

# Advanced Machine Learning Subsidiary Notes

## Lecture 2: Over-Fitting

Adam Prügel-Bennett

February 27, 2024

## 1 Keywords

- Overfitting, regularisation, feature selection

## 2 Main Points

- Over-fitting
  - Over-fitting is when we learn **spurious rules** that explain the training set
  - If we use an infinitely flexible machine then we can't generalise at all
- Controlling Complexity
  - More training example improves generalisation by allowing us to eliminate spurious rules
  - To achieve good generalisation performance machine must match the underlying structure of the problem
  - By preprocessing or careful feature engineering you can sometimes make the learning task easier
  - Deep learning works because
    - \* It uses a lot of training data
    - \* They use convolutions that
      - have many fewer parameter to learn than fully connected layers
      - respect translation invariance
      - find local features (at different scales through the network)
  - We usually don't know the structure of the inputs (they are too high dimensional)
    - \* Try different machines to see what fits
    - \* Fine-tune the models (adjusting hyper-parameters)
    - \* Feature Engineering very often help
      - Feature selection
      - Using PCA or clustering
      - Normalise your input features
    - \* Need a validation set to do this
    - \* Beware you are very likely to over-estimate your generalisation error if you use your test set to select your model
    - \* Can use  $K$ -fold cross-validation to get a better estimate of generalisation error
- Regularisation
  - Adding regularisation terms can help choose a simpler model

- $L_2$  regularisers punish large weights making the fitting function smoother
- $L_1$  regularisers can do automatic feature selection
- Can do subtle forms of regularisation such as choosing a maximum margin solution

## 3 Exercises

### 3.1 Estimating the error in the mean

- Consider a binary classification problem
- Suppose we measure the accuracy on a validation set of size  $n = 10\,000$
- If we correctly predict 8 000 of the validation set what is
  1. an estimate of the mean accuracy?
  2. the accuracy of this estimate?
- State you assumptions
- What would the answer be if  $n = 100$  and we got 80 correct on the validation set?
- Answer at the end

## 4 Experiments

### 4.1 Overfitting Game

- Write your own simulation (see lecture notes)
- Generate  $n$  random numbers  $X_i \sim U(0, 3)$  (you can try different distributions)
- Add some noise to generate  $Y_i = X_i + Z_i$  where  $Z_i \sim \mathcal{N}(0, 1)$
- Choose  $i^*$  with the largest value of  $Y_i$
- Compute  $\Delta = Z_{i^*} = Y_{i^*} - X_{i^*}$
- Repeat many times and plot a histogram

### 4.2 Cross-validation

- Try using cross validation

```
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn import datasets
from sklearn import svm
```

```
X, y = datasets.load_iris(return_X_y=True)
X.shape, y.shape
```

```
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.4, random_state=0)
```

```
X_train.shape, y_train.shape
```

```
X_test.shape, y_test.shape
```

```
clf = svm.SVC(kernel='linear', C=1).fit(X_train, y_train)
clf.score(X_test, y_test)
```

### 4.3 Practical work

- There are a lot of practical tips that you can use in your project
  - Try them out and see what works and what doesn't
  - Many modifications won't make much difference, but occasionally one will make a lot of difference
  - Be very wary of estimating your generalisation performance
    - \* You are measuring on a finite validation set so you will have errors
    - \* Once you've used your validation set for selecting models, your likely to over-estimate your generalisation performance (under-estimate the generalisation error)

## 5 Answers

### 5.1 Estimating the error in the mean

- We assume that the examples in the validation set are independent so the number of successes,  $k$ , will be binomially distributed

$$\text{Bin}(k|n, p) = \binom{n}{k} p^k (1 - p)^{n-k}$$

- We don't know the success probability  $p$ , but an unbiased estimate of it is the number of observed success divided by  $n$ , that is 0.8
- For a binomial the variance in  $k$  is given by  $np(1 - p)$  or 1600 if  $p = 0.8$  and  $n = 10\,000$
- The typical size of the fluctuations in  $k$  would be  $\sqrt{np(1 - p)} = 40$
- Our estimate of  $p$  would therefore typically have fluctuations of size  $40/n = 0.004$
- Thus our estimated success rate is  $0.8 \pm 0.004$  or 80% with an error of about half a percent
- This is small because we used a large validation set. If  $n = 100$  and we got 80 successes then we would have an error of  $0.8 \pm 0.04$  so we have an error of 4%.
- Note that by choosing the best of 10 machines we might expect to over-estimate the performance by around 5% in this case