# Super Learning (SL) and sl3

Rachael Victoria Phillips
rachaelvphillips@berkeley.edu

May 23, 2022
ACIC 2022 Workshop

Berkeley
UNIVERSITY OF CALIFORNIA

# Overview of Super Learner

# Super Learner (SL)

## Library

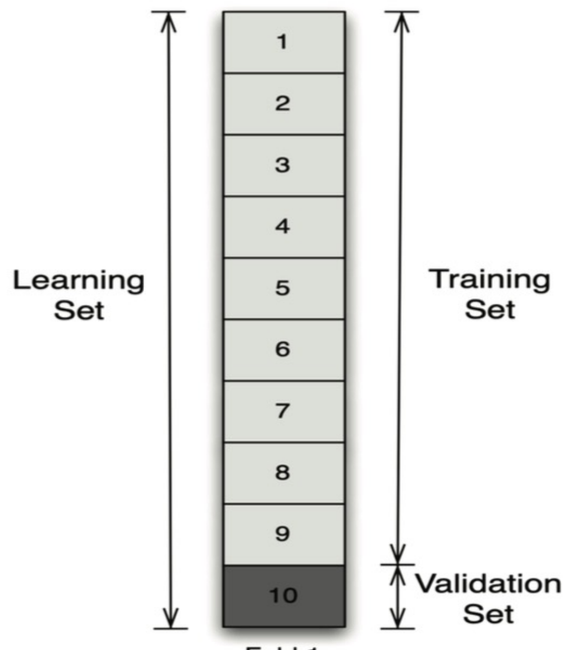
Linear model
BART
Random Forest
Neural Network
Lasso
HAL
Regression splines

# Super Learner (SL)

**LIBRARY** | **COMPETITION**

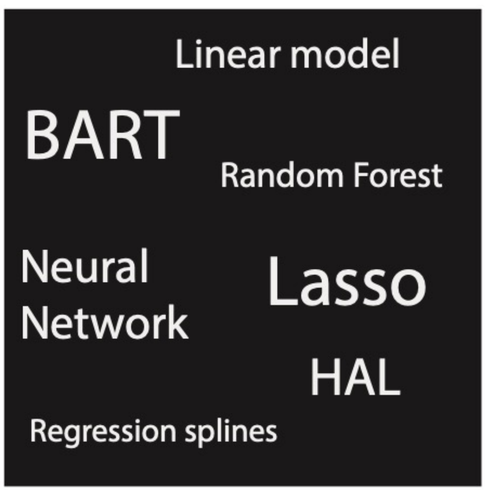
Cross-validated
performance of
learners + ensembles

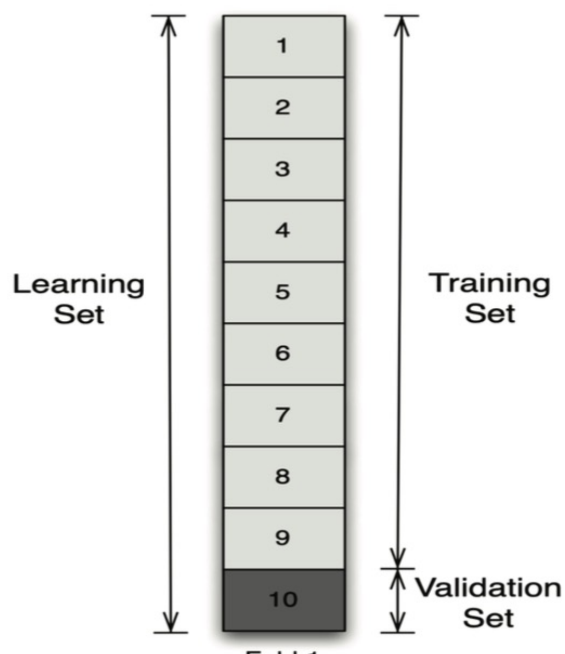
Linear model
BART
Random Forest
Neural
Network
Lasso
HAL
Regression splines

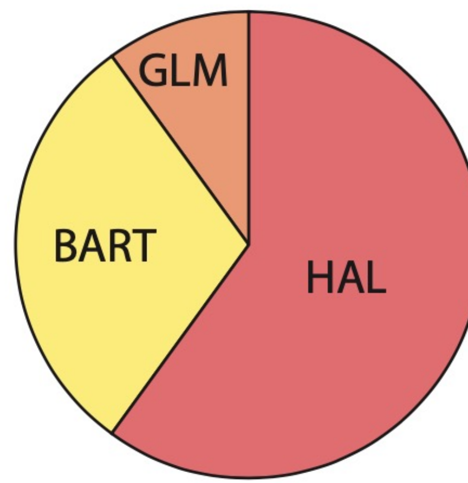


# Super Learner (SL)


Cross-validated
performance of
learners + ensembles



# Super Learner (SL)

- **Uses a library of algorithms for estimating a prediction function**
  - Analyst specifies $Alg_1$, … $Alg_K$
  - Create an optimal combination
    - Optimal with respect to V-fold cross-validated (CV) risk
    - Example risk functions: Negative log likelihood, mean squared error, 1-AUC
- **SL predicted values, $\hat{Y}_{SL}$, are a combination of $\hat{Y}_{Alg_1}$, …, $\hat{Y}_{Alg_K}$**
  - Discrete SL: "winner-take-all", predictions from algorithm with best CV risk
  - Ensemble SL: predictions from multiple algorithms are combined
    - weighted combination
    - some other, possibly complex function of the algorithms' predictions

Berkeley
UNIVERSITY OF CALIFORNIA

# Defining estimation problem

# Defining estimation problem

- A performance metric quantifies the success of an estimated prediction function (i.e., a trained algorithm)

  - Expectation of the squared error loss / MSE
  - The area under the ROC curve, AUC

# Defining estimation problem

- A performance metric quantifies the success of an estimated prediction function (i.e., a trained algorithm)

- The chosen metric needs to be optimized (minimized or maximized) by the true prediction function
  - **This guarantees that the evaluation corresponds to the trained algorithm's success in approximating the true prediction function**

# Defining estimation problem

- A performance metric quantifies the success of an estimated prediction function (i.e., a trained algorithm)
- The chosen metric needs to be optimized (minimized or maximized) by the true prediction function
  - **This guarantees that the evaluation corresponds to the trained algorithm's success in approximating the true prediction function**
- The chosen metric should align with the intended real-world use of the predictions

# Why super learner (SL)?

- Alleviates concerns over selecting the one “right” algorithm while benefiting from considering a diverse set
- Grounded in optimality theory that guarantees for large sample sizes the SL will perform as well as possible, given the algorithms in the library
- Pre-specified and data-adaptive
- Conveying knowledge about DGP through the library can mitigate statistical model misspecification

# Super learner

1. Specify
    a) Measure of performance
    b) Cross-validation scheme
    c) Diverse library of candidate learners

# Super learner

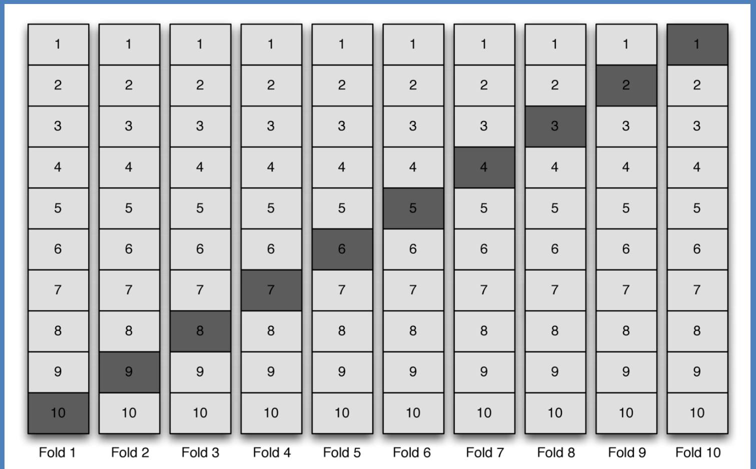
What are you learning from the data?
What do you want to optimize for?

Performance measure should be valid (i.e., optimized by underlying target), bounded, corresponds to desired goal

1. Specify
   a) Measure of performance
   b) Cross-validation scheme
   c) Diverse library of candidate learners

Berkeley
UNIVERSITY OF CALIFORNIA

# Super learner




1. Specify
   a) Measure of performance
   b) Cross-validation scheme
   c) Diverse library of candidate learners

# Super learner

1. Specify
   a) Measure of performance
   b) Cross-validation scheme
   c) Diverse library of candidate learners

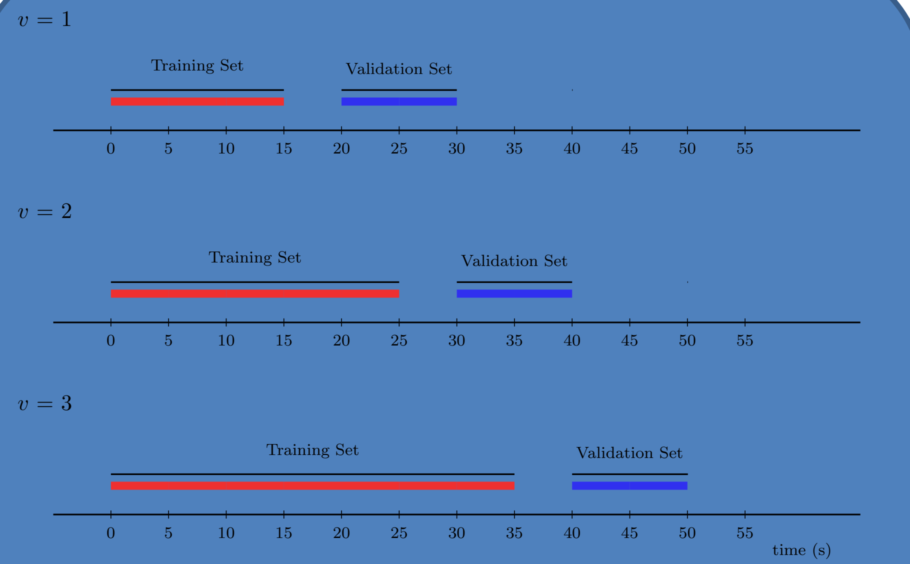
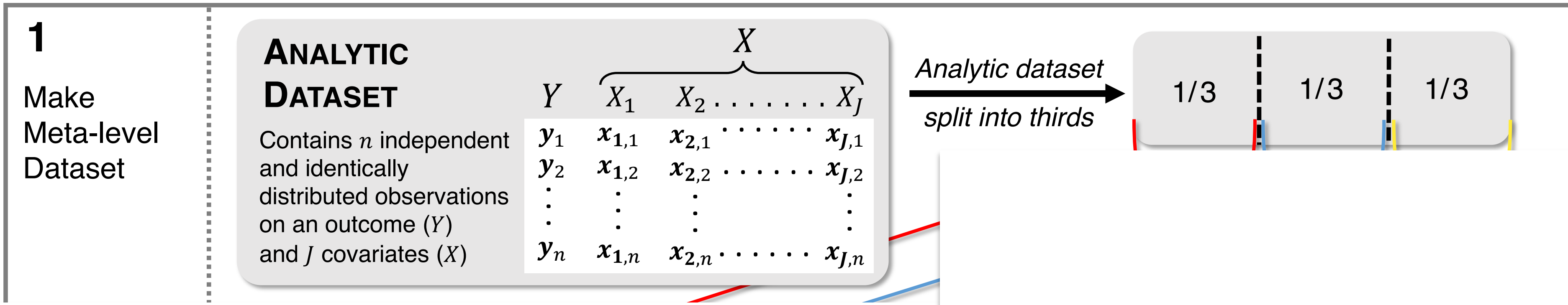


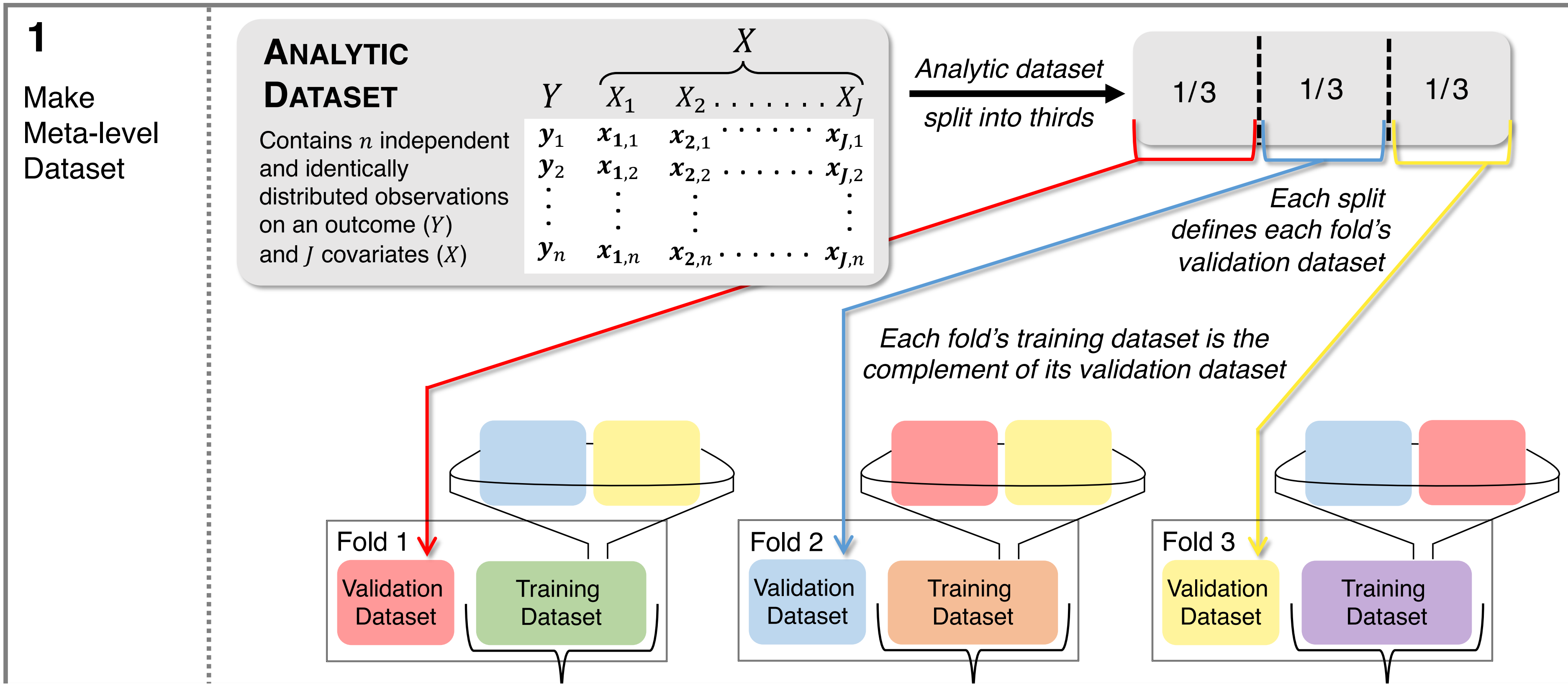

Figure 1: Rolling origin cross-validation scheme for $V = 3$ folds with first window size $n^0_{t,v_1} = 15$, validation size $n^1_{t,v} = 10$, batch size $m = 10$ and gap $h = 5$ for sample $i$.

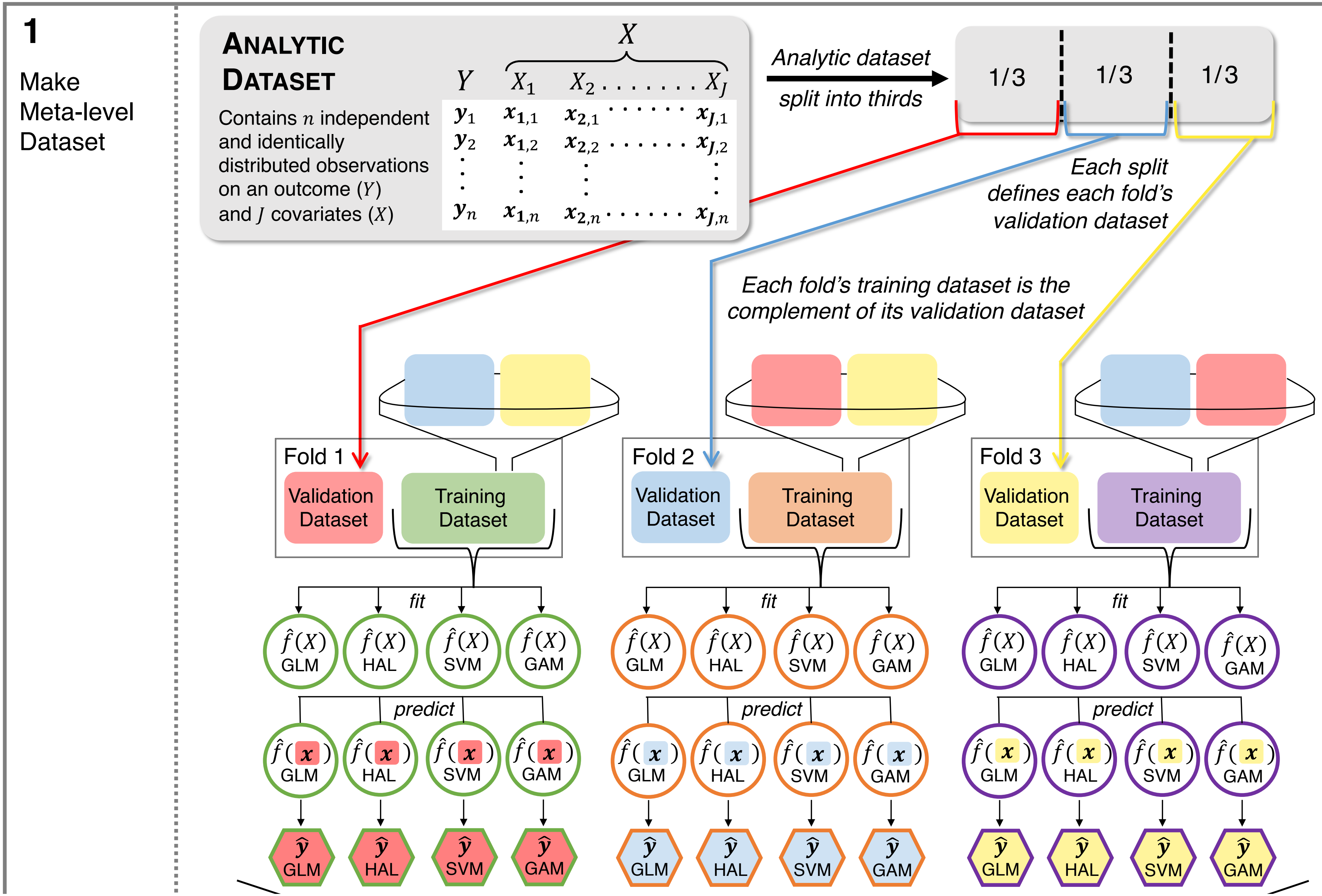
Berkeley
UNIVERSITY OF CALIFORNIA

# Overview of the super learner

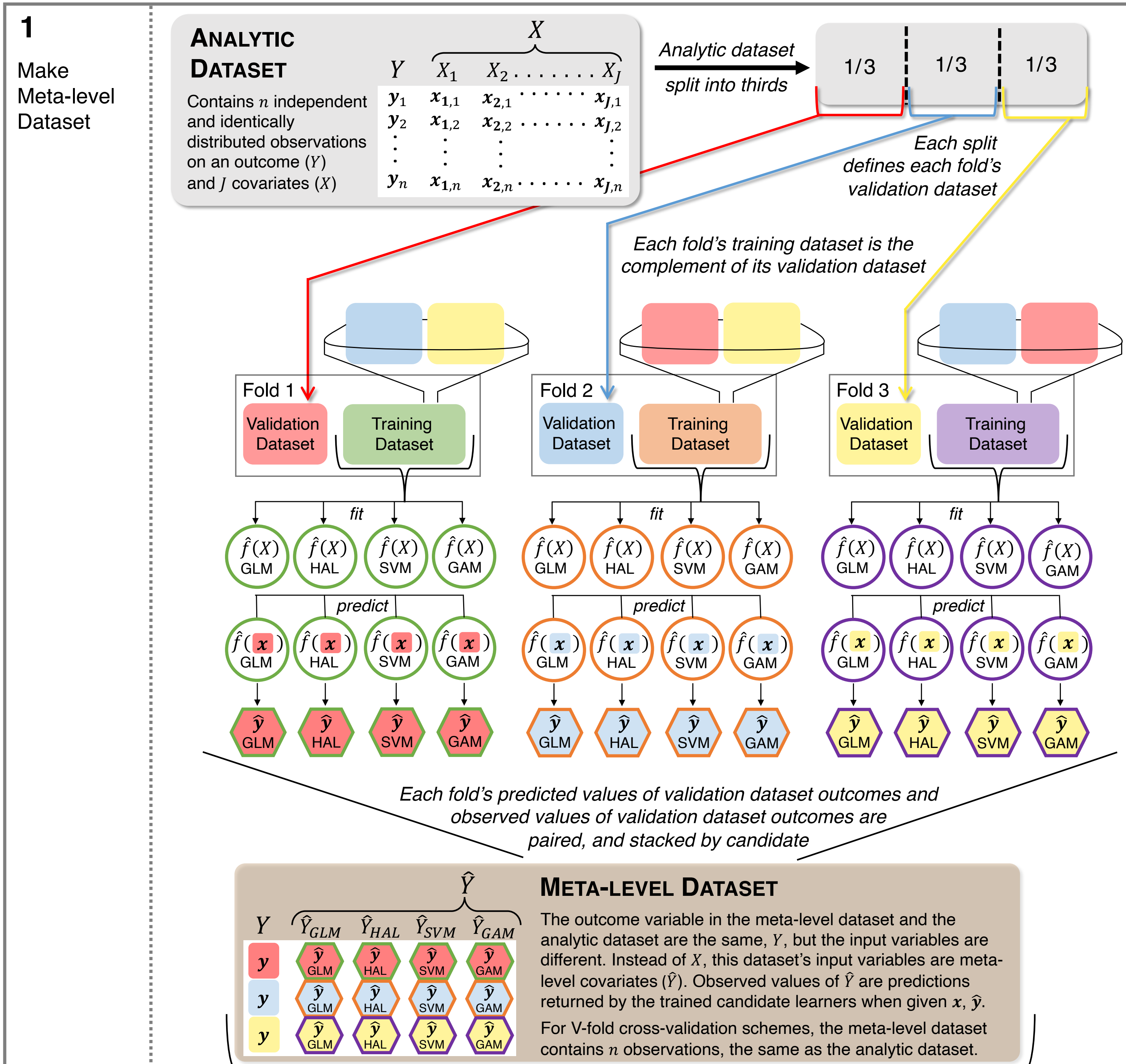
1. Make metalevel dataset with cross-validated candidate predictions and validation set outcomes
2. Fit meta-learner to the metalevel dataset
3. Full-fit candidates
4. Define the SL


1
Make
Meta-level
Dataset
ANALYTIC
DATASET
Contains $n$ independent
and identically
distributed observations
on an outcome $(Y)$
and $J$ covariates $(X)$
$X$
$Y$ $X_1$ $X_2$ ........ $X_J$
$y_1$ $x_{1,1}$ $x_{2,1}$ ...... $x_{J,1}$
$y_2$ $x_{1,2}$ $x_{2,2}$ ...... $x_{J,2}$
$y_n$ $x_{1,n}$ $x_{2,n}$ ...... $x_{J,n}$
Analytic dataset
split into thirds
1/3
1/3
1/3

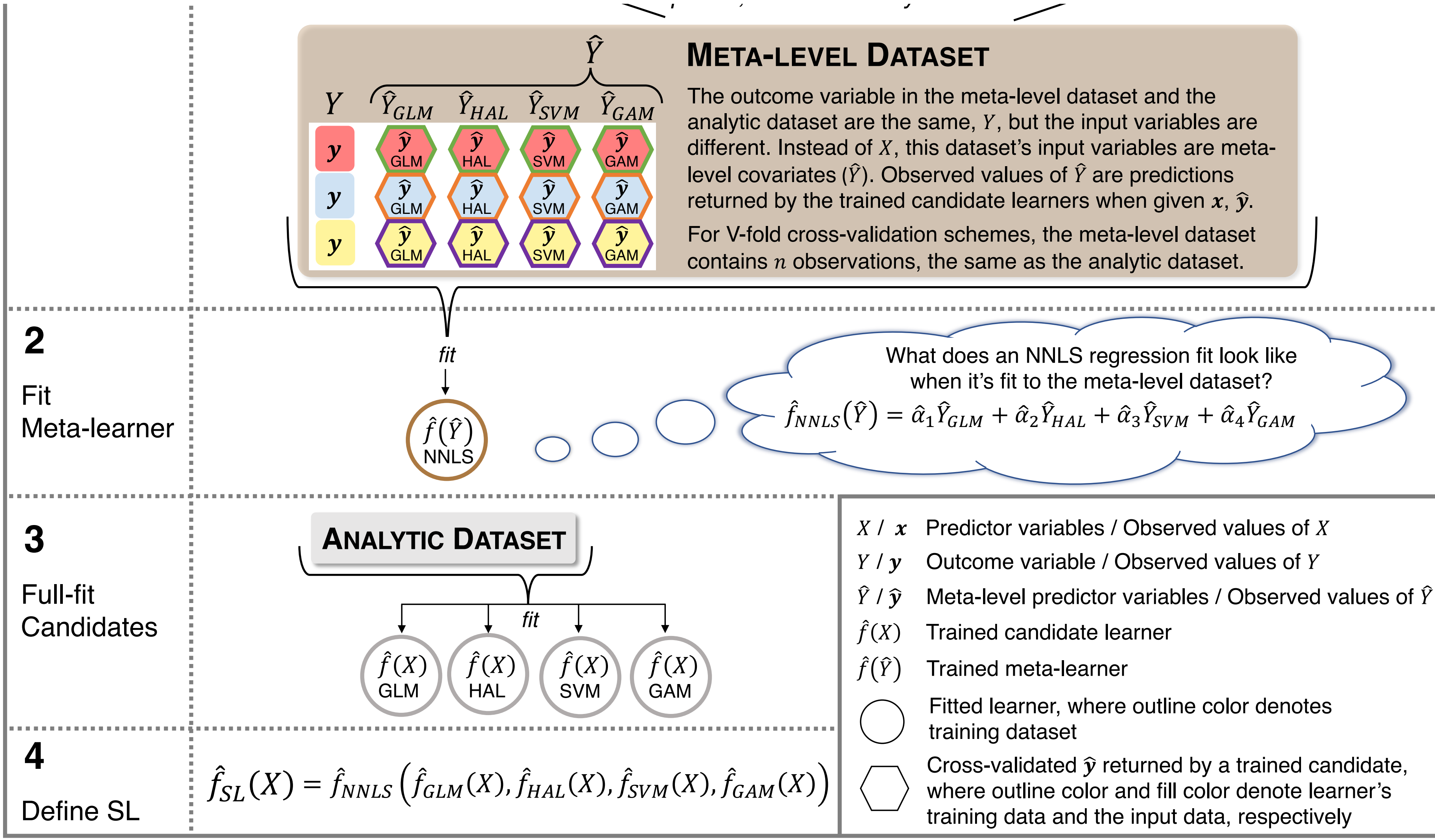

1
Make Meta-level Dataset
ANALYTIC DATASET
Contains $n$ independent and identically distributed observations on an outcome ($Y$) and $J$ covariates ($X$)
$X$
$Y$ $X_1$ $X_2$ ........ $X_J$
$y_1$ $x_{1,1}$ $x_{2,1}$ ...... $x_{J,1}$
$y_2$ $x_{1,2}$ $x_{2,2}$ ...... $x_{J,2}$
$y_n$ $x_{1,n}$ $x_{2,n}$ ...... $x_{J,n}$
Analytic dataset split into thirds
1/3
1/3
1/3
Each split defines each fold's validation dataset
Each fold's training dataset is the complement of its validation dataset
Fold 1
Validation Dataset
Training Dataset
Fold 2
Validation Dataset
Training Dataset
Fold 3
Validation Dataset
Training Dataset



1
Make Meta-level Dataset
ANALYTIC DATASET
Contains $n$ independent and identically distributed observations on an outcome ($Y$) and $J$ covariates ($X$)
$Y$ $X$ $X_1$ $X_2$ $X_J$
Analytic dataset split into thirds
1/3 1/3 1/3
Each split defines each fold's validation dataset
Each fold's training dataset is the complement of its validation dataset
Fold 1
Validation Dataset
Training Dataset
Fold 2
Validation Dataset
Training Dataset
Fold 3
Validation Dataset
Training Dataset
fit
$\hat{f}(X)$ GLM
$\hat{f}(X)$ HAL
$\hat{f}(X)$ SVM
$\hat{f}(X)$ GAM
predict
$\hat{f}(x)$ GLM
$\hat{f}(x)$ HAL
$\hat{f}(x)$ SVM
$\hat{f}(x)$ GAM
$\hat{y}$ GLM
$\hat{y}$ HAL
$\hat{y}$ SVM
$\hat{y}$ GAM



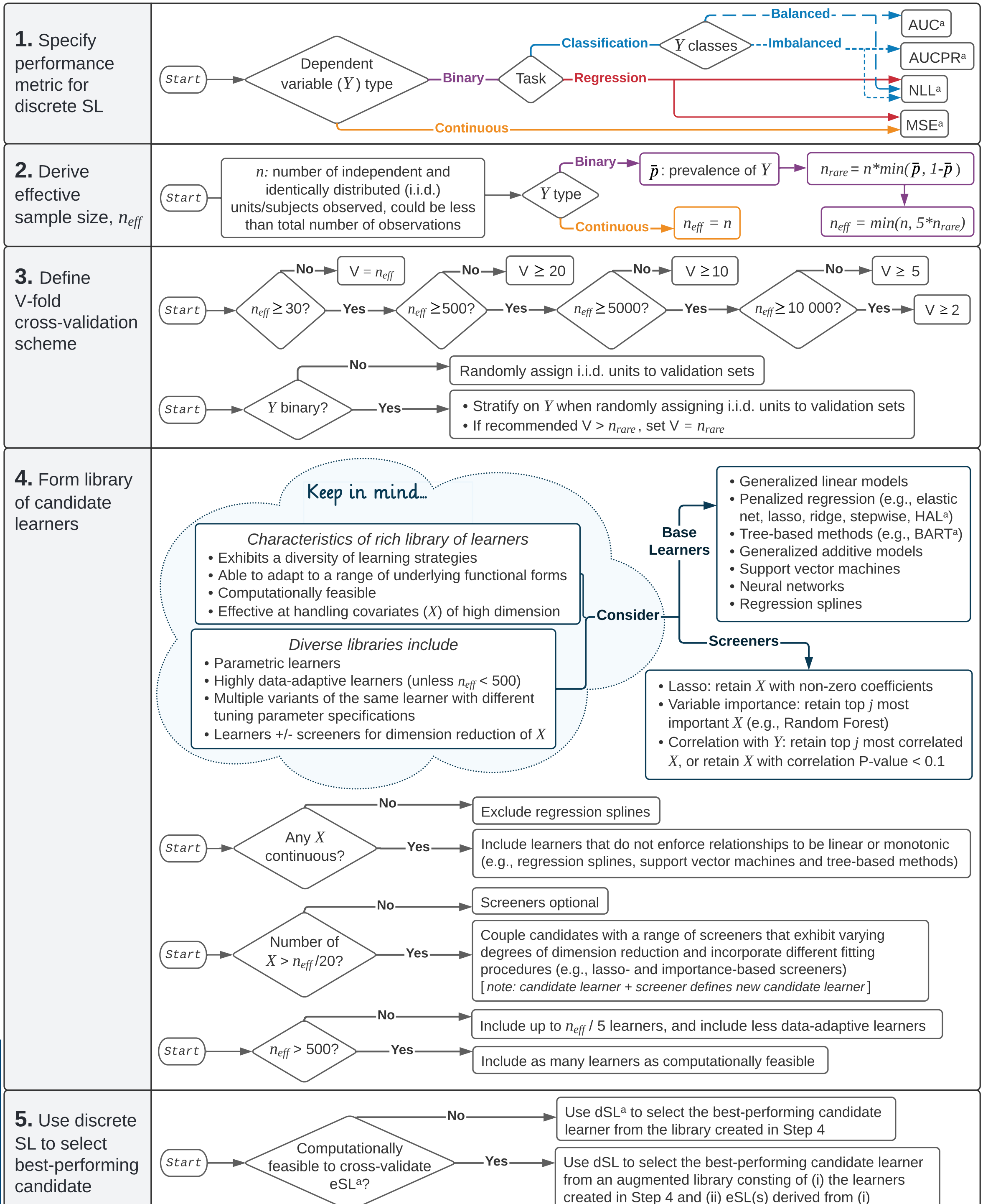
1
Make Meta-level Dataset
ANALYTIC DATASET
Contains $n$ independent and identically distributed observations on an outcome ($Y$) and $J$ covariates ($X$)
Analytic dataset split into thirds
1/3
1/3
1/3
Each split defines each fold's validation dataset
Each fold's training dataset is the complement of its validation dataset
Fold 1
Validation Dataset
Training Dataset
Fold 2
Validation Dataset
Training Dataset
Fold 3
Validation Dataset
Training Dataset
fit
predict
GLM
HAL
SVM
GAM
Each fold's predicted values of validation dataset outcomes and observed values of validation dataset outcomes are paired, and stacked by candidate
META-LEVEL DATASET
The outcome variable in the meta-level dataset and the analytic dataset are the same, $Y$, but the input variables are different. Instead of $X$, this dataset's input variables are meta-level covariates ($\hat{Y}$). Observed values of $\hat{Y}$ are predictions returned by the trained candidate learners when given $\boldsymbol{x}$, $\hat{\boldsymbol{y}}$.
For V-fold cross-validation schemes, the meta-level dataset contains $n$ observations, the same as the analytic dataset.


META-LEVEL DATASET
$\hat{Y}$
$Y$
$\hat{Y}_{GLM}$ $\hat{Y}_{HAL}$ $\hat{Y}_{SVM}$ $\hat{Y}_{GAM}$
$\boldsymbol{y}$
$\hat{\boldsymbol{y}}$ GLM
$\hat{\boldsymbol{y}}$ HAL
$\hat{\boldsymbol{y}}$ SVM
$\hat{\boldsymbol{y}}$ GAM
The outcome variable in the meta-level dataset and the analytic dataset are the same, $Y$, but the input variables are different. Instead of $X$, this dataset's input variables are meta-level covariates ($\hat{Y}$). Observed values of $\hat{Y}$ are predictions returned by the trained candidate learners when given $\boldsymbol{x}$, $\hat{\boldsymbol{y}}$.
For V-fold cross-validation schemes, the meta-level dataset contains $n$ observations, the same as the analytic dataset.
2
Fit
Meta-learner
fit
$\hat{f}(\hat{Y})$
NNLS
What does an NNLS regression fit look like when it's fit to the meta-level dataset?
$\hat{f}_{NNLS}(\hat{Y}) = \hat{\alpha}_1\hat{Y}_{GLM} + \hat{\alpha}_2\hat{Y}_{HAL} + \hat{\alpha}_3\hat{Y}_{SVM} + \hat{\alpha}_4\hat{Y}_{GAM}$
3
Full-fit
Candidates
ANALYTIC DATASET
fit
$\hat{f}(X)$ GLM
$\hat{f}(X)$ HAL
$\hat{f}(X)$ SVM
$\hat{f}(X)$ GAM
4
Define SL
$\hat{f}_{SL}(X) = \hat{f}_{NNLS}\left(\hat{f}_{GLM}(X), \hat{f}_{HAL}(X), \hat{f}_{SVM}(X), \hat{f}_{GAM}(X)\right)$
$X$ / $\boldsymbol{x}$ Predictor variables / Observed values of $X$
$Y$ / $\boldsymbol{y}$ Outcome variable / Observed values of $Y$
$\hat{Y}$ / $\hat{\boldsymbol{y}}$ Meta-level predictor variables / Observed values of $\hat{Y}$
$\hat{f}(X)$ Trained candidate learner
$\hat{f}(\hat{Y})$ Trained meta-learner
Fitted learner, where outline color denotes training dataset
Cross-validated $\hat{\boldsymbol{y}}$ returned by a trained candidate, where outline color and fill color denote learner's training data and the input data, respectively

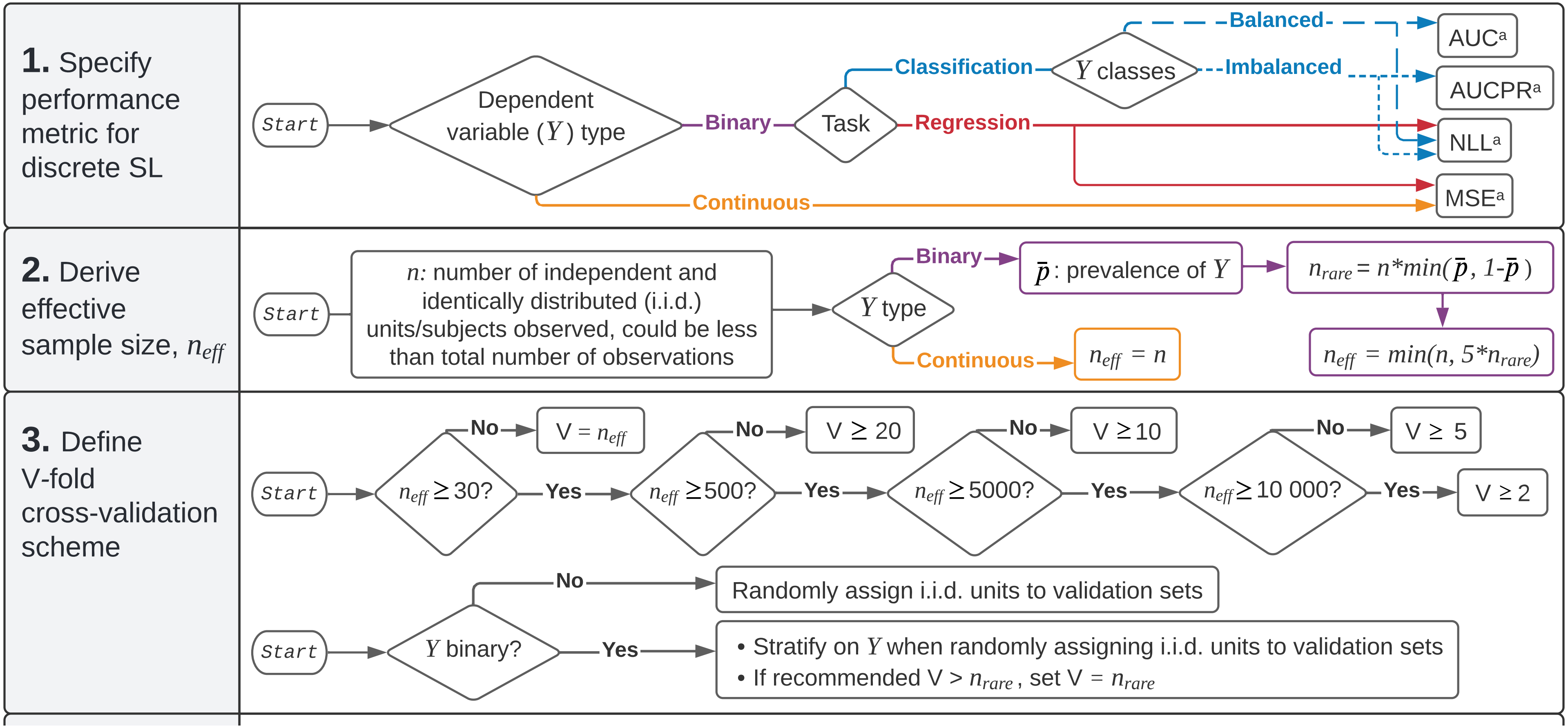
# Practical considerations for specifying SL

Phillips RV, van der Laan MJ, Lee H, Gruber S. Practical considerations for specifying a super learner. arXiv preprint arXiv:2204.06139. 2022.

https://arxiv.org/abs/2204.06139

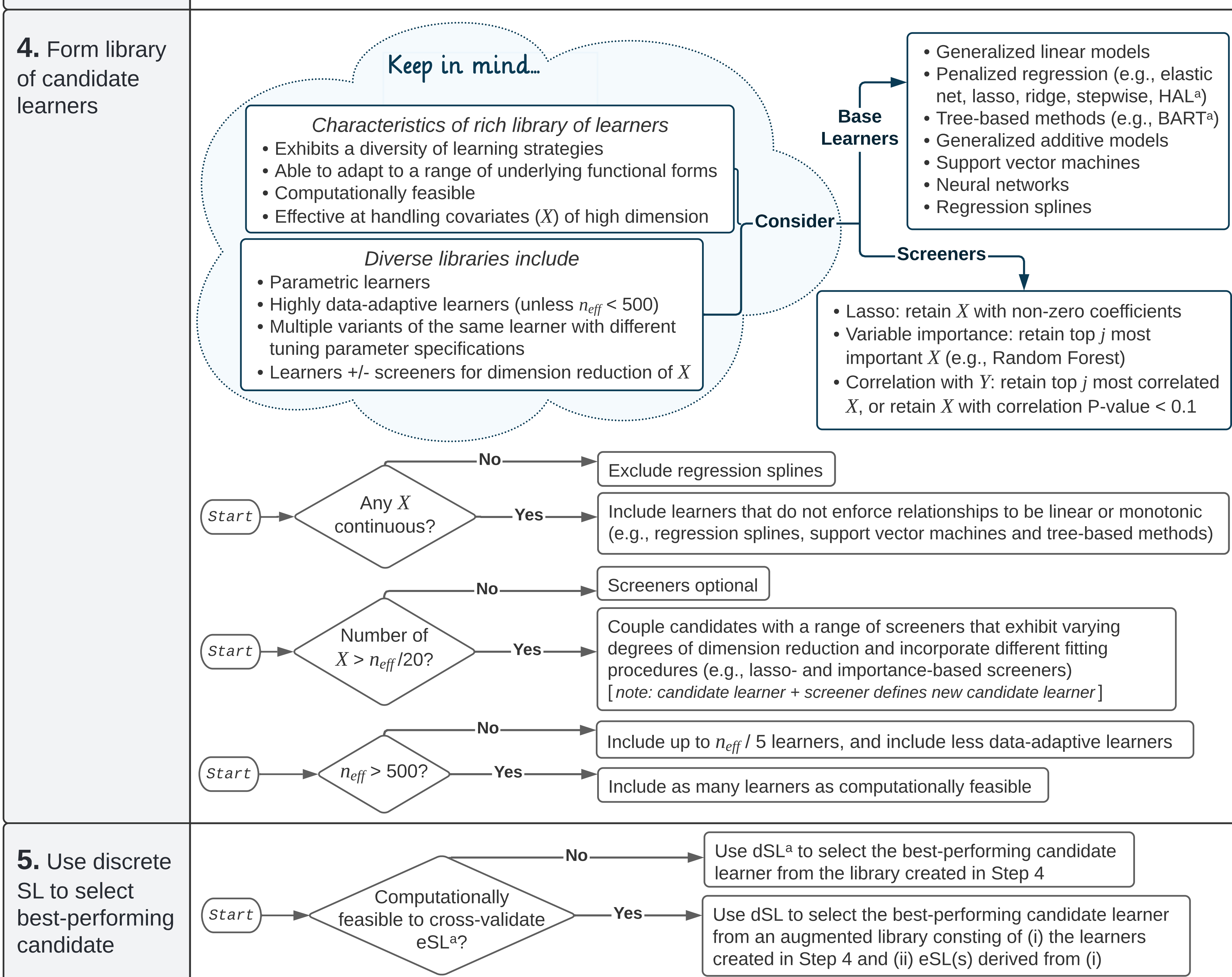



Berkeley
UNIVERSITY OF CALIFORNIA




Berkeley
UNIVERSITY OF CALIFORNIA

Phillips RV, van der Laan MJ, Lee H, Gruber S.
Practical considerations for specifying a super learner.
arXiv preprint arXiv:2204.06139. 2022.



4. Form library of candidate learners
Keep in mind…
Characteristics of rich library of learners
• Exhibits a diversity of learning strategies
• Able to adapt to a range of underlying functional forms
• Computationally feasible
• Effective at handling covariates ($X$) of high dimension
Diverse libraries include
• Parametric learners
• Highly data-adaptive learners (unless $n_{eff}$ < 500)
• Multiple variants of the same learner with different tuning parameter specifications
• Learners +/- screeners for dimension reduction of $X$
Consider
Base Learners
• Generalized linear models
• Penalized regression (e.g., elastic net, lasso, ridge, stepwise, HAL[a])
• Tree-based methods (e.g., BART[a])
• Generalized additive models
• Support vector machines
• Neural networks
• Regression splines
Screeners
• Lasso: retain $X$ with non-zero coefficients
• Variable importance: retain top $j$ most important $X$ (e.g., Random Forest)
• Correlation with $Y$: retain top $j$ most correlated $X$, or retain $X$ with correlation P-value < 0.1
Start
Any $X$ continuous?
No
Exclude regression splines
Yes
Include learners that do not enforce relationships to be linear or monotonic (e.g., regression splines, support vector machines and tree-based methods)
Start
Number of $X > n_{eff}/20$?
No
Screeners optional
Yes
Couple candidates with a range of screeners that exhibit varying degrees of dimension reduction and incorporate different fitting procedures (e.g., lasso- and importance-based screeners)
[note: candidate learner + screener defines new candidate learner]
Start
$n_{eff} > 500$?
No
Include up to $n_{eff}$ / 5 learners, and include less data-adaptive learners
Yes
Include as many learners as computationally feasible
5. Use discrete SL to select best-performing candidate
Start
Computationally feasible to cross-validate eSL[a]?
No
Use dSL[a] to select the best-performing candidate learner from the library created in Step 4
Yes
Use dSL to select the best-performing candidate learner from an augmented library consting of (i) the learners created in Step 4 and (ii) eSL(s) derived from (i)

| Term | Definition |
|---|---|
| **Algorithm, learner, machine learning algorithm** | A set of instructions that define a prediction function estimator when tuning parameters are specified. Estimating the prediction function (i.e., algorithm training/fitting) is an optimization problem; in learning the function of the input variables, the algorithm aims to optimize some performance metric / risk function (e.g., minimize the mean squared error). |

| Term | Definition |
| --- | --- |
| **Algorithm, learner, machine learning algorithm** | A set of instructions that define a prediction function estimator when tuning parameters are specified. Estimating the prediction function (i.e., algorithm training/fitting) is an optimization problem; in learning the function of the input variables, the algorithm aims to optimize some performance metric / risk function (e.g., minimize the mean squared error). |
| **Base algorithm, base learner** | An algorithm that is not fully specified but defines a particular learning strategy (e.g., random forest). A base learner is used as a building block to define one or more fully specified learners, i.e., one or more estimators of the true prediction function. |
| **Library** | The set of specified algorithms that will be considered by the super learner. |
| **Candidate, candidate learner** | A specified algorithm included in the super learner library, with values provided for all tuning parameters, optionally coupled with a screening algorithm. Candidates are trained to consider $X$ as input variables. |

| Term | Definition |
|---|---|
| **Algorithm, learner, machine learning algorithm** | A set of instructions that define a prediction function estimator when tuning parameters are specified. Estimating the prediction function (i.e., algorithm training/fitting) is an optimization problem; in learning the function of the input variables, the algorithm aims to optimize some performance metric / risk function (e.g., minimize the mean squared error). |
| **Base algorithm, base learner** | An algorithm that is not fully specified but defines a particular learning strategy (e.g., random forest). A base learner is used as a building block to define one or more fully specified learners, i.e., one or more estimators of the true prediction function. |
| **Library** | The set of specified algorithms that will be considered by the super learner. |
| **Candidate, candidate learner** | A specified algorithm included in the super learner library, with values provided for all tuning parameters, optionally coupled with a screening algorithm. Candidates are trained to consider $X$ as input variables. |
| **Screener, screening algorithm** | A function that returns a subset of $X$.<br>A screener can be coupled with a candidate learner to define a new candidate learner that considers the reduced set of screener-returned $X$ as its covariates. |

Berkeley
UNIVERSITY OF CALIFORNIA

| Term | Definition |
|---|---|
| **Meta-learner, meta-learning algorithm** | A specified algorithm that is trained to consider $\hat{Y}$ as input variables. Hence, the “meta” nature of the meta-learner: it learns from what is learned by the candidate learners (see Figure 2). |
| **Super learner (SL)** | Just like any other algorithm, the SL is a prediction function estimator. The fitted SL’s input variables are $X$. The SL’s estimated prediction function is special in that it has two layers: the inner layer is the set of prediction functions learned by the candidates, and the outer layer is the prediction function learned by the meta-learner (see Figure 2). |

| Term | Definition |
| --- | --- |
| **Meta-learner, meta-learning algorithm** | A specified algorithm that is trained to consider $\hat{Y}$ as input variables. Hence, the "meta" nature of the meta-learner: it learns from what is learned by the candidate learners (see Figure 2). |
| **Super learner (SL)** | Just like any other algorithm, the SL is a prediction function estimator. The fitted SL's input variables are $X$. The SL's estimated prediction function is special in that it has two layers: the inner layer is the set of prediction functions learned by the candidates, and the outer layer is the prediction function learned by the meta-learner (see Figure 2). |
| **Discrete SL (dSL)** | A SL that uses a winner-take-all meta-learner called the cross-validated selector. The dSL is therefore identical to the candidate with the best cross-validated performance; its predictions will be the same as this candidate's predictions. |
| **Ensemble SL (eSL)** | A SL that uses any parametric or non-parametric algorithm as its meta-learner. Therefore, the eSL is defined by a combination of multiple candidates; its predictions are defined by a combination of multiple candidates' predictions. (Note that the dSL can be thought of as a highly constrained or superficial type of eSL, in which dSL predictions are a weighted combination of the candidates' predictions, with predictions from the candidate with the best cross-validated performance given weight one and those from all other candidates given weight zero.) |

# `sl3`

## SL software package in `tlverse`

Berkeley
UNIVERSITY OF CALIFORNIA

# Introductory overview of `sl3`

- Task
- Learners
- Functions

# Introductory overview of `sl3`

- **Task**
- Learners
- Functions

# Tasks in `sl3`

***What is the prediction task?***

data, covariates, outcome,

weights, id, outcome_type,offset,
drop_missing_outcome, folds

https://tlverse.org/sl3/reference/sl3_Task.html

Berkeley
UNIVERSITY OF CALIFORNIA

# Introductory overview of `sl3`

- Task
- **Learners**
- Functions

# Learners in `sl3`

***How to estimate prediction function?***



https://tlverse.org/sl3/reference/index.html#sl-learners

# Learners in `sl3`

## *How to estimate prediction function?*

Other Learners: `Lrnr_HarmonicReg`, `Lrnr_arima`, `Lrnr_bartMachine`, `Lrnr_base`, `Lrnr_bayesglm`, `Lrnr_bilstm`, `Lrnr_caret`, `Lrnr_cv_selector`, `Lrnr_cv`, `Lrnr_dbarts`, `Lrnr_define_interactions`, `Lrnr_density_discretize`, `Lrnr_density_hse`, `Lrnr_density_semiparametric`, `Lrnr_earth`, `Lrnr_expSmooth`, `Lrnr_gam`, `Lrnr_ga`, `Lrnr_gbm`, `Lrnr_glm_fast`, `Lrnr_glmnet`, `Lrnr_glm`, `Lrnr_grf`, `Lrnr_gru_keras`, `Lrnr_gts`, `Lrnr_h2o_grid`, `Lrnr_hal9001`, `Lrnr_haldensify`, `Lrnr_hts`, `Lrnr_independent_binomial`, `Lrnr_lightgbm`, `Lrnr_lstm_keras`, `Lrnr_mean`, `Lrnr_multiple_ts`, `Lrnr_multivariate`, `Lrnr_nnet`, `Lrnr_nnls`, `Lrnr_optim`, `Lrnr_pca`, `Lrnr_pkg_SuperLearner`, `Lrnr_polspline`, `Lrnr_pooled_hazards`, `Lrnr_randomForest`, `Lrnr_ranger`, `Lrnr_revere_task`, `Lrnr_rpart`, `Lrnr_rugarch`, `Lrnr_screener_augment`, `Lrnr_screener_coefs`, `Lrnr_screener_correlation`, `Lrnr_screener_importance`, `Lrnr_sl`, `Lrnr_solnp_density`, `Lrnr_solnp`, `Lrnr_stratified`, `Lrnr_subset_covariates`, `Lrnr_svm`, `Lrnr_tsDyn`, `Lrnr_ts_weights`, `Lrnr_xgboost`, `Pipeline`, `Stack`, `define_h2o_X()`,

Berkeley
UNIVERSITY OF CALIFORNIA

https://tlverse.org/sl3/reference/index.html#sl-learners

# Learners in `sl3`

***How to estimate prediction function?***

Other Learners: `Lrnr_HarmonicReg`, `Lrnr_arima`, `Lrnr_bartMachine`, `Lrnr_base`, `Lrnr_bayesglm`, `Lrnr_bilstm`, `Lrnr_caret`, `Lrnr_cv_selector`, `Lrnr_cv`, `Lrnr_dbarts`, `Lrnr_define_interactions`, `Lrnr_density_discretize`, `Lrnr_density_hse`, `Lrnr_density_semiparametric`, `Lrnr_earth`, `Lrnr_expSmooth`, `Lrnr_gam`, `Lrnr_ga`, `Lrnr_gbm`, `Lrnr_glm_fast`, `Lrnr_glmnet`, `Lrnr_glm`, `Lrnr_grf`, `Lrnr_gru_keras`, `Lrnr_gts`, `Lrnr_h2o_grid`, `Lrnr_hal9001`, `Lrnr_haldensify`, `Lrnr_hts`, `Lrnr_independent_binomial`, `Lrnr_lightgbm`, `Lrnr_lstm_keras`, `Lrnr_mean`, `Lrnr_multiple_ts`, `Lrnr_multivariate`, `Lrnr_nnet`, `Lrnr_nnls`, `Lrnr_optim`, `Lrnr_pca`, `Lrnr_pkg_SuperLearner`, `Lrnr_polspline`, `Lrnr_pooled_hazards`, `Lrnr_randomForest`, `Lrnr_ranger`, `Lrnr_revere_task`, `Lrnr_rpart`, `Lrnr_rugarch`, `Lrnr_screener_augment`, `Lrnr_screener_coefs`, `Lrnr_screener_correlation`, `Lrnr_screener_importance`, `Lrnr_sl`, `Lrnr_solnp_density`, `Lrnr_solnp`, `Lrnr_stratified`, `Lrnr_subset_covariates`, `Lrnr_svm`, `Lrnr_tsDyn`, `Lrnr_ts_weights`, `Lrnr_xgboost`, `Pipeline`, `Stack`, `define_h2o_X()`,

Berkeley
UNIVERSITY OF CALIFORNIA

https://tlverse.org/sl3/reference/index.html#sl-learners

# Learners in `sl3`

## *How to estimate prediction function?*

Other Learners: `Lrnr_HarmonicReg`, `Lrnr_arima`, `Lrnr_bartMachine`, `Lrnr_base`, `Lrnr_bayesglm`, `Lrnr_bilstm`, `Lrnr_caret`, `Lrnr_cv_selector`, `Lrnr_cv`, `Lrnr_dbarts`, `Lrnr_define_interactions`, `Lrnr_density_discretize`, `Lrnr_density_hse`, `Lrnr_density_semiparametric`, `Lrnr_earth`, `Lrnr_expSmooth`, `Lrnr_gam`, `Lrnr_ga`, `Lrnr_gbm`, `Lrnr_glm_fast`, `Lrnr_glmnet`, `Lrnr_glm`, `Lrnr_grf`, `Lrnr_gru_keras`, `Lrnr_gts`, `Lrnr_h2o_grid`, `Lrnr_hal9001`, `Lrnr_haldensify`, `Lrnr_hts`, `Lrnr_independent_binomial`, `Lrnr_lightgbm`, `Lrnr_lstm_keras`, `Lrnr_mean`, `Lrnr_multiple_ts`, `Lrnr_multivariate`, `Lrnr_nnet`, `Lrnr_nnls`, `Lrnr_optim`, `Lrnr_pca`, `Lrnr_pkg_SuperLearner`, `Lrnr_polspline`, `Lrnr_pooled_hazards`, `Lrnr_randomForest`, `Lrnr_ranger`, `Lrnr_revere_task`, `Lrnr_rpart`, `Lrnr_rugarch`, `Lrnr_screener_augment`, `Lrnr_screener_coefs`, `Lrnr_screener_correlation`, `Lrnr_screener_importance`, `Lrnr_sl`, `Lrnr_solnp_density`, `Lrnr_solnp`, `Lrnr_stratified`, `Lrnr_subset_covariates`, `Lrnr_svm`, `Lrnr_tsDyn`, `Lrnr_ts_weights`, `Lrnr_xgboost`, `Pipeline`, `Stack`, `define_h2o_X()`,

Berkeley
UNIVERSITY OF CALIFORNIA

https://tlverse.org/sl3/reference/index.html#sl-learners

# Learners in `sl3`

## *How to estimate prediction function?*

Other Learners: `Lrnr_HarmonicReg`, `Lrnr_arima`, `Lrnr_bartMachine`, `Lrnr_base`, `Lrnr_bayesglm`, `Lrnr_bilstm`, `Lrnr_caret`, `Lrnr_cv_selector`, `Lrnr_cv`, `Lrnr_dbarts`, `Lrnr_define_interactions`, `Lrnr_density_discretize`, `Lrnr_density_hse`, `Lrnr_density_semiparametric`, `Lrnr_earth`, `Lrnr_expSmooth`, `Lrnr_gam`, `Lrnr_ga`, `Lrnr_gbm`, `Lrnr_glm_fast`, `Lrnr_glmnet`, `Lrnr_glm`, `Lrnr_grf`, `Lrnr_gru_keras`, `Lrnr_gts`, `Lrnr_h2o_grid`, `Lrnr_hal9001`, `Lrnr_haldensify`, `Lrnr_hts`, `Lrnr_independent_binomial`, `Lrnr_lightgbm`, `Lrnr_lstm_keras`, `Lrnr_mean`, `Lrnr_multiple_ts`, `Lrnr_multivariate`, `Lrnr_nnet`, `Lrnr_nnls`, `Lrnr_optim`, `Lrnr_pca`, `Lrnr_pkg_SuperLearner`, `Lrnr_polspline`, `Lrnr_pooled_hazards`, `Lrnr_randomForest`, `Lrnr_ranger`, `Lrnr_revere_task`, `Lrnr_rpart`, `Lrnr_rugarch`, `Lrnr_screener_augment`, `Lrnr_screener_coefs`, `Lrnr_screener_correlation`, `Lrnr_screener_importance`, `Lrnr_sl`, `Lrnr_solnp_density`, `Lrnr_solnp`, `Lrnr_stratified`, `Lrnr_subset_covariates`, `Lrnr_svm`, `Lrnr_tsDyn`, `Lrnr_ts_weights`, `Lrnr_xgboost`, `Pipeline`, `Stack`, `define_h2o_X()`,

Berkeley
UNIVERSITY OF CALIFORNIA

https://tlverse.org/sl3/reference/index.html#sl-learners

# Learners in `sl3`

## *How to estimate prediction function?*

Other Learners: `Lrnr_HarmonicReg`, `Lrnr_arima`, `Lrnr_bartMachine`, `Lrnr_base`, `Lrnr_bayesglm`, `Lrnr_bilstm`, `Lrnr_caret`, `Lrnr_cv_selector`, `Lrnr_cv`, `Lrnr_dbarts`, `Lrnr_define_interactions`, `Lrnr_density_discretize`, `Lrnr_density_hse`, `Lrnr_density_semiparametric`, `Lrnr_earth`, `Lrnr_expSmooth`, `Lrnr_gam`, `Lrnr_ga`, `Lrnr_gbm`, `Lrnr_glm_fast`, `Lrnr_glmnet`, `Lrnr_glm`, `Lrnr_grf`, `Lrnr_gru_keras`, `Lrnr_gts`, `Lrnr_h2o_grid`, `Lrnr_hal9001`, `Lrnr_haldensify`, `Lrnr_hts`, `Lrnr_independent_binomial`, `Lrnr_lightgbm`, `Lrnr_lstm_keras`, `Lrnr_mean`, `Lrnr_multiple_ts`, `Lrnr_multivariate`, `Lrnr_nnet`, `Lrnr_nnls`, `Lrnr_optim`, `Lrnr_pca`, `Lrnr_pkg_SuperLearner`, `Lrnr_polspline`, `Lrnr_pooled_hazards`, `Lrnr_randomForest`, `Lrnr_ranger`, `Lrnr_revere_task`, `Lrnr_rpart`, `Lrnr_rugarch`, `Lrnr_screener_augment`, `Lrnr_screener_coefs`, `Lrnr_screener_correlation`, `Lrnr_screener_importance`, `Lrnr_sl`, `Lrnr_solnp_density`, `Lrnr_solnp`, `Lrnr_stratified`, `Lrnr_subset_covariates`, `Lrnr_svm`, `Lrnr_tsDyn`, `Lrnr_ts_weights`, `Lrnr_xgboost`, `Pipeline`, `Stack`, `define_h2o_X()`,



https://tlverse.org/sl3/reference/index.html#sl-learners

# Learners in `sl3`

## *How to estimate prediction function?*

Other Learners: `Lrnr_HarmonicReg`, `Lrnr_arima`, `Lrnr_bartMachine`, `Lrnr_base`, `Lrnr_bayesglm`, `Lrnr_bilstm`, `Lrnr_caret`, `Lrnr_cv_selector`, `Lrnr_cv`, `Lrnr_dbarts`, `Lrnr_define_interactions`, `Lrnr_density_discretize`, `Lrnr_density_hse`, `Lrnr_density_semiparametric`, `Lrnr_earth`, `Lrnr_expSmooth`, `Lrnr_gam`, `Lrnr_ga`, `Lrnr_gbm`, `Lrnr_glm_fast`, `Lrnr_glmnet`, `Lrnr_glm`, `Lrnr_grf`, `Lrnr_gru_keras`, `Lrnr_gts`, `Lrnr_h2o_grid`, `Lrnr_hal9001`, `Lrnr_haldensify`, `Lrnr_hts`, `Lrnr_independent_binomial`, `Lrnr_lightgbm`, `Lrnr_lstm_keras`, `Lrnr_mean`, `Lrnr_multiple_ts`, `Lrnr_multivariate`, `Lrnr_nnet`, `Lrnr_nnls`, `Lrnr_optim`, `Lrnr_pca`, `Lrnr_pkg_SuperLearner`, `Lrnr_polspline`, `Lrnr_pooled_hazards`, `Lrnr_randomForest`, `Lrnr_ranger`, `Lrnr_revere_task`, `Lrnr_rpart`, `Lrnr_rugarch`, `Lrnr_screener_augment`, `Lrnr_screener_coefs`, `Lrnr_screener_correlation`, `Lrnr_screener_importance`, `Lrnr_sl`, `Lrnr_solnp_density`, `Lrnr_solnp`, `Lrnr_stratified`, `Lrnr_subset_covariates`, `Lrnr_svm`, `Lrnr_tsDyn`, `Lrnr_ts_weights`, `Lrnr_xgboost`, `Pipeline`, `Stack`, `define_h2o_X()`,

Berkeley UNIVERSITY OF CALIFORNIA

https://tlverse.org/sl3/reference/index.html#sl-learners

# Learners in `sl3`

## *How to estimate prediction function?*

Other Learners: `Lrnr_HarmonicReg`, `Lrnr_arima`, `Lrnr_bartMachine`, `Lrnr_base`, `Lrnr_bayesglm`, `Lrnr_bilstm`, `Lrnr_caret`, `Lrnr_cv_selector`, `Lrnr_cv`, `Lrnr_dbarts`, `Lrnr_define_interactions`, `Lrnr_density_discretize`, `Lrnr_density_hse`, `Lrnr_density_semiparametric`, `Lrnr_earth`, `Lrnr_expSmooth`, `Lrnr_gam`, `Lrnr_ga`, `Lrnr_gbm`, `Lrnr_glm_fast`, `Lrnr_glmnet`, `Lrnr_glm`, `Lrnr_grf`, `Lrnr_gru_keras`, `Lrnr_gts`, `Lrnr_h2o_grid`, `Lrnr_hal9001`, `Lrnr_haldensify`, `Lrnr_hts`, `Lrnr_independent_binomial`, `Lrnr_lightgbm`, `Lrnr_lstm_keras`, `Lrnr_mean`, `Lrnr_multiple_ts`, `Lrnr_multivariate`, `Lrnr_nnet`, `Lrnr_nnls`, `Lrnr_optim`, `Lrnr_pca`, `Lrnr_pkg_SuperLearner`, `Lrnr_polspline`, `Lrnr_pooled_hazards`, `Lrnr_randomForest`, `Lrnr_ranger`, `Lrnr_revere_task`, `Lrnr_rpart`, `Lrnr_rugarch`, `Lrnr_screener_augment`, `Lrnr_screener_coefs`, `Lrnr_screener_correlation`, `Lrnr_screener_importance`, `Lrnr_sl`, `Lrnr_solnp_density`, `Lrnr_solnp`, `Lrnr_stratified`, `Lrnr_subset_covariates`, `Lrnr_svm`, `Lrnr_tsDyn`, `Lrnr_ts_weights`, `Lrnr_xgboost`, `Pipeline`, `Stack`, `define_h2o_X()`,



https://tlverse.org/sl3/reference/index.html#sl-learners

# Learners in `sl3`

## *How to estimate prediction function?*

Other Learners: `Lrnr_HarmonicReg`, `Lrnr_arima`, `Lrnr_bartMachine`, `Lrnr_base`, `Lrnr_bayesglm`, `Lrnr_bilstm`, `Lrnr_caret`, `Lrnr_cv_selector`, `Lrnr_cv`, `Lrnr_dbarts`, `Lrnr_define_interactions`, `Lrnr_density_discretize`, `Lrnr_density_hse`, `Lrnr_density_semiparametric`, `Lrnr_earth`, `Lrnr_expSmooth`, `Lrnr_gam`, `Lrnr_ga`, `Lrnr_gbm`, `Lrnr_glm_fast`, `Lrnr_glmnet`, `Lrnr_glm`, `Lrnr_grf`, `Lrnr_gru_keras`, `Lrnr_gts`, `Lrnr_h2o_grid`, `Lrnr_hal9001`, `Lrnr_haldensify`, `Lrnr_hts`, `Lrnr_independent_binomial`, `Lrnr_lightgbm`, `Lrnr_lstm_keras`, `Lrnr_mean`, `Lrnr_multiple_ts`, `Lrnr_multivariate`, `Lrnr_nnet`, `Lrnr_nnls`, `Lrnr_optim`, `Lrnr_pca`, `Lrnr_pkg_SuperLearner`, `Lrnr_polspline`, `Lrnr_pooled_hazards`, `Lrnr_randomForest`, `Lrnr_ranger`, `Lrnr_revere_task`, `Lrnr_rpart`, `Lrnr_rugarch`, `Lrnr_screener_augment`, `Lrnr_screener_coefs`, `Lrnr_screener_correlation`, `Lrnr_screener_importance`, `Lrnr_sl`, `Lrnr_solnp_density`, `Lrnr_solnp`, `Lrnr_stratified`, `Lrnr_subset_covariates`, `Lrnr_svm`, `Lrnr_tsDyn`, `Lrnr_ts_weights`, `Lrnr_xgboost`, `Pipeline`, `Stack`, `define_h2o_X()`,

Berkeley
UNIVERSITY OF CALIFORNIA

https://tlverse.org/sl3/reference/index.html#sl-learners

# Learners in `sl3`

## *How to estimate prediction function?*

Other Learners: `Lrnr_HarmonicReg`, `Lrnr_arima`, `Lrnr_bartMachine`, `Lrnr_base`, `Lrnr_bayesglm`, `Lrnr_bilstm`, `Lrnr_caret`, `Lrnr_cv_selector`, `Lrnr_cv`, `Lrnr_dbarts`, `Lrnr_define_interactions`, `Lrnr_density_discretize`, `Lrnr_density_hse`, `Lrnr_density_semiparametric`, `Lrnr_earth`, `Lrnr_expSmooth`, `Lrnr_gam`, `Lrnr_ga`, `Lrnr_gbm`, `Lrnr_glm_fast`, `Lrnr_glmnet`, `Lrnr_glm`, `Lrnr_grf`, `Lrnr_gru_keras`, `Lrnr_gts`, `Lrnr_h2o_grid`, `Lrnr_hal9001`, `Lrnr_haldensify`, `Lrnr_hts`, `Lrnr_independent_binomial`, `Lrnr_lightgbm`, `Lrnr_lstm_keras`, `Lrnr_mean`, `Lrnr_multiple_ts`, `Lrnr_multivariate`, `Lrnr_nnet`, `Lrnr_nnls`, `Lrnr_optim`, `Lrnr_pca`, `Lrnr_pkg_SuperLearner`, `Lrnr_polspline`, `Lrnr_pooled_hazards`, `Lrnr_randomForest`, `Lrnr_ranger`, `Lrnr_revere_task`, `Lrnr_rpart`, `Lrnr_rugarch`, `Lrnr_screener_augment`, `Lrnr_screener_coefs`, `Lrnr_screener_correlation`, `Lrnr_screener_importance`, `Lrnr_sl`, `Lrnr_solnp_density`, `Lrnr_solnp`, `Lrnr_stratified`, `Lrnr_subset_covariates`, `Lrnr_svm`, `Lrnr_tsDyn`, `Lrnr_ts_weights`, `Lrnr_xgboost`, `Pipeline`, $\boxed{\texttt{Stack}}$, `define_h2o_X()`,

# Learners in `sl3`

## ***How to estimate prediction function?***

Other Learners: `Lrnr_HarmonicReg`, `Lrnr_arima`, `Lrnr_bartMachine`, `Lrnr_base`, `Lrnr_bayesglm`, `Lrnr_bilstm`, `Lrnr_caret`, `Lrnr_cv_selector`, `Lrnr_cv`, `Lrnr_dbarts`, `Lrnr_define_interactions`, `Lrnr_density_discretize`, `Lrnr_density_hse`, `Lrnr_density_semiparametric`, `Lrnr_earth`, `Lrnr_expSmooth`, `Lrnr_gam`, `Lrnr_ga`, `Lrnr_gbm`, `Lrnr_glm_fast`, `Lrnr_glmnet`, `Lrnr_glm`, `Lrnr_grf`, `Lrnr_gru_keras`, `Lrnr_gts`, `Lrnr_h2o_grid`, `Lrnr_hal9001`, `Lrnr_haldensify`, `Lrnr_hts`, `Lrnr_independent_binomial`, `Lrnr_lightgbm`, `Lrnr_lstm_keras`, `Lrnr_mean`, `Lrnr_multiple_ts`, `Lrnr_multivariate`, `Lrnr_nnet`, `Lrnr_nnls`, `Lrnr_optim`, `Lrnr_pca`, `Lrnr_pkg_SuperLearner`, `Lrnr_polspline`, `Lrnr_pooled_hazards`, `Lrnr_randomForest`, `Lrnr_ranger`, `Lrnr_revere_task`, `Lrnr_rpart`, `Lrnr_rugarch`, `Lrnr_screener_augment`, `Lrnr_screener_coefs`, `Lrnr_screener_correlation`, `Lrnr_screener_importance`, `Lrnr_sl`, `Lrnr_solnp_density`, `Lrnr_solnp`, `Lrnr_stratified`, `Lrnr_subset_covariates`, `Lrnr_svm`, `Lrnr_tsDyn`, `Lrnr_ts_weights`, `Lrnr_xgboost`, `Pipeline`, `Stack`, `define_h2o_X()`,

Berkeley
UNIVERSITY OF CALIFORNIA

https://tlverse.org/sl3/reference/index.html#sl-learners

# Learners in `sl3`

## ***How to estimate prediction function?***

Other Learners: `Lrnr_HarmonicReg`, `Lrnr_arima`, `Lrnr_bartMachine`, `Lrnr_base`, `Lrnr_bayesglm`, `Lrnr_bilstm`, `Lrnr_caret`, `Lrnr_cv_selector`, `Lrnr_cv`, `Lrnr_dbarts`, `Lrnr_define_interactions`, `Lrnr_density_discretize`, `Lrnr_density_hse`, `Lrnr_density_semiparametric`, `Lrnr_earth`, `Lrnr_expSmooth`, `Lrnr_gam`, `Lrnr_ga`, `Lrnr_gbm`, `Lrnr_glm_fast`, `Lrnr_glmnet`, `Lrnr_glm`, `Lrnr_grf`, `Lrnr_gru_keras`, `Lrnr_gts`, `Lrnr_h2o_grid`, `Lrnr_hal9001`, `Lrnr_haldensify`, `Lrnr_hts`, `Lrnr_independent_binomial`, `Lrnr_lightgbm`, `Lrnr_lstm_keras`, `Lrnr_mean`, `Lrnr_multiple_ts`, `Lrnr_multivariate`, `Lrnr_nnet`, `Lrnr_nnls`, `Lrnr_optim`, `Lrnr_pca`, `Lrnr_pkg_SuperLearner`, `Lrnr_polspline`, `Lrnr_pooled_hazards`, `Lrnr_randomForest`, `Lrnr_ranger`, `Lrnr_revere_task`, `Lrnr_rpart`, `Lrnr_rugarch`, `Lrnr_screener_augment`, `Lrnr_screener_coefs`, `Lrnr_screener_correlation`, `Lrnr_screener_importance`, `Lrnr_sl`, `Lrnr_solnp_density`, `Lrnr_solnp`, `Lrnr_stratified`, `Lrnr_subset_covariates`, `Lrnr_svm`, `Lrnr_tsDyn`, `Lrnr_ts_weights`, `Lrnr_xgboost`, `Pipeline`, `Stack`, `define_h2o_X()`,



https://tlverse.org/sl3/reference/index.html#sl-learners

# Introductory overview of `sl3`

- Task
- Learners
- **Other functions**

Berkeley
UNIVERSITY OF CALIFORNIA

# Other sl3 Functions

- Performance measures:
  - loss functions (e.g., squared error, negative log-likelihood, multinomial log-likelihood )
  - metrics based on `ROCR` software package, like AUC, AUCPR, accuracy, sensitivity, with `custom_ROCR_risk()`
- Variable importance with `importance`
- Table with each candidate learner's cross-validated predictive performance with `cv_risk`
- Cross-validated SL with `cv_sl`

# Live coding exercise with `sl3`

https://tlverse.org/acic2022-adv-workshop/sl3.html

# WASH Benefits Bangladesh Example Dataset

- Study aiming to understand the effect of water quality, sanitation, hand washing, and nutritional interventions on child development in rural Bangladesh (WASH Benefits Bangladesh): a cluster randomized controlled trial (Tofail et al. 2018).

Berkeley
UNIVERSITY OF CALIFORNIA

# WASH Benefits Bangladesh Example Dataset

- Study aiming to understand the effect of water quality, sanitation, hand washing, and nutritional interventions on child development in rural Bangladesh (WASH Benefits Bangladesh): a cluster randomized controlled trial (Tofail et al. 2018).
- Enrolled pregnant women in their first or second trimester from the rural villages of Gazipur, Kishoreganj, Mymensingh, and Tangail districts of central Bangladesh, with an average of 8 women per cluster.

# WASH Benefits Bangladesh Example Dataset

- Groups of eight geographically adjacent clusters were block randomized, using a random number generator, into
  - six intervention groups (all received weekly visits from a community health promoter for the first 6 months, and every 2 weeks for next 18 months) and
  - a double-sized control group (no intervention or health promoter visit).

# WASH Benefits Bangladesh Example Dataset

- Six intervention groups:
  - chlorinated drinking water;
  - improved sanitation;
  - hand-washing with soap;
  - combined water, sanitation, and hand washing;
  - improved nutrition through counseling and provision of lipid-based nutrient supplements; and
  - combined water, sanitation, handwashing, and nutrition.
- We concentrate on child growth (size for age) as the outcome of interest.