Homogenous Parallel Ensemble

08 December 2024 13:20

Ensemble learning :

one highly sophisticated model w/ insane accuracy ✗ — high bias (low dof) / high variance (high dof)

↓ combine

multiple models ✓

(easy to build + not computationally expensive + lower predictive power)

mostly correct → combined learner has low bias .

high confidence → most models predict same class .

↓

Weak learners — accuracy slightly better than chance — overfitting is good .

× n = strong learner .

Constructing multiple diverse prediction models .

Key idea = learners have to be $\boxed{\text{independent}}$ + $\boxed{\text{different.}}$

independent ↓ Errors should be independent & random in nature

↓

Combined opinion should be better than individual opinion .

different ↓ Main aspect : inexpensive build .

from there choose diff

a) Algos

b) Hyperparams

c) Subsets of training data .

d) Features .

How do you combine?

a) Mean

b) Mode

c) Weights

combined model must have low bias & high variance .

Errors come from :

a) Noise

b) Bias

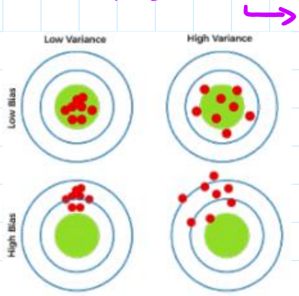
c) Variance .

→ Ensemble learning ensures outcome has low variance

$Z_1, Z_2 \dots Z_n$ independent observations

→ Each have variance σ^2

↓



Averaging a set of observations reduces variance .

So

Low variance = reliable results .

low bias = valid results .

But why exactly does the confidence ↑ ?

Scene : N models combined

prob. that output is correct = A . (accuracy)

prob. of misclassification = 1 − A .

All weak learners have same accuracy A .

Math : P(correctly classified) = A

P (misclassifies) = 1 − A

P (all N models misclassify) = $(1-A)^N$ → v.v. low

P (at least one of N predicts correctly) = $1-(1-A)^N$

So how can we go about all this ?

① Manipulate data DISTRIBUTION

- bagging
- boosting

② Manipulate INPUT

- random forests .

② Manipulate INPUT
- random forests.

③ Manipulate CLASS LABELS
- error correcting o/p coding.

K. We start.

Bagging

- Bootstrap AGGregatING (Special case of model averaging) — Combine o/p of multiple classifiers trained on diff samples.
- Why?
  → decrease variance → make unstable classifiers robust.
  → avoid overfitting
- When?
  → decision tree methods.
- How many?
  → 100 seems good enough.
  → More trees → improvements ↑
- How to implement:
  ① Multiple subsets created from original dataset w/ equal no. of tuples (select obs. w/ replacement)

  ② Base model created on each subset.

  ③ Each model learns independently and in parallel.

  ④ Final predictions by combining predictions of all models.

This means we get about 67% of data for training + 37% isn't selected.

P(record x not getting selected) = $\left(1-\frac{1}{n}\right)^n$ → approx. of e.

we draw from sample n times.

So two independent sets are created:
a. bootstrap sample — in the bag — sampling w/ replacement.
b. out-of-bag set — not chosen in sampling process.

→ Good test for model performance.

↓ How?

a. Find all models not trained by OOB instance
b. Take majority vote + compare w/ true label.
c. Compile OOB error for all OOB instances.

Random Forests

Namita Achyuthan

Random Forests

- Combines the o/p of multiple decision trees
  (ensemble of many decision trees)

- How the algo works:
  → random seed — pull out random collection of samples
    └ maintains class distribution
  → random set of attr. from OG dataset is chosen.
  → ALL i/p variables not considered.
  → M = no. of i/p attr.
    R = no. of attr. chosen at random } R < M
  → This gives best possible split to develop decision tree model.

- Evaluating the algo:
  → create bootstrapped dataset. → 1/3 not used → OOB MSE.
  → create dt. using random subset of features
  → get OOB error
  → run samples thro' RF + classify based on voting

- Advantages:
  → no need for pruning
  → accuracy + variable importance — automatic.
  → overfitting = no prob
  → doesn't care about outliers.
  → easy to set parameters.
  → good performance.

- Okay but what if missing attribute?
  → fill in missing attr. using normal imputation methods.
  → make a proximity matrix — HELP.

Extra Trees

- EXTremely RAndomized trees
- Increases randomness in tree construction

- But WHY?
  → High randomness (random split points)
  → Super fast (avoids search for optimal split)
  → Robustness (more stability)
  → Less overfitting

- How does it work?
  → Sample data (random sample)
  → Random feature selection
  → Random split points.
  → Aggregate predictions

Namita Achyuthan

→ Random split points.
→ Aggregate predictions

- Difference between extra trees and random forest:

Extra Trees	Random Forest
Random splits for each feature	Optimal split for selected features
More randomness	Less randomness
Lower computational cost	Higher computational cost
May have higher bias	Generally lower bias

- Advantages
  → Efficient training
  → Reduced variance
  → No bootstrapping

- Disadvantages
  → higher bias
  → less interpretability