Advanced Tree-Based Methods (MSBA 7027)
CART, Bagging, Random Forest

Zhengli Wang

Faculty of Business and Economics
The University of Hong Kong

2023

Classification & Regression Tree (CART)

Algorithm class: Non-parametric

Mechanism: Partition feature space into smaller regions
using a set of binary splitting rules

Applicable: Both classification and regression problem

Advantage: Easy to interpret & visualize; Little Data Preprocessing

CART usually the 1st few models to try when faced with new datasets

CART Structure

Partition feature space into subgroups using binary splitting rules

Subgroups formed recursively using binary partitions (with simple yes-or-no response from features, e.g. is age < 18?)

Repeat until a stopping criteria is satisfied (e.g. min. node size / max. tree depth)

Output Prediction:
Regression problem: Mean from subgroup
Classification problem: Majority vote from subgroup (Prob. by proportion)

CART Example

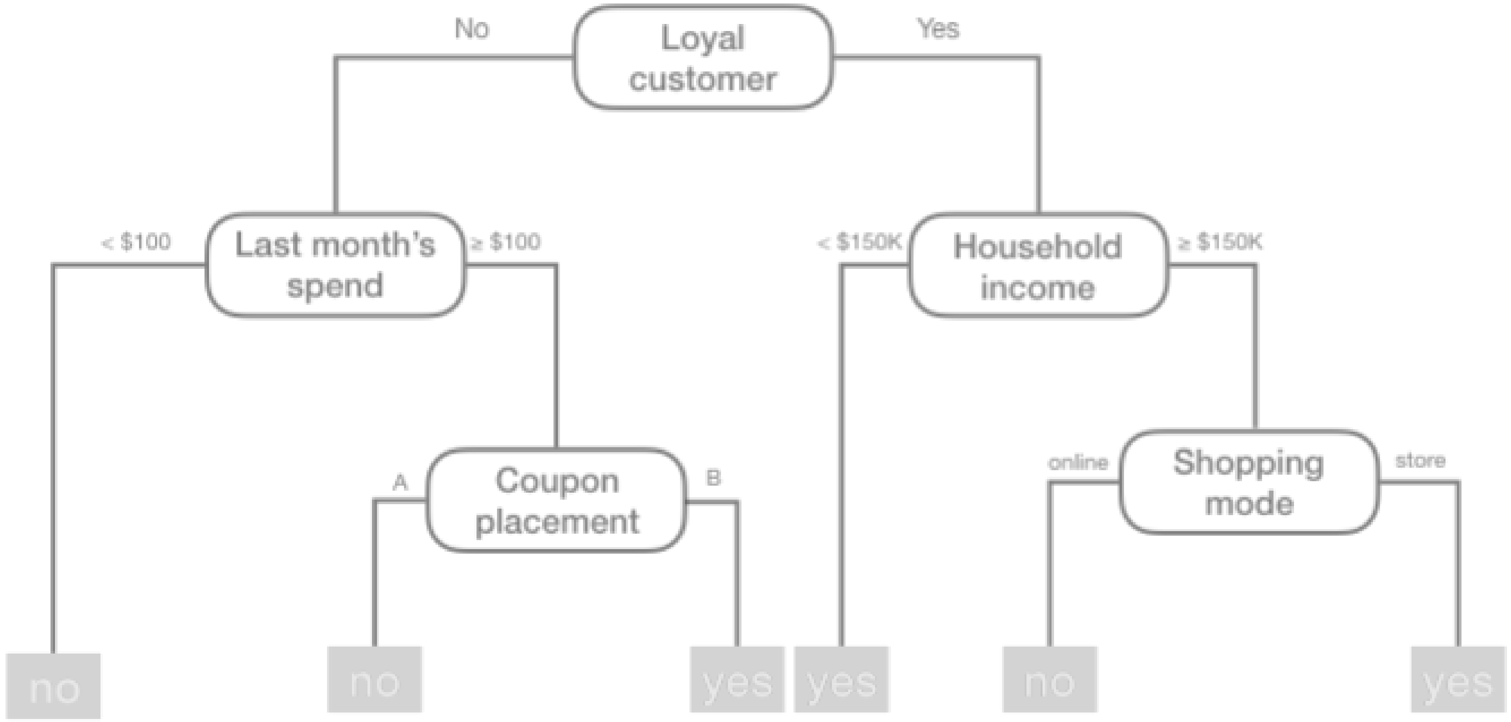
Predict whether a customer redeem a coupon

y	x1	x2	x3	x4	x5
Customer Response	Customer Loyalty	House Income	Last Month Spend	Coupon Placement	Shopping Mode

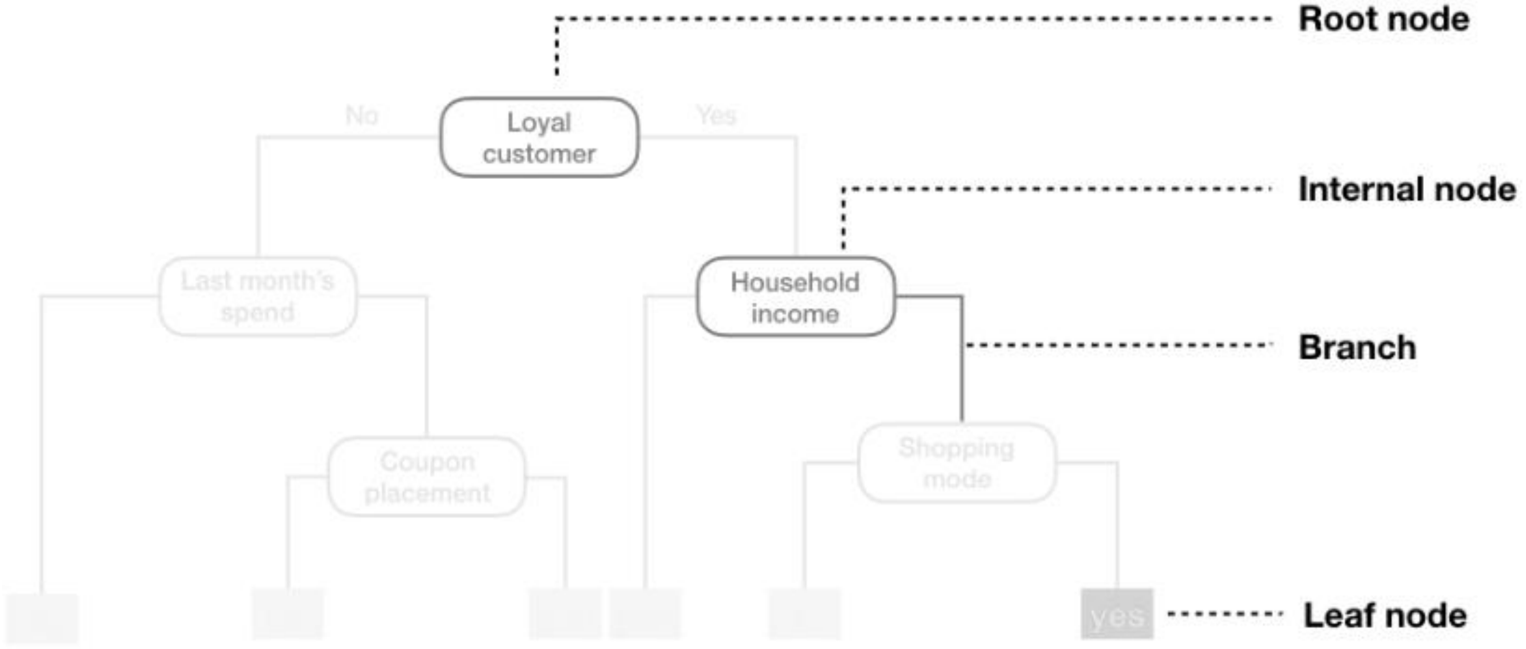
CART Example

Predict whether a customer redeem a coupon

y	x1	x2	x3	x4	x5
Customer Response	Customer Loyalty	House Income	Last Month Spend	Coupon Placement	Shopping Mode

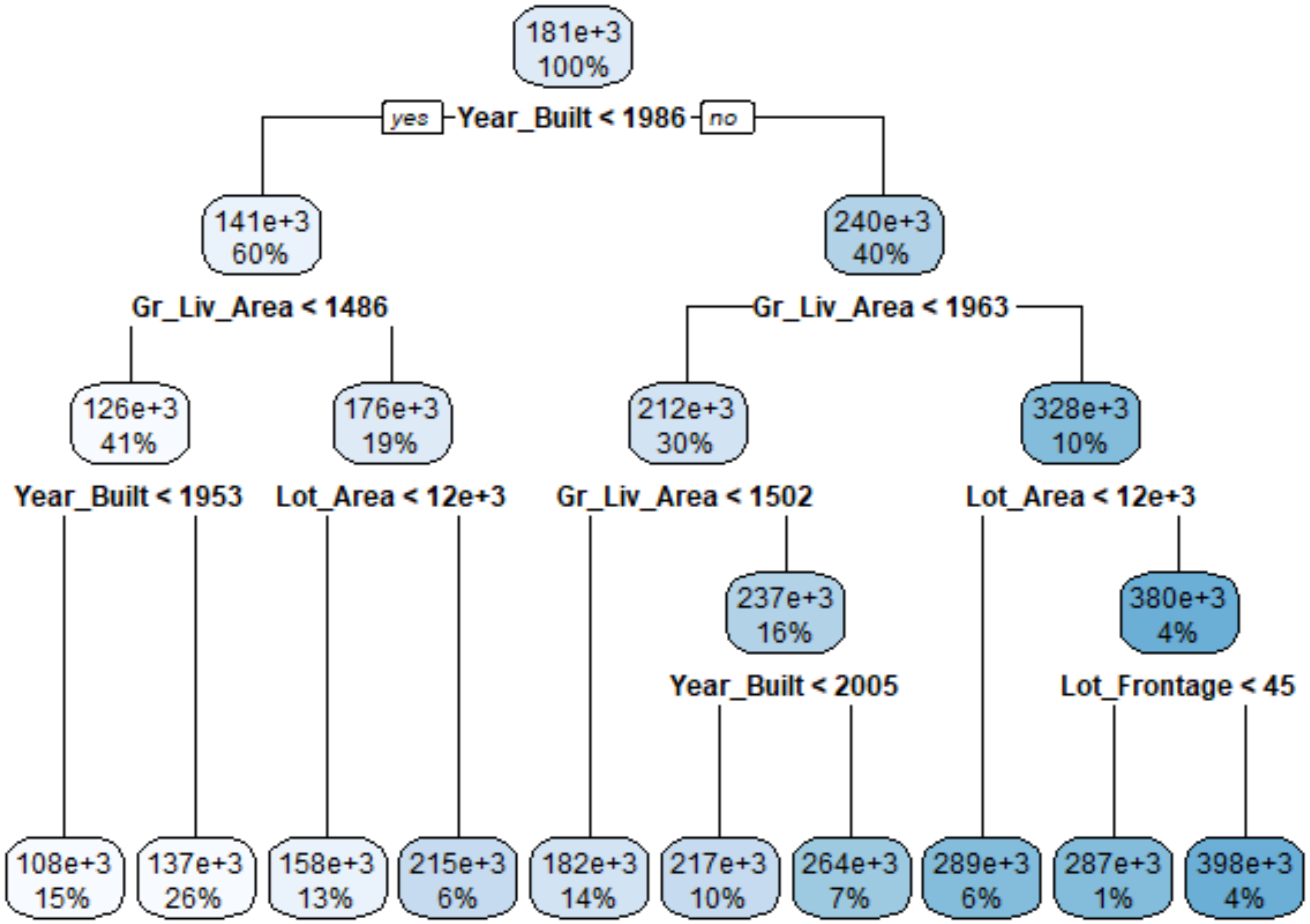


CART Terminology



Size of tree = # terminal nodes
Max Depth of tree = # branches to the most distant leaf node

Another CART Example



How to Grow a Tree

Binary splitting rule

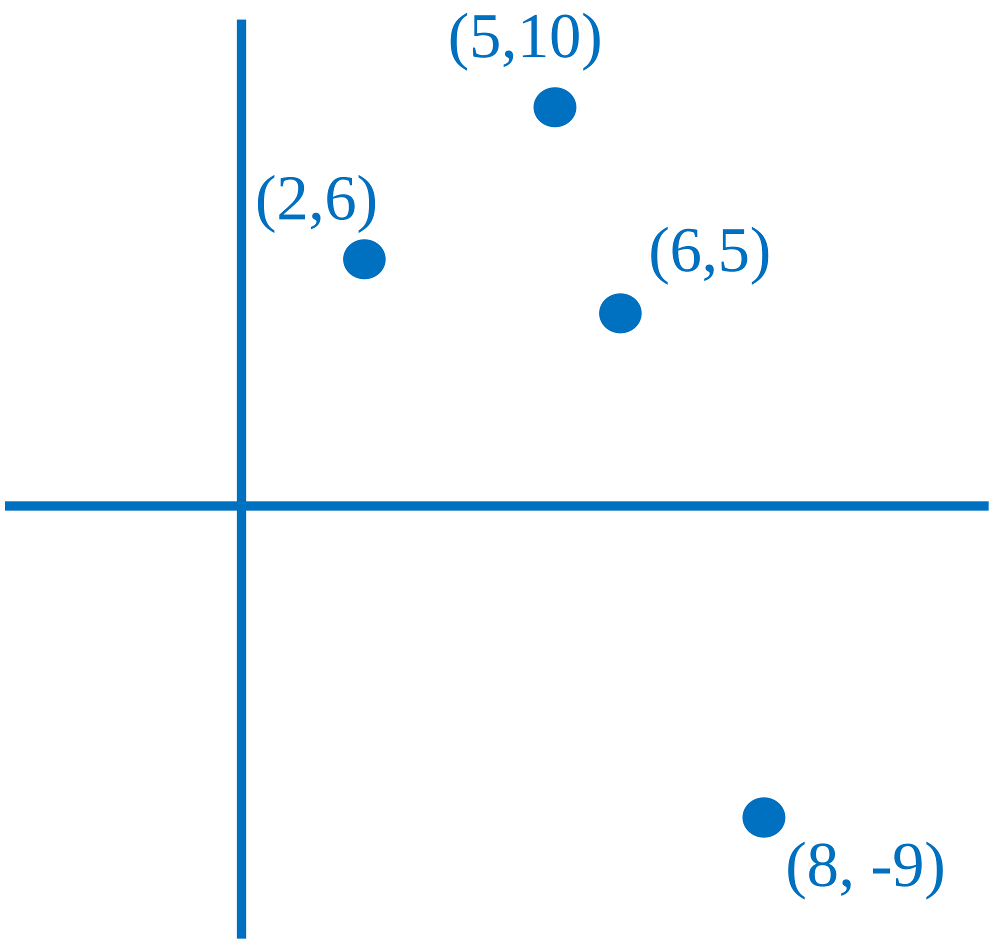
Regression Problem: split to achieve small SSE

Classification Problem: split to achieve small entropy

Keep spltting until **stopping criteria** reached (e.g. min. node size / max. tree depth)

How to Grow a Tree

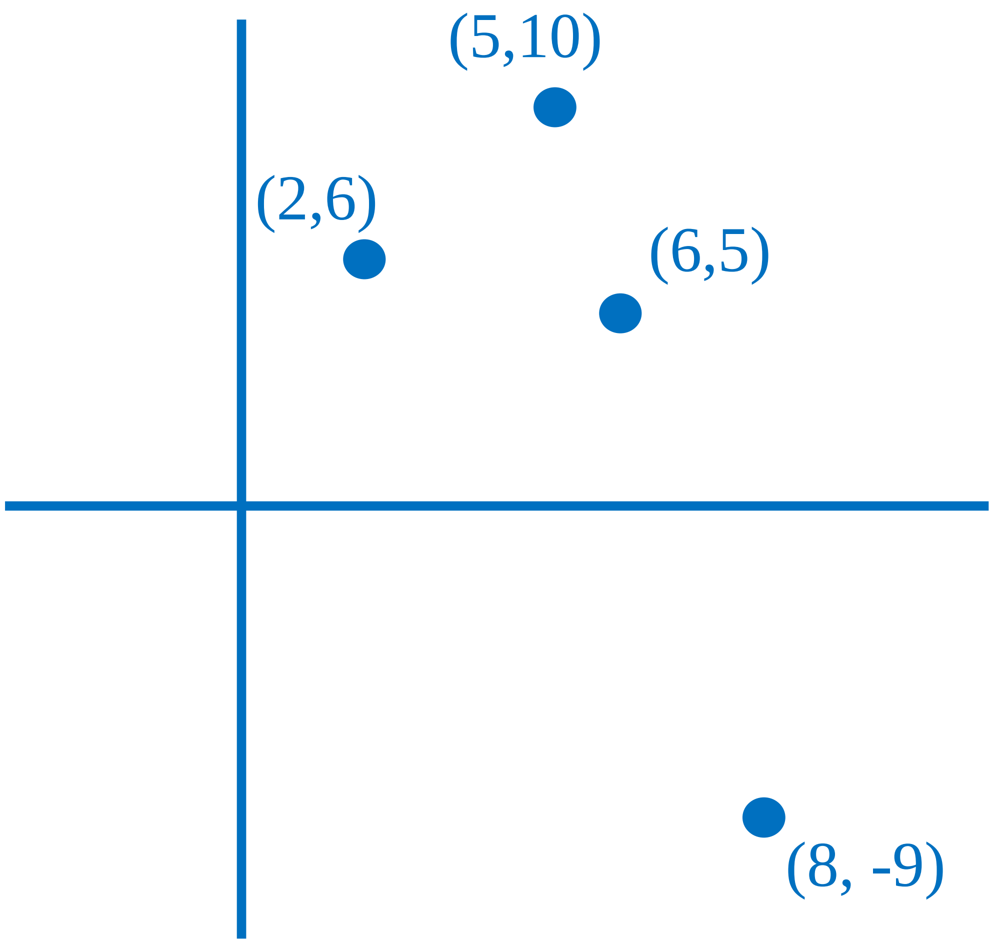
Numerical Example: Partition in Regression Problem



Split	L Pred	R Pred	L SSE	R SSE	T SSE
X=3.5	6	2	0	194	194
X=5.5	8	-2			
X=7	7	-9			

How to Grow a Tree

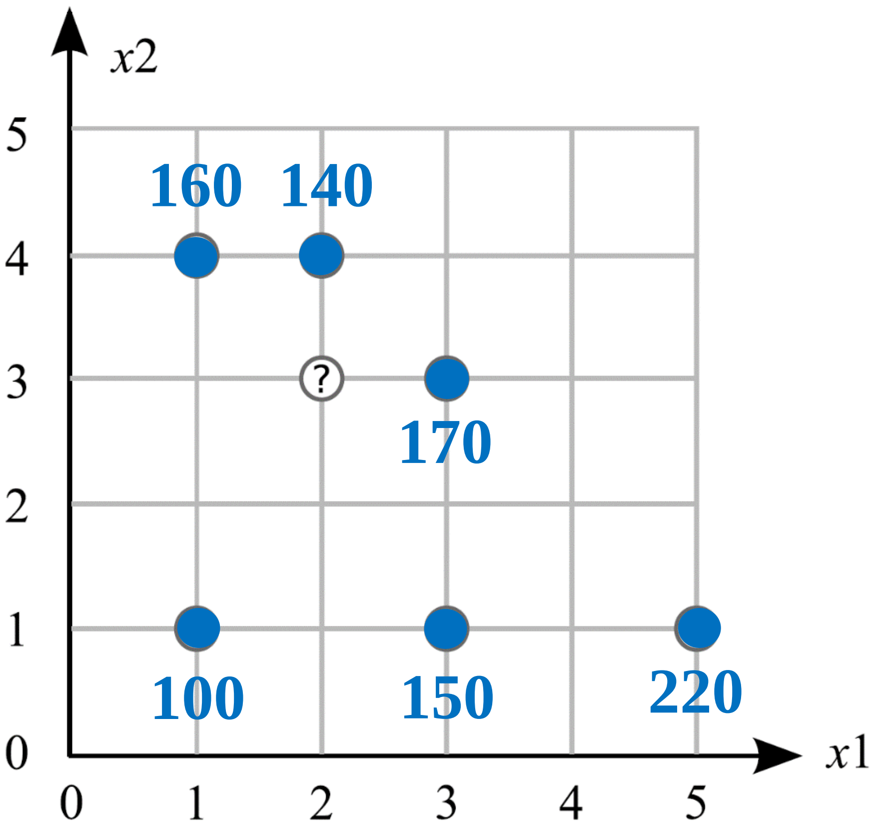
Numerical Example: Partition in Regression Problem



Split	L Pred	R Pred	L SSE	R SSE	T SSE
X=3.5					
X=5.5					

How to Grow a Tree

Numerical Example: Partition in Regression Problem



Split at x1 = 1.5:

Split at x1 = 2.5:

$(160 - 400/3)^2 + (140 - 400/3)^2 + (100 - 400/3)^2$

$+ (170 - 180)^2 + (150 - 180)^2 + (220 - 180)^2 = 4466.66$

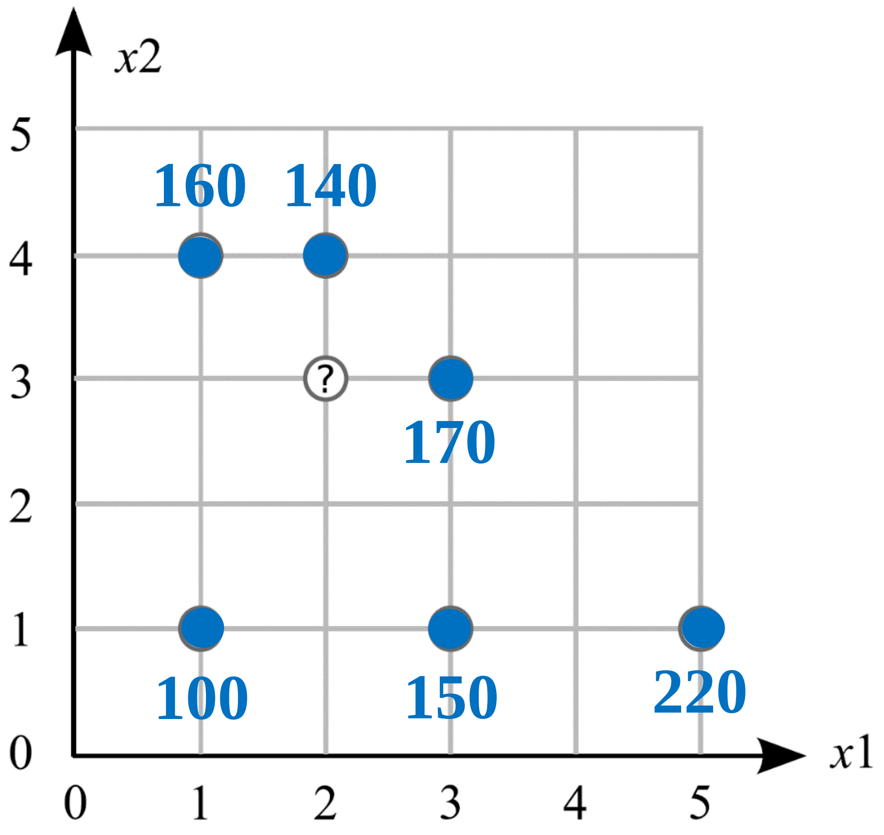
Split at x1 = 4:

$(160 - 144)^2 + (140 - 144)^2 + (170 - 144)^2 + (100 - 144)^2 +$

$(150 - 144)^2 + (220 - 220)^2 = 2920$

How to Grow a Tree

Numerical Example: Partition in Regression Problem



Split at x2 = 2:

$(160 - 470/3)^2 + (140 - 470/3)^2 + (170 - 470/3)^2$

$+ (100 - 470/3)^2 + (150 - 470/3)^2 + (220 - 470/3)^2 = 7733.33$

Split at x2 = 3.5:

Which is the best split?

How to Grow a Tree

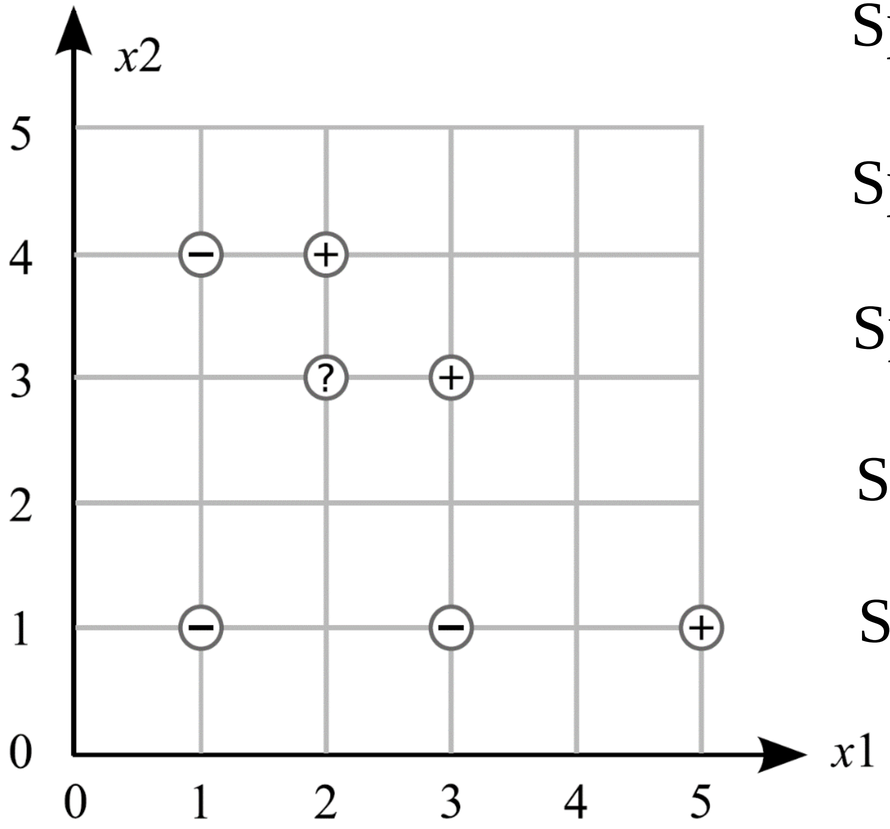
Numerical Example: Partition in Classification Problem

- Entropy $\sum_{i=1}^{K} -p_i log_2(p_i)$ where K = #classes, p_i=proportion of class i
  - value >= 0, smaller value is better
    - 2 classes: 0 <= value <= 1,
    - >2 classes: 0 <= value <= ∞

$$\sum_{i=1}^{2} -(1/2)log(1/2) = 1, \sum_{i=1}^{K} -(1/K)log(1/K) = logK$$

How to Grow a Tree

Numerical Example: Partition in Classification Problem



Split at x1 = 1.5: $\dfrac{2(0) + 4\left(-\frac{3}{4}log\frac{3}{4} - \frac{1}{4}log\frac{1}{4}\right)}{6} = 0.541$

Split at x1 = 2.5: $\dfrac{3\left(-\frac{1}{3}log\frac{1}{3} - \frac{2}{3}log\frac{2}{3}\right) + 3\left(-\frac{1}{3}log\frac{1}{3} - \frac{2}{3}log\frac{2}{3}\right)}{6} = 0.918$

Split at x1 = 4: $\dfrac{1(0) + 5\left(-\frac{2}{5}log\frac{2}{5} - \frac{3}{5}log\frac{3}{5}\right)}{6} = 0.809$

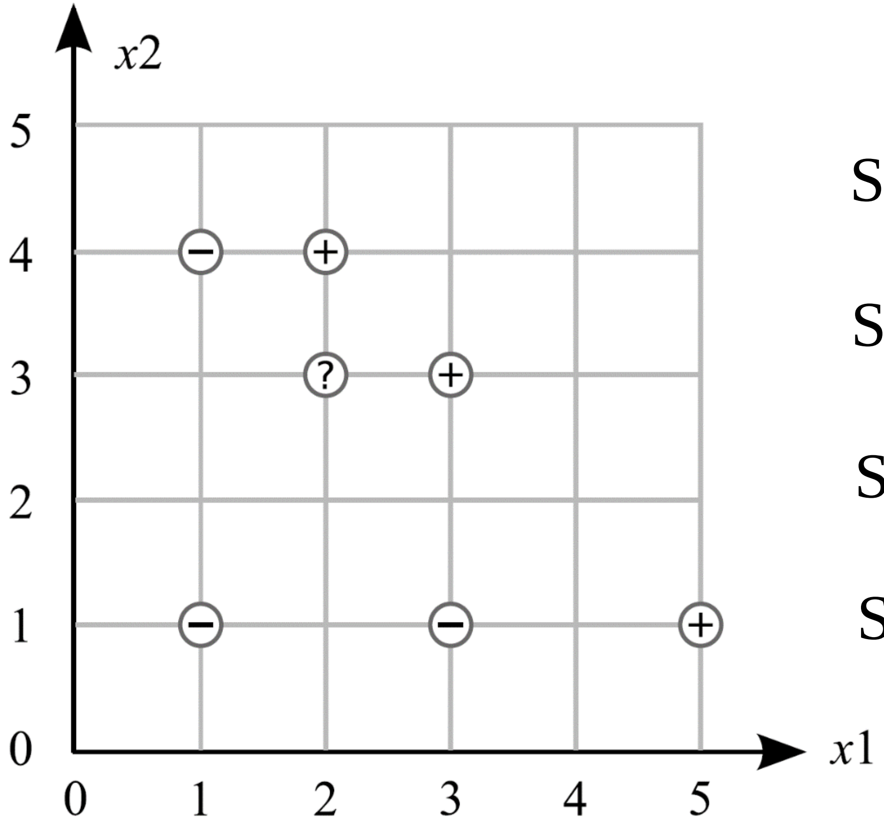
Split at x2 = 2:

Split at x2 = 3.5:

Which is the best split?

How to Grow a Tree

Numerical Example: Partition in Classification Problem



Split at x1 = 2.5: $\frac{1(0) + 3\left(-\frac{1}{3}log\frac{1}{3} - \frac{2}{3}log\frac{2}{3}\right)}{4} = 0.689$

Split at x1 = 4:

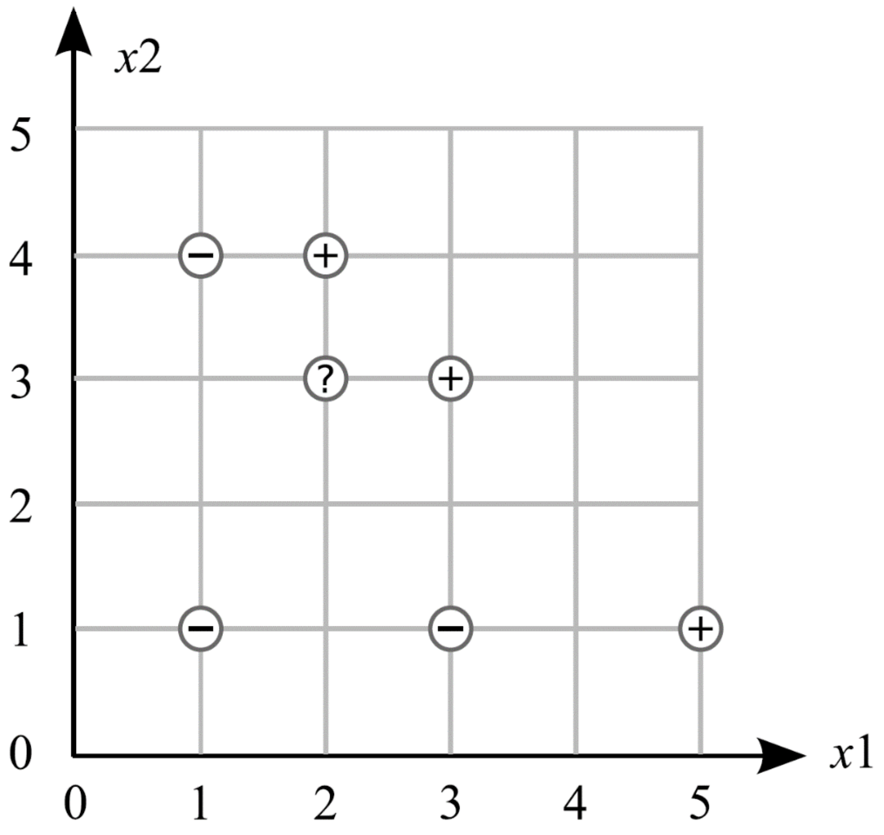
Split at x2 = 2:

Split at x2 = 3.5:

Which is the best split?

How to Grow a Tree

Numerical Example: Partition in Classification Problem



One final split & Draw Resulting Tree

How to Grow a Tree

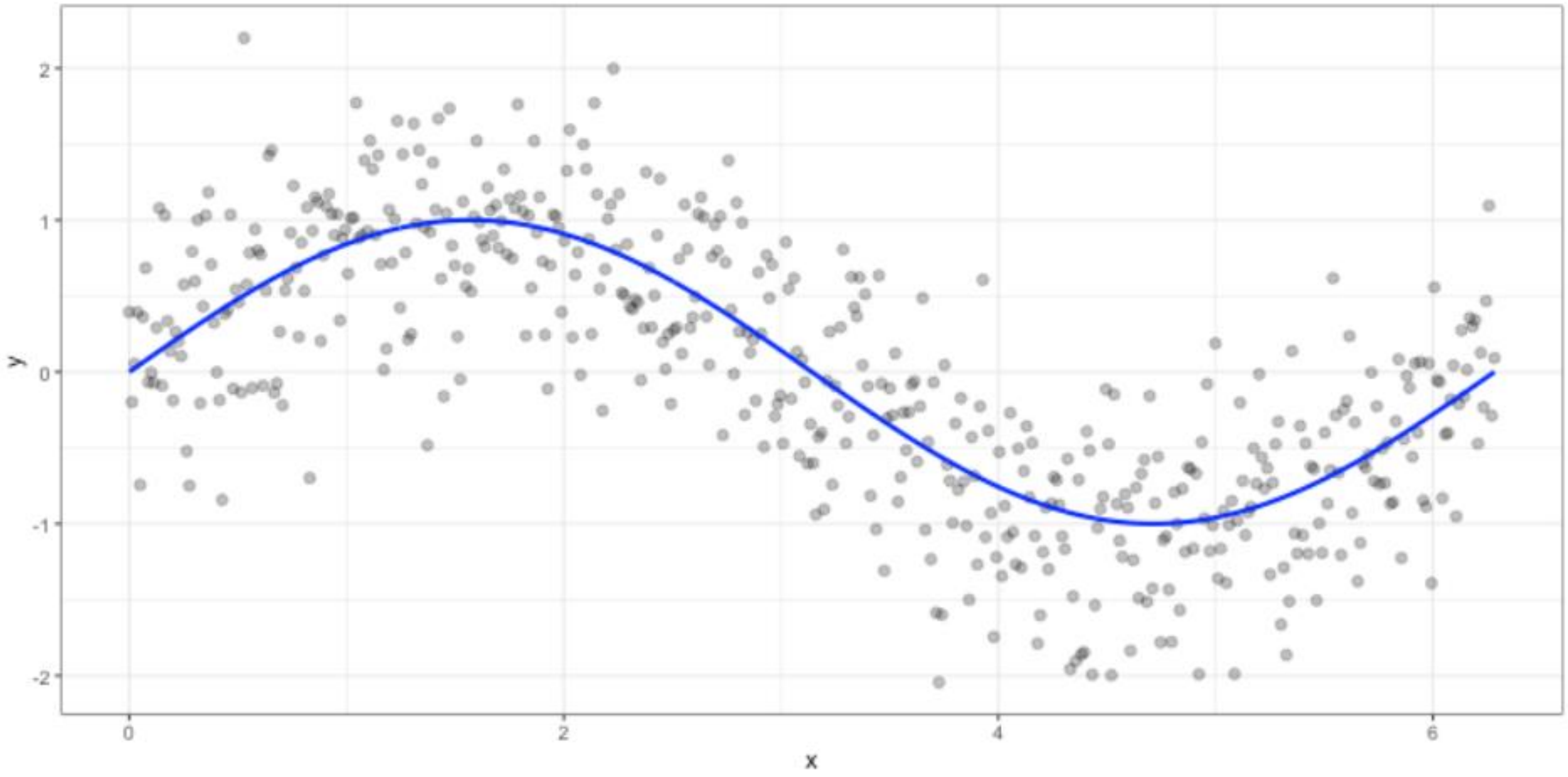
Note:

1. Way of splitting: greedy (may NOT be globally optimal)

2. The same variable can be involved in splitting multiple times

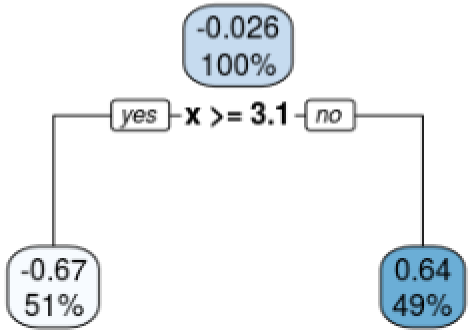
How Deep to Grow the Tree

Say we have the given data generated from the underlying ‘truth' function

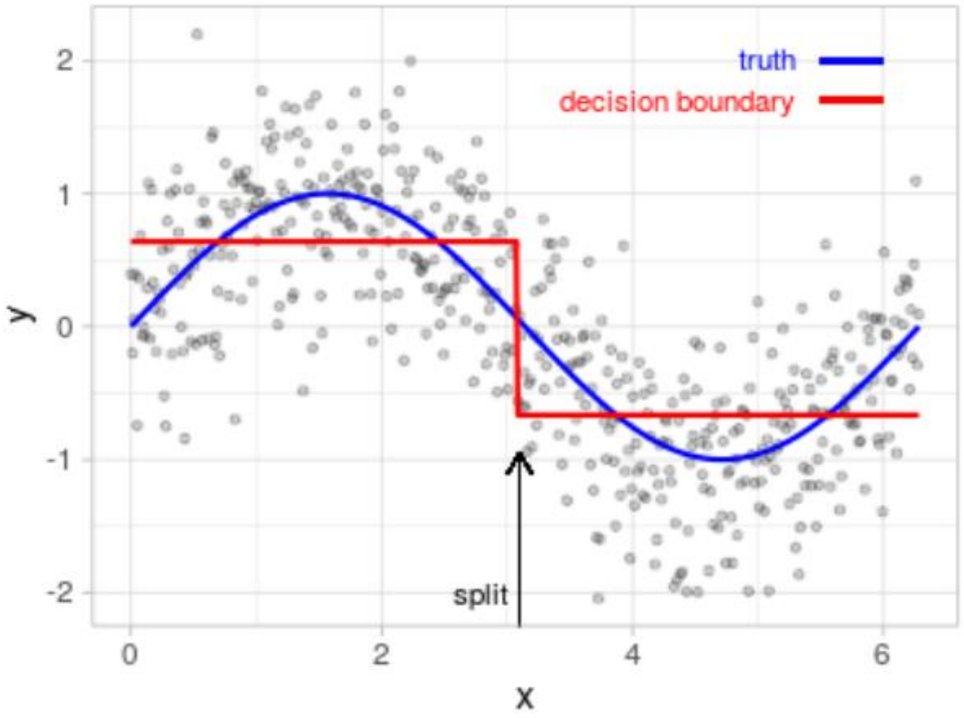


How Deep to Grow the Tree

Depth = 1 (decision stump)

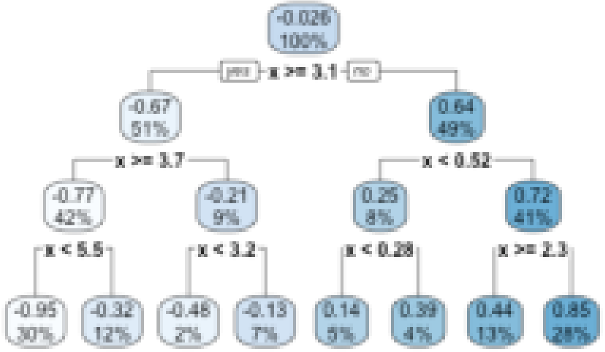



```
## Model formula:
## y ~ x
##
## Fitted party:
## [1] root
## |   [2] x >= 3.07863: -0.665 (n = 255, err = 95.5)
## |   [3] x < 3.07863: 0.640 (n = 245, err = 75.9)
##
## Number of inner nodes: 1
## Number of terminal nodes: 2
```

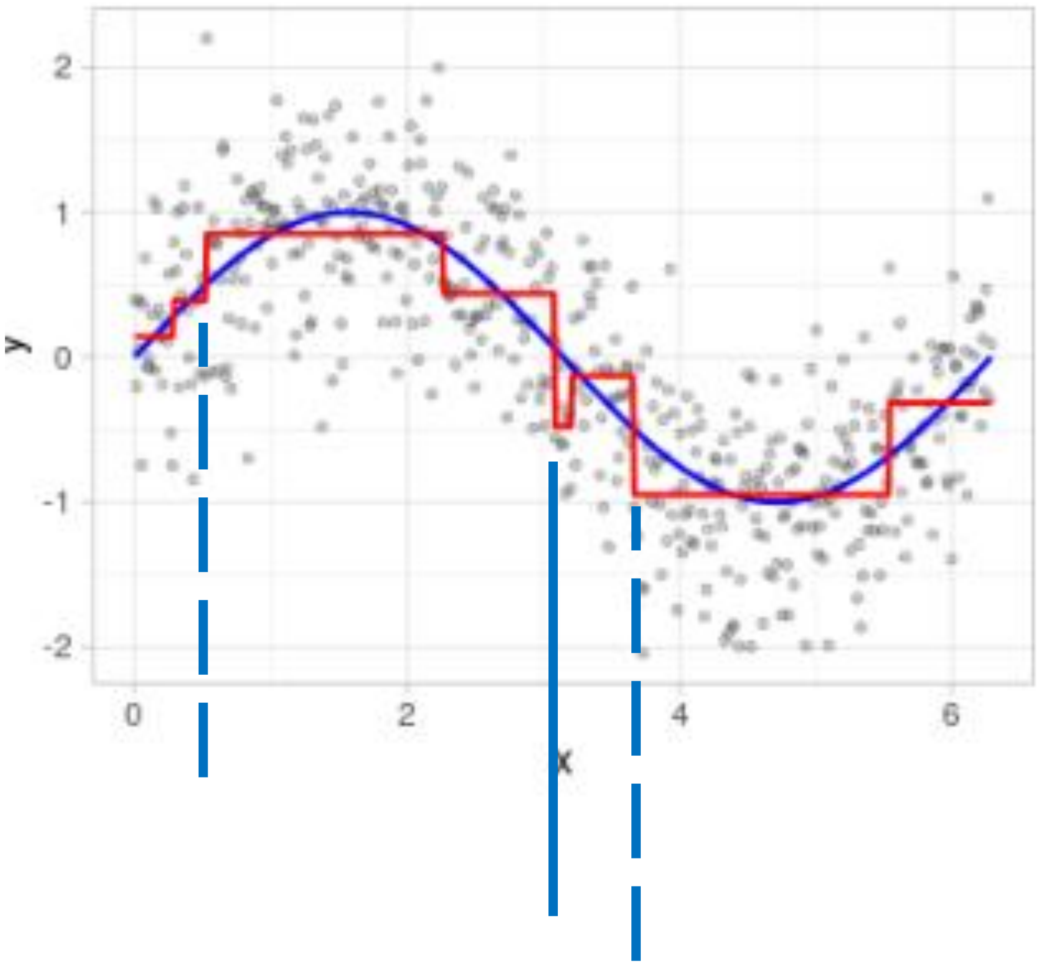


How Deep to Grow the Tree

Depth = 3

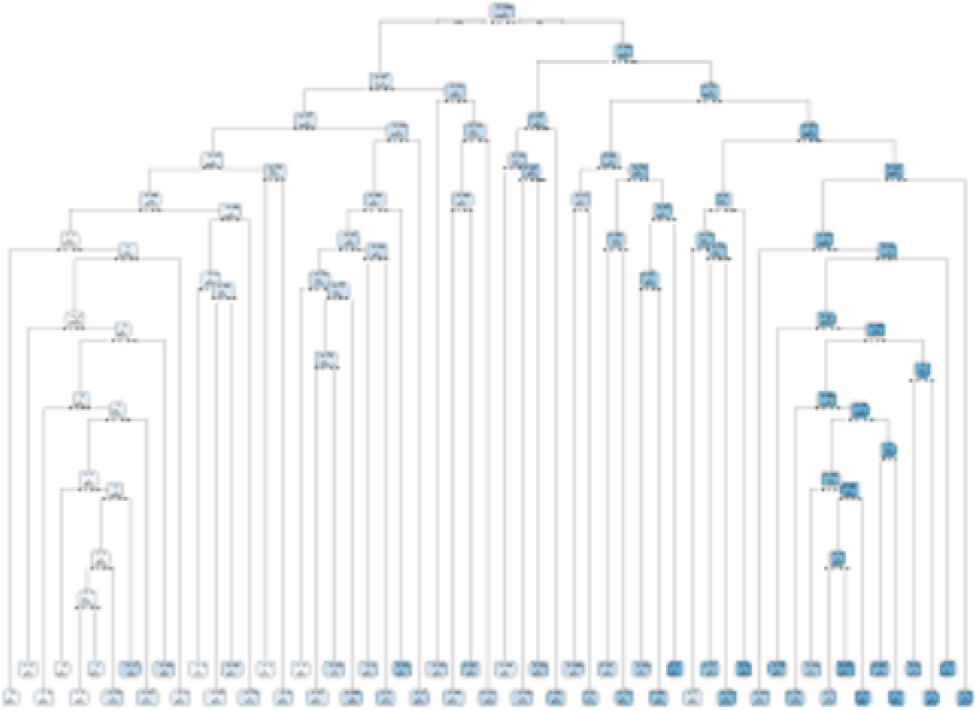


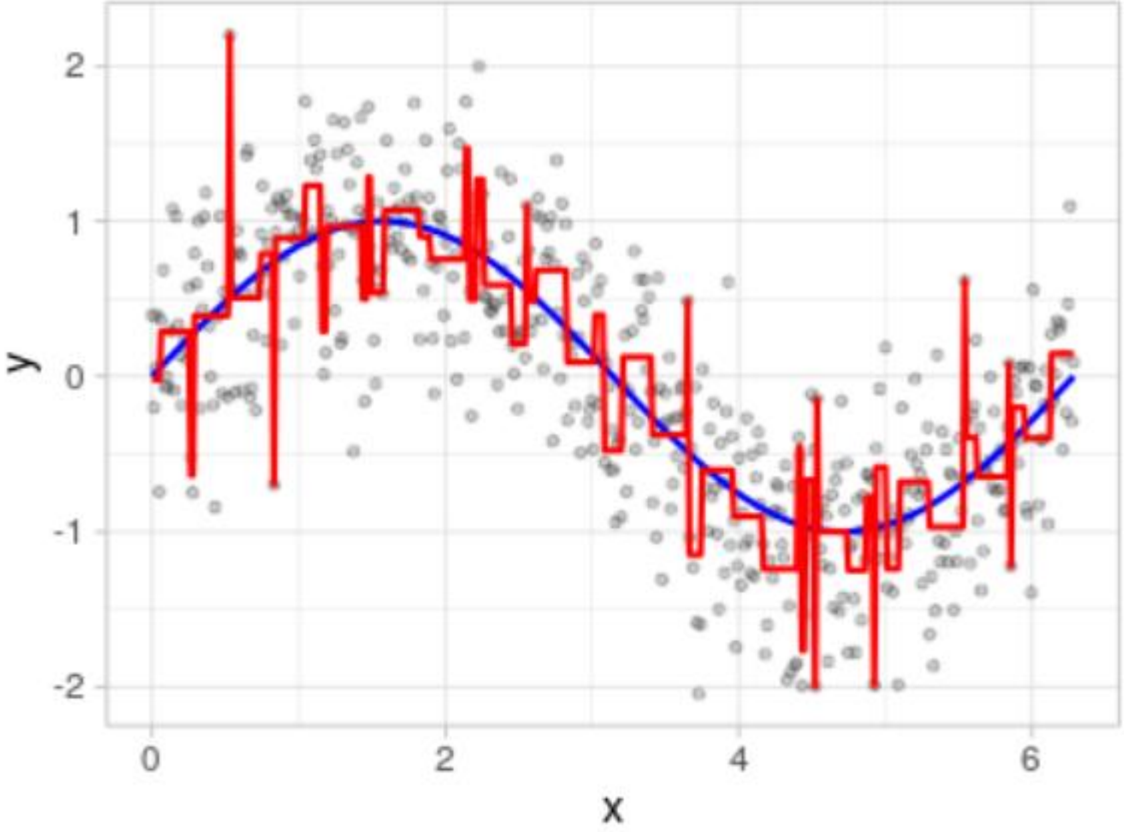

```
## Fitted party:
## [1] root
## | [2] x >= 3.07863
## | | [3] x >= 3.65785
## | | | [4] x < 5.53399: -0.948 (n = 149, err = 40.0)
## | | | [5] x >= 5.53399: -0.316 (n = 60, err = 15.6)
## | | [6] x < 3.65785
## | | | [7] x < 3.20455: -0.476 (n = 10, err = 0.9)
## | | | [8] x >= 3.20455: -0.130 (n = 36, err = 9.0)
## | [9] x < 3.07863
## | | [10] x < 0.52255
## | | | [11] x < 0.28331: 0.142 (n = 23, err = 4.8)
## | | | [12] x >= 0.28331: 0.390 (n = 19, err = 5.1)
## | | [13] x >= 0.52255
## | | | [14] x >= 2.26018: 0.440 (n = 65, err = 13.7)
## | | | [15] x < 2.26018: 0.852 (n = 138, err = 36.6)
##
## Number of inner nodes: 7
## Number of terminal nodes: 8
```



How Deep to Grow the Tree

Depth = 20 (complex tree)



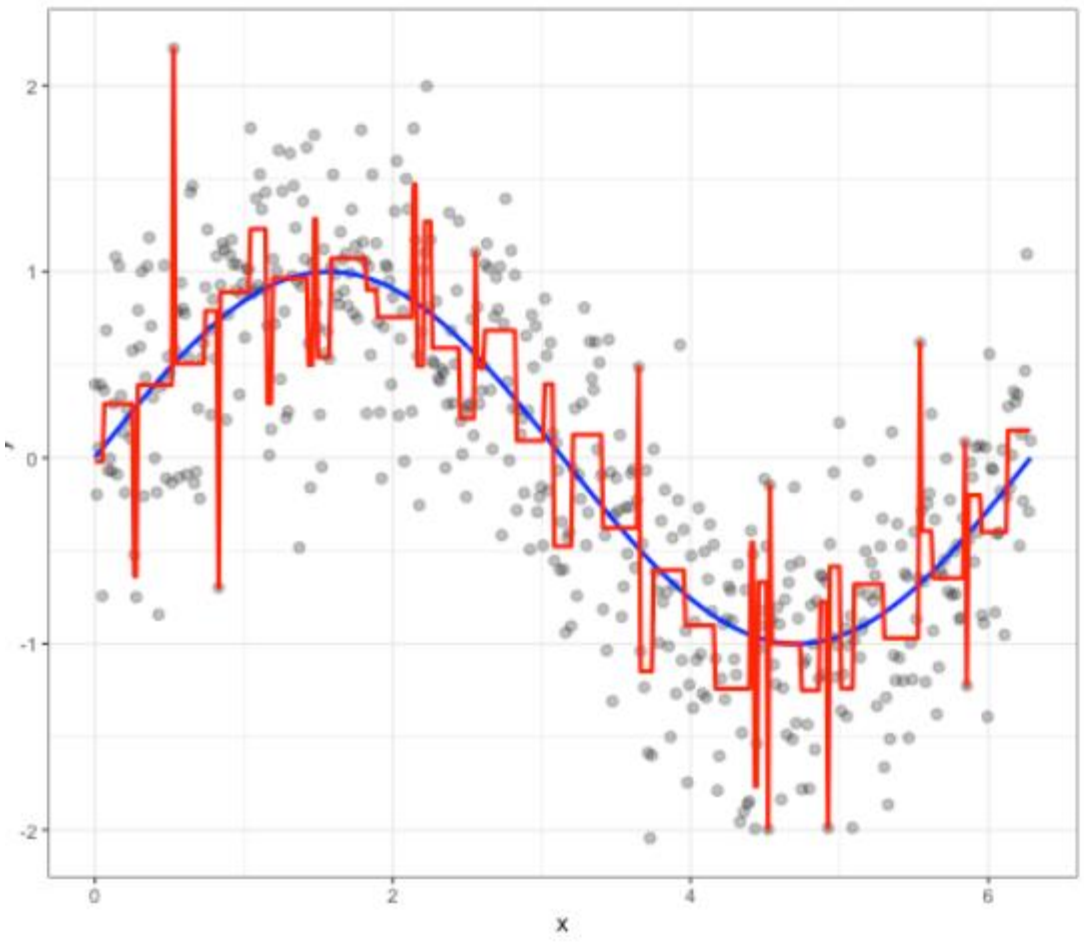


Minimize overfitting

- Trees have a tendency to overfit

- Avoid overfitting: Control tree complexity

- Restrict tree depth
- Restrict node size
- Complexity parameter cp



Minimize overfitting

- Restrict tree depth: Specify max tree depth
  - stop splitting after a certain depth

- Restrict node size: Specify minimum node size
  - do not split intermediate node which contains too few data points

Minimize overfitting

- Complexity parameter cp:
    - Any split that does NOT improve loss function by a factor of cp is NOT attempted
    - e.g. cp = 0.01

Source: https://stat.ethz.ch/R-manual/R-devel/library/rpart/html/rpart.control.html

Ames Housing Example

Initial Setup

Response variable

```
## Ames housing data
ames <- AmesHousing::make_ames()
summary(ames)

   Sale_Type     Sale_Condition    Sale_Price        Longitude         Latitude
 WD      :2536   Abnorml: 190   Min.   : 12789   Min.   :-93.69   Min.   :41.99
 New     : 239   AdjLand:  12   1st Qu.:129500   1st Qu.:-93.66   1st Qu.:42.02
 COD     :  87   Alloca :  24   Median :160000   Median :-93.64   Median :42.03
 ConLD   :  26   Family :  46   Mean   :180796   Mean   :-93.64   Mean   :42.03
 CWD     :  12   Normal :2413   3rd Qu.:213500   3rd Qu.:-93.62   3rd Qu.:42.05
 ConLI   :   9   Partial: 245   Max.   :755000   Max.   :-93.58   Max.   :42.06
 (Other):   21
 ...

## Create training set (70 %)for the Ames housing data
set.seed(123)
split  <- rsample::initial_split(ames, prop = 0.7, strata = "Sale_Price")
ames_train  <- rsample::training(split)
```

Ames Housing Example

Grow a tree (simple way) & Visualize, using **rpart** package

Train a default decision tree model

```
ames_dt1 <- rpart(
    formula = Sale_Price ~ .,
    data    = ames_train,
    method  = "anova"
)
```

- **For regression problem: method = "anova"**
- **For classification problem: method = "class"**

```
rpart.plot(ames_dt1, cex=0.5)
```

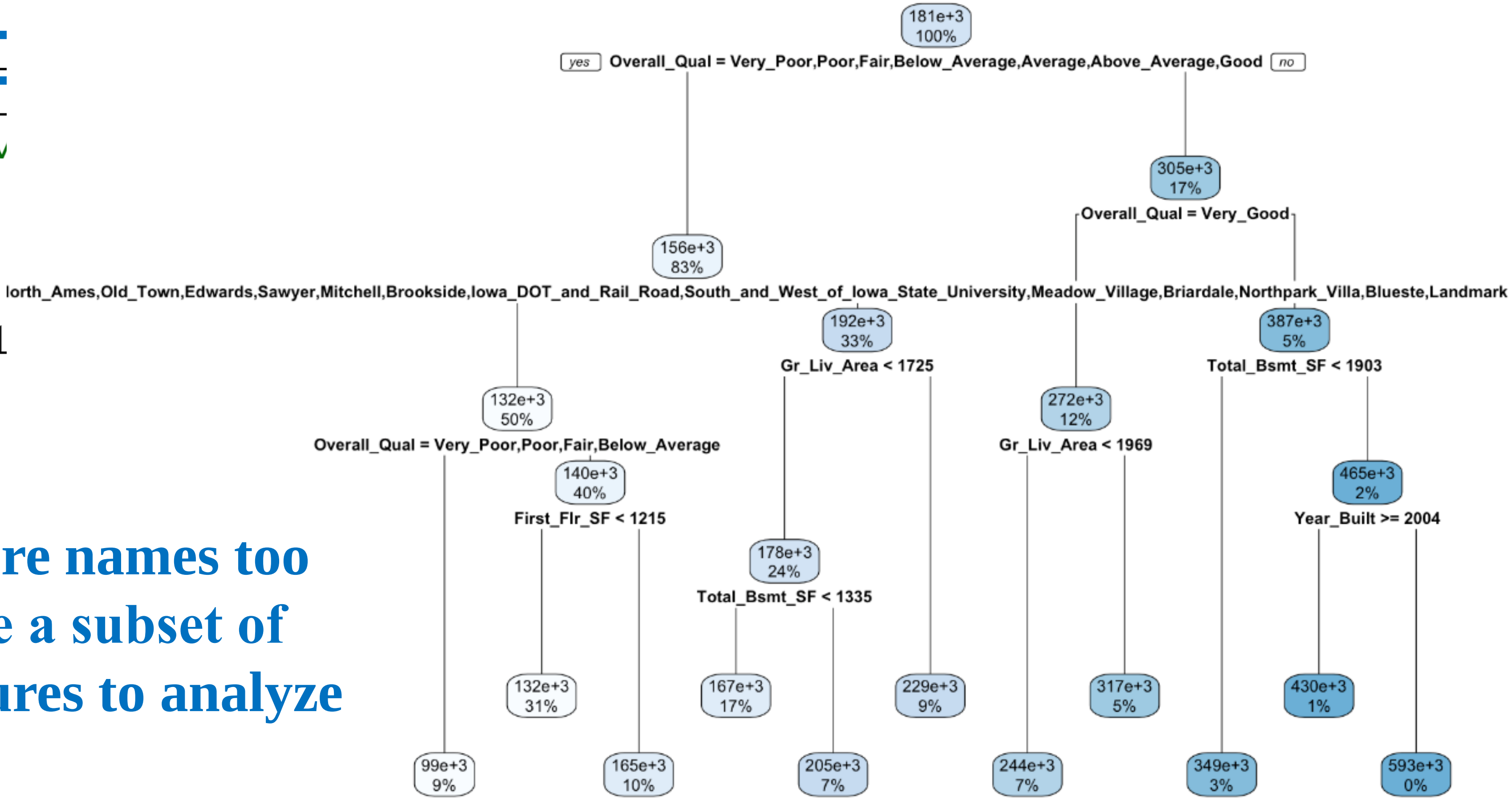
- **Even if this is NOT specified, rpart() will make an intelligent guess on whether it is a regression / classification problem**

Ames Housing Example

Grow a tree (simple way) & Visualize, using rpart package

```
## Fit a regression
ames_dt1 <- rpart(
    formula = Sale_
    data    = ames_
    method  = "anov
)


rpart.plot(ames_dt1
```



Categorical feature names too long, let's just use a subset of the numeric features to analyze

Ames Housing Example

Grow a tree (simple way) & Visualize, using rpart package

Demonstrations in R Studio

Ames Housing Example

Grow Tree using **Caret package**

```
# caret cross validation results
# Runtime: 30 seconds on i7 CPU
ames_dt3 <- train(
    Sale_Price ~ .,
    data = ames_train,
    method = "rpart",
    trControl = trainControl(method = "cv", number = 10),
    tuneLength = 20
)
```

Ames Housing Example

Feature Importance Measure: Impurity

- based on total reduction in SSE in CART

- Scale (of feature importance) standardize to 0-100

```
vip(ames_dt3, num_features = 40, scale = TRUE)
```

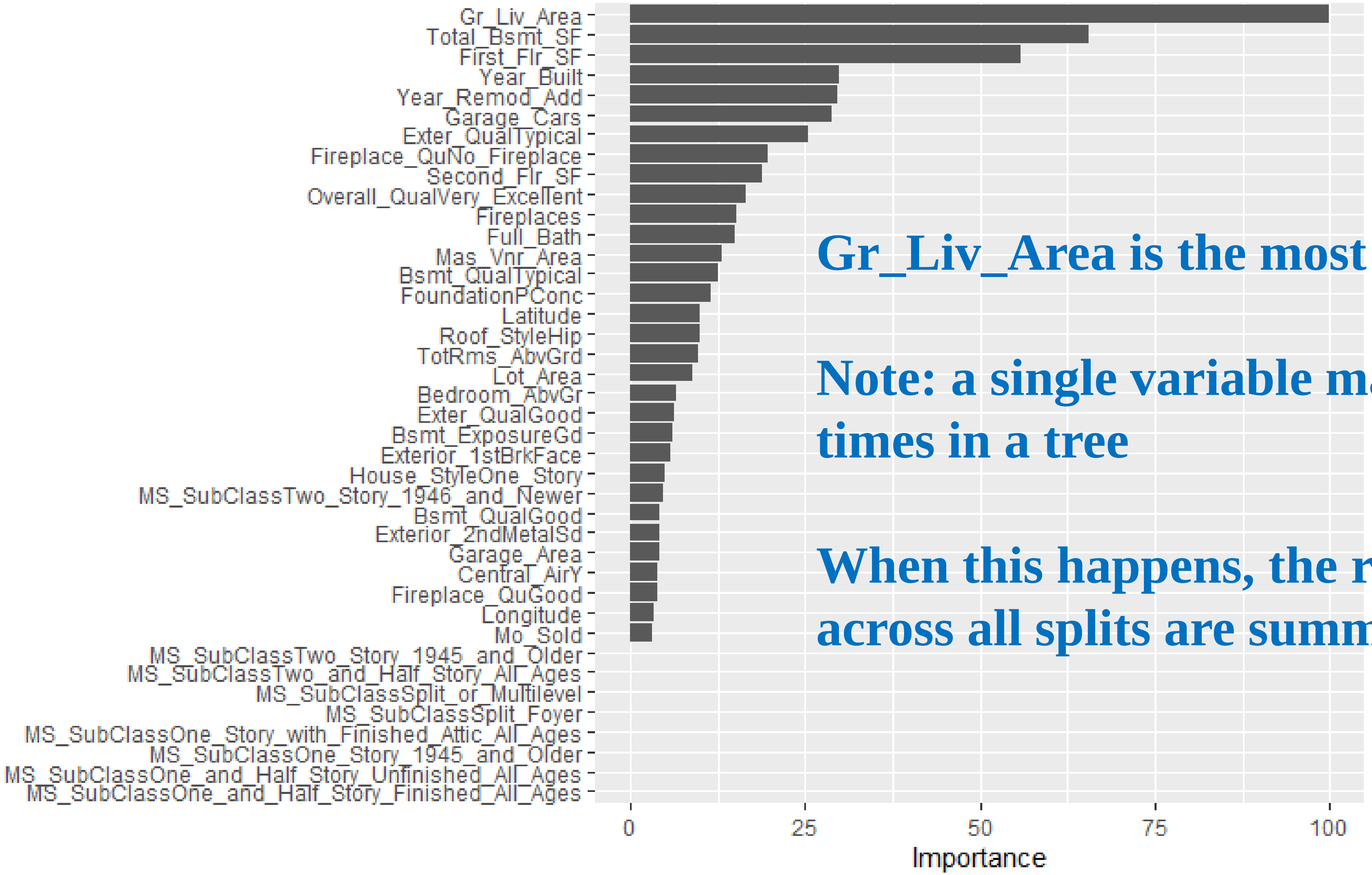
plot vip (from vip package)

- **Syntax: vip(model, num_features)**
- **num_features default = 10, here we use 40**
- **method = “model” (default), impurity-based var. importance (specific to tree-based models)**
- **will see other option (e.g. method = “permute”) later**

Ames Housing Example

Feature Importance Measure: Impurity

- based on total reduction in SSE in CART



Gr_Liv_Area is the most important feature

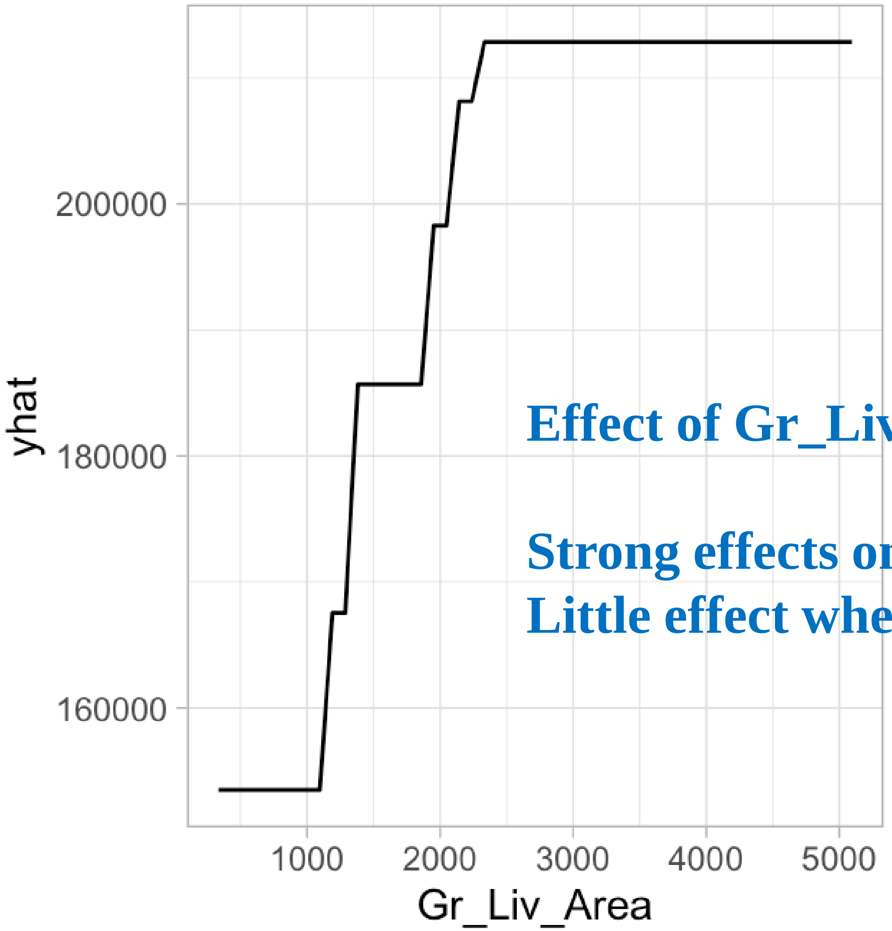
Note: a single variable may be used multi-times in a tree

When this happens, the reductions in MSE across all splits are summed up

Ames Housing Example

Partial Dependence Plot

```
partial(ames_dt3, pred.var = "Gr_Liv_Area") %>% autoplot()
```



Effect of Gr_Liv_Area is nonlinear

Strong effects on sales price when between 1000-2500
Little effect when it exceeds 2500