;

cout<<re1<<"\t"<<re2<<endl;

int tclass=re1>re2?-1:1;

cout<<"("<<userx1<<","<<userx2<<")"<<"的类别是"<<tclass<<endl;

return 0;

}

