# 基于SVM的人脸识别

## 5.1 实验目的

①熟悉人脸检测及识别的基本原理和实现方法。

②学习支持向量机SVM的基础知识，了解其本质、软硬间隔和升维转换及核技巧入门。

③了解SVM的数学模型及推导，学习求解SVM决策超平面。

④学习SVM支持向量机-核技巧Kernel Trick详解（多项式核函数，无限维度高斯核函数）。

⑤了解SVM支持向量机的软间隔数学思想。

## 5.2 实验原理

### 5.2.1 问题引入

SVM(Support Vector Machine)支持向量机是机器学习里非常常见的算法，其本质是把两个不同类别的数据样本，通过找到一条理想中的最优超平面（平面内是一条线，三维内是一个面，如果是高维的则需要找到一个最优的超平面），使得后续的数据添加进来后仍然可以划分。理想的是找到两类数据中最中间的分界。

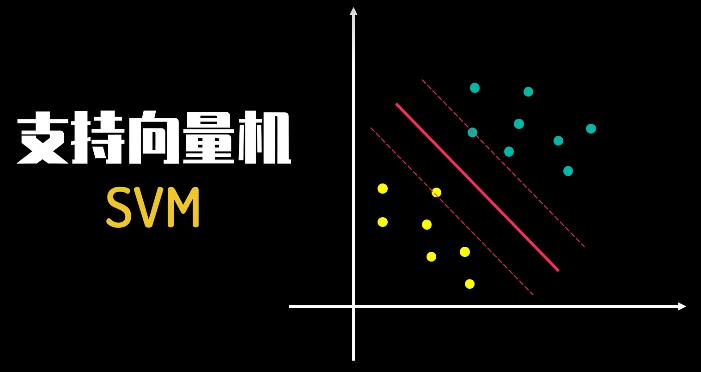


图5.1 支持向量机SVM示意

如图5.2，在这个二维的空间里存在着如图所示的一些数据点，不同的颜色代表着两种不同的数据类别。

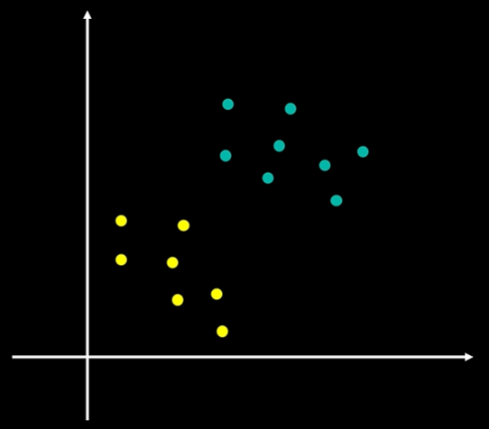


图5.2 空间内的一些数据点

如果需要画一条直线能将两类数据区分开来，并且当有新数据加入时，根据这条线，也能够判别它属于哪一类，那么应该如何画这条线呢？

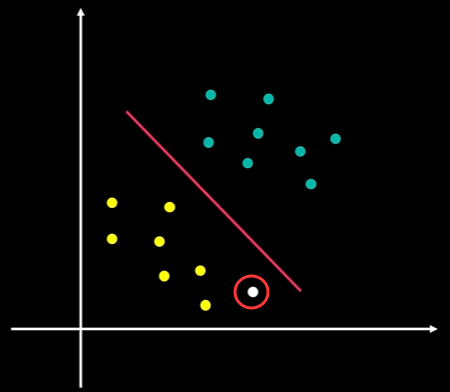
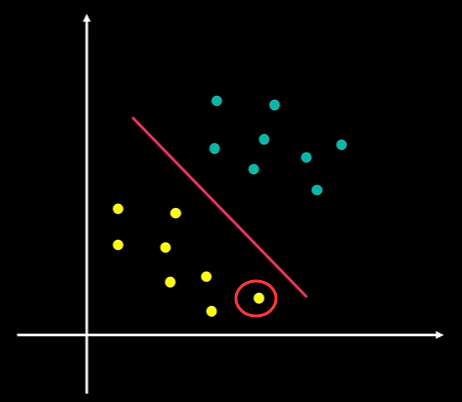
 

图5.3 新数据的加入并分类

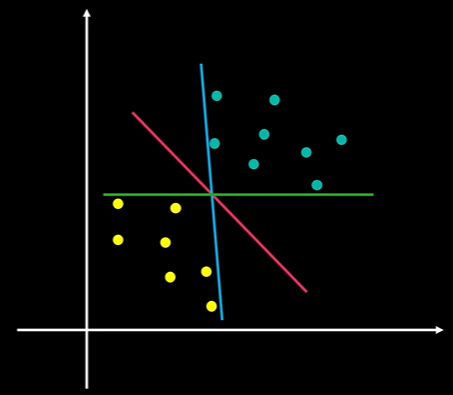


图5.4 不同直线画法

这个问题可以进一步延伸到三维空间中，如何让一个二维平面区分两类数据？

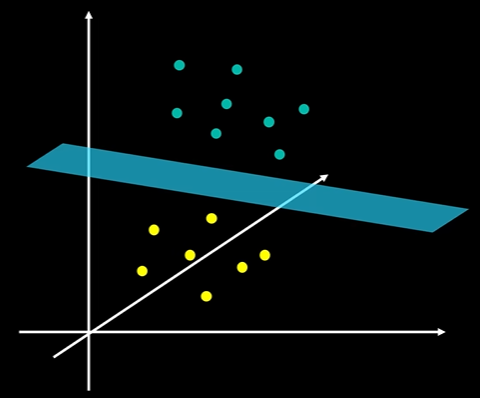


图5.5 三维空间中用一个二维平面区分两类数据

也可以再进一步推演到四维空间，如何让三位超平面区分两类数据？以此类推，以上这类问题就可以被归纳为：为了区分两类维度数据，N为数据的样本数，M为维度数，如何设计一个维度为M-1的超平面，将两类数据区分开来（W可以理解为X对应的权重）。

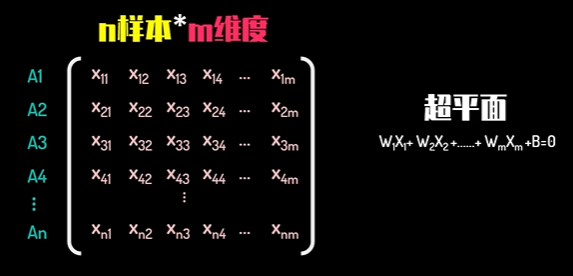


图5.6 问题归纳

### 5.2.2 在二维空间内的讨论

为了便于直观理解，以下先以二维空间的问题为例找到那条分割线，即决策边界（Decision Boundary）

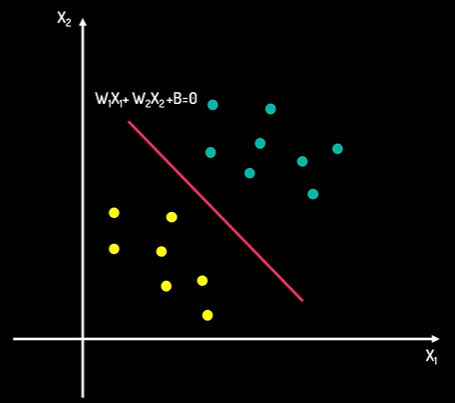
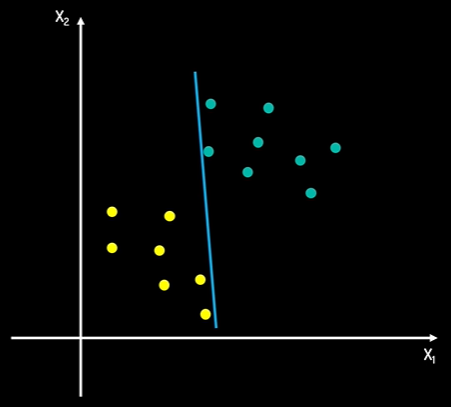
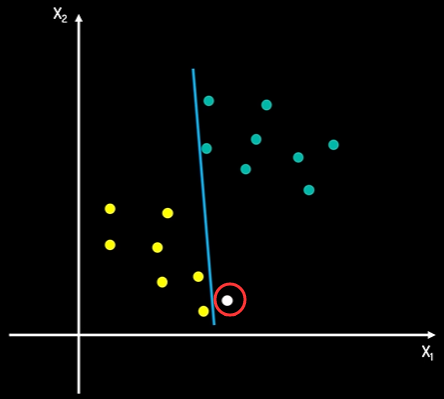


图5.7 二维空间内的决策边界

**一种画法及其存在的问题：**

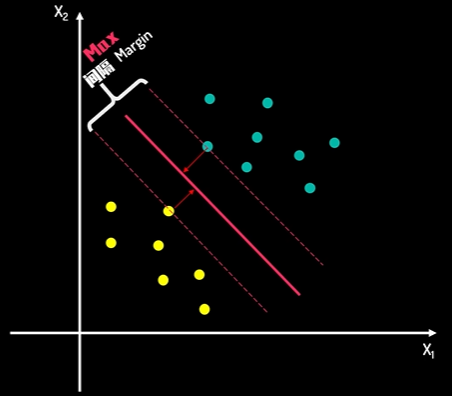


两类数据都有相应的数据点与决策边界线非常接近，这种画法是很危险的。当有一个新的数据同样接近于该直线时，这样分类错误的概率是非常的大的。

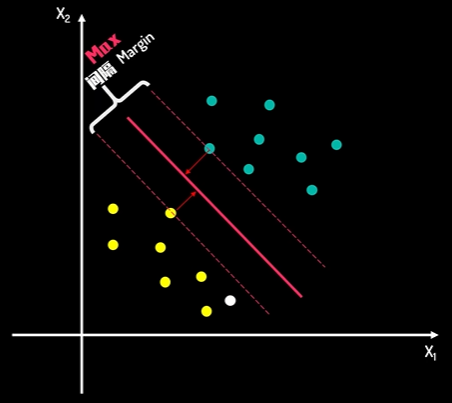
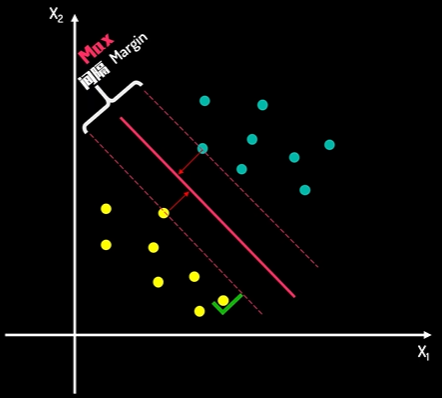


**第二种画法：**

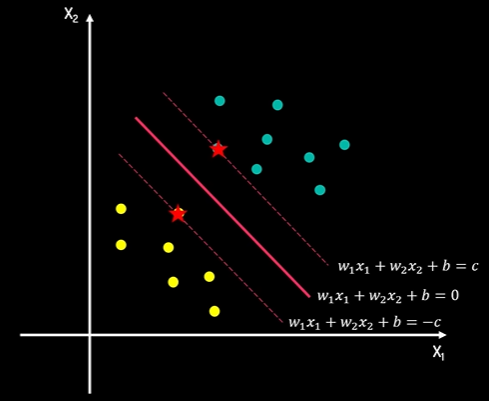
在这种画法中，两类数据中所有的点都与决策边界线保持了一定的距离，这个距离起到了缓冲区的作用。当这个缓冲区足够大时，分类结果的可信度就更高了，把这个缓冲区成为间隔（Margin）。这个间隔把两类数据所处的空间分隔开来，间隔距离可以体现出两类数据的差异大小，越大的间隔，意味着两位数据差异越大，那么数据区分起来就越容易。因此，寻找最佳决策边界线的问题可以转化为求解两类数据的最大间隔问题，而间隔的正中间就是决策边界。



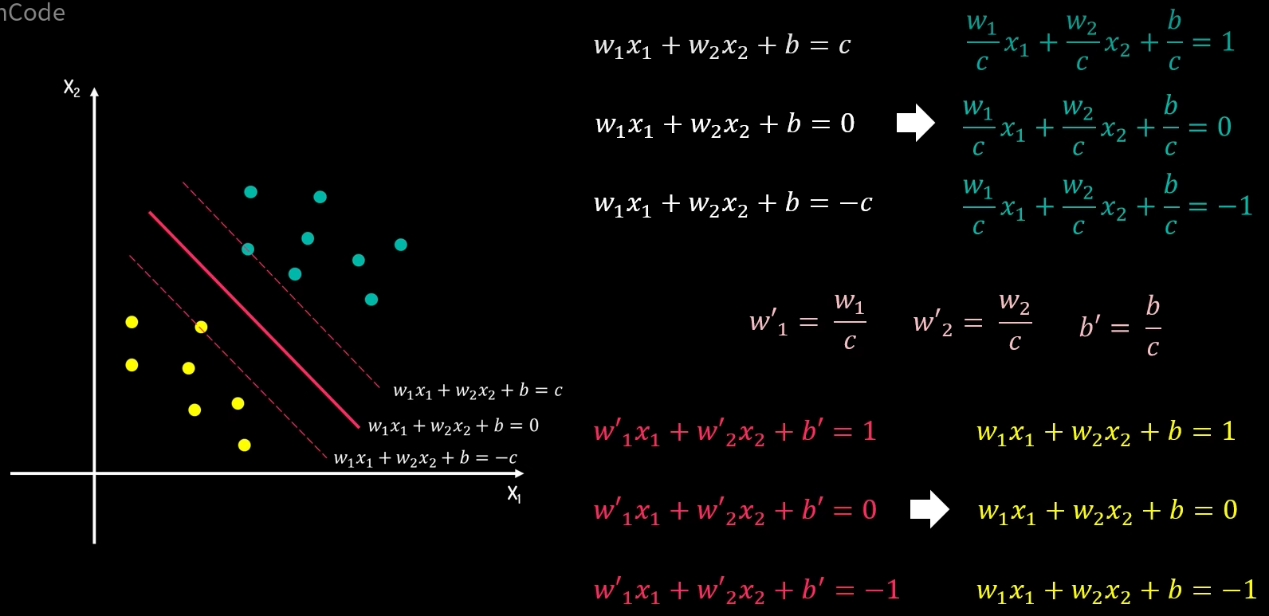
当有新的数据需要判断时，根据它处于决策边界的相对位置就可以进行分类了。

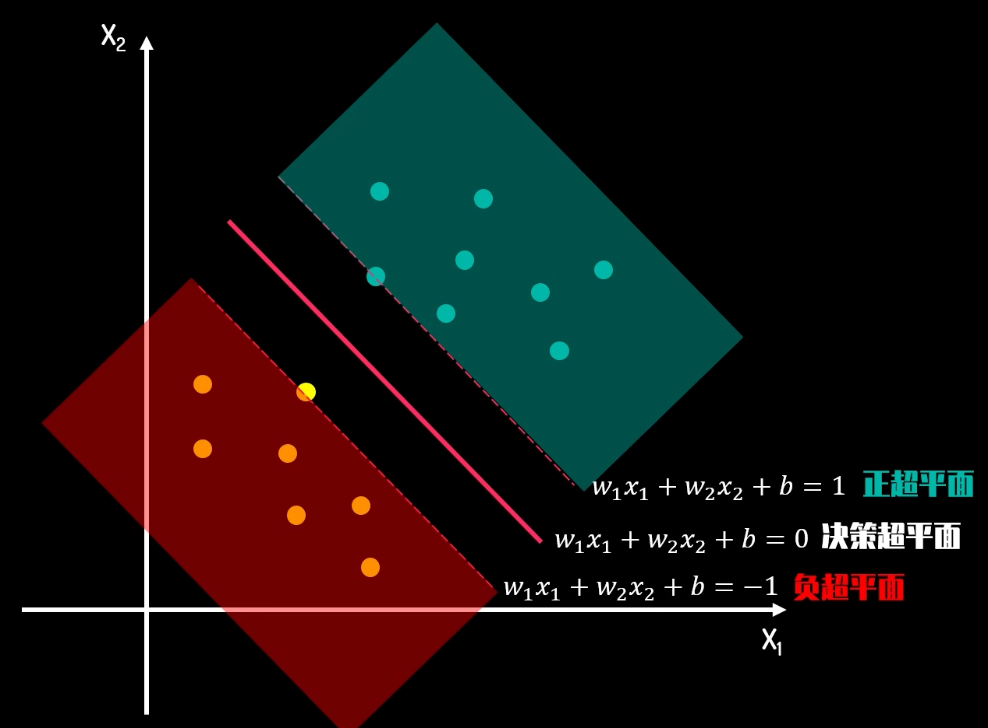
假设决策边界的超平面方程式为它向上、下分别移动C来到对应的间隔上下边界，由于上下边界一定会经过一些样本数据点，而这些点距离决策边界最近，它们决定了间隔距离，称它们为支持向量（Support Vector），这也是为什么把该方法命名为支持向量机（Support Vector Machine——SVM）。



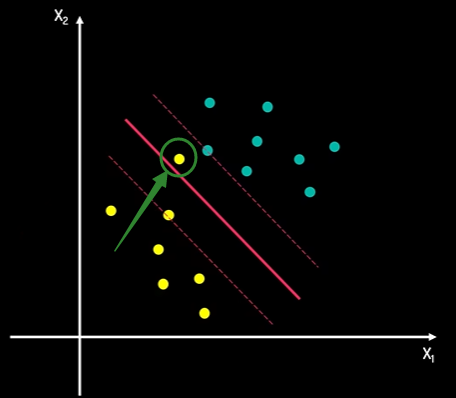
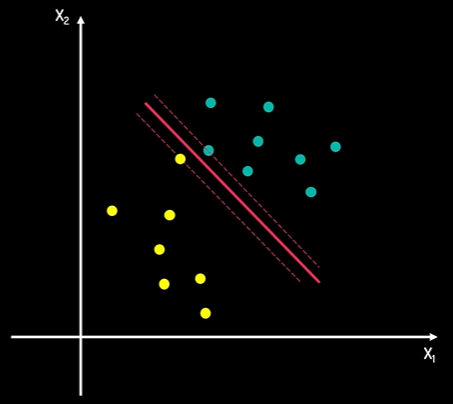
把等式两边分别除C得到如下图所示结果，利用、、分别替换原方程变量，得到新的方程，方程式右侧被转换为±1，这样就可以将上述两个方程式定义为正负超平面。由于、只是我们需要求解的代号，把它们替换成也不影响计算，所以最终只需要求解。



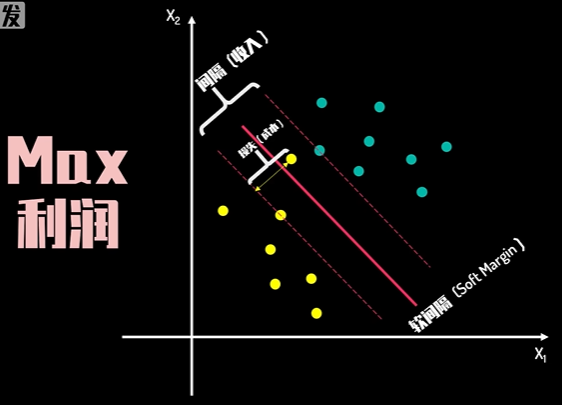
从而得到下面3个超平面方程式：正超平面、决策超平面和负超平面。所有正超平面及其上方的数据点颜色相同，都属于正类；负超平面及其下方的点为负类。当有新数据加入需要分类判断时，那么可以利用新加入的数据与决策超平面的相对位置来进行分类。

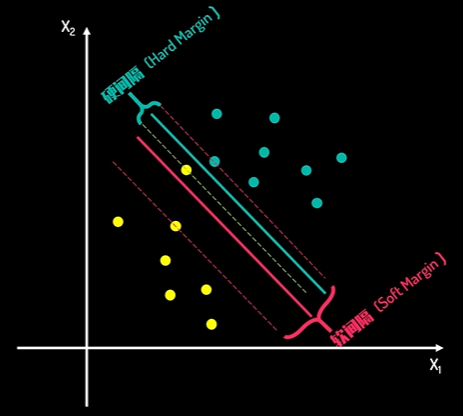


假如初始数据发生了如左下图的变化，根据之前的方法，最后间隔会变成右下图所示的样子。

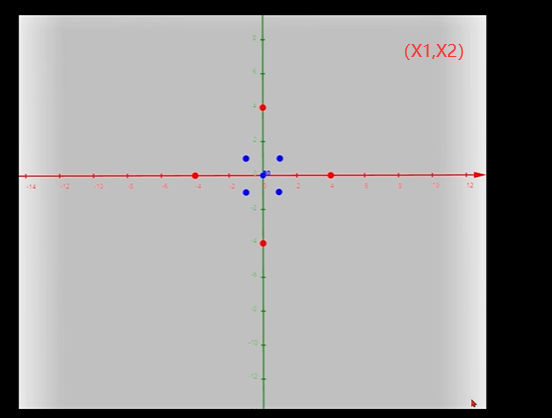
如果忽略中间这个异常点，那么间隔距离也会明显增加，那么问题是需不需要为这一个异常值，而牺牲的间隔距离呢（由于异常点的出现导致间隔变小，即牺牲了间隔距离）？当有更多的异常值呢？此时，需要引入损失因子这个概念，那些违背规则的异常点都会有对应的损失值。比如，把间隔距离想象成经营收入，而损失值想象成经营成本，目标则转化为同时考虑“收入”和“成本”因素去最大化我们的“利润”，这个最优解下形成的间隔称之为软间隔（Soft Margin）。它有一定的容错率，目的是在间隔距离和错误大小间找到一个平衡，而之前推导出的间隔则被成为硬间隔（Hard Margin）。



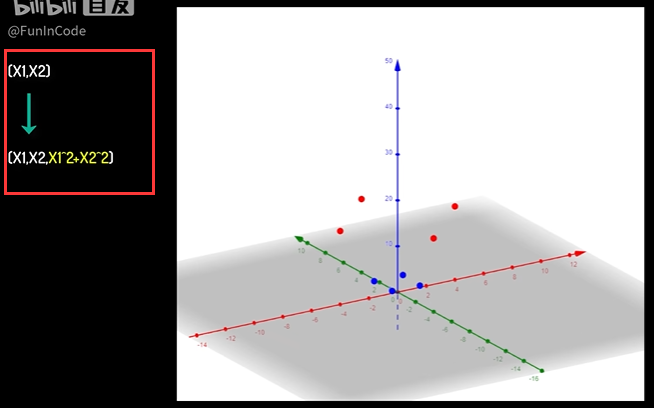


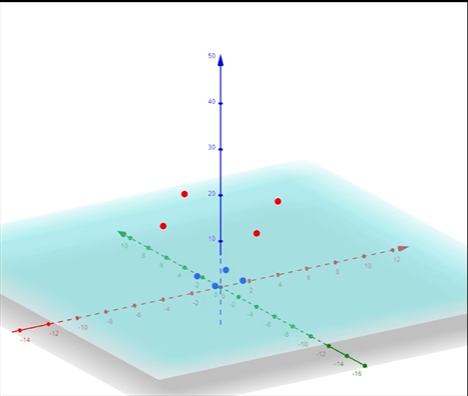
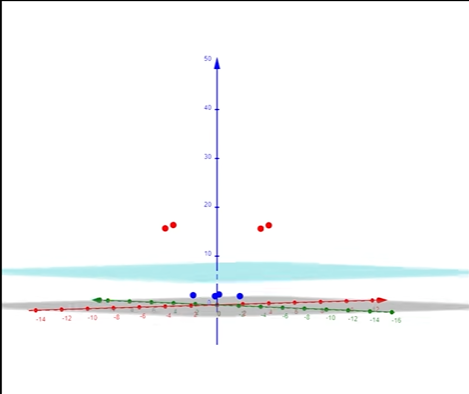
### 5.2.3 升维转换与核技巧（Kernel Trick）

在了解核技巧之前，首先需要了解什么是升维转换。如下图所示例子，在这个二维空间中，很显然我们不能通过一条直线将其有效区分。

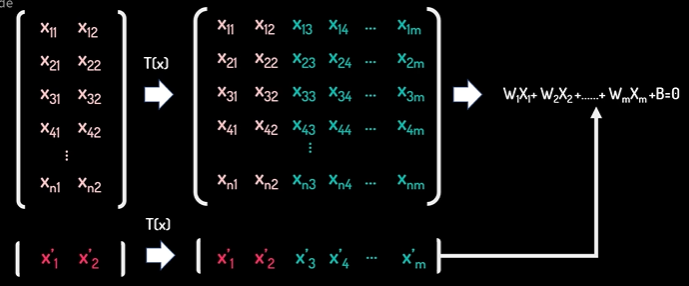


但如果进行升维转换，增加一个新维度的话，就可以在这个三维空间中通过超平面将数据区分了。

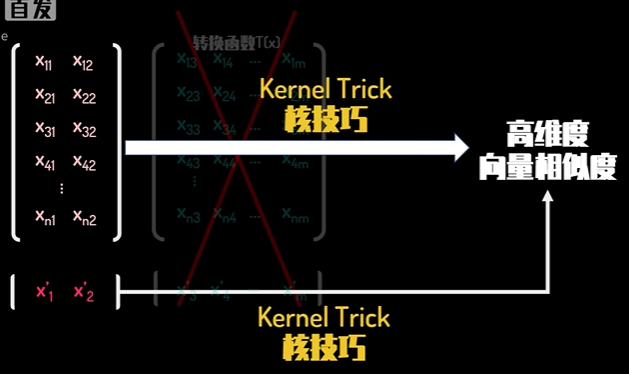


对于这些在低维度下无法方便分类的数据，可以采用类似的方法。通过合适的维度转换函数将低维数据进行升维，然后在高维度下求解SVM模型，找到对应的分隔超平面。当有新数据需要进行分类预测时，可以对其先做升维转换操作，再根据高维度下的决策边界超平面进行判断。



不过，提升维度需要明确的维度转换函数，以及更多的数据存储和计算需求。那么，有没有一种方法能够避免将数据送入高维度进行计算，却又能获得同样的分类效果呢？那么就需要运用Kernel Trick核技巧了。



由于SVM的本质是量化两类数据差异的方法，而和函数（Kernel Function）能够提供高维度向量相似度的测量，通过选取合适的核公式，可以不用知晓具体的维度转换函数，直接获得数据的高维度差异度，并以此来进行分类判断。

（

本节内容均摘自B站视频：【数之道】支持向量机SVM是什么，八分钟直觉理解其本质。

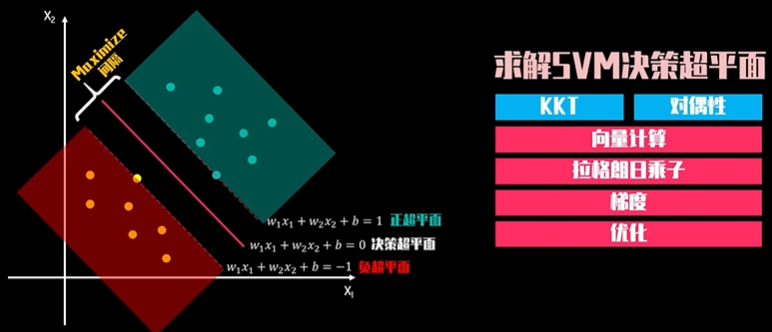
视频链接：

<https://www.bilibili.com/video/BV16T4y1y7qj/?spm_id_from=333.788&vd_source=f927924e46d4f85d34ee898c33f7337c>

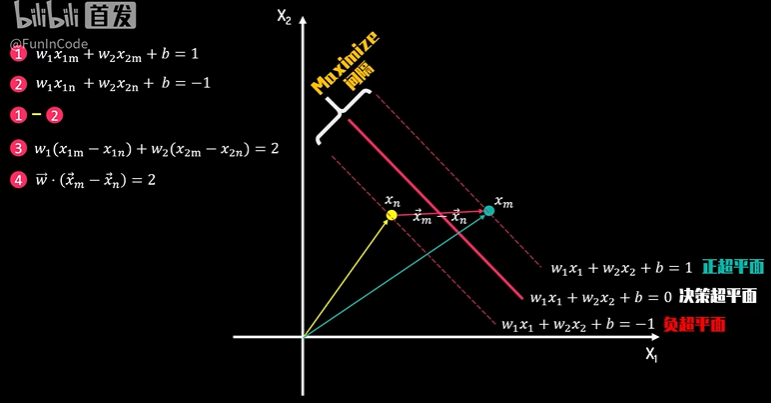
）

## 5.3 基本原理

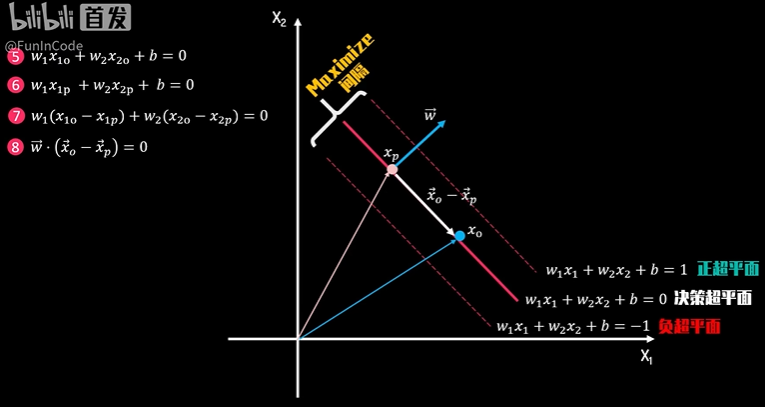
### 5.3.1 SVM支持向量机的数学内涵



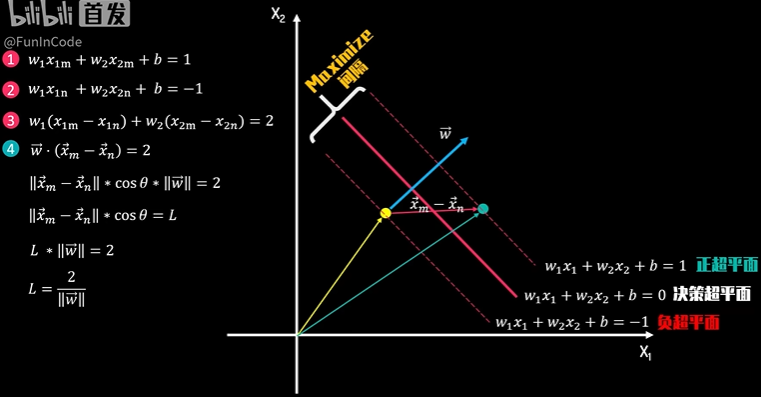
以二维数据为例，我们的目的是最大化正负超平面间的间隔距离L，那么构建L的表达式就非常重要了。分别在正负超平面上随机选取一个支持向量点。



在决策超平面上随机选择两个点，利用等式5减6得到等式7。

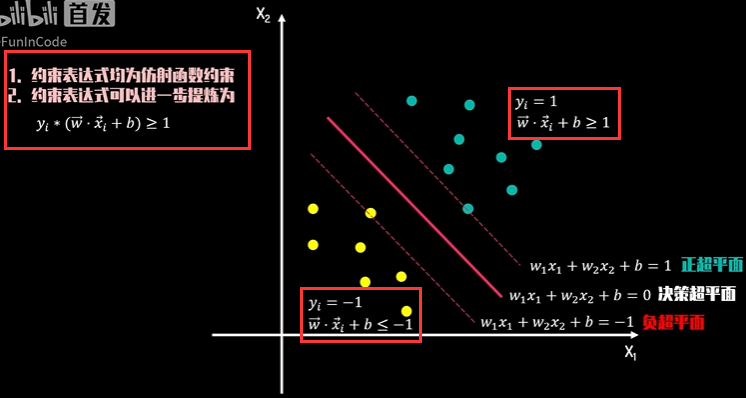


由上面可以得到向量w与决策超平面垂直。

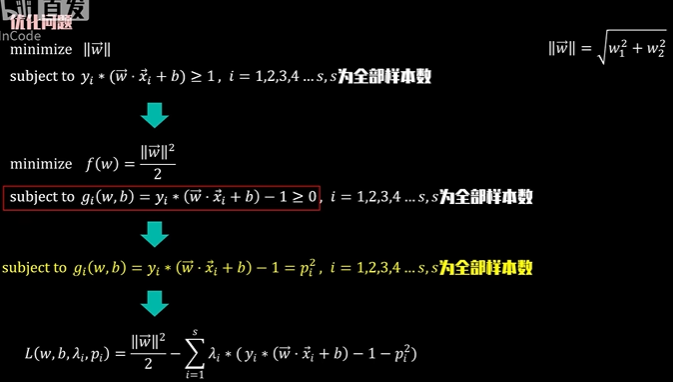


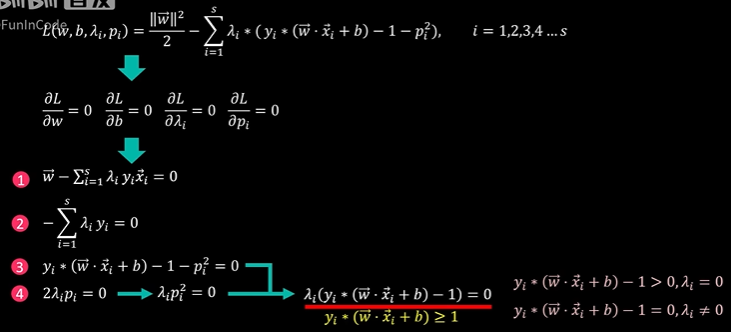
由于需要最大化L，这等同于求解在约束条件下向量w长度的最小化问题。

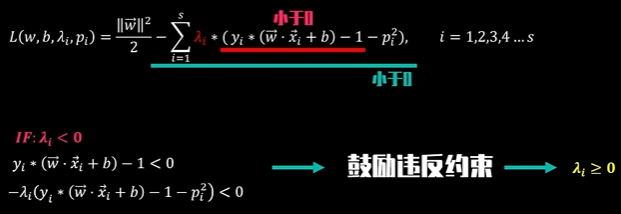
约束条件如下：



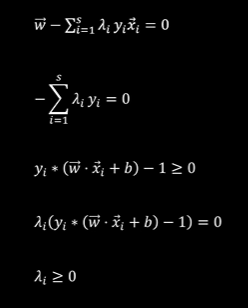
最后的优化问题可以总结为：





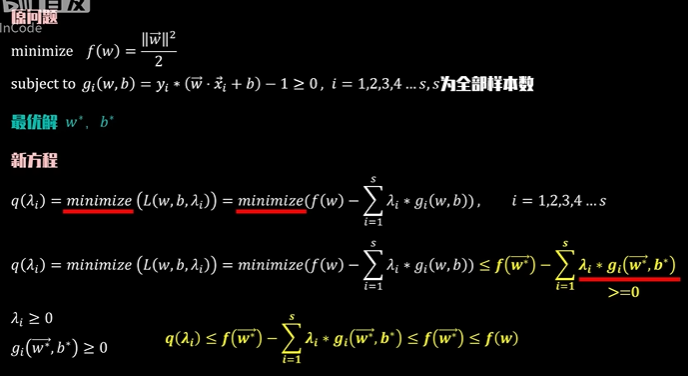


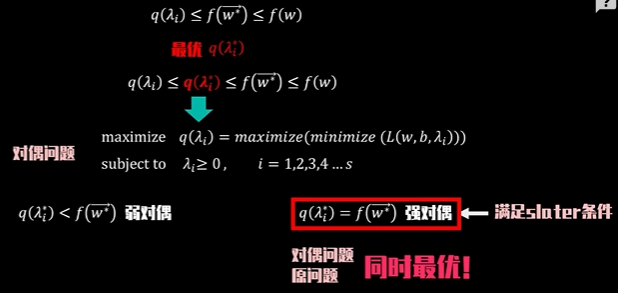
KKT条件如下：

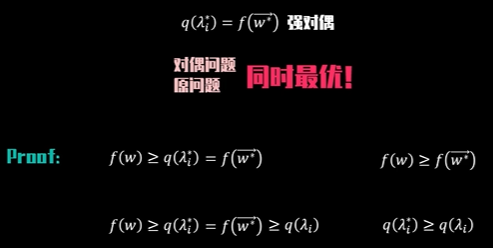


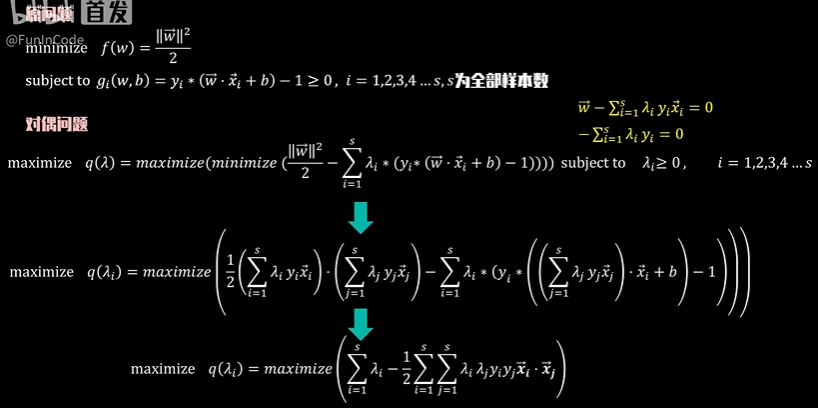
基于KKT条件，就可以求解最终的决策超平面了。

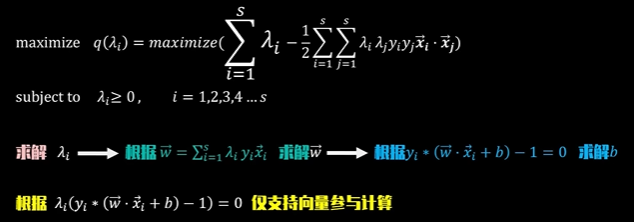
利用SVM对偶性求解问题：











关于利用数学推导求解SVM的最佳决策超平面，理解KKT条件、对偶性等抽象概念等内容，由于图文描述很抽象，不能很好的体现推导过程，因此推荐一个B站视频进行理解学习。视频链接如下：

【数之道25】机器学习必经之路-SVM支持向量机的数学精华

<https://www.bilibili.com/video/BV13r4y1z7AG/?spm_id_from=333.788&vd_source=f927924e46d4f85d34ee898c33f7337c>

### 5.3.2 SVM支持向量机——核技巧Kernel Trick详解

同5.3.1节，关于SVM的核技巧问题推荐学习B站视频，视频链接如下：

【数之道26】SVM支持向量机-核技巧Kernel Trick详解(多项式核函数，无限…)

<https://www.bilibili.com/video/BV1Nb4y1s7pE/?spm_id_from=333.788&vd_source=f927924e46d4f85d34ee898c33f7337c>

### 5.3.3 SVM支持向量机软间隔数学思想

同5.3.1节，关于SVM软间隔的数学本质问题推荐学习B站视频，视频链接如下：

【数之道27】详解SVM支持向量机软间隔数学思想

<https://www.bilibili.com/video/BV1AS4y1K7Jf/?spm_id_from=333.788&vd_source=f927924e46d4f85d34ee898c33f7337c>

## 5.4 实验步骤

### 5.4.1数据规范化

1. disp('训练特征数据规范化...')
2. disp('.................................................')
3. lowvec=min(pcatrainface);
4. upvec=max(pcatrainface);
5. scaledtrainface = scaling( pcatrainface,lowvec,upvec);

通过得到训练集特征数据的最大值和最小值，就可以实现将训练集数据归一化处理，将所有特征都等比压缩在0和1之间，防止某一特征数值过大导致权值过大，影响训练精度。

此处所使用的规范化函数scaling()如下：

1. function [ scaledface] = scaling( faceMat,lowvec,upvec )
2. %特征数据规范化
3. %输入——faceMat需要进行规范化的图像数据，
4. %       lowvec原来的最小值
5. %       upvec原来的最大值
6. upnew=1;
7. lownew=-1;
8. [m,n]=size(faceMat);
9. scaledface=zeros(m,n);
10. for i=1:m
11. scaledface(i,:)=lownew+(faceMat(i,:)-lowvec)./(upvec-lowvec)\*(upnew-lownew);
12. end
13. end

### 5.4.2 SVM样本训练

1. disp('SVM样本训练...')
2. disp('.................................................')
3. model=multiSVMtrain(scaledtrainface,trainlabel);
4. save('V.mat',"V")
5. save('mA.mat',"mA")
6. save('lowvec.mat',"lowvec")
7. save('upvec.mat',"upvec")
8. save('orl.mat',"model")       %保存模型，做GUI界面时就可以省的再训练了

将规范化后的训练集(scaledtrainface)，训练集对应的正确标签共同投喂给SVM分类器，给分类器输入特定参数使之训练得到最佳模型。至于libsvm分类器的安装和详细参数介绍，可以参考附录中的两篇博客。

此处可以直接展示SVM训练程序

1. function [ model ] =multiSVMtrain(traindata, trainlabel)
2. %多类别的SVM训练器
3. model=svmtrain(trainlabel,traindata,'-c  9e-04 -g 0.014');
4. end

通过libsvm工具箱，可以很方便的对分类器进行训练，即svmtrain程序。其中第一个参数为训练集对应标签，第二个参数为训练集，第三个参数为选择分类器内核等，附录有详细介绍。

至此，数据的预处理、PCA处理、规范化、训练等操作均已完成，接下来就可以投入测试集看看准确率，或是投入单个图片查看机器识别的是哪个人物。

### 5.4.3 输入测试集预测查看正确率

1. disp('读取测试数据...')
2. disp('.................................................')
3. [testface,testlabel]=ReadFace(npersons,1);
4. disp('测试数据特征降维...')
5. disp('.................................................')
6. m=size(testface,1);
7. for i=1:m
8. testface(i,:)=testface(i,:)-mA;
9. end
10. pcatestface=testface\*V;
11. disp('测试特征数据规范化...')
12. disp('.................................................')
13. scaledtestface = scaling( pcatestface,lowvec,upvec);
14. disp('SVM样本分类...')
15. disp('.................................................')
16. [pred,accuracy] = multiSVM(scaledtestface,testlabel,model);

首先读取测试集数据及其对应的标签，然后通过投影到特征向量V上，并进行规范化处理得到规范化后的测试集scaledtestface，通过multiSVM函数即可得到最终的预测准确率，该函数需要输入三个参数，第一个为测试集。第二个为测试集对应的标签，第三个为前面训练得到的SVM模型。

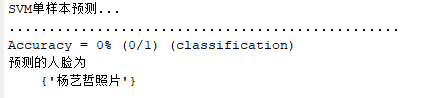


可知预测准确率仅25%，这意味着对仍然需要调试参数，包括pca维度、svm训练时的参数、训练集数量的选择等，有兴趣的同学可以自行调试。

### 5.4.4 输入单样本查看机器所预测的人物

1. load("dic\_cell.mat")
2. disp('SVM单样本预测...')
3. disp('.................................................')
4. [pred,accuracy] = multiSVM(scaledtestface(7),testlabel(7),model);
5. disp('预测的人脸为')
6. disp(dic\_cell(pred))

此处首先加载date\_process()函数的预处理程序所得到的名字字典dic\_cell，随后可随机输入一个样本scaledtestface(7)，输入到SVM分类器后，其会将预测的标签返回到pred中，通过查字典dic\_cell可得该标签对应的人名。预测结果如下：



综上，有关PCA+SVM人脸识别的实验程序到此结束。相关的6个主要的大步骤都有较为详细的说明，同学们也可以将程序copy下来自行运行调试，理解每段程序的用处及实现思路，对于特定函数里的个别语句如果难以理解则并不影响，只要对每个模块都有清晰的逻辑即可。

### 5.4.5 程序附录

1、如何将所有的命名不一的图片数据预处理？

1. function [dic\_cell] = data\_process()
2. %本函数的作用是将杂乱无章的原始数据集进行人脸检测后重新存放,用了很大部分工作量在这部分内容上!所以同学们以后命名应该统一！！
3. FilePath  = ["srcData\2021明月\2021明月" "srcData\2022明月一班照片\2022明月一班照片\明月一班照片" "srcData\2022明月二班照片\2022明月二班照片"];
4. %     stImageSavePath  = 'dstData\';
5. arr\_count = 1;
6. for fil = 1:3
7. par\_Path = FilePath(fil);                   %父文件夹
8. dir\_class\_PathList = dir(par\_Path);        %读取该目录下全部图片的路径（字符串格式）
9. i\_class\_Num  = length(dir\_class\_PathList);                    %获取文件夹的总数
10. for i = 3:i\_class\_Num
11. par\_Path = FilePath(fil);                   %父文件夹
12. arr\_name = dir\_class\_PathList(i).name;
13. filename = arr\_name;
14. tempPath = strcat(par\_Path,'\',filename);
15. temp\_PathList = dir(tempPath);
16. while isfolder(strcat(tempPath,'\',temp\_PathList(1).name))&&isfolder(strcat(tempPath,'\',temp\_PathList(2).name))&&isfolder(strcat(tempPath,'\',temp\_PathList(3).name))
17. filename = temp\_PathList(3).name;
18. par\_Path = tempPath;
19. tempPath = strcat(par\_Path,'\',filename);
20. temp\_PathList = dir(tempPath);
21. end
22. stImageSavePath = strcat('dstData\',int2str(arr\_count));
23. dic\_cell{arr\_count} = arr\_name;
24. arr\_count = arr\_count+1;
25. mkdir(stImageSavePath);
26. temp\_Num = length(temp\_PathList);
27. counts = 1;
28. for j = 1:12
29. minj = min(j,temp\_Num);
30. if temp\_PathList(minj).isdir == 0
31. iSaveNum      = int2str(counts);
32. stImagePath   = temp\_PathList(minj).name;
33. mImageCurrent = uint8(imread(strcat(tempPath,'\',stImagePath)));
34. mFaceResult   = face\_judge(mImageCurrent);                  %人脸识别截取后的图片，可用于进一步处理、
35. if numel(size(mFaceResult))>2                              %rgb图
36. mmFaceResult  = rgb2gray(mFaceResult);
37. else
38. mmFaceResult = mFaceResult;
39. end
40. finalResult = imresize(mmFaceResult,[128,128]) ;
41. imwrite(finalResult,strcat(stImageSavePath,'\',iSaveNum,'.jpg'));
42. counts = counts+1;
43. end
44. end
45. end
46. end
47. save('dic\_cell.mat',"dic\_cell");
48. end

其中包含有人脸识别的程序，该程序使用的是matlab自带的人脸识别工具箱，不为本实验的重点，故而不作赘述。

1. function IFaces = face\_judge(I)
2. faceDetector = vision.CascadeObjectDetector; % 构造检测器
3. bboxes = step(faceDetector, I); % 开始检测，返回结果到bboxes，[x,y,width,heigth]
4. if isempty(bboxes)
5. IFaces = I;
6. else
7. boxe = bboxes(:,3).\*bboxes(:,4);
8. [~,indexboxes] = max(boxe);
9. if (bboxes(indexboxes,3)<200)||(bboxes(indexboxes,4)<200)
10. IFaces = I;
11. else
12. IFaces = imcrop(I,bboxes(indexboxes,:));
13. end
14. end
15. %     figure, imshow(IFaces), title('Detected faces');
16. end

2、Libsvm工具箱的安装使用以及其相关参数的介绍。

MATLAB LibSVM安装——以MATLAB R2018B为例

<https://blog.csdn.net/Mactav/article/details/124106774>

3、Libsvm参数使用说明

<https://blog.csdn.net/FDA_sq/article/details/120012295>