Inspire…Educate…Transform.

Data Science: Foundations, Ensembles

Big Picture

Dr. K. V Dakshinamurthy

President, INSOFE

Ensembling strategies

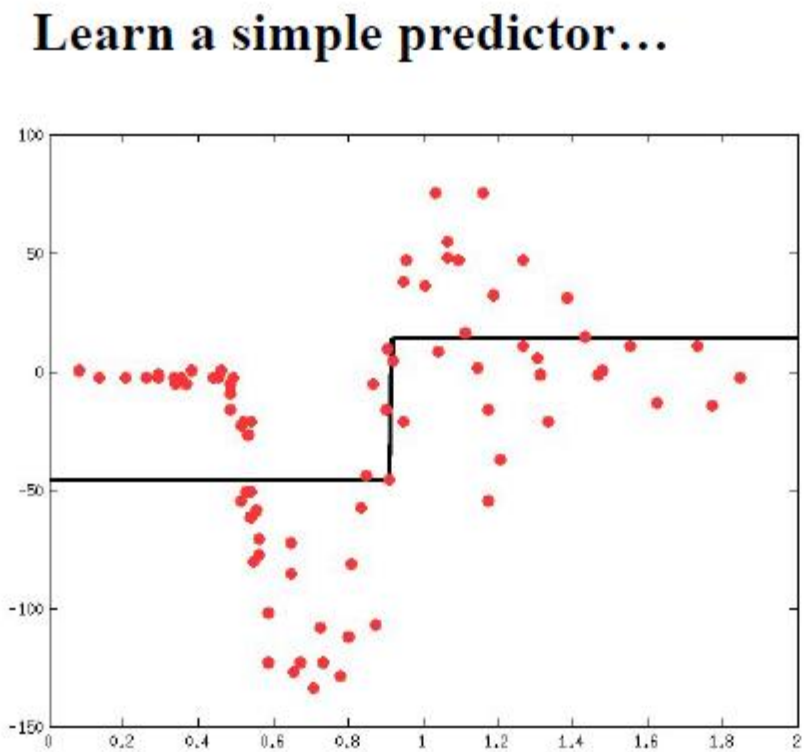
- With large data, there are two strategies
  - One mega model on entire data
  - Multiple models on small sets of data and combining the predictions
  - Later is mostly better

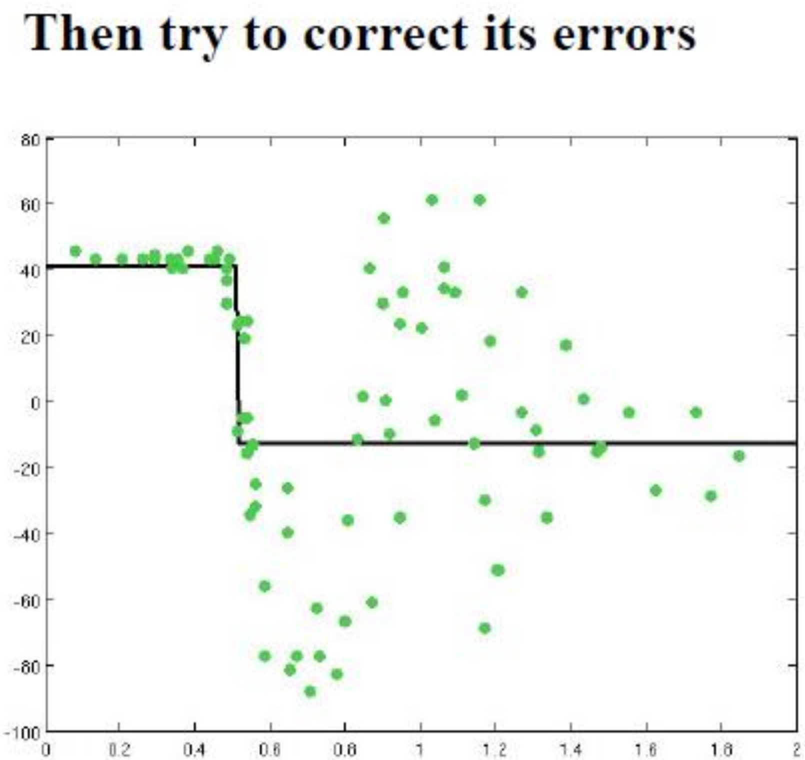
- How do we do multiple models? Again two strategies
  - Bagging: Randomly take subsets of data, build a base model on each. Let them vote for predictions (eg. Random forests)
  - Boosting: Make the data progressively tougher (second model is built on the records on which the first model failed and so on). Take combined weighted average for prediction.

Two most powerful algorithms for boosting

- Adaboost
  - Second model is built on the samples where first model failed
- Gradient boosting machines
  - Second model is built on the error made by first model

Error residual

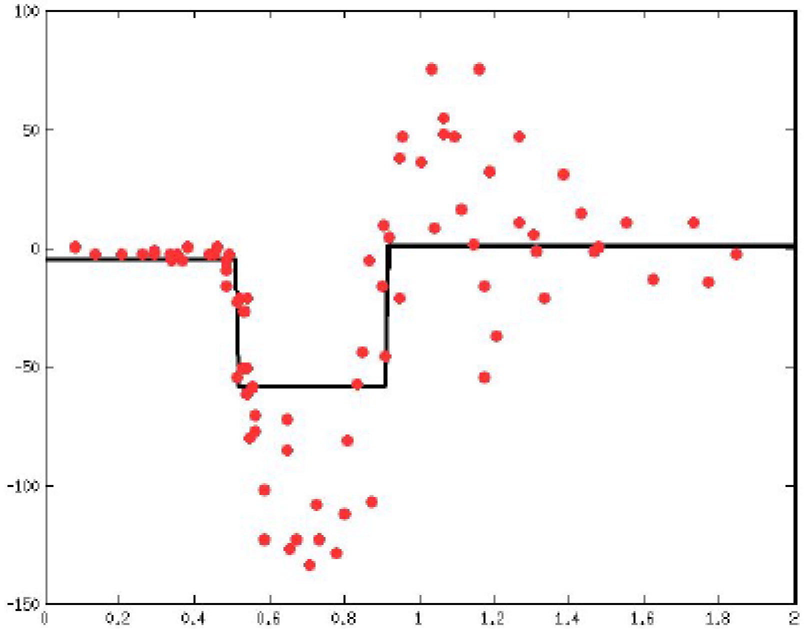




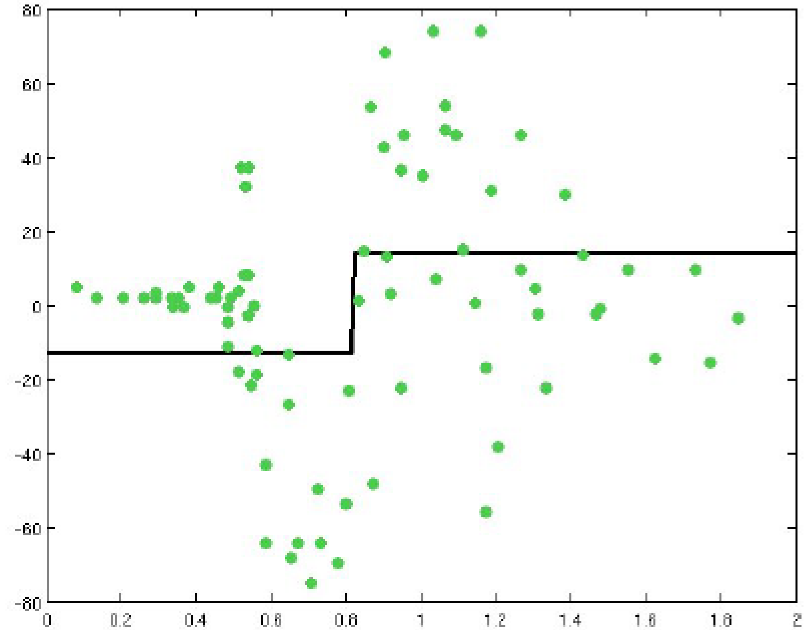
Excellent video: http://www.youtube.com/watch?v=sRktKszFmSk
Tutorial: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3885826/

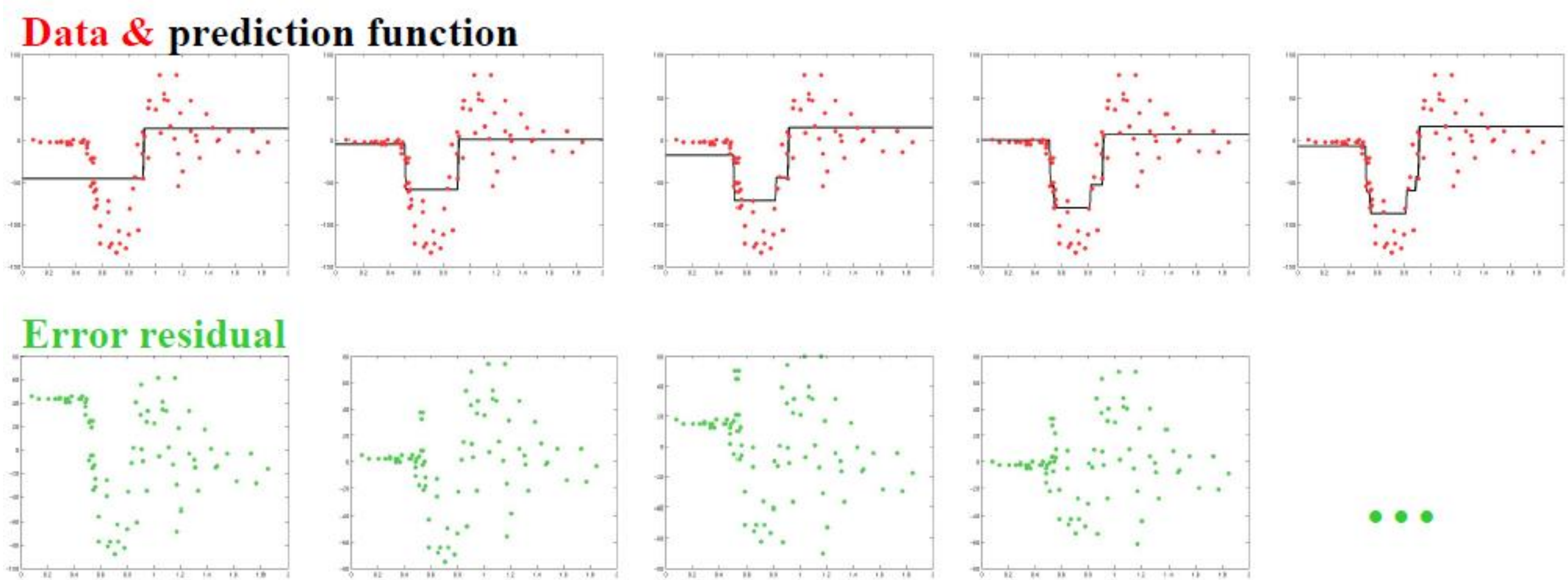
Model gets complex with each addition

Combining gives a better predictor…



Can try to correct its errors also, & repeat




Data & prediction function
Error residual

GBM choices

Choice of the loss-function $\Psi(y, f)$ (least squares, logistic etc.)
Choice of the base-learner model $h(x, \theta)$ (regression, trees etc.)

Algorithm: 1: initialize $\hat{f}0$ with a constant

for t = 1 to M **do**
compute the negative gradient $g_t(x)$
fit a new base-learner function $h(x, \theta_t)$
find the best gradient descent step-size ρ_t:

$$\widehat{f}_t \leftarrow \widehat{f}_{t-1} + \rho_t h(x, \theta_t)$$

Loss functions

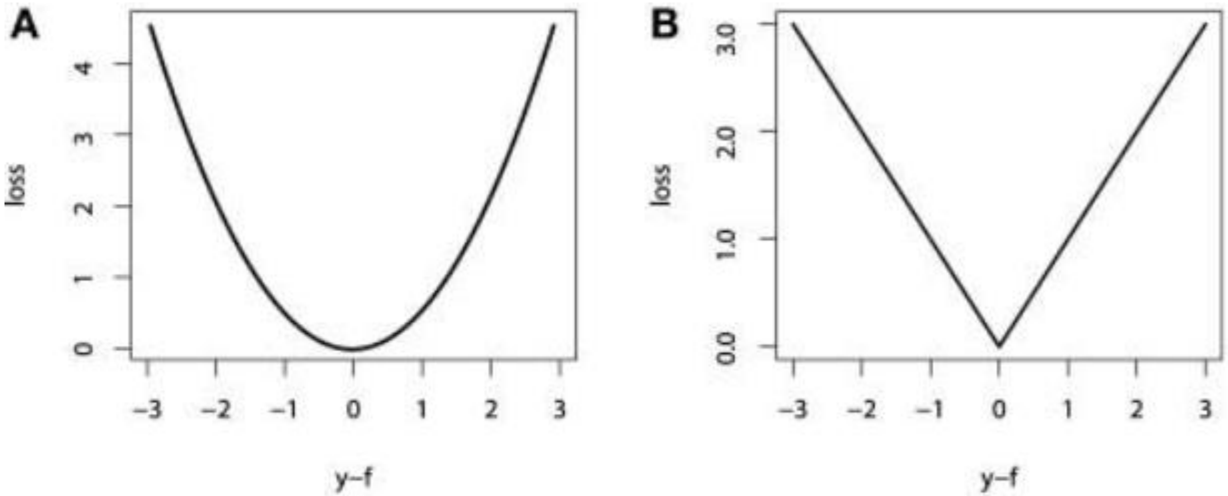
Continuous

$$\Psi(y,f)_{L_2} = \frac{1}{2}(y-f)^2$$

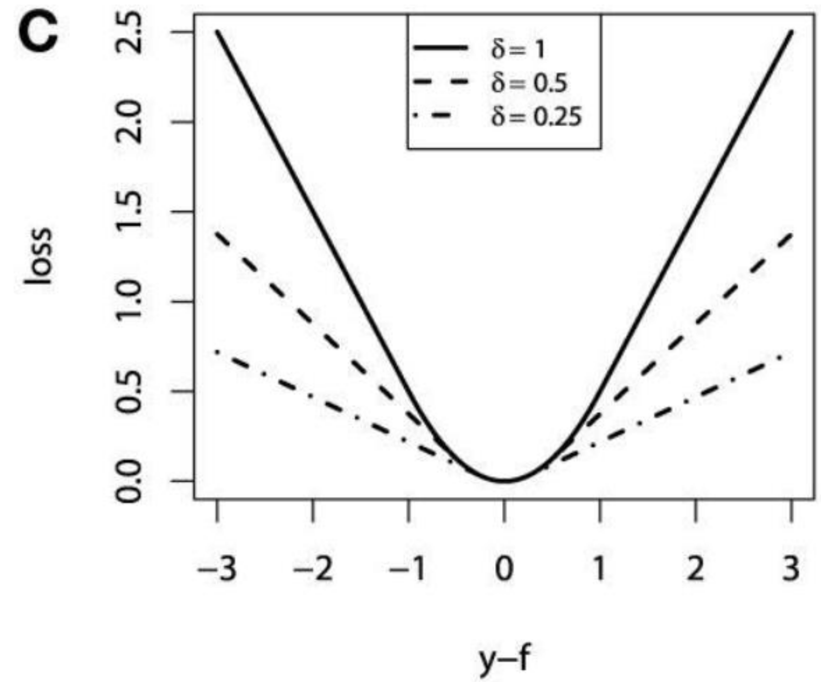
$$\Psi(y,f)_{L_1} = |y-f|$$

$$\Psi(y,f)_{\mathrm{Huber},\delta} = \begin{cases} \frac{1}{2}(y-f)^2 & |y-f| \le \delta \\ \delta(|y-f| - \delta/2) & |y-f| > \delta \end{cases}$$

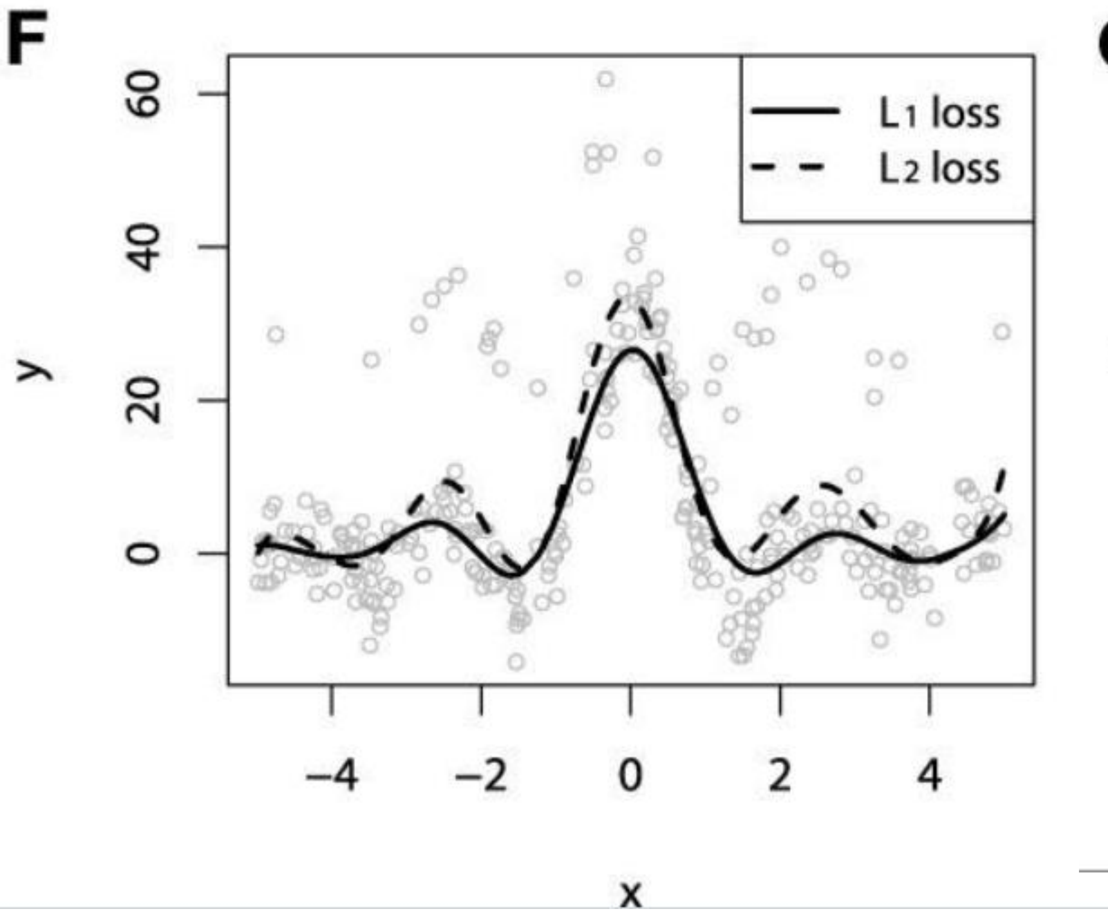
Loss functions

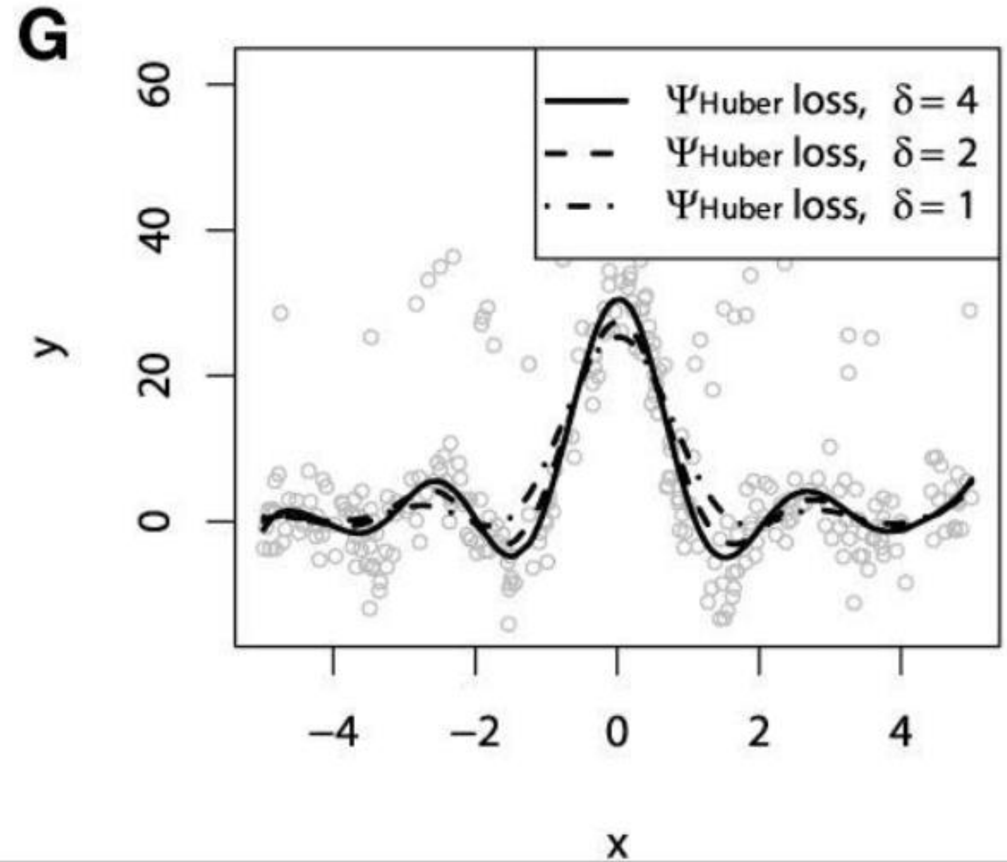


Continuous loss functions: (A) L_2 squared loss function; (B) L_1 absolute loss function;



Loss function is a hyper parameter





Categorical loss functions

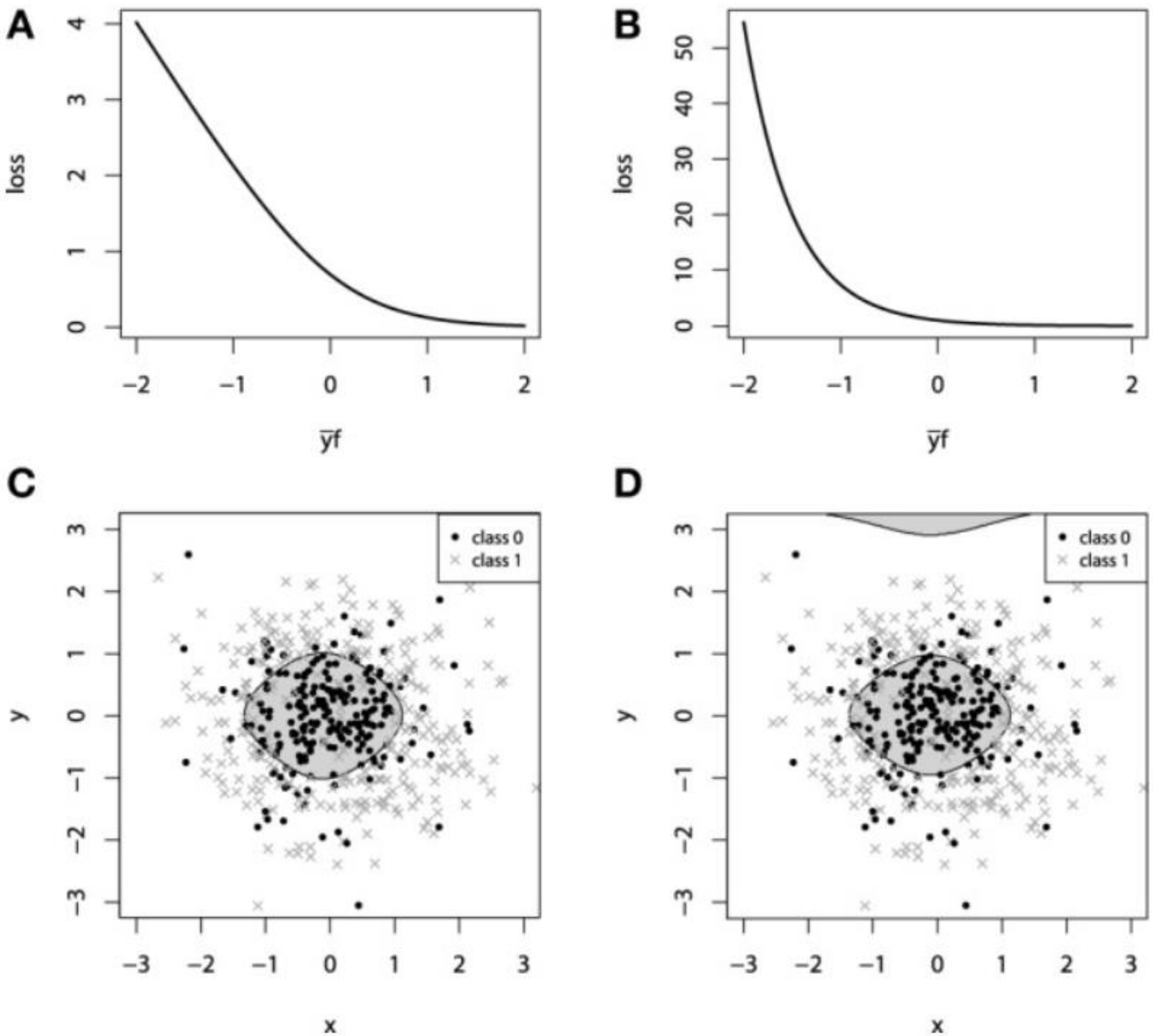
$$\Psi(y,f)_{\mathrm{Bern}} = \log(1+\exp(-2\bar{y}f))$$

Bernouli

$$\Psi(y,f)_{\mathrm{Ada}} = \exp(-\bar{y}f)$$

AdaBoost

Figure 2



(A) Bernoulli loss function. **(B)** Adaboost loss function. **(C)** GBM 2d classification with Bernoulli loss. **(D)** GBM 2d classification with Adaboost loss.

Base learners

- Linear models:
  - Ordinary linear regression
  - Ridge penalized linear regression
  - Random effects
- Smooth models:
  - P-splines
  - Radial basis functions
- Decision trees
  - Decision tree stumps
  - Decision trees with arbitrary interaction depth
- Other models:
  - Markov Random Fields
  - Wavelets
  - Custom base-learner functions

How to avoid over fitting

- Implicit measures (play with these parameters and pick the one that gives best accuracy on both train and validation sets)
  - Interaction depth
  - Sub sampling
  - Shrinkage or Learning rate
  - Early stopping

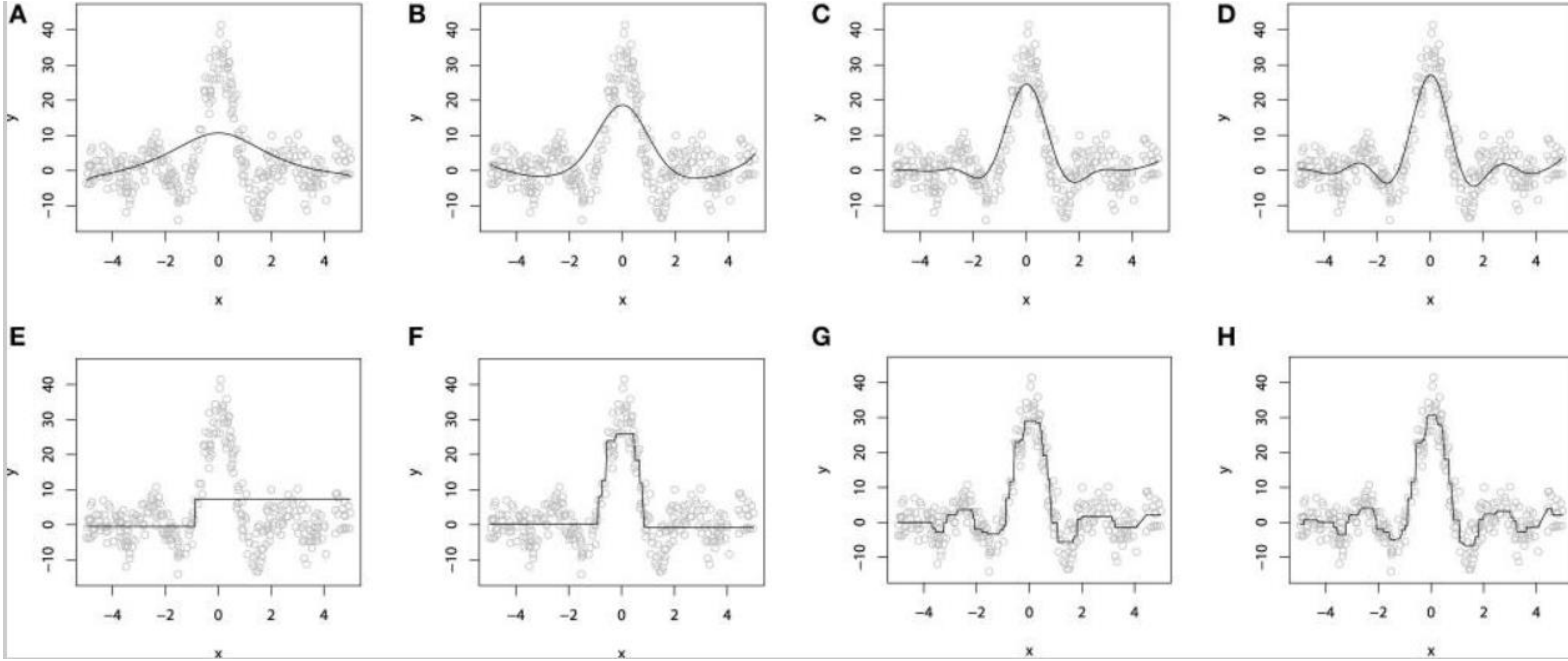
Interaction depth

- Regression
  - No interaction -> y=f(x1,x2,x3…).
  - Binary interactions; y=f(x1x2,x2x3,…)
  - Ternary interactions -> y=f(x1x2x3…)
- Decision trees
  - Interactions are naturally included as the depth of the branch
    - If x1 is between k1 and k2 & x2 is between l1 and l2 etc.
  - The depth of the tree is called interaction depth

Tree stumps (additive models)

- No interaction models are called additive models
  - A special case of a decision tree with only one split (i.e., a tree with two terminal nodes) is called a tree stump. Therefore, if one wants to fit an additive model with tree base-learners, it is possible to do this using the tree stumps.
- In many practical applications small trees (and tree-stumps), lower interaction depths provide considerably accurate results (Wenxin, 2002).

Small models can explain complexity



Sub sampling

- At each learning iteration only a random part of the training data is used to fit a consecutive base-learner.
- The training data is typically sampled without replacement, however, replacement sampling, just as it is done in bootstrapping, is yet another possible design choice.
- The subsampling procedure requires a parameter called the “bag fraction.” Bag fraction is a positive value not greater than one, which specifies the ratio of the data to be used at each iteration. For example, *bag* = 0.1 corresponds to sampling and using only 10% of the data at each iteration.
- Another useful property of the subsampling is that it naturally adapts the GBM learning procedures to large datasets when there is no reason to use all the potentially enormous amounts of data at once.

Smaller bags are better

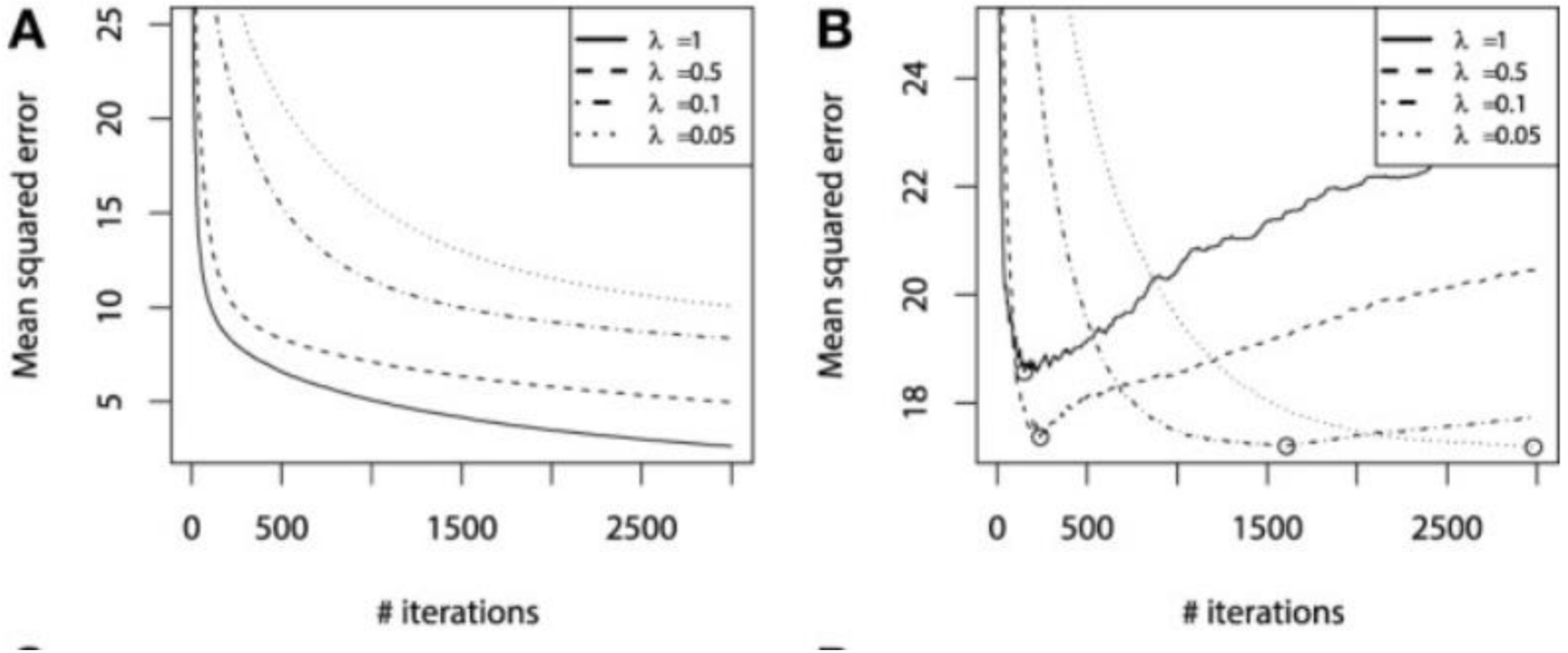
- GBM ensemble will reach the desired accuracy with a larger number of base-learners and lower bag than the one with smaller amount of more carefully fitted base-learners with larger bag.

Shrinkage or learning parameter

- In the context of GBMs, shrinkage is used for reducing, or shrinking, the impact of each additional fitted base-learner.
- It penalizes the importance of each consecutive iteration. The intuition behind this technique is that it is better to improve a model by taking many small steps than by taking fewer large steps.

$$\widehat{f}_t \leftarrow \widehat{f}_{t-1} + \lambda \rho_t h(x, \theta_t)$$

Early stopping and shrinkage



(A) training set error; (B) validation set error.

If the ensemble was trimmed by the number of trees, corresponding to the validation set minima on the error curve, the overfitting would be circumvented at the minimal accuracy expense. Another observation is that the optimal number of boosts, at which the early stopping is considered, varies with respect to the shrinkage parameter λ. Therefore, a trade-off between the number of boosts and λ should be considered.

Explicit regularization

http://xgboost.readthedocs.io/en/latest/model.html

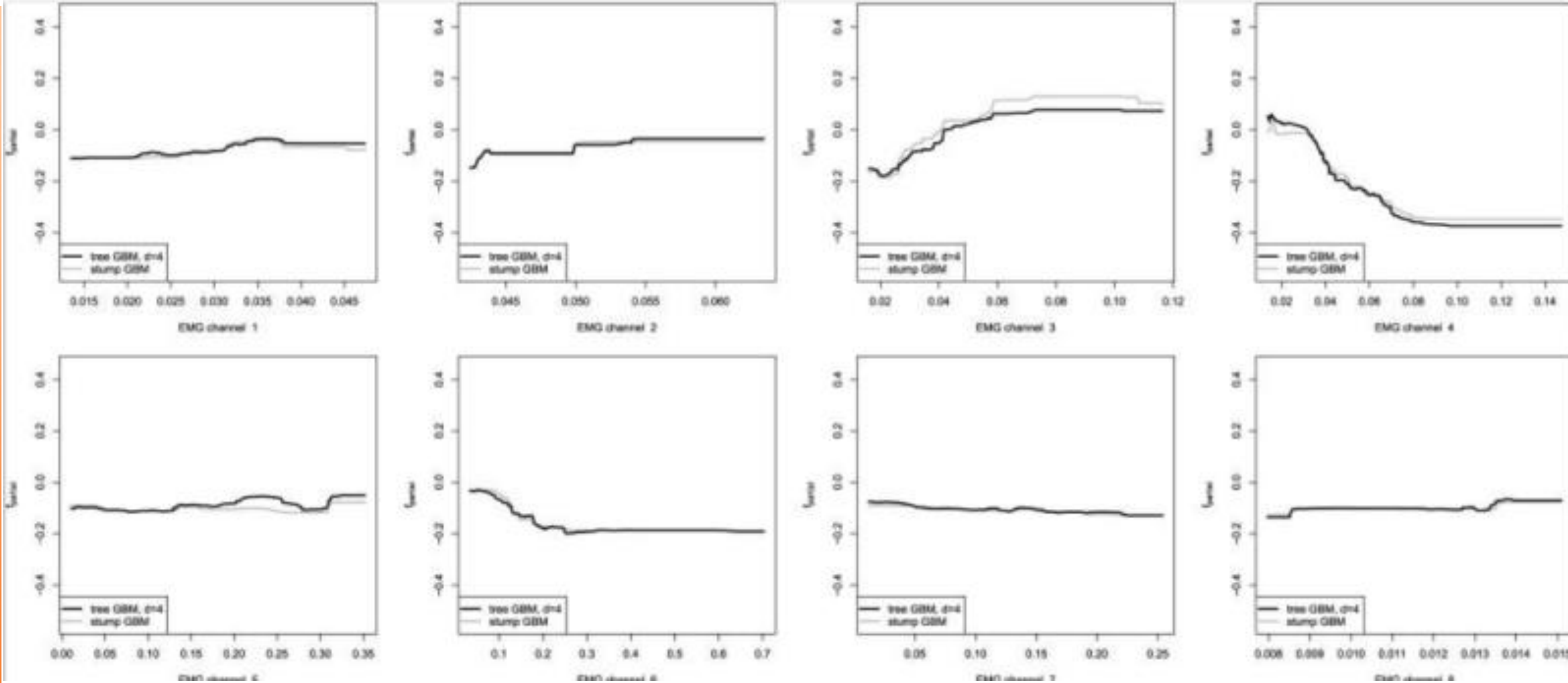
XGBoost defines complexity explicitly.

w is the vector of scores on leaves and T is the number of leaves.

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$$

It works excellent in practice

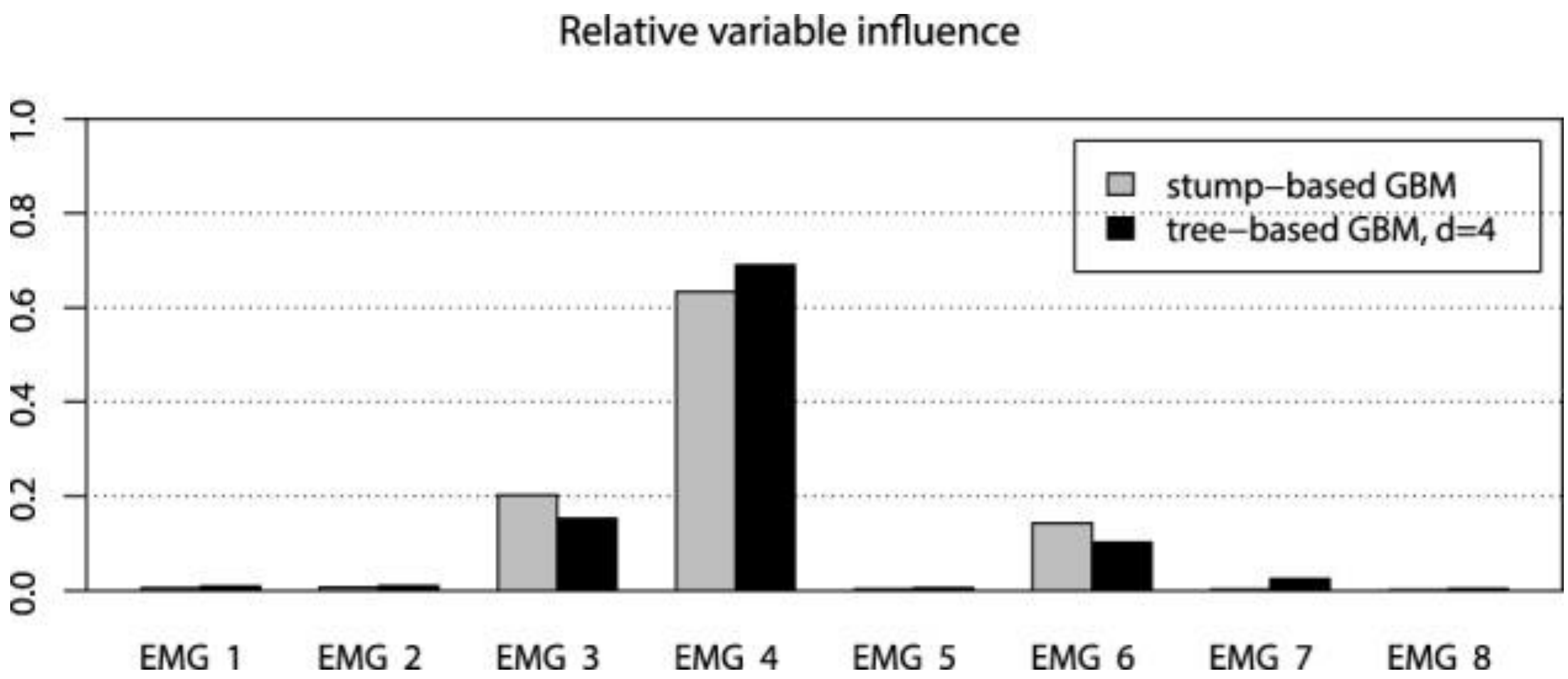
Partial dependency plots



Interpretability

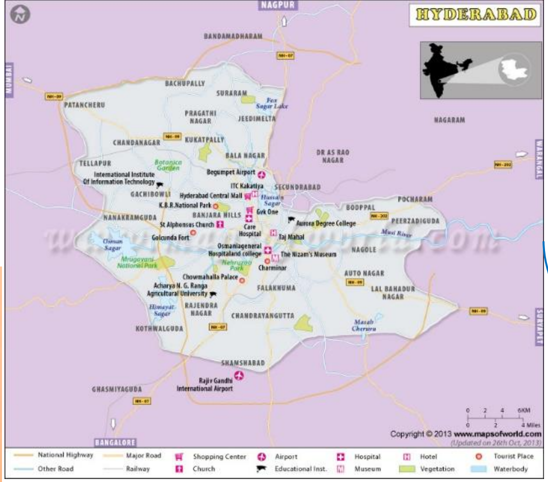
$$\text{Influence}_j(T) = \sum_{i=1}^{L-1} I_i^2 1(S_i = j)$$

This measure is based on the number of times a variable is selected for splitting, i.e., current splitting variable S is the same as the queried variable j. The measure also captures weights of the influence with the empirical squared improvement I , assigned to the model as a result of this split.

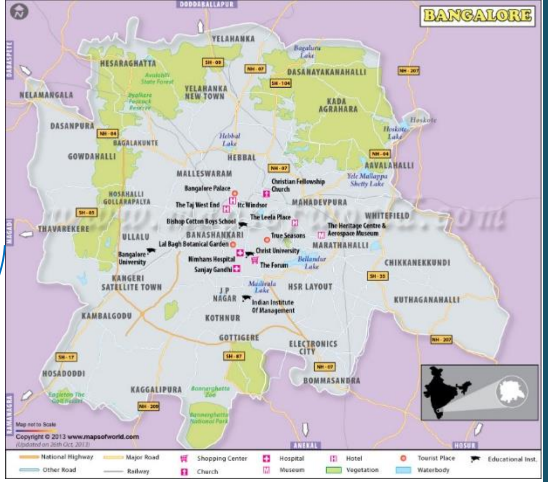

Relative variable influence
stump–based GBM
tree–based GBM, d=4
0.0
0.2
0.4
0.6
0.8
1.0
EMG 1
EMG 2
EMG 3
EMG 4
EMG 5
EMG 6
EMG 7
EMG 8

Good packages

- XGBBoost
  - Implemented in Python and Interface available in R.
  - Implements ridge regularization explicitly (http://xgboost.readthedocs.io/en/latest/model.html)
- GBM: A native R package



Inspire...Educate...Transform.



HYDERABAD

Office and Classrooms

Plot 63/A, Floors 1&2, Road # 13, Film Nagar,
Jubilee Hills, Hyderabad - 500 033
+91-9701685511 (Individuals)
+91-9618483483 (Corporates)

BENGALURU

Office

Incubex, #728, Grace Platina, 4th Floor, CMH Road,
Indira Nagar, 1st Stage, Bengaluru – 560038
+91-9502334561 (Individuals)
+91-9502799088 (Corporates)

Classroom

KnowledgeHut Solutions Pvt. Ltd., Reliable Plaza,
Jakkasandra Main Road, Teacher's Colony, 14th Main
Road, Sector – 5, HSR Layout, Bengaluru - 560102

Social Media

Web:	http://www.insofe.edu.in
Facebook:	https://www.facebook.com/insofe
Twitter:	https://twitter.com/Insofeedu
YouTube:	http://www.youtube.com/InsofeVideos
SlideShare:	http://www.slideshare.net/INSOFE
LinkedIn:	http://www.linkedin.com/company/international-school-of-engineering

This presentation may contain references to findings of various reports available in the public domain. INSOFE makes no representation as to their accuracy or that the organization subscribes to those findings.