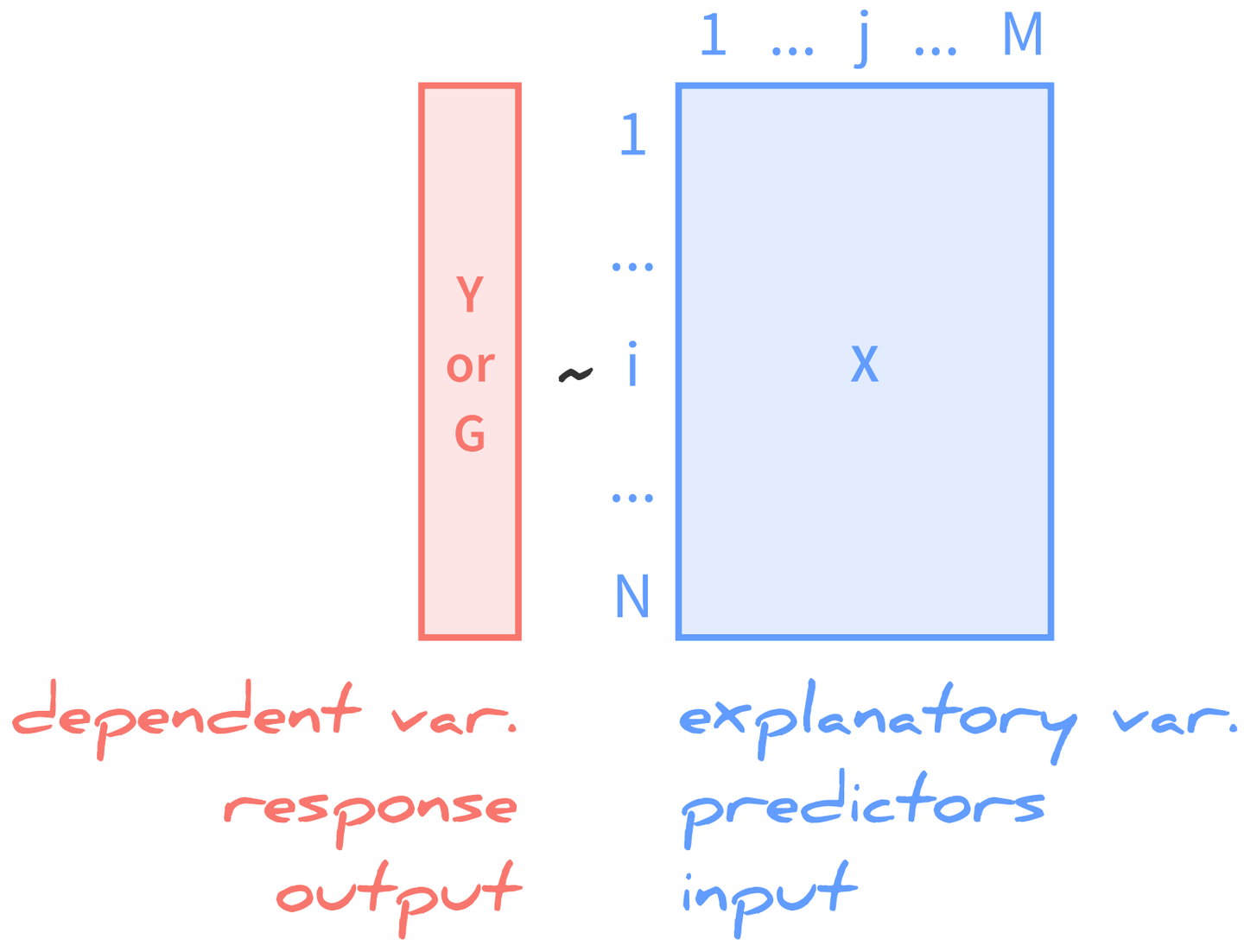
ML4Oceans
Jean-Olivier Irisson

Gradient Boosted Trees

For all Hobbits share a love of things that grow

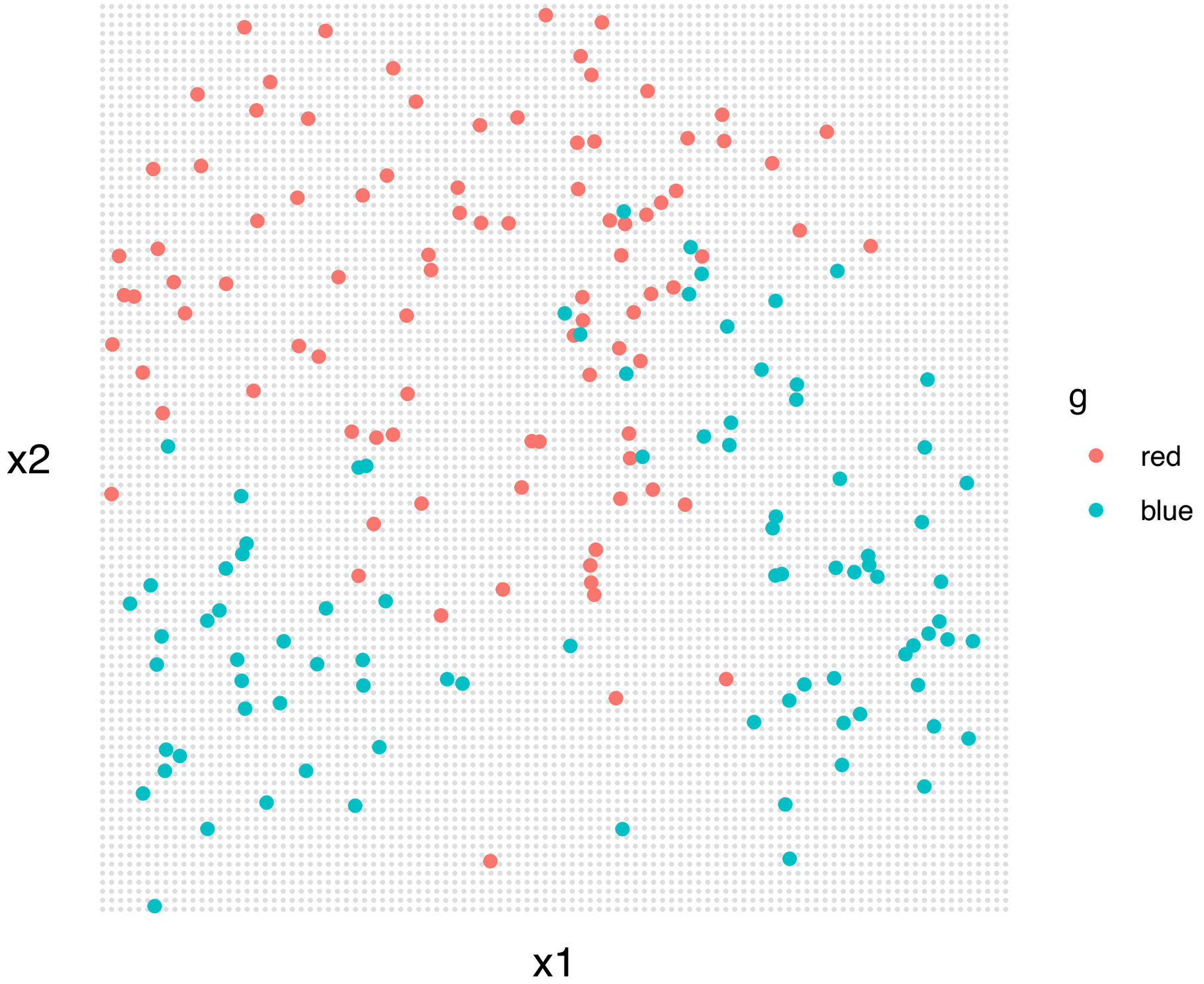
Vocabulary


1 … j … M
1
…
i
…
N
Y
or
G
~
X
dependent var.
response
output
explanatory var.
predictors
input

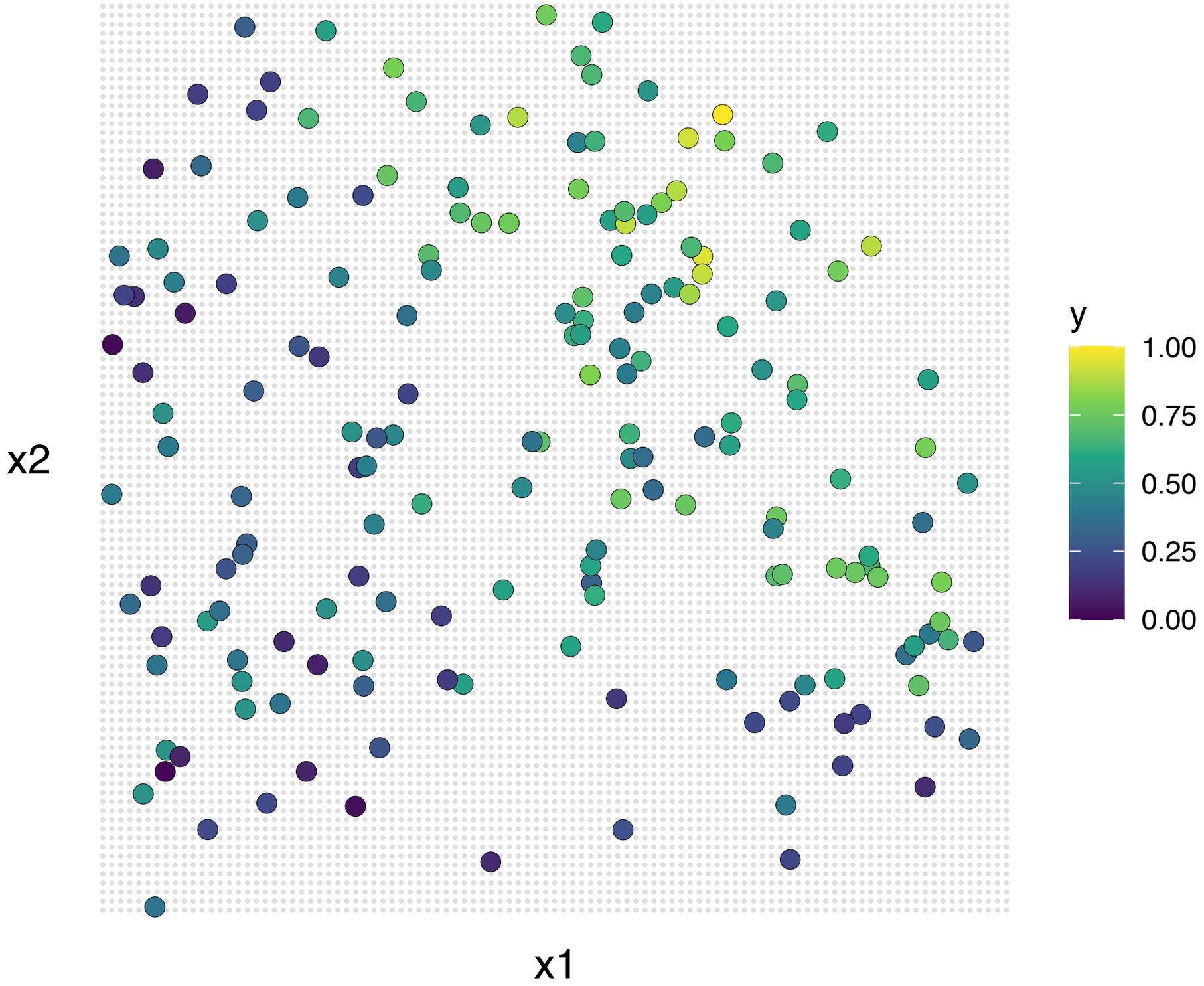
Classification and regression trees

It all starts with a branch

A simple classification problem



A simple *regression* problem



Classification And Regression Trees (CART)

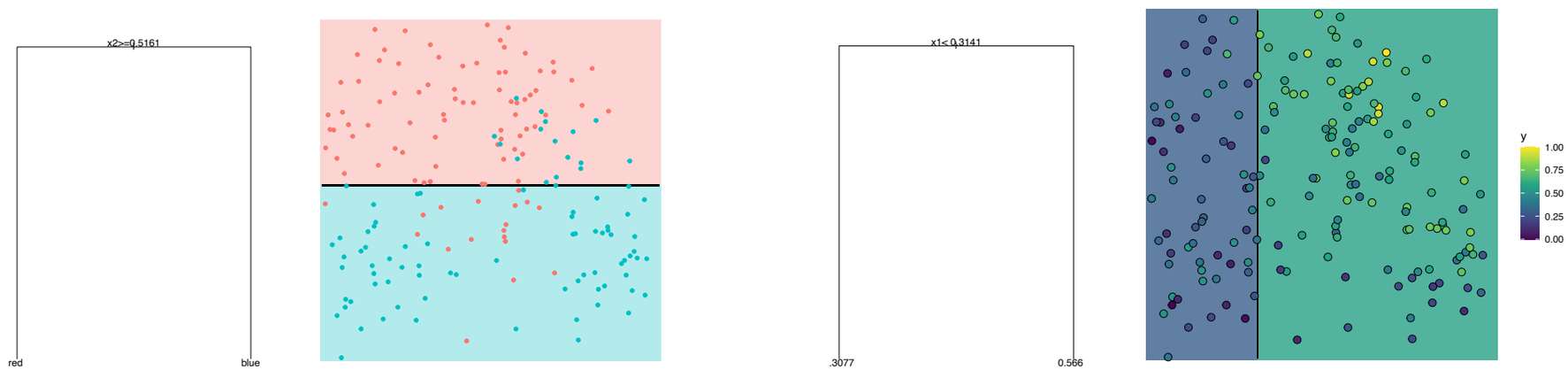
Binary splits

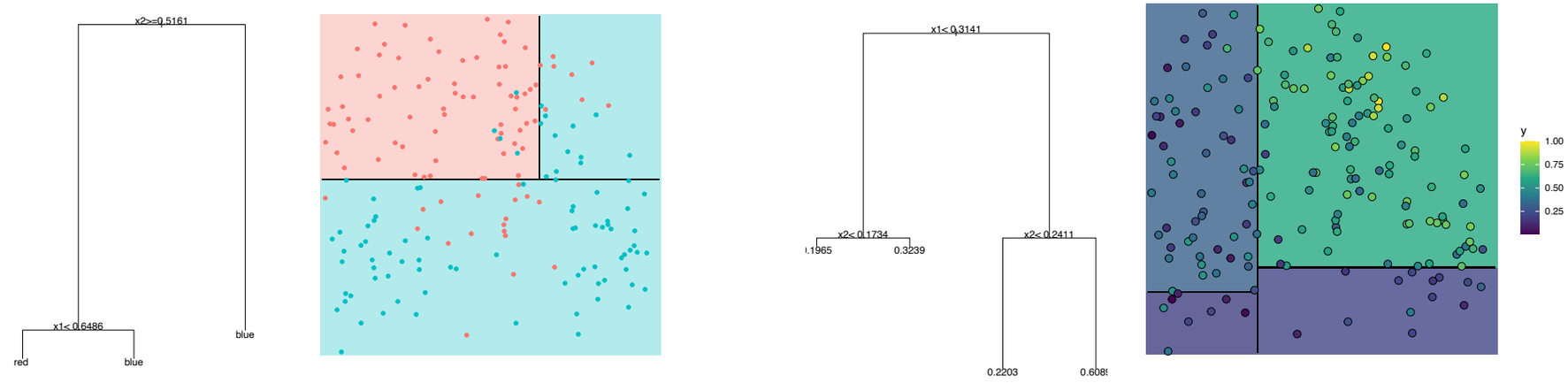
Based on **predictors**

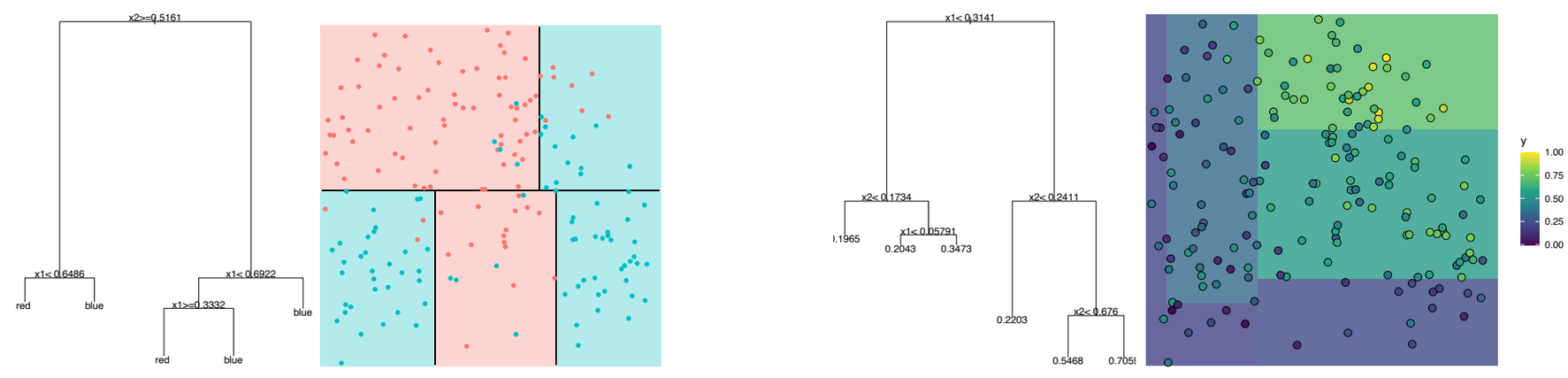
Categorical response = **classification**

Continuous response = **regression**

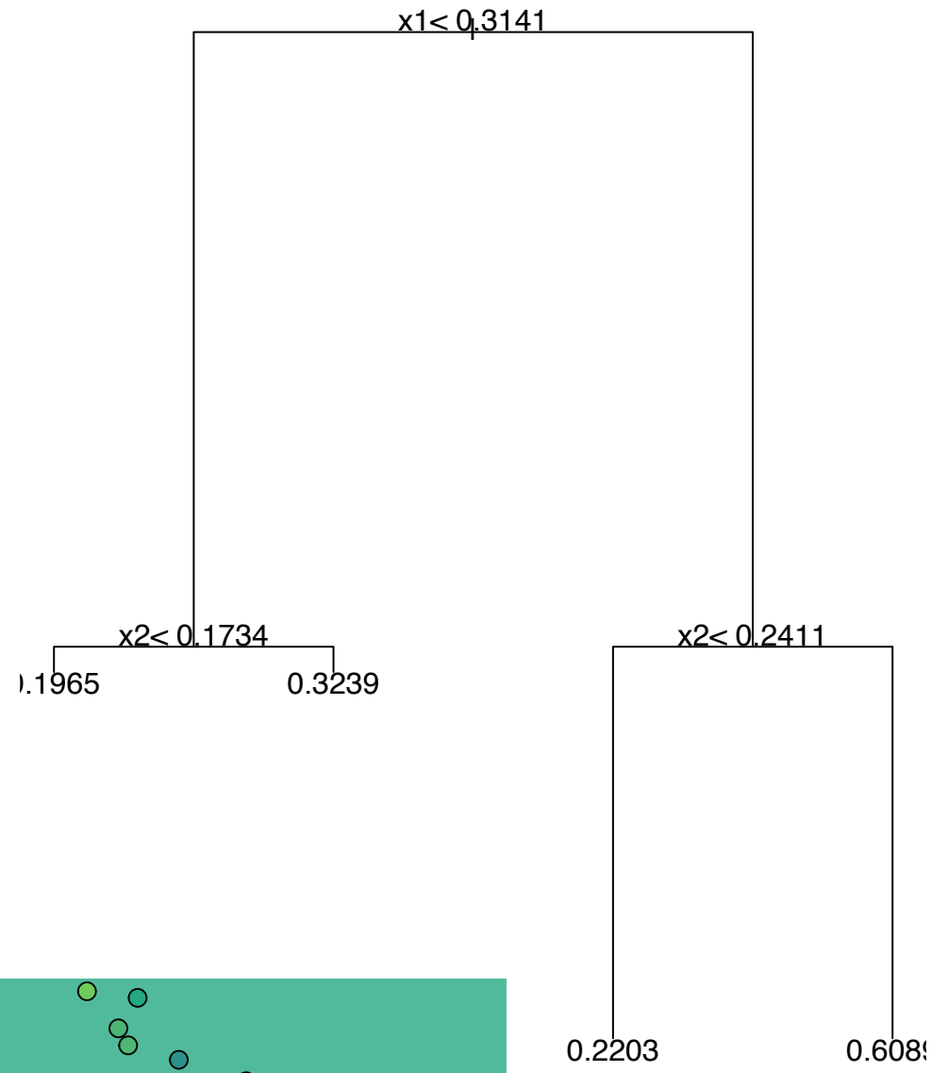
Predictors can be **continuous** or **categorical**







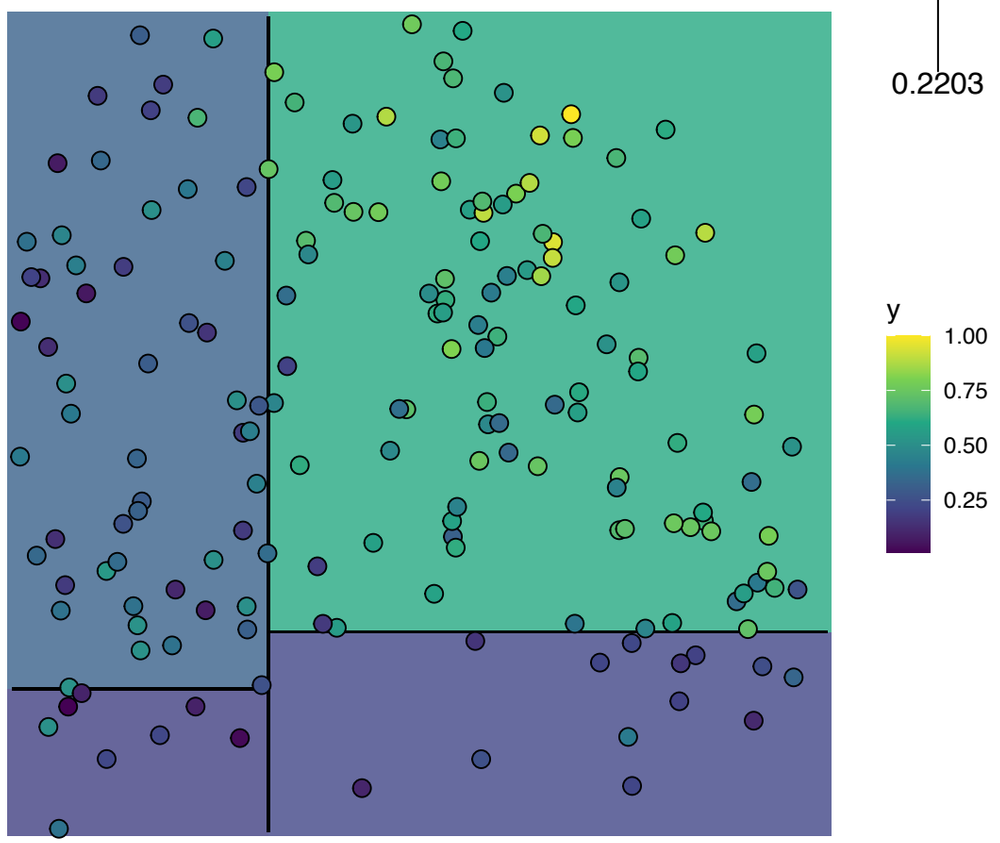
Regression trees



Constant in each leaf = **average** of observations in leaf

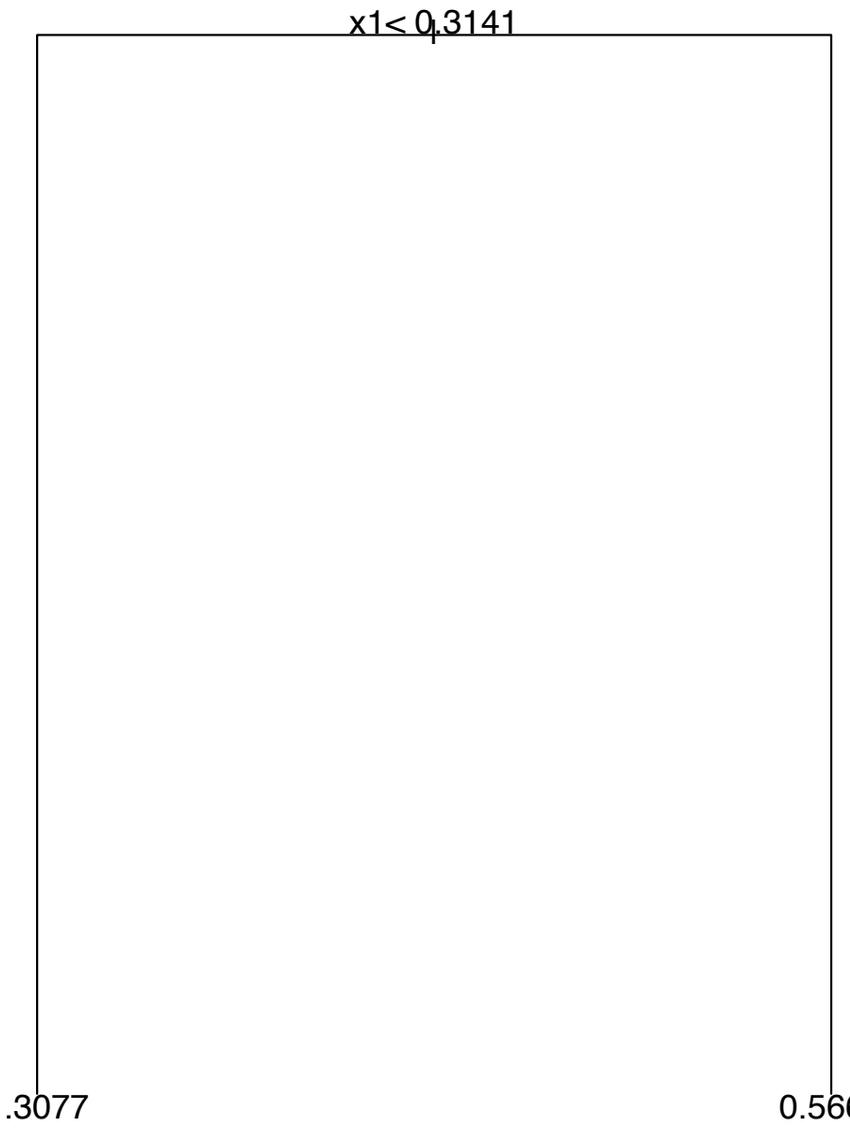
Minimise

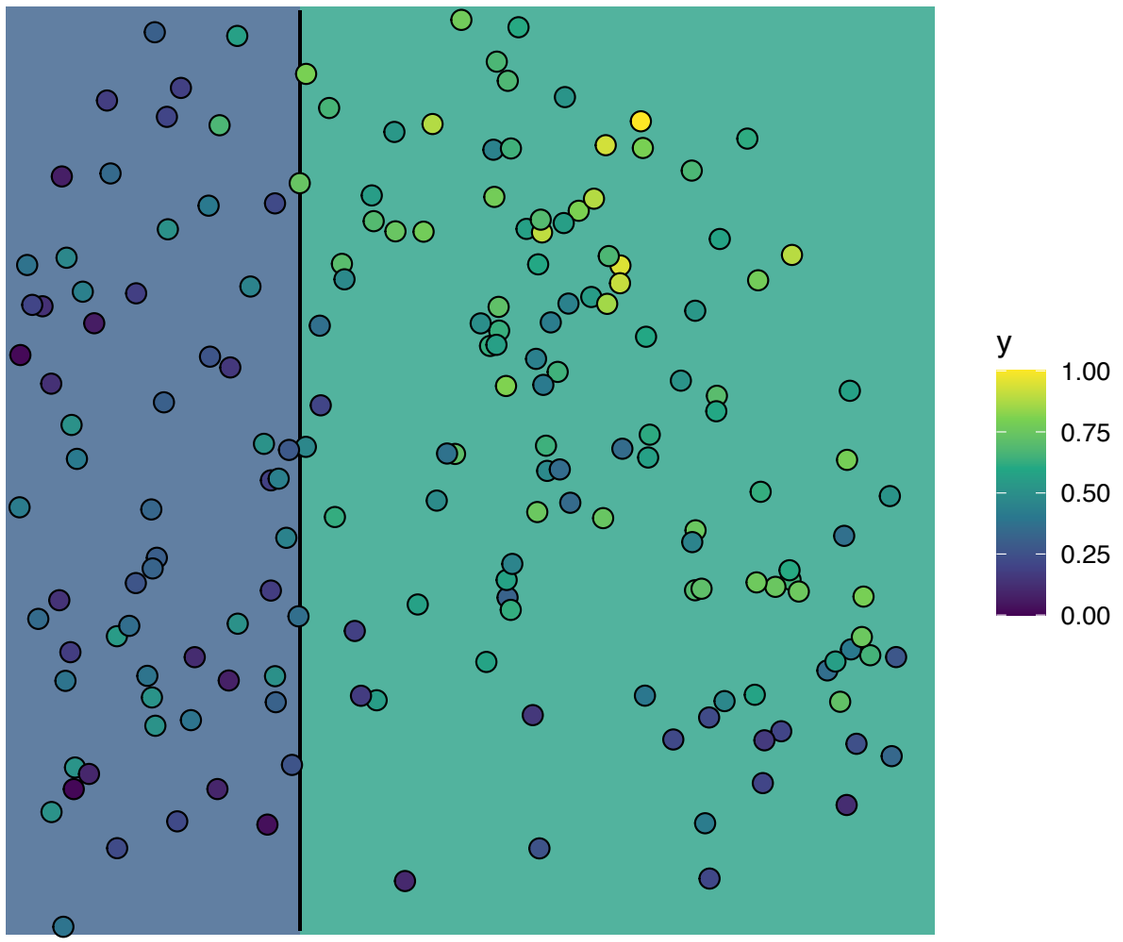
$$\sum_{r=1}^{R} \sum_{\forall i \in R_r} (y_i - \bar{y}_r)^2$$



Growing the tree

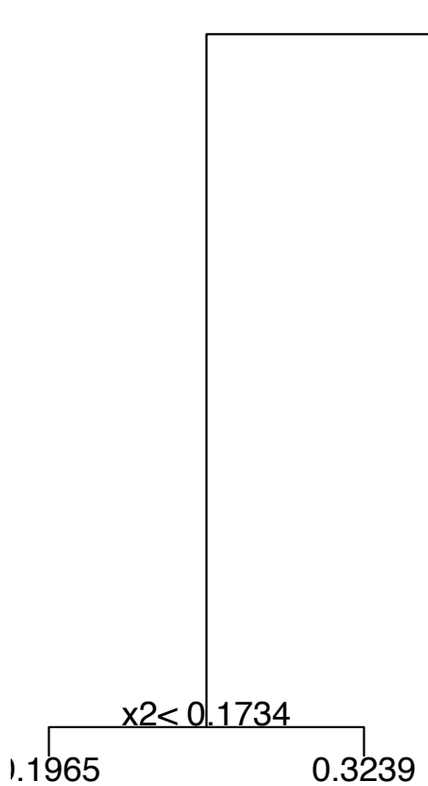
Finding the **completely optimal tree** is infeasible computationally. Use a **stepwise** algorithm ("greedy" method) and find the optimal variable and value for each split **successively**.

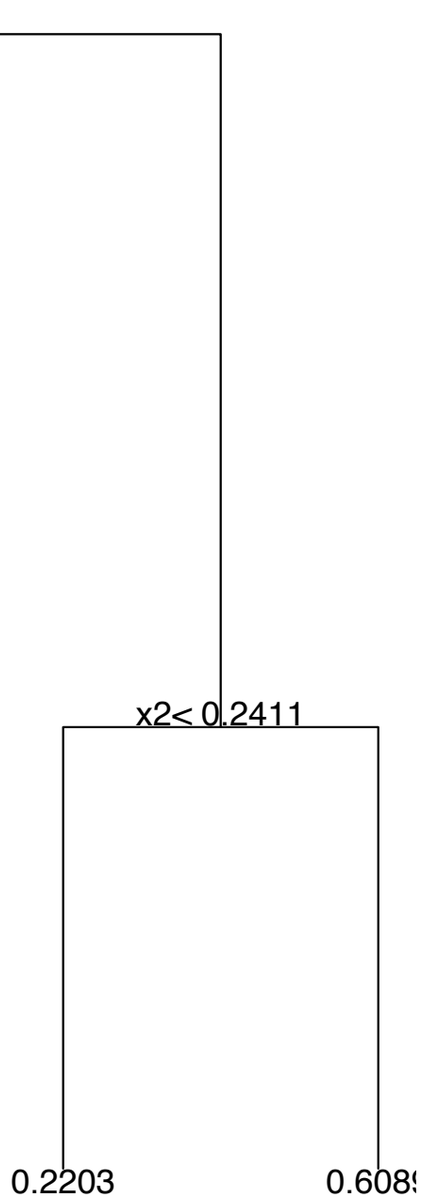


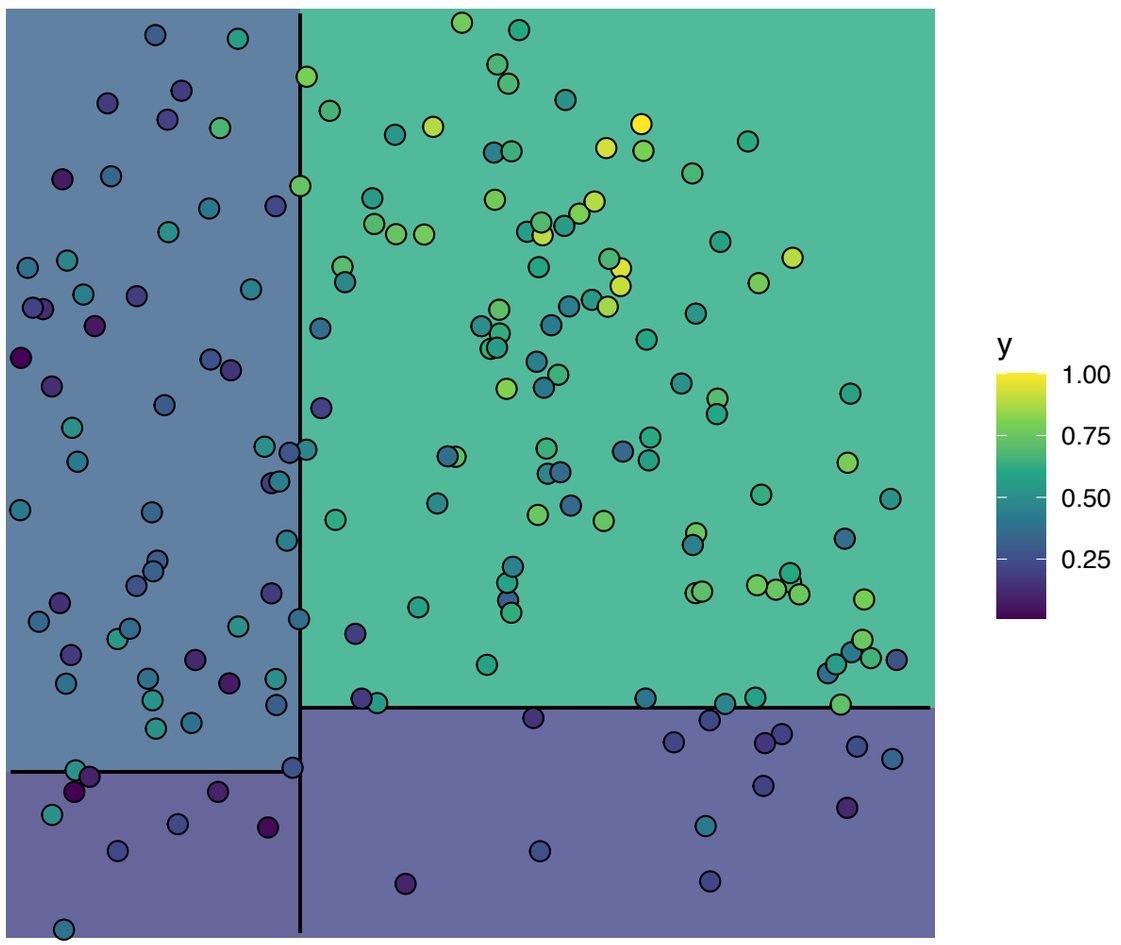


Growing the tree

Finding the **completely optimal tree** is infeasible computationally. Use a **stepwise** algorithm ("greedy" method) and find the optimal variable and value for each split **successively**.

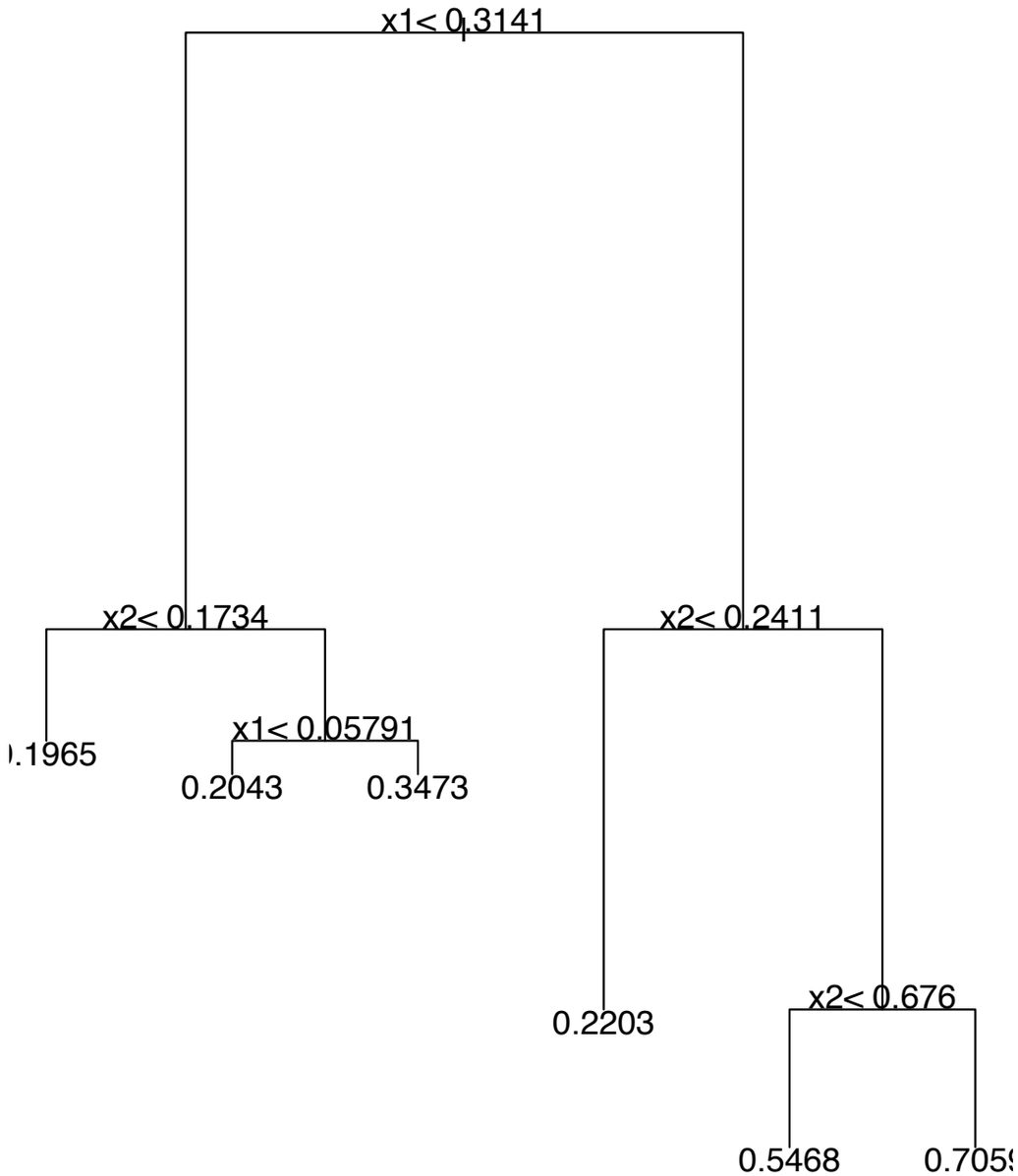


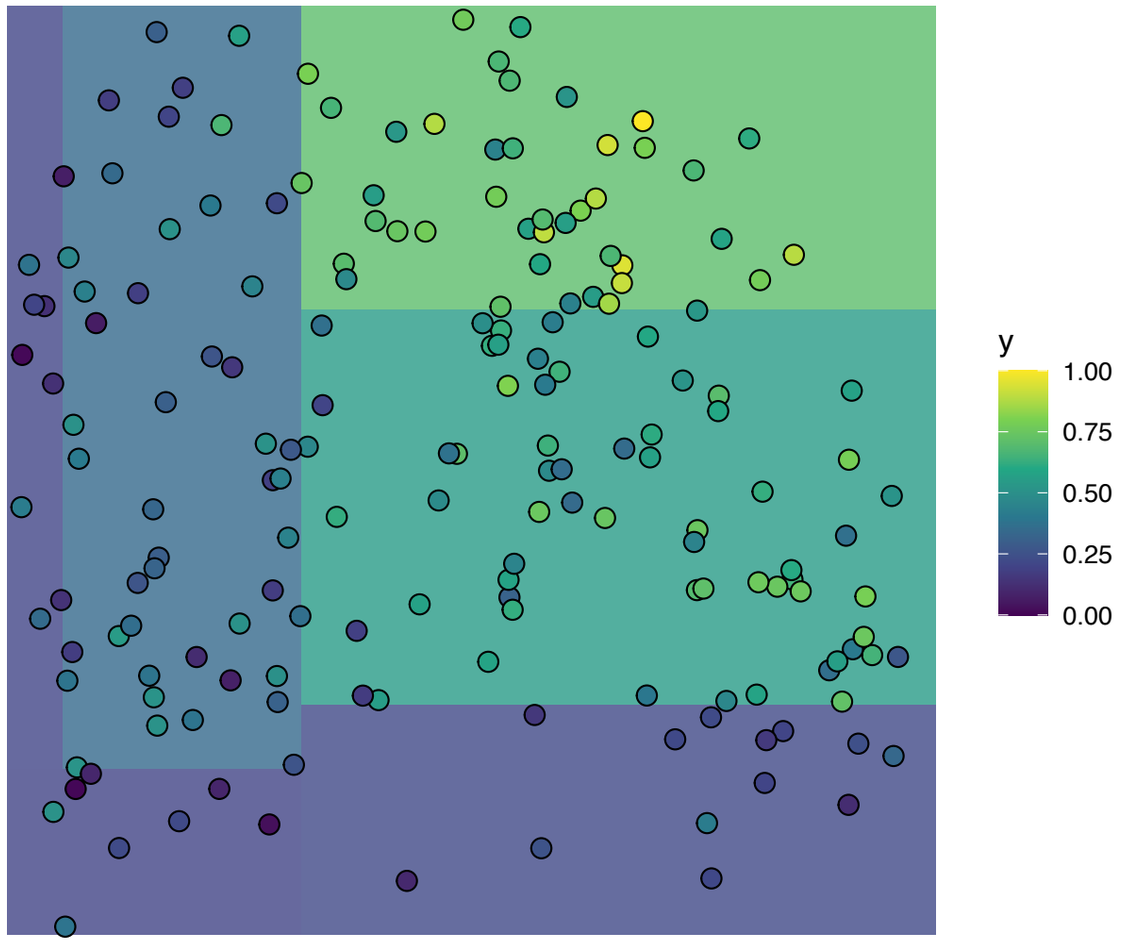




Growing the tree

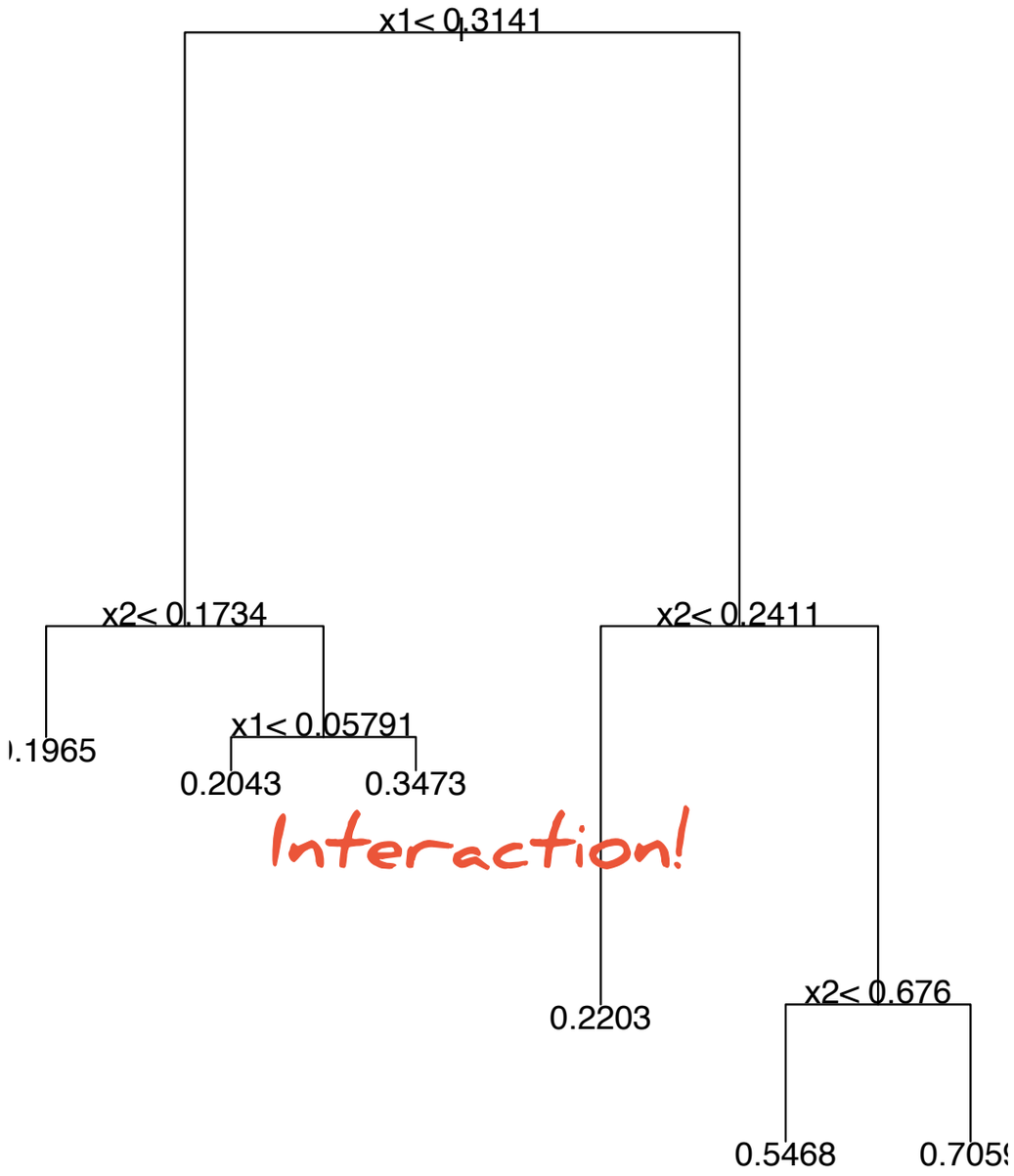
Finding the **completely optimal tree** is infeasible computationally. Use a **stepwise** algorithm ("greedy" method) and find the optimal variable and value for each split **successively**.

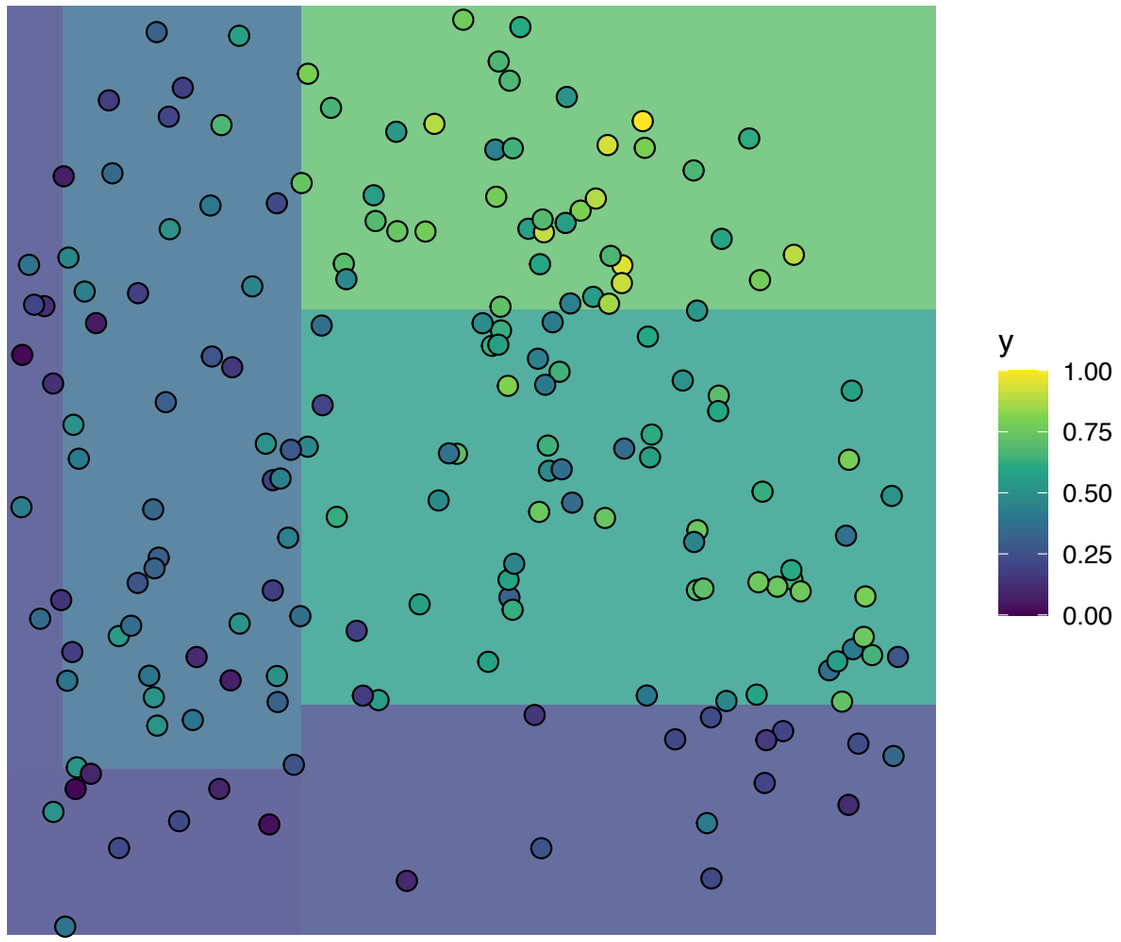




Growing the tree

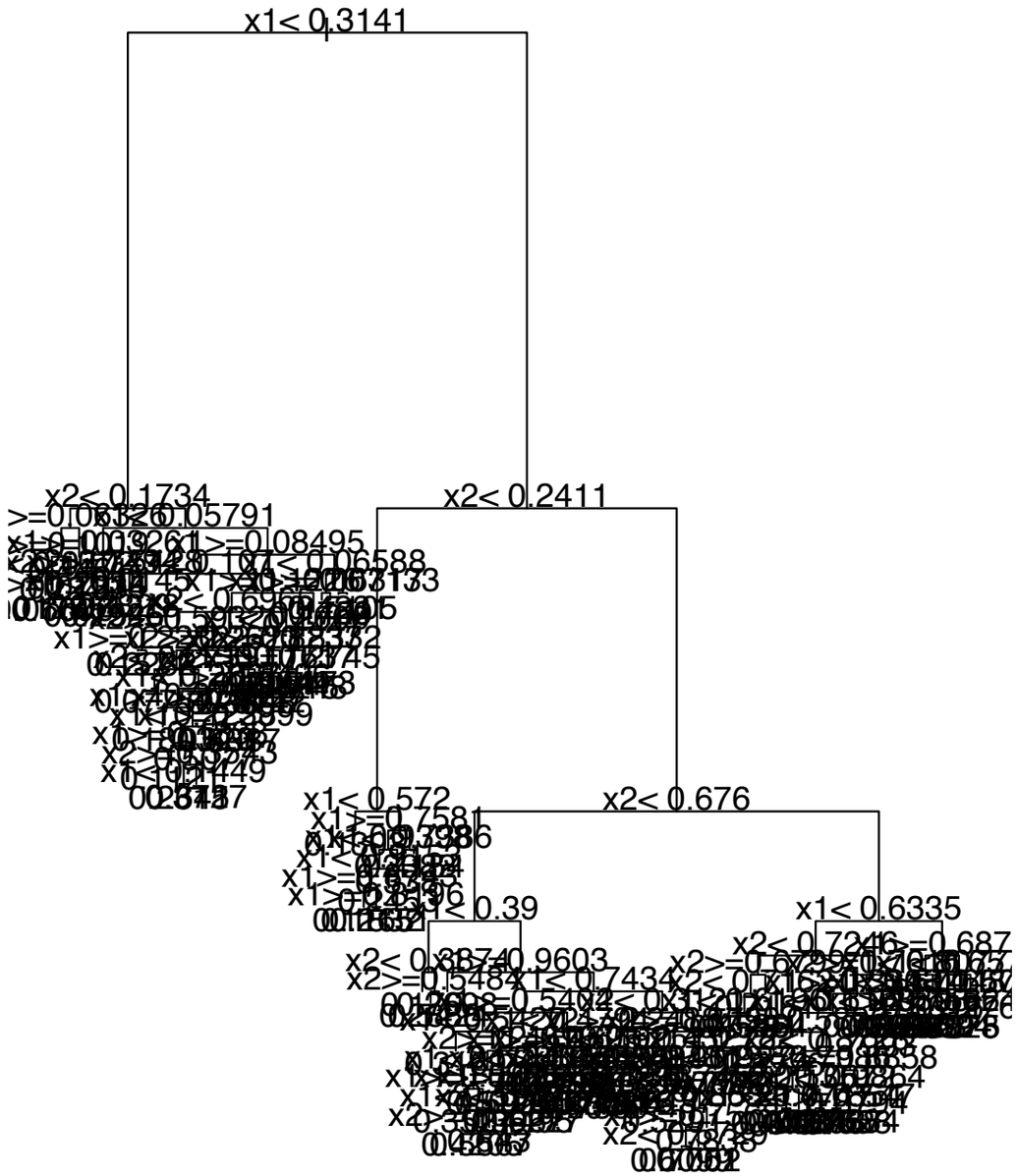
Finding the **completely optimal tree** is infeasible computationally. Use a **stepwise** algorithm ("greedy" method) and find the optimal variable and value for each split **successively**.

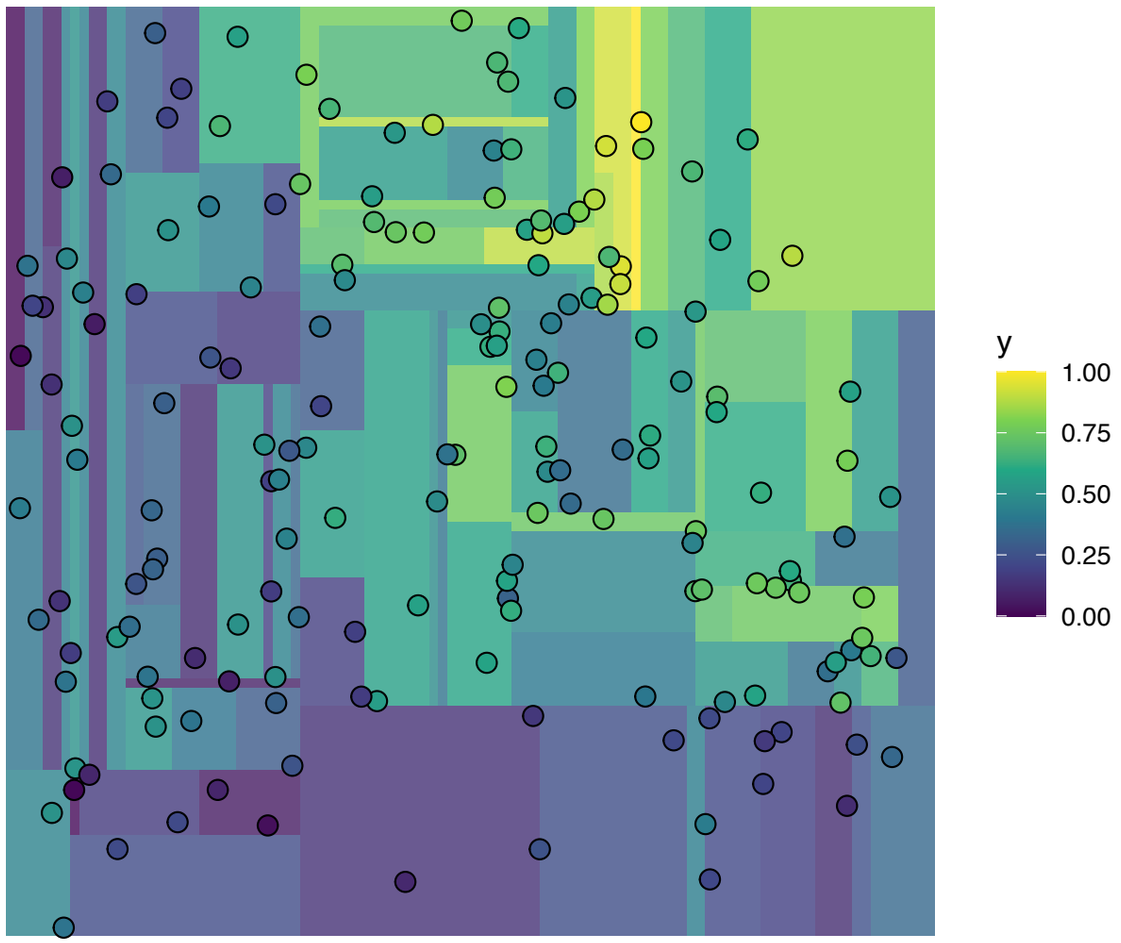




Growing the tree

Finding the **completely optimal tree** is infeasible computationally. Use a **stepwise** algorithm ("greedy" method) and find the optimal variable and value for each split **successively**.





Classification trees

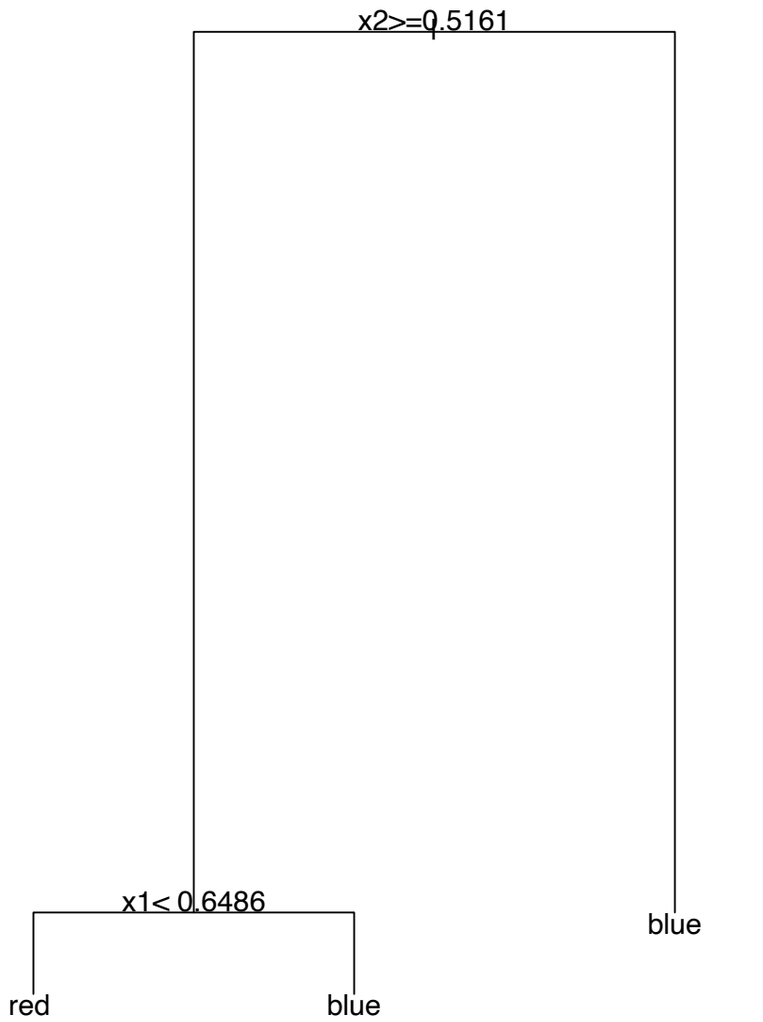
Constant in each leaf = **majority class** in the leaf

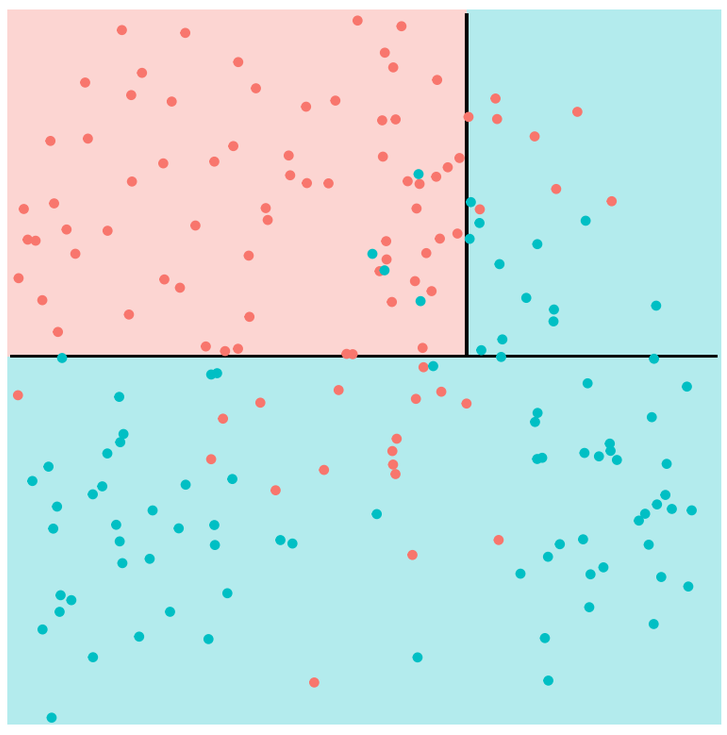
Minimise

$$\sum_{r=1}^{R} \frac{1}{N_r} \sum_{\forall i \in R_r} \mathbf{1}(y_i \neq k_r)$$

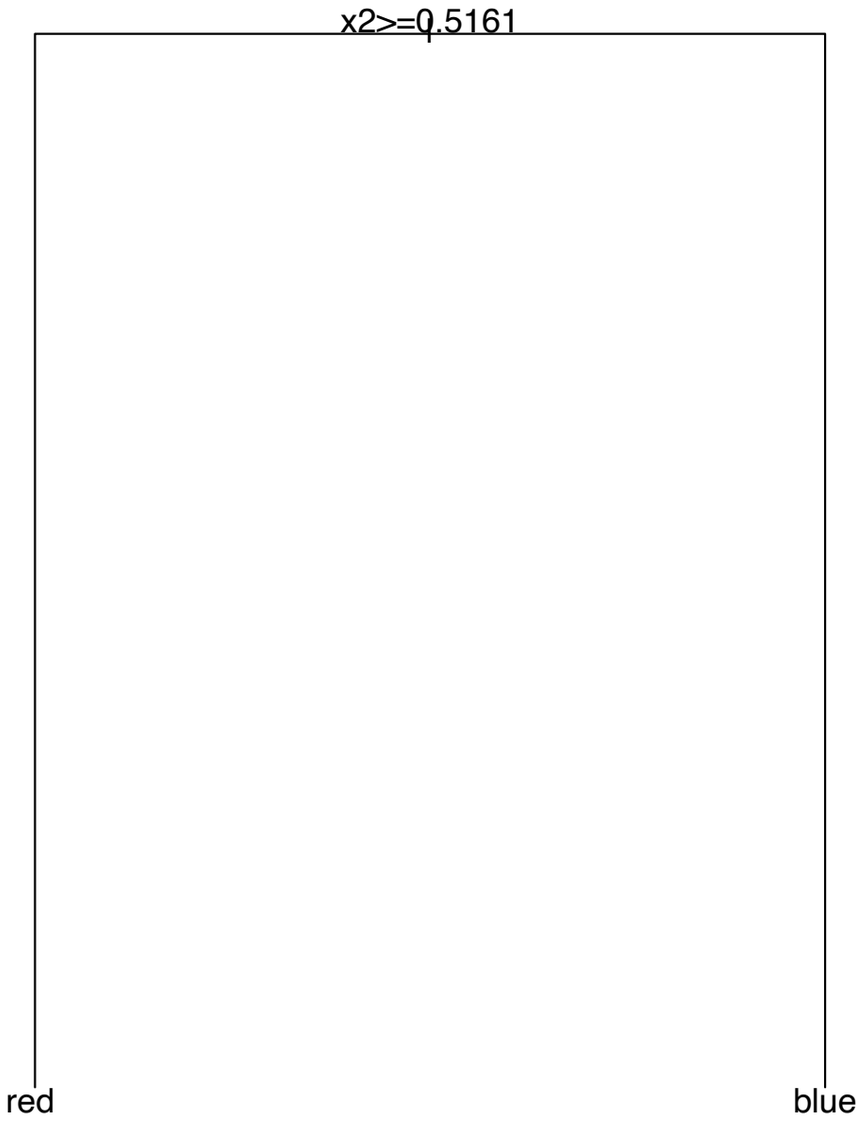
or Gini impurity, or Cross-entropy, etc.

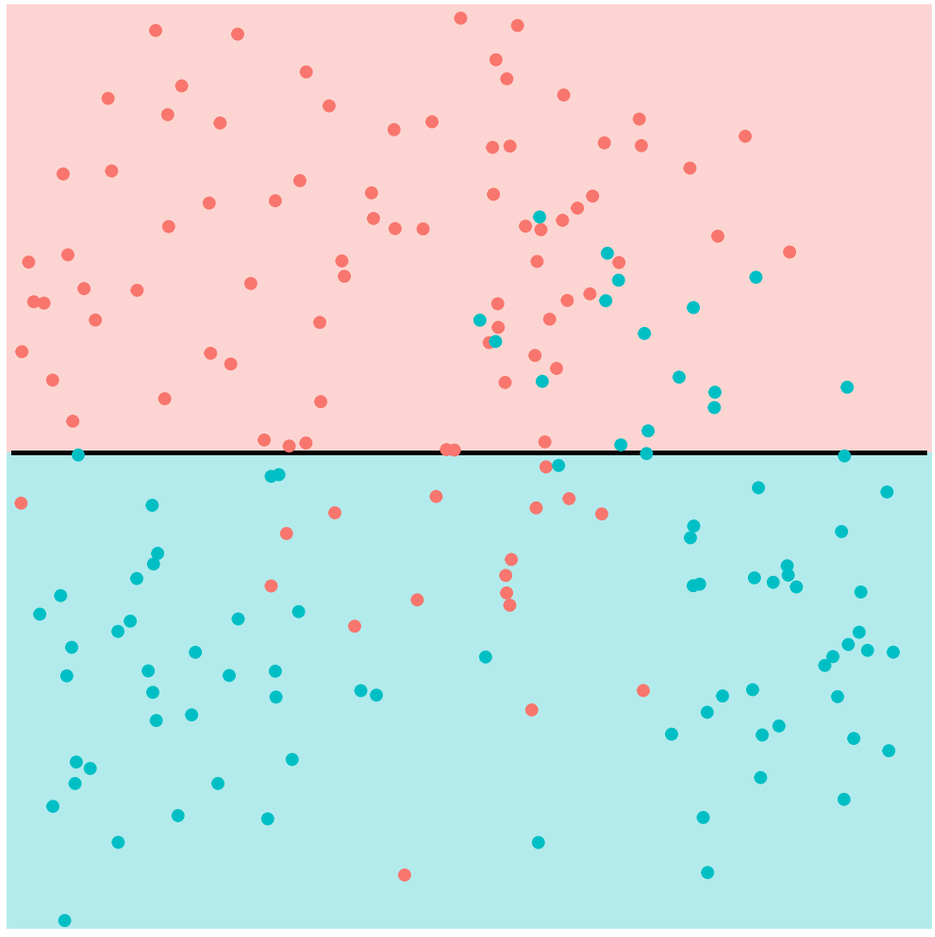
Proportions in leaves are estimates for the **probabilities** to be in each class



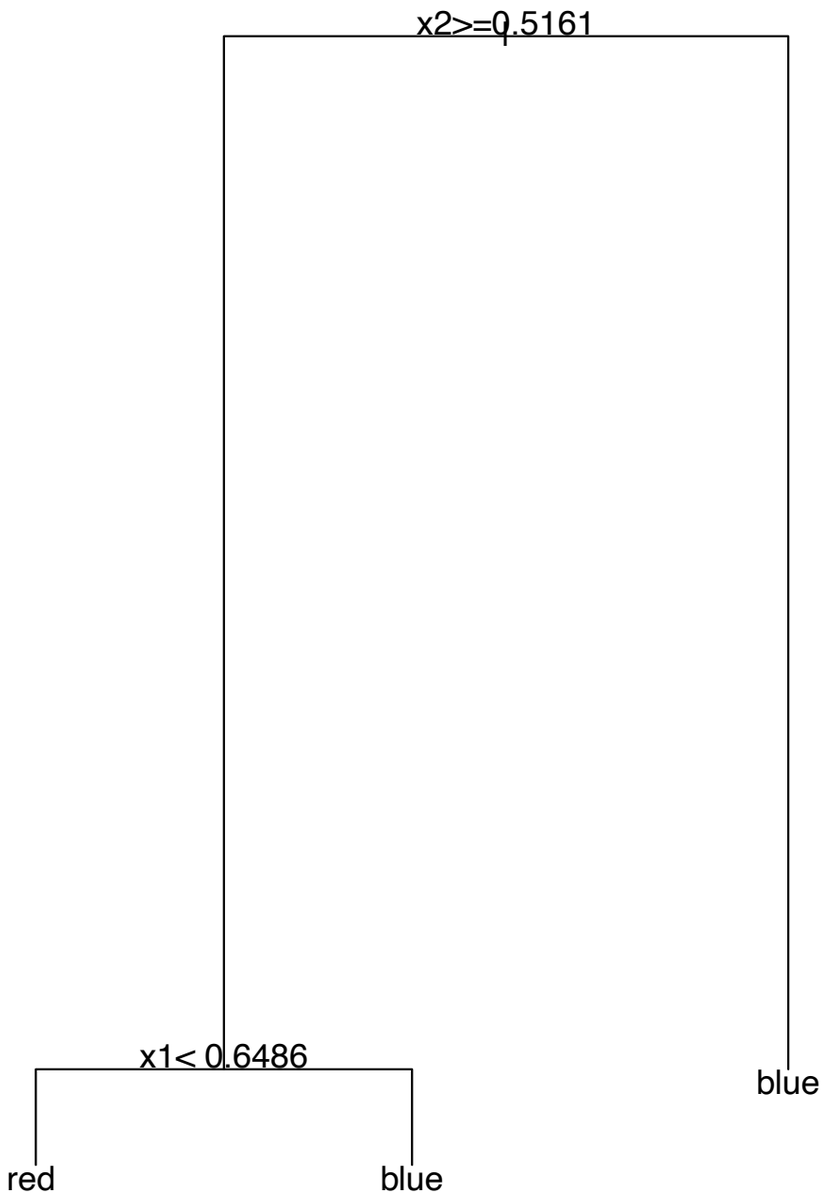


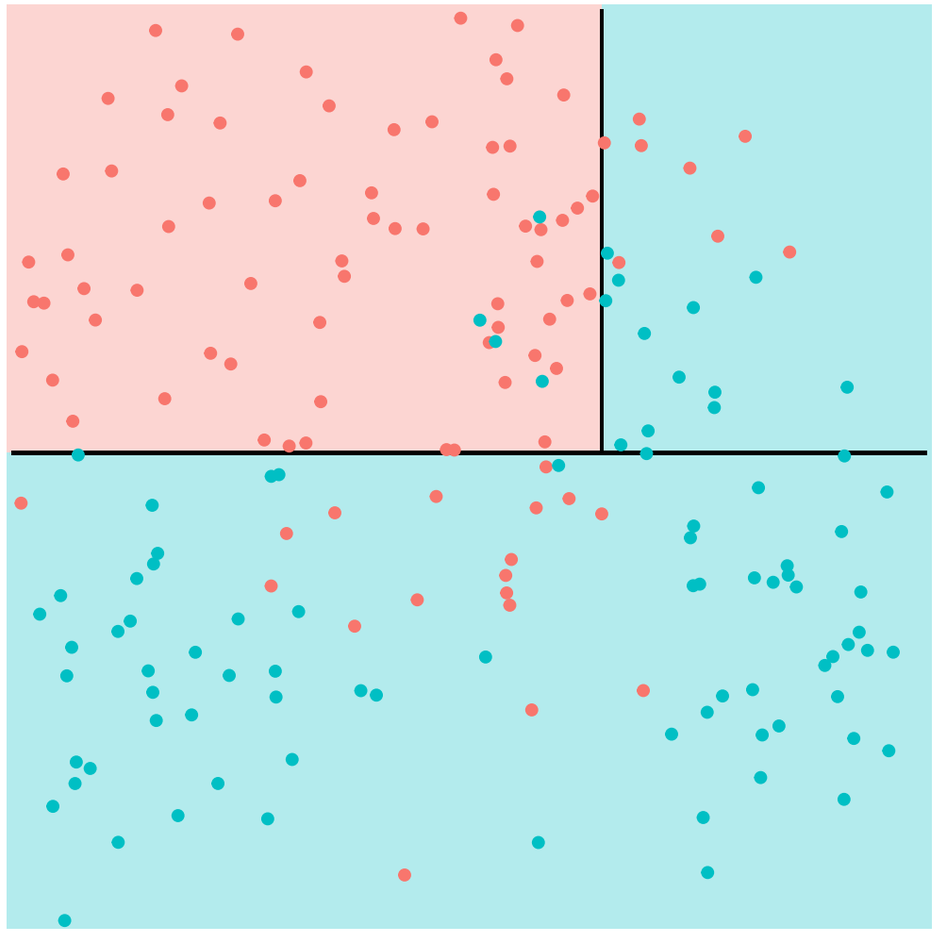
Growing the tree



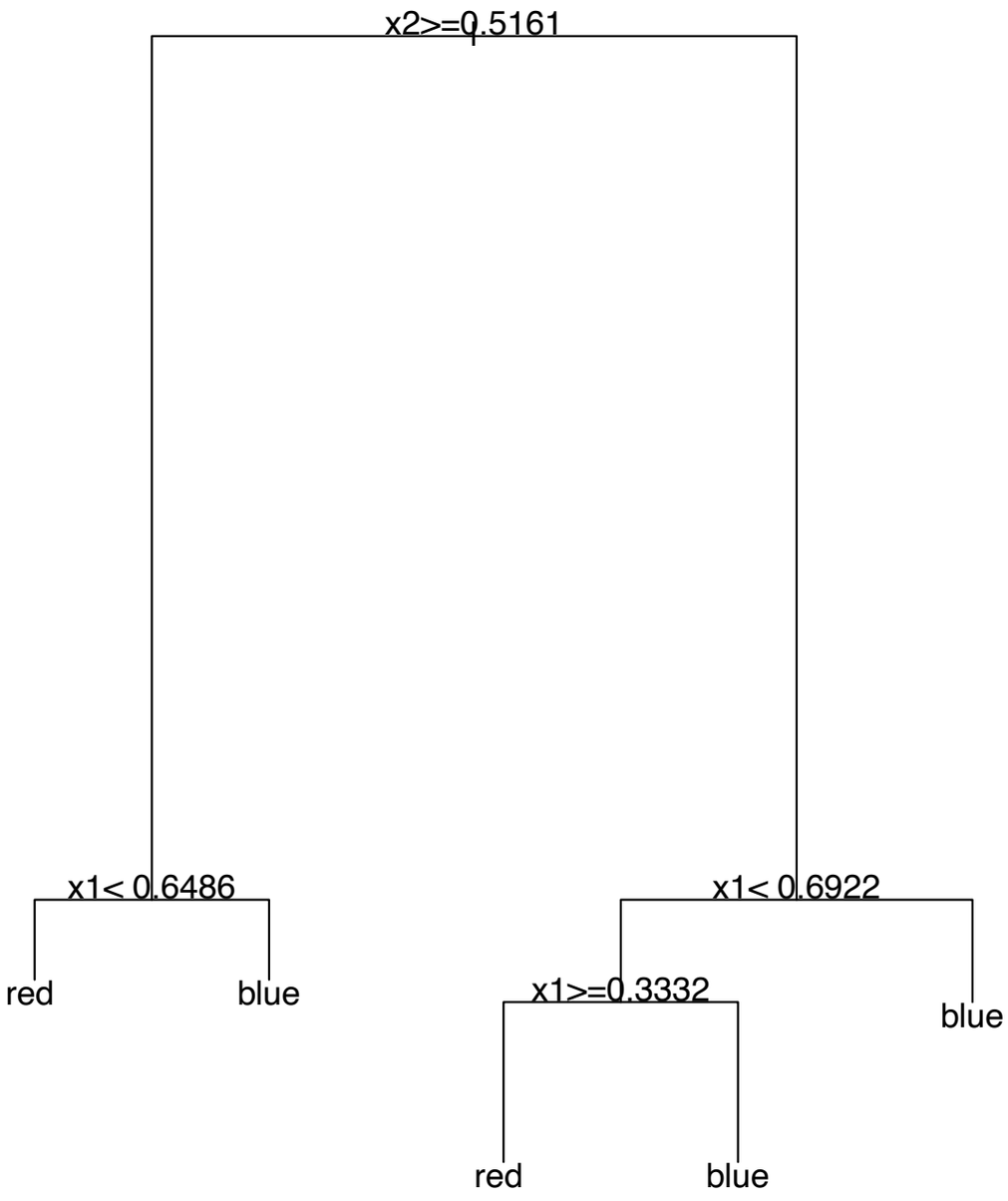


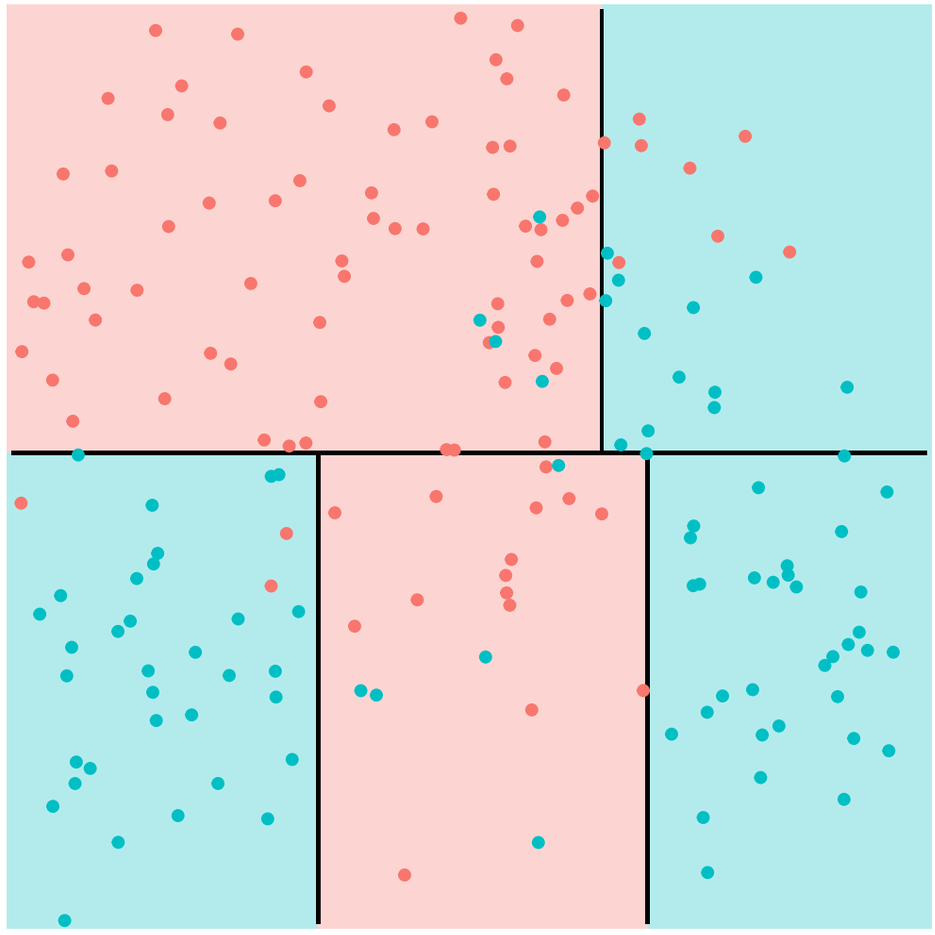
Growing the tree



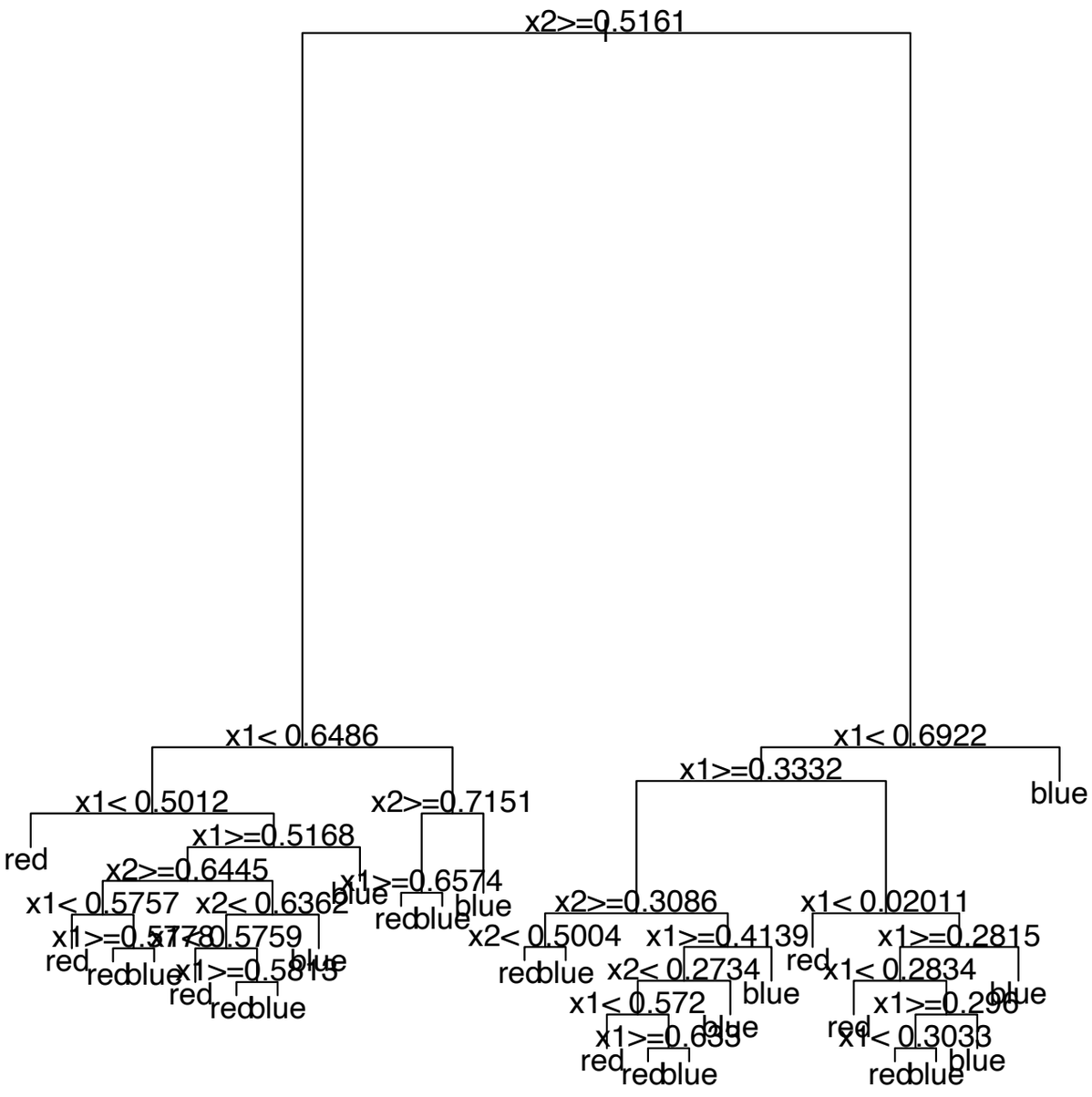


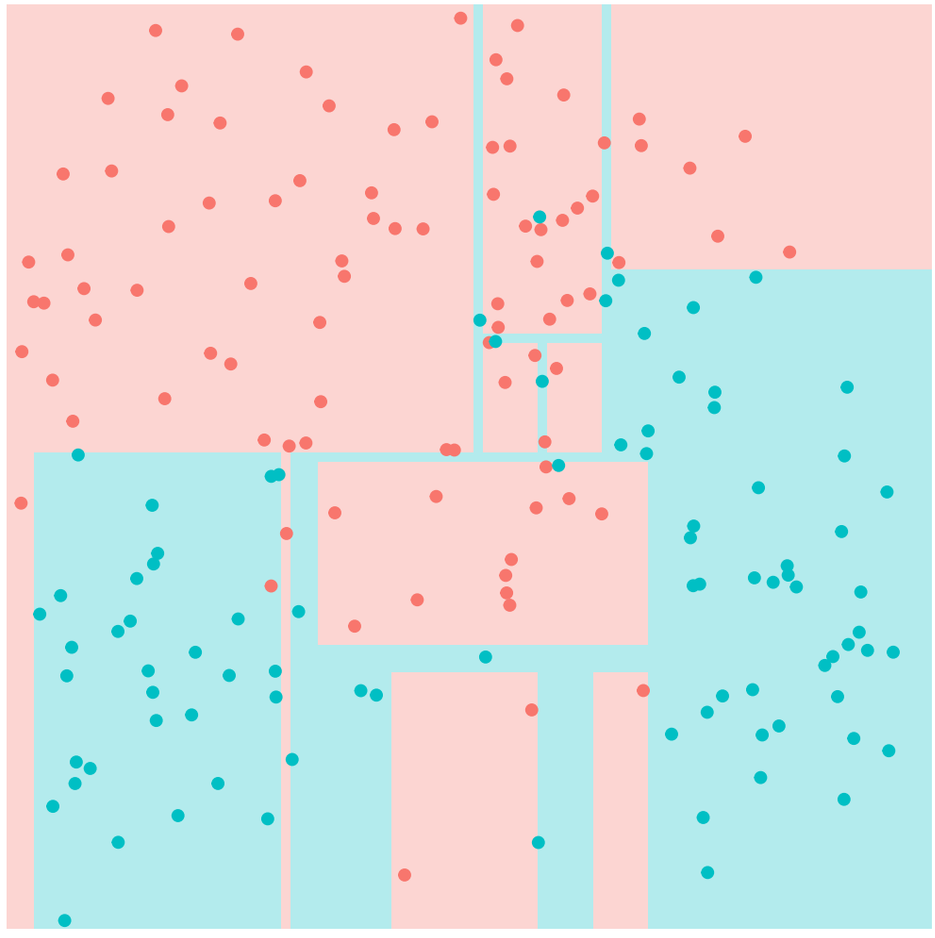
Growing the tree





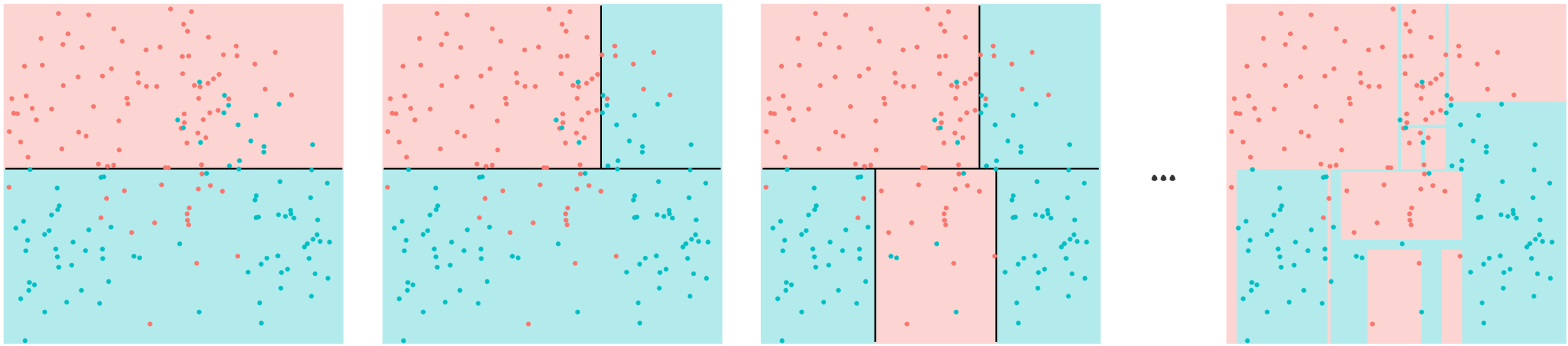
Growing the tree



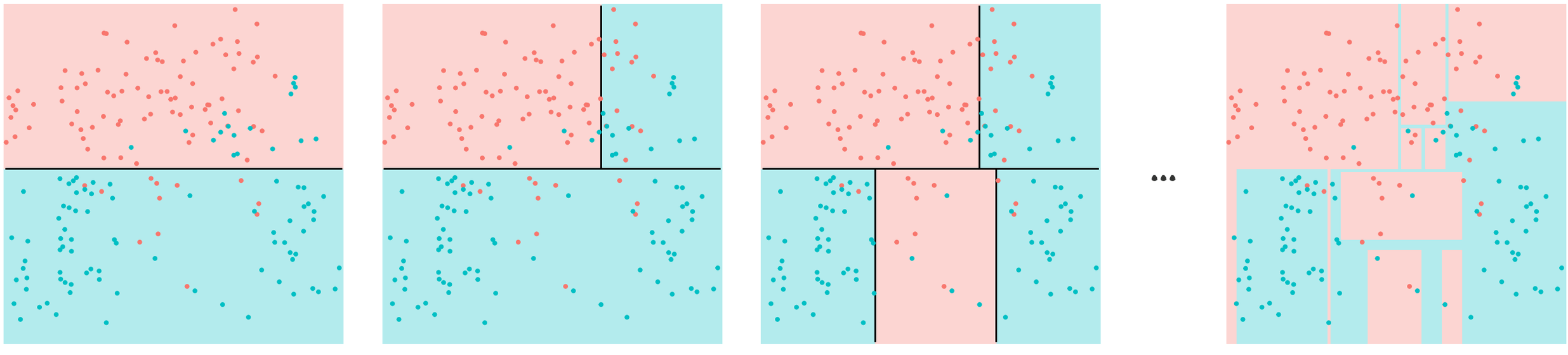


When to stop?

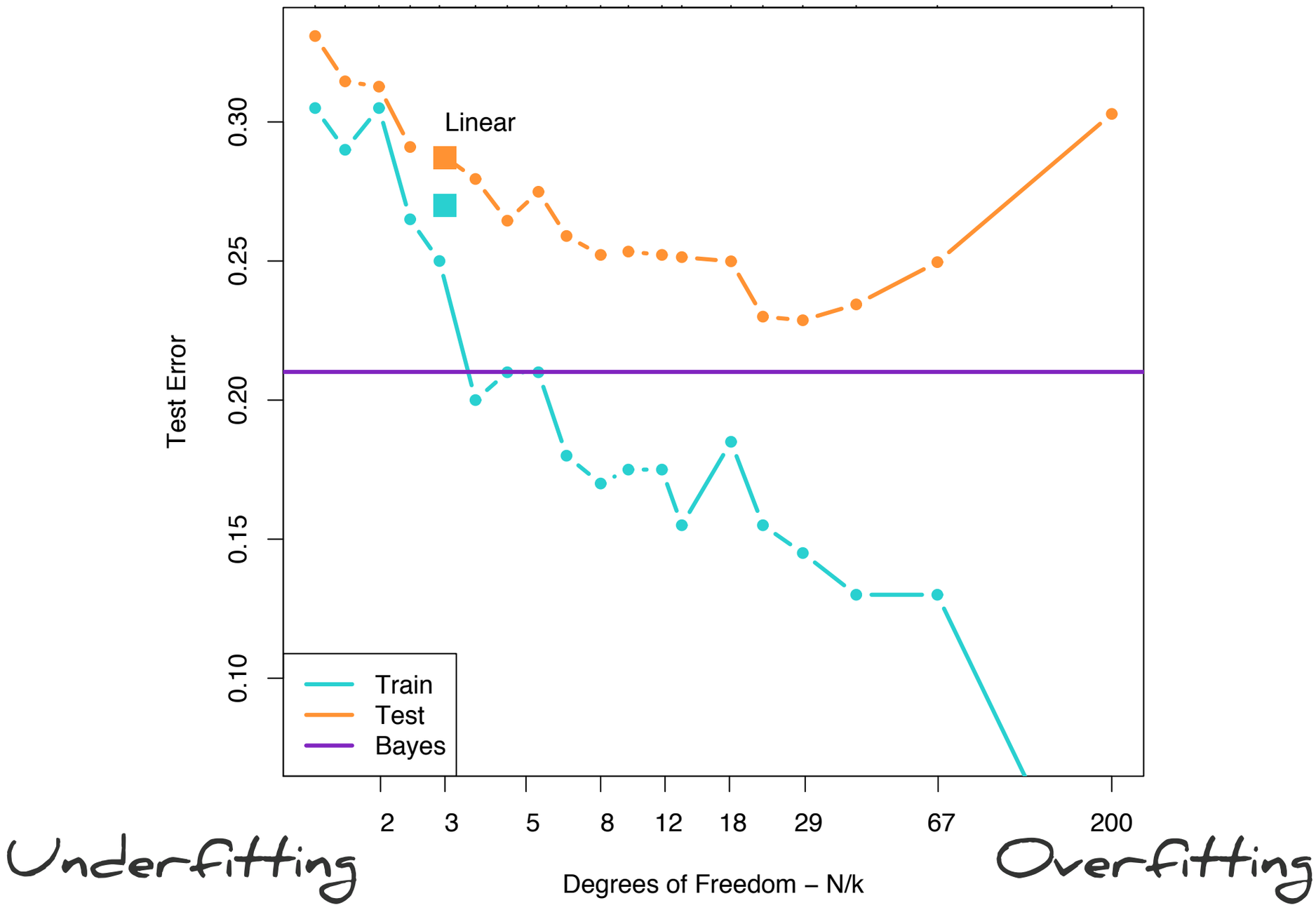
Training data



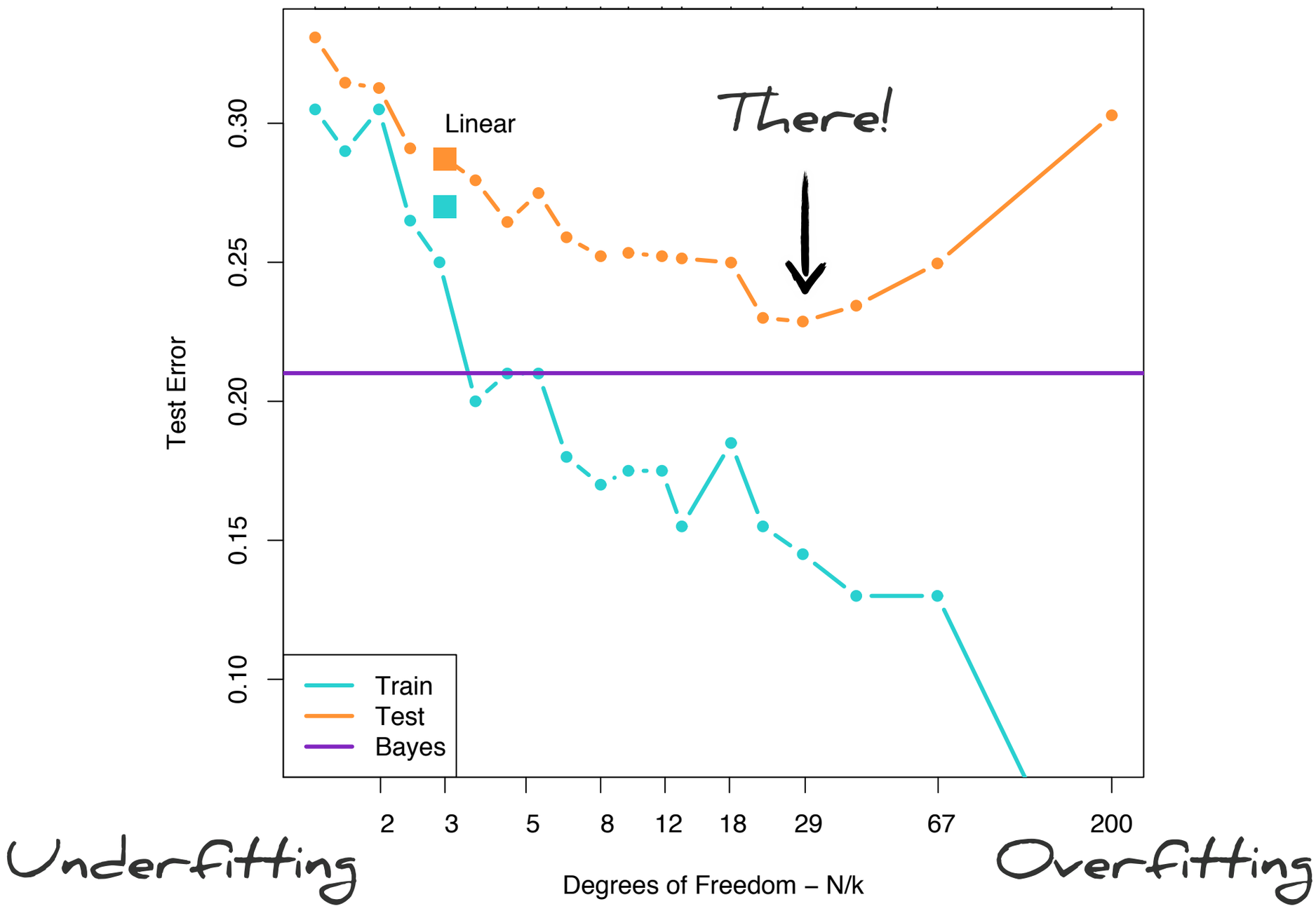
New data



When to stop?


Linear
Test Error
0.10
0.15
0.20
0.25
0.30
Train
Test
Bayes
2
3
5
8
12
18
29
67
200
Degrees of Freedom – N/k
Underfitting
Overfitting

When to stop?


Linear
There!
Test Error
0.30
0.25
0.20
0.15
0.10
Train
Test
Bayes
2
3
5
8
12
18
29
67
200
Underfitting
Overfitting
Degrees of Freedom – N/k

Choosing an hyperparameter

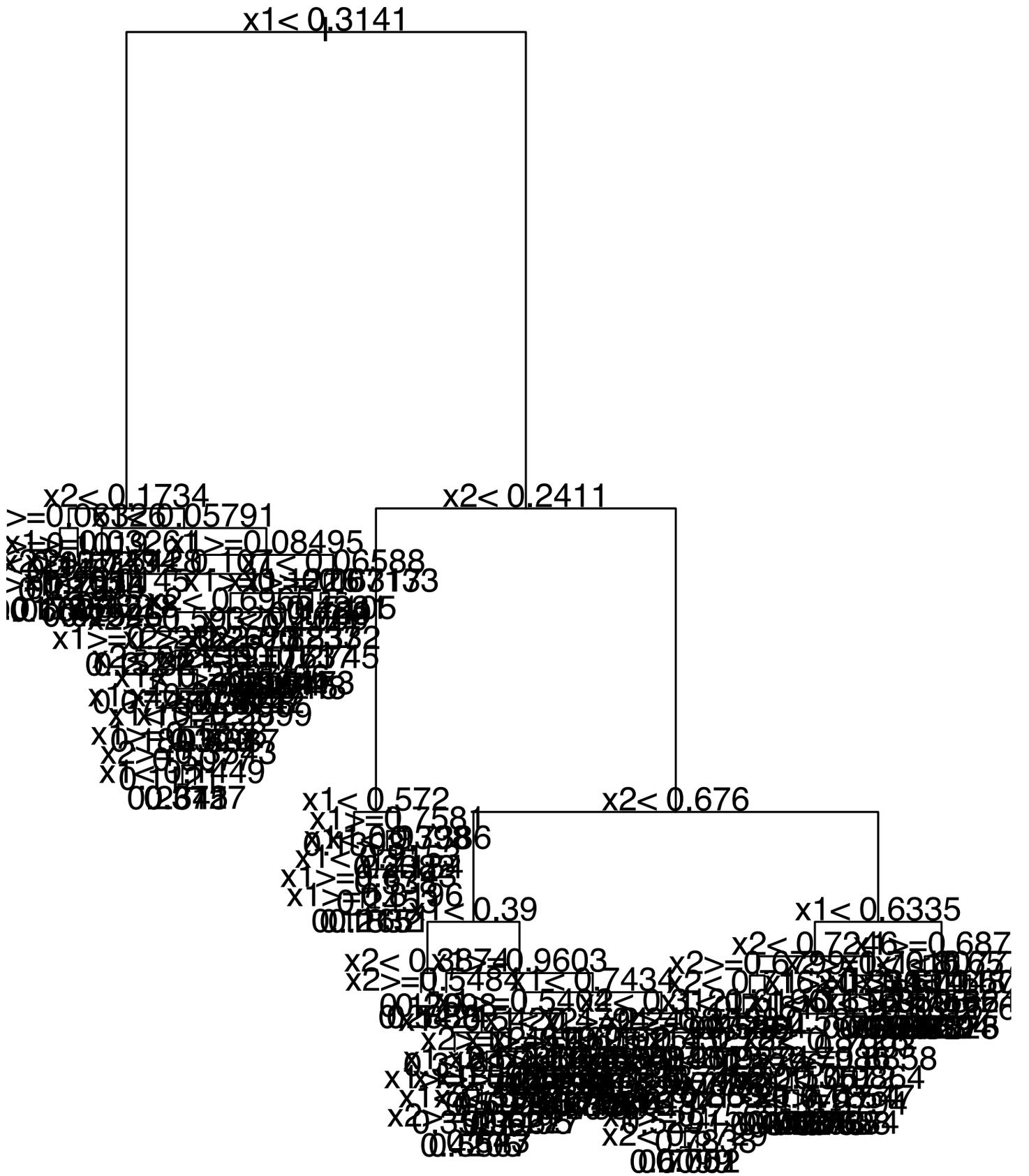
Here: the tree depth

Grow the tree **too far**

Prune back using a **cost-complexity** parameter

The best tree depth is chosen with a **validation set** or by **cross-validation** or through **bootstrapping** or...

NOT using the **test set**!



Choosing an hyperparameter

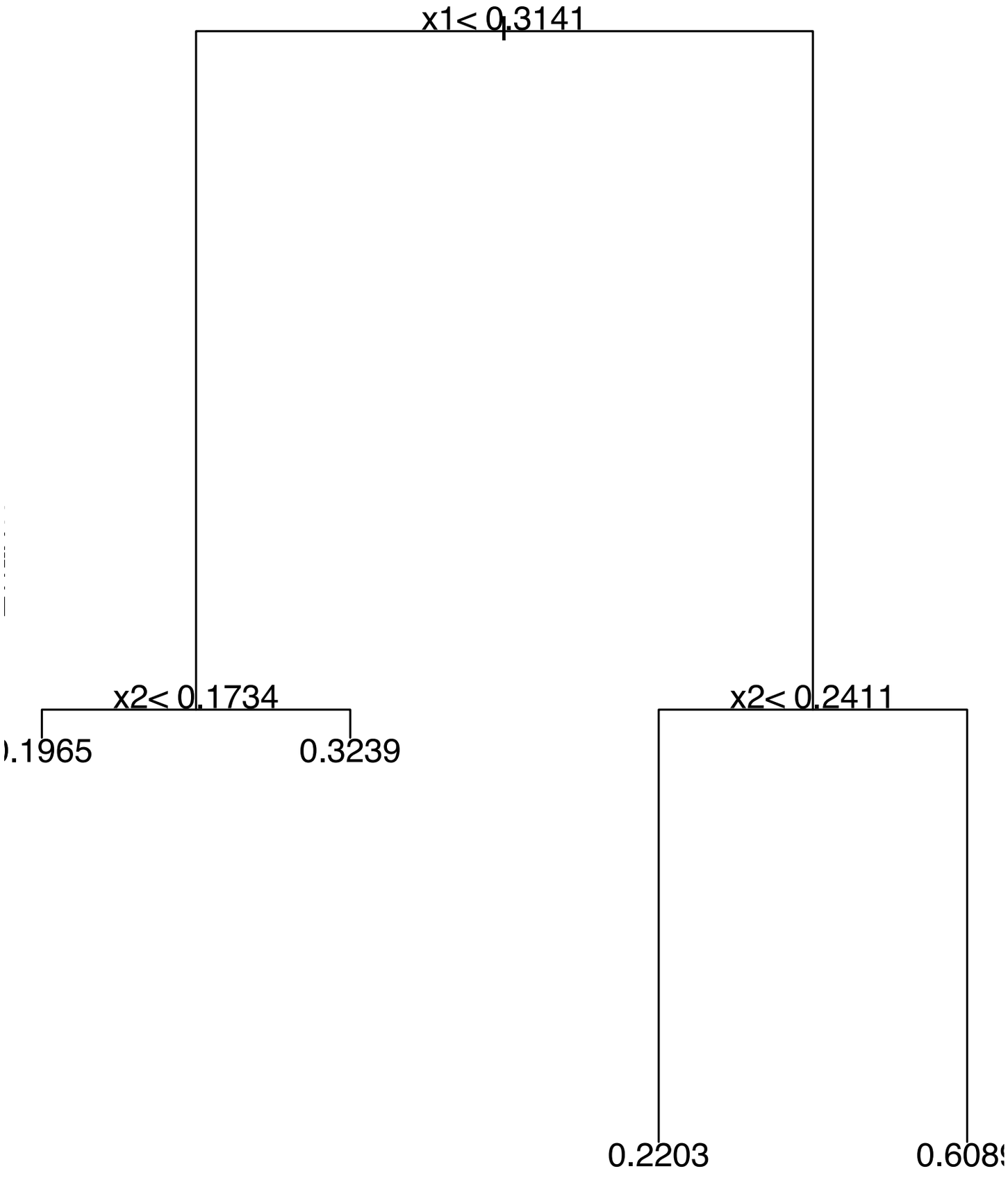
Here: the tree depth

Grow the tree **too far**

Prune back using a **cost-complexity** parameter

The best tree depth is chosen with a **validation set** or by **cross-validation** or through **bootstrapping** or...

NOT using the **test set**!



Pros and Cons of CART

Pros

Easy to **compute**

No assumption on the data, insensitive to monotone **transformations** and robust to **outliers**

Eliminate **irrelevant** variables

Easy to **interpret**

Deals with **interactions** naturally

Cons

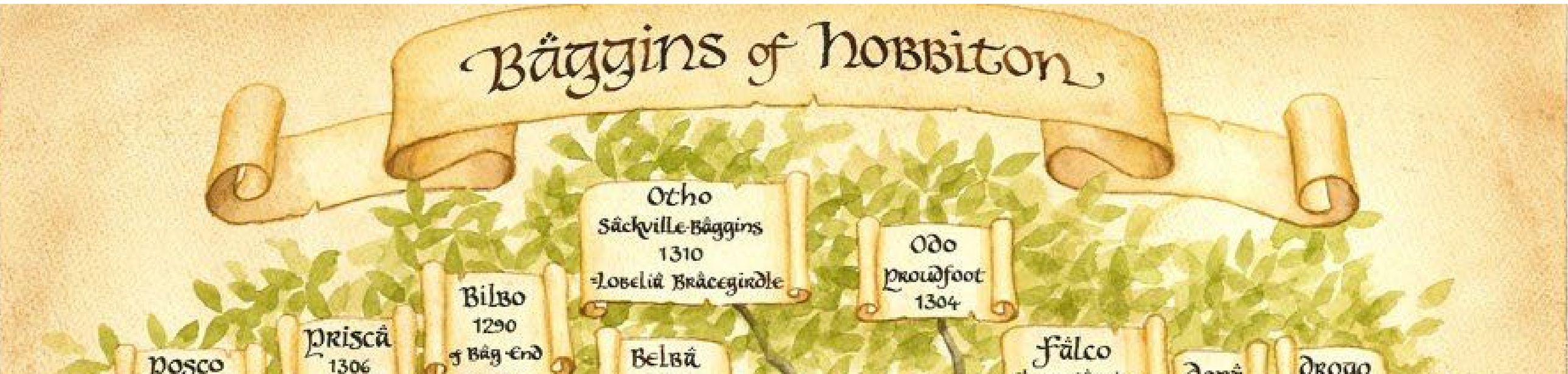
Instable (high variance = a small change in the data can yield a large change in the tree)

Prone to **overfitting** $\Longrightarrow$ lower predictive power

Lack of **smoothness** (which is particularly problematic for regression)

Bagging

Decision by committee



Bootstrapping

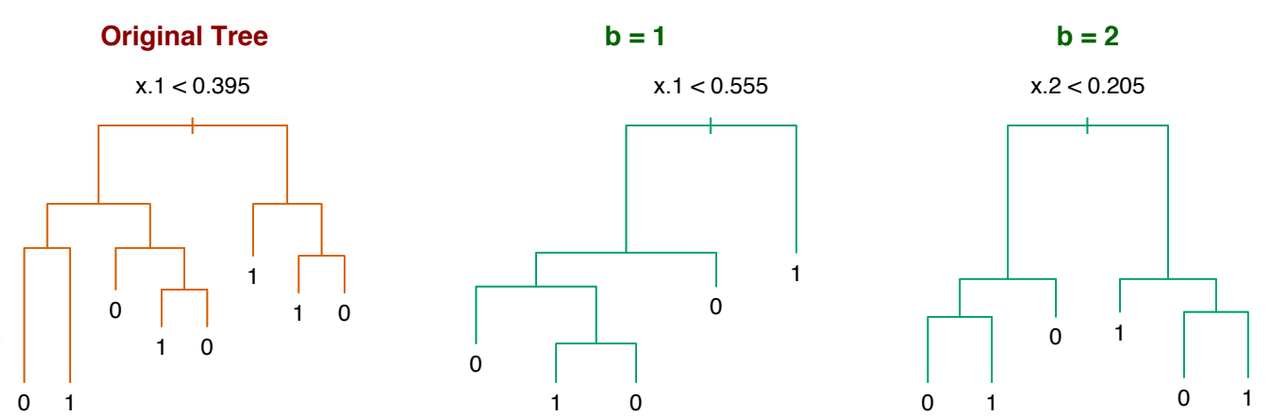
Bootstrap **agg**regat**ing** = **bagging**

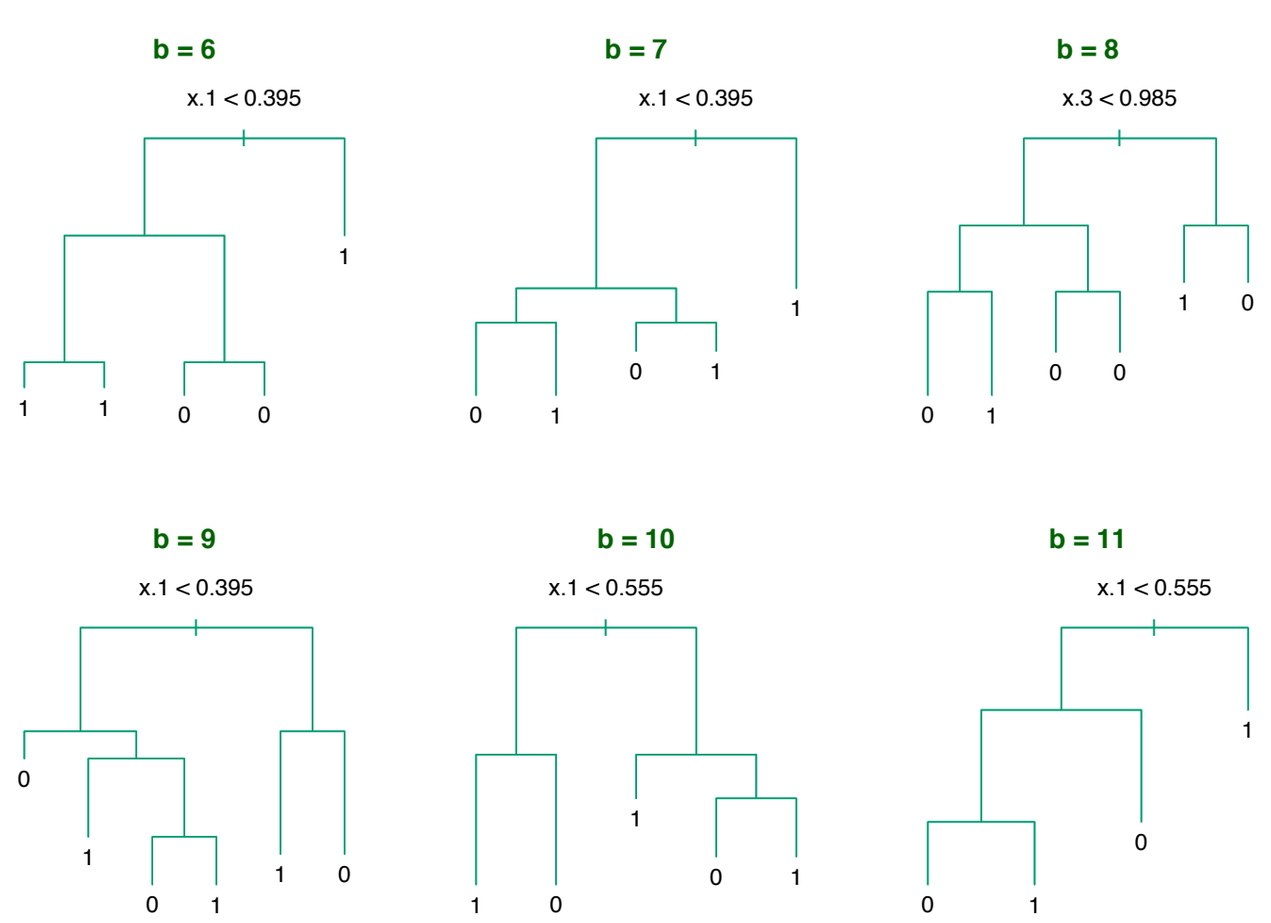
Algorithm:

1. draw B **bootstrap** samples
2. compute B trees
3. **average** result of all B trees

   for classification = majority vote among B trees or average probabilities

   for regression = average value across B trees



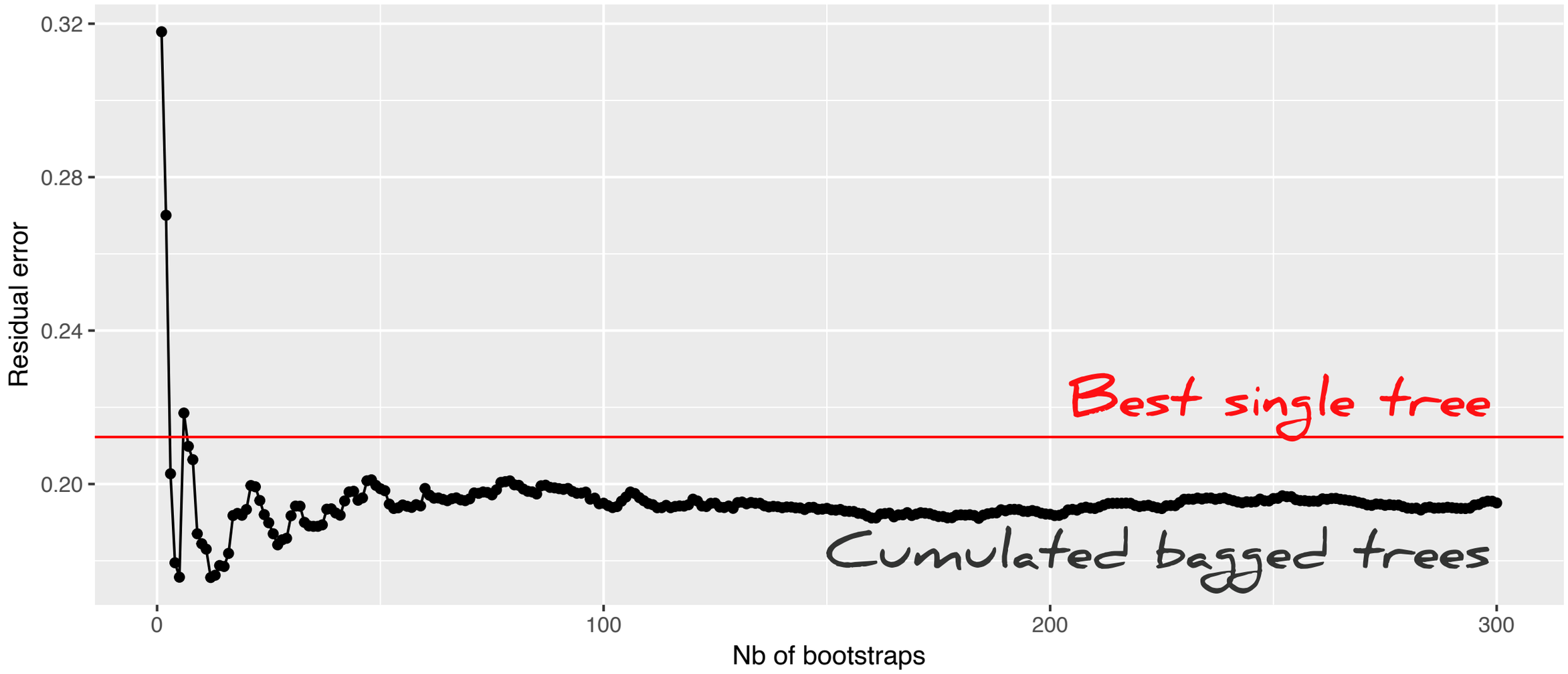


Bagging improves performance

Reduces variance/instability

Particularly adapted for **trees** (high variance, low bias)

Can even **improve prediction** (particularly when smooth = regression)

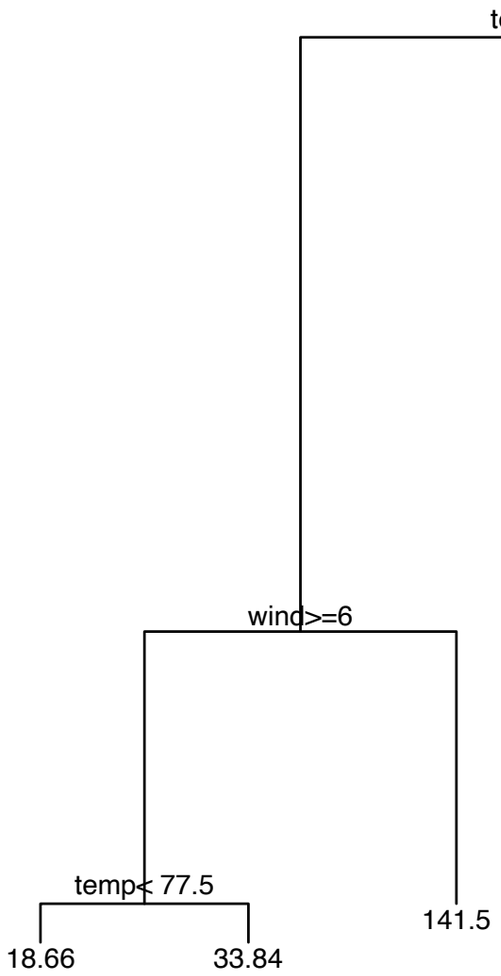


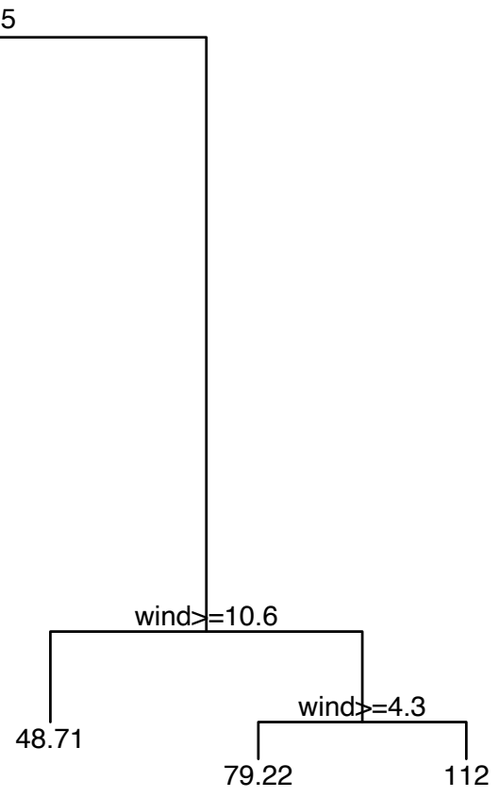
Gradient boosting

Because is sounds cool

Principle of boosting

Replace one deep tree





with **many** shallow ones, applied in sequence

Algorithm for regression

Algorithm 10.3 *Gradient Tree Boosting Algorithm.*

1. Initialize $f_0(x) = \arg\min_\gamma \sum_{i=1}^{N} L(y_i, \gamma)$.
2. For $m = 1$ to M:
   (a) For $i = 1, 2, \ldots, N$ compute

$$r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}.$$

   (b) Fit a regression tree to the targets r_{im} giving terminal regions $R_{jm},\ j = 1, 2, \ldots, J_m$.
   (c) For $j = 1, 2, \ldots, J_m$ compute

$$\gamma_{jm} = \arg\min_{\gamma} \sum_{x_i \in R_{jm}} L\left(y_i, f_{m-1}(x_i) + \gamma\right).$$

   (d) Update $f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$.
3. Output $\hat{f}(x) = f_M(x)$.

Translation 😅

Initialise with a constant that minimises the loss

Compute the "residuals"

Fit *j* trees to those residuals

Pick the tree that decreases the loss the most

Update the previous predicted values and residuals

Repeat steps in 2 M times

Compute the final value as the last update of the values

Algorithm 10.3 *Gradient Tree Boosting Algorithm.*

1. Initialize $f_0(x) = \arg\min_\gamma \sum_{i=1}^{N} L(y_i, \gamma)$.
2. For $m = 1$ to M:
   (a) For $i = 1, 2, \ldots, N$ compute
   $$r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}.$$
   (b) Fit a regression tree to the targets r_{im} giving terminal regions $R_{jm},\ j = 1, 2, \ldots, J_m$.
   (c) For $j = 1, 2, \ldots, J_m$ compute
   $$\gamma_{jm} = \arg\min_\gamma \sum_{x_i \in R_{jm}} L\left(y_i, f_{m-1}(x_i) + \gamma\right).$$
   (d) Update $f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$.
3. Output $\hat{f}(x) = f_M(x)$.

NB: notations different from rest of slides

(One, old) algorithm for classification

Algorithm 10.1 *AdaBoost.M1.*

1. Initialize the observation weights $w_i = 1/N,\ i = 1, 2, \ldots, N$.
2. For $m = 1$ to M:
   (a) Fit a classifier $G_m(x)$ to the training data using weights w_i.
   (b) Compute
   $$\text{err}_m = \frac{\sum_{i=1}^{N} w_i I(y_i \neq G_m(x_i))}{\sum_{i=1}^{N} w_i}.$$
   (c) Compute $\alpha_m = \log((1 - \text{err}_m)/\text{err}_m)$.
   (d) Set $w_i \leftarrow w_i \cdot \exp[\alpha_m \cdot I(y_i \neq G_m(x_i))],\ i = 1, 2, \ldots, N$.
3. Output $G(x) = \text{sign}\left[\sum_{m=1}^{M} \alpha_m G_m(x)\right]$.

Translation

Start with uniform weights

Fit a classifier, with weighted observations

Compute the (weighted) error of this classifier

Increase the weights for incorrectly classified observations

Repeat steps in 2 a large number of times

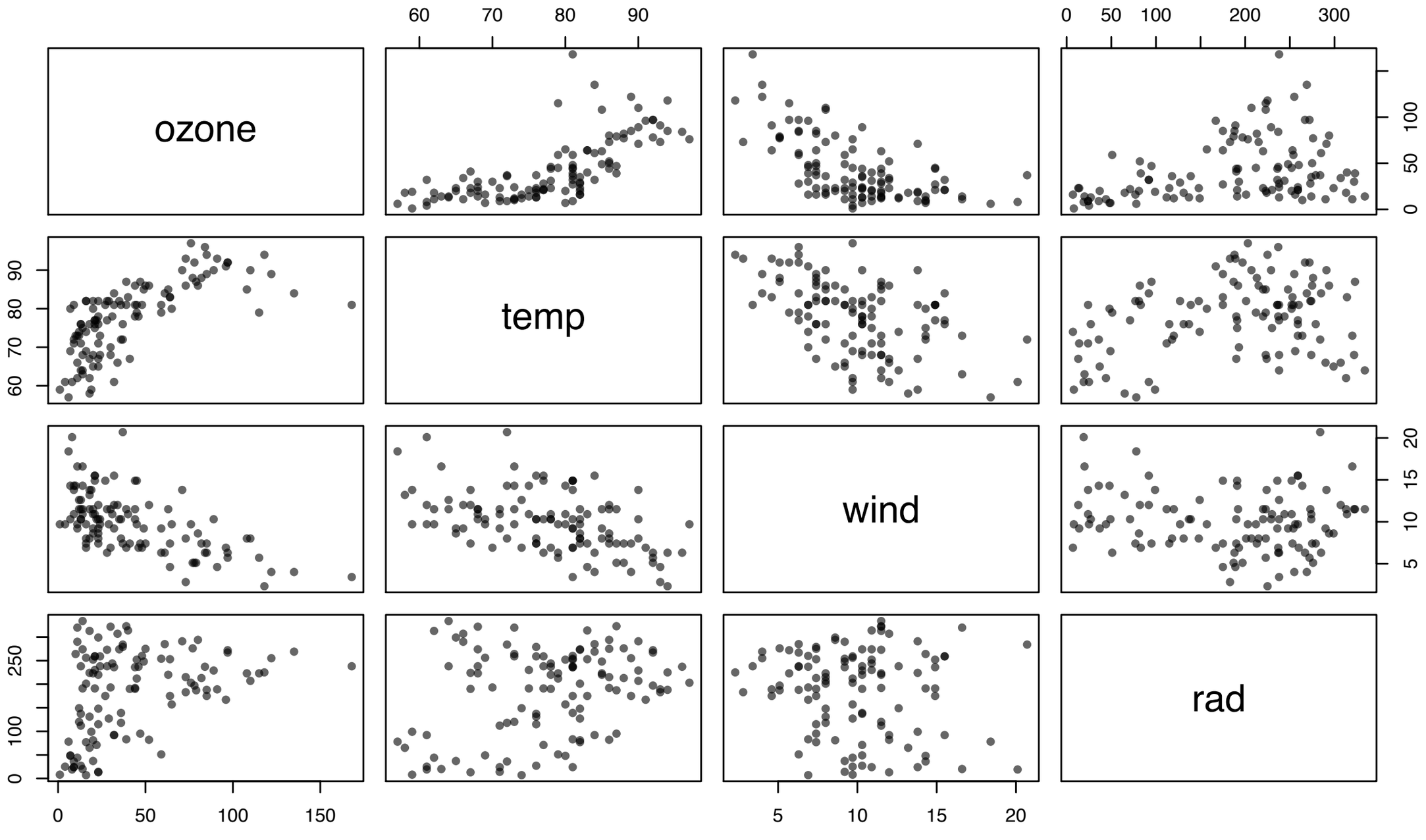
Predict final group as the weighted compound prediction of all trees

Algorithm 10.1 *AdaBoost.M1.*

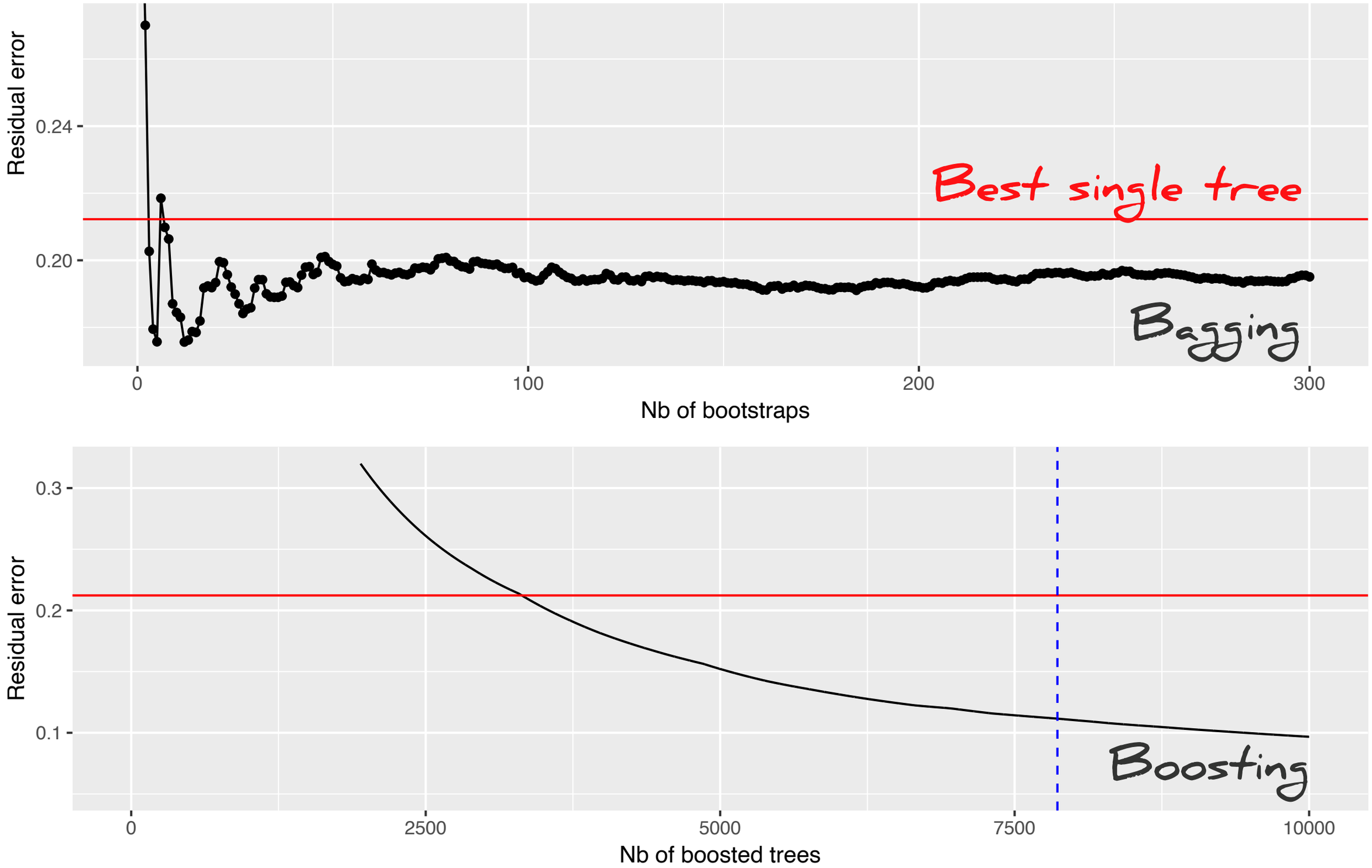
1. Initialize the observation weights $w_i = 1/N,\ i = 1, 2, \ldots, N$.
2. For $m = 1$ to M:
   (a) Fit a classifier $G_m(x)$ to the training data using weights w_i.
   (b) Compute
   $$\text{err}_m = \frac{\sum_{i=1}^{N} w_i I(y_i \neq G_m(x_i))}{\sum_{i=1}^{N} w_i}.$$
   (c) Compute $\alpha_m = \log((1 - \text{err}_m)/\text{err}_m)$.
   (d) Set $w_i \leftarrow w_i \cdot \exp[\alpha_m \cdot I(y_i \neq G_m(x_i))],\ i = 1, 2, \ldots, N$.
3. Output $G(x) = \text{sign}\left[\sum_{m=1}^{M} \alpha_m G_m(x)\right]$.

NB: notations different from rest of slides

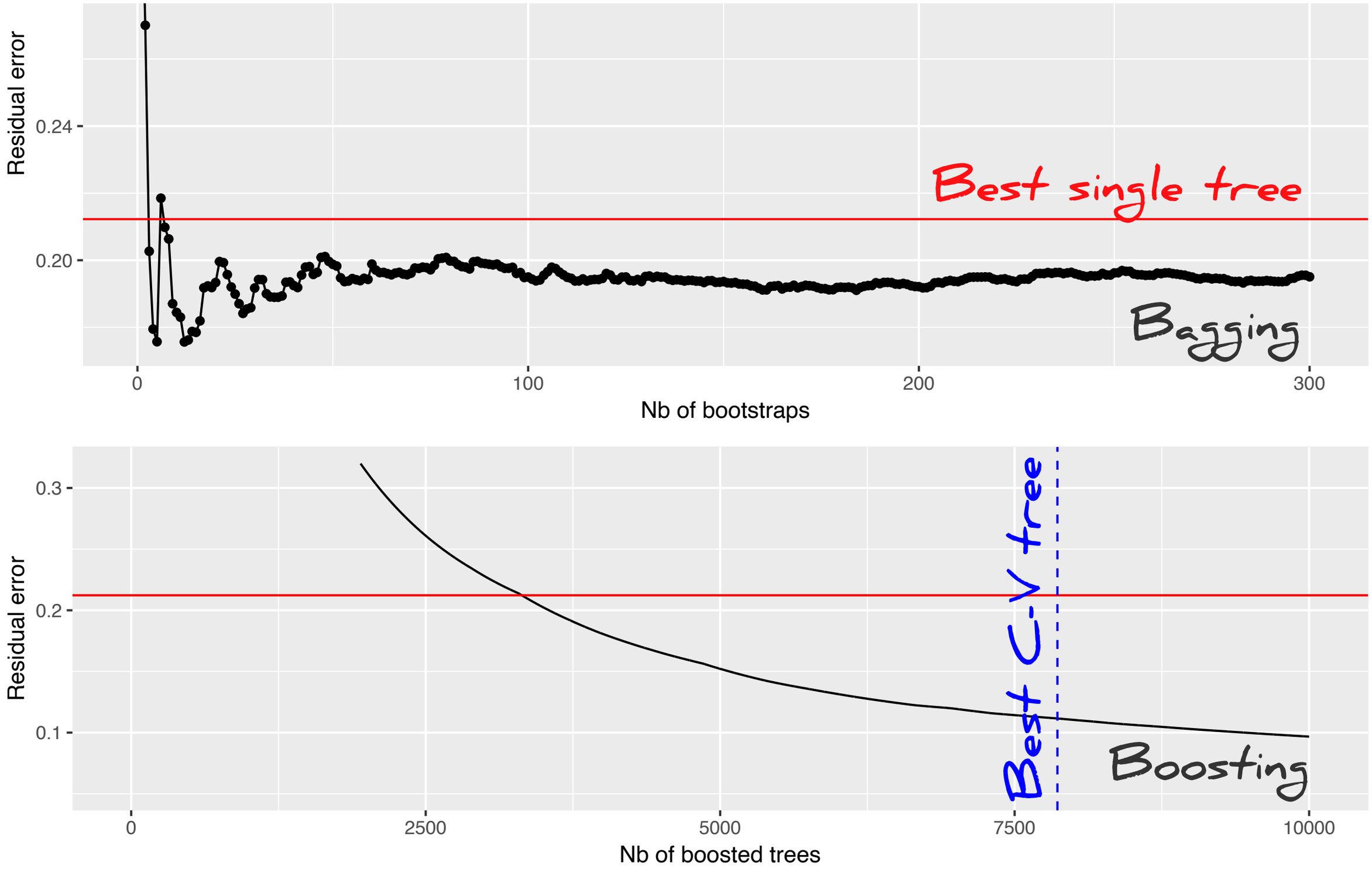
Example: Ozone concentration in atmosphere



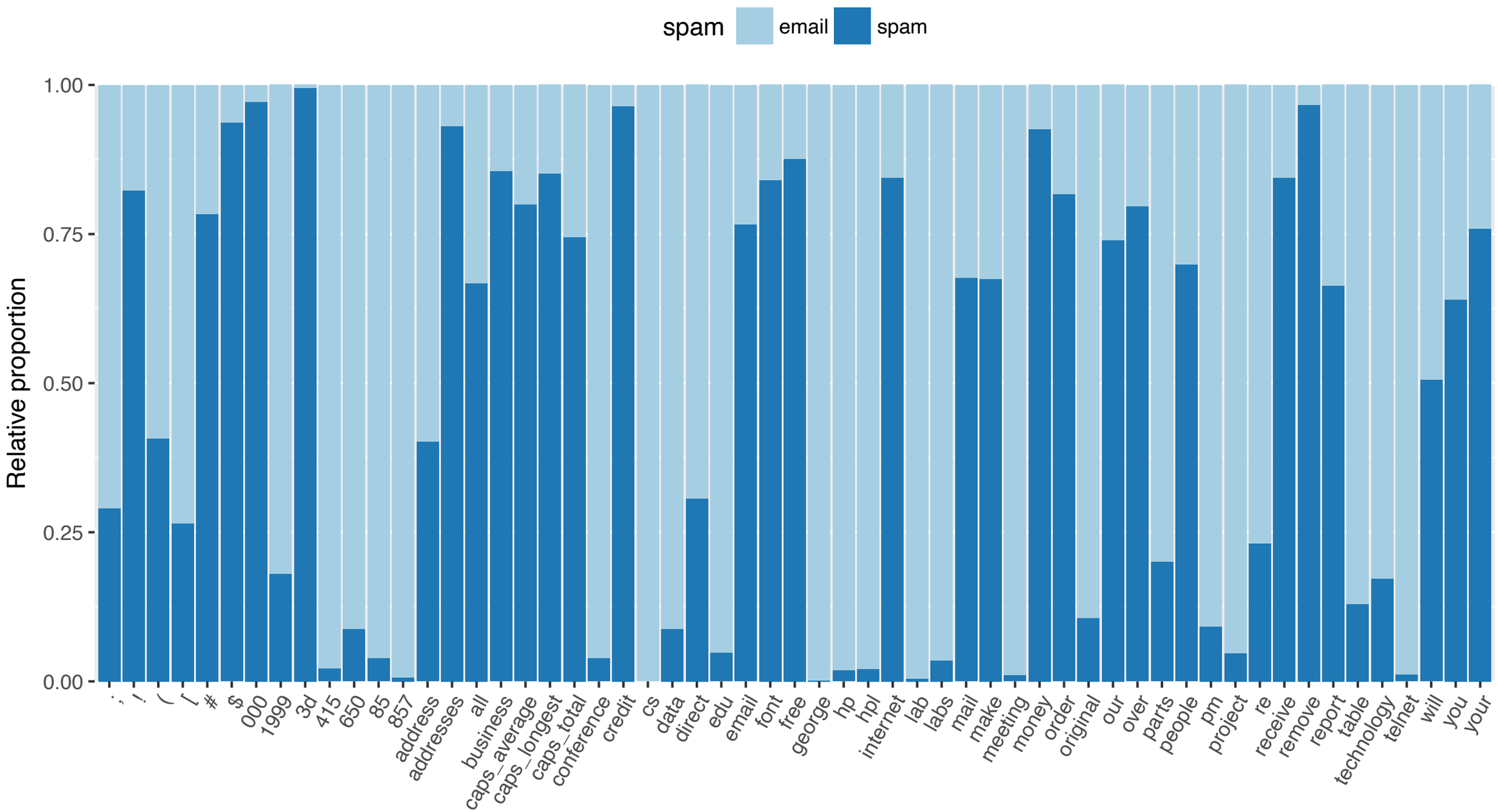
Performance comparison



Performance comparison



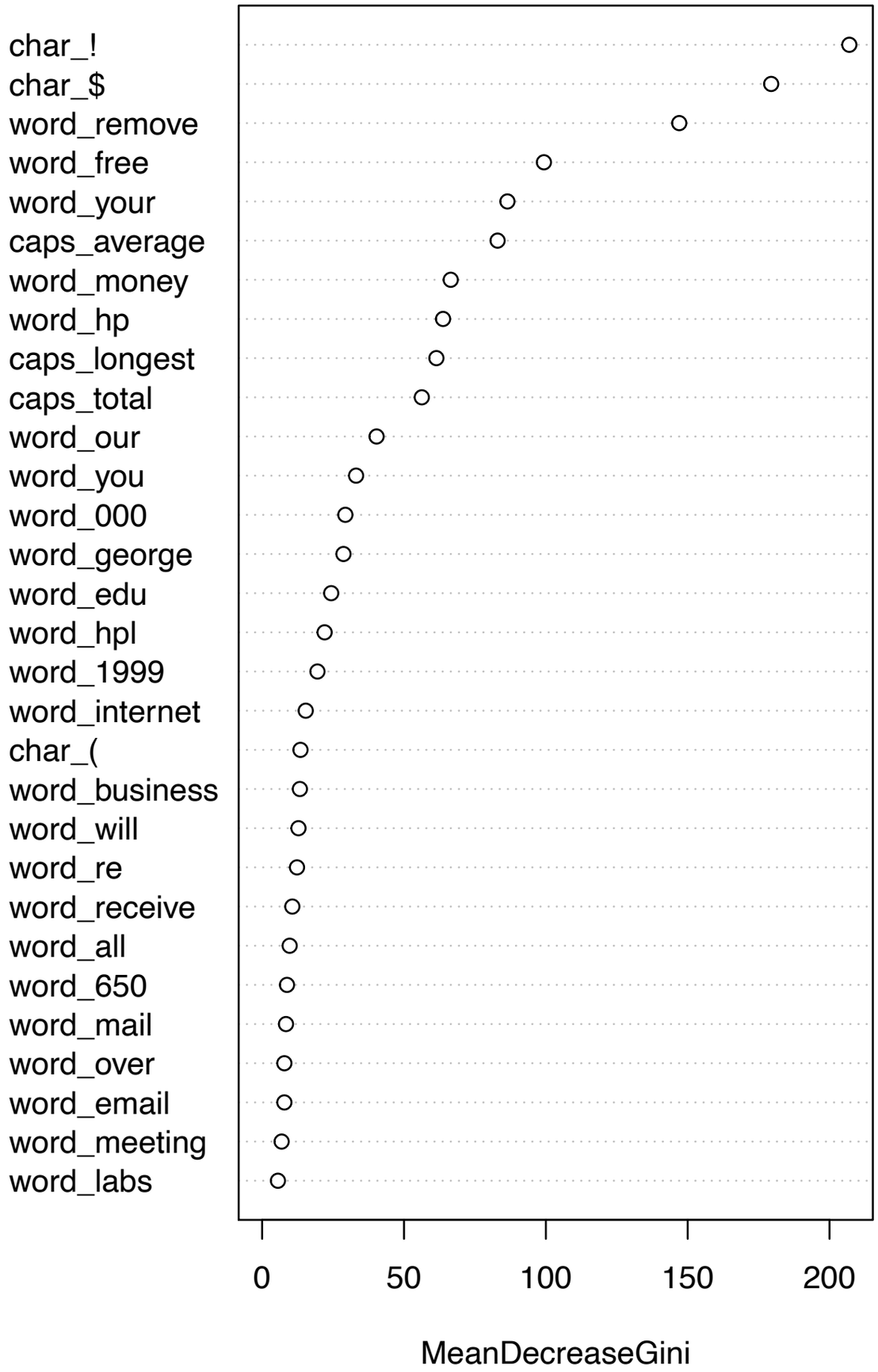
Example: spam detection



Other outputs beyond prediction

Estimate **relative importance** of variables (by counting how many times and with which weight they occur in the trees)

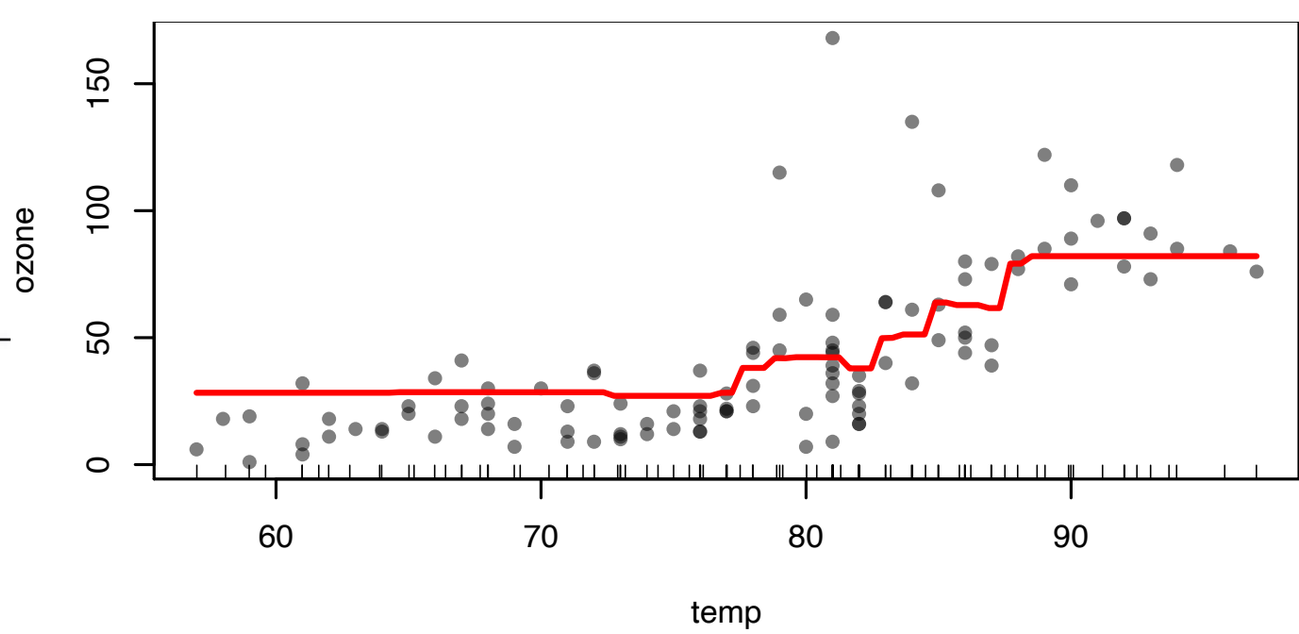
Produce **partial dependence plots** as the mean of the predictions on the training set with the target variable set to a given value

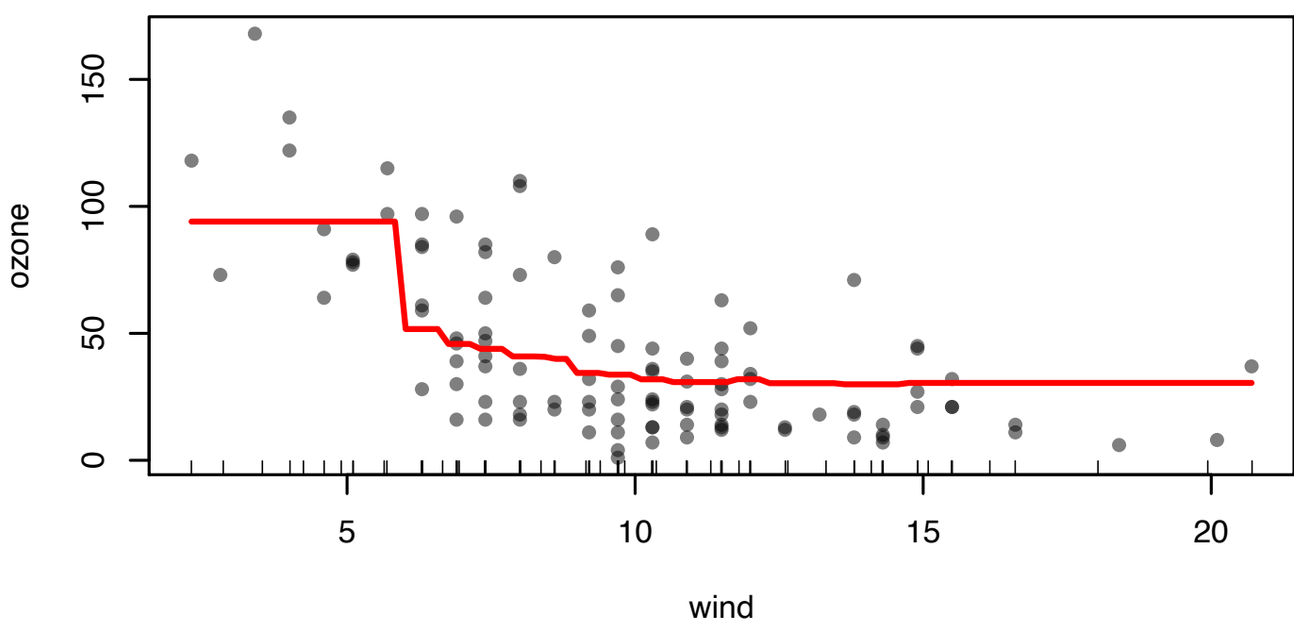


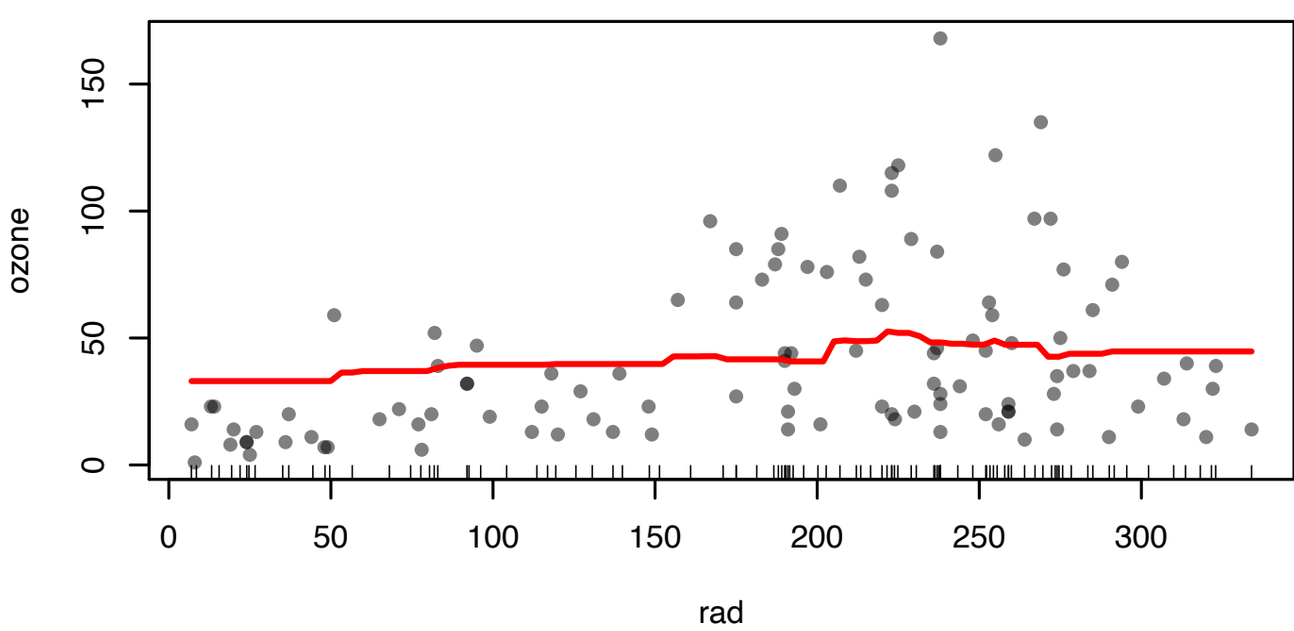
Other outputs beyond prediction

Estimate **relative importance** of variables (by counting how many times and with which weight they occur in the trees)

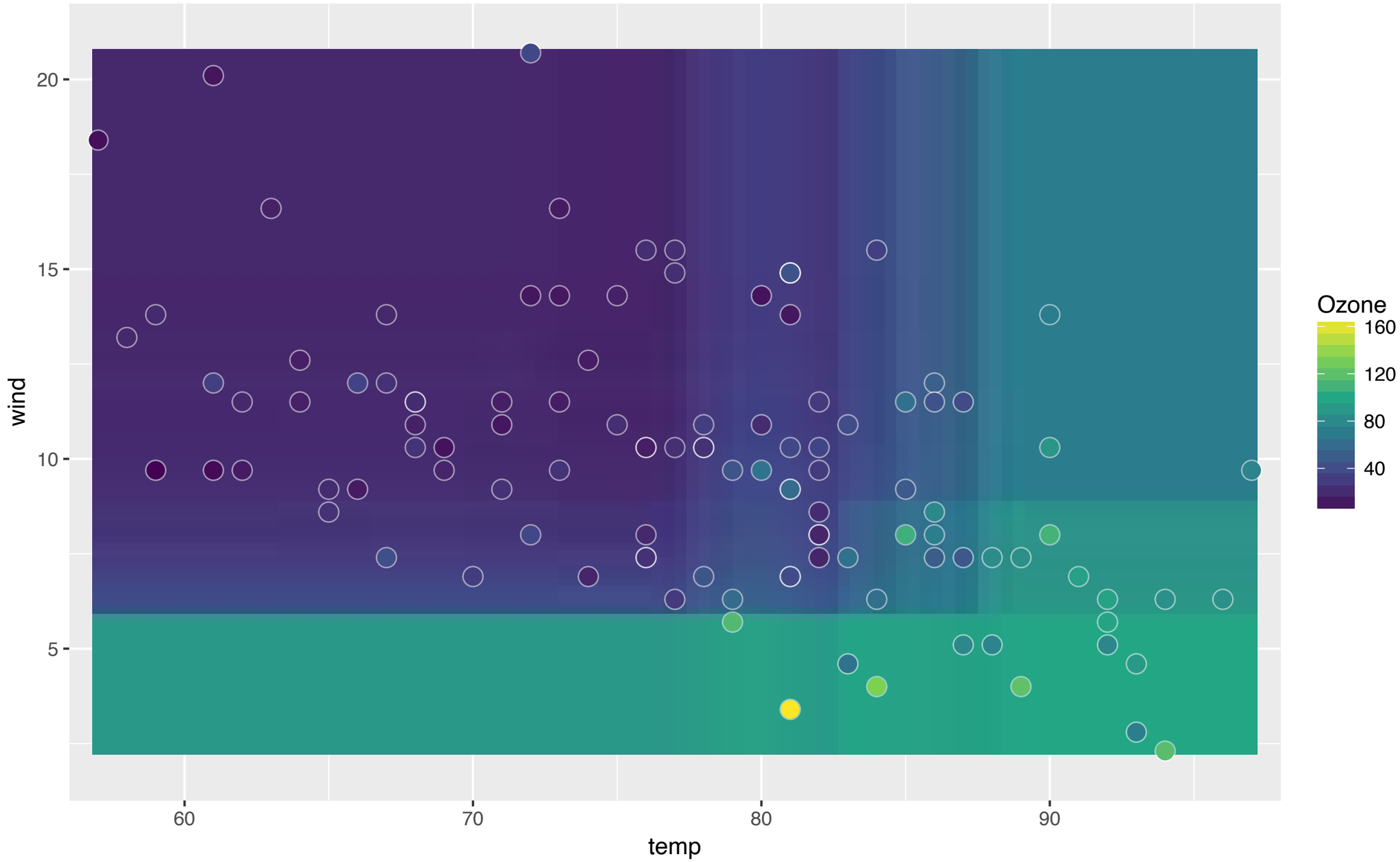
Produce **partial dependence plots** as the mean of the predictions on the training set with the target variable set to a given value







2D partial dependence plot



Choices in a gradient boosted trees procedure

Loss function (`loss`)

Squared error for regression, log-loss for classification

Number of trees (`n_estimators`)

As large as possible, often thousands.

Depth of each tree (`max_depth`, `min_samples_split`, `min_samples_leaf`)

Maximum depth is often low: often 2 to 4, usually < 10. Simpler trees require more trees.

Learning rate (`learning_rate`)

How much of the prediction of each tree is retained. As small as possible, usually $\in[10^{-3},10^{-5}]$. A small learning rate requires more trees.

Proportion of data sampled for each tree (`subsample`)

Smaller values induce better generalization but more bias, commonly ~ 0.5.

Multivariate extension

Response is a vector

Loss function: multivariate R^2, L^2 Norm

Regularisation

(Non constant response)

Nespoli and Medici. arXiv, 2022 http://arxiv.org/abs/2003.03835

Multivariate Boosted Trees and Applications to Forecasting and Control

Lorenzo Nespoli LORENZO.NESPOLI@SUPSI.CH
ISAAC, SUPSI
Mendrisio, CH

Vasco Medici VASCO.MEDICI@SUPSI.CH
ISAAC, SUPSI
Mendrisio, CH

Editor: arXiv

Abstract

Gradient boosted trees are competition-winning, general-purpose, non-parametric regressors, which exploit sequential model fitting and gradient descent to minimize a specific loss function. The most popular implementations are tailored to univariate regression and classification tasks, precluding the possibility of capturing multivariate target cross-correlations and applying structured penalties to the predictions. In this paper, we present a computationally efficient algorithm for fitting multivariate boosted trees. We show that multivariate trees can outperform their univariate counterpart when the predictions are correlated. Furthermore, the algorithm allows to arbitrarily regularize the predictions, so that properties like smoothness, consistency and functional relations can be enforced. We present applications and numerical results related to forecasting and control.

Keywords: boosted trees, multivariate regression, forecasting, control, statistical learning

1. Introduction

We propose the use of multivariate boosted trees (MBTs) to induce arbitrary regularization and consistency properties in the tree output. This can be done both via penalization of the multivariate output or requiring it to be a superposition of basis functions. Inducing regularization in multivariate output is not new, but while this is common for example in neural network architectures (Oreshkin et al., 2019; Belharbi et al., 2018; Bronstein et al., 2017), they are currently not exploited in tree-based algorithms. One exception is the possibility of LightGBM and XGBoost to express monotonicity conditions of the univariate prediction, with respect to a given input (LightGBM, 2020). This is obtained by inhibiting the tree growth if the new leaf causes a non-monotonic split in the selected feature. However, this may produce unnecessarily shallow trees if not enough split candidates are tested, which could be the case if the tree is grown using histogram search, one of the most popular methods for finding candidate splits.

1.1 Related work

In Pande et al. (2017), an MBT tailored to predicting longitudinal data is presented. This kind of data is typically generated in medical studies, sampling the population at different

Gradient boosting for biogeography

Examples from the wild

Krill distribution in the Austral Ocean

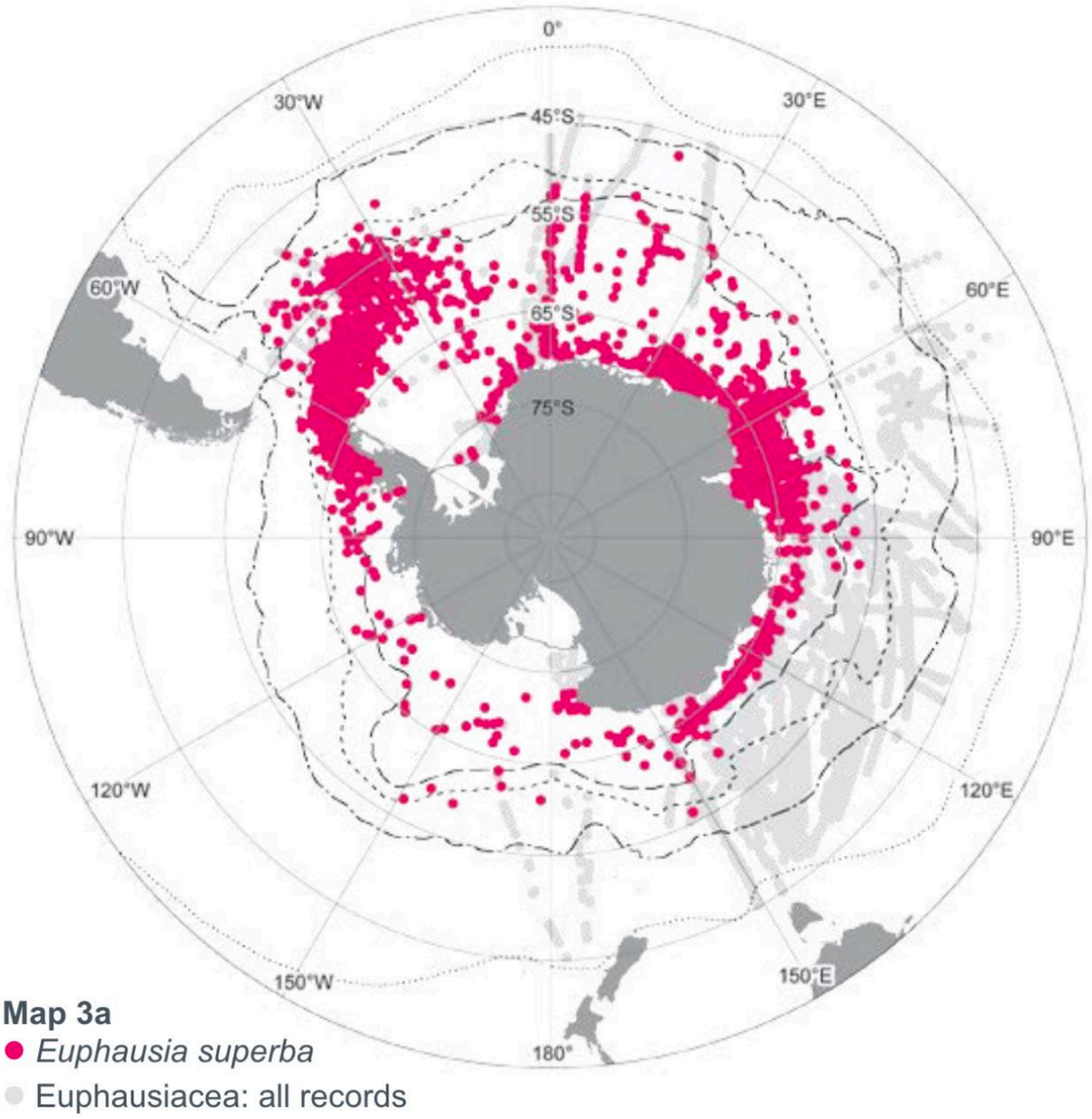
Presence + absence records

Logistic regression

Use ~15 environmental fields as predictors

Projected within the observed environmental space

Irisson et al. in Biogeographic Atlas of the Southern Ocean, 2014



Krill distribution in the Austral Ocean

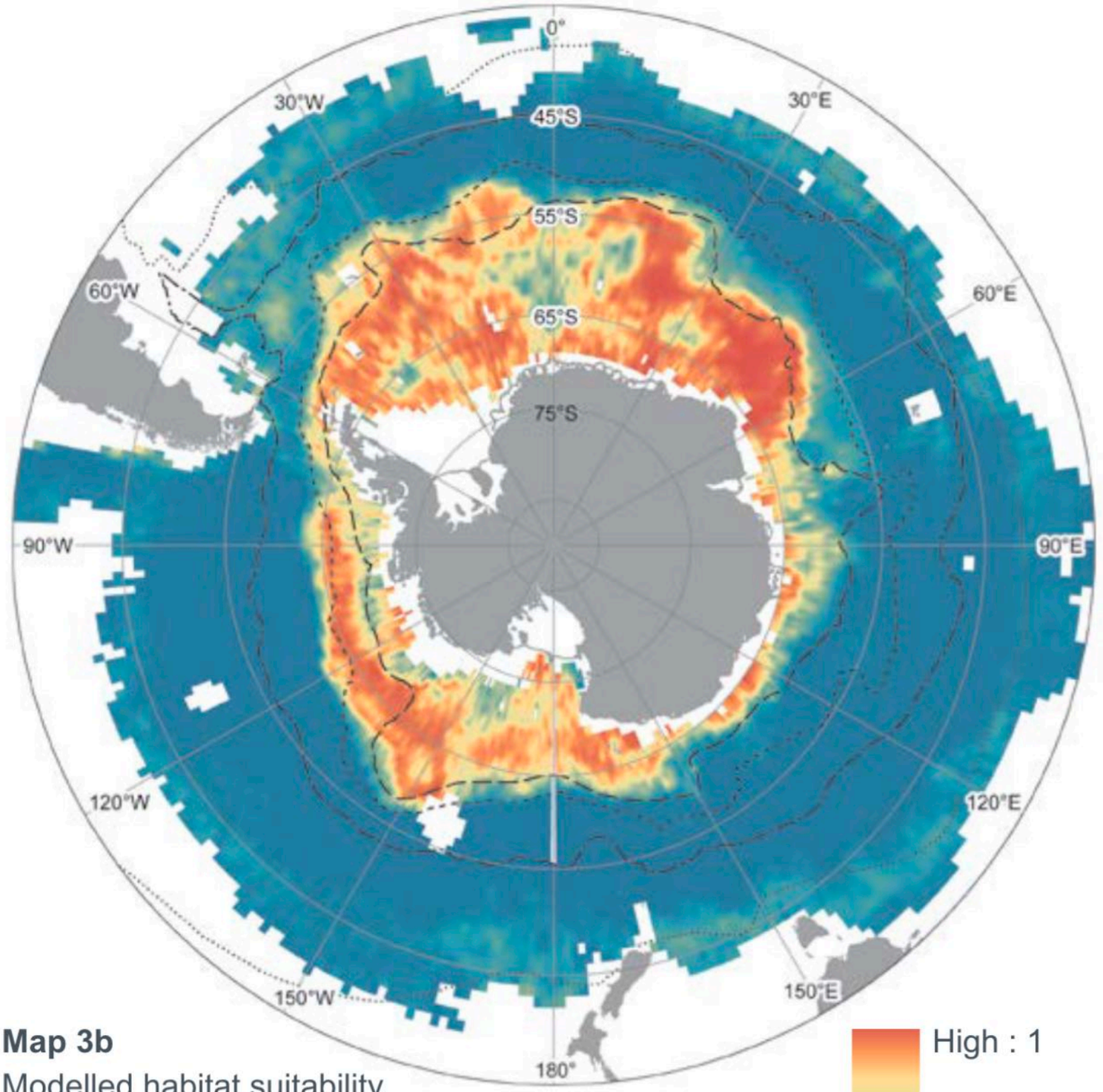
Presence + absence records

Logistic regression

Use ~15 environmental fields as predictors

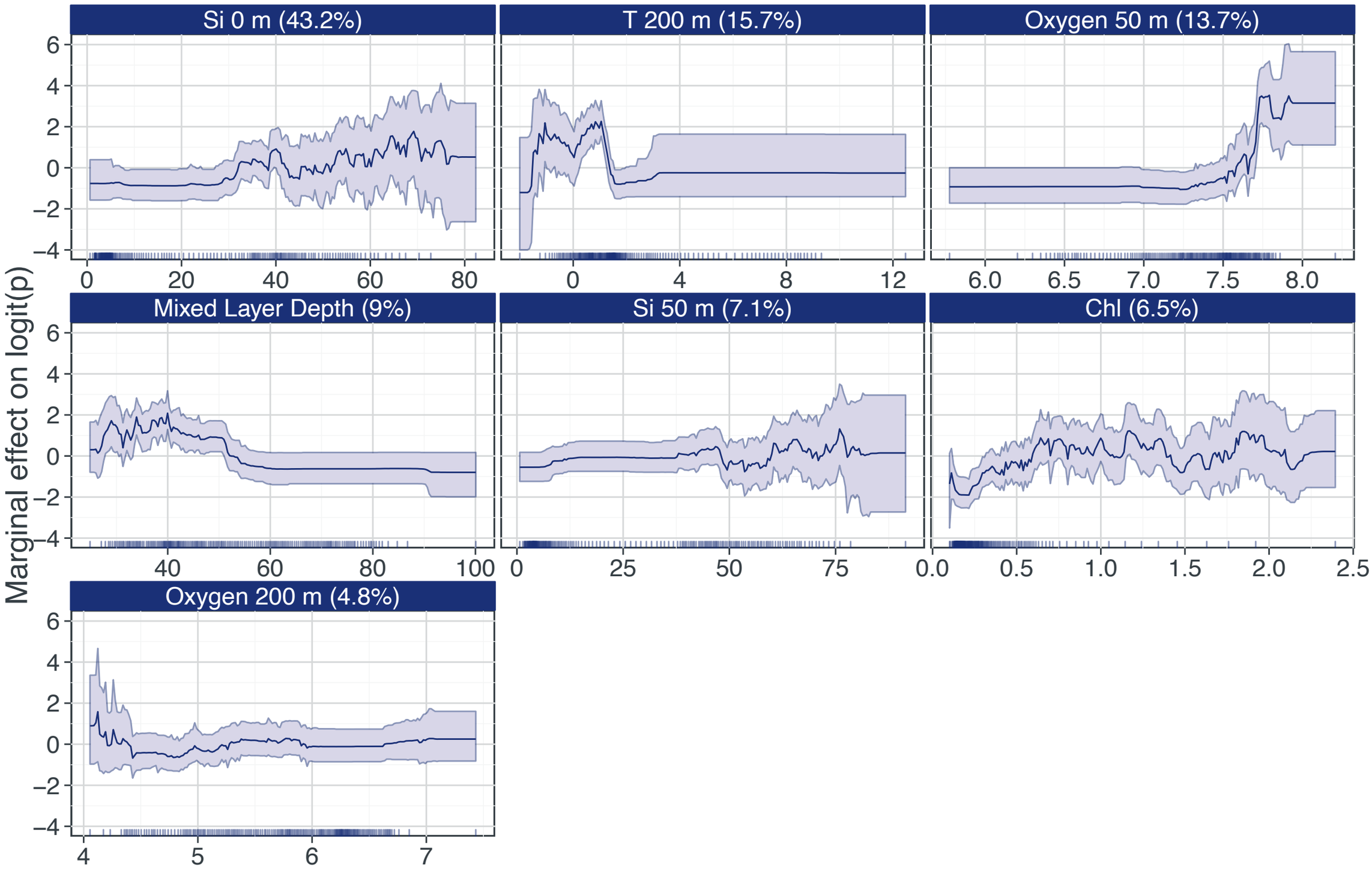
Projected within the observed environmental space

Irisson et al. in Biogeographic Atlas of the Southern Ocean, 2014



Map 3b
Modelled habitat suitability

Krill distribution in the Austral Ocean



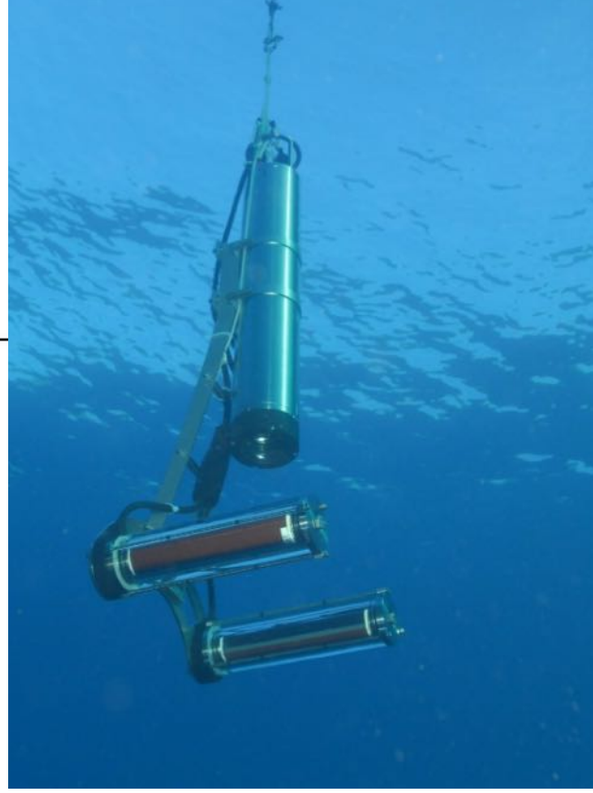
Predicting global plankton biomass

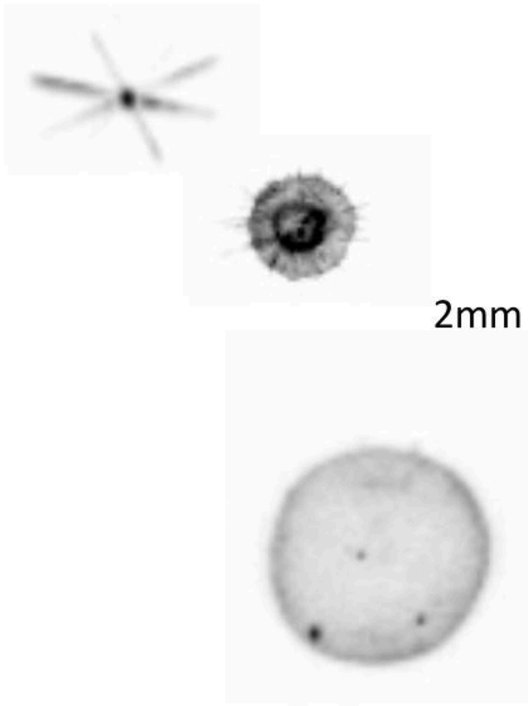
Individual biomass records, from images

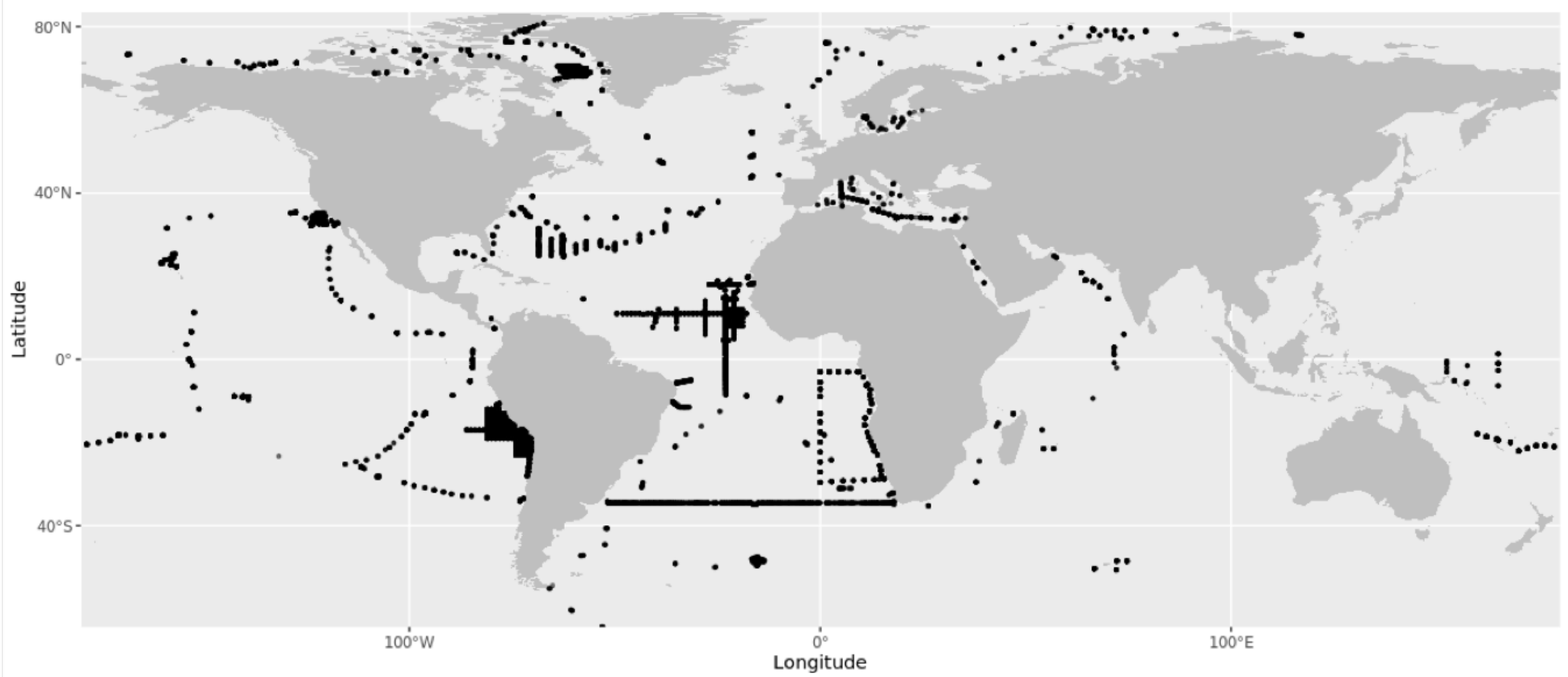
Poisson regression

On ~10 environmental climatologies

Drago et al. Frontiers in Marine Science, 2022







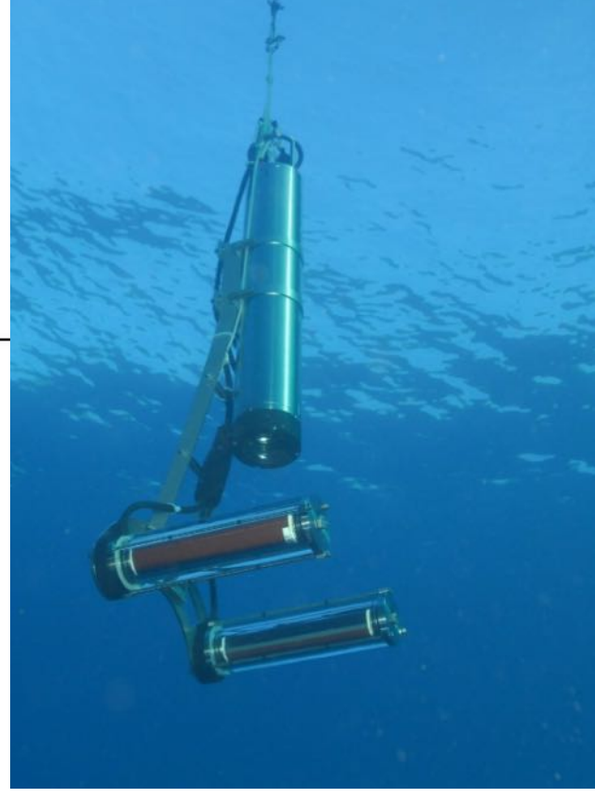
Predicting global plankton biomass

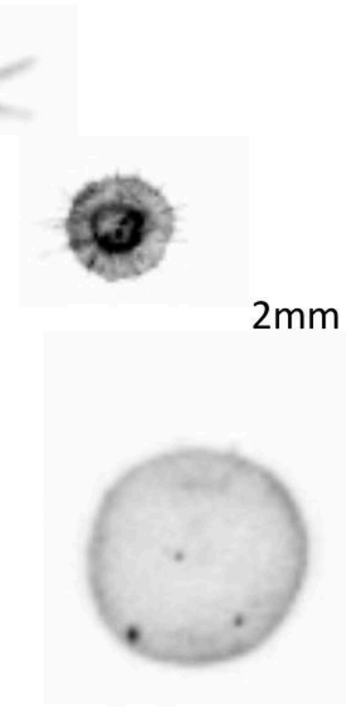
Individual biomass records, from images

Poisson regression

On ~10 environmental climatologies

Drago et al. Frontiers in Marine Science, 2022





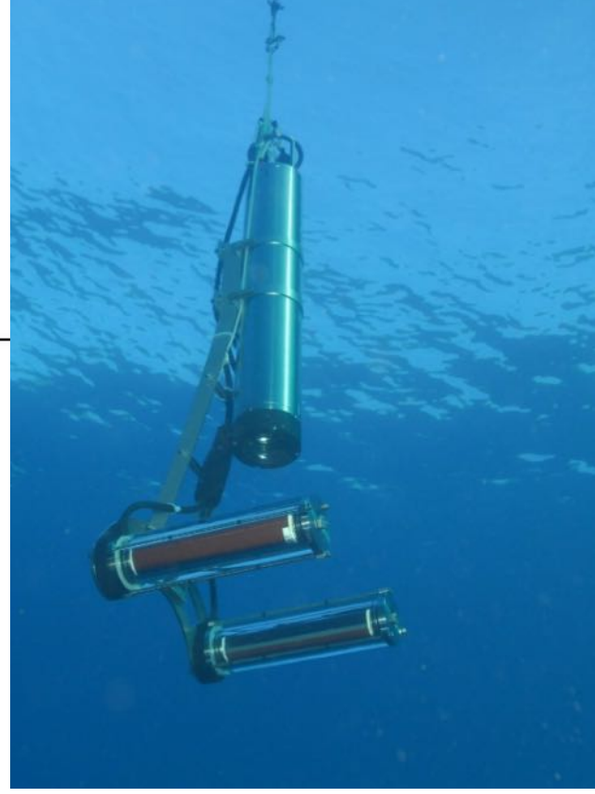
Predicting global plankton biomass

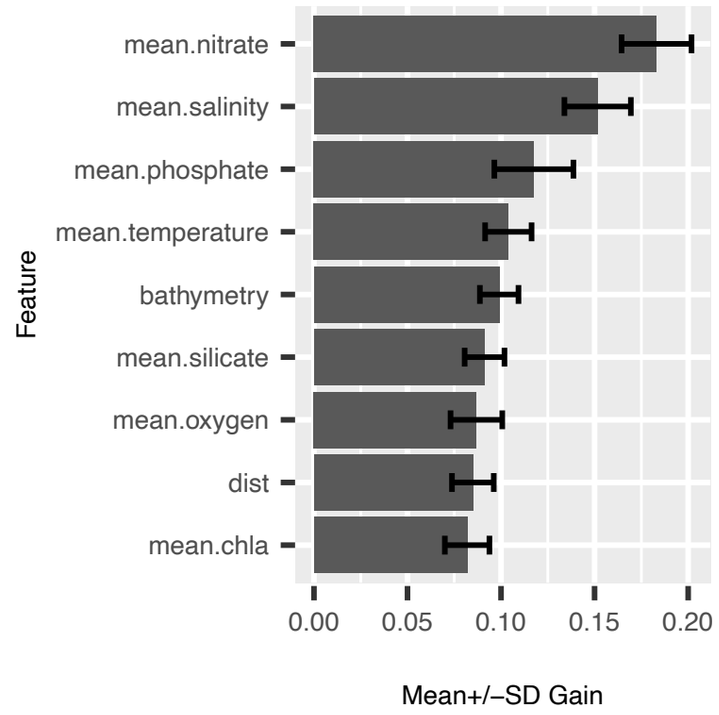
Individual biomass records, from images

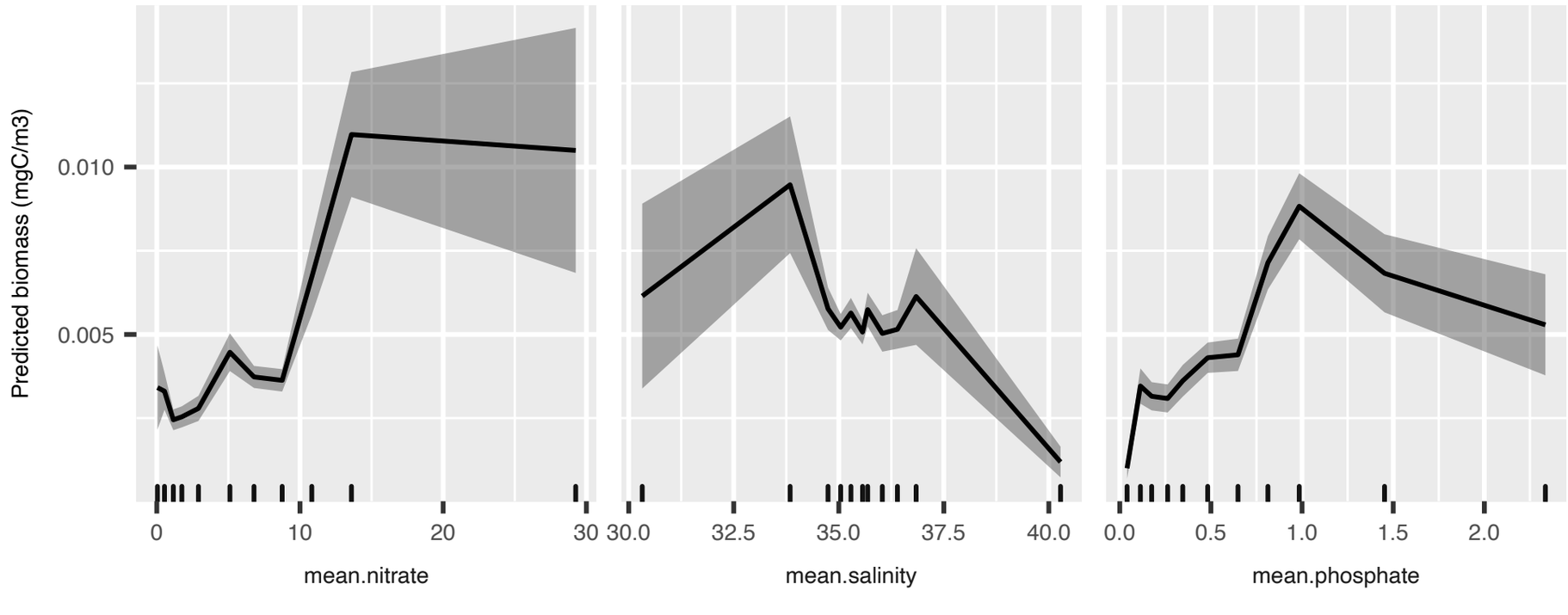
Poisson regression

On ~10 environmental climatologies

Drago et al. Frontiers in Marine Science, 2022







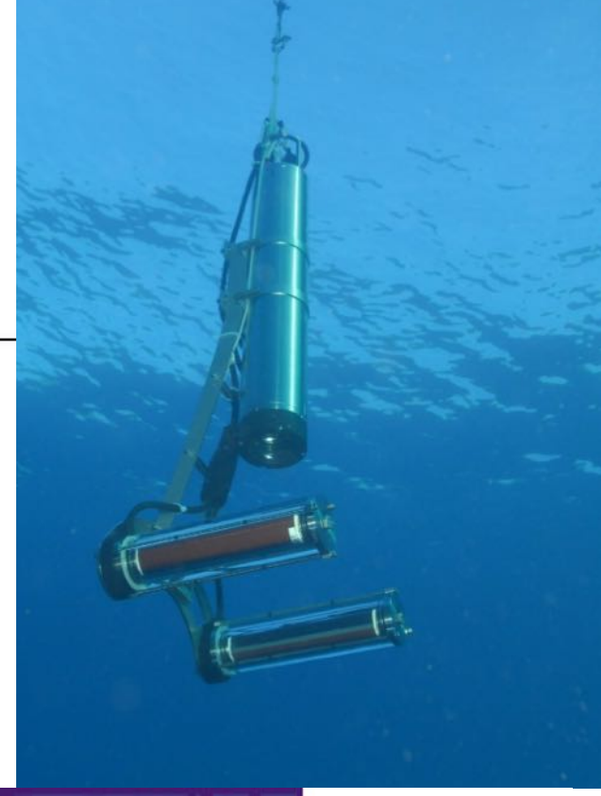
Predicting global plankton biomass

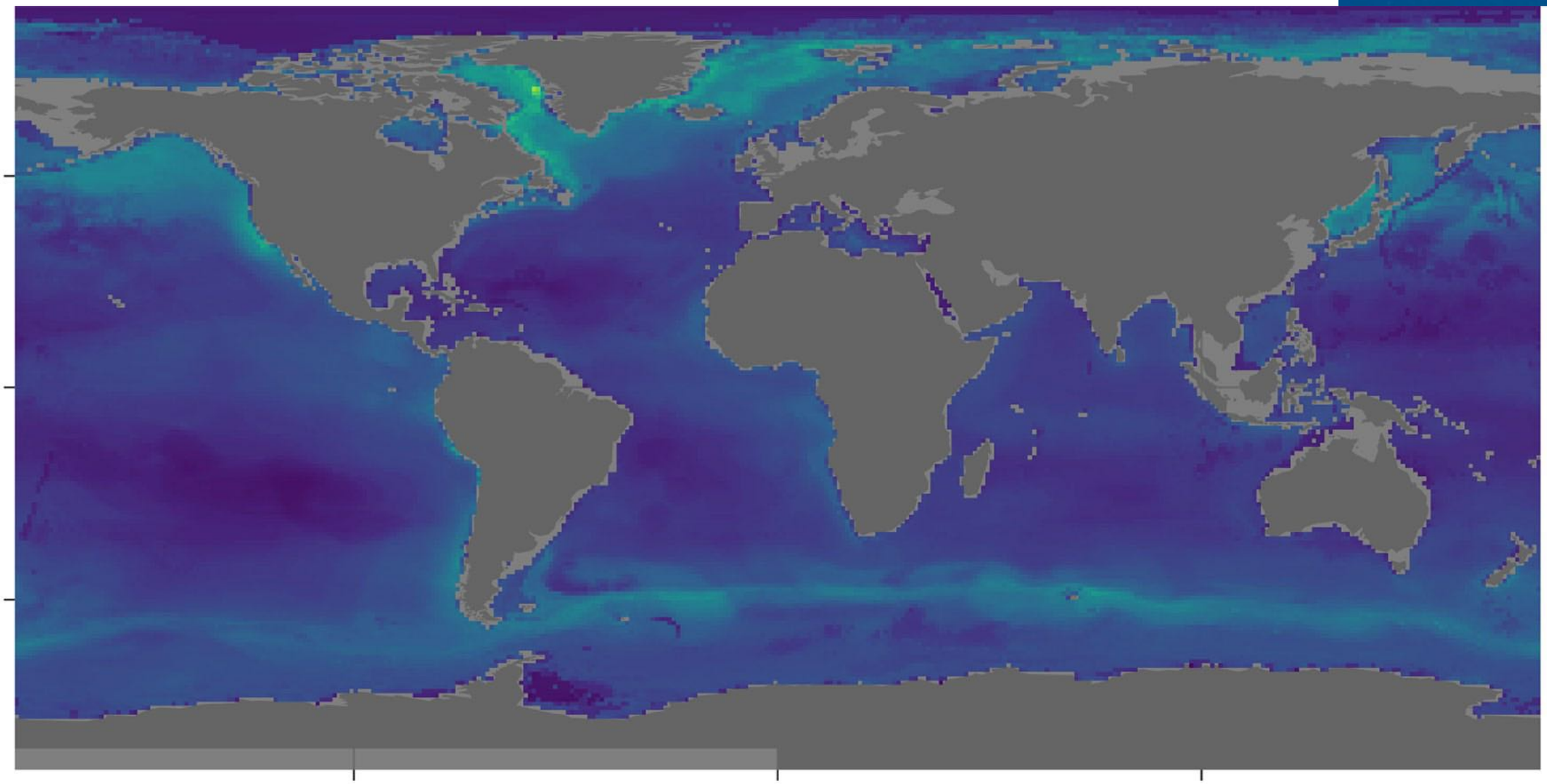
Individual biomass records, from images

Poisson regression

On ~10 environmental climatologies

Drago et al. Frontiers in Marine Science, 2022





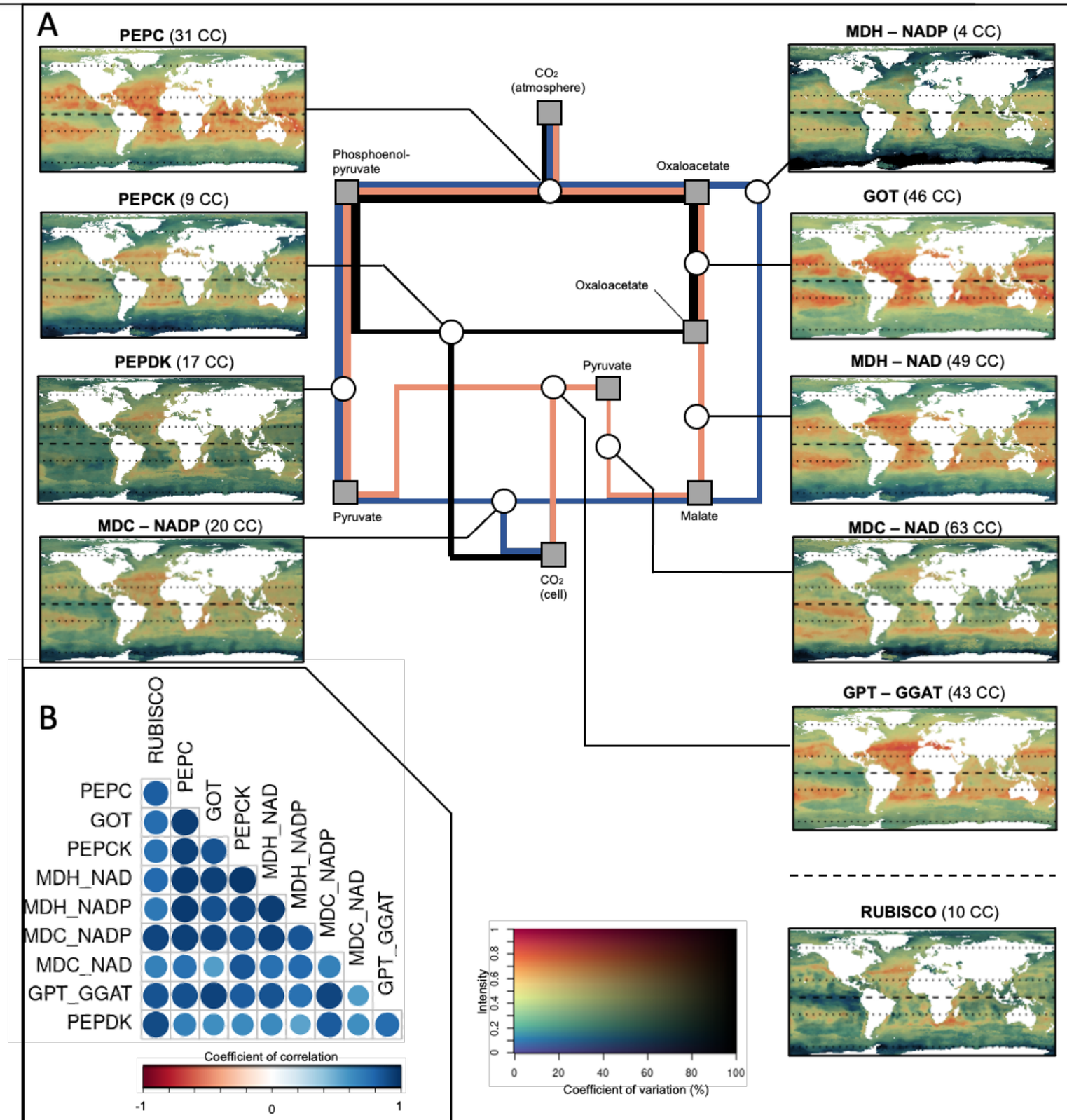
Potential distribution of carbon concentration mechanisms in picoeukaryotes

Proportions of different enzymes in the sample, from metagenomics

Multi-output least-square regression

On environmental climatologies of ~10 variables, including mean and variance

Schickele et al., in progress, 2022 (hopefully 😉)



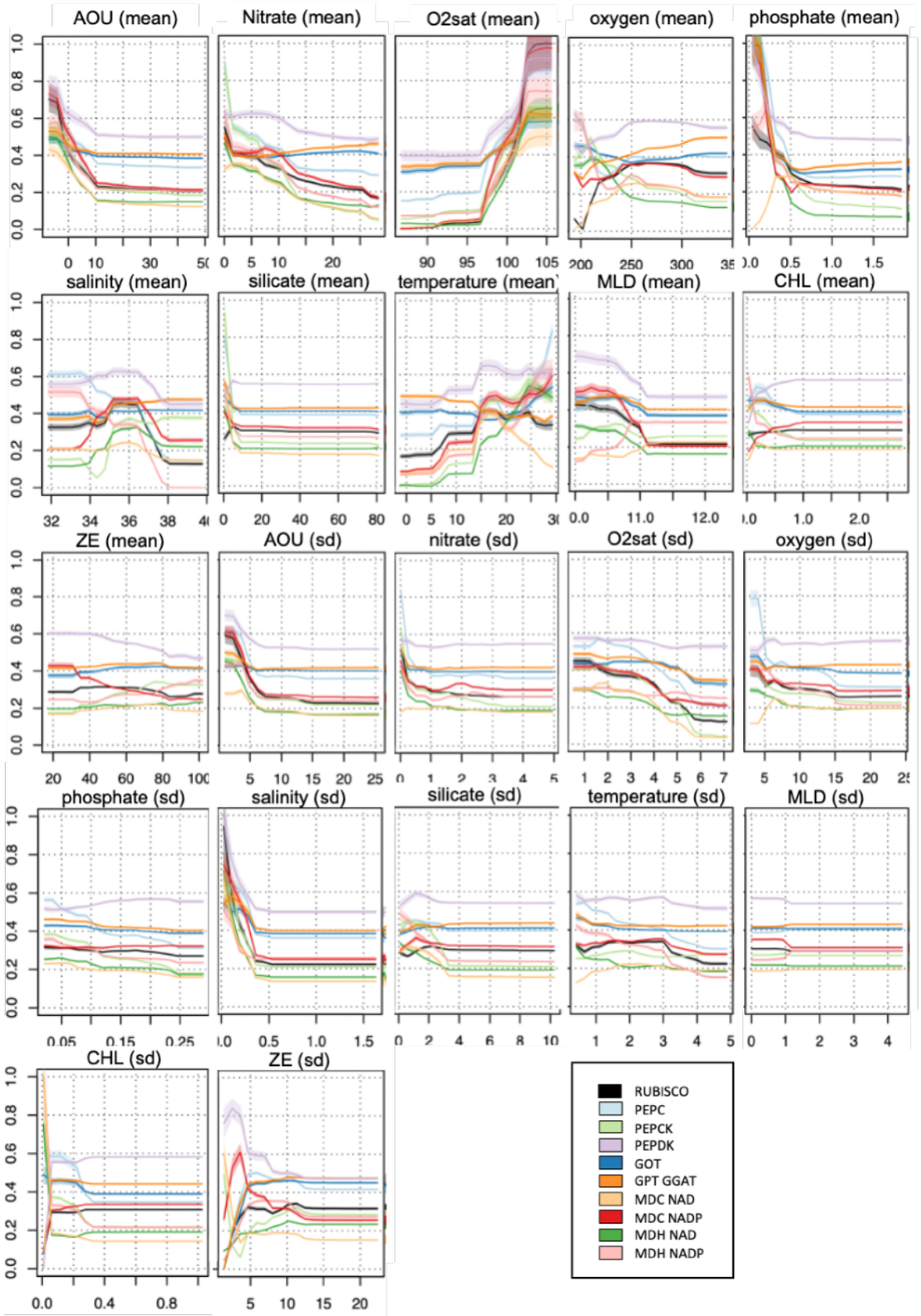
Potential distribution of carbon concentration mechanisms in picoeukaryotes

Proportions of different enzymes in the sample, from metagenomics

Multi-output least-square regression

On environmental climatologies of ~10 variables, including mean and variance

Schickele et al., in progress, 2022 (hopefully 😉)



Merci

Jean-Olivier Irisson
Laboratoire d'Océanographie de Villefranche (LOV)
irisson@normalesup.org