作业 1：线性模型和支持向量机

清华大学软件学院
机器学习, 2023 年秋季学期

1 介绍

本次作业需要提交说明文档（PDF 形式）和 Python 的源代码。注意事项如下：

- 本次作业线性模型和支持向量机各占 50 分，另有部分附加题，若总得分超过 100 分，则按照 100 分截断。
- 作业按点给分，因此请在说明文档中按点回答，方便助教批改。示例如下：
  2.2.1 $J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x_i) - y_i\right)^2 + \cdots$。
- 除非特别说明，本次作业不能直接使用机器学习的开源库，例如 sklearn、pytorch 等。
- 你需要熟练掌握 numpy 及其广播（broadcast）机制，使用 for 循环实现的矩阵运算不会得到编程部分的分数。
- 请提供易懂的注释，若代码的可读性过差，会酌情扣除对应题的分数。
- 在 PDF 中记录你在写程序的哪些模块时与他人有过交流，与他人交流需具体到姓名（网络论坛上的交流可填写用户名），参考网络资料需具体到链接。
- 不要使用他人的作业（含代码和文档），也不要向他人公开自己的作业，否则处罚很严厉，会扣至-100（倒扣本次作业的全部分值）。

2 线性模型与梯度下降 (50pt)

在本题中，你将使用梯度下降法（Gradient Descent）实现岭回归（Ridge Regression）算法。

2.1 特征归一化 (3pt)

在实际应用中，当数据的不同维特征差异很大时，梯度下降的收敛速度会变得很慢。此外，使用正则化时，具有较大绝对值的特征对正则化有更大的影响。因此，我们需要进行特征归一化。一种常见方法是执行仿射变换，将**训练集**中的所有特征值映射至 $[0, 1]$，对验证集上的每个特征也需

要使用相同的仿射变换。你需要修改函数 `feature_normalization` 实现特征的归一化。

2.2 目标函数与梯度 (15pt+3pt)

在**线性回归**中，我们考虑线性函数的假设空间 $h_\theta : \mathbf{R}^d \to \mathbf{R}$,

$$h_{\theta,b}(x) = \theta^T x + b,$$

其中 $\theta, x \in \mathbf{R}^d$, 为了书写和编程的方便，我们通常为 x 添加一个额外的维度，该维度始终是一个固定值，例如 1，用于消除 h_θ 的偏移量参数 b。此时，我们可以将待优化的函数写成:

$$h_\theta(x) = \theta^T x,$$

其中 $\theta, x \in \mathbf{R}^{d+1}$。我们需要找到合适的 θ 最小化:

$$J_{\mathrm{MSE}}(\theta) = \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x_i) - y_i\right)^2,$$

其中 $(x_1, y_1), \ldots, (x_m, y_m) \in \mathbf{R}^{d+1} \times \mathbf{R}$ 是训练数据。

岭回归是使用 L_2 正则化的线性回归，其目标函数是

$$J(\theta) = J_{\mathrm{MSE}}(\theta) + \lambda R(\theta) = \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x_i) - y_i\right)^2 + \lambda \theta^T \theta,$$

其中 λ 是正则化参数，它控制正则化程度。

1. 将训练数据的特征记作 $X = (x_1, x_2, \ldots, x_m)^T \in \mathbf{R}^{m \times (d+1)}$ ，即 X 的第 i 行是 x_i^T, 训练数据的输出记作 $y = (y_1, y_2, \ldots, y_m)^T \in \mathbf{R}^m$，请写出 $J(\theta)$ 的矩阵形式。
2. 修改函数 `compute_regularized_square_loss`，给定 θ，计算的 $J(\theta)$。
3. 请写出 $J(\theta)$ 对 θ 梯度的矩阵形式。
4. 修改函数 `compute_regularized_square_loss_gradient`，给定 θ，计算 $J(\theta)$ 的梯度。
5. **(附加题)** 为了检查梯度的实现是否正确，可以用数值法来计算梯度。可导函数的方向导数可以通过下述公式计算，
$$\lim_{\varepsilon \to 0} \frac{J(\theta + \varepsilon h) - J(\theta - \varepsilon h)}{2\varepsilon}$$
在实际操作中，我们可以选择一个较小的 $\varepsilon > 0$，通过逼近方向导数来逼近梯度。更具体地，可以在每个坐标方向上首先取 $h = (1, 0, 0, \ldots, 0)^T$ 计算第一维的梯度，然后取 $h = (0, 1, 0, \ldots, 0)^T$ 得到第二维的梯度，以此类推。将每一维的梯度合起来，就得到 $J(\theta)$ 在 θ 处的梯度。根据提示，请实现函数 `grad_checker`。(3pt)
6. 在岭回归问题中，我们可能希望偏置项 b 不被正则化。一种可行的方法是修改目标函数 $J(\theta)$，将偏置项从其他参数中分离出来，使其不受正则化影响。另一种可行的方法是将 x 添加的额外维度置为一个非常大的数 B，而非 1，从而减小偏置项受正则化的影响。请说明为何将额外维度置为一个更大的数可以降低正则化对偏置项的影响。

2.3 梯度下降 (15pt)

1. 在最小化 $J(\theta)$ 时，假设取 θ 到 $\theta + \eta h$ 的一步，其中 $h \in \mathbf{R}^{d+1}$ 是前进方向（不一定是单位向量），$\eta \in (0, \infty)$ 是步长。请用梯度写出目标函数值变化的近似表达式 $J(\theta + \eta h) - J(\theta)$,

并思考 h 为哪一前进方向时目标函数下降最快。

2. 令 η 为步长，写出梯度下降中，更新 θ 的表达式。
3. `main` 函数已经加载了数据，将其拆分成了一个训练集和验证集，并完成了归一化。你现在需要修改 `batch_gradient_descent` 实现**梯度下降**算法，使得模型可以在训练集上进行训练。
4. 选择合适的步长：固定 $\lambda = 0$，从 0.1 的步长开始，尝试各种不同的固定步长（至少包括 0.5、0.1、0.05 和 0.01），记录哪个步长收敛最快，哪个步长会导致发散，并绘制目标函数在不同步长下随着训练时间变化的曲线。

2.4 随机梯度下降 (12pt)

当训练数据集非常大时，梯度下降算法需要遍历整个数据集，因此效率较低。实际中用的往往是**随机梯度下降**（SGD）算法。令 $f_i(\theta) = (h_\theta(x_i) - y_i)^2$ 为第 i 个数据点上的平方误差，则

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} f_i(\theta) + \lambda\theta^T\theta$$

在 SGD 中，每一步会随机采样一个小批量（batch）$\{(x_{i_k}, y_{i_k})\}_{k=1}^{n}$ 来计算目标函数

$$J_{\mathrm{SGD}}(\theta) = \frac{1}{n}\sum_{k=1}^{n} f_{i_k}(\theta) + \lambda\theta^T\theta$$

和梯度方向，其中 $n \ll m$ 并且 i_k 从 $\{1, 2, \ldots, m\}$ 中独立同分布采样。

1. 请给出 $J_{\mathrm{SGD}}(\theta)$ 对应的梯度 $\nabla J_{\mathrm{SGD}}(\theta)$ 的表达式。
2. 证明随机梯度 $\nabla J_{\mathrm{SGD}}(\theta)$ 是 $\nabla J(\theta)$ 的无偏估计，即证明

$$\mathbb{E}_{i_1,i_2,\ldots,i_n}\left[\nabla J_{\mathrm{SGD}}(\theta)\right] = \nabla J(\theta)$$

（提示：可以利用期望的线性性质）

3. 实现函数 `stochastic_grad_descent`。
4. 随机梯度下降算法的噪音较大，因此模型较难收敛。你需要选择合适的批大小：固定 $\lambda = 0$，并根据 2.3.4 的结果固定选取一个合适的步长或步长策略，从批大小 1 开始，尝试各种不同的批大小，并记录随着批大小逐渐增大时，训练曲线发生的变化。

2.5 模型选择 (5pt)

L_2 正则化往往可以有效避免过拟合。可以通过验证集上的均方误差 $J_{\mathrm{MSE}}(\theta)$ 来选择合适的 λ，例如，$\lambda \in \left\{10^{-7}, 10^{-5}, 10^{-3}, 10^{-1}, 1, 10, 100\right\}$。根据 2.4.4 的结果固定选取一组合适的批大小和学习步长（或步长策略），使用梯度下降或随机梯度下降来训练模型，并用表格记录不同超参数 λ 下的模型在验证集上的均方误差。

2.6 （附加题）牛顿法 (3pt+3pt)

模型也可以使用二阶优化方法（例如牛顿法）训练。考虑多元函数 $J(\theta)$ 在 θ_0 处的泰勒展开

$$J(\theta) = J(\theta_0) + \nabla J(\theta_0)^T(\theta - \theta_0) + \frac{1}{2}(\theta - \theta_0)^T\nabla^2 J(\theta_0)(\theta - \theta_0) + o((\theta - \theta_0)^2)$$

忽略二次及以上的项，对上式两端同时求梯度可得

$$\nabla J(\theta) = \nabla J(\theta_0) + \nabla^2 J(\theta_0)(\theta - \theta_0)$$

我们的目标是寻找函数的驻点，即梯度等于 0 的点。令梯度等于 0 可得

$$\theta = \theta_0 - (\nabla^2 J(\theta_0))^{-1} \nabla J(\theta_0) = \theta_0 - H^{-1} g$$

其中 H 是函数 $J(\theta)$ 的 Hessian 矩阵，而 g 则是函数的梯度。此即为一步牛顿迭代的过程。

我们考虑**岭回归**问题

$$J_{\mathrm{MSE}}(\theta) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i)^2 + \lambda \theta^T \theta,$$

1. 请推导 $J_{\mathrm{MSE}}(\theta)$ 对 θ 的海森矩阵 H 的矩阵形式，并证明在 $\lambda > 0$ 的情况下，该矩阵一定非奇异。(3pt)
2. 实现函数 `newton_method`。根据模型在验证集上的的表现，选取合适的正则项 λ 以及步长**(不要求搜索最优的超参数组合)**，绘制函数的训练曲线及验证集上损失函数曲线，并观察牛顿法是否能让模型更快收敛 (提示：可从模型收敛需要的迭代步数以及单步迭代所需时间等方面进行分析)。(3pt)

3 支持向量机 (50pt)

在本题，我们将使用支持向量机（Support Vector Machine）对文本数据进行情绪检测。

3.1 次梯度 (4pt+3pt)

对于 $f : \mathbf{R}^d \to \mathbf{R}$，在 x 如果对于所有 z，

$$f(z) \geq f(x) + g^T(z - x),$$

则 $g \in \mathbf{R}^d$ 为 x 处的**次梯度**（subgradients）。次梯度并不唯一，f 在 x 处的次梯度集合记为 $\partial f(x)$，那么 $g \in \partial f(x)$。大部分时候，例如在进行次梯度下降的时候，我们只需要次梯度集合中的一个次梯度即可，这时也可以写作 $\partial f(x) = g$。

1. 基于以上的定义，你需要给出合页损失函数（Hinge Loss）的次梯度。合页损失如下。

$$J(w) = \max\left\{0, 1 - yw^T x\right\}$$

2. **(附加题)** 设函数 $f : \mathbf{R}^d \to \mathbf{R}$ 的次梯度处处存在，亦即 $\partial f(x) \neq \emptyset$ 对于任意 $x \in \mathbf{R}^d$ 成立。试说明函数 f 是一凸函数[1]。注意 f 为凸是指对任意定义域中的 x, y，以及任意 $\lambda \in (0, 1)$, 有

$$\lambda f(x) + (1 - \lambda) f(y) \geq f(\lambda x + (1 - \lambda) y)$$

(**提示**) 使用反证法。(3pt)

3.2 感知机 (6pt)

给定数据集 $(x_1, y_1), \ldots, (x_n, y_n) \in \mathbf{R}^d \times \{-1, 1\}$，感知器算法希望能找到一个超平面将两个类别完全区分开，即找到 $w \in \mathbf{R}^d$ 使得

$$y_i w^T x_i > 0 \; \forall i \in \{1, \ldots, n\}$$

这意味着所有标签 $y = 1$ 的 x 都在超平面 $\left\{x \mid w^T x = 0\right\}$ 的一侧，并且所有标签 $y = -1$ 的 x 都在另一侧。若这样的超平面存在，则称数据是**线性可分**的。具体算法如算法1所示。

感知机中还定义了一个感知损失，

$$\ell(\hat{y}, y) = \max\{0, -\hat{y}y\}$$

1. 令 $\mathcal{H}$ 是由函数组成的线性假设空间 $x \mapsto w^T x$。考虑使用随机次梯度下降 (SSGD) 最小化感知器损失。证明当选择合适的次梯度，并使用固定步长 1 的时候，这个过程等价于感知机算法。
2. 假设感知器算法返回 w。证明 w 是输入数据的线性组合。即 $w = \sum_{i=1}^n \alpha_i x_i$ 对于某些 $\alpha_1, \ldots, \alpha_n \in \mathbf{R}$。当 $\alpha_i \neq 0$ 时，x_i 称为支持向量（support vector）。

[1]实际上，次梯度处处存在是函数 $f : \mathbf{R}^d \to \mathbf{R}$ 为凸的一个充要条件，具体可参考《Convex Analysis and Nonlinear Optimization》一书中关于次梯度的部分。

Algorithm 1: 感知机算法

输入：训练集 $(x_1, y_1), \ldots, (x_n, y_n) \in \mathbf{R}^d \times \{-1, 1\}$
$w^{(0)} = (0, \ldots, 0) \in \mathbf{R}^d$
$k = 0$ # 迭代次数
repeat
  all_correct = TRUE
  for $i = 1, 2, \ldots, n$ # 遍历数据集
    if ($y_i x_i^T w^{(k)} \leq 0$)
      $w^{(k+1)} = w^{(k)} + y_i x_i$
      all_correct = FALSE
    else
      $w^{(k+1)} = w^{(k)}$
    end if
    $k = k + 1$
  end for
until (all_correct == TRUE)
return $w^{(k)}$

3.3 软间隔支持向量机 (20pt+5pt)

线性可分是一种理想的情况，现实数据集经常会有噪声，无法约束所有的数据点都能被正确分类，因此实际中的支持向量机都会引入软间隔，目标是让不满足约束的数据点尽可能少，即

$$\begin{aligned} \min_{w \in \mathbf{R}^d, b \in \mathbf{R}, \xi \in \mathbf{R}^m} & \frac{\lambda}{2}\|w\|_2^2 + \frac{1}{m}\sum_{i=1}^{m} \xi_i. \\ \text{s.t.} \quad & y_i(w^T x_i + b) \geq 1 - \xi_i \\ & \xi_i \geq 0, \quad 1 \leq i \leq m \end{aligned} \tag{1}$$

1. 求该问题的拉格朗日（Lagrange）方程。
2. 求该问题的对偶形式（Dual Form）。
3. 已知非线性映射 Φ 和其对应的核函数 $k(\cdot, \cdot)$，若在本节中的支持向量机中使用该核函数，请写出对应的原问题和对偶问题。
4. 在求出对偶问题后，使用 Sequential Minimal Optimization (SMO) 算法，可以实现软间隔支持向量机的实际求解。上文中的优化问题也可以等价表示为：

$$\min_{w \in \mathbf{R}^d, b \in \mathbf{R}} \frac{\lambda}{2}\|w\|^2 + \frac{1}{m}\sum_{i=1}^{m} \max\left\{0, 1 - y_i(w^T x_i + b)\right\}. \tag{2}$$

   SVM 求解还有另外一种基于**次梯度下降**的算法。证明软间隔支持向量机中 $J_i(w) = \frac{\lambda}{2}\|w\|^2 + \max\left\{0, 1 - y_i(w^T x_i + b)\right\}$ 的次梯度由下式[2]给出（提示：参考合页损失函数的次梯度）。

$$\partial J_i|_w = \begin{cases} \lambda w - y_i x_i & \text{for } y_i(w^T x_i + b) < 1 \\ \lambda w & \text{for } y_i(w^T x_i + b) \geq 1. \end{cases}, \quad \partial J_i|_b = \begin{cases} -y_i & \text{for } y_i(w^T x_i + b) < 1 \\ 0 & \text{for } y_i(w^T x_i + b) \geq 1. \end{cases}$$

5. 请根据上一问中给出的次梯度方向，写出用随机次梯度下降（SSGD）训练 SVM 的伪代码。

[2] $\partial J_i|_w$ 表示次梯度 ∂J_i 在 w 对应的维度上的分量，$\partial J_i|_b$ 同理。

6. 考虑如下带 l_2 正则项的软间隔支持向量机：

$$\begin{aligned} \min_{w\in\mathbf{R}^d, b\in\mathbf{R}, \xi\in\mathbf{R}^m} & \frac{\lambda}{2}\|w\|_2^2 + \frac{1}{m}\sum_{i=1}^{m}\xi_i^2. \\ \text{s.t.} \quad & y_i(w^T x_i + b) \geq 1 - \xi_i \end{aligned} \tag{3}$$

注意到在该问题中，我们省略了 $\xi_i \geq 0$ 这一约束。请说明这一约束是可以被移除的，也即无论有没有这一约束，问题的最优解不变。

7. **(附加题)** 对于题6中的支持向量机，给出该问题的拉格朗日方程和问题的对偶形式。(5pt)

3.4 情绪检测 (20pt)

我们将使用**线性软间隔支持向量机**进行情绪检测，类别包括：开心（joy）和伤心（sadness）。我们在 `start_code.py` 中已经实现了数据集的加载与预处理，你也可以自己实现。你需要完成

1. 参考 3.3.4 中写出的伪代码，在函数 `linear_svm_subgrad_descent` 中实现 SVM 的随机次梯度下降算法，并在情绪检测数据集上进行训练（提示：可以参考 2.4 节随机梯度下降部分的代码）。
2. 调整超参数，例如批大小、正则化参数 λ、步长、步长衰减策略等，观察训练完成时训练集上准确率与验证集上准确率随着超参数发生的变化，绘制曲线或者表格来记录你的超参数调优过程和结果。（提示：仅需要如实地记录自己的超参数调优过程即可，**不要求完备地或有计划地搜索超参数空间**）
3. 软间隔 SVM 的目标函数2是所谓的 λ-强凸函数，因此存在一种 SGD 算法使得收敛速度为 $O(\frac{\log(T)}{T})$，T 为迭代步数。相比之下，使用 SGD 求解普通凸函数的优化问题时收敛速度一般为 $O(\frac{1}{\sqrt{T}})$。我们给出该收敛速度更快的 SGD 算法如算法2所示。[3]

Algorithm 2: 求解 λ-强凸问题的 SGD 算法

输入：训练集 $(x_1, y_1), \ldots, (x_n, y_n) \in \mathbf{R}^d \times \{-1, 1\}$
目标：求解 $\min_w f(w)$
初始化：$w^{(1)} = \mathbf{0}$
for $t = 1, 2, \ldots, T$ # 迭代步数
  randomly choose x_i, y_i from training dataset
  choose $v_t \in \partial f(w^{(t)}, x_i, y_i)$
  compute $\eta_t = 1/(\lambda * t)$
  update $w^{(t+1)} = w^{(t)} - \eta_t v_t$
end for
return $w^{(T)}$

请基于该算法在函数 `linear_svm_subgrad_descent_lambda` 中实现软间隔 SVM 的优化过程，并记录训练过程中模型在训练集上损失函数或准确率的变化过程。相比你在题1中实现的算法，该算法是否有效加速了模型的收敛过程？试分析原因 (**提示**：可以参考书中相关章节关于收敛速度的详细推导进行分析，例如收敛速度的常数项等)。

[3]我们对算法进行了一些简化，更为一般形式的算法以及收敛速度推导请参考《Understanding Machine Learning: From Theory to Algorithms》一书第 14.4 小节。

4. 在函数 `kernel_svm_subgrad_descent` 中实现基于核函数的 SVM（例如基于线性核或者高斯核），调整相关的超参数，记录它在测试集上的准确率。核函数的引入能提高当前模型的准确率吗？试解释原因。
5. 计算并汇报最终的 SVM 模型在验证集上的准确率，F1-Score 以及混淆矩阵（Confusion Matrix）。