Stanford University

CS 329P : Practical Machine Learning (2021 Fall)

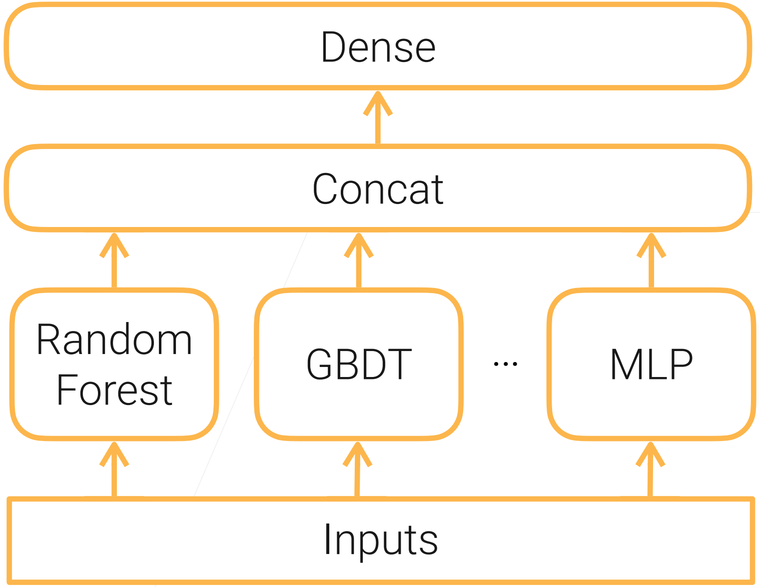
5.4 Stacking

Qingqing Huang, Mu Li, Alex Smola

https://c.d2l.ai/stanford-cs329p

Stacking

- Combine multiple base learners to reduce variance
  - Base learners can be different model types
  - Linearly combine base learners outputs by learned parameters
- Widely used in competitions
- bagging VS boosting
  - Bagging: bootstrap samples to get diversity
  - Boosting: different types of models extract different features



Stacking Results

- Evaluate on house sales data, compare to bagging and GBDT we implemented before

	Test Error
GBDT	0.259
RandomForest	0.243
Stacking (AutoGluon)	**0.229**

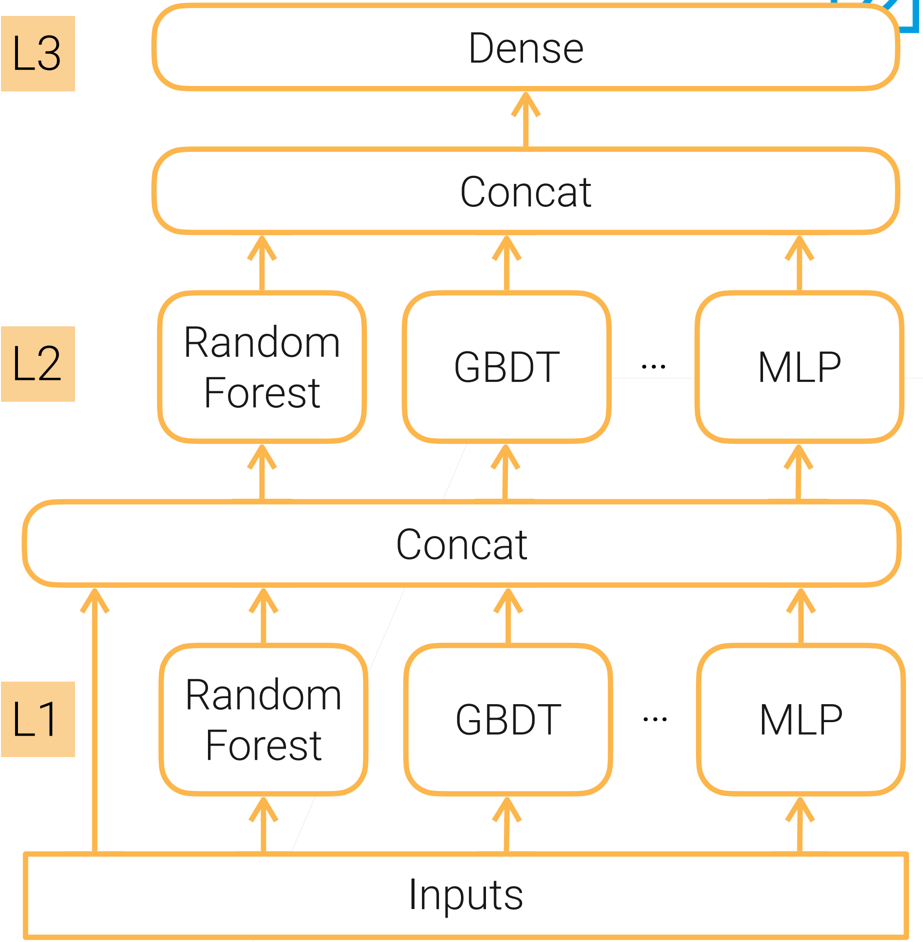
	model	score_test	score_val	pred_time_test
0	WeightedEnsemble_L2	-0.229626	-0.222406	3.953961
1	ExtraTrees	-0.232468	-0.232027	1.831407
2	CatBoost	-0.238690	-0.230338	0.016279
3	LightGBM	-0.239441	-0.226900	0.203499
4	NeuralNetMXNet	-0.241847	-0.246979	4.397190
5	RandomForest	-0.242904	-0.234931	1.750319
6	XGBoost	-0.249837	-0.240684	0.086715
7	KNeighbors	-0.457131	-0.443591	0.107212

```
from autogluon.tabular import TabularPredictor

predictor = TabularPredictor(label=label).fit(train)
```

Multi-layer Stacking

- Stacking base learners in multiple levels to reduce bias
  - Can use a different set of base learners at each level
- Upper levels (e.g. L2) are trained on the outputs of the level below (e.g. L1)
  - Concatenating original inputs helps



Overfitting in Multi-layer Stacking

- Train leaners from different levels on different data to alleviate overfitting
  - Split training data into *A* and *B*, train L1 learners on *A*, run inference on *B* to generate training data for L2 learners
- Repeated k-fold bagging:
  - Train k models as in k-fold cross validation
  - Combine predictions of each model on out-of-fold data
  - Repeat step 1,2 by n times, average the n predictions of each example for the next level training

Multi-layer Stacking Results

- Use 1 additional staked level, with 5-fold repeated bagging
  - Error: 0.229 → 0.227
  - Training time: 39 sec → 207 sec (5x)

```
from autogluon.tabular import TabularPredictor

predictor = TabularPredictor(label=label).fit(
    train, num_stack_levels=1, num_bag_folds=5)
```

	model	score_test	score_val
0	NeuralNetMXNet_BAG_L2	-0.225332	-0.219718
1	WeightedEnsemble_L3	-0.226921	-0.216254
2	CatBoost_BAG_L2	-0.227525	-0.217471
3	WeightedEnsemble_L2	-0.228386	-0.218298
4	LightGBM_BAG_L2	-0.228400	-0.218374
5	XGBoost_BAG_L2	-0.228660	-0.218824
6	ExtraTrees_BAG_L2	-0.228751	-0.217563
7	ExtraTrees_BAG_L1	-0.233527	-0.224974
8	RandomForest_BAG_L2	-0.234270	-0.220346
9	CatBoost_BAG_L1	-0.237356	-0.227126
10	LightGBM_BAG_L1	-0.238102	-0.225848
11	NeuralNetMXNet_BAG_L1	-0.238413	-0.238786
12	XGBoost_BAG_L1	-0.241698	-0.235570
13	RandomForest_BAG_L1	-0.242029	-0.227800
14	KNeighbors_BAG_L1	-0.457909	-0.447980

Model Combination Summary

- The goal is to reduce bias and variance

Reduce	Bias	Variance	Computation Cost	Parallelization
Bagging	Y		*n*	*n*
Boosting		Y	*n*	*1*
Stacking	Y		*n*	*n*
K-fold multi-level stacking		Y	*nxlxk*	*nxk*

n: number of learners, *l*: number of levels, *k*: k-fold