Model Selection and Optimization

Satyanarayana Gajjarapu
AI24BTECH11009
Department of Artificial Intelligence
`ai24btech11009@iith.ac.in`

The goal in machine learning is to select a hypothesis that will optimally fit future examples. A stationarity assumption is made that future examples are similar to the past examples. Also assume each E_j has the same prior probability distribution and is independent of the previous examples.

$$\mathbf{P}(E_j) = \mathbf{P}(E_{j+1}) = \mathbf{P}(E_{j+2}) = \cdots$$
$$\mathbf{P}(E_j) = \mathbf{P}(E_j|E_{j-1}, E_{j-2}, \cdots)$$

Examples that satisfy these equations are called **independent and identically distributed** (i.i.d). **Hyperparameters** are the parameters of the model class, not of an individual model.

Available examples are divided into two sets, a **training set** to create the hypothesis and a **test set** to evaluate it. Apart from training set and test set, a **validation set** (development set or dev set) is also required to evaluate and select best candidate model. The technique of using data as both training data and validation data by dividing into k equal subsets is called k**-fold cross-validation**. The extreme case of $k = n$ is called **leave-one-out cross-validation** (LOOCV).

1 Model Selection

In finding a good hypothesis, **model selection**, to find good hypothesis space and **optimization** (training), to find best hypothesis within the space are required. For many model classes, the training set error reaches zero as the complexity increases. Model selection picks the value where it has least validation error rate. Model classes start to overfit as the capacity approaches the point of interpolation.

2 Loss function and Loss

The main aim of a model is to maximize the utility function, but in machine learning it is expressed as to minimize the loss function. The loss function $L(x, y, \hat{y})$ is defined as the amount of utility lost by predicting $h(x) = \hat{y}$ when the correct answer is $f(x) = y$.

$L(x, y, \hat{y}) =$ Utility(result of using y given an input x) - Utility(result of using $\hat{y}$ given an input x)

A simplified version of loss function is $L(y, \hat{y})$, which is independent of x. The different types of loss functions are **Absolute-value loss**: $L_1(y, \hat{y}) = |y - \hat{y}|$ and **Squared-error loss**: $L_2(y, \hat{y}) = (y - \hat{y})^2$. Expected **generalization loss** for a hypothesis h (with respect to loss function L) is

$$GenLoss_L(h) = \sum_{(x,y)\in\varepsilon} L(y, h(x))P(x, y)$$
$$h^* = \operatorname*{argmin}_{h\in\mathcal{H}} GenLoss_L(h)$$

Preprint. Under review.

Where P is the probability distribution over examples, ε is the set of all possible input–output examples and h^* is the best hypothesis with the minimum expected generalization loss. The learning agent can only estimate generalization loss with **empirical loss** on a set of examples E of size N,

$$EmpLoss_{L,E}(h) = \sum_{(x,y)\in E} L(y, h(x))\frac{1}{N}$$
$$\hat{h}^* = \operatorname*{argmin}_{h\in\mathcal{H}} EmpLoss_{L,E}(h)$$

Where $\hat{h}^*$ is the estimated best hypothesis with minimum empirical loss. $\hat{h}^*$ may differ from the true function f due to four reasons unrealizability, variance, noise and computational complexity.

3 Regularization and Hyperparameter tuning

The weighted sum of empirical loss and the complexity of the hypothesis is called total cost. λ is hyperparameter.

$$Cost(h) = EmpLoss(h) + \lambda Complexity(h)$$
$$\hat{h}^* = \operatorname*{argmin}_{h\in\mathcal{H}} Cost(h)$$

The process of explicitly penalizing complex hypotheses is called **regularization**. The major use of regularization is to make a model simple and to improve model's generalization ability. The simplest approach to hyperparameter tuning is **hand tuning**, which uses parameter values from past experiences to train the model and suggests new parameter values. Bayesian optimization, Gaussian process and population-based training (PBT) are used to address hyperparameter tuning tasks.