Model selection

Training error

在训练数据的误差

Generalization error

在新数据的误差

Validation dataset

评估模型优秀程度的数据集（模考）

Test dataset

只用一次的数据集（高考）

Overfitting

产生的分析过于接近或精确地对应于一组特定的数据,因此可能无法可靠地拟合额外的数据或预测未来的观测结果

Underfitting

欠拟合是指模型在训练数据和测试数据上都表现不佳。这通常是因为模型过于简单,无法捕捉数据中的规律

Code Explain

指定一个三阶多项式

$$y = 5 + 1.2x - 3.4\frac{x^2}{2!} + 5.6\frac{x^3}{3!} + \epsilon \text{ where } \epsilon \sim \mathcal{N}(0, 0.1^2)$$

```python
import torch
from torch import nn
from d2l import torch as d2l
max_degree = 20
n_train, n_test = 100, 100
true_w = np.zeros(max_degree)
true_w[0:4] = np.array([5,1.2,-3.4,5.6])
features = np.random.normal(size=(n_train+n_test,1))
np.random.shuffle(features)
poly_features = np.power(features, np.arange(max_degree).reshape(1,-1)) # 取相应次方
for i in range(max_degree):
    poly_features[:,i] /= math.gamma(i+1) # 阶乘
labels = np.dot(poly_features,true_w)
labels += np.random.normal(scale=0.1,size=labels.shape)

true_w, features, poly_features, labels = [torch.tensor(x,dtype=torch.float32) for x
in [true_w, features, poly_features, labels]]
```

```
def evaluate_loss(net, data_iter, loss):
    """评估给定数据集上模型的损失"""
    metric = d2l.Accumulator(2) # 累加
    for X, y in data_iter:
        out = net(X)
        y = y.reshape(out.shape) # 将真实标签改为网络输出标签的形式，统一形式
        l = loss(out, y)
        metric.add(l.sum(), l.numel()) # 平均
    return metric[0] / metric[1]

def train(train_features, test_features, train_labels, test_labels, num_epochs=400):
    loss = nn.MSELoss()
    input_shape = train_features.shape[-1]
    net = nn.Sequential(nn.Linear(input_shape, 1, bias=False))
    batch_size = min(10,train_labels.shape[0])
    train_iter = 
d2l.load_array((train_features,train_labels.reshape(-1,1)),batch_size)
    test_iter = 
d2l.load_array((test_features,test_labels.reshape(-1,1)),batch_size,is_train=False)
    trainer = torch.optim.SGD(net.parameters(),lr=0.01)
    animator = d2l.Animator(xlabel='epoch',ylabel='loss',yscale='log',xlim=
[1,num_epochs],ylim=[1e-3,1e2],legend=['train','test'])
    for epoch in range(num_epochs):
        d2l.train_epoch_ch3(net, train_iter, loss, trainer)
        if epoch == 0 or (epoch + 1) % 20 == 0:
            animator.add(epoch + 1, (evaluate_loss(net, train_iter, loss),
evaluate_loss(net,test_iter,loss)))
    print('weight',net[0].weight.data.numpy())

# 三阶多项式函数拟合(正态)
train(poly_features[:n_train,:4],poly_features[n_train:,:4],labels[:n_train],labels[n
_train:])  # 最后返回的weight值和公式真实weight值很接近
d2l.plt.show()

# 一阶多项式函数拟合(欠拟合)
# 这里相当于用一阶多项式拟合真实的三阶多项式，欠拟合了，损失很高，根本就没降
train(poly_features[:n_train,:2],poly_features[n_train:,:2],labels[:n_train],labels[n
_train:])
d2l.plt.show()

# 十九阶多项式函数拟合(过拟合)
# 这里相当于用十九阶多项式拟合真实的三阶多项式，过拟合了
train(poly_features[:n_train,:],poly_features[n_train:,:],labels[:n_train],labels[n_t
rain:])
d2l.plt.show()
```