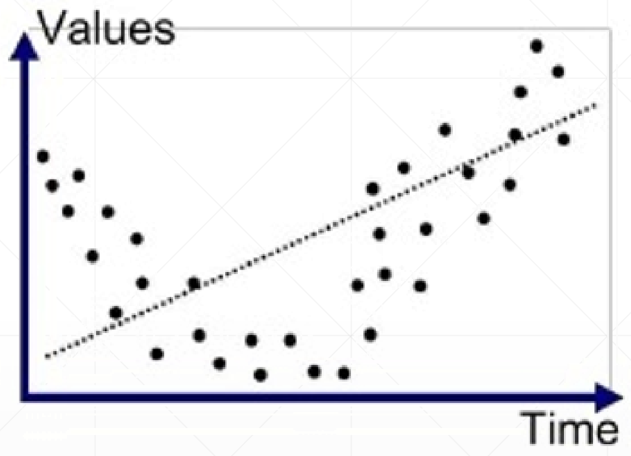
PyTorch

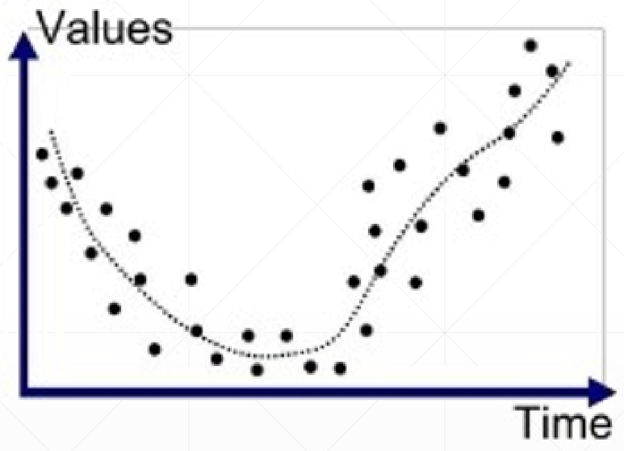
Train-Val-Test划分

主讲人：龙良曲

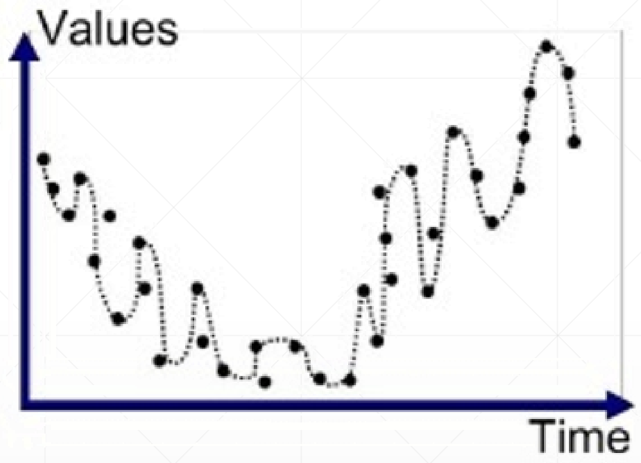
Recap



Underfitted

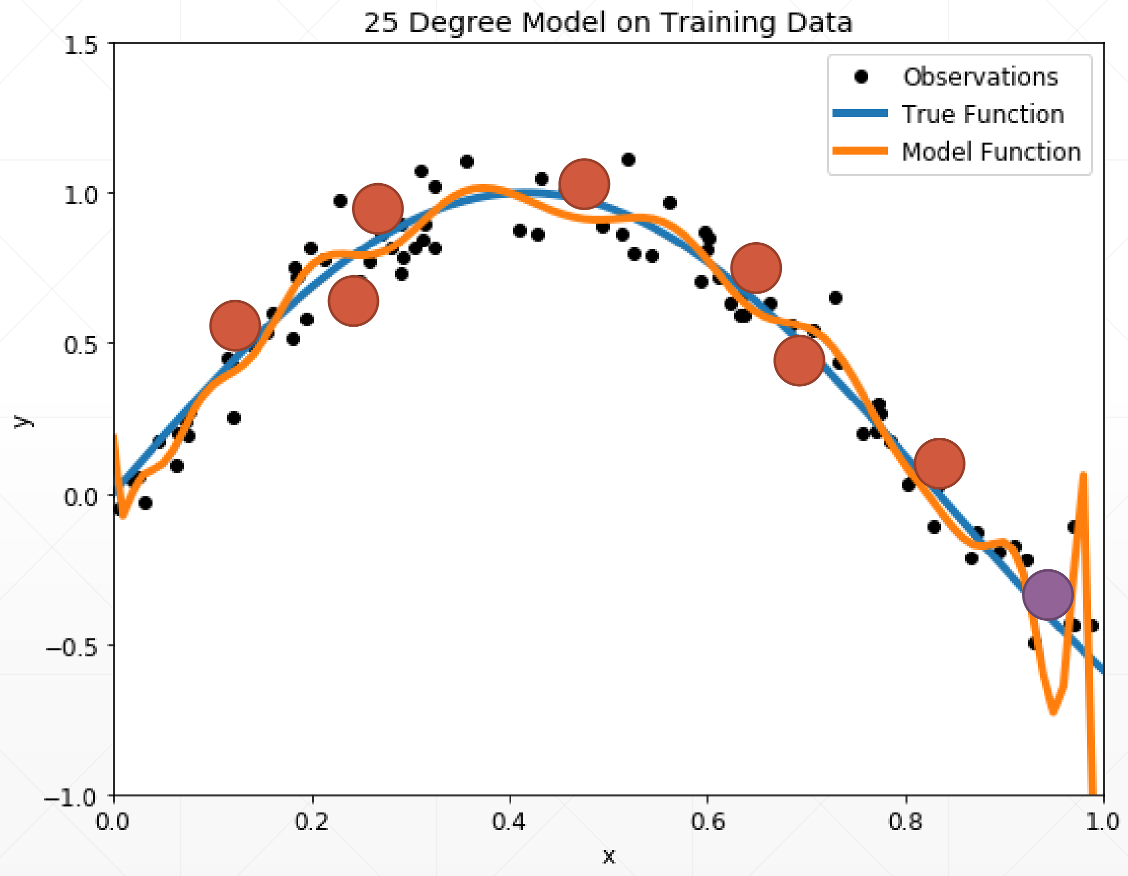


Good Fit/Robust



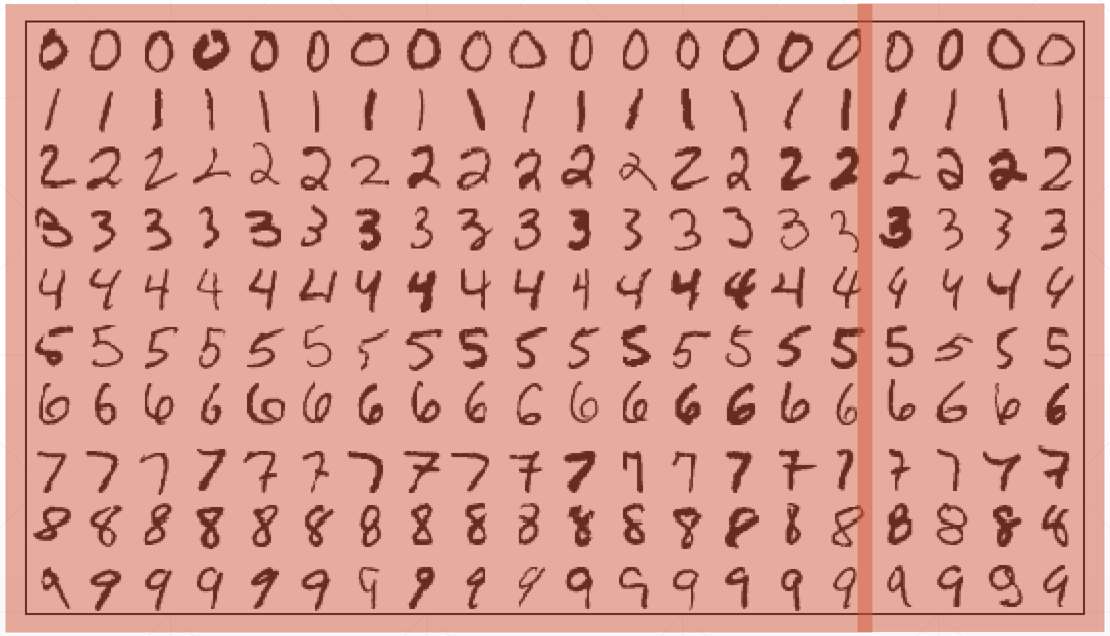
Overfitted

How to detect



Splitting

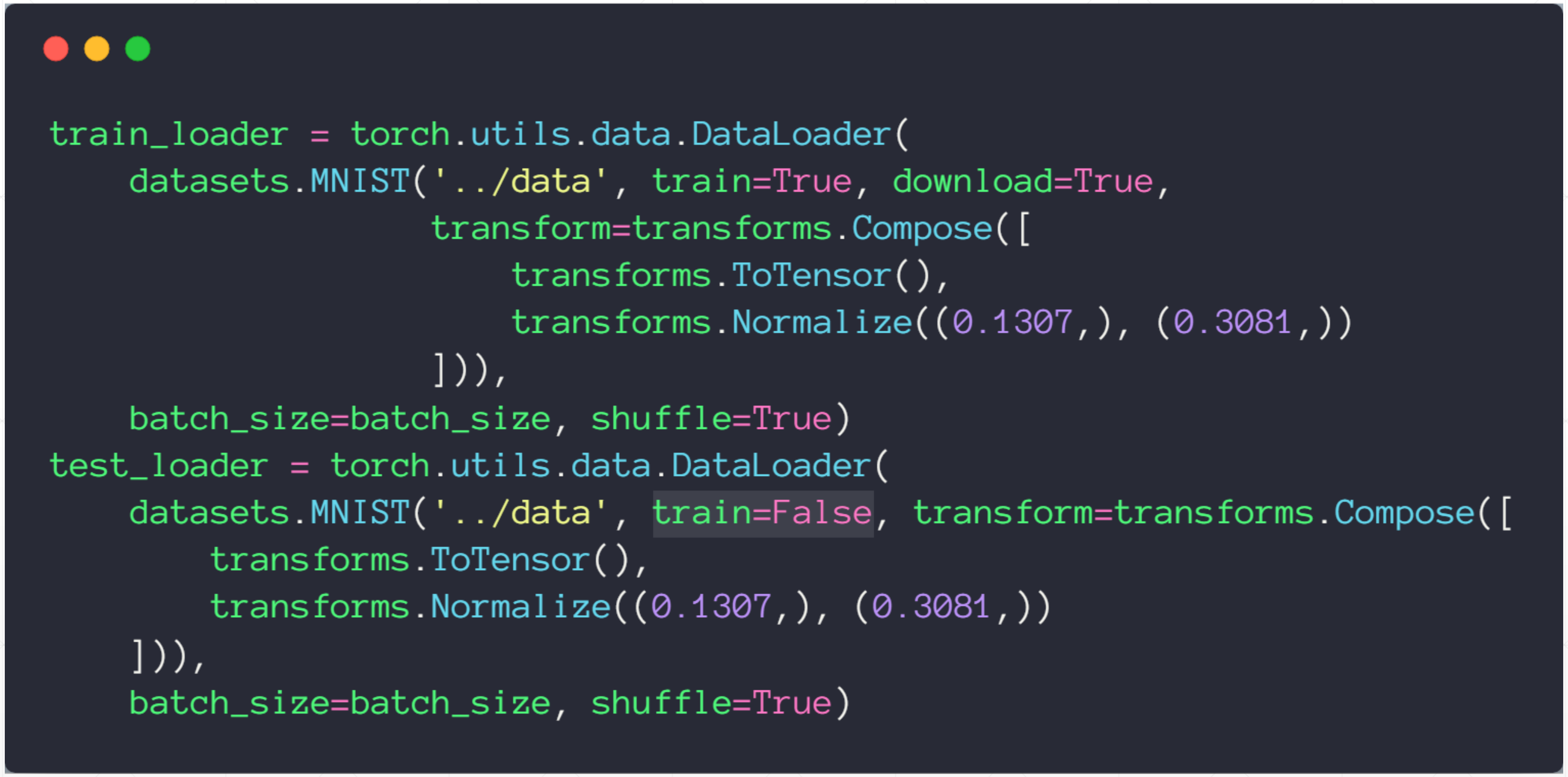
Train Set



Test Set

For example

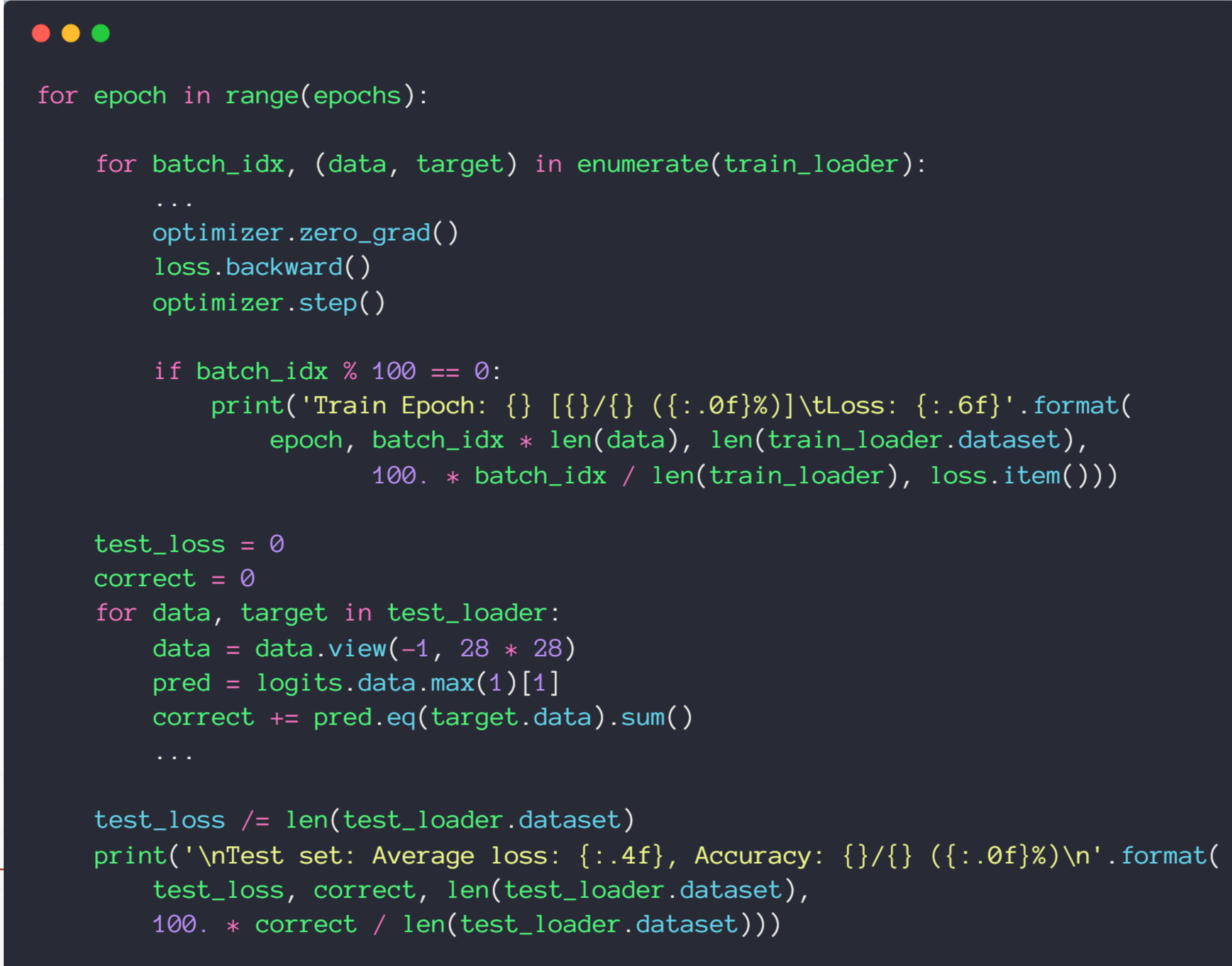
60K



```
train_loader = torch.utils.data.DataLoader(
    datasets.MNIST('../data', train=True, download=True,
                   transform=transforms.Compose([
                       transforms.ToTensor(),
                       transforms.Normalize((0.1307,), (0.3081,))
                   ])),
    batch_size=batch_size, shuffle=True)
test_loader = torch.utils.data.DataLoader(
    datasets.MNIST('../data', train=False, transform=transforms.Compose([
        transforms.ToTensor(),
        transforms.Normalize((0.1307,), (0.3081,))
    ])),
    batch_size=batch_size, shuffle=True)
```

10K

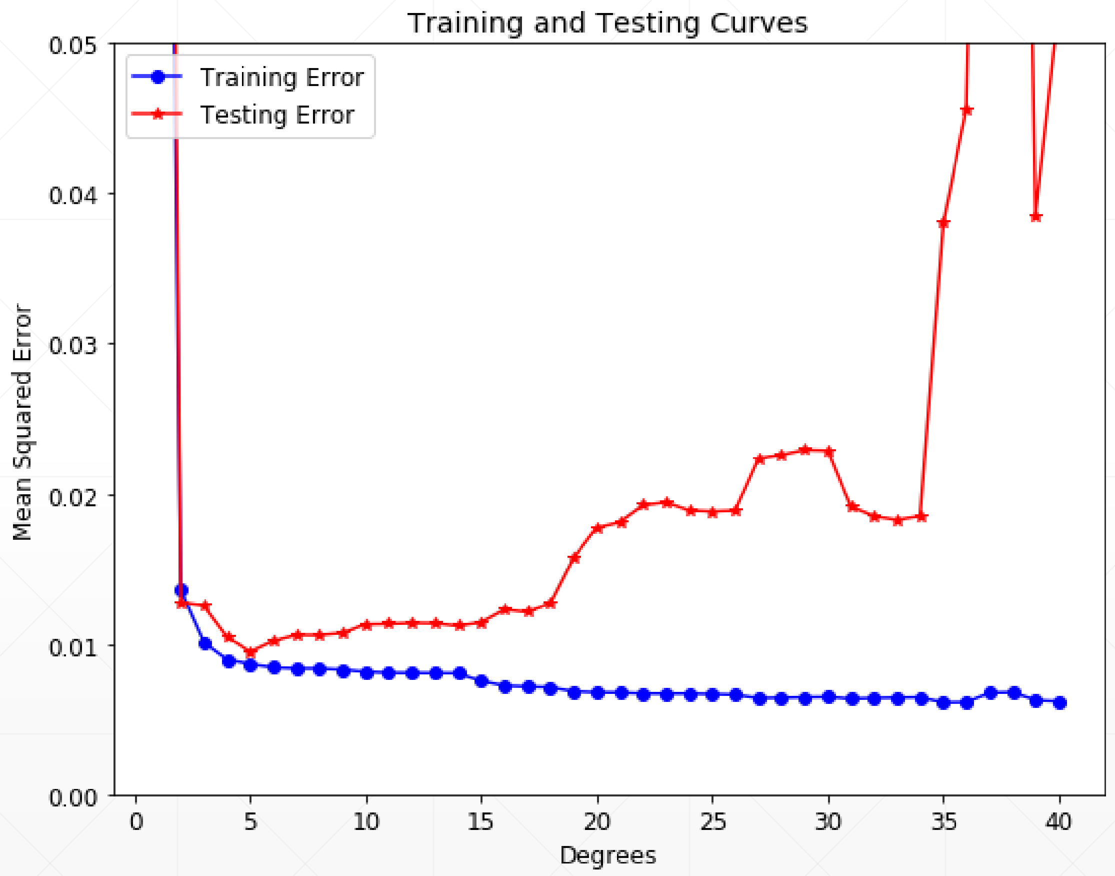
test while train



```python
for epoch in range(epochs):

    for batch_idx, (data, target) in enumerate(train_loader):
        ...
        optimizer.zero_grad()
        loss.backward()
        optimizer.step()

        if batch_idx % 100 == 0:
            print('Train Epoch: {} [{}/{} ({:.0f}%)]\tLoss: {:.6f}'.format(
                epoch, batch_idx * len(data), len(train_loader.dataset),
                       100. * batch_idx / len(train_loader), loss.item()))

    test_loss = 0
    correct = 0
    for data, target in test_loader:
        data = data.view(-1, 28 * 28)
        pred = logits.data.max(1)[1]
        correct += pred.eq(target.data).sum()
        ...

    test_loss /= len(test_loader.dataset)
    print('\nTest set: Average loss: {:.4f}, Accuracy: {}/{} ({:.0f}%)\n'.format(
        test_loss, correct, len(test_loader.dataset),
        100. * correct / len(test_loader.dataset)))
```

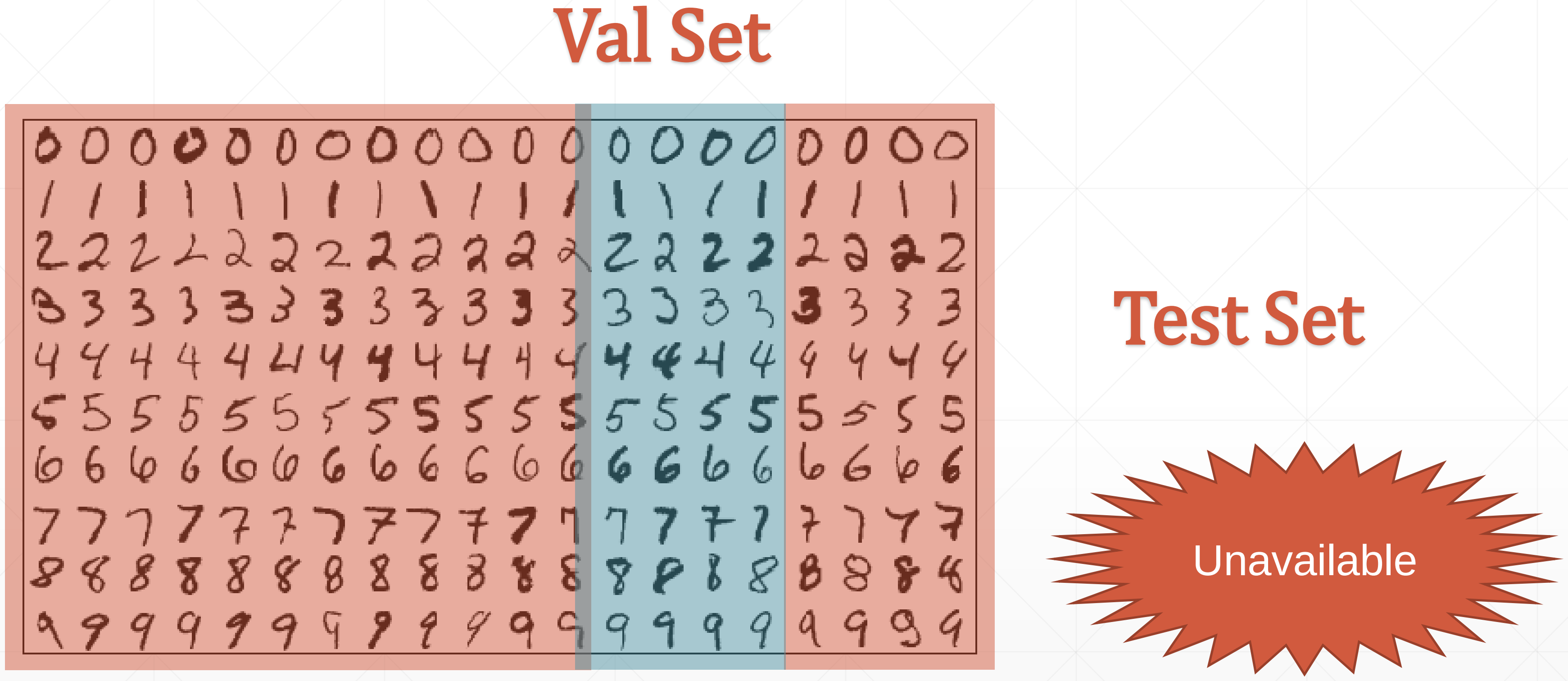
train test trade-off

Overfitting

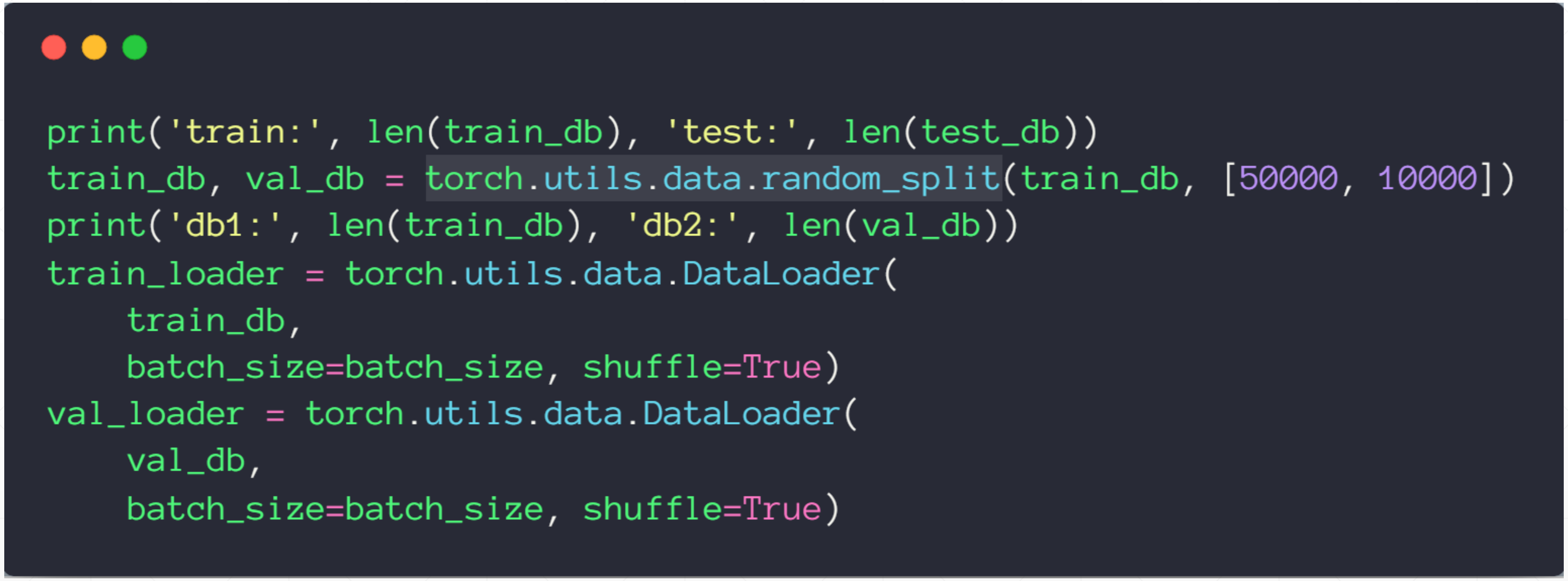


For others judge

- Kaggle

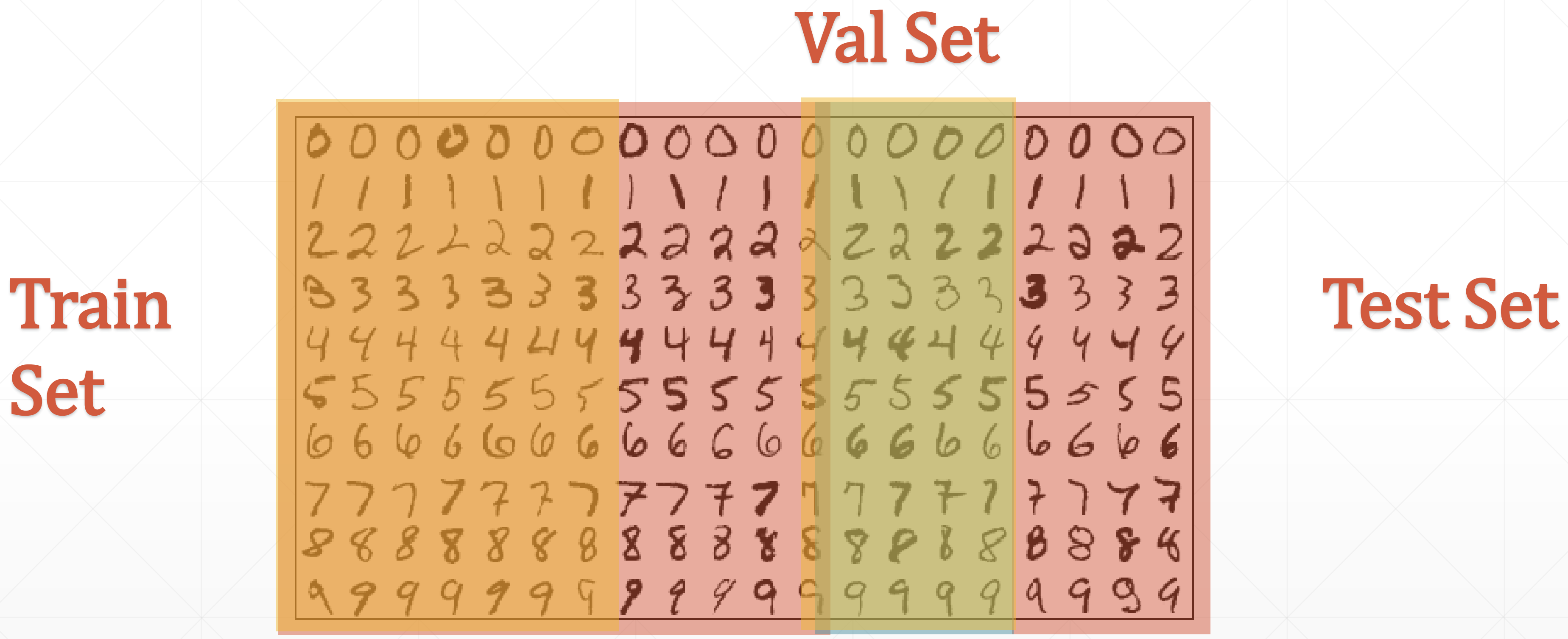


train-val-test



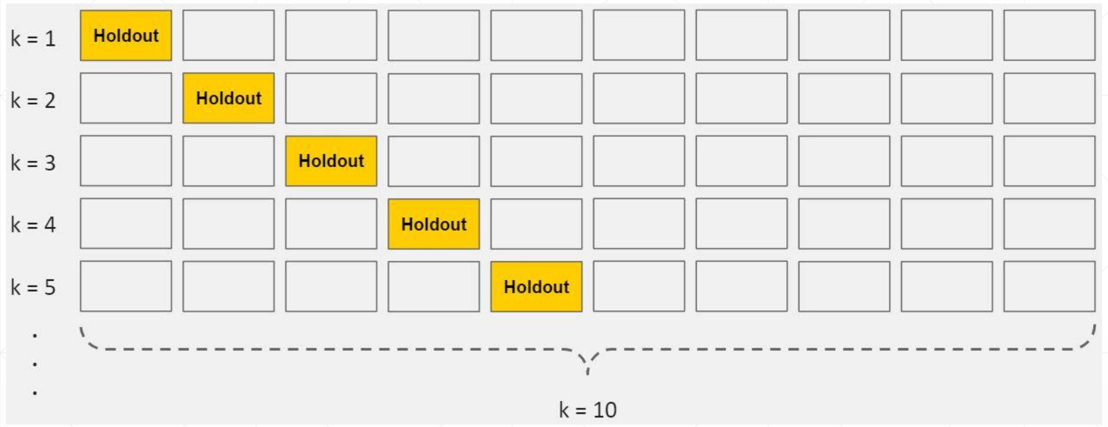
```
print('train:', len(train_db), 'test:', len(test_db))
train_db, val_db = torch.utils.data.random_split(train_db, [50000, 10000])
print('db1:', len(train_db), 'db2:', len(val_db))
train_loader = torch.utils.data.DataLoader(
    train_db,
    batch_size=batch_size, shuffle=True)
val_loader = torch.utils.data.DataLoader(
    val_db,
    batch_size=batch_size, shuffle=True)
```

K-fold cross-validation



k-fold cross validation

- merge train/val sets
- randomly sample 1/k as val set



下一课时

减轻Overfitting

Thank You.