Become a SUPER LEARNER

Using {sl3} to build ensemble learning models

Kat Hoffman
R-Ladies NYC

September 10, 2019

What is **Ensemble Learning?**

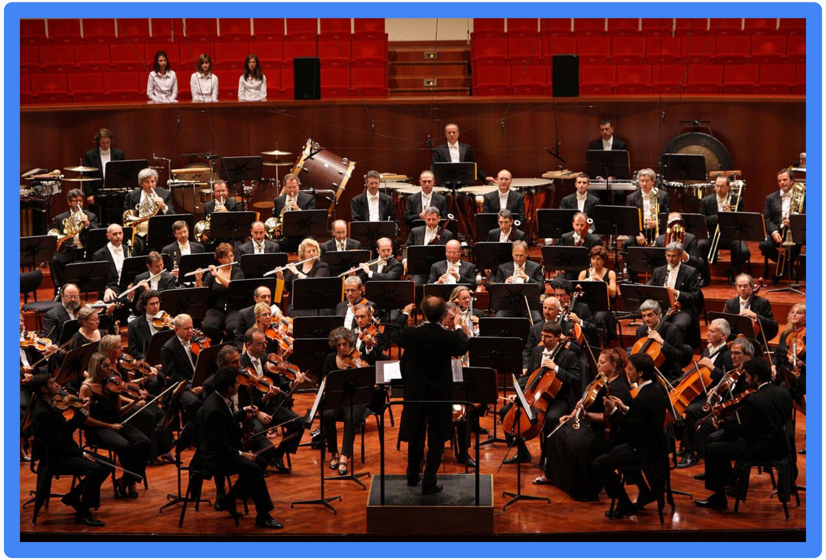


Image source: Royal Philharmonic Society

What is **Ensemble Learning**?

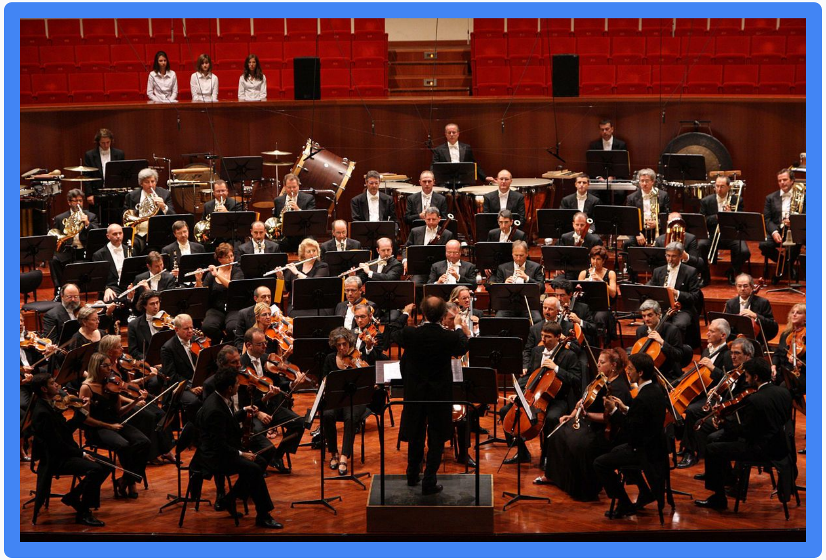


Image source: Royal Philharmonic Society

Ensemble learning: The process of combining multiple models to improve the overall model's prediction performance

What is **Ensemble Learning**?

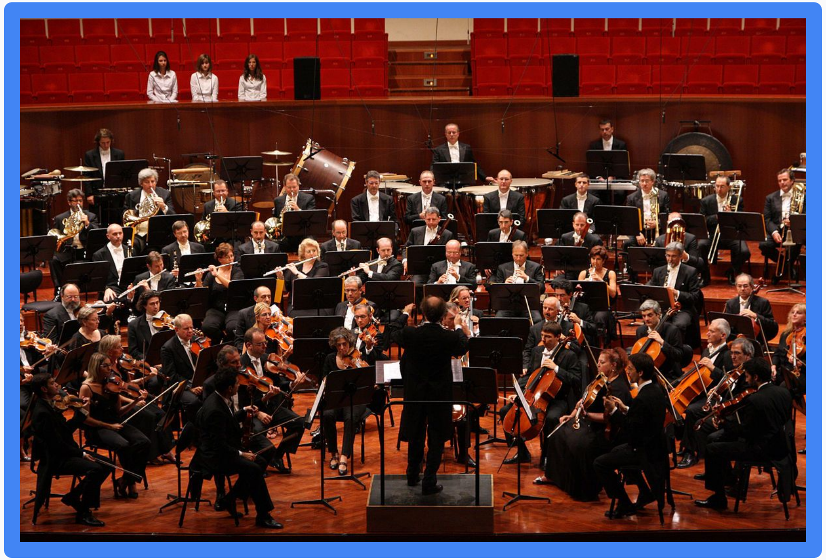


Image source: Royal Philharmonic Society

Ensemble learning: The process of combining multiple models to improve the overall model's prediction performance

Common techniques:

1. Bagging
2. Boosting
3. Stacking

Ensembling Technique 1: BAGGING

Bootstrap Aggregating

sample data
with replacement

```
bootstrap <-
  dplyr::sample_n(
    tbl = mtcars,
    size = 100,
    replace = T)
```

Ensembling Technique 1: BAGGING

Bootstrap Aggregating

sample data
with replacement

```
bootstrap <-
  dplyr::sample_n(
    tbl = mtcars,
    size = 100,
    replace = T)
```

combine
multiple models

Ensembling Technique 1: BAGGING

Bootstrap Aggregating

1 sample data with replacement

```
bootstrap <-
  dplyr::sample_n(
    tbl = mtcars,
    size = 100,
    replace = T)
```

2 fit a model on every bootstrapped data set

3 combine multiple models

Ensembling Technique 1: BAGGING

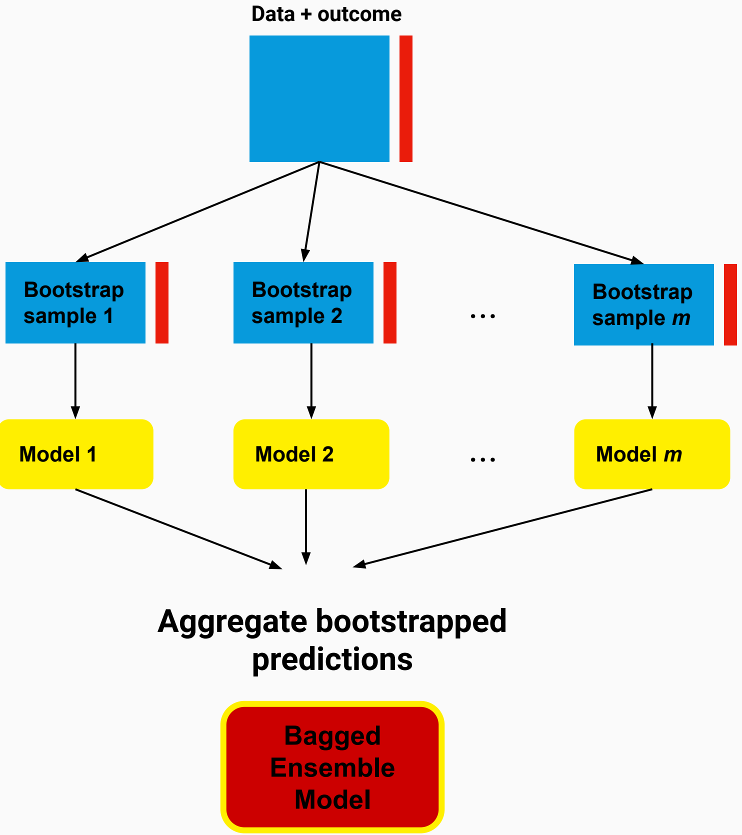
Bootstrap Aggregating

1 sample data with replacement

```
bootstrap <-
  dplyr::sample_n(
    tbl = mtcars,
    size = 100,
    replace = T)
```

2 fit a model on every bootstrapped data set

3 combine multiple models



BAGGING with Decision Trees

Bagging is most effective for unstable models, i.e. decision trees

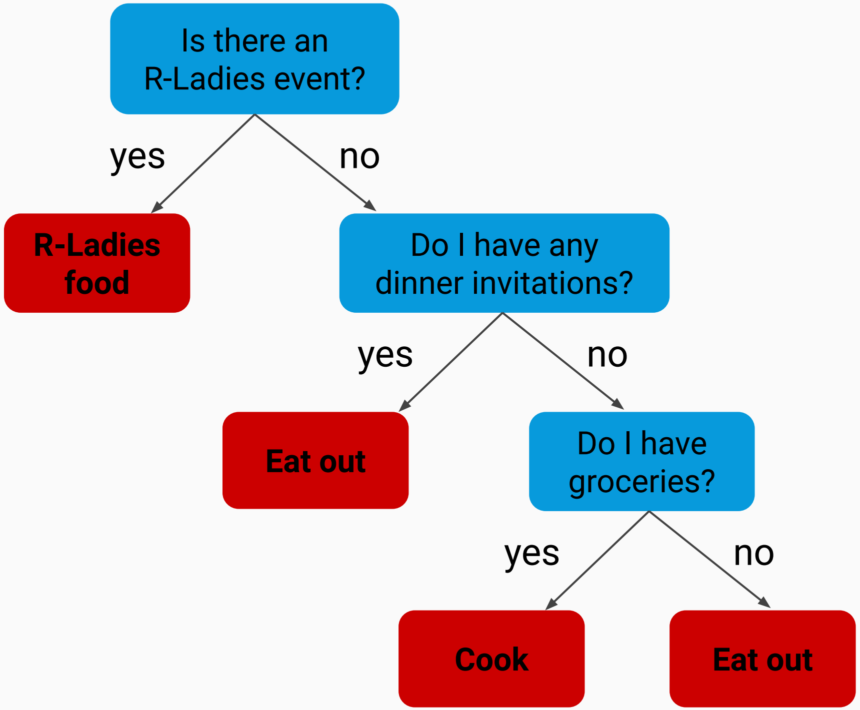
Decision tree: repeatedly subsetting your data in whichever way best predicts the final outcome

BAGGING with Decision Trees

Bagging is most effective for unstable models, i.e. decision trees

Decision tree: repeatedly subsetting your data in whichever way best predicts the final outcome

A decision tree for the categorical outcome of:
Dinner Plans



BAGGING with Decision Trees

Bagging is most effective for unstable models, i.e. decision trees

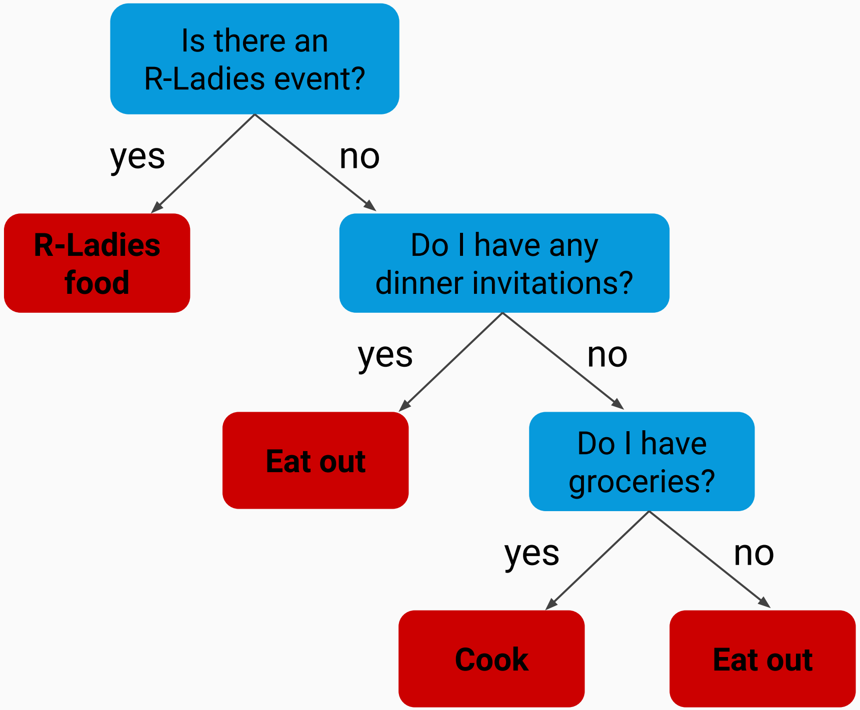
Decision tree: repeatedly subsetting your data in whichever way best predicts the final outcome

A very common, slight variation of bagging:

Random Forest: aggregated predictions from different decision trees

- Bootstrapped samples (Bagging)
- Limiting and randomizing the predictors to choose from at each decision branch

A decision tree for the categorical outcome of:
Dinner Plans



Random Forests in R

- Basic implementation: `RandomForest`
    - Main function: `randomForest()`
    - Simple tuning: `tuneRF()`
- For increased speed and easier tuning of parameters:
    - `ranger`
- Well-known interface for many models, not just random forests
    - `caret`

Random Forests in R

- Basic implementation: `RandomForest`
  - Main function: `randomForest()`
  - Simple tuning: `tuneRF()`
- For increased speed and easier tuning of parameters:
  - `ranger`
- Well-known interface for many models, not just random forests
  - `caret`

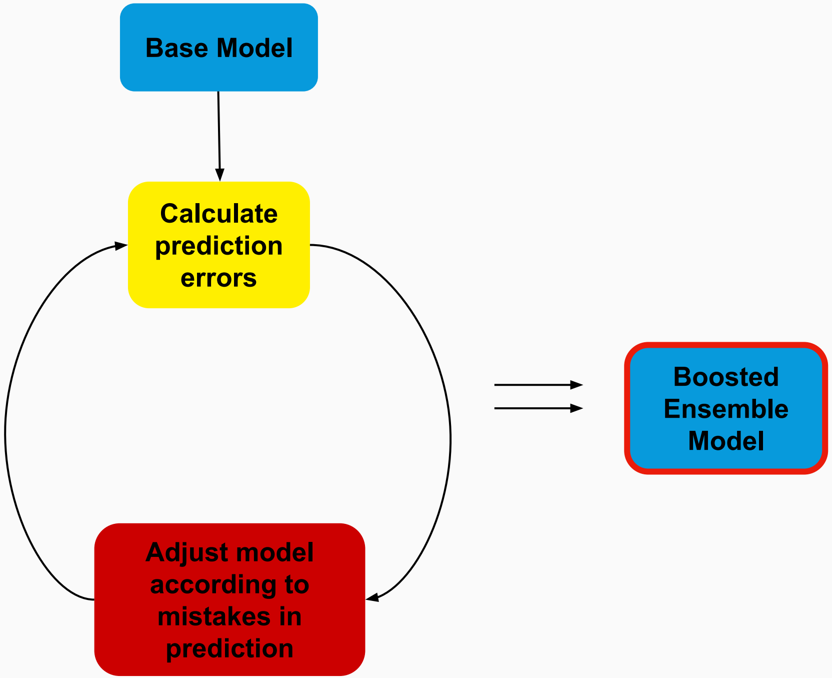
`brickr` + `rayshader` "random forest"
Source: Twitter, @ryantimpe

Ensembling Technique 2: BOOSTING

During **bagging**, models are fit ***in parallel***, but in **boosting**, models are fit ***sequentially*** with the goal to learn from past mistakes

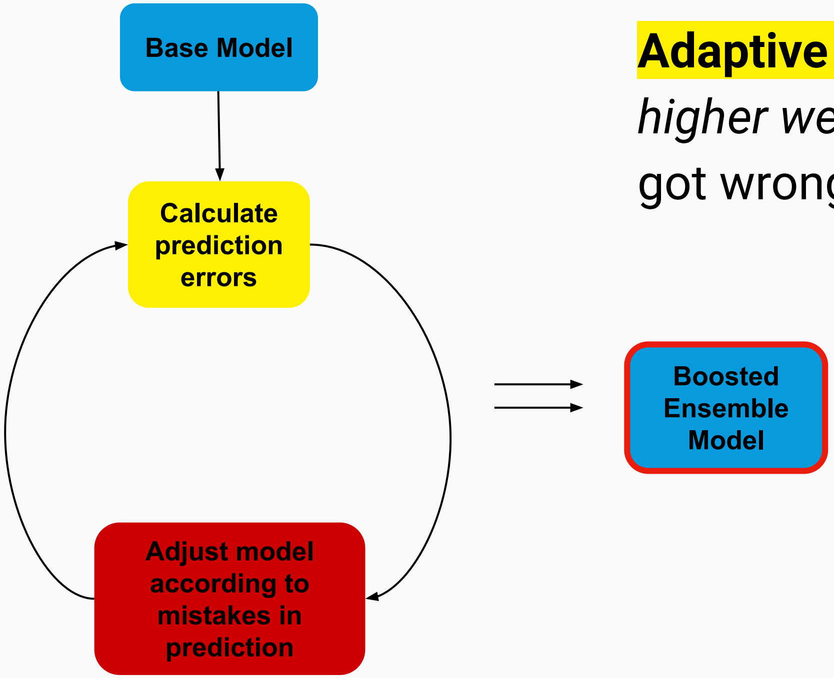
Ensembling Technique 2: BOOSTING

During **bagging**, models are fit ***in parallel***, but in **boosting**, models are fit ***sequentially*** with the goal to learn from past mistakes



Ensembling Technique 2: BOOSTING

During **bagging**, models are fit ***in parallel***, but in **boosting**, models are fit ***sequentially*** with the goal to learn from past mistakes



Adaptive boosting: Adjust model by *assigning a higher weight* to the predictions the previous model got wrong

Gradient boosting: Adjust model by *making a new model to predict the errors* of the previous model and adding that error prediction to the previous model

BOOSTING in R

- Adaptive boosting:
  - `Adabag`
- Gradient boosting:
  - `gbm`
  - `Xgboost`
    - Computationally efficient, adds regularization to help with overfitting
- Generalized interface:
  - `caret`
  - `h2o`
  - `mlr/mlr3`

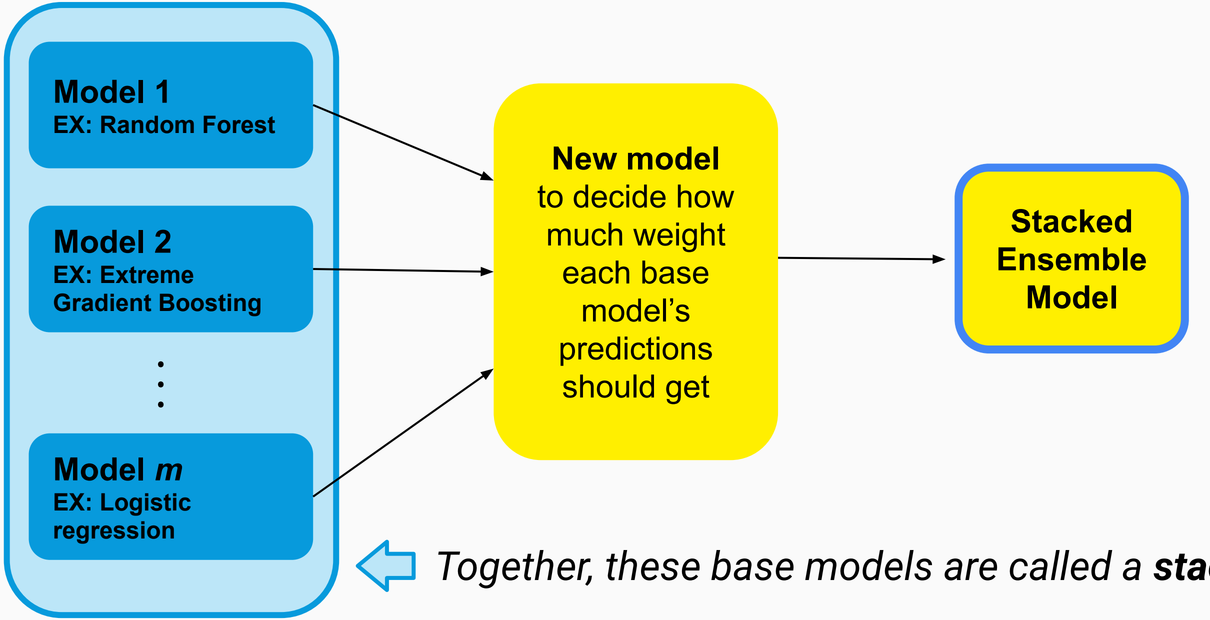
Check out [Rika Gorn's slides]() on **`xgboost`** from her R-Ladies Lightning Talk!

Common Technique 3: STACKING

Stacking: Several different types of models are built to predict an outcome, and a **new, separate model** is used to decide how much weight each base model's predictions should receive

Common Technique 3: STACKING

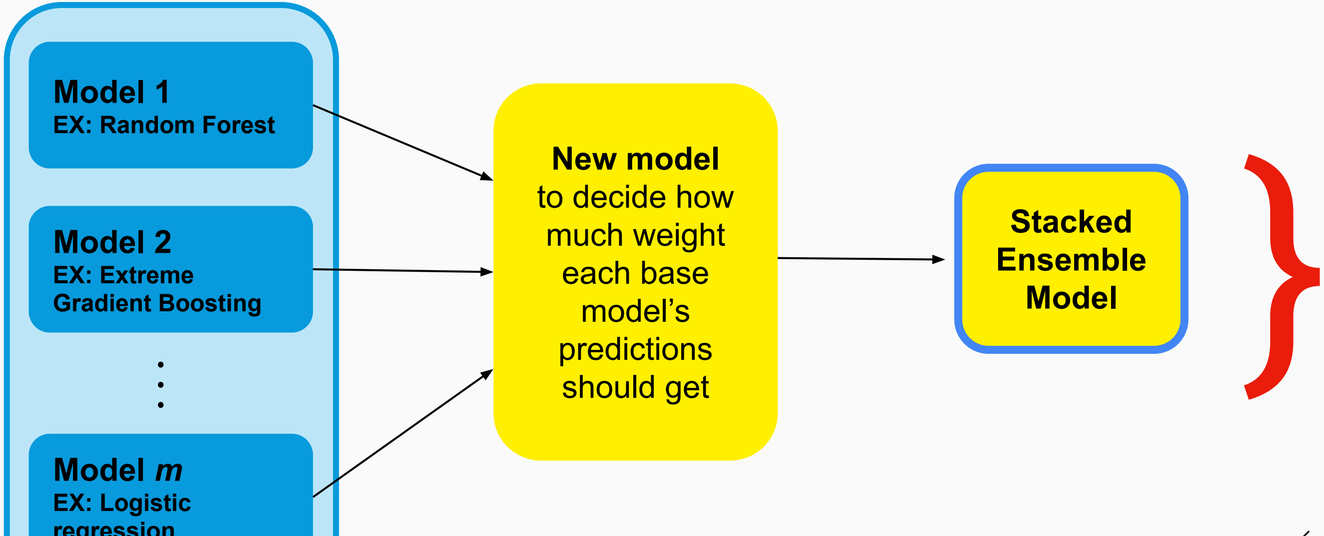
Stacking: Several different types of models are built to predict an outcome, and a **new, separate model** is used to decide how much weight each base model's predictions should receive



*Together, these base models are called a **stack*** 📚

Common Technique 3: STACKING

Stacking: Several different types of models are built to predict an outcome, and a **new, separate model** is used to decide how much weight each base model's predictions should receive



*Together, these base models are called a **stack***

A little jargon:
The base models are often called "**learners**" and the new model is often referred to as the "**meta-learner**"

History of Stacking AKA SUPERLEARNING

Fun stats fact of the day! In the early 2000s, a group of statisticians proved that stacking or “superlearning” would always perform as good or better than the best base model in your stack as sample size approaches ∞

An important point:
The predictions you input to your meta-learner must come from out-of-sample data (using methods like bootstrapping or cross-validating)

A quick aside: cross-validation

K-fold cross-validation:

1. splitting your data into equal parts

A quick aside: cross-validation

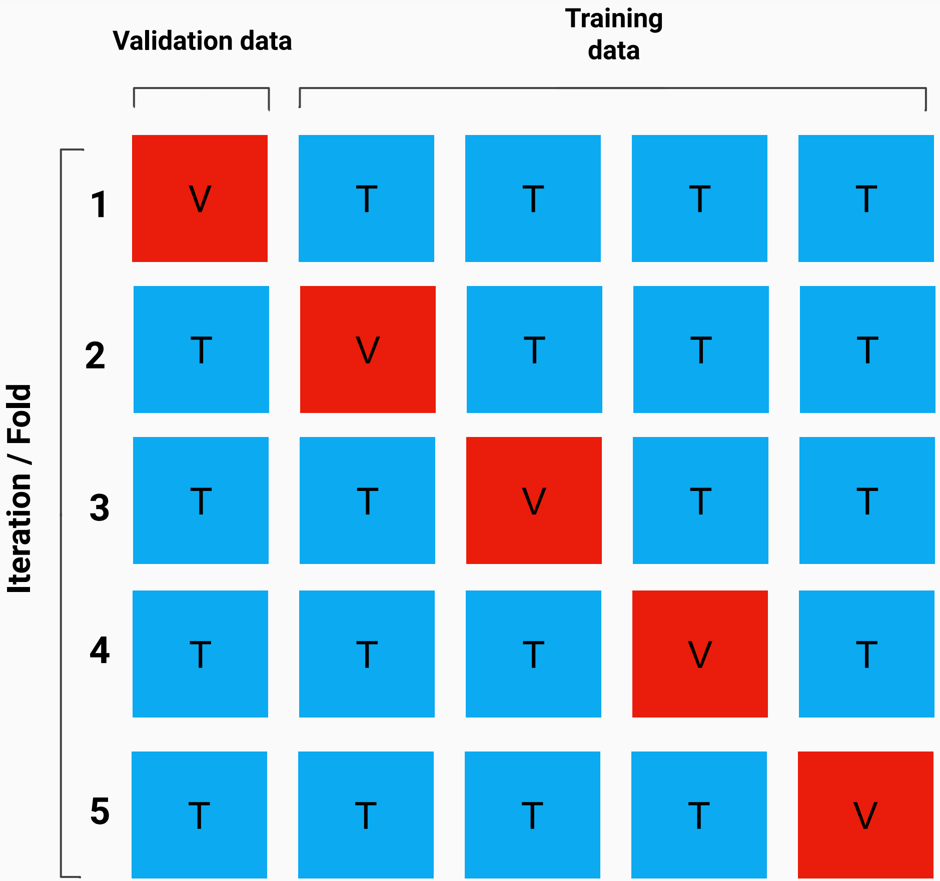
K-fold cross-validation:

1. splitting your data into equal parts
2. Training a model on all but one parts of the data
3. Validating, or testing, your model's performance on the remaining piece of data

A quick aside: cross-validation

K-fold cross-validation:

1. splitting your data into equal parts
2. Training a model on all but one parts of the data
3. Validating, or testing, your model's performance on the remaining piece of data
4. Repeating with each piece of data taking its turn as the validation set



Deep Dive of Stacking AKA SUPERLEARNING

One example of a super learner:

Step 1: Pick base learners

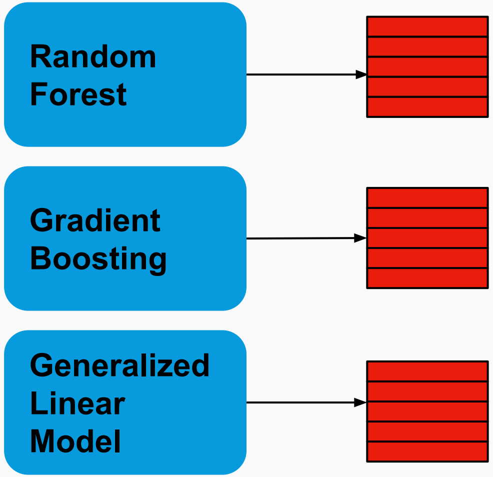
- Random Forest
- Gradient Boosting
- Generalized Linear Model

Deep Dive of Stacking AKA SUPERLEARNING

One example of a super learner:

Step 1: Pick base learners

Step 2: Cross validate to get out of sample predictions



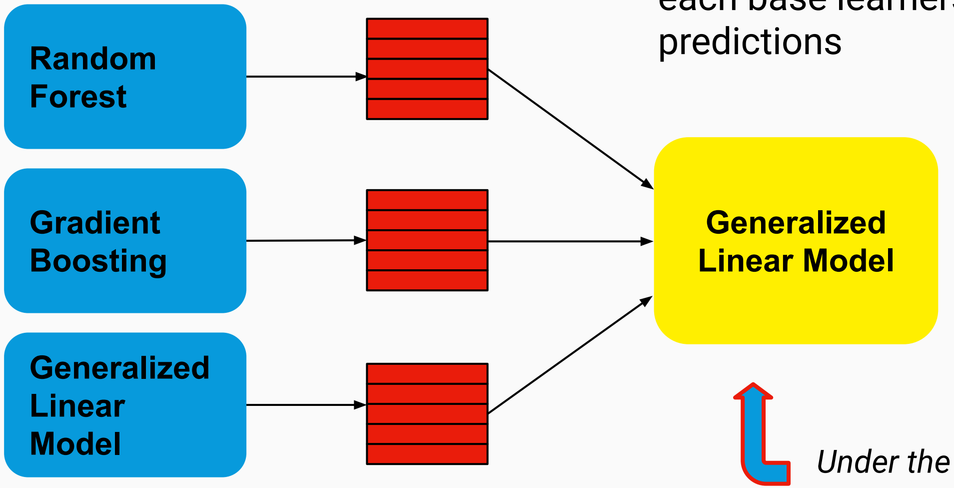
Deep Dive of Stacking AKA SUPERLEARNING

One example of a super learner:

Step 1: Pick base learners

Step 2: Cross validate to get out of sample predictions

Step 3: Pick a meta-learner, predict true outcome from each base learners predictions



Under the hood:

```
True_Outcome ~ RF_pred + GBM_pred + GLM_pred
```

Deep Dive of Stacking AKA SUPERLEARNING

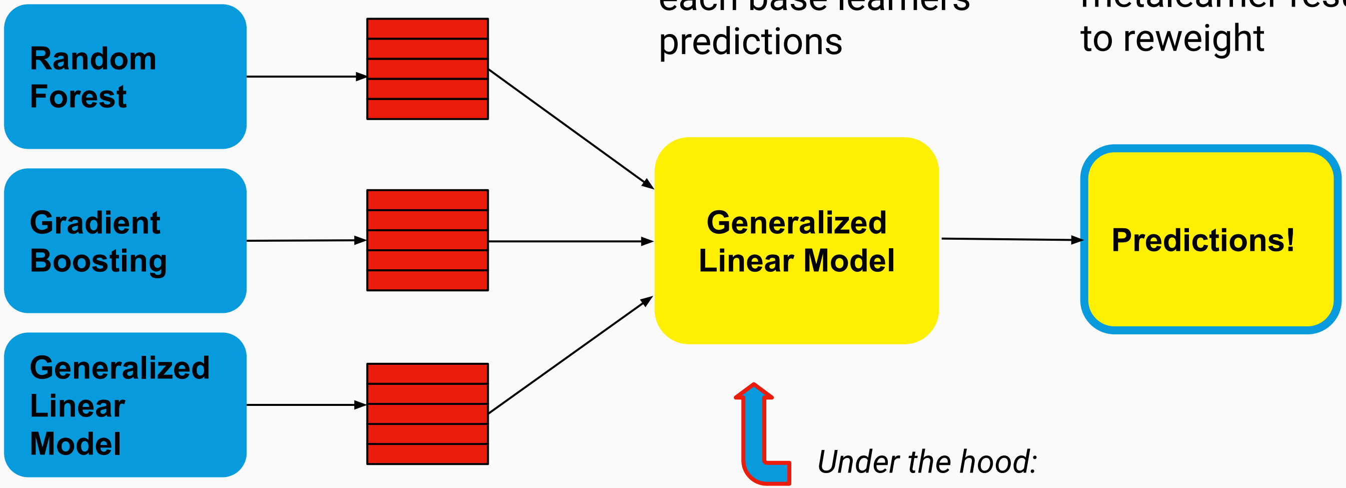
One example of a super learner:

Step 1: Pick base learners

Step 2: Cross validate to get out of sample predictions

Step 3: Pick a meta-learner, predict true outcome from each base learners predictions

Step 4: Fit base learners on your entire data sample and use the metalearner results to reweight



Under the hood:

`True_Outcome ~ RF_pred + GBM_pred + GLM_pred`

Deep Dive of Stacking AKA SUPERLEARNING

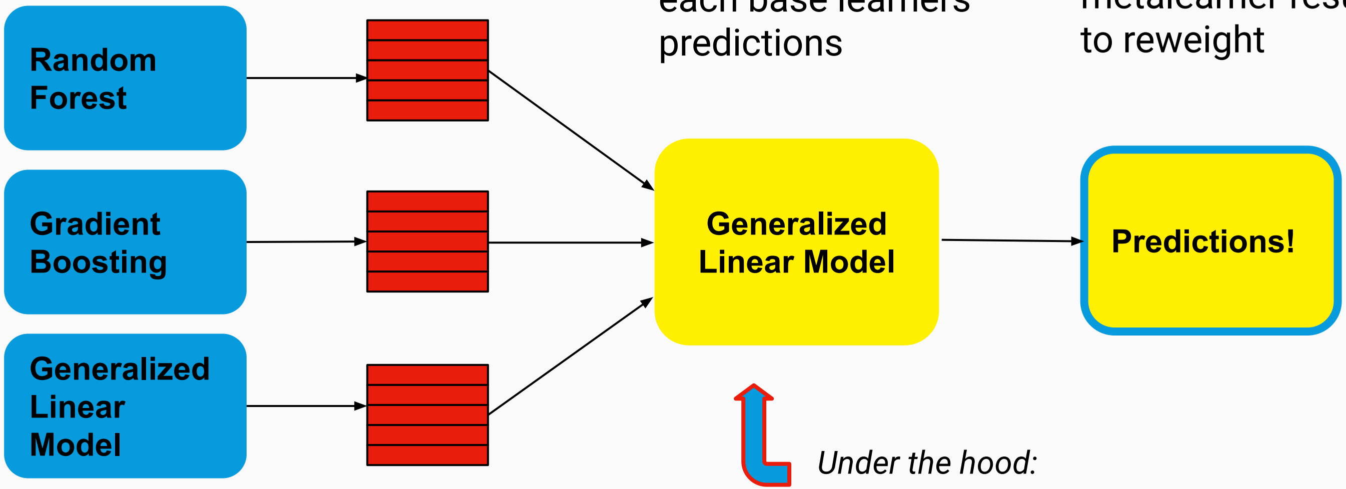
One example of a super learner:

Step 1: Pick base learners

Step 2: Cross validate to get out of sample predictions

Step 3: Pick a meta-learner, predict true outcome from each base learners predictions

Step 4: Fit base learners on your entire data sample and use the metalearner results to reweight



Under the hood:

`True_Outcome ~ RF_pred + GBM_pred + GLM_pred`

Stacking AKA SUPERLEARNING in `R`

There are many packages in `R` to implement stacking/ Superlearning. Some examples:

- `SuperLearner`
- `mlr / mlr3`
- `caretEnsemble`
- `h2o`

Stacking AKA SUPERLEARNING in R

There are many packages in `R` to implement stacking/ Superlearning. Some examples:

- `SuperLearner`
- `mlr / mlr3`
- `caretEnsemble`
- `h2o`

Fun R-Ladies fact of the day!

One of R-Ladies' co-founders, Erin Ledell, is the Chief Machine Learning Scientist at h2o (the software company which maintains h2o across a variety of programming platforms)

Stacking AKA SUPERLEARNING in R

There are many packages in `R` to implement stacking/ Superlearning. Some examples:

- `SuperLearner`
- `mlr / mlr3`
- `caretEnsemble`
- `h2o`

Why `sl3`?

- Comprehensive, faster, modernized syntax update to the older `SuperLearner` package
- Open source, written entirely in `R`
- Syntax modeled after popular machine learning packages such as scikit-learn

Fun R-Ladies fact of the day!
One of R-Ladies' co-founders, Erin Ledell, is the Chief Machine Learning Scientist at h2o (the software company which maintains h2o across a variety of programming platforms)

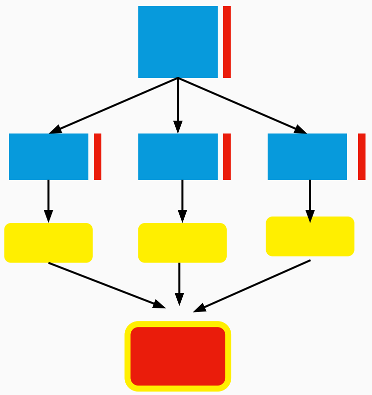
`sl3` Demo

WASH Benefits data set: measures of water quality, sanitation, hand washing, and nutritional interventions in rural Bangladesh and Kenya

We will use it to predict: children's weight-to-height z-scores

SuperReview:

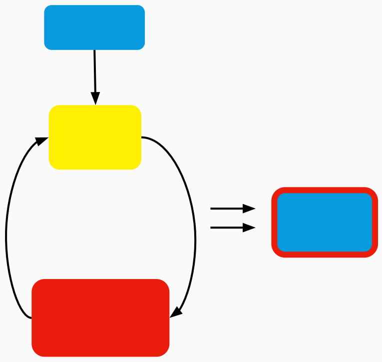
BAGGING



Aggregating bootstrapped predictions

`RandomForest`
`ranger`

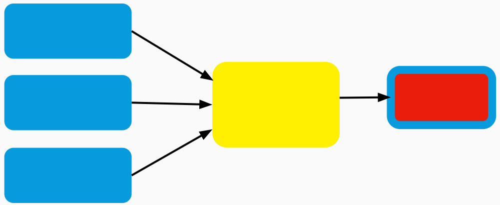
BOOSTING



Sequentially correcting models' mistakes

`AdaBoost`
`gbm`
`xgboost`

STACKING / SUPERLEARNING



Using a new model to blend together base models

`caretEnsemble`
`mlr/mlr3`
`h2o`

`sl3`

Fast, modern update to `SuperLearner` package

Similar syntax to popular ML packages in other languages

Written entirely in **R**, contributions welcomed

Helpful Resources:

Ensemble Learning:

- *Towards Data Science* articles:
  - "Understanding Random Forests"
  - "Ensemble Methods: Bagging, Boosting and Stacking"

- Bradley Boehmke's "Hands on Machine Learning with R," Chapters 10-15

- Datacamp's course: "Machine Learning with Tree-Based Models in R"

- Erin Ledell's "Introduction to Practical Ensemble Learning"

Superlearning and `sl3`:

- Teaching materials from the authors of sl3:
  - https://tlverse.org/tlverse-handbook/ensemble-machine-learning.html
  - https://tlverse.org/acic2019-workshop/ensemble-machine-learning.html
  - https://github.com/tlverse/sl3_lecture

- Peterson and Balzar's Causal Inference Seminar, Lab #3: "Super Learner" https://www.ucbbiostat.com/labs

- Polley, Eric C. and van der Laan, Mark J., "Super Learner In Prediction" (May 2010). *U.C. Berkeley Division of Biostatistics Working Paper Series*. Working Paper 266. https://biostats.bepress.com/ucbbiostat/paper266

Special thanks to one of `sl3`'s authors, Nima Hejazi, for answering questions.