更高效的支持向量机算法实现及其在手写数字识别中的应用——02手动实现SGD优化的软间隔线性SVM

大数据机器学习课程第二次实验项目

叶璨铭 (2024214500) ycm24@mails.tsinghua.edu.cn

2024年11月11日星期一

目录

本文是[更高效的支持向量机算法实现及其在手写数字识别中的应用](./00svm.html)系列文章第02篇——手动实现SGD优化的软间隔线性SVM。

dataset\_dict\_uci\_digits = load\_digits(as\_frame=False)
X\_train, X\_val, X\_test, y\_train, y\_val, y\_test, train\_set, val\_set, test\_set, data\_module, categories = process\_sklearn\_dataset\_dict(dataset\_dict\_uci\_digits, 'all')
dataset\_dict\_full\_mnist = fetch\_openml("mnist\_784", as\_frame=True)
X\_train\_full, X\_val\_full, X\_test\_full, y\_train\_full, y\_val\_full, y\_test\_full, train\_set\_full, val\_set\_full, test\_set\_full, data\_module\_full, categories\_full = process\_sklearn\_dataset\_dict(dataset\_dict\_full\_mnist, 'all')

(1797, 64) float32 (1797,) int64 [0 1 2 3 4 5 6 7 8 9]
1293 144 360
(70000, 784) float32 (70000,) int64 [0 1 2 3 4 5 6 7 8 9]
50400 5600 14000

### 0.1 实现 Hinge Loss+SGD 版本的 Soft Margin Linear SVM

我们现在来实现与from sklearn.linear\_model import SGDClassifier等价的 SVM，但是我们基于PyTorch实现，在GPU上面运行，期望能在大型数据集上比sklearn的实现快。

#### 0.1.1 Hinge Loss 二分类实现

首先回顾二分类情况下，李航《统计学习方法》131页的优化目标：

$$\min\_{w,b}\sum\_{i=1}^{N}\left[1−y\_{i}\left(w⋅x\_{i}+b\right)\right]\_{+}+λ∥w∥^{2}$$

李航随即给出了证明，证明这个无约束最优化问题等价于优化软间隔线性支持向量机的原始问题。我感觉上面的式子有点丑，莫名其妙地引入了没有物理意义的超参数$λ$, 让读者无法与软间隔问题中的$C$联系起来。所以我们使用李航证明中用来说明这两个问题等效性的式子:

$$\min\_{w,b}\frac{1}{2}∥w∥^{2}+C\sum\_{i=1}^{N}\left[1−y\_{i}\left(w⋅x\_{i}+b\right)\right]\_{+}$$

其中 $*i =* + $ 是松弛变量，$\left[\right]\_{+}$ 自动地让松弛变量满足了约束。

我们现在就可以用 PyTorch 实现这个 二分类情况下的 Hinge Loss了。

注意，在下面的代码中我们不会去实现$\frac{1}{2}∥w∥^{2}$!

因为在实际PyTorch项目中，这个是写在 Optimizer的 weight decay里面的。 1/2 其实和 C 、学习率等 只是相对比例关系。

这是因为 PyTorch 的设计哲学中，loss 函数的输入就是 logits 和 labels，不会包括模型的参数，这样才能把逻辑解耦开来。

所以如果我们在loss就实现 l2 regularization，是很麻烦而且没有必要的。

Optimizer 有权限访问到 model 的参数，所以我们待会可以精确控制，只对 weight 而非 bias 做 l2 regularization，从而满足 SVM 的要求。

[source](https://github.com/Open-Book-Studio/THU-Coursework-Machine-Learning-for-Big-Data/blob/main/thu_big_data_ml/svm/handy_crafted/linear.py#L19)

BinaryHingeLoss

 BinaryHingeLoss (C=1.0, squared=False, margin=1.0)

*Binary Hinge Loss. For SVM,* $$\min\_{w,b}rac12∥wVert^{2}+C\sum\_{i=1}^{N}\left[1−y\_{i}\left(w⋅x\_{i}+b\right)ight]\_{+}\right.$$

 *we compute* $$C\sum\_{i=1}^{N}\left[1−y\_{i}\left(w⋅x\_{i}+b\right)ight]\_{+}\right.$$

import torch
import torch.nn as nn
from fastcore.all import store\_attr

class BinaryHingeLoss(nn.Module):
 """
 Binary Hinge Loss.
 For SVM,
 $$
 \min\_{w, b} \frac{1}{2} \lVert w \rVert^2 + C \sum\_{i=1}^N \left[ 1 - y\_i(w \cdot x\_i + b) \right]\_+
 $$
 we compute
 $$
 C \sum\_{i=1}^N \left[ 1 - y\_i(w \cdot x\_i + b) \right]\_+
 $$
 """
 def \_\_init\_\_(self, C=1.0,
 squared = False,
 margin = 1.0,
 ):
 super().\_\_init\_\_()
 store\_attr() # 保存参数到实例变量中

 def forward(self, y\_pred\_logits:torch.Tensor, y\_true:torch.Tensor)->torch.Tensor:
 functional\_margin = y\_true \* y\_pred\_logits # 函数间隔
 how\_small\_than\_required\_margin = self.margin - functional\_margin
 xi = torch.clamp(how\_small\_than\_required\_margin, min=0) # 计算 xi 也就是 松弛变量
 if self.squared:
 xi = xi \*\* 2
 return self.C \* xi.sum()

其中 squared 参数表示是否使用 squared hinge loss。 什么叫做 squared hinge loss 呢？我们做过理论作业，李航《统计学习方法》153页例题 7.3 提到了 (但是书上没有给出Reference)

$$\min\_{w,b}\frac{1}{2}∥w∥^{2}+C\sum\_{i=1}^{N}ξ\_{i}^{2}$$

来取代

$$\min\_{w,b}\frac{1}{2}∥w∥^{2}+C\sum\_{i=1}^{N}ξ\_{i}$$

其实上面那个式子就是 squared hinge loss。

from fastcore.test import test\_eq

# 尝试一下
critierion = BinaryHingeLoss(margin=1, C=1, squared=False)
y\_pred\_logits = torch.tensor([1.0, 2.0, 3.0], requires\_grad=True)
y\_true = torch.tensor([1.0, -1.0, 1.0])
test\_eq(critierion(y\_pred\_logits, y\_true).item(), 1-(-1.0)\*2.0) # 第二个样本分类错误，函数间隔为 -2， 差了 3

#### 0.1.2 Hinge Loss 多分类实现

现在我们已经成功实现 BinaryHingeLoss， 接下来我们要考虑 多分类情况下 要如何变化。

参考

* https://lightning.ai/docs/torchmetrics/stable/classification/hinge\_loss.html
* https://pytorch.org/docs/stable/generated/torch.nn.MultiMarginLoss.html

按照我们上文 理论回顾 一节说的，有两类方法。首先是直接法，具体来说 [Crammer and Singer在论文中](https://jmlr.csail.mit.edu/papers/volume2/crammer01a/crammer01a.pdf) 提出， 对于每一个样本$x\_i, y\_i, t\_i $ (y\_i 是 one hot vector, t\_i 是 标签，假如是第k类也就是$t\_i = k $，$y\_i^{t\_i} = 1, y\_i^{k!=t\_i} = 0$),使用这一个公式来作为多分类的HingeLoss$$ L\_i = \_+ $$ 其中 $\hat{y\_{i}}^{k}=\left(w^{k}⋅x\_{i}+b^{k}\right)$ 是针对第k个类的那一个SVM($w^{k},b^{k}$表示)预测出来的值，不是函数间隔哦，没有乘$y\_{i}^{k}$。

而间接法则是使用 one vs all(a.k.a., rest) 来计算 hinge loss。（one vs one 比较少见）

具体而言，首先 $t\_i = k $，$y\_i^{t\_i} = 1, y\_i^{k!=t\_i} = -1$, $\hat{y\_{i}}^{k}=\left(w^{k}⋅x\_{i}+b^{k}\right)$ 表示 是否为 第k个类。 对于每一个样本i，对于每一个样本k都能计算出一个binary loss：

$$L\_{i}^{k}=\left[1−y\_{i}^{k}\left(w^{k}⋅x\_{i}+b^{k}\right)\right]\_{+}$$

相当于同时在训练K个分类器，所以这些loss都可以加在一起，反向传播训练对于$w^{k},b^{k}$独立操作。

所以推导一下，如果每一个 $1−y\_{i}^{k}\left(w^{k}⋅x\_{i}+b^{k}\right)$ 都 $>0$ 就相当于

$$L\_{i}=\sum\_{k=1}^{K}L\_{i}^{k}≈\left[K−\left(\hat{y\_{i}}^{t\_{i}}−\sum\_{k\ne t\_{i}}^{​}\hat{y\_{i}}^{k}\right))\right]\_{+}$$

但是并不是都>0的，没法这样进去推导，所以还是得老老实实相加。

理论清晰之后，现在我们开始实现代码：

[source](https://github.com/Open-Book-Studio/THU-Coursework-Machine-Learning-for-Big-Data/blob/main/thu_big_data_ml/svm/handy_crafted/linear.py#L53)

MultiClassHingeLoss

 MultiClassHingeLoss (C=1.0, squared=False, margin=1.0,
 strategy:Literal['crammer\_singer','one\_vs\_all']='cra
 mmer\_singer')

*MultiClassHingeLoss*

from fastcore.all import patch
from typing import Literal

Strategy = Literal['crammer\_singer', 'one\_vs\_all']
class MultiClassHingeLoss(nn.Module):
 """MultiClassHingeLoss"""
 def \_\_init\_\_(self, C=1.0,
 squared = False,
 margin = 1.0,
 strategy: Strategy = 'crammer\_singer',
 # \*args, \*\*kwargs
 ):
 super().\_\_init\_\_()
 store\_attr()
 self.binary\_critieria = None
 def forward(self, y\_pred\_logits:torch.Tensor,
 y\_true:torch.Tensor # 并非 one hot 编码，而是 int/long 类型 的 label
 )->torch.Tensor:
 if self.strategy == 'crammer\_singer':
 return self.forward\_crammer\_singer(y\_pred\_logits, y\_true)
 elif self.strategy == 'one\_vs\_all':
 return self.forward\_one\_vs\_all(y\_pred\_logits, y\_true)
 else:
 raise ValueError(f"Invalid strategy: {self.strategy}")
 def forward\_one\_vs\_all(self, y\_pred\_logits:torch.Tensor, y\_true:torch.Tensor)->torch.Tensor:
 num\_of\_classes = y\_pred\_logits.size(1)
 if self.binary\_critieria is None:
 self.binary\_critieria = nn.ModuleList([
 BinaryHingeLoss(C=self.C, squared=self.squared, margin=self.margin)
 for \_ in range(num\_of\_classes)
 ])
 losses = []
 for k, critierion in enumerate(self.binary\_critieria):
 y\_true\_binary = 2 \* (y\_true == k) - 1 # 转换为 -1/1 编码、
 y\_pred\_that\_class = y\_pred\_logits[:, k]
 loss = critierion(y\_pred\_that\_class, y\_true\_binary)
 losses.append(loss)
 return sum(losses)
 def forward\_crammer\_singer(self, y\_pred\_logits:torch.Tensor, y\_true:torch.Tensor)->torch.Tensor: ...

[source](https://github.com/Open-Book-Studio/THU-Coursework-Machine-Learning-for-Big-Data/blob/main/thu_big_data_ml/svm/handy_crafted/linear.py#L93)

get\_max\_values\_without\_true

 get\_max\_values\_without\_true (y\_pred\_logits, y\_true)

\*获取去掉y\_true对应元素后，y\_pred\_logits每行的最大值。

参数: y\_pred\_logits: torch.Tensor, 形状为 (N, K) y\_true: torch.Tensor, 形状为 (N,)

返回: torch.Tensor, 形状为 (N,), 去掉y\_true对应元素后每行的最大值\*

import torch

def get\_max\_values\_without\_true(y\_pred\_logits, y\_true):
 """
 获取去掉y\_true对应元素后，y\_pred\_logits每行的最大值。

 参数:
 y\_pred\_logits: torch.Tensor, 形状为 (N, K)
 y\_true: torch.Tensor, 形状为 (N,)

 返回:
 torch.Tensor, 形状为 (N,), 去掉y\_true对应元素后每行的最大值
 """
 # 将y\_true转换为适当的索引格式
 indices = y\_true.unsqueeze(1).expand\_as(y\_pred\_logits)

 # 创建一个与y\_pred\_logits形状相同的掩码，真实标签位置为False，其余为True
 mask = torch.ones\_like(y\_pred\_logits, dtype=torch.bool)
 mask.scatter\_(1, indices, False)

 # 使用掩码来排除y\_true对应的列，并计算每一行的最大值
 max\_values = y\_pred\_logits[mask].view(y\_pred\_logits.size(0), -1).max(dim=1)[0]

 return max\_values

我们调用已经实现的[BinaryHingeLoss](https://Open-Book-Studio.github.io/THU-Coursework-Machine-Learning-for-Big-Data/coding_projects/P2_SVM/handy_crafted_linear.html#binaryhingeloss)来实现forward\_one\_vs\_all。

为了方便实现 forward\_crammer\_singer 方法，首先我们实现一个辅助函数，

注意 PyTorch中的索引有广播机制，我们刚才编程的时候遇到这个问题，实现了一阵才正确实现逻辑。

下面的代码是说明这个问题的最小重现单元，它的运行结果可能超出你的预期，所以编程的时候我们刚才特别注意了这一点。

#
y\_pred = torch.ones(2, 10)
y\_true = torch.zeros(2).to(int)
y\_pred[:, y\_true] # 很多人以为会是 [1, 1], 实际输出 [[1., 1.], [1., 1.]]

tensor([[1., 1.],
 [1., 1.]])

测试一下函数正确性

y\_pred\_logits = torch.tensor([[1, 2, 3, 4, 5],
 [6, 7, 8, 9, 10],
 [11, 12, 13, 14, 15],
 [16, 17, 18, 19, 20]], dtype=torch.float)
y\_true = torch.tensor([4, 4, 4, 4])

# 函数调用
max\_values = get\_max\_values\_without\_true(y\_pred\_logits, y\_true)
max\_values

tensor([ 4., 9., 14., 19.])

正确。 现在我们继续实现 [MultiClassHingeLoss.forward\_crammer\_singer](https://Open-Book-Studio.github.io/THU-Coursework-Machine-Learning-for-Big-Data/coding_projects/P2_SVM/handy_crafted_linear.html#multiclasshingeloss.forward_crammer_singer), fastcore 提供 patch 扩展了 Python 的能力，可以支持 Mixin 风格的软件编程。

[source](https://github.com/Open-Book-Studio/THU-Coursework-Machine-Learning-for-Big-Data/blob/main/thu_big_data_ml/svm/handy_crafted/linear.py#L118)

MultiClassHingeLoss.forward\_crammer\_singer

 MultiClassHingeLoss.forward\_crammer\_singer (y\_pred\_logits:torch.Tensor,
 y\_true:torch.Tensor)

\*L\_i = \_+\*

@patch
def forward\_crammer\_singer(self:MultiClassHingeLoss, y\_pred\_logits:torch.Tensor, y\_true:torch.Tensor)->torch.Tensor:
 """L\_i = \left[ 1 - (\hat{y\_i}^{t\_i} - \max\_{k \neq t\_i} \hat{y\_i}^k)) \right]\_+"""
 batch\_size, num\_classes = y\_pred\_logits.size()
 y\_true\_one\_hot = torch.eye(num\_classes).to(y\_pred\_logits.device)[y\_true]

 # 计算真实类别的预测值
 y\_true\_logits = (y\_pred\_logits \* y\_true\_one\_hot).sum(dim=1)
 # y\_true\_logits = y\_pred\_logits[:, y\_true]

 max\_other\_logits = get\_max\_values\_without\_true(y\_pred\_logits, y\_true)

 functional\_margin\_differences = (y\_true\_logits - max\_other\_logits)

 # 计算hinge loss
 xi = torch.clamp(self.margin - functional\_margin\_differences, min=0)

 if self.squared:
 xi = xi \*\* 2
 return self.C \* xi.sum()

现在我们简单测试一下上面实现的两个Loss是否计算正确，与理论公式一样。

y\_pred\_logits = torch.tensor([[0.5, -0.5], [-0.5, 0.5]])
y\_true = torch.tensor([0, 1])
critierion = MultiClassHingeLoss(strategy="crammer\_singer")
loss = critierion(y\_pred\_logits, y\_true)
loss

tensor(0.)

上面的结果是正确的，因为 自己这个类别的 预测值 是 0.5, 其他类别的最大预测值是 -0.5, 比它大1, 正好是margin，所以loss为0

critierion = MultiClassHingeLoss(strategy="one\_vs\_all")
loss = critierion(y\_pred\_logits, y\_true)
loss

tensor(2.)

上面的结果也是正确的，这两个数据都是 正确的那个类 自己只有0.5的值，不到1， 而错误的类不到-1，所以有正负类两个样本各有0.5的loss，一共为2。

以上我们实现的Loss在真实情况中会使用哪个组合呢？这也是个值得探讨的问题。

对于多类别的情况，[论文](https://www.csie.ntu.edu.tw/~cjlin/papers/l2mcsvm/l2mcsvm.pdf) 详细研究了 crammer\_singer 方法 + squared hinge loss 的有效性，所以这两者是可以结合的。

#### 0.1.3 SVM 模型与优化算法实现

在 crammer\_singer 方法 和 one vs all 的情况下，SVM的参数$w,b$的大小是 (n\_features, n\_classes), (n\_classes,)，这与多分类的感知机、Linear+Softmax是一样的。

[source](https://github.com/Open-Book-Studio/THU-Coursework-Machine-Learning-for-Big-Data/blob/main/thu_big_data_ml/svm/handy_crafted/linear.py#L148)

HingeSupportVectorClassifier

 HingeSupportVectorClassifier (input\_dim, num\_classes, learning\_rate=0.01,
 weight\_decay=0.5, optimizer\_type=<class
 'torch.optim.sgd.SGD'>, C:float=1,
 squared:bool=False, margin:float=1, strateg
 y:Literal['crammer\_singer','one\_vs\_all']='c
 rammer\_singer', experiment\_index=0)

*Hooks to be used in LightningModule.*

|  | **Type** | **Default** | **Details** |
| --- | --- | --- | --- |
| input\_dim |  |  |  |
| num\_classes |  |  | model related |
| learning\_rate | float | 0.01 |  |
| weight\_decay | float | 0.5 | optimization related |
| optimizer\_type | type | SGD |  |
| C | float | 1 | loss related |
| squared | bool | False |  |
| margin | float | 1 |  |
| strategy | Literal | crammer\_singer |  |
| experiment\_index | int | 0 | experiment related |

from overrides import override
from lightning.pytorch.utilities.types import EVAL\_DATALOADERS, TRAIN\_DATALOADERS, STEP\_OUTPUT, OptimizerLRScheduler
import torch.optim as optim
import lightning as L

# lightning imports

class HingeSupportVectorClassifier(L.LightningModule):
 def \_\_init\_\_(self,
 # model related
 input\_dim, num\_classes,
 # optimization related
 learning\_rate=0.01, weight\_decay=0.5,
 optimizer\_type = optim.SGD,
 # loss related
 C: float = 1,
 squared: bool = False,
 margin: float = 1,
 strategy: Strategy = 'crammer\_singer',
 # experiment related
 experiment\_index=0,
 ):
 super().\_\_init\_\_()
 self.save\_hyperparameters()
 L.seed\_everything(experiment\_index)
 self.model = nn.Linear(input\_dim, num\_classes)
 self.loss\_fn = MultiClassHingeLoss()

 self.example\_input\_array = torch.randn(1, input\_dim)
 self.dummy\_inputs = dict(input\_ids=self.example\_input\_array) # for opendelta and huggingface
 self.automatic\_optimization = True
 # 评价策略
 self.evaluation\_steps\_outputs = dict()

 @override
 def forward(self, image\_tensor:torch.Tensor, \*args, \*\*kwargs)-> torch.Tensor:
 """
 Returns:
 torch.Tensor: the predicted functional margin to each class's decision hyperplane
 """
 return self.model(image\_tensor)

 def predict\_geometric\_margin(self, image\_tensor:torch.Tensor)->torch.Tensor:
 w\_norm\_each\_line = torch.norm(self.model.weight, dim=1)
 return torch.clamp(self(image\_tensor) / w\_norm\_each\_line, min=0)

 def predict\_class(self, image\_tensor:torch.Tensor)->torch.Tensor:
 return torch.argmax(self(image\_tensor), dim=1)

 def forward\_loss(self, image\_tensor: torch.Tensor, label\_tensor:torch.Tensor)->torch.Tensor:
 logits = self(image\_tensor)
 return self.loss\_fn(logits, label\_tensor)

 @override
 def training\_step(self, batch, batch\_idx=None, \*args, \*\*kwargs)-> STEP\_OUTPUT:
 loss = self.forward\_loss(\*batch)
 self.log("train\_loss", loss, prog\_bar=True)
 return loss

from namable\_classify.infra import print\_model\_pretty

test\_model = HingeSupportVectorClassifier(28\*28, 10)
test\_model.print\_model\_pretty()
y\_pred\_logits =test\_model(test\_model.example\_input\_array)
test\_model.predict\_class(test\_model.example\_input\_array)
test\_model.predict\_geometric\_margin(test\_model.example\_input\_array)
# y\_pred\_logits
# test\_model.loss\_fn(y\_pred\_logits, torch.tensor([0]))
test\_model.forward\_loss(test\_model.example\_input\_array, torch.tensor([0]))

Seed set to 0

Mon 2024-11-18 20:38:01.629617

INFO ╭─────────────────────── Model Tree for HingeSupportVectorClassifier ────────────────────────╮ d=271493;file:///home/ye\_canming/repos/novelties/cv/ScholarlyInfrastructure/scholarly\_infrastructure/logging/torch.py\torch.py;;\:d=536110;file:///home/ye\_canming/repos/novelties/cv/ScholarlyInfrastructure/scholarly\_infrastructure/logging/torch.py#72\72;;\
 │ root │
 │ └── model (Linear) weight:[10, 784] bias:[10] │
 ╰────────────────────────────────────────────────────────────────────────────────────────────╯

tensor(1.4472, grad\_fn=<MulBackward0>)

我们定义好了 loss 和 model（w与b的参数，以及决策超平面如何用于决策）。 现在我们要定义训练算法。

首先我们来首先一下 上文我们提到的sklearn使用的 Leon Bottou optimal learning rate。从现代深度学习的角度来看，Leon Bottou 本质上是提出了一个 learning rate scheduler。

参照 [Leon Bottou 的SVM网站](https://leon.bottou.org/projects/sgd), 他对学习率的规划如下

SGD

$$η\_{t}=\frac{η\_{0}}{\left(1+λη\_{0}t\right)}$$

ASGD

$$η\_{t}=\frac{η\_{0}}{\left(1+λη\_{0}t\right)^{0.75}}$$

其中对于第一个形式，是否是一个特殊的[指数/gamma 衰减的学习率]？(https://pytorch.org/docs/stable/generated/torch.optim.lr\_scheduler.ExponentialLR.html)。

$$η\_{t+1}=\frac{η\_{0}}{\left(1+λη\_{0}\left(t+1\right)\right)}$$

$$γ=\frac{η\_{t+1}}{η\_{t}}=\frac{1}{1+\frac{1}{t+\frac{1}{λη\_{0}}}}$$

答案是否定的，因为 $γ$ 推导出来不是和t无关的常数。 所以Leon Bottou不是 gamma衰减的特殊情况。

scheduler\_lmd\_leon\_bottou\_sgd = lambda epoch, init\_lr=0.1, lmd=0.001: init\_lr / (1+ lmd\*init\_lr\*epoch)
scheduler\_lmd\_leon\_bottou\_asgd = lambda epoch, init\_lr=0.1, lmd=0.001: init\_lr / (1+ lmd\*init\_lr\*epoch)\*\*0.75

不要忘了刚才还有 $\frac{1}{2}\left|\right|w\left|\right|^{2}$ 项， 需要特别注意 bias $b$ 没有被decay。[博客](https://discuss.pytorch.org/t/weight-decay-only-for-weights-of-nn-linear-and-nn-conv/114348)提到，尽管很多论文都指出了分开这两者的weight decay的重要性，但是在torch中实现这一点还是有一定的难度的。

[source](https://github.com/Open-Book-Studio/THU-Coursework-Machine-Learning-for-Big-Data/blob/main/thu_big_data_ml/svm/handy_crafted/linear.py#L206)

separate\_weight\_decay

 separate\_weight\_decay (model:torch.nn.modules.module.Module,
 weight\_decay:float, otherwise\_set\_to:float=0.0,
 verbose:bool=False)

def separate\_weight\_decay(model:nn.Module, weight\_decay: float, otherwise\_set\_to:float=0.0, verbose:bool=False):
 decay = list() # 不能使用 set，由于Pytorch优化器需要顺序
 no\_decay = list()
 for name, param in model.named\_parameters():
 do\_weight\_decay = 'weight' in name
 if verbose:
 print(f'{name} should do weight decay? {do\_weight\_decay}')
 if do\_weight\_decay:
 decay.append(param)
 else:
 no\_decay.append(param)
 # return decay, no\_decay
 return [
 dict(params=decay, weight\_decay=weight\_decay),
 dict(params=no\_decay, weight\_decay=otherwise\_set\_to)
 ]

test\_model = HingeSupportVectorClassifier(10, 10)
weight\_decay = 0.001
separate\_weight\_decay(test\_model, weight\_decay)
pass

Seed set to 0

[source](https://github.com/Open-Book-Studio/THU-Coursework-Machine-Learning-for-Big-Data/blob/main/thu_big_data_ml/svm/handy_crafted/linear.py#L226)

HingeSupportVectorClassifier.configure\_optimizers

 HingeSupportVectorClassifier.configure\_optimizers ()

\*Choose what optimizers and learning-rate schedulers to use in your optimization. Normally you’d need one. But in the case of GANs or similar you might have multiple. Optimization with multiple optimizers only works in the manual optimization mode.

Return: Any of these 6 options.

- \*\*Single optimizer\*\*.
- \*\*List or Tuple\*\* of optimizers.
- \*\*Two lists\*\* - The first list has multiple optimizers, and the second has multiple LR schedulers
 (or multiple ``lr\_scheduler\_config``).
- \*\*Dictionary\*\*, with an ``"optimizer"`` key, and (optionally) a ``"lr\_scheduler"``
 key whose value is a single LR scheduler or ``lr\_scheduler\_config``.
- \*\*None\*\* - Fit will run without any optimizer.

The lr\_scheduler\_config is a dictionary which contains the scheduler and its associated configuration. The default configuration is shown below.

.. code-block:: python

lr\_scheduler\_config = {
 # REQUIRED: The scheduler instance
 "scheduler": lr\_scheduler,
 # The unit of the scheduler's step size, could also be 'step'.
 # 'epoch' updates the scheduler on epoch end whereas 'step'
 # updates it after a optimizer update.
 "interval": "epoch",
 # How many epochs/steps should pass between calls to
 # `scheduler.step()`. 1 corresponds to updating the learning
 # rate after every epoch/step.
 "frequency": 1,
 # Metric to to monitor for schedulers like `ReduceLROnPlateau`
 "monitor": "val\_loss",
 # If set to `True`, will enforce that the value specified 'monitor'
 # is available when the scheduler is updated, thus stopping
 # training if not found. If set to `False`, it will only produce a warning
 "strict": True,
 # If using the `LearningRateMonitor` callback to monitor the
 # learning rate progress, this keyword can be used to specify
 # a custom logged name
 "name": None,
}

When there are schedulers in which the .step() method is conditioned on a value, such as the :class:torch.optim.lr\_scheduler.ReduceLROnPlateau scheduler, Lightning requires that the lr\_scheduler\_config contains the keyword "monitor" set to the metric name that the scheduler should be conditioned on.

.. testcode::

# The ReduceLROnPlateau scheduler requires a monitor
def configure\_optimizers(self):
 optimizer = Adam(...)
 return {
 "optimizer": optimizer,
 "lr\_scheduler": {
 "scheduler": ReduceLROnPlateau(optimizer, ...),
 "monitor": "metric\_to\_track",
 "frequency": "indicates how often the metric is updated",
 # If "monitor" references validation metrics, then "frequency" should be set to a
 # multiple of "trainer.check\_val\_every\_n\_epoch".
 },
 }

# In the case of two optimizers, only one using the ReduceLROnPlateau scheduler
def configure\_optimizers(self):
 optimizer1 = Adam(...)
 optimizer2 = SGD(...)
 scheduler1 = ReduceLROnPlateau(optimizer1, ...)
 scheduler2 = LambdaLR(optimizer2, ...)
 return (
 {
 "optimizer": optimizer1,
 "lr\_scheduler": {
 "scheduler": scheduler1,
 "monitor": "metric\_to\_track",
 },
 },
 {"optimizer": optimizer2, "lr\_scheduler": scheduler2},
 )

Metrics can be made available to monitor by simply logging it using self.log('metric\_to\_track', metric\_val) in your :class:~lightning.pytorch.core.LightningModule.

Note: Some things to know:

- Lightning calls ``.backward()`` and ``.step()`` automatically in case of automatic optimization.
- If a learning rate scheduler is specified in ``configure\_optimizers()`` with key
 ``"interval"`` (default "epoch") in the scheduler configuration, Lightning will call
 the scheduler's ``.step()`` method automatically in case of automatic optimization.
- If you use 16-bit precision (``precision=16``), Lightning will automatically handle the optimizer.
- If you use :class:`torch.optim.LBFGS`, Lightning handles the closure function automatically for you.
- If you use multiple optimizers, you will have to switch to 'manual optimization' mode and step them
 yourself.
- If you need to control how often the optimizer steps, override the :meth:`optimizer\_step` hook.\*

@override
@patch
def configure\_optimizers(self:HingeSupportVectorClassifier) -> OptimizerLRScheduler:
 init\_lr = self.hparams.learning\_rate
 # lmd = self.hparams.weight\_decay / self.hparams.C
 lmd = self.hparams.weight\_decay # 需要考证，Leon Bottou的lamda 是什么情况下推导的。

 weight\_decay = self.hparams.weight\_decay

 if self.hparams.optimizer\_type == torch.optim.ASGD:
 optimizer = torch.optim.ASGD(
 # self.parameters(),
 separate\_weight\_decay(self, weight\_decay),
 lr=init\_lr, # 刚才 save\_hyperparameters() 保存了，这是为了方便是使用 Lightning 调学习率
 # weight\_decay = self.hparams.weight\_decay, # 李航书上的 hinge loss的第一项
 )
 scheduler = torch.optim.lr\_scheduler.LambdaLR(optimizer,
 lr\_lambda=lambda epoch: scheduler\_lmd\_leon\_bottou\_asgd(epoch, init\_lr, lmd))
 elif self.hparams.optimizer\_type == torch.optim.SGD:
 optimizer = torch.optim.SGD(
 # self.parameters(),
 separate\_weight\_decay(self, weight\_decay),
 lr=init\_lr,
 # weight\_decay=self.hparams.weight\_decay
 )
 scheduler = torch.optim.lr\_scheduler.LambdaLR(optimizer,
 lr\_lambda=lambda epoch: scheduler\_lmd\_leon\_bottou\_sgd(epoch, init\_lr, lmd))
 else:
 return self.hparams.optimizer\_type(self.parameters(), lr=init\_lr, weight\_decay=self.hparams.weight\_decay) # 只 return optimizer

 return ([optimizer], [scheduler])

最后我们补上评价指标

[source](https://github.com/Open-Book-Studio/THU-Coursework-Machine-Learning-for-Big-Data/blob/main/thu_big_data_ml/svm/handy_crafted/linear.py#L331)

HingeSupportVectorClassifier.test\_step

 HingeSupportVectorClassifier.test\_step (batch, batch\_idx=None, \*args,
 \*\*kwargs)

\*Operates on a single batch of data from the test set. In this step you’d normally generate examples or calculate anything of interest such as accuracy.

Args: batch: The output of your data iterable, normally a :class:~torch.utils.data.DataLoader. batch\_idx: The index of this batch. dataloader\_idx: The index of the dataloader that produced this batch. (only if multiple dataloaders used)

Return: - :class:~torch.Tensor - The loss tensor - dict - A dictionary. Can include any keys, but must include the key 'loss'. - None - Skip to the next batch.

.. code-block:: python

# if you have one test dataloader:
def test\_step(self, batch, batch\_idx): ...

# if you have multiple test dataloaders:
def test\_step(self, batch, batch\_idx, dataloader\_idx=0): ...

Examples::

# CASE 1: A single test dataset
def test\_step(self, batch, batch\_idx):
 x, y = batch

 # implement your own
 out = self(x)
 loss = self.loss(out, y)

 # log 6 example images
 # or generated text... or whatever
 sample\_imgs = x[:6]
 grid = torchvision.utils.make\_grid(sample\_imgs)
 self.logger.experiment.add\_image('example\_images', grid, 0)

 # calculate acc
 labels\_hat = torch.argmax(out, dim=1)
 test\_acc = torch.sum(y == labels\_hat).item() / (len(y) \* 1.0)

 # log the outputs!
 self.log\_dict({'test\_loss': loss, 'test\_acc': test\_acc})

If you pass in multiple test dataloaders, :meth:test\_step will have an additional argument. We recommend setting the default value of 0 so that you can quickly switch between single and multiple dataloaders.

.. code-block:: python

# CASE 2: multiple test dataloaders
def test\_step(self, batch, batch\_idx, dataloader\_idx=0):
 # dataloader\_idx tells you which dataset this is.
 ...

Note: If you don’t need to test you don’t need to implement this method.

Note: When the :meth:test\_step is called, the model has been put in eval mode and PyTorch gradients have been disabled. At the end of the test epoch, the model goes back to training mode and gradients are enabled.\*

from namable\_classify.infra import append\_dict\_list, ensure\_array
from typing import Any
import numpy as np

@patch
def on\_evaluation\_epoch\_start(self:HingeSupportVectorClassifier, stage:str=""):
 self.evaluation\_steps\_outputs = dict()
 self.evaluation\_steps\_outputs[f'{stage}\_batch\_probs'] = []
 self.evaluation\_steps\_outputs[f'{stage}\_label\_tensor'] = []
 # self.evaluation\_steps\_outputs[f'{stage}\_batch\_preds'] = []

@patch
def evaluation\_step(self:HingeSupportVectorClassifier, batch, batch\_idx=None, stage:str="", \*args: Any, \*\*kwargs: Any) -> STEP\_OUTPUT:
 image\_tensor, label\_tensor = batch
 batch\_probs = self(image\_tensor)
 # batch\_preds = self.predict\_class(image\_tensor)
 append\_dict\_list(self.evaluation\_steps\_outputs, f'{stage}\_batch\_probs', ensure\_array(batch\_probs))
 append\_dict\_list(self.evaluation\_steps\_outputs, f'{stage}\_label\_tensor', ensure\_array(label\_tensor))
 # append\_dict\_list(self.evaluation\_steps\_outputs, f'{stage}\_batch\_preds', ensure\_array(batch\_preds))
 batch\_loss = self.loss\_fn(batch\_probs, label\_tensor)
 self.log(f"{stage}\_loss", batch\_loss, prog\_bar=True)
 return batch\_loss

@patch
def on\_evaluation\_epoch\_end(self:HingeSupportVectorClassifier, stage:str=""):
 # https://github.com/Lightning-AI/pytorch-lightning/discussions/9845
 # labels = self.lit\_data.classes
 # labels = list(range(self.lit\_data.num\_of\_classes))
 labels = list(range(self.hparams.num\_classes))
 # labels = None
 # print(labels)
 # stack 是 new axis， concat是existing axis
 all\_pred\_probs = np.concatenate(self.evaluation\_steps\_outputs[f'{stage}\_batch\_probs'])
 all\_label\_tensor = np.concatenate(self.evaluation\_steps\_outputs[f'{stage}\_label\_tensor'])
 # all\_preds = np.concatenate(self.evaluation\_steps\_outputs[f'{stage}\_batch\_preds'])
 # logger.debug(self.evaluation\_steps\_outputs[f'{stage}\_label\_tensor'])
 # logger.debug(all\_label\_tensor)
 eval\_dict = compute\_classification\_metrics(all\_label\_tensor, all\_pred\_probs,
 # logits\_to\_prob=False,
 logits\_to\_prob=True,
 labels=labels)
 eval\_dict = {f"{stage}\_{k}": v for k,v in eval\_dict.items()}
 self.log\_dict(eval\_dict)
 self.evaluation\_steps\_outputs.clear()

@override
@patch
def on\_validation\_epoch\_start(self:HingeSupportVectorClassifier):
 return self.on\_evaluation\_epoch\_start(stage="val")

@override
@patch
def on\_test\_epoch\_start(self:HingeSupportVectorClassifier):
 return self.on\_evaluation\_epoch\_start(stage="test")

@override
@patch
def on\_validation\_epoch\_end(self:HingeSupportVectorClassifier):
 return self.on\_evaluation\_epoch\_end(stage="val")

@override
@patch
def on\_test\_epoch\_end(self:HingeSupportVectorClassifier):
 return self.on\_evaluation\_epoch\_end(stage="test")

@override
@patch
def validation\_step(self:HingeSupportVectorClassifier, batch, batch\_idx=None, \*args, \*\*kwargs):
 return self.evaluation\_step(batch, batch\_idx, stage="val")

@override
@patch
def test\_step(self:HingeSupportVectorClassifier, batch, batch\_idx=None, \*args, \*\*kwargs):
 return self.evaluation\_step(batch, batch\_idx, stage="test")

[source](https://github.com/Open-Book-Studio/THU-Coursework-Machine-Learning-for-Big-Data/blob/main/thu_big_data_ml/svm/handy_crafted/linear.py#L326)

HingeSupportVectorClassifier.validation\_step

 HingeSupportVectorClassifier.validation\_step (batch, batch\_idx=None,
 \*args, \*\*kwargs)

\*Operates on a single batch of data from the validation set. In this step you’d might generate examples or calculate anything of interest like accuracy.

Args: batch: The output of your data iterable, normally a :class:~torch.utils.data.DataLoader. batch\_idx: The index of this batch. dataloader\_idx: The index of the dataloader that produced this batch. (only if multiple dataloaders used)

Return: - :class:~torch.Tensor - The loss tensor - dict - A dictionary. Can include any keys, but must include the key 'loss'. - None - Skip to the next batch.

.. code-block:: python

# if you have one val dataloader:
def validation\_step(self, batch, batch\_idx): ...

# if you have multiple val dataloaders:
def validation\_step(self, batch, batch\_idx, dataloader\_idx=0): ...

Examples::

# CASE 1: A single validation dataset
def validation\_step(self, batch, batch\_idx):
 x, y = batch

 # implement your own
 out = self(x)
 loss = self.loss(out, y)

 # log 6 example images
 # or generated text... or whatever
 sample\_imgs = x[:6]
 grid = torchvision.utils.make\_grid(sample\_imgs)
 self.logger.experiment.add\_image('example\_images', grid, 0)

 # calculate acc
 labels\_hat = torch.argmax(out, dim=1)
 val\_acc = torch.sum(y == labels\_hat).item() / (len(y) \* 1.0)

 # log the outputs!
 self.log\_dict({'val\_loss': loss, 'val\_acc': val\_acc})

If you pass in multiple val dataloaders, :meth:validation\_step will have an additional argument. We recommend setting the default value of 0 so that you can quickly switch between single and multiple dataloaders.

.. code-block:: python

# CASE 2: multiple validation dataloaders
def validation\_step(self, batch, batch\_idx, dataloader\_idx=0):
 # dataloader\_idx tells you which dataset this is.
 ...

Note: If you don’t need to validate you don’t need to implement this method.

Note: When the :meth:validation\_step is called, the model has been put in eval mode and PyTorch gradients have been disabled. At the end of validation, the model goes back to training mode and gradients are enabled.\*

[source](https://github.com/Open-Book-Studio/THU-Coursework-Machine-Learning-for-Big-Data/blob/main/thu_big_data_ml/svm/handy_crafted/linear.py#L321)

HingeSupportVectorClassifier.on\_test\_epoch\_end

 HingeSupportVectorClassifier.on\_test\_epoch\_end ()

*Called in the test loop at the very end of the epoch.*

[source](https://github.com/Open-Book-Studio/THU-Coursework-Machine-Learning-for-Big-Data/blob/main/thu_big_data_ml/svm/handy_crafted/linear.py#L316)

HingeSupportVectorClassifier.on\_validation\_epoch\_end

 HingeSupportVectorClassifier.on\_validation\_epoch\_end ()

*Called in the validation loop at the very end of the epoch.*

[source](https://github.com/Open-Book-Studio/THU-Coursework-Machine-Learning-for-Big-Data/blob/main/thu_big_data_ml/svm/handy_crafted/linear.py#L311)

HingeSupportVectorClassifier.on\_test\_epoch\_start

 HingeSupportVectorClassifier.on\_test\_epoch\_start ()

*Called in the test loop at the very beginning of the epoch.*

[source](https://github.com/Open-Book-Studio/THU-Coursework-Machine-Learning-for-Big-Data/blob/main/thu_big_data_ml/svm/handy_crafted/linear.py#L306)

HingeSupportVectorClassifier.on\_validation\_epoch\_start

 HingeSupportVectorClassifier.on\_validation\_epoch\_start ()

*Called in the validation loop at the very beginning of the epoch.*

[source](https://github.com/Open-Book-Studio/THU-Coursework-Machine-Learning-for-Big-Data/blob/main/thu_big_data_ml/svm/handy_crafted/linear.py#L283)

HingeSupportVectorClassifier.on\_evaluation\_epoch\_end

 HingeSupportVectorClassifier.on\_evaluation\_epoch\_end (stage:str='')

[source](https://github.com/Open-Book-Studio/THU-Coursework-Machine-Learning-for-Big-Data/blob/main/thu_big_data_ml/svm/handy_crafted/linear.py#L271)

HingeSupportVectorClassifier.evaluation\_step

 HingeSupportVectorClassifier.evaluation\_step (batch, batch\_idx=None,
 stage:str='', \*args:Any,
 \*\*kwargs:Any)

[source](https://github.com/Open-Book-Studio/THU-Coursework-Machine-Learning-for-Big-Data/blob/main/thu_big_data_ml/svm/handy_crafted/linear.py#L264)

HingeSupportVectorClassifier.on\_evaluation\_epoch\_start

 HingeSupportVectorClassifier.on\_evaluation\_epoch\_start (stage:str='')

input\_dim = X\_val.shape[1]
num\_of\_classes = len(categories)
input\_dim, num\_of\_classes

(64, 10)

get\_cls\_task = lambda: HingeSupportVectorClassifier(input\_dim, num\_of\_classes,
 learning\_rate=1,
 weight\_decay=0.5,
 optimizer\_type=torch.optim.ASGD, C=1, squared=True, margin=1,
 strategy="crammer\_singer", experiment\_index=0)
cls\_task = get\_cls\_task()
cls\_data = data\_module
# cls\_data = data\_module\_full

Seed set to 0

#### 0.1.4 SVM 训练与评价

from thu\_big\_data\_ml.help import runs\_path
from lightning.pytorch.callbacks.early\_stopping import EarlyStopping
from lightning.pytorch.callbacks import ModelSummary, LearningRateMonitor, LearningRateFinder, BatchSizeFinder

trainer = L.Trainer(default\_root\_dir=runs\_path,
 enable\_checkpointing=True,
 enable\_model\_summary=True,
 # num\_sanity\_val\_steps=2, # 防止 val 在训了好久train才发现崩溃
 callbacks=[
 EarlyStopping(monitor="val\_acc1", mode="max", check\_finite=True,
 patience=5,
 check\_on\_train\_epoch\_end=False, # check on validation end
 verbose=True),
 ModelSummary(max\_depth=3),
 # LearningRateFinder(),
 # BatchSizeFinder(),
 LearningRateMonitor(logging\_interval="epoch")
 ]
 # , fast\_dev\_run=True
 # , limit\_train\_batches=10,
 # limit\_val\_batches=5
 , accelerator="gpu", devices=1, strategy="auto"
 )

/home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/fabric/plugins/environments/slurm.py:204: The `srun` command is available on your system but is not used. HINT: If your intention is to run Lightning on SLURM, prepend your python command with `srun` like so: srun python /home/ye\_canming/program\_files/managers/conda/envs/y ...
Trainer already configured with model summary callbacks: [<class 'lightning.pytorch.callbacks.model\_summary.ModelSummary'>]. Skipping setting a default `ModelSummary` callback.
GPU available: True (cuda), used: True
TPU available: False, using: 0 TPU cores
HPU available: False, using: 0 HPUs

trainer.fit(cls\_task, datamodule=cls\_data)
val\_res = trainer.validate(cls\_task, datamodule=cls\_data)
test\_res = trainer.test(cls\_task, datamodule=cls\_data)
pass

/home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/loops/utilities.py:72: `max\_epochs` was not set. Setting it to 1000 epochs. To train without an epoch limit, set `max\_epochs=-1`.
INFO: LOCAL\_RANK: 0 - CUDA\_VISIBLE\_DEVICES: [0,1,2,3,4,5,6,7]

Sat 2024-11-16 00:58:49.020341

INFO LOCAL\_RANK: 0 - CUDA\_VISIBLE\_DEVICES: [0,1,2,3,4,5,6,7] d=262674;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/accelerators/cuda.py\cuda.py;;\:d=953938;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/accelerators/cuda.py#61\61;;\

INFO:
 | Name | Type | Params | Mode | In sizes | Out sizes
-------------------------------------------------------------------------------
0 | model | Linear | 650 | train | [1, 64] | [1, 10]
1 | loss\_fn | MultiClassHingeLoss | 0 | train | ? | ?
-------------------------------------------------------------------------------
650 Trainable params
0 Non-trainable params
650 Total params
0.003 Total estimated model params size (MB)
2 Modules in train mode
0 Modules in eval mode

Sat 2024-11-16 00:58:49.057947

INFO d=945989;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/callbacks/model\_summary.py\model\_summary.py;;\:d=154100;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/callbacks/model\_summary.py#104\104;;\
 | Name | Type | Params | Mode | In sizes | Out sizes
 -------------------------------------------------------------------------------
 0 | model | Linear | 650 | train | [1, 64] | [1, 10]
 1 | loss\_fn | MultiClassHingeLoss | 0 | train | ? | ?
 -------------------------------------------------------------------------------
 650 Trainable params
 0 Non-trainable params
 650 Total params
 0.003 Total estimated model params size (MB)
 2 Modules in train mode
 0 Modules in eval mode

Sanity Checking: | | 0/? [00:00<?, ?it/s]

/home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/loops/fit\_loop.py:298: The number of training batches (11) is smaller than the logging interval Trainer(log\_every\_n\_steps=50). Set a lower value for log\_every\_n\_steps if you want to see logs for the training epoch.

Training: | | 0/? [00:00<?, ?it/s]

Validation: | | 0/? [00:00<?, ?it/s]

INFO: Metric val\_acc1 improved. New best score: 0.514

Sat 2024-11-16 00:58:52.534348

INFO Metric val\_acc1 improved. New best score: 0.514 d=942500;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/callbacks/early\_stopping.py\early\_stopping.py;;\:d=891786;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/callbacks/early\_stopping.py#273\273;;\

Validation: | | 0/? [00:00<?, ?it/s]

INFO: Metric val\_acc1 improved by 0.229 >= min\_delta = 0.0. New best score: 0.743

Sat 2024-11-16 00:58:55.733727

INFO Metric val\_acc1 improved by 0.229 >= min\_delta = 0.0. New best score: 0.743 d=105592;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/callbacks/early\_stopping.py\early\_stopping.py;;\:d=370977;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/callbacks/early\_stopping.py#273\273;;\

Validation: | | 0/? [00:00<?, ?it/s]

Validation: | | 0/? [00:00<?, ?it/s]

INFO: Metric val\_acc1 improved by 0.007 >= min\_delta = 0.0. New best score: 0.750

Sat 2024-11-16 00:59:01.783794

INFO Metric val\_acc1 improved by 0.007 >= min\_delta = 0.0. New best score: 0.750 d=957361;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/callbacks/early\_stopping.py\early\_stopping.py;;\:d=214410;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/callbacks/early\_stopping.py#273\273;;\

Validation: | | 0/? [00:00<?, ?it/s]

Validation: | | 0/? [00:00<?, ?it/s]

Validation: | | 0/? [00:00<?, ?it/s]

INFO: Metric val\_acc1 improved by 0.111 >= min\_delta = 0.0. New best score: 0.861

Sat 2024-11-16 00:59:09.920693

INFO Metric val\_acc1 improved by 0.111 >= min\_delta = 0.0. New best score: 0.861 d=546678;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/callbacks/early\_stopping.py\early\_stopping.py;;\:d=273145;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/callbacks/early\_stopping.py#273\273;;\

Validation: | | 0/? [00:00<?, ?it/s]

Validation: | | 0/? [00:00<?, ?it/s]

INFO: Metric val\_acc1 improved by 0.042 >= min\_delta = 0.0. New best score: 0.903

Sat 2024-11-16 00:59:14.194501

INFO Metric val\_acc1 improved by 0.042 >= min\_delta = 0.0. New best score: 0.903 d=960489;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/callbacks/early\_stopping.py\early\_stopping.py;;\:d=14723;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/callbacks/early\_stopping.py#273\273;;\

Validation: | | 0/? [00:00<?, ?it/s]

Validation: | | 0/? [00:00<?, ?it/s]

Validation: | | 0/? [00:00<?, ?it/s]

Validation: | | 0/? [00:00<?, ?it/s]

Validation: | | 0/? [00:00<?, ?it/s]

INFO: Monitored metric val\_acc1 did not improve in the last 5 records. Best score: 0.903. Signaling Trainer to stop.

Sat 2024-11-16 00:59:26.505406

INFO Monitored metric val\_acc1 did not improve in the last 5 records. Best score: 0.903. d=744754;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/callbacks/early\_stopping.py\early\_stopping.py;;\:d=864912;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/callbacks/early\_stopping.py#273\273;;\
 Signaling Trainer to stop.

/home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/fabric/plugins/environments/slurm.py:204: The `srun` command is available on your system but is not used. HINT: If your intention is to run Lightning on SLURM, prepend your python command with `srun` like so: srun python /home/ye\_canming/program\_files/managers/conda/envs/y ...
INFO: LOCAL\_RANK: 0 - CUDA\_VISIBLE\_DEVICES: [0,1,2,3,4,5,6,7]

Sat 2024-11-16 00:59:27.127571

INFO LOCAL\_RANK: 0 - CUDA\_VISIBLE\_DEVICES: [0,1,2,3,4,5,6,7] d=641620;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/accelerators/cuda.py\cuda.py;;\:d=517553;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/accelerators/cuda.py#61\61;;\

Validation: | | 0/? [00:00<?, ?it/s]

┏━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━━━━┓
┃ Validate metric ┃ DataLoader 0 ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━━━━┩
│ val\_acc1 │ 0.8541666865348816 │
│ val\_acc10 │ 1.0 │
│ val\_acc2 │ 0.9513888955116272 │
│ val\_acc20 │ 1.0 │
│ val\_acc3 │ 0.9652777910232544 │
│ val\_acc5 │ 0.9791666865348816 │
│ val\_balanced\_accuracy │ 0.8519047498703003 │
│ val\_cohen\_kappa │ 0.8378465175628662 │
│ val\_f1 │ 0.8545861840248108 │
│ val\_hinge\_loss │ 0.29842936992645264 │
│ val\_log\_loss │ 0.9072062373161316 │
│ val\_loss │ 263.0315856933594 │
│ val\_matthews\_corrcoef │ 0.8427900671958923 │
│ val\_precision │ 0.8835164904594421 │
│ val\_recall │ 0.8519047498703003 │
│ val\_roc\_auc │ 0.9750232100486755 │
└───────────────────────────┴───────────────────────────┘

/home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/fabric/plugins/environments/slurm.py:204: The `srun` command is available on your system but is not used. HINT: If your intention is to run Lightning on SLURM, prepend your python command with `srun` like so: srun python /home/ye\_canming/program\_files/managers/conda/envs/y ...
INFO: LOCAL\_RANK: 0 - CUDA\_VISIBLE\_DEVICES: [0,1,2,3,4,5,6,7]

Sat 2024-11-16 00:59:28.313136

INFO LOCAL\_RANK: 0 - CUDA\_VISIBLE\_DEVICES: [0,1,2,3,4,5,6,7] d=765752;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/accelerators/cuda.py\cuda.py;;\:d=341001;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/accelerators/cuda.py#61\61;;\

Testing: | | 0/? [00:00<?, ?it/s]

┏━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━━━━┓
┃ Test metric ┃ DataLoader 0 ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━━━━┩
│ test\_acc1 │ 0.855555534362793 │
│ test\_acc10 │ 1.0 │
│ test\_acc2 │ 0.9027777910232544 │
│ test\_acc20 │ 1.0 │
│ test\_acc3 │ 0.9444444179534912 │
│ test\_acc5 │ 0.9694444537162781 │
│ test\_balanced\_accuracy │ 0.8538845777511597 │
│ test\_cohen\_kappa │ 0.8394717574119568 │
│ test\_f1 │ 0.8514753580093384 │
│ test\_hinge\_loss │ 0.30003729462623596 │
│ test\_log\_loss │ 1.314468264579773 │
│ test\_loss │ 328.9302062988281 │
│ test\_matthews\_corrcoef │ 0.8407779932022095 │
│ test\_precision │ 0.8590342998504639 │
│ test\_recall │ 0.8538845777511597 │
│ test\_roc\_auc │ 0.9722300171852112 │
└───────────────────────────┴───────────────────────────┘

### 0.2 对手动实现的SVM进行调参

要想进行调参，我们需要首先思考一下，实际上有哪些元参数是必要的。

首先optimizer\_type选择ASGD还是SGD，squared是hinge loss还是squared hinge loss，strategy使用crammer\_singer还是one vs all, 这三个参数的选择自然是会影响到最终的结果，是重要的目标元参数或者冗余元参数。然而，剩下的learning\_rate,weight\_decay和C, margin我们可以发现，本质上只有前两个。C和learning\_rate是一样的效果，而margin是一个函数间隔，也是可以任意选取的，不影响优化问题的最优解（当然会一定程度影响landscape）。

因此，对于软间隔的Linear SVM的原始问题，我们只需要在给定优化器、损失函数、多分类策略的情况下，只需要调整现代深度学习中也很重要的learning\_rate,weight\_decay参数。

我在上次Project作业KD树中详细描述了谷歌AI团队《深度学习调优指南》的思想，涉及到的概念包括目标元参数、冗余元参数和固定元参数，贝叶斯优化、演化计算、近似随机搜索，科学实验的控制变量法与调参实验设计中的探索与利用、调参结果的假设检验分析等。这里我们不再赘述，需要的话可以阅读[上次作业的文档](https://github.com/Open-Book-Studio/THU-Coursework-Machine-Learning-for-Big-Data/blob/main/notebooks/coding_projects/P1_KNN/kd_tree.ipynb)。

与上次作业不同，今天我们调的参数（学习率）具有一定的特殊性，因为我们今天的模型是个可以SGD优化的模型，所以我们可以用到深度学习论文中提出的一些调参方法来进行调参。具体而言，我们今天学习的是这篇经典论文[Cyclical Learning Rates for Training Neural Networks](https://ieeexplore.ieee.org/abstract/document/7926641) 的调参方法，具体的实现已经集成到 lightning 库中，我们可以直接使用。

from lightning.pytorch.tuner import Tuner

cls\_task = get\_cls\_task()
tuner\_trainer = L.Trainer()
tuner = Tuner(tuner\_trainer)
lr\_finder = tuner.lr\_find(cls\_task, datamodule=cls\_data, max\_lr=10, min\_lr=1e-6, num\_training=100, mode = "exponential")
# print(lr\_finder.results)
fig = lr\_finder.plot(suggest=True)
fig.show()
new\_lr = lr\_finder.suggestion()
new\_lr, cls\_task.hparams.learning\_rate

INFO: Seed set to 0

Sat 2024-11-16 00:52:49.912254

INFO Seed set to 0 d=717209;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/fabric/utilities/seed.py\seed.py;;\:d=346236;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/fabric/utilities/seed.py#57\57;;\

INFO: Trainer will use only 1 of 8 GPUs because it is running inside an interactive / notebook environment. You may try to set `Trainer(devices=8)` but please note that multi-GPU inside interactive / notebook environments is considered experimental and unstable. Your mileage may vary.

Sat 2024-11-16 00:52:49.928983

INFO Trainer will use only 1 of 8 GPUs because it is running inside an interactive / notebook d=441001;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning\_utilities/core/rank\_zero.py\rank\_zero.py;;\:d=42450;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning\_utilities/core/rank\_zero.py#63\63;;\
 environment. You may try to set `Trainer(devices=8)` but please note that multi-GPU inside
 interactive / notebook environments is considered experimental and unstable. Your mileage
 may vary.

INFO: GPU available: True (cuda), used: True

Sat 2024-11-16 00:52:49.953772

INFO GPU available: True (cuda), used: True d=962838;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning\_utilities/core/rank\_zero.py\rank\_zero.py;;\:d=821872;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning\_utilities/core/rank\_zero.py#63\63;;\

INFO: TPU available: False, using: 0 TPU cores

Sat 2024-11-16 00:52:49.958918

INFO TPU available: False, using: 0 TPU cores d=611720;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning\_utilities/core/rank\_zero.py\rank\_zero.py;;\:d=934973;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning\_utilities/core/rank\_zero.py#63\63;;\

INFO: HPU available: False, using: 0 HPUs

Sat 2024-11-16 00:52:49.962648

INFO HPU available: False, using: 0 HPUs d=295528;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning\_utilities/core/rank\_zero.py\rank\_zero.py;;\:d=146534;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning\_utilities/core/rank\_zero.py#63\63;;\

INFO: LOCAL\_RANK: 0 - CUDA\_VISIBLE\_DEVICES: [0,1,2,3,4,5,6,7]

Sat 2024-11-16 00:52:49.972189

INFO LOCAL\_RANK: 0 - CUDA\_VISIBLE\_DEVICES: [0,1,2,3,4,5,6,7] d=262674;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/accelerators/cuda.py\cuda.py;;\:d=953938;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/accelerators/cuda.py#61\61;;\

Finding best initial lr: 0%| | 0/100 [00:00<?, ?it/s]

INFO: LR finder stopped early after 85 steps due to diverging loss.

Sat 2024-11-16 00:53:03.916698

INFO LR finder stopped early after 85 steps due to diverging loss. d=325213;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/tuner/lr\_finder.py\lr\_finder.py;;\:d=103560;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/tuner/lr\_finder.py#282\282;;\

INFO: Learning rate set to 3.467368504525316e-05

Sat 2024-11-16 00:53:03.941939

INFO Learning rate set to 3.467368504525316e-05 d=346236;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/tuner/lr\_finder.py\lr\_finder.py;;\:d=495077;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/tuner/lr\_finder.py#301\301;;\

INFO: Restoring states from the checkpoint path at /home/ye\_canming/repos/assignments/THU-Coursework-Machine-Learning-for-Big-Data/notebooks/coding\_projects/P2\_SVM/.lr\_find\_eeb25b7b-6568-46de-b7da-ccdf5af0b22b.ckpt

Sat 2024-11-16 00:53:03.946531

INFO Restoring states from the checkpoint path at d=331556;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning\_utilities/core/rank\_zero.py\rank\_zero.py;;\:d=640561;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning\_utilities/core/rank\_zero.py#63\63;;\
 /home/ye\_canming/repos/assignments/THU-Coursework-Machine-Learning-for-Big-Data/notebooks/
 coding\_projects/P2\_SVM/.lr\_find\_eeb25b7b-6568-46de-b7da-ccdf5af0b22b.ckpt

INFO: Restored all states from the checkpoint at /home/ye\_canming/repos/assignments/THU-Coursework-Machine-Learning-for-Big-Data/notebooks/coding\_projects/P2\_SVM/.lr\_find\_eeb25b7b-6568-46de-b7da-ccdf5af0b22b.ckpt

Sat 2024-11-16 00:53:03.962106

INFO Restored all states from the checkpoint at d=500181;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning\_utilities/core/rank\_zero.py\rank\_zero.py;;\:d=464197;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning\_utilities/core/rank\_zero.py#63\63;;\
 /home/ye\_canming/repos/assignments/THU-Coursework-Machine-Learning-for-Big-Data/notebooks/
 coding\_projects/P2\_SVM/.lr\_find\_eeb25b7b-6568-46de-b7da-ccdf5af0b22b.ckpt

(3.467368504525316e-05, 3.467368504525316e-05)



这个方法我认为也可以用于调整weight decay。

cls\_task = get\_cls\_task()
cls\_task.hparams.learning\_rate = new\_lr
tuner\_trainer = L.Trainer()
tuner = Tuner(tuner\_trainer)
weight\_decay\_finder = tuner.lr\_find(cls\_task, datamodule=cls\_data, max\_lr=1e-3, min\_lr=1e-5, num\_training=50, attr\_name='weight\_decay')

INFO: Seed set to 0

Sat 2024-11-16 00:54:03.777345

INFO Seed set to 0 d=942500;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/fabric/utilities/seed.py\seed.py;;\:d=891786;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/fabric/utilities/seed.py#57\57;;\

INFO: Trainer will use only 1 of 8 GPUs because it is running inside an interactive / notebook environment. You may try to set `Trainer(devices=8)` but please note that multi-GPU inside interactive / notebook environments is considered experimental and unstable. Your mileage may vary.

Sat 2024-11-16 00:54:03.800328

INFO Trainer will use only 1 of 8 GPUs because it is running inside an interactive / notebook d=441001;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning\_utilities/core/rank\_zero.py\rank\_zero.py;;\:d=42450;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning\_utilities/core/rank\_zero.py#63\63;;\
 environment. You may try to set `Trainer(devices=8)` but please note that multi-GPU inside
 interactive / notebook environments is considered experimental and unstable. Your mileage
 may vary.

INFO: GPU available: True (cuda), used: True

Sat 2024-11-16 00:54:03.836096

INFO GPU available: True (cuda), used: True d=962838;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning\_utilities/core/rank\_zero.py\rank\_zero.py;;\:d=821872;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning\_utilities/core/rank\_zero.py#63\63;;\

INFO: TPU available: False, using: 0 TPU cores

Sat 2024-11-16 00:54:03.852469

INFO TPU available: False, using: 0 TPU cores d=611720;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning\_utilities/core/rank\_zero.py\rank\_zero.py;;\:d=934973;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning\_utilities/core/rank\_zero.py#63\63;;\

INFO: HPU available: False, using: 0 HPUs

Sat 2024-11-16 00:54:03.868754

INFO HPU available: False, using: 0 HPUs d=295528;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning\_utilities/core/rank\_zero.py\rank\_zero.py;;\:d=146534;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning\_utilities/core/rank\_zero.py#63\63;;\

INFO: LOCAL\_RANK: 0 - CUDA\_VISIBLE\_DEVICES: [0,1,2,3,4,5,6,7]

Sat 2024-11-16 00:54:03.892937

INFO LOCAL\_RANK: 0 - CUDA\_VISIBLE\_DEVICES: [0,1,2,3,4,5,6,7] d=262674;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/accelerators/cuda.py\cuda.py;;\:d=953938;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/accelerators/cuda.py#61\61;;\

Finding best initial lr: 0%| | 0/50 [00:00<?, ?it/s]

INFO: `Trainer.fit` stopped: `max\_steps=50` reached.

Sat 2024-11-16 00:54:11.843827

INFO `Trainer.fit` stopped: `max\_steps=50` reached. d=945989;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning\_utilities/core/rank\_zero.py\rank\_zero.py;;\:d=154100;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning\_utilities/core/rank\_zero.py#63\63;;\

INFO: Learning rate set to 7.585775750291839e-05

Sat 2024-11-16 00:54:11.865877

INFO Learning rate set to 7.585775750291839e-05 d=942500;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/tuner/lr\_finder.py\lr\_finder.py;;\:d=891786;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/tuner/lr\_finder.py#301\301;;\

INFO: Restoring states from the checkpoint path at /home/ye\_canming/repos/assignments/THU-Coursework-Machine-Learning-for-Big-Data/notebooks/coding\_projects/P2\_SVM/.lr\_find\_c0a44246-c85b-407a-8dae-440075c5ae1a.ckpt

Sat 2024-11-16 00:54:11.877609

INFO Restoring states from the checkpoint path at d=105592;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning\_utilities/core/rank\_zero.py\rank\_zero.py;;\:d=370977;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning\_utilities/core/rank\_zero.py#63\63;;\
 /home/ye\_canming/repos/assignments/THU-Coursework-Machine-Learning-for-Big-Data/notebooks/
 coding\_projects/P2\_SVM/.lr\_find\_c0a44246-c85b-407a-8dae-440075c5ae1a.ckpt

INFO: Restored all states from the checkpoint at /home/ye\_canming/repos/assignments/THU-Coursework-Machine-Learning-for-Big-Data/notebooks/coding\_projects/P2\_SVM/.lr\_find\_c0a44246-c85b-407a-8dae-440075c5ae1a.ckpt

Sat 2024-11-16 00:54:11.900391

INFO Restored all states from the checkpoint at d=957361;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning\_utilities/core/rank\_zero.py\rank\_zero.py;;\:d=214410;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning\_utilities/core/rank\_zero.py#63\63;;\
 /home/ye\_canming/repos/assignments/THU-Coursework-Machine-Learning-for-Big-Data/notebooks/
 coding\_projects/P2\_SVM/.lr\_find\_c0a44246-c85b-407a-8dae-440075c5ae1a.ckpt

fig = weight\_decay\_finder.plot(suggest=True)
fig.show()
new\_weight\_decay = weight\_decay\_finder.suggestion()
new\_weight\_decay, cls\_task.hparams.weight\_decay

(7.585775750291839e-05, 7.585775750291839e-05)



由于代码限制，第二张图中显示的learning rate其实是weight decay。

现在我们对调整后的参数进行重新训练

cls\_task = get\_cls\_task()
cls\_task.hparams.learning\_rate = new\_lr
cls\_task.hparams.weight\_decay = new\_weight\_decay

INFO: Seed set to 0

Sat 2024-11-16 00:54:57.243884

INFO Seed set to 0 d=945989;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/fabric/utilities/seed.py\seed.py;;\:d=154100;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/fabric/utilities/seed.py#57\57;;\

trainer.fit(cls\_task, datamodule=cls\_data)
val\_res\_tuned = trainer.validate(cls\_task, datamodule=cls\_data)
test\_res\_tuned = trainer.test(cls\_task, datamodule=cls\_data)
pass

/home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/fabric/plugins/environments/slurm.py:204: The `srun` command is available on your system but is not used. HINT: If your intention is to run Lightning on SLURM, prepend your python command with `srun` like so: srun python /home/ye\_canming/program\_files/managers/conda/envs/y ...
/home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/callbacks/model\_checkpoint.py:654: Checkpoint directory /home/ye\_canming/repos/assignments/THU-Coursework-Machine-Learning-for-Big-Data/runs/lightning\_logs/version\_16/checkpoints exists and is not empty.
INFO: LOCAL\_RANK: 0 - CUDA\_VISIBLE\_DEVICES: [0,1,2,3,4,5,6,7]

Sat 2024-11-16 00:58:01.204967

INFO LOCAL\_RANK: 0 - CUDA\_VISIBLE\_DEVICES: [0,1,2,3,4,5,6,7] d=495077;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/accelerators/cuda.py\cuda.py;;\:d=587007;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/accelerators/cuda.py#61\61;;\

INFO:
 | Name | Type | Params | Mode | In sizes | Out sizes
-------------------------------------------------------------------------------
0 | model | Linear | 650 | train | [1, 64] | [1, 10]
1 | loss\_fn | MultiClassHingeLoss | 0 | train | ? | ?
-------------------------------------------------------------------------------
650 Trainable params
0 Non-trainable params
650 Total params
0.003 Total estimated model params size (MB)
2 Modules in train mode
0 Modules in eval mode

Sat 2024-11-16 00:58:01.224387

INFO d=640561;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/callbacks/model\_summary.py\model\_summary.py;;\:d=671532;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/callbacks/model\_summary.py#104\104;;\
 | Name | Type | Params | Mode | In sizes | Out sizes
 -------------------------------------------------------------------------------
 0 | model | Linear | 650 | train | [1, 64] | [1, 10]
 1 | loss\_fn | MultiClassHingeLoss | 0 | train | ? | ?
 -------------------------------------------------------------------------------
 650 Trainable params
 0 Non-trainable params
 650 Total params
 0.003 Total estimated model params size (MB)
 2 Modules in train mode
 0 Modules in eval mode

Sanity Checking: | | 0/? [00:00<?, ?it/s]

/home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/loops/fit\_loop.py:298: The number of training batches (11) is smaller than the logging interval Trainer(log\_every\_n\_steps=50). Set a lower value for log\_every\_n\_steps if you want to see logs for the training epoch.
/home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/fabric/plugins/environments/slurm.py:204: The `srun` command is available on your system but is not used. HINT: If your intention is to run Lightning on SLURM, prepend your python command with `srun` like so: srun python /home/ye\_canming/program\_files/managers/conda/envs/y ...
INFO: LOCAL\_RANK: 0 - CUDA\_VISIBLE\_DEVICES: [0,1,2,3,4,5,6,7]

Sat 2024-11-16 00:58:03.645924

INFO LOCAL\_RANK: 0 - CUDA\_VISIBLE\_DEVICES: [0,1,2,3,4,5,6,7] d=464197;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/accelerators/cuda.py\cuda.py;;\:d=907343;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/accelerators/cuda.py#61\61;;\

Validation: | | 0/? [00:00<?, ?it/s]

┏━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━━━━┓
┃ Validate metric ┃ DataLoader 0 ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━━━━┩
│ val\_acc1 │ 0.1319444477558136 │
│ val\_acc10 │ 1.0 │
│ val\_acc2 │ 0.2361111044883728 │
│ val\_acc20 │ 1.0 │
│ val\_acc3 │ 0.3541666567325592 │
│ val\_acc5 │ 0.5347222089767456 │
│ val\_balanced\_accuracy │ 0.13571429252624512 │
│ val\_cohen\_kappa │ 0.03511122986674309 │
│ val\_f1 │ 0.1358683854341507 │
│ val\_hinge\_loss │ 1.1006195545196533 │
│ val\_log\_loss │ 2.3971312046051025 │
│ val\_loss │ 204.75518798828125 │
│ val\_matthews\_corrcoef │ 0.03540673106908798 │
│ val\_precision │ 0.13904762268066406 │
│ val\_recall │ 0.13571429252624512 │
│ val\_roc\_auc │ 0.544339656829834 │
└───────────────────────────┴───────────────────────────┘

/home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/fabric/plugins/environments/slurm.py:204: The `srun` command is available on your system but is not used. HINT: If your intention is to run Lightning on SLURM, prepend your python command with `srun` like so: srun python /home/ye\_canming/program\_files/managers/conda/envs/y ...
INFO: LOCAL\_RANK: 0 - CUDA\_VISIBLE\_DEVICES: [0,1,2,3,4,5,6,7]

Sat 2024-11-16 00:58:04.947243

INFO LOCAL\_RANK: 0 - CUDA\_VISIBLE\_DEVICES: [0,1,2,3,4,5,6,7] d=963080;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/accelerators/cuda.py\cuda.py;;\:d=575352;file:///home/ye\_canming/program\_files/managers/conda/envs/yuequ/lib/python3.10/site-packages/lightning/pytorch/accelerators/cuda.py#61\61;;\

Testing: | | 0/? [00:00<?, ?it/s]

┏━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━━━━┓
┃ Test metric ┃ DataLoader 0 ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━━━━┩
│ test\_acc1 │ 0.11666666716337204 │
│ test\_acc10 │ 1.0 │
│ test\_acc2 │ 0.25555557012557983 │
│ test\_acc20 │ 1.0 │
│ test\_acc3 │ 0.3444444537162781 │
│ test\_acc5 │ 0.5166666507720947 │
│ test\_balanced\_accuracy │ 0.11671385914087296 │
│ test\_cohen\_kappa │ 0.018240757286548615 │
│ test\_f1 │ 0.1174597293138504 │
│ test\_hinge\_loss │ 1.1020585298538208 │
│ test\_log\_loss │ 2.420191526412964 │
│ test\_loss │ 219.2089385986328 │
│ test\_matthews\_corrcoef │ 0.018393170088529587 │
│ test\_precision │ 0.12187045812606812 │
│ test\_recall │ 0.11671385914087296 │
│ test\_roc\_auc │ 0.5245167016983032 │
└───────────────────────────┴───────────────────────────┘

val\_res = val\_res[0]
test\_res = test\_res[0]
val\_res\_tuned = val\_res\_tuned[0]
test\_res\_tuned = test\_res\_tuned[0]

results = []
for key in val\_res.keys():
 delta = val\_res\_tuned[key] - val\_res[key]
 is\_better = delta>0
 if 'loss' in key:
 is\_better = not is\_better
 results.append(dict(metric=key, value=val\_res[key], tuned\_value=val\_res\_tuned[key], delta=delta, is\_better=is\_better))
for key in test\_res.keys():
 delta = test\_res\_tuned[key] - test\_res[key]
 is\_better = delta>0
 if 'loss' in key:
 is\_better = not is\_better
 results.append(dict(metric=key, value=test\_res[key], tuned\_value=test\_res\_tuned[key], delta=delta, is\_better=is\_better))

pd.DataFrame(results)

|  | metric | value | tuned\_value | delta | is\_better |
| --- | --- | --- | --- | --- | --- |
| 0 | val\_loss | 263.031586 | 204.755188 | -58.276398 | True |
| 1 | val\_acc1 | 0.854167 | 0.131944 | -0.722222 | False |
| 2 | val\_acc2 | 0.951389 | 0.236111 | -0.715278 | False |
| 3 | val\_acc3 | 0.965278 | 0.354167 | -0.611111 | False |
| 4 | val\_acc5 | 0.979167 | 0.534722 | -0.444444 | False |
| 5 | val\_acc10 | 1.000000 | 1.000000 | 0.000000 | False |
| 6 | val\_acc20 | 1.000000 | 1.000000 | 0.000000 | False |
| 7 | val\_matthews\_corrcoef | 0.842790 | 0.035407 | -0.807383 | False |
| 8 | val\_f1 | 0.854586 | 0.135868 | -0.718718 | False |
| 9 | val\_precision | 0.883516 | 0.139048 | -0.744469 | False |
| 10 | val\_recall | 0.851905 | 0.135714 | -0.716190 | False |
| 11 | val\_balanced\_accuracy | 0.851905 | 0.135714 | -0.716190 | False |
| 12 | val\_cohen\_kappa | 0.837847 | 0.035111 | -0.802735 | False |
| 13 | val\_roc\_auc | 0.975023 | 0.544340 | -0.430684 | False |
| 14 | val\_hinge\_loss | 0.298429 | 1.100620 | 0.802190 | False |
| 15 | val\_log\_loss | 0.907206 | 2.397131 | 1.489925 | False |
| 16 | test\_loss | 328.930206 | 219.208939 | -109.721268 | True |
| 17 | test\_acc1 | 0.855556 | 0.116667 | -0.738889 | False |
| 18 | test\_acc2 | 0.902778 | 0.255556 | -0.647222 | False |
| 19 | test\_acc3 | 0.944444 | 0.344444 | -0.600000 | False |
| 20 | test\_acc5 | 0.969444 | 0.516667 | -0.452778 | False |
| 21 | test\_acc10 | 1.000000 | 1.000000 | 0.000000 | False |
| 22 | test\_acc20 | 1.000000 | 1.000000 | 0.000000 | False |
| 23 | test\_matthews\_corrcoef | 0.840778 | 0.018393 | -0.822385 | False |
| 24 | test\_f1 | 0.851475 | 0.117460 | -0.734016 | False |
| 25 | test\_precision | 0.859034 | 0.121870 | -0.737164 | False |
| 26 | test\_recall | 0.853885 | 0.116714 | -0.737171 | False |
| 27 | test\_balanced\_accuracy | 0.853885 | 0.116714 | -0.737171 | False |
| 28 | test\_cohen\_kappa | 0.839472 | 0.018241 | -0.821231 | False |
| 29 | test\_roc\_auc | 0.972230 | 0.524517 | -0.447713 | False |
| 30 | test\_hinge\_loss | 0.300037 | 1.102059 | 0.802021 | False |
| 31 | test\_log\_loss | 1.314468 | 2.420192 | 1.105723 | False |

我们发现其实 test\_loss 与 val\_loss 其实我们调参之后的模型确实变小了，说明了调参的有效性。

但是其他所有指标，包括sklearn里面的hinge\_loss (one vs all, 不是 crammer\_singer，并且有对样本平均) 都变差了。这说明 crammer\_singer 方法，或者”sum”而非”mean“的损失函数，是带有一定局限性的。

### 0.3 附加题: 对比不同 kernel 方法下的 SVM 分类器 （对完整SVM进行调参）

现在我们手动实现了线性SVM，接下来我们对比不同 kernel 方法下的 SVM 分类器。

接下来的内容请见文件 [03svm\_kernel\_hpo](./03svm_kernel_hpo.html)。

本次Project的目录请见绪论 [00svm](./00svm.html)。