Ensmeble Training

Xiaoyong Pan, Peter R. Rijnbeek,

April 20, 2018

Contents

1 Introduction

In general, different machine learning models will perform differently on the same dataset, and they can complement each other. Thus, different models can be combined to improve prediction performance. In the `PatientLevelPrediction` package, we implemented different ensemble strategies for combining different models.

2 Background

Supervised learning algorithms try to find a good hypothesis space well-suited for a particular problem, but it is very diffcult to find it. Ensmeble models combine different hypothesis to form a better but not optimum one. How to combine different models is a important research direction.

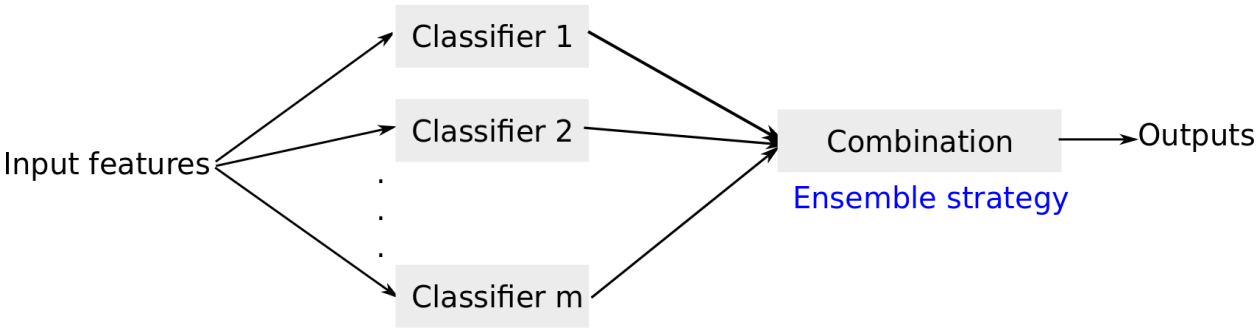

Figure 1: The ensemble training

Ensemble models train multiple models, Figure 1 illustrates the ensmenle training. Ensemble models yield better results for multiple models with signficant diversity, and also ensemble strategies can paly a curical role in final prediction performance. Currently, many ensemble strategies have been proposed. In PatientLevelPrediction package, four ensmeble strategies have been implemented, they are average ensembble, ensmeble with product rule, weighted ensemble and stakced ensemble. 1. average ensembble: Calculate the average probability from differnt models 2. product ensemble: Claculate the product of probabilites from different models. 3. weighted ensmeble: Calculate the weighted average probability from different models using train AUC as weights. 4. stacked ensemble: Train a logistics regression on outputs from different models, and then the trained logitstics regression is used to combine different models.

3 Usage

Use the OHDSI tool ecosystem to generate a `population` and `plpData` object. Alternatively, you can make use of the data simulator. The following code snippet creates a population of 12000 patients.

```
set.seed(1234)
data(plpDataSimulationProfile)
sampleSize <- 12000
plpData <- simulatePlpData(
  plpDataSimulationProfile,
  n = sampleSize
)
```

```
population <- createStudyPopulation(
  plpData,
  outcomeId = 2,
  binary = TRUE,
  firstExposureOnly = FALSE,
  washoutPeriod = 0,
  removeSubjectsWithPriorOutcome = FALSE,
  priorOutcomeLookback = 99999,
  requireTimeAtRisk = FALSE,
  minTimeAtRisk = 0,
  riskWindowStart = 0,
  addExposureDaysToStart = FALSE,
  riskWindowEnd = 365,
  addExposureDaysToEnd = FALSE,
  verbosity = futile.logger::INFO
)
```

Specify the prediction algorithms to be combined.

```
# Use LASSO logistic regression and Random Forest as base predictors
model1 <- setLassoLogisticRegression()
model2 <- setRandomForest()
```

Specify a test fraction and a sequence of training set fractions.

```
testFraction <- 0.2
```

Specify an ensembleStrategy to combine multiple predictors. The strategy used for ensembling the outputs from different models, it can be 'mean', 'product', 'weighted' and 'stacked': 'mean' the average probability from differnt models 'product' the product rule 'weighted' the weighted average probability from different models using train AUC as weights. 'stacked' the stakced ensemble trains a logistics regression on different models.

```
ensembleStrategy <- 'stacked'
```

Specify the test split to be used.

```
# Use a split by person, alterantively a time split is possible
testSplit <- 'person'
```

Run the ensemble learning to combine model1 and model2. You can also use different plpData for different models.

```
results <- PatientLevelPrediction::runEnsembleModel(population, dataList = list(plpData, plpData),
                                                    modelList = list(model1, model2),
                                                    testSplit=testSplit,
                                                    testFraction=testFraction,
                                                    nfold=3, splitSeed=1000, ensembleStrategy = ensembl
```

4 Apply Ensemble model

```
plpData <- loadPlpData("plpdata/")
modelList <- loadEnsemblePlpModel("/data/home/xpan/git/PatientLevelPrediction/plpmodels/20180607153811")
use the same population settings as the model:
populationSettings <- plpModel$populationSettings
```

```
populationSettings$plpData <- plpData
population <- do.call(createStudyPopulation, populationSettings)
```

Get the prediction, please make sure the ensemble strategy for training and applying is the same:

```
prediction <- applyEnsembleModel(population,
                                 dataList = list(plpData, plpData),
                                 modelList = modelList,
                                 analysisId = NULL,
                                 save = NULL,
                                 ensembleStrategy = "stacked")$prediction
```