Assignment2 报告

一、 使用线性回归进行分类

1.最小平方算法

对于最小平方算法，每个类别有自己的线性模型描述即$y = w_k^T x + w_{k0}$,设 $\hat{x} = (1, \ x^T)^T$，对于数据集$\{x_n, t_n\}$，为了方便计算，令其中一个类别的$t = 1$，另一个类别为$t = -1$

平方和误差函数为

$$E_D(\widehat{W}) = \frac{1}{2} Tr\{(\hat{X}\widehat{W} - T)^T(\hat{X}\widehat{W} - T)\}$$

当$E_D(\widehat{W})$取最小值即$\frac{\partial E_D(\widehat{W})}{\partial \widehat{W}}$为 0 时，W为

$$\widehat{W} = (\hat{X}^T \ \hat{X})^{-1} \hat{X}^T T$$

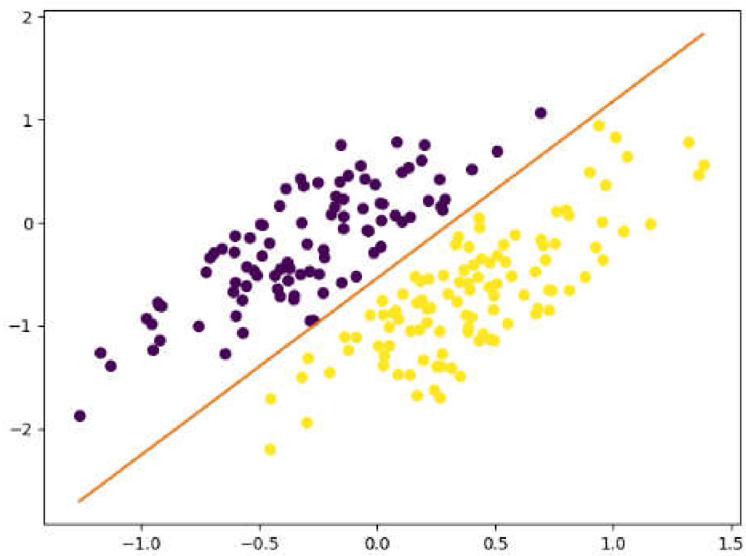
判别函数为

$$y = \widehat{W} \hat{x}$$

通过 **get_linear_seperatable_2d_2c_dataset** 得到了x后，计算出$\widehat{W}$。

$\widehat{W}$ =[-0.51867256,1.65455779,-0.96409225]。准确率为 100%

判别函数和样本点如图所示。



2.感知器算法

对于输入向量x，和最小平方方法一样得到$\hat{x}$，然后构造一个一般的线性模型$y = f(w^T x)$。考虑误差函数

$$E_p(w) = -\sum_{n \in M} w^T x_n t_n$$

其中M为误分类的集合。

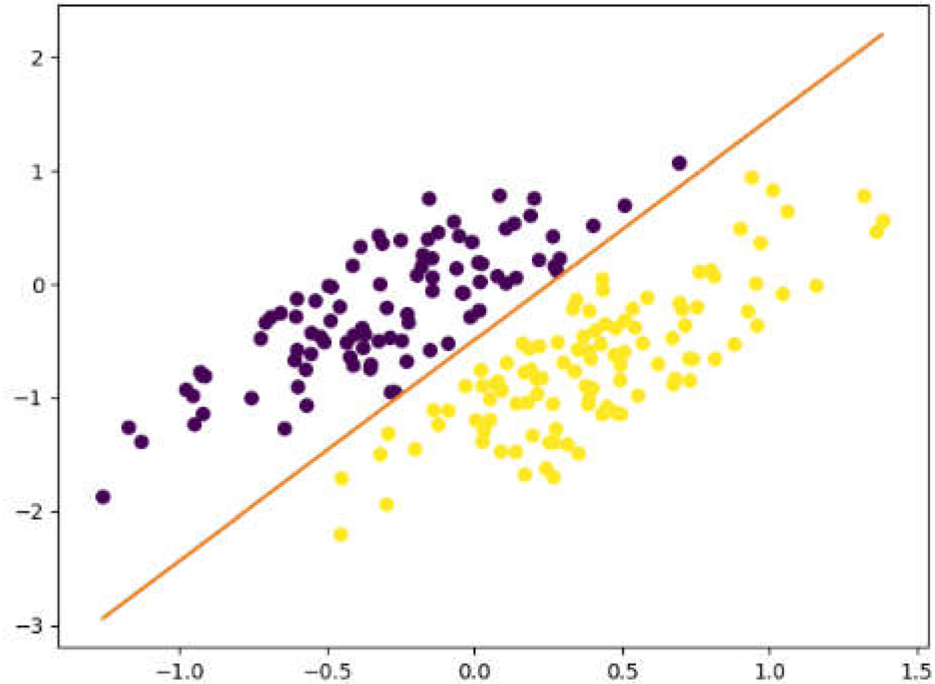
我们采用随机梯度下降法，在这里误差函数对w求导

$$\frac{\partial E_p(w)}{\partial w} = -\sum_{n \in M} x_n t_n$$

每一次误分类是权向量w的变化为

$$w^{t+1} = w^t - \alpha \frac{\partial E_p(w)}{\partial w}$$

设置学习率α为 0.1，迭代 10000 次后得到 W，其中一次w为[-0.25567304,1.01376203,-0.52116001] ,准确率为 100%。判别函数和样本点如图所示。



二、 文本分类

1. 字符串处理

对于一个字符串，逐位判断，忽略所有标点，将 string.whitespace 转化为空格，将大写字母转化为小写字母，得到一个新的字符串。然后对空格进行分割得到一个含有单词的列表，其中无视空字符串。对于每个单词，用字典记录该单词的出现次数。对训练集所有的文本处理后得到所有出现次数大于 10 次的单词。用名为 vocabulary 的字典记录这些单词并进行编号。对每个文本，建立一个长度为D的向量，其中D表示单词总数。对每个文本对应的单词列表，当该单词出现在 vocabulary 中且编号为 i 时，将第 i 维的数字变为 1。于是每个文本都建立出一个向量。

2.对损失函数求导

L对$w_{i,j}$求导

$$\frac{\partial L}{\partial W_{i,j}} = \frac{-\frac{1}{N}\sum_{n=1}^{N} y_n \log \widehat{y_n} + \left|\left|W\right|\right|^2}{\partial W_{i,j}} = -\frac{1}{N}\sum_{n=1}^{N}\frac{\partial y_n \log \widehat{y_n}}{\partial W_{i,j}} + 2\lambda W_{i,j}$$

$$\log \widehat{y_n} = \log \mathrm{softmax}(W^T x_n + b) = \log \frac{e^{W^T x_n + b}}{\sum_{i=1}^{D} e^{W_i^T x_n + b}} = W^T x_n + b - \log \sum_{i=1}^{D} e^{W_i^T x_n + b}$$

$$\frac{\partial y_n \log \widehat{y_n}}{\partial W_{i,j}} = \frac{\partial \sum_{k=1}^{D}(y_{n,k}(w_k^T x_{n,k} + b - \log \sum_{i=1}^{D} e^{W_i^T x_n}))}{\partial W_{i,j}} = y_{n,j}(x_{n,i} - \frac{e^{W_i^T x_n + b} x_{n,i}}{\sum_{i=1}^{D} e^{W_i^T x_n + b}})$$

$$= y_{n,j}(x_{n,i} - \widehat{y_j} x_{n,i})$$

当$y_{n,k}=0$时

$$\frac{\partial y_{n,k}(w_k^T x_{n,k}+b-\log\sum_{i=1}^{D} e^{W_i^T x_n})}{\partial W_{i,j}}=0$$

必然存在一个k使得$y_{n,k}=1$时。若$k!=j$

$$\frac{\partial y_{n,k}(w_k^T x_{n,k}+b-\log\sum_{i=1}^{D} e^{W_i^T x_n})}{\partial W_{i,j}}=\frac{\partial-\log\sum_{i=1}^{D} e^{W_i^T x_n}}{\partial W_{i,j}}=\frac{e^{W_i^T x_n+b} x_{n,i}}{\sum_{i=1}^{D} e^{W_i^T x_n+b}}=\widehat{y_j} x_{n,i}$$

若$k=j$,则

$$\frac{\partial y_{n,k}(w_k^T x_{n,k}+b-\log\sum_{i=1}^{D} e^{W_i^T x_n})}{\partial W_{i,j}}=x_{n,i}-\widehat{y_j} x_{n,i}$$

综上所述，可以统一成

$$\frac{\partial\sum_{k=1}^{D}(y_{n,k}(w_k^T x_{n,k}+b-\log\sum_{i=1}^{D} e^{W_i^T x_n}))}{\partial W_{i,j}}=y_{n,j} x_{n,i}-\widehat{y_j} x_{n,i}=\left(y_{n,j}-\widehat{y_j}\right) x_{n,i}$$

所以

$$\frac{\partial L}{\partial W_{i,j}}=-\frac{1}{N}\sum_{n=1}^{N}\left(y_{n,j}-\widehat{y_j}\right) x_{n,i}+2\lambda W_{i,j}$$

同理可得

$$\frac{\partial L}{\partial b_i}=-\frac{1}{N}\sum_{n=1}^{N} y_{n,i}-\widehat{y_i}$$

因为采用向量计算而不是使用显式循环计算。
所以

$$\frac{\partial L}{\partial W}=-\frac{1}{N}\sum_{n=1}^{N}(y-\hat{y})x+2\lambda W$$

$$\frac{\partial L}{\partial b}=-\frac{1}{N}\sum_{n=1}^{N} y-\widehat{y_n}$$

问题 1：需要对偏置项进行 L2 正则化吗？
答：不需要。L2 正则化是因为防止w数值过大导致过拟合，在该问题上，防止个别$w_{i,j}$过大主要是防止极个别单词决定整篇文章的分类。而偏置 b 只起到偏置作用，所有的文章分类都要加上偏置 b，如果偏置 b 的值较大才能起到偏置作用的话，那么进行正则化则会影响偏置起到的效果。因此不需要对偏置项进行 L2 正则化。
问题 2：如何检查梯度计算是否正确？
答：对于导数的定义

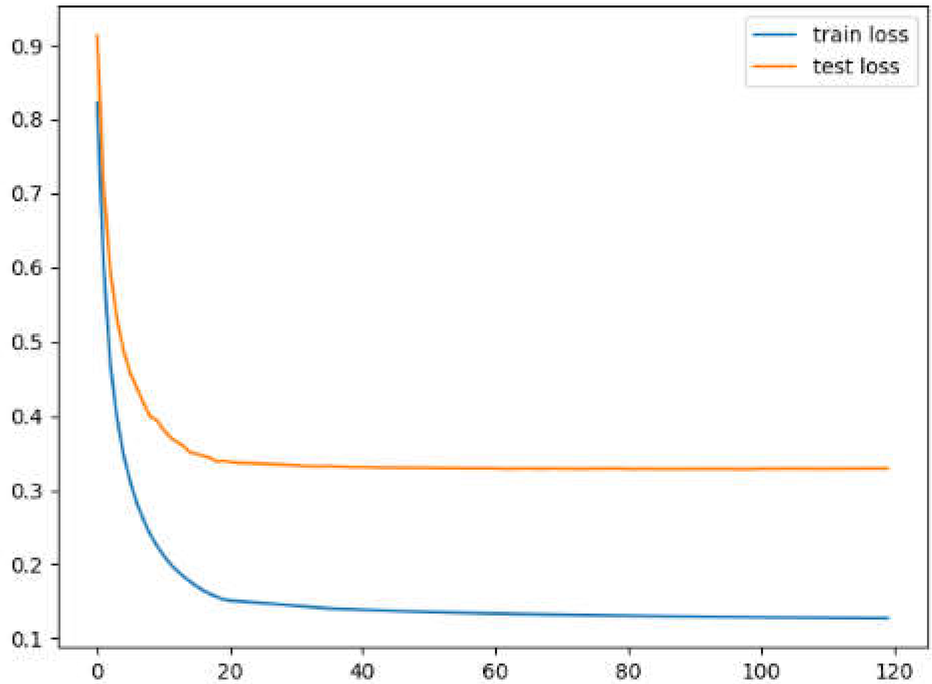
$$\frac{\partial L(w)}{\partial w_{i,j}}=\lim_{\epsilon\to 0}\frac{L\left(w_{i,j}+\epsilon\right)-L(w_{i,j})}{\epsilon}$$

在这里我采用$\epsilon=0.001$来进行验证$\frac{\partial L(w)}{\partial w_{i,j}}$，随机计算一组梯度判断是否正确，过程写在

check_gradient函数里，在程序 230 行取消注释取较小的 batch size 即可检验（较大的 batch size 要计算很久）。事实证明梯度计算没有计算错误。

3. 如何决定学习速度和终止程序

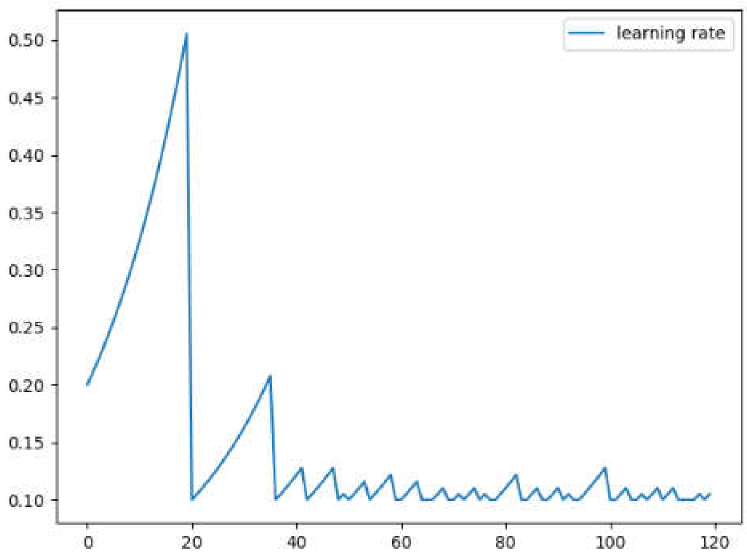
随机梯度下降每批数据大小$N = 200$，正则项$\lambda = 0.001$时，损耗曲线图如下：



(1)如何决定学习速度？

如果学习速度过大，则很难收敛，如果学习速度过小，则训练时间要很长。在这里采用一个方法。当测试集的 loss 函数减小时，可以加大学习速度，在这里去$\alpha_{t+1} = \alpha_t \times 1.05$，当 loss 函数增大时说明学习速率过大需要减小学习速率进行收敛，则$\alpha_{t+1} = 0.1$。初始学习速率$\alpha = 0.2$

学习速率如图所示：



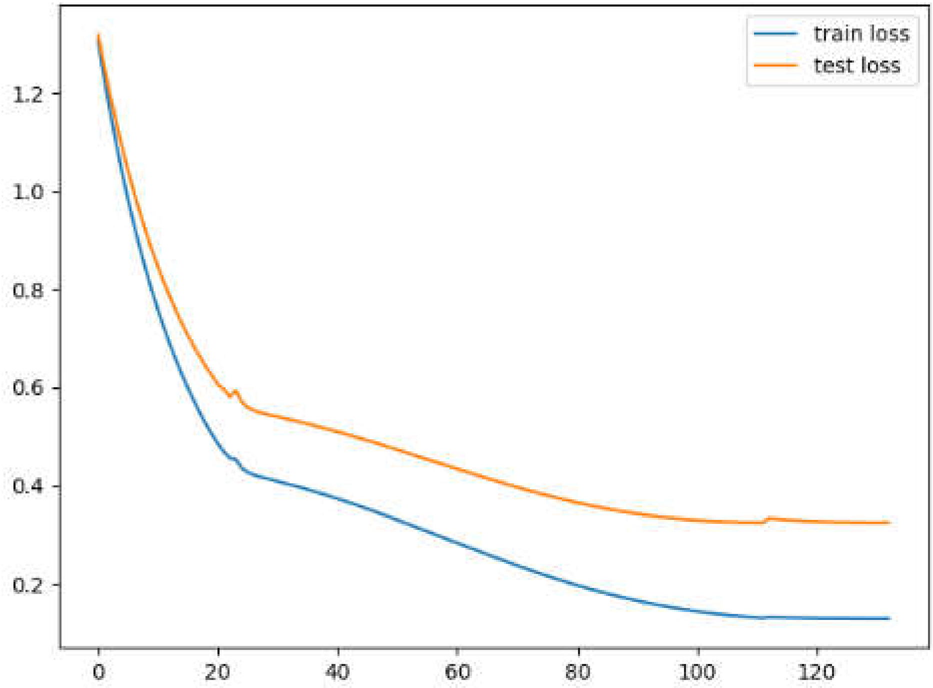
(2)如何决定终止程序？

设定一定轮数的随机梯度下降算法之内没有取到比最小的测试集 loss 函数还小的函数时终止程序。在这里取 20 轮。

4. 不同类别的梯度下降算法

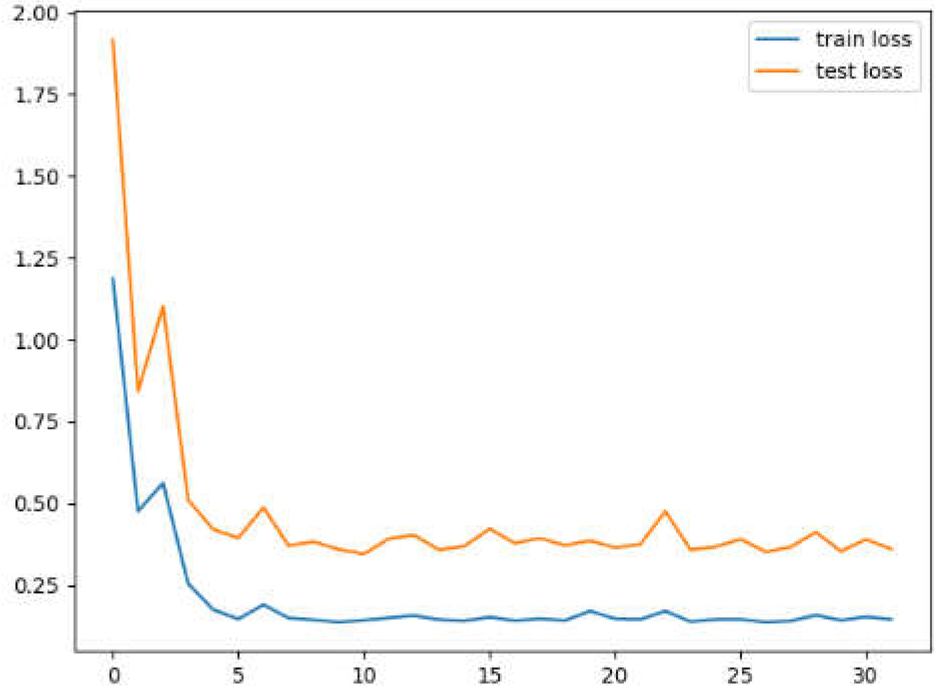
上文采用随机梯度下降算法每批大小为 200，测试集预测分类**准确率为 93.11%**

用所有的训练数据做梯度下降算法时，损失函数如图所示：



测试集预测分类**准确率为 92.98%。**

当随机梯度下降算法每批大小为 **1 个**时，损失函数如图所示：



测试集预测分类**准确率为 93.18%。**

（1）通过另外两种梯度下降方法观察到了什么？

总体的预测分类准确率相近。当采用所有数据进行梯度下降时，损失函数收敛速度很慢，而每批只取一个收敛速度很快。因为所有数据分批处理同样的测试集，每批越小一轮就进行越

多轮次的梯度下降算法，所以很快就能收敛，反之就收敛越慢。但是每批越小后面很容易因为不同数据特征差异过大，导致损失函数不稳定。

（2）三种不同梯度更新更新策略的优缺点？

取所有数据进行梯度下降算法

优点：损失函数收敛后起伏较小，权重取值不会很差。

缺点：占用大量内存资源，损失函数收敛时间较长。

每批只取一个数据进行梯度下降算法

优点：损失函数下降较快，占用内存资源少

缺点：损失函数较难稳定至一个值

每批取适当大小数据进行梯度下降算法

优点：平衡了资源和收敛的问题

缺点：需要额外调参，找到具体适合的批大小比较麻烦

5.结果

损失函数曲线图和准确率在上文二.3 和二.4 有提及。

程序使用方法

执行程序后按照输入提示输入相应参数即可。