Deep Learning

2.2 Over and Under fitting

Dr. Konda Reddy Mopuri
kmopuri@iittp.ac.in
Dept. of CSE, IIT Tirupati

Generalization

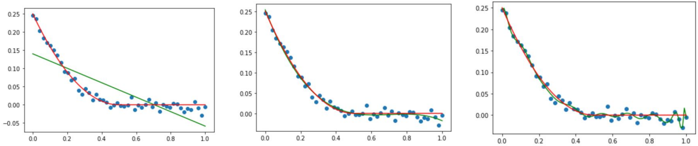
1. Ability of an ML model to perform on unseen data

Generalization

1. Ability of an ML model to perform on unseen data
2. Goal of good ML model is to generalize well from training data to any data from the task domain

Fit

1. Refers to how well the model can approximate a target



Fit

1. Refers to how well the model can approximate a target function
2. Goodness of the fit refers to measures used to estimate how well the approximation matches the target

Fit

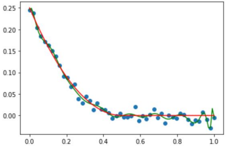
1. Refers to how well the model can approximate a target function
2. Goodness of the fit refers to measures used to estimate how well the approximation matches the target
3. In ML we don't know the target function under approximation

Over and under fitting

1. Cause of poor performance in ML is either overfitting or underfitting to the data

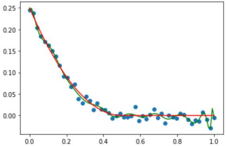
Overfitting

1. Refers to a model which learns the training data too well



Overfitting

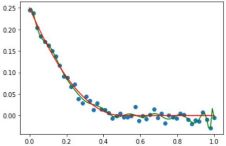
1. Refers to a model which learns the training data too well



2. Model learns the noise and random fluctuations in the data as concepts (to an extent that affects its generalization)

Overfitting

1. Refers to a model which learns the training data too well



2. Model learns the noise and random fluctuations in the data as concepts (to an extent that affects its generalization)
3. More likely to occur in case of nonparametric and nonlinear models with more flexibility

Example

1. Decision trees are a nonparametric model

Example

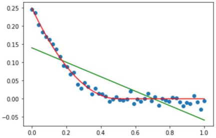
1. Decision trees are a nonparametric model
2. Flexible and prone to overfitting training data

Example

1. Decision trees are a nonparametric model
2. Flexible and prone to overfitting training data
3. Can be addressed by pruning the tree after learning (removes some of the detail picked up)

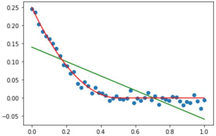
Underfitting

1. Refers to a scenario where the model can neither model the training data nor generalize to new data



Underfitting

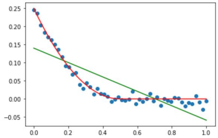
1. Refers to a scenario where the model can neither model the training data nor generalize to new data



2. Obvious since the performance on the training data is poor (hence often not discussed)

Underfitting

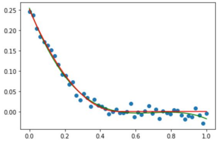
1. Refers to a scenario where the model can neither model the training data nor generalize to new data



2. Obvious since the performance on the training data is poor (hence often not discussed)
3. Can be alleviated by trying alternate ML algorithms (e.g. relatively complex)

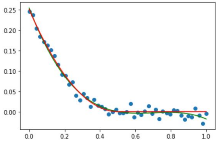
Good fit in ML

1. Ideally, one should select a model at the sweet spot between over and underfitting



Good fit in ML

1. Ideally, one should select a model at the sweet spot between over and underfitting



2. Very difficult in practice

Good fit in ML

1. One can observe the behavior of the model during the training

Figure credits: https://ds100.org/

Good fit in ML

1. One can observe the behavior of the model during the training
    2. Error on train and held out/validation sets

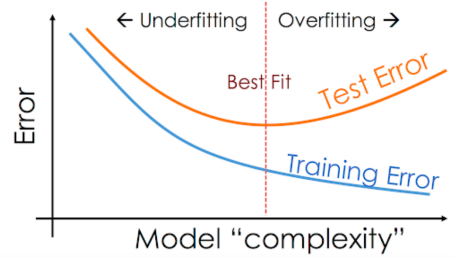
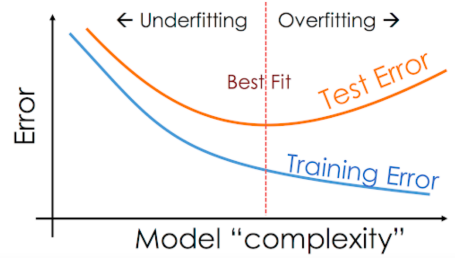


Figure credits: https://ds100.org/

Good fit in ML

1. One can observe the behavior of the model during the training
    2. Error on train and held out/validation sets



3. Cross validation is often used for estimating the generalization (hence limit overfitting)

Figure credits: https://ds100.org/

Capacity

1. Vaguely, it is the ability to model an arbitrary function

Capacity

1. Vaguely, it is the ability to model an arbitrary function
2. More rigorous notion is VC dimension

Capacity

1. Although it is difficult to define precisely, in practice it is not very hard to manipulate it for a given class of models

Capacity

1. Although it is difficult to define precisely, in practice it is not very hard to manipulate it for a given class of models
2. In general overfitting can be controlled by
   - Restricting the space of functions $\mathcal{F}$ (regularization, constrained optimization)
   - Making the choice of optimal function f^* less dependent on the data (e.g. ensemble methods)

Polynomial Model

1. Given a polynomial model
   $\forall x, \alpha_0, \ldots, \alpha_D \in \mathcal{R}, f(x, \alpha) = \sum_{\mathbf{d=0}}^{\mathbf{D}} \alpha_{\mathbf{d}} \mathbf{x}^{\mathbf{d}}$,
   and a training set $(x_n, y_n) \in \mathcal{R}^2, n = 1, \ldots, N$, the quadratic loss is

Polynomial Model

1. Given a polynomial model
   $\forall x, \alpha_0, \ldots, \alpha_D \in \mathcal{R}, f(x, \alpha) = \sum_{\mathbf{d=0}}^{\mathbf{D}} \alpha_{\mathbf{d}} \mathbf{x}^{\mathbf{d}}$,
   and a training set $(x_n, y_n) \in \mathcal{R}^2, n = 1, \ldots, N$, the quadratic loss is

2.

$$
\begin{aligned}
\mathcal{L}(\alpha) &= \sum_n (f(x_n; \alpha) - y_n)^2 \\
&= \sum_n (\sum_{d=0}^{D} \alpha_d x^d - y_n)^2 \\
&= \left\| \begin{bmatrix} x_1^0 & \ldots & x_1^D \\ \vdots & \ddots & \vdots \\ x_N^0 & \ldots & x_N^D \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \vdots \\ \alpha_D \end{bmatrix} - \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} \right\|^2
\end{aligned}
$$

Polynomial Model

- $\underset{\alpha}{\operatorname{argmin}} \left\| \begin{bmatrix} x_1^0 & \dots & x_1^D \\ \vdots & \ddots & \vdots \\ x_N^0 & \dots & x_N^D \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \vdots \\ \alpha_D \end{bmatrix} - \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} \right\|^2$ is a standard least squares problem

Polynomial Model- Prediction with degree