CS 337, Fall 2023
Basis Functions for Linear Regression, Under/Overfitting

Scribes: Mridul Agarwal, Hruday Nandan Tudu, Soupati Sri Nithya
Arnav Aditya Singh, Chaitanya Garg, Rajkumar
Edited by: Ashwin Ramachandran

Monday 7th August, 2023

Disclaimer. Please note this document has not received the usual scrutiny that formal publications enjoy. This may be distributed outside this class only with the permission of the instructor.

1 Recap

- The hypothesis $h_{\mathbf{w}}(\mathbf{x}) = w_0 + w_1x_1 + \ldots + w_dx_d = \mathbf{w}^T\mathbf{x}$ represents a linearly weighted combination of features. Hypothesis class, $H = \{h_{\mathbf{w}} : \mathbf{w} \in \mathbb{R}^{d+1}\}$ where $d+1$ is the number of features in each datapoint.

$$h_{\mathbf{w}}(\mathbf{x}) = \mathbf{w}^T\mathbf{x},$$

  where

$$\mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_d \end{bmatrix}_{(d+1)\times 1}, \quad \mathbf{x} = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_d \end{bmatrix}_{(d+1)\times 1}$$

- Least squares loss: $L(\mathbf{w}, D_{\text{train}}) = \sum_i (y_i - \mathbf{w}^T\mathbf{x}_i)^2$ where $\mathbf{x}_i$ is a $(d+1)$ dimensional datapoint, y_i is the target value, $\mathbf{w}$ is the parameter or weight vector.
- Least squares solution: $W_{\text{LS}} = \arg\min_{\mathbf{w}} \sum_i (y_i - \mathbf{w}^T\mathbf{x}_i)^2$, $\quad W_{\text{LS}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}$

  where $\mathbf{X} = \begin{bmatrix} \longleftarrow \mathbf{x}_1^T \longrightarrow \\ \vdots \\ \longleftarrow \mathbf{x}_n^T \longleftarrow \end{bmatrix}_{n\times(d+1)}, \quad \mathbf{x}_i = \begin{bmatrix} 1 \\ x_{i1} \\ \vdots \\ x_{id} \end{bmatrix}_{(d+1)\times 1}, \quad \mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_d \end{bmatrix}_{(d+1)\times 1}$

2 Basis Functions

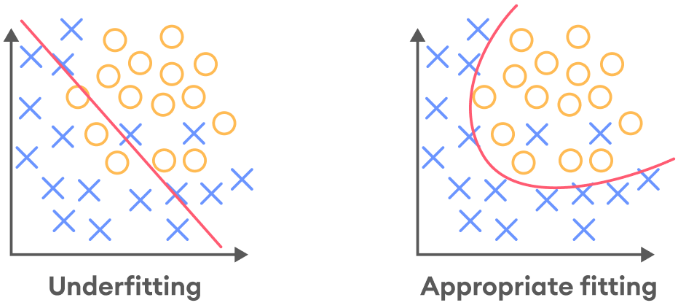

Figure 1: Basis Transformation

- In the above figure, the left-side model represent $h_{\mathbf{w}}(x) = w_0 + w_1 x$ for some w_0, w_1, but it doesn't fit the given dataset well.
- Right-side model fits the dataset well, and represents $h_{\mathbf{w}}(x) = w_0 + w_1 x + w_2 x^2$ for some $w_0, w_1, w2$.
- This suggests we may want to change/transform the feature space we are working with for improved model fits. We detail the procedure below using basis functions.

Given a basis function $\phi : \mathcal{X} \to \mathbb{R}^{m+1}$, where

$$\phi(\mathbf{x}) = \begin{bmatrix} 1 \\ \phi_1(\mathbf{x}) \\ \phi_2(\mathbf{x}) \\ \vdots \\ \phi_m(\mathbf{x}) \end{bmatrix}_{(m+1)\times 1}$$

The hypothesis space becomes

$$\mathcal{H}_\phi = \left\{ h_{\mathbf{w}} \in \mathcal{X}^{\mathbb{R}} \middle| \mathbf{w} \in \mathbb{R}^{m+1}, h_{\mathbf{w}}(\mathbf{x}) = \mathbf{w}^\top \phi(\mathbf{x}) \text{ for } \mathbf{x} \in \mathcal{X} \right\}$$

The design matrix with basis function-based transformations can be written as:

$$\mathbf{\Phi}_{\mathcal{D}} = \begin{bmatrix} \phi(\mathbf{x}_1)^\top \\ \phi(\mathbf{x}_2)^\top \\ \vdots \\ \phi(\mathbf{x}_n)^\top \end{bmatrix}_{n\times(m+1)}$$

Note that $\mathbf{\Phi}_{\mathcal{D}}$ yields m-dimensional feature vectors, that could be smaller or larger than the original d-dimensional input feature space.

The least-squares objective using basis functions can be written as:

$$\begin{aligned}\mathcal{L}_{\mathrm{LS}}(h_{\mathbf{w}}, \mathcal{D}) &= \frac{1}{n}\sum_{i=1}^{n}(y_i - h_{\mathbf{w}}(\mathbf{x}_i))^2 \\ &= \frac{1}{n}\sum_{i=1}^{n}(y_i - \mathbf{w}^\top \phi(\mathbf{x}_i))^2 \\ &= \frac{1}{n}\|\mathbf{y} - \mathbf{\Phi}_{\mathcal{D}}\mathbf{w}\|_2^2\end{aligned}$$

and as before

$$\mathbf{w}_{\mathrm{LS}}^{*} = \left(\mathbf{\Phi}_{\mathcal{D}}^\top \mathbf{\Phi}_{\mathcal{D}}\right)^{-1} \mathbf{\Phi}_{\mathcal{D}}^\top \mathbf{y}$$

2.1 Examples of Basis Functions

- **Polynomial Basis**
  For 1-D data, $\Phi_j(x) = x^j$ for $1 \leq j \leq d$.
- **Radial Basis Function(RBF)**
  For d-dimensional data, $\Phi_j(\mathbf{x}) = e^{-\frac{\|\mathbf{x}-\mu_j\|_2^2}{\sigma_j^2}}$ for $\mathbf{x}, \mu_j \in \mathbb{R}^d, \sigma_j \in \mathbb{R}$.
- **Fourier Basis**
- **Piecewise Linear Basis**
- **Periodic Basis (**$\sin(x)$**,** $\cos(x)$ **etc.)**

Data Splits (Preliminaries) : The original data in a machine learning model is typically taken and split into three sets.

- Training data (Training set): Dataset used to train the model and learn the parameters. (Below, n is the number of data points in the training set)

$$D = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$$

$$\text{Training Error : } \frac{\Sigma_{i=1}^{n}(y_i - h_{\mathbf{w}}(\mathbf{x}_i))^2}{n}$$

- Development Set (Validation Set) : Mainly used to tune *hyperparameters* of the model. Hyperparameters are not learnable parameters, and are instead predetermined quantities that are used with the model. For example, σ in an RBF basis function is a hyperparameter.
- Test Set (Evaluation Set) : This is the unseen/test set used for a final evaluation of the model, after choosing the best hyperparameters determined via tuning on the development set. The error on the test set is also referred to as the generalization error.

Errors on the development/test sets are a measure of the generalization ability of the trained machine learning model.

3 Hyperparameter tuning

Two general methods :

- Grid Search: Define a search space as a grid of hyperparameter values and evaluate every position in the grid.
- Random Search: Define a search space as a bounded domain of hyperparameter values and randomly sample points in that domain.

3.1 Underfitting

Underfitting is a scenario where a data model is overly simple and unable to capture the relationship between the input and output variables accurately.

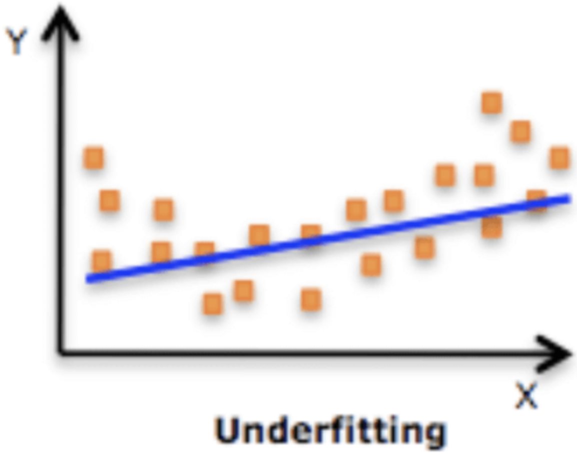

Figure 2: Model is too simple and underfits the data

3.2 Overfitting

Overfitting is an undesirable machine learning behavior that occurs when the machine learning model gives accurate predictions for training data but not for new data (i.e., does not generalise well). Model fits the training data perfectly but is overly complex.

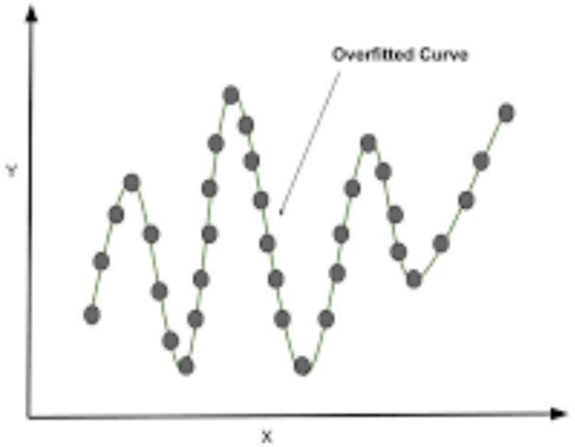

Figure 3: Model fits to the training data perfectly but is overly complex

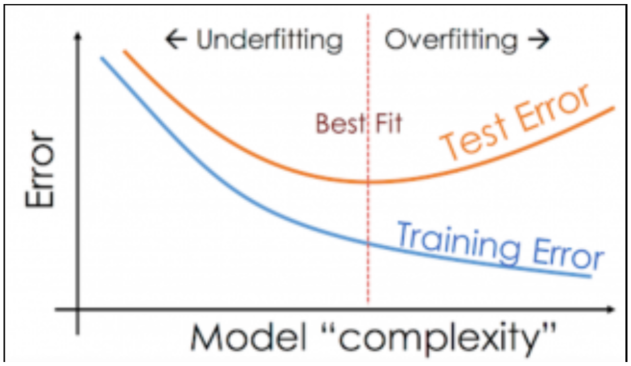

Figure 4: Hyperparameter tuning