TorchAutogardLinearRegression

2022 年 4 月 25 日

[ ]:
```
from csv import reader
import numpy as np

train_file=open("covid.train.csv")
rdr=reader(train_file)
a=[]
tar=[]
fst=True
for row in rdr:
    if fst:
        fst=False
    else:
        a.append(list(map(float,[1]+row[1:-1])))
        tar.append(float(row[-1]))
train_file.close()
x=np.array(a,dtype=np.float32)
y=np.array(tar,dtype=np.float32)
print(x.shape)
print(y.shape)
```

```
(2700, 94)
(2700,)
```

[ ]:
```
def normalize(x):
    nx=np.empty([x.shape[0],x.shape[1]],dtype=float)
    for j in range(1,x.shape[1]):
        now_mean=x[:,j].mean()
        now_dlt=x[:,j].max()-x[:,j].min()
        for i in range(x.shape[0]):
```

```
            nx[i][j]=(x[i][j]-now_mean)/now_dlt
    for i in range(x.shape[0]):
        nx[i][0]=1
    return nx
x=normalize(x)
```

[ ]:
```
import torch
dev='cuda' if torch.cuda.is_available() else 'cpu'
x=torch.from_numpy(x).to(dev)
y=torch.from_numpy(y).to(dev)
x=x.float()
y=y.float()
```

[ ]:
```
theta=torch.rand(x.shape[1],dtype=torch.float32).to(dev)
b=torch.zeros(1,dtype=torch.float32).to(dev)
print(theta.shape)
print(b)
theta.requires_grad_(requires_grad=True)
b.requires_grad_(requires_grad=True)
```

```
torch.Size([94])
tensor([0.], device='cuda:0')
```

[ ]:
```
tensor([0.], device='cuda:0', requires_grad=True)
```

[ ]:
```
def loss(theta):
    return ((theta@x.T+b-y)**2).sum()/(2*y.shape[0])
loss(theta)
```

[ ]:
```
tensor(0.4102, device='cuda:0', grad_fn=<DivBackward0>)
```

[ ]:
```
T=10000
alpha=0.0001
while T>0:
    l=loss(theta)
    l.backward()
    theta.data-=alpha*theta.grad
    b.data-=alpha*b.grad
    theta.grad.data.zero_()
```

```
    b.grad.data.zero_()
    #print(l)
    T-=1
    if l<0.4:
        break
```

[ ]:
```
print(b)
```

```
tensor([8.1562], device='cuda:0', requires_grad=True)
```

[ ]:
```
test_file=open("covid.test.csv")
rdrt=reader(test_file)
bx=[]
fst=True
for row in rdrt:
    if fst:
        fst=False
    else:
        bx.append(list(map(float,[1]+row[1:])))
test_file.close()
tx=np.array(bx,dtype=np.float32)
tx=normalize(tx)
tx=torch.from_numpy(tx).to(dev)
tx=tx.float()
print(tx.shape)
```

```
torch.Size([893, 94])
```

[ ]:
```
from csv import writer
outfile=open("result3.csv","w")
wtr=writer(outfile,lineterminator='\n')
header=["id","tested_positive"]
wtr.writerow(header)
for i in range(tx.shape[0]):
    wtr.writerow([i,(theta@tx[i]+b).item()])
outfile.close()
```