线性回归的简洁实现

Eng Teong Cheah

目录

线性回归的简洁实现

生成数据集

```python
from mxnet import autograd, nd

num_inputs = 2
num_examples = 1000
true_w = [2, -3.4]
true_b = 4.2
features = nd.random.normal(scale=1, shape=(num_examples, num_inputs))
labels = true_w[0] * features[:, 0] + true_w[1] * features[:, 1] +
true_b
labels += nd.random.normal(scale=0.01, shape=labels.shape)
```

读取数据

```python
# Save to the d2l package.
def load_array(data_arrays, batch_size, is_train=True):
    """Construct a Gluon data loader"""
    dataset = gluon.data.ArrayDataset(*data_arrays)
    return gluon.data.DataLoader(dataset, batch_size, shuffle=is_train)

batch_size = 10
data_iter = load_array((features, labels), batch_size)
```
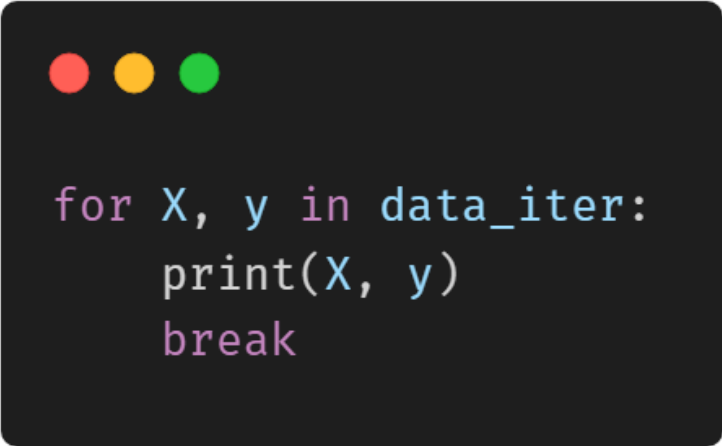
读取数据

```
for X, y in data_iter:
    print(X, y)
    break
```

```
[-1.130275   0.3379702 ]
[ 0.39320782  1.5395709 ]
[-0.06822178 0.72335476]
[-2.0644426  0.50765425]
[ 0.03313668  1.3074721 ]
[-0.52480155 0.3005414 ]
[ 1.4612247  0.9584059 ]
[ 0.4652423 -0.5965788 ]]
<NDArray 10x2 @cpu(0)>
[-2.7828512 12.15272    0.7970579 -
0.25150746 1.5961332 -1.6511697
 -0.18884937 2.1383817  3.8612688
7.1612334 ]
<NDArray 10 @cpu(0)>
```

定义模型

```
from mxnet.gluon import nn
net = nn.Sequential()
```

初始化模型参数

```
from mxnet import init
net.initialize(init.Normal(sigma=0.01))
```

定义损失函数

```python
from mxnet.gluon import loss as gloss
loss = gloss.L2Loss()  # The squared loss is also known as the L2 norm loss
```

定义优化算法

```python
from mxnet import gluon
trainer = gluon.Trainer(net.collect_params(), 'sgd', {'learning_rate':
0.03})
```

训练模型

```python
num_epochs = 3
for epoch in range(1, num_epochs + 1):
    for X, y in data_iter:
        with autograd.record():
            l = loss(net(X), y)
        l.backward()
        trainer.step(batch_size)
    l = loss(net(features), labels)
    print('epoch %d, loss: %f' % (epoch, l.mean().asnumpy()))
```

谢谢!

Does anyone have any questions?

Twitter: @walkercet

Blog: https://ceteongvanness.wordpress.com

资源

Dive into Deep Learning