Predictive Modeling in R

A Brief Introduction

NUIT Summer Data Science and Programming Workshop

July 25, 2019, 1 – 4 pm Chambers Hall

Workshop materials: https://github.com/andrewnolanhall/NUITPredictiveModelingSummer2019

Andrew N Hall

Northwestern University

Who am I?

Who are you?

Introduce yourselves to the person sitting next to you!

- Who you are
- What department you are coming from
- Why you want to learn about predictive modeling

Our Agenda

- Introduction
- Regression-Based Methods
- Single Decision Tree
- Ensemble Methods: Random Forest, GBMs
- Taking it Further: Neural Nets, NLP

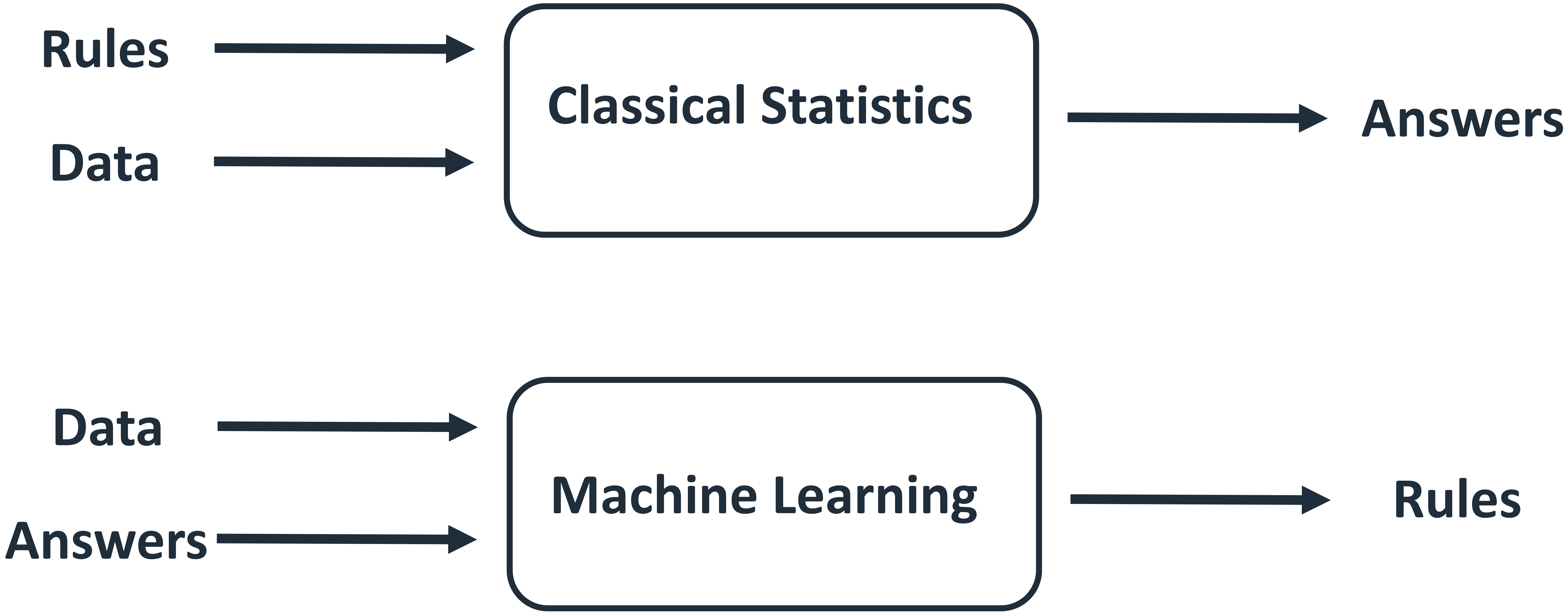
Our Agenda

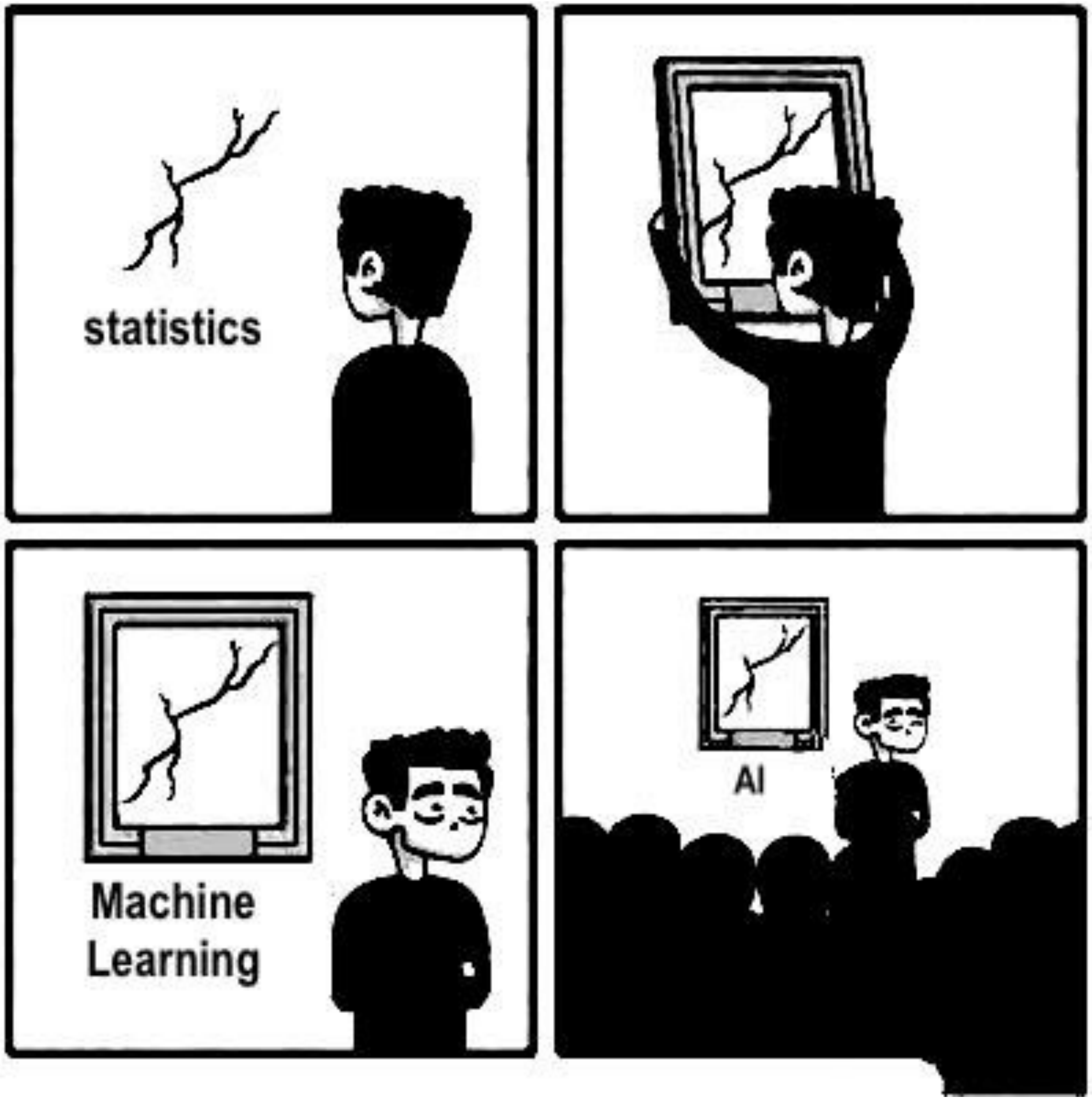
Introduction	Regression-Based Methods	Single Decision Tree	Ensemble Methods: Random Forest, GBMs	Taking it Further: Neural Nets, NLP
What is ML? Major Concepts	Application of classical statistics and introduction to cross-validation	Introduction to tree-based ML methods	The power of averaging weak learners	More complex methods (Touched upon briefly)

Introduction: What is Predictive Modeling?


THIS IS YOUR MACHINE LEARNING SYSTEM?
YUP! YOU POUR THE DATA INTO THIS BIG PILE OF LINEAR ALGEBRA, THEN COLLECT THE ANSWERS ON THE OTHER SIDE.
WHAT IF THE ANSWERS ARE WRONG?
JUST STIR THE PILE UNTIL THEY START LOOKING RIGHT.
DATA
ANSWERS


Rules
Data
Classical Statistics
Answers


Rules
Data
Classical Statistics
Answers
Data
Answers
Machine Learning
Rules


statistics
Machine
Learning
AI

Machine learning is…..

The “subfield of computer science that is concerned with building algorithms which, to be useful, rely on a collection of examples of some phenomenon. These examples can come from nature, be handcrafted by humans or generated by another algorithm.”

”Machine learning can also be defined as the process of solving a practical problem by 1) gathering a dataset, and 2) algorithmically building a statistical model based on that dataset. That statistical model is assumed to be used somehow to solve the practical problem.”

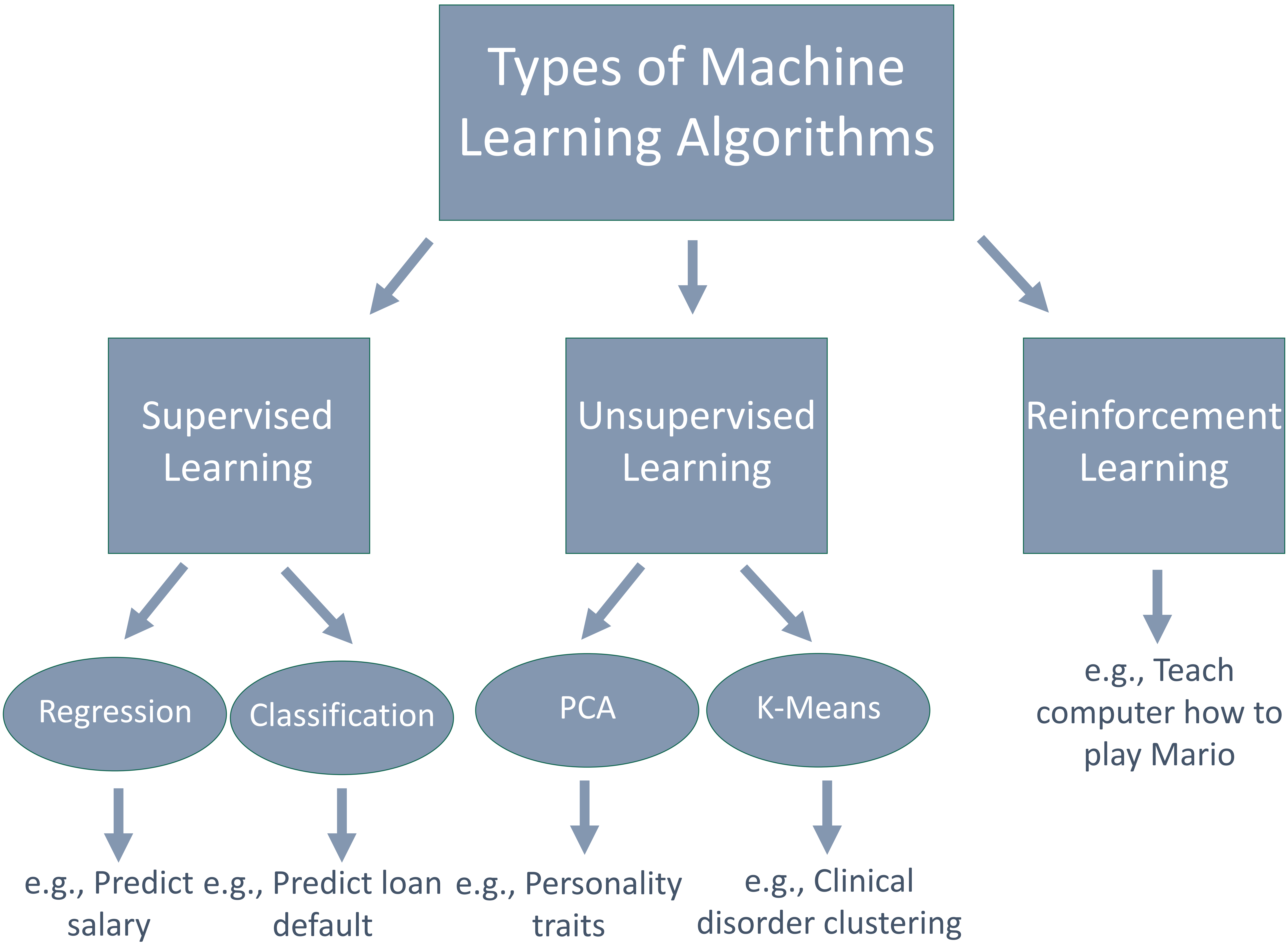
- Andriy Burkov

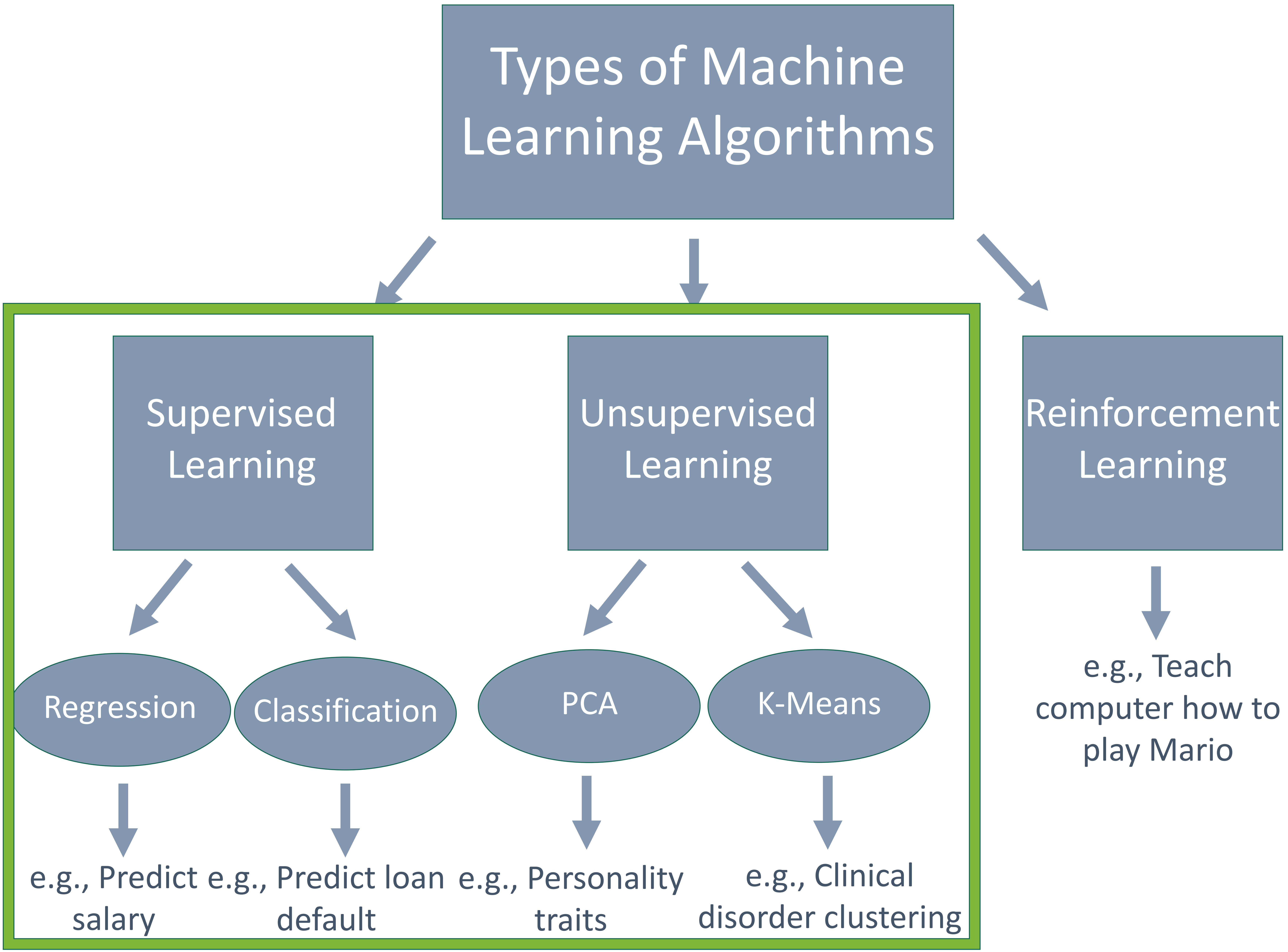
Machine learning is…..

The “subfield of computer science that is concerned with building algorithms which, to be useful, rely on a collection of examples of some phenomenon. These examples can come from nature, be handcrafted by humans or generated by another algorithm.”

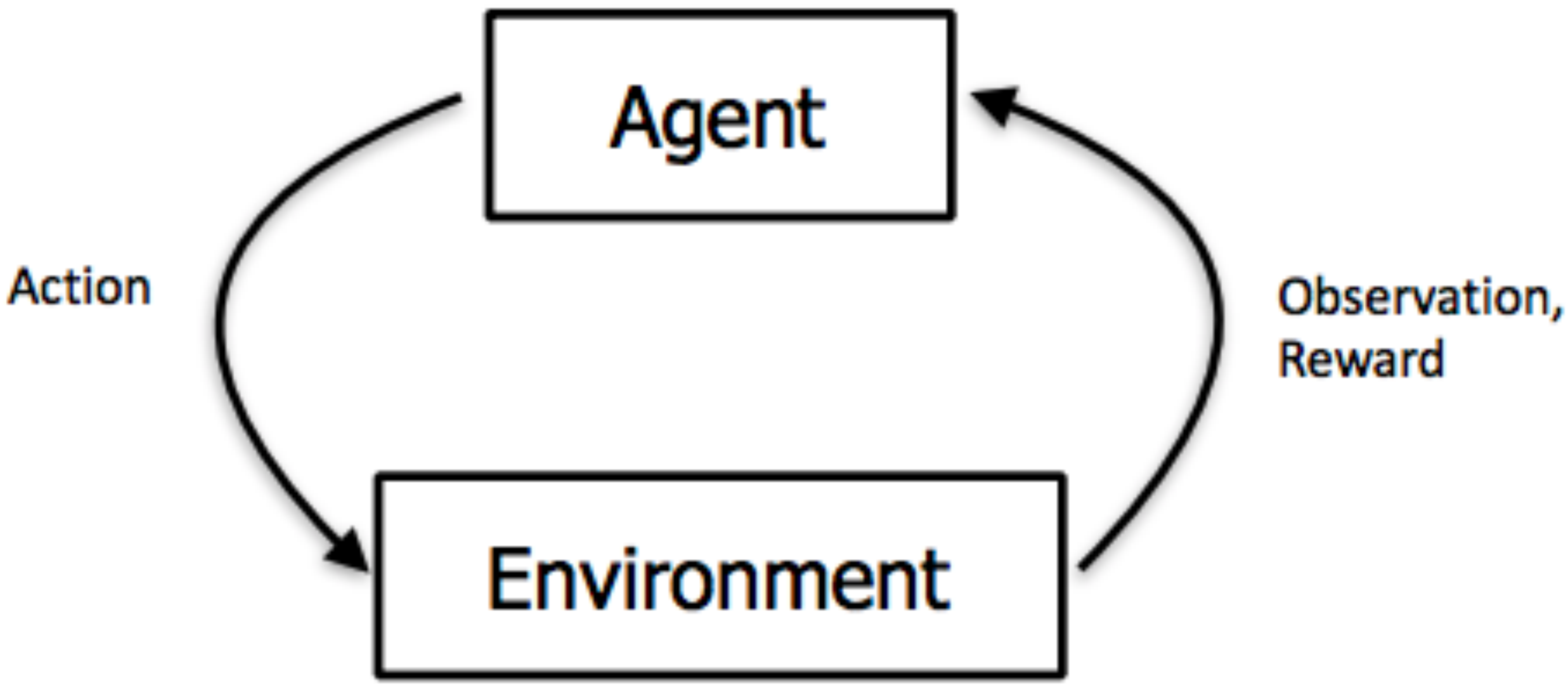
”Machine learning can also be defined as the process of solving a practical problem by 1) gathering a dataset, and 2) <u>algorithmically</u> building a statistical model based on that dataset. That statistical model is assumed to be used somehow to solve the practical problem.”

- Andriy Burkov


Types of Machine Learning Algorithms
Supervised Learning
Unsupervised Learning
Reinforcement Learning
Regression
Classification
PCA
K-Means
e.g., Teach computer how to play Mario
e.g., Predict salary
e.g., Predict loan default
e.g., Personality traits
e.g., Clinical disorder clustering


Types of Machine Learning Algorithms
Supervised Learning
Unsupervised Learning
Reinforcement Learning
Regression
Classification
PCA
K-Means
e.g., Predict salary
e.g., Predict loan default
e.g., Personality traits
e.g., Clinical disorder clustering
e.g., Teach computer how to play Mario

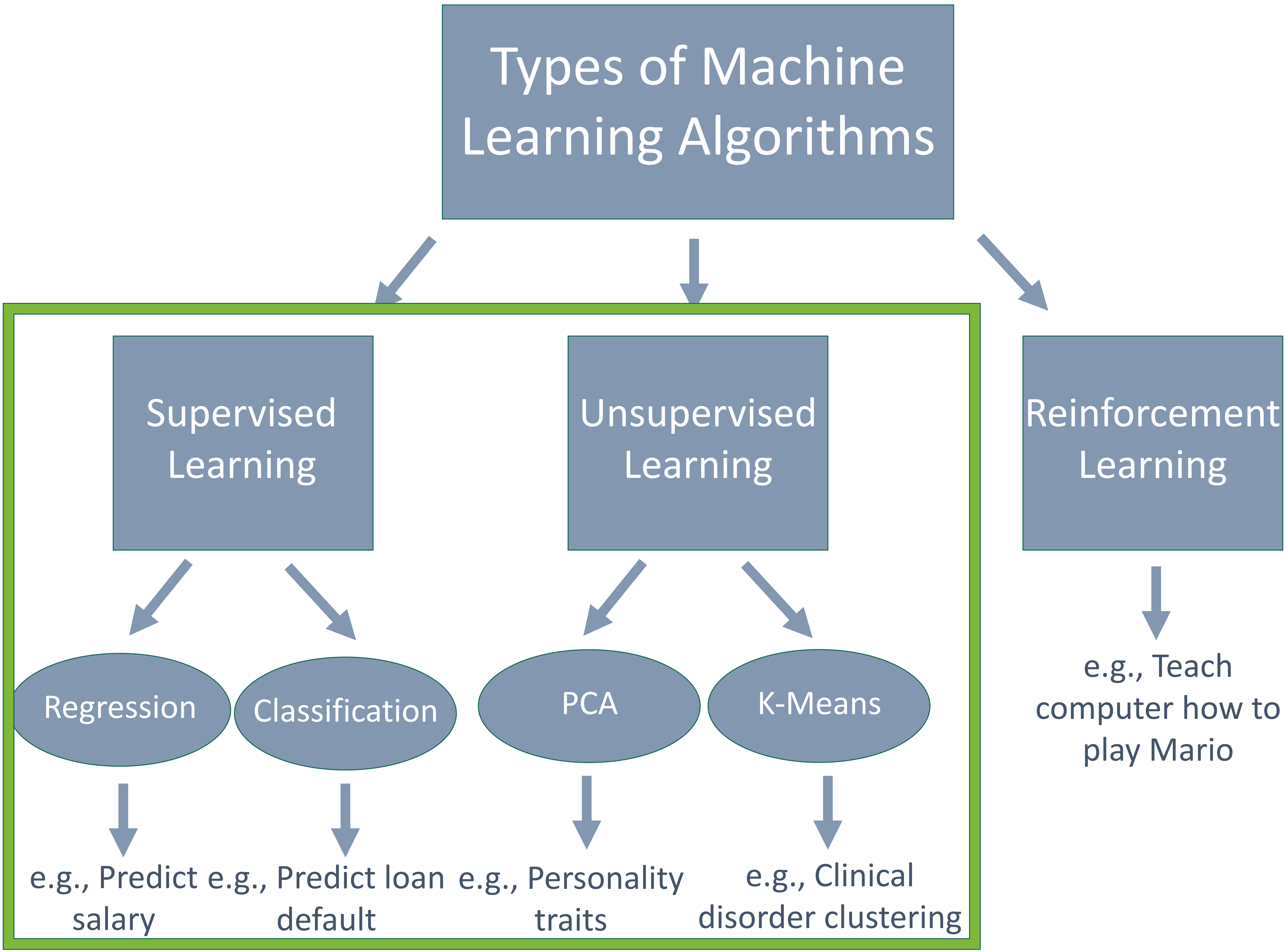
Reinforcement Learning (in *very* brief)

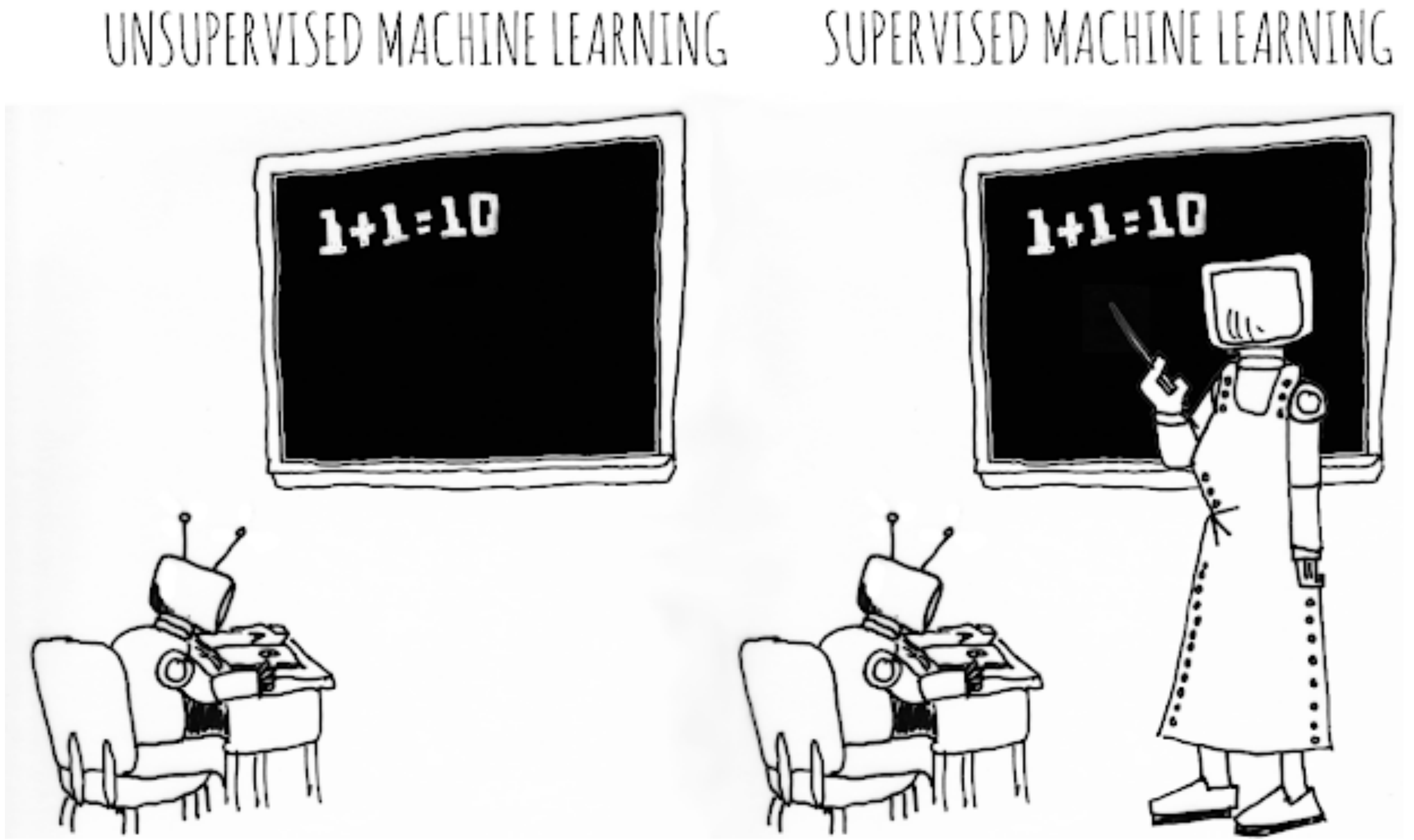


Machine placed in an environment where it executes actions, observes results, and responds to rewards to learn a policy of "behavior."

Useful when: decision making sequential, goal is long term (e.g., game playing, logistics, robotics). Differs from one-shot predictions on data from past.

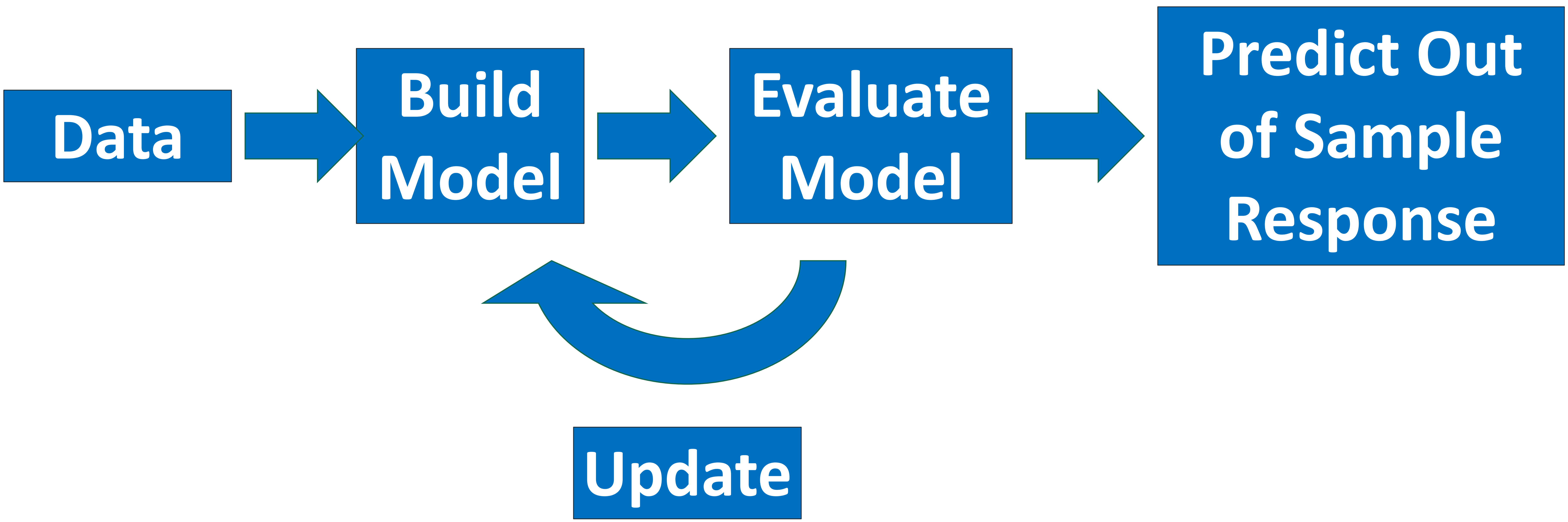
https://www.youtube.com/watch?v=qv6UVOQ0F44&t=2s

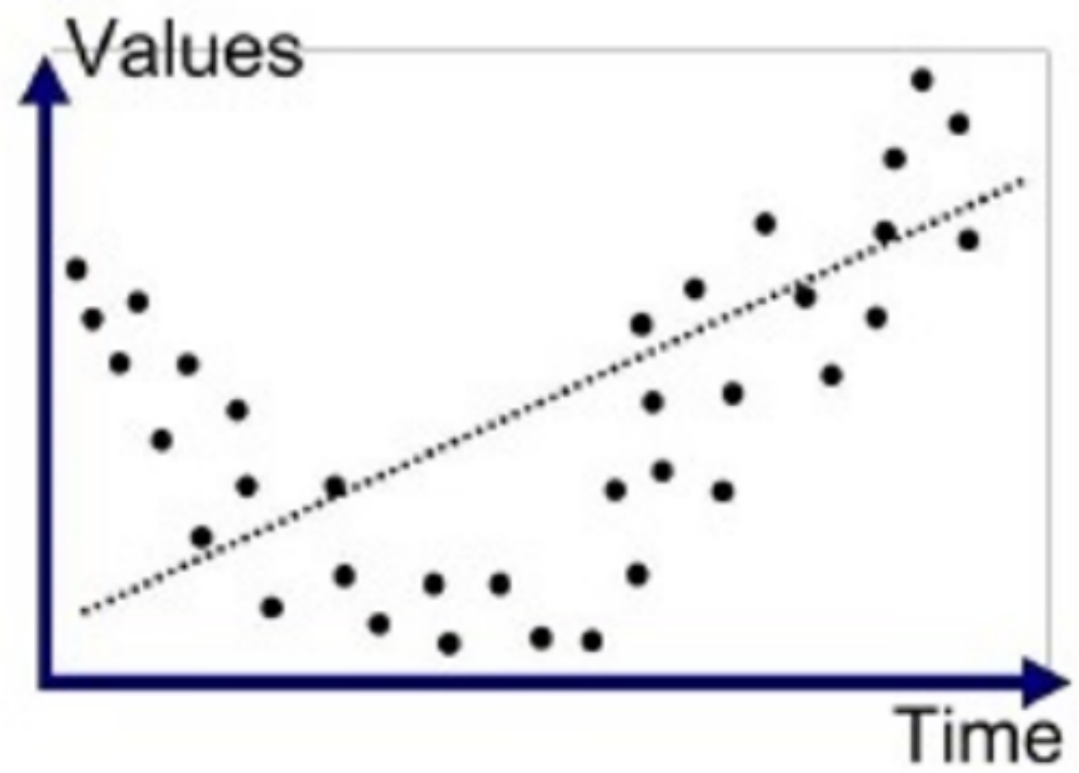

Types of Machine Learning Algorithms
Supervised Learning
Unsupervised Learning
Reinforcement Learning
Regression
Classification
PCA
K-Means
e.g., Teach computer how to play Mario
e.g., Predict salary
e.g., Predict loan default
e.g., Personality traits
e.g., Clinical disorder clustering


UNSUPERVISED MACHINE LEARNING
SUPERVISED MACHINE LEARNING
1+1=10
1+1=10
PROOFREADERSWHIMSY.BLOGSPOT.CA

Key ML tenet #1: Emphasis on Prediction

- Out-of-sample generalizability is key
- Machine Learning “pipeline”





Underfit

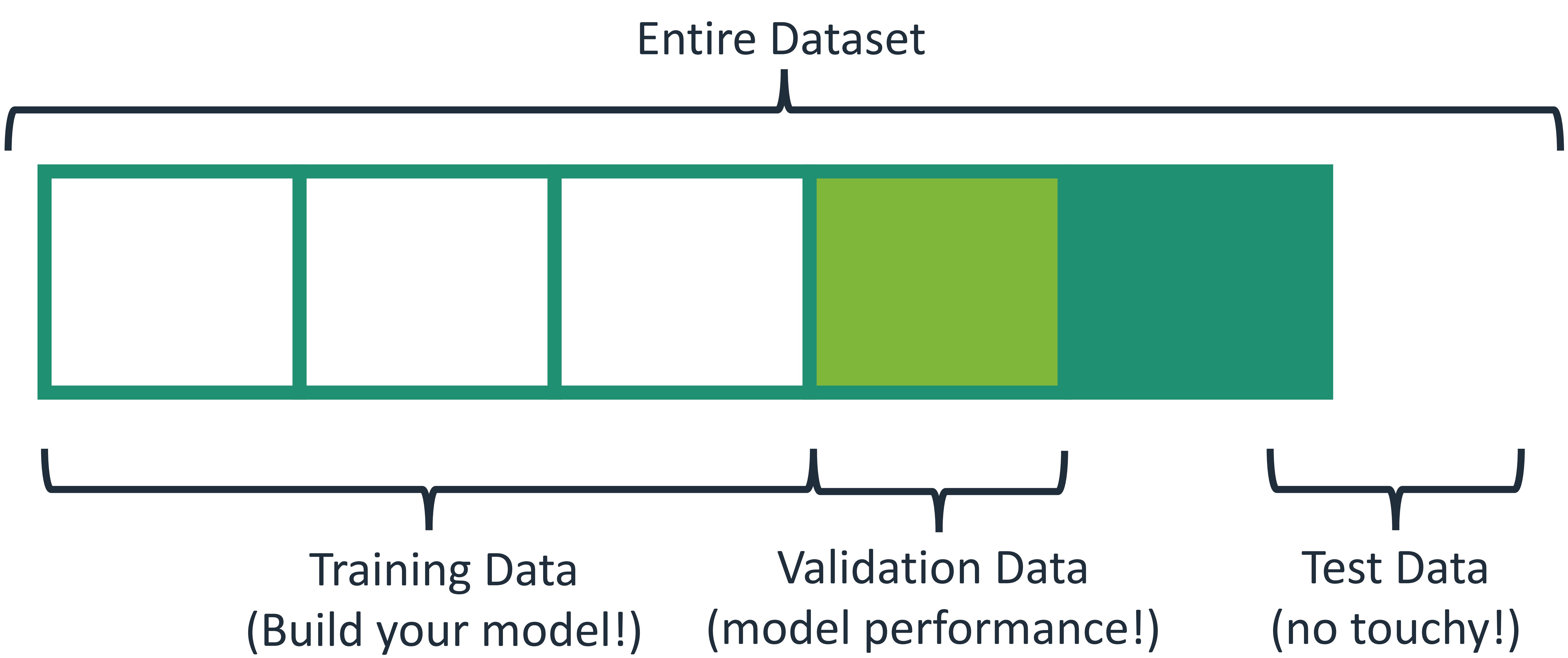
Key ML tenet #2: Bias-Variance Trade-off

Competing desires:

- Fit model to make accurate predictions for given data
- Vs. Fit model to predict future data

How do we solve this?

- Focus on *Test Set* performance



Thanks to machine-learning algorithms,
the robot apocalypse was short-lived.

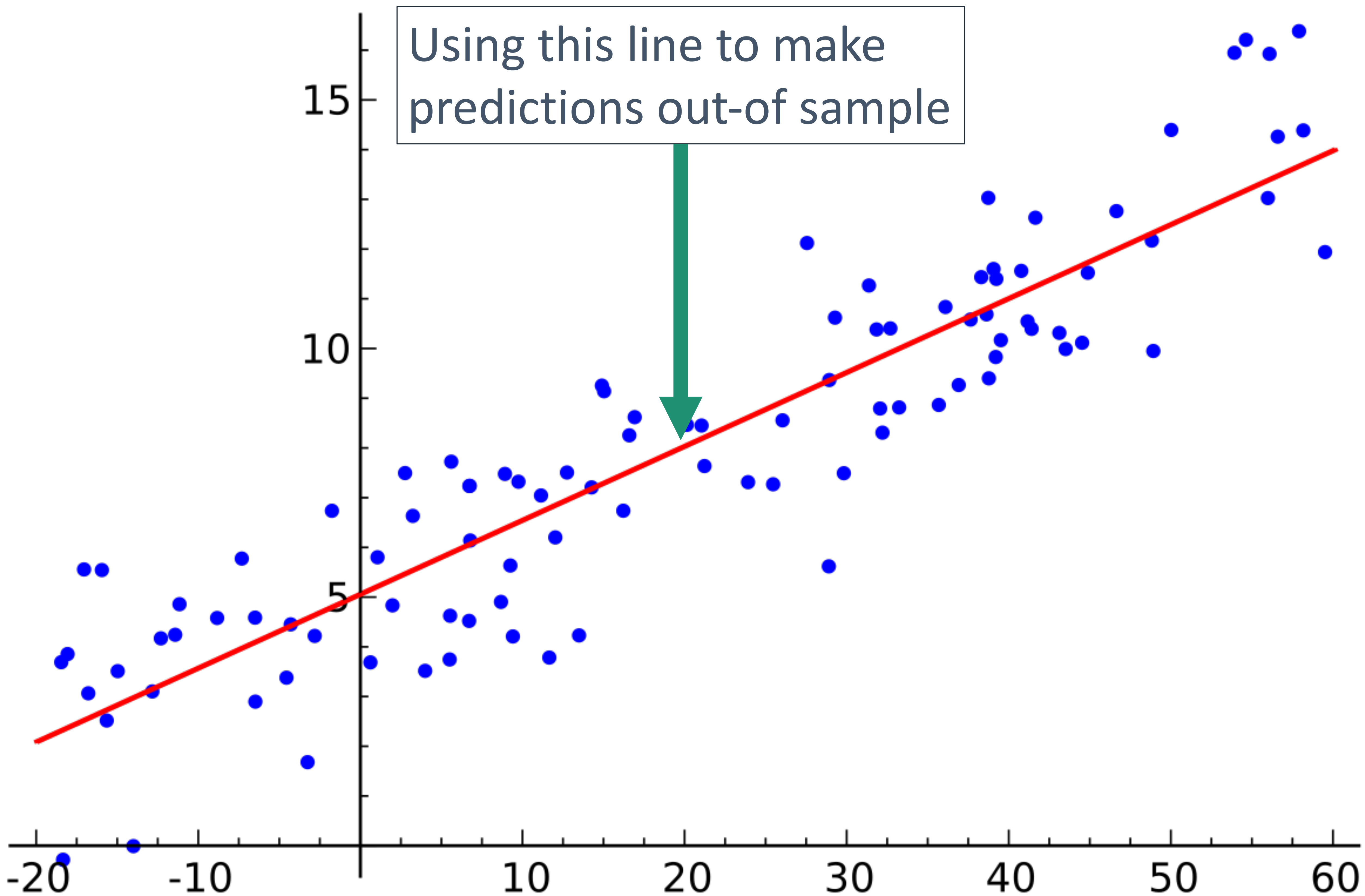
Regression-Based Methods

Classification vs. Regression

- Essentially: labelling outcomes vs. assigning outcomes a numeric value
- “Regression” not always used in the same way it’s used in statistics
  - Typically means “quantitative outcome”
- Fundamental “algorithm”:
  - Linear regression (typically least squares)
  - Logistic regression (typically log-odds, etc.)

Regression Models to Predict Quantitative Outcomes

- Least-squares regression: finding line of "best fit" that minimizes the sum of the squared residuals.

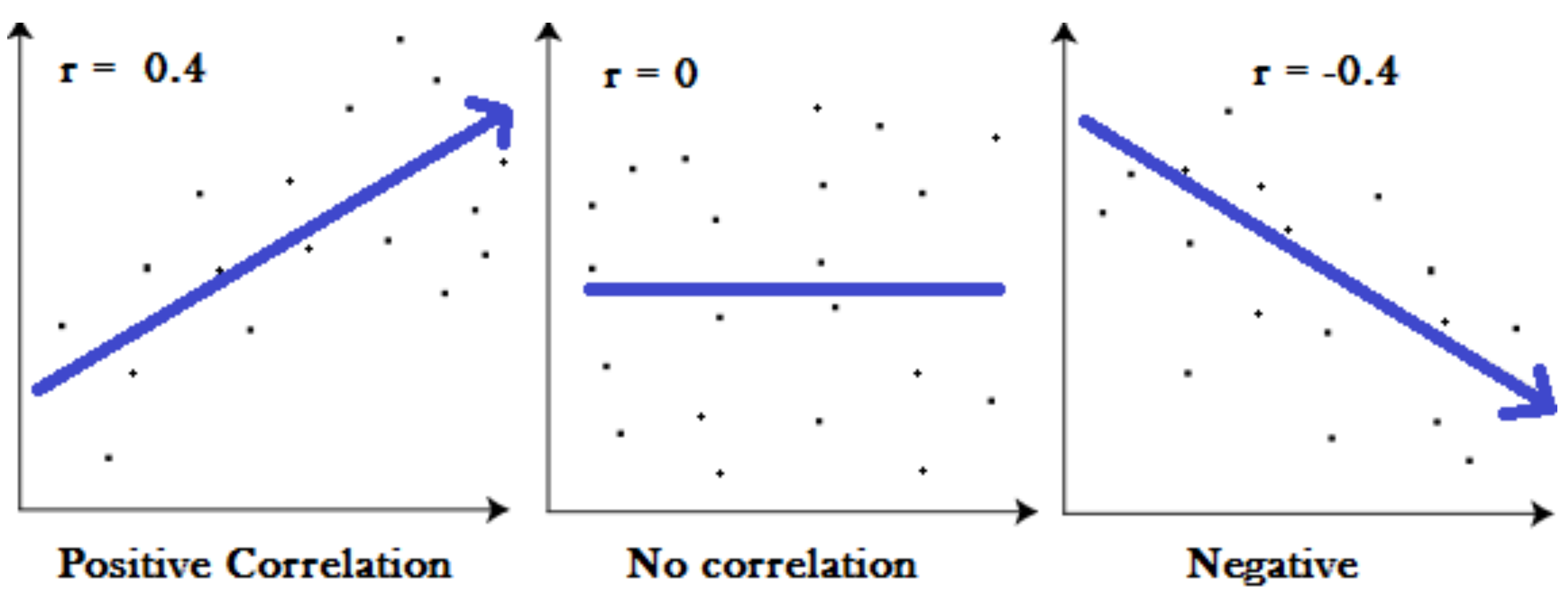


Ways to Evaluate Model

- Two primary ways to evaluate regression-based models
  - 1) Root Mean Squared Error (RMSE)
    - Very common in machine learning applications
    - Quantifies how much the prediction-actual results differ

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (Predicted_i - Actual_i)^2}{N}}$$

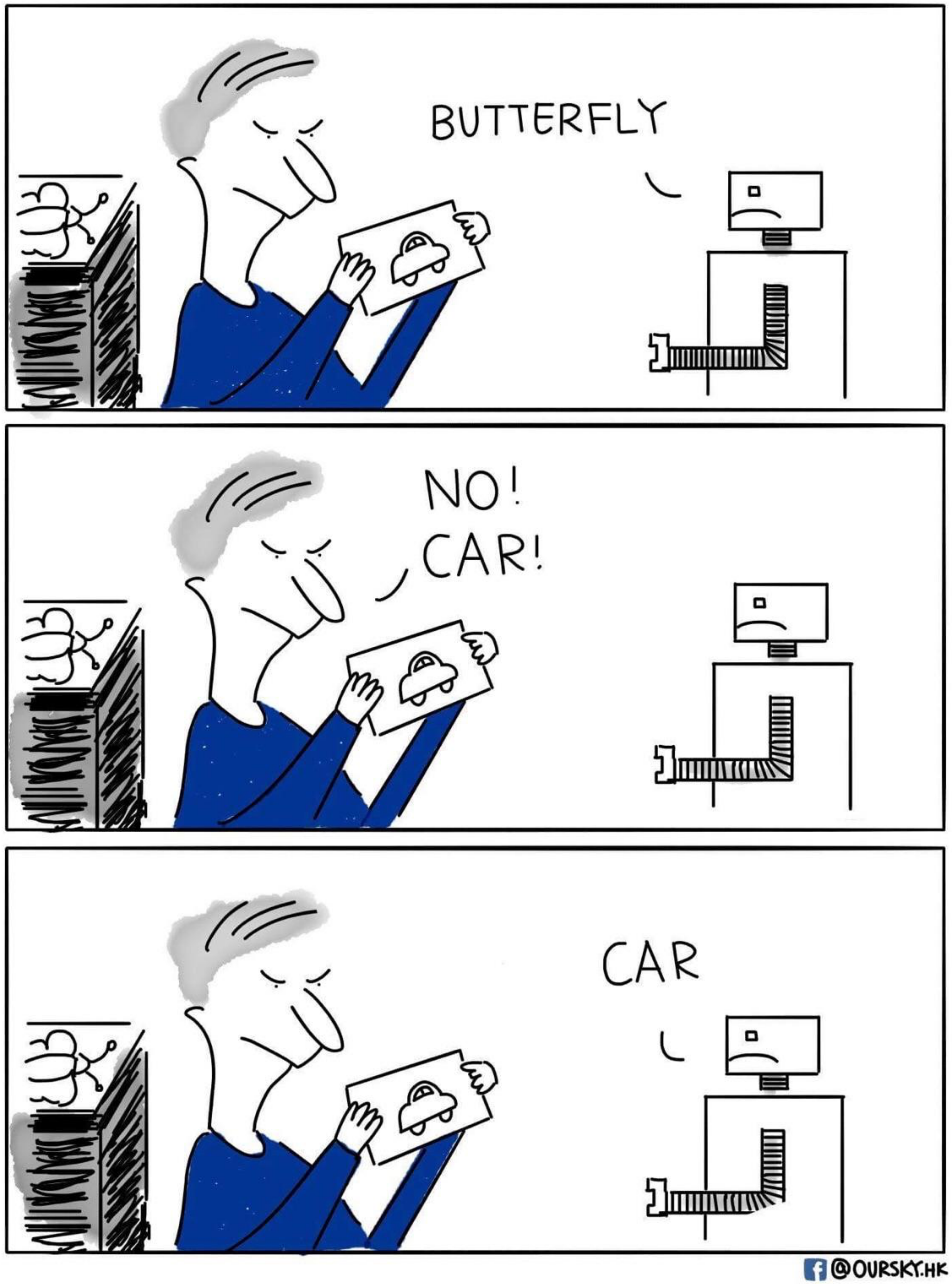
  - 2) Correlation
    - Effect size measure, no units
    - Frequently more interpretable for scientific audiences



Time to open R!

- Steps:
  - 1) Split data into training and testing datasets
  - 2) Construct linear model on the training dataset
    - Using `lm` command in R.
  - 3) Use linear model from training to make predictions on the test dataset
    - Using `predict` command in R

- Walk through example by me, then you will try

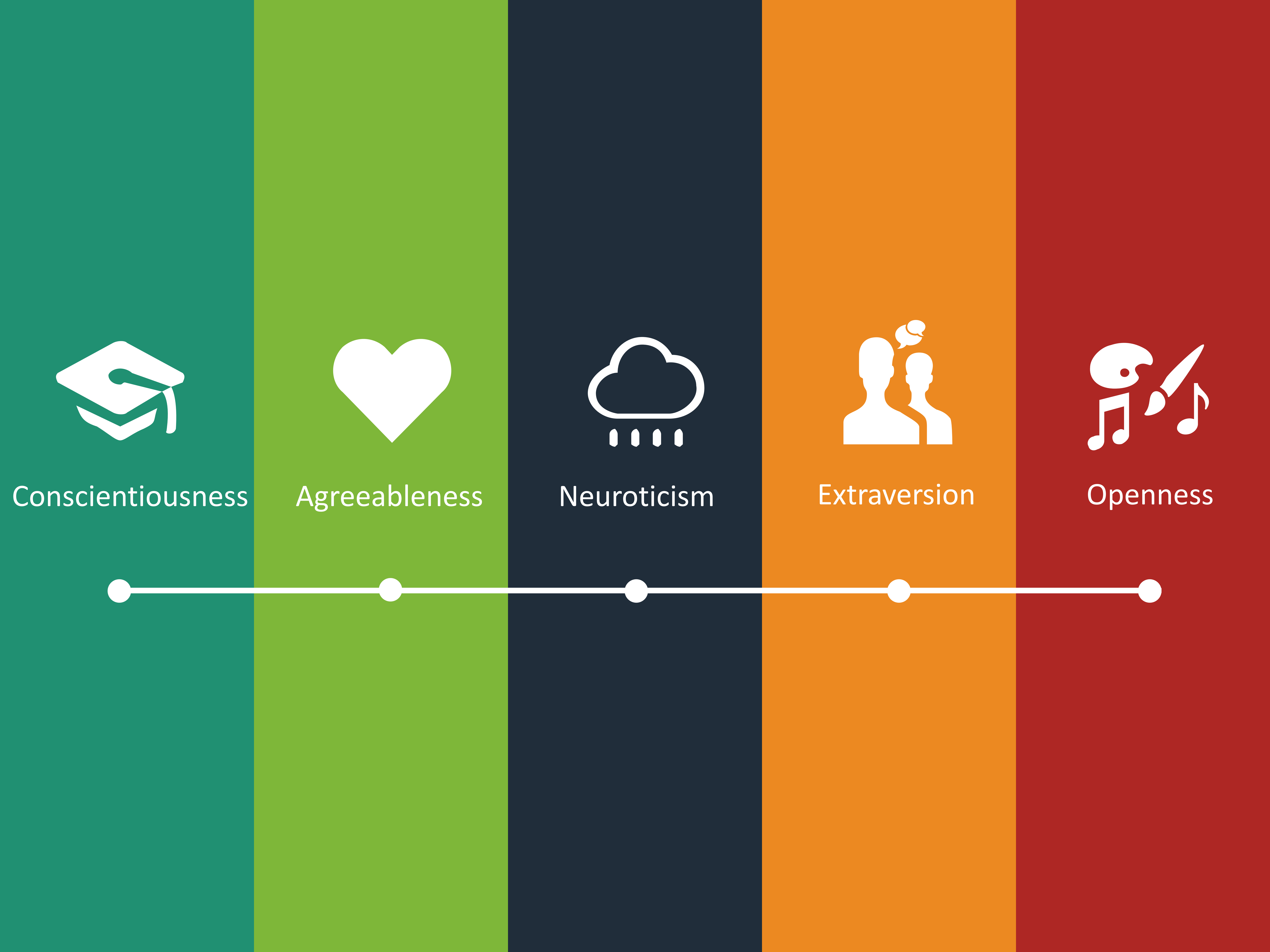

BUTTERFLY
NO! CAR!
CAR
@OURSKY.HK

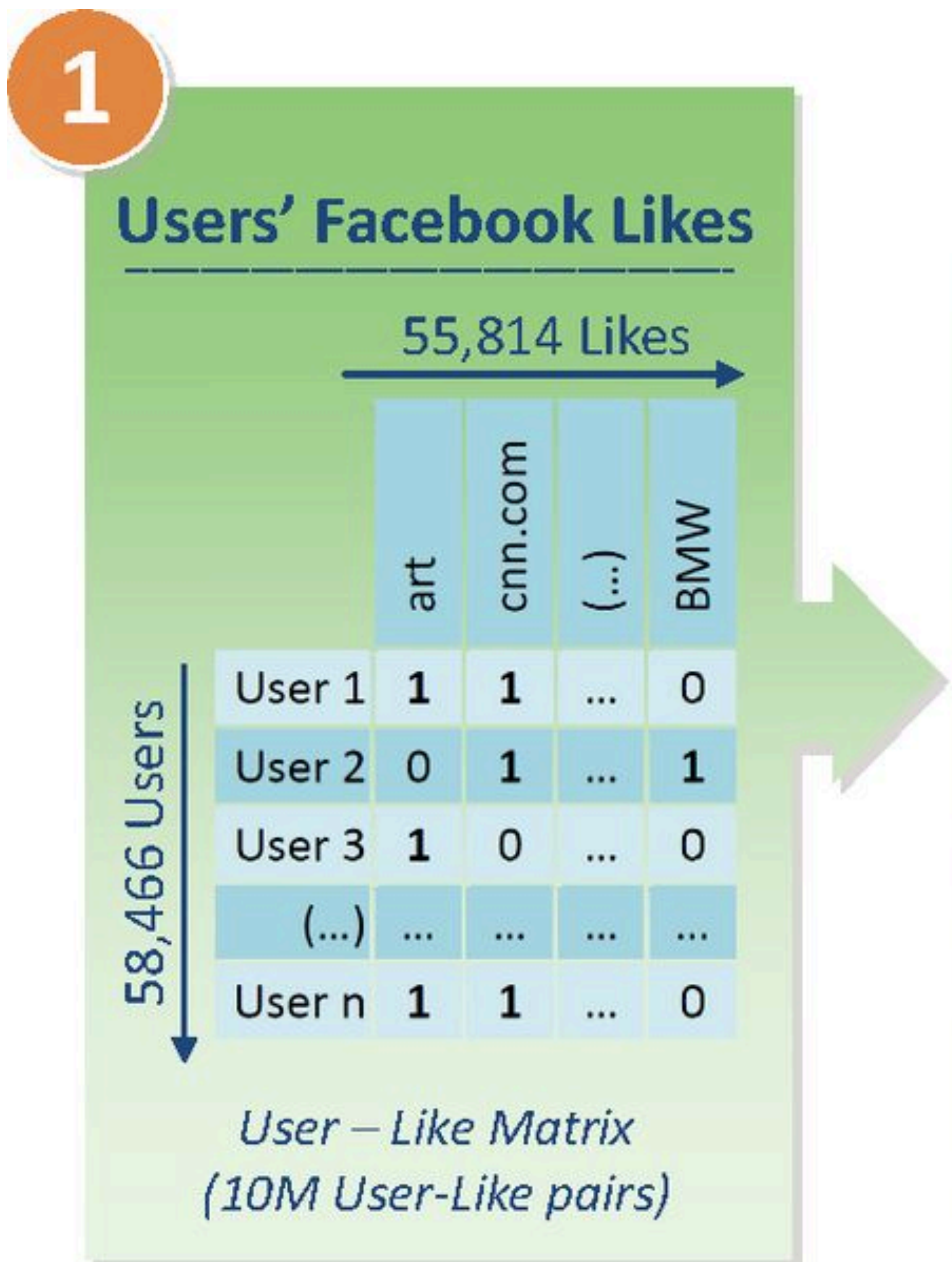

LinkedIn
LiVEJOURNAL
foursquare
vimeo
Insta
YouTube
facebook
Blogger
my
twitter
flickr
tumblr.
Like

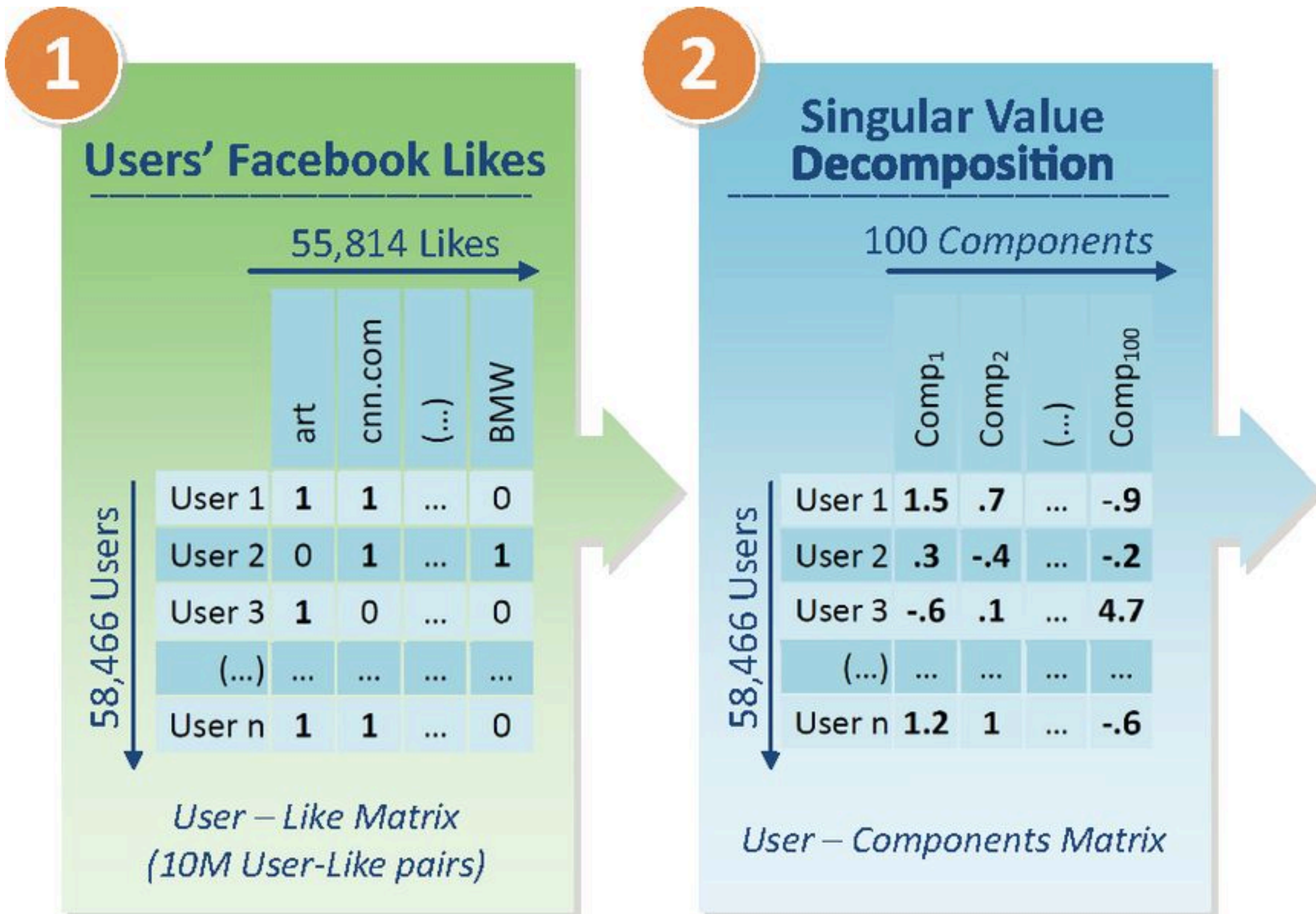
1) Collect

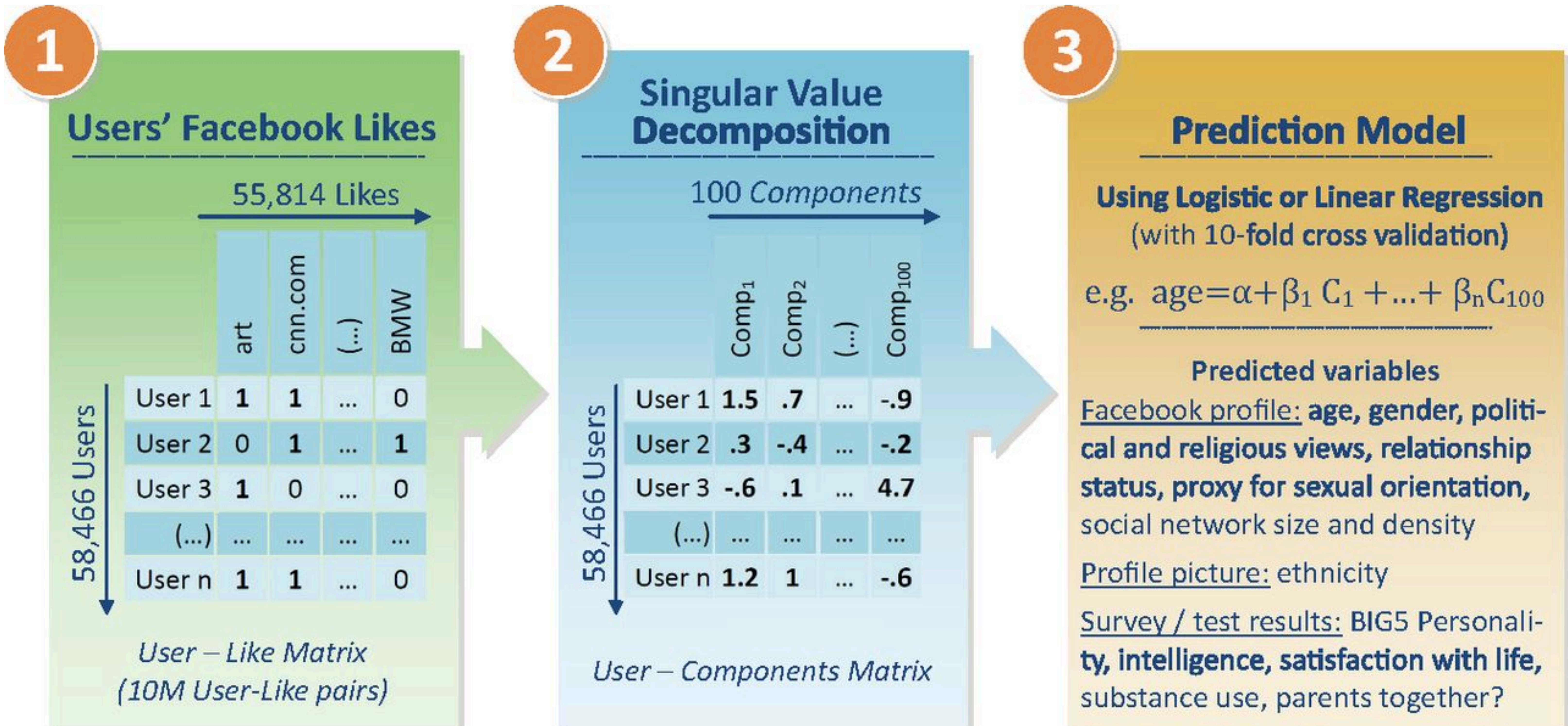
2) Correlate

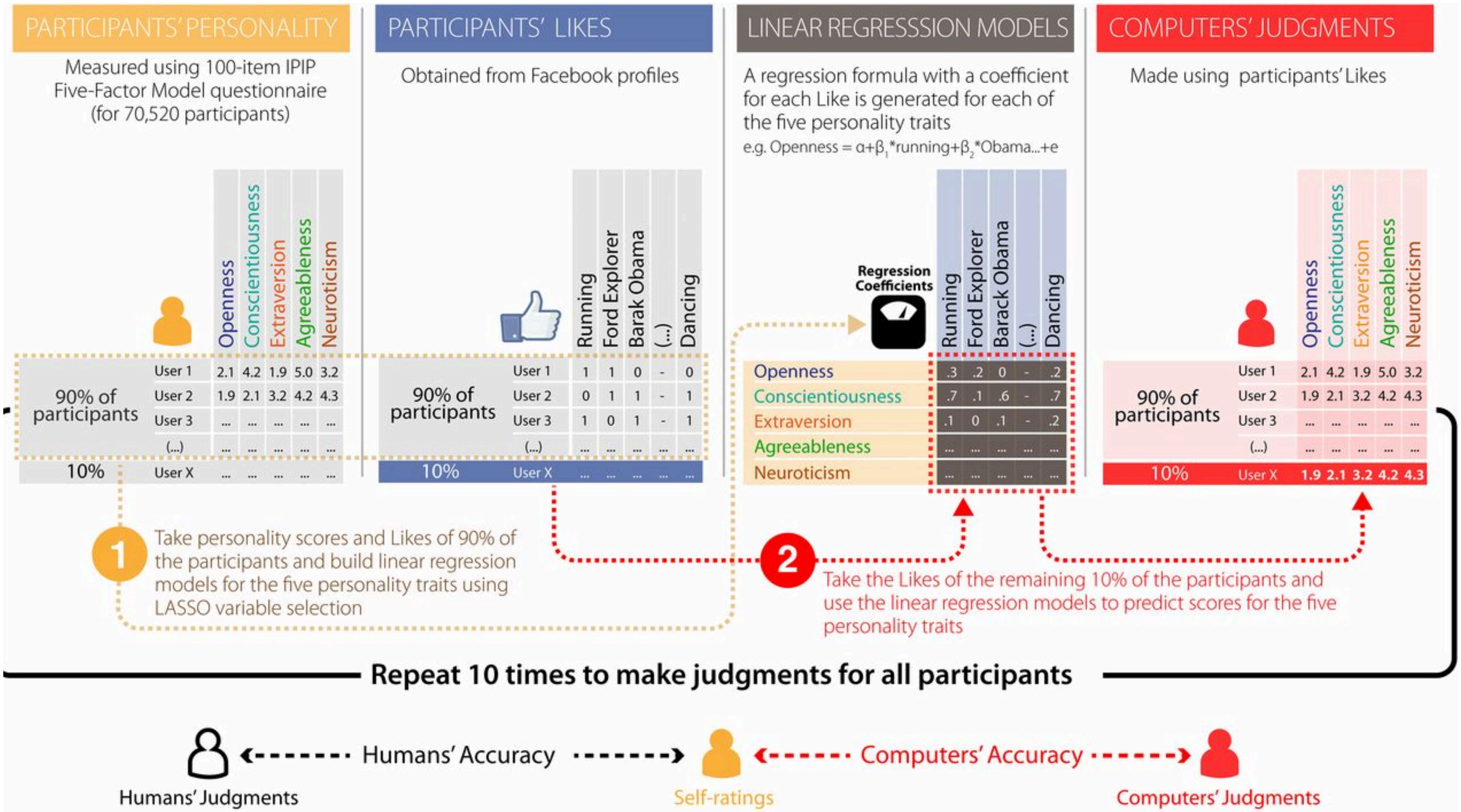
3) Predict


Conscientiousness
Agreeableness
Neuroticism
Extraversion
Openness

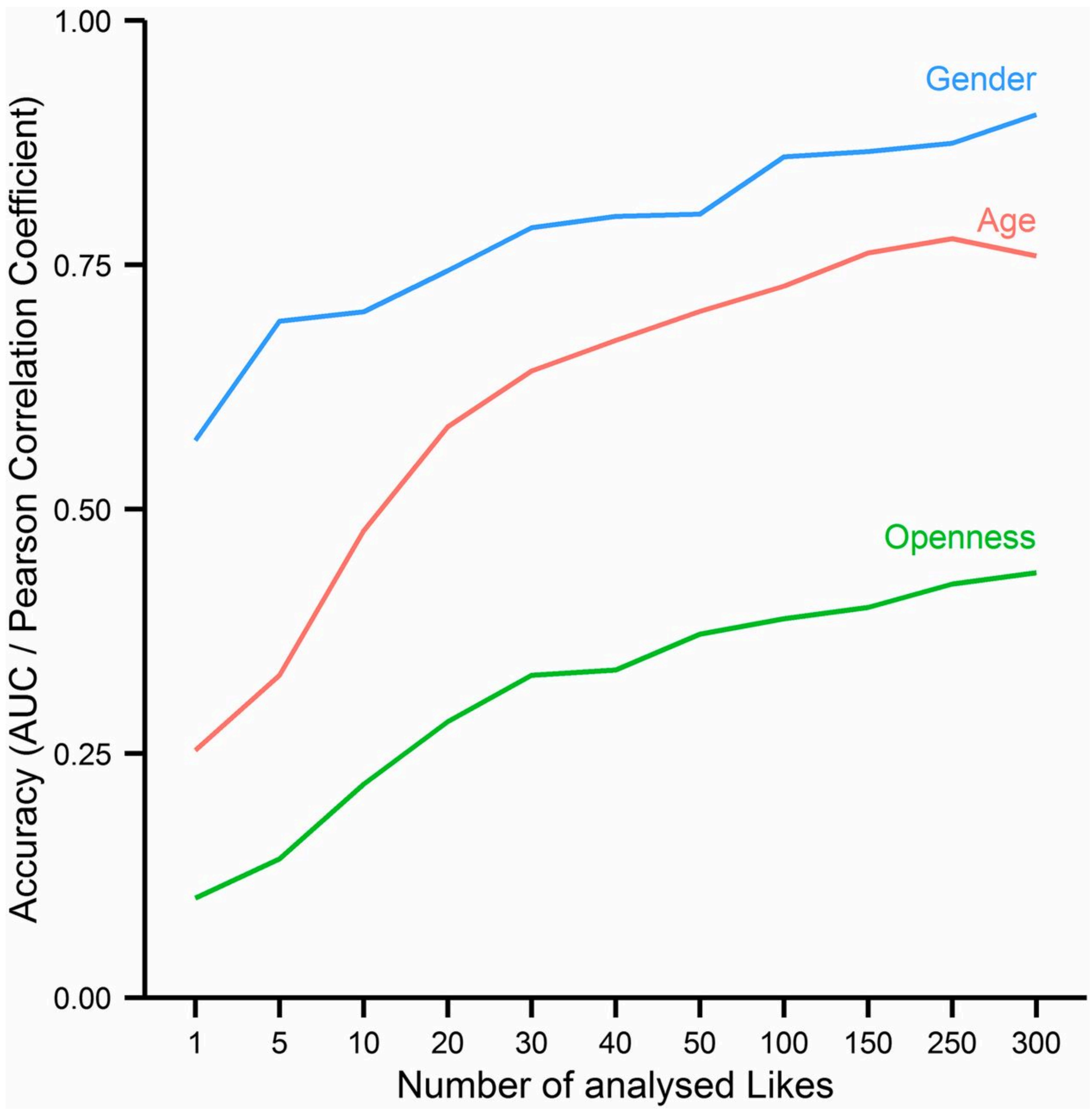

1
Users' Facebook Likes
55,814 Likes
art
cnn.com
(...)
BMW
58,466 Users
User 1 1 1 ... 0
User 2 0 1 ... 1
User 3 1 0 ... 0
(...) ... ...
User n 1 1 ... 0
User – Like Matrix
(10M User-Like pairs)


1
Users' Facebook Likes
55,814 Likes
art
cnn.com
(...)
BMW
58,466 Users
User 1 1 1 ... 0
User 2 0 1 ... 1
User 3 1 0 ... 0
(...) ... ...
User n 1 1 ... 0
User – Like Matrix
(10M User-Like pairs)
2
Singular Value Decomposition
100 Components
Comp1
Comp2
(...)
Comp100
58,466 Users
User 1 1.5 .7 ... -.9
User 2 .3 -.4 ... -.2
User 3 -.6 .1 ... 4.7
(...) ... ...
User n 1.2 1 ... -.6
User – Components Matrix


1
Users' Facebook Likes
55,814 Likes
art
cnn.com
(...)
BMW
58,466 Users
User 1 1 1 ... 0
User 2 0 1 ... 1
User 3 1 0 ... 0
(...) ... ...
User n 1 1 ... 0
User – Like Matrix
(10M User-Like pairs)
2
Singular Value Decomposition
100 Components
$Comp_1$
$Comp_2$
(...)
$Comp_{100}$
58,466 Users
User 1 1.5 .7 ... -.9
User 2 .3 -.4 ... -.2
User 3 -.6 .1 ... 4.7
(...) ... ...
User n 1.2 1 ... -.6
User – Components Matrix
3
Prediction Model
Using Logistic or Linear Regression
(with 10-fold cross validation)
e.g. $age=\alpha+\beta_1 C_1 +...+ \beta_n C_{100}$
Predicted variables
Facebook profile: age, gender, political and religious views, relationship status, proxy for sexual orientation, social network size and density
Profile picture: ethnicity
Survey / test results: BIG5 Personality, intelligence, satisfaction with life, substance use, parents together?



PARTICIPANTS' PERSONALITY
Measured using 100-item IPIP Five-Factor Model questionnaire (for 70,520 participants)
Openness
Conscientiousness
Extraversion
Agreeableness
Neuroticism
90% of participants
User 1 2.1 4.2 1.9 5.0 3.2
User 2 1.9 2.1 3.2 4.2 4.3
User 3
(...)
10%
User X
PARTICIPANTS' LIKES
Obtained from Facebook profiles
Running
Ford Explorer
Barak Obama
(...)
Dancing
User 1 1 1 0 - 0
User 2 0 1 1 - 1
User 3 1 0 1 - 1
LINEAR REGRESSSION MODELS
A regression formula with a coefficient for each Like is generated for each of the five personality traits
e.g. Openness = α+β1*running+β2*Obama...+e
Regression Coefficients
Barack Obama
Openness .3 .2 0 - .2
Conscientiousness .7 .1 .6 - .7
Extraversion .1 0 .1 - .2
Agreeableness
Neuroticism
COMPUTERS' JUDGMENTS
Made using participants' Likes
User X 1.9 2.1 3.2 4.2 4.3
1
Take personality scores and Likes of 90% of the participants and build linear regression models for the five personality traits using LASSO variable selection
2
Take the Likes of the remaining 10% of the participants and use the linear regression models to predict scores for the five personality traits
Repeat 10 times to make judgments for all participants
Humans' Accuracy
Computers' Accuracy
Humans' Judgments
Self-ratings
Computers' Judgments



1.00
0.75
0.50
0.25
0.00
Accuracy (AUC / Pearson Correlation Coefficient)
Gender
Age
Openness
1
5
10
20
30
40
50
100
150
250
300
Number of analysed Likes

Key take-aways from Regression-Based Methods

- Same methods from classical statistics
  - But with an emphasis on prediction
  - You can use these methods with little change to your current research program!
- Can use numeric outcomes (”regression”) or binary/factor (“classification”)
- Parametric in nature
- Assume underlying linearity between predictors and outcome
- Utilize cross-validation to reduce likelihood of overfitting
- Dirty secret of machine learning….frequently these methods outperform the “more complicated” methods

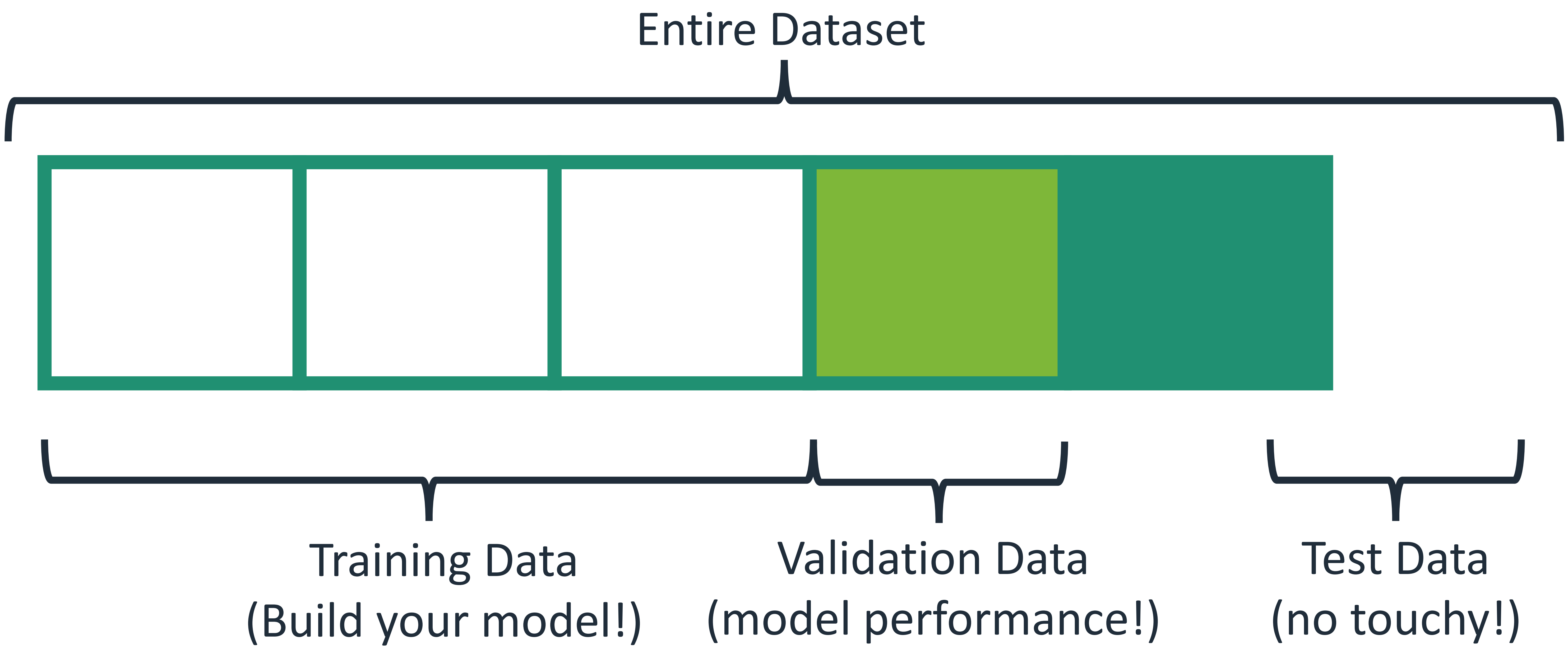
Key ML tenet #2: Bias-Variance Trade-off

Competing desires:

- Fit model to make accurate predictions for given data
- Vs. Fit model to predict future data

How do we solve this?

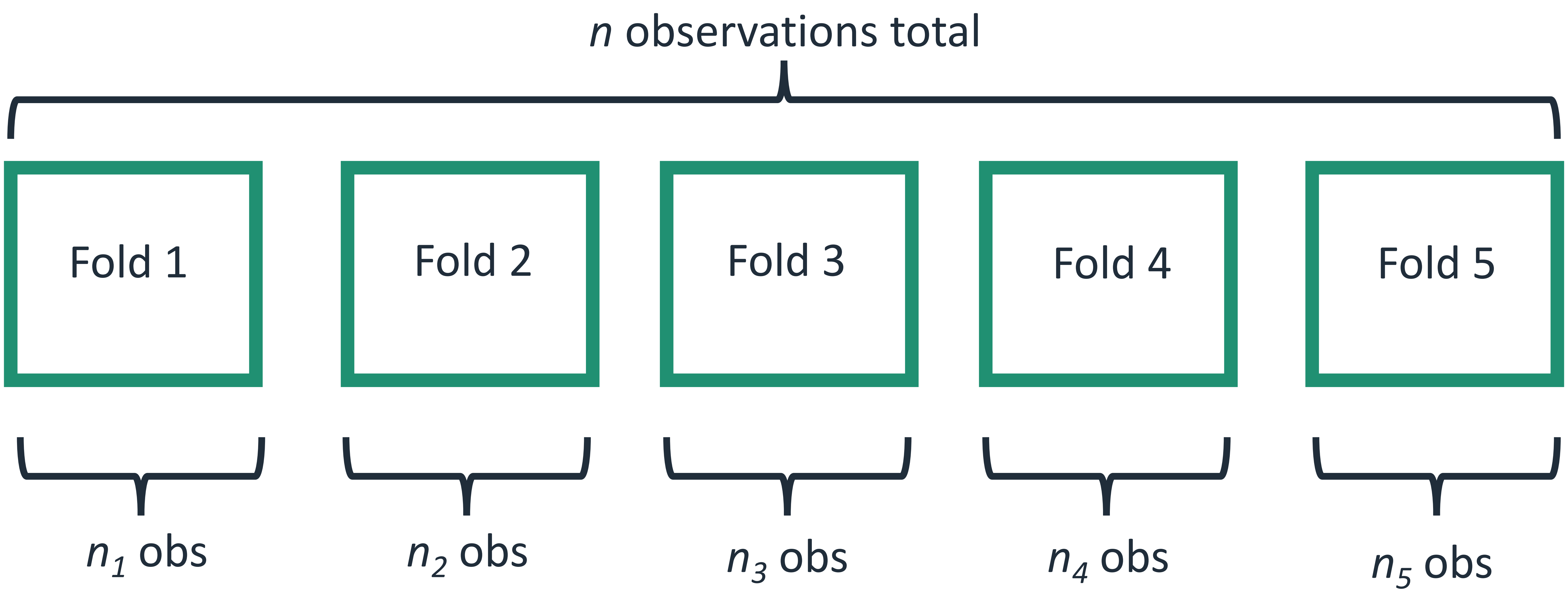
- Focus on *Test Set* performance



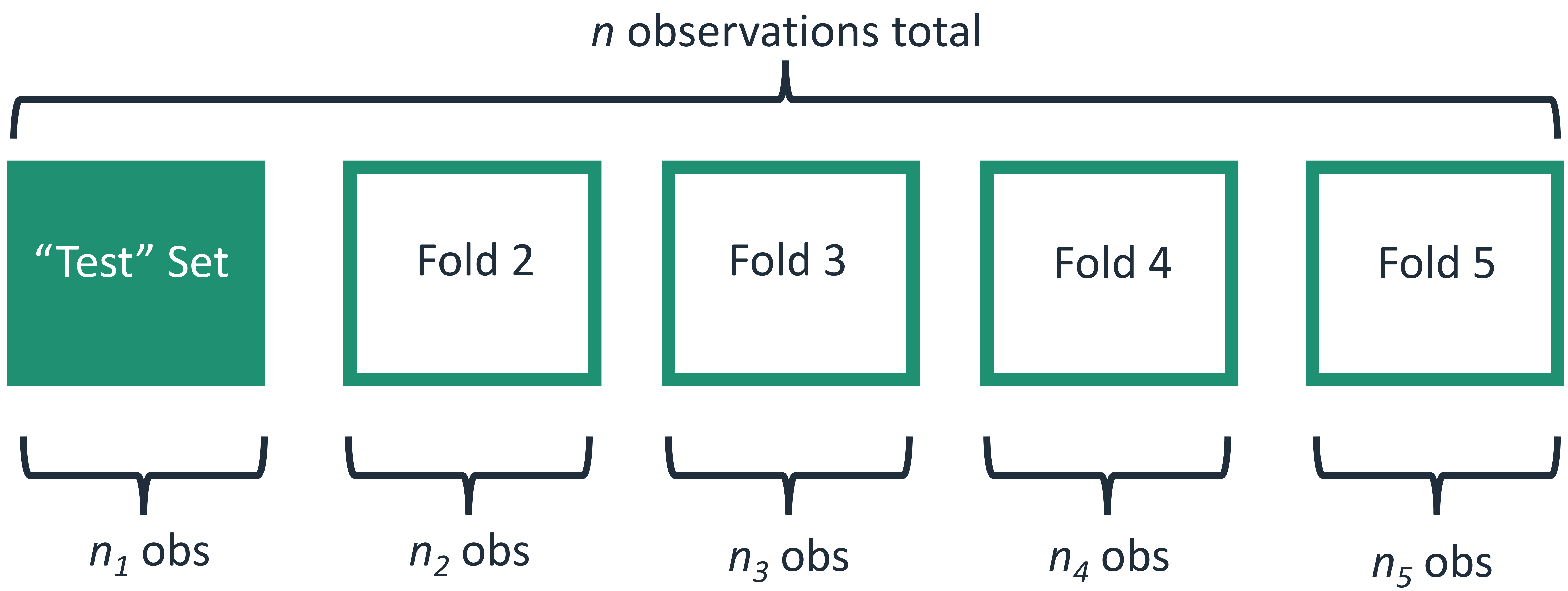
Extension of test-set separation: Cross-Validation

- Separation of test set is great: completely unimplicated data, more confidence in generalizability of results.
- BUT!! You restrict data used in training.
  - If 100 observations, 40 as test set → only 60 for training
  - In general, want as many in training as possible while still leaving enough for test dataset.
- Solution? Cross-validation!

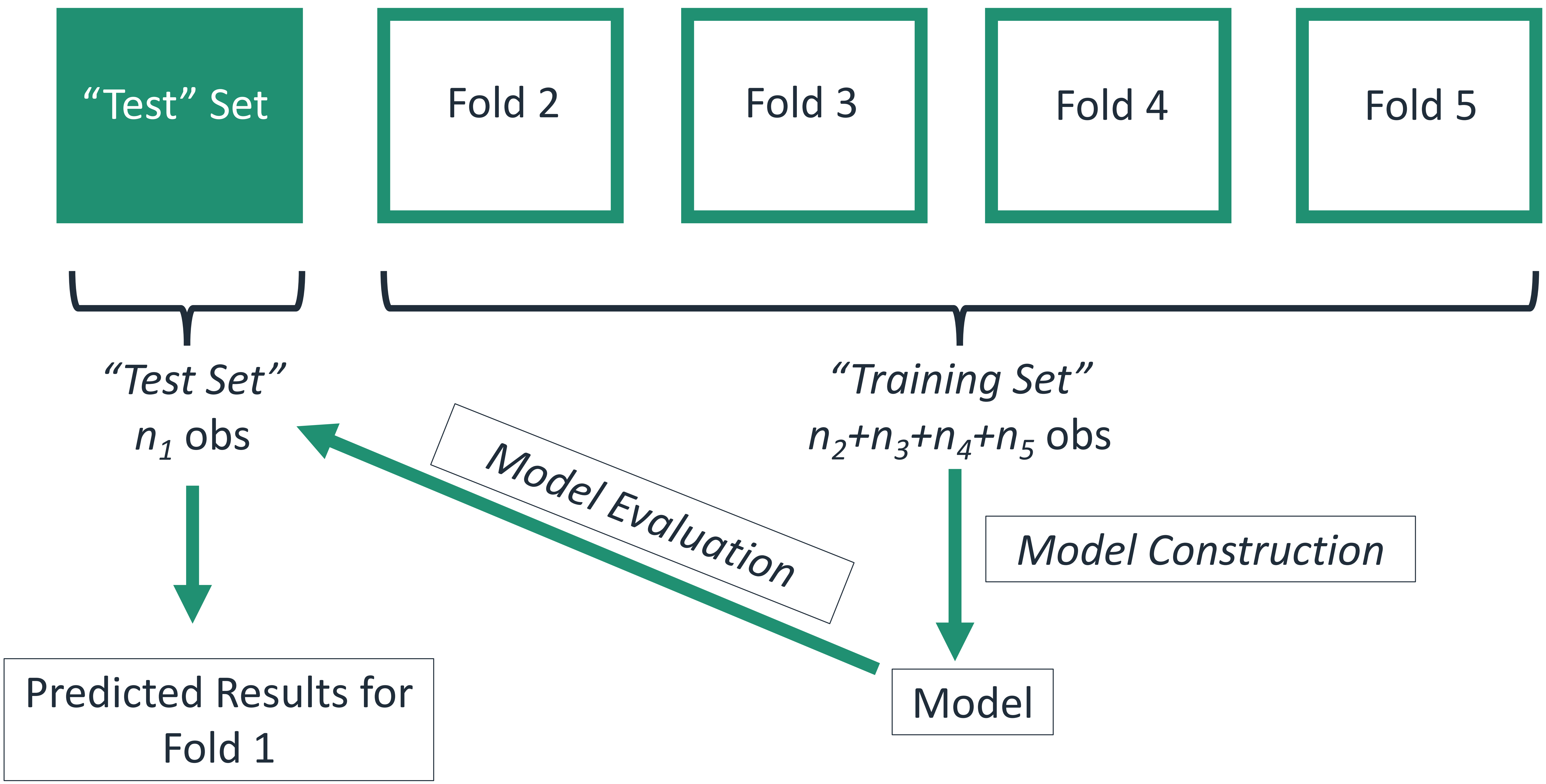
Cross-Validation (5-fold)



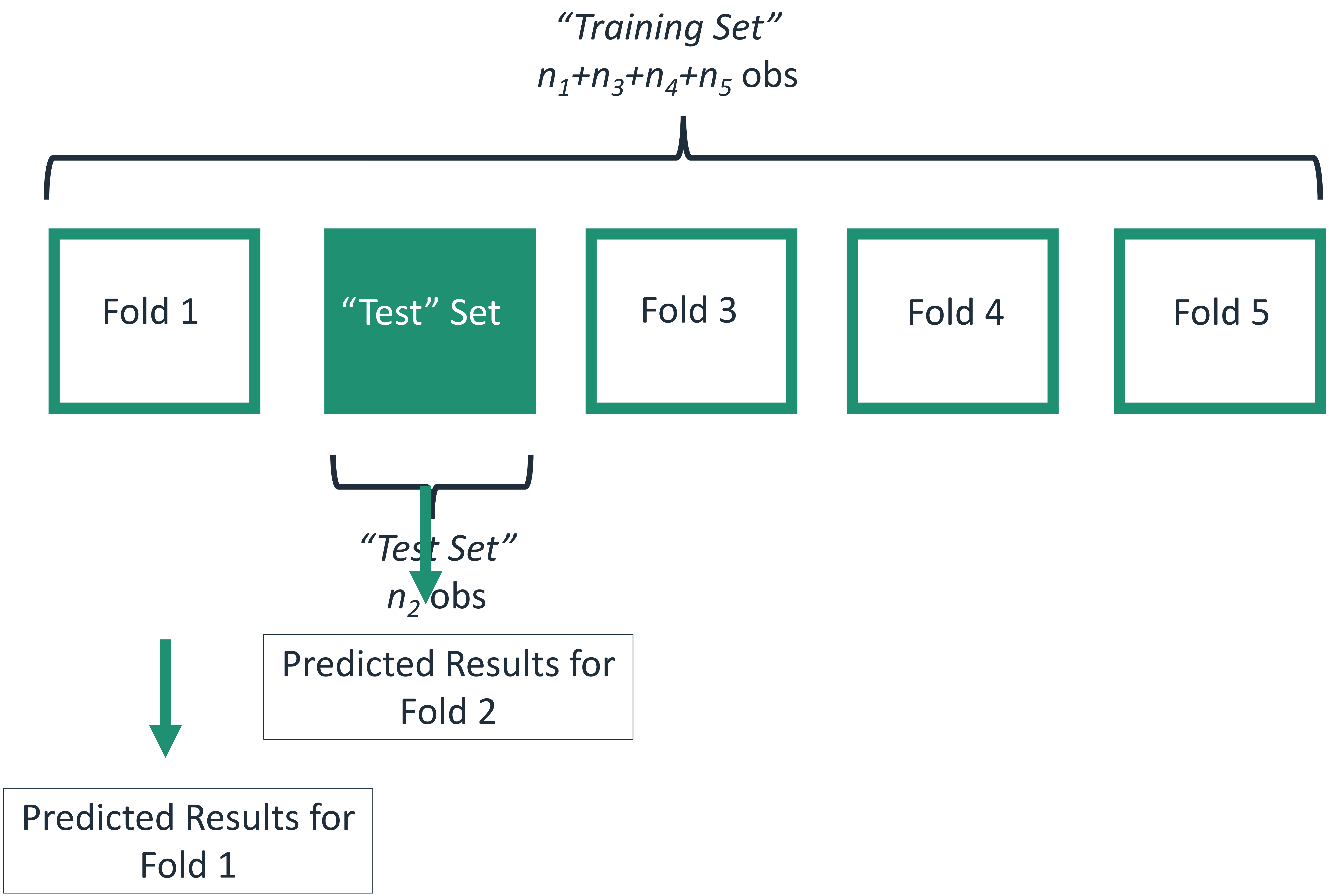
Cross-Validation (5-fold)



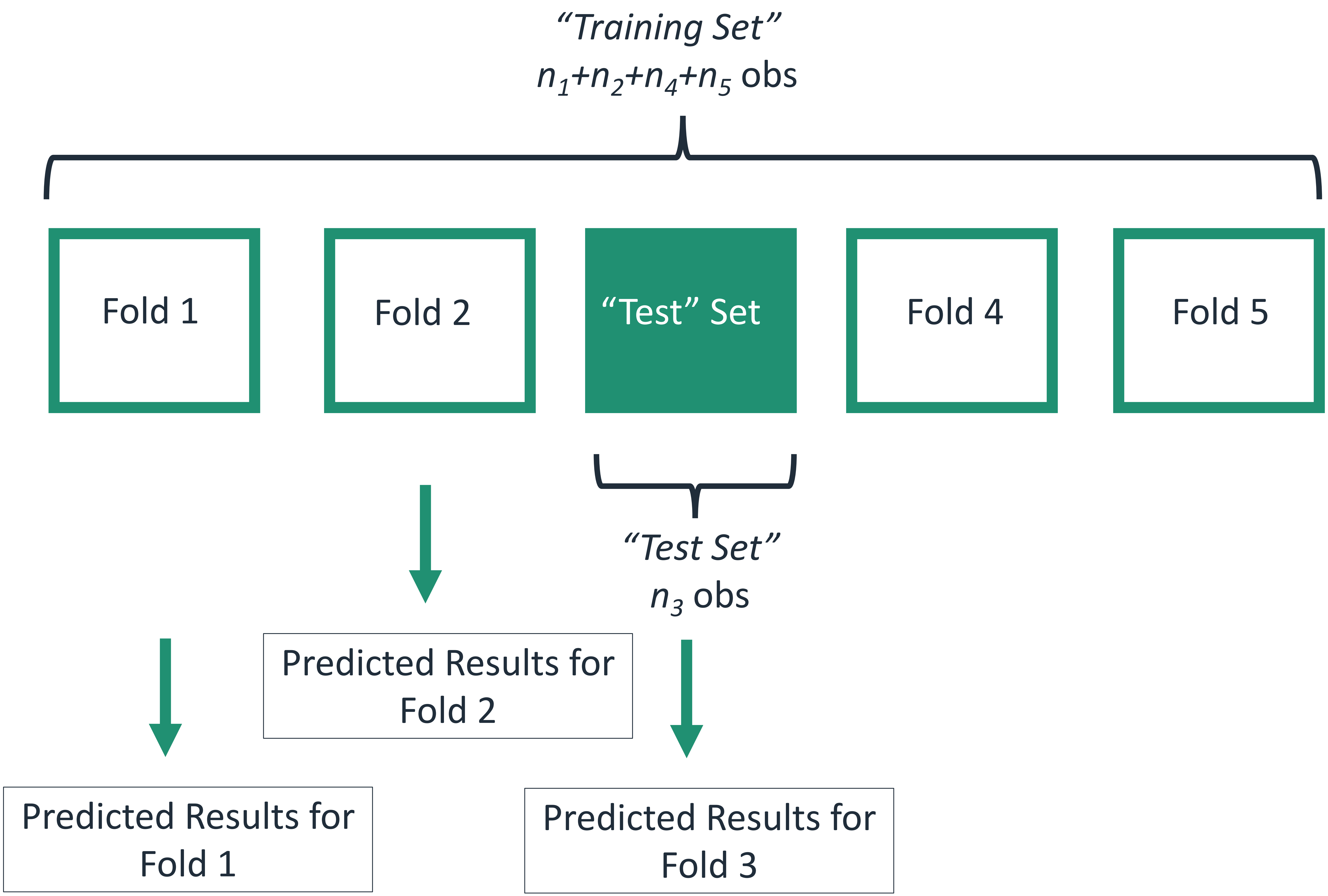
Cross-Validation (5-fold)



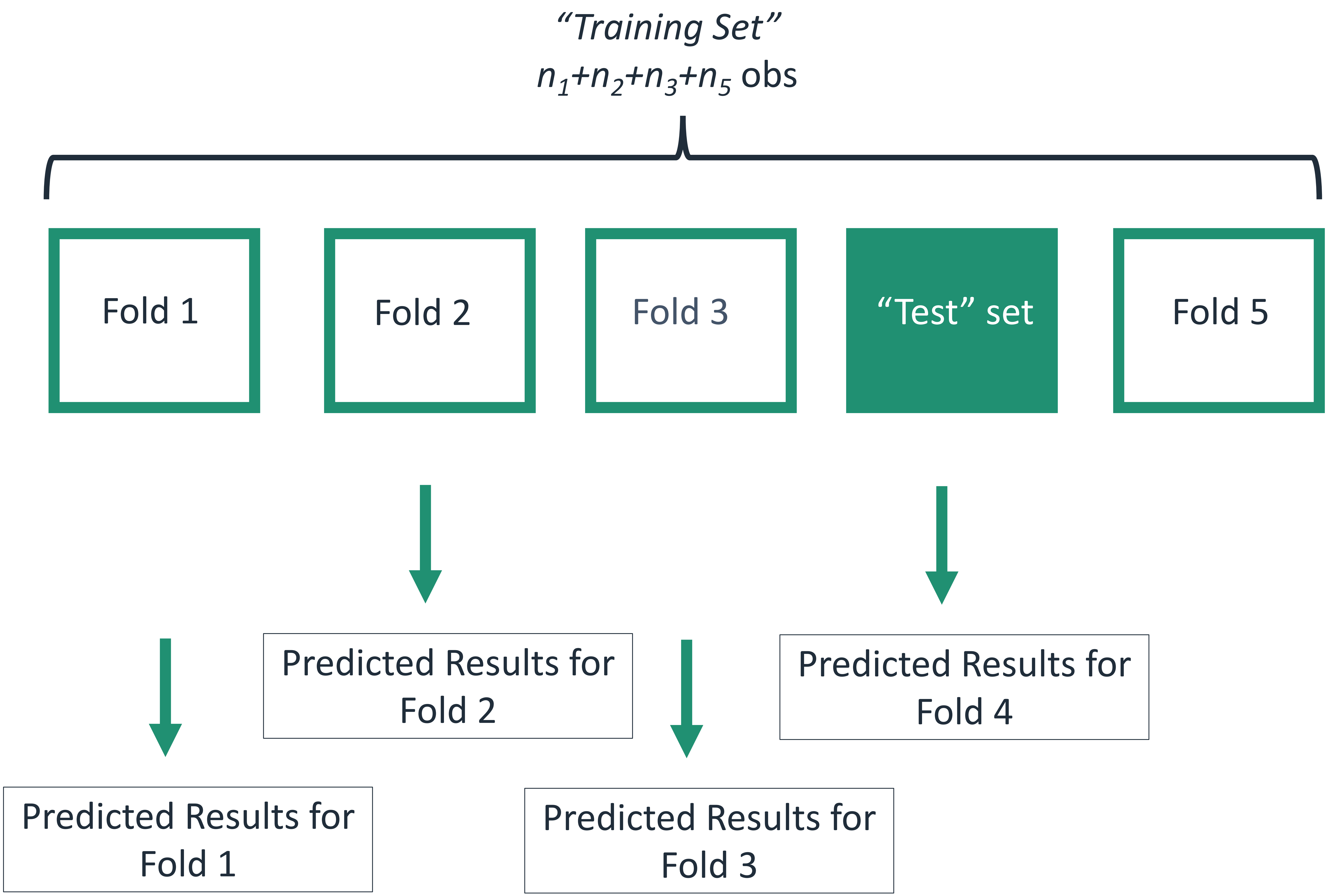
Cross-Validation (5-fold)



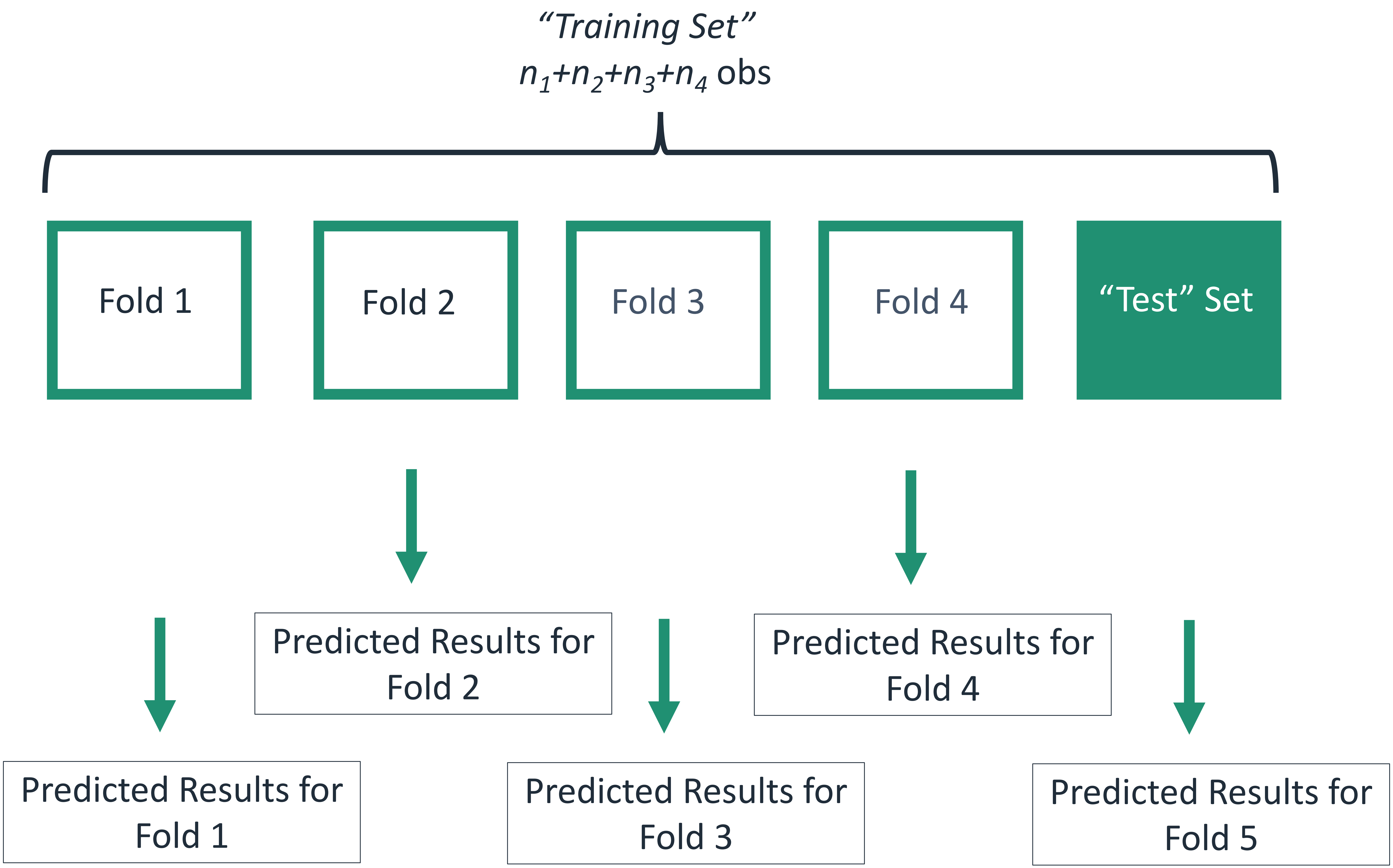
Cross-Validation (5-fold)



Cross-Validation (5-fold)

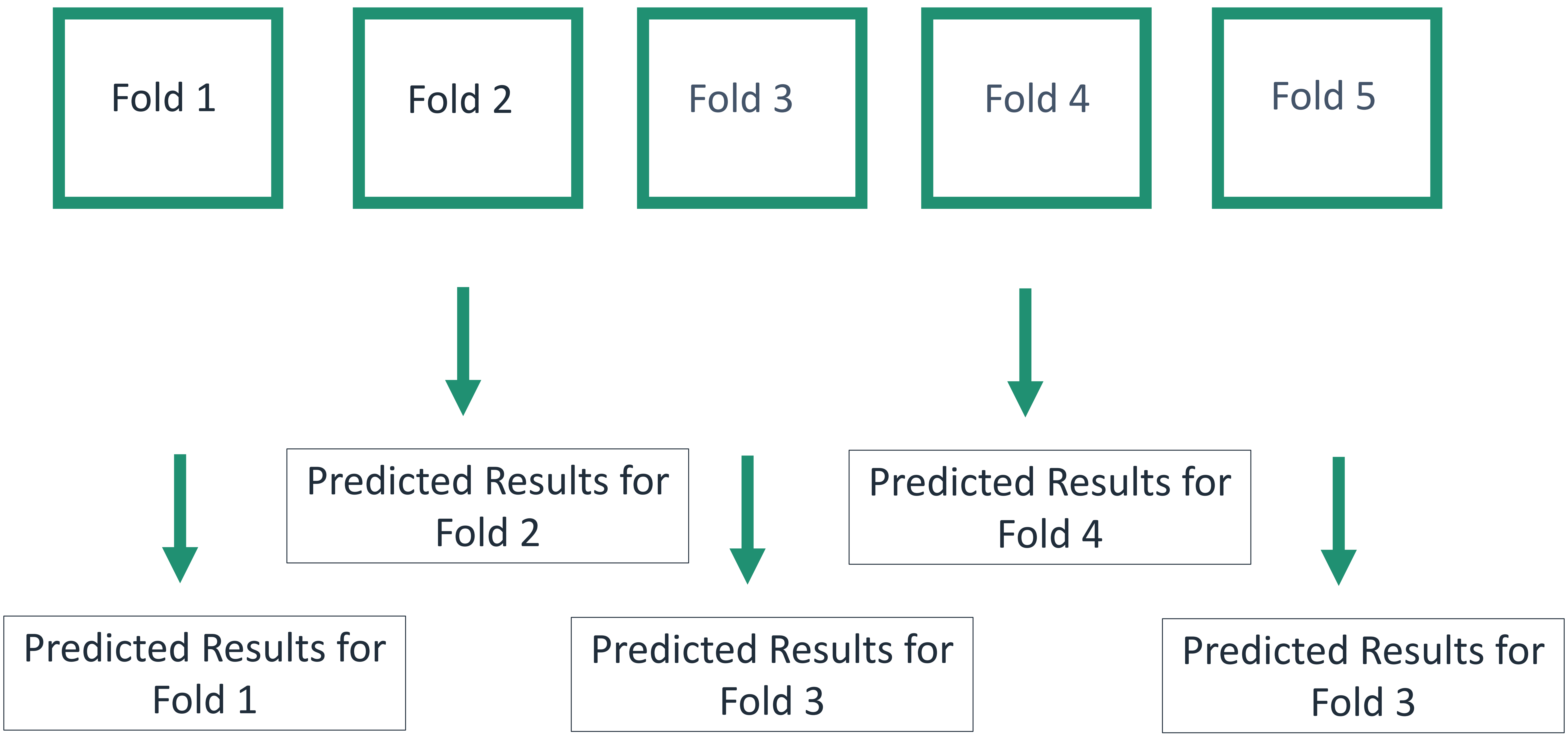


Cross-Validation (5-fold)

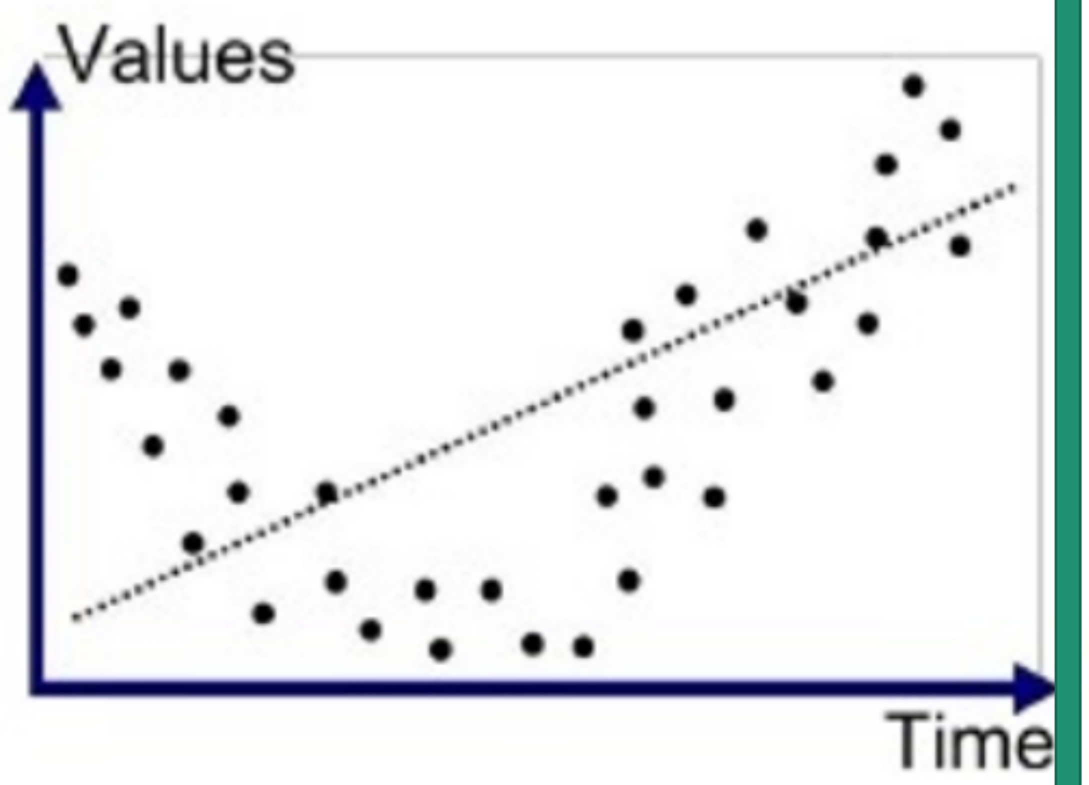


Cross-Validation (5-fold)

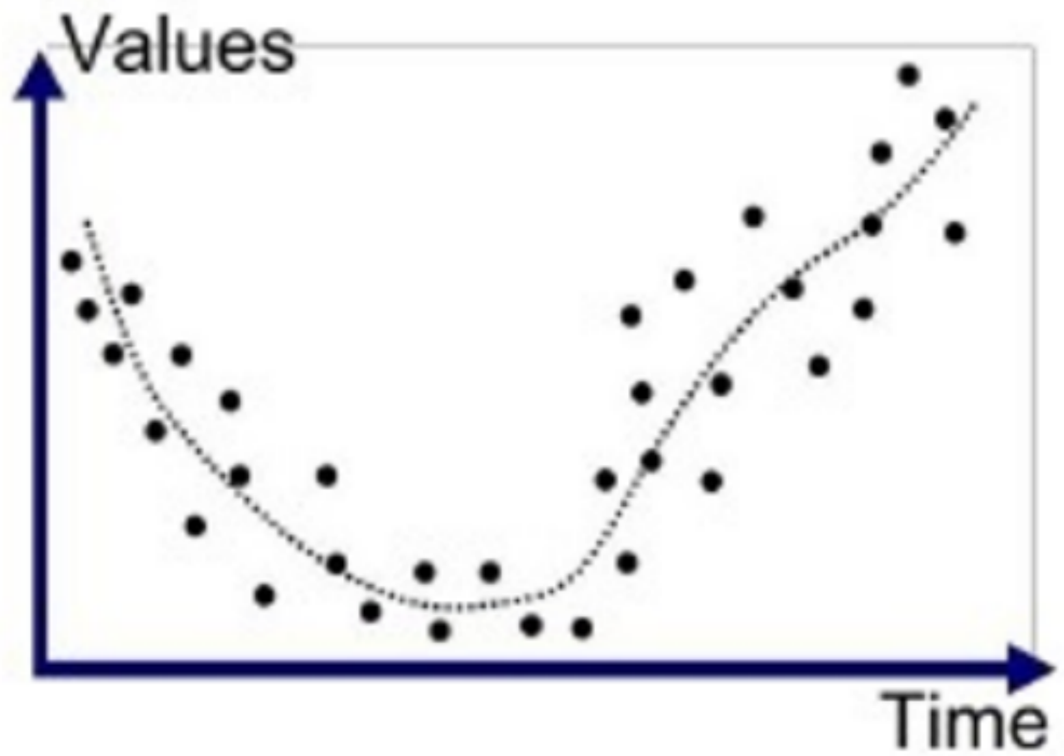
We have predicted results for each fold based on the observations in all the other folds (e.g., fold 5 based on model trained on $n_1+n_2+n_3+n_4$ observations)



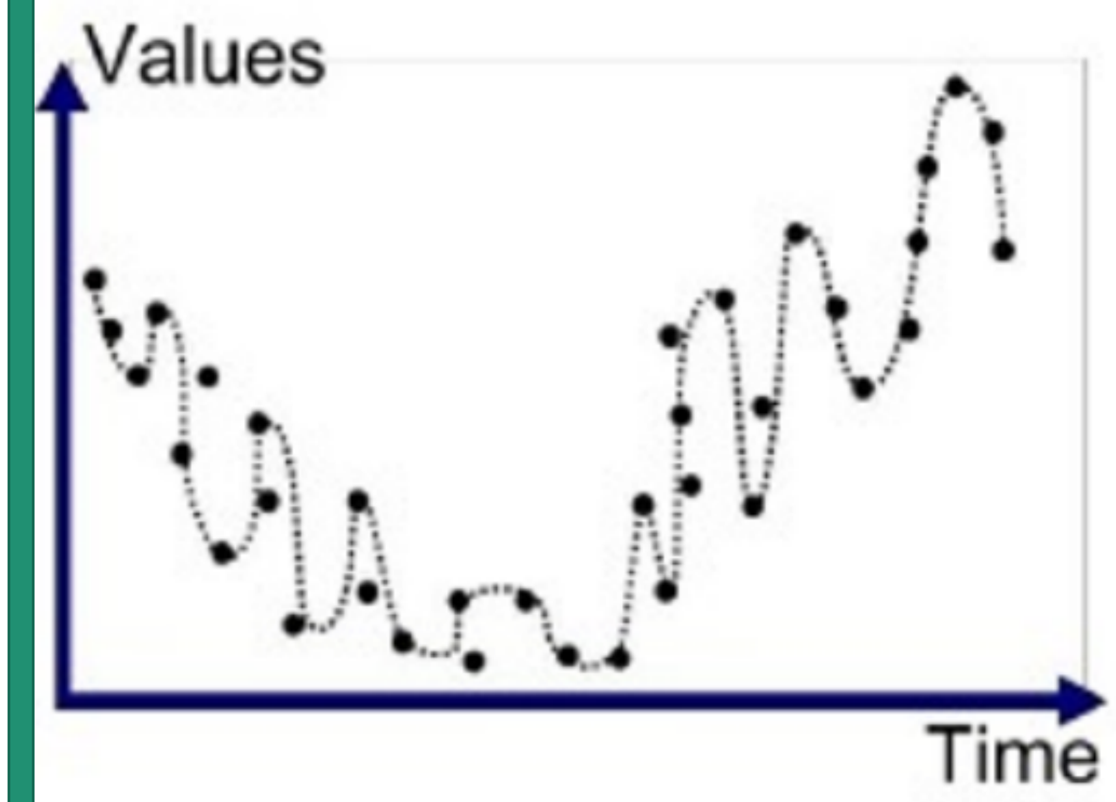
Cross-Validation helps avoid overfitting



Underfit



Good Fit/Robust



Overfit

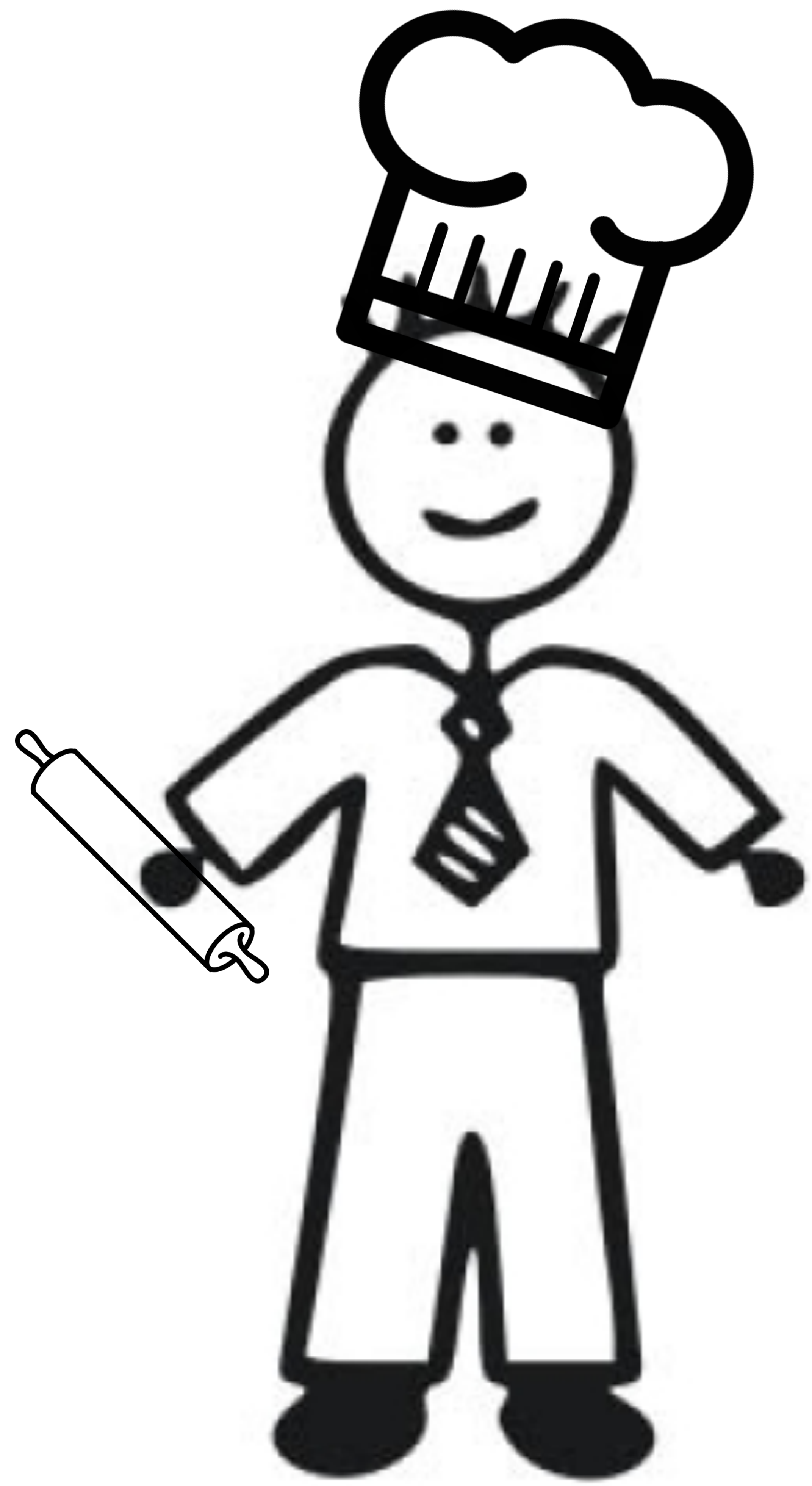
Time for Cross-Validation Coding

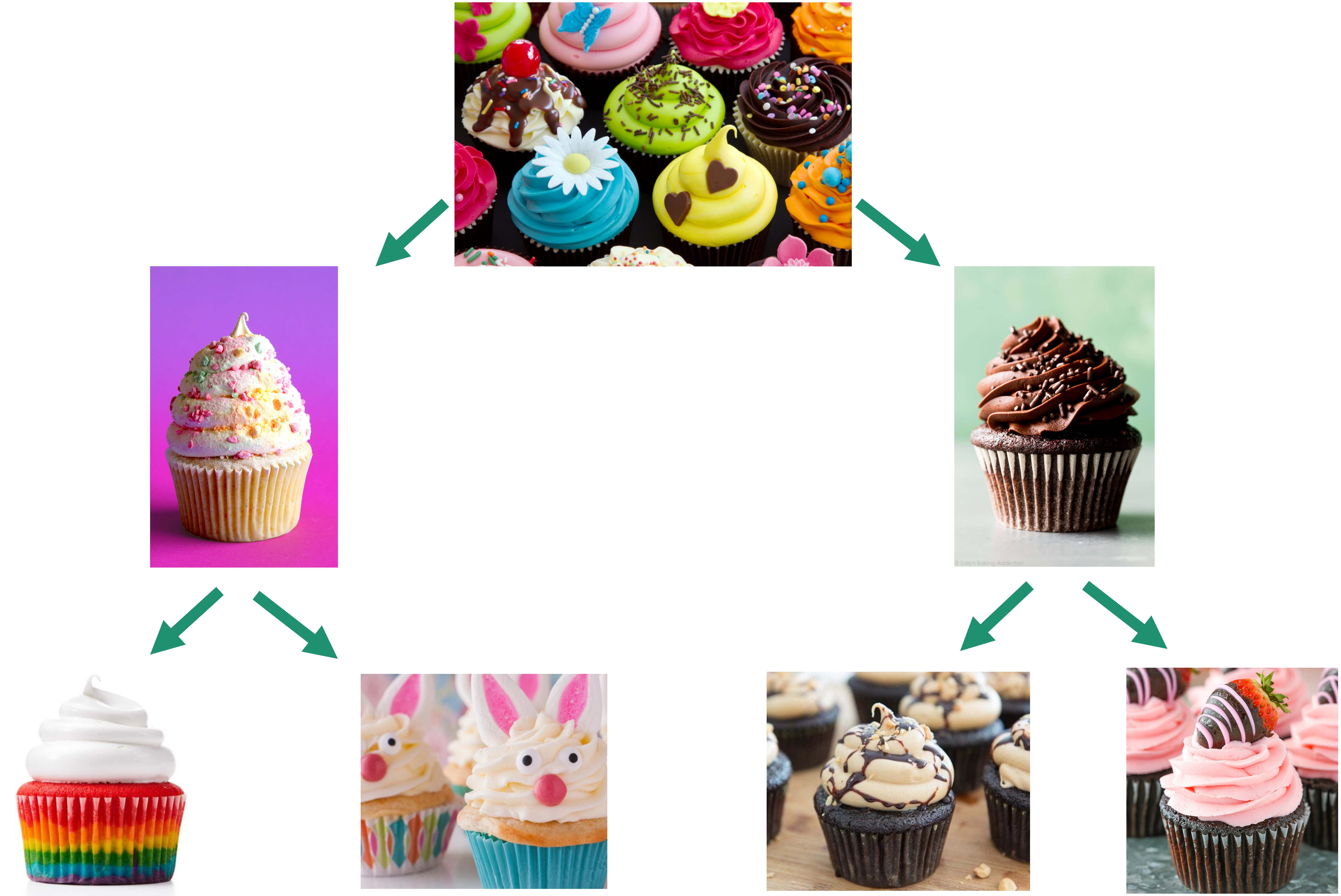
- Steps:
  - 1) Split training data into five “folds”
  - 2) Assign one ”fold” to be temporary test data
  - 3) Construct model and test on this fold getting predicted values
  - 4) Save these predicted values
  - 5) Repeat above process four more times, making each fold the test dataset in turn.

- Walk through example by me, then you will try

Time for Break

Decision Trees



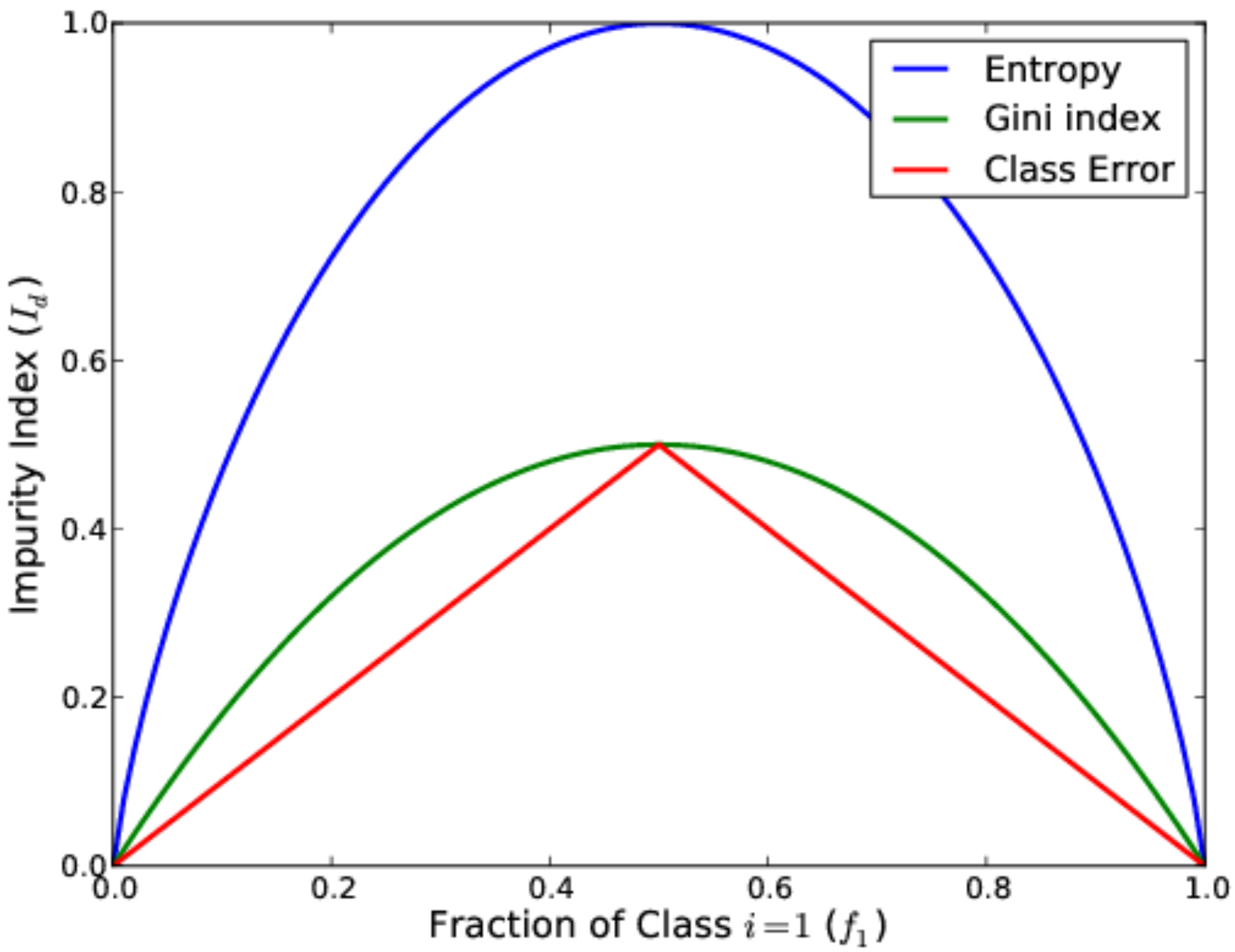


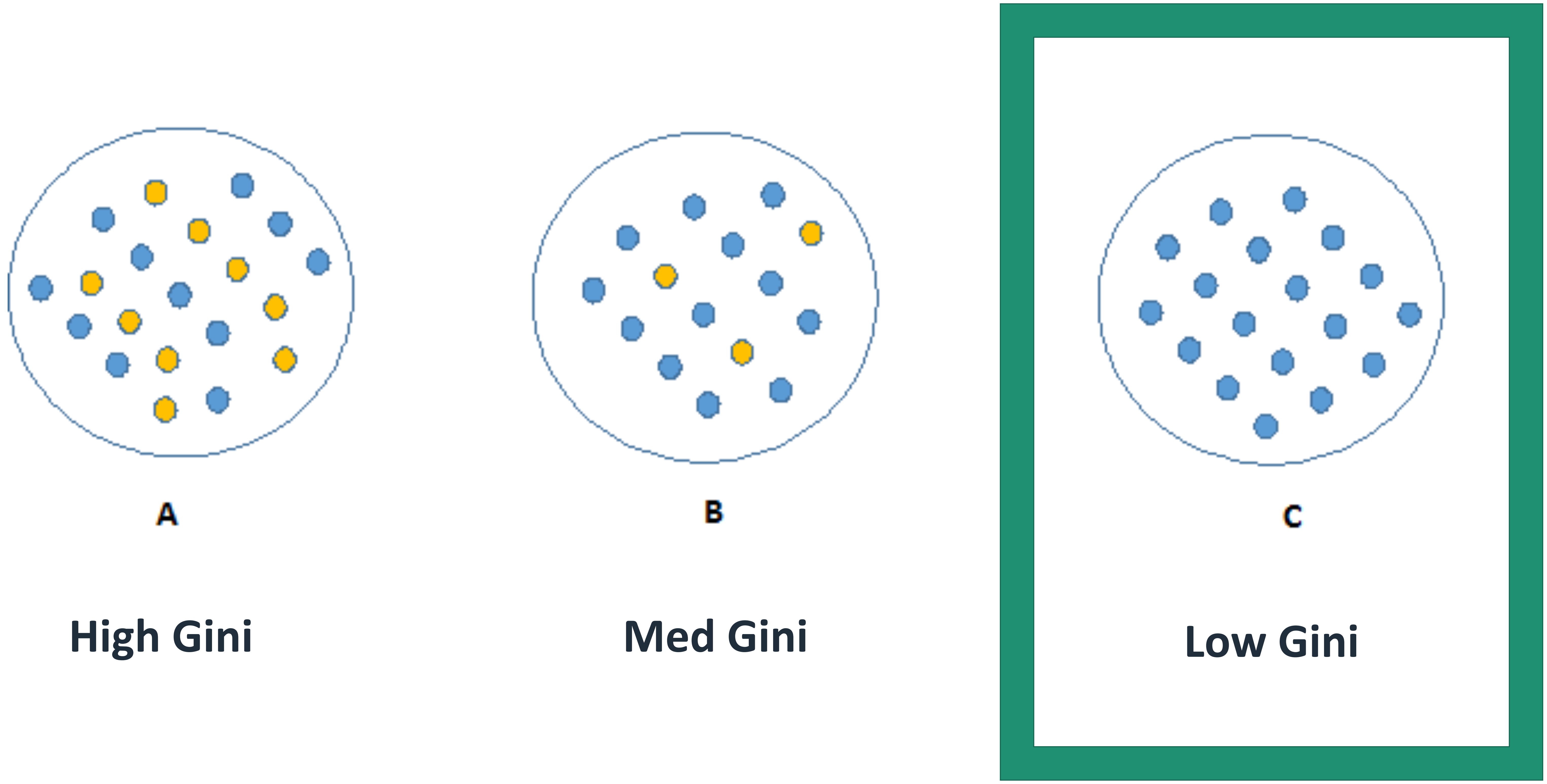
Non-Vegan Vegan Non-Vegan Vegan

How do we determine homogeneity of partitions?

- Many different methods possible (e.g., accuracy, information, entropy).
- Here, we use Gini impurity index:

$$I(p,q) = 2pq = P(Y_1 \neq Y_2)$$




A
B
C
High Gini
Med Gini
Low Gini

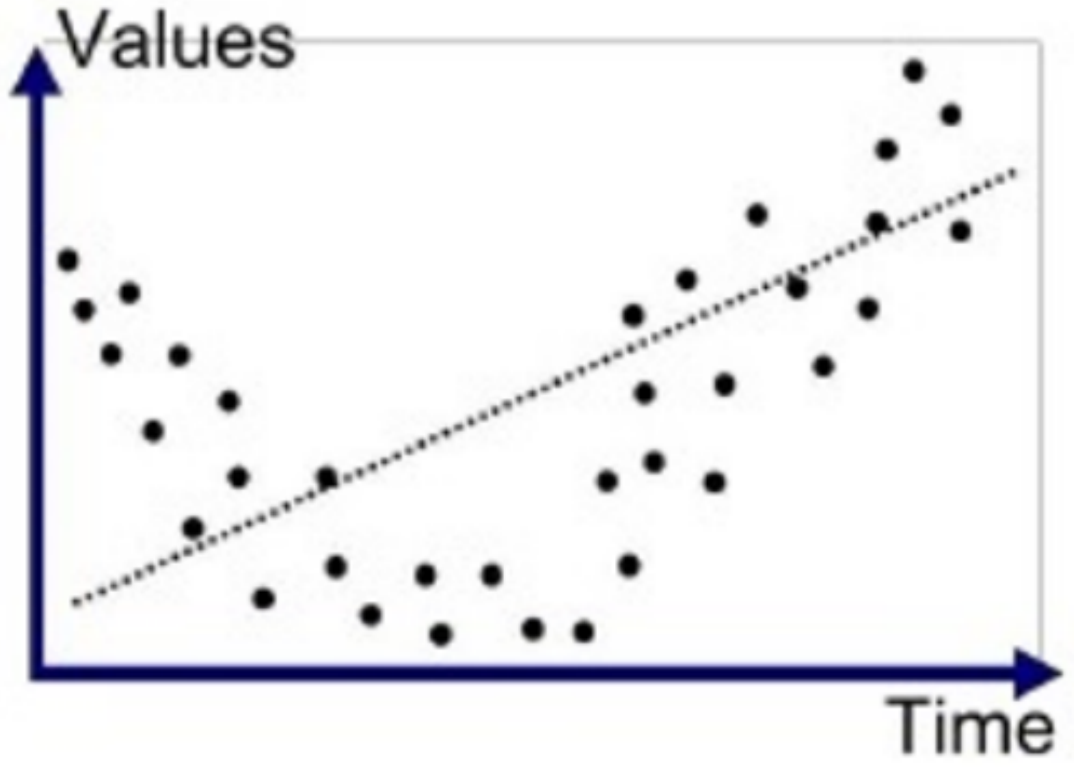
Recursive Partitioning Algorithm

- Suppose we have a response variable *Y* and a set of *P* predictor variables X_j for j = 1,…, P. For a partition *A* of records, recursive partitioning will find the best way to partition A into two subpartitions:
  1. For each predictor variable X_j:
     - For each value s_j of X_j:
       - Split the records in A with X_j values $< s_j$ as one partition, and the remaining records where $X_j \geq s_j$ as another partition
       - Measure the homogeneity of class within each subpartition of A
     - Select the value of s_j that produces maximum within-partition homogeneity of class.
  2. Select the variable X_j and the split value s_j that produces maximum within-partition homogeneity of class across all variables.
- **Then comes the recursive part:**
  1. Initialize A with the entire data set
  2. Apply the partitioning algorithm to split A into two subparitions A_1 and A_2
  3. Repeat step 2 on subpartitions A_1 and A_2
  4. Algorithm terminates when no further partition can be made that sufficiently improves the homogeneity of the partitions

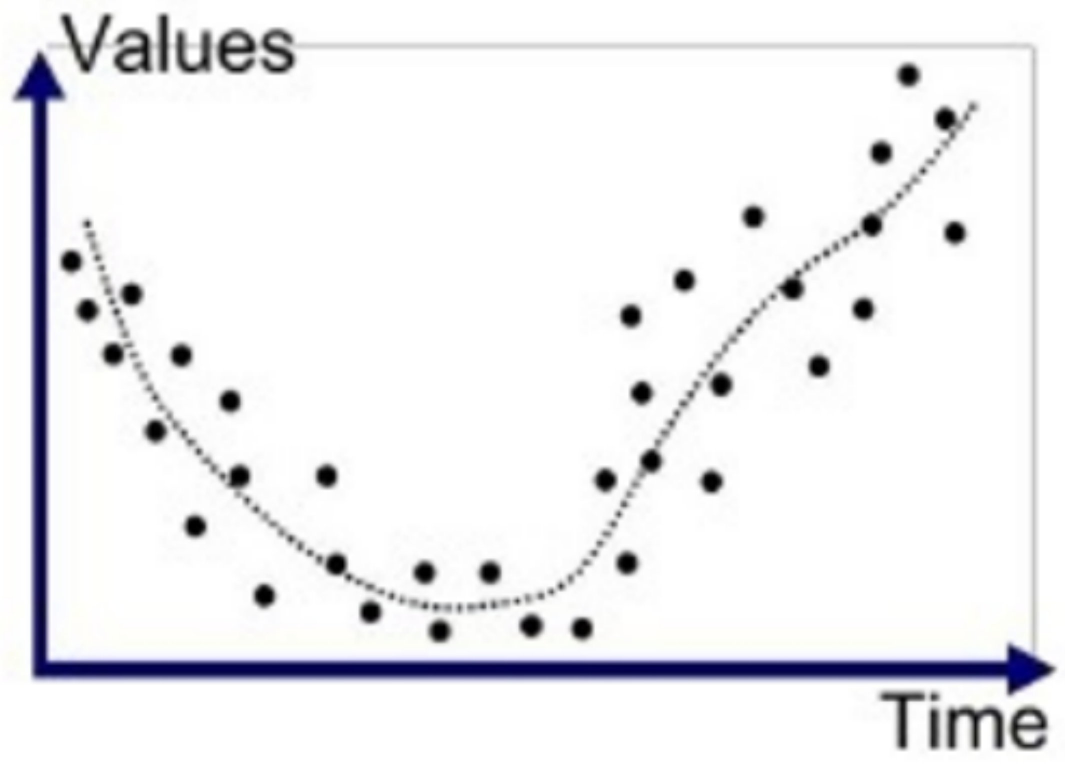
Bruce, P., & Bruce, A. (2017). *Practical statistics for data scientists.* Sebastopol, CA: O'Reilley Media, Inc.

Vegan

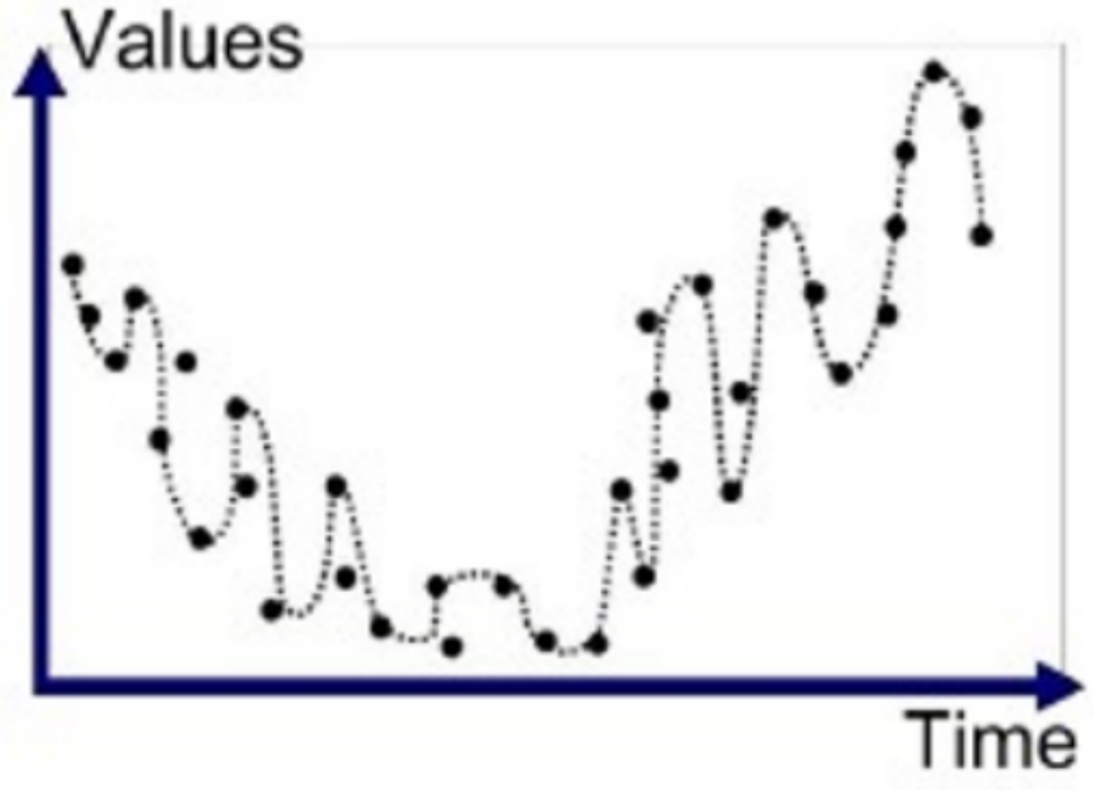
Non-Vegan



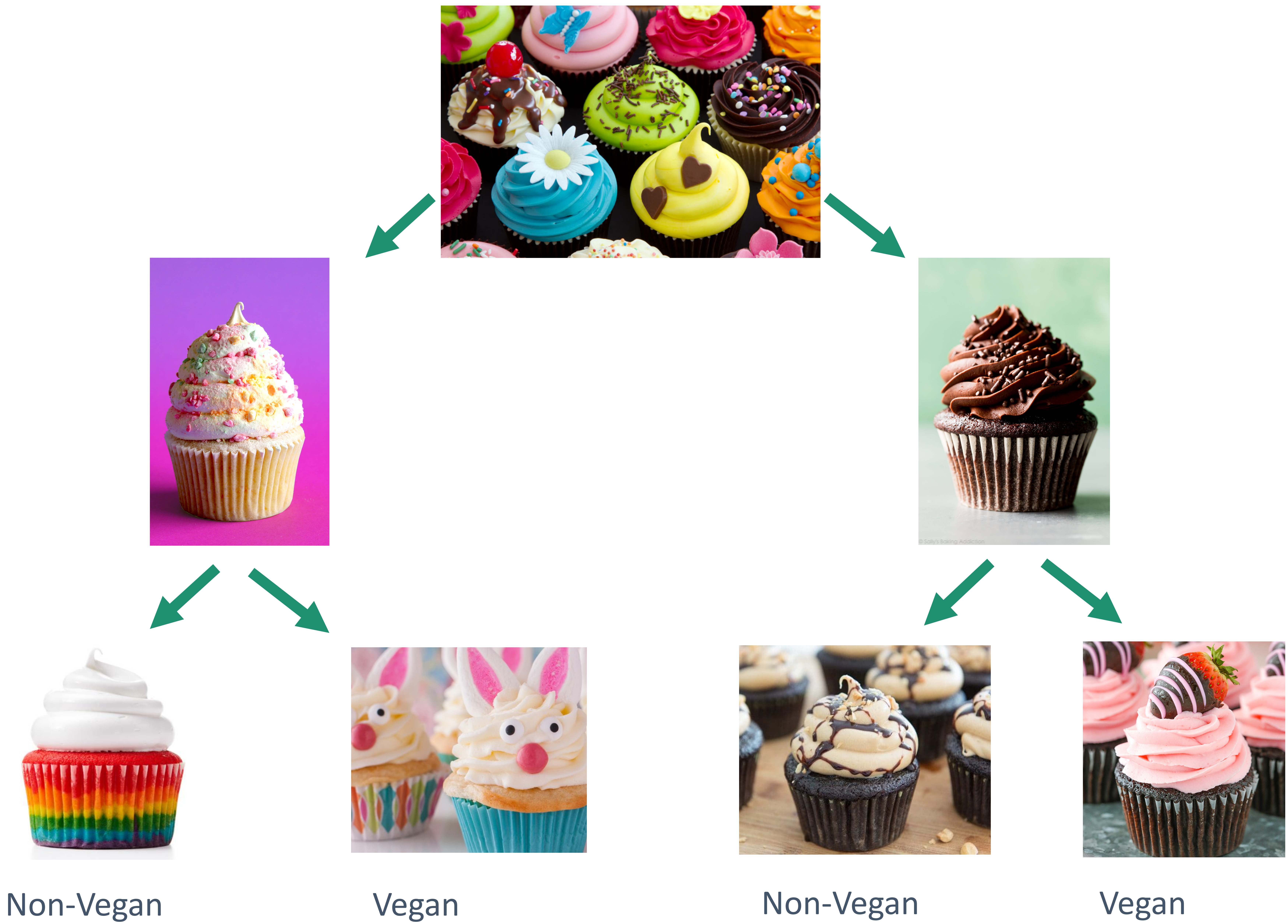
Underfit



Good Fit/Robust



Overfit

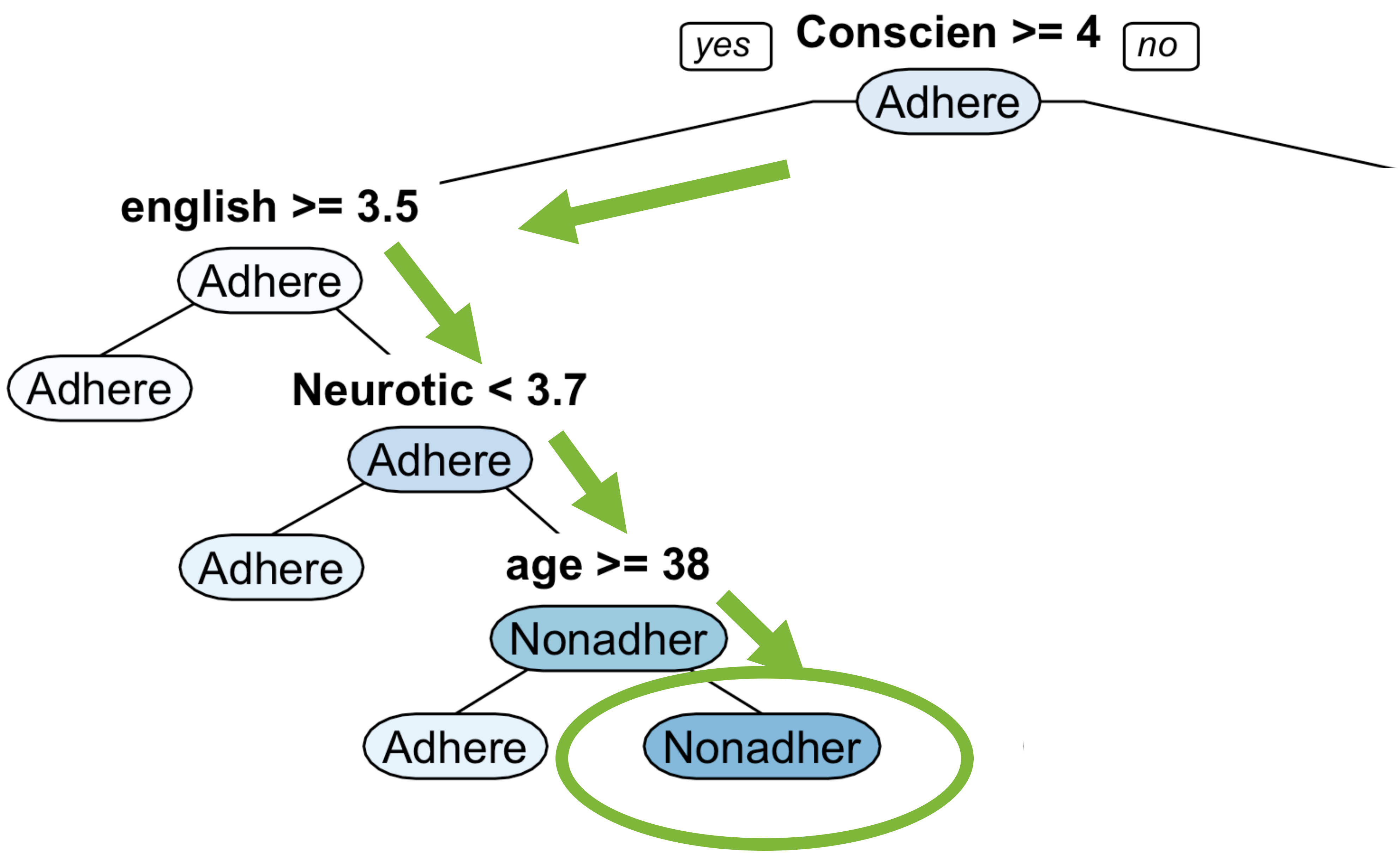

Non-Vegan
Vegan
Non-Vegan
Vegan

Medication Adherence Decision Tree

Baker Bob

- Conscientiousness: 4.5
- English Proficiency: 3
- Neuroticism: 4
- Age: 35
- Prediction?

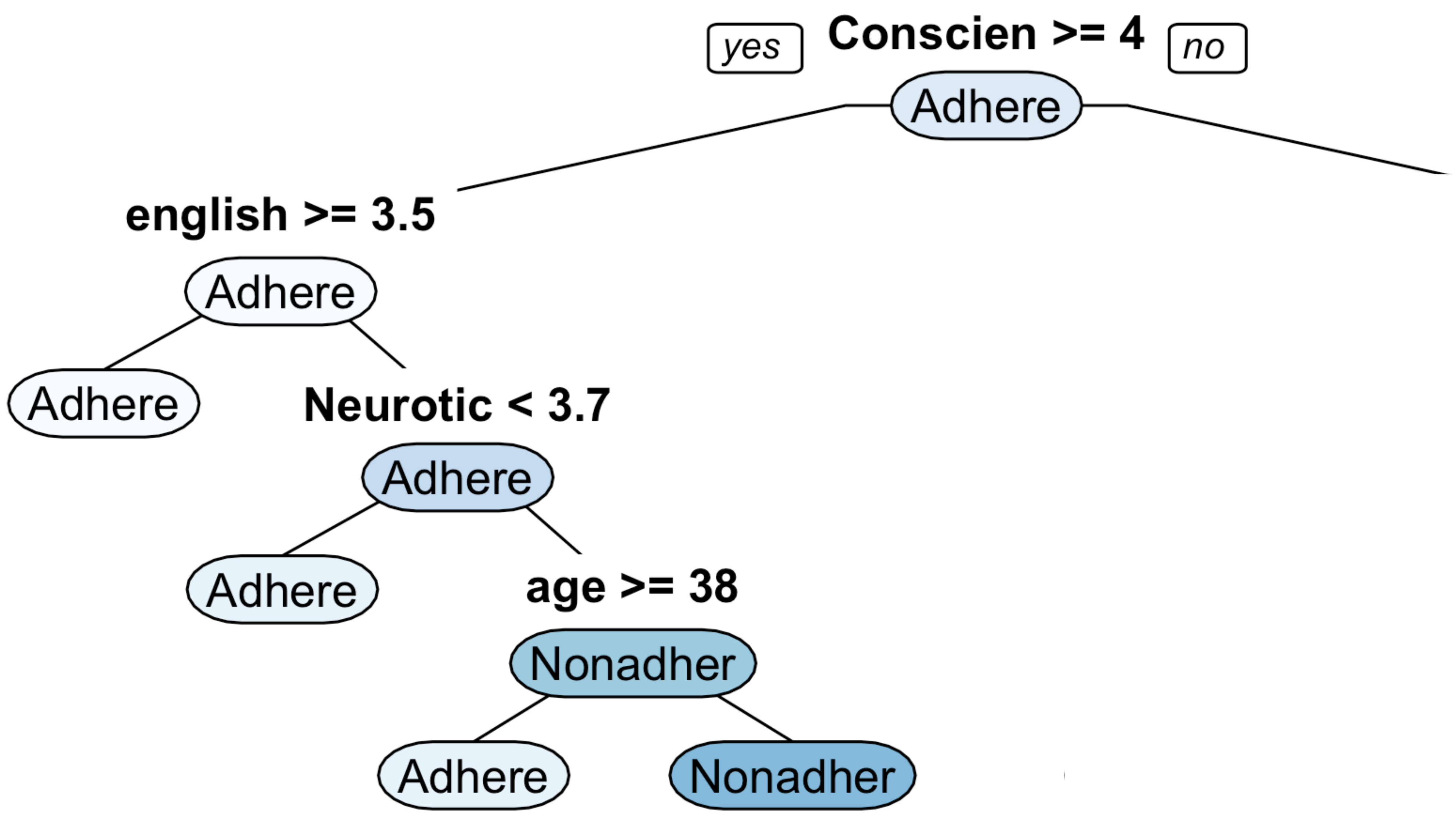
Medication Adherence Decision Tree



Baker Bob

- Conscientiousness: 4.5
- English Proficiency: 3
- Neuroticism: 4
- Age: 35
- Prediction?

Medication Adherence Decision Tree



Time for Decision Tree Coding

- Steps:
  - 1) Upload, score new personality data
  - 2) Separate training, test
  - 3) Construct model using `rpart` on training data
  - 4) Test model on test dataset

- Walk through example by me, then you will try

Key take-aways from tree-based methods

- Decision-focused, not trying to explain
- Pretty pictures!
- Intuitive, no stats background needed
- But…….
  - Pretty shitty when it comes to predicting anything. So like…..why use?

Ensemble Methods: Random Forest


cupcake
All your ~~base~~ are
belong to us!

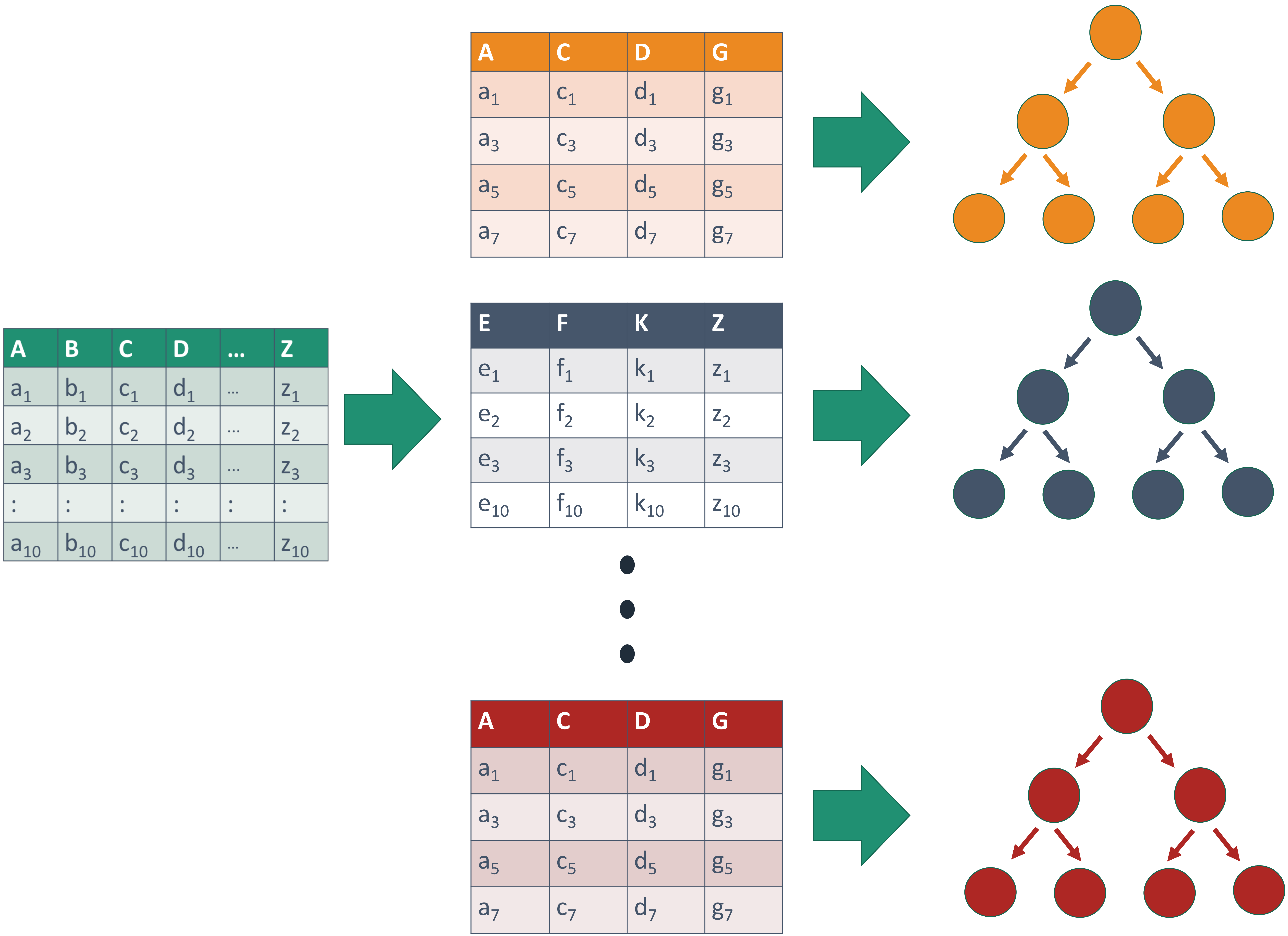
Seeing the forest for the trees

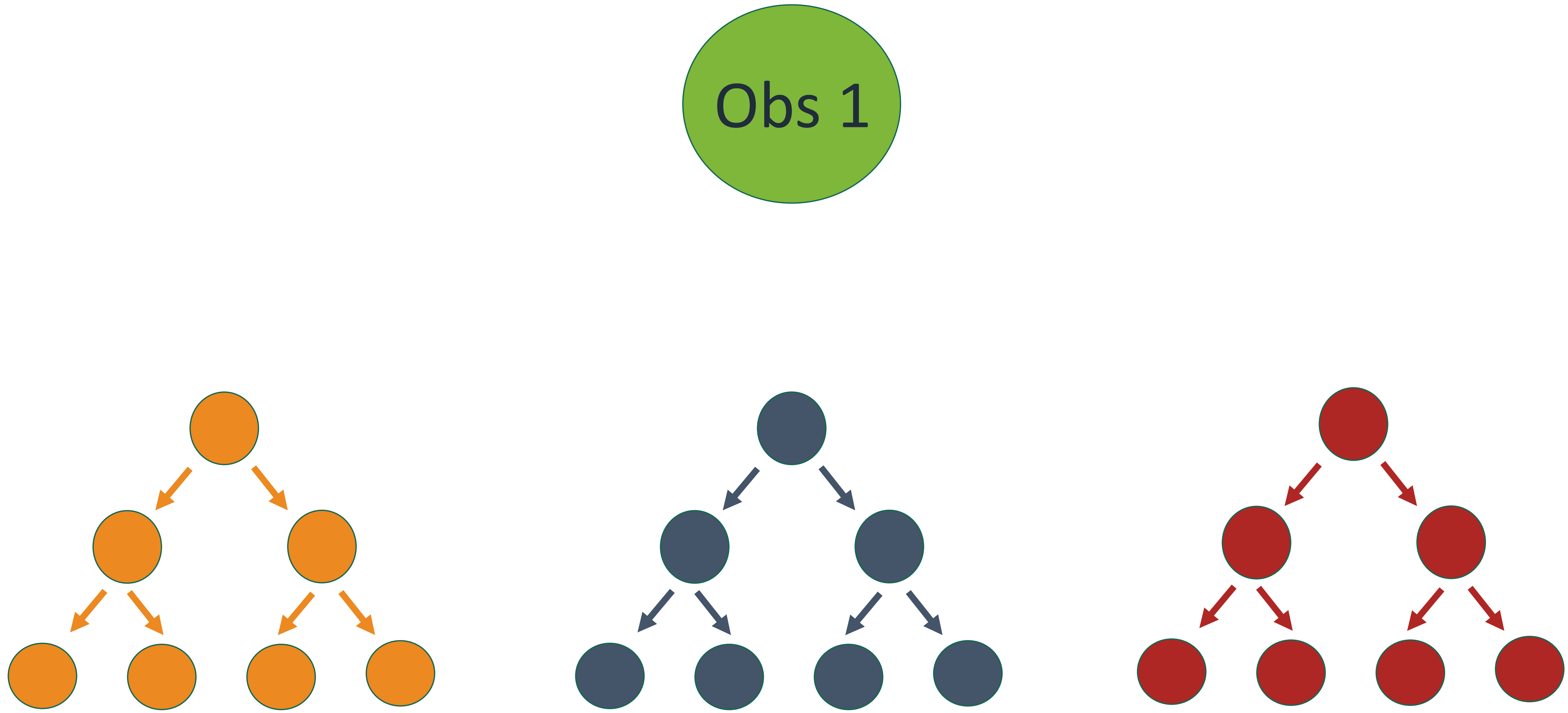
- In general, follows recursive partition algorithm, but for a randomly selected subsample from the records.
- Algorithm:
  1. Take a bootstrap (with replacement) subsample from the records
  2. For the first split, sample $p < P$ variables at random without replacement
  3. For each of the sample variables $X_{j(1)}, X_{j(2)},\ldots,X_{j(p)}$, apply the splitting algorithm:
     - For each value $s_{j(k)}$ of $X_{j(k)}$:
       - Split the records in A with $X_{j(k)}$ values $< s_{j(k)}$ as one partition, and the remaining records where $X_{j(k)} \geq s_{j(k)}$ as another partition
       - Measure the homogeneity of class within each subpartition of A
     - Select the value of $s_{j(k)}$ that produces maximum within-partition homogeneity of class.
  4. Select the variable $X_{j(k)}$ and the split value $s_{j(k)}$ that produces maximum within-partition homogeneity of class.
  5. Proceed to the next split and repeat the previous steps, starting with step 2.
  6. Continue with additional splits following the same procedure until the tree is grown
  7. Go back to step 1, take another bootstrap subsample and start the process over again
- How many at each step? Sqrt(P) is the rule of thumb for variabels, 2/3 of records.

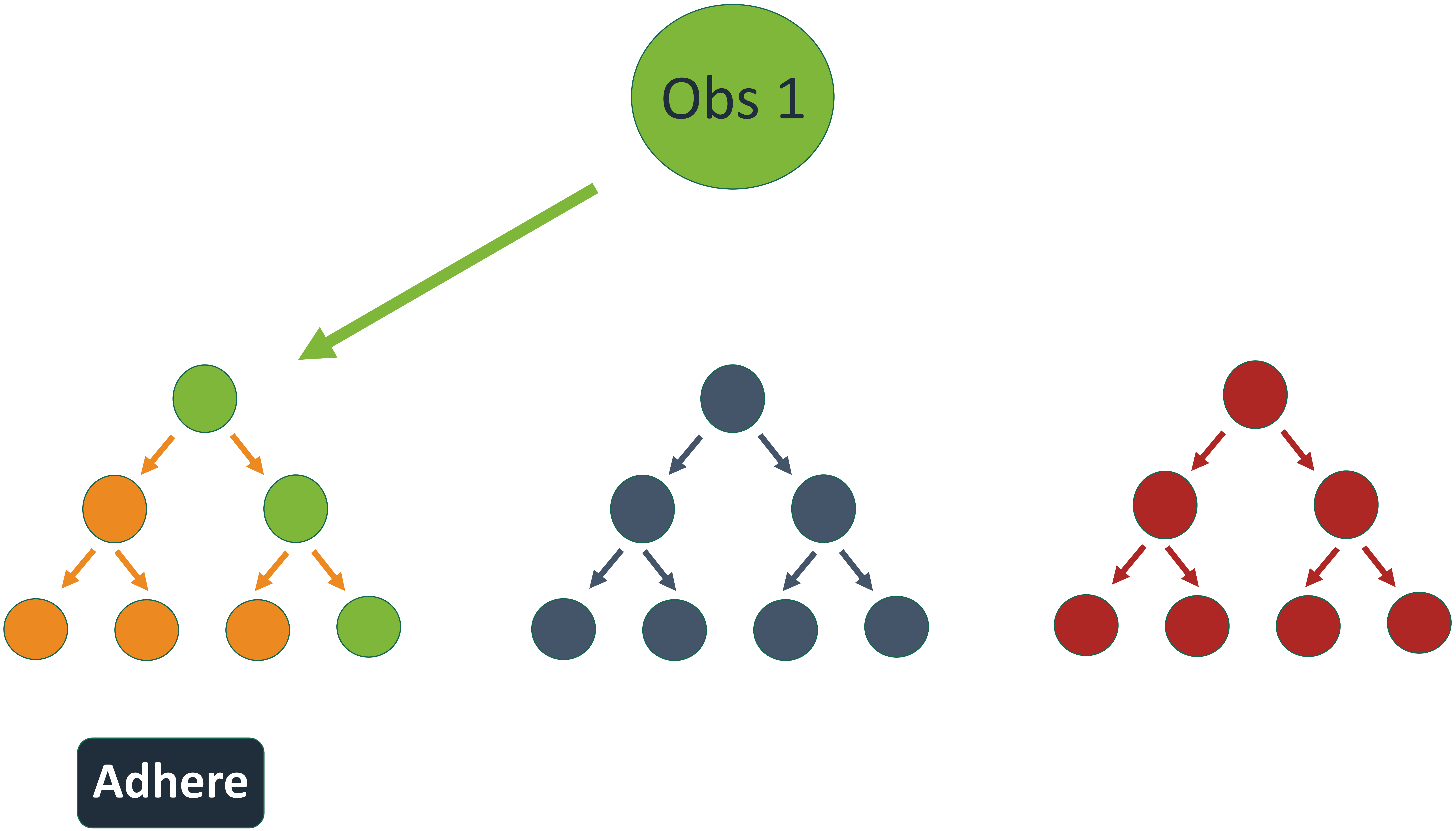
Bruce, P., & Bruce, A. (2017). *Practical statistics for data scientists.* Sebastopol, CA: O'Reilley Media, Inc.

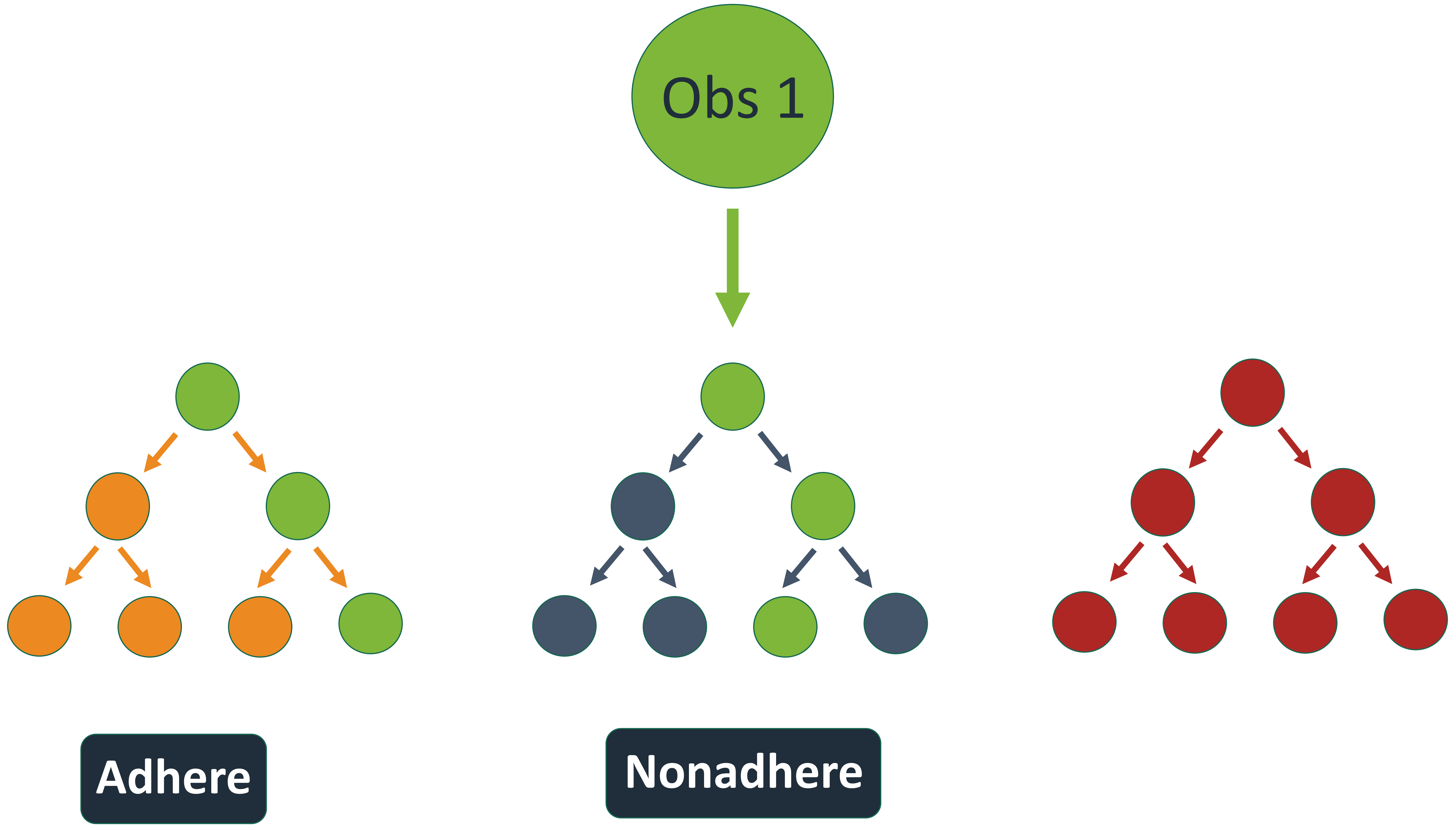
Taking advantage of the power of averages

- Ensemble methods derive their utility from averages
- General ensemble algorithm:
  - Develop a predictive model and record the predictions for a given dataset
  - Repeat for multiple models, on the same data
  - For each record to be predicted, take and average (or a weighted average or a majority vote) of the predictions

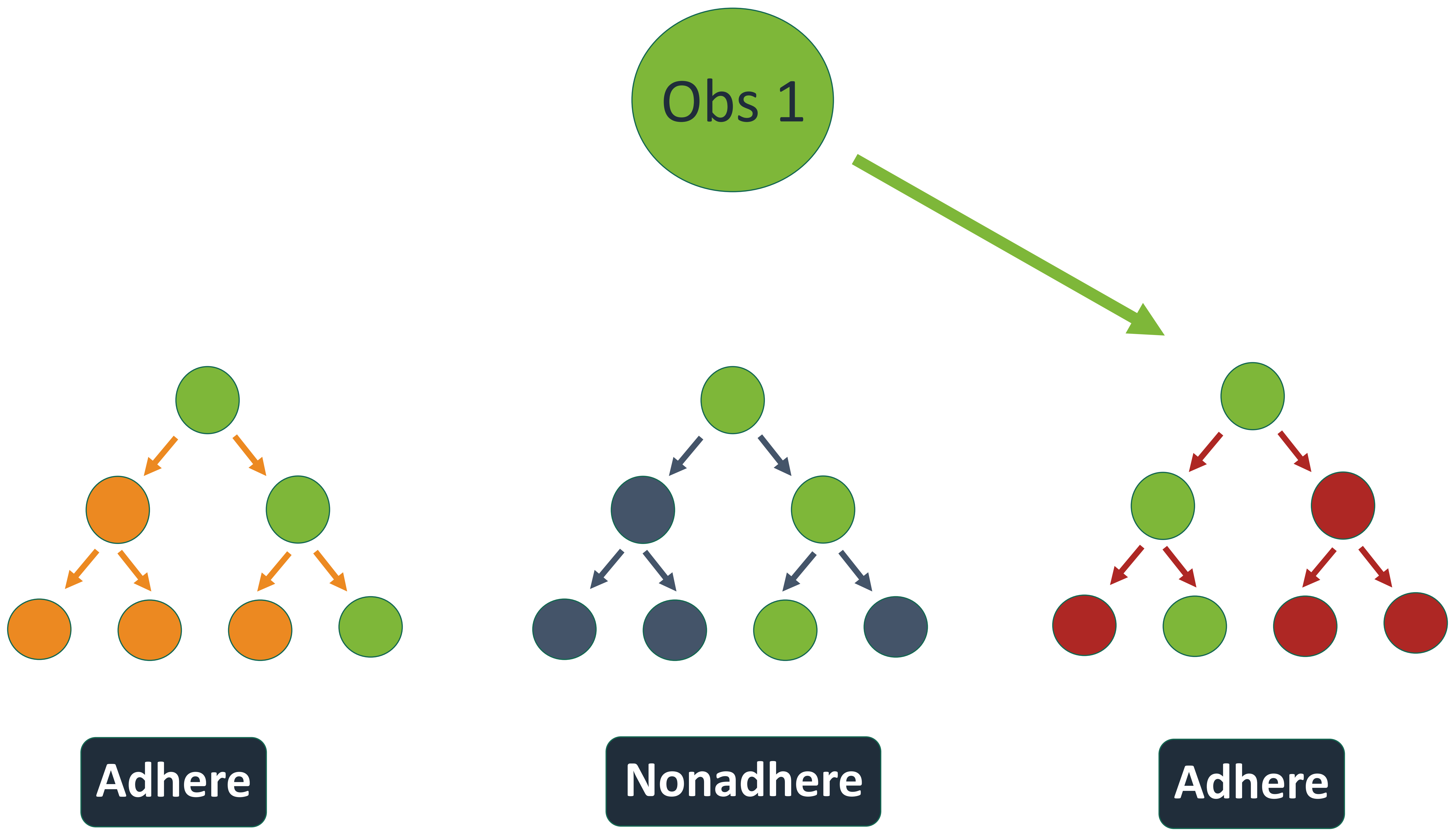

A B C D … Z
a1 b1 c1 d1 … z1
a2 b2 c2 d2 … z2
a3 b3 c3 d3 … z3
: : : : : :
a10 b10 c10 d10 … z10
A C D G
a1 c1 d1 g1
a3 c3 d3 g3
a5 c5 d5 g5
a7 c7 d7 g7
E F K Z
e1 f1 k1 z1
e2 f2 k2 z2
e3 f3 k3 z3
e10 f10 k10 z10
A C D G
a1 c1 d1 g1
a3 c3 d3 g3
a5 c5 d5 g5
a7 c7 d7 g7


Obs 1

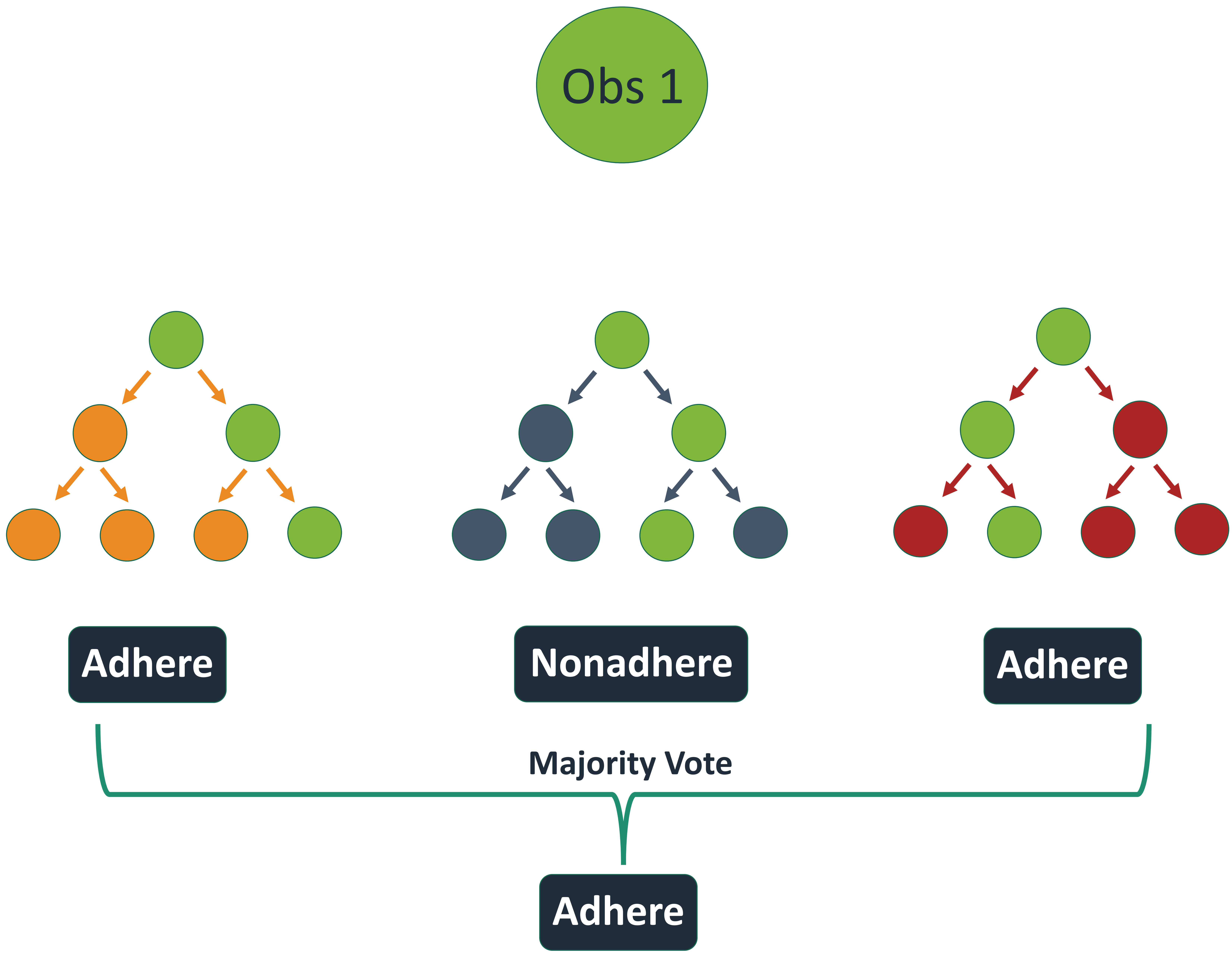

Obs 1
Adhere



Obs 1
Adhere
Nonadhere



Obs 1
Adhere
Nonadhere
Adhere


Obs 1
Adhere
Nonadhere
Adhere
Majority Vote
Adhere

Hyperparameters

Random Forest is “Black box”

- Can’t see inside
- But can control how the box functions!

Unable to estimate directly

- Use cross-validation

Hyperparameters for Random forest:

- Number of trees
- Number of variables selected at each node split

Glimpse Into Performance with Variable Importance Metrics

- How “important” a variable is to the overall performance
- Think: “if this variable were excluded from the model, how would my results differ?”
  - This is what the algorithm is doing behind the scenes
  - Removing the effect of each variable by randomly permuting, then examining how that impacts performance on OOB observations

Time for Random Forest Coding

- Steps:
  - 1) Take complete data from personality data
  - 2) Separate training, test
  - 3) Construct model using `randomForest` on training data
  - 4) View variable importance
  - 5) Test model on test dataset

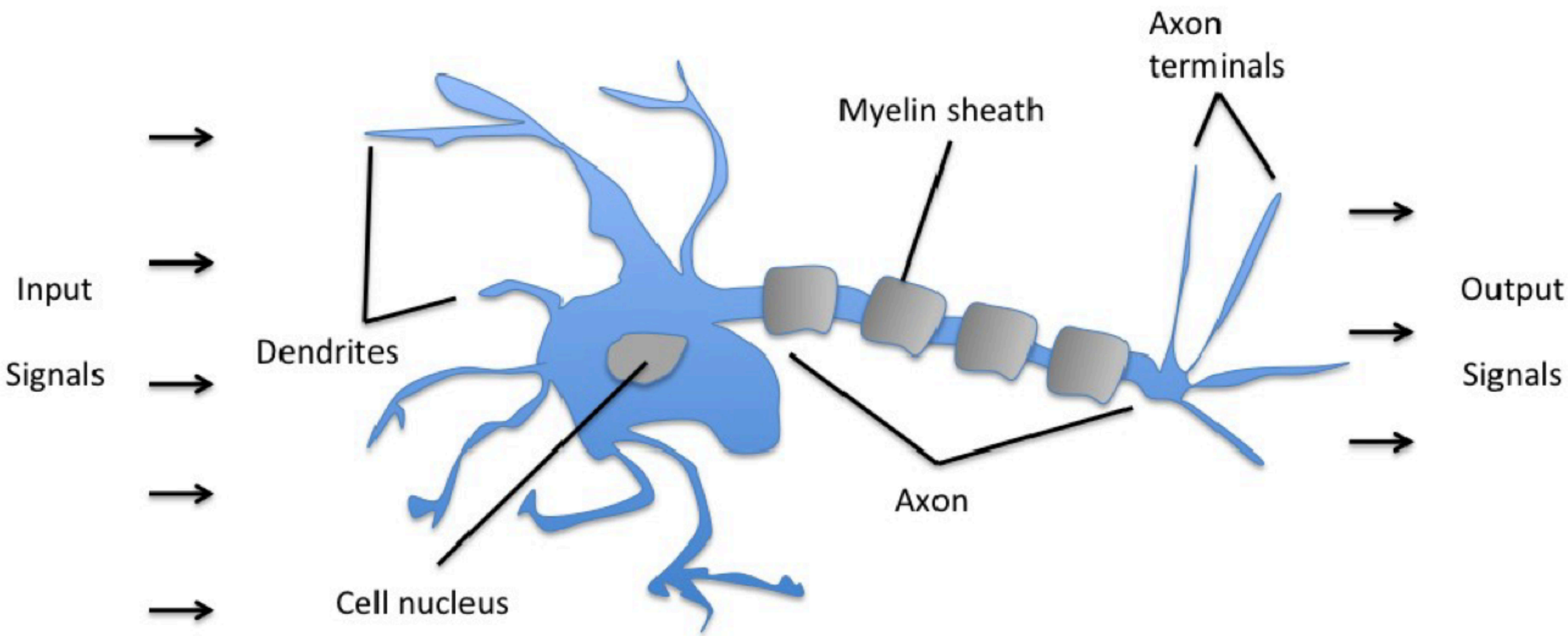
- Walk through example by me, then you will try

Key take-aways from ensemble methods

- Uses many different “weak learners” to create a strong learner
- GBMs and RFs provide top-tier accuracy in competitions
- BUT…..
    - Little interpretability (no visible trees)
    - Black box methods

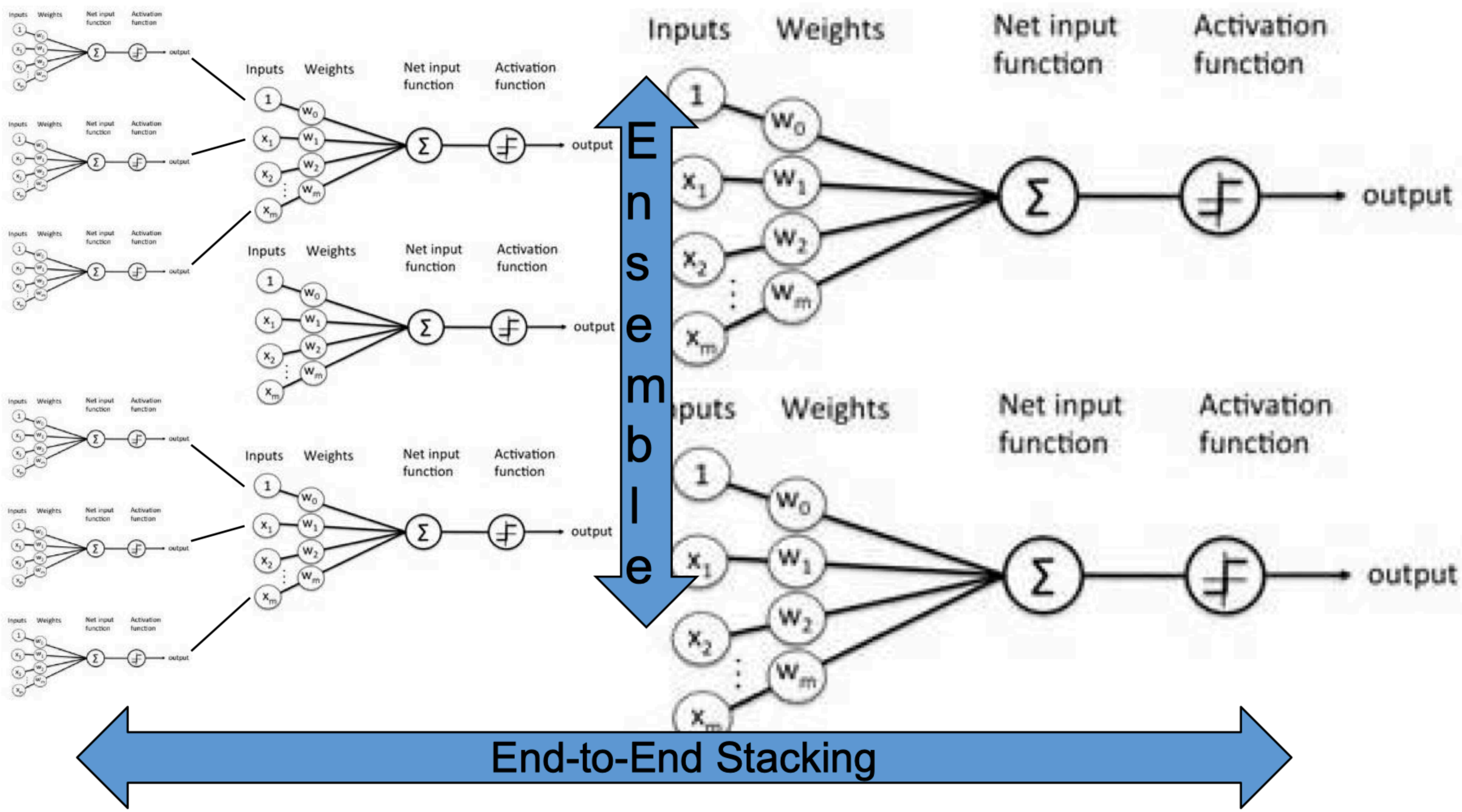
Advanced Methods: Neural Networks

It’s like a human neuron…..right?

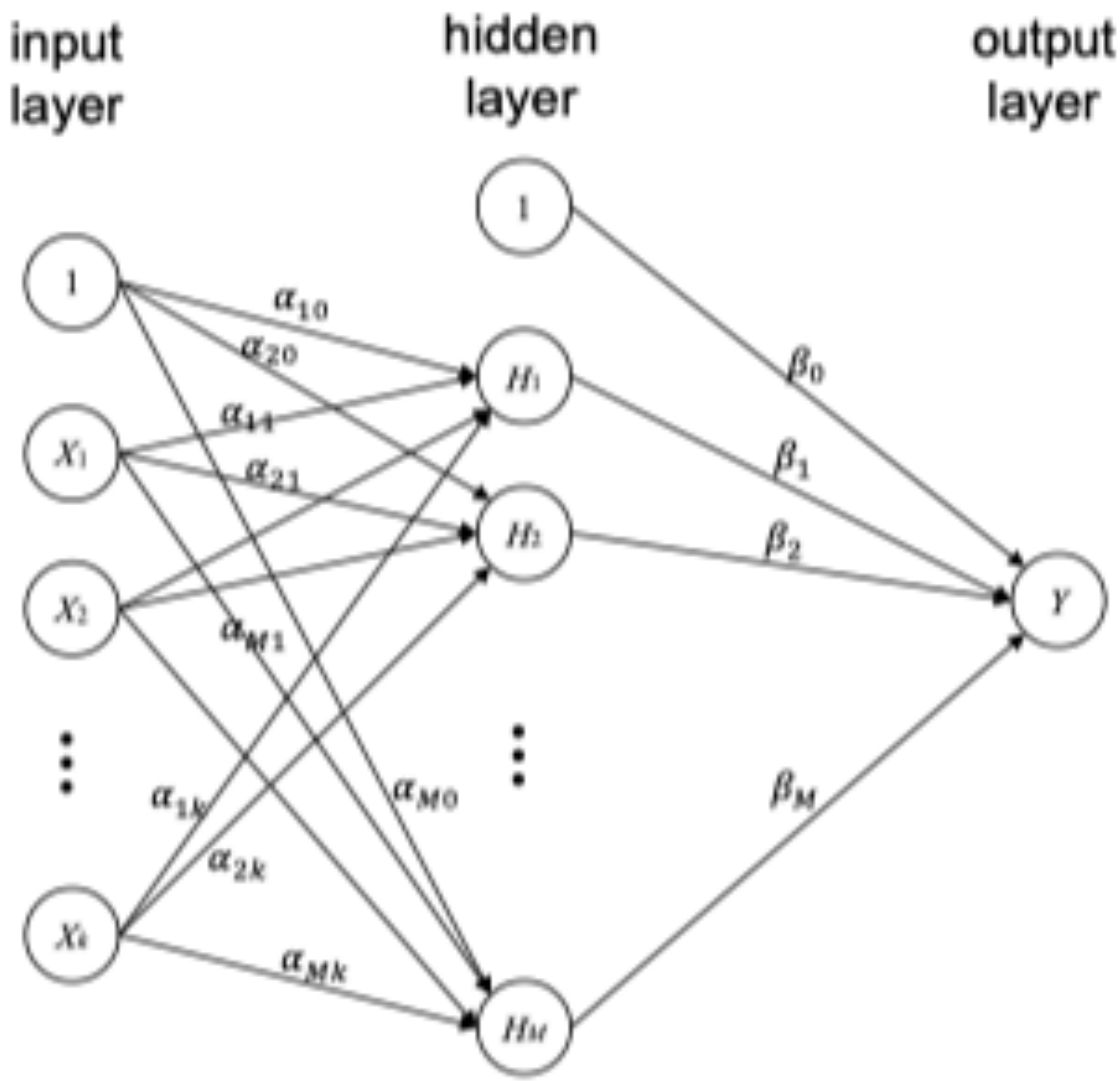


Schematic of a biological neuron.

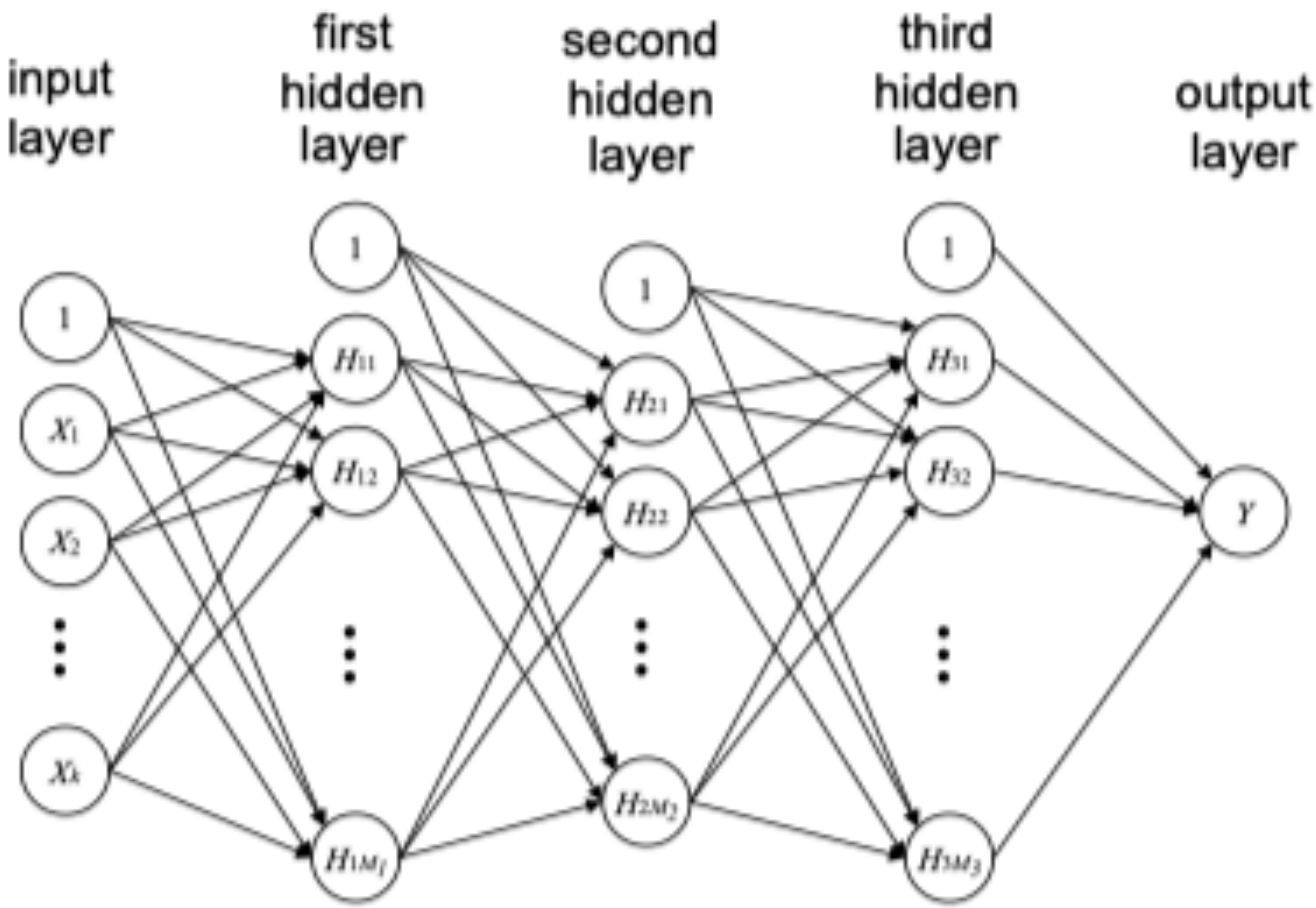
More accurate: Many logistic regressions



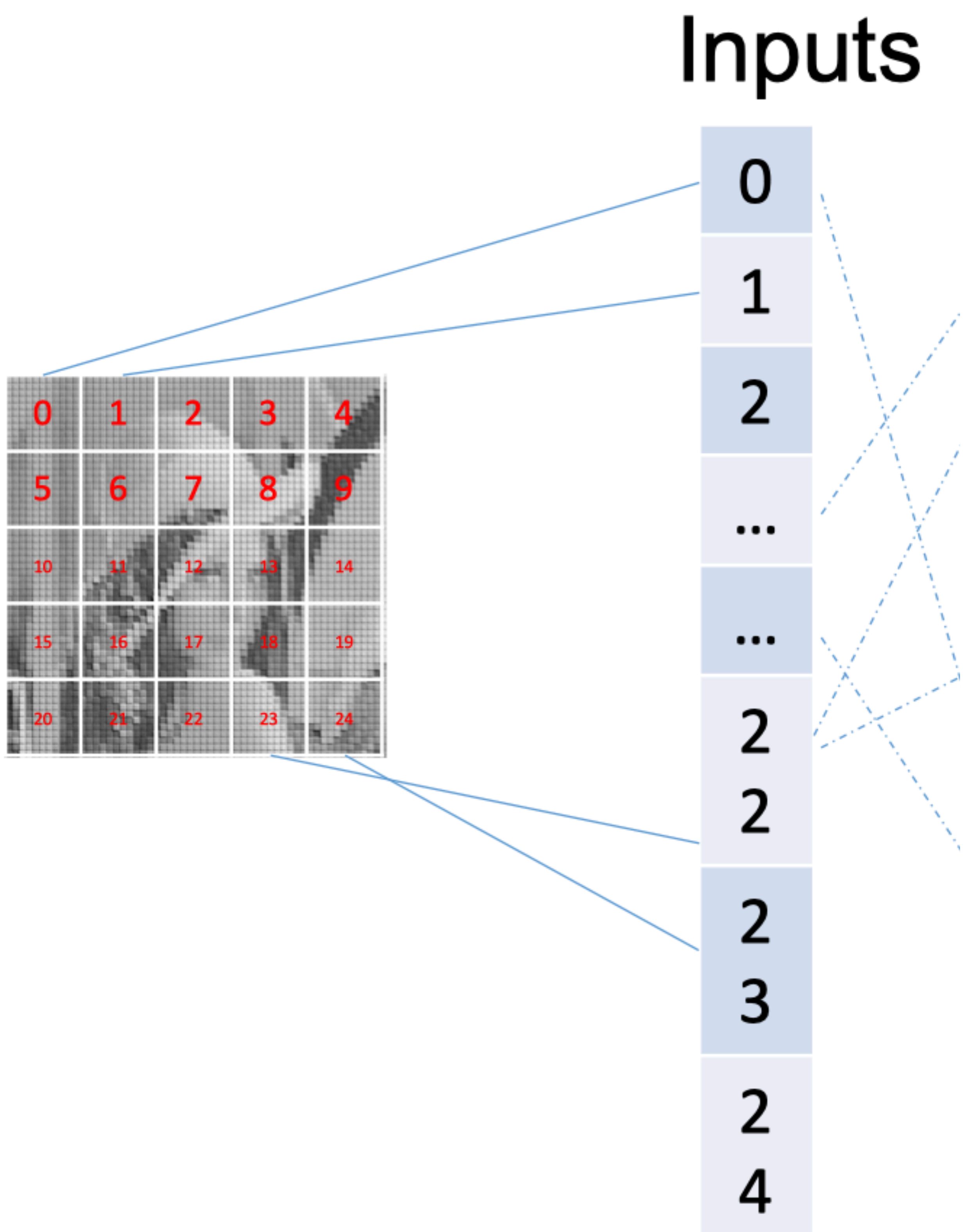
Neural Network with a single "hidden layer"



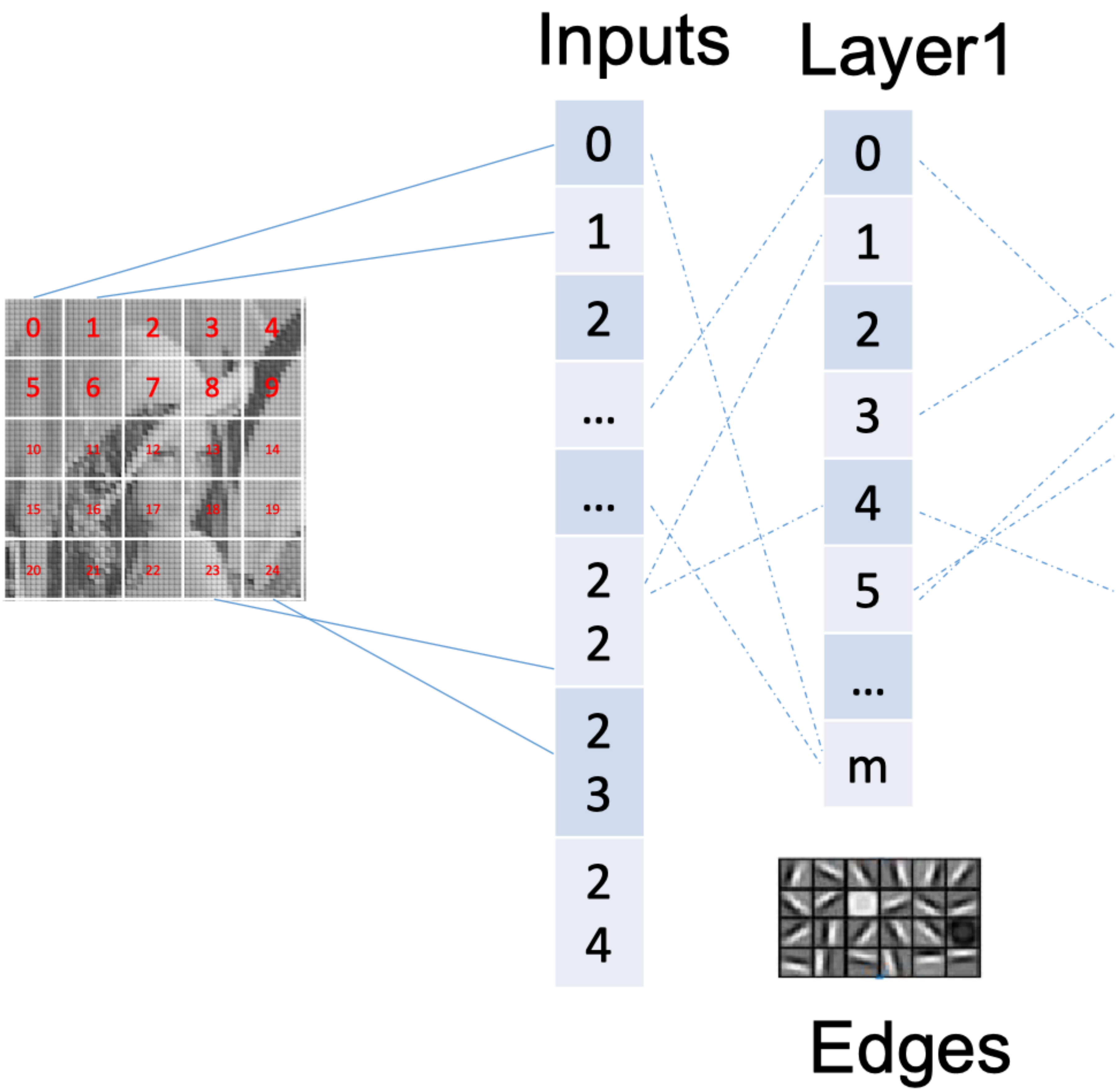
Neural Network with 3 Hidden Layers



How does a neural network “learn?”

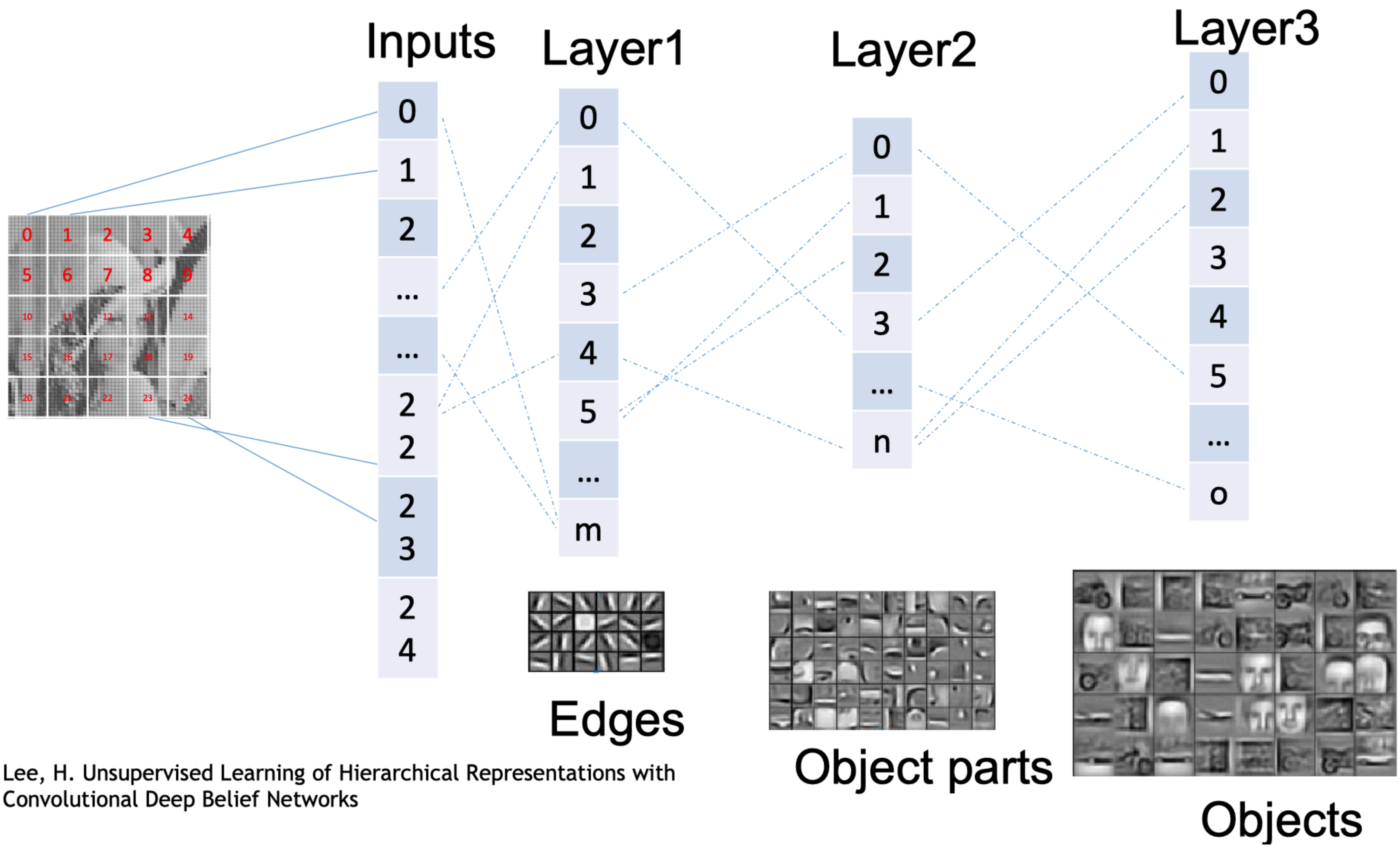


How does a neural network "learn?"



Lee, H. Unsupervised Learning of Hierarchical Representations with
Convolutional Deep Belief Networks

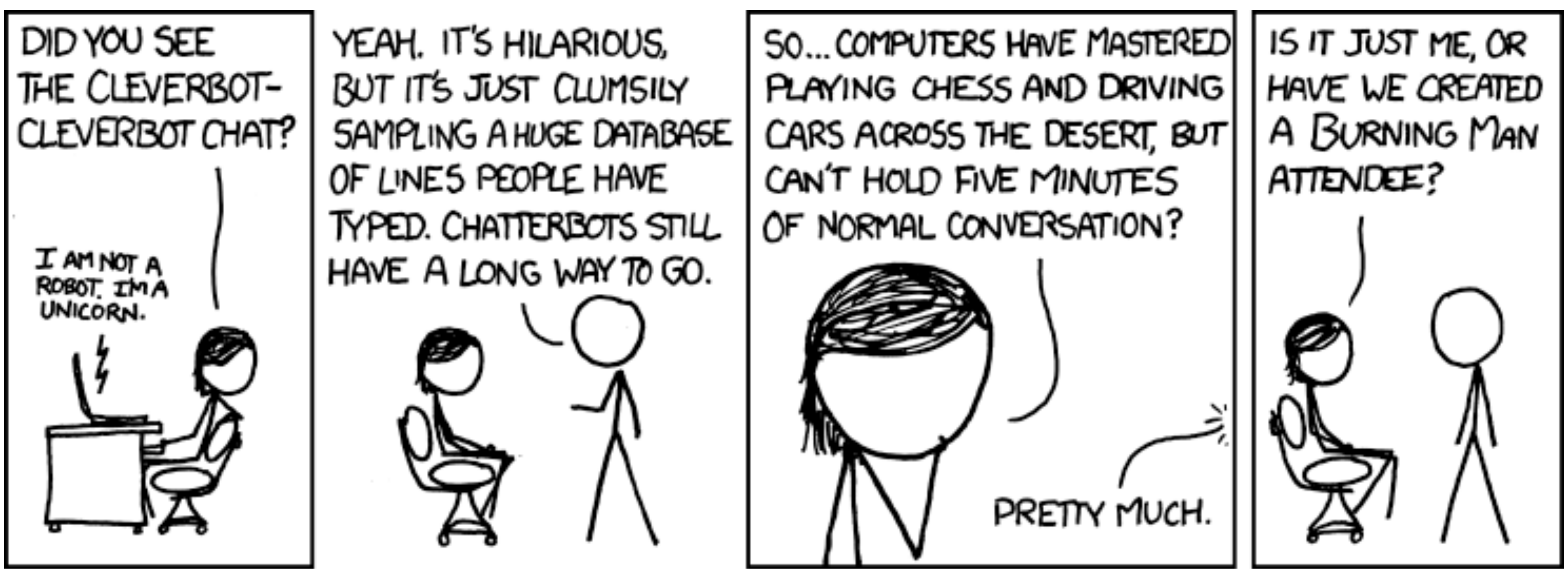
How does a neural network "learn?"



Lee, H. Unsupervised Learning of Hierarchical Representations with Convolutional Deep Belief Networks

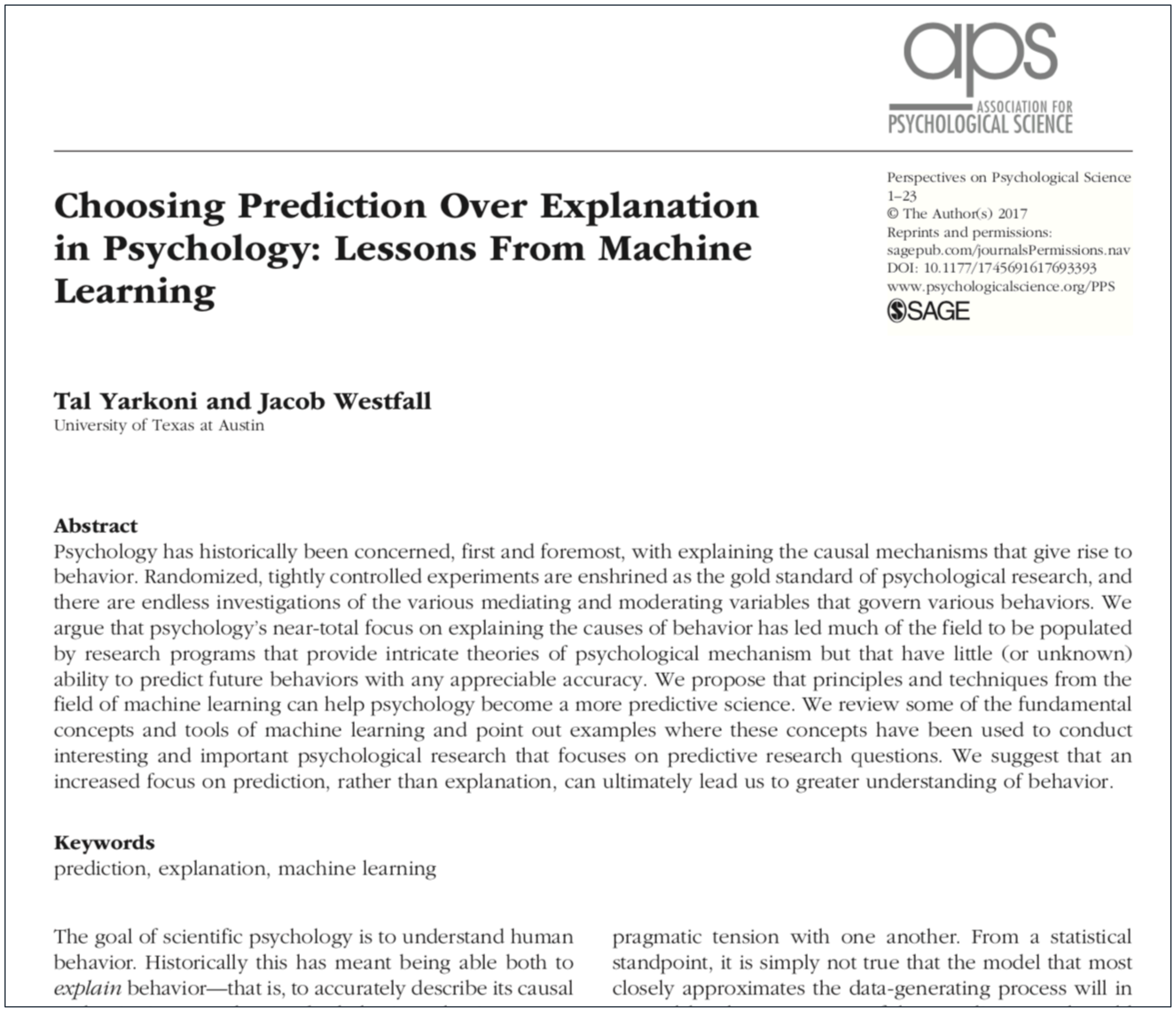
Future Directions

- Recurrent neural networks, convolutional neural networks
- Natural language processing
- Lots to learn!



Conclusions: Where to Learn More

Interested in Learning More?


aps ASSOCIATION FOR PSYCHOLOGICAL SCIENCE

Perspectives on Psychological Science
1–23

DOI: 10.1177/1745691617693393
www.psychologicalscience.org/PPS
SAGE

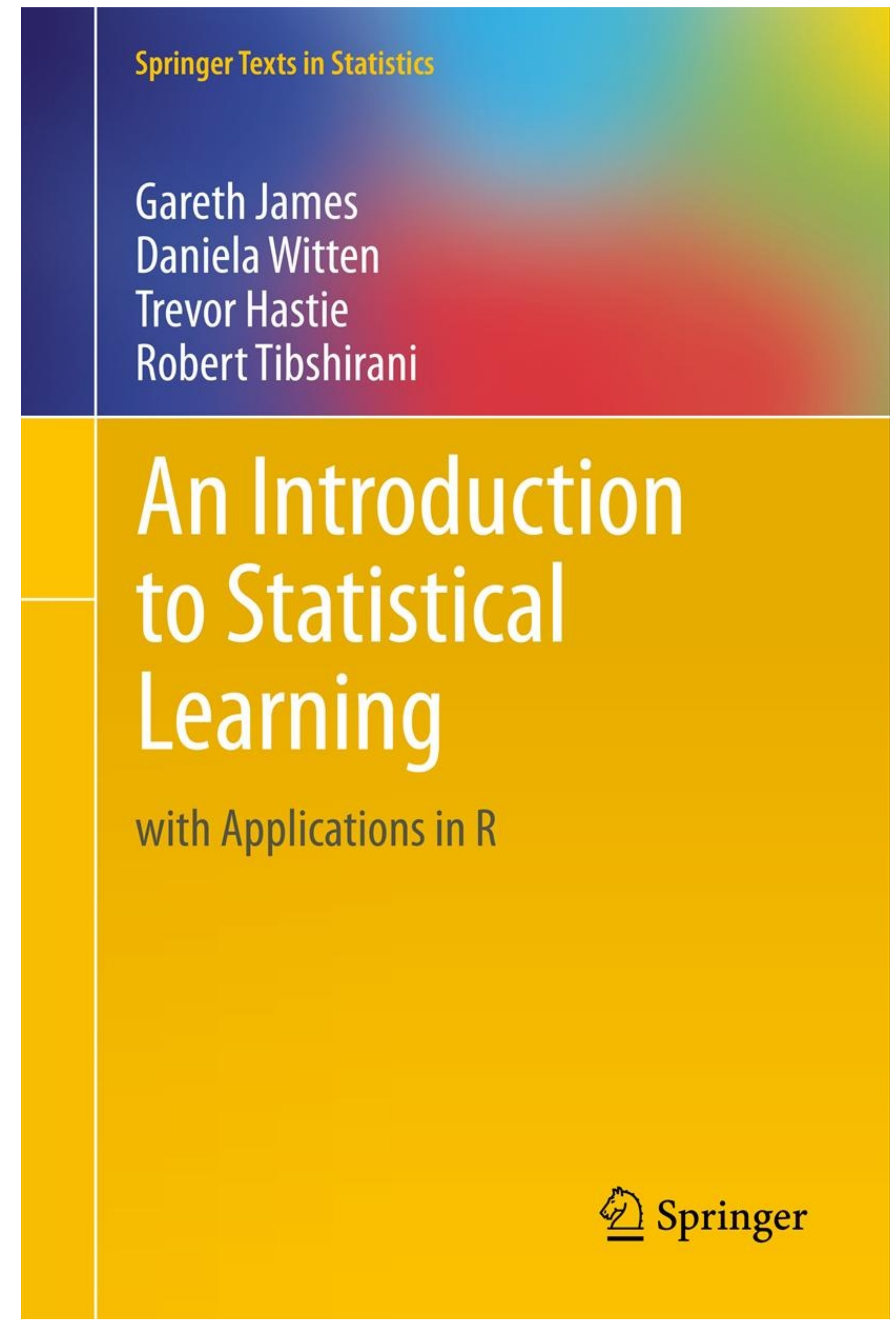
Choosing Prediction Over Explanation in Psychology: Lessons From Machine Learning

Tal Yarkoni and Jacob Westfall
University of Texas at Austin

Abstract
Psychology has historically been concerned, first and foremost, with explaining the causal mechanisms that give rise to behavior. Randomized, tightly controlled experiments are enshrined as the gold standard of psychological research, and there are endless investigations of the various mediating and moderating variables that govern various behaviors. We argue that psychology's near-total focus on explaining the causes of behavior has led much of the field to be populated by research programs that provide intricate theories of psychological mechanism but that have little (or unknown) ability to predict future behaviors with any appreciable accuracy. We propose that principles and techniques from the field of machine learning can help psychology become a more predictive science. We review some of the fundamental concepts and tools of machine learning and point out examples where these concepts have been used to conduct interesting and important psychological research that focuses on predictive research questions. We suggest that an increased focus on prediction, rather than explanation, can ultimately lead us to greater understanding of behavior.

Keywords
prediction, explanation, machine learning

The goal of scientific psychology is to understand human behavior. Historically this has meant being able both to *explain* behavior—that is, to accurately describe its causal pragmatic tension with one another. From a statistical standpoint, it is simply not true that the model that most closely approximates the data-generating process will in



Workshop on Statistical Modeling Next Week
July 31 9 am – 12 pm, Location Chambers Hall

The end

THANK YOU!

Full Citations from Slides

Bruce, P., & Bruce, A. (2017). *Practical statistics for data scientists.* Sebastopol, CA: O'Reilley Media, Inc.

Hastie, T., Tibshirani, R., & Friedman, J. (2001). *The Elements of Statistical Learning: Data mining, inference, and prediction 2nd Edition*. New York, NY: Springer.

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An introduction to statistical learning with applications in R*. New York, NY: Springer.

McCarthy, R. (1998). The price you pay for the drug not taken. *Business Health, 16,* 27-28.

[1]Neiheisel, M. B., Wheeler, K. J., & Roberts, M. E. (2013). Medication adherence part one: Understanding and assessing the problem. *Journal of the American Association of Nurse Practitioners, 26,* 49-55.

[2]Osterberg, L. & Blaschke, T. (2005). Adherence to medication. *New England Journal of Medicine, 353,* 487-497.

[3]Bosworth, H. B., Granger, B. B., Mendys, P., Brindis, R., Burkholder, R., Czajkowski, M., Grander, C. B, (2011). Medication adherence: A call for action. *American Heart Journal, 162,* 412-424.

[4]McCarthy, R. (1998). The price you pay for the drug not taken. *Business Health, 16,* 27-28.

[5]Roberts, B. W., Kuneel, N. R., Shiner, R., Caspi, A., & Goldberg, L. R., (2007). The power of personality: The comparative validity of personality traits, socioeconomic status, and cognitive ability for predicting important life outcomes. *Perspectives in Psychological Science, 2,* 313-345.

[6] Hill, P. L., & Roberts, B. W. (2011). The role of adherence in the relationship between conscientiousness and perceived health. *Health Psychology, 30,* 797-804.

[7] Molloy, G. J., O'Carroll, R. E., Ferguson, E. (2014). Conscientiousness and medication adherence: A meta-analysis. *Annals of Behavioral Medicine, 47,* 92-101.

Additional Code

```
########################
## R Script accompanying "Recursive Binary Partitioning and the Random Forest: An Introduction to
Tree-Based Machine Learning Methods in R"
## In-text Walkthrough
## Authors: Andrew N Hall, David M Condon, Daniel K Mroczek
########################

# Install and load required packages
packages = c("psych", "rpart", "randomForest", "dplyr", "ggplot2") #packages needed for
walkthrough
packages.notinstalled <- packages[!(packages %in% installed.packages()[,"Package"])] #check for
packages installed
if(length(packages.notinstalled)) install.packages(packages.notnstalled)
library(psych); library(rpart); library(randomForest); library(dplyr); library(ggplot2) #load
relevant packages

# Load dataset from psych package
data(spi) #load spi data
dat = spi #assign spi data to the variable `data`

# Split data into training and test datasets
set.seed(44) #set seed for reproducible results
dat_train = sample_frac(dat, size = 7/10) #take 70% of observations for training data
dat_test = setdiff(dat, dat_train) #leave remaining 30% for test data
```

Additional Code

```
####
# Decision Tree - Regression Using Recursive Binary Partitioning
###
dtree = rpart(health ~ ., method = "anova", parms = list(split= "gini"), data = dat_train) #apply
rpart to data to create decision tree

plot(dtree, uniform = T, main = "Regression Decision Tree Predicting Health") #create plot for
decision tree
text(dtree, pretty = 0, use.n = TRUE, cex = .9) #add text to decision tree

## Decision Tree -- Pruning
printcp(dtree) #cost complexity tuning parameters

plotcp(dtree, upper = c("none"), main = "Cross Validated Cost Complexity Results") #plot cross-
validated cost complexity results

dtree_pruned <- prune(dtree, cp=0.022) #prune tree using the relevant cp value

plot(dtree_pruned, uniform = T, main = "Pruned Regression Decision Tree Predicting Health") #plot
pruned decision tree
text(dtree_pruned, pretty = 0, use.n = TRUE) #add text to pruned decision tree

## Decision Tree -- Prediction
pred_dtree <- predict(dtree_pruned, newdata = dat_test) #predict outcome value for test set using
the pruned decision tree
dtree_RMSE <- sqrt(mean((pred_dtree-dat_test$health)^2, na.rm=T)) #calculate RMSE values from
test set predictions
print(dtree_RMSE) #print RMSE value
```

Additional Code

```
####
# Random Forest
###
# Create a complete dataset
set.seed(44)
dat_complete <- dat[complete.cases(dat),] #takes only complete cases from the dataset
dat_complete_train <- dat_complete %>%
  sample_frac(size = 7/10) #sample 7/10 of the observations for the training dataset
dat_complete_test <- dat_complete %>%
  setdiff(dat_complete_train) #take the rest for the test dataset

# Random Forest
set.seed(44)

rf <- randomForest(health ~ ., data = dat_complete_train, ntree = 500, importance = TRUE)
#function to create a random forest using default values for ntree and mtry
sqrt(mean(rf$mse)) #OOB RMSE value

## Tuning RF
set.seed(44)
tuneRF(dat_complete_train[,-3], dat_complete_train[,3], ntreeTry=1000, stepFactor=1.5,
improve=0.05,
       trace=TRUE, plot=TRUE, doBest=FALSE) #tries different values of mtry and looks at impact
on performance.

## Final RF model
set.seed(44)
rf_final <- randomForest(health ~ ., mtry = 48, ntree = 500, importance = TRUE, data =
dat_complete_train) #final model using the mtry=48 value from above.

## Variable importance
varImpPlot(rf_final, type = 1, main = "Variable Importance for Random Forest Regression")
#creates importance plot

# Evaluate RF performance
pred_rf = predict(rf_final, newdata = dat_complete_test) #predict outcome value for test set
using the random forest
rf_RMSE = sqrt(mean((pred_rf-dat_complete_test$health)^2)) #calculate RMSE values from test set
predictions
print(rf_RMSE)
```

Additional Code

```
## Multiple Regression
multreg = lm(health ~ ., data = dat_train) #create a multiple regression model predicting health
pred_multreg = predict(multreg, newdata = dat_test, type = "response") #predcit outcome value for
test set using multiple regression
multreg_RMSE = sqrt(mean((pred_multreg -dat_test$health)^2, na.rm = T)) #calculate RMSE values
from test set predictions
print(multreg_RMSE)
```

Additional Code

```
########################
## R Script accompanying "Recursive Binary Partitioning and the Random Forest: An Introduction to
Tree-Based Machine Learning Methods in R"
## Code for supplemental classification example
## Authors: Andrew N Hall, David M Condon, Daniel K Mroczek
########################

## Load packages
packages = c("psych", "rpart", "randomForest", "dplyr", "ggplot2")
#install.packages(packages)
library(psych); library(rpart); library(randomForest); library(dplyr); library(ggplot2)

## Extract data from psych package
dat <- spi # using same dataset as regression but will manipulate one variable to create binary.
dat <-  dat[complete.cases(dat),] #take only complete cases for this example
dat <- dat %>%
  mutate(ER = if_else(ER == 1, 0, 1)) %>% #create binary variable for binary classification. In
original data, 1 = Never been to ER, 2-4 represented increasing number of visits. We create a
binary variable such that 0 = Never been to ER, 1 = Been to ER at least once.
  mutate(ER = as.factor(ER))

## Recursive Binary Partitioning (Decision Tree) for Classification
## Separate training and test datasets
set.seed(44)
dat_train <- dat %>%
  sample_frac(size = 7/10) #sample 7/10 of the observations for the training dataset
dat_test <- dat %>%
  setdiff(dat_train) #take the rest for the test dataset

## Build decision tree
dtree <- rpart(ER~., method = "class", parms = list(split= "gini"), data = dat_train) #construct
classification decision tree

plot(dtree, uniform = T, main = "Decision Tree for Classification of ER Visits") #create decision
tree plot
text(dtree, pretty = 0) #add text to decision tree plot

pred_dtree <- predict(dtree, newdata = dat_test, type = "class") #make predictions on test set
cm_dtree <- table(pred_dtree, dat_test$ER) #create a confusion matrix of accurate vs. inaccurate
predictions. We see the model is doing a poor job of classifying people who did visit ER!
(cm_dtree[1] + cm_dtree[4])/nrow(dat_test) #calculate
```

Additional Code

```
## Random Forest Classification

set.seed(44)
tuneRF(dat_train[,-57], dat_train[,57], mtryStart = 8, ntreeTry=100, stepFactor=2, improve=0.05,
       trace=TRUE, plot=TRUE, doBest=FALSE) #run tuning plot of mtry vs. OOB error. Tells us to
use mtry = 32.

set.seed(44)
rf <- randomForest(ER ~ ., data = dat_train, ntree = 500, importance = TRUE, mtry = 32) #run
random forest model using mtry = 32

varImpPlot(rf, type = 1) #variable importance plot. type = 1 tells it to only select the plot
based on accuracy.

pred_rf <- predict(rf, newdata = dat_test, type = "class") #construct predictions on the test
dataset
cm_rf <- table(pred_rf, dat_test$ER) #construct the confustion matrix. We see RF model is
predicting everyone to be a value of 0! Thus, accuracy may be high, but it will just be the
proportion of people who reported 0. Illustrates a danger in only reporting overall
classification accuracy.
(cm_rf[1] + cm_rf[4])/nrow(dat_test) #overall accuracy of RF
```

Additional Code

```
## Comparison to logistic regression
logreg <- glm(ER ~ ., family = "binomial", data = dat_train) #construct logistic regression model
with ER as binary outcome

pred_logreg <- predict(logreg, newdata = dat_test, type = "response") #make predictions on test
dataset making "response" outcomes of probability of inclusion in a class.
pred_logreg <- if_else(pred_logreg > 0.5, 1, 0) #cutoff for probability of inclusion. Here we use
0.5, which is arbitrary. Would likely want a different cutoff due to unbalanced groupings in a
real scenario.
pred_logreg <- as.factor(pred_logreg) #make predictions a factor
cm_log <- table(pred_logreg, dat_test$ER) #construct confusion matrix of predictions by actual
values.
(cm_log[1] + cm_log[4])/nrow(dat_test) #calculates overall accuracy rates
```

Additional Code

```
#########################
## R Script accompanying "Recursive Binary Partitioning and the Random Forest: An Introduction to
Tree-Based Machine Learning Methods in R"
## Code for examples in paper pre walkthrough section using iris dataset
## Authors: Andrew N Hall, David M Condon, Daniel K Mroczek
#########################

# In-text example code using iris (pre-tutorial section)
library(datasets) #load datasets library
library(tidyverse) #load tidyverse for data manipulation
library(rpart) #load rpart for construction of decision tree
data(iris)

# Split data into training and test datasets
set.seed(44)
iris_train = sample_frac(iris, size = 1/2) #sample 1/2 of observations for training set
iris_test = setdiff(iris, iris_train) #take remaining 1/2 for test set

# Decision tree Iris
dtree_iris <- rpart(Species~., data = iris_train) #create basic decision tree based on iris data

plot(dtree_iris, uniform = T, main = "Example Decision Tree Using Iris Dataset") #plot decision
tree
text(dtree_iris, pretty = 0, use.n = T, cex = .9) #add text to decision tree
```