Motivations
○○○

Reviews: Mixture of models and EM algorithm
○○○○
○○○○○

Expectation-Boosting algorithm
○○○○○○○○
○○○○○○○○○○○○○○○○

Applications
○○○○○
○○○○○○○

Conclusions
○○

Gradient tree-boosted mixture models and their applications in insurance loss prediction

GAO, Guangyuan[1]

Center for Applied Statistics and School of Statistics, Renmin University of China

Department of Mathematics at SUSTech, November 2021
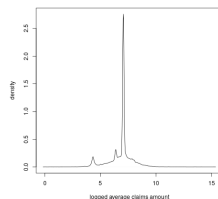

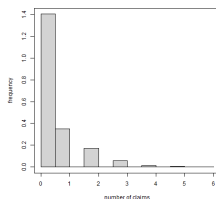
[1]Joint work with Li, Jiahong (Peking University)

Motivations
000

Reviews: Mixture of models and EM algorithm
0000
00000

Expectation-Boosting algorithm
00000000
0000000000000000

Applications
00000
0000000

Conclusions
00

Table of Contents

Insurance loss

Insurance loss data sometimes cannot be well modeled by a single
distribution.

- For claims counts data, there may be an excess of zero claims, so
  a Poisson distribution is not the best choice.
- For claims amount data, there may be a heavy tail, so a gamma
  distribution is not enough to describe the entire data.

One solution is to apply mixture of distributions.

Motivations
○●○

Reviews: Mixture of models and EM algorithm
○○○○
○○○○○

Expectation-Boosting algorithm
○○○○○○○○
○○○○○○○○○○○○○○○

Applications
○○○○○
○○○○○○○

Conclusions
○○

Insurance loss

- For claims counts data, we can utilize zero-modified Poisson distribution.
- For claims amount data, we can utilize mixture of gamma distribution and heavy-tailed distribution such as Pareto distribution.

When the individual risk factors are available, we can extend mixture of distributions to mixture of regressions to address risk heterogeneity in the portfolio.

Challenges with mixture models

- **Parameter estimation** in mixture models is challenging: each component distribution/regression parameters are related to each other.
- **Variable selection** in mixture models is also challenging: we need to perform variable selection for each component regression.

Motivations
○○○

Reviews: Mixture of models and EM algorithm
●○○○
○○○○○

Expectation-Boosting algorithm
○○○○○○○○
○○○○○○○○○○○○○○○

Applications
○○○○○
○○○○○○○

Conclusions
○○

Mixture of models

Mixture of distributions

Suppose a random variable y follows the k-th component distribution from

$$\{f_1(y; \mu_1, \phi_1), \ldots, f_K(y; \mu_K, \phi_K)\}$$

with mixing probability p_k, $k = 1, \ldots, K$, then the probability density function for y is

$$f(y) = \sum_{k=1}^{K} p_k f_k(y; \mu_k, \phi_k).$$

Motivations
○○○

Reviews: Mixture of models and EM algorithm
○●○○
○○○○○

Expectation-Boosting algorithm
○○○○○○○○○
○○○○○○○○○○○○○○○○○

Applications
○○○○○
○○○○○○○

Conclusions
○○

Mixture of models

Mixture of regressions

If individual features $\boldsymbol{x}_i$ are available and they have systematic effects on the distribution of y_i, then we establish a mixture of regressions:

$$f(y|\boldsymbol{x}) = \sum_{k=1}^{K} p_k(\boldsymbol{x}) f_k(y; \mu_k(\boldsymbol{x}), \phi_k(\boldsymbol{x})).$$

Above is the most general form of mixing. In applications, we always put some constraints on the mixing structure.

Motivations
000

Reviews: Mixture of models and EM algorithm
0000
00000

Expectation-Boosting algorithm
00000000
0000000000000000

Applications
00000
0000000

Conclusions
00

Mixture of models

Mixing structures

- Mixing probabilities are not related to x

$$f(y) = \sum_{k=1}^{K} p_k f_k(y; \mu_k(\boldsymbol{x}), \phi_k(\boldsymbol{x})).$$

- Both mixing probabilities and dispersions are not related to x

$$f(y) = \sum_{k=1}^{K} p_k f_k(y; \mu_k(\boldsymbol{x}), \phi_k),$$

Motivations   Reviews: Mixture of models and EM algorithm   Expectation-Boosting algorithm   Applications   Conclusions
○○○   ○○○●   ○○○○○○○○   ○○○○○   ○○
                     ○○○○○   ○○○○○○○○○○○○○○○○○   ○○○○○○○

Mixture of models

Mixing structures

- If component distributions are from the same distribution family, we might assume different component distributions have the same dispersion

$$f(y) = \sum_{k=1}^{K} p_k f_k(y; \mu_k(\boldsymbol{x}), \phi).$$

- Covariates $\boldsymbol{x}$ are only related to the mixing probabilities:

$$f(y) = \sum_{k=1}^{K} p_k(\boldsymbol{x}) f_k(y; \mu_k, \phi_k).$$

We need to determine the mixture structure according to the data and our aim. Imposing suitable constraints on mixture, we can accelerate model fitting without compromising predictive performance.

Motivations
○○○

Reviews: Mixture of models and EM algorithm
○○○○
●○○○○

Expectation-Boosting algorithm
○○○○○○○○
○○○○○○○○○○○○○○○

Applications
○○○○○
○○○○○○○

Conclusions
○○

EM algorithm

Likelihood function given full information

Suppose we know which component distribution each sample is from. That is we know the full information $(Y, Z, \boldsymbol{x})$ where

$$Z = (Z_1, \ldots, Z_K)^\top = (\mathbb{1}_1(k), \ldots, \mathbb{1}_K(k))^\top$$

is the one-hot encoding of component indicator variable.
The joint distribution function for full information (one sample) is given by

$$f(y, z|\boldsymbol{x}) = \prod_{k=1}^{K} \left[p_k(\boldsymbol{x}) f_k(y; \mu_k(\boldsymbol{x}), \phi_k(\boldsymbol{x})) \right]^{z_k}$$

The log-likelihood function is given by

$$l(p, \mu, \phi|y, z, \boldsymbol{x}) = \sum_{k=1}^{K} z_k \left[\log p_k(\boldsymbol{x}) + \log f_k(y; \mu_k(\boldsymbol{x}), \phi_k(\boldsymbol{x})) \right] \quad (1)$$

Motivations
○○○

Reviews: Mixture of models and EM algorithm
○○○○
○●○○○

Expectation-Boosting algorithm
○○○○○○○○
○○○○○○○○○○○○○○

Applications
○○○○○
○○○○○○○

Conclusions
○○

EM algorithm

Likelihood function given full information

The parameters in p, μ, ϕ can be estimated by $K + 1$ independent optimizations

$$\hat{\theta}_p = \arg\max_{\theta_p} \sum_{i=1}^{n} \sum_{k=1}^{K} z_{i,k} \log p_k(\boldsymbol{x}_i; \theta_p) \qquad (2)$$

$$\left(\hat{\theta}_{\mu}^{(k)}, \hat{\theta}_{\phi}^{(k)}\right) = \arg\max_{\theta_{\mu}^{(k)}, \theta_{\phi}^{(k)}} \sum_{i=1}^{n} \sum_{k=1}^{K} z_{i,k} \log f_k \left(y_i; \mu_k\left(\boldsymbol{x}_i; \theta_{\mu}^{(k)}\right), \phi_k\left(\boldsymbol{x}_i; \theta_{\phi}^{(k)}\right)\right)$$

$$(3)$$

Motivations
○○○

Reviews: Mixture of models and EM algorithm
○○○○
○○●○○

Expectation-Boosting algorithm
○○○○○○○○○
○○○○○○○○○○○○○○○○

Applications
○○○○○
○○○○○○○

Conclusions
○○

EM algorithm

Likelihood function given full information

Those optimizations are corresponding to a multinomial logistic classification and K regressions.

The multinomial logistic classification are fitted to all samples, while K regressions are fitted to partial samples with $\{i : z_{i,k} = 1\}$.

Motivations
○○○

Reviews: Mixture of models and EM algorithm
○○○○
○○○●○

Expectation-Boosting algorithm
○○○○○○○○
○○○○○○○○○○○○○○○

Applications
○○○○○
○○○○○○○

Conclusions
○○

EM algorithm

Expectation step

With iterated $\hat{p}, \hat{\mu}, \hat{\phi}$, calculate the conditional expectation of z:

$$\hat{z}_{i,k} = \hat{z}_k(\boldsymbol{x}_i) = \frac{\hat{p}_{i,k} f_k(y_i; \hat{\mu}_{i,k}, \hat{\phi}_{i,k})}{\sum_{l=1}^{K} \hat{p}_{i,l} f_l(y_i; \hat{\mu}_{i,l}, \hat{\phi}_{i,l})},$$

where $\hat{p}_{i,k} = \hat{p}_k(\boldsymbol{x}_i), \hat{\mu}_{i,k} = \hat{\mu}_k(\boldsymbol{x}_i), \hat{\phi}_{i,k} = \hat{\phi}_k(\boldsymbol{x}_i)$.

Motivations   Reviews: Mixture of models and EM algorithm   Expectation-Boosting algorithm   Applications   Conclusions
○○○   ○○○○   ○○○○○○○○   ○○○○○   ○○
                   ○○○○●   ○○○○○○○○○○○○○○○○○   ○○○○○○○

EM algorithm

Maximization step

Based on the following likelihood function for full information, calculate the MLE of parameters.

$$
\begin{aligned}
& l(p, \mu, \phi | y, \hat{z}, \boldsymbol{x}) \\
&= \sum_{i=1}^{n} \sum_{k=1}^{K} \hat{z}_{i,k} \left[\log p_k(\boldsymbol{x}_i) + \log f_k(y_i; \mu_k(\boldsymbol{x}_i), \phi_k(\boldsymbol{x}_i)) \right] \\
&= \sum_{i=1}^{n} \sum_{k=1}^{K} \hat{z}_{i,k} \log p_k(\boldsymbol{x}_i) + \sum_{i=1}^{n} \sum_{k=1}^{K} \hat{z}_{i,k} \log f_k(y_i; \mu_k(\boldsymbol{x}_i), \phi_k(\boldsymbol{x}_i))
\end{aligned}
\tag{4}
$$

This step is similar to (2) and (3). However, now we have $\hat{z}_{i,k} \in (0, 1)$, so we need to consider

- A multinomial logistic classification with fractional response.
- K weighted-regressions fitted to all samples.

Generic functional gradient descent algorithm

Our proposal

We replace the maximization step in the EM algorithm by a boosting step.

The boosting step increases the likelihood, but overfitting-sensitively.

The boosting step follows a generic functional gradient descent algorithm.

Motivations
○○○

Reviews: Mixture of models and EM algorithm
○○○○
○○○○○

Expectation-Boosting algorithm
○●○○○○○○
○○○○○○○○○○○○○○○○

Applications
○○○○○
○○○○○○○

Conclusions
○○

Generic functional gradient descent algorithm

General setting

Suppose non-parametric regression function as $F : \mathbb{R}^P \to \mathbb{R}$. Our purpose is to estimate F to minimize the expected loss

$$\hat{F} = \arg\min_F \mathbb{E}\left[C(Y, F(\boldsymbol{x}))\right],$$

where $C : \mathbb{R} \times \mathbb{R} \to \mathbb{R}_+$ is the loss function.

We always replace the expected loss by sample average loss:

$$\hat{F} = \arg\min_F \frac{1}{n} \sum_{i=1}^{n} C(y_i, F(\boldsymbol{x}_i))$$

Motivations   Reviews: Mixture of models and EM algorithm   **Expectation-Boosting algorithm**   Applications   Conclusions
○○○   ○○○○   ○○●○○○○○   ○○○○○   ○○
             ○○○○○   ○○○○○○○○○○○○○○○○   ○○○○○○○

Generic functional gradient descent algorithm

Link functions and loss functions

Commonly used link function and loss functions:

- AdaBoost:
$$C(y, F) = \exp(yF), \quad y \in \{-1, 1\}$$
$$F(\boldsymbol{x}) = \frac{1}{2} \log \left(\frac{\Pr[Y = 1|\boldsymbol{x}]}{\Pr[Y = -1|\boldsymbol{x}]} \right)$$

- LogitBoost:
$$C(y, F) = \log_2(1 + \exp(-2yF)), \quad y \in \{-1, 1\}$$
$$F(\boldsymbol{x}) = \frac{1}{2} \log \left(\frac{\Pr[Y = 1|\boldsymbol{x}]}{\Pr[Y = -1|\boldsymbol{x}]} \right)$$

- L_2Boost:
$$C(y, F) = (y - F)^2/2, \quad y \in \mathbb{R}$$
$$F(\boldsymbol{x}) = \mathbb{E}(Y|\boldsymbol{x})$$

Generic functional gradient descent algorithm

Loss function for insurance data

We choose the negative log-likelihood function as the loss function. Hence, minimizing the loss function is equivalent to maximizing the likelihood function.

Types of boosting algorithms

Commonly used boosting algorithms:

1 Binary classification: AdaBoost, LogitBoost (real, discrete, gentle AdaBoost), AdaBoost.M1

2 Multinomial classification: Stagewise Additive Modeling using a Multi-class Exponential loss (SAMME), SAMME.R (multi-class real AdaBoost),

3 Gradient based: gradient boosting machine/model (GBM), Newton boosting, gradient boosting decision tree (GBDT), eXtreme Gradient Boosting (XGBoost), light gradient boosting machine (LightGBM)

The last type is based on gradient of loss function, and our EB algorithm uses this version of boosting.

Generic functional gradient descent algorithm

Generic functional gradient descent algorithm

1. Initialization: $\hat{F}^{[0]}(\boldsymbol{x}; \hat{\theta}^{[0]})$, where $\hat{\theta}^{[0]}$ are determined by $(y_i, \boldsymbol{x}_i)_{i=1:n}$. Let $m = 0$.

2. Projection of gradient to weak learner: Calculate negative gradient

$$u_i = -\left.\frac{\partial C(y_i, F)}{\partial F}\right|_{F=\hat{F}^{[m]}(\boldsymbol{x}_i)}, i = 1, \ldots, n.$$

Data $(u_i, \boldsymbol{x}_i)_{i=1:n}$ is used to calibrate a weak learner $\hat{f}^{[m+1]}(\boldsymbol{x}; \hat{\theta}^{[m+1]})$ with loss function L_2.

Motivations   Reviews: Mixture of models and EM algorithm   Expectation-Boosting algorithm   Applications   Conclusions
○○○   ○○○○   ○○○○○○○●○   ○○○○○   ○○
              ○○○○○   ○○○○○○○○○○○○○○○○○   ○○○○○○○

Generic functional gradient descent algorithm

Generic functional gradient descent algorithm

3. One-dimensional optimization: Solve an one-dimensional optimization to find expansion coefficient $\hat{\omega}^{[m+1]}$:

$$\hat{\omega}^{[m+1]} = \arg\min_{\omega} \sum_{i=1}^{n} C(y_i, \hat{F}^{[m]}(\boldsymbol{x}_i) + \omega \hat{f}^{[m+1]}(\boldsymbol{x}_i))$$

Update

$$\hat{F}^{[m+1]} = \hat{F}^{[m]} + s\hat{\omega}^{[m+1]} \hat{f}^{[m+1]},$$

where s is shrinkage factor or learning rate.

4. Iteration: Let m increase by 1, and repeat steps 2-3.

Motivations
○○○

Reviews: Mixture of models and EM algorithm
○○○○
○○○○○

Expectation-Boosting algorithm
○○○○○○○●
○○○○○○○○○○○○○○○○○

Applications
○○○○○
○○○○○○○

Conclusions
○○

Generic functional gradient descent algorithm

Generic functional gradient descent algorithm

- Weak learners are fitted to negative gradient U rather than Y.
- Loss function in weak learners is always L_2, independently with model loss function C.
- If weak learners are trees, the algorithm is called gradient boosting decision tree (GBDT).
- If weak learners are trees, calibration and variable selection are performed simultaneously.
- Step 3 of one-dimensional optimization seems unnecessary given learning rate is sufficiently small according to some empirical experiments.

Preparation

- Notations: Regression functions for p, μ, ϕ are denoted by F, G, H.
- Key points: Link functions, cost/loss functions, negative gradient, .

Motivations   Reviews: Mixture of models and EM algorithm   Expectation-Boosting algorithm   Applications   Conclusions
000   0000   0000000000000000   00000   00
               00000   0000000

EB algorithm

Multiple logistic link function for mixing probabilities

$$p_k(\boldsymbol{x}) = \Pr(Z_k = 1|\boldsymbol{x}) = \frac{\exp F_k(\boldsymbol{x})}{\sum_{l=1}^{K} \exp F_l(\boldsymbol{x})}, \tag{5}$$

or equivalently

$$F_k(\boldsymbol{x}) = \log p_k(\boldsymbol{x}) - \frac{1}{K} \sum_{l=1}^{K} \log p_l(\boldsymbol{x}), \quad k = 1, \ldots, K. \tag{6}$$

Motivations   Reviews: Mixture of models and EM algorithm   Expectation-Boosting algorithm   Applications   Conclusions
000   0000   00000000   00000   00
              00000   00●00000000000000000   0000000

EB algorithm

Link functions in component models

- Component Gaussian model:

$$\mu_k(\boldsymbol{x}) = G_k(\boldsymbol{x})$$

$$\phi_k(\boldsymbol{x}) = \exp H_k(\boldsymbol{x})$$

- Component Poisson model:

$$\mu_k(\boldsymbol{x}) = \exp G_k(\boldsymbol{x})$$

- Component gamma model:

$$\mu_k(\boldsymbol{x}) = \exp G_k(\boldsymbol{x})$$

$$\phi_k(\boldsymbol{x}) = \exp H_k(\boldsymbol{x})$$

Motivations
○○○

Reviews: Mixture of models and EM algorithm
○○○○
○○○○○

Expectation-Boosting algorithm
○○○●○○○○○○○○○○○○

Applications
○○○○○
○○○○○○○

Conclusions
○○

EB algorithm

Negative gradient for mixing probabilities

We use the negative log-likelihood function as the cost function

$$C_Z(Z, F(\boldsymbol{x})) = -\sum_{k=1}^{K} Z_k \log p_k(\boldsymbol{x}). \tag{7}$$

The negative gradient of the cost function w.r.t. F_k is given by

$$U_k(Z, F(\boldsymbol{x})) = -\frac{\partial C_Z(Z, F(\boldsymbol{x}))}{\partial F_k(\boldsymbol{x})} = Z_k - p_k(\boldsymbol{x}), \quad k = 1, \ldots, K \tag{8}$$

EB algorithm

Negative gradient for component models

Similarly, we use negative log-likelihood function of each component model as its cost function:

$$C_k(Y, Z, G(\boldsymbol{x})) = -Z_k \log f_k(Y; \mu_k(G_k(\boldsymbol{x})), \phi_k), \ k = 1 : K. \quad (9)$$

The negative gradient of C_k w.r.t. G_k is denoted by $V_k(Y, Z, G(\boldsymbol{x}))$

Note that we have assumed that dispersion ϕ_k is fixed among samples (not related to $\boldsymbol{x}$).

Motivations   Reviews: Mixture of models and EM algorithm   Expectation-Boosting algorithm   Applications   Conclusions
000   0000   00000000   00000   00
               00000   00000●000000000000   0000000

EB algorithm

Overview of EB algorithm

Step 1: Initialization of EB algorithm $\hat{p}^{[0]}, \hat{\mu}^{[0]}, \hat{\phi}^{[0]}$. Set $t = 0$.

Step 2: Calculating conditional expectation of latent variable $\hat{z}^{[t]}$ given $\hat{p}^{[t]}, \hat{\mu}^{[t]}, \hat{\phi}^{[t]}$.

Step 3.1: Gradient boosting mixing probabilities $\hat{p}^{[t+1]}$ given latent variable $\hat{z}^{[t]}$.

Step 3.2: Gradient boosting component models $\hat{\mu}^{[t+1]}$ given latent variable $\hat{z}^{[t]}$.

Step 4: Calculate the MLE $\hat{\phi}^{[t+1]}$. Increase t by 1. Repeat steps 2-3 until t reaches to T.

Motivations   Reviews: Mixture of models and EM algorithm   Expectation-Boosting algorithm   Applications   Conclusions
000   0000   00000000   00000   00
              00000   000000●00000000000   0000000

EB algorithm

Step 1: Initialization of EB algorithm

1. Initialize $\hat{p}_1^{[0]}, \ldots, \hat{p}_K^{[0]}$ and $\hat{F}_1^{[0]}, \ldots, \hat{F}_K^{[0]}$:
   - $\hat{p}_k^{[0]} = \frac{1}{K}$
   - $\hat{F}_1^{[0]}, \ldots, \hat{F}_K^{[0]}$ are obtained by (6).

2. Initialize $\hat{\mu}_1^{[0]}, \ldots, \hat{\mu}_K^{[0]}$ and $\hat{G}_1^{[0]}, \ldots, \hat{G}_K^{[0]}$:
   - $\hat{\mu}_k^{[0]} = \frac{\sum_{i=1}^n Y_i}{n}$
   - $\hat{G}_k^{[0]} = \mu_k^{-1}(\hat{\mu}_k^{[0]})$

3. Initialize $\hat{\phi}_1^{[0]}, \ldots, \hat{\phi}_K^{[0]}$ as the sample variance. (Assume that dispersion is independent with covariates.)

4. Set $t = 0$.

Motivations Reviews: Mixture of models and EM algorithm Expectation-Boosting algorithm Applications Conclusions
000   0000   00000000   00000   00
             00000   0000000●0000000000   0000000

EB algorithm

Step 2: Conditional expectation of latent variable

Set $\hat{z}_{i,k}^{[t]}$ as

$$\hat{z}_{i,k}^{[t]} = \frac{\hat{p}_k^{[t]}(\boldsymbol{x}_i) f_k \left(y_i; \hat{\mu}_k^{[t]}(\boldsymbol{x}_i), \phi_k^{[t]} \right)}{\sum_{l=1}^{K} \hat{p}_l^{[t]}(\boldsymbol{x}_i) f_l \left(y_i; \hat{\mu}_l^{[t]}(\boldsymbol{x}_i), \phi_l^{[t]} \right)}, \quad k = 1 : K.$$

Motivations · · · · Reviews: Mixture of models and EM algorithm · · · · Expectation-Boosting algorithm · · · · Applications · · · · Conclusions

EB algorithm

Step 3.1: Gradient boosting mixing probabilities

1. Initialization. Set $\hat{p}_1^{[t,0]}, \ldots, \hat{p}_K^{[t,0]}$ and $\hat{F}_1^{[t,0]}, \ldots, \hat{F}_K^{[t,0]}$ as

$$\hat{p}_k^{[t,0]} = \frac{1}{K}, \quad \hat{F}_k(\boldsymbol{x})^{[t,0]} = \log \hat{p}_k^{[t,0]} - \frac{1}{K} \sum_{l=1}^{K} \log \hat{p}_l^{[t,0]}$$

Set $m = 0$.

2. Projection of gradient to learner. Compute the negative gradient sample $u_{1,k}^{[t,m]}, \ldots, u_{n,k}^{[t,m]}$, in which

$$u_{i,k}^{[t,m]} = U_k(\hat{z}_i^{[t]}, \hat{F}^{[t,m]}(\boldsymbol{x}_i)) = \hat{z}_{i,k}^{[t]} - \hat{p}_{i,k}^{[t,m]}.$$

Then the data $(u_{i,k}^{[t,m]}, \boldsymbol{x}_i)_{i=1:n}$ is used to calibrate a L-terminal node regression trees $\hat{f}_k^{[t,m+1]}\left(\boldsymbol{x}; R_{l=1:L}^{[t,m+1]}, \bar{u}_{l=1:L}^{[t,m+1]}\right)$ with L_2 loss, where $R_{l=1:L}^{[t,m+1]}$ is the partition of covariate space and $\bar{u}_{l=1:L}^{[t,m+1]}$ contains the average gradient in each terminal node. K regression trees are fitted independently.

Motivations   Reviews: Mixture of models and EM algorithm   Expectation-Boosting algorithm   Applications   Conclusions
000   0000   00000000   00000   00
                  00000   000000000●00000000   0000000

EB algorithm

Step 3.1: Gradient boosting mixing probabilities

3. The one-dimensional optimization for expansion coefficient leads
   to the following update (Friedman, 2001):

$$\hat{F}_k^{[t,m+1]}(\boldsymbol{x}_i) = \hat{F}_k^{[t,m]}(\boldsymbol{x}_i) + s \sum_{l=1}^{L} \gamma_l^{[t,m+1]} \mathbb{1}(\boldsymbol{x} \in R_l^{[t,m+1]}), \ k = 1, \ldots, K$$

where

$$\gamma_l^{[t,m+1]} = \frac{K-1}{K} \frac{\sum_{\boldsymbol{x}_i \in R_l^{[t,m+1]}} u_{i,k}^{[t,m]}}{\sum_{\boldsymbol{x}_i \in R_l^{[t,m+1]}} |u_{i,k}^{[t,m]}|(1 - |u_{i,k}^{[t,m]}|)}$$

We then derive the updated mixing probability $\hat{p}_{k=1:K}^{[t,m+1]}$ using (5).

Motivations	Reviews: Mixture of models and EM algorithm	Expectation-Boosting algorithm	Applications	Conclusions
○○○	○○○○	●●●●●●●●●○●○○○○○○	○○○○○	○○
	○○○○○		○○○○○○○	

EB algorithm

Step 3.1: Gradient boosting mixing probabilities

4. Increase *m* by 1, and repeat steps 2-3 until *m* reaches to the pre-determined M_p. Set

$$\hat{F}_k^{[t+1]} = \hat{F}_k^{[t,M_P]}, \quad \hat{p}_k^{[t+1]} = \hat{p}_k^{[t,M_P]},$$

Motivations   Reviews: Mixture of models and EM algorithm   **Expectation-Boosting algorithm**   Applications   Conclusions
000   0000   00000000   00000   00
              00000   0000000000000●000000   0000000

EB algorithm

Step 3.2: Gradient boosting component models

1 Initialization. Set $\hat{\mu}_1^{[t,0]}, \ldots, \hat{\mu}_K^{[t,0]}$ and $\hat{G}_1^{[t,0]}, \ldots, \hat{G}_K^{[t,0]}$:

$$\hat{\mu}_k^{[t,0]} = \frac{\sum_{i=1}^n Y_i}{n}, \hat{G}_k^{[t,0]} = \mu_k^{-1}(\hat{\mu}_k^{[t,0]})$$

Set $m = 0$.

2 Projection of gradient to learner. Compute the negative gradient samples $v_{1,k}^{[t,m]}, \ldots, v_{n,k}^{[t,m]}$, in which

$$v_{i,k}^{[t,m]} = V_k(y_i, \hat{z}_i^{[t]}, \hat{G}^{[t,m]}(\boldsymbol{x}_i)),$$

Then the data $(v_{i,k}^{[t,m]}, \boldsymbol{x}_i)_{i=1:n}$ is used to calibrate a L-terminal node regression trees $\hat{g}_k^{[t,m+1]}\left(\boldsymbol{x}; S_{l=1:L}^{[t,m+1]}, \bar{v}_{l=1:L}^{[t,m+1]}\right)$ with L_2 loss, where $S_{l=1:L}^{[t,m+1]}$ is the partition of covariate space and $\bar{v}_{l=1:L}^{[t,m+1]}$ contains the average gradient in each terminal node.
K regression trees are fitted independently.

Motivations
000

Reviews: Mixture of models and EM algorithm
0000
00000

Expectation-Boosting algorithm
00000000
000000000000●00000

Applications
00000
0000000

Conclusions
00

EB algorithm

Step 3.2: Gradient boosting component models

3. Conduct the following K independent one-dimensional optimizations to find the best expansion coefficients.

$$\hat{w}_k^{[t,m+1]} = \arg\min_w \sum_{i=1}^n C_k(y_i, \hat{z}_i^{[t]}, \hat{G}_k^{[t,m]}(\boldsymbol{x}_i) + w\hat{g}_k^{[t,m+1]}(\boldsymbol{x}_i)).$$

4. Compute the updates

$$\hat{G}_k^{[t,m+1]}(\boldsymbol{x}_i) = \hat{G}_k^{[t,m]}(\boldsymbol{x}_i) + s\hat{w}_k^{[t,m+1]}\hat{g}_k^{[t,m+1]}(\boldsymbol{x}_i),$$

$$\hat{\mu}_k^{[t,m+1]}(\boldsymbol{x}_i) = \mu_k(\hat{G}_k^{[t,m+1]}(\boldsymbol{x}_i)).$$

Motivations   Reviews: Mixture of models and EM algorithm   Expectation-Boosting algorithm   Applications   Conclusions
000   0000   0000000000000000000   00000   00
              00000   0000000   0000000

EB algorithm

Step 3.2: Gradient boosting component models

5. Increase m by 1, and repeat steps 2-4 until m reaches to the pre-determined M_μ.

$$\hat{G}_k^{[t+1]} = \hat{G}_k^{[t,M_\mu]}, \quad \hat{\mu}_k^{[t+1]} = \hat{\mu}_k^{[t,M_\mu]}.$$

Motivations   Reviews: Mixture of models and EM algorithm   Expectation-Boosting algorithm   Applications   Conclusions
○○○   ○○○○   ○○○○○○○○   ○○○○○   ○○
               ○○○○○   ○○○○○○○○○○○○○○●○○○   ○○○○○○○

EB algorithm

Step 4: EB iteration

1. For parameters not related to covariate x, compute the MLE $\hat{\phi}_k^{[t+1]}$ given all the other parameters $\hat{p}_k^{[t+1]}, \hat{\mu}_k^{[t+1]}$.

2. Increase t by 1, and repeat steps 2-3 until t reaches to the pre-determined T.

Motivations
ooo

Reviews: Mixture of models and EM algorithm
ooooo
ooooo

Expectation-Boosting algorithm
ooooooooo
oooooooooooooooo●oo

Applications
ooooo
ooooooo

Conclusions
oo

EB algorithm

Independent boosting v.s. forward boosting

- Independent boosting: In initialization of boosting, we initialize parameters independently with the previously boosting estimates $\hat{p}_k^{[t-1]}, \hat{\mu}_k^{[t-1]}$.

- Forward boosting: In contrast, we might initialize parameters as the previously boosting estimates $\hat{p}_k^{[t-1]}, \hat{\mu}_k^{[t-1]}$. This would lead a smaller required boosting iterations M (or a earlier stop on validation loss).

Motivations   Reviews: Mixture of models and EM algorithm   Expectation-Boosting algorithm   Applications   Conclusions
○○○   ○○○○   ○○○○○○○○○   ○○○○○   ○○
               ○○○○○   ○○○○○○○○○○○○○○○○○●○   ○○○○○○○

EB algorithm

Independent boosting v.s. forward boosting

Issues with forward boosting: difficult to predict for new data due to the iterative initialization.

An additional boosting with default initialization needs to be conducted given the last expected hidden variables $\hat{z}_i^{[T]}$.

Motivations
○○○

Reviews: Mixture of models and EM algorithm
○○○○
○○○○○

Expectation-Boosting algorithm
○○○○○○○○
○○○○○○○○○○○○○○○○●

Applications
○○○○○
○○○○○○○

Conclusions
○○

EB algorithm

Tuning parameters

- Number of EB iterations T. Determined by the trace plot of loss.
- Number of iterations in boosting M_p, M_μ. We can specify a sufficiently large M_p, M_μ and use early stop according to validation loss.
- Learning rate s. Smaller learning rate tends to lead to a better fitting and predictive performance but with more iterations.
- Tuning parameters in base learner tree: complexity parameter, maximum depth, number of terminal nodes.

Motivations   Reviews: Mixture of models and EM algorithm   Expectation-Boosting algorithm   **Applications**   Conclusions
000   0000   00000000   ●0000   00
               00000   0000000000000000000   0000000

A simulated example: mixture of Gaussians

Underlying model

$$Y_i = Z_i Y_{i,1} + (1 - Z_i) Y_{i,2}$$
$$Y_{i,1} | \boldsymbol{x}_i \sim N(\mu_1(\boldsymbol{x}_i), 0.9^2)$$
$$Y_{i,2} | \boldsymbol{x}_i \sim N(\mu_2(\boldsymbol{x}_i), 0.5^2)$$
$$Z_i \sim Bernuli(p(\boldsymbol{x}_i))$$
$$\boldsymbol{x}_i = (x_{i,1}, x_{i,2}, x_{i,3})^\top$$

A simulated example: mixture of Gaussians

Underlying model

$$\mu_1(\boldsymbol{x}_i) = 2 + 2 \times x_{i,1} + \log x_{i,2}$$

$$\mu_2(\boldsymbol{x}_i) = 0.5 - 0.5 \times x_{i,1} + x_{i,3} - x_{i,1} \times x_{i,2}$$

$$p(\boldsymbol{x}_i) = \frac{\exp \eta(\boldsymbol{x}_i)}{1 + \exp \eta(\boldsymbol{x}_i)}$$

$$\eta(\boldsymbol{x}_i) = 1 + 0.1 \times x_1 + \log x_2 + x_3 \times x_1$$

$$x_{1,i} \sim N(2, 1^2)$$

$$x_{i,2} \sim Exp(2)$$

$$x_{i,3} \sim Bernuli(0.5)$$

$$n = 5,000$$

Motivations   Reviews: Mixture of models and EM algorithm   Expectation-Boosting algorithm   Applications   Conclusions
○○○   ○○○○   ○○○○○○○○   ○○●○○   ○○
             ○○○○○   ○○○○○○○○○○○○○○○○○   ○○○○○○○

A simulated example: mixture of Gaussians

Data distribution
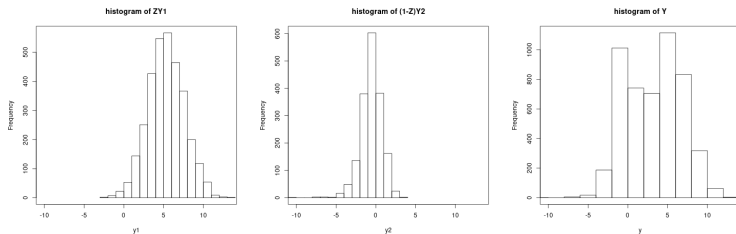
Figure 1: Distribution of $ZY, (1 - Z)Y, Y$.

Motivations   Reviews: Mixture of models and EM algorithm   Expectation-Boosting algorithm   **Applications**   Conclusions
000   0000   00000000   00000   00
               00000   0000000000000000000   0000000

A simulated example: mixture of Gaussians

Models fitted

Mixture of distributions (homo):

$$f(y; p, \mu, \sigma) = p_1 f_N(y; \mu_1, \sigma_1^2) + (1 - p_1) f_N(y; \mu_2, \sigma_2^2)$$

Mixture of linear regressions or mixture of boosting with constant mixing probabilities (glm_cp or boosting_cp)

$$f(y; p, \mu, \sigma) = p_1 f_N(y; \mu_1(\boldsymbol{x}), \sigma_1^2) + (1 - p_1) f_N(y; \mu_2(\boldsymbol{x}), \sigma_2^2)$$

Mixture of linear regressions or mixture of boosting with varying mixing probabilities (glm_vp or boosting_vp)

$$f(y; p, \mu, \sigma) = p_1(\boldsymbol{x}) f_N(y; \mu_1(\boldsymbol{x}), \sigma_1^2) + (1 - p_1(\boldsymbol{x})) f_N(y; \mu_2(\boldsymbol{x}), \sigma_2^2)$$

Motivations	Reviews: Mixture of models and EM algorithm	Expectation-Boosting algorithm	Applications	Conclusions
०००	०००० ०००००	००००००० ००००००००००००००००	०००० ००००००	००

A simulated example: mixture of Gaussians

Results and comparisons

$$e_{\mu 1} = \frac{\sum_{i=1}^{n}(\mu_{i,1} - \hat{\mu}_{i,1})^2}{n}$$

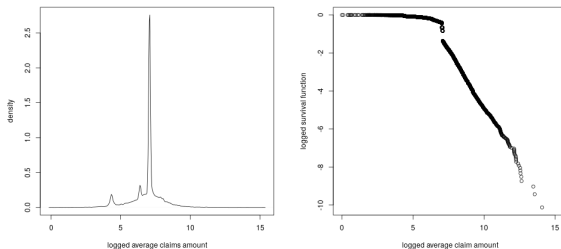$$e_{\mu 2} = \frac{\sum_{i=1}^{n}(\mu_{i,2} - \hat{\mu}_{i,2})^2}{n}$$

$$e_{\eta} = \frac{\sum_{i=1}^{n}(\eta_i - \hat{\eta}_i)^2}{n}, \eta_i = \log\frac{p_i}{1-p_i}$$

Table 1: Summary of test losses for different models.

model	neg LL	$e_{\mu 1}$	$e_{\mu 2}$	e_{η}
homo	2.5868	6.2657	2.6688	3.3304
glm_cp	1.8978	0.7775	0.3233	3.2440
glm_vp	1.7698	0.8028	0.3260	1.2667
boosting_cp	1.7492	0.2461	**0.1062**	3.2242
boosting_vp	**1.6407**	**0.2334**	0.1105	**0.8435**

Motivations
○○○

Reviews: Mixture of models and EM algorithm
○○○○
○○○○○

Expectation-Boosting algorithm
○○○○○○○○
○○○○○○○○○○○○○○○

Applications
○○○○○
●○○○○○○

Conclusions
○○

A real data example: claims severity modelling

Distribution of average claim amount

Claims amount data `freMTPL2sev` from R package **CASdatasets**.
Sample size $n = 24,938$. Three peaks and heavy tail.

Figure 2: Histogram and logged survival function of logged average claims.

Motivations
000

Reviews: Mixture of models and EM algorithm
0000
00000

Expectation-Boosting algorithm
00000000
00000000000000

Applications
00000
0●00000

Conclusions
00

A real data example: claims severity modelling

Mixture of distributions

Three gamma distributions for three peaks. One gamma distribution for resting non-tail part. One Pareto distribution for tail.

$$f(y) = \sum_{k=1}^{4} p_k f_{gamma}(y; \mu_k, \phi_k) + p_5 f_{pareto}(y; \alpha, M)$$

where μ, ϕ are mean and dispersion parameter, and α, M are tail index and threshold. The threshold is pre-determined as 8158.13 according to Hill plot.

Initialization of hidden variable

Figure 2 indicates a way to initialize the hidden variable:

$$\hat{z}_i^{[0]} = (\hat{z}_{i,1}^{[0]}, \hat{z}_{i,2}^{[0]}, \hat{z}_{i,3}^{[0]}, \hat{z}_{i,4}^{[0]}, \hat{z}_{i,5}^{[0]})^\top$$
$$= (\mathbb{1}_{(0,500]} y_i, \mathbb{1}_{(500,1000]} y_i, \mathbb{1}_{(1000,1200]} y_i, \mathbb{1}_{(1200,8158.13]} y_i, \mathbb{1}_{(8158.13,\infty)} y_i$$

(10)

Other parameters can be initialized as the MLE based on the full likelihood function.
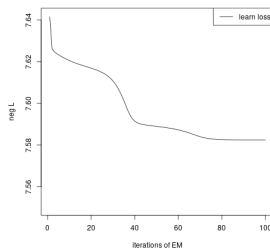
EM algorithm

Figure 3: Learning loss of mixture of distributions.

Estimated parameters

Table 2: MLE of four component gamma distributions. Tail index is estimated as $\hat{\alpha} = 1.0773$

component k	μ_k	shape ($1/\phi_k$)	scale	rate
1	76.8727	105.556	0.7283	1.3731
2	592.5909	653.539	0.9067	1.1029
3	1171.3811	999.9999	1.1714	0.8537
4	1534.5143	1.0377	1478.7768	7e-04

Those large shape parameters (small dispersion) implies the difficulties with gamma mean modeling.

Motivations Reviews: Mixture of models and EM algorithm Expectation-Boosting algorithm **Applications** Conclusions
000   0000   00000000   00000   00
             00000   0000000000000000000   0000000●0

A real data example: claims severity modelling

Boosting mixing probabilities

$$f(y|\boldsymbol{x}) = \sum_{k=1}^{4} p_k(\boldsymbol{x}) f_{gamma}(y; \mu_k, \phi_k) + p_5(\boldsymbol{x}) f_{pareto}(y; \alpha, M)$$
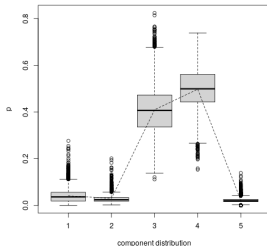
Figure 4: Boxplot of estimated mixing probabilities.

Motivations Reviews: Mixture of models and EM algorithm Expectation-Boosting algorithm **Applications** Conclusions
000   0000   00000000   00000   00
           00000   0000000000000000   0000000●

A real data example: claims severity modelling

Boosting the forth component mean

$$f(y|\boldsymbol{x}) = \sum_{k=1}^{3} p_k(\boldsymbol{x}) f_{gamma}(y; \mu_k, \phi_k)+$$

$$p_4(\boldsymbol{x}) f_{gamma}(y; \mu_4(\boldsymbol{x}), \phi_4) + p_5(\boldsymbol{x}) f_{pareto}(y; \alpha, M)$$

Test loss (negative log-likelihood)

1. Mixture of distributions: 7.5815
2. Boosting mixing probabilities: **7.5588**
3. Boosting mixing probabilities and the forth component mean: 7.5573.

Motivations
ooo

Reviews: Mixture of models and EM algorithm
oooo
ooooo

Expectation-Boosting algorithm
oooooooo
ooooooooooooooooo

Applications
ooooo
ooooooo

Conclusions
●o

Our proposal: Expectation-Boosting algorithm

Expectation-Boosting (EB) algorithm:

- Replaces the maximization step by an overfitting-sensitive boosting step.
- The boosting step follows a generic functional gradient descent algorithm.

Motivations
○○○

Reviews: Mixture of models and EM algorithm
○○○○
○○○○○

Expectation-Boosting algorithm
○○○○○○○○
○○○○○○○○○○○○○○○

Applications
○○○○○
○○○○○○○

Conclusions
○●

Advantages

Several advantages of EB algorithm over the EM algorithm.

- No need for specifying the form of component regression functions and performing covariate transformation.
- Only need for component loss functions.
- Boosting algorithm is a flexible non-parametric regression facilitating both non-linear effects and interaction.
- Boosting algorithm is overfitting-sensitive, we can perform variable selection simultaneously during the EB algorithm.

Thank you!
Q & A