**练习**

**题目：**

在总共200人的样本中： 有1 0人事实上患有A病（阳性）；经过检测后，9人判定患有A病，而1 人判定并不患有A病；外的1 90人实际上并不患有A病（阴性），经过检测后，其中的1 0人被判定患有A病，另外的1 80人判定不患有A病。

计算精度、错误率、特异度、敏感度。

**解答：**

TP（True Positive）表示真阳性，FP（False Positive）表示假阳性，TN（True Negative）表示真阴性，FN（False Negative）表示假阴性。

根据给定的数据：

10人实际上患有A病（阳性），其中9人被正确判定患病（TP = 9），1人被错误判定为不患病（FN = 1）。

190人实际上不患有A病（阴性），其中180人被正确判定不患病（TN = 180），10人被错误判定为患病（FP = 10）。

我们可以代入公式计算各个指标：

**精度（Accuracy）：**表示正确预测的比例。



**错误率（Error Rate）：**表示错误预测的比例。



**特异度（Specificity）：**表示正确识别阴性样本的比例。



**敏感度（Sensitivity）：**表示正确识别阳性样本的比例。



**作业1**

1. **原理**

**（一）Fashion-MNIST**

**来源**：论文Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms

**制作方法**：最原始图片是背景为浅灰色的，分辨率为762\*1000的jepg图片。然后经过resampled 到 51\*73 的彩色图片。然后依次经过7个步骤，最终得到28\*28的灰度图。

1. 将输入转换为png图像

2. 修剪任何与边角像素颜色相近的边缘。“接近度”是由RGB空间中最大可能强度的5%以内的距离来定义的。

3. 对像素进行采样，将图像最长边缘的大小调整为28，即跳过一些行和列。

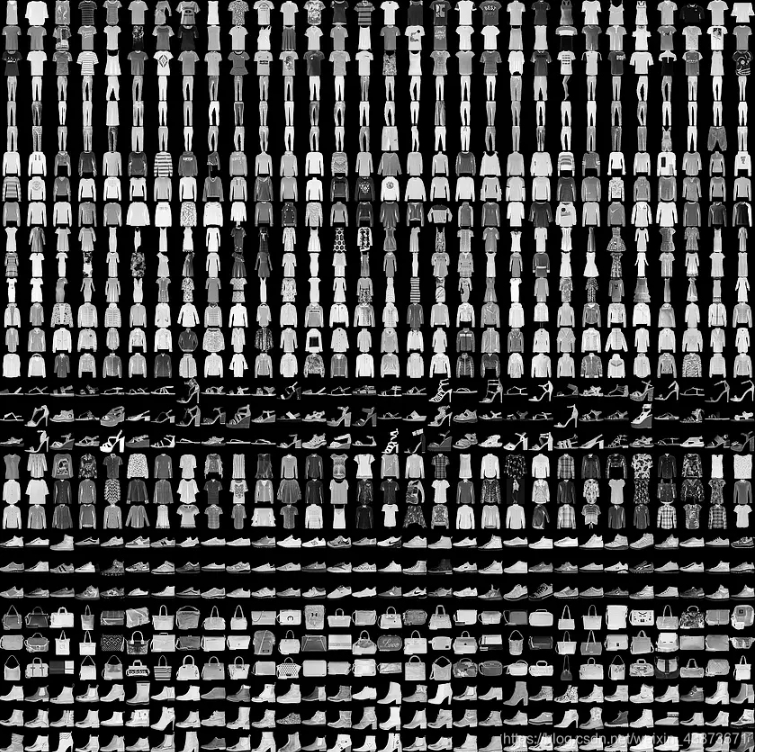
4. 使用半径和标准差为1.0的高斯算子锐化像素，轮廓附近效果越来越好。

5. 将最短的边延长到28，并将图像放在画布的中心。

6. 消去图像的强度。

7. 将图像转换为8位灰度像素

**类别：**Fashion-MNIST不再是抽象符号，而是更加具象化的人类必需品：服装，共10大类,分别是：t-shirt（T恤），trouser（牛仔裤），pullover（套衫），dress（裙子），coat（外套），sandal（凉鞋），shirt（衬衫），sneaker（运动鞋），bag（包），ankle boot（短靴）。



**（二）灰度图**

[灰度图像](https://so.csdn.net/so/search?q=%E7%81%B0%E5%BA%A6%E5%9B%BE%E5%83%8F&spm=1001.2101.3001.7020" \t "https://blog.csdn.net/weixin_43329700/article/details/_blank)避免了彩色带来的复杂性。放大图片中的某一小部分，会发现它是一个二维网络值，亦被称之为具有宽度和高度的数组（单个颜色强度很小的单位）。

这个网格中每个像素颜色都有一个对应的数值，每个像素的值范围是0~255。0 表示黑色 255 表示白色，我们可以通过定位像素网格的横纵坐标来获取某一特定位置的像素值。



1. **HOG特征**

**灰度化：**可选步骤，灰度化是将彩色图片变成灰度图。彩色图片也可以直接处理。不过是分别对三通道的颜色值进行梯度计算，最后选择梯度最大的那个。为简单起见，假设输入为灰度图，同时大小是64\*128。

|  |  |  |
| --- | --- | --- |
|  | 32 |  |
| 20 | A | 30 |
|  | 64 |  |

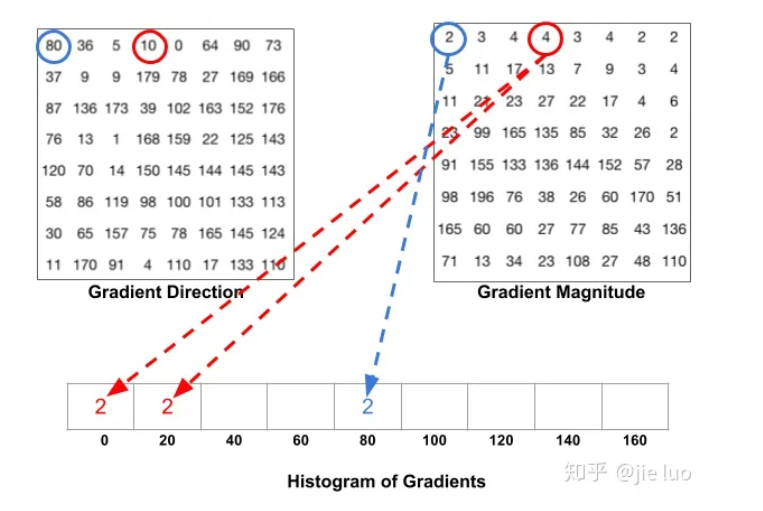
**像素点梯度图：**像素点A的水平梯度 gx=30-20=10,竖直梯度gy=64-32=32.

总的梯度强度值g和梯度方向将按照以下公式计算：

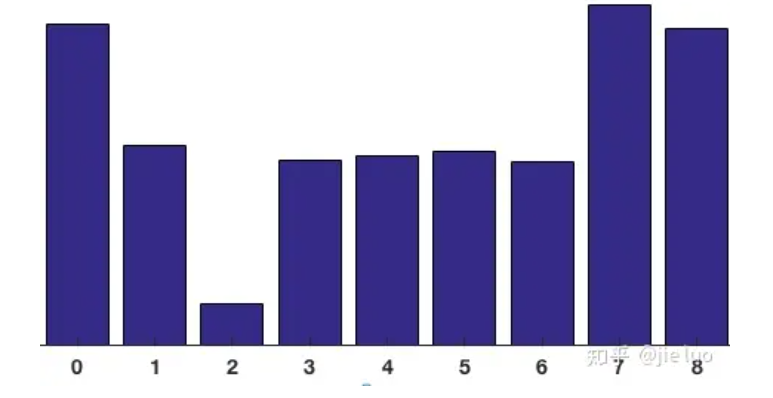


**梯度直方图：**每一个像素点都会有两个值：梯度强度/梯度方向。梯度直方图是在一个8\*8的cell里面计算的。那么在8\*8的cell里面就会有8\*8\*2=128个值，2是包括了梯度强度和梯度方向。通过统计形成梯度直方图，128个值将会变成9个值，大大降低了计算量。

将0-180度分成9个bins，分别是0，20，40...160。然后统计每一个像素点所在的bin，如下图：



左上图是8\*8的梯度方向值，右上图是8\*8的梯度强度值，下图是9个bins。先看两个蓝色圈圈。因为蓝圈的方向是80度，大小是2，所以该点就投给80这个bin；再看两个红色圈圈。因为红色圈圈的方向是10，大小是4，因为10距离0点为10，距离20点为也为10，那么有一半的大小是投给0这个bin，还有一半的大小（即是2）投给20这个bin。那么统计完64个点的投票数以后，每个bin就会得到一个数值，可以得到一个直方图，在计算机里面就是一个大小为9的数组。从直方图可以看到，更多的点的梯度方向是倾向于0度和160度，也就是说这些点的梯度方向是向上或者向下，表明图像这个位置存在比较明显的横向边缘。因此HOG是对边角敏感的，由于这样的统计方法，也是对部分像素值变化不敏感的，所以能够适应不同的环境



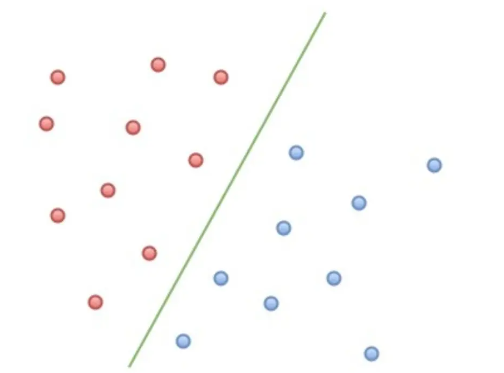
**归一化：**向量的每一个值除以向量的模长，目的是降低光照的影响。绿色方块是8\*8大小的cell，蓝色方块就是由4个cell组成的block。作者提出要对block进行normalize。那么由于一个cell就会有大小为9的vector，四个cell就有36大小的vector。对block进行normalize就是对这大小为36的vector进行归一化。而每一个block将按照上图篮框移动的方式进行迭代截取。



**HOG特征向量：**每一个16\*16大小的block将会得到36大小的vector。那么对于一个64\*128大小的图像，按照上图的方式提取block，将会有7个水平位置和15个竖直位可以取得，所以一共有7\*15=105个block，所以我们整合所有block的vector，形成一个大的一维vector的大小将会是36\*105=3780。得到HOG特征向量，就可以用来可视化和分类了。

1. **SVM算法**

**线性可分：**在二维空间上，两类点被一条直线完全分开叫做线性可分，可以推广至n维，三维空间对应二维平面分割，多维空间对应超平面分割

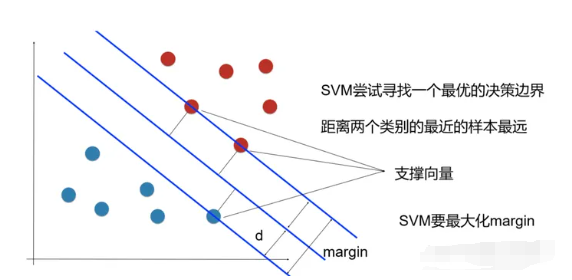


### **最大间隔超平面：**以最大间隔把两类样本分开的最佳超平面，使分类更准确，具体要求：

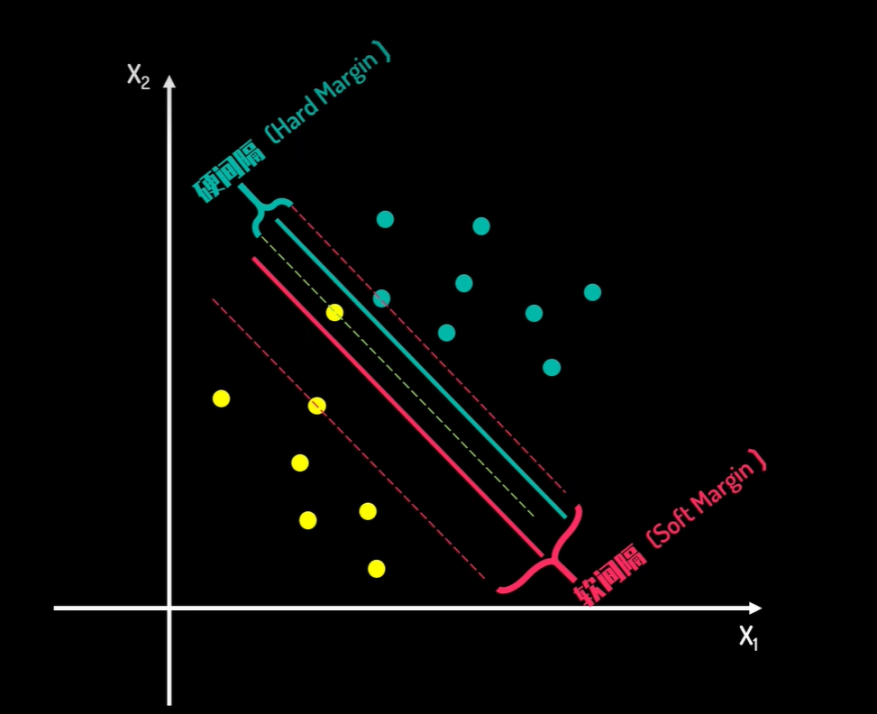
两类样本分别分割在该超平面的两侧；

两侧距离超平面最近的样本点到超平面的距离被最大化

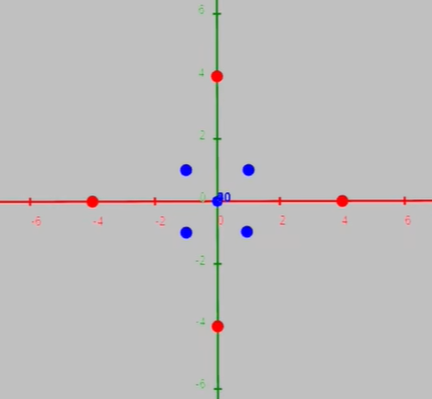
**支持向量：**样本中距离超平面最近的一些点



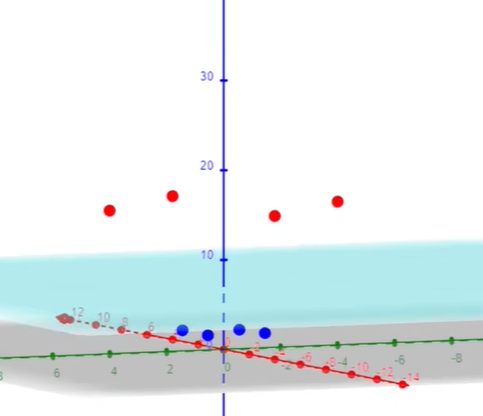
**硬间隔和软间隔：**当两组数据是完全线性可分，我们可以找出一个决策边界使得训练集上的分类误差为0，这两种数据就被称为是存在”硬间隔“的。（考虑所有数据点，包括异常点）当两组数据几乎是完全线性可分的，但决策边界在训练集上存在较小的训练误差，这两种数据就被称为是存在“软间隔”（通过一定规则忽略异常点，考虑针对大多数点的分割）



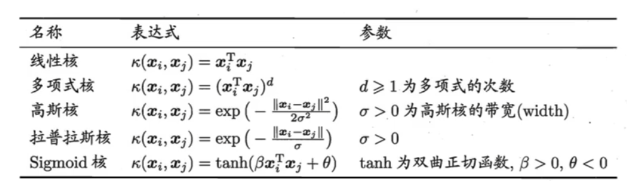
**核函数：**一个从低维空间到高维空间的映射，而这个映射可以把低维空间中线性不可分的两类点变成线性可分的。下图所示的两类数据在二维平面中，这样的数据本身就是线性不可分的，此时我们就要思考该如何把这两类数据分开：



当我们通过一定的映射，将二维数据升为三维数据，就成为线性可分的数据



**常见核函数：**



**线性核和多项式核：**

1）这两种核的作用也是首先在属性空间中找到一些点，把这些点当做base，核函数的作用就是找与该点距离和角度满足某种关系的样本点。

2）样本点与该点的夹角近乎垂直时，两个样本的欧式长度必须非常长才能保证满足线性核函数大于0；而当样本点与base点的方向相同时，长度就不必很长；而当方向相反时，核函数值就是负的，被判为反类。即，它在空间上划分出一个梭形，按照梭形来进行正反类划分。

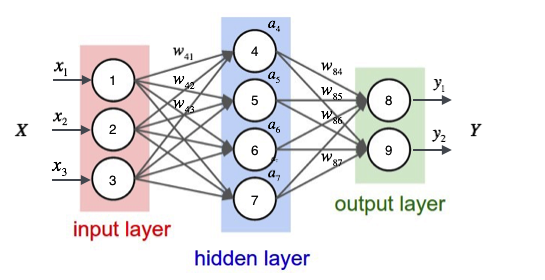
RBF核：

1）高斯核函数就是在属性空间中找到一些点，这些点可以是也可以不是样本点，把这些点当做base，以这些base为圆心向外扩展，扩展半径即为带宽，即可划分数据。

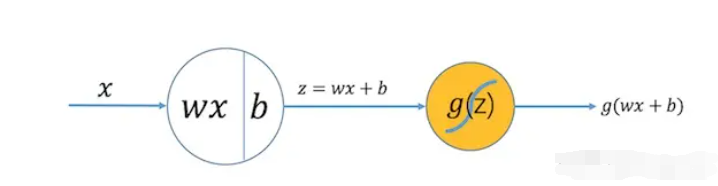
2）换句话说，在属性空间中找到一些超圆，用这些超圆来判定正反类。

**（五）前馈神经网络**

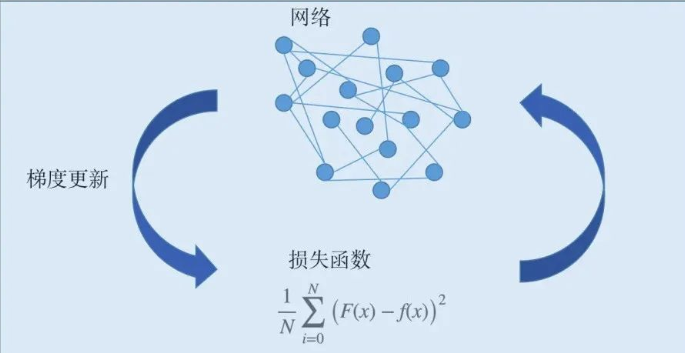
**结构：**输入层，即输入x的那一层；输出层，即输出y的那一层；隐藏层，输入层和输出层之间不管隔了多少层都叫隐藏层。



**激活函数：**假设输入特征只有一个特征，那么上图的输出就是个线性函数。你也可以将隐藏层设为多个节点，算出来的输出仍然是关于的线性函数。所以如果激活函数是线性的，神经网络叠再多层也没什么意义，和最基本的线性回归是一模一样的。因此将激活函数应用到隐藏层上，可以增加非线性因素，解决线性模型表达能力不足的缺陷。



**损失函数：**损失函数用于量化模型预测与真实值之间的差异。它是预测值与真实值之间差距的计算方法，并通过深度学习框架（如PyTorch、TensorFlow）进行封装。



**训练过程：**

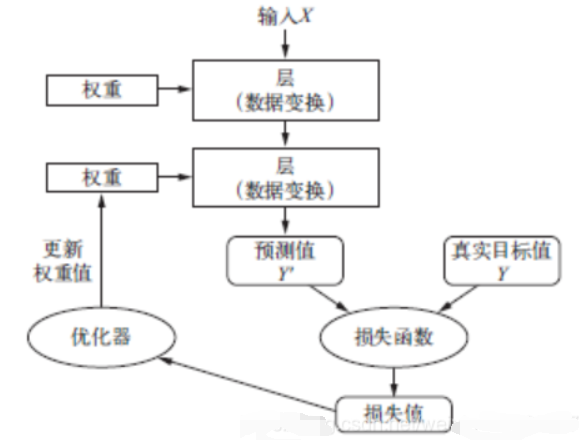
1）抽取训练样本x和对应的目标y组成数据批量

2）在x上运行网络（这一步被称为前向传播），得到预测值y\_pred

3）计算网络在这批数据上的损失，用于衡量y\_pred和y之间的距离

4）计算损失相对于网络参数的梯度（一次反向传播）

5）将参数沿着梯度的反方向移动一点（步长），从而使得这批数据上的损失减小一点，这种方法就是小批量随机梯度下降

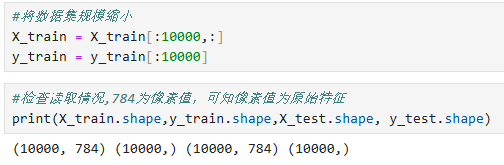


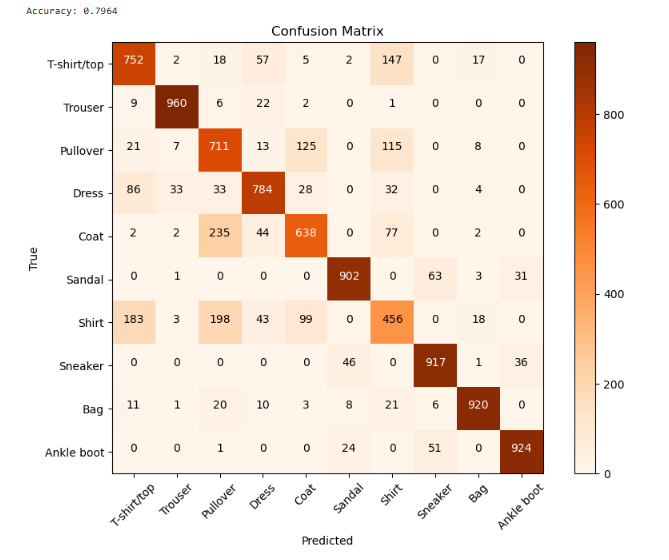
1. **问题**

**（一）运行时间长**

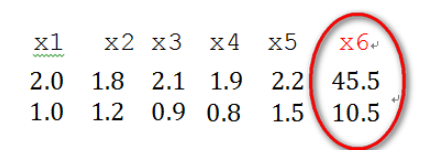
无论是使用SVM还是前馈神经网络进行训练，当使用完整的训练集（60000张图片）时，训练时间很长，但是也并没有占用很多内存，最终长时间不能生成结果。

**解决办法一：缩减数据集规模。**首先将训练集切片至10000，再去执行，发现可以在较短时间内生成结果，因此排除代码问题。

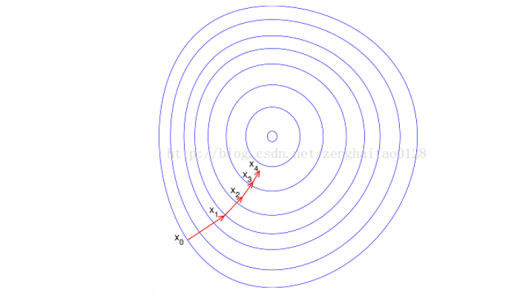
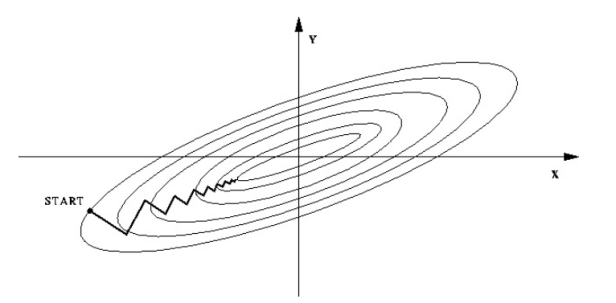




**解决办法二：标准化处理**。归一化或者标准化的目的就是使得预处理的数据被限定在一定的范围内（比如[0,1]或者[-1,1]），从而消除奇异样本数据导致的不良影响。譬如，下面为具有两个特征的样本数据x1、x2、x3、x4、x5、x6,其中x6这个样本的两个特征相对其他样本而言相差比较大，因此，x6认为是奇异样本数据。



奇异样本数据的存在会引起训练时间增大，同时也可能导致无法收敛，因此，当存在奇异样本数据时，在进行训练之前需要对预处理数据进行归一化。如果不进行归一化，那么由于特征向量中不同特征的取值相差较大，会导致目标函数变“扁”。这样在进行梯度下降的时候，梯度的方向就会偏离最小值的方向，走很多弯路，即训练时间过长。如果进行归一化以后，目标函数会呈现比较“圆”，这样训练速度大大加快，少走很多弯路。



**（二）参数调优**

神经网络可以设置层数，节点数，激活函数，求解器等多个参数，不同组合下其训练的效果并不相同，如果和SVM算法进行对比的话，理论上应该选取具有最优参数的神经网络与SVM进行对比，如何实现？

**解决办法：网格搜索。**网格搜索的基本思想是通过穷举搜索指定参数的所有可能组合，以找到在验证集上表现最佳的参数集。对于神经网络，常见的超参数包括：隐藏层大小（hidden\_layer\_sizes），学习率（learning\_rate\_init），激活函数（activation），优化算法（solver）最大迭代次数（max\_iter）

**工作原理：**

定义参数网格：确定需要优化的超参数及其可能的取值范围。

生成参数组合：生成所有可能的参数组合。

评估模型性能：对于每一组参数组合，使用交叉验证评估模型性能。

选择最佳参数组合：根据交叉验证的平均性能指标，选择表现最好的参数组合。



1. **训练及验证结果**
2. **准确度分析：**

**SVM**不同的求解器类型和参数设置，其对应的准确度如下：

|  |  |  |
| --- | --- | --- |
| 准确度 | 线性核函数 | 多项式核函数 |
| 像素值特征 | 0.837 | 0.8755 |
| HOG特征 | 0.8052 | 0.8414 |

**结论一：像素值特征的效果优于HOG特征**，主要原因可能有以下几点：

1、简单图像结构：Fashion-MNIST的数据集包含的图像相对简单，每个图像是28x28像素的灰度图，表示10类不同的服饰。这些图像中的对象通常是居中的，并且背景简单。因此，直接使用像素值作为特征就已经包含了足够的区分信息。

2、低分辨率：Fashion-MNIST图像的低分辨率（28x28像素）意味着像素值特征的维度是784，尽管维度较高，但在当前的计算能力下仍然可以有效处理。而HOG特征在提取过程中可能丢失一些关键细节，尤其是对于低分辨率图像。

3、HOG特征的维度通常比原始像素值要低。尽管这在某些情况下有助于减少计算复杂度和过拟合，但对于Fashion-MNIST这样简单的任务，较高维度的像素值特征反而可能提供更多的区分信息，有助于分类器的性能。

4、不同的特征类型可能需要不同的超参数设置。例如，HOG特征提取后的数据可能需要不同的SVM参数设置（如C值、gamma值等）来达到最佳性能。如果没有针对HOG特征进行细致的超参数调整，可能导致其性能不如直接使用像素值。

**结论二：多项式核函数的效果要好于线性核函数**，主要原因可能有以下几点：

1. 数据的非线性可分性：Fashion-MNIST数据集中的样本通常不是线性可分的。线性核函数假设数据在高维空间中是线性可分的，这对某些复杂的数据集来说是一个限制。相比之下，多项式核函数能够映射数据到更高维的多项式特征空间，使得在这个空间中更容易找到一个超平面来分离不同类别的样本

2. 多项式核函数的灵活性：多项式核函数通过参数调整（如多项式的度数）可以灵活地控制模型的复杂度。较低度数的多项式核可以捕捉到较简单的模式，而较高度数的多项式核则可以捕捉到更复杂的模式。这种灵活性使得多项式核函数在处理复杂数据时比线性核函数更有优势。

3. 适应复杂模式：Fashion-MNIST数据集中的服饰类别（如T恤、裤子、鞋子等）具有不同的形状和结构，这些模式往往是非线性的。多项式核函数能够更好地适应这些复杂的模式，而线性核函数则可能不足以捕捉到这些模式的所有细节。

**前馈神经网络：**进行训练，准确度如下，并将其与svm的准确度进行对比

|  |  |
| --- | --- |
| 类别 | 准确度 |
| 前馈神经网络 | 0.8888 |
| Svm | 0.8755  0.8414  0.837  0.8052 |

**结论三：前馈神经网络准确度略优于svm。**其原因可能是：

1. 非线性特征学习能力

FNN能够通过多层结构自动学习输入数据的非线性特征。每一层都可以对前一层的输出进行复杂的非线性变换，从而捕捉到数据的复杂模式。SVM尽管使用核函数（如高斯核、RBF核）可以让SVM处理非线性数据，但它的非线性学习能力有限，尤其是在高维数据集上，可能无法捕捉到数据的复杂模式。

2. 适应性

前馈神经网络FNN可以通过改变网络结构（如增加层数和神经元数量）来适应不同的任务和数据集。它的灵活性使其在处理各种类型的数据时具有优势。尽管SVM的核函数可以调整以适应不同的任务，但其整体结构相对固定，缺乏灵活性。

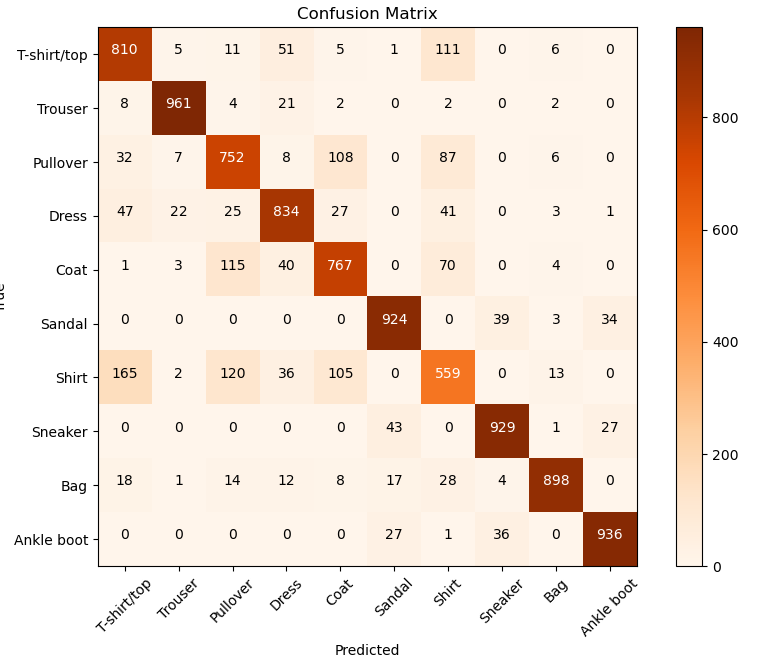
3.处理大规模数据

FNN在处理大规模数据时，利用并行计算和GPU加速可以更有效地训练模型。它能够处理大量的输入数据并从中学习复杂的模式。SVM在处理大规模数据时，训练时间和内存需求都会急剧增加，尤其是使用复杂的核函数时，这可能成为瓶颈。

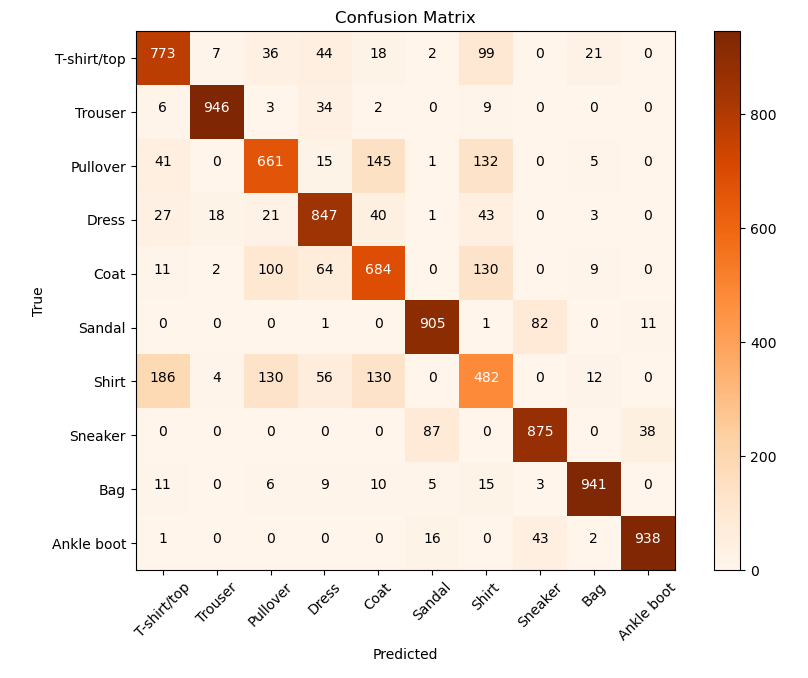
4. 泛化能力

FNN通过正则化技术（如Dropout、L2正则化）和早停（Early Stopping）等方法，可以有效地防止过拟合，提高泛化能力。SVM的泛化能力依赖于核函数的选择和参数调优，但在一些复杂数据集上可能不如FNN。

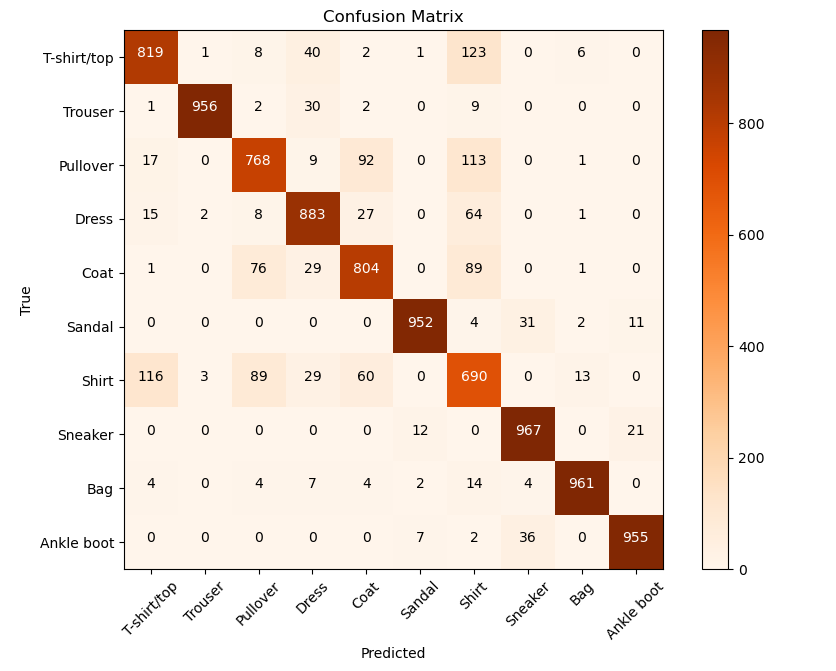
**（二）混淆矩阵分析**



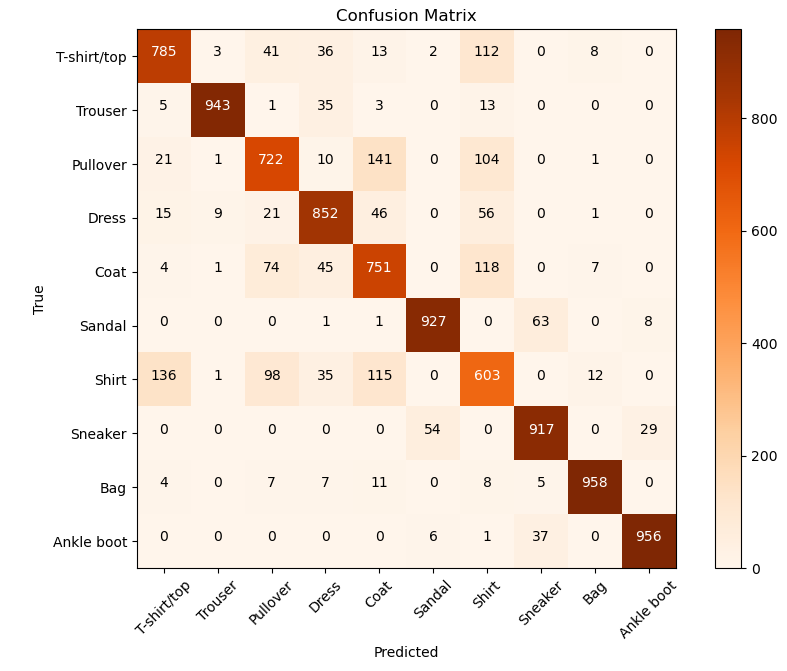
SVM+LINEAR+像素值



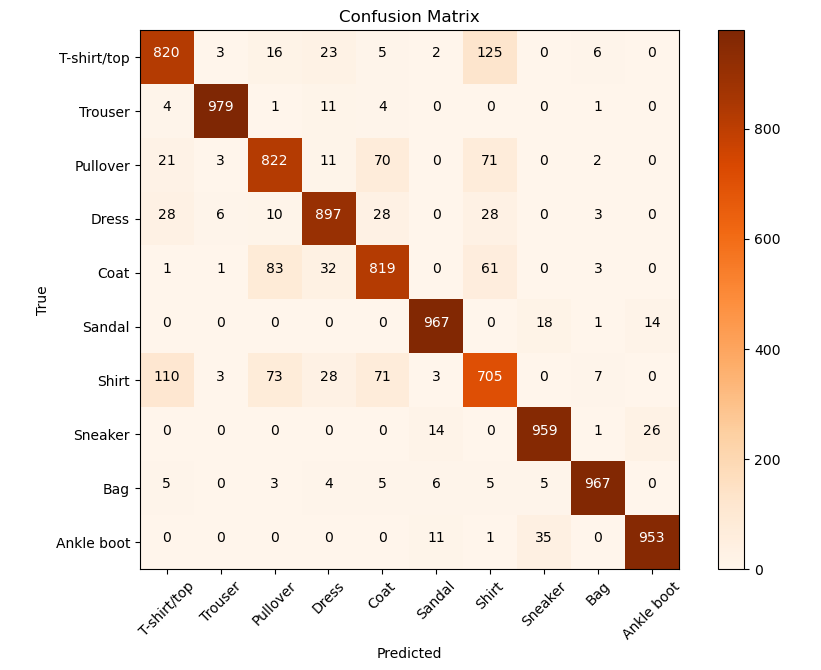
SVM+LINEAR+HOG



SVM+POLY+像素值



SVM+POLY+HOG



前馈神经网络

**准确率和召回率：**每个类别的准确率表示正确分类该类别的比例。通过观察分类报告中的准确率，可以确定哪些类别的分类效果较好，哪些较差。每个类别的召回率表示该类别中被正确分类的样本比例。低召回率表明模型对该类别的识别能力较差。

**类别间混淆：**T-shirt/top 和 Shirt：这两个类别容易混淆，因为它们在视觉上比较相似。混淆矩阵中会显示 T-shirt/top 和 Shirt 之间的误分类数量。

Pullover 和 Coat：这两个类别也可能因为形状和外观的相似而被误分类。Sneaker 和 Ankle boot：这两个类别同为鞋类，有可能在某些角度或特定情况下被误分类。

**改进:**可以增加这些容易混淆类别的样本数量，以提供更多的训练数据。提高图像的分辨率或使用更多的预处理技术（如数据增强），以帮助模型更好地区分不同类别。尝试不同的特征提取方法（如更高级的HOG参数，或使用深度学习方法自动提取特征）来提高分类性能。