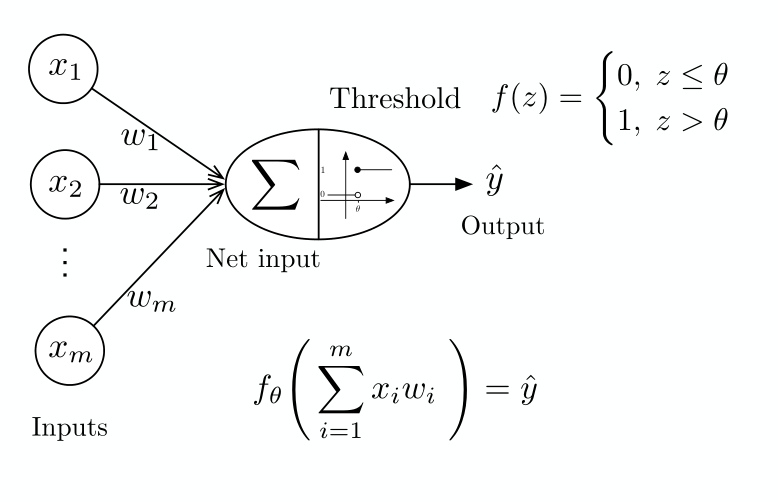
对于感知器这一单层神经网络，看过PDF教程文件和进行coding训练后。

我的理解是：

结合数学层面进行理解，感知器的组成为



可以看到，这里有的input为多个x，这里的多个x对应到机器学习中正是数据集输入X的每一维特征，且这里还有对应的w1,w2...wm，这里在数学上是给每一个x加上了一个权重系数，对应到机器学习中就是输入X的每一维特征添加了权重系数，不同的权重系数表示了此维特征在对于output Y的影响作用大小。

然后我们得到了x和w的累加，同时为了分类过程更加灵活，这里添加了偏置项bias，进而得到了一个加权和。

这里需要说明权重w和偏置b的作用，对应到最终的分割界限上，w可以调节此分界线的方向，而b则可以调节分界线左右移动，假如不添加偏置的话，所有的分界线最终都是经过原点的，然而这对应到现实问题往往就无法解决了。

所以在添加了偏置b以后，分类针对实际问题会变得更加灵活且准确。

之后是加权和经过一个阈值函数f(z)的判断得到输出y，这里的y就是模型的预测值。而阈值函数的结果的0和1正是刚好对应了分类的过程。这里的θ是根据偏置bias来确定的。

所以这就是感知器的结构，但需要注意的是这只是其静态的结构，其真正起作用还是需要多次的更新迭代才能达到分类的目的。

跳出其单次的静态结构，那么感知器是如何运作的呢？

首先需要知道的是作为一种单层神经网络，感知器在训练的过程中是在不断更新的，这个更新正是其学习的过程。

不同于之前学过的其它模型，感知器对于数据集的训练是一个个进行训练的，以此次做的例子为例，训练集有75个数据点，感知器是将数据点一个一个的输入，所以每个数据点都会单独进入模型训练一次，且每进入一个数据点模型都会更新一次，这里的更新是为了什么呢？

我们可以把感知器比作一个人，这个人每次都对一个数据点进行分类，每得到一个数据点他都会通过判断来决定将这个数据点放在第一堆还是第二堆，但是感知器在物理层次上其此参数是静态的，所以他需要确定一组参数来尽可能将所有的数据点都能准确的分类。所以训练的过程中它的参数是需要针对每一次的数据来进行更新的，或者说它力求让自己的参数决定的模型能够尽可能学习到“数据中的分类原则”，这样在测试集中就可以得到尽量高的得分。

所以总的来说，感知器的参数的更新过程其实是针对每一个输入数据点的协同过程。

所以最终经过训练完的模型是比刚开始的模型要“聪明”很多的，这也正是学习的目的。

然后来具体说说参数是怎么更新的呢？

在每一次数据点跑完以后，都会得到一个预测值，通过将预测值和真实值进行比对，来进行判断，假如两者判断的类别一致，那么就不需要更新参数。而如果两者的类别不一致，就需要更新，这里的更新在物理层面上就相当于这个数据点的类别判断错了，它本来属于第一堆，我把它放到第二堆了，这时我就需要把这个数据点从第二堆拿回第一堆，所以可以说这是一个不断更正的过程，而这个更新的过程在数学层面上，就是通过真实值与预测值之间的误差来进行更新的，用权值向量和输入向量的点积来进行判断，假如判断错了，就需要将调整权值向量。

所以可以说单层神经网络横向看包含两个过程，即正向传播和反向传播，其中正向传播得到预测值，而反向传播则是经过“纠正”判断以后来反过来更新感知器的参数。

而从纵向来看，单层神经网络又是一个多次训练的过程，这里训练的次数正是对应于数据点的个数，通过不断的更新，才使得神经网络最终更加“强壮”。

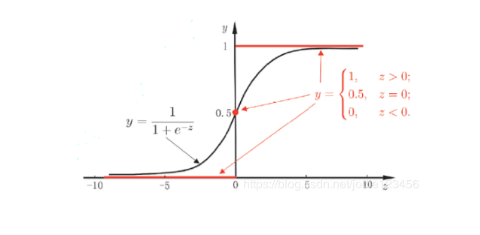
以上就是感知器整体的运行过程和原理。

那么，接下来再说说最终画二维决策边界的原理。

对于二维决策边界的绘制，散点的绘制自然是不需要赘述，关键是如何绘制边界？

需要注意的是，决策边界是假设函数的属性，由参数确定，而不是由数据集的特征确定。

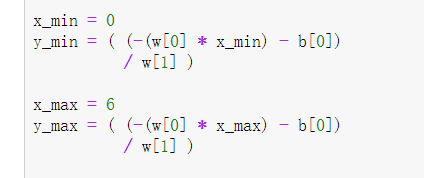
这里使用logistic逻辑回归决策边界。 Logistic回归假设该函数内的阈值函数为S型函数。 根据其性质，可以看出Z的正负只能区分两种类型的数据点，并且当Z取0时，函数值对应于0.5，恰好是数据集的分界线。



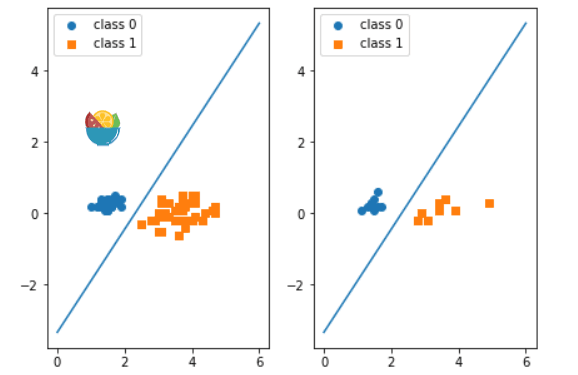
有了这一判断条件，我们就可以画出函数图像了，对应到此问题中就是w1x1+w2x2+b的值（加权和）为0。

具体的做法就是，由于边界是线性的，所以需要两个点来确定这一折线，为了让所有数据点都能够得以区分，所以分别取Xmax和Xmin，之后Y的表达式就可以根据上述公式变形得到了。

需要注意的是，这里的X和Y对应的就是特征1和特征2，所以此做法是将上述公式转换为两个特征之间的关系了。其中w和b用的是最后一次更新的，这在上一步已经得到。



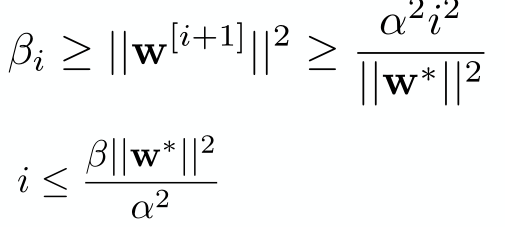
有了这些参数，我们就可以确定数据集的决策边界了。



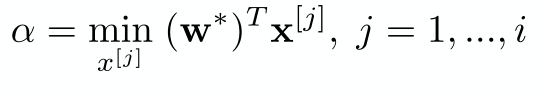
感知器的代码实现部分和说明在ipynb中有更详细的说明。

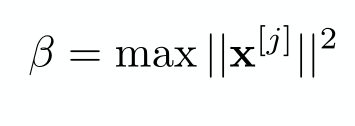
针对感知器的课程，我的问题是：

在证明感知器在有限步数会收敛的过程中，为什么通过以下公式可以说由于迭代次数i有一个上界，我们可以得出结论，权重只改变有限次，并且如果类是线性可分的，则会收敛？



其中





而对于老师上次提出的非线性边界的问题：

其画法的原理和线性边界类似，具体的判断条件或者说函数形式根据感知器内部的权值和的公式来进行判断，在代码上的实现还在研究之中。