Multi-Task Predictions: A Revised Ensemble Machine Learning Approach

Data Science Bootcamp
University of Saskatchewan
06/14/2023
Xuekui Zhang

Q/A

- There are many well developed prediction methods: linear regression, logistic regression, SVM, KNN, decision tree, elastic net, LDA, QDA, etc
  - How to quickly develop methods works better than those?
    - - Ensemble learning

- Most prediction methods predict one outcome at a time.
  - How to predict multiple outcomes? Non-Gaussian?
    - - MTPS and other multitask prediction methods

ENSEMBLE LEARNING

ENSEMBLE LEARNING

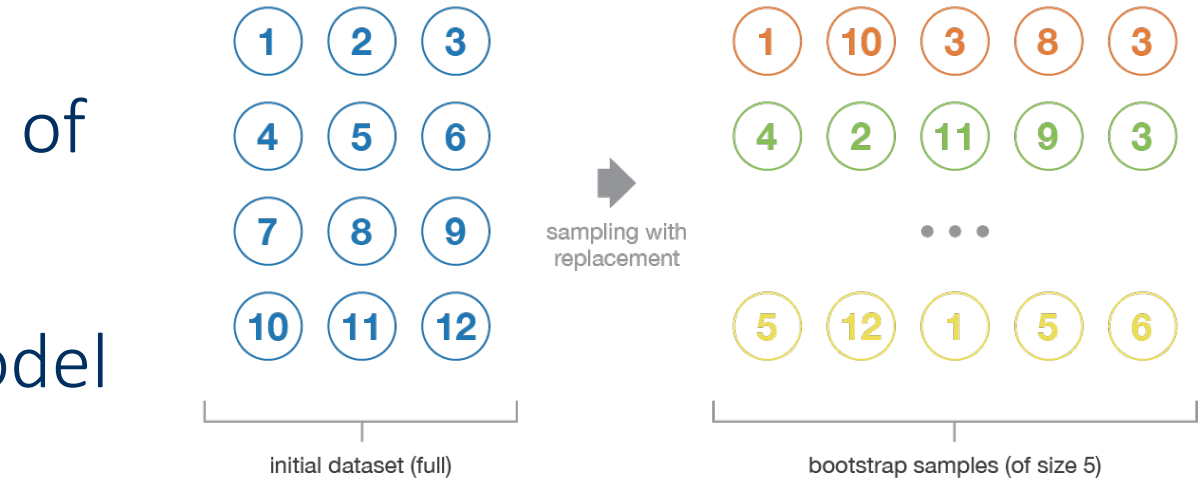
- Ensemble learning: Combine multiple models to produce a more powerful ensemble model
- Single model/weak learner/**base learner** (models to be combined) – the models we introduced in previous lectures
- Ensemble model/strong learner/ensemble learner (combined model) – a more accurate and/or robust model
- More details [link]

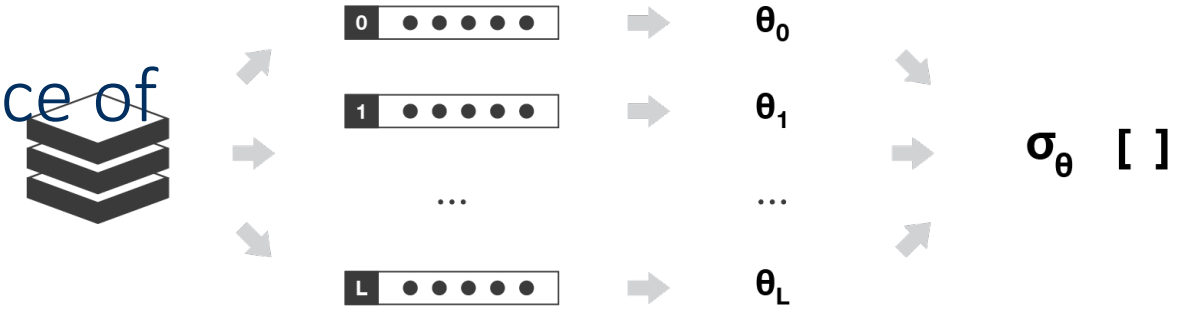
THREE TYPES OF ENSEMBLE LEARNING

- Bagging
- Boosting
- Stacking

BOOTSTRAP

- Sampling many random subsets of original data
- Fit the same model to each random subsets
- Often used to estimate variance of an estimated parameter

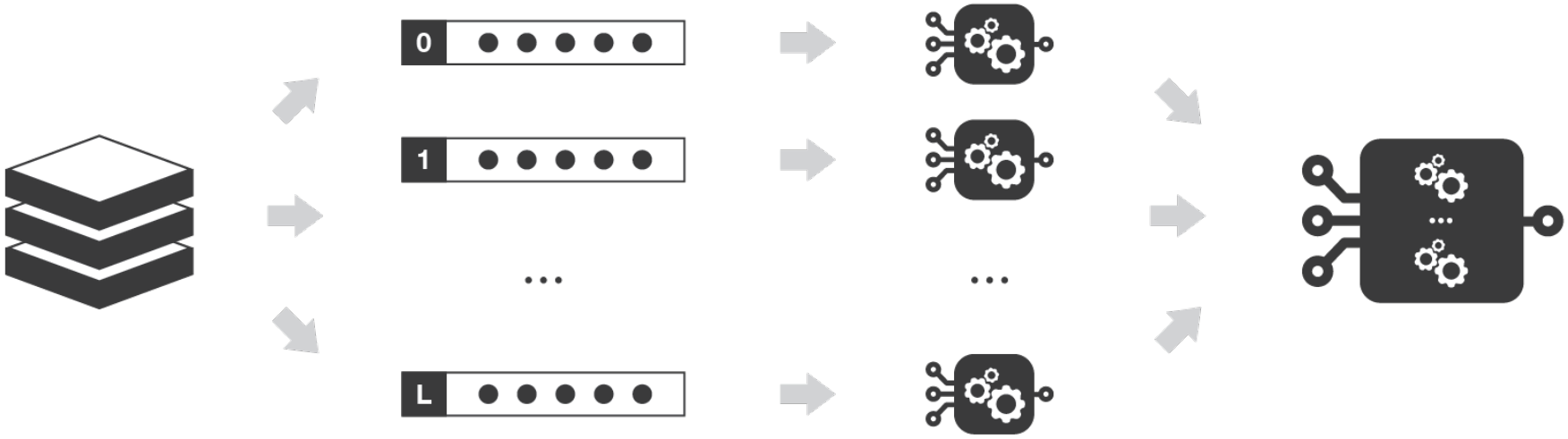




BAGGING - (BOOTSTRAP AGGREGATING)

Bagging: average many bootstrap outputs as the final output

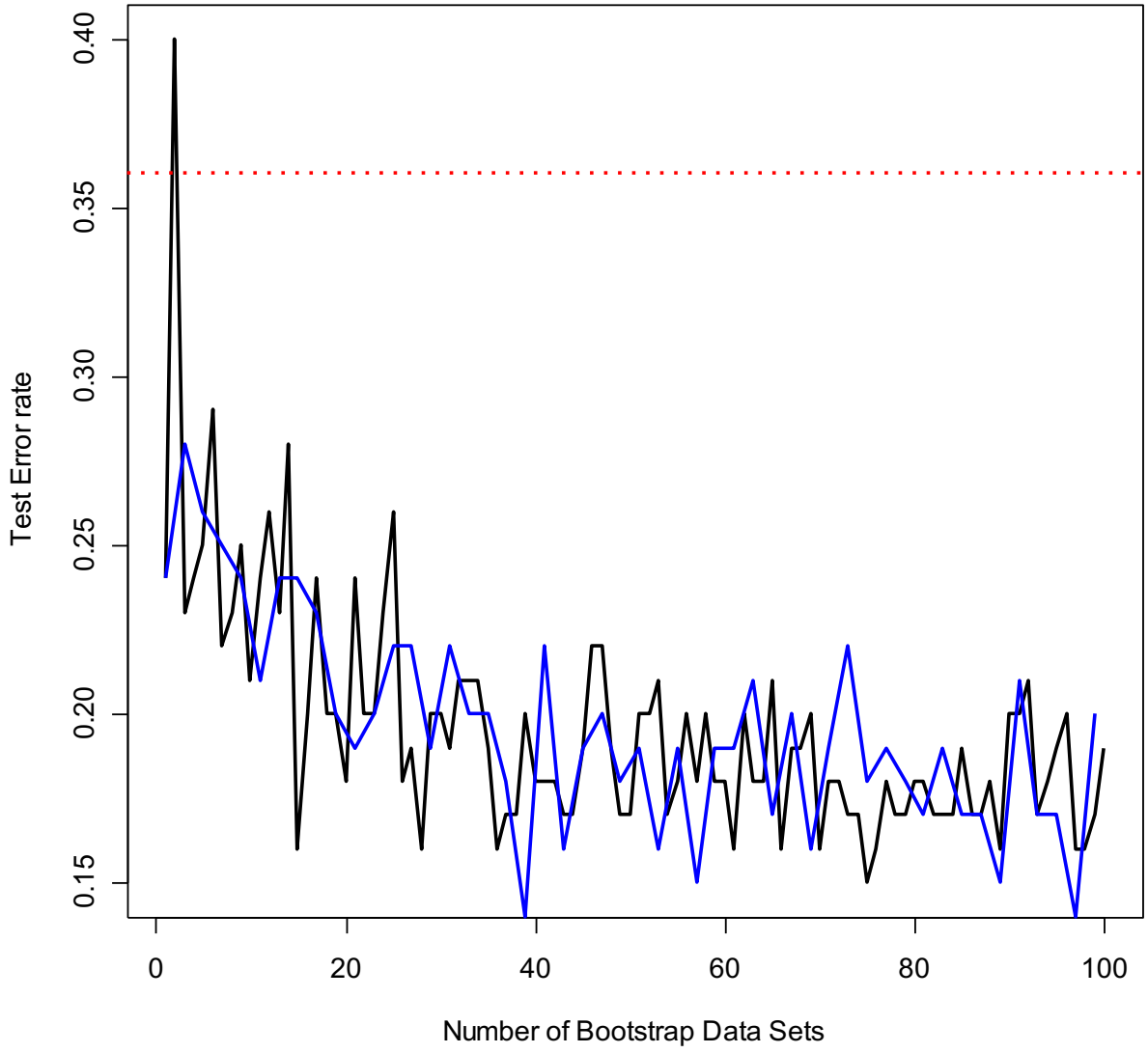
- Continuous outcomes: simple averaging
- Categorical outcomes: majority voting



initial dataset | L bootstrap samples | weak learners fitted on each bootstrap sample | ensemble model (kind of average of the weak learners)

CAR SEAT DATA EXAMPLE

- The red line represents the test error rate using a single tree.
- The black line corresponds to the bagging error rate using majority vote while the blue line averages the probabilities.

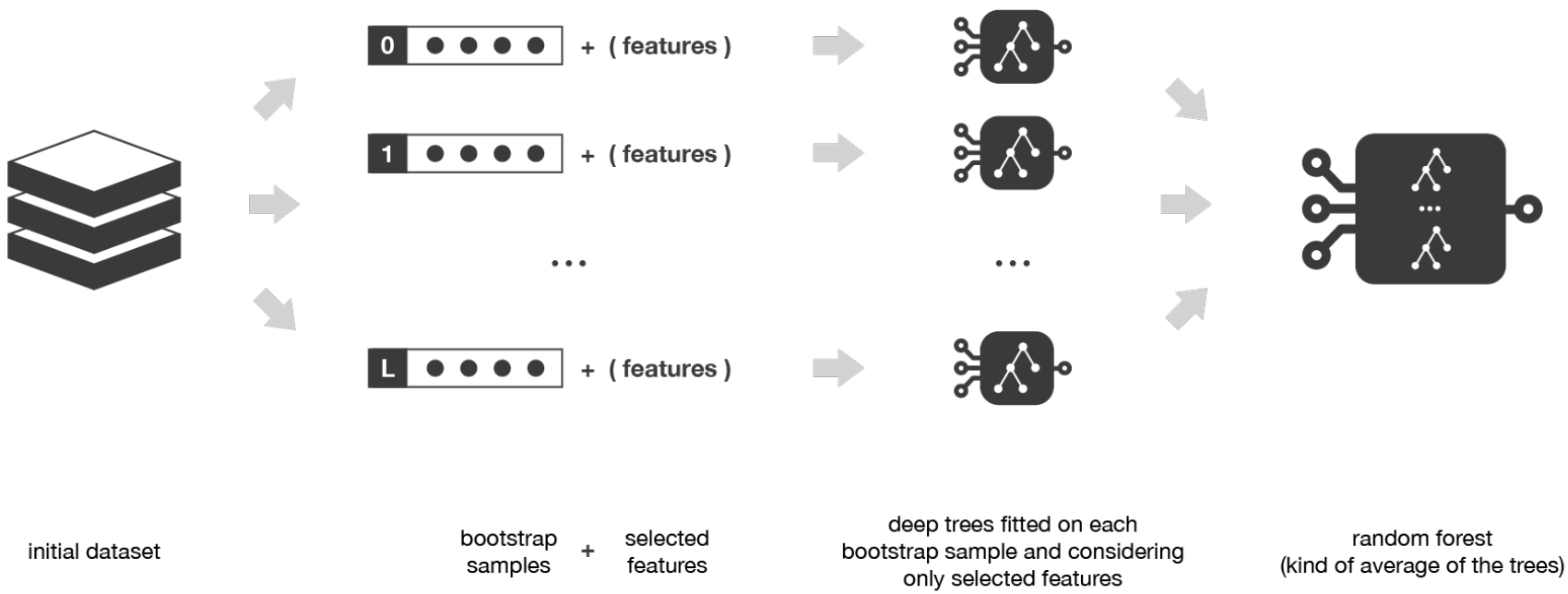


RANDOM FORESTS

- Builds on the idea of bagging, but it provides an improvement because it de-correlates the trees
- Averaging over deep trees to reduce variance
  - Deep tree - high variance, low bias
  - Shallow tree - low variance, high bias
- Sample both records and **features**
- Not as easy to interpret as a single tree

HOW DOES RANDOM FOREST WORKS?

- Build many decision trees on bootstrapped training samples
- Each tree only considered, a random sample of *m* predictors
-
- The full set of *p* predictors (Usually $m \approx \sqrt{p}$)



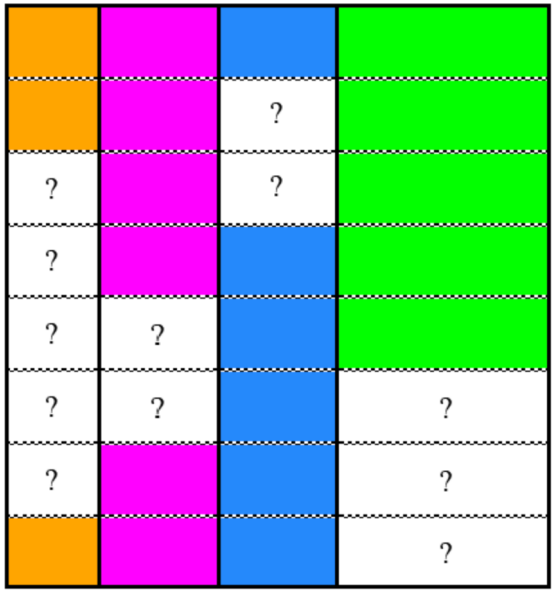
WHY ARE SAMPLING PREDICTORS?

- Suppose a very strong predictor in the data set along with a number of other moderately strong predictors

- In the collection of bagged trees, most or all of them will use the very strong predictor for the first split!
  - First split on the strongest variable might not lead to global optimal solution (problem of greedy algorithm)
  - All bagged trees will look similar. Hence all the predictions from the bagged trees will be highly correlated. Averaging many highly correlated quantities does not lead to a large variance reduction. (need de-correlate the trees)

WHY ARE SAMPLING PREDICTORS?

Better handle missing data



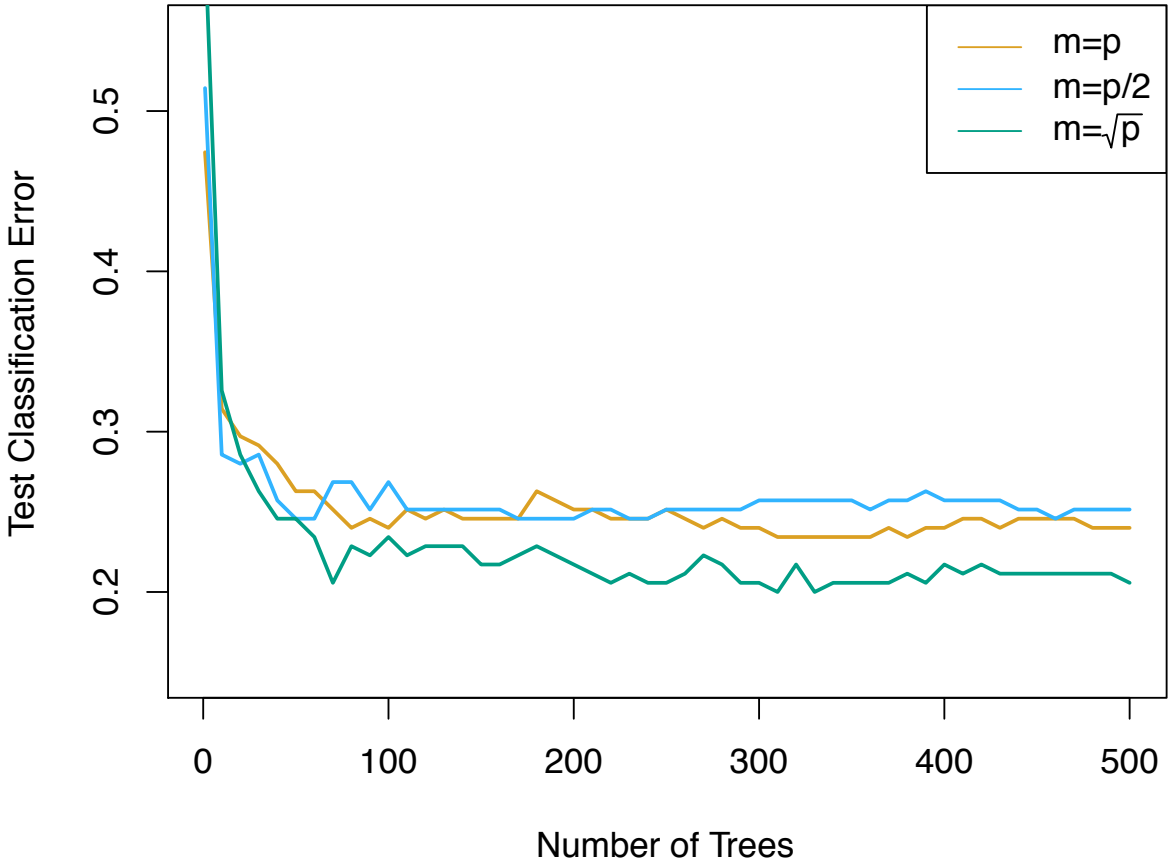
TEST (OUT-OF-BAG) ERROR ESTIMATION

- Since bagging involves random selection of subsets of observations to build a training data set, then the remaining non-selected part could be the testing data.

- On average, each bagged tree makes use of around 2/3 of the observations, so we end up having 1/3 of the observations used for testing

CHOICE OF "M" IN RANDOM FOREST

- Notice when random forests are built using m = p, then this amounts simply to bagging.



VARIABLE IMPORTANCE MEASURE

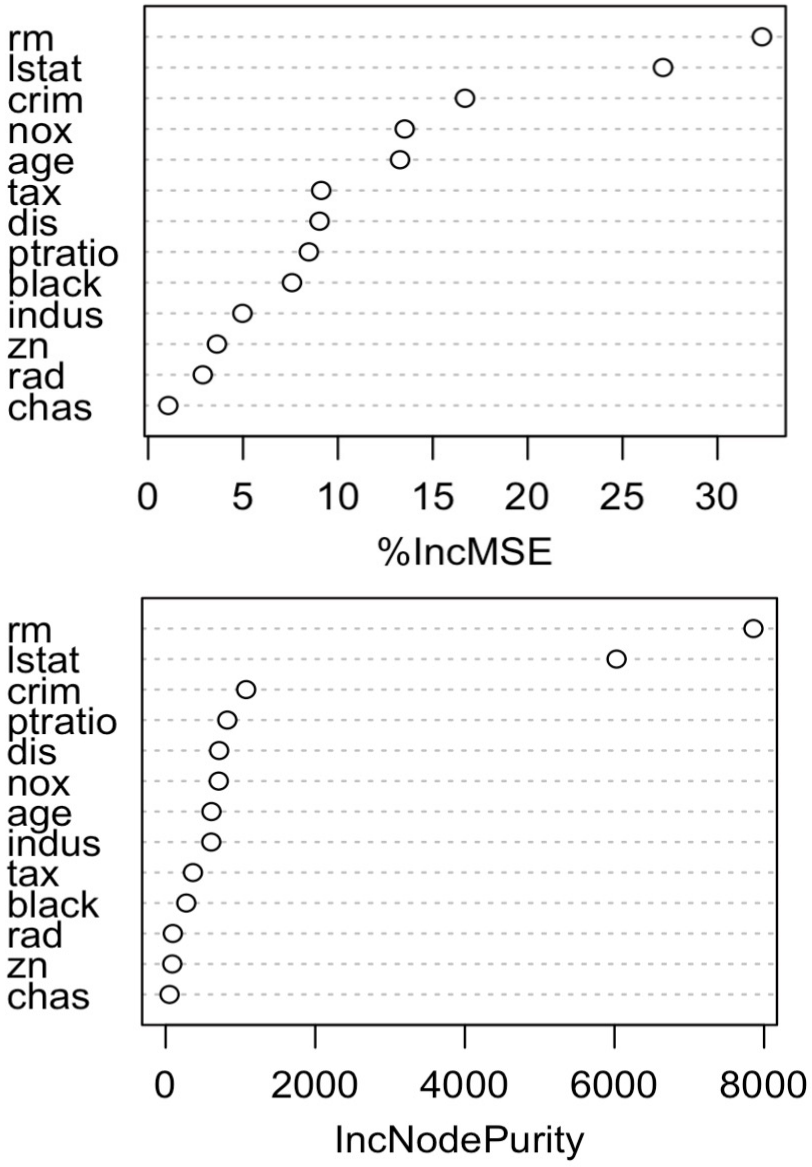
- Bagging typically improves the accuracy over prediction using a single tree, but it is now hard to interpret the model!
- We have hundreds of trees, and it is no longer clear which variables are most important to the procedure
- Thus, bagging improves prediction accuracy at the expense of interpretability
- But, we can still get an overall summary of the importance of each predictor using Relative Influence Plots

RELATIVE INFLUENCE PLOTS

How do we decide which variables are most useful in predicting the response?

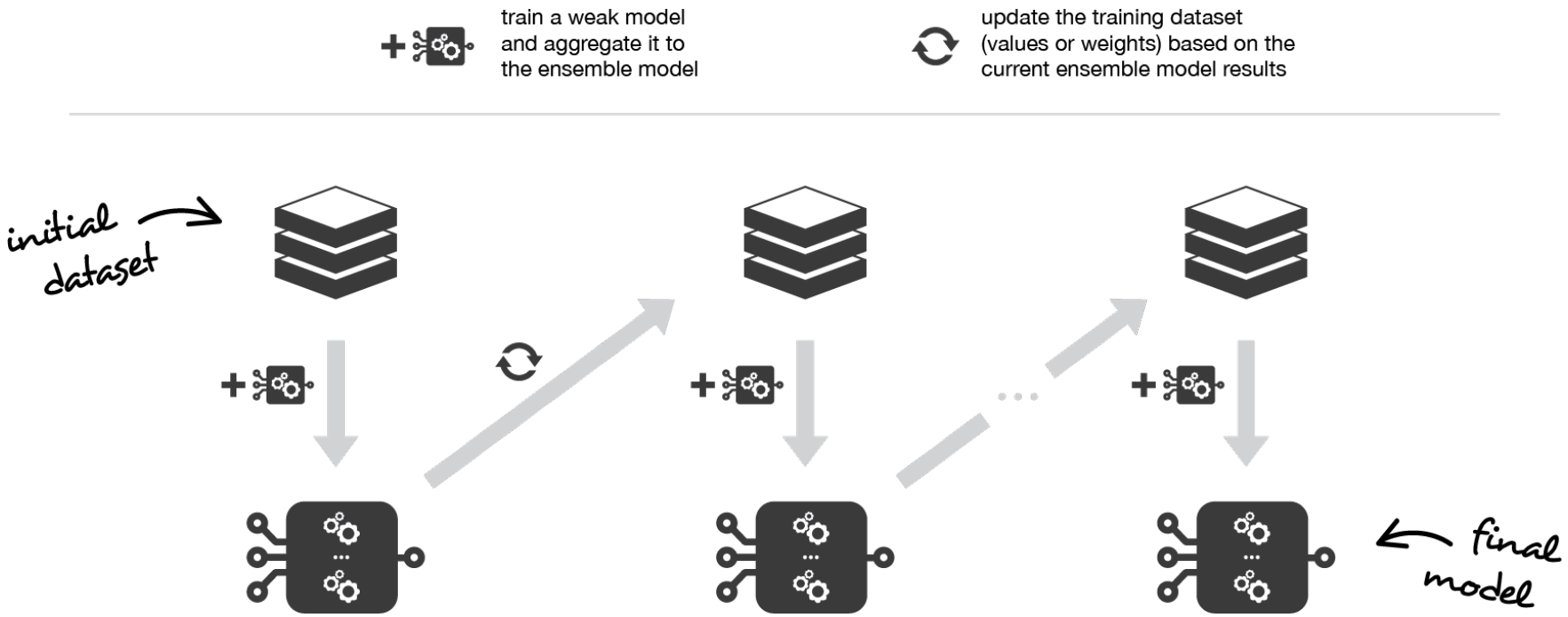
- We can compute something called relative influence plots.
- These plots give a score for each variable.
- These scores represents the decrease in MSE when splitting on a particular variable
- A number close to zero indicates the variable is not important and could be dropped.
- The larger the score the more influence the variable has.

Boston Data



BOOSTING

- Main focus: reduce bias (could also have effect of reducing variance)
- Base models often have low variance and high bias
- Fit the same model iteratively and sequentially
- Focus on most difficult-to-predict observations in each iteration



TWO TYPES OF BOOSTING

Both types of boosting represent ensemble learner as weighted average of base learners

$$s_L(.) = \sum_{l=1}^{L} c_l \times w_l(.) \qquad \text{where } c_l\text{'s are coefficients and } w_l\text{'s are weak learners}$$

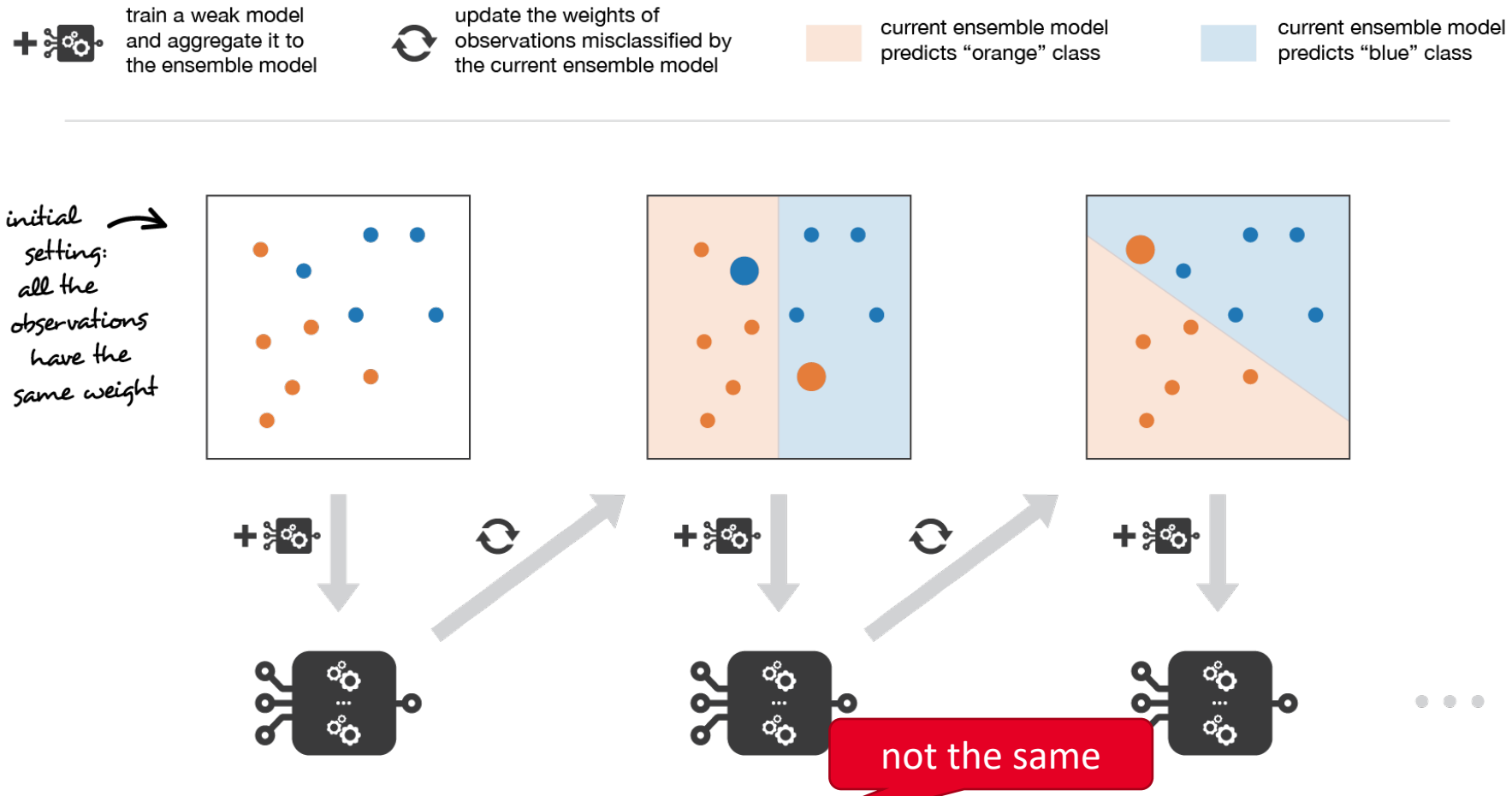
1. adaboost (Adaptive boosting)

   Update weights of records in training data

2. Gradient boosting

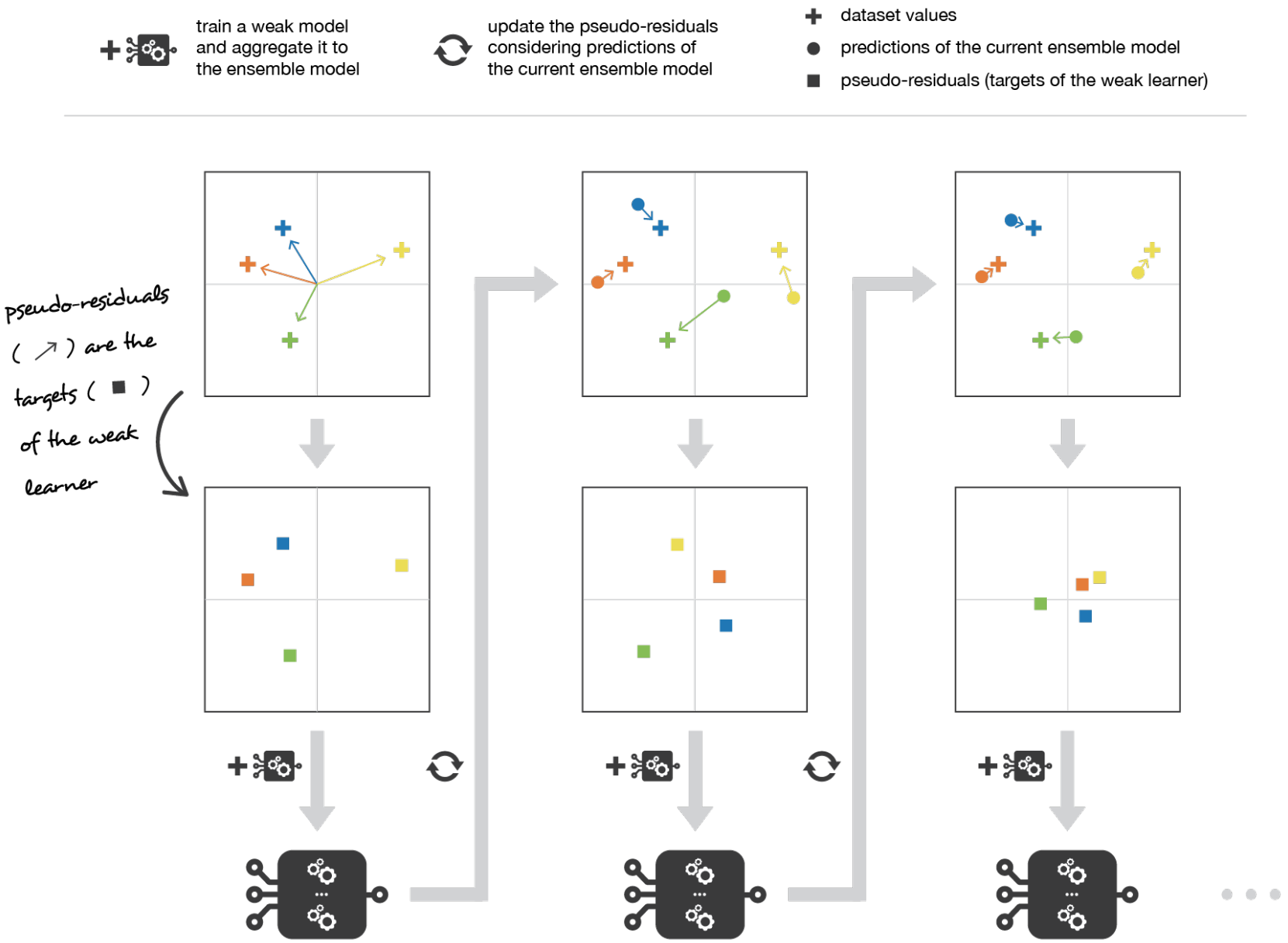
   Update values of records in training data

ADABOOST



Adaboost updates weights of the observations at each iteration. Weights of well classified observations decrease relatively to weights of misclassified observations. Models that perform better have higher weights $\mathbf{c}_\ell$ in the final ensemble model.

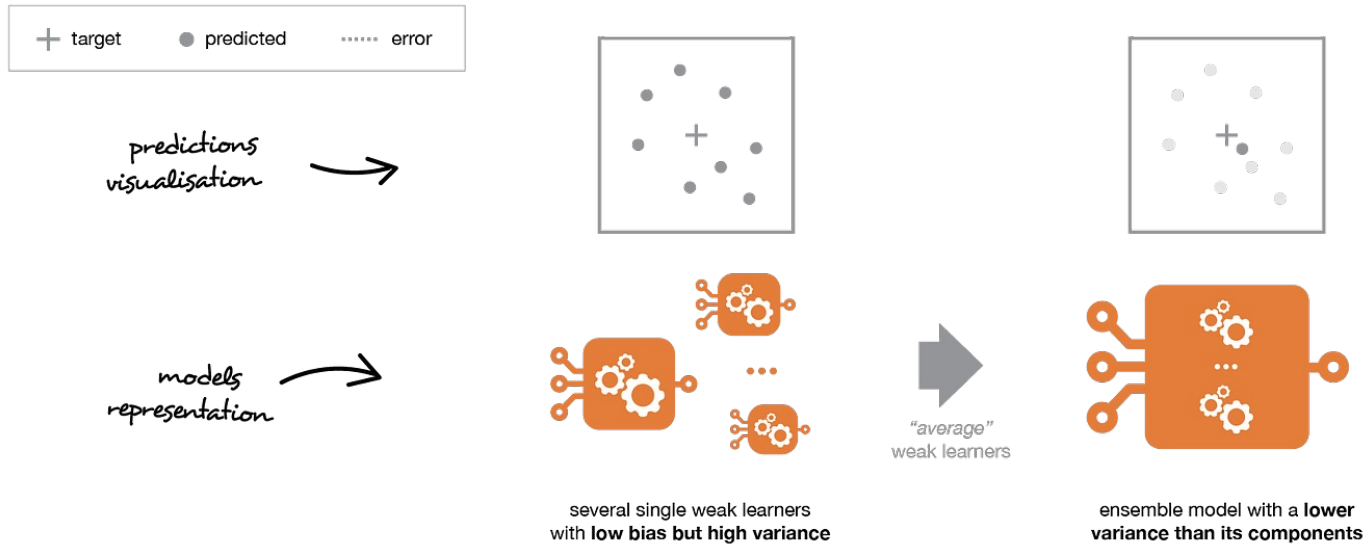
GRADIENT BOOSTING



XGBOOST (EXTREME GRADIENT BOOSTING)

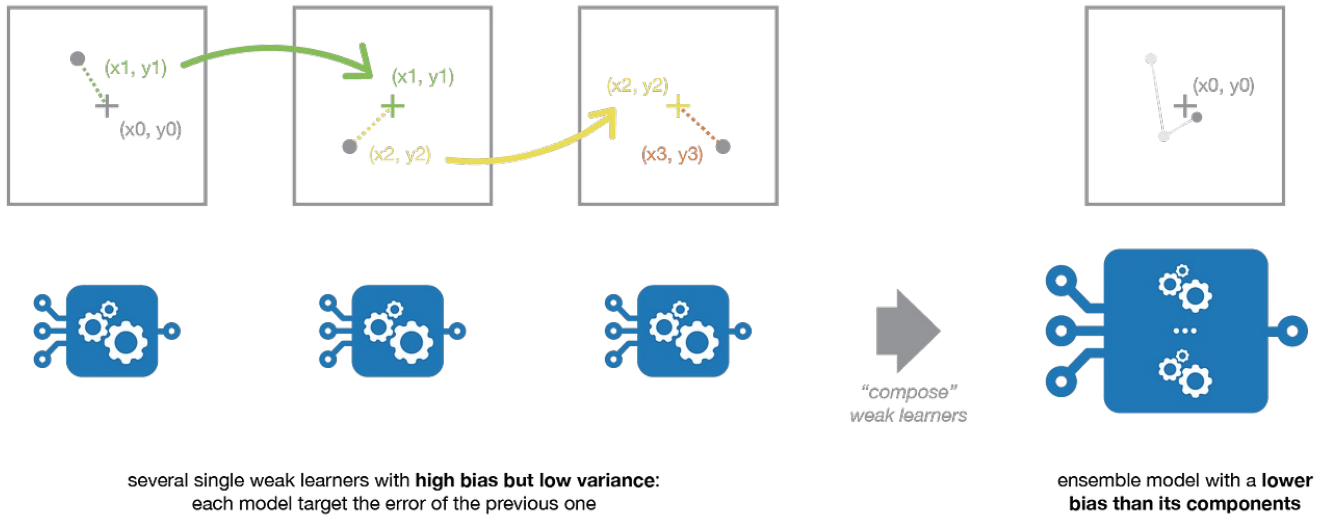
- A special gradient boosting, with many good features
- Winning solutions of many machine learning competitions (XGBoost - ML winning solutions (incomplete list))
- Tutorial
  https://xgboost.readthedocs.io/en/latest/tutorials/model.html

BAGGING VS BOOSTING



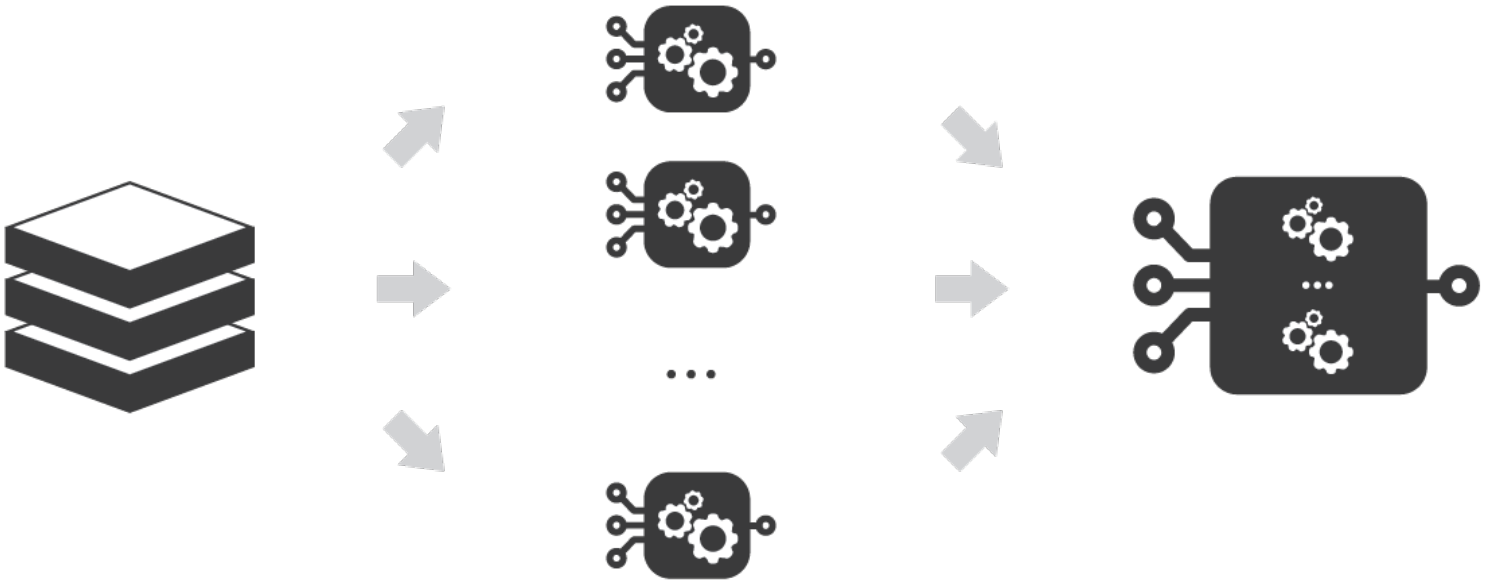
LOW BIAS HIGH VARIANCE WEAK LEARNERS

LOW VARIANCE HIGH BIAS WEAK LEARNERS



STACKING

- Heterogeneous weak learners
- Combine them by training a meta-model
- Let various weak learners borrow strength from each other



STEPS OF STACKING

- Split the training data in two folds
- Choose L weak learners and fit them to data of the first fold
- For each of the L weak learners, make predictions for observations in the second fold
- Fit the meta-model on the second fold, using predictions made by the weak learners as inputs

- Can replace the two-folds splitting by a “K-fold cross-validation” like approach

SUMMARY: 3 TYPES OF ENSEMBLE LEARNING

Bagging

- ***Homogeneous*** weak learners
- Learned independently from each other in ***parallel***
- Combined with ***deterministic averaging***
- Focused on reducing ***variance***

Boosting

- ***Homogeneous*** weak learners
- Learned ***sequentially*** in a very adaptative way (a base model depends on the previous ones)
- Combined with ***deterministic strategy***
- Focused reducing ***bias***

Stacking

- ***Heterogeneous*** weak learners
- Learned in ***parallel***
- Combined by training a ***meta-model***
- Focused reducing ***bias***

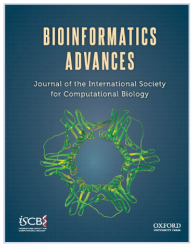
APPLICATION

OXFORD ACADEMIC | Journals | Books

BIOINFORMATICS ADVANCES

ISCB INTERNATIONAL SOCIETY FOR COMPUTATIONAL BIOLOGY

Issues | Advance articles | Submit ▾ | Alerts | About ▾



Volume 3, Issue 1
2023
(In Progress)

Get citation

JOURNAL ARTICLE

scAnnotate: an automated cell-type annotation tool for single-cell RNA-sequencing data

Xiangling Ji (Data curation), Danielle Tsao (Data curation), Kailun Bai (Data curation), Min Tsao (Conceptualization), Li Xing (Conceptualization) ✉, Xuekui Zhang (Conceptualization) ✉

Bioinformatics Advances, Volume 3, Issue 1, 2023, vbad030,
https://doi.org/10.1093/bioadv/vbad030
Published: 13 March 2023 **Article history** ▾

PDF Help

PDF | Views ▾ | Cite | Permissions | Share ▾

R package: https://cran.r-project.org/web/packages/scAnnotate/index.html

CELL TYPE CLASSIFICATION

Gene expression distribution is important for cell classification, but most classification methods do not incorporate the distribution of predictors.

Modelling the expressions of many genes involves high dimensional distribution with too many parameters in correlation structure. (curse of dimensionality)

Modelling the multivariate non-Gaussian distribution can be hard.

We build a classification model using each gene's expression distribution and use an ensemble approach to aggregate their classification results.

Split data in two parts, one part learns individual models, and the rest learns aggregation parameters.

MULTI-TASK PREDICTION

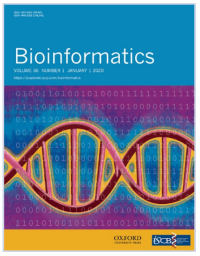
AVAILABILITY

OXFORD ACADEMIC | Journals | Books

Bioinformatics

ISCB INTERNATIONAL SOCIETY FOR COMPUTATIONAL BIOLOGY

Issues | Advance articles | Submit ▾ | Alerts | About ▾



Volume 36, Issue 1
January 2020

Article Contents

Get citation

JOURNAL ARTICLE

Simultaneous prediction of multiple outcomes using revised stacking algorithms FREE

Li Xing, Mary L Lesperance, Xuekui Zhang

Bioinformatics, Volume 36, Issue 1, January 2020, Pages 65–72,
https://doi.org/10.1093/bioinformatics/btz531

Published: 02 July 2019 **Article history** ▾

PDF Help

PDF | Split View | Cite | Permissions | Share ▾

R package: https://cran.r-project.org/web/packages/MTPS/index.html

SCIENTIFIC PROBLEM: PERSONALIZED TREATMENT FOR HIV DISEASE

Problem:

HIV is difficult to treat

►HIV virus mutates at a high rate.

►Mutated viruses easily develop resistance to current drugs.

Solution:

Personalized Medicine

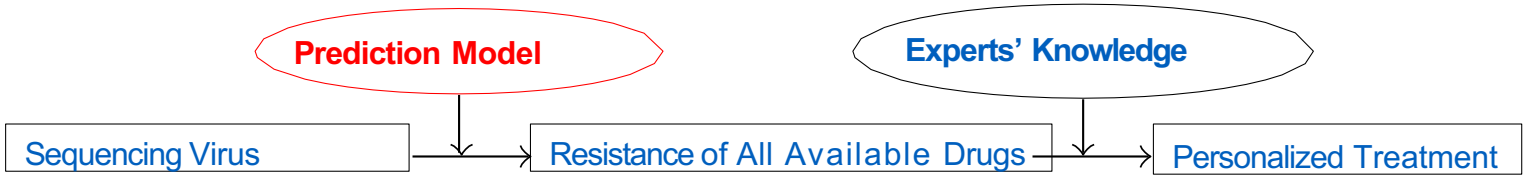
►Use mutation information to predict resistance of **multiple** HIV drugs

►Pick the most suitable drug

MULTI-TASK PREDICTION PROBLEM

- **Aim**: Build a multi-task **prediction model** (mutation → resistance of drugs)

- **Application**: For a new patient, we obtain his/her HIV virus.



- **Data**: The HIV Drug Resistance Database is built by a Stanford team. (https://hivdb.stanford.edu/)

HIV DRUG RESISTANCE DATA

- **Sample size** : 1498.
- **Outcomes**: Resistance information of 5 drugs in the NRTI class.

$$\text{Resistance} = \frac{IC_{50}(\text{drug concentration for resistant strain})}{IC_{50}(\text{drug concentration for wild type})}$$

- **Predictors**: 240 mutations of each isolate/virus.

Sample Size = 1498

	ABC	3TC	AZT	D4T	DDI	X.4S	X.6D	X.6E	X.6K	X.8I	X.8V	X.1
1	0.63346846	2.30103000	0.59106461	0.14612804	0.07918125	0	0	0	0	0	0	0
3	0.04139269	0.14612804	1.44715803	0.00000000	-0.09691001	0	0	0	0	0	0	0
4	0.17609126	0.25527251	0.85125835	0.07918125	0.04139269	0	0	0	0	0	0	0
6	0.85125835	2.30103000	-0.09691001	0.11394335	0.27875360	0	0	0	0	0	0	0
7	0.71600334	0.40654018	0.82607480	0.73239376	0.72427587	0	0	0	0	0	0	0
8	0.77085201	2.30103000	0.27875360	0.11394335	0.23044892	0	0	0	0	0	0	0
9	0.79934055	2.30103000	0.71600334	0.14612804	0.27875360	0	0	0	0	0	0	0
10	0.75587486	2.30103000	-0.52287875	0.07918125	0.30103000	0	0	0	0	0	0	1
11	0.65321251	0.79239169	3.00000000	0.38021124	0.20411998	0	0	0	0	0	0	0

HIV DRUG RESISTANCE DATA

- **Sample size** : 1498.
- **Outcomes** : Resistance information of 5 drugs in the NRTI class.

$$\text{Resistance} = \frac{IC_{50}(\text{drug concentration for resistant strain})}{IC_{50}(\text{drug concentration for wild type})}$$

- **Predictors**: 240 mutations of each virus.

Sample Size = 1498

	ABC	3TC	AZT	D4T	DDI	X.4S	X.6D	X.6E	X.6K	X.8I	X.8V	X.1
1	0.63346846	2.30103000	0.59106461	0.14612804	0.07918125	0	0	0	0	0	0	0
3	0.04139269	0.14612804	1.44715803	0.00000000	-0.09691001	0	0	0	0	0	0	0
4	0.17609126	0.25527251	0.85125835	0.07918125	0.04139269	0	0	0	0	0	0	0
6	0.85125835	2.30103000	-0.09691001	0.11394335	0.27875360	0	0	0	0	0	0	0
7	0.71600334	0.40654018	0.82607480	0.73239376	0.72427587	0	0	0	0	0	0	0
8	0.77085201	2.30103000	0.27875360	0.11394335	0.23044892	0	0	0	0	0	0	0
9	0.79934055	2.30103000	0.71600334	0.14612804	0.27875360	0	0	0	0	0	0	0
10	0.75587486	2.30103000	-0.52287875	0.07918125	0.30103000	0	0	0	0	0	0	1
11	0.65321251	0.79239169	3.00000000	0.38021124	0.20411998	0	0	0	0	0	0	0

HIV DRUG RESISTANCE DATA

- **Sample size** : 1498.
- **Outcomes** : Resistance information of 5 drugs in the NRTI class.

$$\text{Resistance} = \frac{IC_{50}(\text{drug concentration for resistant strain})}{IC_{50}(\text{drug concentration for wild type})}$$

- **Predictors** : 240 mutations of each virus.

Sample Size = 1498

	ABC	3TC	AZT	D4T	DDI	X.4S	X.6D	X.6E	X.6K	X.8I	X.8V	X.1
1	0.63346846	2.30103000	0.59106461	0.14612804	0.07918125	0	0	0	0	0	0	0
3	0.04139269	0.14612804	1.44715803	0.00000000	-0.09691001	0	0	0	0	0	0	0
4	0.17609126	0.25527251	0.85125835	0.07918125	0.04139269	0	0	0	0	0	0	0
6	0.85125835	2.30103000	-0.09691001	0.11394335	0.27875360	0	0	0	0	0	0	0
7	0.71600334	0.40654018	0.82607480	0.73239376	0.72427587	0	0	0	0	0	0	0
8	0.77085201	2.30103000	0.27875360	0.11394335	0.23044892	0	0	0	0	0	0	0
9	0.79934055	2.30103000	0.71600334	0.14612804	0.27875360	0	0	0	0	0	0	0
10	0.75587486	2.30103000	-0.52287875	0.07918125	0.30103000	0	0	0	0	0	0	1
11	0.65321251	0.79239169	3.00000000	0.38021124	0.20411998	0	0	0	0	0	0	0
12	0.69019608	0.51851394	2.34707327	0.46239800	0.20411998	0	0	0	0	0	0	0

CURRENT POPULAR METHODS

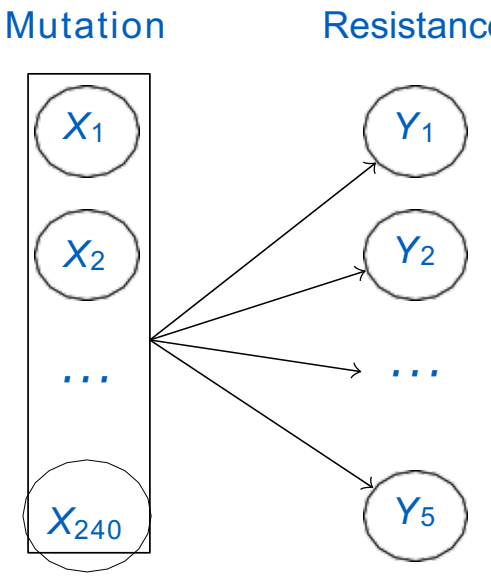
Single-task Prediction Methods

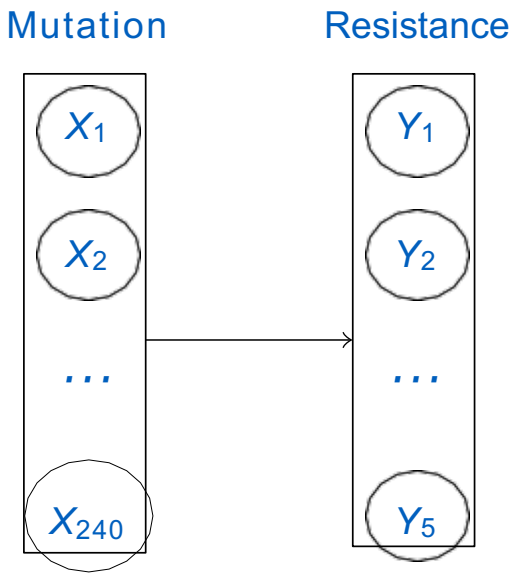
1. Univariate Elastic Net (**uLARS**)
2. Gradient boosting (**XGBoost**)
3. many other methods · · ·

Multi-task Prediction Methods

1. Multivariate Normal Elastic Net (**mLARS**)
2. Neural Network (**NN**)
3. Ensembles of Classifier Chain (**ECC**)

(Heider 2013)





https://www.simplilearn.com/tutorials/deep-learning-tutorial/neural-network

OUR METHODS

The difficulty of multi-task prediction : modelling outcomes' covariance structure

We propose methods based on revisions of the stacking algorithm

Features:

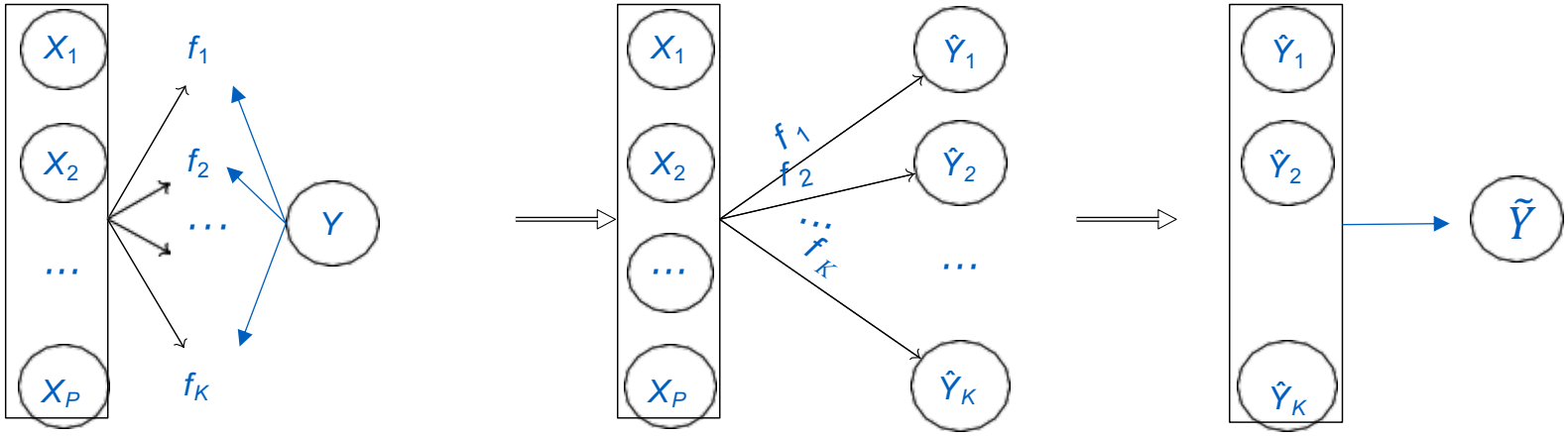
- allow multiple prediction tasks to borrow information from each other;
- Flexible: easy to construct using available univariate methods
- Powerful: outperform the current popular methods.

ORIGINAL STACKING ALGORITHM FOR SINGLE-TASK PREDICTION

Learning univariate prediction functions f's

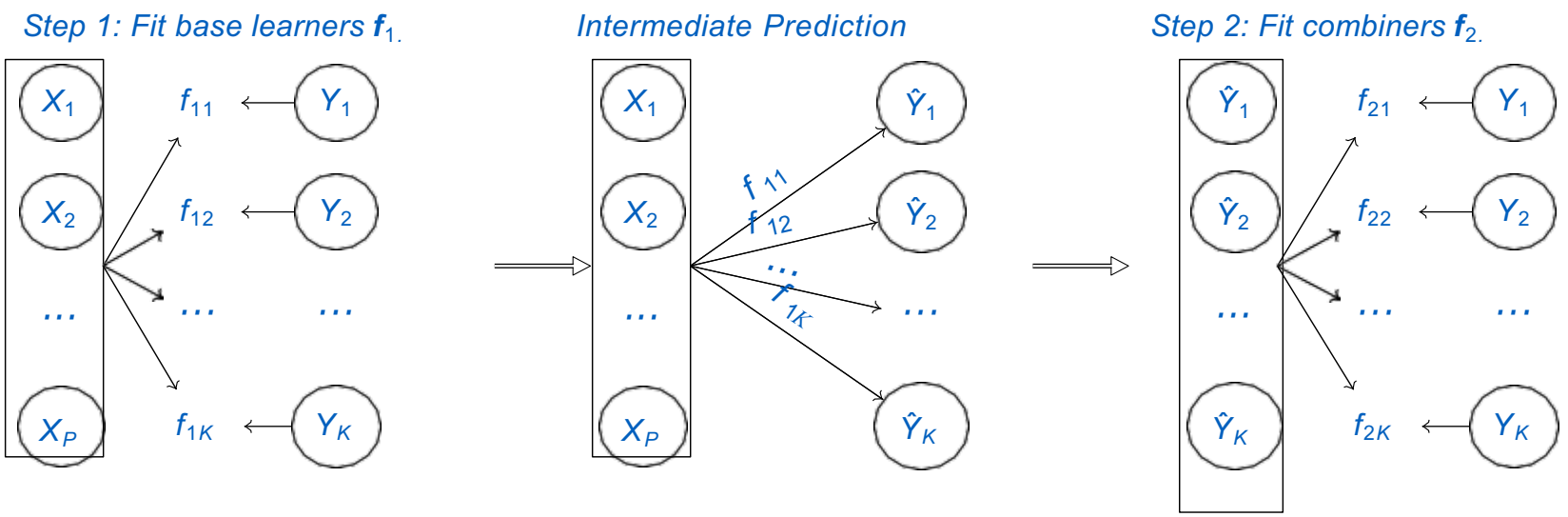
Intermediate Prediction

Final prediction (combiner)



STANDARD STACKING (SS) FOR MULTI-TASK PREDICTION

Learning Process:



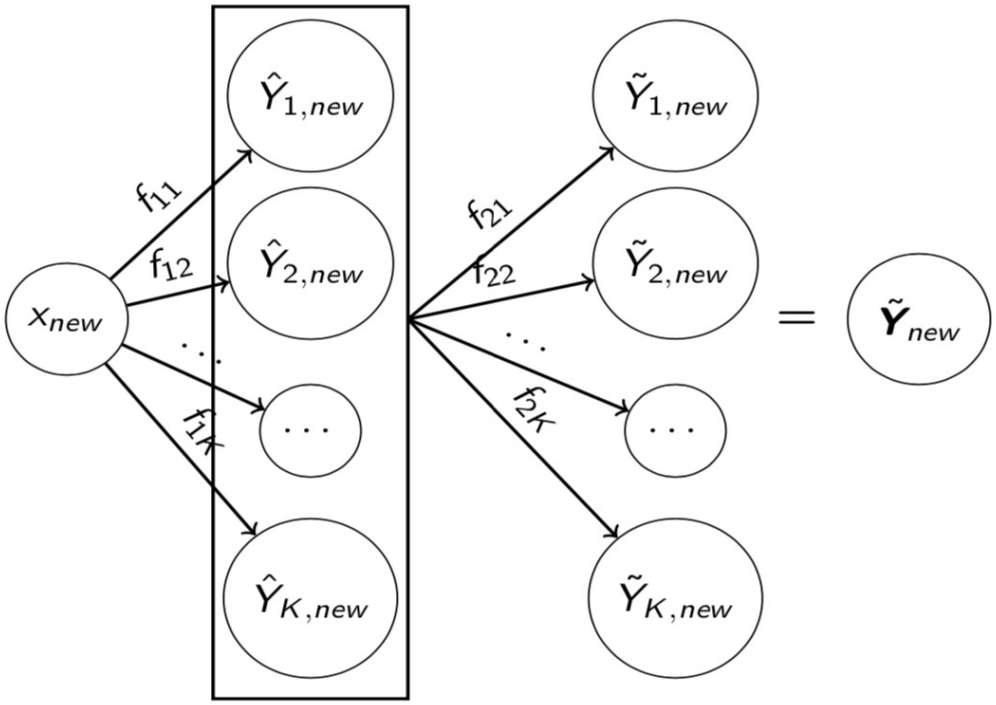
[*Step 1: Individual models*] Learn f_{1k} for the intermediate prediction of the k-th drug, using predictors $\boldsymbol{X}$ and outcome $\boldsymbol{Y}_k$, for $k = 1, \ldots, K$

[*Calculate fitted values*] $\hat{\boldsymbol{Y}}_k = f_{1k}(\boldsymbol{X})$ and $\hat{\boldsymbol{Y}} = (\hat{\boldsymbol{Y}}_1, \ldots, \hat{\boldsymbol{Y}}_K)$

[*Step 2: Combiner models*] Learn combiner models f_{2k} for the final prediction of the k-th drug, using predictors $\hat{\boldsymbol{Y}}$ and outcome $\boldsymbol{Y}_k$

STANDARD STACKING (SS). -CONTINUED

Prediction Process:



[*Final predictions*] Predict resistance of all drugs from mutation data of a new virus $\boldsymbol{X}_{\text{new}}$, for $k = 1, \ldots, K$

$$\tilde{\boldsymbol{Y}}_{k,\text{new}} = f_{2k}\left(f_{11}(\boldsymbol{X}_{\text{new}}), \ldots, f_{1K}(\boldsymbol{X}_{\text{new}})\right). \qquad (1)$$

PROPOSED CROSS VALIDATION STACKING (CVS)

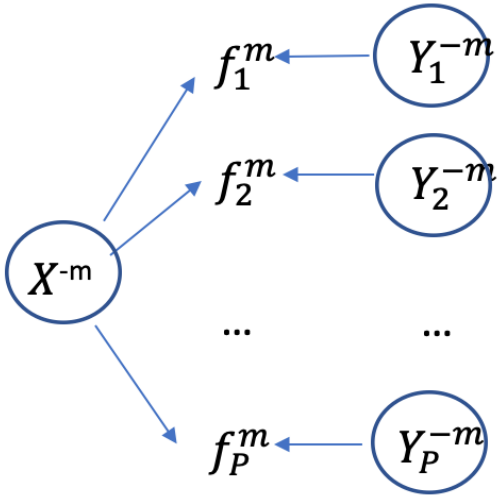
[Problem 1]:

$\hat{Y}$ are fitted values, but $\hat{Y}_{new}$ are predicted values. In general, their variance are different, hence the input data for learning and applying Step 2 functions are different.

[Solution]:

We propose Cross-validation Stacking (**CVS**), which revise Step 1 of SS

PROPOSED CROSS-VALIDATION STACKING (CVS)

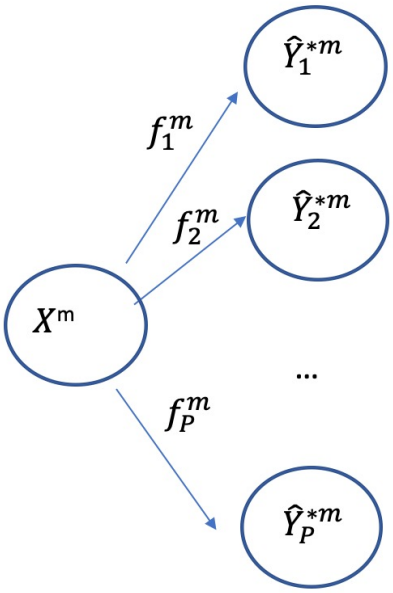


[*Partition*] Randomly partition data into M folds. Let $(\boldsymbol{X}^{[m]}, \boldsymbol{Y}^{[m]})$ and $(\boldsymbol{X}^{[-m]}, \boldsymbol{Y}^{[-m]})$ be the data inside and outside of the m-th fold

for $m = 1$ to M **do**

[*Step 1: Individual models*] Learn $f_{1k}^{[m]}$ for the intermediate prediction, using predictors $\boldsymbol{X}^{[-m]}$ and outcome $\boldsymbol{Y}_k^{[-m]}$, for all k's

[*Calculate cross-validation predicted values*]
$\hat{\boldsymbol{Y}}_k^{*[m]} = f_{1k}^{[m]}(\boldsymbol{X}^{[m]})$ and $\hat{\boldsymbol{Y}}^{*[m]} = (\hat{\boldsymbol{Y}}_1^{*[m]}, \ldots, \hat{\boldsymbol{Y}}_K^{*[m]})$
end for



[*Combine the results from M folds*]

$\hat{\boldsymbol{Y}}^* = (\hat{\boldsymbol{Y}}^{*[1]T}, \ldots, \hat{\boldsymbol{Y}}^{*[M]T})^T$ and $f_{1k} = (f_{1k}^{[1]} + \ldots + f_{1k}^{[M]})/M$

RESIDUAL STACKING (RS)

[Problem 2]:

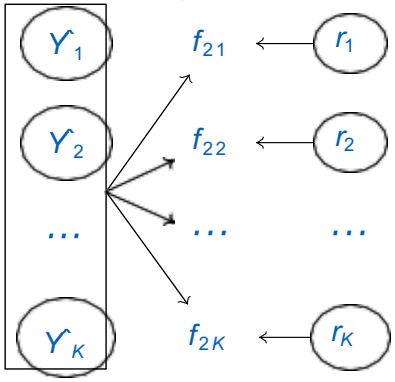
If Step 1 already provides accurate predictions, but the results of Step 2 models are not accurate due to data noise or unstable model fitting, the Step 1 prediction accuracy may be compromised by stacking.

[Solution]:

We propose Residual Stacking (**RS**), *similar to boosting,* which revise Step 2 of SS

OUR METHODS II: RESIDUAL STACKING (RS)

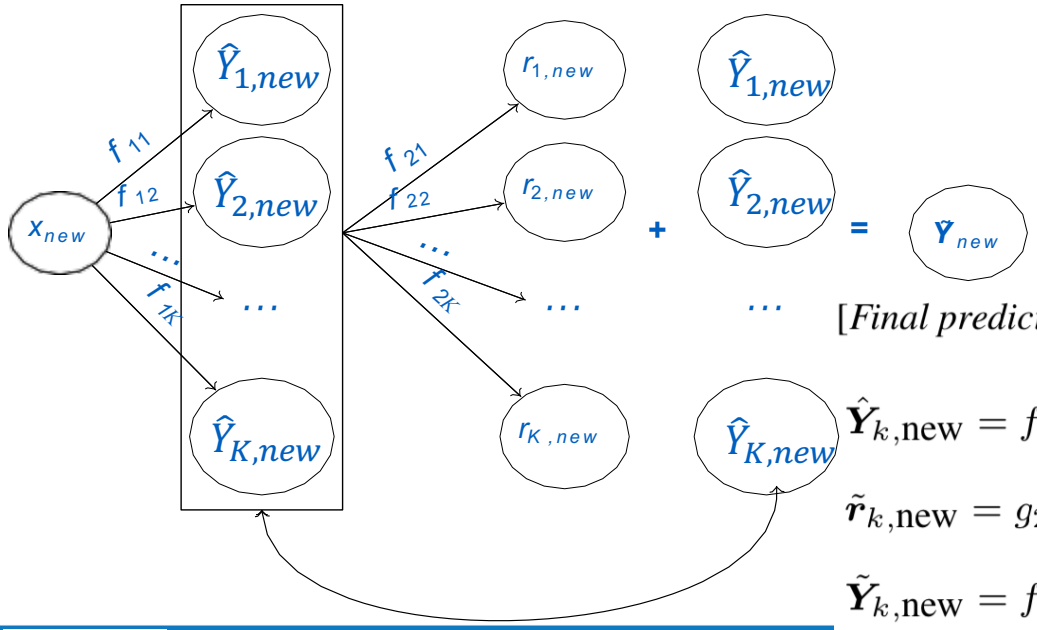
Step 2: Learning $\boldsymbol{f}_2$.



[*Calculate residuals*] $\boldsymbol{r}_k = \boldsymbol{Y}_k - \hat{\boldsymbol{Y}}_k$.

[*Step 2: Combiner models*] Learn combiner models g_{2k} for the final prediction of the k-th drug, using predictors $(\hat{\boldsymbol{Y}}_1, \ldots, \hat{\boldsymbol{Y}}_{k-1}, \hat{\boldsymbol{Y}}_{k+1}, \ldots, \hat{\boldsymbol{Y}}_K)$ and outcome $\boldsymbol{r}_k$, for $k = 1, \ldots, K$

Prediction Process:



[*Final predictions*] Predict resistance of all drugs for a new patient.

$$\hat{\boldsymbol{Y}}_{k,\text{new}} = f_{1k}(\boldsymbol{X}_{\text{new}})$$

$$\tilde{\boldsymbol{r}}_{k,\text{new}} = g_{2k}\left(\hat{\boldsymbol{Y}}_{1,\text{new}}, \ldots, \hat{\boldsymbol{Y}}_{k-1,\text{new}}, \hat{\boldsymbol{Y}}_{k+1,\text{new}}, \ldots, \hat{\boldsymbol{Y}}_{K,\text{new}}\right)$$

$$\tilde{\boldsymbol{Y}}_{k,\text{new}} = f_{1k}(\boldsymbol{X}_{new}) + \tilde{\boldsymbol{r}}_{k,\text{new}} \quad (2)$$

CROSS-VALIDATION RESIDUAL STACKING (CVRS)

We can combine CVS and RS, i.e. revise both step 1 and step 2. We call this algorithm as Cross-validation Residual Stacking (CVRS)

PERFORMANCE EVALUATIONS

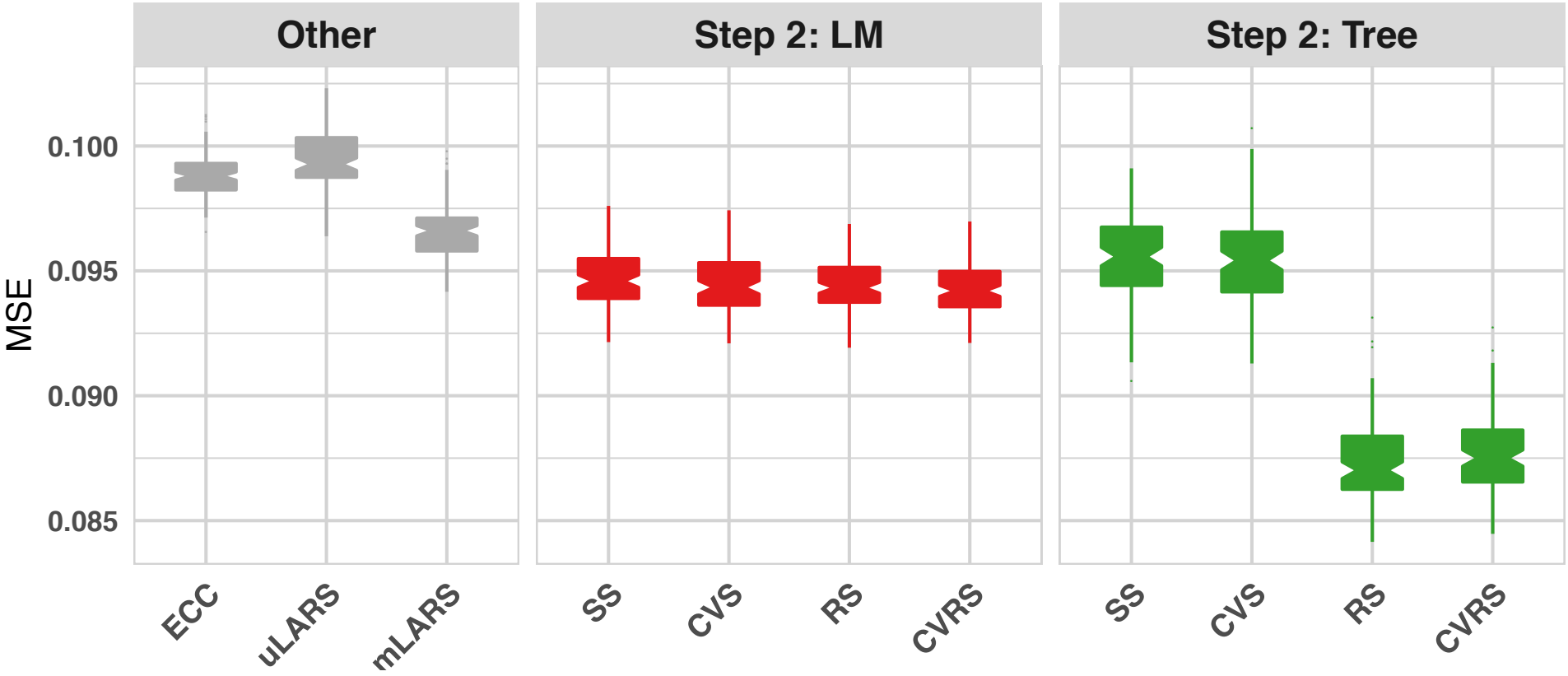
Compare our algorithms with competitors using cross-validation of 100 replicates of random split

To construct stacking algorithm, we use LARS as step 1 learners, and linear model or tree as step 2 models

Using MSE as criteria to compare prediction performance of continuous outcomes (i.e. log IC50 ratios)

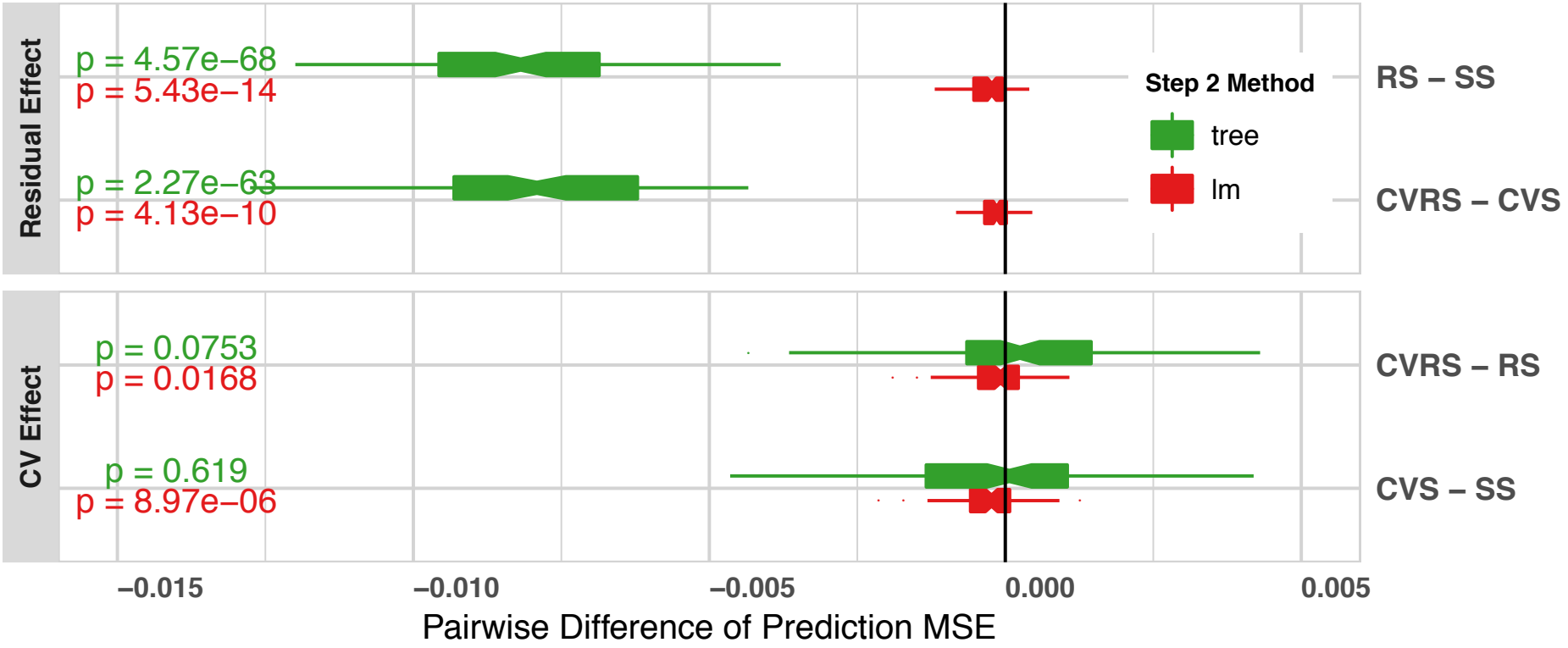
Using Accuracy and AUC to compare prediction performance of binary outcomes (i.e. resistant vs susceptive)

RESULT OF CROSS-VALIDATION: METHODS COMPARISON FOR CONTINUOUS OUTCOMES

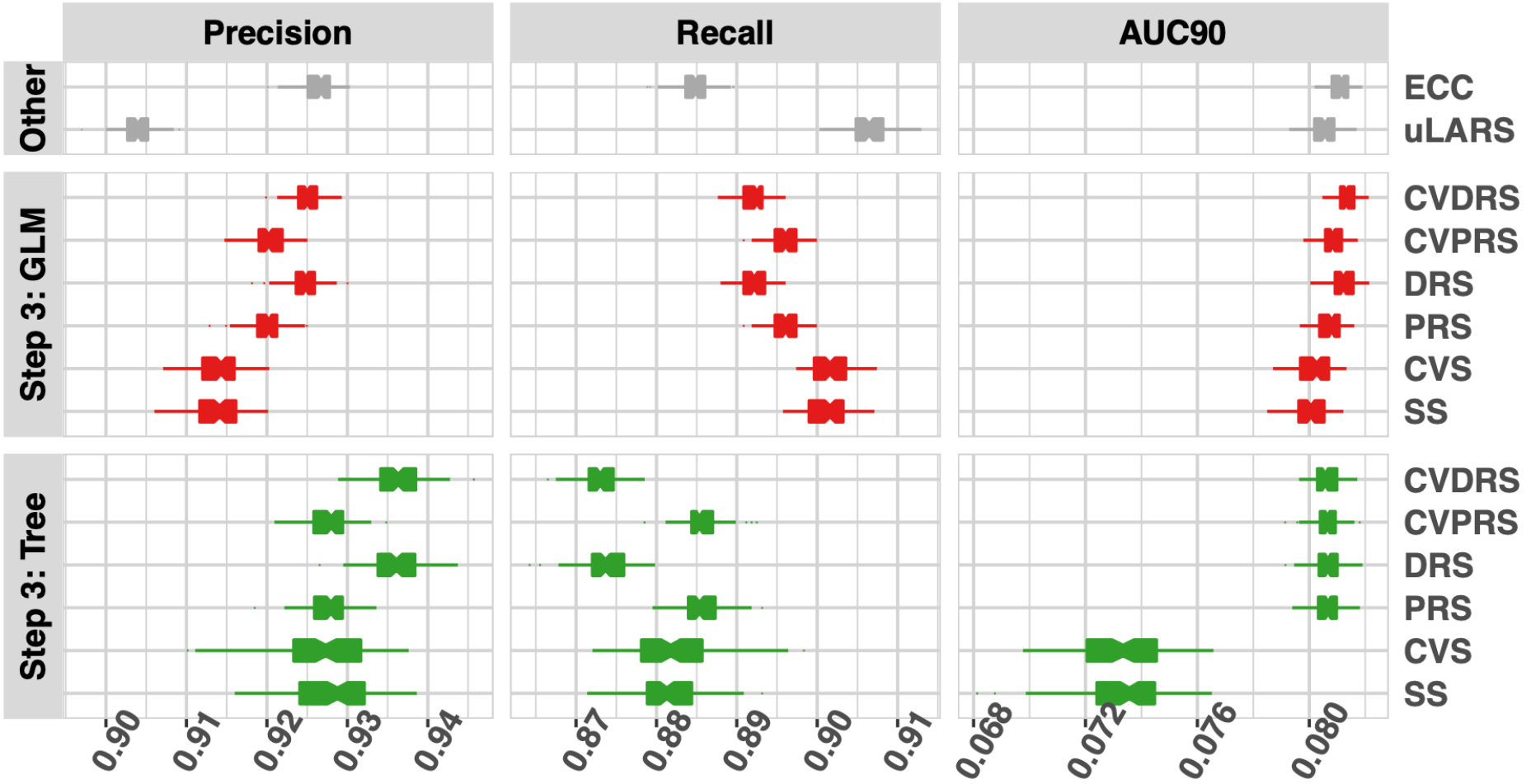


Neural network was excluded from this figure due to its bad performance

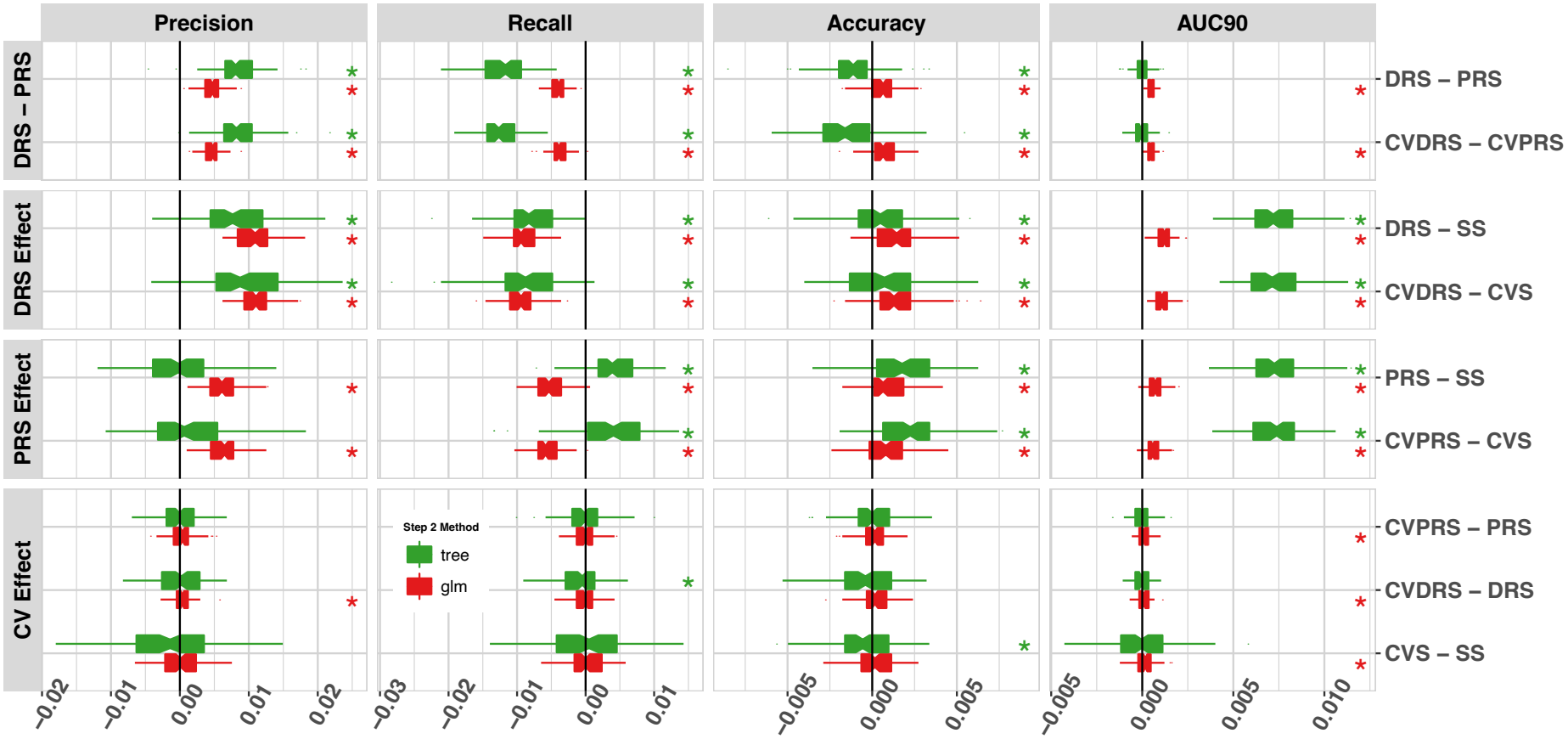
RESULT OF CROSS-VALIDATION: EVALUATION THE EFFECTS OF TWO REVISIONS IN ALGORITHMS



RESULT OF CROSS-VALIDATION: METHODS COMPARISON FOR BINARY OUTCOMES



RESULT OF CROSS-VALIDATION: EVALUATION THE EFFECTS OF TWO REVISIONS IN ALGORITHMS



RELATIONSHIP BETWEEN NEURAL NETWORK AND OUR ALGORITHM

- Our method can be considered as a network with fixed structure of 3 layers. Each layer has fixed number of elements.
  - The processor functions between 1st and 2nd layer are univariate base learners used to construct our algorithm.
  - The processor functions between 2^{nd} and 3^{rd} layer are the combiner functions of our algoirthm
- Neural network can have any number of hidden layers, the number of neurals in the hidden layer can be any number
- The processor funcitons
  - can be quite complex models in our algorithm
  - usually quite simple in Neural network
- Learning/training the processor funcitons
  - Outcome is directly used in training processors in all layers
  - Outcome indirectly affect processors in unconnected layers

RELATIONSHIP BETWEEN NEURAL NETWORK AND OUR ALGORITHM

- Neural network has simple processor functions but a more complex network structure. It can fit very flexible models with many hidden layers and neurons in each layer, if the sample size is large enough.

- Our method used a fixed and simple network structure. With well-selected processor functions, it is more suitable for smaller datasets.

CONCLUSION

- MTPS outperforms current popular methods including, uLARS, mLARS, NN and ECC.
- We recommend RS algorithms in most applications.
- If the Step 1 models are LARS and Step 2 models are linear models, CVRS is suggested for best prediction performance. But RS can be much faster with slightly inferior performance.
- For models with base learners not compared in our study, we suggest to evaluate using similar approach discussed, to ensure the best model is used to analyze the data.

CONTRIBUTION

- **Flexible** (type of outcomes)
- **Flexible** (step 1 and step 2 methods)
- Very suitable for high dimensional data with moderate sample sizes
- Use much less time in model development and implication cycle than other methods such as Bayesian Hierarchical Model

FUTURE WORK

The current method can handle mixed outcomes of binary and continuous

We extend it to handle longitudinal and time-to-event outcomes

- available soon