Homework 2

36-708, Spring 2022

Due March 18 at 5PM EST

Please attach all code to your homework. In an RMarkdown document for example, this can be done in one line (see Yihui Xie's website for how to do this).

1 Taking a closer look at stacking weights

In this question, we consider a model ensemble by weighting the outputs of different models and obtain the optimal set of weights for minimizing the squared risk. Consider the following setup.

- Data model: $(X, Y) \sim P_{X,Y}$ where $P_{X,Y}$ is a distribution on $\mathbb{R}^d \times \mathbb{R}$ with the regression function denoted by $f^\star(x) := \mathbb{E}(Y \mid X = x)$.
- Individual learners: $f_1, f_2, \ldots, f_T$ each mapping $\mathbb{R}^d \to \mathbb{R}$.
- Stacked learner: $f = \sum_{i=1}^T w_i f_i$.

First, we obtain a certain conditions on weights to keep the final prediction of the weighted model within certain reasonable limits. Consider the following requirement on the final prediction.

- Constrained output: Suppose we require that at any input $x \in \mathbb{R}^d$, the final prediction of the weighted learner be within the maximum and minimum of the predictions of all the individual learner predictions, i.e., $f_{\min}(x) \le f(x) \le f_{\max}(x)$, where $f_{\min}(x) = \min\{f_1(x), \ldots, f_T(x)\}$ and $f_{\max}(x) = \max\{f_1(x), \ldots, f_T(x)\}$.

(a) Show that

$$f_{\min}(x) \le f(x) \le f_{\max}(x) \text{ for any } f_1, \ldots, f_T \iff w_i \ge 0 \text{ for each } i \text{ and } \sum_{i=1}^T w_i = 1.$$

1.1 Optimizing the weights subject to the constraints

Next, subject to the simplified sum constraint on the weights (that they need to add to 1), we find the best possible set of weights to minimize the squared risk as defined below.

- Metric: squared risk of f defined as $\mathbb{E}[(f(X) - f^\star(X))^2]$

(a) Show that the squared risk of f can be written as $\sum_{i=1}^T \sum_{j=1}^T w_i C_{ij} w_j$, where $C_{ij} = \mathbb{E}[(f_i(X) - f^\star(X))(f_j(X) - f^\star(X))]$ are certain model correlations.

(b) Define the matrix $\mathbf{C} \in \mathbb{R}^{T \times T}$ by $\mathbf{C} = (C_{ij})$. Assume $\mathbf{C}$ is invertible. Show that subject to the constraints on weights that $\sum_{i=1}^T w_i = 1$, the optimum weights to minimize the squared risk are given by:

$$w_i = \frac{\sum_{j=1}^T \mathbf{C}_{ij}^{-1}}{\sum_{k=1}^T \sum_{j=1}^T \mathbf{C}_{kj}^{-1}}$$

Hint: Use the method of Lagrange multipliers.

(c) Interpret the optimum weights. In particular, comment on the cases when the weights would be more symmetric and when the weights would be more asymmetric. How do the weights simplify if the model correlations, C_{ij} for $i \neq j$, are all 0? Interpret the simplified weights.

2 Stacking versus best model in action

In the previous question, we obtained an optimal set of weights for combining models subject to certain reasonable constraints on the weights. But, in practice, we do no have access to the model correlations as defined in the previous question. Thus, we have to somehow learn the weights as well. Since both the models and the model weights are to be learned from the same data, we need to penalize the complexities of individual models somehow. One way to do is by considering the cross-validated prediction for different models. We then find an optimal set of weights to minimize the empirical squared error subject to certain weight constraints.

In this question, we consider such (stacked) model combination with individual models and see if stacking adds any benefit. Consider the following setup.

- Dataset $\{(x_i, y_i)\}_{i=1}^n$
- Individual learners: $f_1, \ldots, f_T : \mathbb{R}^d \to R$
- Stacked learner $f = \sum_{i=1}^T w_i f_i$
- Stacking weights $w_i, i = 1, \ldots, T$
- Weight constraints $w_i \geq 0, \sum_{i=1}^T w_i \leq 1$
- Stacking objective:

$$\frac{1}{V} \sum_{v=1}^{V} \sum_{j=1}^{n_v} \left(y_j^v - f^{-v}(x_j^v) \right)^2$$

  Here, $V = 10$ is the number of cross-validation folds, (x_j^v, y_j^v) is the j^{th} datapoint in fold v, and f^{-v} is the stacked learner trained on all of the data except for those in fold v, $(x_j^v, y_j^v)_{j=1}^{n_v}$.

2.1 Single best model

(a) Show that if we restrict the weights $w_i \in \{0, 1\}$ in addition to the above set of constraints, then the best combined learned is simply the individual learner with the smallest cross-validation error.

2.2 Stacked versus single best model

Next, we compare the single best model with best combined model for the following specific choices of dataset and individual learners.

- Dataset: Ames Housing Dataset with the following columns:
  - "Bldg_Type",
  - "Lot_Area",
  - "House_Style",
  - "Overall_Qual",
  - "Kitchen_Qual",
  - "Heating_QC",
  - "Sale_Condition",
  - "Year_Remod_Add",

  - "First_Flr_SF",
  - "TotRms_AbvGrd",
  - "Sale_Price"

  Your goal will be to predict `Sale_Price`. This dataset can be found in the R package, `AmesHousing`. Alternatively, the raw data can be found at

  http://jse.amstat.org/v19n3/decock/AmesHousing.txt with the documentation in

  http://jse.amstat.org/v19n3/decock/DataDocumentation.txt

- Train and test split: use 3:1 random split
- Individual learners (4 total):
  1. k-nearest neighbors with $k \in \{1, 6\}$.
  2. Linear regression.
  3. Random forests (you can use the default arguments provided in the software package you use). If using R, the `ranger` package is recommended for computational reasons.

(a) Train individual regressors and obtain cross-validated predictions to be used for stacking.

(b) Find the single best regressor among the regressors and its test error.

(c) Find the best stacking weights with the above set of weight constraints and construct the corresponding stacked regressor and obtain its test error. Compare it with the test error of the single best regressor.

(d) Repeat (a)–(c) *without* random forests. What do you notice about the stacked-versus-best MSE in both cases?

3 Deeper dive into random forests

In this problem, we will use random forests for classification on a labeled spam database, https://archive.ics.uci.edu/ml/datasets/Spambase.

(a) Fit random-forest classifiers on the spam data. Try various suitable values of m, the number of predictor variables selected at random as candidates for splitting when growing the trees.

(b) Plot the train, out-of-bag, and test error of random forests with various values of m that you tried as a function of number of trees.

(c) Summarize your findings. In particular, comment on the following. How fast does the training error decrease? Does the training error reach zero? Does the testing error increase beyond a certain point?

4 Classifier calibration

In this problem, we will calibrate classifiers including the random forest classifier from the previous problem. We will use the same spam data as in the previous problem.

Consider three separate classifiers: random forests with $\sqrt{d}$ variables for splitting at a node, logistic regression, and the constant classifier which outputs 1 if the majority of labels are 1s, and 0 otherwise. Create a 70-30 (train+calibration)-test split. We will use the 'test' set for reliability diagrams.

(a) Using 500 observations from the first split for calibration and the rest for model training, calibrate these classifiers using 10 bins and create a plot with a bin's target probabilities on the x-axis and empirical probabilities (from the test set) on the y-axis. These are also known as 'reliability diagrams'.

(b) Comment on the above plots. In particular, do the classifiers seem calibrated? How does the sharpness compare among these three?