实 验 报 告

年 级 专 业： 2021 级人工智能

学 生 姓 名： 史开松

学 号： 20213003754

课 程 名 称： 机器学习

实 验 名 称： 线性规划与逻辑回归

任 课 老 师： 胡祝华

实验报告成绩	
任课教师签名	

海南大学·信息与通信工程学院

School of Information and Communication Engineering, Hainan University

<table>
<tr><td>实验室</td><td>信院 118</td><td>计算机号</td><td colspan="2"></td></tr>
<tr><td>实验
名称</td><td colspan="2">线性回归与逻辑回归</td><td>成绩评定</td><td></td></tr>
<tr><td>所用
软件</td><td colspan="2">pycharm</td><td>教师签名</td><td></td></tr>
<tr><td>实验目的或要求</td><td colspan="4">一、实验目的：
1. 掌握线性回归（linear regression）的基本原理和实现方法，掌握基本术语和概念：序关系（order）、均方差（square error）最小化、欧式距离（Euclideandistance）、最小二乘法（ least square method）、参数估计（ parameterestimation）、多元线性回归（multivariate linear regression）、广义线性回归（generalized linear model）、对数线性回归（log-linear regression）；
2. 掌握逻辑回归（logistic regression）的基本原理和实现方法，掌握基本术语和概念：分类、 Sigmoid 函数、对数几率（log odds / logit）、极大似然法（maximum likelihood method）；
3. 熟悉 LDA 线性判别分析和多分类转二分类的方法
二、实验要求：
1. 按照实验步骤独立完成实验。
2. 整理并上交实验报告和实验源程序。
3. 考核：以学生的实验报告情况和做实验时的表现为考核依据。</td></tr>
<tr><td>实验内容</td><td colspan="4">一、线性回归实验
<u>原理</u>：线性回归算是回归任务中比较简单的一种模型，它的模型结构可以表示如下：
$f(x)=w^T x^*$， $x^*=\lceil x^T,1\rceil^T$，$x \in R^n$，所以 $w \in R^{n+1}$， w 即是模型需要学习的参数。
步骤 1：导入 numpy 库，并将当前项目根目录加入解释器的搜索路径中。
步骤 2：伪造 100 个样本数据集。
步骤 3：把生成的样本数据集用散点图画出来，同时画出真实的直线。
步骤 4：参数的求解方法
（1）原理：
损失函数：利用等式 $y=3x+2$ 我造了一些伪数据，并给 x 添加了一些噪声数据，线性</td></tr>
</table>

	回归的目标即在只有 x,y 的情况下，求解出最优解：$w=[3,2]^T$；可以通过 MSE（均方误差）来衡量 $f(x)$ 与 y 的相近程度： $$L(w)=\sum_{i=1}^{m}(y_i-f(x_i))^2=\sum_{i=1}^{m}(y_i-w^Tx_i^*)^2=(Y-X^*w)^T(Y-X^*w) \quad (1)$$ 这里 m 表示样本量，本例中 $m=100$，x_i,y_i 表示第 i 个样本，$X^*\in R^{m\times(n+1)},Y\in R^{m\times1}$，损失函数 $L(w)$ 本质上是关于 w 的函数，通过求解最小的 $L(w)$ 即可得到 w 的最优解： $w^*=arg\min_w L(w)$ **方法一：直接求闭式解 (一维的情况下参考 ppt)** 而对 $\min L(w)$ 的求解很明显是一个凸问题（海瑟矩阵 $X^{*T}X^*$ 正定），我们可以直接通过求解 $\frac{dL}{dw}=0$ 得到 w^*，梯度推导如下： $$\frac{dL}{dw}=-2\sum_{i=1}^{m}(y_i-w^Tx_i^*)x_i^*=-2X^{*T}(Y-X^*w) \quad (2)$$ 令 $\frac{dL}{dw}=0$，可得：$w^*=(X^{*T}X^*)^{-1}X^{*T}Y$，实际情景中数据不一定能满足 $X^{*T}X$ 是满秩（比如 $m<n$ 的情况下，w 的解有无数种)，所以没法直接求逆，我们可以考虑用如下的方式求解： $$X^{*+}=\lim_{\alpha\to0}(X^{*T}X^*+\alpha I)^{-1}X^{*T} \quad (3)$$ 上面的公式即是 Moore-Penrose 伪逆的定义，但实际求解更多是通过 SVD 的方式： $$X^{*+}=VD^+U^T \quad (4)$$ 其中，U,D,V 是矩阵 X^* 做奇异值分解（SVD）后得到的矩阵，对角矩阵 D 的伪逆 D^+ 由其非零元素取倒数之后再转置得到，通过伪逆求解到的结果有如下优点： （1）当 w 有解时，$w^*=X^{*+}Y$ 是所有解中欧几里得距离 $\|w\|_2$ 最小的一个； （2）当 w 无解时，通过伪逆得到的 w^* 是使得 X^*w^* 与 Y 的欧几里得距离 $\|X^*w^*-Y\|_2$ 最小 **方法二：梯度下降求解** 但对于数据量很大的情况，求闭式解的方式会让内存很吃力，我们可以通过随机梯度下降法（SGD）对 w 进行更新，首先随机初始化 w，然后使用如下的迭代公式对 w 进行迭代更新： $$w:=w-\eta\frac{dL}{dw} \quad (5)$$ （2）模型训练（利用 SGD 求解） 前面推导出了 w 的更新公式，接下来编码训练过程， 迭代结束后参数就计算出来了。 步骤 5：根据训练得到的参数 w 进行可视化展示。 步骤 6：查看 loss 的变化。通过图只管的查看 loss 值是否随着迭代次数的增加趋于平稳。 （3）线性回归模块的封装，以便于可被重复使用

简单封装线性回归模型，并放到 my_models.linear_model 模块便于后续使用，并在该两个目录下添加__init__.py 文件。

该文件的作用：标识该目录是一个 python 的模块包（module package），如果你是使用 python 的相关 IDE 来进行开发，那么如果目录中存在该文件，该目录就会被识别为 module package。实际上，如果目录中包含了 __init__.py 时，当用 import 导入该目录时，会执行 __init__.py 里面的代码。因此，我们可以在__init__.py 指定默认需要导入的模块，有时候我们在做导入时会偷懒，将包中的所有内容导入，比如 from mypackage import *。因此，该文件就是一个正常的 python 代码文件，可以将初始化代码放入该文件中。

（4）基于封装模块的测试，命名为 linear_regression_test.py，并放到 tests 目录下。有三种测试方式。

1）直接调用封装对象的 SGD 方法进行训练与预测；

2）使用闭式解求解参数的方法进行测试；

3）用 sklearn 中的 linear_model 模块中的 LinearRegression 类测试。

二、逻辑回归实验

步骤 1：导入实验所需的相关资源和模块。

步骤 2：实验原理

逻辑回归简单来看就是在线性回归模型外面再套了一个 sigmoid 函数。

$$\delta(t)=\frac{1}{1+e^{-t}}$$

而将 t 替换为线性回归模型 $w^T x^*$（这里 $x^*=[x^T,1]^T$）即可得到逻辑回归模型：

$$f(x)=\delta(w^T x^*)=\frac{1}{1+e^{-(w^T x^*)}}$$

我们可以发现：Sigmoid 函数决定了模型的输出在 $(0,1)$ 区间，所以逻辑回归模型可以用作区间在 $(0,1)$ 的回归任务，也可以用作 $\{0,1\}$ 的二分类任务；同样，由于模型的输出在 $(0,1)$ 区间，所以逻辑回归模型的输出也可以看作这样的“概率”模型：

$$P(y=1|\ x)=f(x)$$
$$P(y=0|\ x)=1-f(x)$$

所以，逻辑回归的学习目标可以通过极大似然估计求解：$\prod_{j=1}^{n} f(x_j)^{y_j}(1-f(x_j))^{(1-y_j)}$，即使得观测到的当前所有样本的所属类别概率尽可能大；通过对该函数取负对数，即可得到交叉熵损失函数：

$$L(w)=-\sum_{j=1}^{n} y_j log(f(x_j))+(1-y_j)log(1-f(x_j))$$

这里 n 表示样本量，$x_j \in R^m$，m 表示特征量，$y_j \in \{0,1\}$，接下来的与之前推导一样，通过梯度下降求解 w 的更新公式即可：

$$\frac{\partial L}{\partial w}=-\sum_{i=1}^{n}(y_i-f(x_i))x_i^*$$

所以 w 的更新公式：

$$w:=w-\eta\frac{\partial L}{\partial w}$$

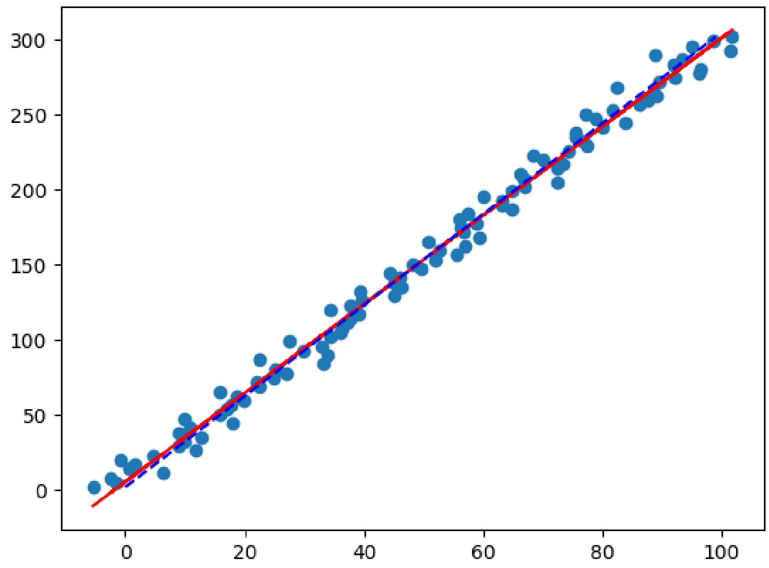
步骤 3：与线性回归类似，模型训练的代码及封装如下所示。代码命名为 logic_regression.py，并放到 my_models\linear_model 目录下。

步骤 4：编写测试程序，命名为 logic_regression_test.py，并放到 tests 目录下。构建数据集，训练模型并进行测试。

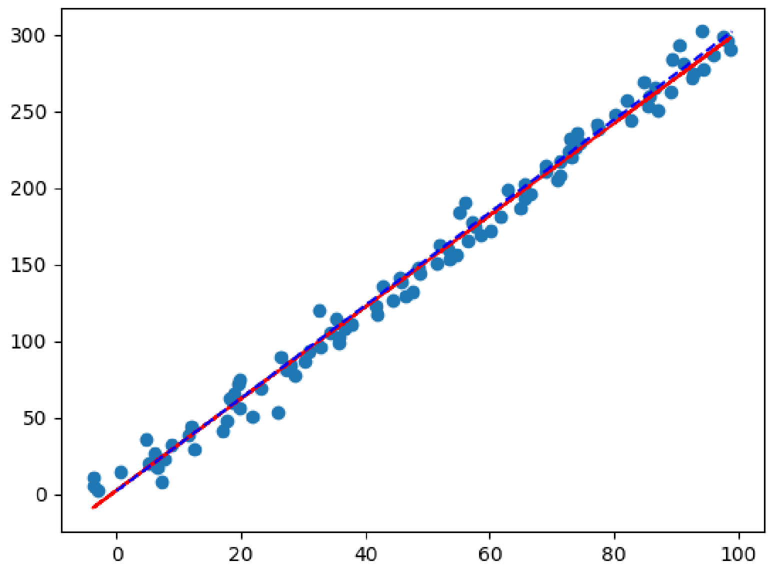
实验结果

一、线性回归实验

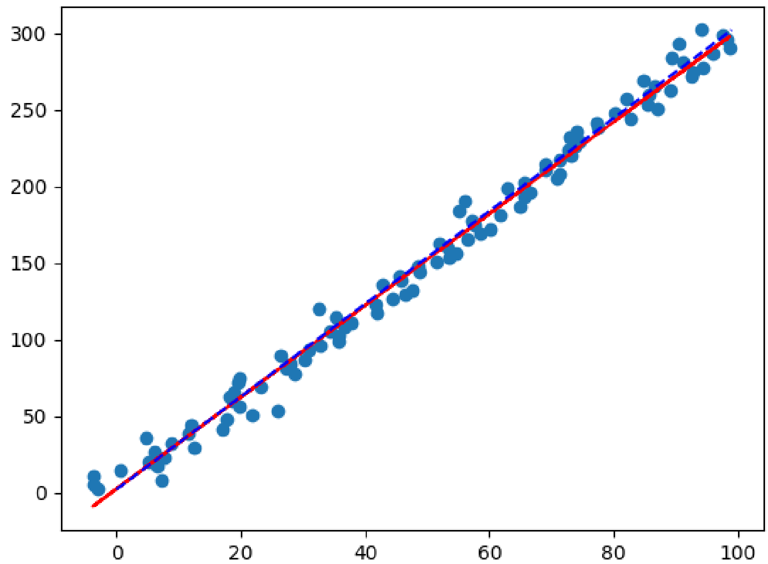
1.直接调用封装对象的 SGD 方法进行训练和预测。



2.使用闭式解求解参数的放大进行测试。

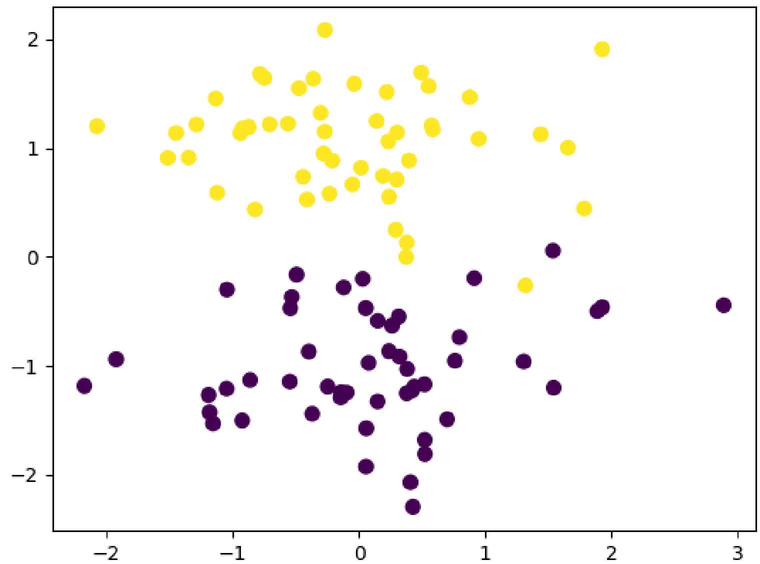


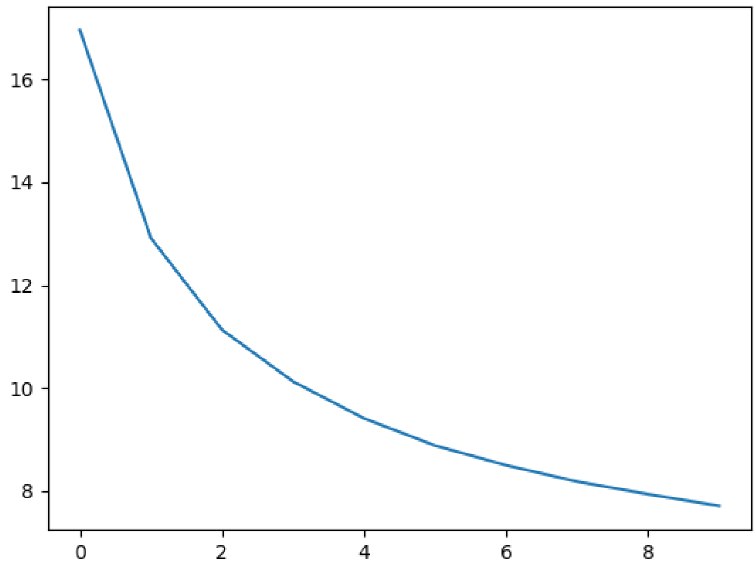
3. 用 sklearn 中的 linear_model 模块中的 LinearRegression 类测试。

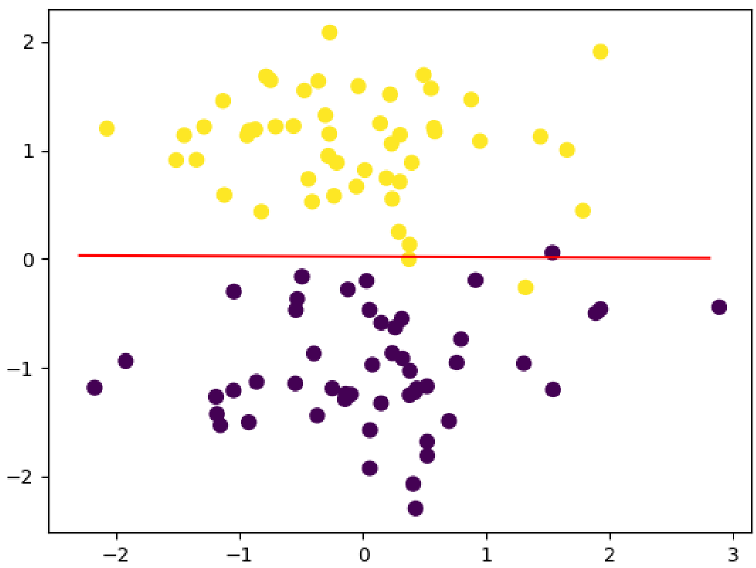


二、逻辑回归实验

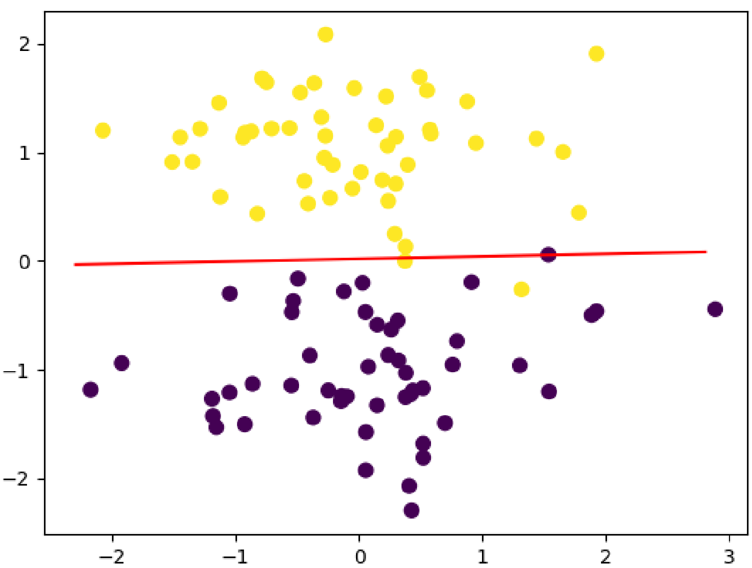
1.







2.与 sklearn 中的逻辑回归模型对比

<table>
<tr>
<td></td>
<td>


实验代码如下：[..\..\deep.rar](..\..\deep.rar)
</td>
</tr>
<tr>
<td>心
得
体
会</td>
<td>本次实验主要学习了线性回归模型与逻辑回归模型的构建、训练与预测，同时将模型进行封装编译后续的使用。</td>
</tr>
</table>