20220508-书&机器学习

1.过程描述

1.1 故事

> 大多数人都相信，生活会带来具有绝对而不可逆转变化的闭合式经历；相信他们最大的冲突源泉都在其自身之外；相信他们是其自身生存状态中的单一主动主人公；相信他们生活在一个连贯而具有因果关联的现实里，其一切生存活动都在一个连续的时间中运行；相信在这个现实里，事件的发生都有其可以解释的、有意义的原因。自从我们人类的始祖凝视着自己升起的一堆火，暗自思忖“我在”以来，人类便是这样看待世界及生活在其间的自己的。

> 故事是我们思想和激情的体现。用埃德蒙德 胡塞尔的话来说，是我们意欲向观众灌输的情感和见识的“一种客观关联”。如果你在写作时一只眼睛盯着稿子，另一只眼睛盯着好莱坞，为了避免商业主义的污染而做出一些违心的选择，那么你便是在拿文学撒气。就像一个生活在强大父亲阴影中的孩子，你打破好莱坞的“规则”，是因为这样能给你一种自由感。但是，对父权的愤怒反抗并不是创造力，而是为了博取关注的忤逆行为。为不同而不同就像对商业法则的盲从一样空洞。你只能写自己相信的东西。

1.2 深度学习

基于pytorch的线性回归的实现

Python | 复制代码

```python
%matplotlib inline
import random
import torch
from d2l import torch as d2l

#正态分布生成训练数据
def synthetic_data(w,b,num_examples):
    X=torch.normal(0,1,(num_examples,len(w)))
    y=torch.matmul(X,w)+b
    y+=torch.normal(0,0.01,y.shape)
    return X,y.reshape((-1,1))

#生成1000个数据点
true_w=torch.tensor([2,-3.4])
true_b=4.2
features,labels=synthetic_data(true_w,true_b,1000)

#绘制第二个特征与label的关系
d2l.set_figsize()
d2l.plt.scatter(features[:,
(1)].detach().numpy(),labels.detach().numpy(),1)

#获取小批量数据
def data_iter(batch_size,features,labels):
    num_examples=len(features)
    indices=list(range(num_examples))
    random.shuffle(indices)
    for i in range(0,num_examples,batch_size):
        batch_indices=torch.tensor(
        indices[i:min(i+batch_size,num_examples)])
        yield features[batch_indices],labels[batch_indices]

#获取一个包含10条数据的小批量数据集
batch_size=10
for X,y in data_iter(batch_size,features,labels):
    print(X,"\n",y)
    break

#初始化模型参数
w=torch.normal(0,0.01,size=(2,1),requires_grad=True)
b=torch.zeros(1,requires_grad=True)

#定义模型
def linreg(X,w,b):
    return torch.matmul(X,w)+b
```

```
45
46  #定义损失函数
47  def squared_loss(y_hat,y):
48      return(y_hat-y.reshape(y_hat.shape))**2/2
49
50  #定义优化算法
51  def sgd(params,lr,batch_size):
52      with torch.no_grad():
53          for param in params:
54              param-=lr*param.grad/batch_size
55              param.grad.zero_()
56
57  #训练
58  lr=0.03
59  num_epochs=3 #迭代周期
60  #总过进行了三轮训练，每轮又分为100个批次（batch_size为10），进行100次权值更新
61  for epoch in range(num_epochs):
62      for X,y in data_iter(batch_size,features,labels):
63          l=squared_loss(linreg(X,w,b),y)
64          l.sum().backward()
65          sgd([w,b],lr,batch_size)
66      with torch.no_grad():
67          train_l=squared_loss(linreg(features,w,b),labels)
68          print(f'epoch {epoch+1},loss{float(train_l.mean()):f}')
69
70  print(f'w的估计误差:{true_w-w.reshape(true_w.shape)}')
71  print(f'b的估计误差:{true_b-b}')
```

2.结果输出

今天主要看了李沐的书，把一些预备知识大概过了一遍，并初步学习了线性回归和softmax回归额实现，其中softmax的算法过程还没完全掌握。跟着代码实践确实有助于深化对算法的认识，但有时候又容易陷在具体的实现细节中无法自拔。明天争取把多层感知机和深度学习计算两部分看完。