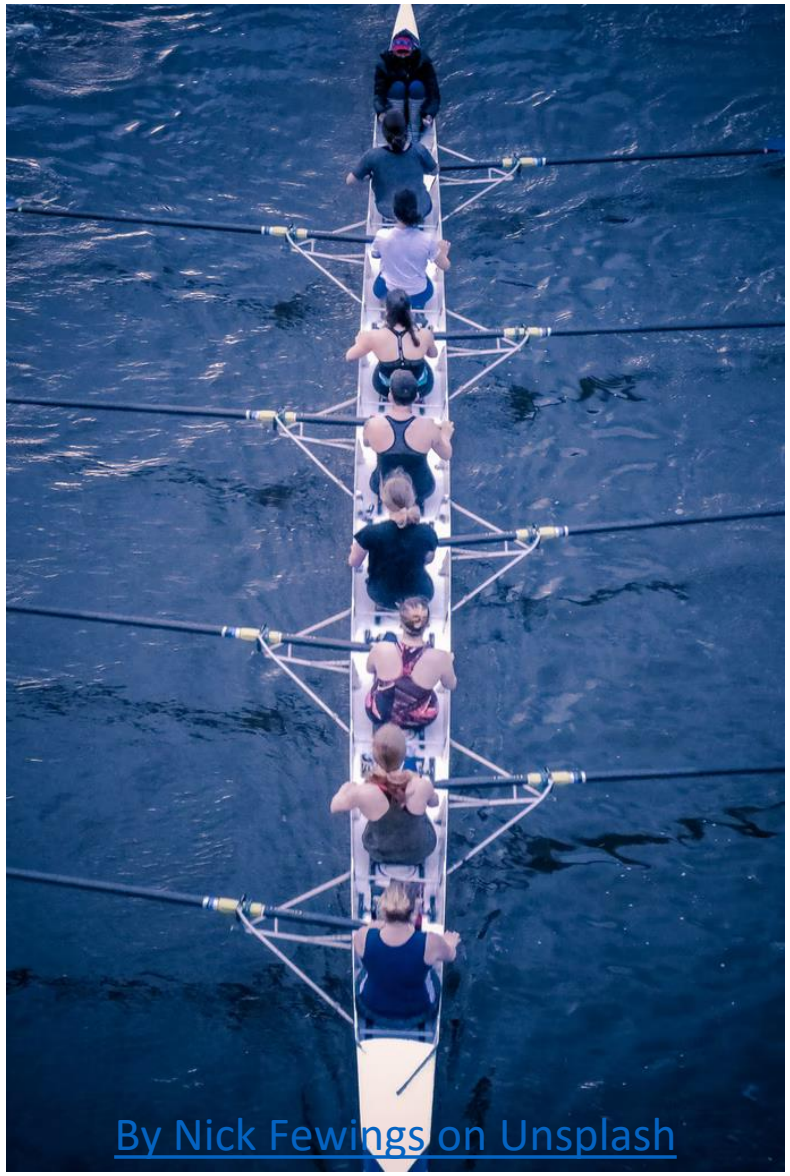
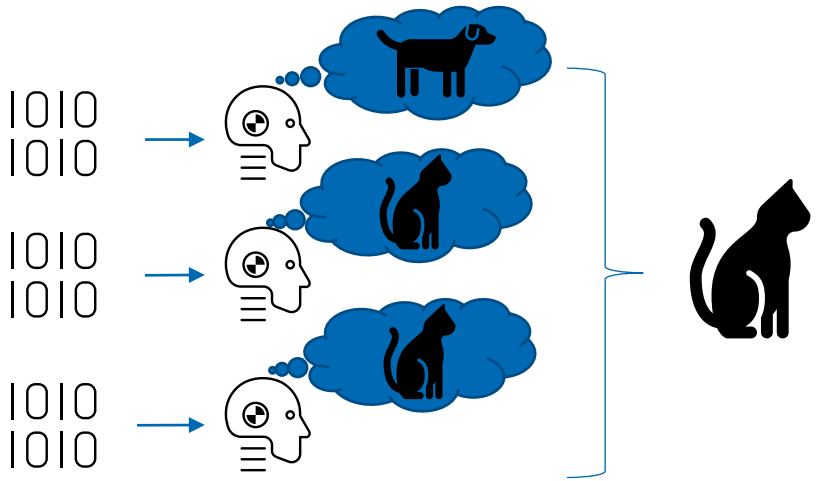
Data Science for Consumer Behavior

Gradient Boosting Trees

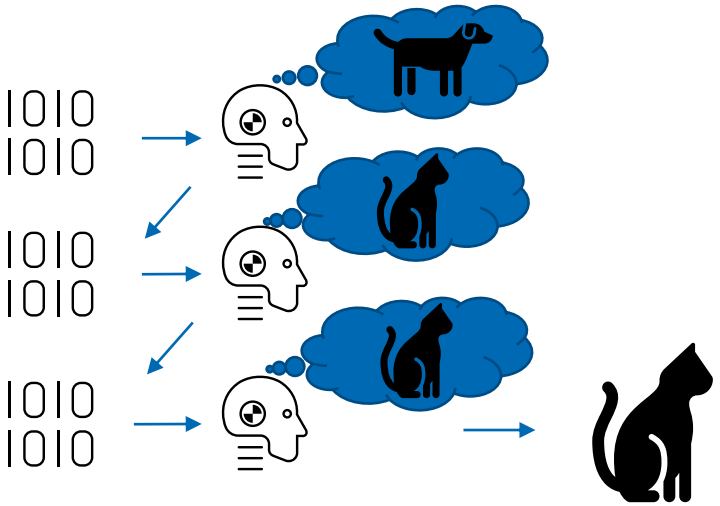


Recap: Ensemble Methods

- Teamwork as metaphor
- Basic Idea: Create a “strong“ learner by means of several “weak” learners
- Two approaches



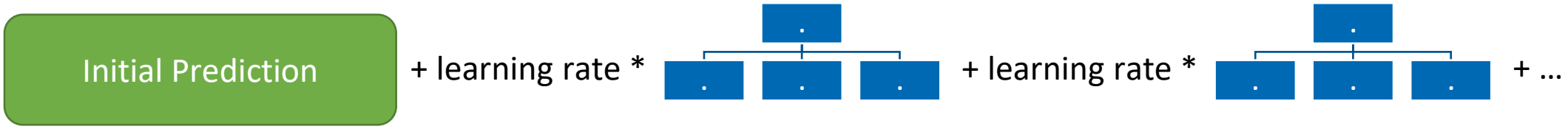
bagging



boosting

Gradient Boosting Trees

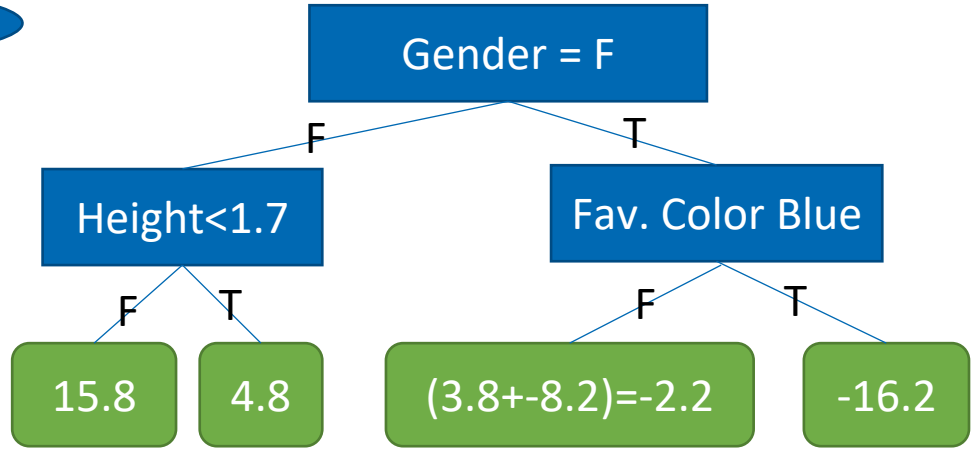
- Basic idea:

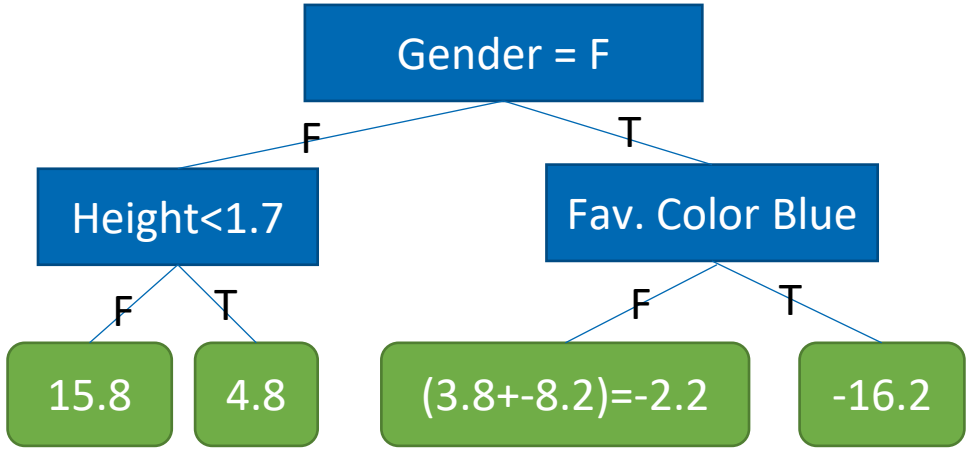


- Easy to understand conceptually with regression example:

Average 72.2 kg

Height (m)	Favorite Color	Gender	Weight (kg)	Residual
1.6	Blue	Male	88	88-72.2=15.8
1.7	Green	Female	76	3.8
1.5	Blue	Female	56	-16.2
1.8	Red	Male	77	4.8
1.6	Green	Female	64	-8.2





Initial Prediction + learning rate 0.8 * 

Average 72.2 kg

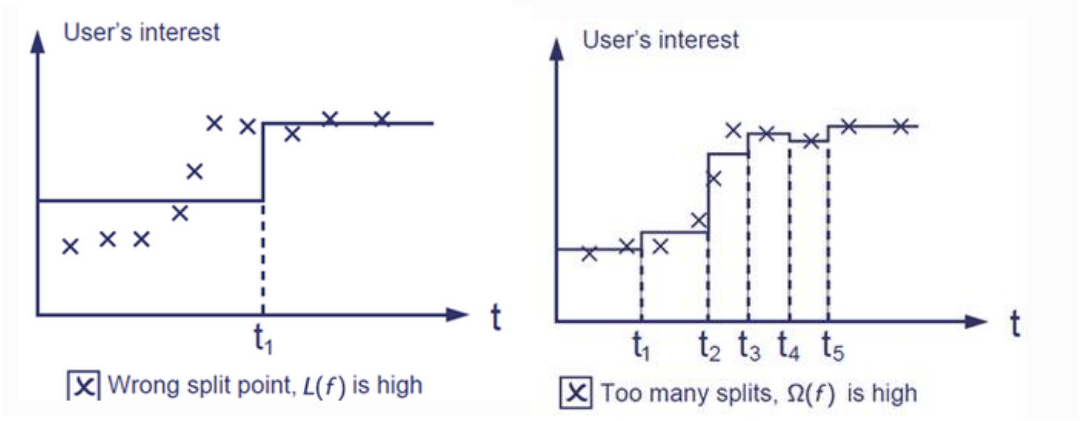
Height (m)	Favorite Color	Gender	Weight (kg)	Residual	2. Prediction
1.6	Blue	Male	88	88-72.2=15.8	72.2+0.8*15.8 = 84.84
1.7	Green	Female	76	3.8	72.2+0.8*-2.2 = 70.44
1.5	Blue	Female	56	-16.2	72.2+0.8*-16.2 = 59.24
1.8	Red	Male	77	4.8	72.2+0.8*4.8 = 76.04
1.6	Green	Female	64	-8.2	72.2+0.8*-2.2 = 70.44

Next step: 2. Prediction + learning rate 0.1 * 

XGBoost

- XGBoost ≡ eXtreme Gradient Boosting
- Challenge with Gradient Boosted Trees:
  How to find the best (next) tree?
- Regularization
- Reduces search space of possible feature splits with an approximation heuristic

$$\text{obj}(\theta) = L(\theta) + \Omega(\theta)$$



- R package available:
  install.packages("xgboost"); library(xgboost)
- Caveat 1: **Numeric class labels** required
  data <- factor(c("a", "b", "c"))
  label = as.integer(data$label) - 1
- Caveat 2: XGBoost uses **own matrix datatype**
  xgb.train = xgb.DMatrix(data=train.data,label=train.label)

```r
# load data
data(agaricus.train, package='xgboost')
data(agaricus.test, package='xgboost')
train <- agaricus.train
test <- agaricus.test
# fit model
bst <- xgboost(data = train$data, label = train$label, max.depth = 2, eta = 1, nrounds = 2,
               nthread = 2, objective = "binary:logistic")
# predict
pred <- predict(bst, test$data)
```

Literature

Chen, T., & Guestrin, C. (2016, August). Xgboost: A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining (pp. 785-794).

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. Annals of statistics, 1189-1232.

XGBoost …

- Homepage: https://xgboost.readthedocs.io
- Talk: https://dl.acm.org/doi/abs/10.1145/2939672.2939785
- R documentation: https://cran.r-project.org/web/packages/xgboost/xgboost.pdf
- Tutorial with IRIS data set: https://www.rpubs.com/dalekube/XGBoost-Iris-Classification-Example-in-R

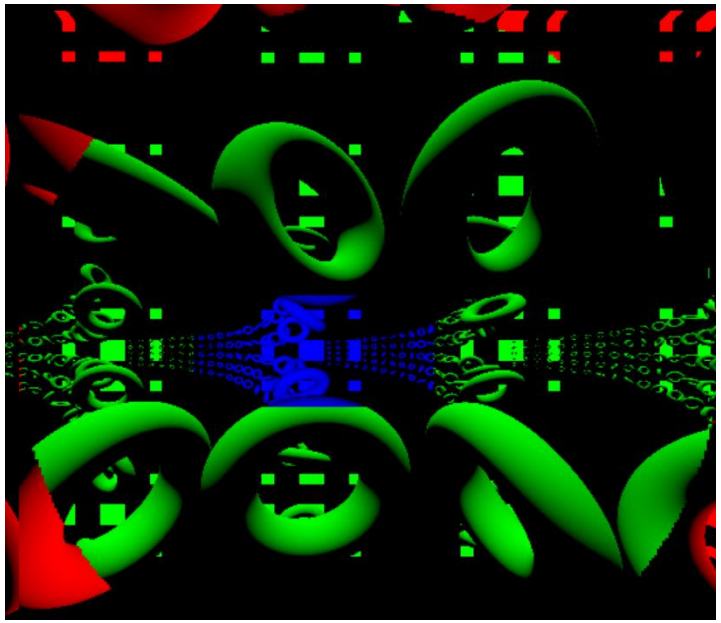
Data Science for Consumer Behavior

XGBoost Use Case: Eye Tracking Data Classification

Motivation: Eye Tracking (ET)

- High-end head-mounted displays include ET today
- Consumer-grade head-mounted displays likely ET tomorrow
- Foveated rendering
- Gaze driven menus

Varjo VR-3

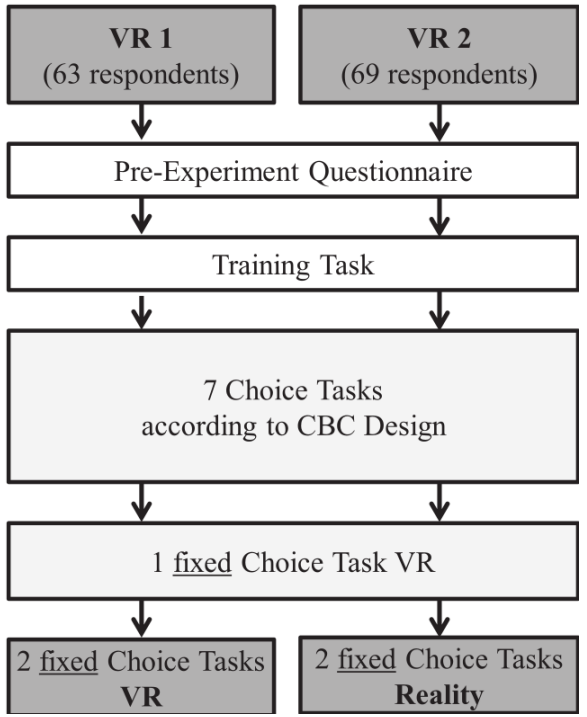


Gaze Driven Menu: HoloLense Example

https://docs.microsoft.com/en-us/dynamics365/mixed-reality/guides/authoring-gestures-hl2

Use Case: The Experiment

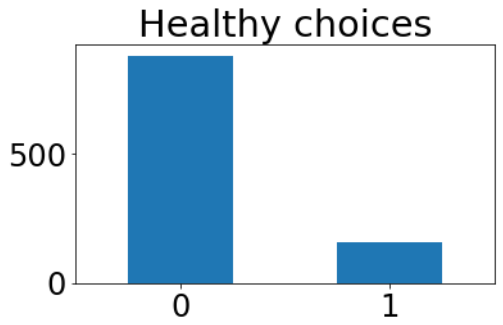
- Starting Point:
  Eye Tracking experiment in Virtual Reality about Muesli purchase decisions (Meißner et al., 2019; Peukert et al., 2019)
- Conjoint-Based Choice Experiment
- 130 participants (each 8 or 10 choice tasks in VR)



Experiment Design
by Meißner et al. (2019)

Use Case: The Data Set

- Goal: Predict healthy customer choices to feed RS
- Labelling criteria: packaging
- Key variables: fixation duration,
  Area of interest
- Challenge: class imbalances
  (7 out of 40 mueslis considered healthy)



Use Case: Feature Engineering

- Total number of fixations
- Average Fixation duration
- **Variance of Fixation durations**

- Number of fixations per product
- Total fixation duration per product
- Average fixation duration per product

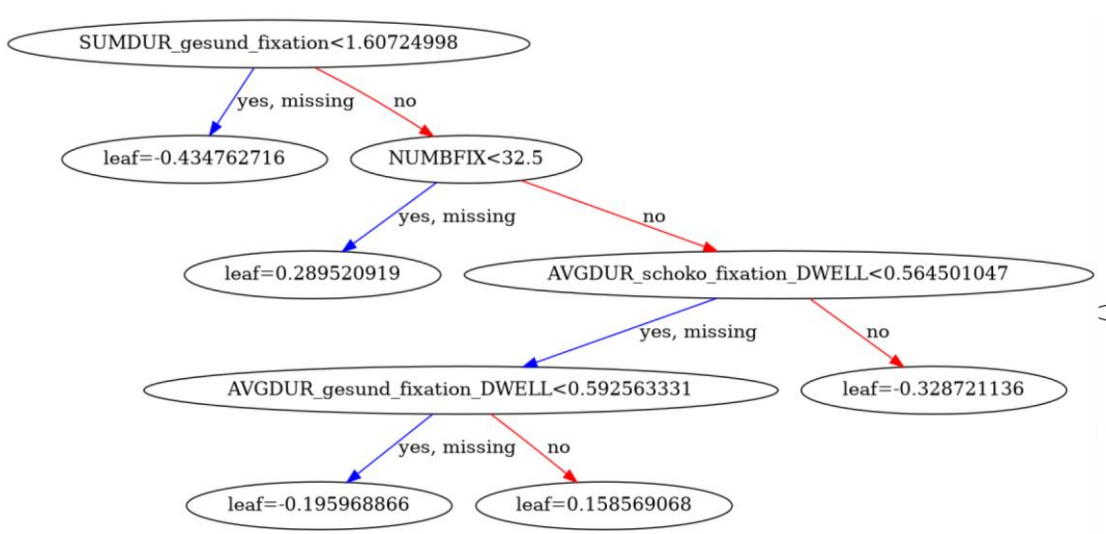
- Number of Fixations per category
- Total fixation duration per category
- Average fixation duration per category

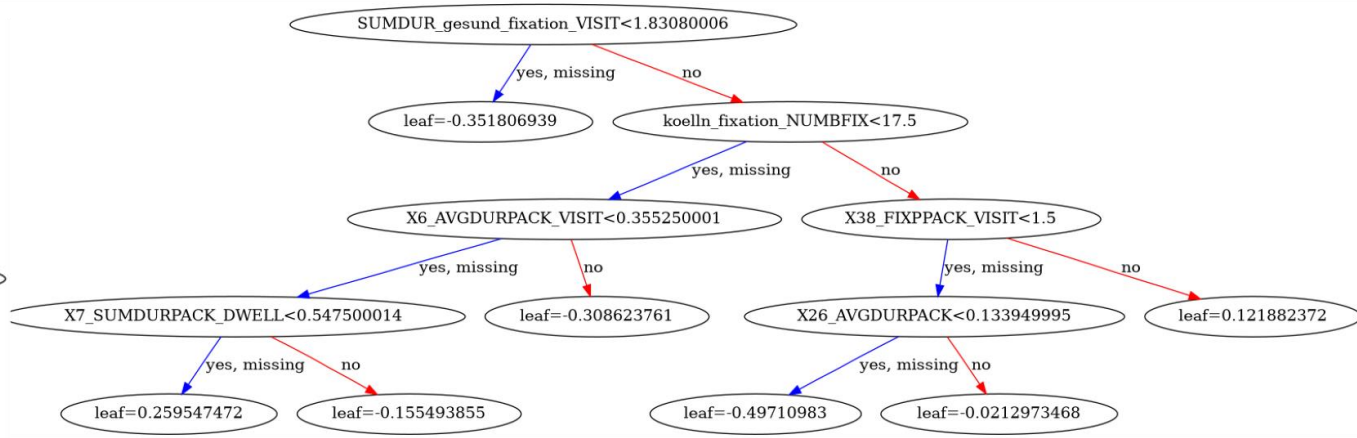
NUMBFIX
39
48
47
44
44
42
34
23
41
56
39
43
46

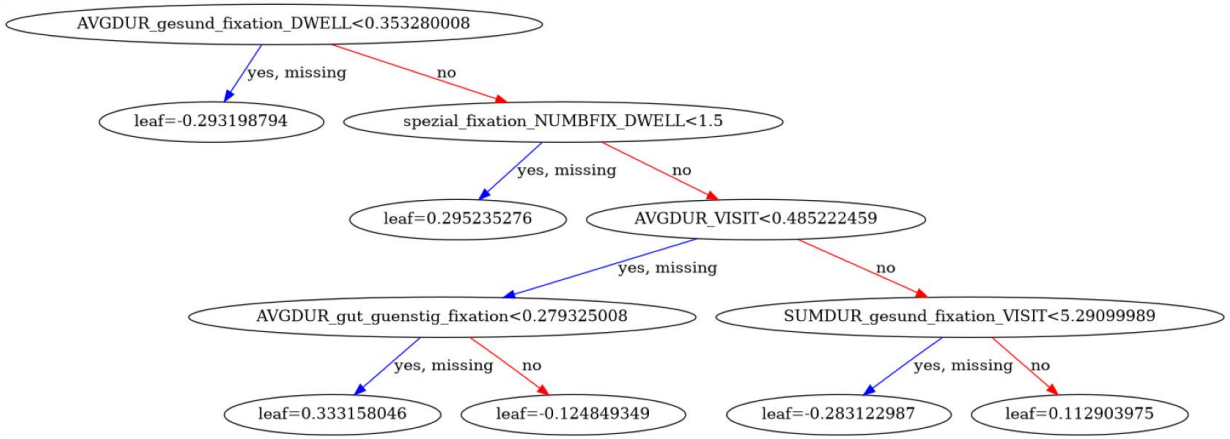
X26_FIXPPACK
0
2
1
1
0
1
2
0
0
1
18
0

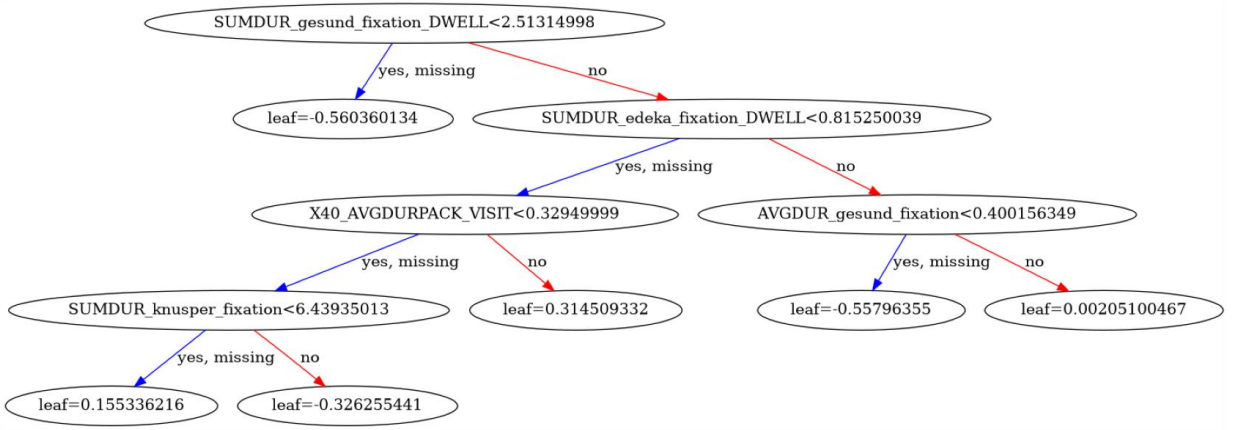
SUMDUR_gesund_fixation
1.6280
2.1670
1.6210
0.5910
3.9410
4.8710
1.6410
0.7380
2.3680
1.2400

Use Case: XGBoost in Action









XGBoost: Take away

- Boosting != Bagging
- Small Trees depending on each other
- State-of-the-art Gradient Boosted Tree library
- Almost Plug-and-play function replacement in R
  - Numeric labels
  - Dmatrix Data Type