Tutorial 5, Mar 5 2021

Blue Estimators For Mean / Var

For samples x_i with true mean and var, μ, σ^2.

$$\bar{X} = \frac{1}{N}\sum_{i=1}^{N} x_i$$

Thus $E((\bar{x}-\mu)) = \frac{\sigma^2}{N}$, larger the sample, smaller the var.

Bootstrapping / Bagging

Since all sampled data contains noise, we can reduce the variance by averaging multiple independent, unbiased estimators of X to get a high confidence model.

Bootstrapping is a test, analysis or metric that relies on random sampling with replacement.

Bagging is combining multiple classfiers each trained on randomly generated data sets.

Ex. Random Forest

This model takes the majority predictor of many overfit decision trees each trained

from a random subset of the training data and a random set of features.

The averaging effect reduces the total variance and noise from each individual tree.

Hyperparameters

Models have parameters and hyperparameters.

Parameters: selected based on loss function

Hyperparameters: selected based on performance on validation set.

To find hyperparams:

1. Randomly partion data into train/validation

2. For all possible hyperparam values, train model using training set.

3 Evaluate each model using validation set and pick the hyperparameters that have lowest validation error

4 Retrain model with all data (train + validation) with chosen hyper parameters.

- Some tips for finding hyperparameters:
  1. Perform a log-scale search to get an approximate range
  2. Do not wait for guesses that make the training times too long
  3. For Decision Trees:
     1. The number of classes is a good starting point for the minimum number of data in a leaf node (overfitting vs underfitting trade-off). This can be done by setting a threshold for minimum entropy
     2. Tree depth is likely <20; data points in a leaf is related to binary tree depth by $N/2^k$
     3. If splitting the data on a feature did not reduce entropy, retry the split with a different feature
  4. For Random Forests:
     1. When building random forests, make sure each model gets enough features and data points: approximately 90-100% of the features and 50-60% of the training data
     2. Number of trees should be related to how much each tree is overfitting