Master studije
GEOINFORMATIKA

# Mašinsko učenje - Stabla odlučivanja. Nasumična šuma.

Tatjana Jakšić Krüger

tatjana@turing.mi.sanu.ac.rs

16. decembar 2021.

Građevinski fakultet Univerziteta u Beogradu

# Podsetnik

- Linearni klasifikatori.
- Merenje gubitka.
- Učenje klasifikatora.
- Perceptron.
- Kvalitet klasifikatora,
  - Linearna razvojivost skupa za obučavanje.
  - *margin of the labelled point*, *margin of the training set*.
- Ukoliko skup za obučavanje nije linearno razdvojiv koristimo logističku regresiju tj. linearnu logističku klasifikaciju.
- Nismo pokrili *Linear Discriminant Analysis* (LDA).

# Cilj za danas

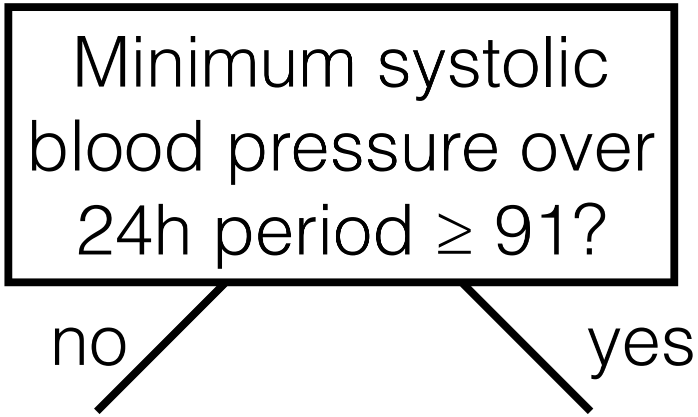
- Stabla odlučivanja (eng. decision trees).
- Ansambli.
- Nasumična šuma (eng. random forest).

# Motivacija

- Jednostavni modeli mašinskog učenja za regresiju i klasifikaciju.
- Simpsonov paradoks.
- Motivacija:
  - Intepretacija modela.
  - Smanjivanje ljudske greške.

# Decision tree

# Decision tree

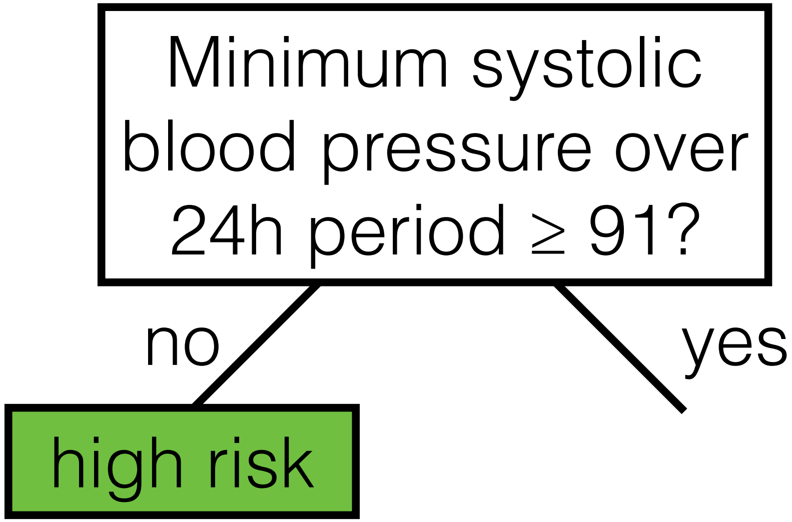

Minimum systolic blood pressure over 24h period ≥ 91?

no

yes

[Breiman, Friedman, Olshen, Stone, 1984]

# Decision tree



Minimum systolic blood pressure over 24h period ≥ 91?

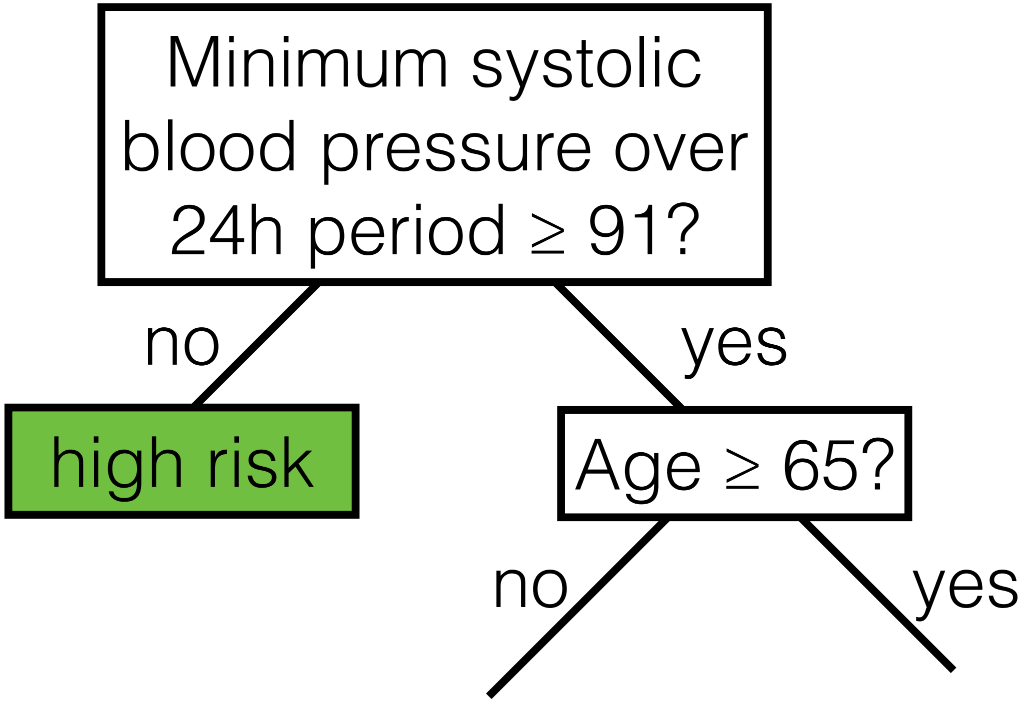
no

yes

high risk

[Breiman, Friedman, Olshen, Stone, 1984]

# Decision tree



Minimum systolic blood pressure over 24h period ≥ 91?

no

yes

high risk

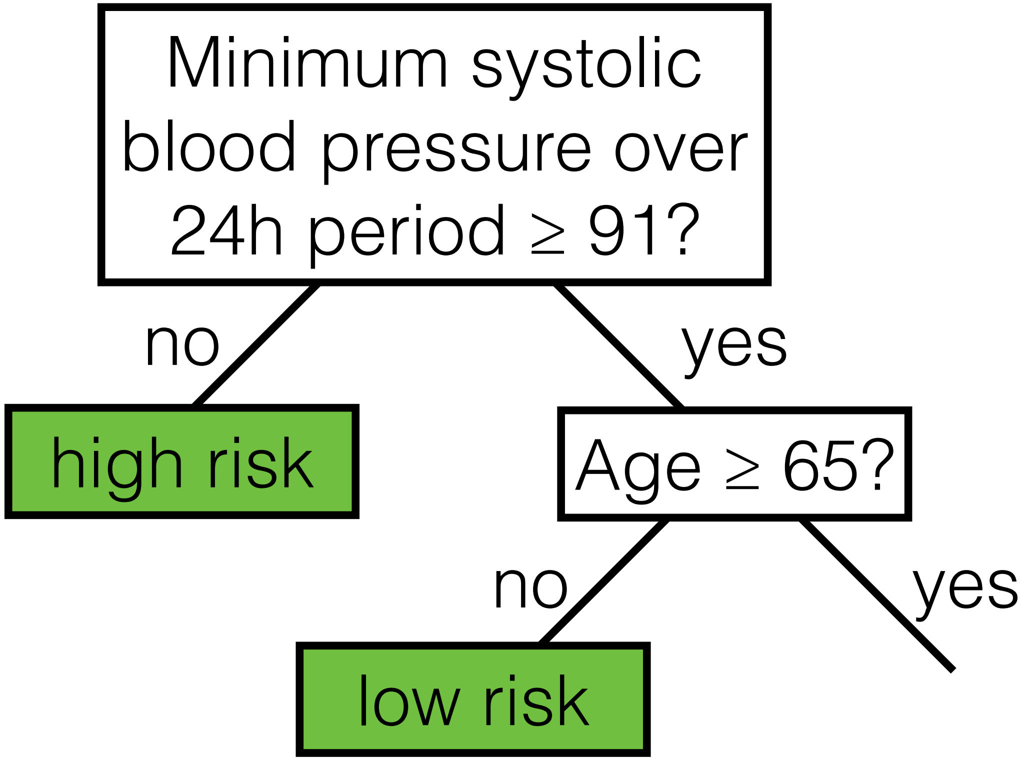
Age ≥ 65?

no

yes

[Breiman, Friedman, Olshen, Stone, 1984]

# Decision tree



Minimum systolic blood pressure over 24h period ≥ 91?

no

yes

high risk

Age ≥ 65?

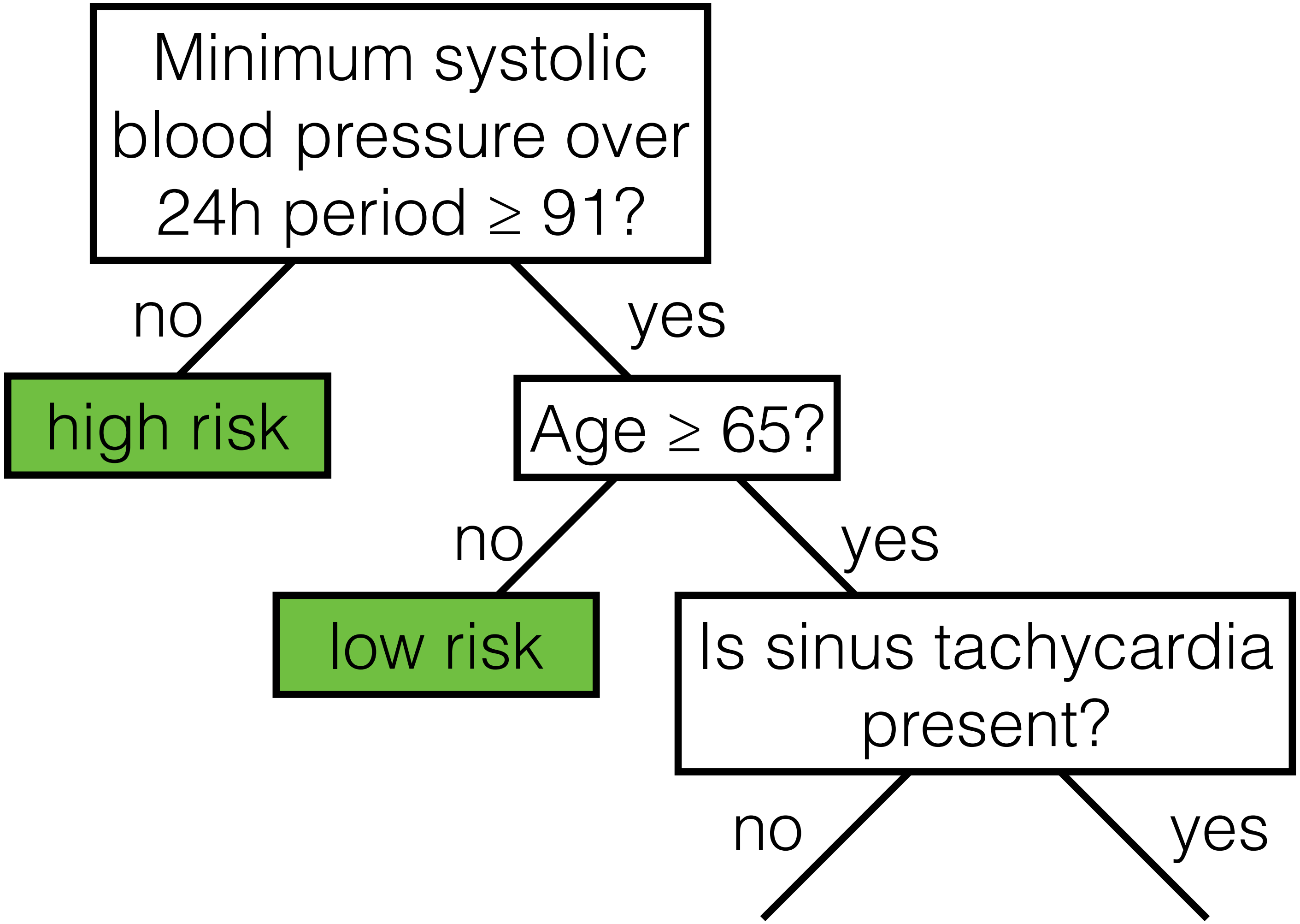
no

yes

low risk

[Breiman, Friedman, Olshen, Stone, 1984]

# Decision tree



Minimum systolic blood pressure over 24h period ≥ 91?

no

high risk

yes

Age ≥ 65?

no

low risk

yes

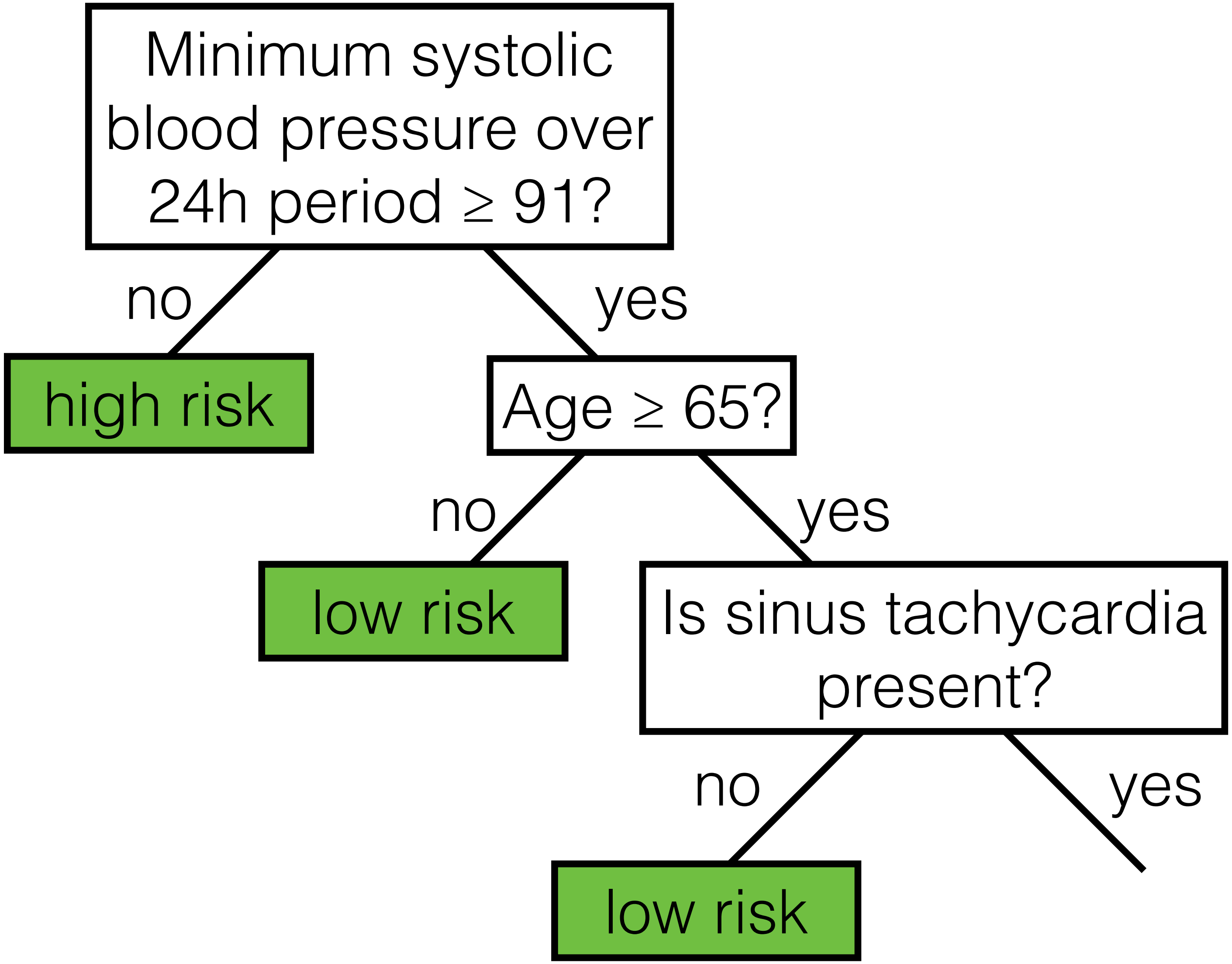
Is sinus tachycardia present?

no

yes

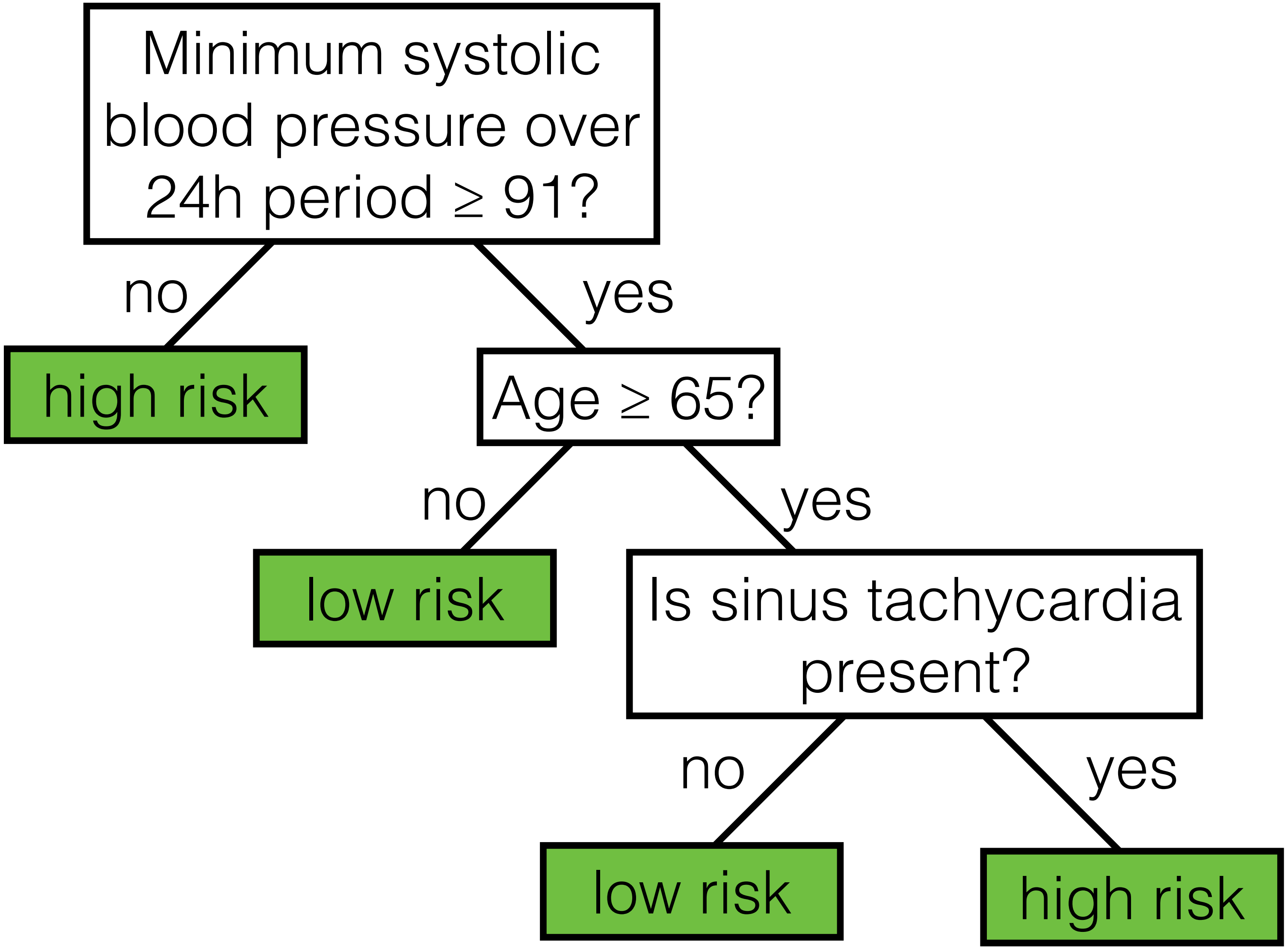
[Breiman, Friedman, Olshen, Stone, 1984]

# Decision tree



[Breiman, Friedman, Olshen, Stone, 1984]

# Decision tree



Minimum systolic blood pressure over 24h period ≥ 91?

no

high risk

yes

Age ≥ 65?

no

low risk

yes

Is sinus tachycardia present?

no

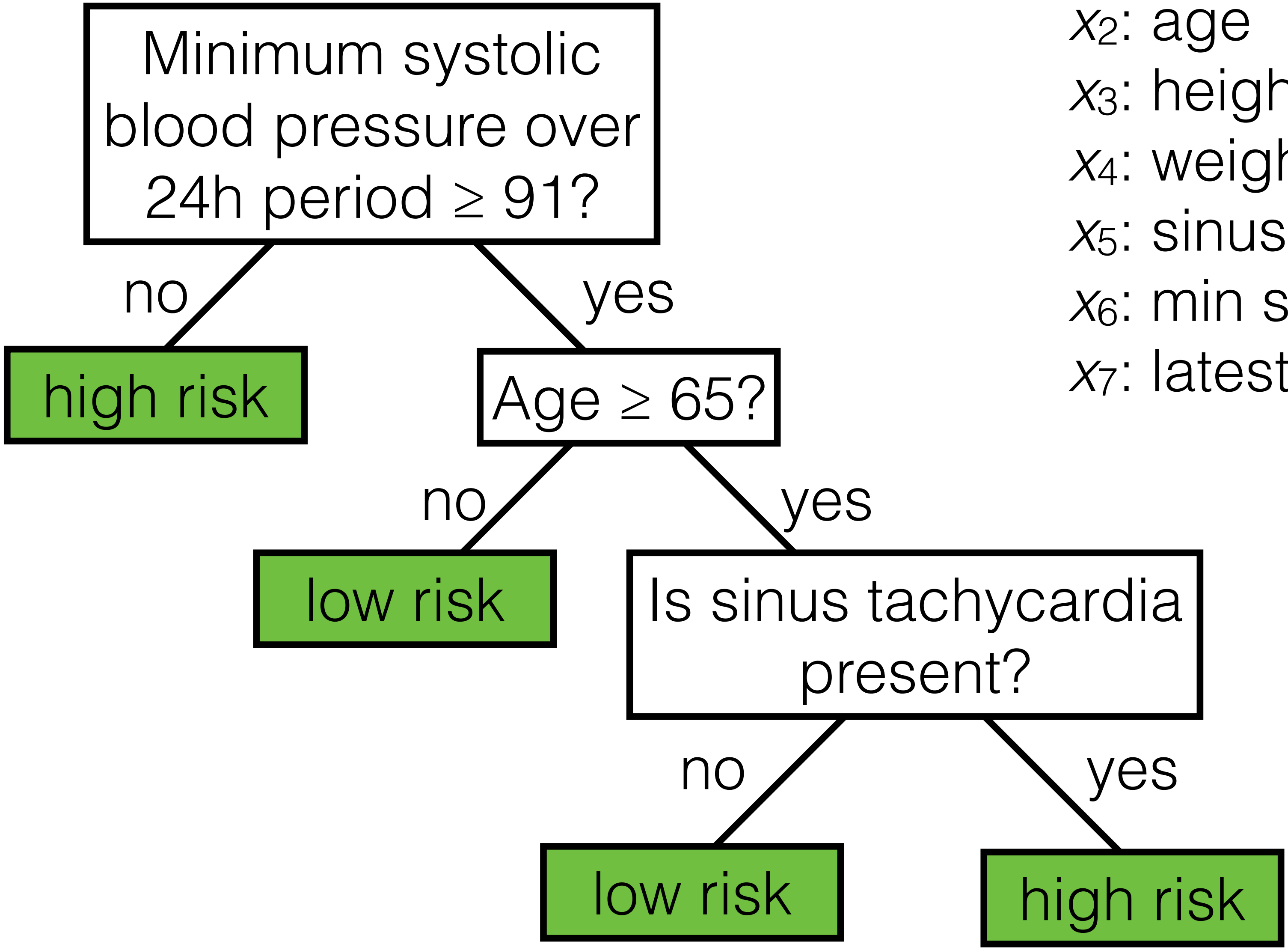
low risk

yes

high risk

[Breiman, Friedman, Olshen, Stone, 1984]

# Decision tree



features:

$x_1$: date

$x_2$: age

$x_3$: height

$x_4$: weight

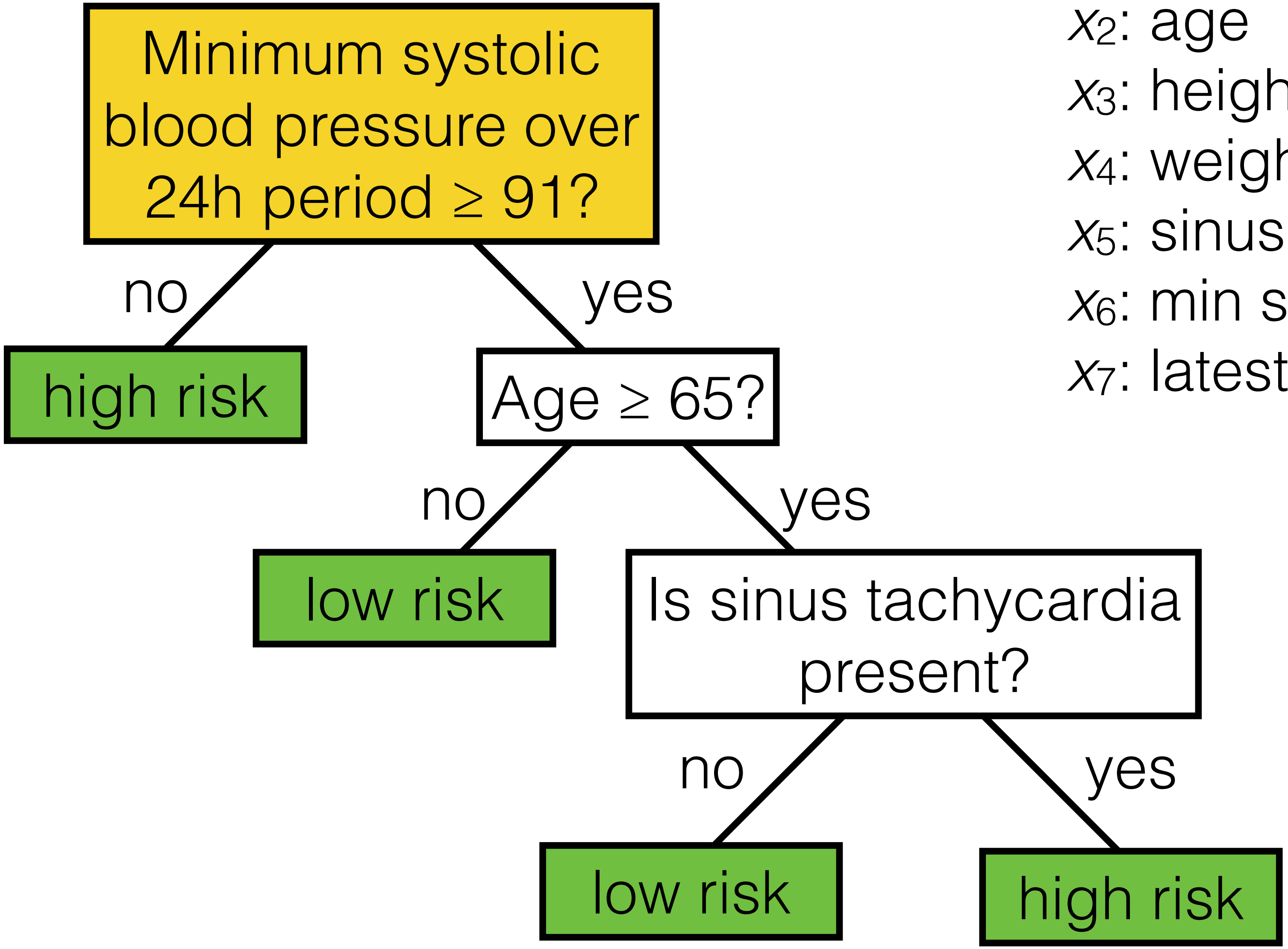
$x_5$: sinus tachycardia?

$x_6$: min systolic bp, 24h

$x_7$: latest diastolic bp

[Breiman, Friedman, Olshen, Stone, 1984]

# Decision tree



features:

$x_1$: date

$x_2$: age

$x_3$: height

$x_4$: weight

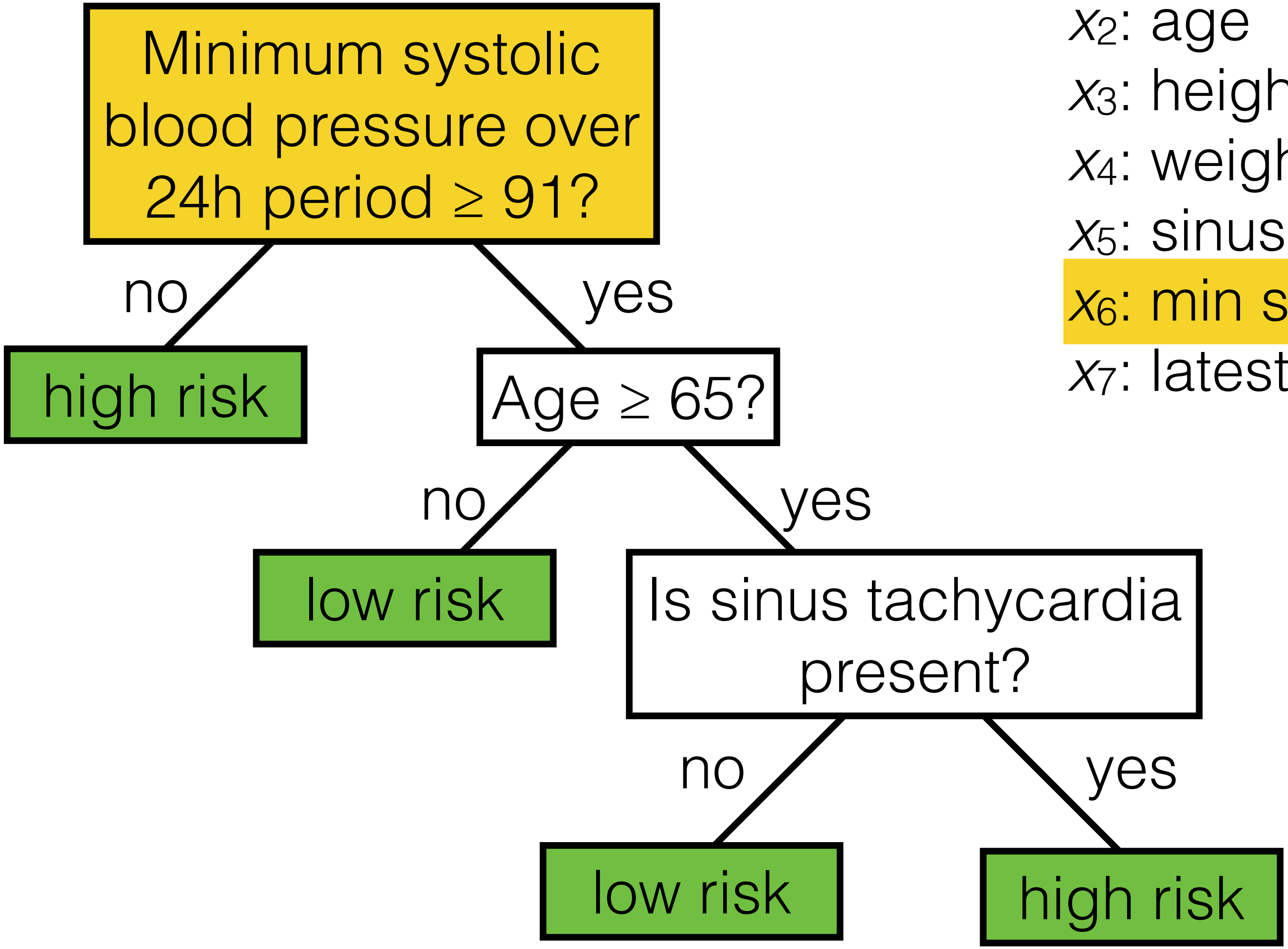
$x_5$: sinus tachycardia?

$x_6$: min systolic bp, 24h

$x_7$: latest diastolic bp

[Breiman, Friedman, Olshen, Stone, 1984]

# Decision tree



features:

$x_1$: date

$x_2$: age

$x_3$: height

$x_4$: weight

$x_5$: sinus tachycardia?

$x_6$: min systolic bp, 24h

$x_7$: latest diastolic bp

[Breiman, Friedman, Olshen, Stone, 1984]

# Decision tree

features:

$x_1$: date

$x_2$: age

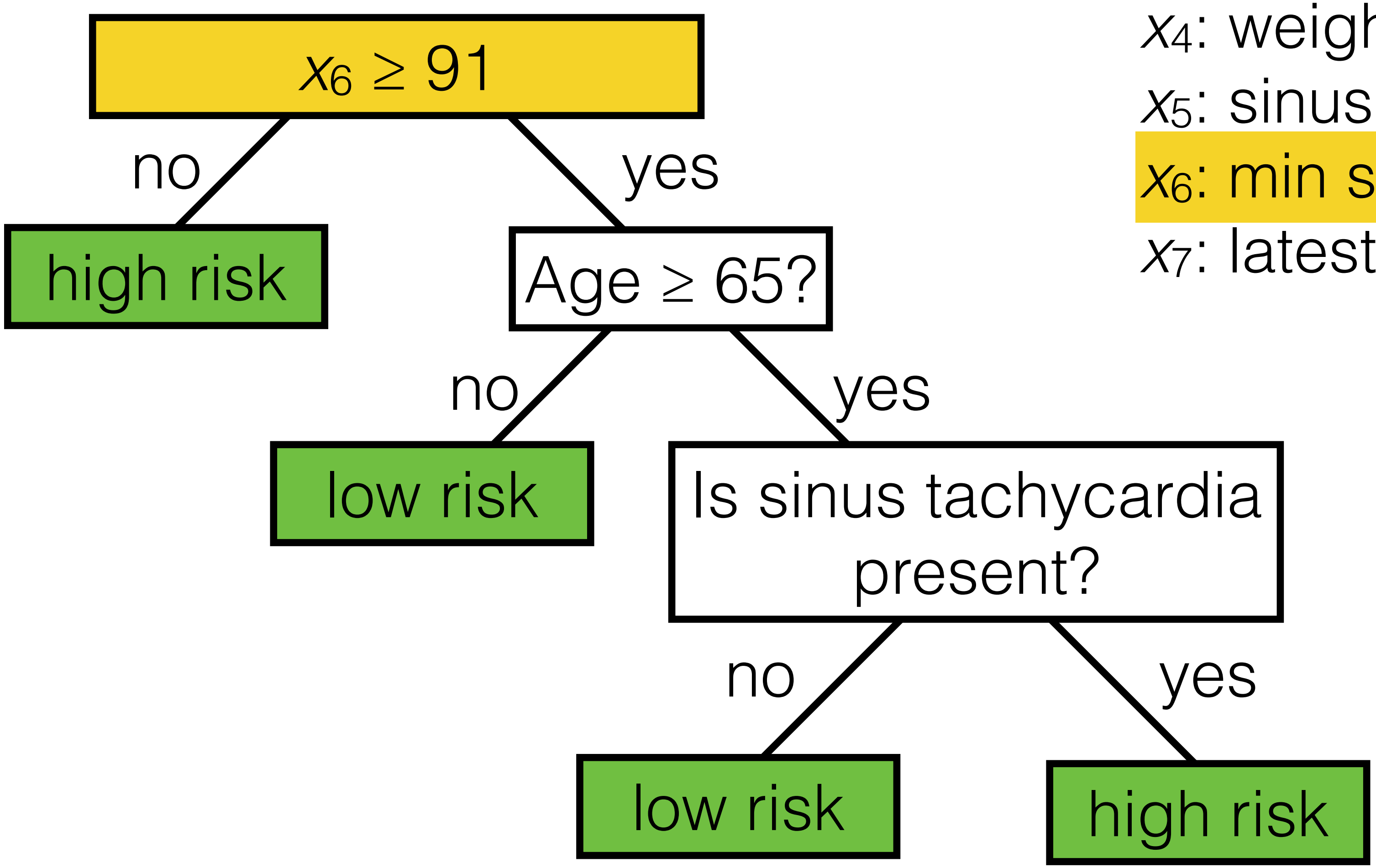
$x_3$: height

$x_4$: weight

$x_5$: sinus tachycardia?

$x_6$: min systolic bp, 24h

$x_7$: latest diastolic bp



[Breiman, Friedman, Olshen, Stone, 1984]

# Decision tree

features:

$x_1$: date

$x_2$: age

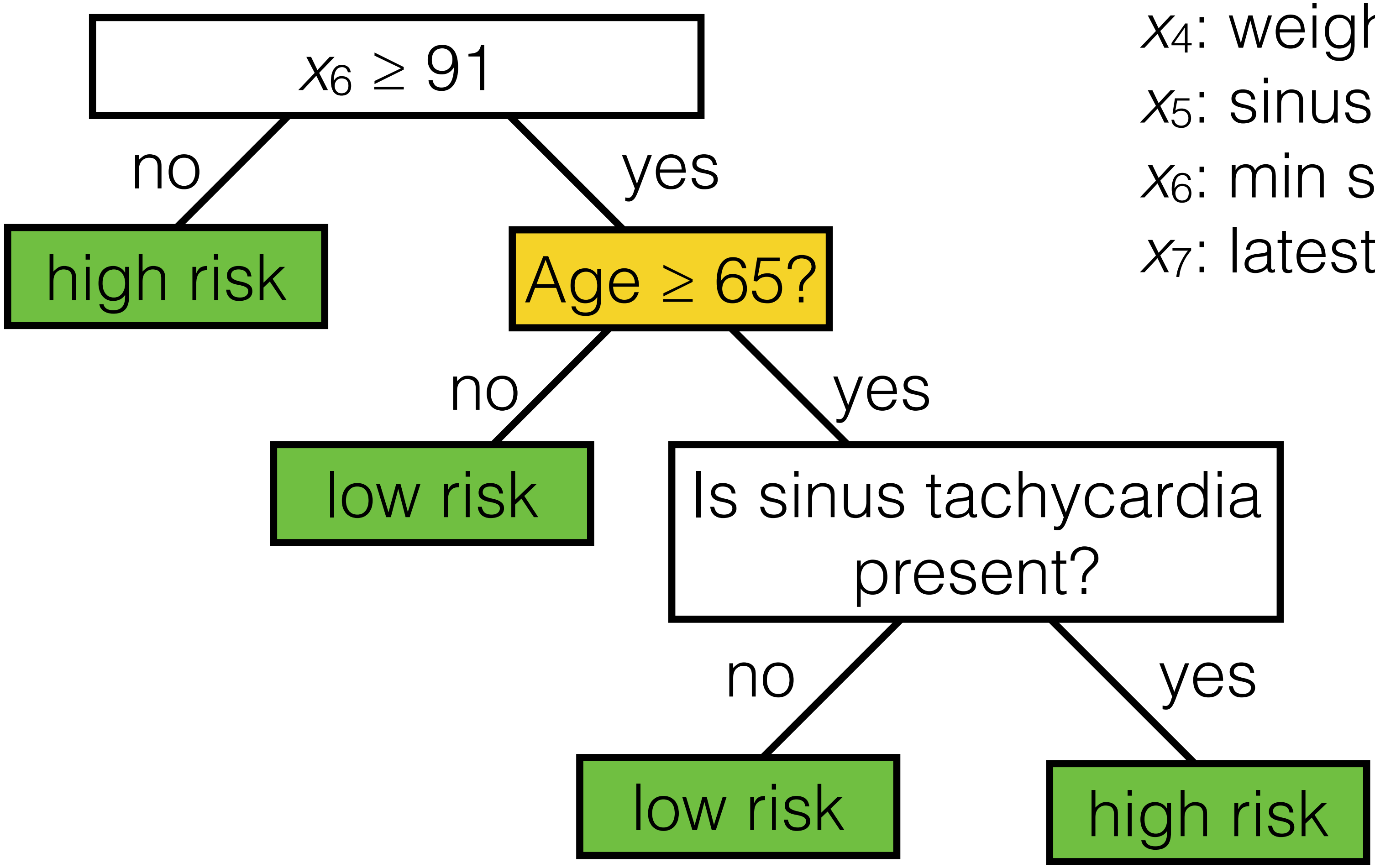
$x_3$: height

$x_4$: weight

$x_5$: sinus tachycardia?

$x_6$: min systolic bp, 24h

$x_7$: latest diastolic bp



$x_6 \geq 91$

no

yes

high risk

Age ≥ 65?

no

yes

low risk

Is sinus tachycardia present?

no

yes

low risk

high risk

[Breiman, Friedman, Olshen, Stone, 1984]

# Decision tree

features:

$x_1$: date

$x_2$: age

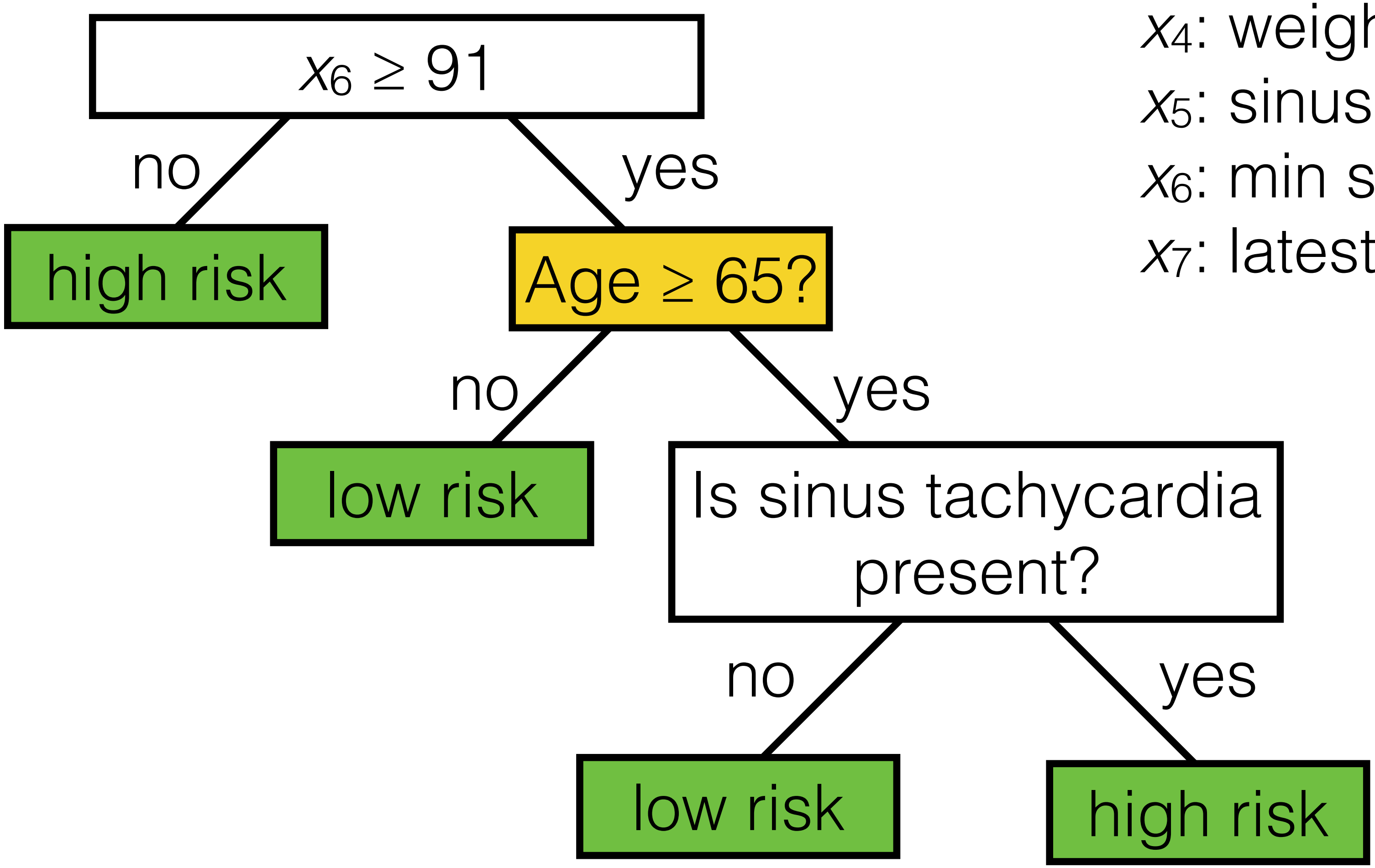
$x_3$: height

$x_4$: weight

$x_5$: sinus tachycardia?

$x_6$: min systolic bp, 24h

$x_7$: latest diastolic bp



[Breiman, Friedman, Olshen, Stone, 1984]

# Decision tree

features:

$x_1$: date

$x_2$: age

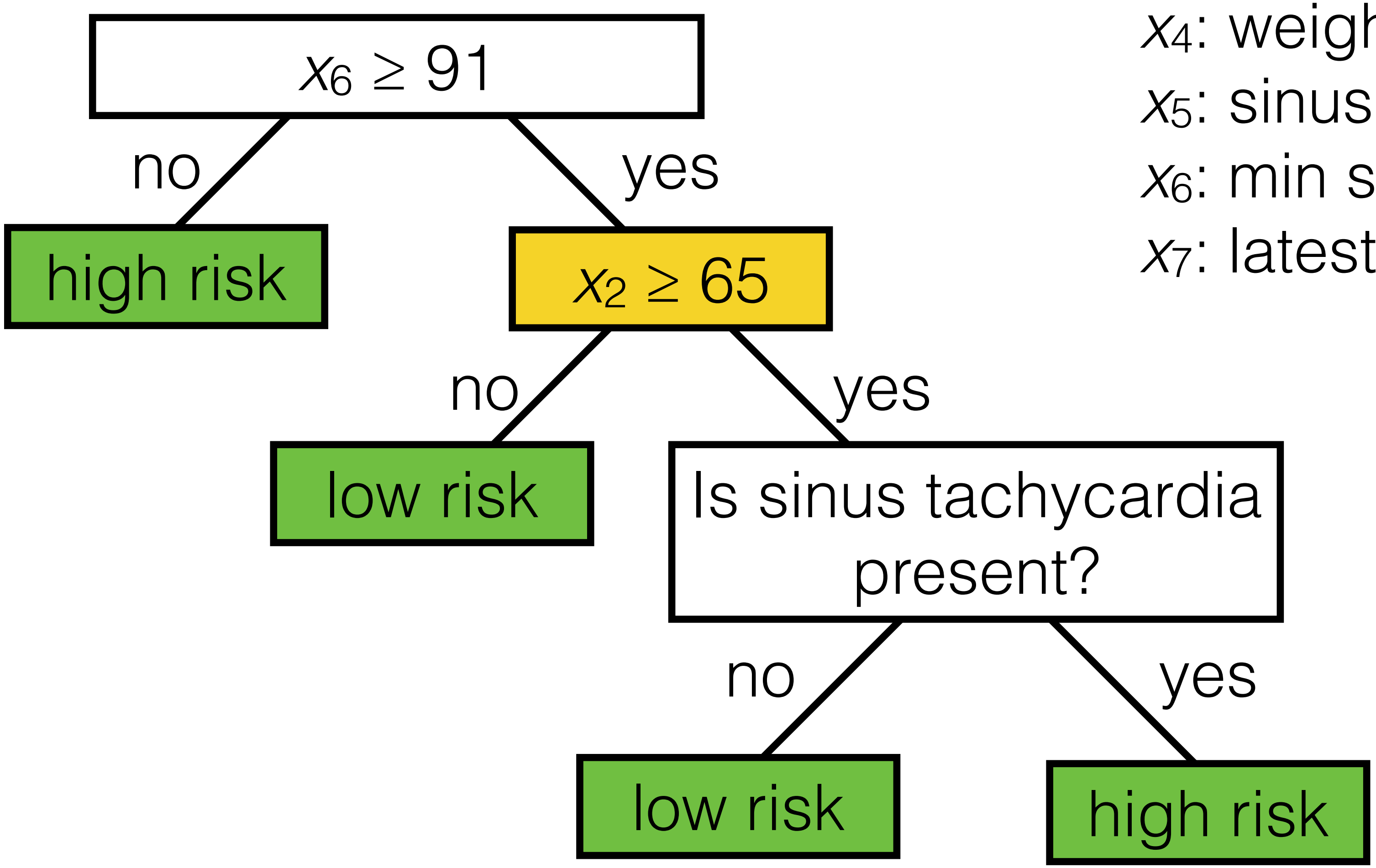
$x_3$: height

$x_4$: weight

$x_5$: sinus tachycardia?

$x_6$: min systolic bp, 24h

$x_7$: latest diastolic bp



[Breiman, Friedman, Olshen, Stone, 1984]

# Decision tree

features:

$x_1$: date

$x_2$: age

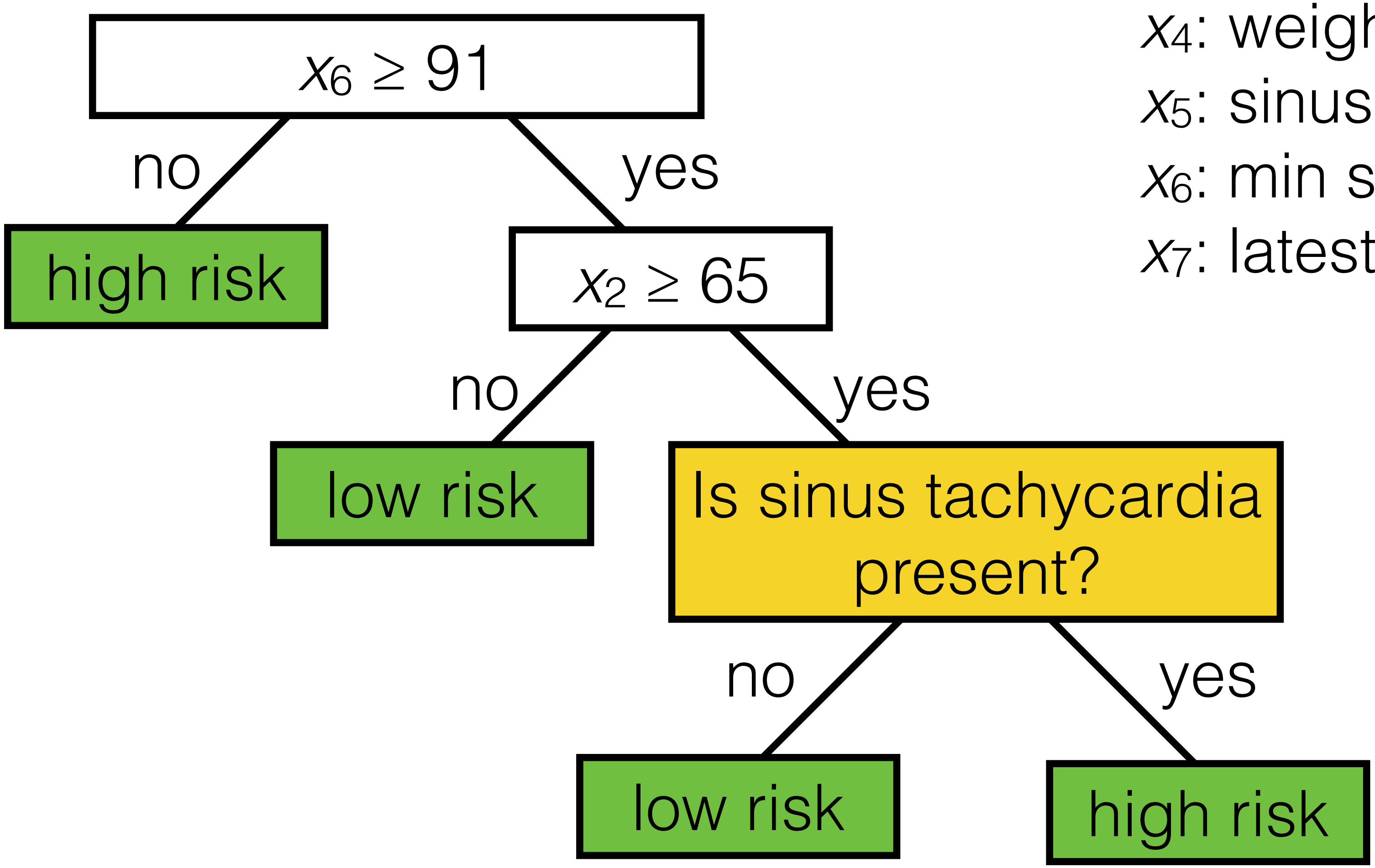
$x_3$: height

$x_4$: weight

$x_5$: sinus tachycardia?

$x_6$: min systolic bp, 24h

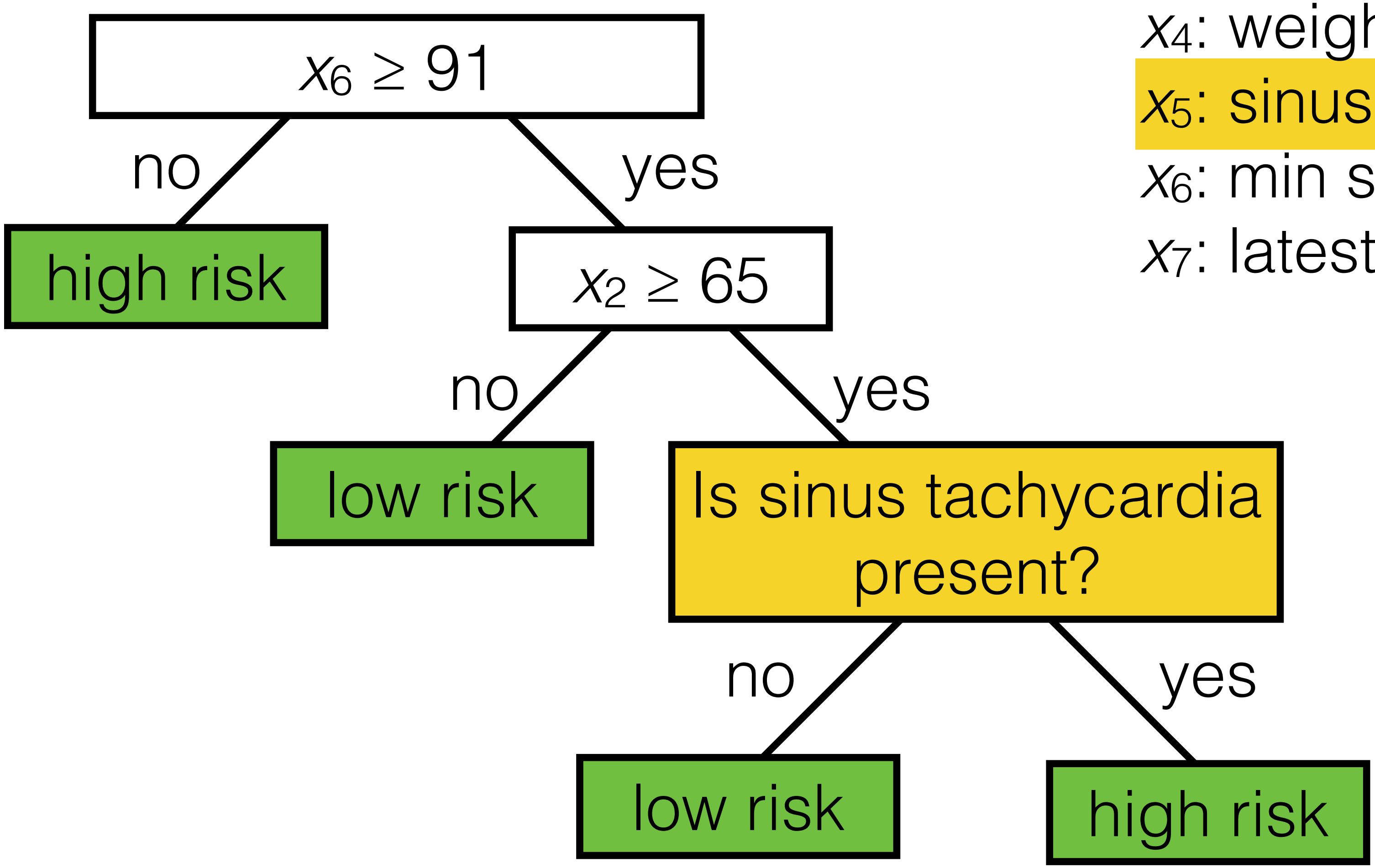
$x_7$: latest diastolic bp



[Breiman, Friedman, Olshen, Stone, 1984]

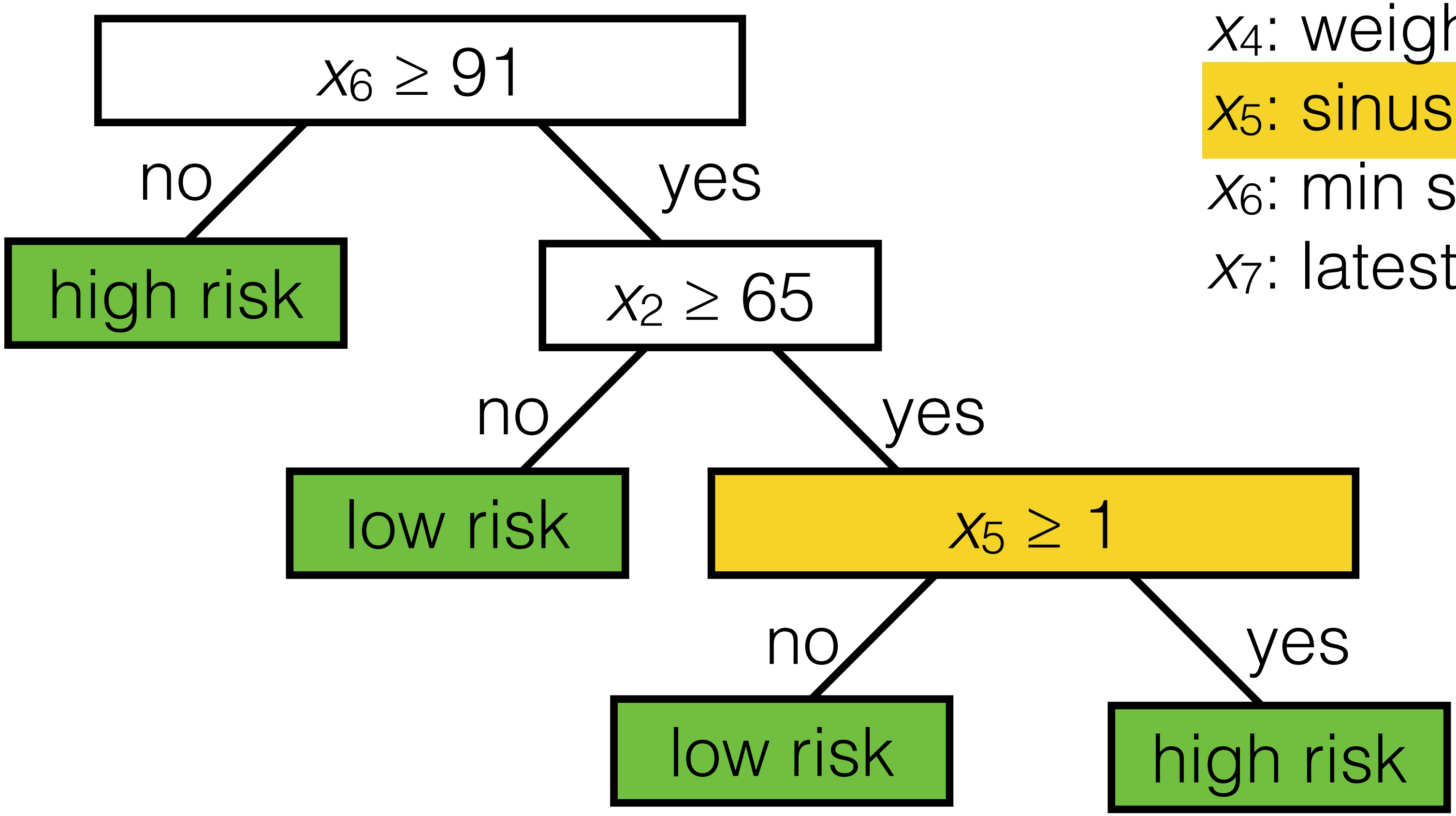
# Decision tree

features:
$x_1$: date
$x_2$: age
$x_3$: height
$x_4$: weight
$x_5$: sinus tachycardia?
$x_6$: min systolic bp, 24h
$x_7$: latest diastolic bp



[Breiman, Friedman, Olshen, Stone, 1984]

# Decision tree



$x_6 \geq 91$

no

high risk

yes

$x_2 \geq 65$

no

low risk

yes

$x_5 \geq 1$

no

low risk

yes

high risk

features:

$x_1$: date

$x_2$: age

$x_3$: height

$x_4$: weight

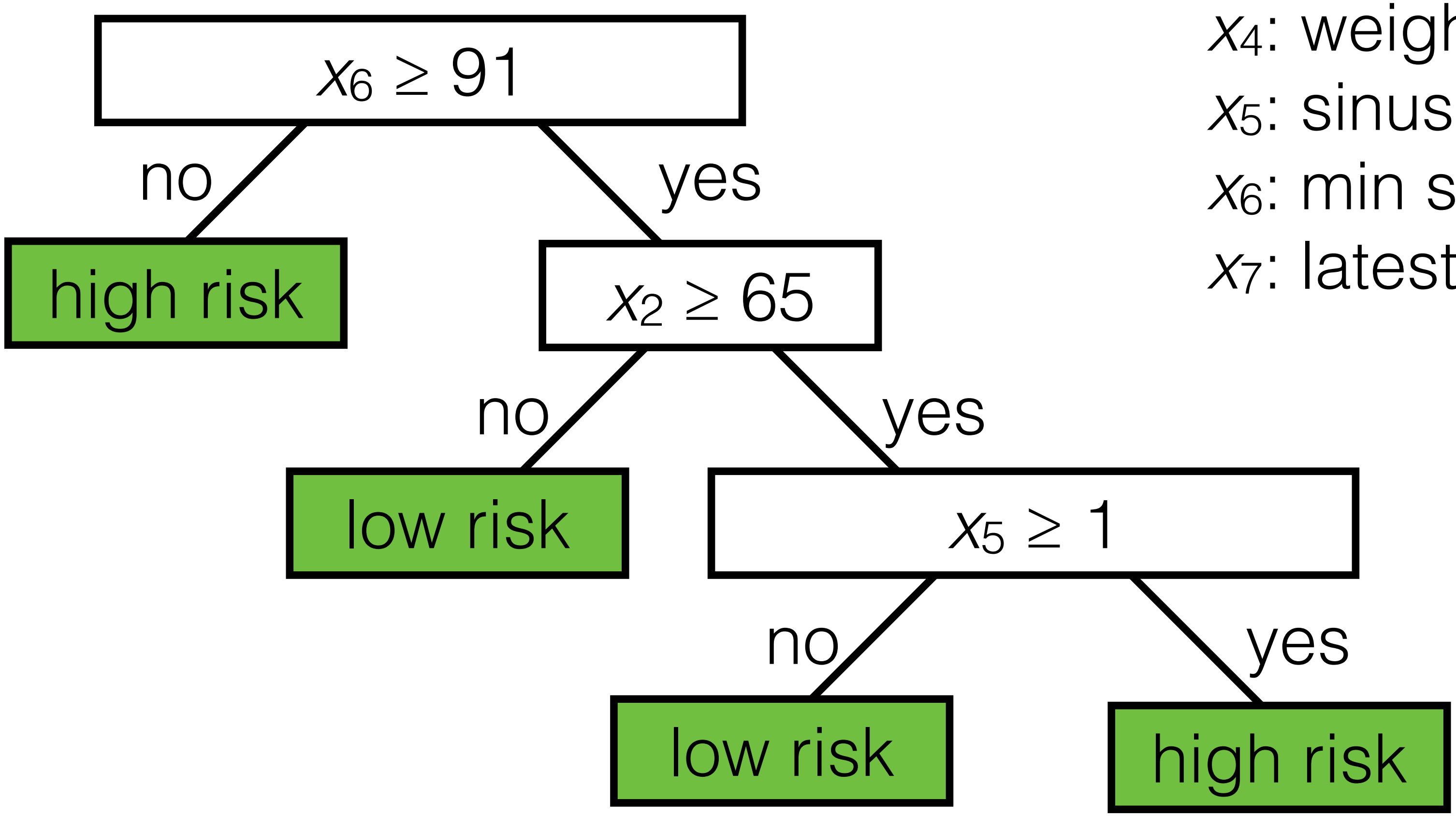
$x_5$: sinus tachycardia?

$x_6$: min systolic bp, 24h

$x_7$: latest diastolic bp

[Breiman, Friedman, Olshen, Stone, 1984]

# Decision tree

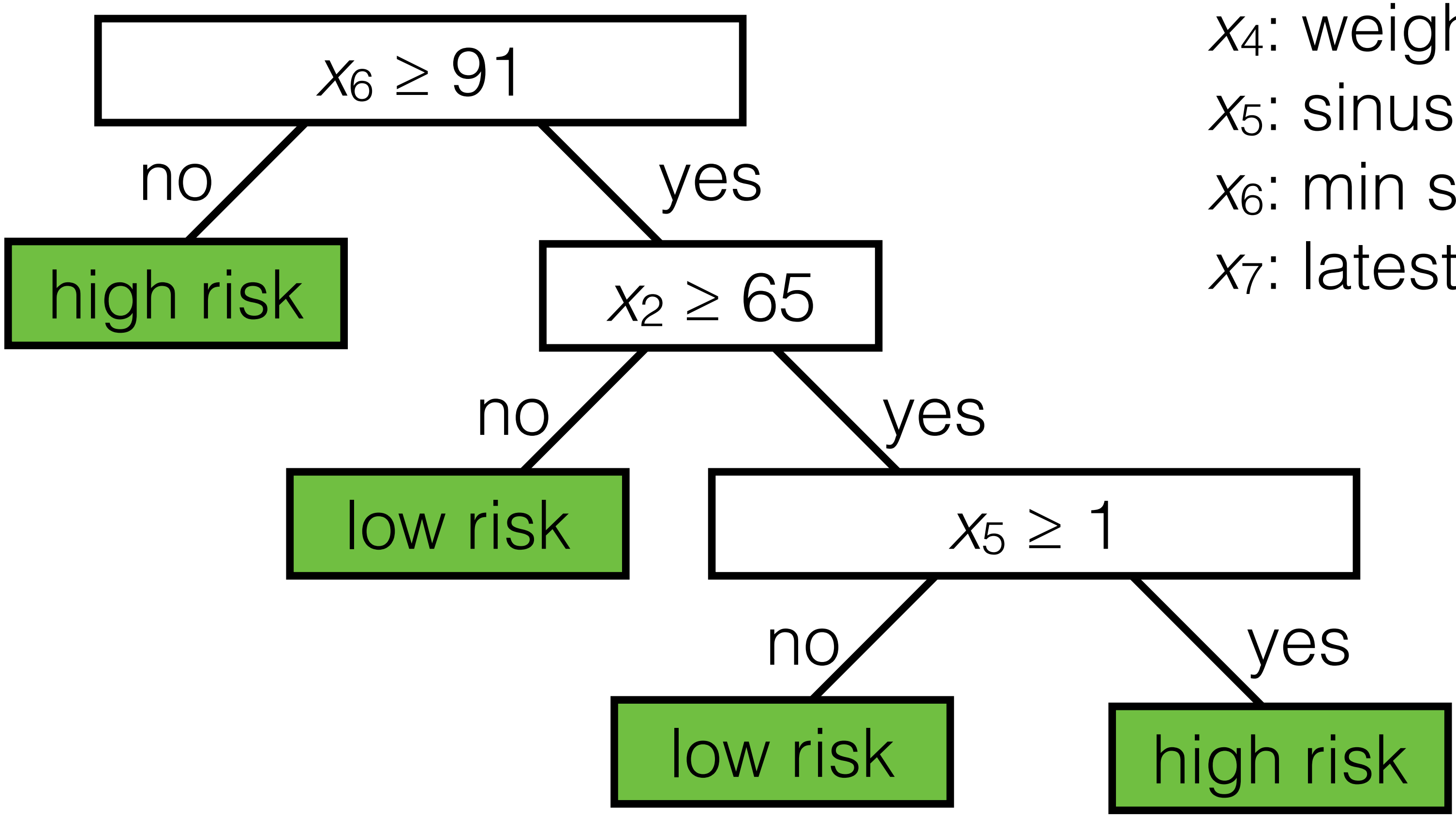


features:
$x_1$: date
$x_2$: age
$x_3$: height
$x_4$: weight
$x_5$: sinus tachycardia?
$x_6$: min systolic bp, 24h
$x_7$: latest diastolic bp

[Breiman, Friedman, Olshen, Stone, 1984]

# Decision tree

features:
$x_1$: date
$x_2$: age
$x_3$: height
$x_4$: weight
$x_5$: sinus tachycardia?
$x_6$: min systolic bp, 24h
$x_7$: latest diastolic bp



$x_6 \geq 91$

no

high risk

yes

$x_2 \geq 65$

no

low risk

yes

$x_5 \geq 1$

no

low risk

yes

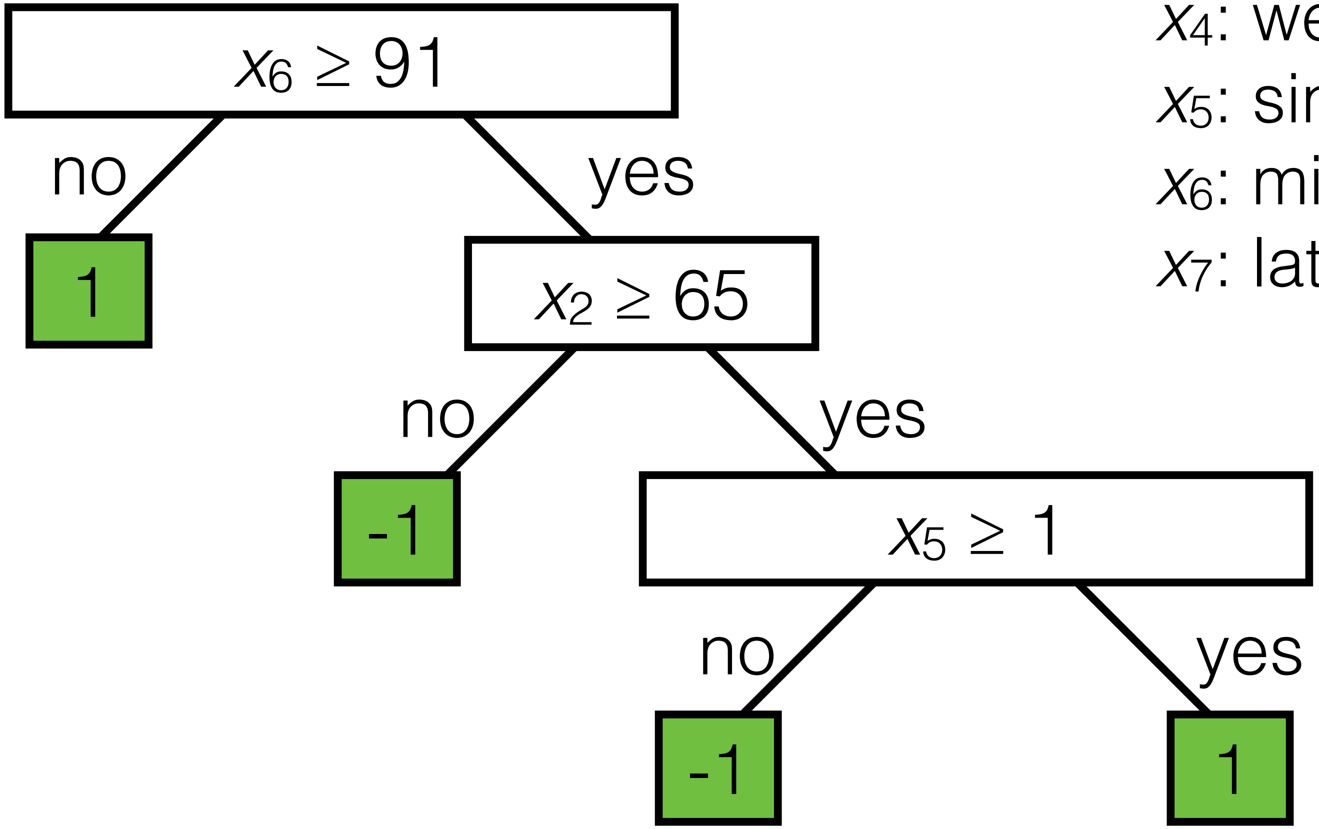
high risk

labels $y$:
1: high risk
-1: low risk

[Breiman, Friedman, Olshen, Stone, 1984]

# Decision tree

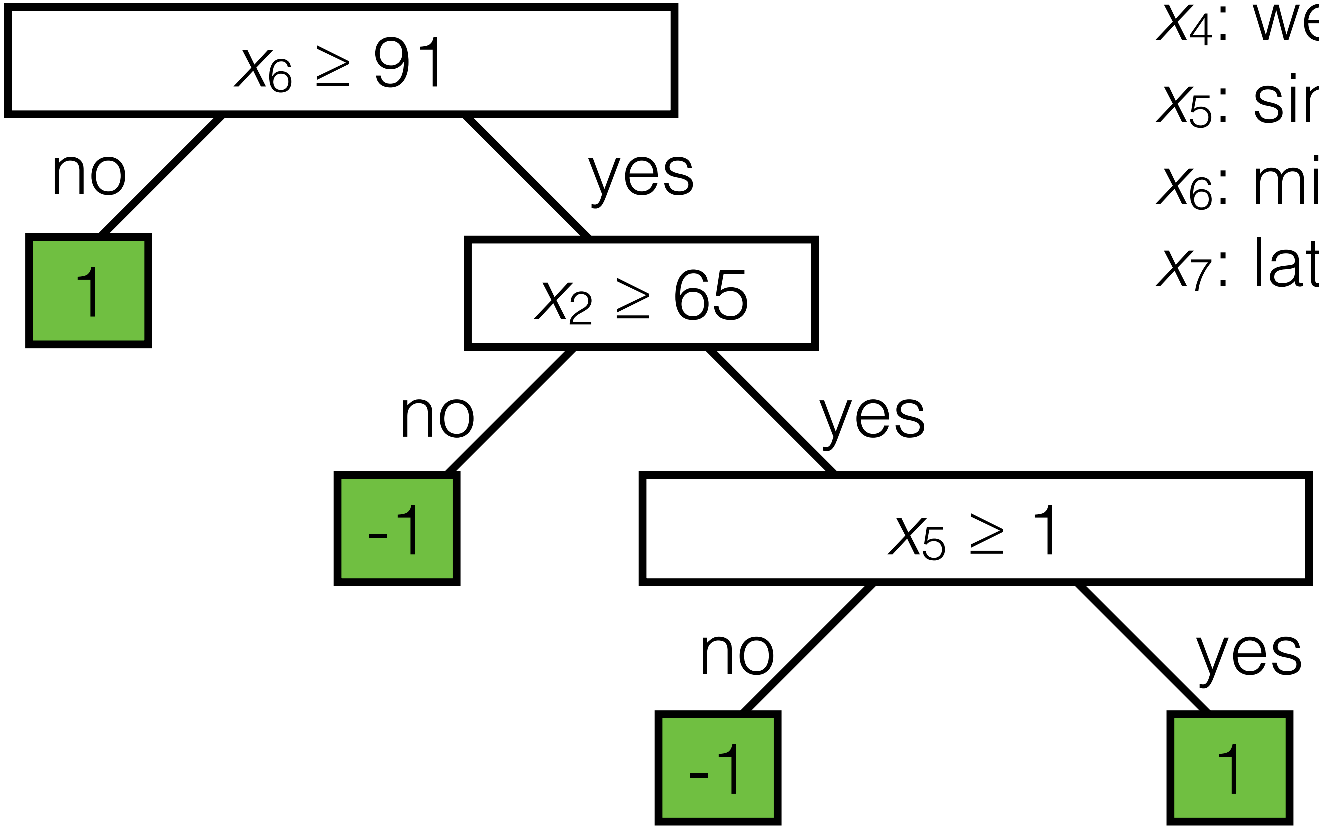
features:
$x_1$: date
$x_2$: age
$x_3$: height
$x_4$: weight
$x_5$: sinus tachycardia?
$x_6$: min systolic bp, 24h
$x_7$: latest diastolic bp



labels $y$:
1: high risk
-1: low risk

[Breiman, Friedman, Olshen, Stone, 1984]

# Decision tree



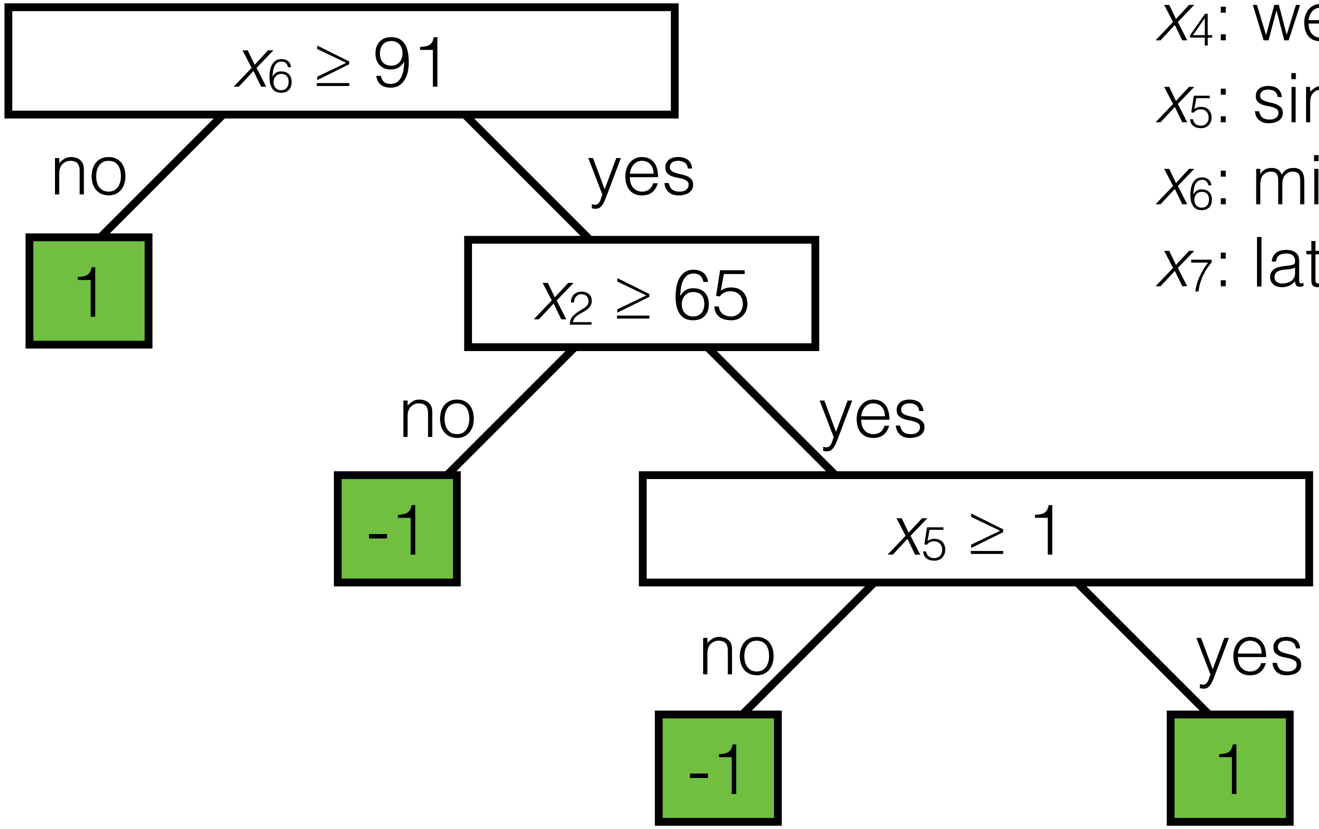
features:
$x_1$: date
$x_2$: age
$x_3$: height
$x_4$: weight
$x_5$: sinus tachycardia?
$x_6$: min systolic bp, 24h
$x_7$: latest diastolic bp

labels $y$:
1: high risk
-1: low risk

[Breiman, Friedman, Olshen, Stone, 1984]

# Decision tree

internal node:



$x_6 \geq 91$

no

yes

1

$x_2 \geq 65$

no

yes

-1

$x_5 \geq 1$

no

yes

-1

1

features:

$x_1$: date

$x_2$: age

$x_3$: height

$x_4$: weight

$x_5$: sinus tachycardia?

$x_6$: min systolic bp, 24h

$x_7$: latest diastolic bp

labels $y$:

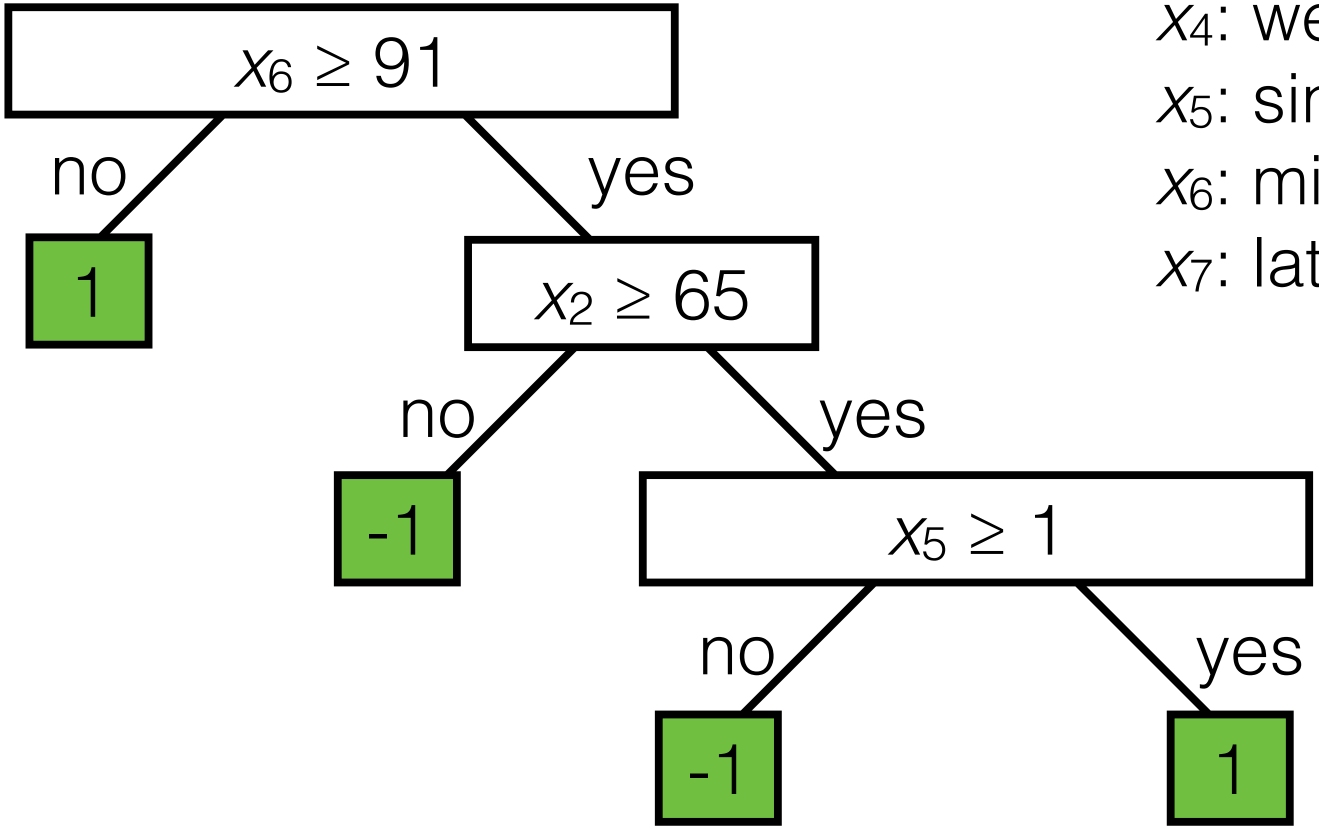
1: high risk

-1: low risk

# Decision tree

internal node:

- dimension index *j*; split value *s*



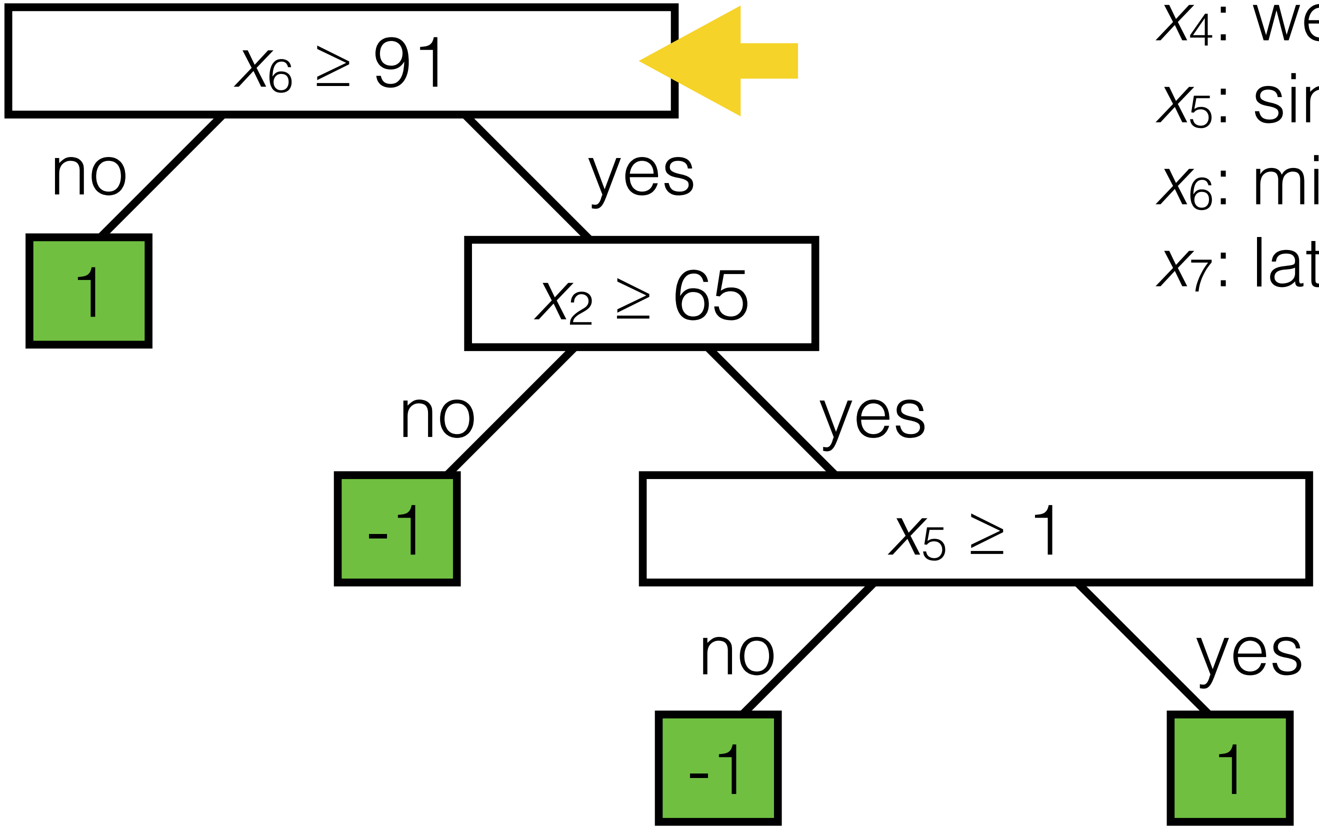
features:
$x_1$: date
$x_2$: age
$x_3$: height
$x_4$: weight
$x_5$: sinus tachycardia?
$x_6$: min systolic bp, 24h
$x_7$: latest diastolic bp

labels $y$:
1: high risk
-1: low risk

# Decision tree

internal node:

- dimension index *j*; split value *s*



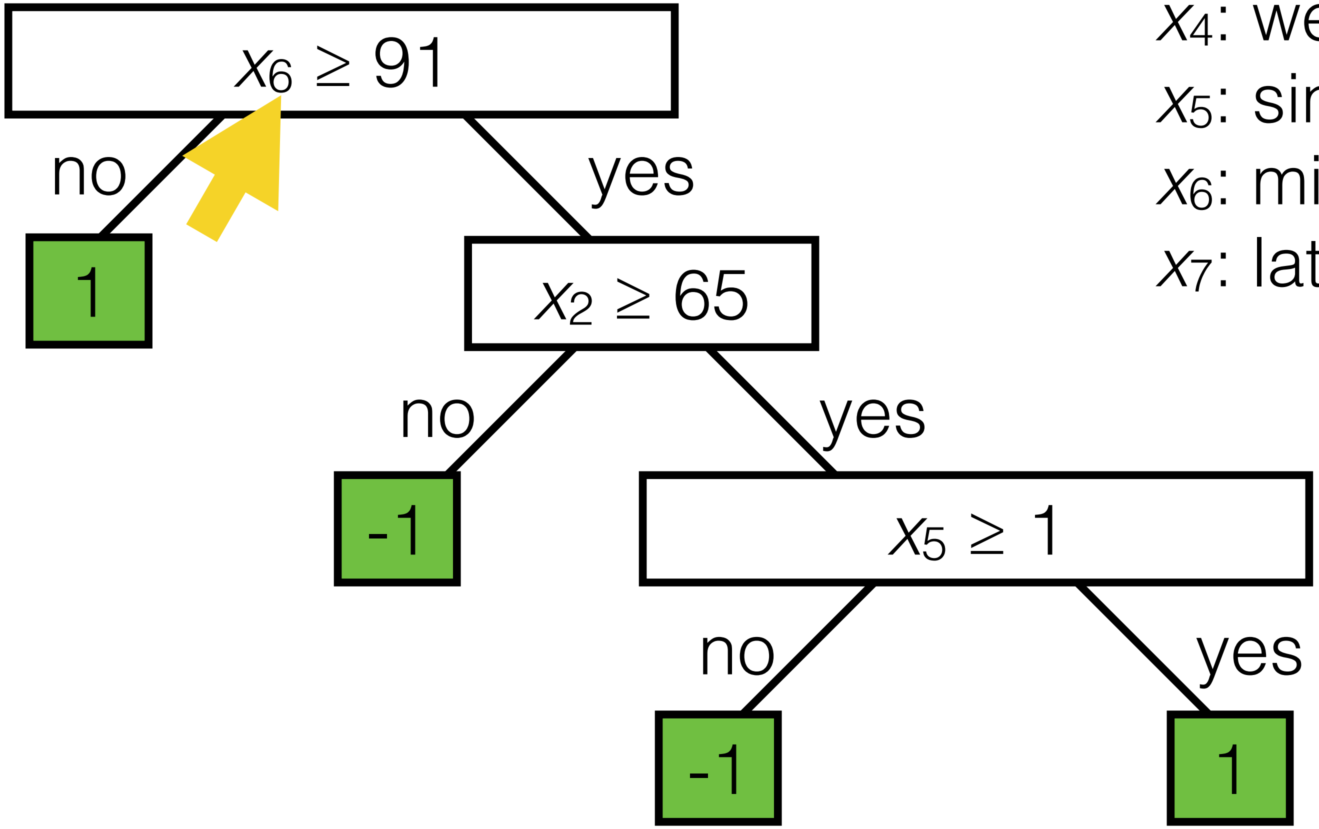
features:
$x_1$: date
$x_2$: age
$x_3$: height
$x_4$: weight
$x_5$: sinus tachycardia?
$x_6$: min systolic bp, 24h
$x_7$: latest diastolic bp

labels $y$:
1: high risk
-1: low risk

# Decision tree

internal node:

- dimension index *j*; split value *s*



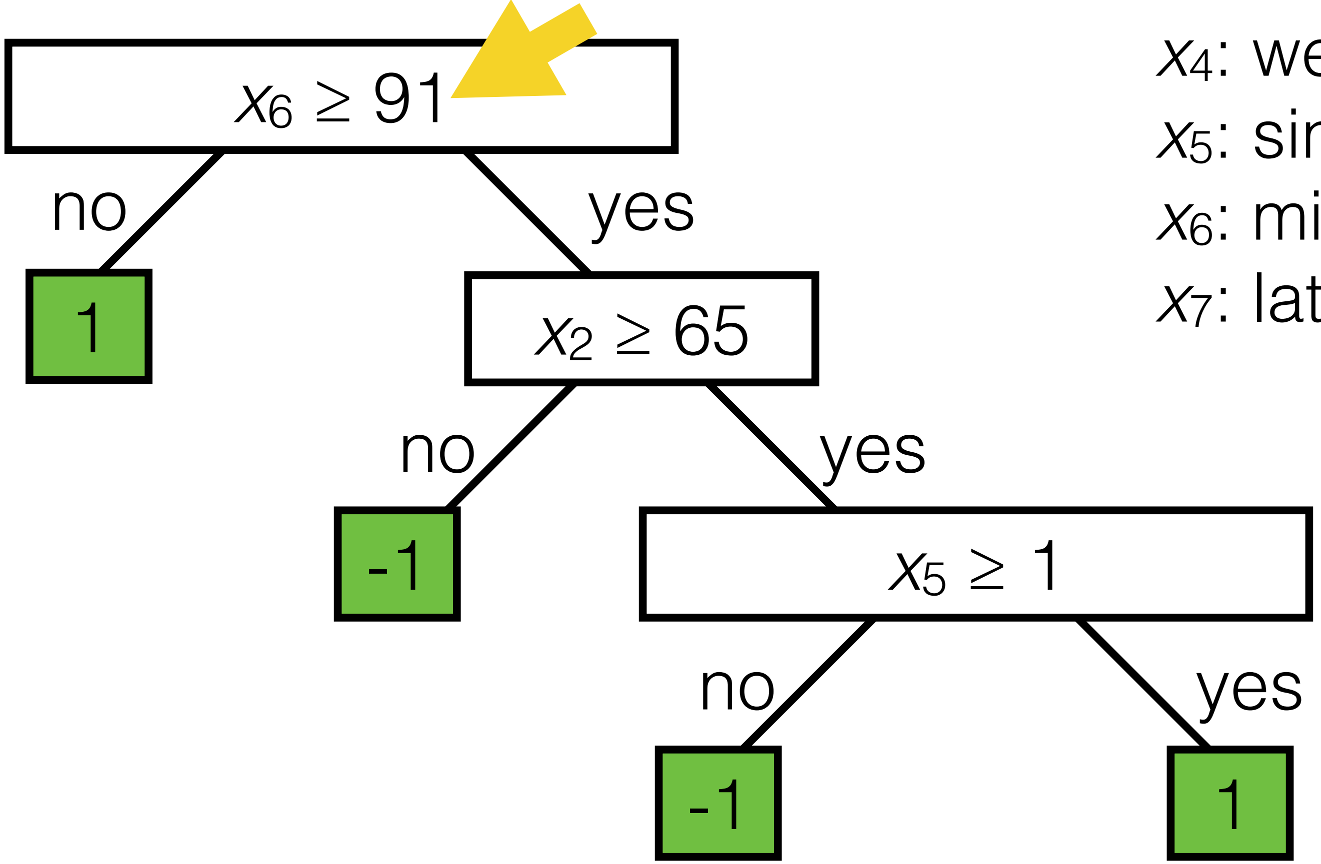
features:
$x_1$: date
$x_2$: age
$x_3$: height
$x_4$: weight
$x_5$: sinus tachycardia?
$x_6$: min systolic bp, 24h
$x_7$: latest diastolic bp

labels $y$:
1: high risk
-1: low risk

# Decision tree

internal node:

- dimension index *j*; split value *s*



features:

$x_1$: date

$x_2$: age

$x_3$: height

$x_4$: weight

$x_5$: sinus tachycardia?

$x_6$: min systolic bp, 24h

$x_7$: latest diastolic bp

labels $y$:

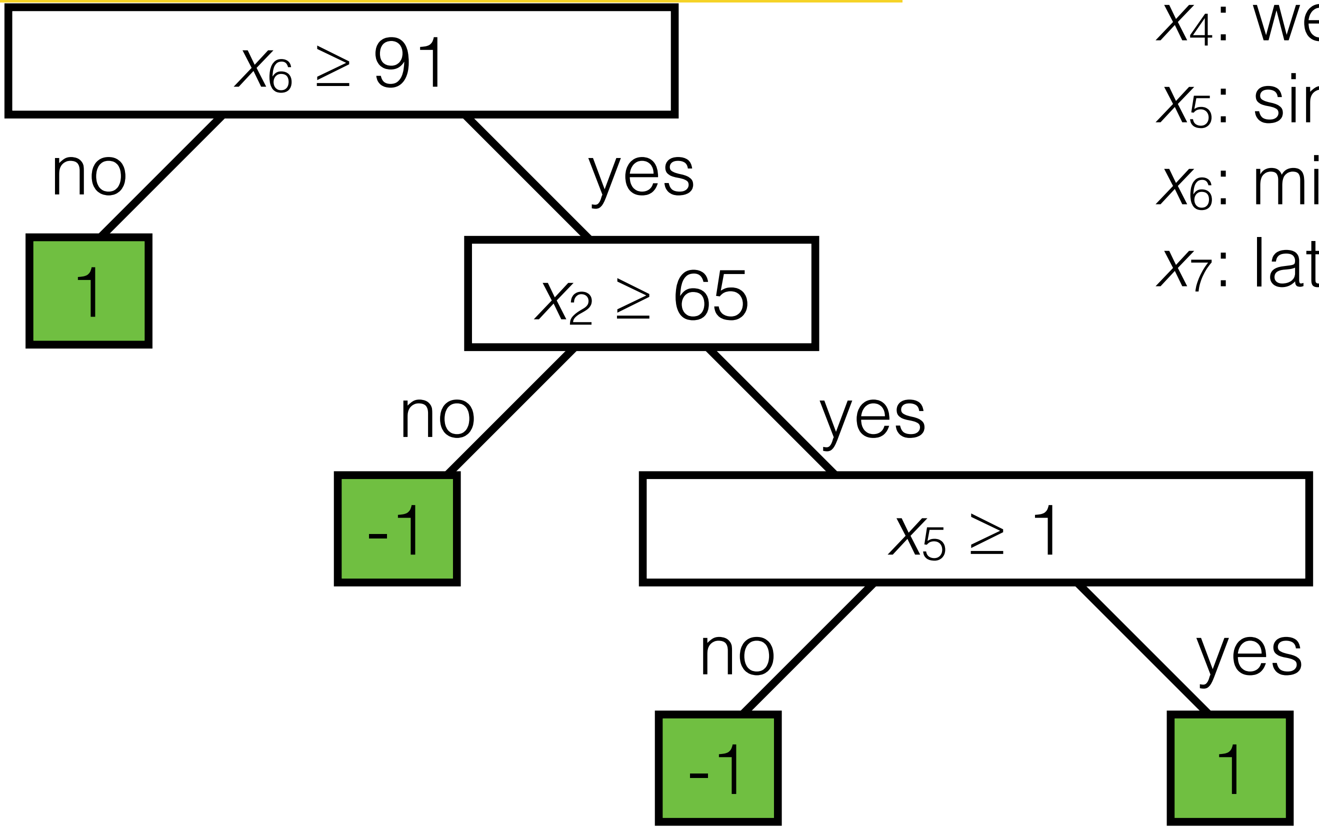
1: high risk

-1: low risk

# Decision tree

internal node:

- dimension index *j*; split value *s*
- two child nodes: internal or leaf

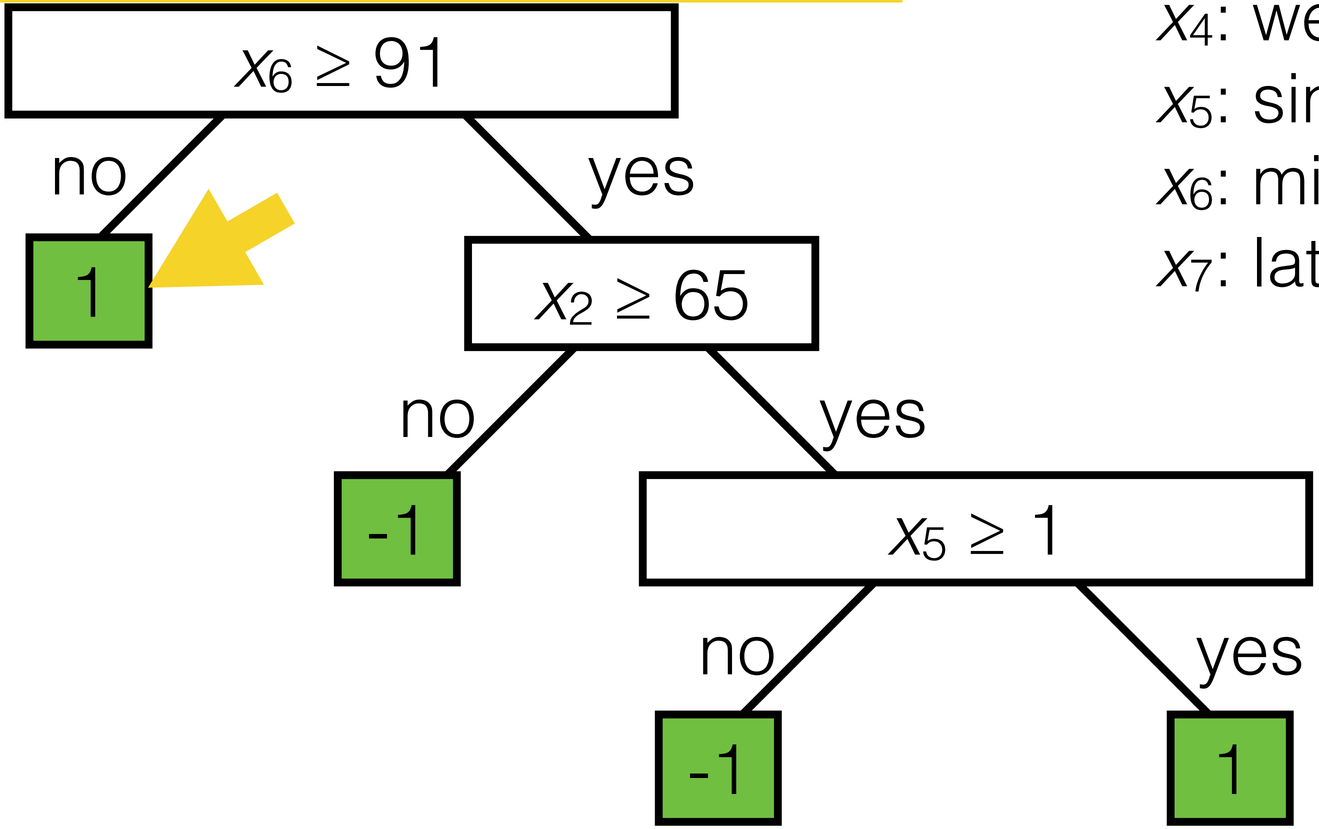


features:
$x_1$: date
$x_2$: age
$x_3$: height
$x_4$: weight
$x_5$: sinus tachycardia?
$x_6$: min systolic bp, 24h
$x_7$: latest diastolic bp

labels *y*:
1: high risk
-1: low risk

# Decision tree

internal node:
- dimension index *j*; split value *s*
- two child nodes: internal or leaf



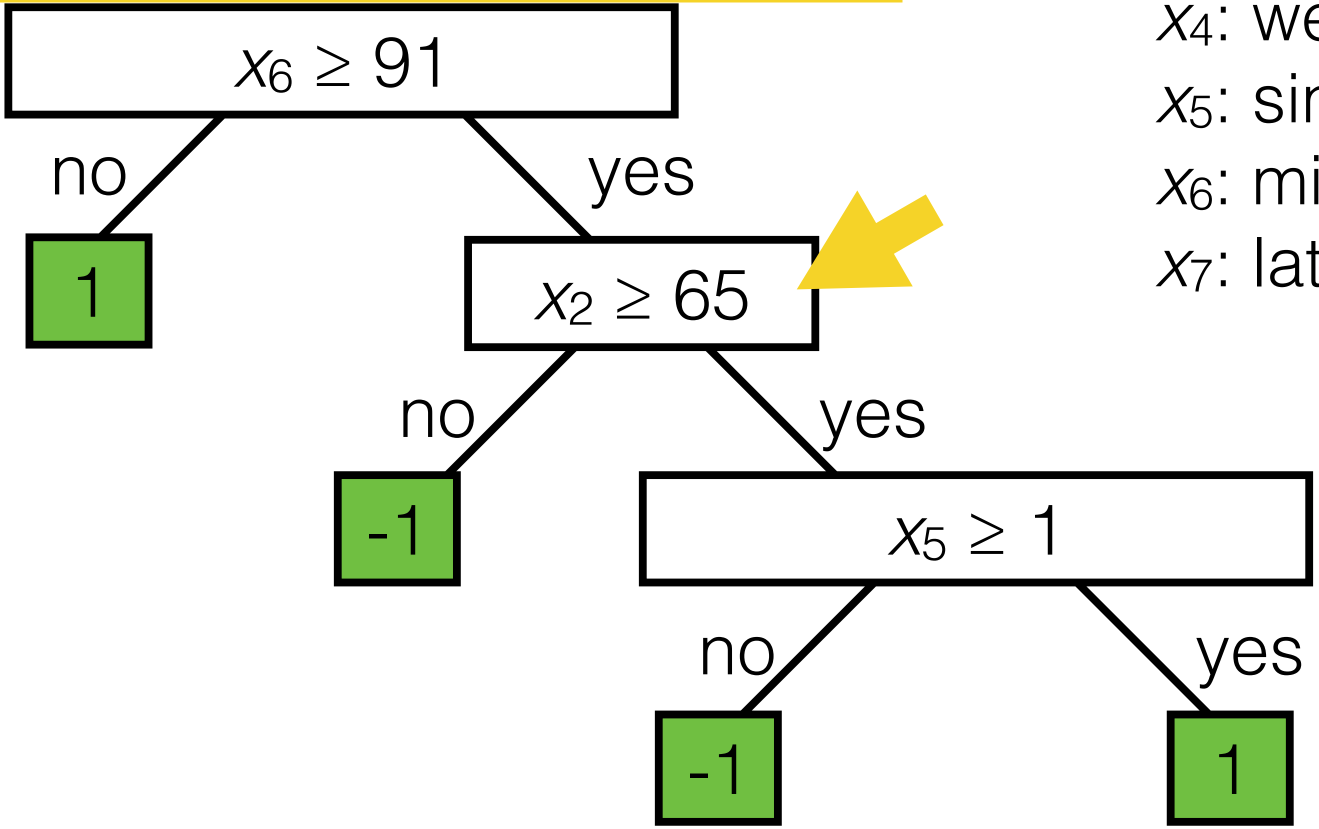
features:
$x_1$: date
$x_2$: age
$x_3$: height
$x_4$: weight
$x_5$: sinus tachycardia?
$x_6$: min systolic bp, 24h
$x_7$: latest diastolic bp

labels $y$:
1: high risk
-1: low risk

# Decision tree

internal node:

- dimension index *j*; split value *s*
- two child nodes: internal or leaf



$x_6 \geq 91$

no

1

yes

$x_2 \geq 65$

no

-1

yes

$x_5 \geq 1$

no

-1

yes

1

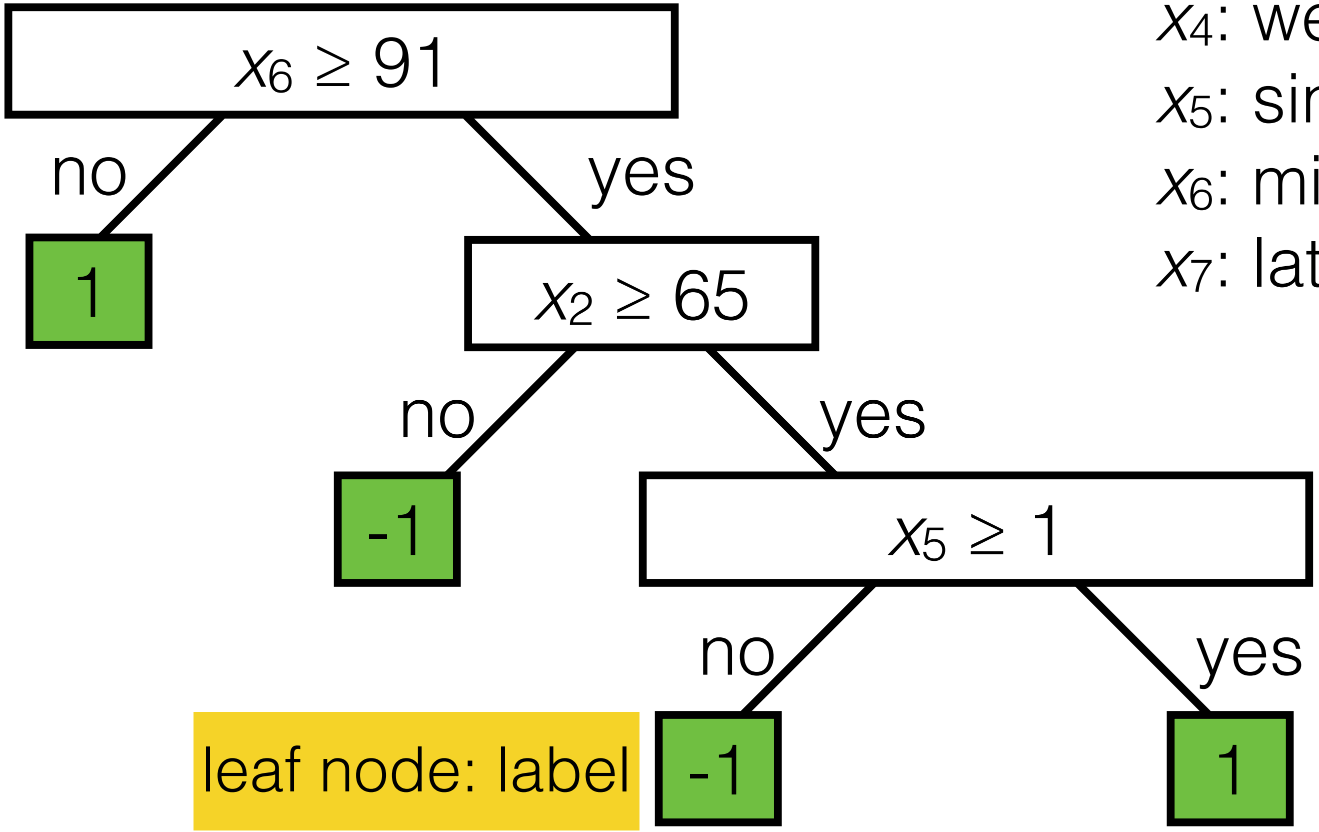
features:
$x_1$: date
$x_2$: age
$x_3$: height
$x_4$: weight
$x_5$: sinus tachycardia?
$x_6$: min systolic bp, 24h
$x_7$: latest diastolic bp

labels *y*:
1: high risk
-1: low risk

# Decision tree

internal node:

- dimension index *j*; split value *s*
- two child nodes: internal or leaf



leaf node: label

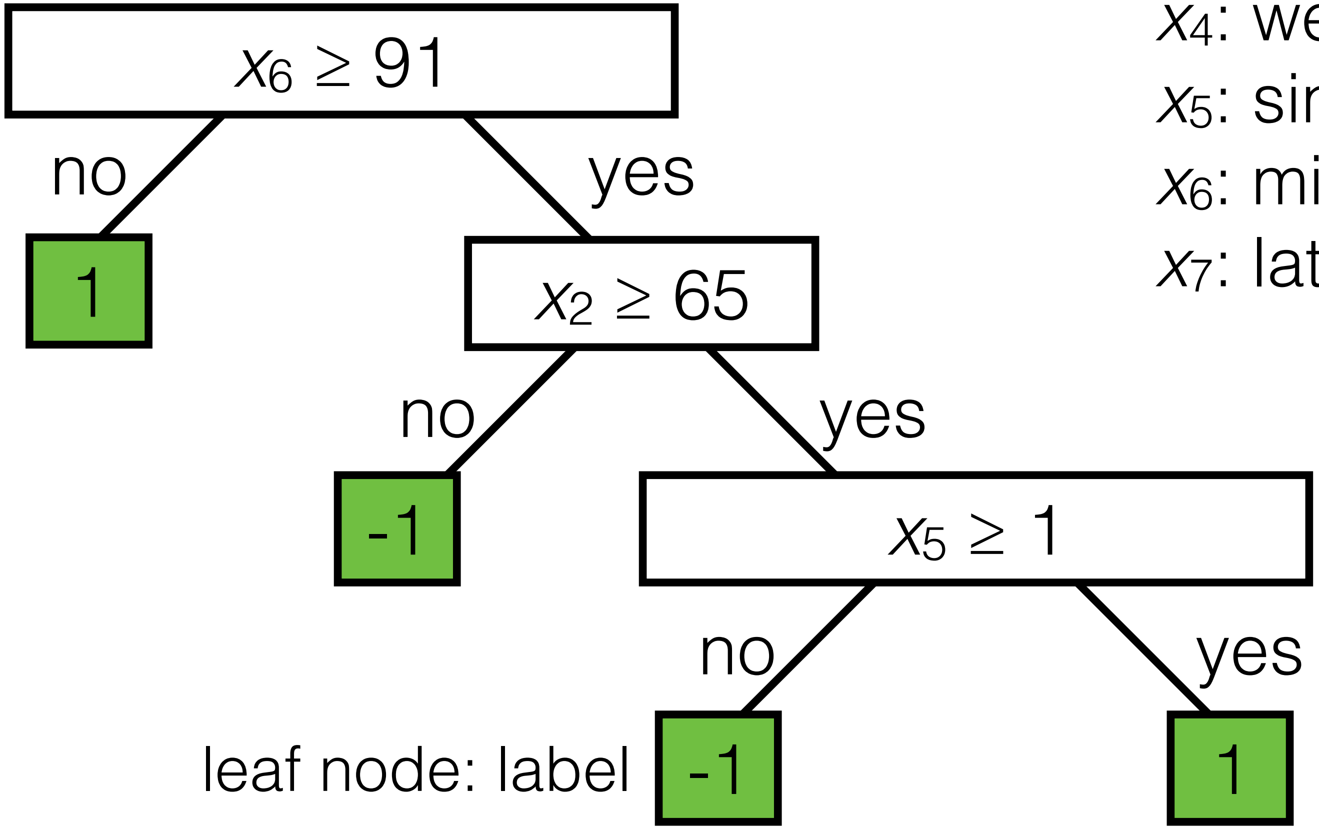
features:
$x_1$: date
$x_2$: age
$x_3$: height
$x_4$: weight
$x_5$: sinus tachycardia?
$x_6$: min systolic bp, 24h
$x_7$: latest diastolic bp

labels $y$:
1: high risk
-1: low risk

# Classification tree

internal node:

- dimension index *j*; split value *s*
- two child nodes: internal or leaf



features:
$x_1$: date
$x_2$: age
$x_3$: height
$x_4$: weight
$x_5$: sinus tachycardia?
$x_6$: min systolic bp, 24h
$x_7$: latest diastolic bp

labels $y$:
1: high risk
-1: low risk

# Classification tree

internal node:

- dimension index *j*; split value *s*
- two child nodes: internal or leaf

features:

$x_1$: date

$x_2$: age

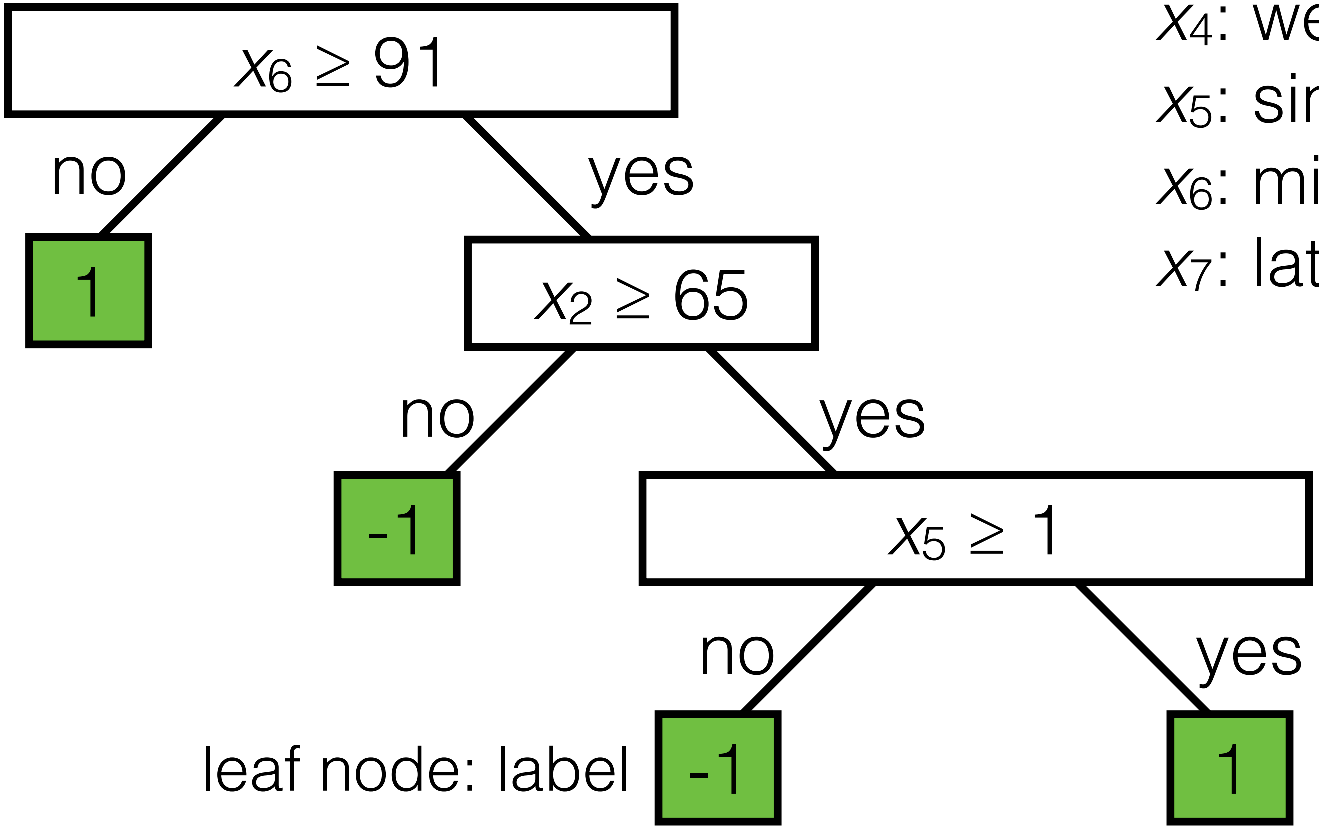
$x_3$: height

$x_4$: weight

$x_5$: sinus tachycardia?

$x_6$: min systolic bp, 24h

$x_7$: latest diastolic bp



leaf node: label

labels *y*:

1: high risk

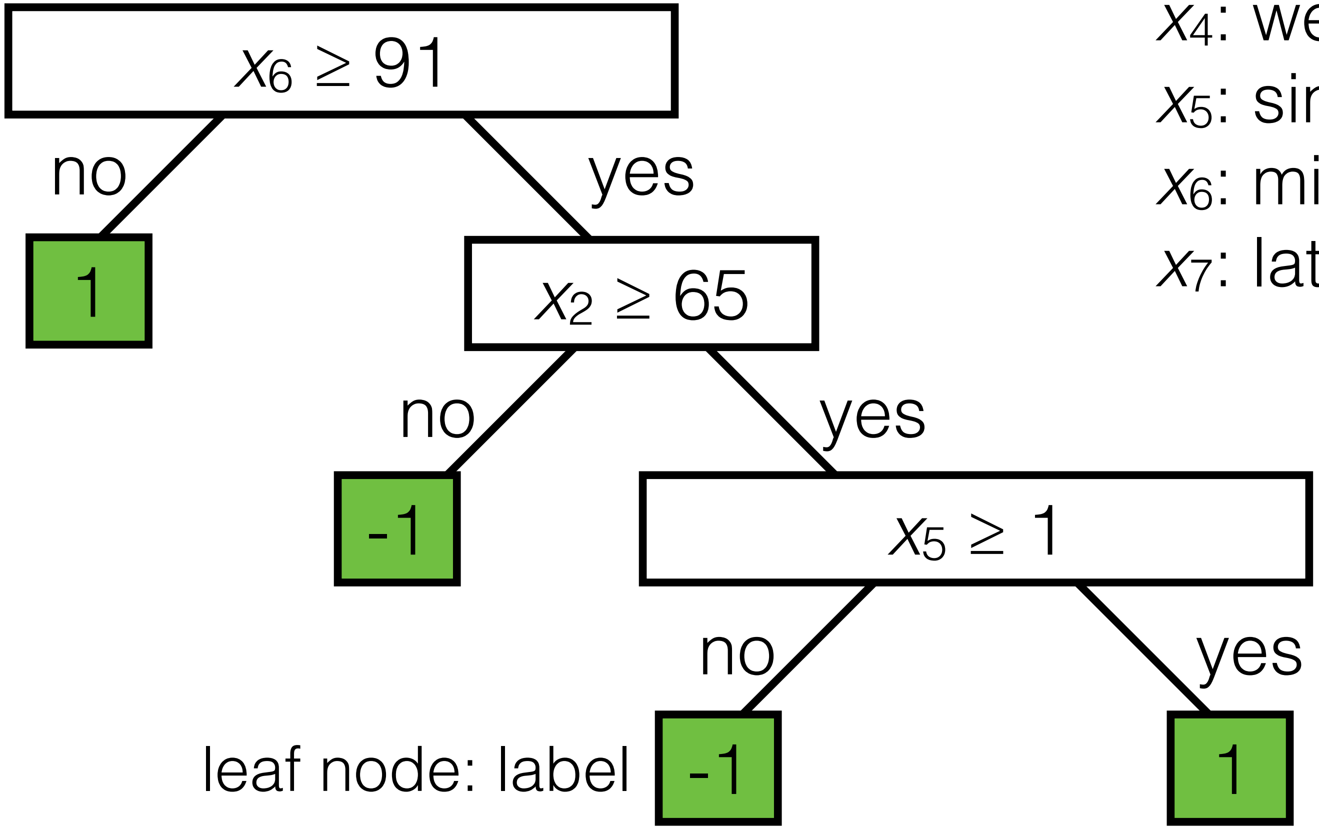
-1: low risk

$x^{(1)}$

# Classification tree

internal node:

- dimension index *j*; split value *s*
- two child nodes: internal or leaf



$x_6 \geq 91$

no

yes

1

$x_2 \geq 65$

no

yes

-1

$x_5 \geq 1$

no

yes

leaf node: label

-1

1

features:

$x_1$: date

$x_2$: age

$x_3$: height

$x_4$: weight

$x_5$: sinus tachycardia?

$x_6$: min systolic bp, 24h

$x_7$: latest diastolic bp

labels *y*:

1: high risk

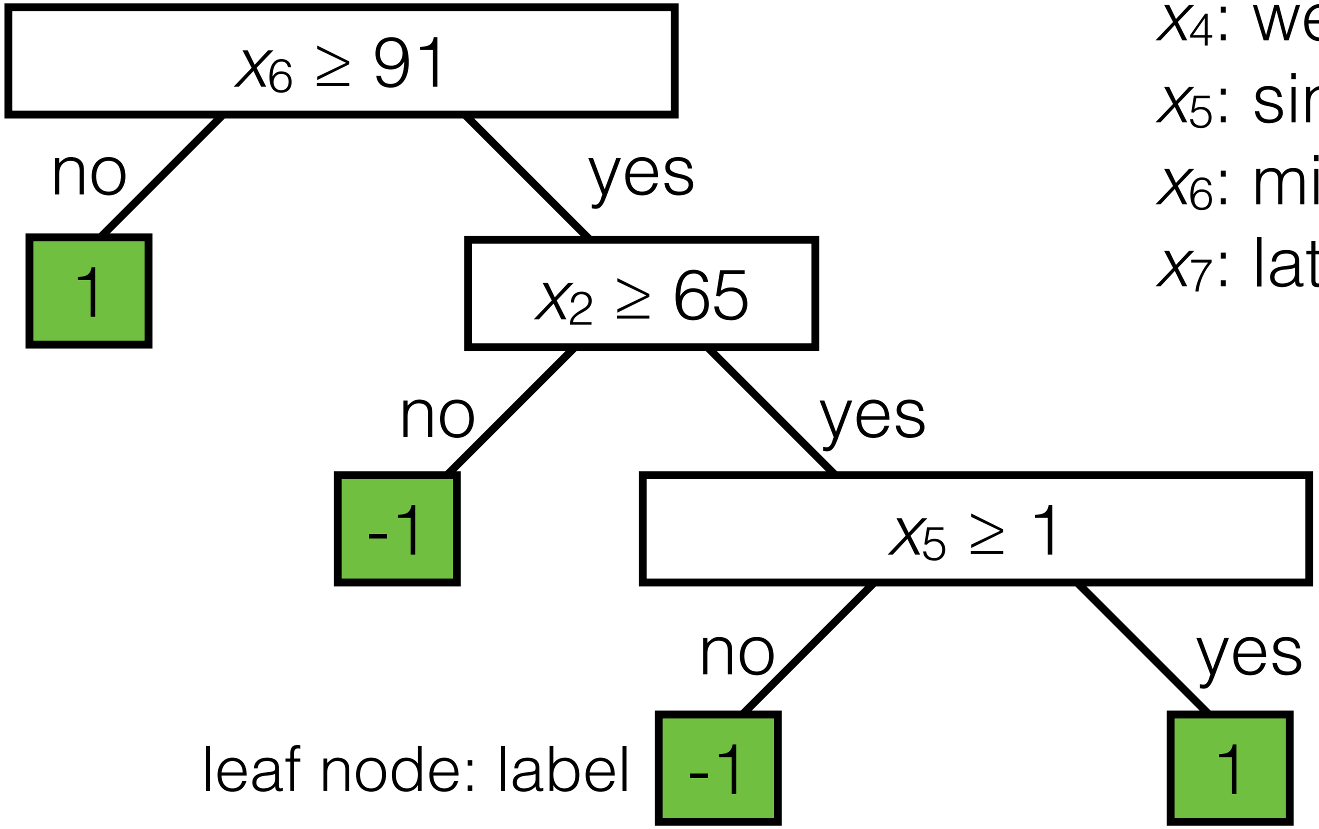
-1: low risk

$x^{(1)}$: 2020/11/17, 49, 172 cm, 70.5 kg, 0, 115, 79

# Classification tree

internal node:

- dimension index *j*; split value *s*
- two child nodes: internal or leaf



leaf node: label

features:
$x_1$: date
$x_2$: age
$x_3$: height
$x_4$: weight
$x_5$: sinus tachycardia?
$x_6$: min systolic bp, 24h
$x_7$: latest diastolic bp

labels $y$:
1: high risk
-1: low risk

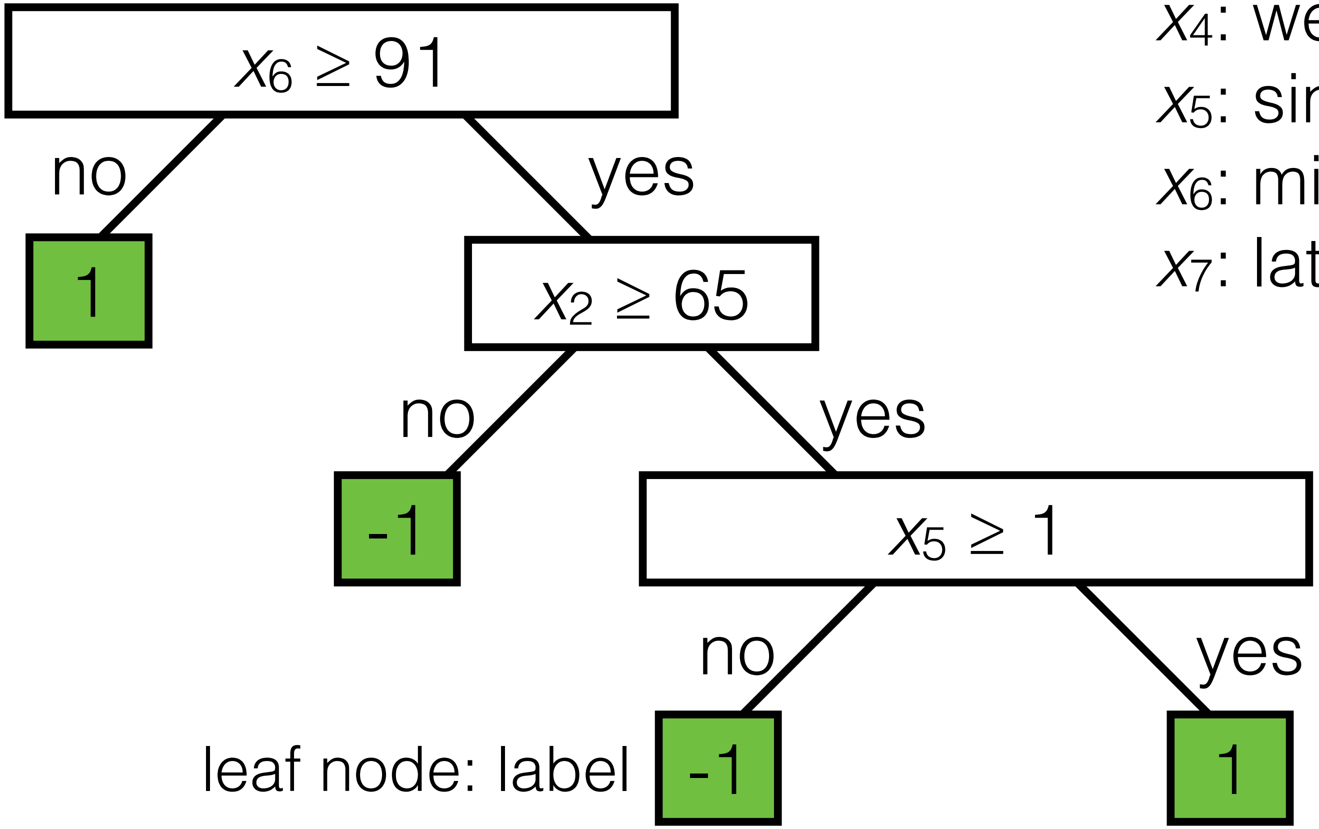
$x^{(1)}$: 2020/11/17, 49, 172 cm, 70.5 kg, 0, 115, 79

$T(x^{(1)})$

# Classification tree

internal node:

- dimension index *j*; split value *s*
- two child nodes: internal or leaf



features:
$x_1$: date
$x_2$: age
$x_3$: height
$x_4$: weight
$x_5$: sinus tachycardia?
$x_6$: min systolic bp, 24h
$x_7$: latest diastolic bp

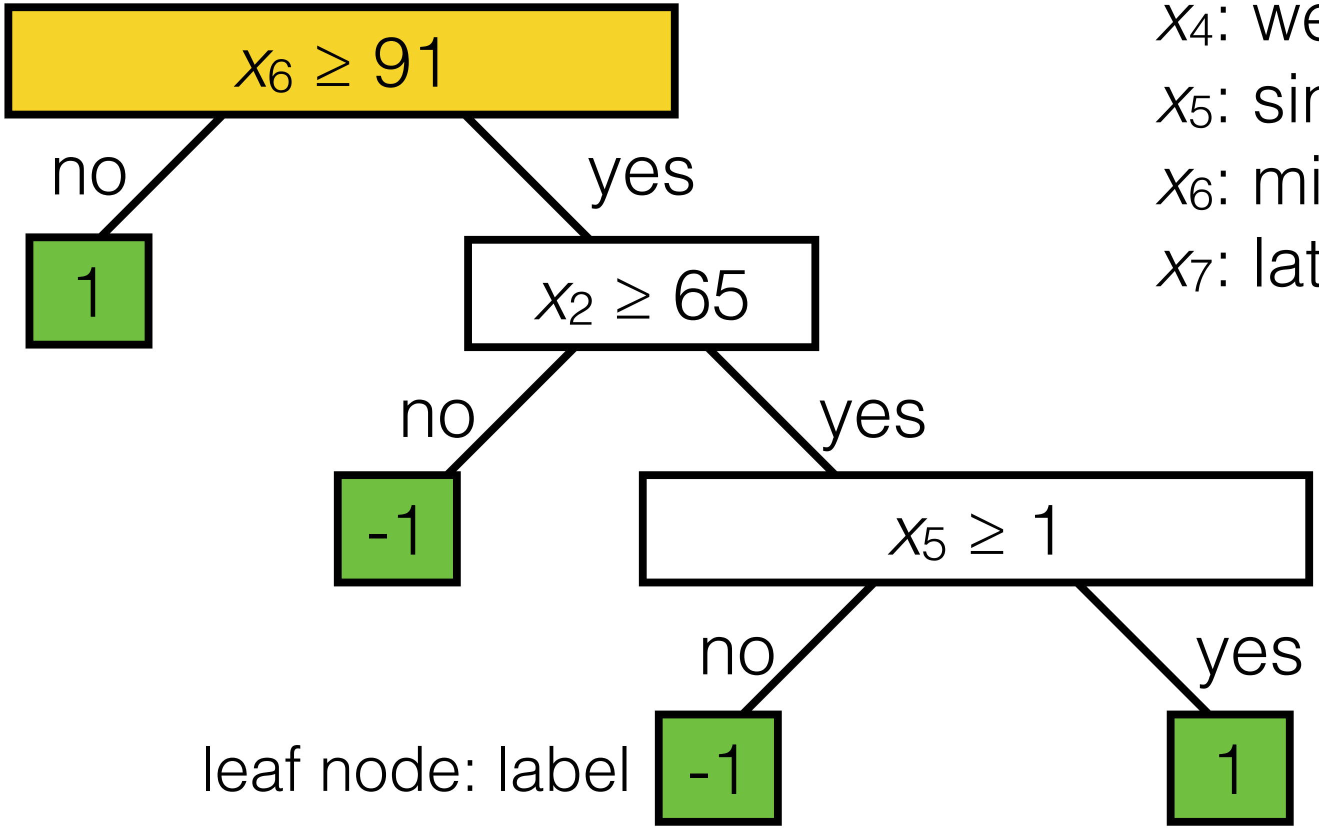
labels *y*:
1: high risk
-1: low risk

$x^{(1)}$: 2020/11/17, 49, 172 cm, 70.5 kg, 0, 115, 79

$T(x^{(1)}) =$

# Classification tree

internal node:
- dimension index *j*; split value *s*
- two child nodes: internal or leaf



leaf node: label

features:
$x_1$: date
$x_2$: age
$x_3$: height
$x_4$: weight
$x_5$: sinus tachycardia?
$x_6$: min systolic bp, 24h
$x_7$: latest diastolic bp

labels *y*:
1: high risk
-1: low risk

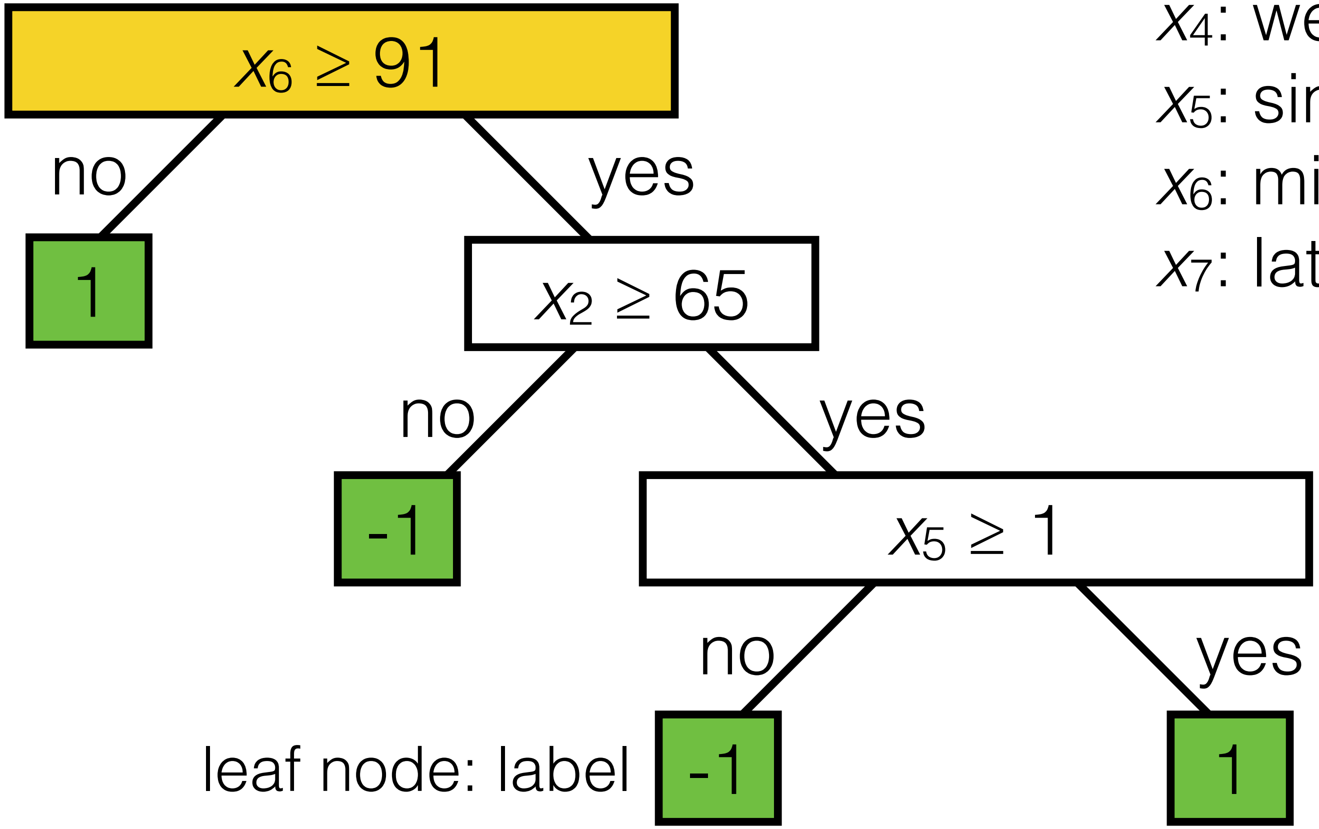
$x^{(1)}$: 2020/11/17, 49, 172 cm, 70.5 kg, 0, 115, 79
$T(x^{(1)}) =$

# Classification tree

internal node:
- dimension index *j*; split value *s*
- two child nodes: internal or leaf

features:
$x_1$: date
$x_2$: age
$x_3$: height
$x_4$: weight
$x_5$: sinus tachycardia?
$x_6$: min systolic bp, 24h
$x_7$: latest diastolic bp

labels *y*:
1: high risk
-1: low risk



$x_6 \geq 91$

no: 1

yes: $x_2 \geq 65$

no: -1

yes: $x_5 \geq 1$

no: -1

yes: 1

leaf node: label

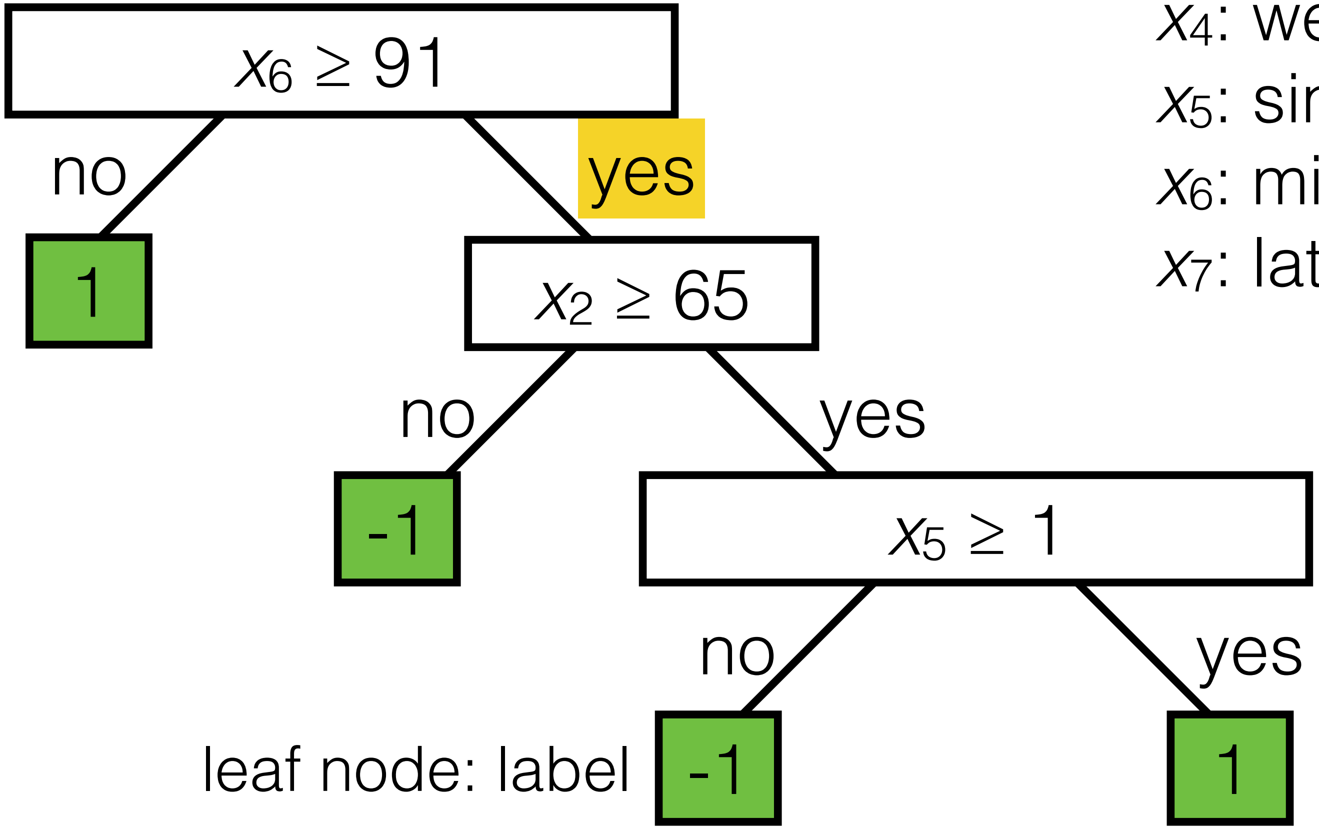
$x^{(1)}$: 2020/11/17, 49, 172 cm, 70.5 kg, 0, 115, 79

$T(x^{(1)}) =$

# Classification tree

internal node:

- dimension index *j*; split value *s*
- two child nodes: internal or leaf



$x_6 \geq 91$

no

1

yes

$x_2 \geq 65$

no

-1

yes

$x_5 \geq 1$

no

-1

yes

1

leaf node: label

features:

$x_1$: date

$x_2$: age

$x_3$: height

$x_4$: weight

$x_5$: sinus tachycardia?

$x_6$: min systolic bp, 24h

$x_7$: latest diastolic bp

labels *y*:

1: high risk

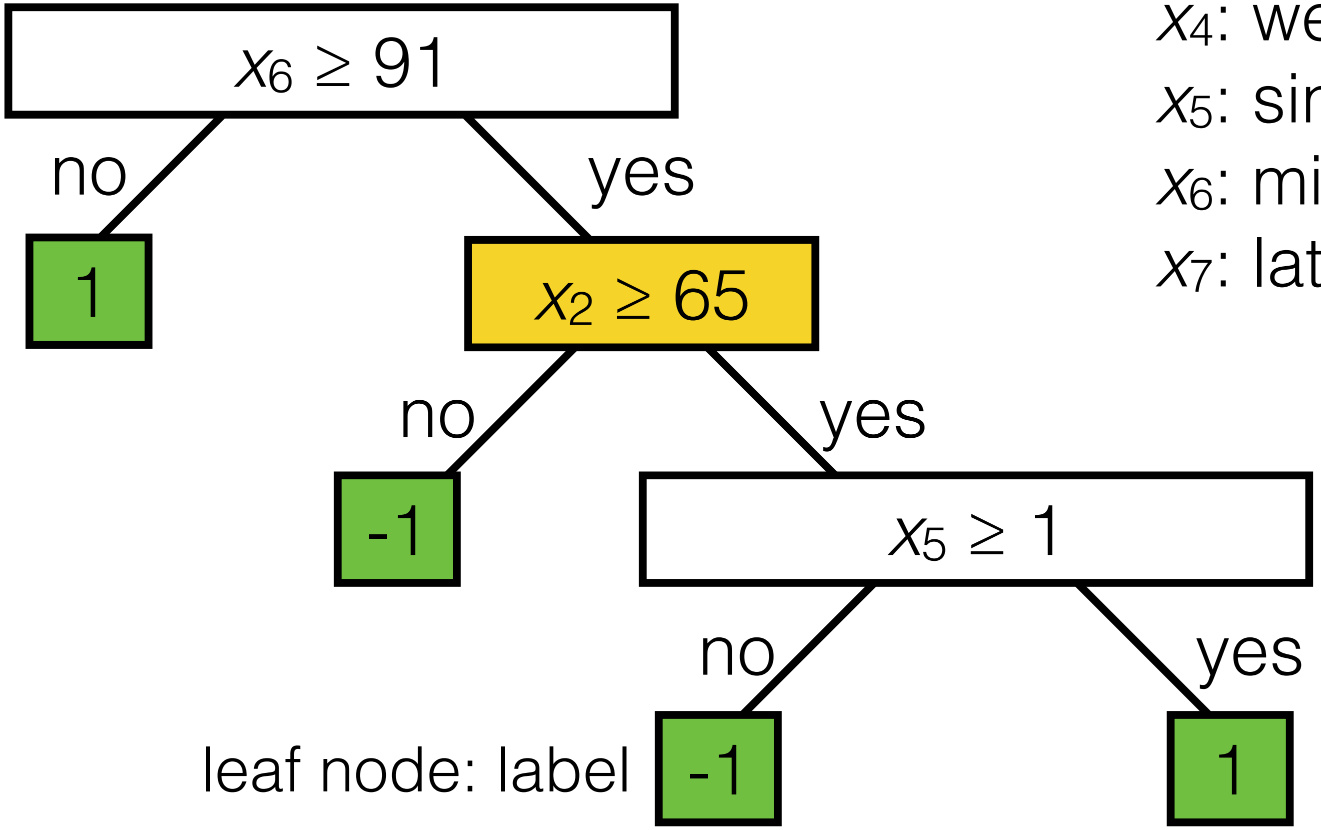
-1: low risk

$x^{(1)}$: 2020/11/17, 49, 172 cm, 70.5 kg, 0, 115, 79

$T(x^{(1)}) =$

# Classification tree

internal node:
- dimension index *j*; split value *s*
- two child nodes: internal or leaf



features:
$x_1$: date
$x_2$: age
$x_3$: height
$x_4$: weight
$x_5$: sinus tachycardia?
$x_6$: min systolic bp, 24h
$x_7$: latest diastolic bp

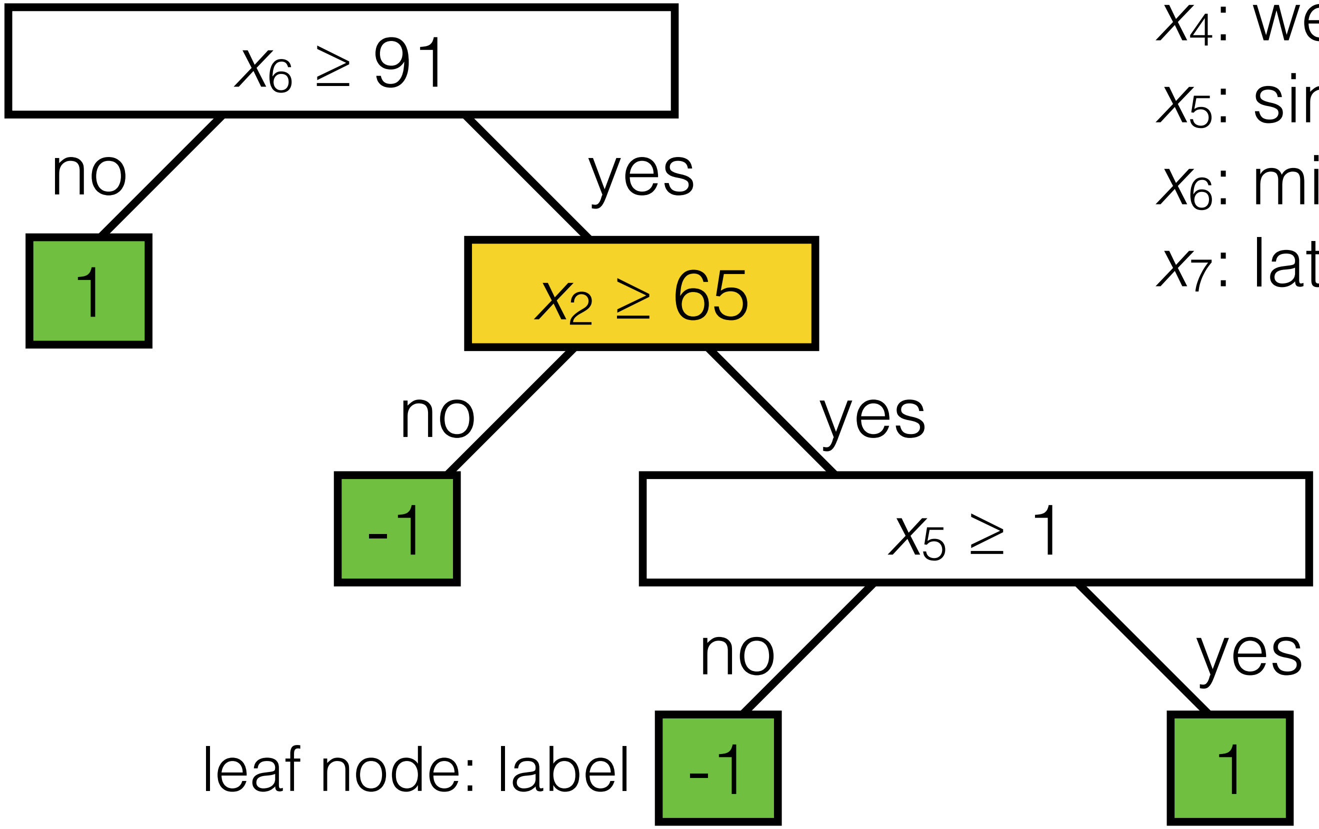
labels *y*:
1: high risk
-1: low risk

$x^{(1)}$: 2020/11/17, 49, 172 cm, 70.5 kg, 0, 115, 79
$T(x^{(1)}) =$

# Classification tree

internal node:

- dimension index *j*; split value *s*
- two child nodes: internal or leaf



leaf node: label

features:
$x_1$: date
$x_2$: age
$x_3$: height
$x_4$: weight
$x_5$: sinus tachycardia?
$x_6$: min systolic bp, 24h
$x_7$: latest diastolic bp

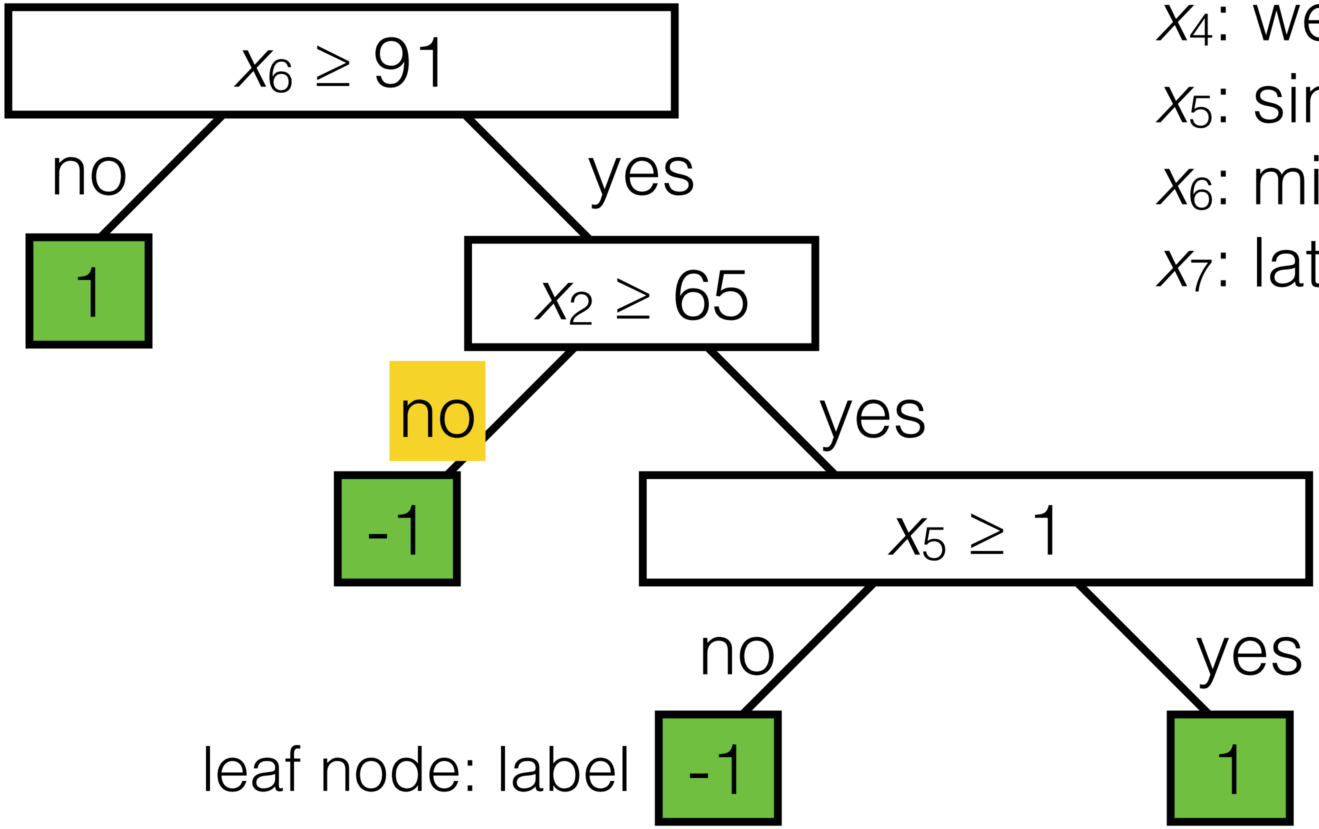
labels *y*:
1: high risk
-1: low risk

$x^{(1)}$: 2020/11/17, 49, 172 cm, 70.5 kg, 0, 115, 79
$T(x^{(1)}) =$

# Classification tree

internal node:

- dimension index *j*; split value *s*
- two child nodes: internal or leaf



leaf node: label

features:
$x_1$: date
$x_2$: age
$x_3$: height
$x_4$: weight
$x_5$: sinus tachycardia?
$x_6$: min systolic bp, 24h
$x_7$: latest diastolic bp

labels *y*:
1: high risk
-1: low risk

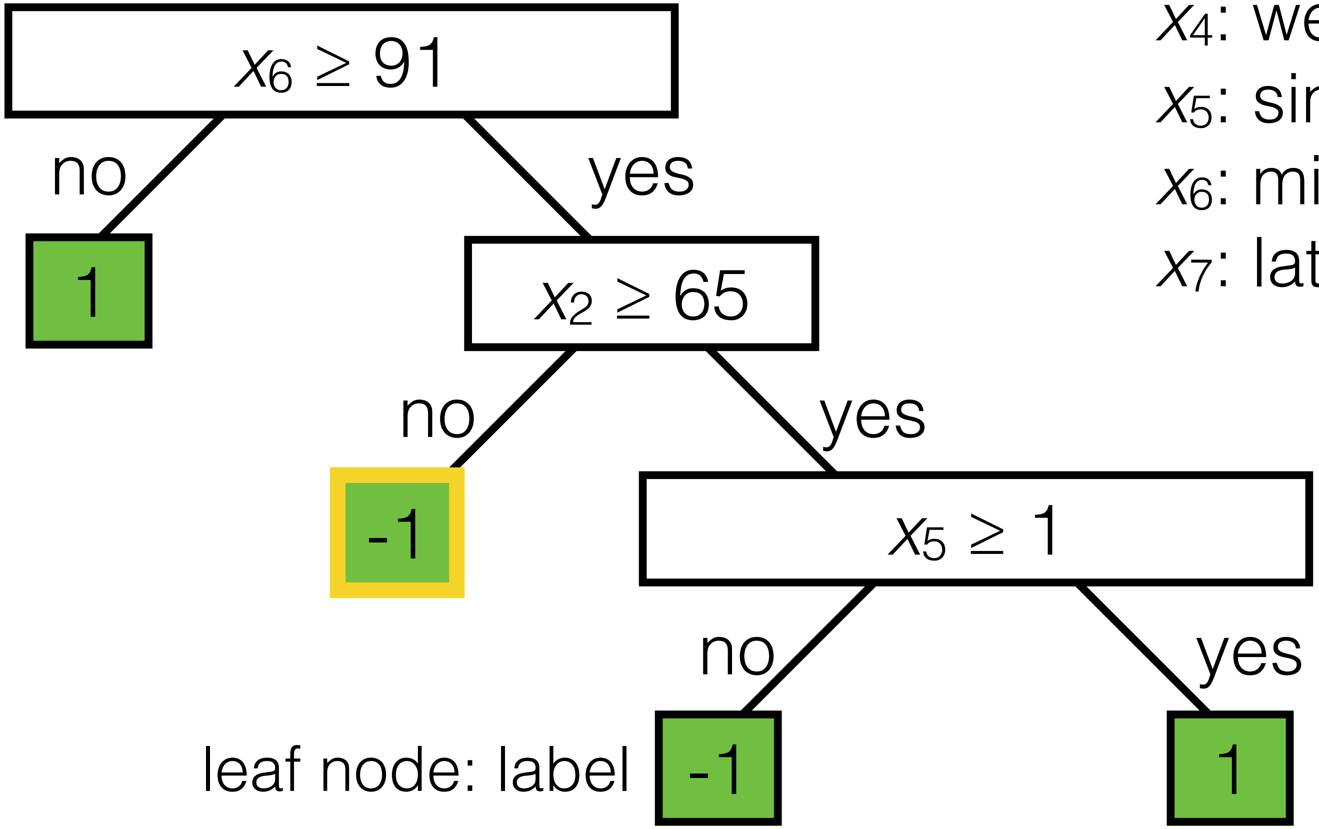
$x^{(1)}$: 2020/11/17, 49, 172 cm, 70.5 kg, 0, 115, 79

$T(x^{(1)}) =$

# Classification tree

internal node:

- dimension index *j*; split value *s*
- two child nodes: internal or leaf



$x_6 \geq 91$

no

1

yes

$x_2 \geq 65$

no

-1

yes

$x_5 \geq 1$

no

-1

yes

1

leaf node: label

features:

$x_1$: date

$x_2$: age

$x_3$: height

$x_4$: weight

$x_5$: sinus tachycardia?

$x_6$: min systolic bp, 24h

$x_7$: latest diastolic bp

labels *y*:

1: high risk

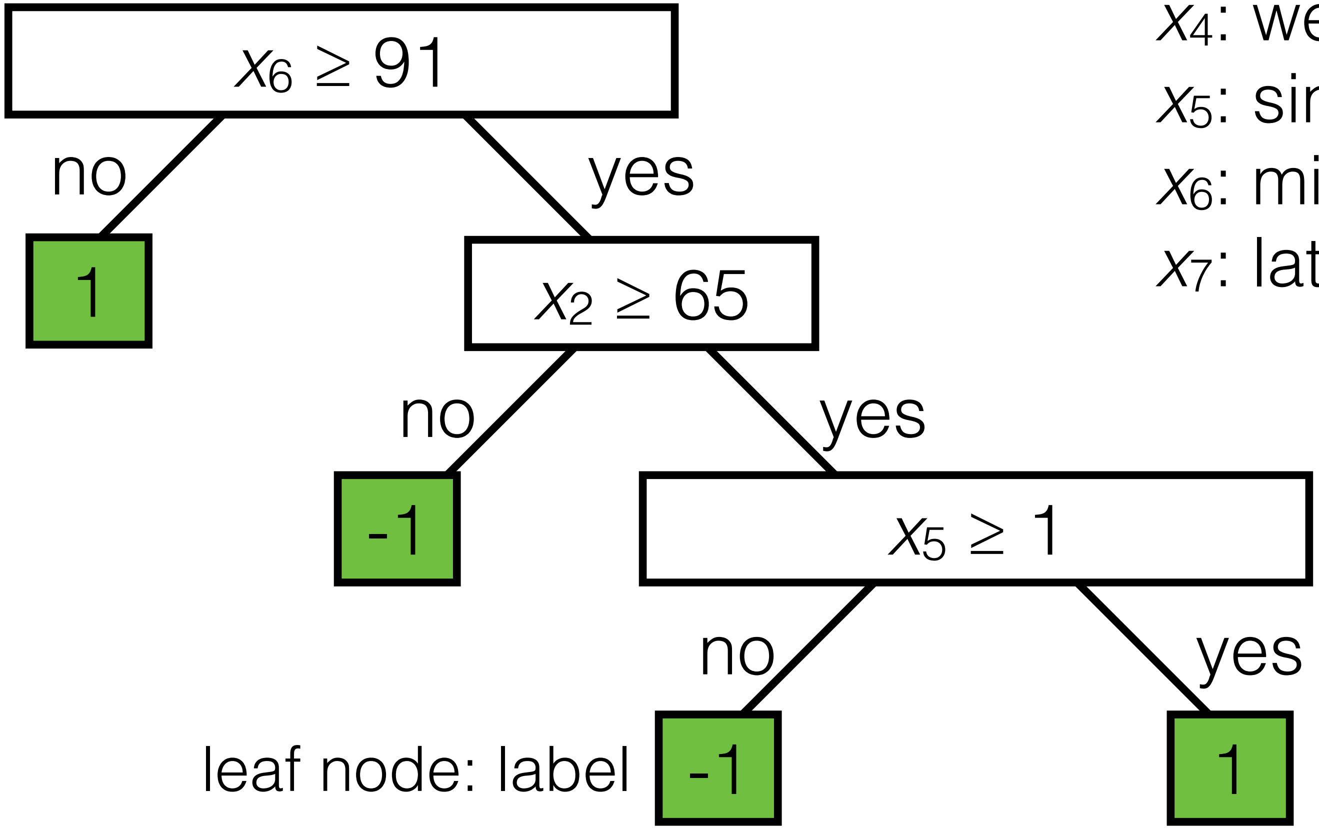
-1: low risk

$x^{(1)}$: 2020/11/17, 49, 172 cm, 70.5 kg, 0, 115, 79

$T(x^{(1)}) =$

# Classification tree

internal node:
- dimension index *j*; split value *s*
- two child nodes: internal or leaf



leaf node: label

features:
$x_1$: date
$x_2$: age
$x_3$: height
$x_4$: weight
$x_5$: sinus tachycardia?
$x_6$: min systolic bp, 24h
$x_7$: latest diastolic bp

labels *y*:
1: high risk
-1: low risk

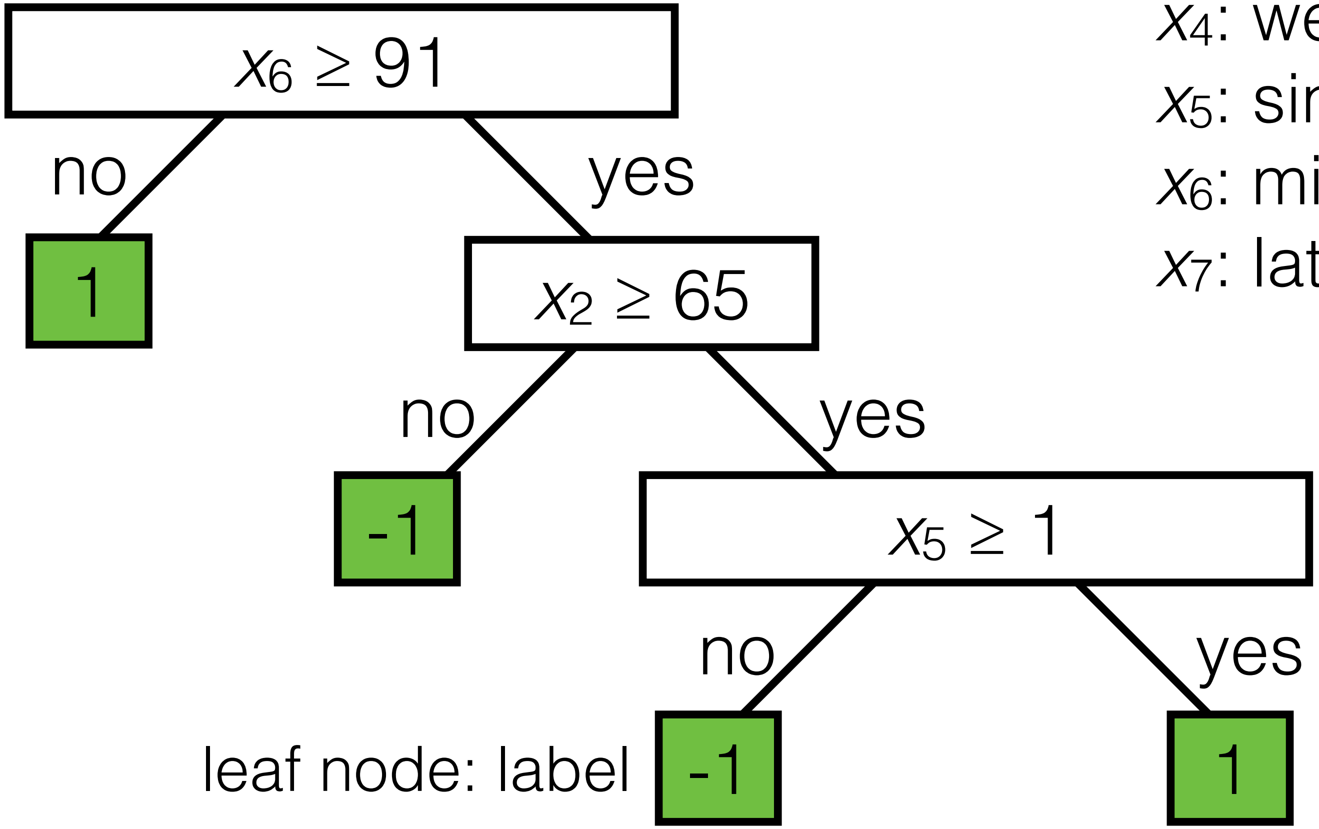
$x^{(1)}$: 2020/11/17, 49, 172 cm, 70.5 kg, 0, 115, 79

$T(x^{(1)}) = -1$

# Classification tree

internal node:

- dimension index *j*; split value *s*
- two child nodes: internal or leaf



leaf node: label

features:
$x_1$: date
$x_2$: age
$x_3$: height
$x_4$: weight
$x_5$: sinus tachycardia?
$x_6$: min systolic bp, 24h
$x_7$: latest diastolic bp

labels *y*:
1: high risk
-1: low risk

$x^{(1)}$: 2020/11/17, 49, 172 cm, 70.5 kg, 0, 115, 79

$$T(x^{(1)}) = -1$$

# Regression tree

# Regression tree

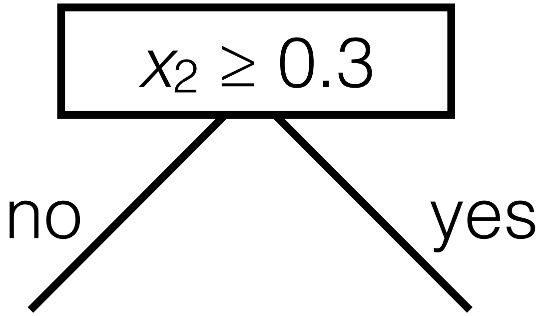
features:
$x_1$: temperature (deg C)
$x_2$: precipitation (cm/hr)

# Regression tree

features:
$x_1$: temperature (deg C)
$x_2$: precipitation (cm/hr)

labels:
$y$: km run

# Regression tree



$x_2 \geq 0.3$

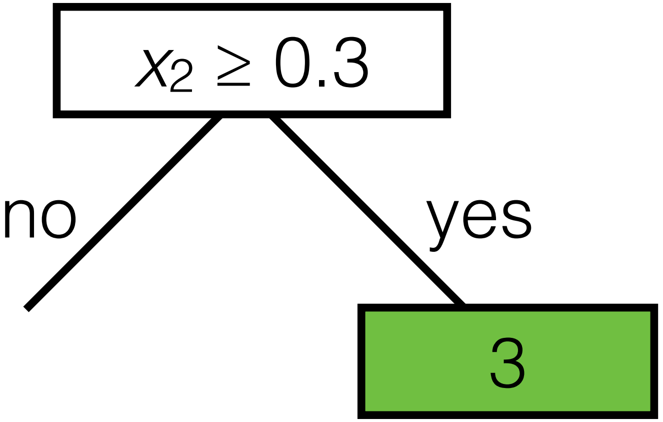
no

yes

features:
$x_1$: temperature (deg C)
$x_2$: precipitation (cm/hr)

labels:
$y$: km run

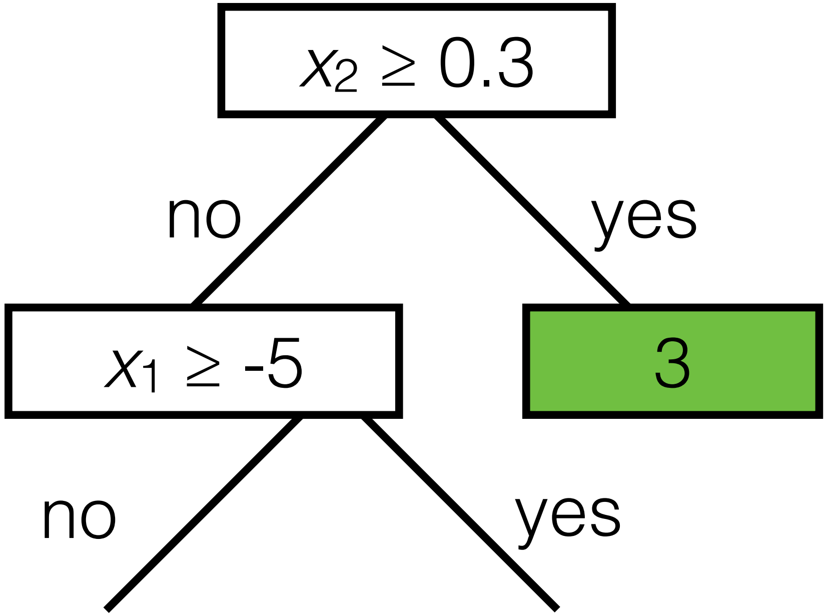
# Regression tree



features:
$x_1$: temperature (deg C)
$x_2$: precipitation (cm/hr)

labels:
$y$: km run

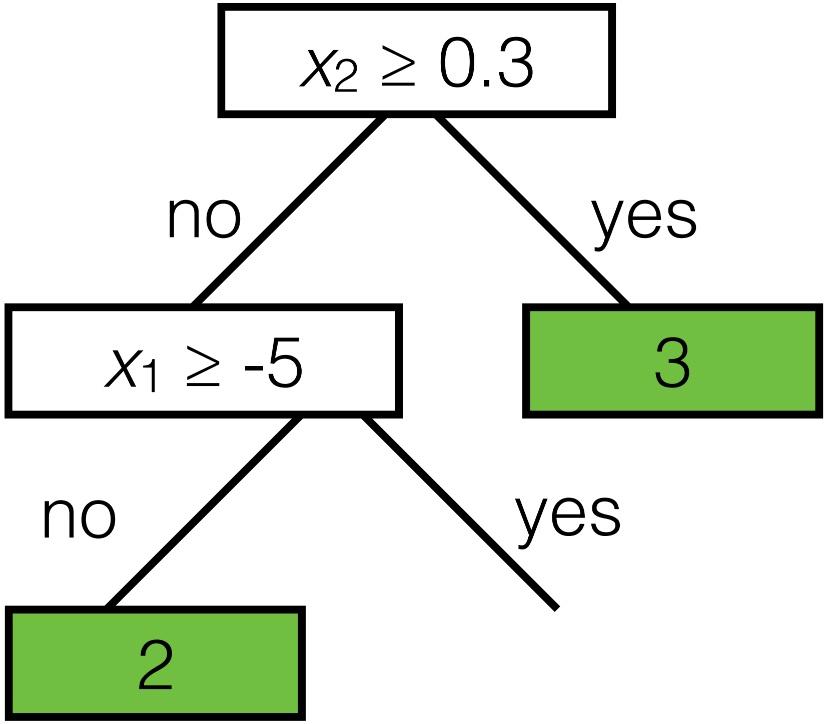
# Regression tree



features:
$x_1$: temperature (deg C)
$x_2$: precipitation (cm/hr)

labels:
$y$: km run

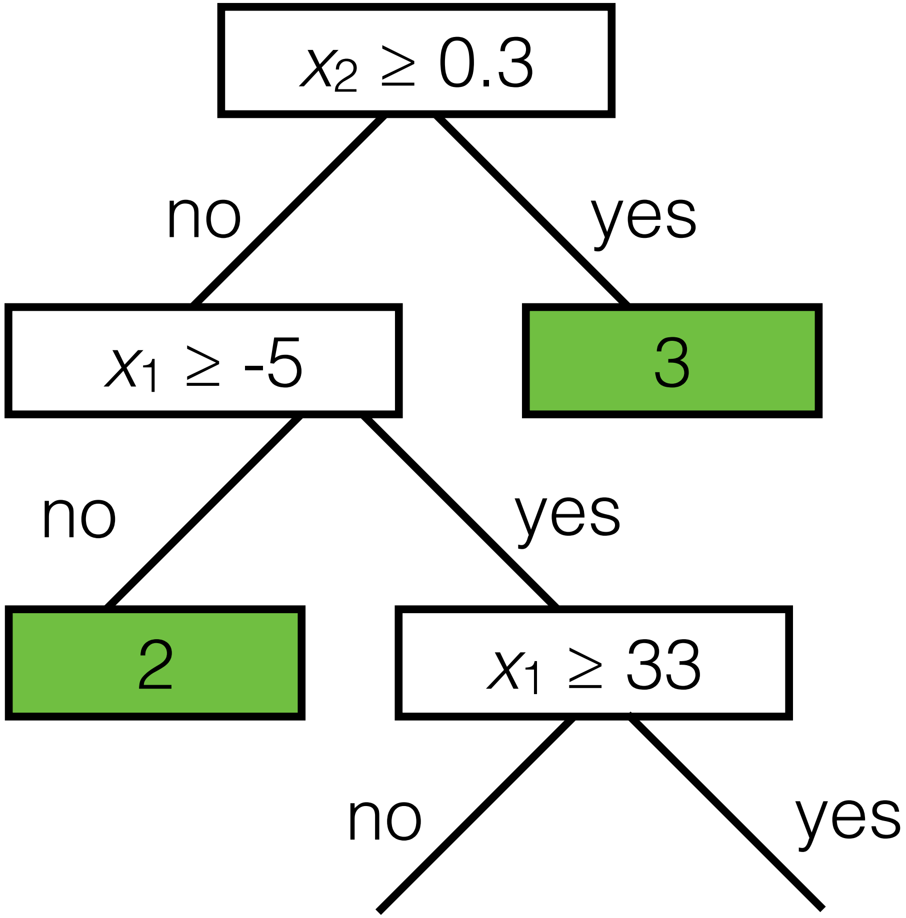
# Regression tree



features:
$x_1$: temperature (deg C)
$x_2$: precipitation (cm/hr)

labels:
$y$: km run

# Regression tree



$x_2 \geq 0.3$

no

yes

$x_1 \geq -5$

3

no

yes

2

$x_1 \geq 33$

no

yes

features:

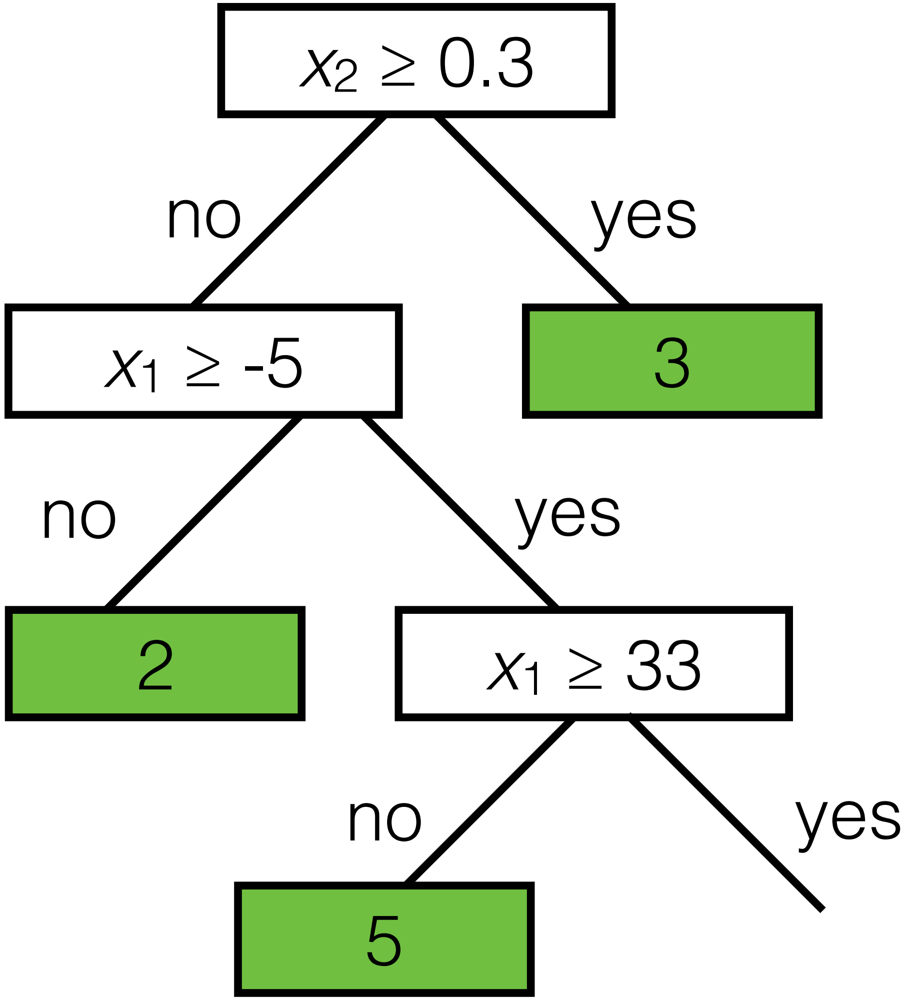
$x_1$: temperature (deg C)

$x_2$: precipitation (cm/hr)

labels:

$y$: km run

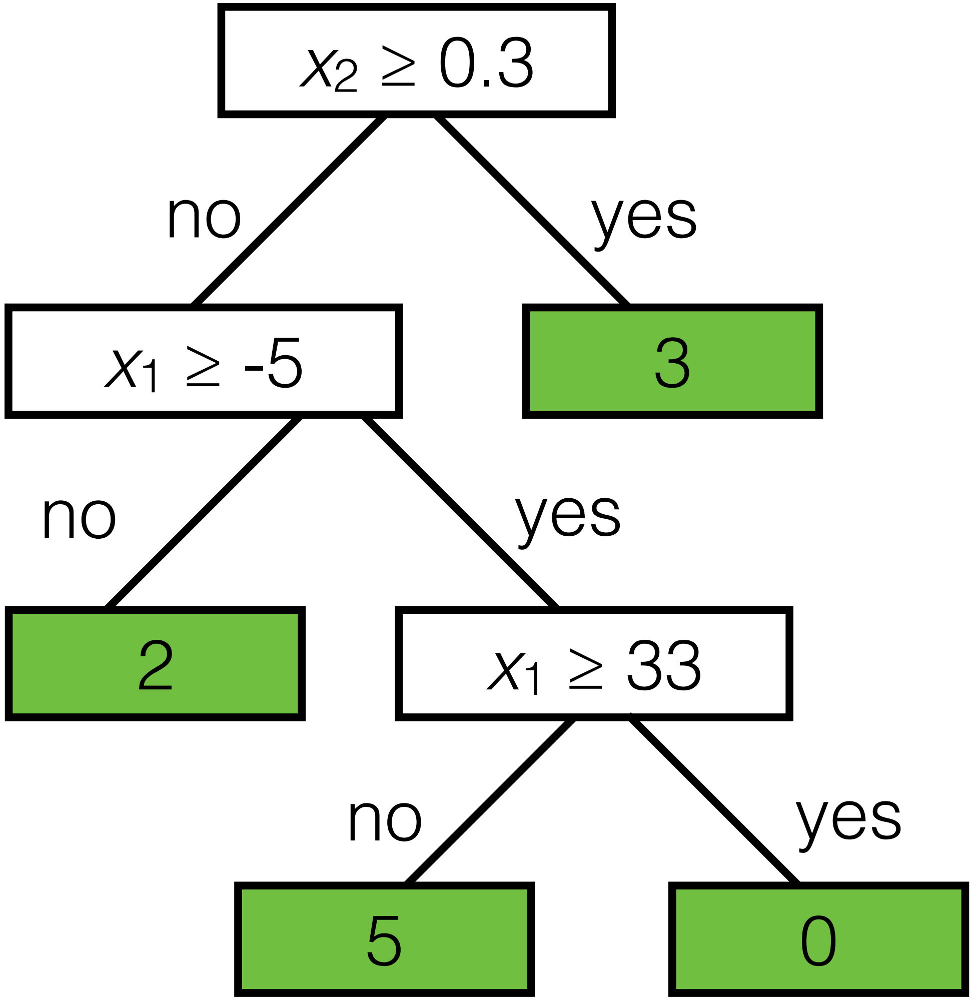
# Regression tree



features:
$x_1$: temperature (deg C)
$x_2$: precipitation (cm/hr)

labels:
$y$: km run

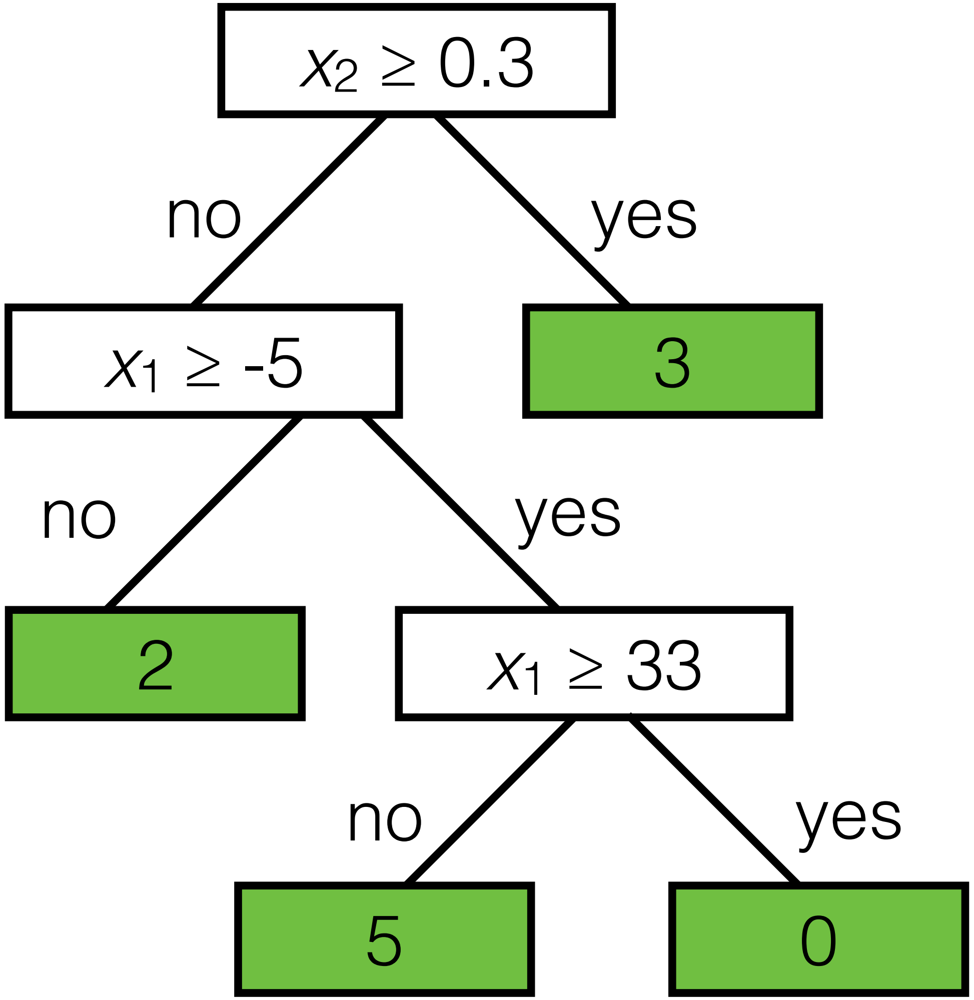
# Regression tree



features:
$x_1$: temperature (deg C)
$x_2$: precipitation (cm/hr)

labels:
$y$: km run

# Regression tree

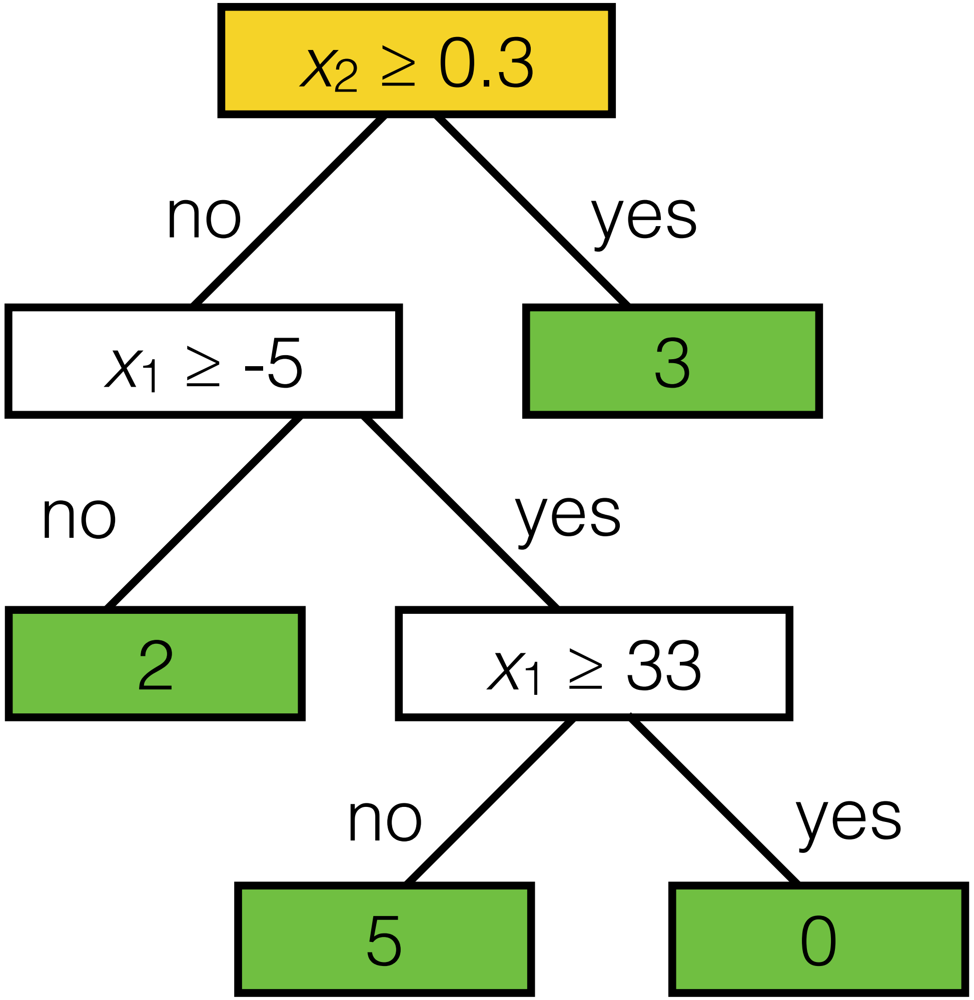


features:
$x_1$: temperature (deg C)
$x_2$: precipitation (cm/hr)

labels:
$y$: km run

- Tree defines an axis-aligned "partition" of the feature space:

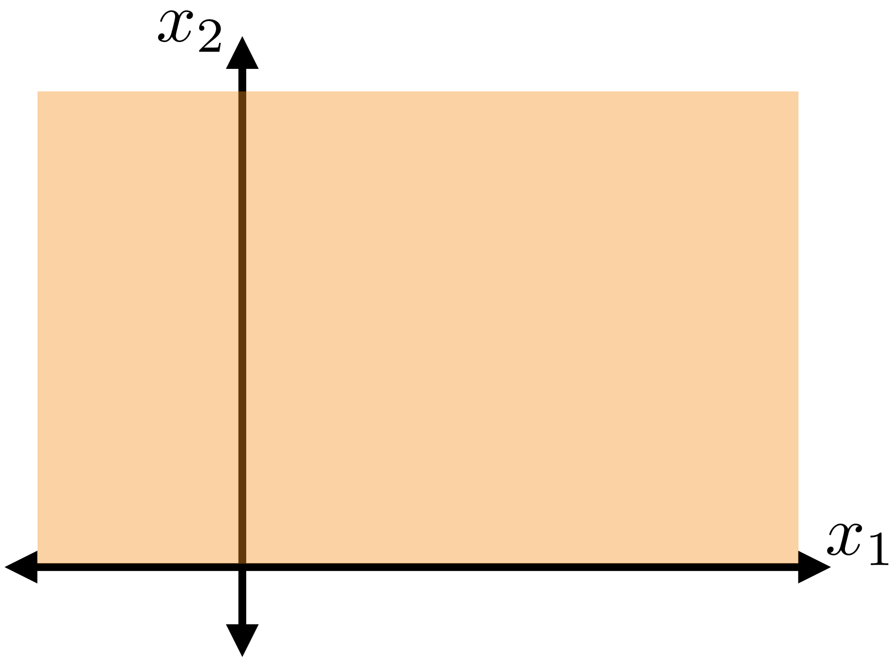
# Regression tree



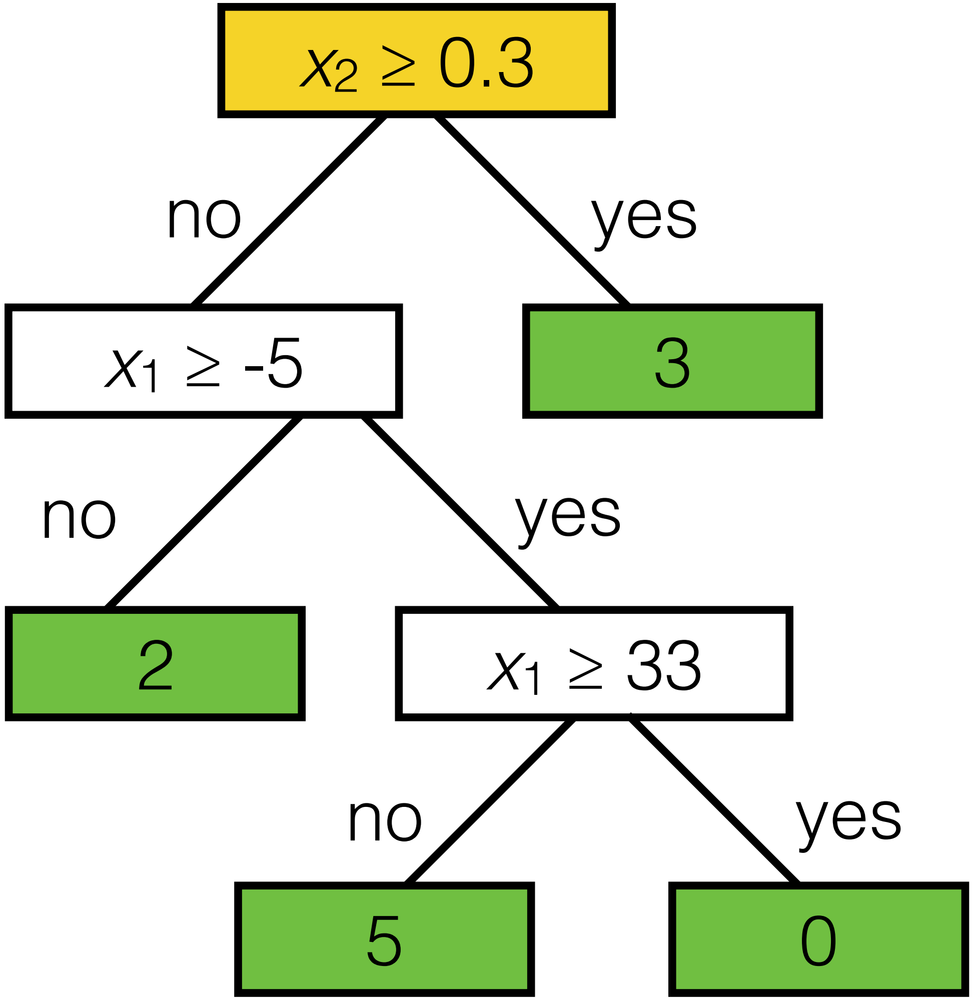
features:
$x_1$: temperature (deg C)
$x_2$: precipitation (cm/hr)

labels:
$y$: km run

- Tree defines an axis-aligned “partition” of the feature space:



# Regression tree



$x_2 \geq 0.3$

no

yes

$x_1 \geq -5$

3

no

yes

2

$x_1 \geq 33$

no

yes

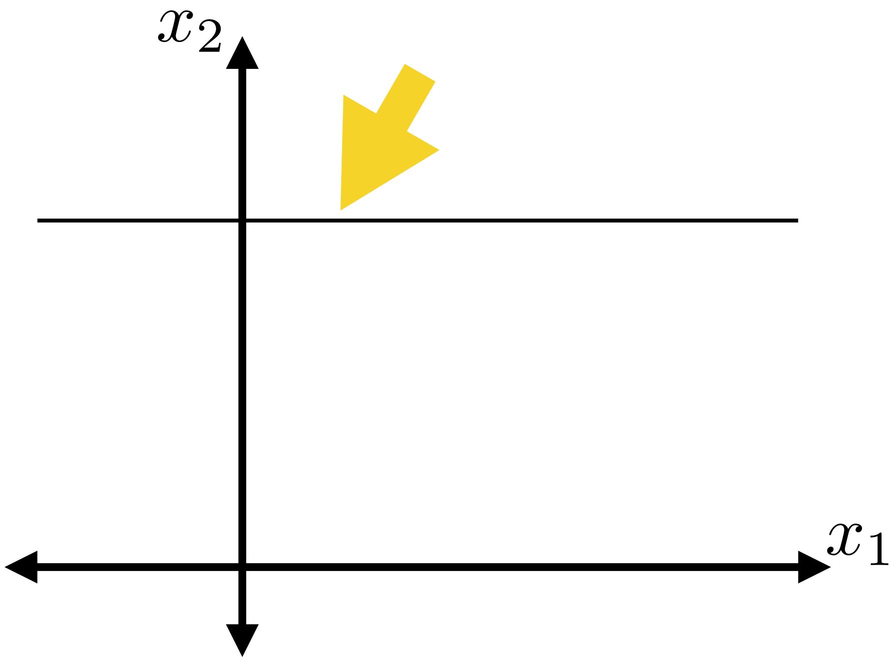
5

0

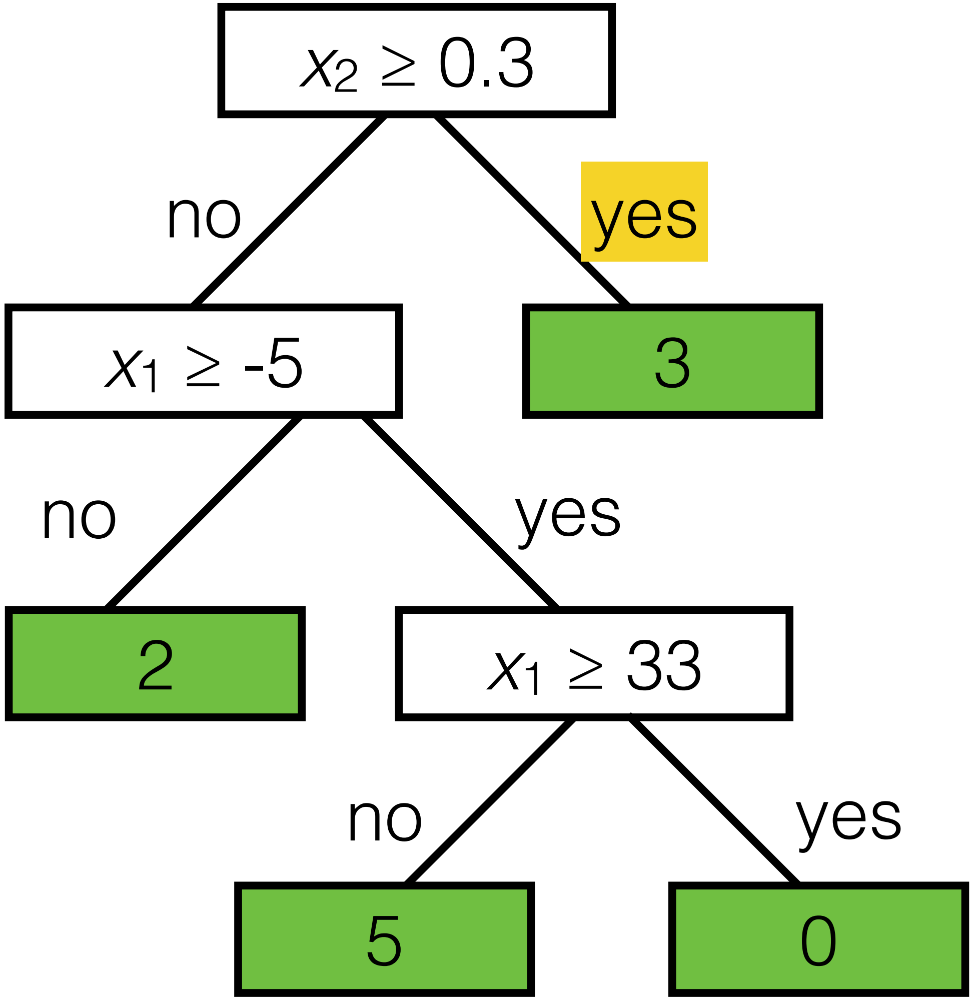
features:
$x_1$: temperature (deg C)
$x_2$: precipitation (cm/hr)

labels:
$y$: km run

- Tree defines an axis-aligned “partition” of the feature space:



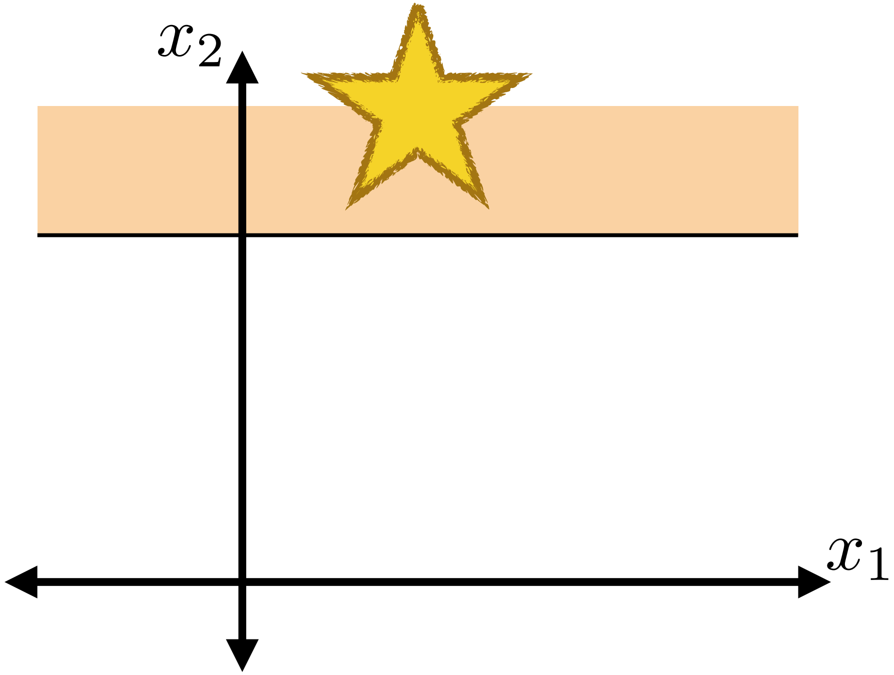
# Regression tree



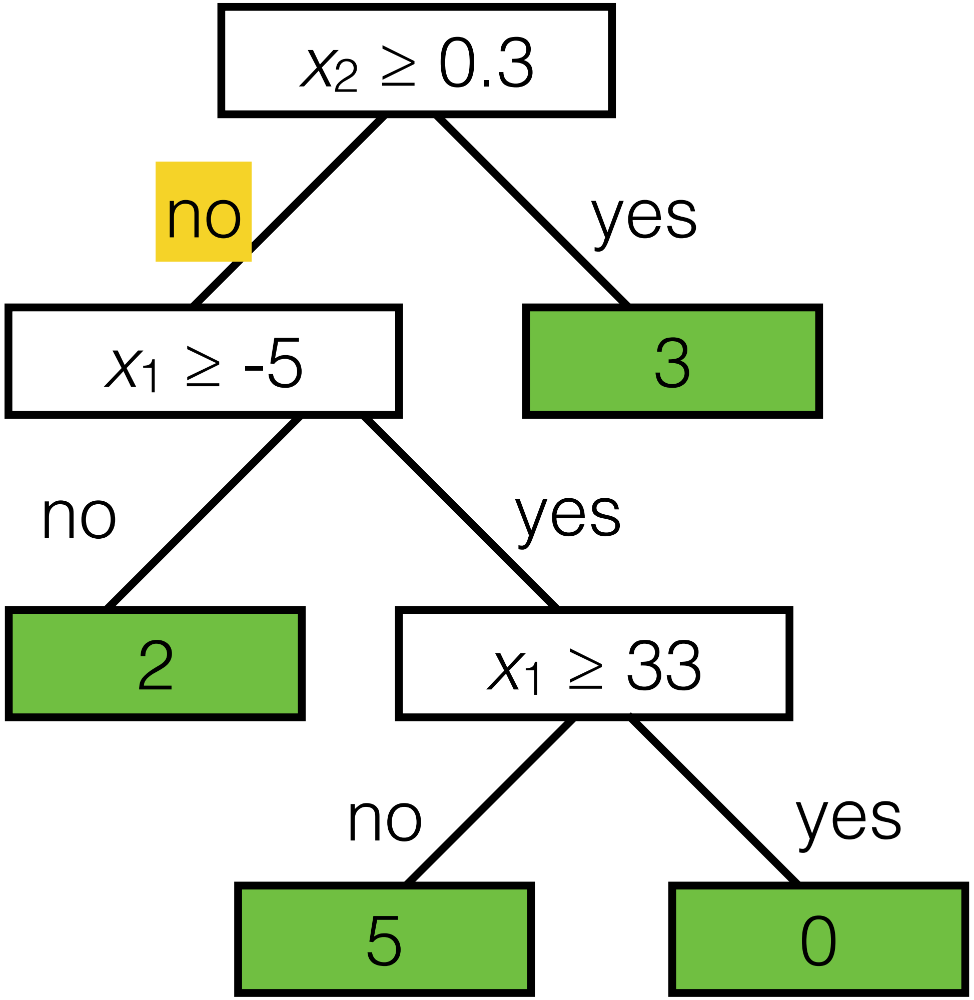
features:
$x_1$: temperature (deg C)
$x_2$: precipitation (cm/hr)

labels:
$y$: km run

- Tree defines an axis-aligned "partition" of the feature space:



# Regression tree



$x_2 \geq 0.3$

no

yes

$x_1 \geq -5$

3

no

yes

2

$x_1 \geq 33$

no

yes

5

0

features:

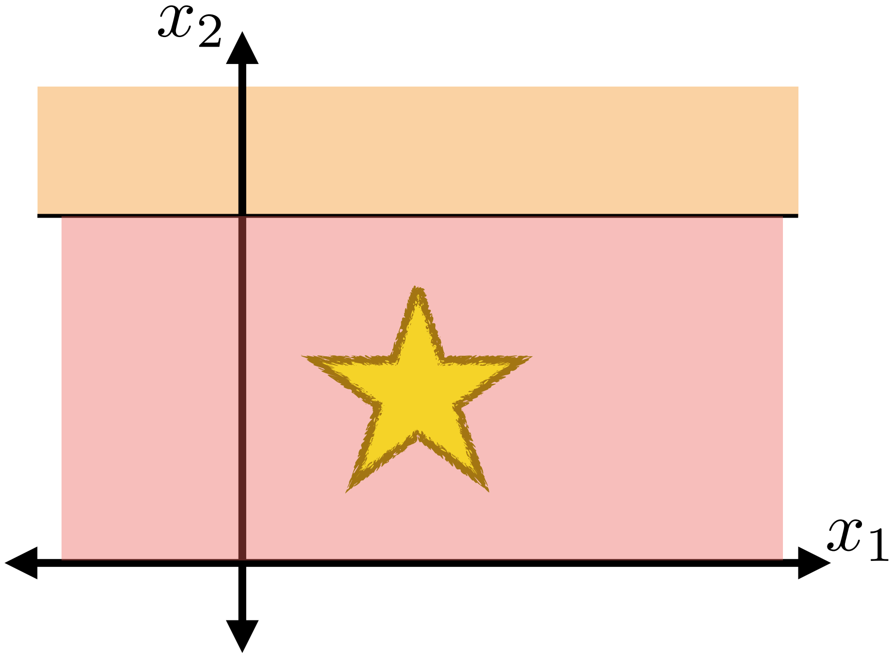
$x_1$: temperature (deg C)

$x_2$: precipitation (cm/hr)

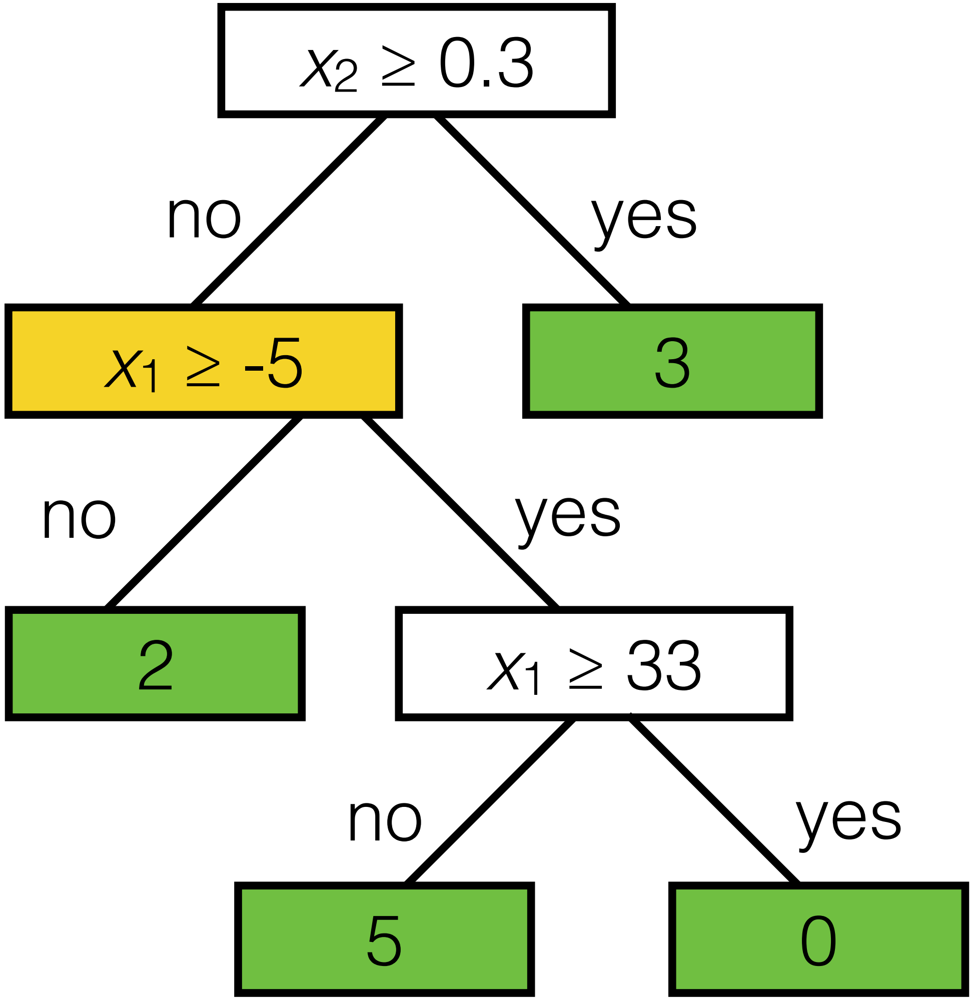
labels:

$y$: km run

- Tree defines an axis-aligned "partition" of the feature space:



# Regression tree



$x_2 \geq 0.3$

no

yes

$x_1 \geq -5$

3

no

yes

2

$x_1 \geq 33$

no

yes

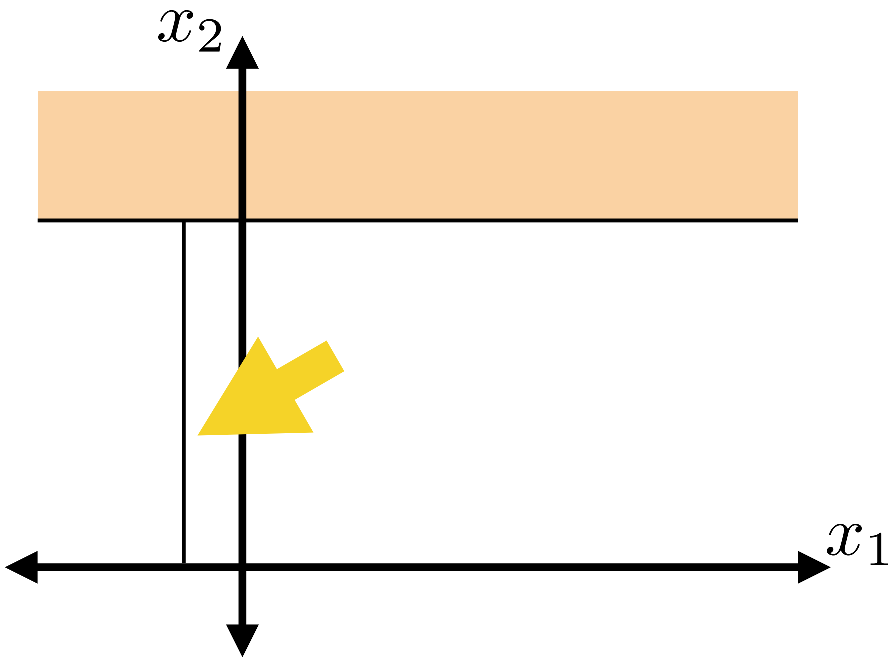
5

0

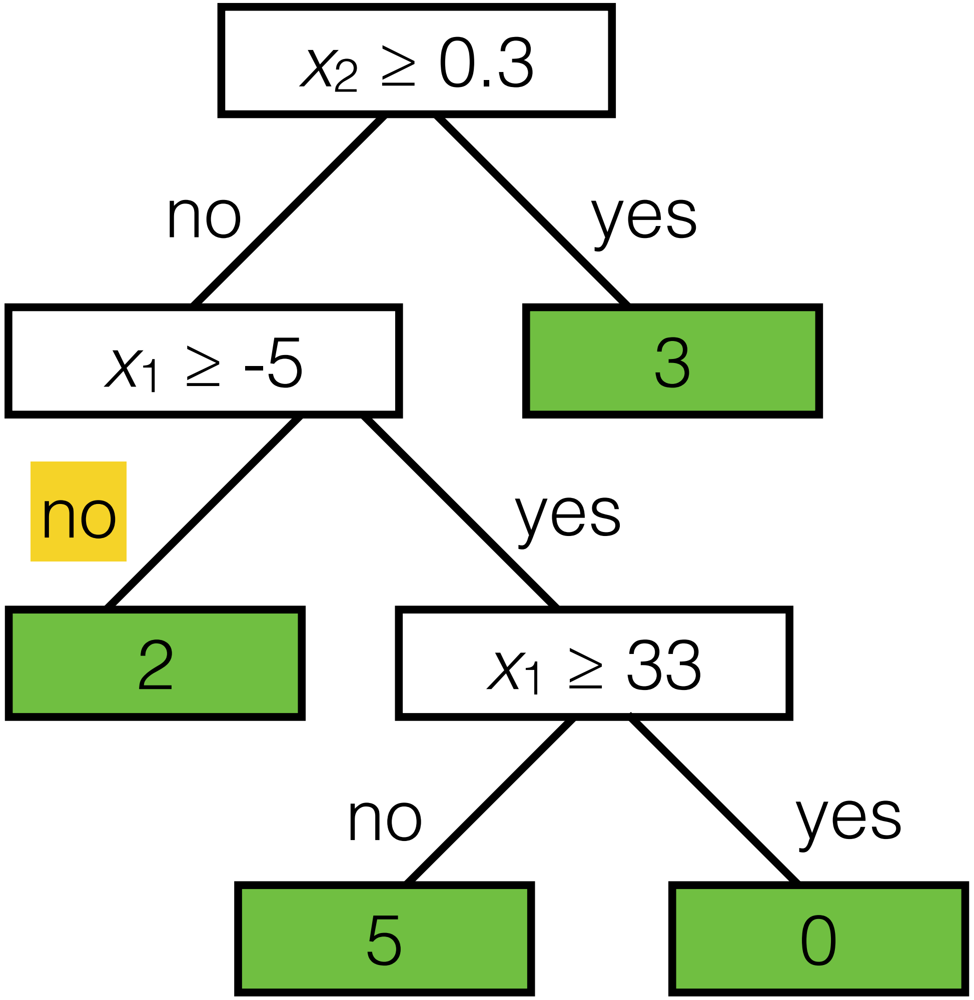
features:
$x_1$: temperature (deg C)
$x_2$: precipitation (cm/hr)

labels:
$y$: km run

- Tree defines an axis-aligned "partition" of the feature space:



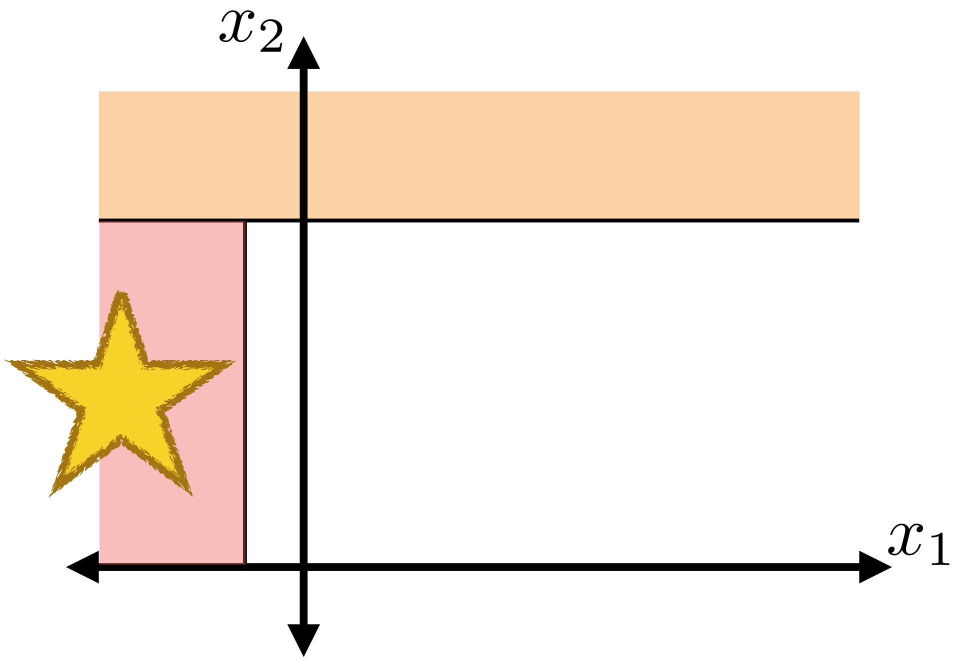
# Regression tree



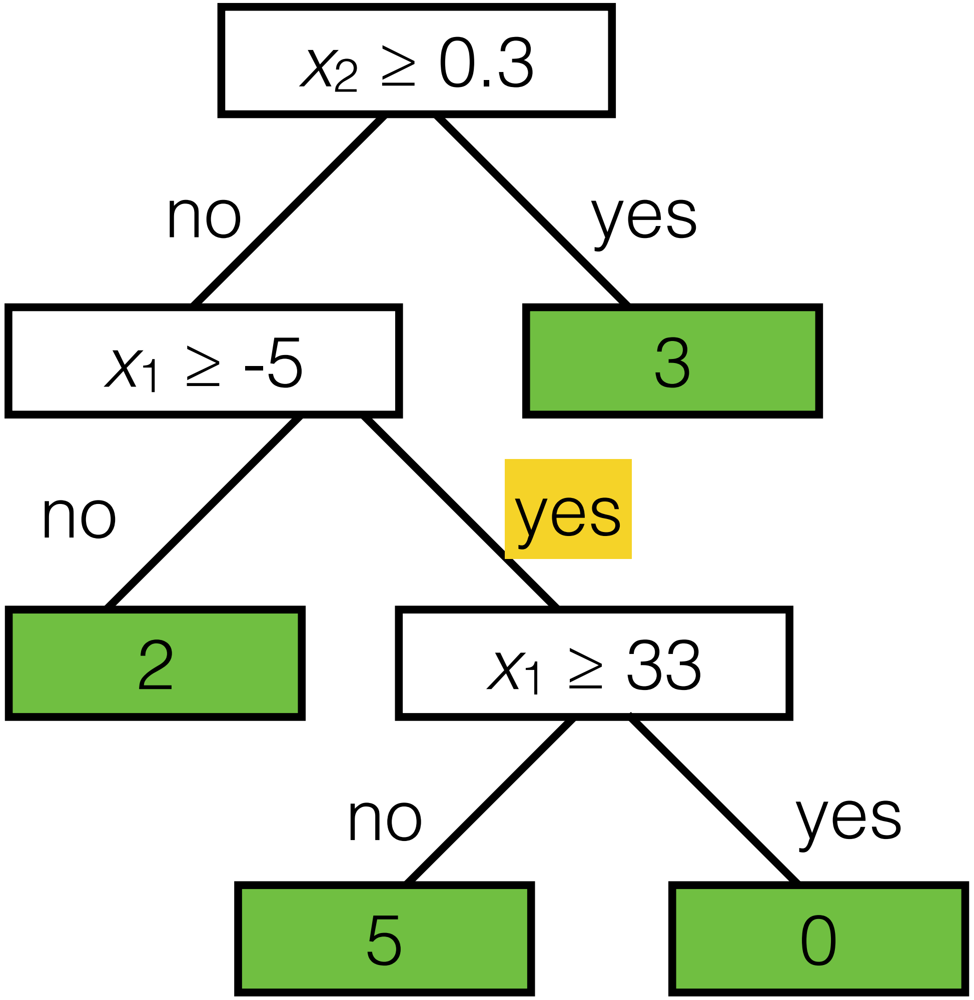
features:
$x_1$: temperature (deg C)
$x_2$: precipitation (cm/hr)

labels:
$y$: km run

- Tree defines an axis-aligned "partition" of the feature space:



# Regression tree



$x_2 \geq 0.3$

no

yes

$x_1 \geq -5$

3

no

yes

2

$x_1 \geq 33$

no

yes

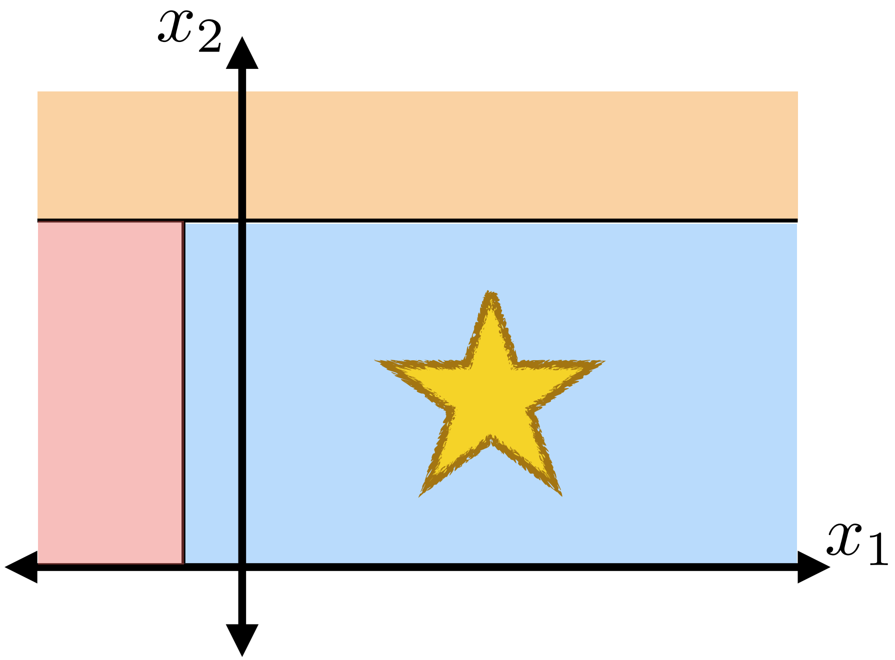
5

0

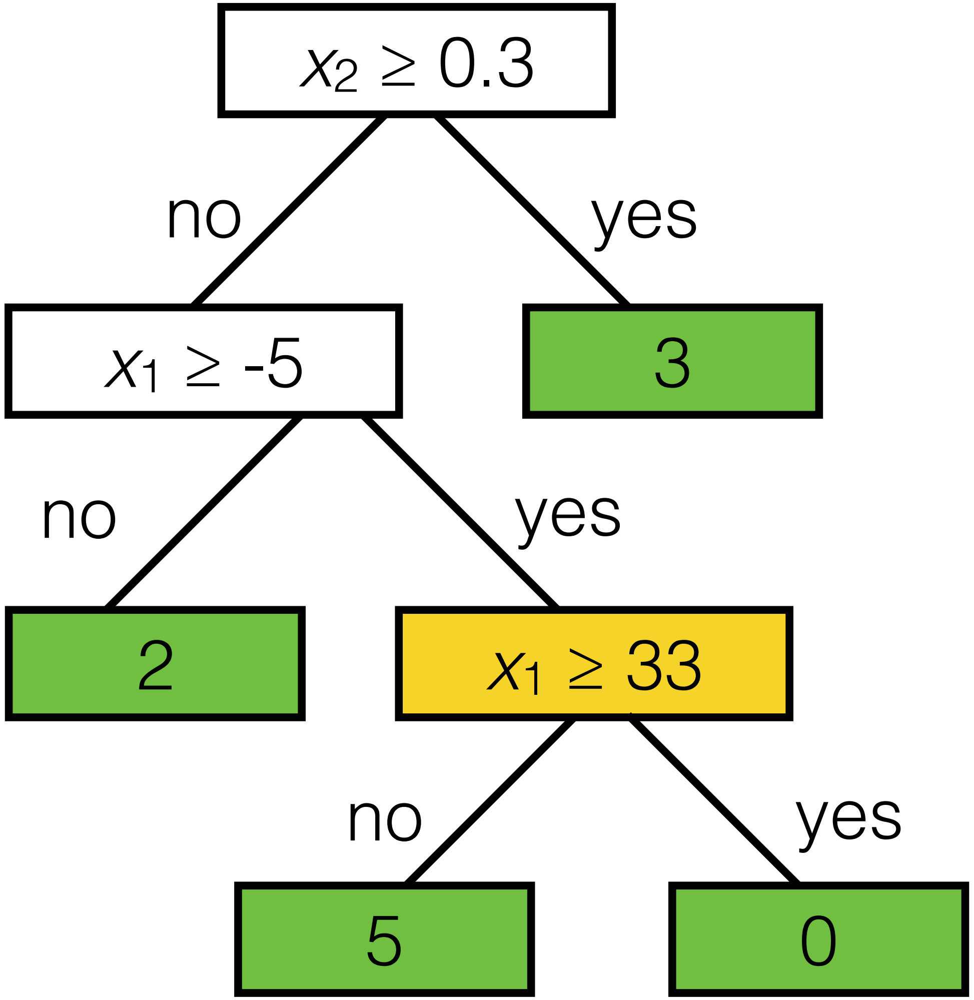
features:
$x_1$: temperature (deg C)
$x_2$: precipitation (cm/hr)

labels:
$y$: km run

- Tree defines an axis-aligned "partition" of the feature space:



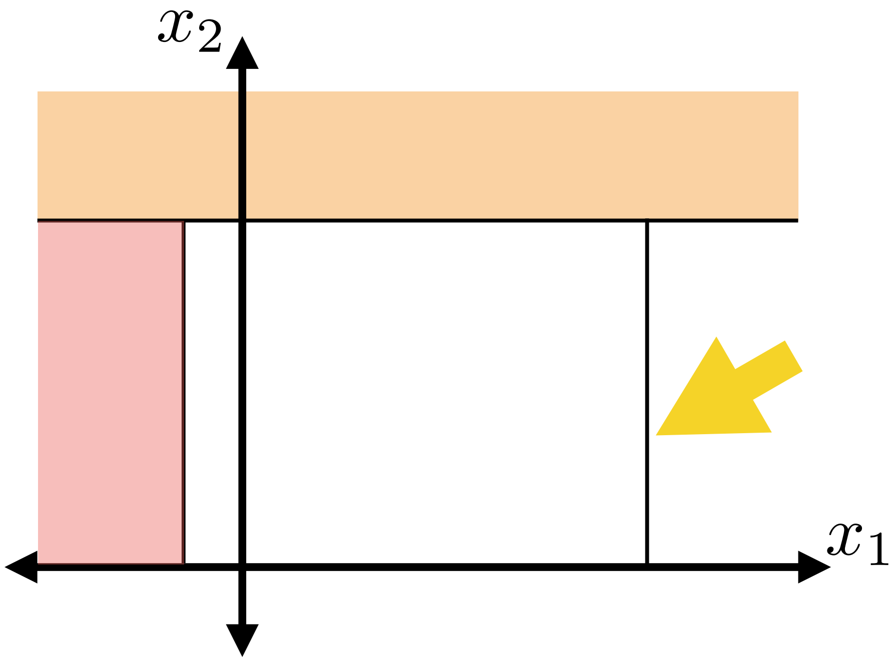
# Regression tree



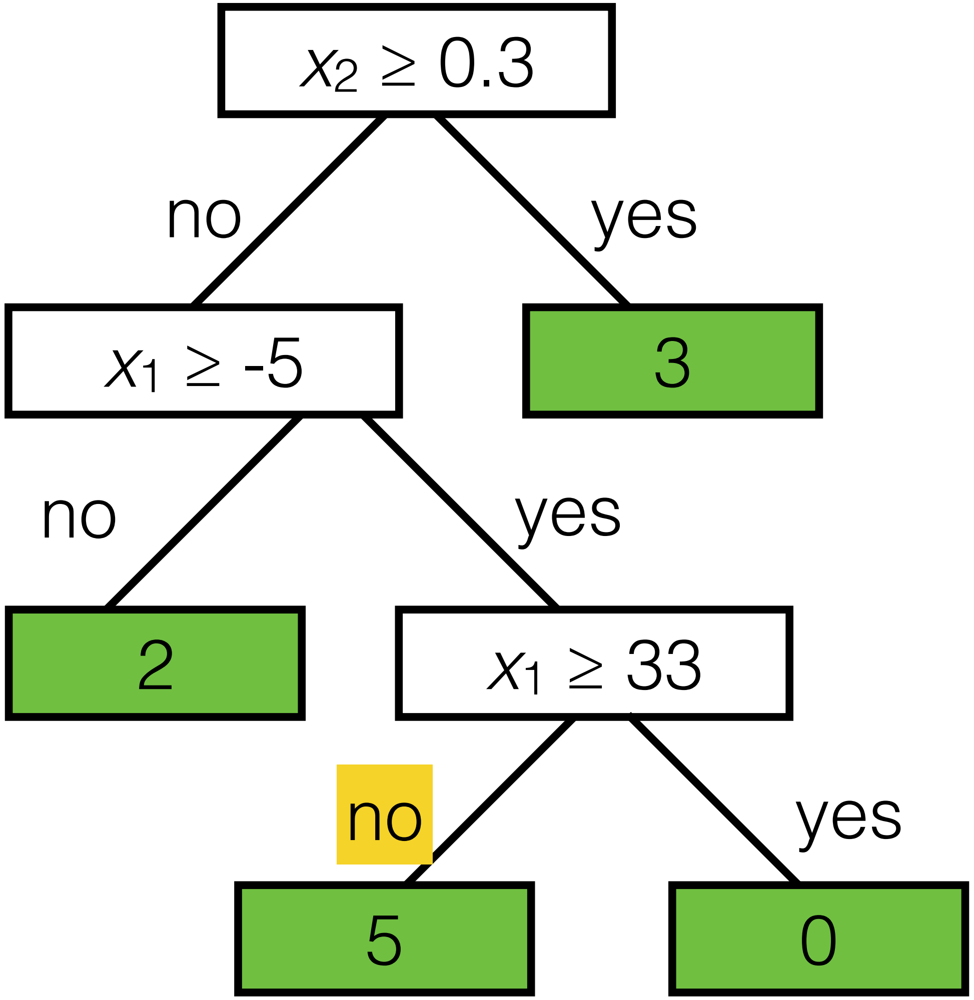
features:
$x_1$: temperature (deg C)
$x_2$: precipitation (cm/hr)

labels:
$y$: km run

- Tree defines an axis-aligned "partition" of the feature space:



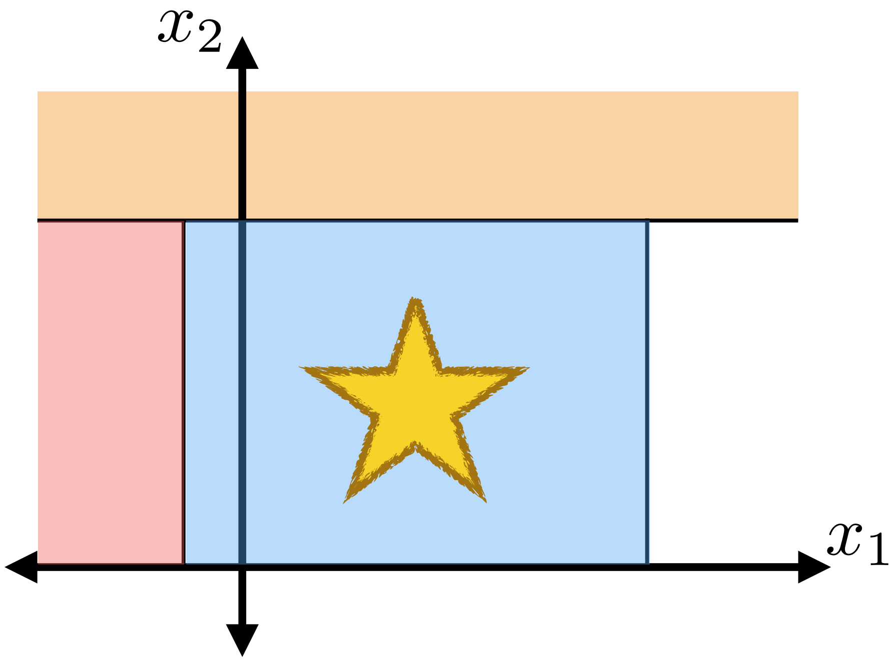
# Regression tree


$x_2 \geq 0.3$
no
yes
$x_1 \geq -5$
3
no
yes
2
$x_1 \geq 33$
no
yes
5
0

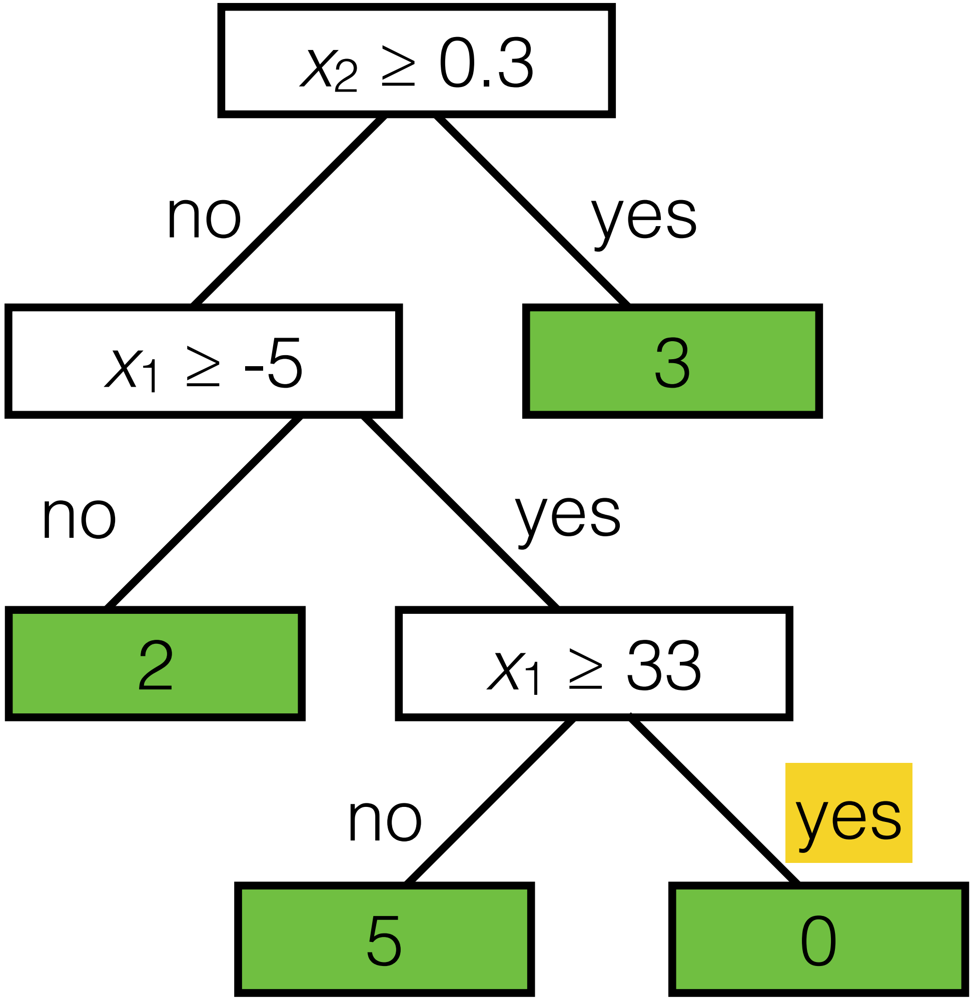
features:
$x_1$: temperature (deg C)
$x_2$: precipitation (cm/hr)

labels:
$y$: km run

- Tree defines an axis-aligned “partition” of the feature space:



# Regression tree



$x_2 \geq 0.3$

no

yes

$x_1 \geq -5$

3

no

yes

2

$x_1 \geq 33$

no

yes

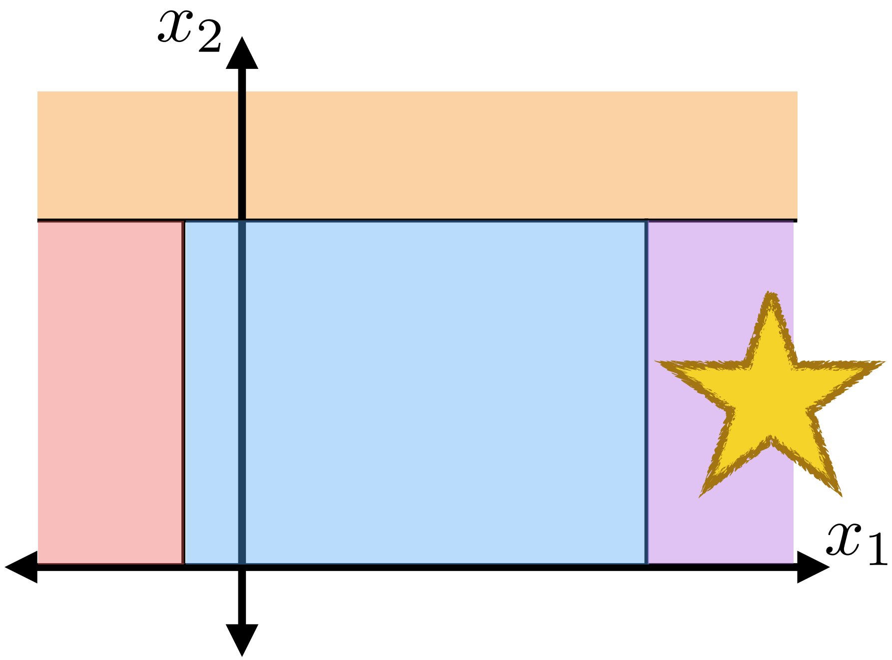
5

0

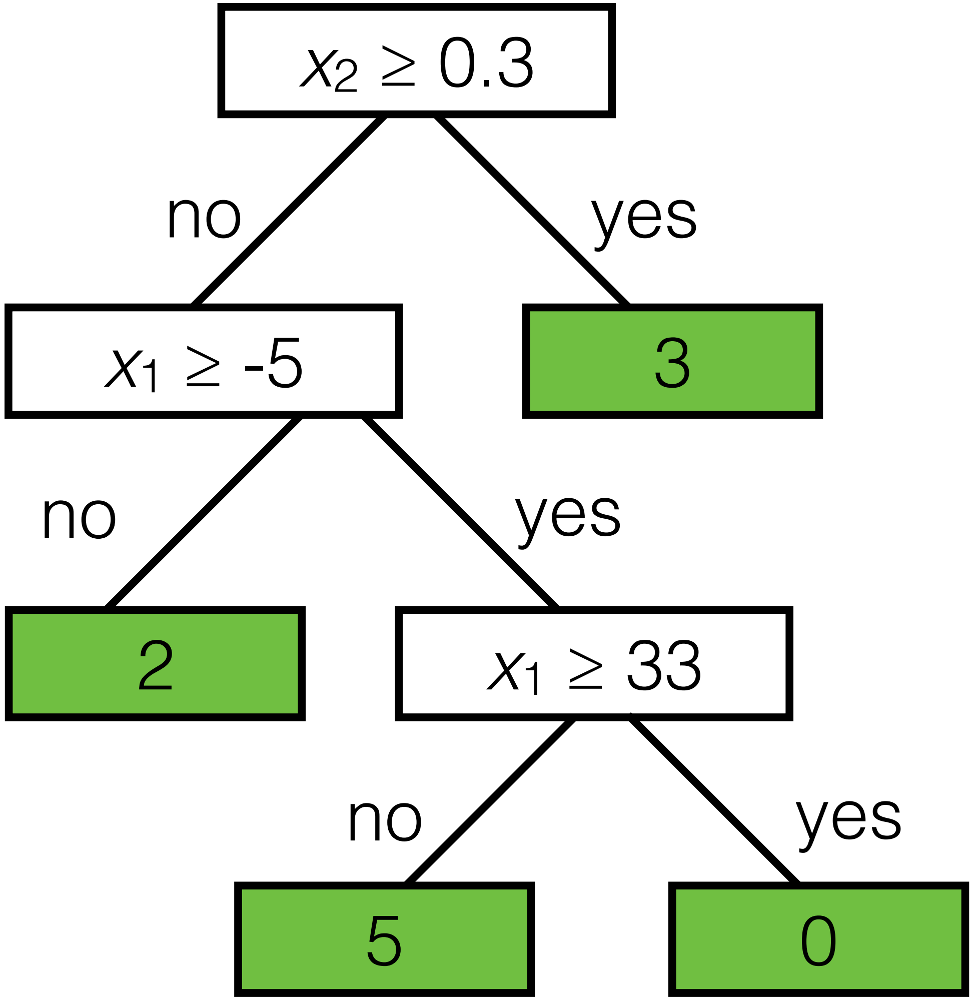
features:
$x_1$: temperature (deg C)
$x_2$: precipitation (cm/hr)

labels:
$y$: km run

- Tree defines an axis-aligned “partition” of the feature space:



# Regression tree



$x_2 \geq 0.3$

no

yes

$x_1 \geq -5$

3

no

yes

2

$x_1 \geq 33$

no

yes

5

0

features:

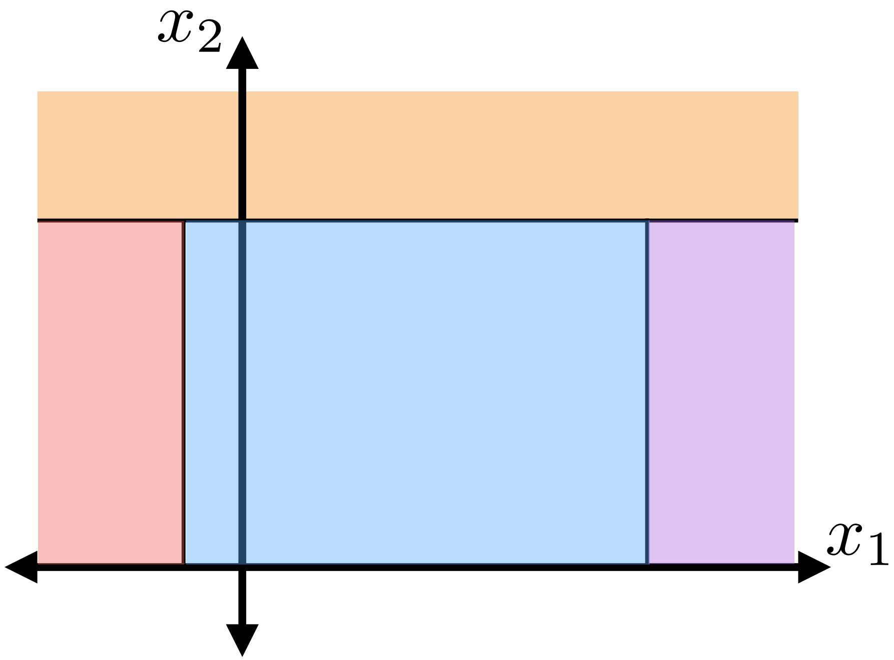
$x_1$: temperature (deg C)

$x_2$: precipitation (cm/hr)

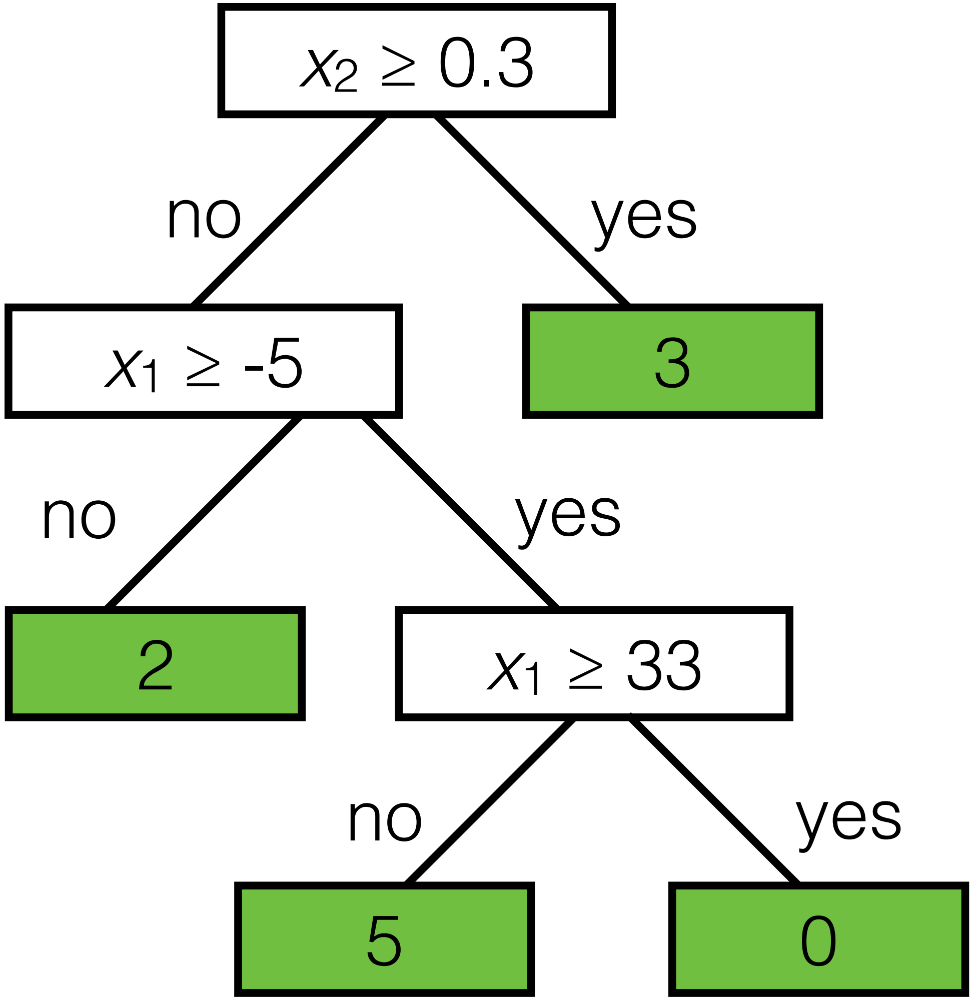
labels:

$y$: km run

- Tree defines an axis-aligned “partition” of the feature space:



# Decision tree



$x_2 \geq 0.3$

no

yes

$x_1 \geq -5$

3

no

yes

2

$x_1 \geq 33$

no

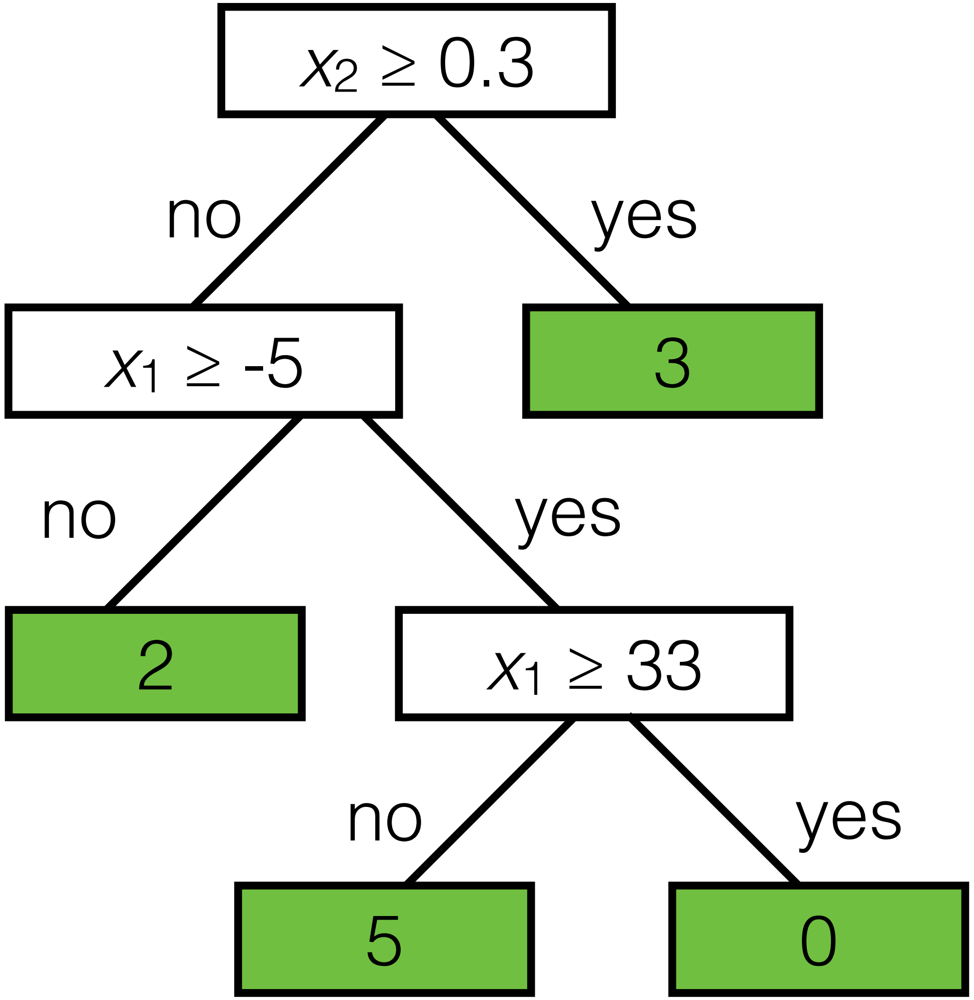
yes

5

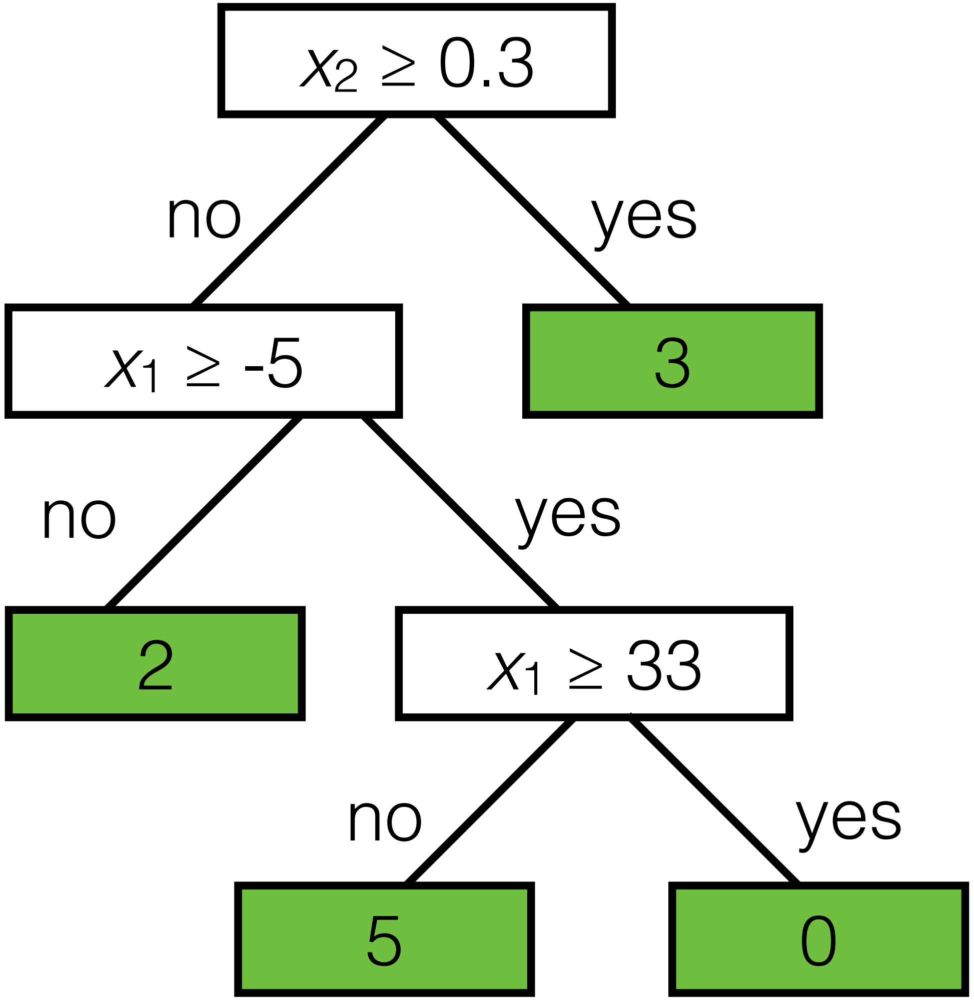
0

# Decision tree

Recall: familiar pattern



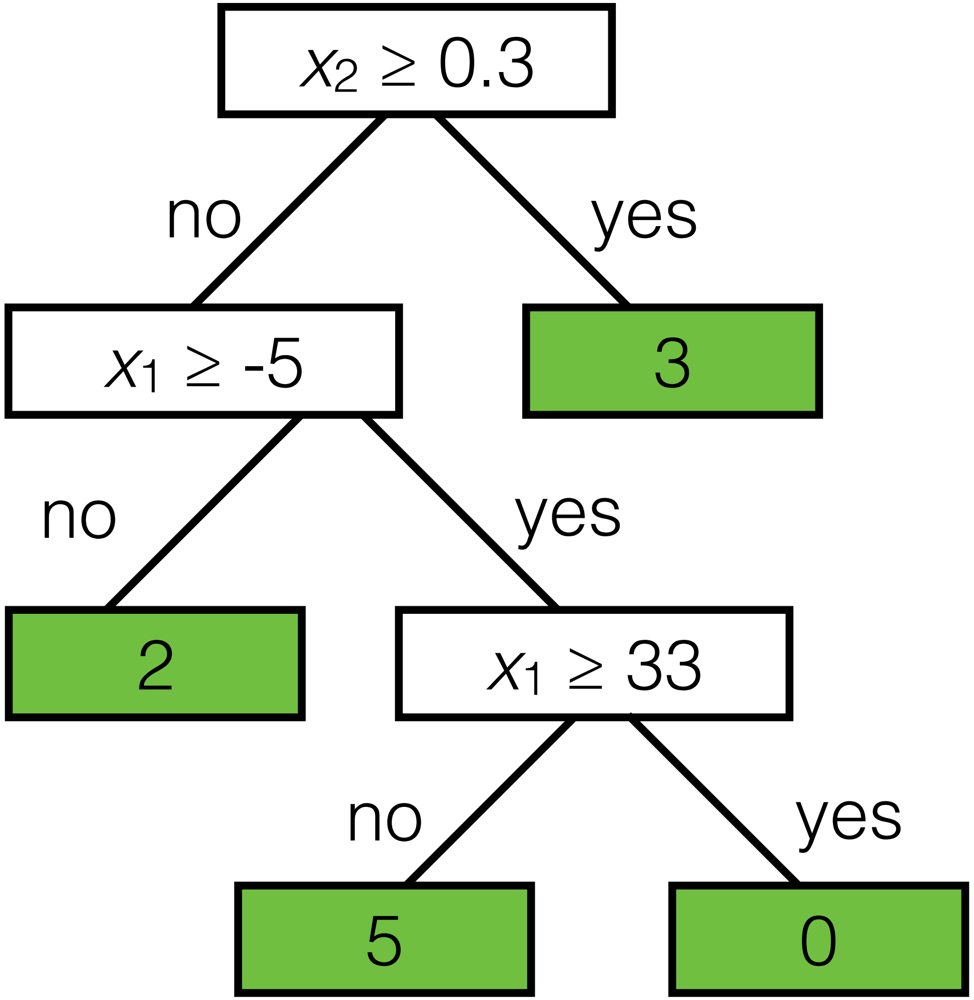
# Decision tree



Recall: familiar pattern

1. Choose how to predict label (given features & parameters)

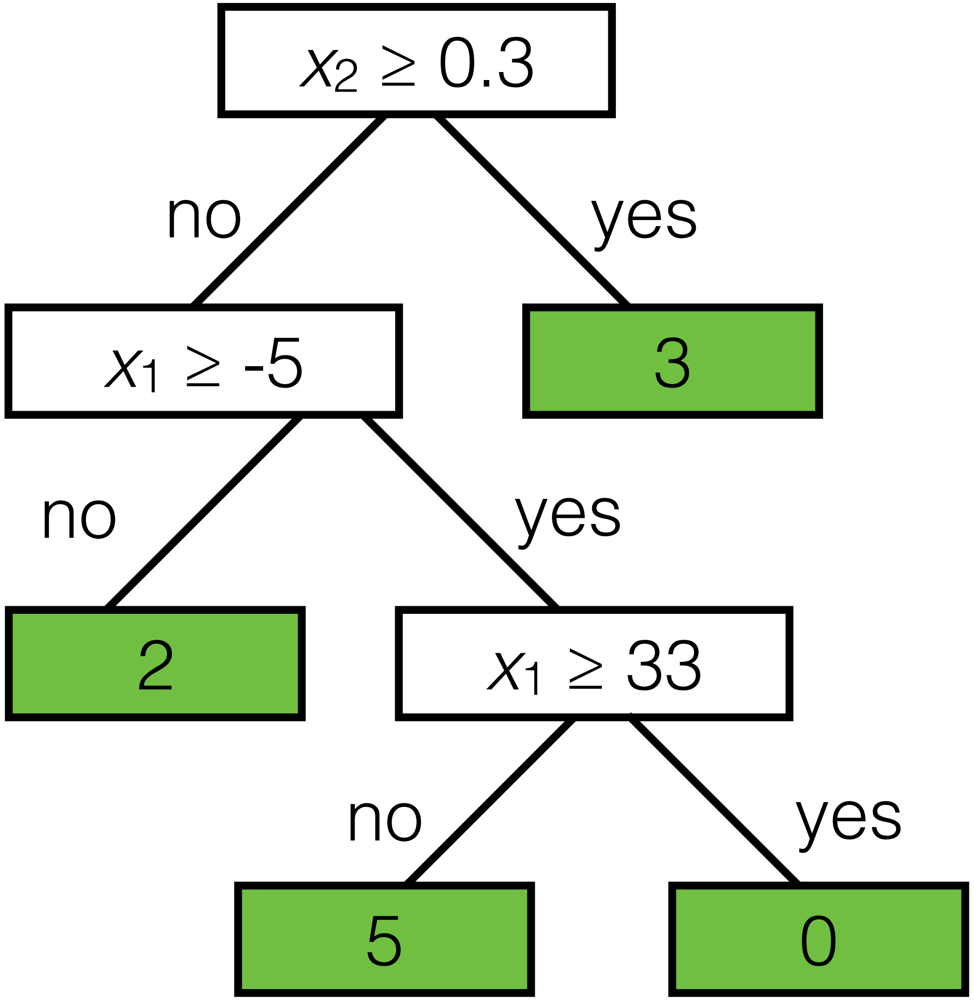
# Decision tree



Recall: familiar pattern

1. Choose how to predict label (given features & parameters)
2. Choose a loss (between guess & actual label)

# Decision tree


$x_2 \geq 0.3$

no

yes

$x_1 \geq -5$

3

no

yes

2

$x_1 \geq 33$

no

yes

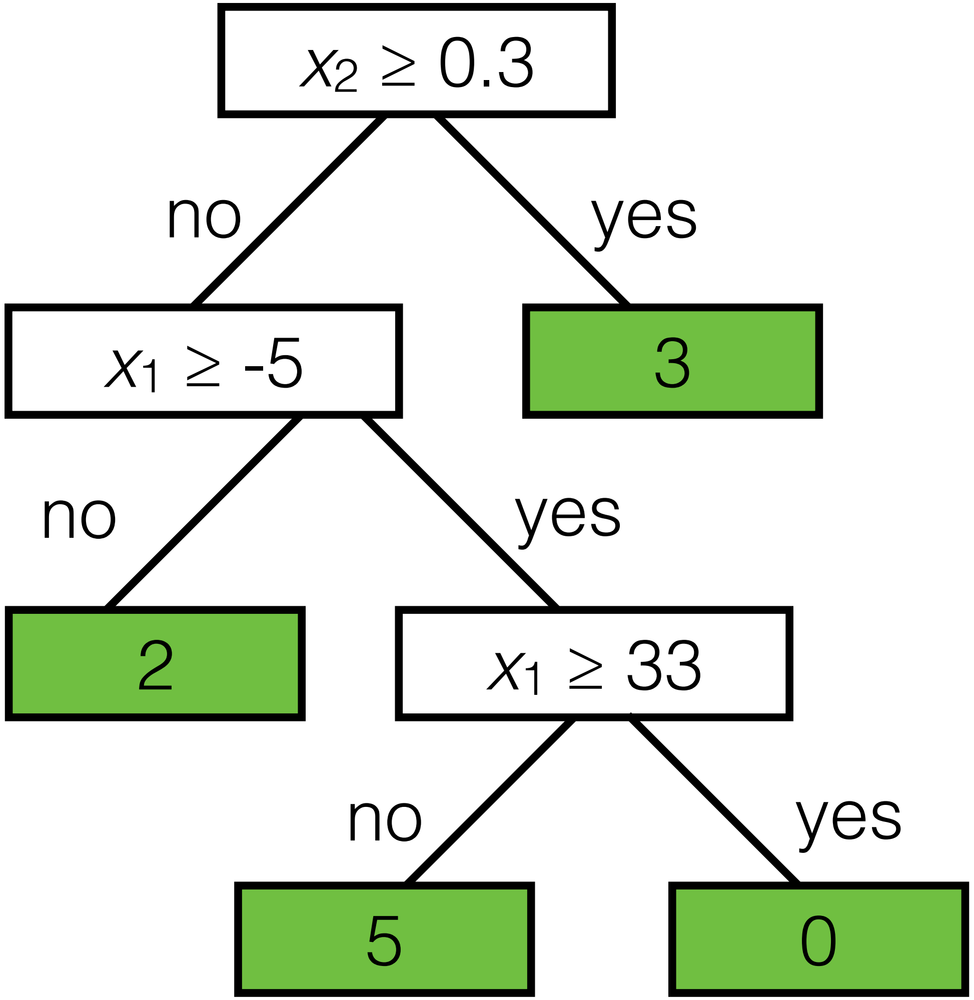
5

0

Recall: familiar pattern

1. Choose how to predict label (given features & parameters)
2. Choose a loss (between guess & actual label)
3. Choose parameters by trying to minimize the training loss

# Decision tree

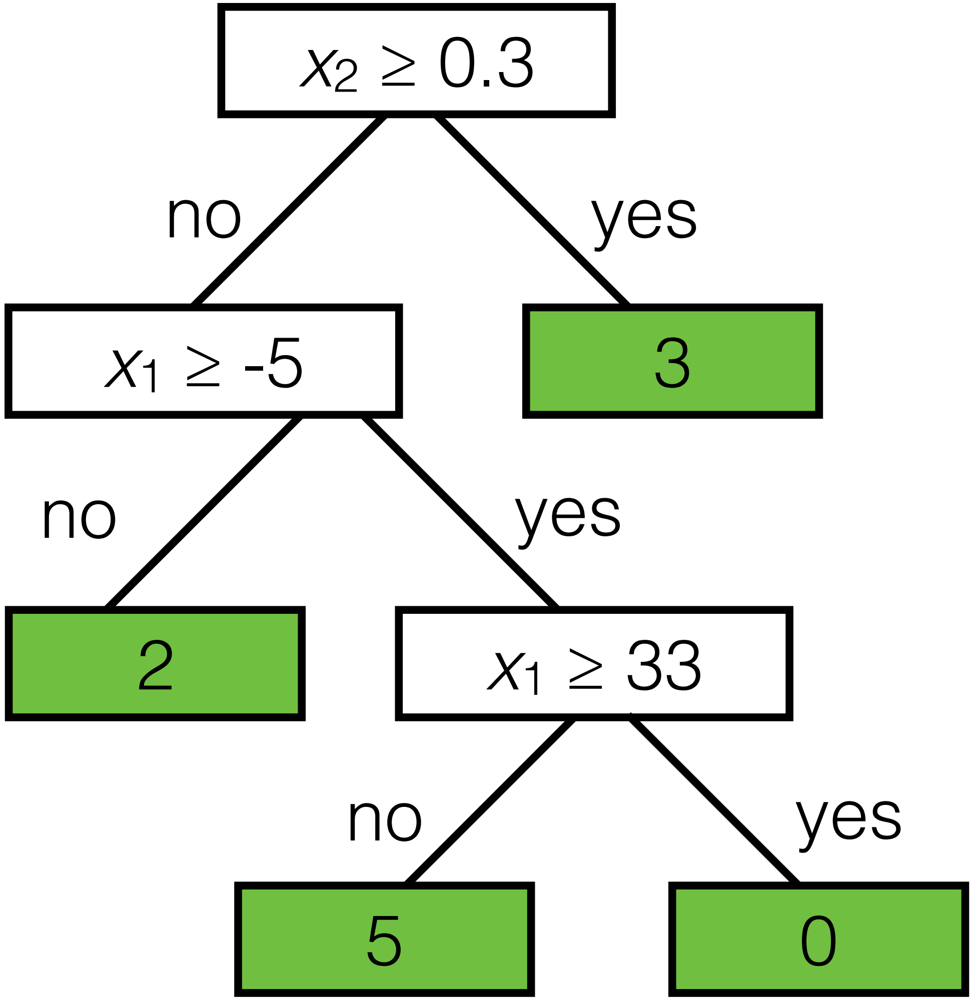


Recall: familiar pattern

1. Choose how to predict label (given features & parameters)
2. Choose a loss (between guess & actual label)
3. Choose parameters by trying to minimize the training loss

- Parameters here:

# Decision tree

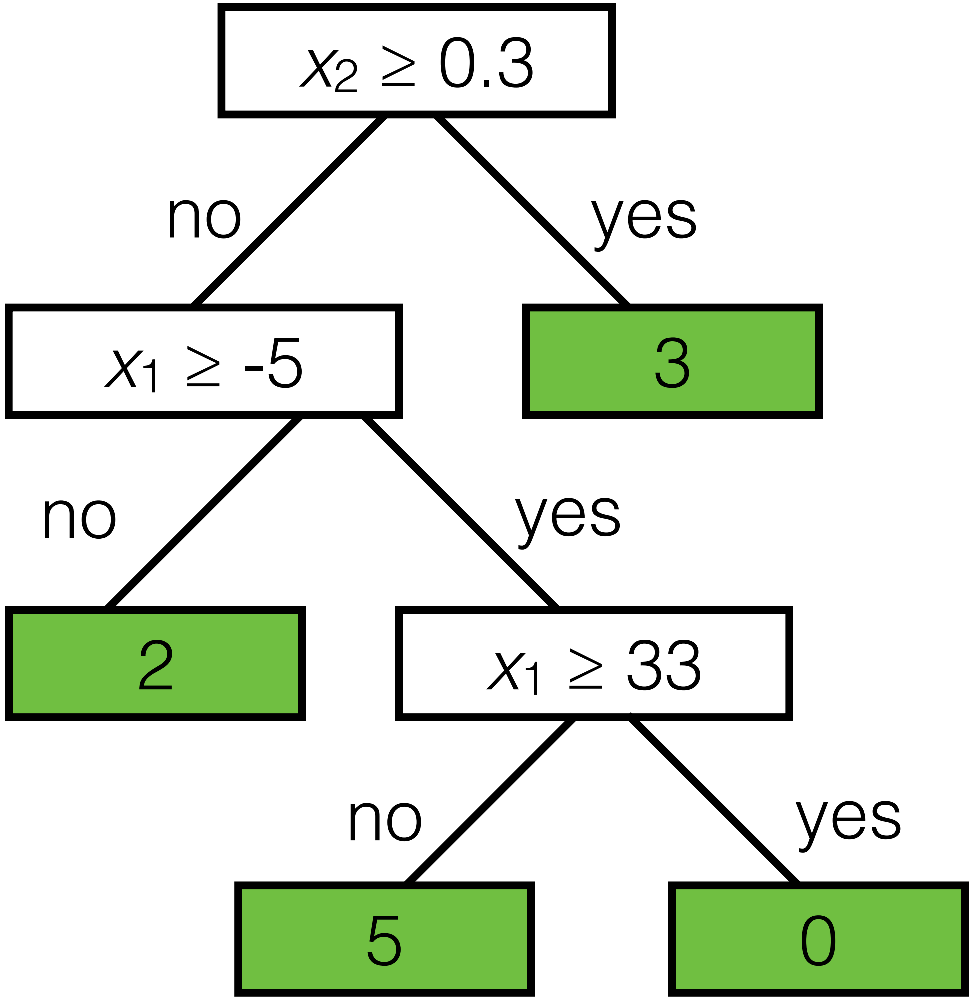


Recall: familiar pattern

1. Choose how to predict label (given features & parameters)
2. Choose a loss (between guess & actual label)
3. Choose parameters by trying to minimize the training loss

- Parameters here:
  - For each internal node:

# Decision tree

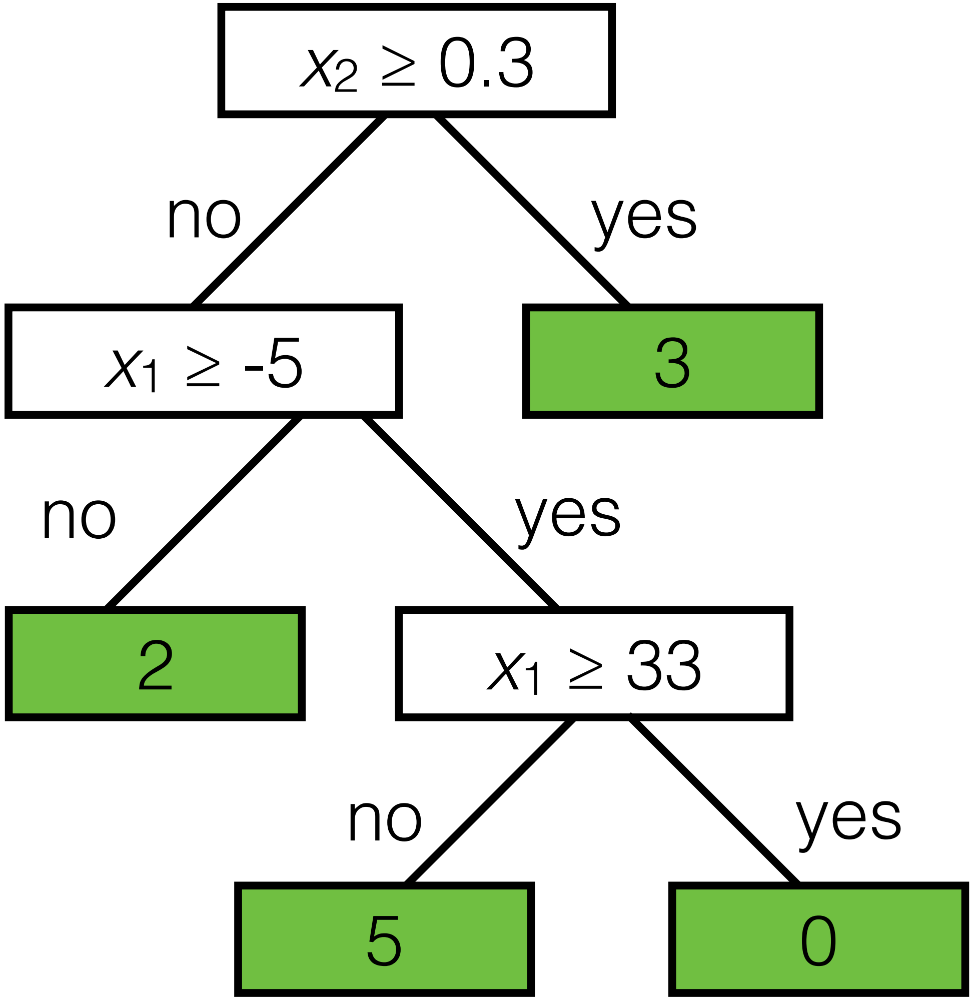


Recall: familiar pattern

1. Choose how to predict label (given features & parameters)
2. Choose a loss (between guess & actual label)
3. Choose parameters by trying to minimize the training loss

- Parameters here:
  - For each internal node:
    - split dimension

# Decision tree

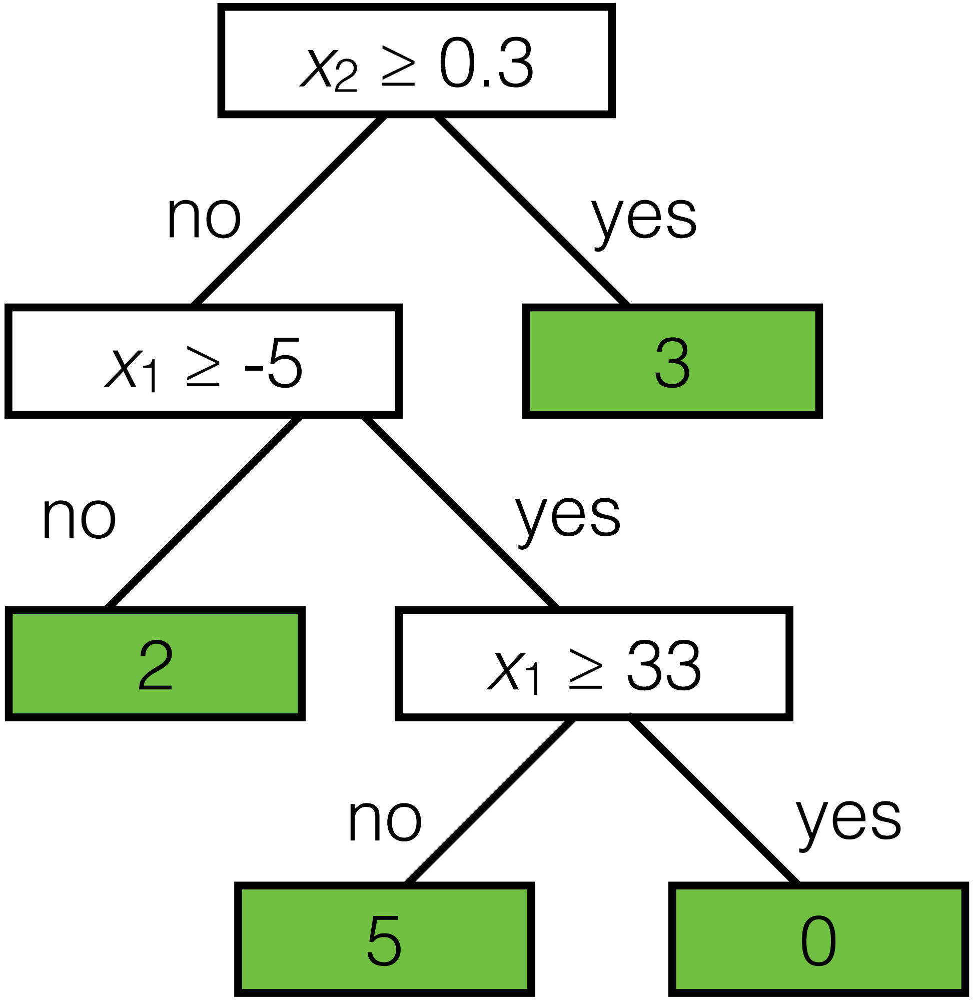


Recall: familiar pattern

1. Choose how to predict label (given features & parameters)
2. Choose a loss (between guess & actual label)
3. Choose parameters by trying to minimize the training loss

- Parameters here:
  - For each internal node:
    - split dimension
    - split value

# Decision tree

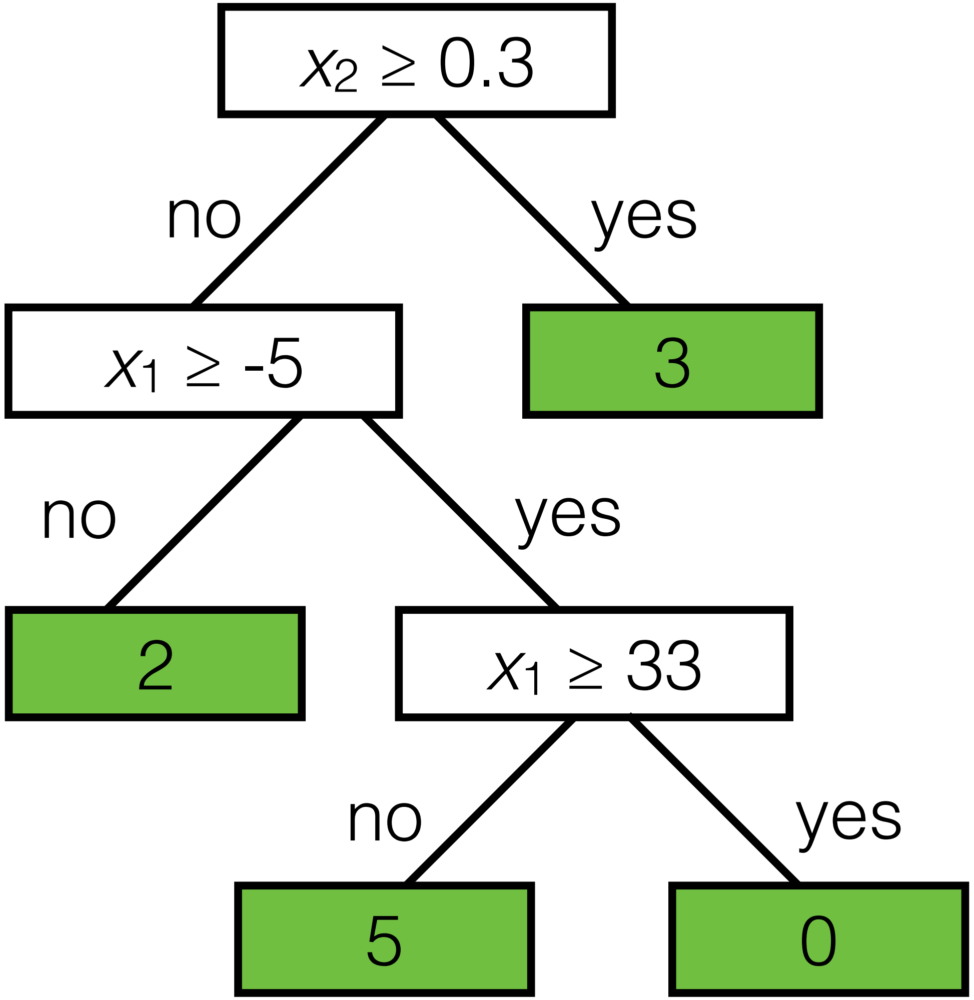


Recall: familiar pattern

1. Choose how to predict label (given features & parameters)
2. Choose a loss (between guess & actual label)
3. Choose parameters by trying to minimize the training loss

- Parameters here:
  - For each internal node:
    - split dimension
    - split value
    - child nodes

# Decision tree

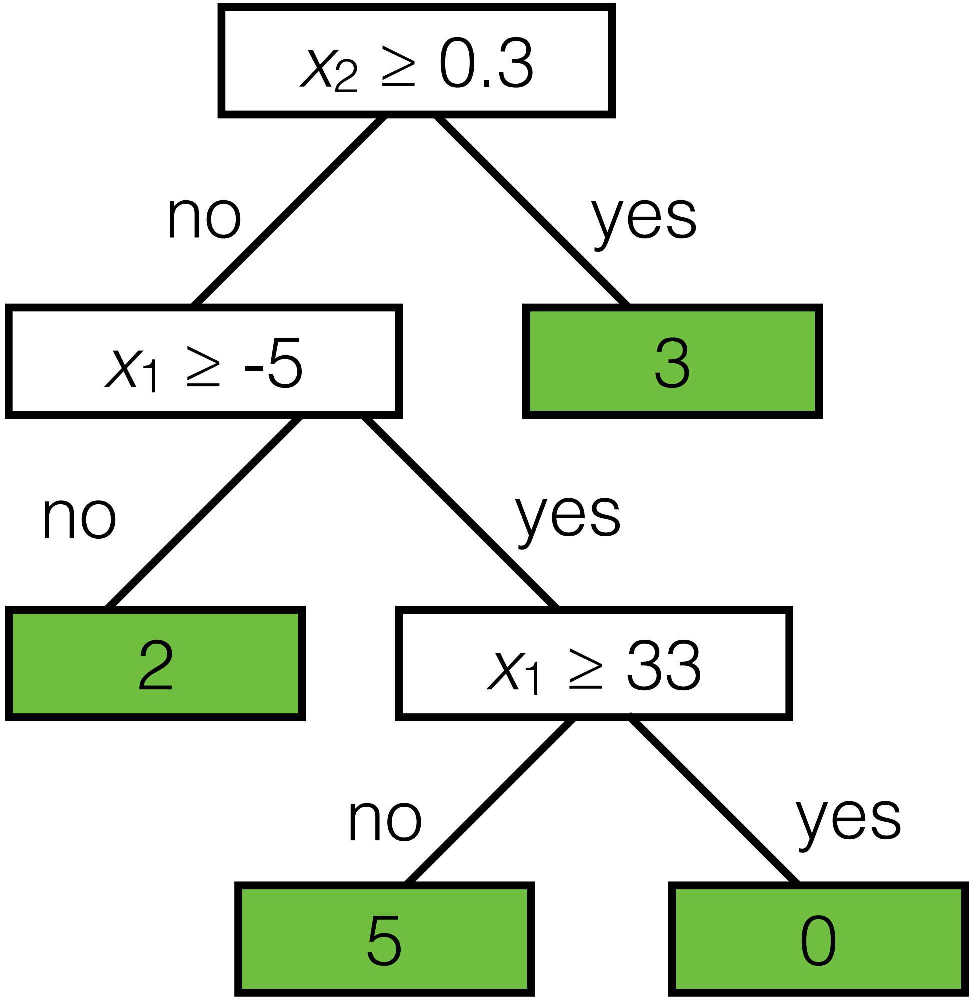


Recall: familiar pattern

1. Choose how to predict label (given features & parameters)
2. Choose a loss (between guess & actual label)
3. Choose parameters by trying to minimize the training loss

- Parameters here:
  - For each internal node:
    - split dimension
    - split value
    - child nodes
  - For each leaf node:
    - label

# Decision tree



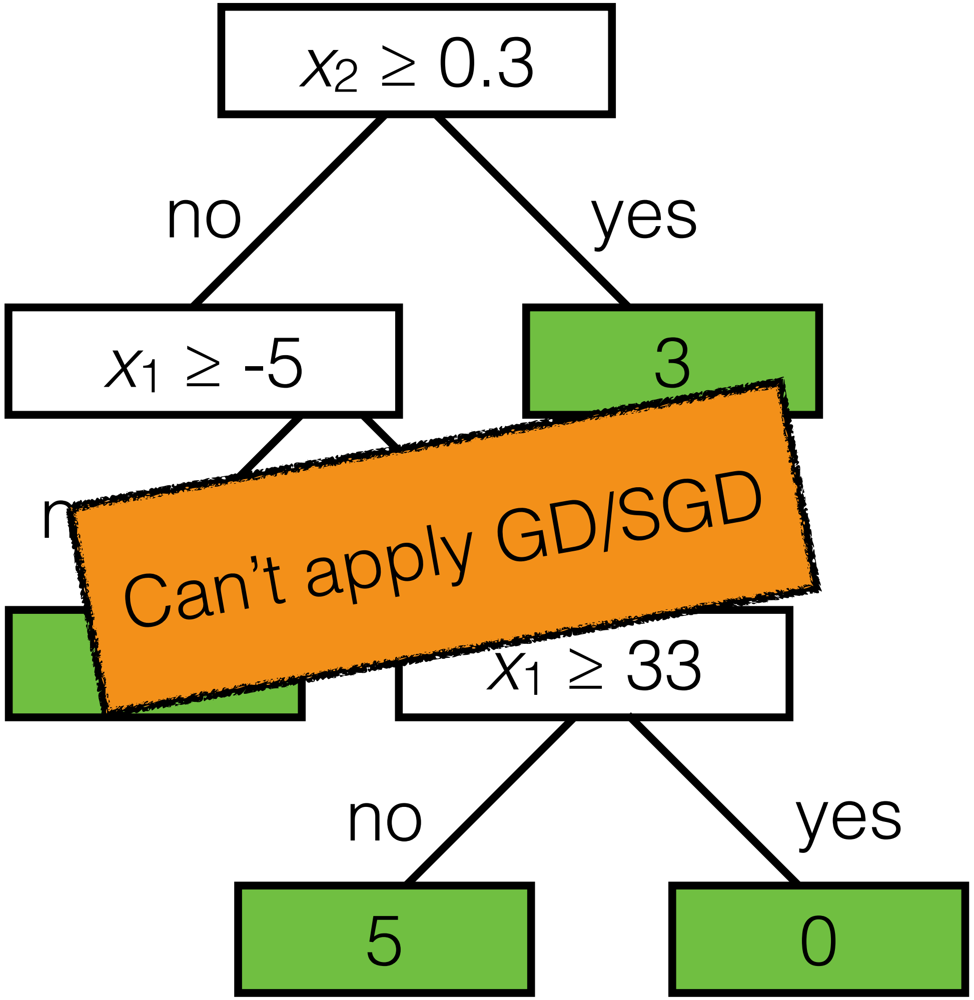
- Note: parameters here don’t have a fixed dimension

Recall: familiar pattern

1. Choose how to predict label (given features & parameters)
2. Choose a loss (between guess & actual label)
3. Choose parameters by trying to minimize the training loss

- Parameters here:
  - For each internal node:
    - split dimension
    - split value
    - child nodes
  - For each leaf node:
    - label

# Decision tree



Can't apply GD/SGD

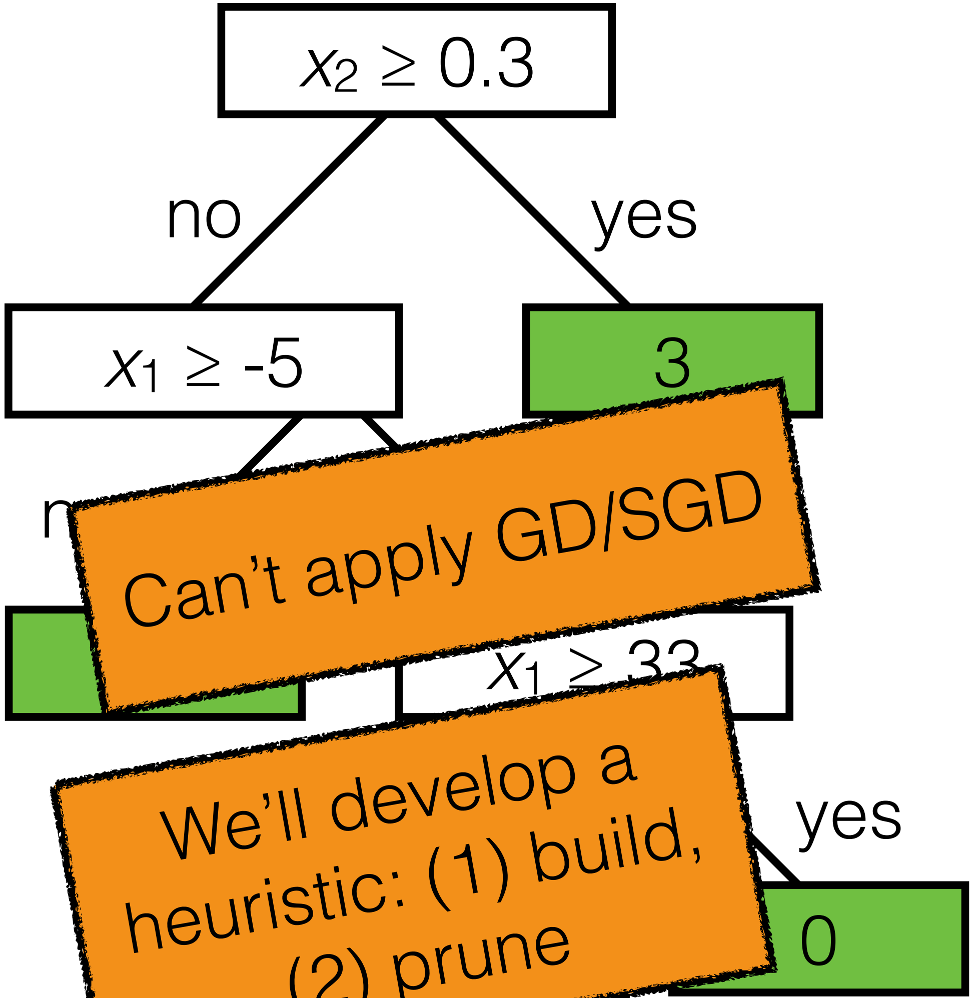
- Note: parameters here don't have a fixed dimension

Recall: familiar pattern

1. Choose how to predict label (given features & parameters)
2. Choose a loss (between guess & actual label)
3. Choose parameters by trying to minimize the training loss

- Parameters here:
  - For each internal node:
    - split dimension
    - split value
    - child nodes
  - For each leaf node:
    - label

# Decision tree



Can't apply GD/SGD

We'll develop a heuristic: (1) build, (2) prune

- Note: parameters here don't have a fixed dimension

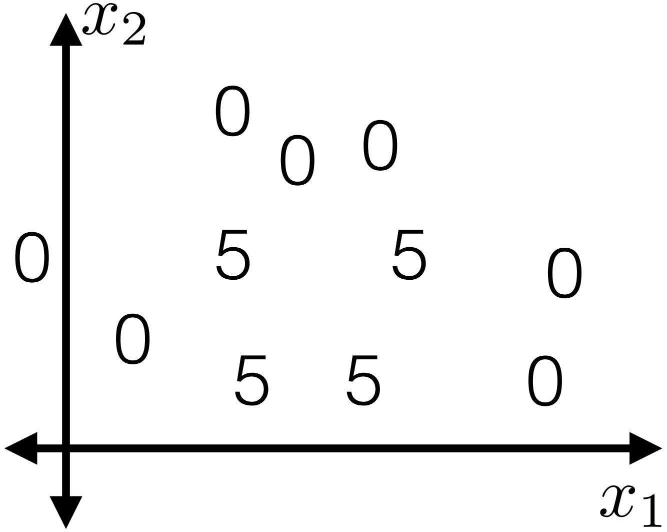
Recall: familiar pattern

1. Choose how to predict label (given features & parameters)
2. Choose a loss (between guess & actual label)
3. Choose parameters by trying to minimize the training loss

- Parameters here:
  - For each internal node:
    - split dimension
    - split value
    - child nodes
  - For each leaf node:
    - label

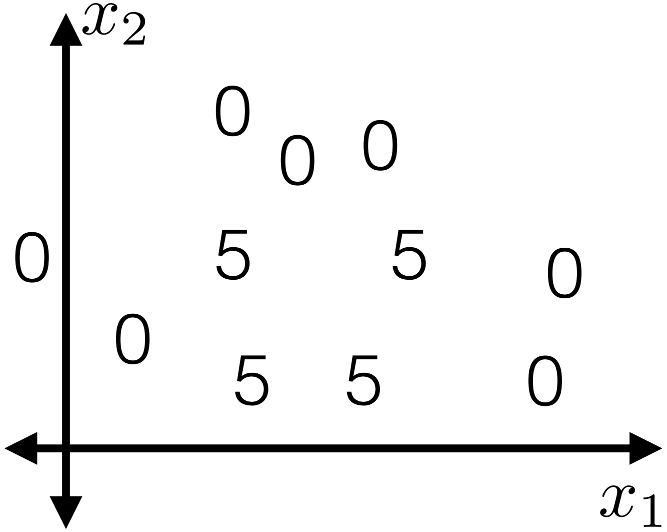
# Building a decision tree

# Building a decision tree



# Building a decision tree

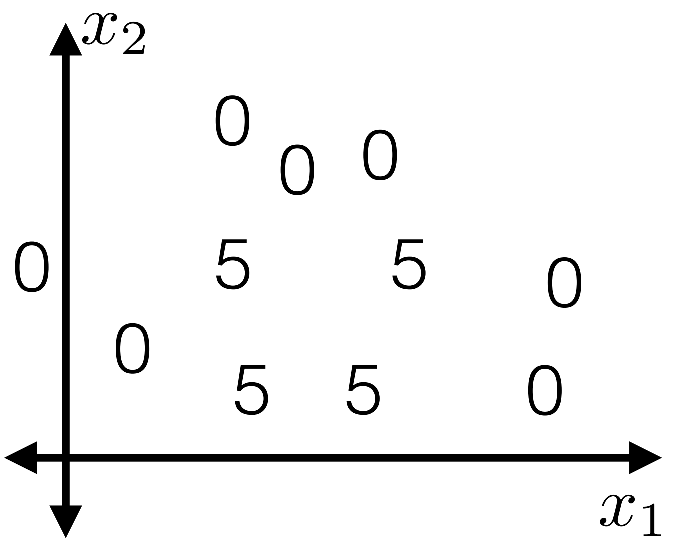
- Regression tree with squared error loss



# Building a decision tree

- Regression tree with squared error loss

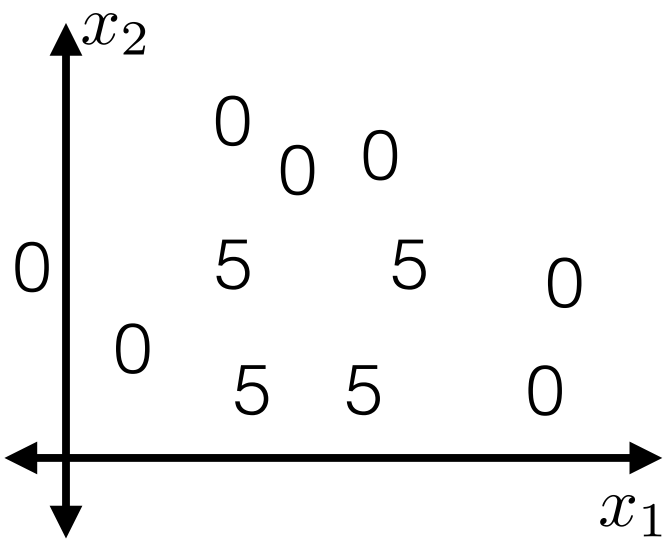
`BuildTree`



# Building a decision tree

- Regression tree with squared error loss

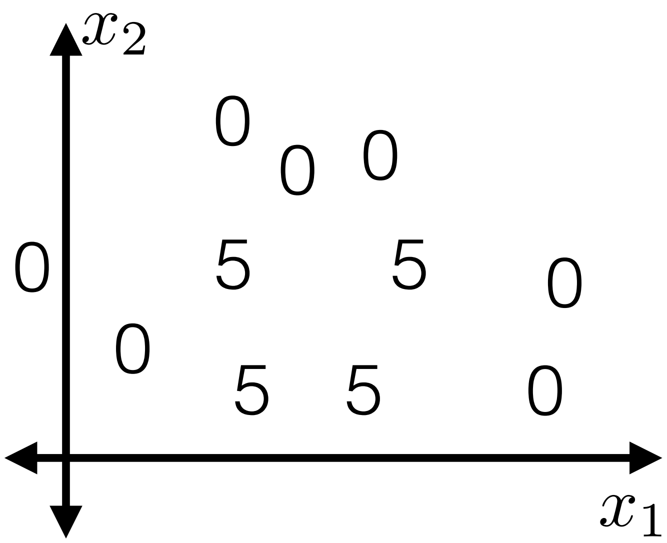
```
BuildTree($I$; $k$)
```



# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
```

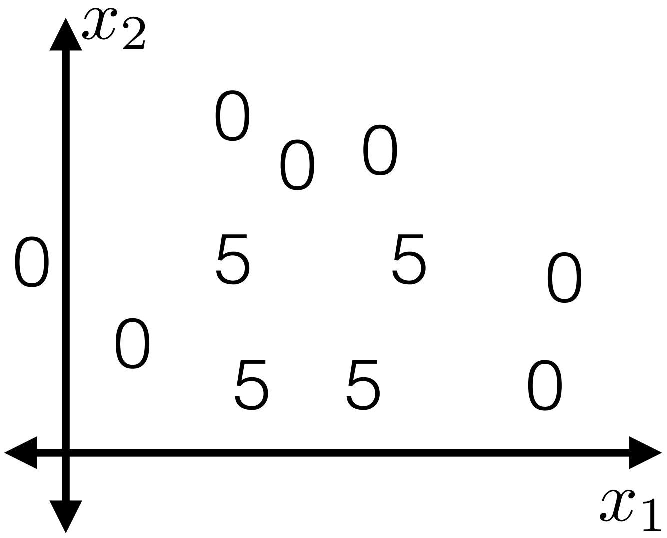


`BuildTree({1,…,n};2)`

# Building a decision tree

- Regression tree with squared error loss

`BuildTree(`$I$`;`$k$`)`

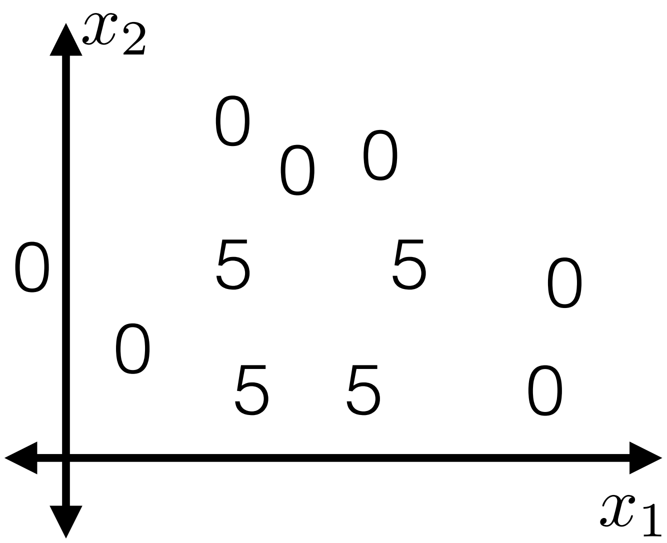


`BuildTree({1,…,n};2)`

# Building a decision tree

- Regression tree with squared error loss

`BuildTree(`$I$`;`$k$`)`

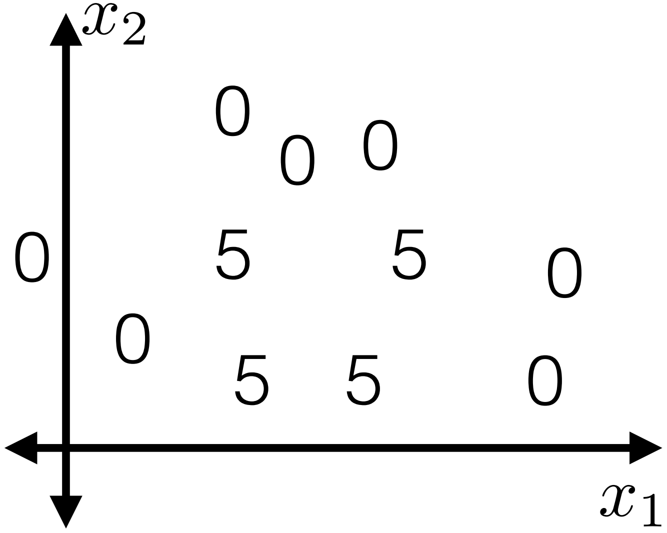


`BuildTree({1,…,n};2)`

# Building a decision tree

- Regression tree with squared error loss

`BuildTree(`$I$`;`$k$`)`

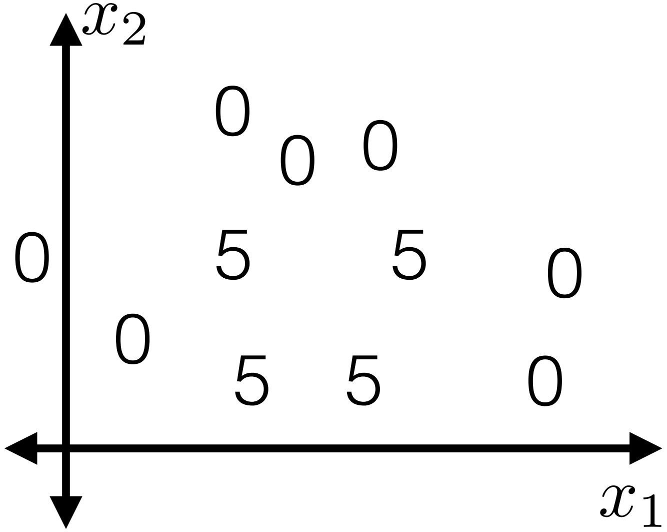


`BuildTree({1,…,n};2)`

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| ≤ k
```

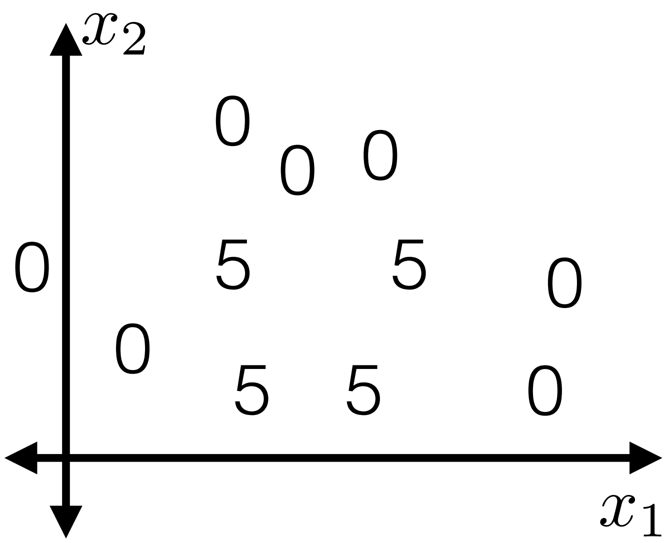


BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| $\leq$ k
```

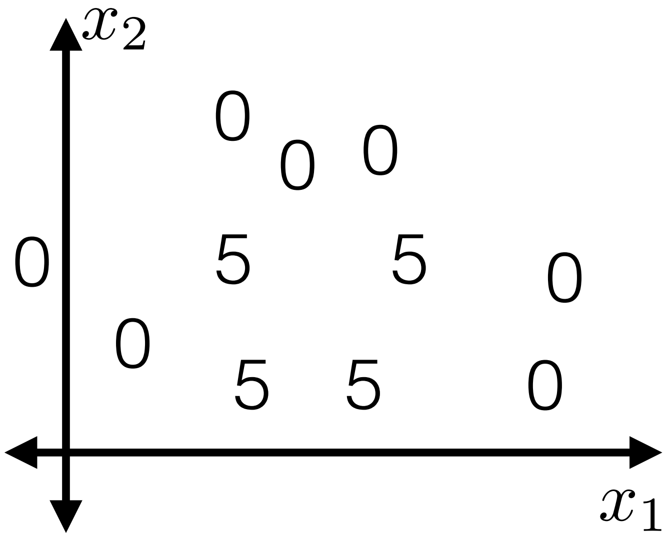


BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| ≤ k
```

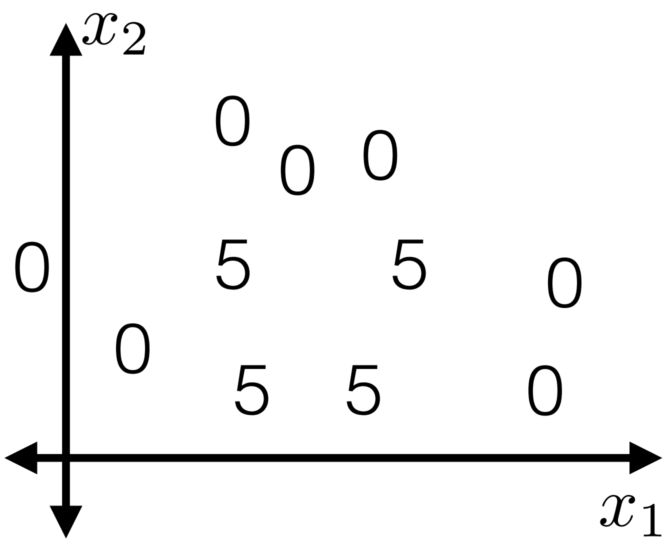


`BuildTree({1,…,n};2)`

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| $\leq$ k
    Set $\hat{y}$ =
```

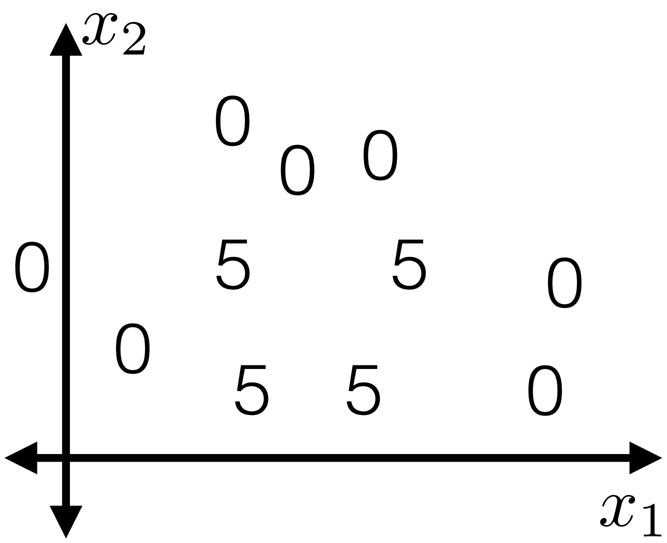


BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| ≤ k
    Set $\hat{y}$ =
```



`BuildTree({1,…,n};2)`

Want to minimize loss/error:

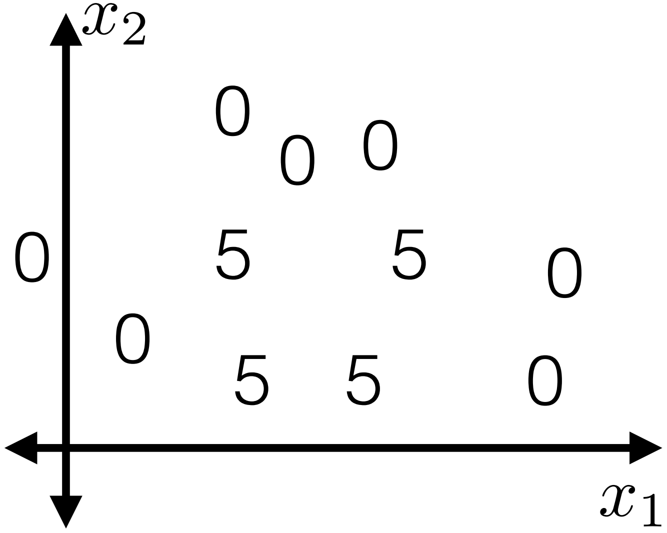
$E = \sum_{i\in I}(y^{(i)} - \hat{y})^2$

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| $\leq$ k
    Set $\hat{y}$ = $\text{average}_{i\in I} y^{(i)}$
```

Want to minimize loss/error: $E = \sum_{i\in I}(y^{(i)} - \hat{y})^2$

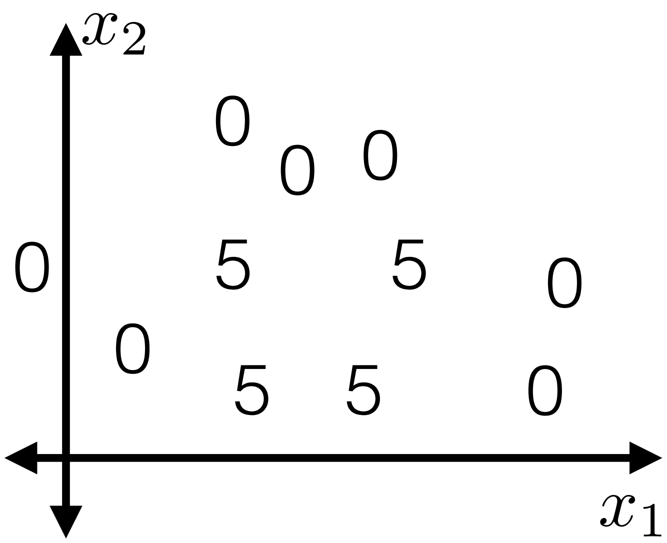


`BuildTree({1,…,n};2)`

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| $\leq$ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
```

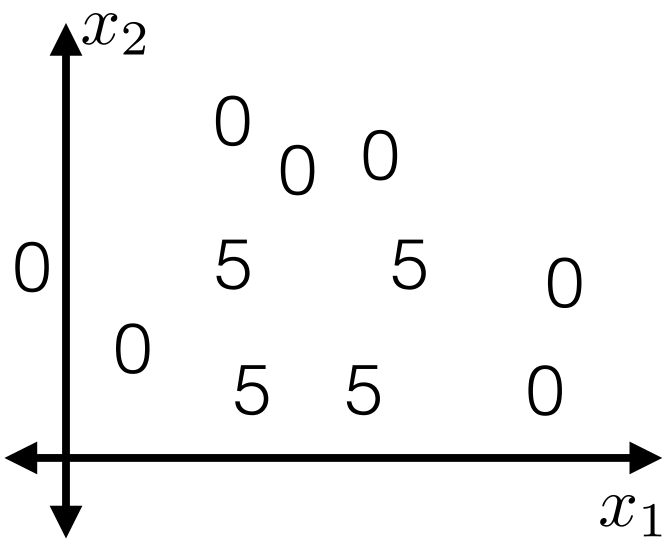


`BuildTree({1,…,n};2)`

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| $\leq$ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
```

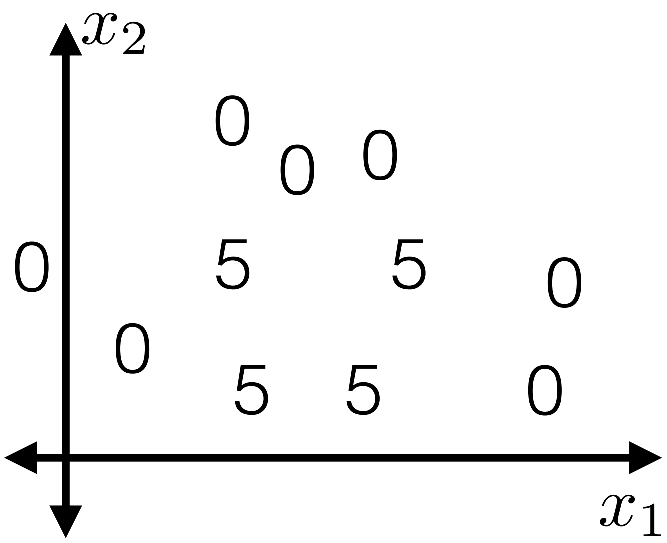


BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| $\leq$ k
    Set $\hat{y}$ = $\text{average}_{i\in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
```

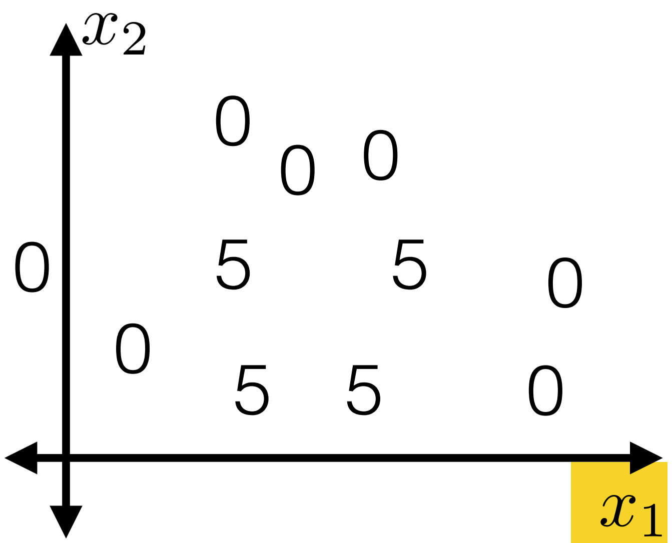


BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| $\leq$ k
    Set $\hat{y}$ = $\text{average}_{i\in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
```

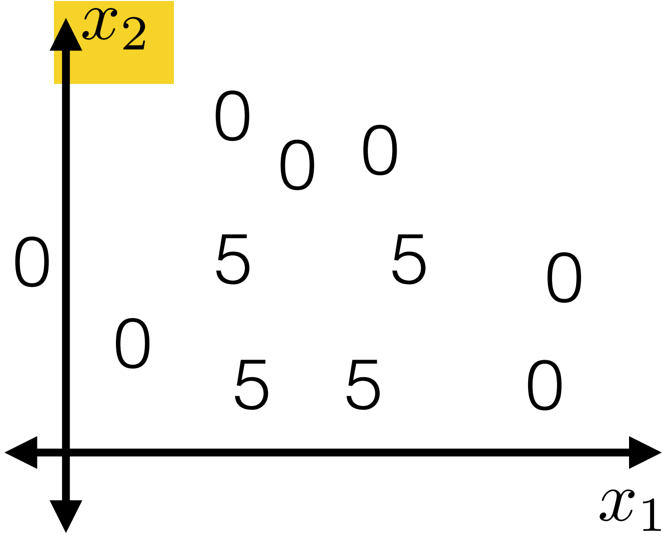


BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| $\leq$ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
```

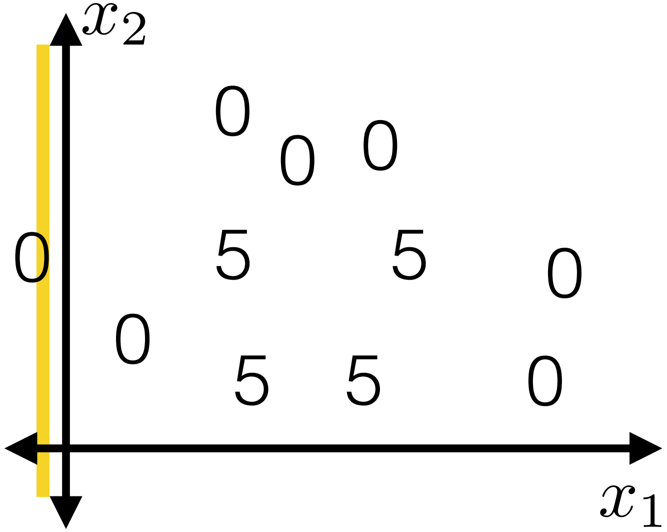


BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| $\leq$ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
```

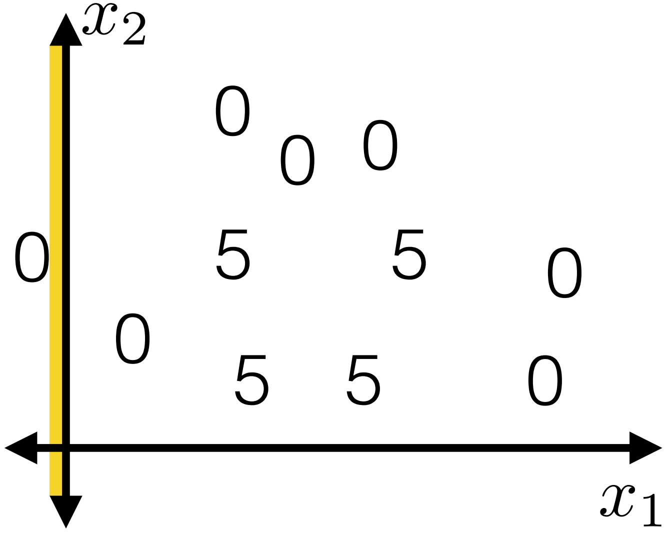


BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| $\leq$ k
    Set $\hat{y}$ = $\mathrm{average}_{i\in I}y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
```

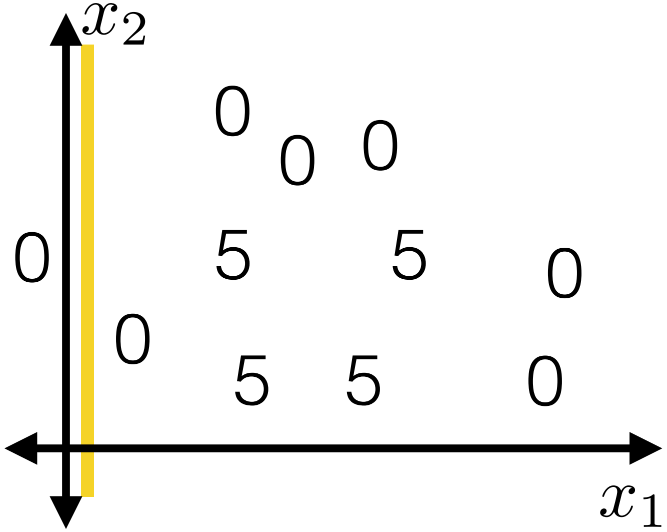


BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i\in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
```

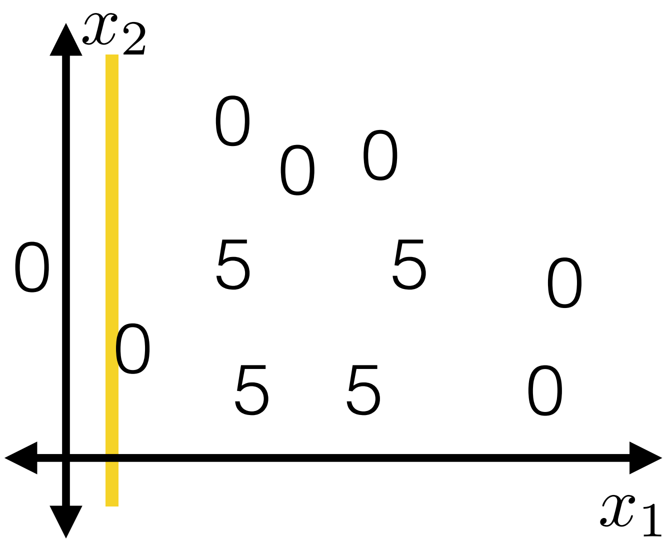


BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| $\leq$ k
    Set $\hat{y}$ = $\text{average}_{i\in I}\, y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
```

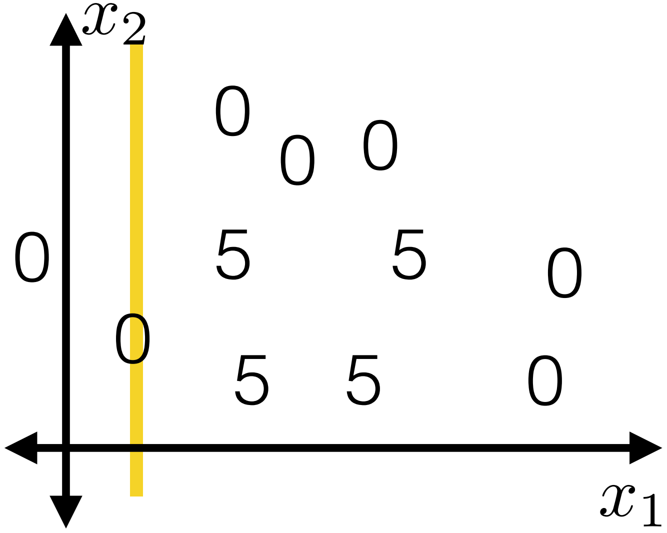


BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| $\leq$ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
```

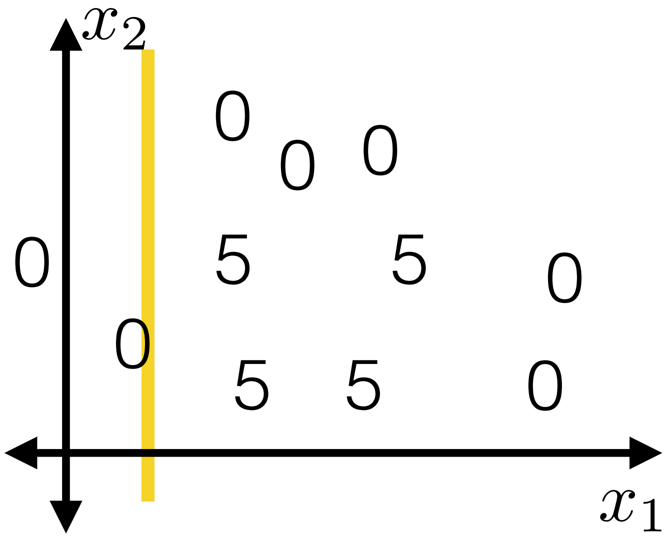


BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I; k$)
  if |$I$| $\leq$ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
```

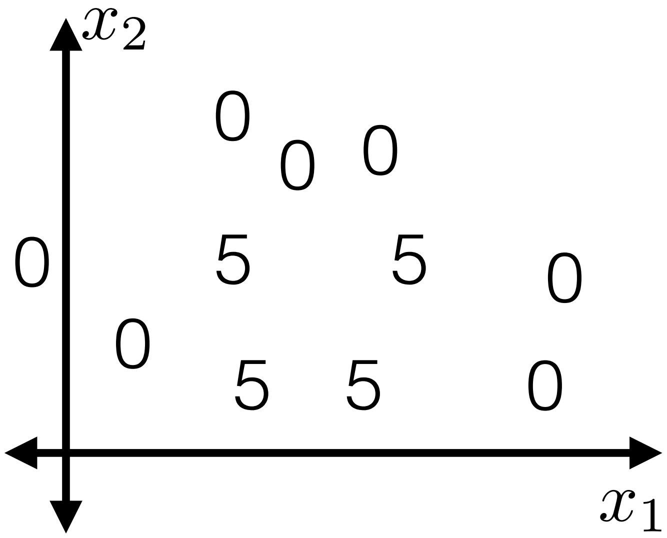


BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| $\leq$ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
```

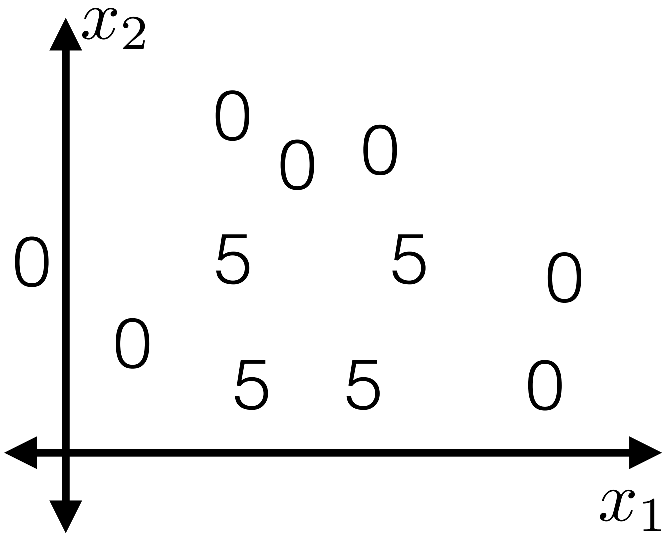


BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
```

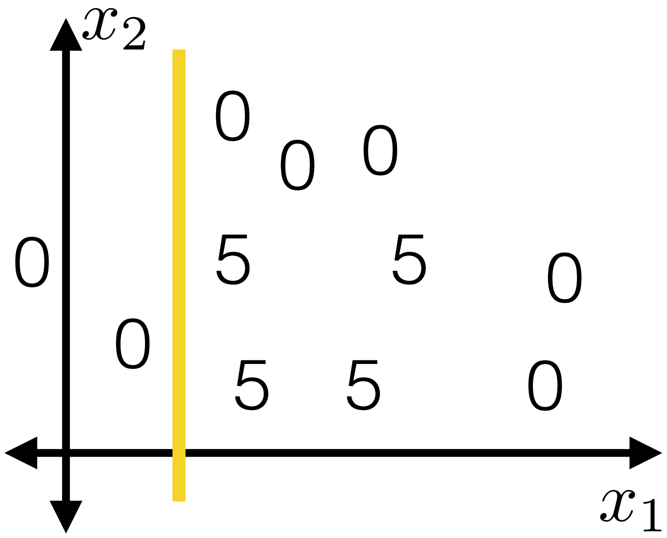


BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| $\leq$ k
    Set $\hat{y}$ = $\text{average}_{i\in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
```

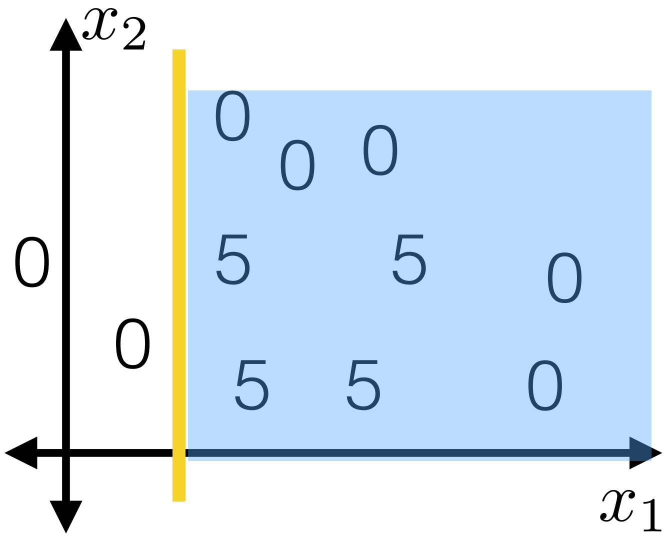


BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss

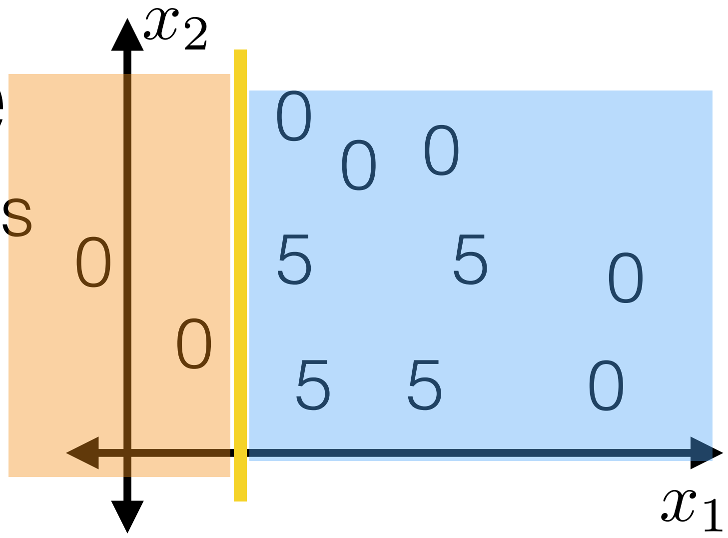
```
BuildTree($I$;$k$)
  if |$I$| $\leq$ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
```



BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss



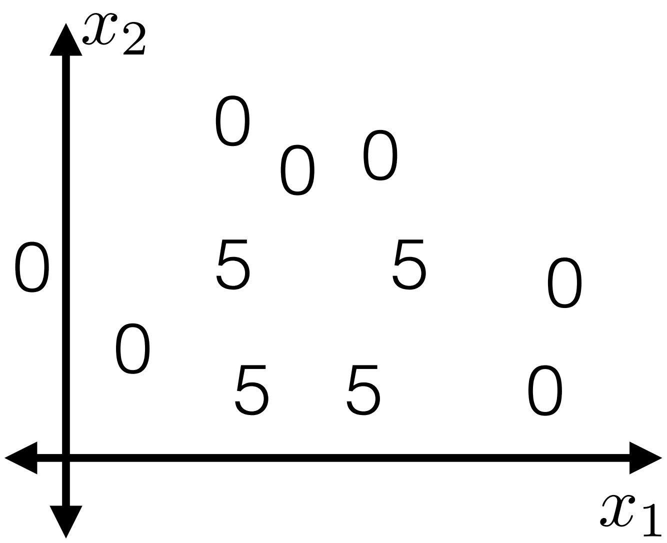
`BuildTree({1,…,n};2)`

```
BuildTree($I$;$k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
```

# Building a decision tree

- Regression tree with squared error loss

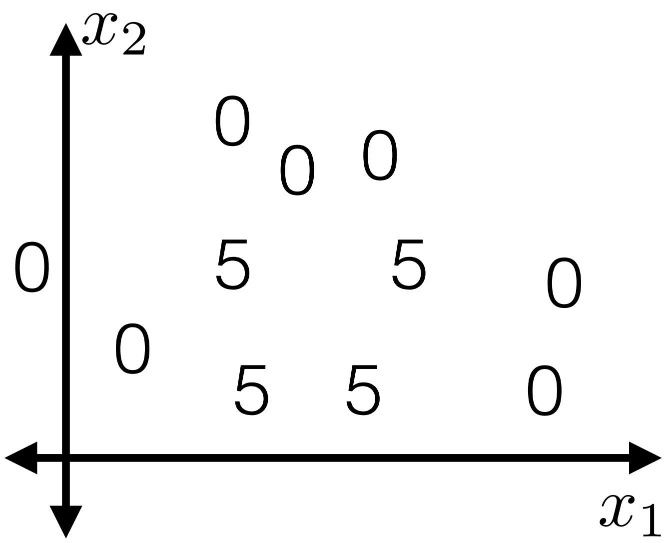
```
BuildTree($I$;$k$)
  if |$I$| $\leq$ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I^{+}_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^{-}_{j,s} = \{i \in I | x^{(i)}_j < s\}$
        $\hat{y}^{+}_{j,s}$
        $\hat{y}^{-}_{j,s}$
```



BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss

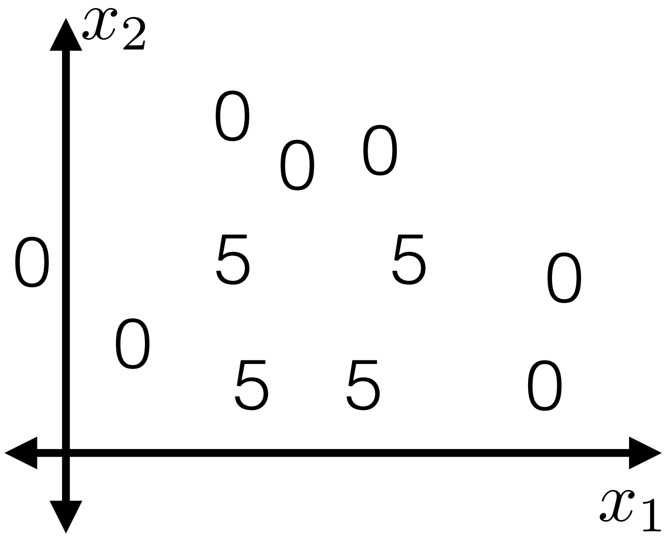


BuildTree({1,…,n};2)

```
BuildTree($I$;$k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
      Set $I^{+}_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^{-}_{j,s} = \{i \in I | x^{(i)}_j < s\}$
          $\hat{y}^{+}_{j,s}$
          $\hat{y}^{-}_{j,s}$
      Set $E_{j,s} = \sum_{i \in I^{+}_{j,s}} (y^{(i)} - \hat{y}^{+}_{j,s})^2 + \sum_{i \in I^{-}_{j,s}} (y^{(i)} - \hat{y}^{-}_{j,s})^2$
```

# Building a decision tree

- Regression tree with squared error loss

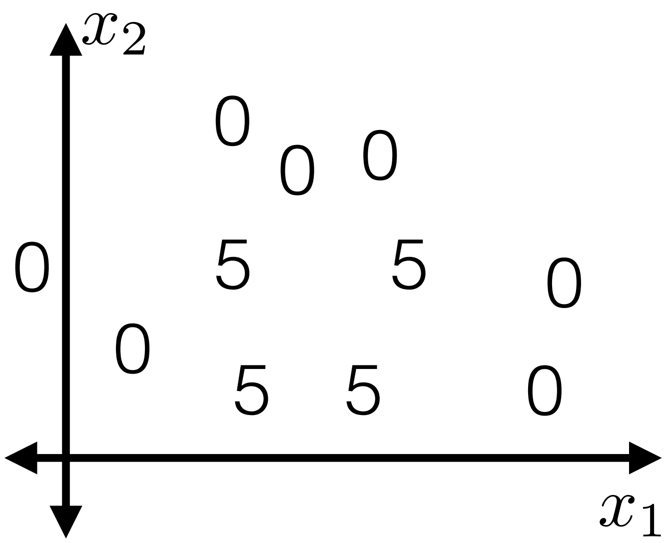


BuildTree({1,…,n};2)

```
BuildTree($I$;$k$)
  if $|I| \leq$ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x_j^{(i)} \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x_j^{(i)} < s\}$
      $\hat{y}^+_{j,s}$
      $\hat{y}^-_{j,s}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
```

# Building a decision tree

- Regression tree with squared error loss



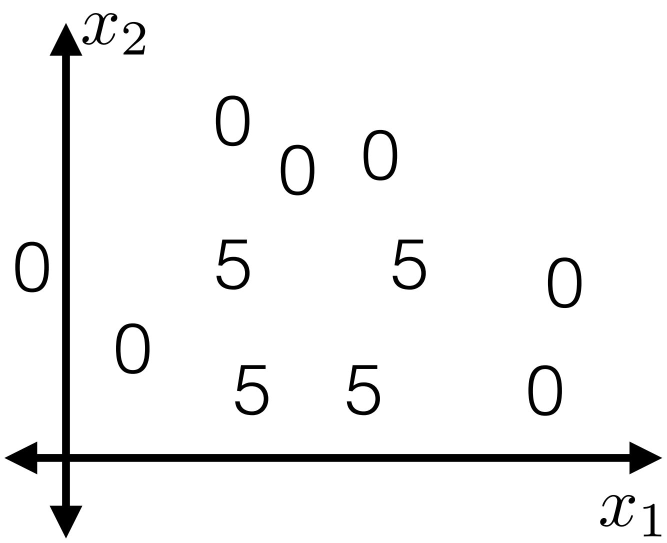
BuildTree({1,…,n};2)

```
BuildTree($I$;$k$)
  if |$I$| $\leq$ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x_j^{(i)} \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x_j^{(i)} < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
```

# Building a decision tree

- Regression tree with squared error loss

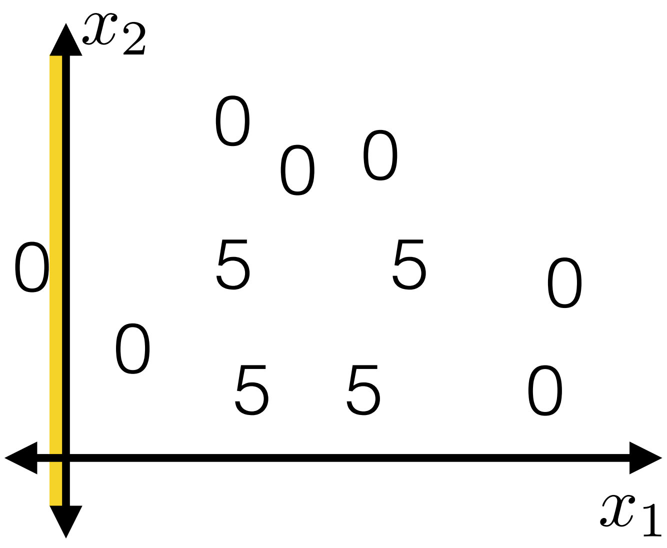
```
BuildTree($I$;$k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i\in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i\in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i\in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i\in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i\in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
```



BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss



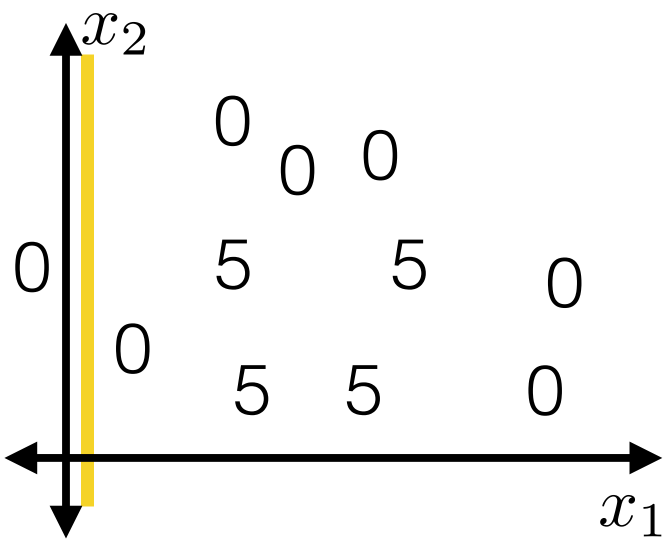
BuildTree({1,…,n};2)

```
BuildTree($I$;$k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i\in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i\in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i\in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i\in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i\in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
```

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
      Set $I^{+}_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^{-}_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^{+}_{j,s} = \text{average}_{i \in I^{+}_{j,s}} y^{(i)}$
      Set $\hat{y}^{-}_{j,s} = \text{average}_{i \in I^{-}_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^{+}_{j,s}} (y^{(i)} - \hat{y}^{+}_{j,s})^2 + \sum_{i \in I^{-}_{j,s}} (y^{(i)} - \hat{y}^{-}_{j,s})^2$
```

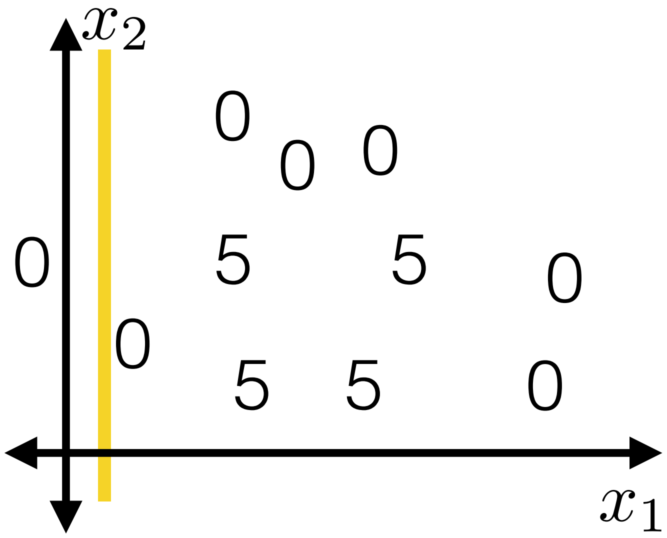


BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss

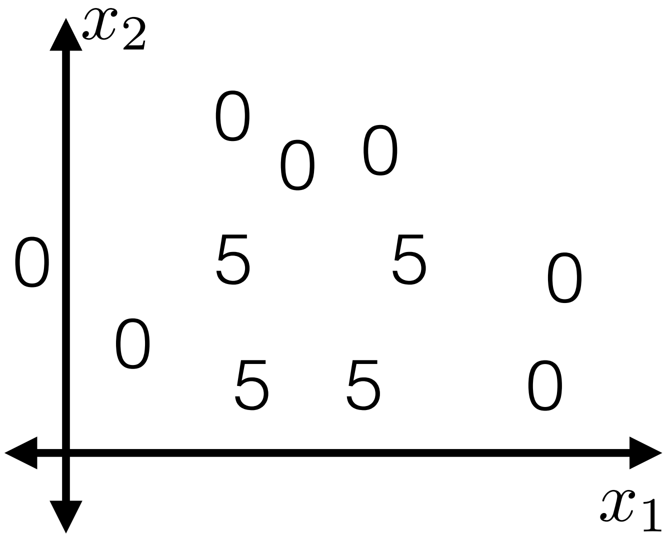
```
BuildTree($I$;$k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
```



BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss



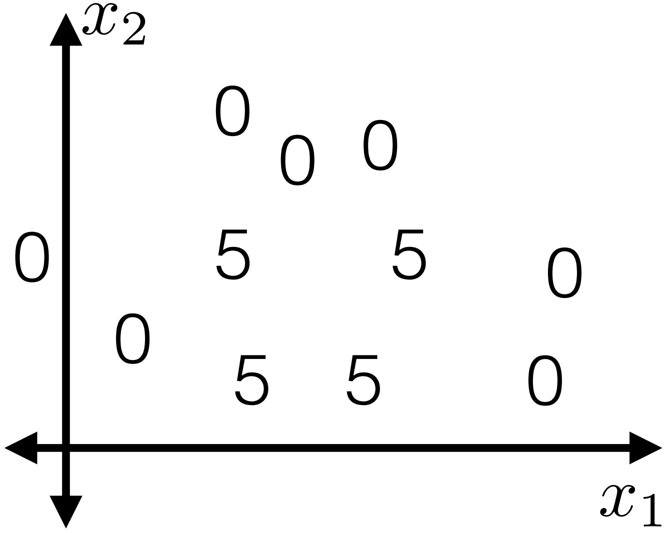
BuildTree({1,…,n};2)

```
BuildTree($I$;$k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
```

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
```

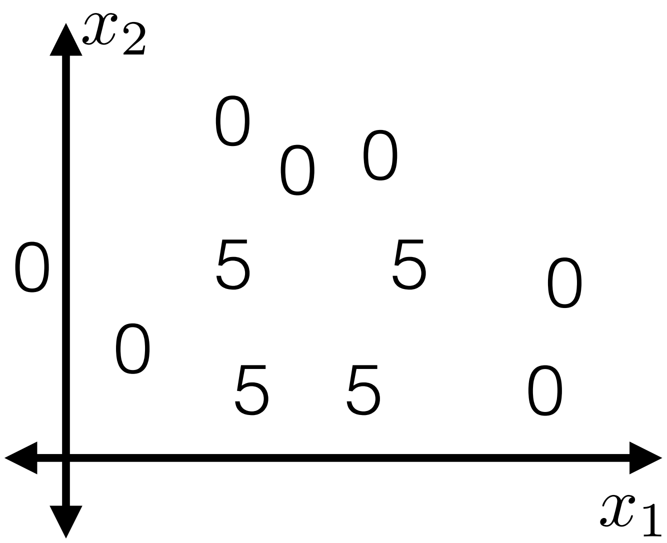


BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss

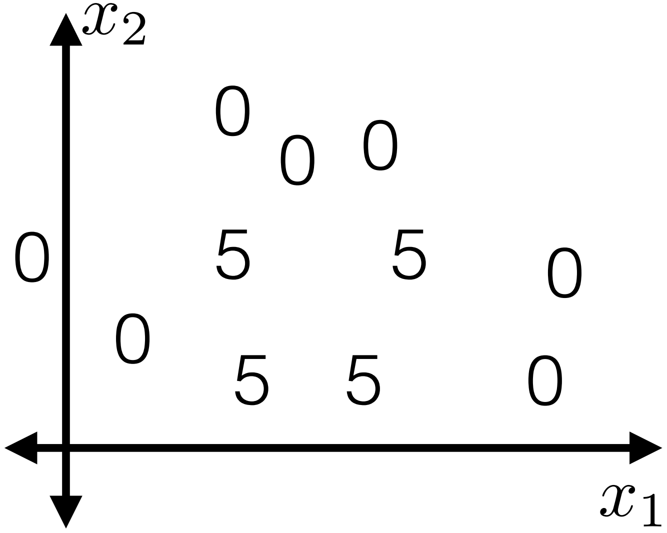
```
BuildTree($I; k$)
  if $|I| \leq$ k
    Set $\hat{y}$ = $\text{average}_{i\in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i\in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i\in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i\in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i\in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node
```



BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss



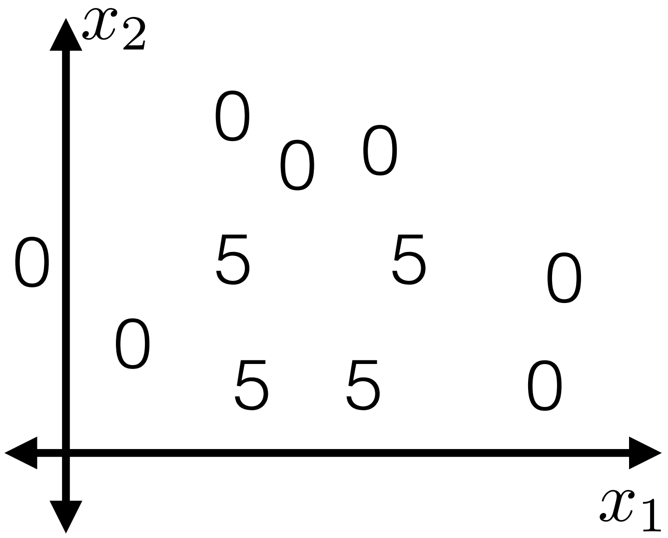
BuildTree({1,…,n};2)

```
BuildTree($I$;$k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node(dim,val,left-child,right-child)
```

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I; k$)
  if $|I| \leq$ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I^{+}_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^{-}_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^{+}_{j,s} = \text{average}_{i \in I^{+}_{j,s}} y^{(i)}$
      Set $\hat{y}^{-}_{j,s} = \text{average}_{i \in I^{-}_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^{+}_{j,s}} (y^{(i)} - \hat{y}^{+}_{j,s})^2 + \sum_{i \in I^{-}_{j,s}} (y^{(i)} - \hat{y}^{-}_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node($j^*$, val, left-child, right-child)
```

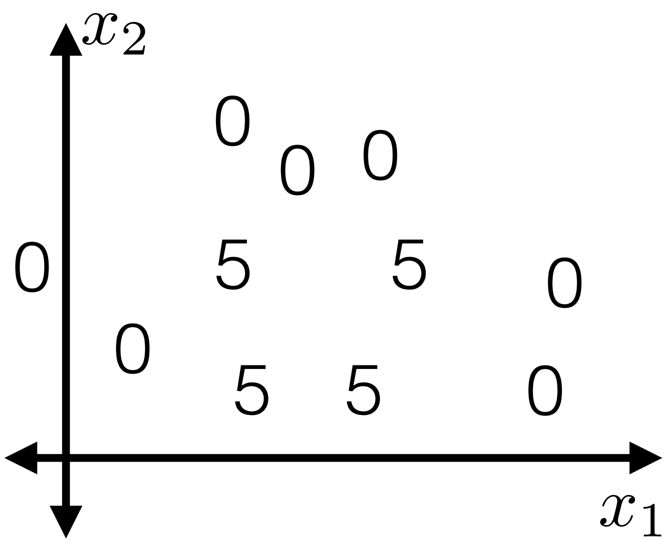


BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss

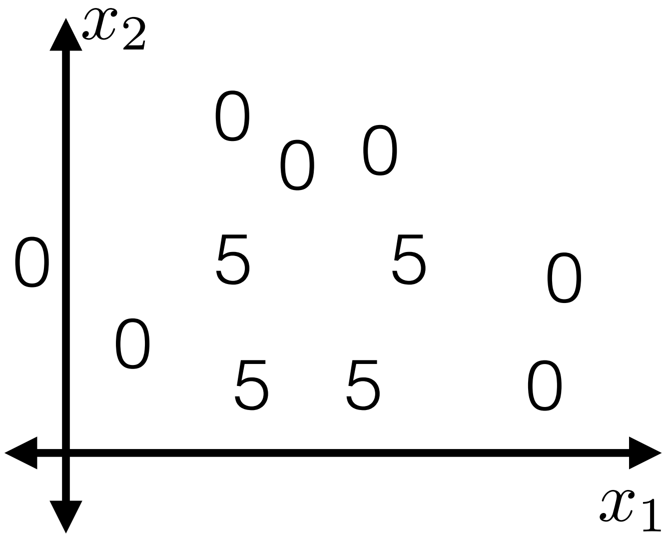
```
BuildTree($I$;$k$)
  if |$I$| $\leq$ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node($j^*$, val, left-child, right-child)
```



BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss

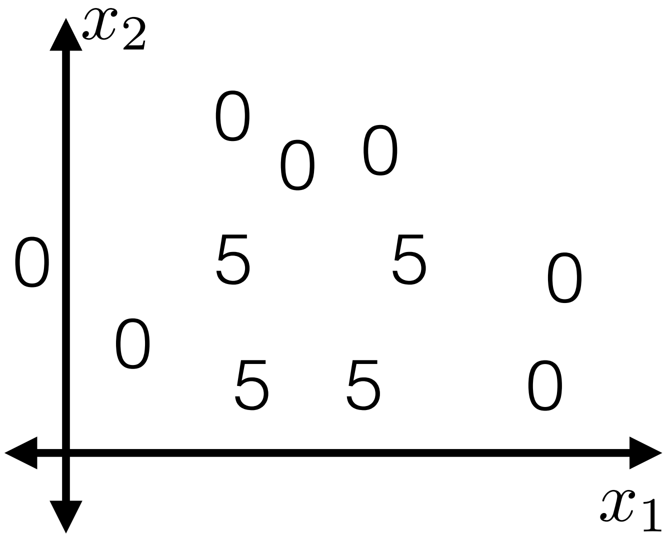


BuildTree({1,…,n};2)

```
BuildTree($I; k$)
  if $|I| \le$ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \ge s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node($j^*, s^*$ ,left-child,right-child)
```

# Building a decision tree

- Regression tree with squared error loss

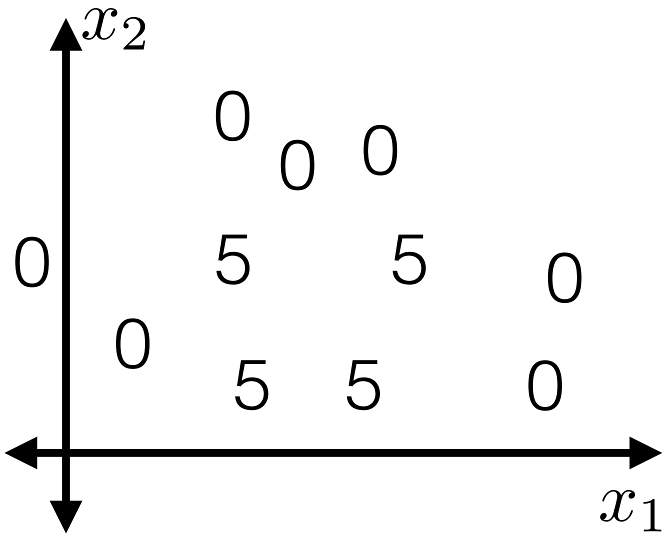


BuildTree({1,…,n};2)

```
BuildTree($I$;$k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i\in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I^{+}_{j,s} = \{i \in I | x_j^{(i)} \geq s\}$
      Set $I^{-}_{j,s} = \{i \in I | x_j^{(i)} < s\}$
      Set $\hat{y}^{+}_{j,s} = \text{average}_{i \in I^{+}_{j,s}} y^{(i)}$
      Set $\hat{y}^{-}_{j,s} = \text{average}_{i \in I^{-}_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^{+}_{j,s}} (y^{(i)} - \hat{y}^{+}_{j,s})^2 + \sum_{i \in I^{-}_{j,s}} (y^{(i)} - \hat{y}^{-}_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node($j^*, s^*$ ,left-child,right-child)
```

# Building a decision tree

- Regression tree with squared error loss

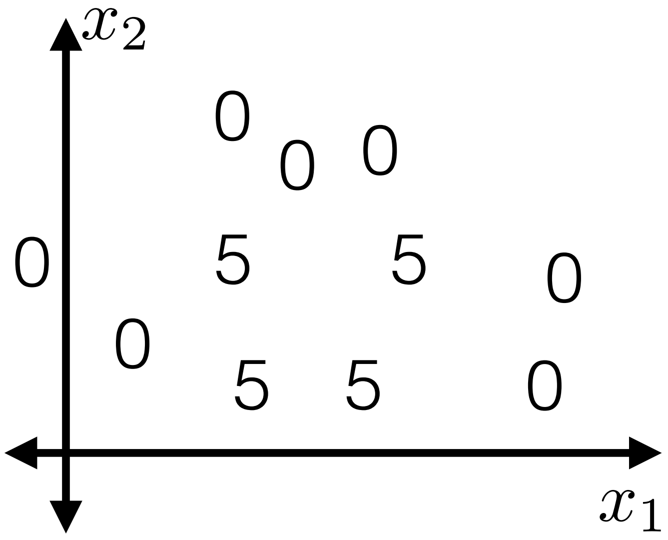


BuildTree({1,…,n};2)

```
BuildTree($I$;$k$)
  if |$I$| $\leq$ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node($j^*, s^*$, $\text{BuildTree}(I^-_{j^*,s^*}, k)$, right-child)
```

# Building a decision tree

- Regression tree with squared error loss



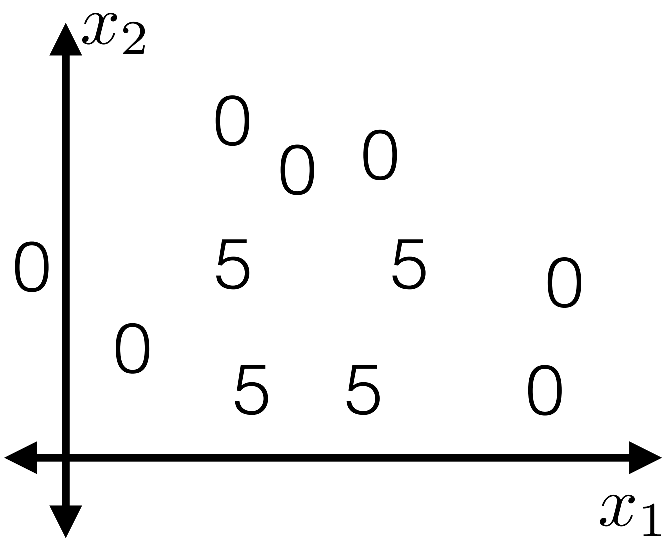
BuildTree({1,…,n};2)

```
BuildTree($I; k$)
  if $|I| \leq$ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node ($j^*, s^*$, BuildTree($I^-_{j^*,s^*}, k$), right-child)
```

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if $|I| \leq$ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node $(j^*, s^*, \text{BuildTree}(I^-_{j^*,s^*}, k), \text{BuildTree}(I^+_{j^*,s^*}, k))$
```

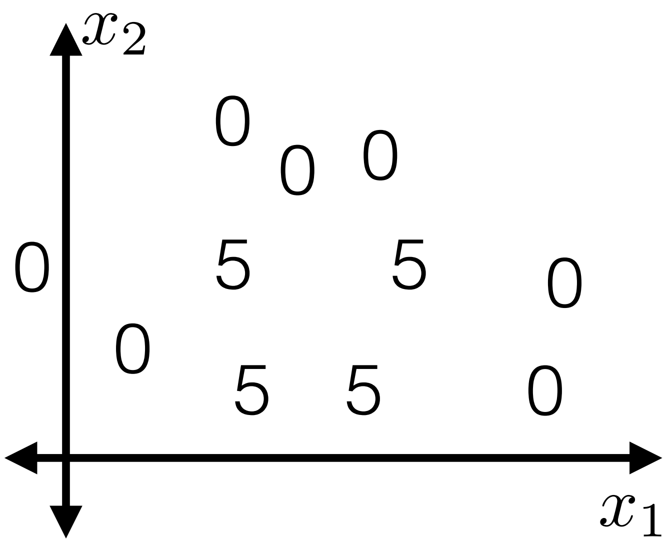


BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree(I;k)
  if |I| ≤ k
    Set $\hat{y} = \text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I^{+}_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^{-}_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^{+}_{j,s} = \text{average}_{i \in I^{+}_{j,s}} y^{(i)}$
      Set $\hat{y}^{-}_{j,s} = \text{average}_{i \in I^{-}_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^{+}_{j,s}} (y^{(i)} - \hat{y}^{+}_{j,s})^2 + \sum_{i \in I^{-}_{j,s}} (y^{(i)} - \hat{y}^{-}_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node $(j^*, s^*, \text{BuildTree}(I^{-}_{j^*,s^*}, k), \text{BuildTree}(I^{+}_{j^*,s^*}, k))$
```

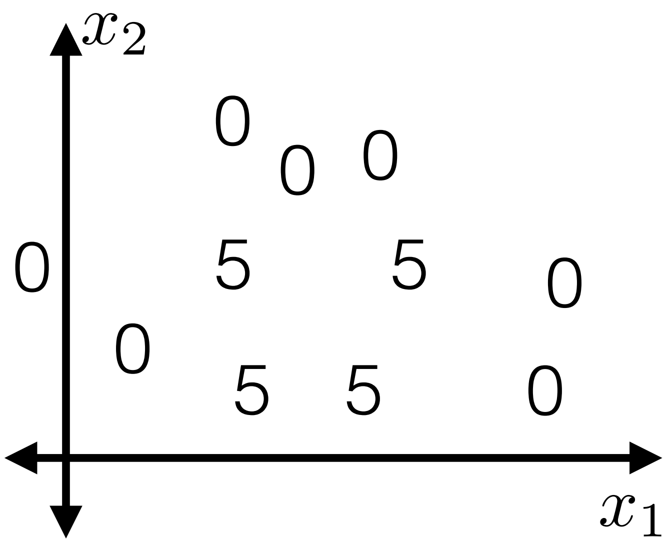


`BuildTree({1,…,n};2)`

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| $\leq$ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I_{j,s}^{+} = \{i \in I | x_j^{(i)} \geq s\}$
      Set $I_{j,s}^{-} = \{i \in I | x_j^{(i)} < s\}$
      Set $\hat{y}_{j,s}^{+} = \text{average}_{i \in I_{j,s}^{+}} y^{(i)}$
      Set $\hat{y}_{j,s}^{-} = \text{average}_{i \in I_{j,s}^{-}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I_{j,s}^{+}} (y^{(i)} - \hat{y}_{j,s}^{+})^2 + \sum_{i \in I_{j,s}^{-}} (y^{(i)} - \hat{y}_{j,s}^{-})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node $(j^*, s^*, \text{BuildTree}(I_{j^*,s^*}^{-}, k), \text{BuildTree}(I_{j^*,s^*}^{+}, k))$
```

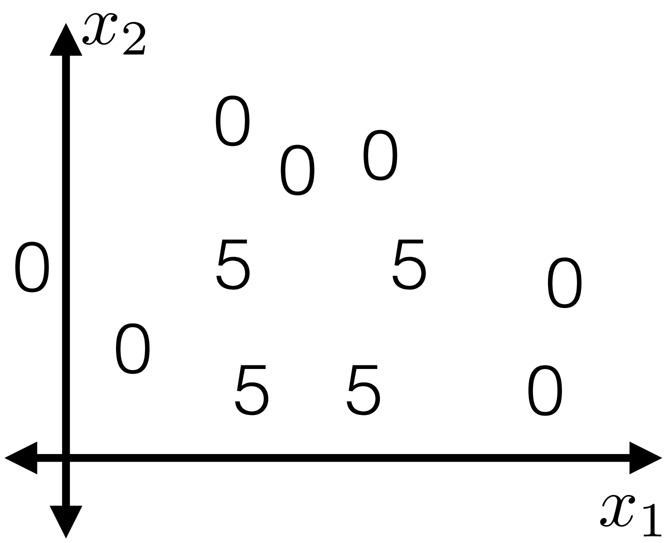


BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I; k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I^{+}_{j,s} = \{i \in I | x^{(i)}_{j} \geq s\}$
      Set $I^{-}_{j,s} = \{i \in I | x^{(i)}_{j} < s\}$
      Set $\hat{y}^{+}_{j,s} = \text{average}_{i \in I^{+}_{j,s}} y^{(i)}$
      Set $\hat{y}^{-}_{j,s} = \text{average}_{i \in I^{-}_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^{+}_{j,s}} (y^{(i)} - \hat{y}^{+}_{j,s})^2 + \sum_{i \in I^{-}_{j,s}} (y^{(i)} - \hat{y}^{-}_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node ($j^*, s^*$, $\text{BuildTree}(I^{-}_{j^*,s^*}, k), \text{BuildTree}(I^{+}_{j^*,s^*}, k)$)
```

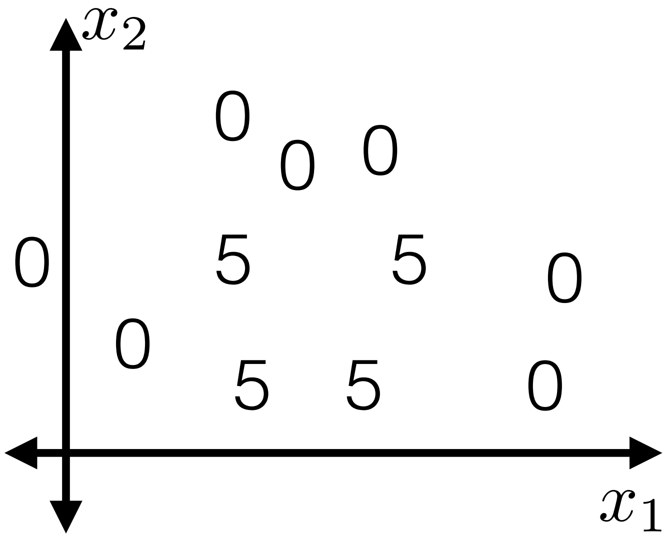


BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss

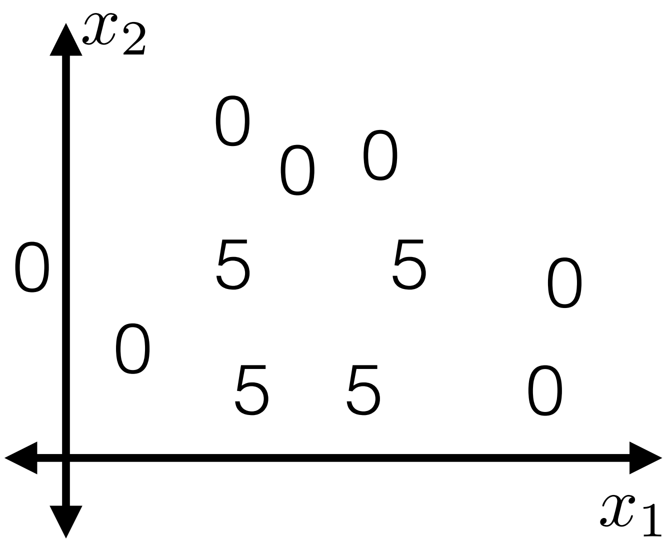
```
BuildTree($I$;$k$)
  if |$I$| $\leq$ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node $(j^*, s^*, \text{BuildTree}(I^-_{j^*,s^*}, k), \text{BuildTree}(I^+_{j^*,s^*}, k))$
```



BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss

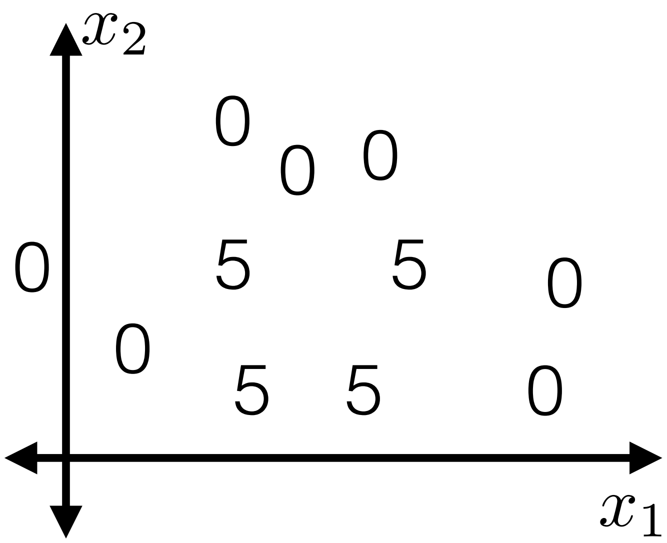


BuildTree({1,…,n};2)

```
BuildTree($I$;$k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node($j^*, s^*$, $\text{BuildTree}(I^-_{j^*,s^*}, k)$, $\text{BuildTree}(I^+_{j^*,s^*}, k)$)
```

# Building a decision tree

- Regression tree with squared error loss



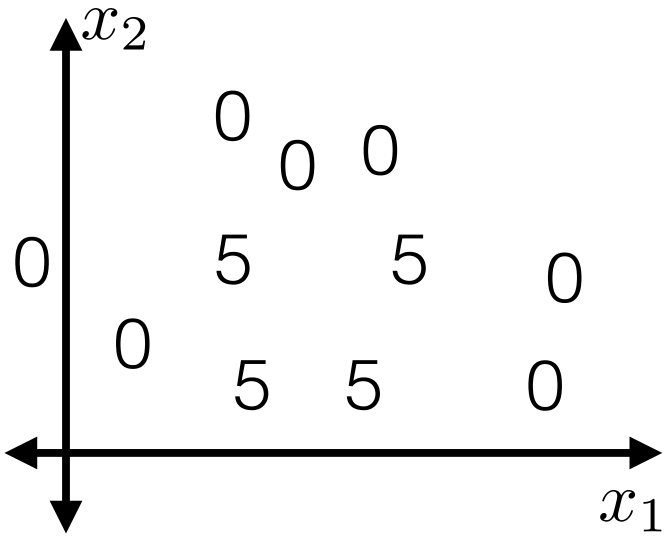
BuildTree({1,…,n};2)

```
BuildTree($I$;$k$)
  if |$I$| $\leq$ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node($j^*, s^*$, $\text{BuildTree}(I^-_{j^*,s^*}, k), \text{BuildTree}(I^+_{j^*,s^*}, k)$)
```

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| $\leq$ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
      Set $I^{+}_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^{-}_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^{+}_{j,s} = \text{average}_{i \in I^{+}_{j,s}} y^{(i)}$
      Set $\hat{y}^{-}_{j,s} = \text{average}_{i \in I^{-}_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^{+}_{j,s}} (y^{(i)} - \hat{y}^{+}_{j,s})^2 + \sum_{i \in I^{-}_{j,s}} (y^{(i)} - \hat{y}^{-}_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node$(j^*, s^*, \text{BuildTree}(I^{-}_{j^*,s^*}, k), \text{BuildTree}(I^{+}_{j^*,s^*}, k))$
```

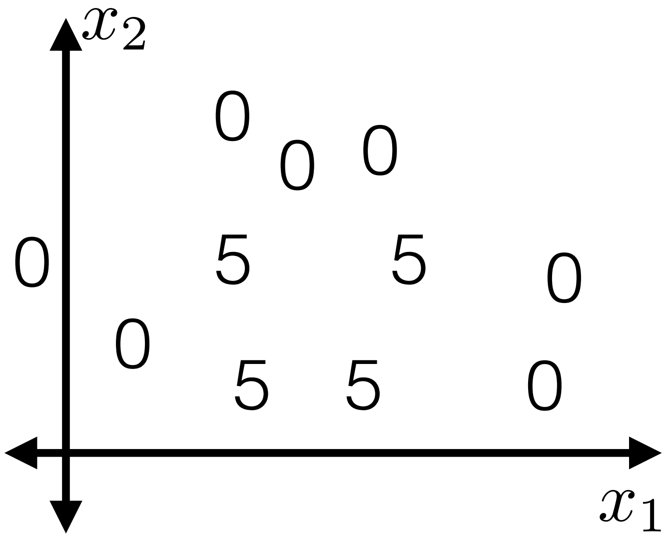


BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss

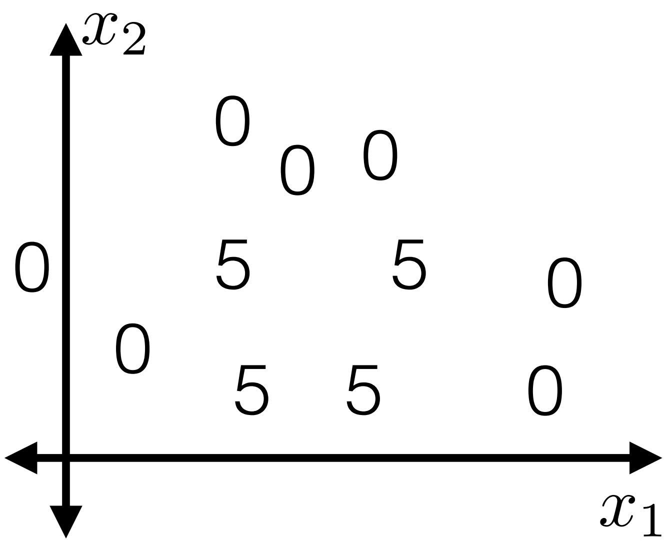
```
BuildTree($I; k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I_{j,s}^{+} = \{i \in I | x_j^{(i)} \geq s\}$
      Set $I_{j,s}^{-} = \{i \in I | x_j^{(i)} < s\}$
      Set $\hat{y}_{j,s}^{+} = \text{average}_{i \in I_{j,s}^{+}} y^{(i)}$
      Set $\hat{y}_{j,s}^{-} = \text{average}_{i \in I_{j,s}^{-}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I_{j,s}^{+}} (y^{(i)} - \hat{y}_{j,s}^{+})^2 + \sum_{i \in I_{j,s}^{-}} (y^{(i)} - \hat{y}_{j,s}^{-})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node ($j^*, s^*$, $\text{BuildTree}(I_{j^*,s^*}^{-}, k), \text{BuildTree}(I_{j^*,s^*}^{+}, k))$
```



BuildTree({1,…,n};2)

# Building a decision tree

- Regression tree with squared error loss



BuildTree({1,…,n};2)

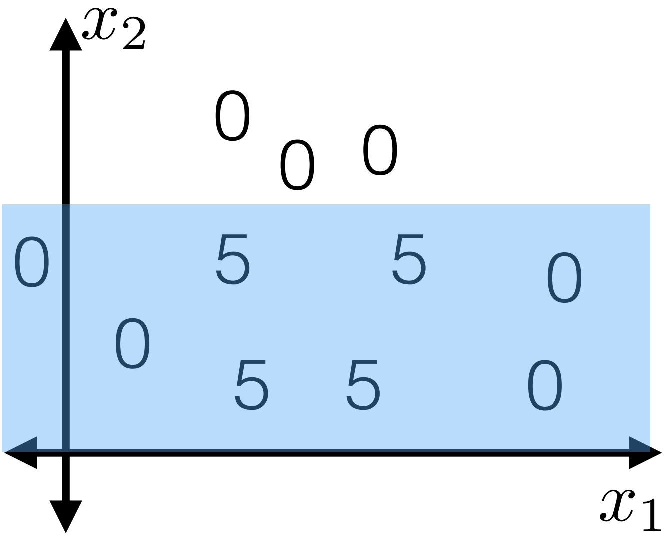
$x_2 \geq 0.28$

```
BuildTree($I$;$k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I^{+}_{j,s} = \{i \in I | x_j^{(i)} \geq s\}$
      Set $I^{-}_{j,s} = \{i \in I | x_j^{(i)} < s\}$
      Set $\hat{y}^{+}_{j,s} = \text{average}_{i \in I^{+}_{j,s}} y^{(i)}$
      Set $\hat{y}^{-}_{j,s} = \text{average}_{i \in I^{-}_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^{+}_{j,s}} (y^{(i)} - \hat{y}^{+}_{j,s})^2 + \sum_{i \in I^{-}_{j,s}} (y^{(i)} - \hat{y}^{-}_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node($j^*$, $s^*$, $\text{BuildTree}(I^{-}_{j^*,s^*}, k)$, $\text{BuildTree}(I^{+}_{j^*,s^*}, k)$)
```

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I;k$)
  if $|I| \leq$ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I^{+}_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^{-}_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^{+}_{j,s} = \text{average}_{i \in I^{+}_{j,s}} y^{(i)}$
      Set $\hat{y}^{-}_{j,s} = \text{average}_{i \in I^{-}_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^{+}_{j,s}} (y^{(i)} - \hat{y}^{+}_{j,s})^2 + \sum_{i \in I^{-}_{j,s}} (y^{(i)} - \hat{y}^{-}_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node ($j^*, s^*$, BuildTree($I^{-}_{j^*,s^*}, k$), BuildTree($I^{+}_{j^*,s^*}, k$))
```



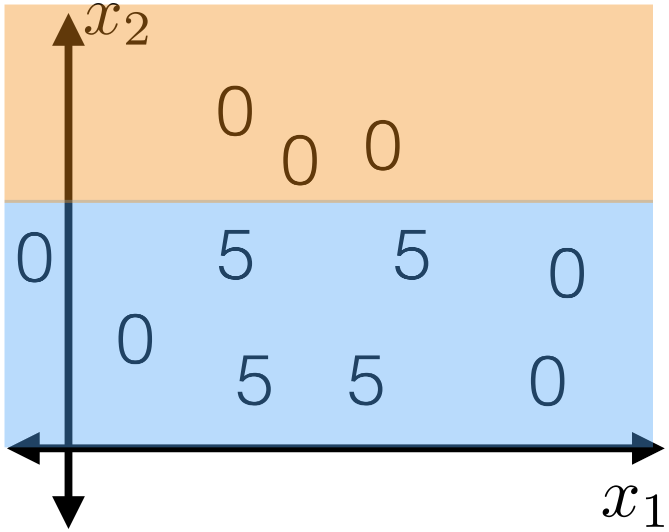
BuildTree({1,…,n};2)

$x_2 \geq 0.28$

# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$; $k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node($j^*$, $s^*$, $\text{BuildTree}(I^-_{j^*,s^*}, k), \text{BuildTree}(I^+_{j^*,s^*}, k)$)
```

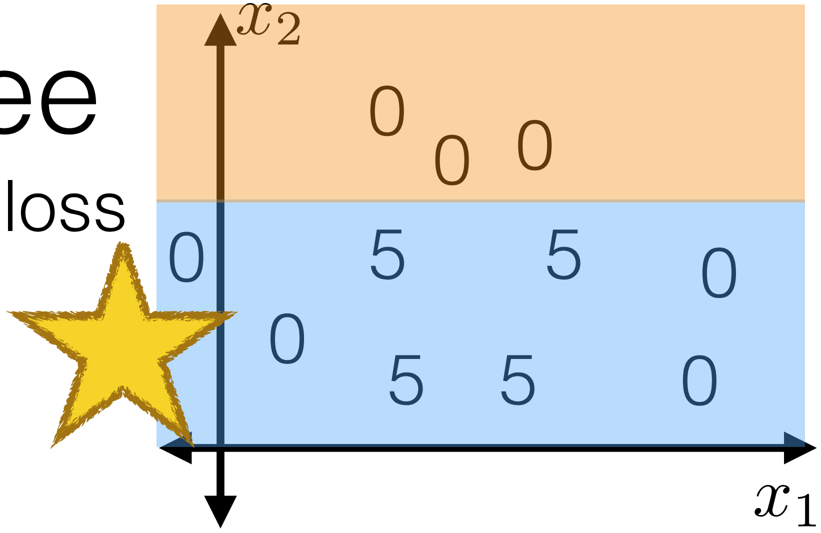


BuildTree({1,…,n};2)

$x_2 \geq 0.28$

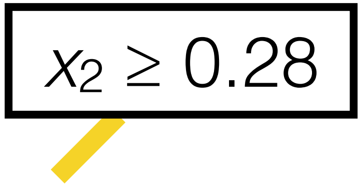
# Building a decision tree

- Regression tree with squared error loss



```
BuildTree($I$; $k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node $(j^*, s^*, \text{BuildTree}(I^-_{j^*,s^*}, k), \text{BuildTree}(I^+_{j^*,s^*}, k))$
```

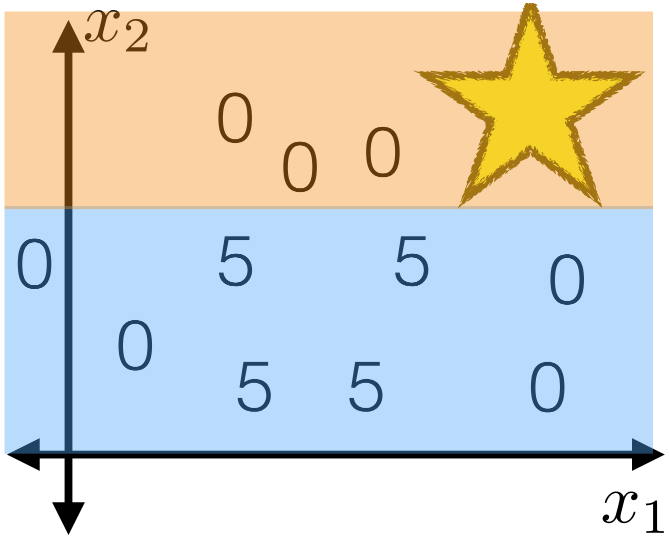
BuildTree({1,…,n};2)



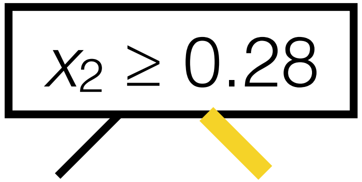
# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
      Set $I^{+}_{j,s} = \{i \in I | x^{(i)}_{j} \geq s\}$
      Set $I^{-}_{j,s} = \{i \in I | x^{(i)}_{j} < s\}$
      Set $\hat{y}^{+}_{j,s} = \text{average}_{i \in I^{+}_{j,s}} y^{(i)}$
      Set $\hat{y}^{-}_{j,s} = \text{average}_{i \in I^{-}_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^{+}_{j,s}} (y^{(i)} - \hat{y}^{+}_{j,s})^2 + \sum_{i \in I^{-}_{j,s}} (y^{(i)} - \hat{y}^{-}_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node($j^*, s^*$, BuildTree($I^{-}_{j^*,s^*}, k$), BuildTree($I^{+}_{j^*,s^*}, k$))
```



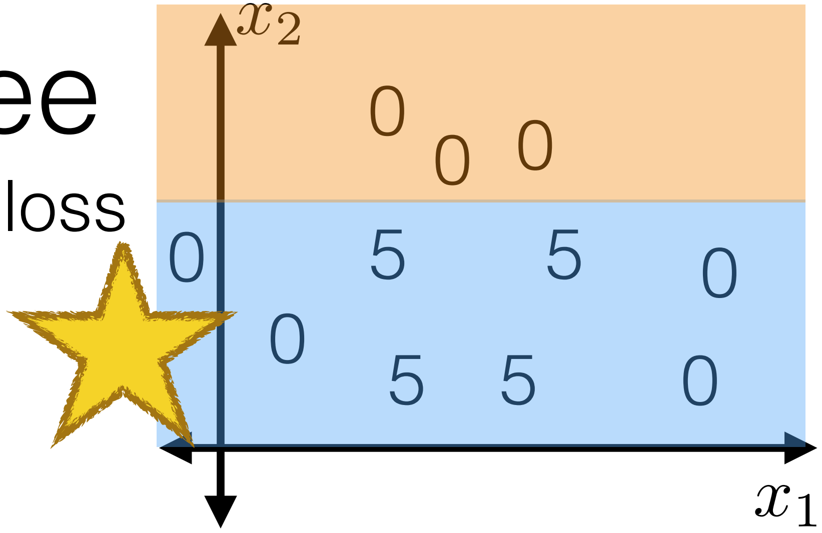
BuildTree({1,…,n};2)


$x_2 \geq 0.28$

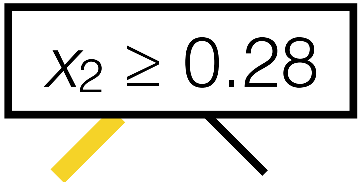
# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I;k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
      Set $I^{+}_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^{-}_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^{+}_{j,s} = \text{average}_{i \in I^{+}_{j,s}} y^{(i)}$
      Set $\hat{y}^{-}_{j,s} = \text{average}_{i \in I^{-}_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^{+}_{j,s}} (y^{(i)} - \hat{y}^{+}_{j,s})^2 + \sum_{i \in I^{-}_{j,s}} (y^{(i)} - \hat{y}^{-}_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node $(j^*, s^*, \text{BuildTree}(I^{-}_{j^*,s^*}, k), \text{BuildTree}(I^{+}_{j^*,s^*}, k))$
```



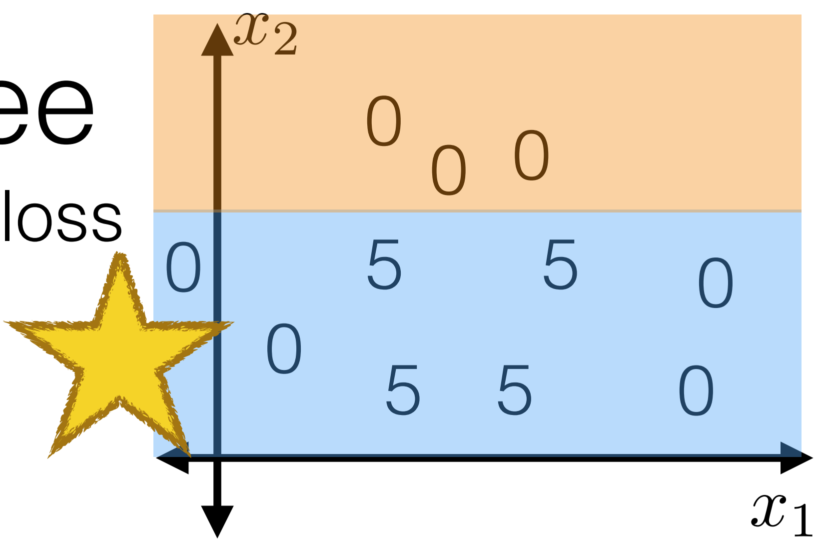
BuildTree({1,…,n};2)



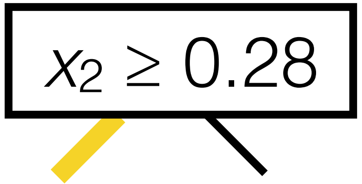
# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$; $k$)
  if $|I| \le$ k
    Set $\hat{y}$ = $\text{average}_{i\in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \ge s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i\in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i\in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i\in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i\in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node $(j^*, s^*, \text{BuildTree}(I^-_{j^*,s^*}, k), \text{BuildTree}(I^+_{j^*,s^*}, k))$
```

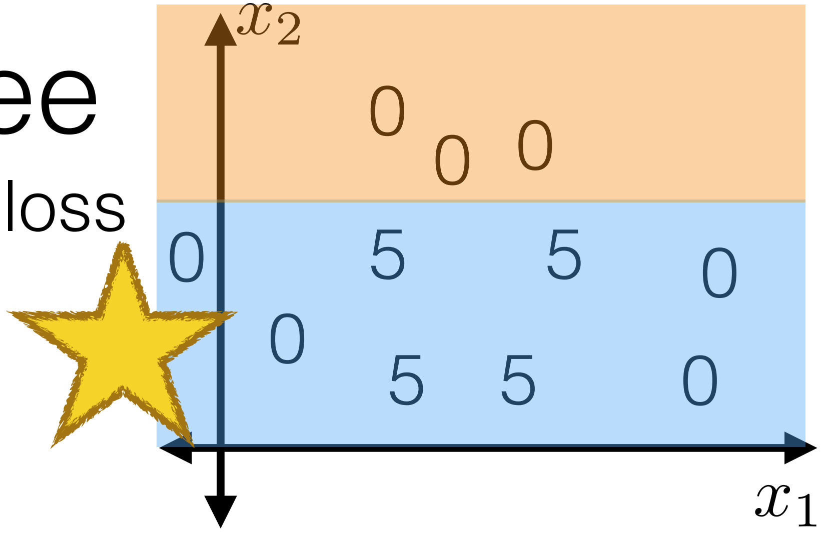


BuildTree({1,…,n};2)

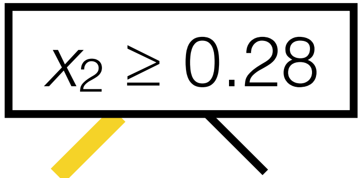


# Building a decision tree

- Regression tree with squared error loss



BuildTree({1,…,n};2)

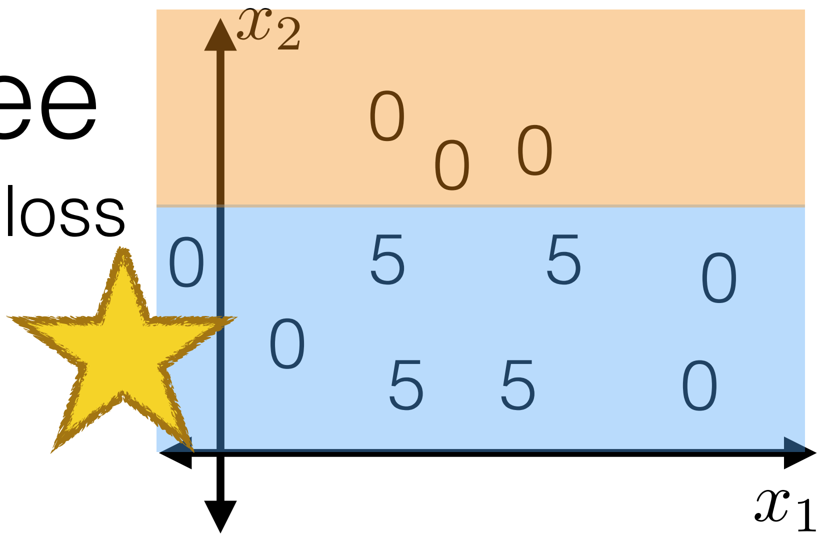

$x_2 \geq 0.28$

```
BuildTree($I$;$k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x_j^{(i)} \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x_j^{(i)} < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node($(j^*, s^*$, $\text{BuildTree}(I^-_{j^*,s^*}, k), \text{BuildTree}(I^+_{j^*,s^*}, k))$
```

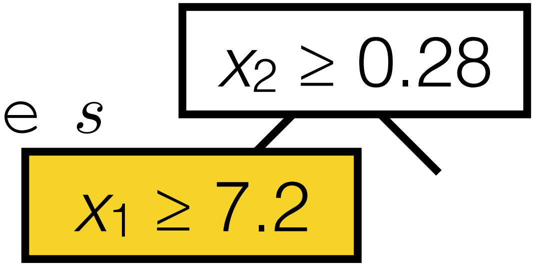
# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I^{+}_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^{-}_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^{+}_{j,s} = \text{average}_{i \in I^{+}_{j,s}} y^{(i)}$
      Set $\hat{y}^{-}_{j,s} = \text{average}_{i \in I^{-}_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^{+}_{j,s}} (y^{(i)} - \hat{y}^{+}_{j,s})^2 + \sum_{i \in I^{-}_{j,s}} (y^{(i)} - \hat{y}^{-}_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node ($j^*, s^*$, $\text{BuildTree}(I^{-}_{j^*,s^*}, k), \text{BuildTree}(I^{+}_{j^*,s^*}, k)$)
```



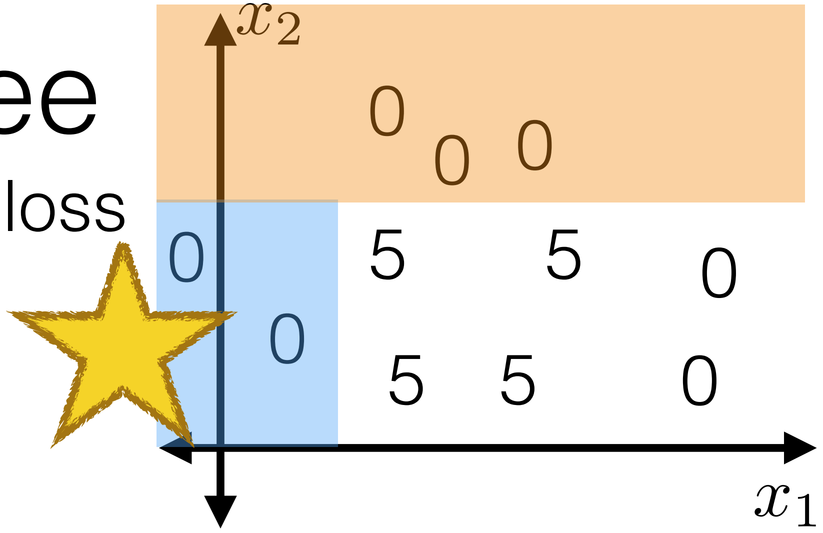
BuildTree({1,…,n};2)



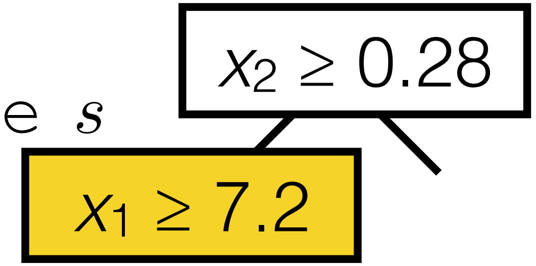
# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i\in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node($j^*, s^*$, $\text{BuildTree}(I^-_{j^*,s^*}, k), \text{BuildTree}(I^+_{j^*,s^*}, k)$)
```

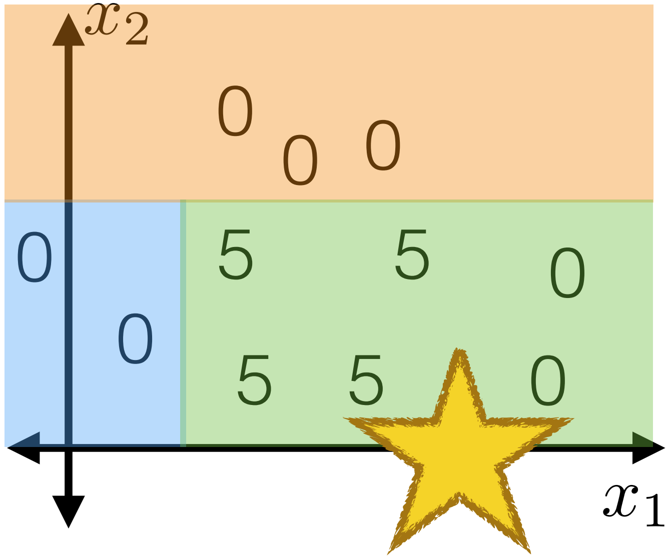


BuildTree({1,…,n};2)

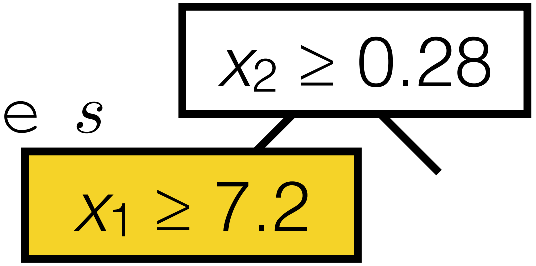


# Building a decision tree

- Regression tree with squared error loss



BuildTree({1,…,n};2)

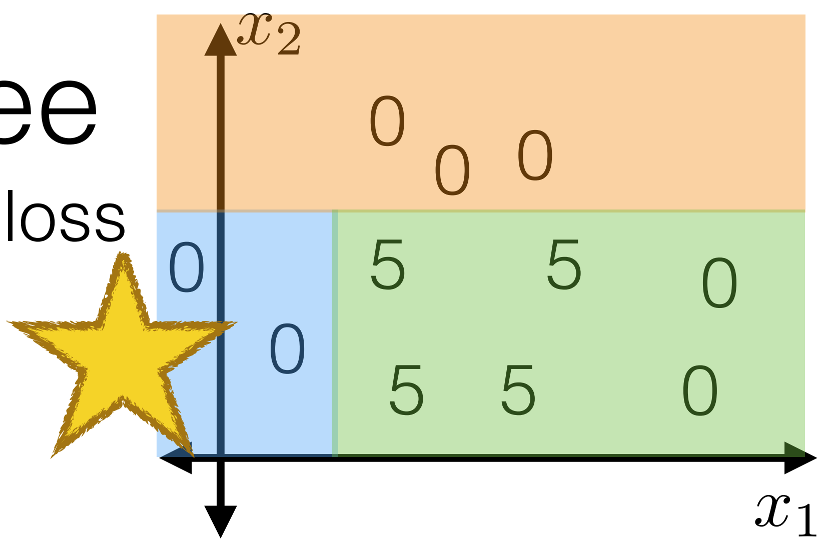


```
BuildTree($I$;$k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
      Set $I^{+}_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^{-}_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^{+}_{j,s} = \text{average}_{i \in I^{+}_{j,s}} y^{(i)}$
      Set $\hat{y}^{-}_{j,s} = \text{average}_{i \in I^{-}_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^{+}_{j,s}} (y^{(i)} - \hat{y}^{+}_{j,s})^2 + \sum_{i \in I^{-}_{j,s}} (y^{(i)} - \hat{y}^{-}_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node($j^*, s^*$, $\text{BuildTree}(I^{-}_{j^*,s^*}, k), \text{BuildTree}(I^{+}_{j^*,s^*}, k)$)
```

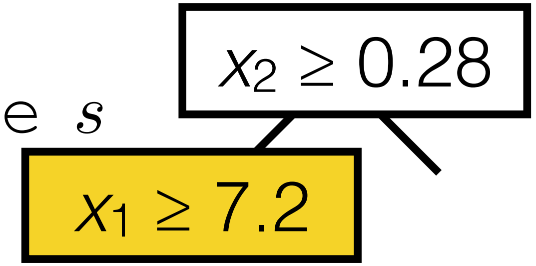
# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if $|I| \leq$ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node$(j^*, s^*$, $\text{BuildTree}(I^-_{j^*,s^*}, k), \text{BuildTree}(I^+_{j^*,s^*}, k))$
```



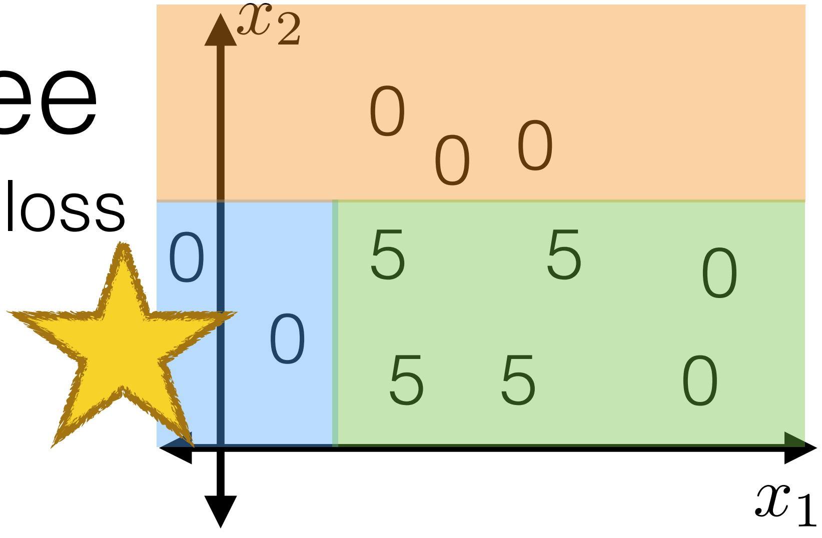
BuildTree({1,…,n};2)



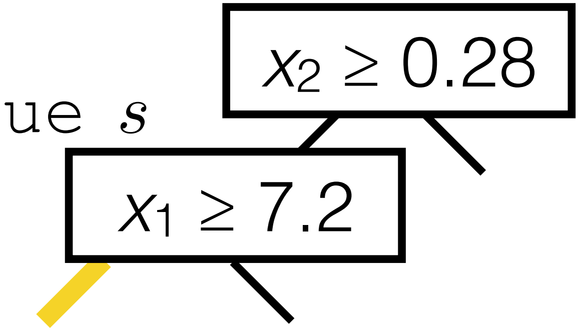
# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node ($j^*, s^*$, $\text{BuildTree}(I^-_{j^*,s^*}, k)$, $\text{BuildTree}(I^+_{j^*,s^*}, k)$)
```



BuildTree({1,…,n};2)



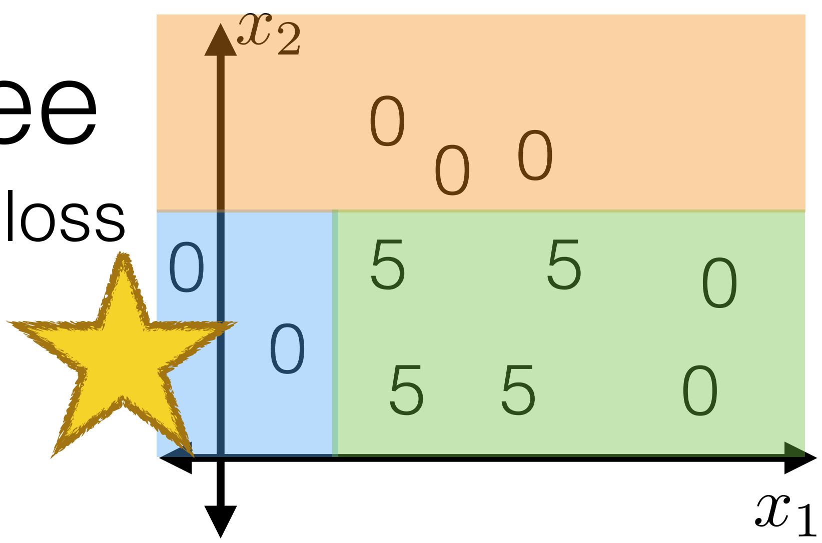
$x_2 \geq 0.28$

$x_1 \geq 7.2$

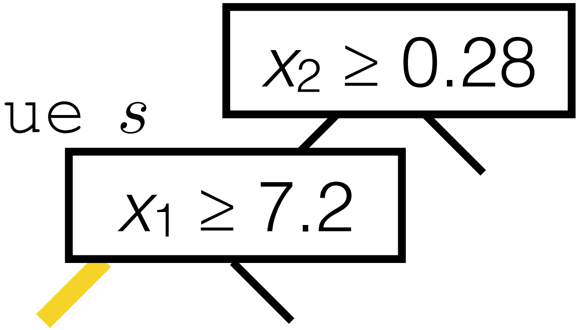
# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$; $k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node $(j^*, s^*, \text{BuildTree}(I^-_{j^*,s^*}, k), \text{BuildTree}(I^+_{j^*,s^*}, k))$
```



BuildTree({1,…,n};2)



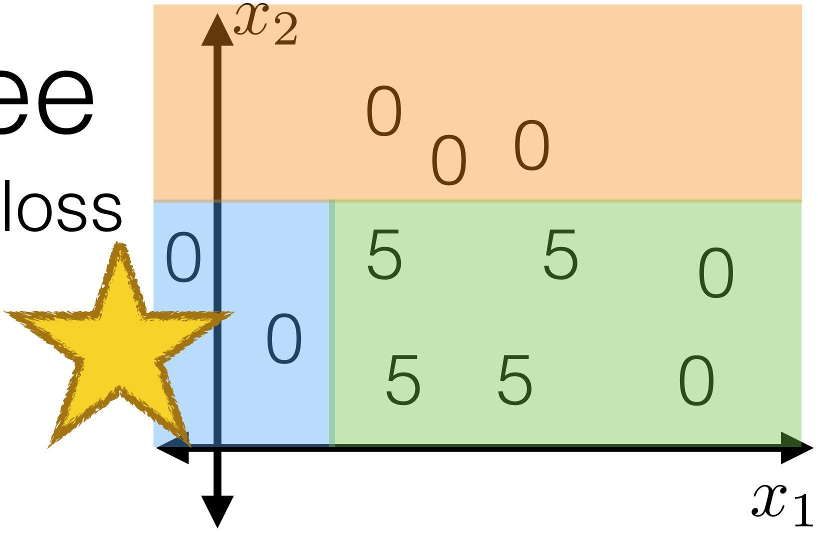
$x_2 \geq 0.28$

$x_1 \geq 7.2$

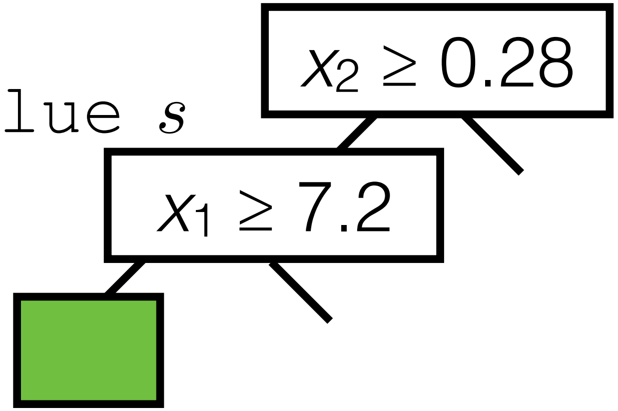
# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$; $k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i\in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i\in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i\in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i\in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i\in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node $(j^*, s^*, \text{BuildTree}(I^-_{j^*,s^*}, k), \text{BuildTree}(I^+_{j^*,s^*}, k))$
```



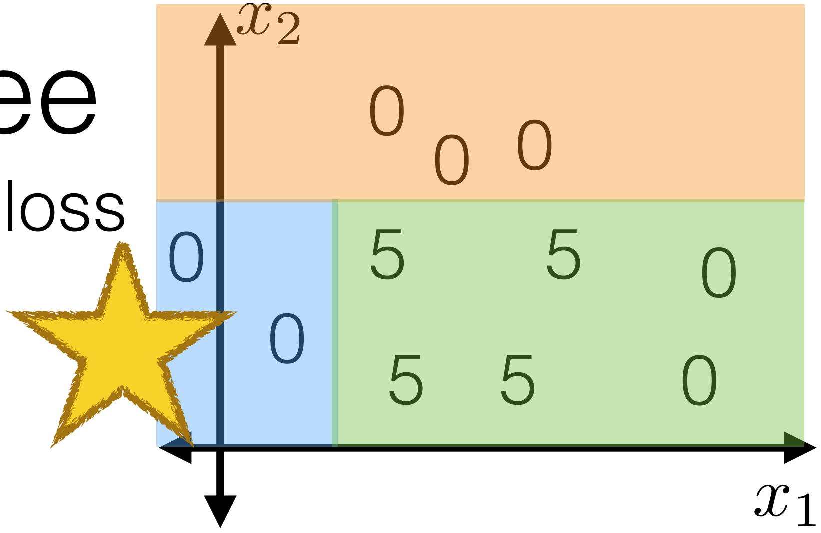
`BuildTree({1,…,n};2)`



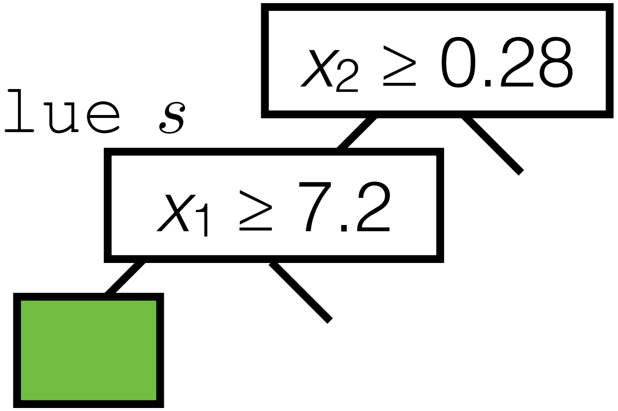
# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i\in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x_j^{(i)} \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x_j^{(i)} < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node $(j^*, s^*, \text{BuildTree}(I^-_{j^*,s^*}, k), \text{BuildTree}(I^+_{j^*,s^*}, k))$
```



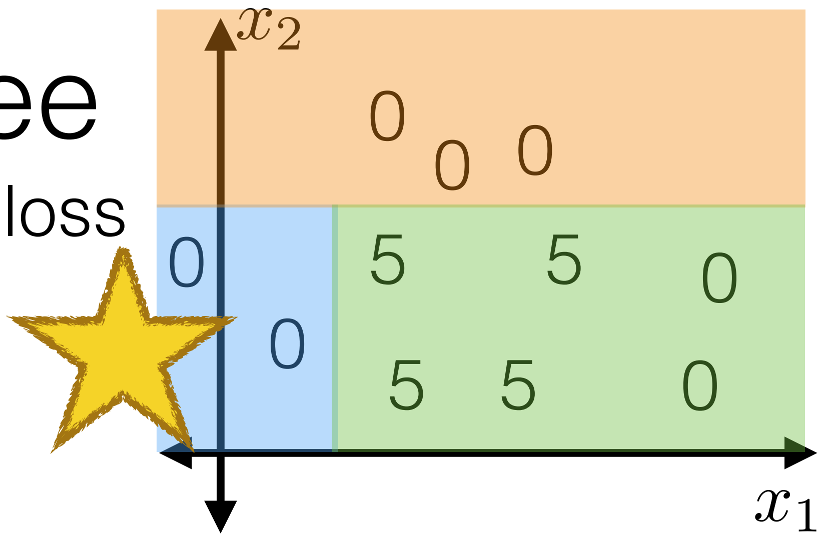
`BuildTree({1,…,n};2)`



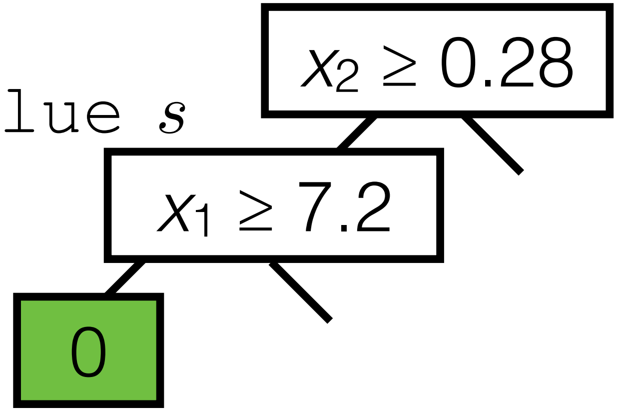
# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$; $k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node($j^*, s^*$, BuildTree($I^-_{j^*,s^*}, k$), BuildTree($I^+_{j^*,s^*}, k$))
```



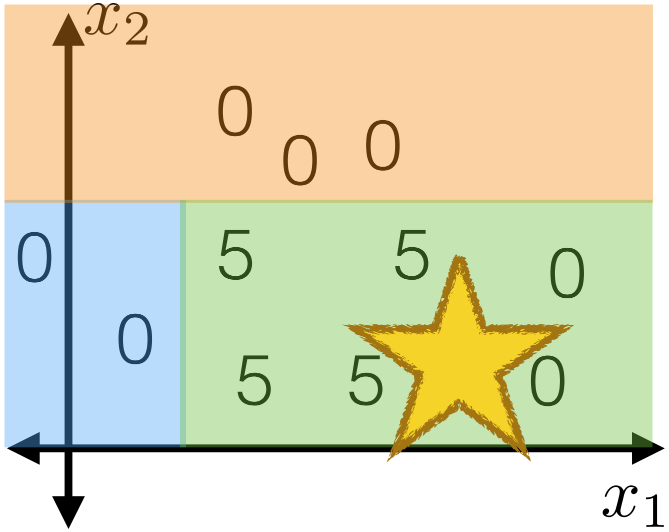
BuildTree({1,…,n};2)



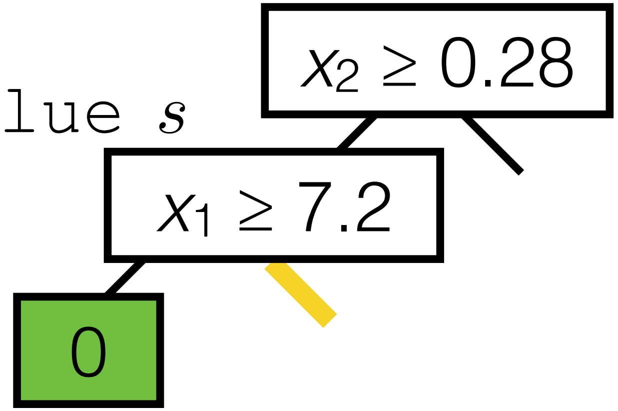
# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| $\leq$ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I^{+}_{j,s} = \{i \in I | x_j^{(i)} \geq s\}$
      Set $I^{-}_{j,s} = \{i \in I | x_j^{(i)} < s\}$
      Set $\hat{y}^{+}_{j,s} = \text{average}_{i \in I^{+}_{j,s}} y^{(i)}$
      Set $\hat{y}^{-}_{j,s} = \text{average}_{i \in I^{-}_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^{+}_{j,s}} (y^{(i)} - \hat{y}^{+}_{j,s})^2 + \sum_{i \in I^{-}_{j,s}} (y^{(i)} - \hat{y}^{-}_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node ($j^*, s^*$, $\text{BuildTree}(I^{-}_{j^*,s^*}, k), \text{BuildTree}(I^{+}_{j^*,s^*}, k)$)
```



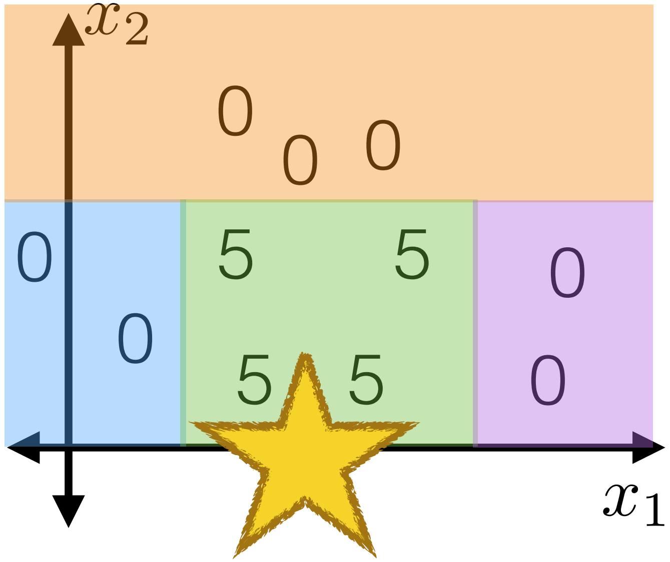
BuildTree({1,…,n};2)



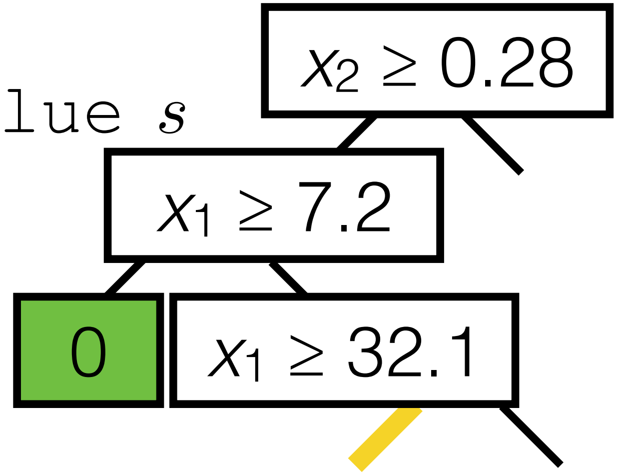
# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node $(j^*, s^*, \text{BuildTree}(I^-_{j^*,s^*}, k), \text{BuildTree}(I^+_{j^*,s^*}, k))$
```



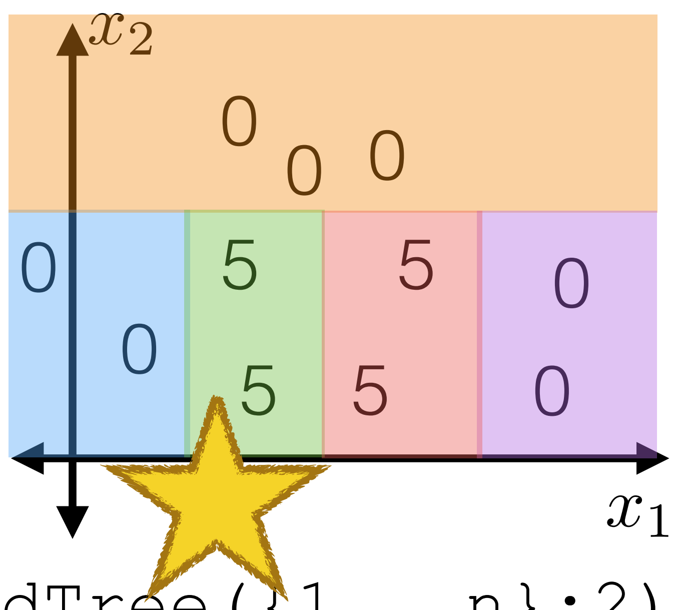
BuildTree({1,…,n};2)



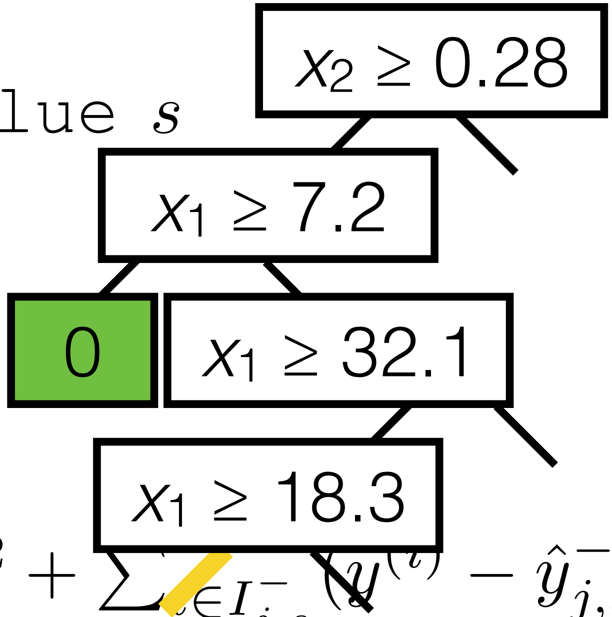
# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node $(j^*, s^*, \text{BuildTree}(I^-_{j^*,s^*}, k), \text{BuildTree}(I^+_{j^*,s^*}, k))$
```



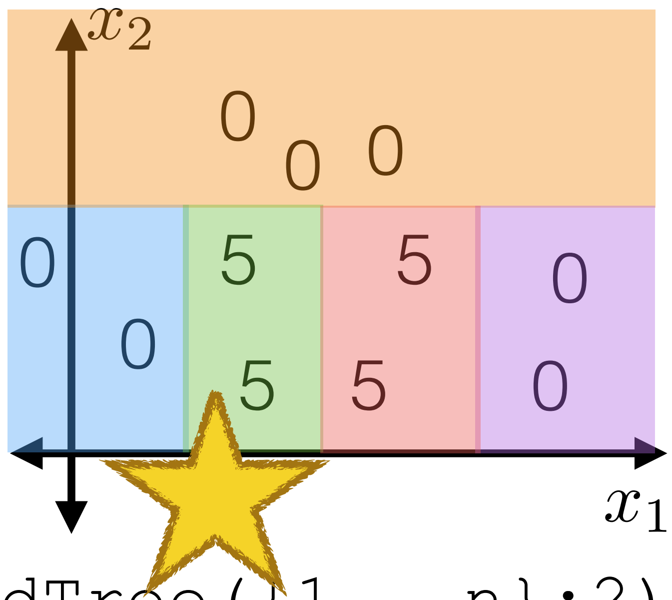
BuildTree({1,…,n};2)



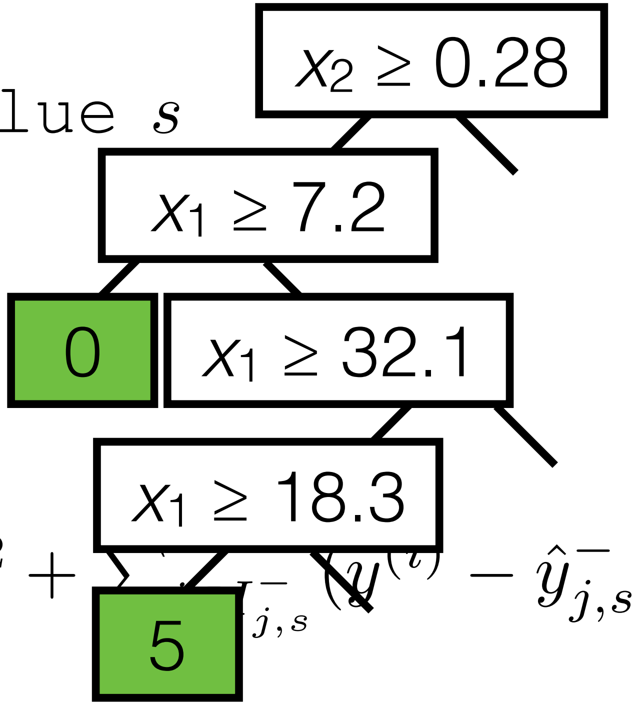
# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i\in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i\in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i\in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i\in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i\in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node $(j^*, s^*, \text{BuildTree}(I^-_{j^*,s^*}, k), \text{BuildTree}(I^+_{j^*,s^*}, k))$
```



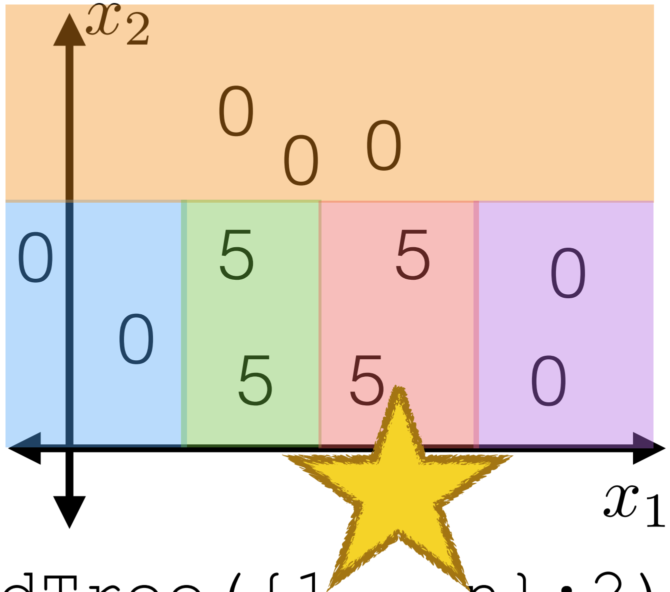
BuildTree({1,…,n};2)



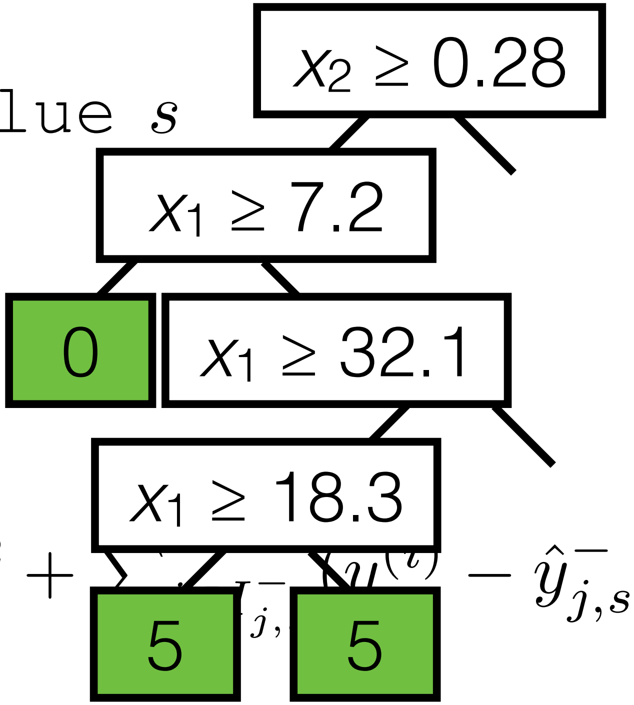
# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node $(j^*, s^*, \text{BuildTree}(I^-_{j^*,s^*}, k), \text{BuildTree}(I^+_{j^*,s^*}, k))$
```

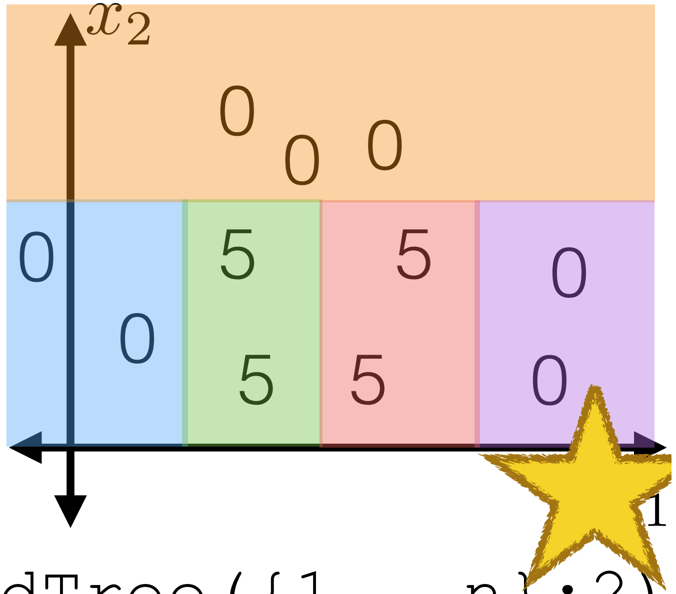


BuildTree({1,…,n};2)



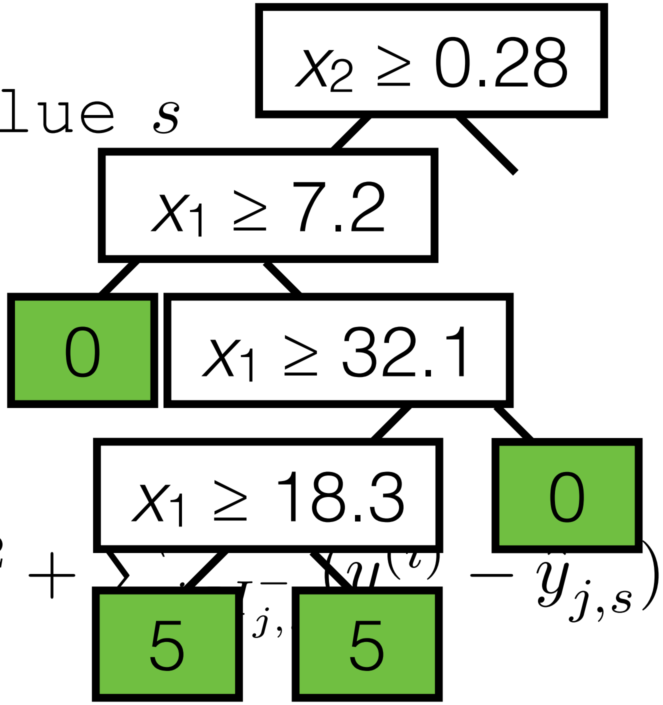
# Building a decision tree

- Regression tree with squared error loss



BuildTree({1,…,n};2)

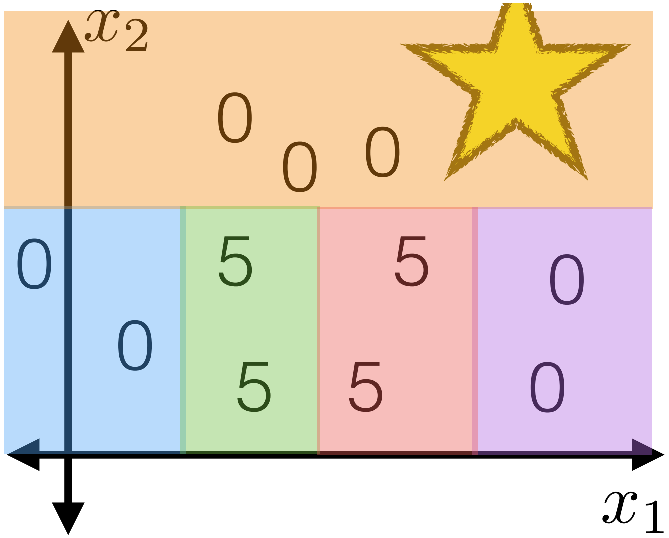
```
BuildTree($I$;$k$)
  if |$I$| $\leq$ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node($j^*, s^*, \text{BuildTree}(I^-_{j^*,s^*}, k), \text{BuildTree}(I^+_{j^*,s^*}, k)$)
```



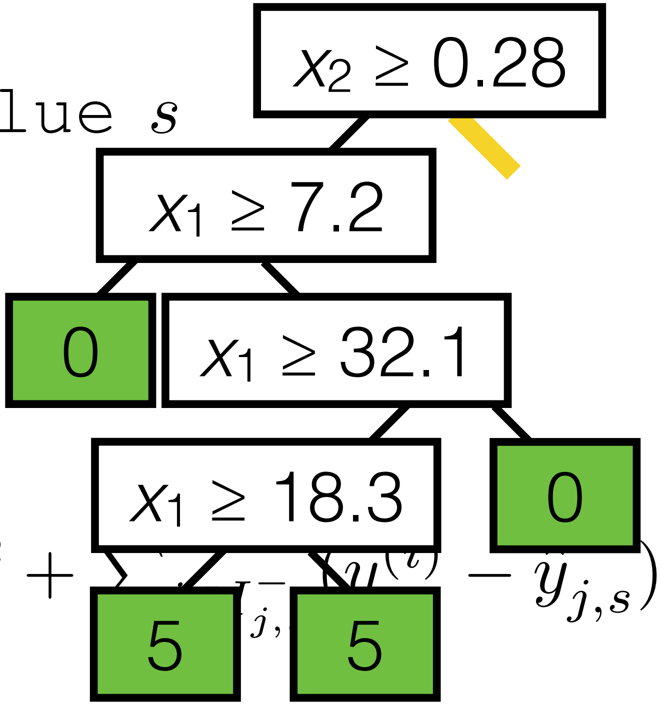
# Building a decision tree

- Regression tree with squared error loss

```
BuildTree($I$;$k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node($j^*, s^*, \text{BuildTree}(I^-_{j^*,s^*}, k), \text{BuildTree}(I^+_{j^*,s^*}, k)$)
```

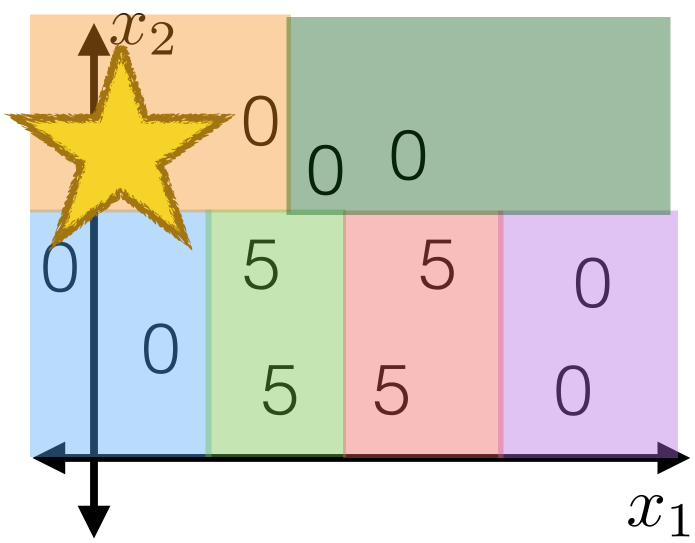


BuildTree({1,…,n};2)



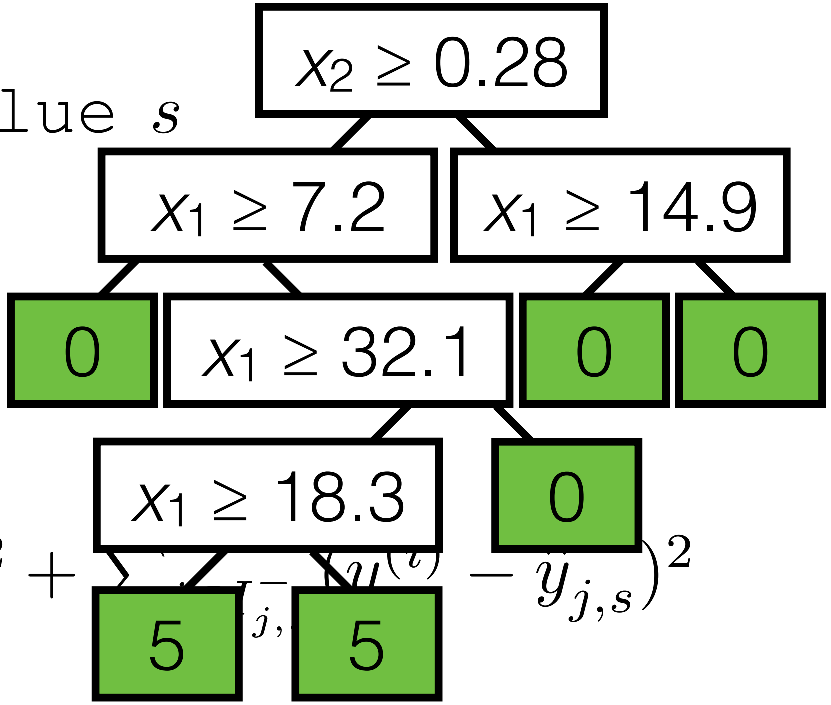
# Building a decision tree

- Regression tree with squared error loss



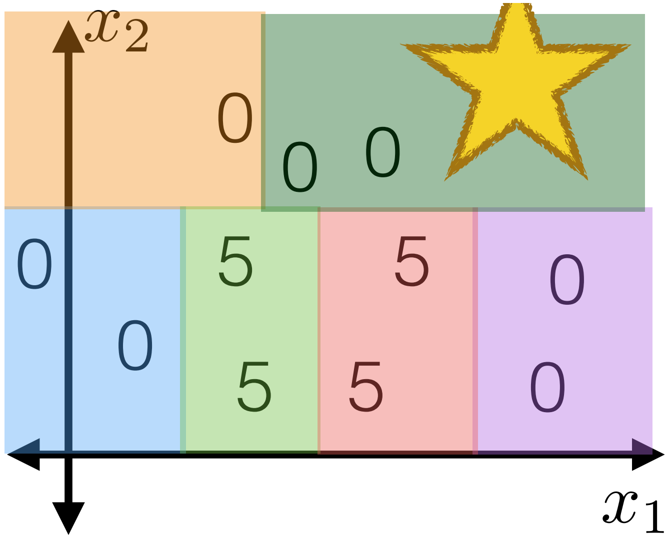
```
BuildTree($I$;$k$)
  if |$I$| ≤ k
    Set $\hat{y}$ = $\text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node($j^*, s^*, \text{BuildTree}(I^-_{j^*,s^*}, k), \text{BuildTree}(I^+_{j^*,s^*}, k)$)
```

BuildTree({1,…,n};2)



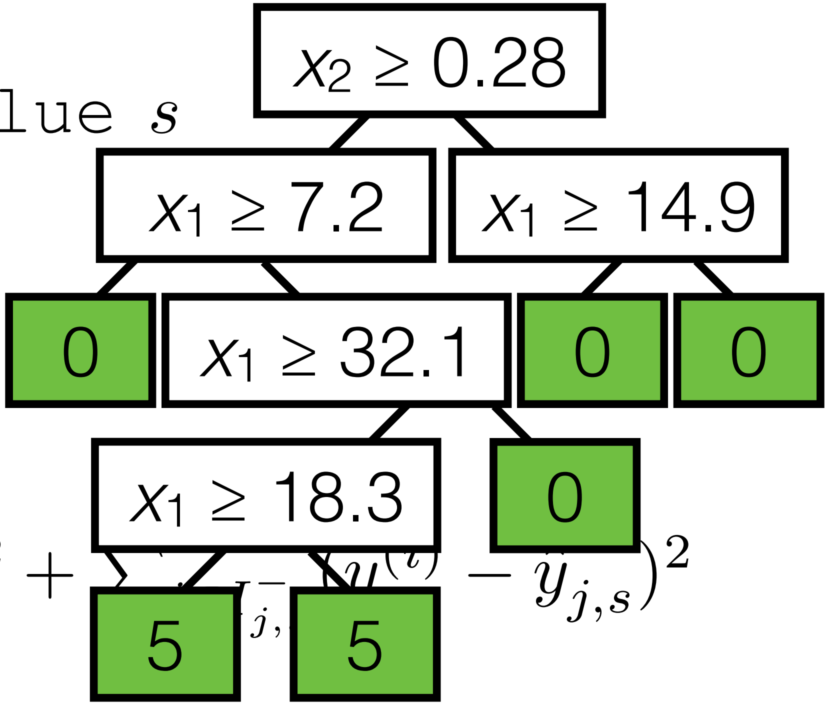
# Building a decision tree

- Regression tree with squared error loss



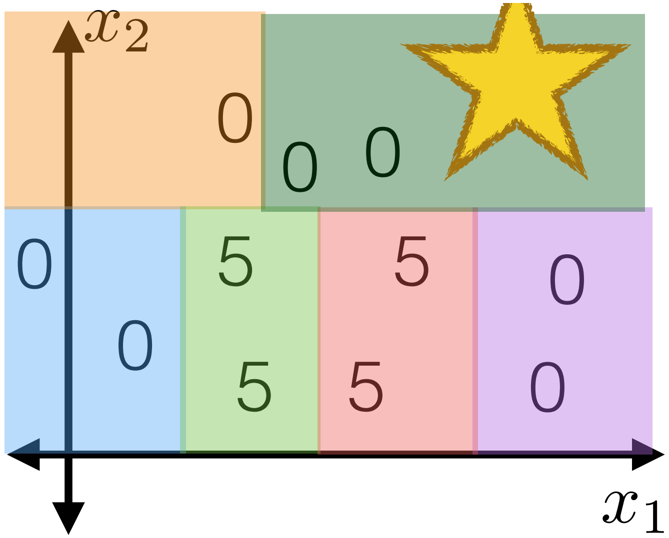
```
BuildTree(I;k)
  if |I| ≤ k
    Set $\hat{y} = \text{average}_{i \in I} y^{(i)}$
    return Leaf(label = $\hat{y}$)
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node $(j^*, s^*, \text{BuildTree}(I^-_{j^*,s^*}, k), \text{BuildTree}(I^+_{j^*,s^*}, k))$
```

BuildTree({1,…,n};2)



# Building a decision tree

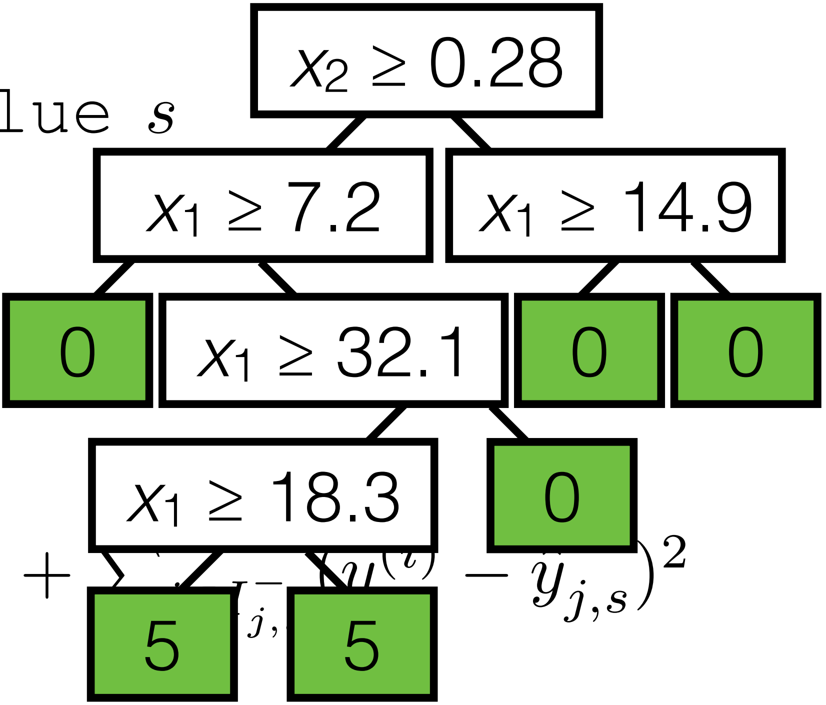
- Regression tree with squared error loss



```
BuildTree($I$;$k$)
  if $|I| <$
                                    )
                 (label = $\hat{y}$ )
  else
    for each split dim $j$ & value $s$
      Set $I^+_{j,s} = \{i \in I | x^{(i)}_j \geq s\}$
      Set $I^-_{j,s} = \{i \in I | x^{(i)}_j < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node $(j^*, s^*, \text{BuildTree}(I^-_{j^*,s^*}, k), \text{BuildTree}(I^+_{j^*,s^*}, k))$
```

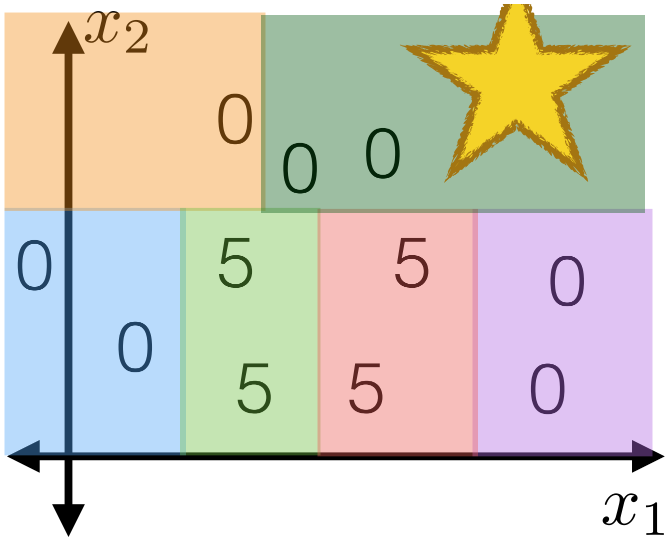
Why keep splitting after our error is low?

BuildTree({1,…,n};2)



# Building a decision tree

- Regression tree with squared error loss

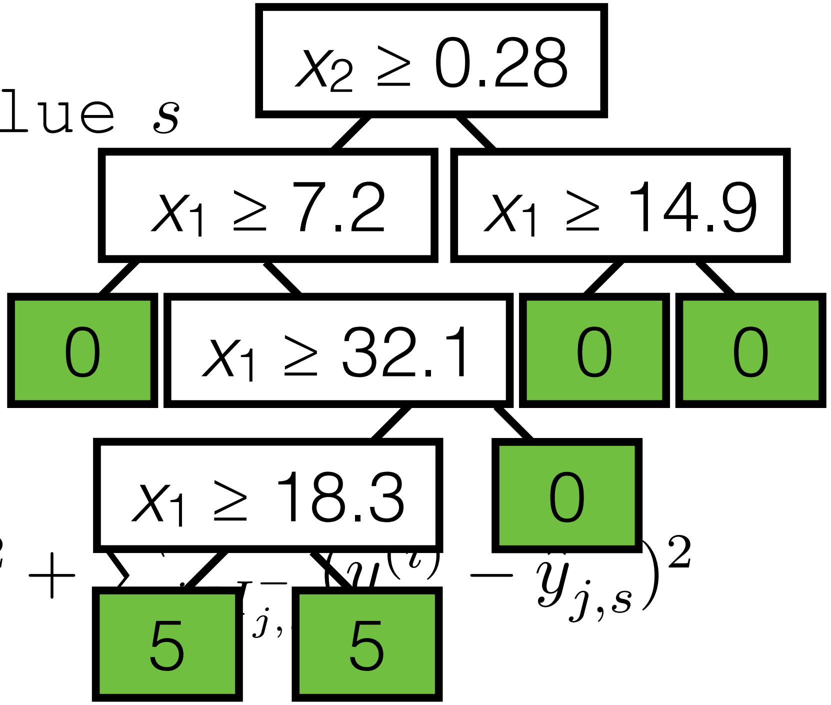


```
BuildTree($I;k$)
  if $|I| <$
     label = $\hat{y}$ )
  else
    f                    n $j$ & value $s$
                            $\geq s\}$
      Set $I^-_{j,s} = \{i \in I | x_j^{(i)} < s\}$
      Set $\hat{y}^+_{j,s} = \text{average}_{i \in I^+_{j,s}} y^{(i)}$
      Set $\hat{y}^-_{j,s} = \text{average}_{i \in I^-_{j,s}} y^{(i)}$
      Set $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum_{i \in I^-_{j,s}} (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node $(j^*, s^*, \text{BuildTree}(I^-_{j^*,s^*}, k), \text{BuildTree}(I^+_{j^*,s^*}, k))$
```

Why keep splitting after our error is low?

Is overfitting an issue?

BuildTree({1,…,n};2)



# Building a decision tree

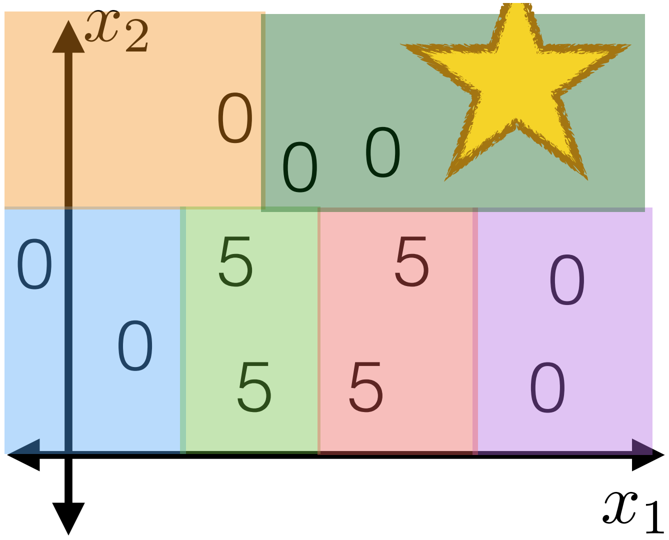
- Regression tree with squared error loss

```
BuildTree($I;k$)
  if |$I$| ≤
    …er(label = $\hat{y}$ )
  else
    f…n $j$ & value $s$
      … $x_j^{(i)} \ge s\}$
      Set $I^-_{j,s} = \{i \in I | x_j^{(i)} < s\}$
      Set … $y^{(i)}$
      … $y^{(i)}$
      … $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + \sum … (y^{(i)} - \hat{y}^-_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node $(j^*, s^*, \text{BuildTree}(I^-_{j^*,s^*}, k), \text{BuildTree}(I^+_{j^*,s^*}, k))$
```

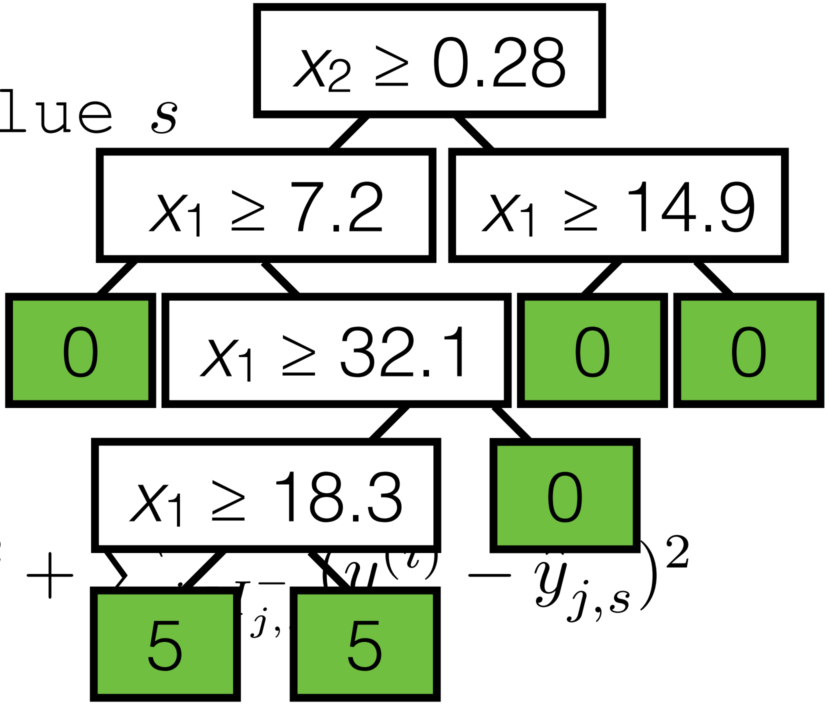
Why keep splitting after our error is low?

Is overfitting an issue?

Idea: stop when no/small change in loss



BuildTree({1,…,n};2)



# Building a decision tree

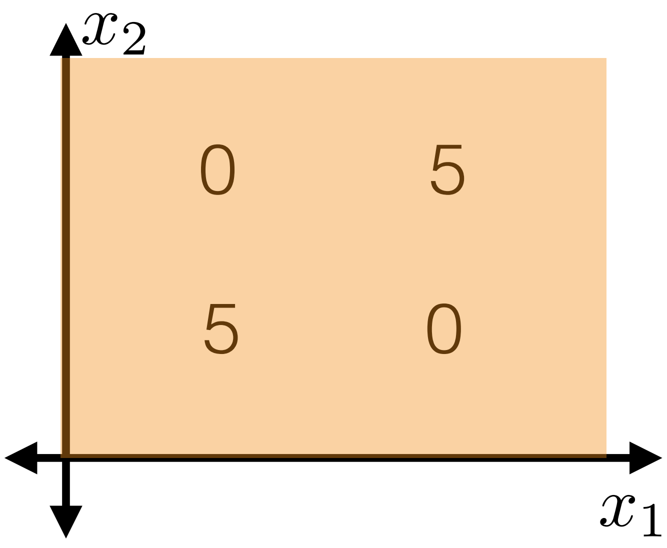
- Regression tree with squared error loss

```
BuildTree($I; k$)
  if $|I| <$
    …)
    …(label = $\hat{y}$ )
  else
    f…  …n $j$ & value $s$
      … $\geq s\}$
      Set $I^-_{j,s} = \{i \in I | x … < s\}$
      Se… $y^{(i)}$
      … $y^{(i)}$
      $E_{j,s} = \sum_{i \in I^+_{j,s}} (y^{(i)} - \hat{y}^+_{j,s})^2 + … (y^{(i)} - \hat{y}_{j,s})^2$
    Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
    return Node $(j^*, s^*, \mathrm{BuildTree}(I^-_{j^*,s^*}, k), \mathrm{BuildTree}(I^+_{j^*,s^*}, k))$
```

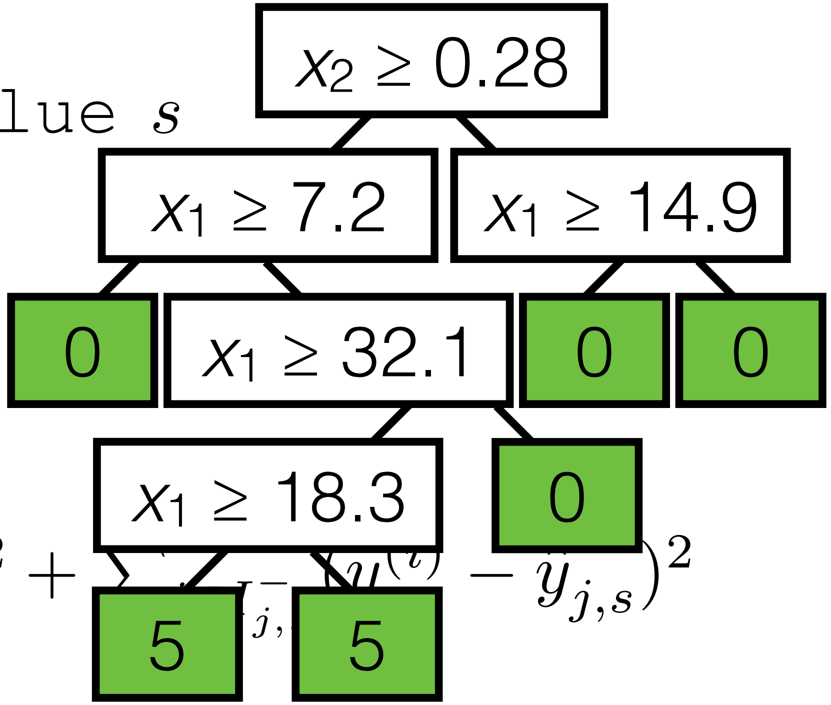
Why keep splitting after our error is low?

Is overfitting an issue?

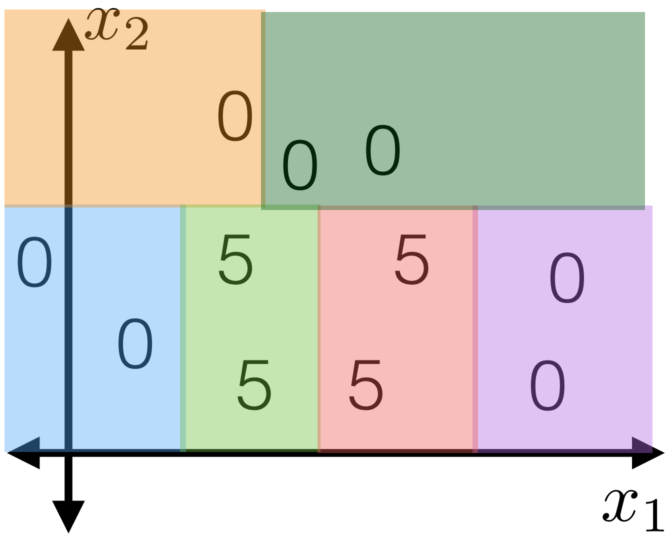
Idea: stop when no/ small change in loss



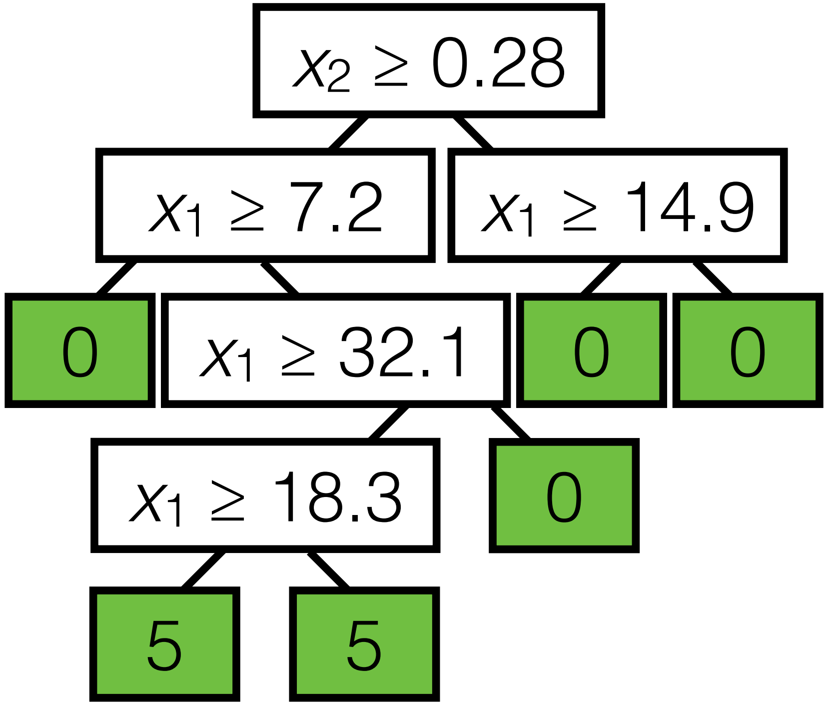
BuildTree({1,…,n};2)



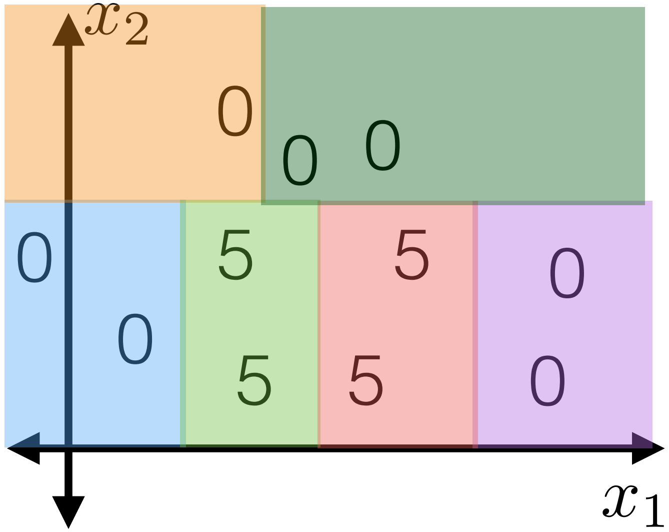
# How to regularize?



`BuildTree({1,…,n};2)`

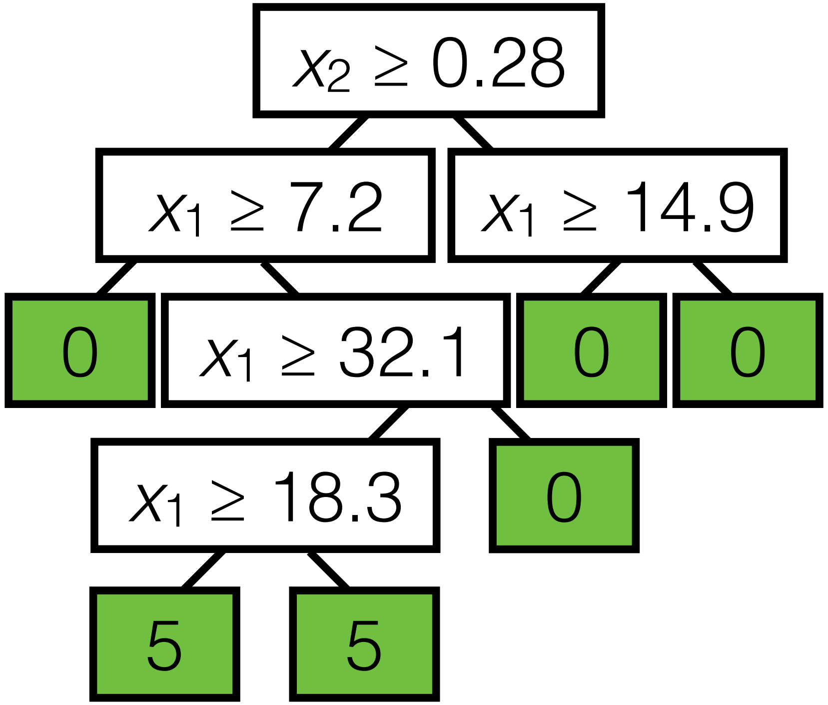


# How to regularize?



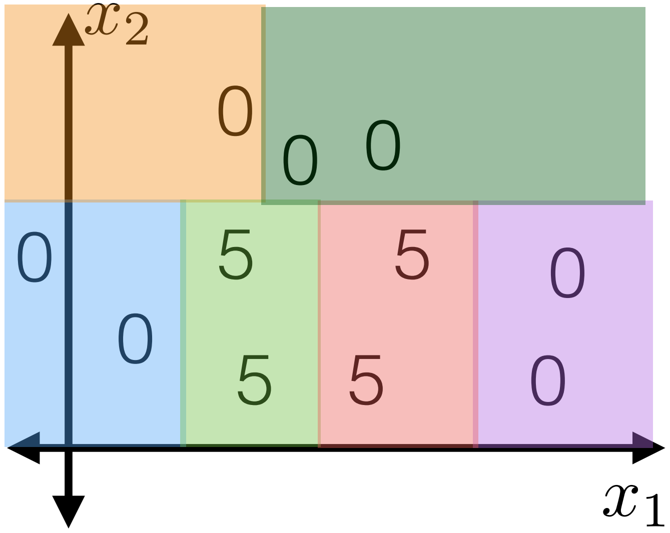
- Objective = training loss
  + constant * regularizer

`BuildTree({1,…,n};2)`

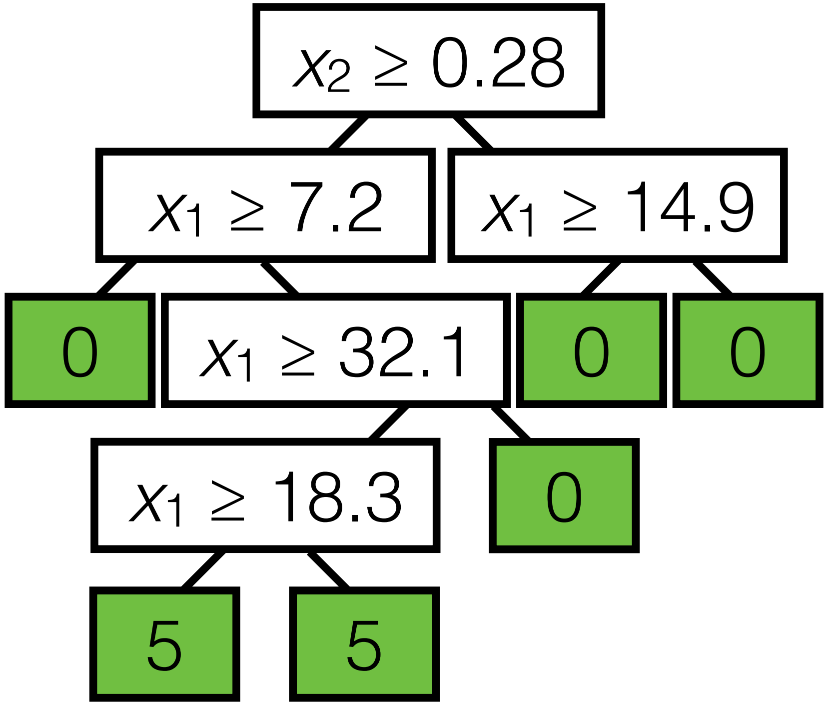


# How to regularize?

$$C_\alpha(T) = \sum_{i=1}^{n} L(T(x^{(i)}), y^{(i)}) + \alpha \overbrace{|T|}^{= \# \text{ leaves}}$$

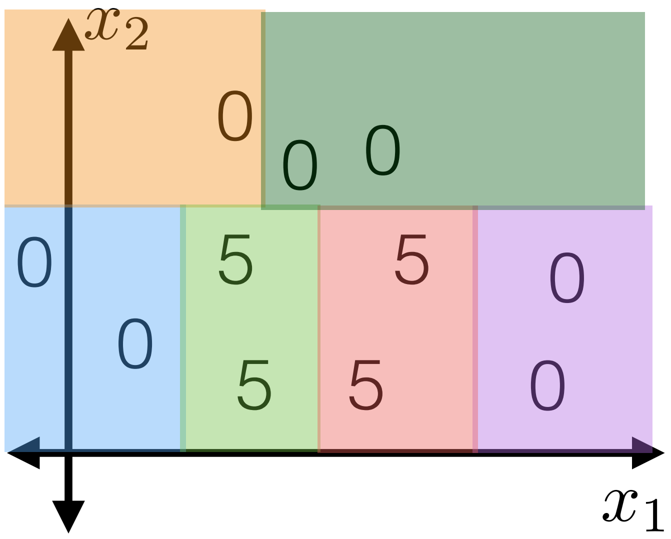


`BuildTree({1,…,n};2)`

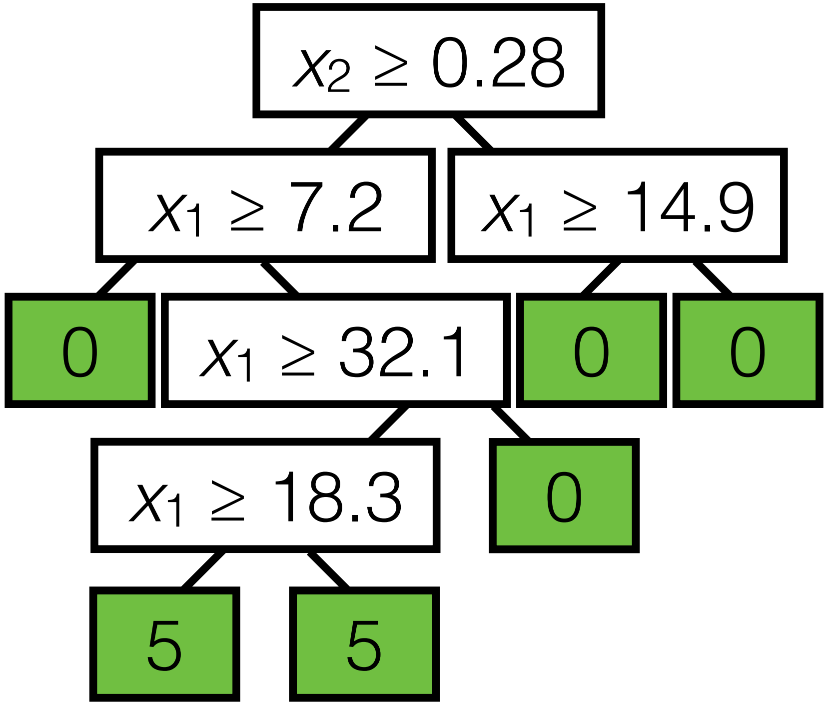

$x_2 \geq 0.28$
$x_1 \geq 7.2$
$x_1 \geq 14.9$
0
$x_1 \geq 32.1$
0
0
$x_1 \geq 18.3$
0
5
5

# How to regularize?

$$C_\alpha(T) = \sum_{i=1}^{n} L(T(x^{(i)}), y^{(i)}) + \alpha \overbrace{|T|}^{= \text{\# leaves}}$$

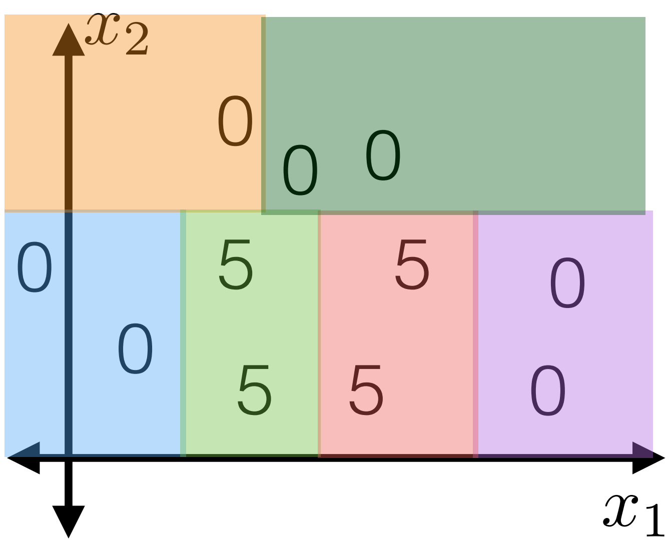


`BuildTree({1,…,n};2)`

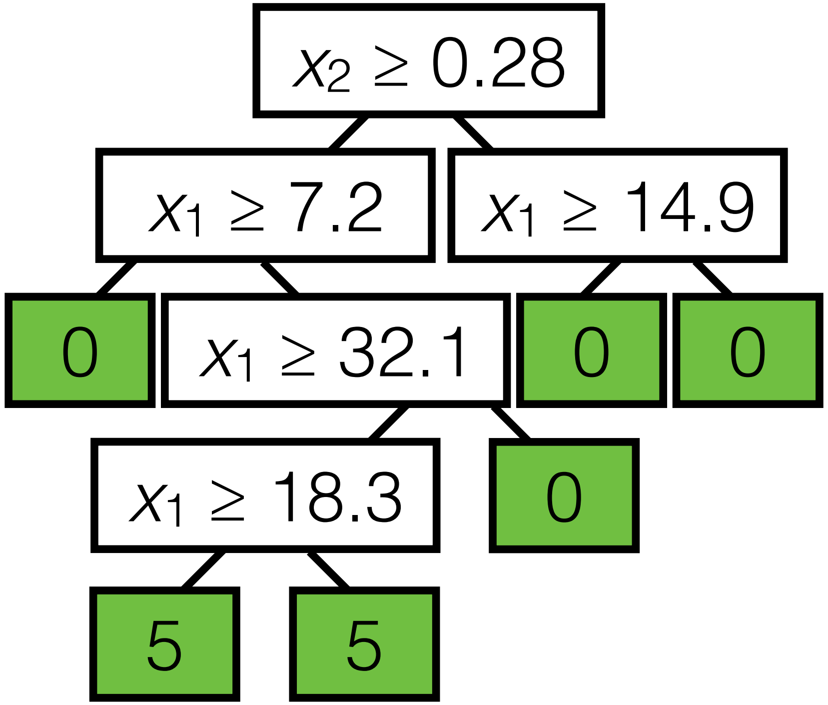


# How to regularize?

$$C_\alpha(T) = \sum_{i=1}^{n} L(T(x^{(i)}), y^{(i)}) + \alpha \overbrace{|T|}^{= \#\text{ leaves}}$$

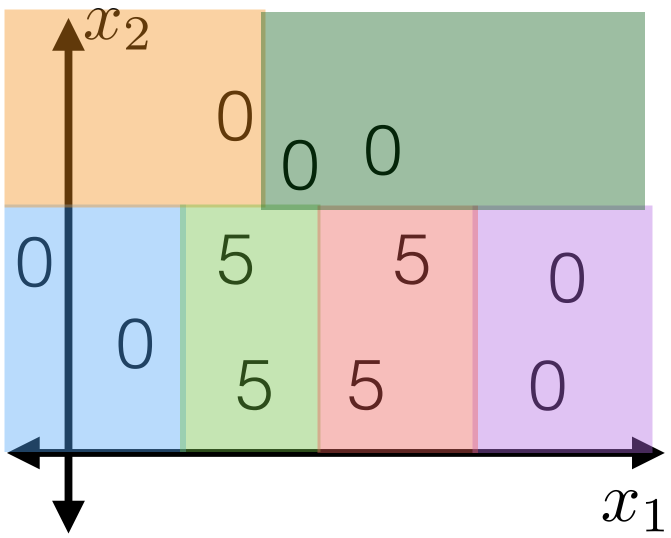


`BuildTree({1,…,n};2)`

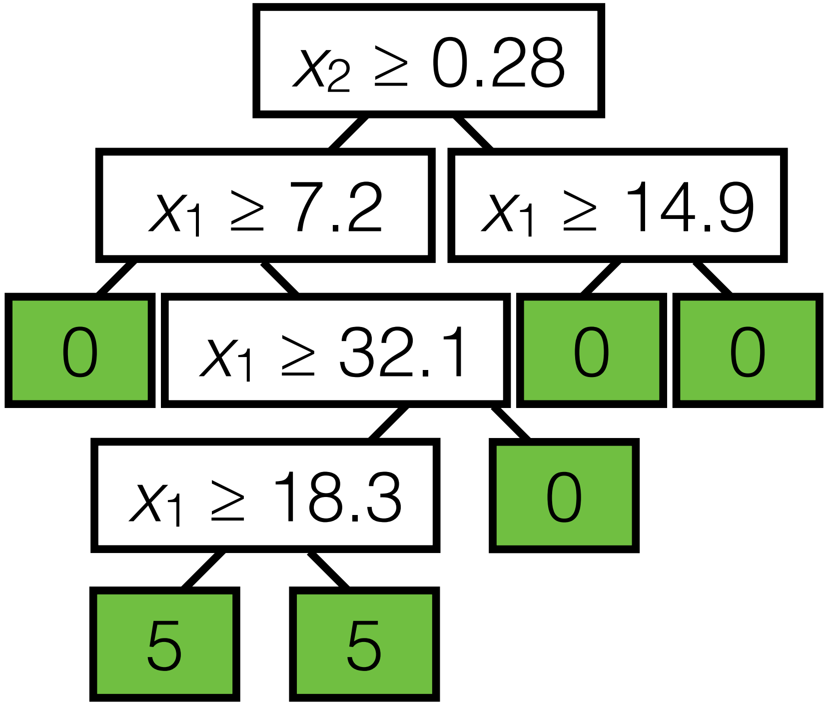


# How to regularize?

$$C_\alpha(T) = \sum_{i=1}^{n} L(T(x^{(i)}), y^{(i)}) + \alpha \overbrace{|T|}^{= \# \text{ leaves}}$$

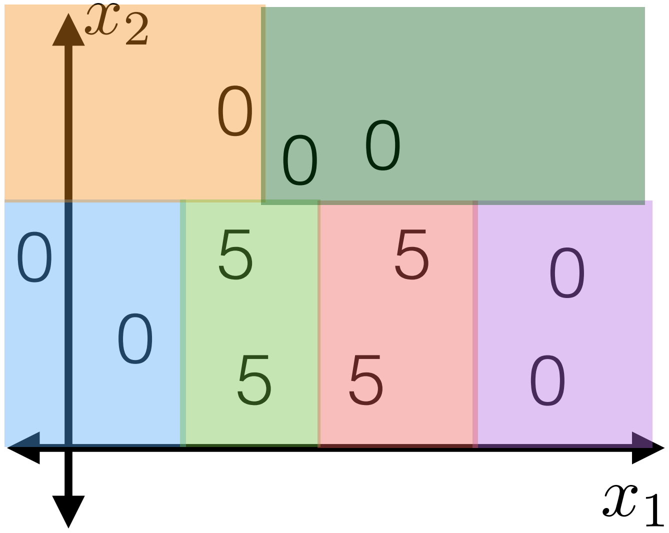


BuildTree({1,…,n};2)

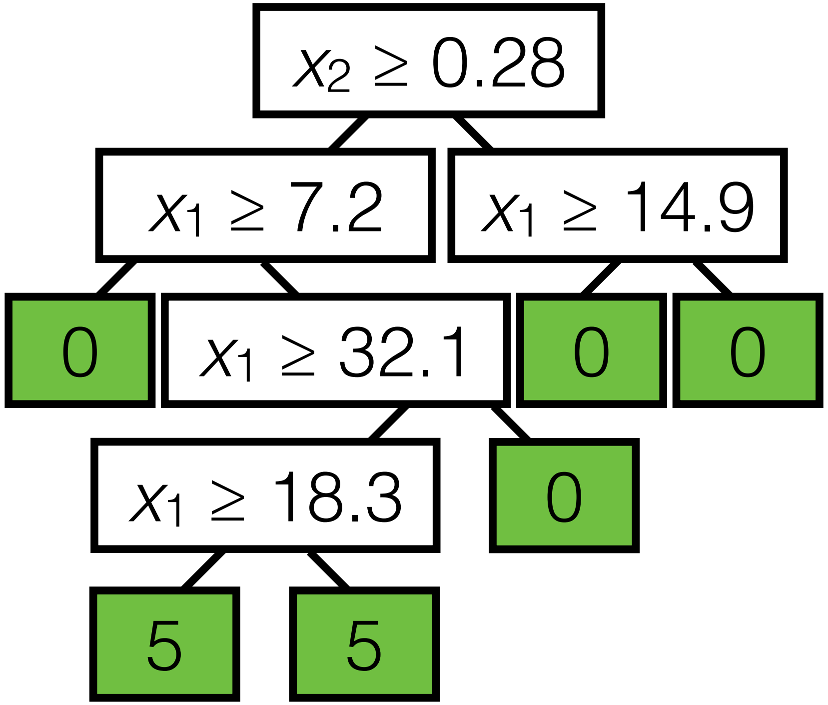


# How to regularize?

$$C_\alpha(T) = \sum_{i=1}^{n} L(T(x^{(i)}), y^{(i)}) + \alpha \overbrace{|T|}^{= \# \text{ leaves}}$$



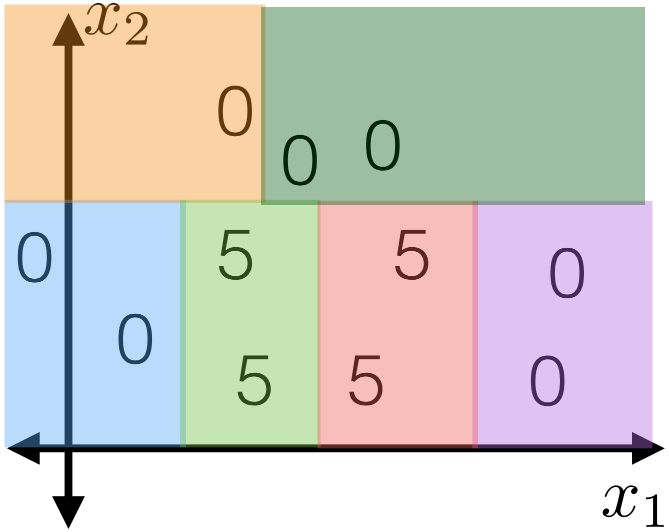
`BuildTree({1,…,n};2)`



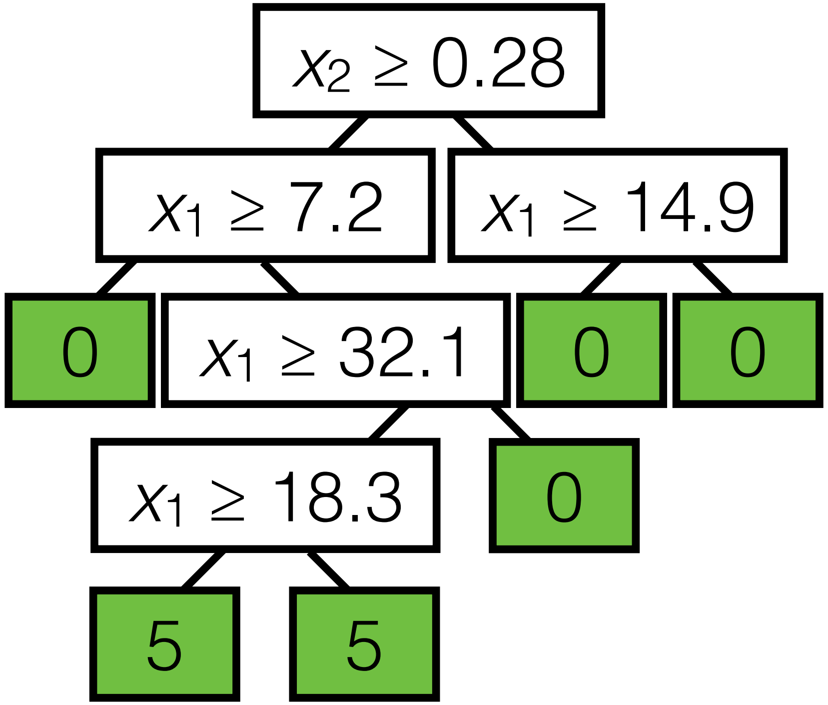
# How to regularize?

- "Cost complexity" of a tree $T$

$$C_\alpha(T) = \sum_{i=1}^{n} L(T(x^{(i)}), y^{(i)}) + \alpha \overbrace{|T|}^{= \#\ \text{leaves}}$$



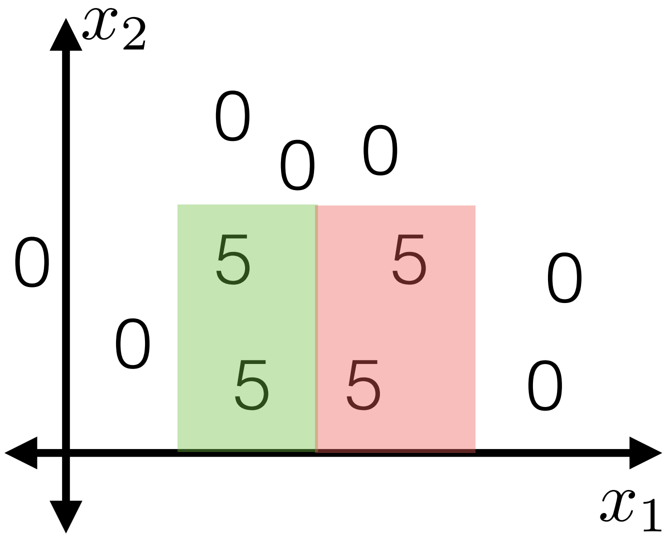
`BuildTree({1,…,n};2)`



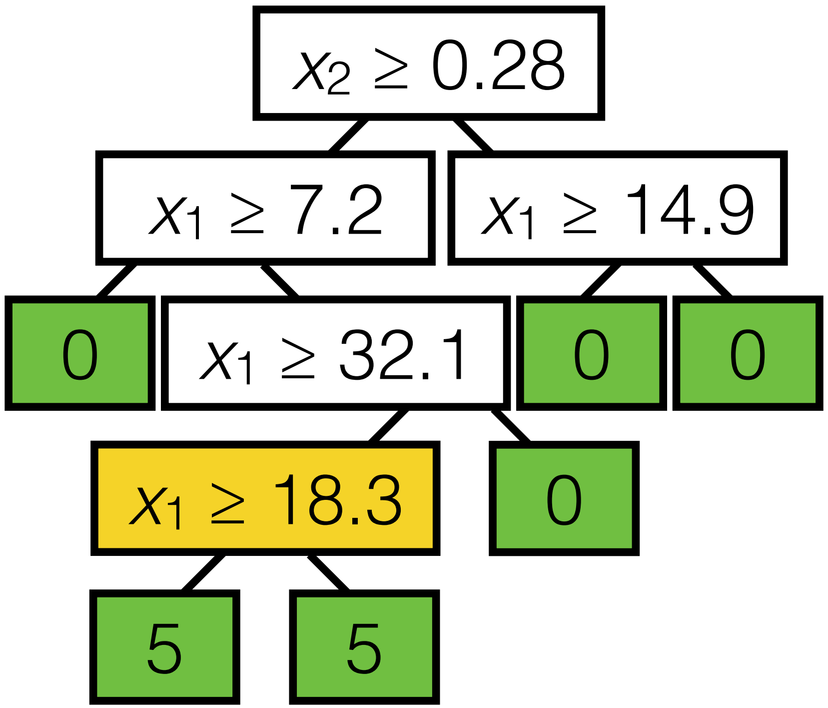
# How to regularize?

- “Cost complexity” of a tree *T*

$$C_\alpha(T) = \sum_{i=1}^{n} L(T(x^{(i)}), y^{(i)}) + \alpha \overbrace{|T|}^{= \text{\# leaves}}$$



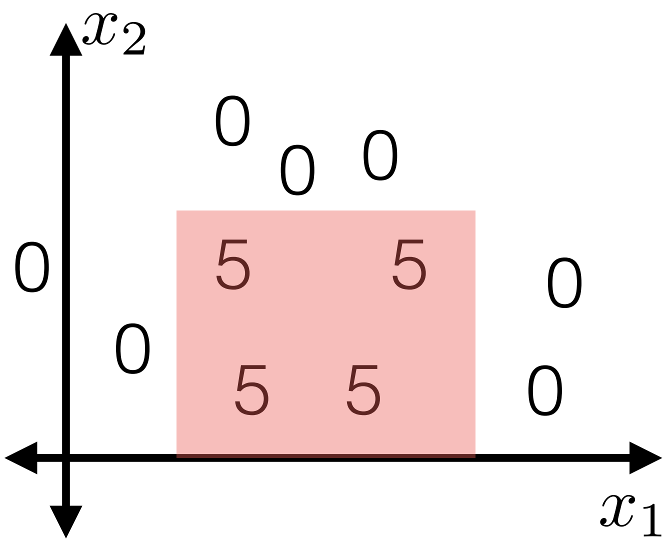
`BuildTree({1,…,n};2)`



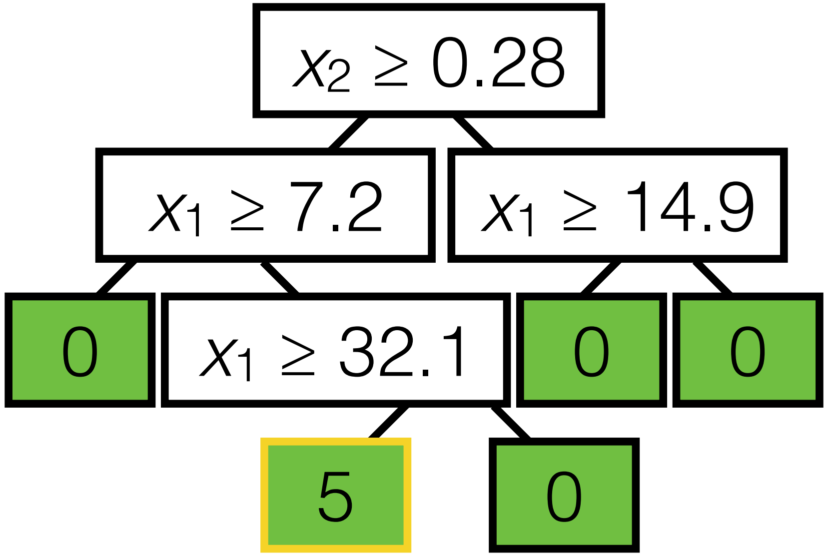
# How to regularize?

- "Cost complexity" of a tree $T$

$$C_\alpha(T) = \sum_{i=1}^{n} L(T(x^{(i)}), y^{(i)}) + \alpha \overbrace{|T|}^{= \text{\# leaves}}$$



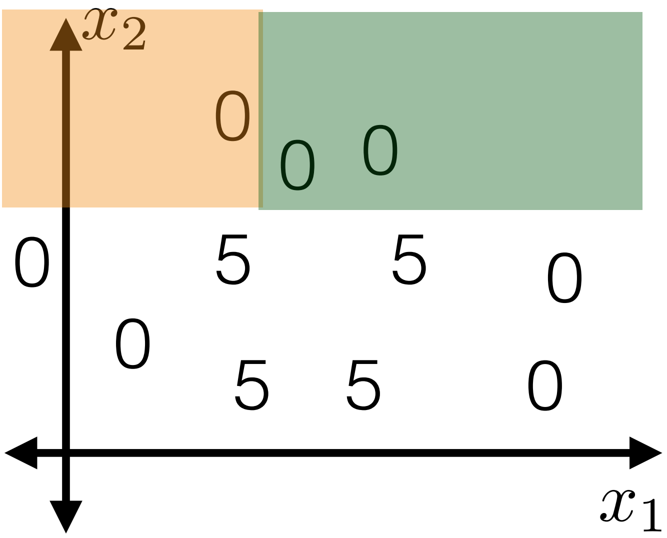
`BuildTree({1,…,n};2)`



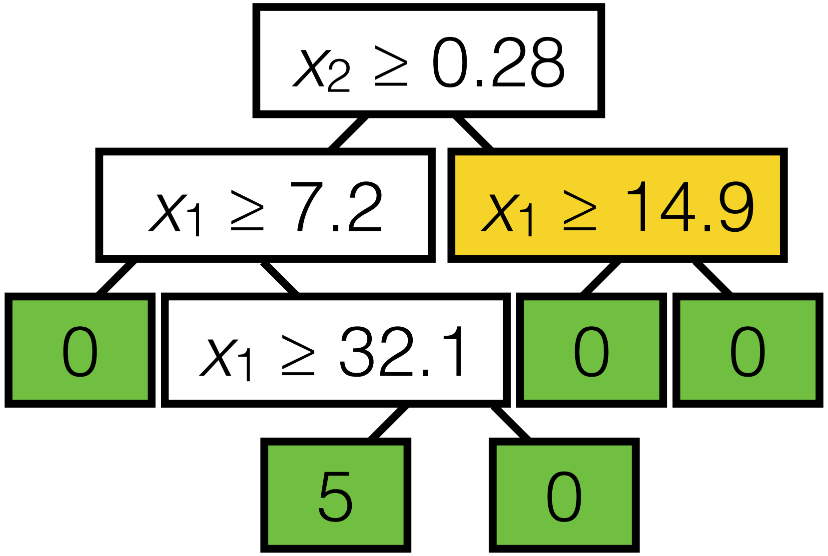
# How to regularize?

- “Cost complexity” of a tree $T$

$$C_\alpha(T) = \sum_{i=1}^{n} L(T(x^{(i)}), y^{(i)}) + \alpha \overbrace{|T|}^{= \#\text{ leaves}}$$



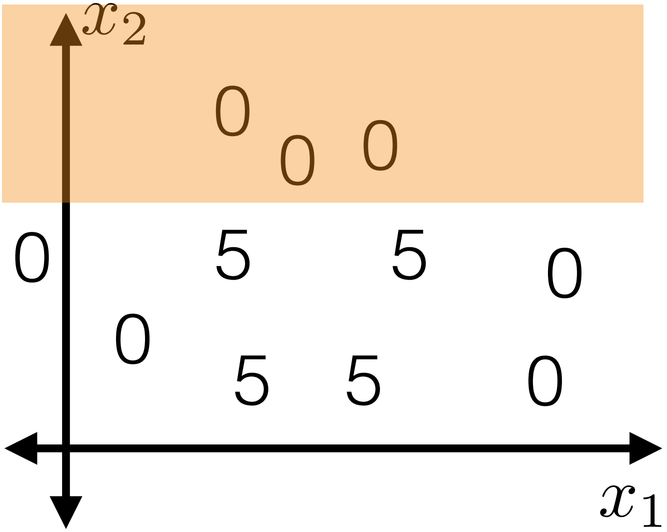
`BuildTree({1,…,n};2)`



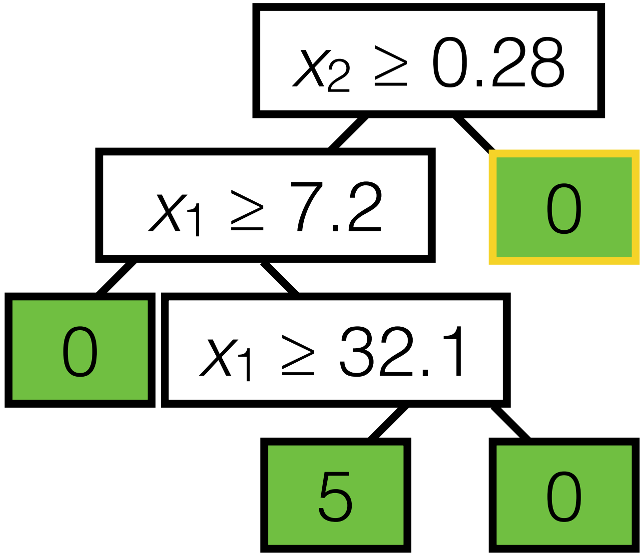
# How to regularize?

- “Cost complexity” of a tree *T*

$$C_\alpha(T) = \sum_{i=1}^{n} L(T(x^{(i)}), y^{(i)}) + \alpha \overbrace{|T|}^{= \# \text{ leaves}}$$



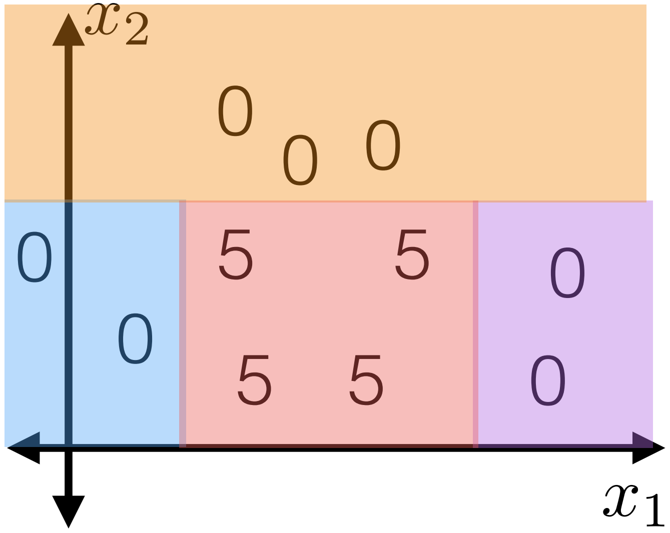
`BuildTree({1,…,n};2)`



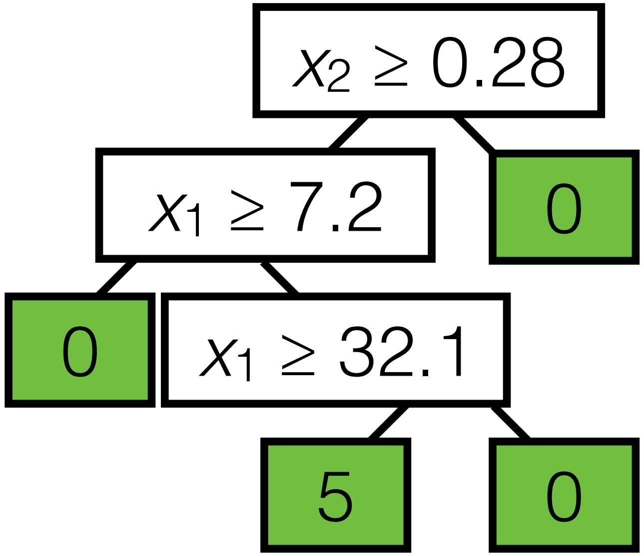
# How to regularize?

- “Cost complexity” of a tree $T$

$$C_\alpha(T) = \sum_{i=1}^{n} L(T(x^{(i)}), y^{(i)}) + \alpha \overbrace{|T|}^{= \text{\# leaves}}$$



`BuildTree({1,…,n};2)`



$x_2 \geq 0.28$

$x_1 \geq 7.2$

0

0

$x_1 \geq 32.1$

5

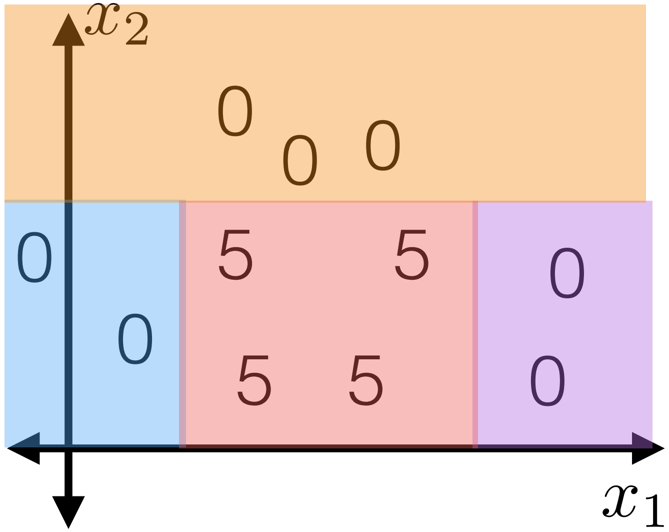
0

# How to regularize?

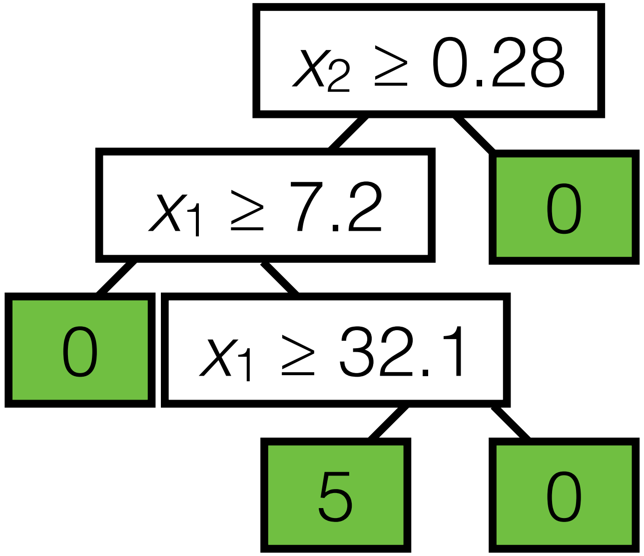
- "Cost complexity" of a tree $T$

$$C_\alpha(T) = \sum_{i=1}^{n} L(T(x^{(i)}), y^{(i)}) + \alpha \overbrace{|T|}^{= \#\text{ leaves}}$$

- Pruning



`BuildTree({1,…,n};2)`

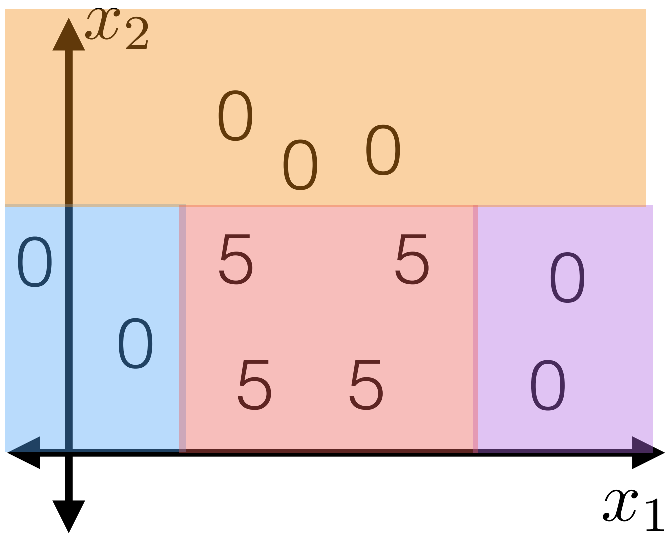


# How to regularize?

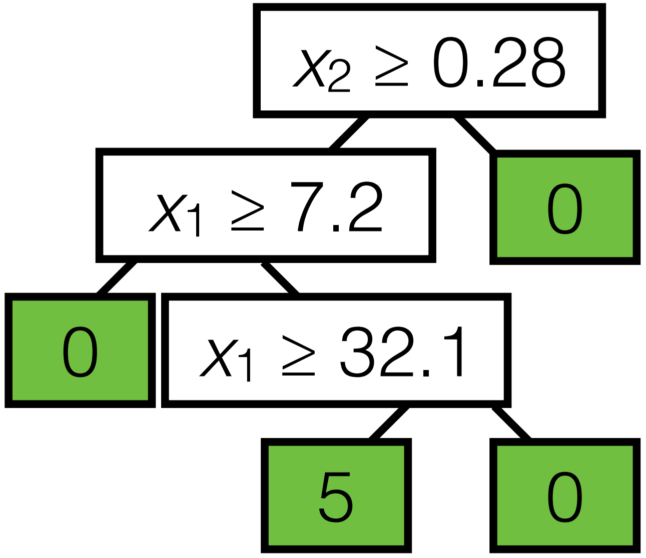
- “Cost complexity” of a tree *T*

$$C_\alpha(T) = \sum_{i=1}^{n} L(T(x^{(i)}), y^{(i)}) + \alpha \overbrace{|T|}^{= \text{\# leaves}}$$

- Pruning
    - For each $\alpha$, choose $T_\alpha$ by pruning subtrees until it’s not worthwhile



`BuildTree({1,…,n};2)`

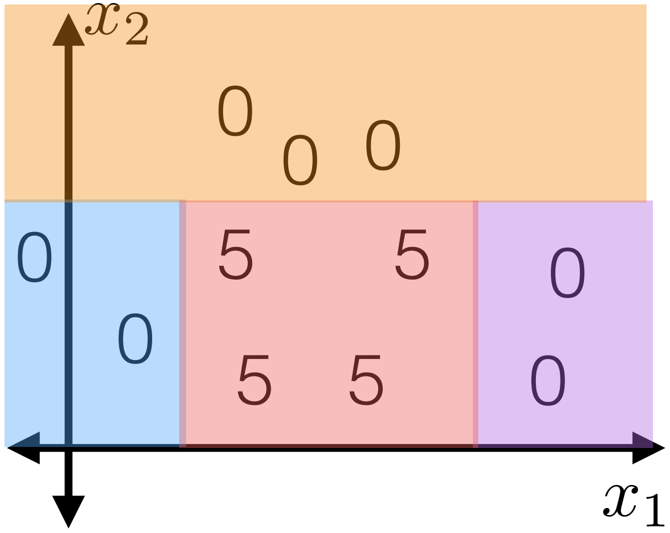


# How to regularize?

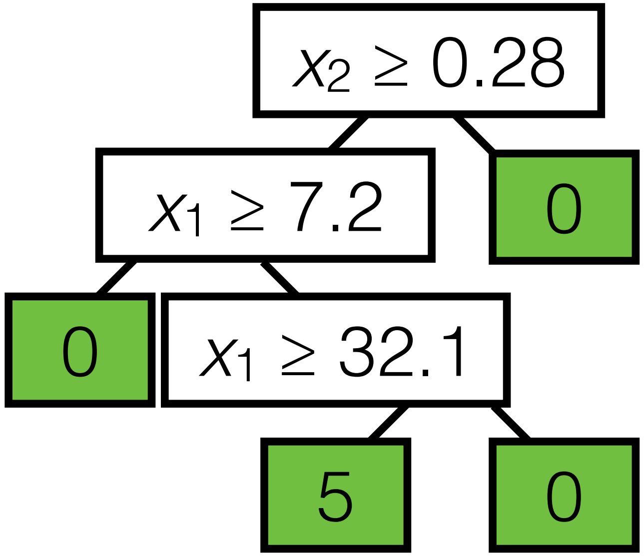
- "Cost complexity" of a tree $T$

$$C_\alpha(T) = \sum_{i=1}^{n} L(T(x^{(i)}), y^{(i)}) + \alpha \overbrace{|T|}^{= \# \text{ leaves}}$$

- Pruning
  - For each $\alpha$, choose $T_\alpha$ by pruning subtrees until it's not worthwhile
  - Choose a final tree by cross validation



`BuildTree({1,…,n};2)`



# Bagging

# Bagging

- One of multiple ways to make and use an ensemble

# Bagging

- One of multiple ways to make and use an ensemble
- Bagging = **B**ootstrap **agg**regat**ing**

# Bagging

- One of multiple ways to make and use an ensemble
- Bagging = **B**ootstrap **agg**regat**ing**
    - Training data $\mathcal{D}_n$

# Bagging

- One of multiple ways to make and use an ensemble
- Bagging = **B**ootstrap **agg**regat**ing**
  - Training data $\mathcal{D}_n$
  - For $b$ = 1, …, $B$

# Bagging

- One of multiple ways to make and use an ensemble
- Bagging = **B**ootstrap **agg**regat**ing**
    - Training data $\mathcal{D}_n$
    - For *b* = 1, …, *B*
        - Draw a new "data set" $\tilde{\mathcal{D}}_n^{(b)}$ of size *n* by sampling *with* replacement from $\mathcal{D}_n$

# Bagging

- One of multiple ways to make and use an ensemble
- Bagging = **B**ootstrap **agg**regat**ing**
    - Training data $\mathcal{D}_n$
    - For *b* = 1, …, *B*
        - Draw a new “data set” $\tilde{\mathcal{D}}_n^{(b)}$ of size *n* by sampling *with* replacement from $\mathcal{D}_n$

- Sampling with replacement

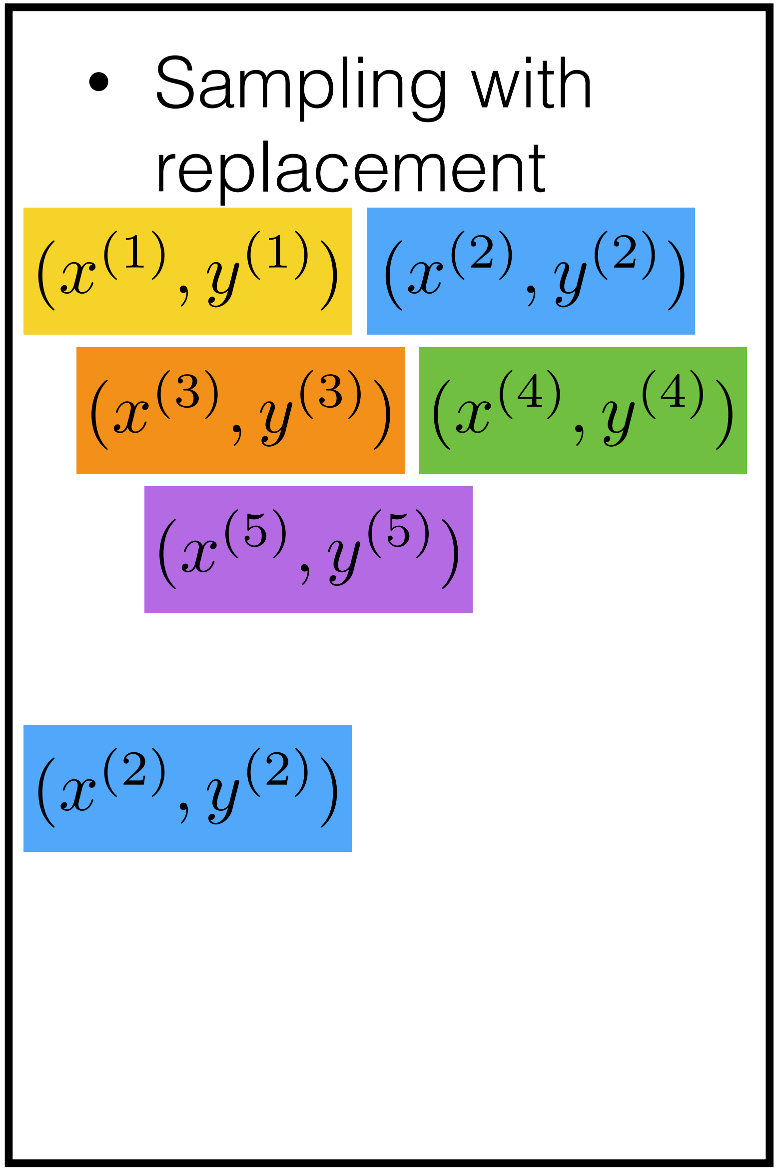
$(x^{(1)}, y^{(1)})$ $(x^{(2)}, y^{(2)})$

$(x^{(3)}, y^{(3)})$ $(x^{(4)}, y^{(4)})$

$(x^{(5)}, y^{(5)})$

# Bagging

- One of multiple ways to make and use an ensemble
- Bagging = **B**ootstrap **agg**regat**ing**
  - Training data $\mathcal{D}_n$
  - For *b* = 1, …, *B*
    - Draw a new “data set” $\tilde{\mathcal{D}}_n^{(b)}$ of size *n* by sampling *with* replacement from $\mathcal{D}_n$



- Sampling with replacement

$(x^{(1)}, y^{(1)})$ $(x^{(2)}, y^{(2)})$

$(x^{(3)}, y^{(3)})$ $(x^{(4)}, y^{(4)})$

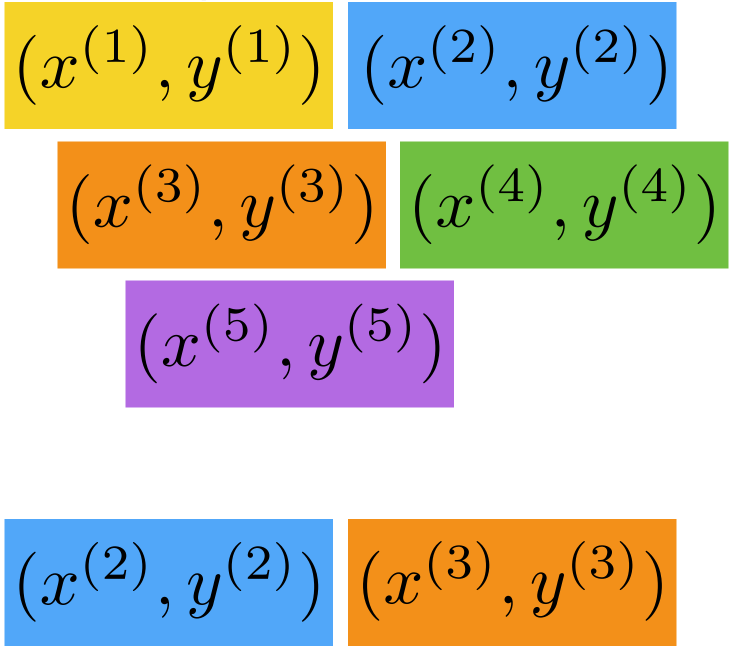
$(x^{(5)}, y^{(5)})$

$(x^{(2)}, y^{(2)})$

# Bagging

- One of multiple ways to make and use an ensemble
- Bagging = **B**ootstrap **agg**regat**ing**
  - Training data $\mathcal{D}_n$
  - For *b* = 1, …, *B*
    - Draw a new "data set" $\tilde{\mathcal{D}}_n^{(b)}$ of size *n* by sampling *with* replacement from $\mathcal{D}_n$

- Sampling with replacement



$(x^{(1)}, y^{(1)})$ $(x^{(2)}, y^{(2)})$

$(x^{(3)}, y^{(3)})$ $(x^{(4)}, y^{(4)})$

$(x^{(5)}, y^{(5)})$

$(x^{(2)}, y^{(2)})$ $(x^{(3)}, y^{(3)})$

# Bagging

- One of multiple ways to make and use an ensemble
- Bagging = **B**ootstrap **agg**regat**ing**
  - Training data $\mathcal{D}_n$
  - For *b* = 1, …, *B*
    - Draw a new “data set” $\tilde{\mathcal{D}}_n^{(b)}$ of size *n* by sampling *with* replacement from $\mathcal{D}_n$

- Sampling with replacement

$(x^{(1)}, y^{(1)})$ $(x^{(2)}, y^{(2)})$

$(x^{(3)}, y^{(3)})$ $(x^{(4)}, y^{(4)})$

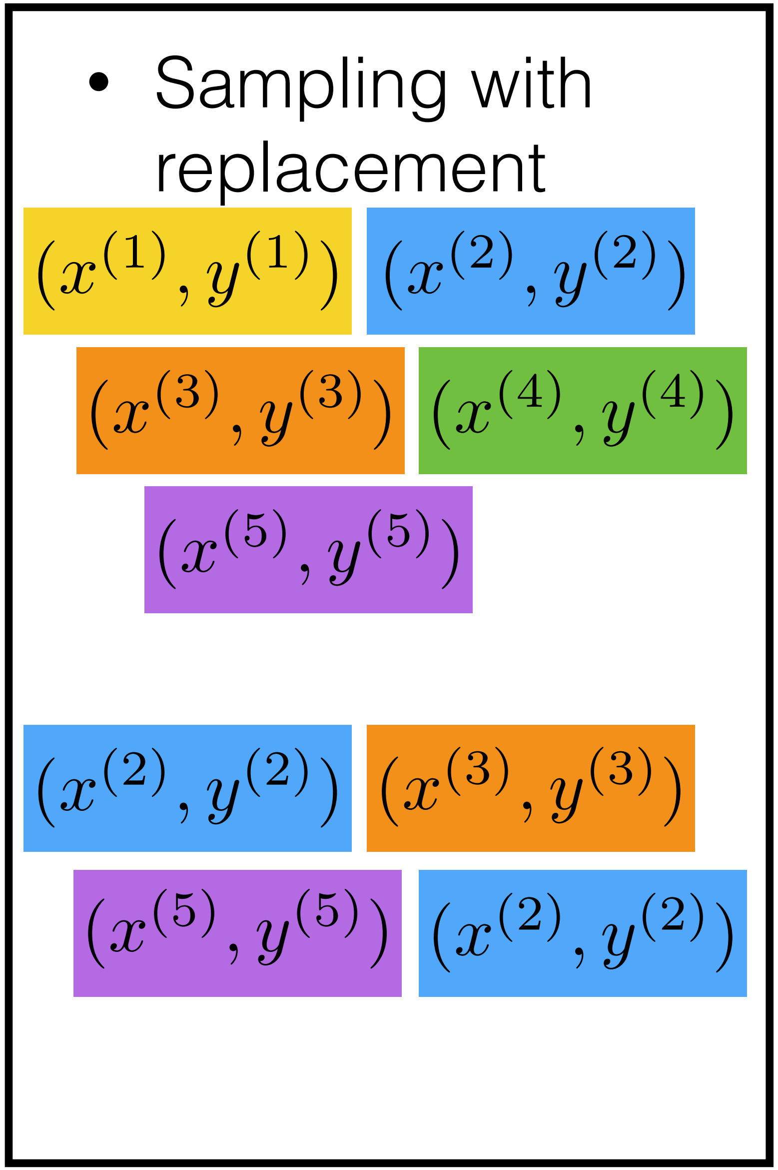
$(x^{(5)}, y^{(5)})$

$(x^{(2)}, y^{(2)})$ $(x^{(3)}, y^{(3)})$

$(x^{(5)}, y^{(5)})$

# Bagging

- One of multiple ways to make and use an ensemble
- Bagging = **B**ootstrap **agg**regat**ing**
  - Training data $\mathcal{D}_n$
  - For *b* = 1, …, *B*
    - Draw a new “data set” $\tilde{\mathcal{D}}_n^{(b)}$ of size *n* by sampling *with* replacement from $\mathcal{D}_n$



- Sampling with replacement

$(x^{(1)}, y^{(1)})$ $(x^{(2)}, y^{(2)})$

$(x^{(3)}, y^{(3)})$ $(x^{(4)}, y^{(4)})$

$(x^{(5)}, y^{(5)})$

$(x^{(2)}, y^{(2)})$ $(x^{(3)}, y^{(3)})$

$(x^{(5)}, y^{(5)})$ $(x^{(2)}, y^{(2)})$

# Bagging

- One of multiple ways to make and use an ensemble
- Bagging = **B**ootstrap **agg**regat**ing**
  - Training data $\mathcal{D}_n$
  - For *b* = 1, …, *B*
    - Draw a new "data set" $\tilde{\mathcal{D}}_n^{(b)}$ of size *n* by sampling *with* replacement from $\mathcal{D}_n$

- Sampling with replacement

$(x^{(1)}, y^{(1)})$ $(x^{(2)}, y^{(2)})$

$(x^{(3)}, y^{(3)})$ $(x^{(4)}, y^{(4)})$

$(x^{(5)}, y^{(5)})$

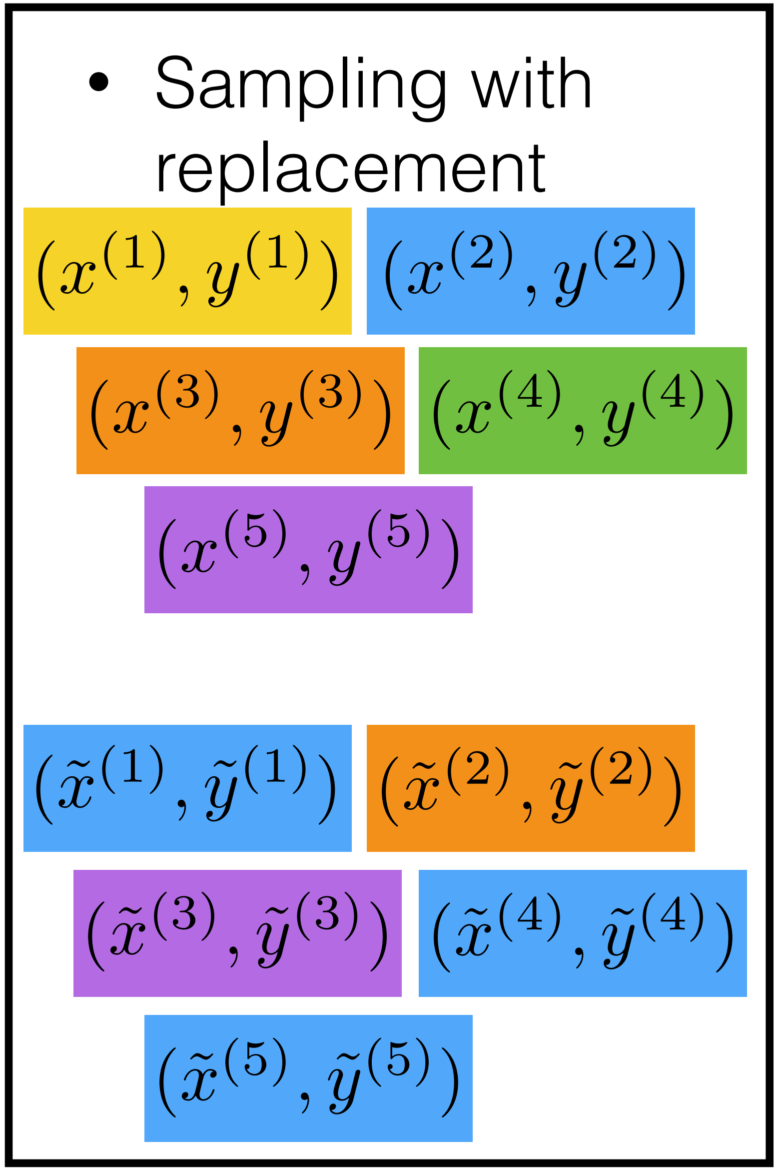
$(x^{(2)}, y^{(2)})$ $(x^{(3)}, y^{(3)})$

$(x^{(5)}, y^{(5)})$ $(x^{(2)}, y^{(2)})$

$(x^{(2)}, y^{(2)})$

# Bagging

- One of multiple ways to make and use an ensemble
- Bagging = **B**ootstrap **agg**regat**ing**
  - Training data $\mathcal{D}_n$
  - For *b* = 1, …, *B*
    - Draw a new “data set” $\tilde{\mathcal{D}}_n^{(b)}$ of size *n* by sampling *with* replacement from $\mathcal{D}_n$



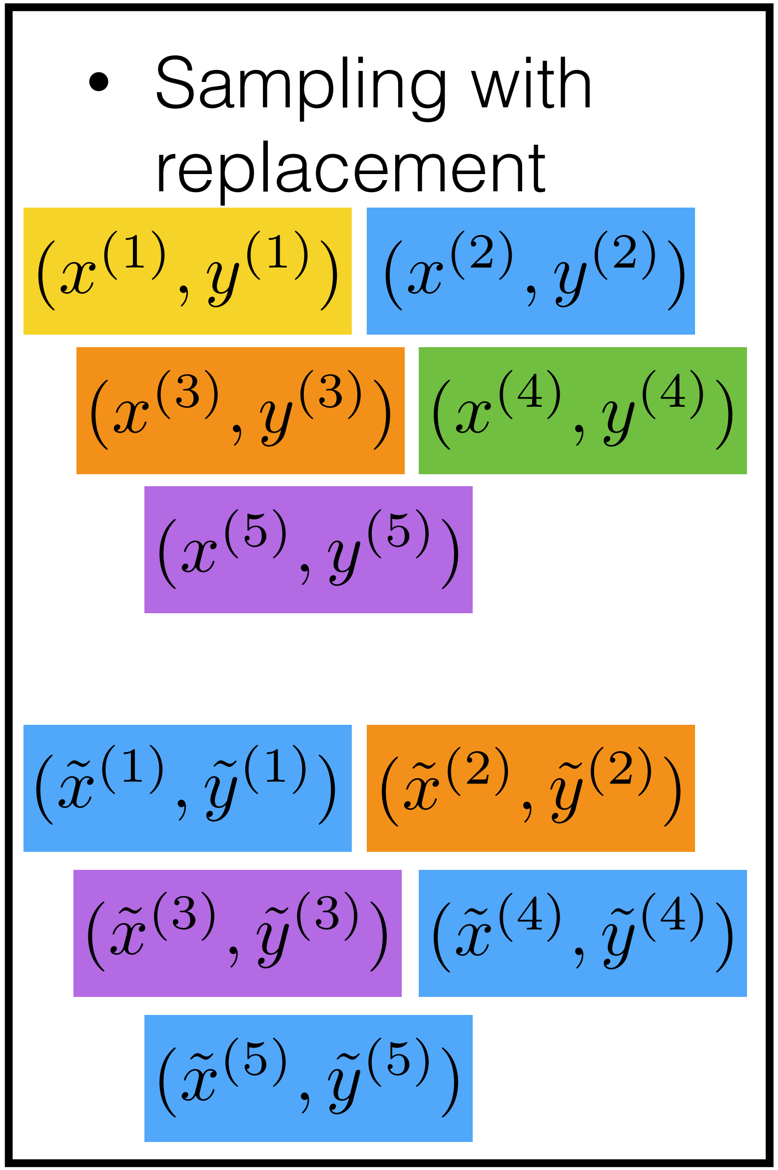
- Sampling with replacement

$(x^{(1)}, y^{(1)})$ $(x^{(2)}, y^{(2)})$ $(x^{(3)}, y^{(3)})$ $(x^{(4)}, y^{(4)})$ $(x^{(5)}, y^{(5)})$

$(\tilde{x}^{(1)}, \tilde{y}^{(1)})$ $(\tilde{x}^{(2)}, \tilde{y}^{(2)})$ $(\tilde{x}^{(3)}, \tilde{y}^{(3)})$ $(\tilde{x}^{(4)}, \tilde{y}^{(4)})$ $(\tilde{x}^{(5)}, \tilde{y}^{(5)})$

# Bagging

- One of multiple ways to make and use an ensemble
- Bagging = **B**ootstrap **agg**regat**ing**
  - Training data $\mathcal{D}_n$
  - For *b* = 1, …, *B*
    - Draw a new “data set” $\tilde{\mathcal{D}}_n^{(b)}$ of size *n* by sampling *with* replacement from $\mathcal{D}_n$
    - Train a predictor $\hat{f}^{(b)}$ on $\tilde{\mathcal{D}}_n^{(b)}$



- Sampling with replacement

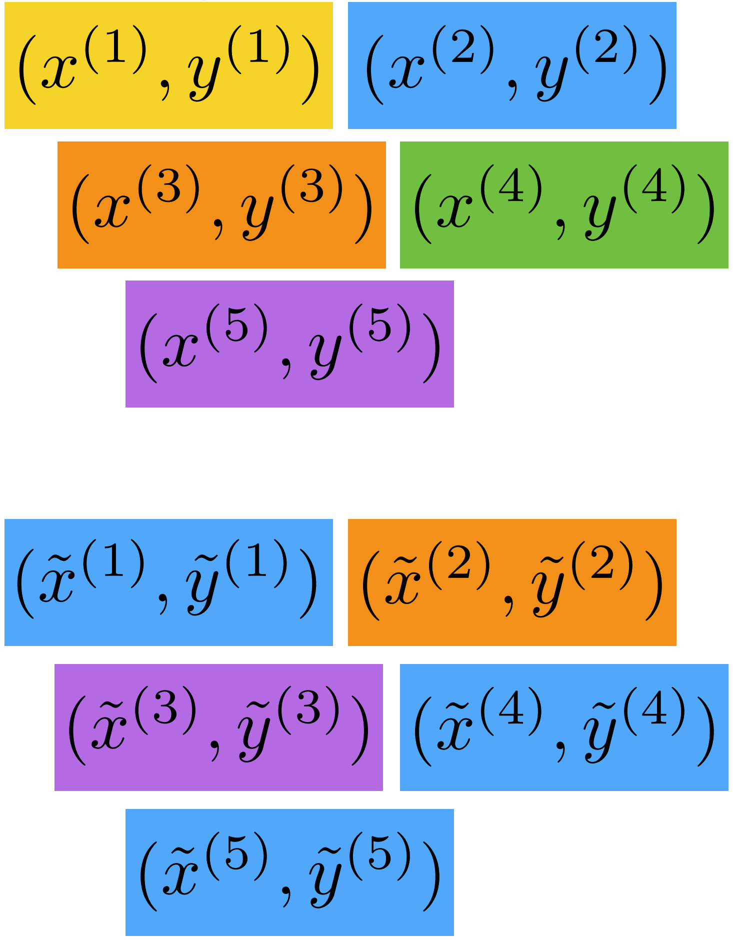
$(x^{(1)}, y^{(1)})$ $(x^{(2)}, y^{(2)})$ $(x^{(3)}, y^{(3)})$ $(x^{(4)}, y^{(4)})$ $(x^{(5)}, y^{(5)})$

$(\tilde{x}^{(1)}, \tilde{y}^{(1)})$ $(\tilde{x}^{(2)}, \tilde{y}^{(2)})$ $(\tilde{x}^{(3)}, \tilde{y}^{(3)})$ $(\tilde{x}^{(4)}, \tilde{y}^{(4)})$ $(\tilde{x}^{(5)}, \tilde{y}^{(5)})$

# Bagging

- One of multiple ways to make and use an ensemble
- Bagging = **B**ootstrap **agg**regat**ing**
  - Training data $\mathcal{D}_n$
  - For *b* = 1, …, *B*
    - Draw a new “data set” $\tilde{\mathcal{D}}_n^{(b)}$ of size *n* by sampling *with* replacement from $\mathcal{D}_n$
    - Train a predictor $\hat{f}^{(b)}$ on $\tilde{\mathcal{D}}_n^{(b)}$
  - Return

- Sampling with replacement



$(x^{(1)}, y^{(1)})$ $(x^{(2)}, y^{(2)})$

$(x^{(3)}, y^{(3)})$ $(x^{(4)}, y^{(4)})$

$(x^{(5)}, y^{(5)})$

$(\tilde{x}^{(1)}, \tilde{y}^{(1)})$ $(\tilde{x}^{(2)}, \tilde{y}^{(2)})$

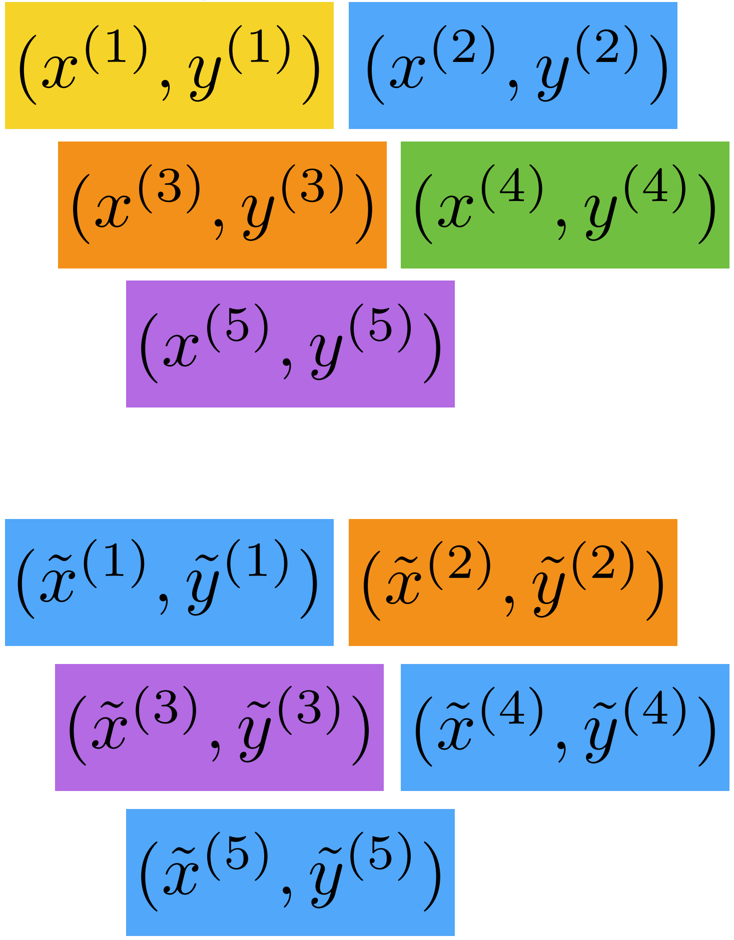
$(\tilde{x}^{(3)}, \tilde{y}^{(3)})$ $(\tilde{x}^{(4)}, \tilde{y}^{(4)})$

$(\tilde{x}^{(5)}, \tilde{y}^{(5)})$

# Bagging

- One of multiple ways to make and use an ensemble
- Bagging = **B**ootstrap **agg**regat**ing**
  - Training data $\mathcal{D}_n$
  - For *b* = 1, …, *B*
    - Draw a new “data set” $\tilde{\mathcal{D}}_n^{(b)}$ of size *n* by sampling *with* replacement from $\mathcal{D}_n$
    - Train a predictor $\hat{f}^{(b)}$ on $\tilde{\mathcal{D}}_n^{(b)}$
  - Return
    - For regression:

- Sampling with replacement



$(x^{(1)}, y^{(1)})$ $(x^{(2)}, y^{(2)})$

$(x^{(3)}, y^{(3)})$ $(x^{(4)}, y^{(4)})$

$(x^{(5)}, y^{(5)})$

$(\tilde{x}^{(1)}, \tilde{y}^{(1)})$ $(\tilde{x}^{(2)}, \tilde{y}^{(2)})$

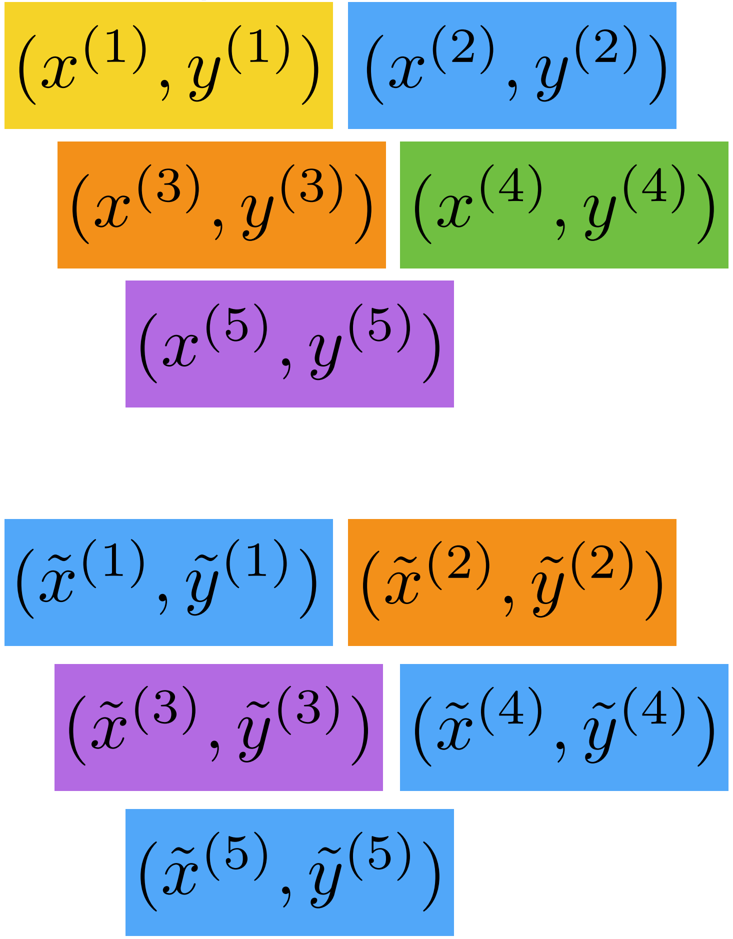
$(\tilde{x}^{(3)}, \tilde{y}^{(3)})$ $(\tilde{x}^{(4)}, \tilde{y}^{(4)})$

$(\tilde{x}^{(5)}, \tilde{y}^{(5)})$

# Bagging

- One of multiple ways to make and use an ensemble
- Bagging = **B**ootstrap **agg**regat**ing**
  - Training data $\mathcal{D}_n$
  - For *b* = 1, …, *B*
    - Draw a new “data set” $\tilde{\mathcal{D}}_n^{(b)}$ of size *n* by sampling *with* replacement from $\mathcal{D}_n$
    - Train a predictor $\hat{f}^{(b)}$ on $\tilde{\mathcal{D}}_n^{(b)}$
  - Return
    - For regression: the predictor

- Sampling with replacement

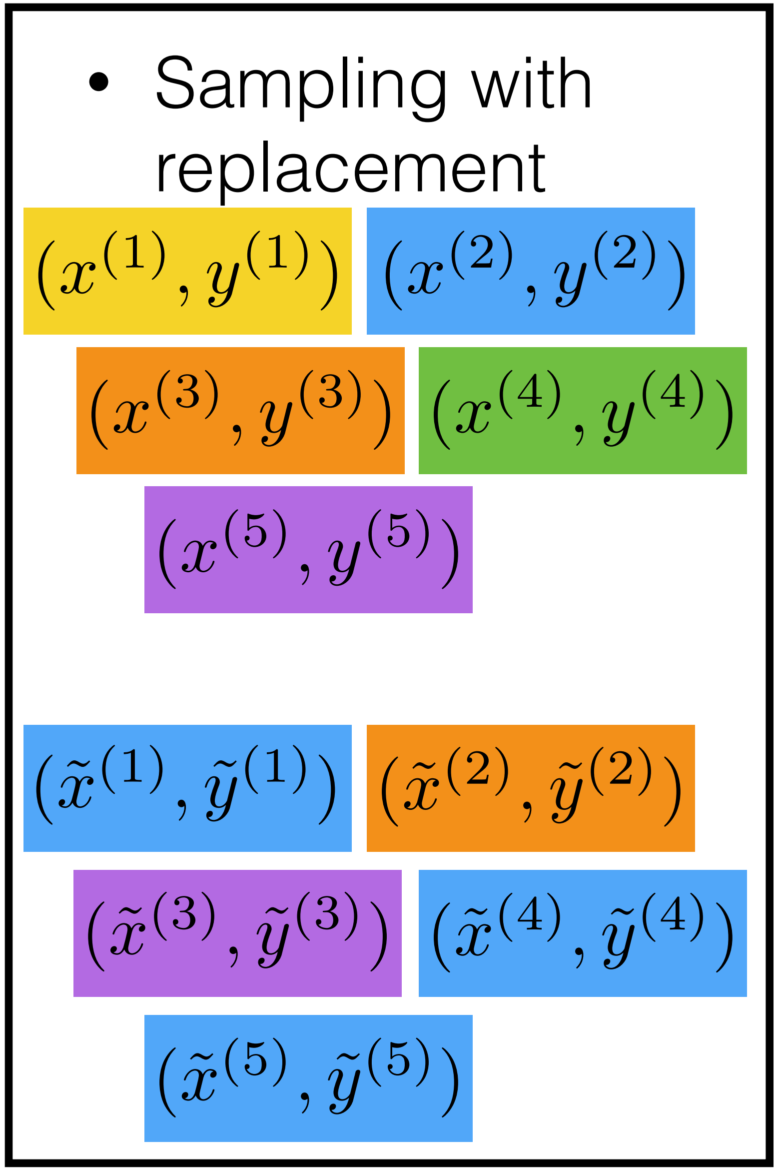


$(x^{(1)}, y^{(1)})$ $(x^{(2)}, y^{(2)})$ $(x^{(3)}, y^{(3)})$ $(x^{(4)}, y^{(4)})$ $(x^{(5)}, y^{(5)})$

$(\tilde{x}^{(1)}, \tilde{y}^{(1)})$ $(\tilde{x}^{(2)}, \tilde{y}^{(2)})$ $(\tilde{x}^{(3)}, \tilde{y}^{(3)})$ $(\tilde{x}^{(4)}, \tilde{y}^{(4)})$ $(\tilde{x}^{(5)}, \tilde{y}^{(5)})$

# Bagging

- One of multiple ways to make and use an ensemble
- Bagging = **B**ootstrap **agg**regat**ing**
  - Training data $\mathcal{D}_n$
  - For *b* = 1, …, *B*
    - Draw a new "data set" $\tilde{\mathcal{D}}_n^{(b)}$ of size *n* by sampling *with* replacement from $\mathcal{D}_n$
    - Train a predictor $\hat{f}^{(b)}$ on $\tilde{\mathcal{D}}_n^{(b)}$
  - Return
    - For regression: the predictor $\hat{f}_{\text{bag}}(x) =$



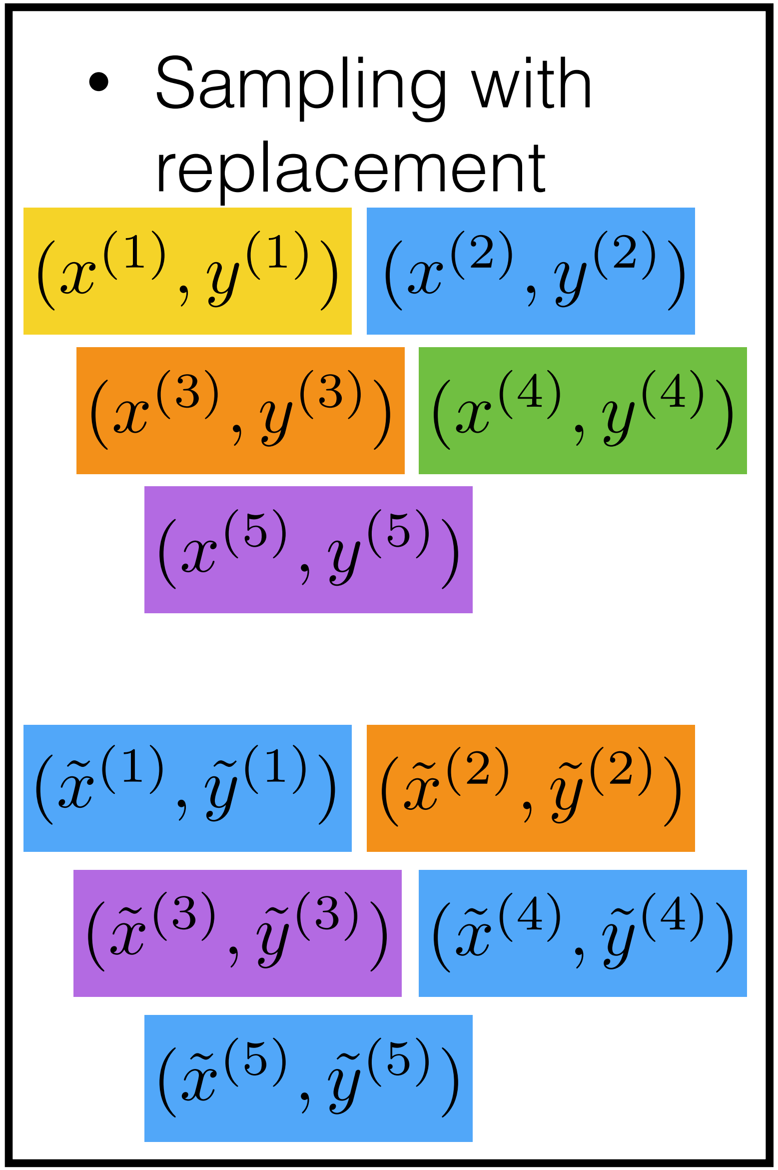
- Sampling with replacement

$(x^{(1)}, y^{(1)})$ $(x^{(2)}, y^{(2)})$ $(x^{(3)}, y^{(3)})$ $(x^{(4)}, y^{(4)})$ $(x^{(5)}, y^{(5)})$

$(\tilde{x}^{(1)}, \tilde{y}^{(1)})$ $(\tilde{x}^{(2)}, \tilde{y}^{(2)})$ $(\tilde{x}^{(3)}, \tilde{y}^{(3)})$ $(\tilde{x}^{(4)}, \tilde{y}^{(4)})$ $(\tilde{x}^{(5)}, \tilde{y}^{(5)})$

# Bagging

- One of multiple ways to make and use an ensemble
- Bagging = **B**ootstrap **agg**regat**ing**
  - Training data $\mathcal{D}_n$
  - For *b* = 1, …, *B*
    - Draw a new “data set” $\tilde{\mathcal{D}}_n^{(b)}$ of size *n* by sampling *with* replacement from $\mathcal{D}_n$
    - Train a predictor $\hat{f}^{(b)}$ on $\tilde{\mathcal{D}}_n^{(b)}$
  - Return
    - For regression: the predictor $\hat{f}_{\text{bag}}(x) = \frac{1}{B}\sum_{b=1}^{B} \hat{f}^{(b)}(x)$



- Sampling with replacement

$(x^{(1)}, y^{(1)})$ $(x^{(2)}, y^{(2)})$

$(x^{(3)}, y^{(3)})$ $(x^{(4)}, y^{(4)})$

$(x^{(5)}, y^{(5)})$

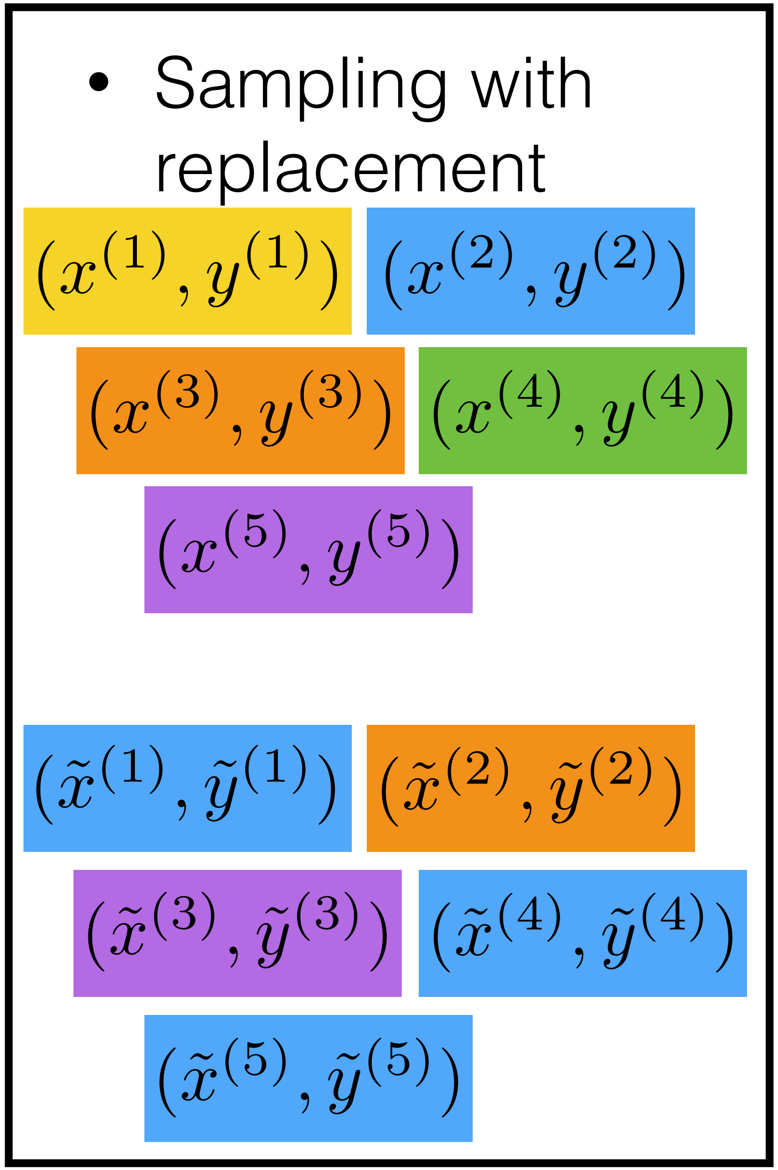
$(\tilde{x}^{(1)}, \tilde{y}^{(1)})$ $(\tilde{x}^{(2)}, \tilde{y}^{(2)})$

$(\tilde{x}^{(3)}, \tilde{y}^{(3)})$ $(\tilde{x}^{(4)}, \tilde{y}^{(4)})$

$(\tilde{x}^{(5)}, \tilde{y}^{(5)})$

# Bagging

- One of multiple ways to make and use an ensemble
- Bagging = **B**ootstrap **agg**regat**ing**
  - Training data $\mathcal{D}_n$
  - For *b* = 1, …, *B*
    - Draw a new “data set” $\tilde{\mathcal{D}}_n^{(b)}$ of size *n* by sampling *with* replacement from $\mathcal{D}_n$
    - Train a predictor $\hat{f}^{(b)}$ on $\tilde{\mathcal{D}}_n^{(b)}$
  - Return
    - For regression: the predictor $\hat{f}_{\text{bag}}(x) = \frac{1}{B}\sum_{b=1}^{B} \hat{f}^{(b)}(x)$
    - Classification: predictor at a point is class with highest vote count at that point



- Sampling with replacement

$(x^{(1)}, y^{(1)})$ $(x^{(2)}, y^{(2)})$

$(x^{(3)}, y^{(3)})$ $(x^{(4)}, y^{(4)})$

$(x^{(5)}, y^{(5)})$

$(\tilde{x}^{(1)}, \tilde{y}^{(1)})$ $(\tilde{x}^{(2)}, \tilde{y}^{(2)})$

$(\tilde{x}^{(3)}, \tilde{y}^{(3)})$ $(\tilde{x}^{(4)}, \tilde{y}^{(4)})$

$(\tilde{x}^{(5)}, \tilde{y}^{(5)})$

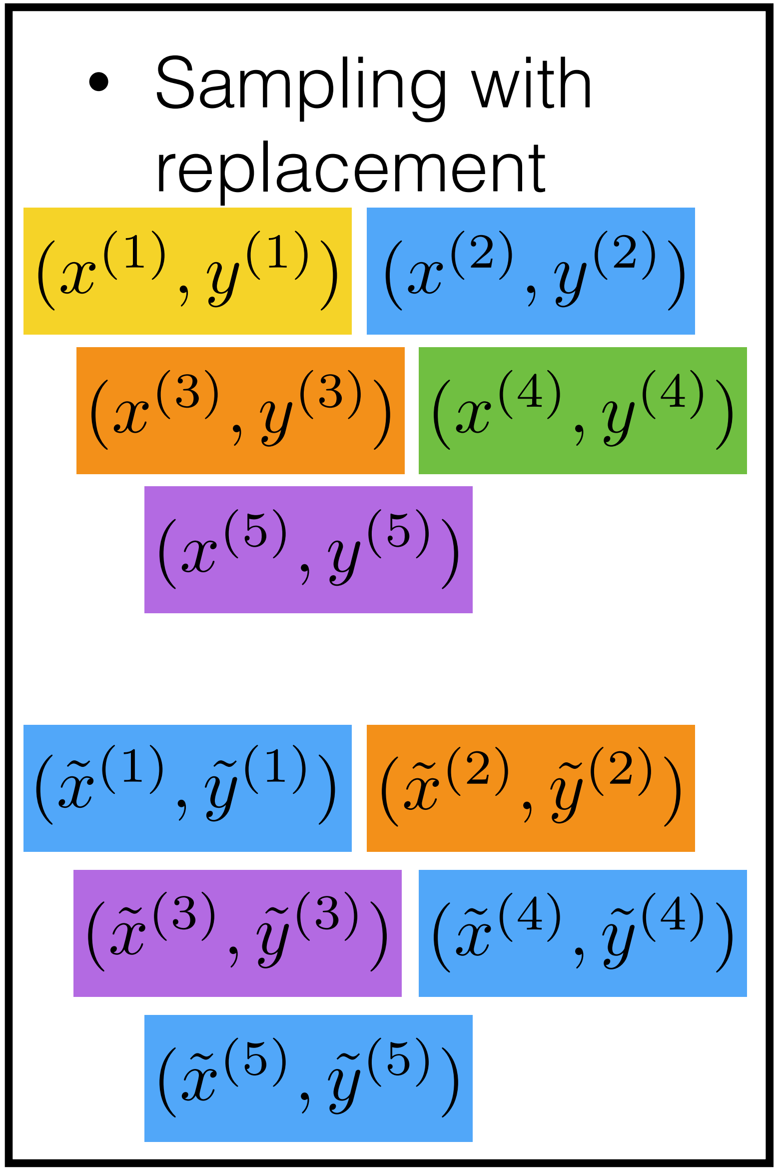
# Bagging

- One of multiple ways to make and use an ensemble
- Bagging = **B**ootstrap **agg**regat**ing**
  - Training data $\mathcal{D}_n$
  - For *b* = 1, …, *B*
    - Draw a new “data set” $\tilde{\mathcal{D}}_n^{(b)}$ of size *n* by sampling *with* replacement from $\mathcal{D}_n$
    - Train a predictor $\hat{f}^{(b)}$ on $\tilde{\mathcal{D}}_n^{(b)}$

bootstrap

aggregating

  - For regression: the predictor $\hat{f}_{\text{bag}}(x) = \frac{1}{B}\sum_{b=1}^{B} \hat{f}^{(b)}(x)$
  - Classification: predictor at a point is class with highest vote count at that point



- Sampling with replacement

$(x^{(1)}, y^{(1)})$ $(x^{(2)}, y^{(2)})$ $(x^{(3)}, y^{(3)})$ $(x^{(4)}, y^{(4)})$ $(x^{(5)}, y^{(5)})$

$(\tilde{x}^{(1)}, \tilde{y}^{(1)})$ $(\tilde{x}^{(2)}, \tilde{y}^{(2)})$ $(\tilde{x}^{(3)}, \tilde{y}^{(3)})$ $(\tilde{x}^{(4)}, \tilde{y}^{(4)})$ $(\tilde{x}^{(5)}, \tilde{y}^{(5)})$

# Random forests

# Random forests

- Bagging + decision trees + extra randomness

# Random forests

- Bagging + decision trees + extra randomness
- Random forest

# Random forests

- Bagging + decision trees + extra randomness
- Random forest
    - For $b = 1, \ldots, B$

# Random forests

- Bagging + decision trees + extra randomness
- Random forest
  - For *b* = 1, …, *B*
    - Draw a new “data set” $\tilde{\mathcal{D}}_n^{(b)}$ of size *n* by sampling *with* replacement from $\mathcal{D}_n$

# Random forests

- Bagging + decision trees + extra randomness
- Random forest
  - For *b* = 1, …, *B*
    - Draw a new “data set” $\tilde{\mathcal{D}}_n^{(b)}$ of size *n* by sampling *with* replacement from $\mathcal{D}_n$
    - Build a tree on $\tilde{\mathcal{D}}_n^{(b)}$ by recursively repeating the following until minimum node size *k* is reached:

# Random forests

- Bagging + decision trees + extra randomness
- Random forest
  - For *b* = 1, …, *B*
    - Draw a new “data set” $\tilde{\mathcal{D}}_n^{(b)}$ of size *n* by sampling *with* replacement from $\mathcal{D}_n$
    - Build a tree on $\tilde{\mathcal{D}}_n^{(b)}$ by recursively repeating the following until minimum node size *k* is reached:
      - Select *m* features uniformly at random, *without* replacement, from the *d* features

# Random forests

- Bagging + decision trees + extra randomness
- Random forest
  - For $b = 1, \ldots, B$
    - Draw a new "data set" $\tilde{\mathcal{D}}_n^{(b)}$ of size *n* by sampling *with* replacement from $\mathcal{D}_n$
    - Build a tree on $\tilde{\mathcal{D}}_n^{(b)}$ by recursively repeating the following until minimum node size *k* is reached:
      - Select *m* features uniformly at random, *without* replacement, from the *d* features
      - Pick the best split dimension and split value among the *m* features

# Random forests

- Bagging + decision trees + extra randomness
- Random forest
  - For *b* = 1, …, *B*
    - Draw a new "data set" $\tilde{\mathcal{D}}_n^{(b)}$ of size *n* by sampling *with* replacement from $\mathcal{D}_n$
    - Build a tree on $\tilde{\mathcal{D}}_n^{(b)}$ by recursively repeating the following until minimum node size *k* is reached:
      - Select *m* features uniformly at random, *without* replacement, from the *d* features
      - Pick the best split dimension and split value among the *m* features
      - Build two children

# Random forests

- Bagging + decision trees + extra randomness
- Random forest
  - For *b* = 1, …, *B*
    - Draw a new “data set” $\tilde{\mathcal{D}}_n^{(b)}$ of size *n* by sampling *with* replacement from $\mathcal{D}_n$
    - Build a tree on $\tilde{\mathcal{D}}_n^{(b)}$ by recursively repeating the following until minimum node size *k* is reached:
      - Select *m* features uniformly at random, *without* replacement, from the *d* features
      - Pick the best split dimension and split value among the *m* features
      - Build two children
  - Return: average for regression; vote for classification

# Random forests

- Bagging + decision trees + extra randomness
- Random forest
    - For *b* = 1, …, *B*
        - Draw a new “data set” $\tilde{\mathcal{D}}_n^{(b)}$ of size *n* by sampling *with* replacement from $\mathcal{D}_n$
        - Build a tree on $\tilde{\mathcal{D}}_n^{(b)}$ by recursively repeating the following until minimum node size *k* is reached:
            - Select *m* features uniformly at random, *without* replacement, from the *d* features
            - Pick the best split dimension and split value among the *m* features
            - Build two children
    - Return: average for regression; vote for classification

# Random forests

- Bagging + decision trees + extra randomness
- Random forest
  - For $b$ = 1, …, $B$
    - Draw a new “data set” $\tilde{\mathcal{D}}_n^{(b)}$ of size $n$ by sampling *with* replacement from $\mathcal{D}_n$
    - Build a tree on $\tilde{\mathcal{D}}_n^{(b)}$ by recursively repeating the following until minimum node size $k$ is reached:
      - Select $m$ features uniformly at random, *without* replacement, from the $d$ features
      - Pick the best split dimension and split value among the $m$ features
      - Build two children
  - Return: average for regression; vote for classification

# Random forests

- Bagging + decision trees + extra randomness
- Random forest
  - For *b* = 1, …, *B*
    - Draw a new "data set" $\tilde{\mathcal{D}}_n^{(b)}$ of size *n* by sampling *with* replacement from $\mathcal{D}_n$
    - Build a tree on $\tilde{\mathcal{D}}_n^{(b)}$ by recursively repeating the following until minimum node size *k* is reached:
      - Select *m* features uniformly at random, *without* replacement, from the *d* features
      - Pick the best split dimension and split value among the *m* features
      - Build two children
  - Return: average for regression; vote for classification

bagging

bagging

# Random forests

- Bagging + decision trees + extra randomness
- Random forest
  - For $b = 1, \ldots, B$

    bagging

    - Draw a new "data set" $\tilde{\mathcal{D}}_n^{(b)}$ of size $n$ by sampling *with* replacement from $\mathcal{D}_n$

    trees

    - Build a tree on $\tilde{\mathcal{D}}_n^{(b)}$ by recursively repeating the following until minimum node size $k$ is reached:
      - Select $m$ features uniformly at random, *without* replacement, from the $d$ features
      - Pick the best split dimension and split value among the $m$ features
      - Build two children

  bagging

  - Return: average for regression; vote for classification

# Random forests

- Bagging + decision trees + extra randomness
- Random forest
  - For $b = 1, \ldots, B$
    - Draw a new "data set" $\tilde{\mathcal{D}}_n^{(b)}$ of size *n* by sampling *with* replacement from $\mathcal{D}_n$
    - Build a tree on $\tilde{\mathcal{D}}_n^{(b)}$ by recursively repeating the following until minimum node size *k* is reached:
      - Select *m* features uniformly at random, *without* replacement, from the *d* features
      - Pick the best split dimension and among the *m* features
      - Build two children
  - Return: average for regression; vote for classification

bagging

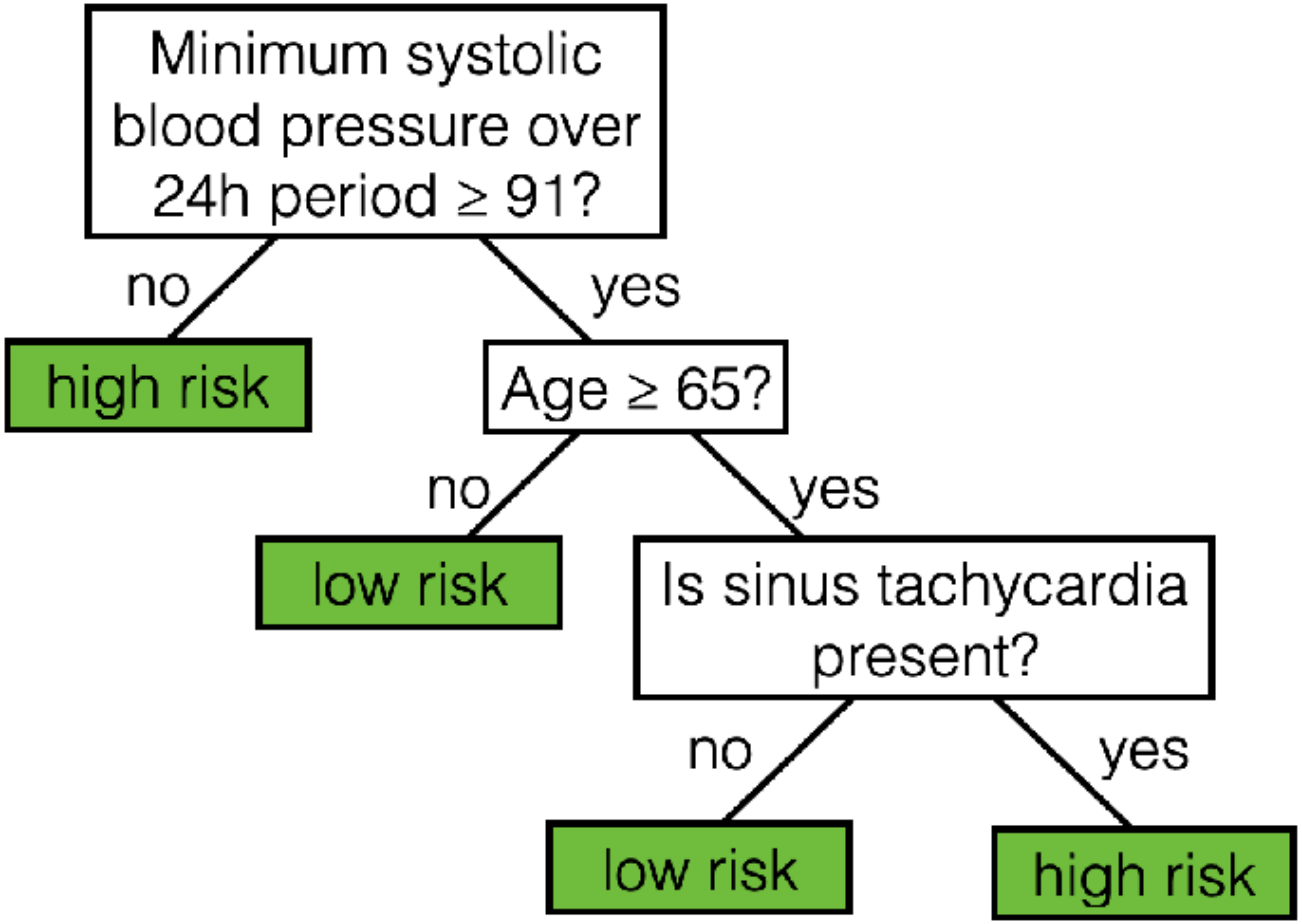
trees

extra randomness

bagging

# Decision trees & random forests

# Decision trees & random forests



Minimum systolic blood pressure over 24h period ≥ 91?

no

high risk

yes

Age ≥ 65?

no

low risk

yes

Is sinus tachycardia present?

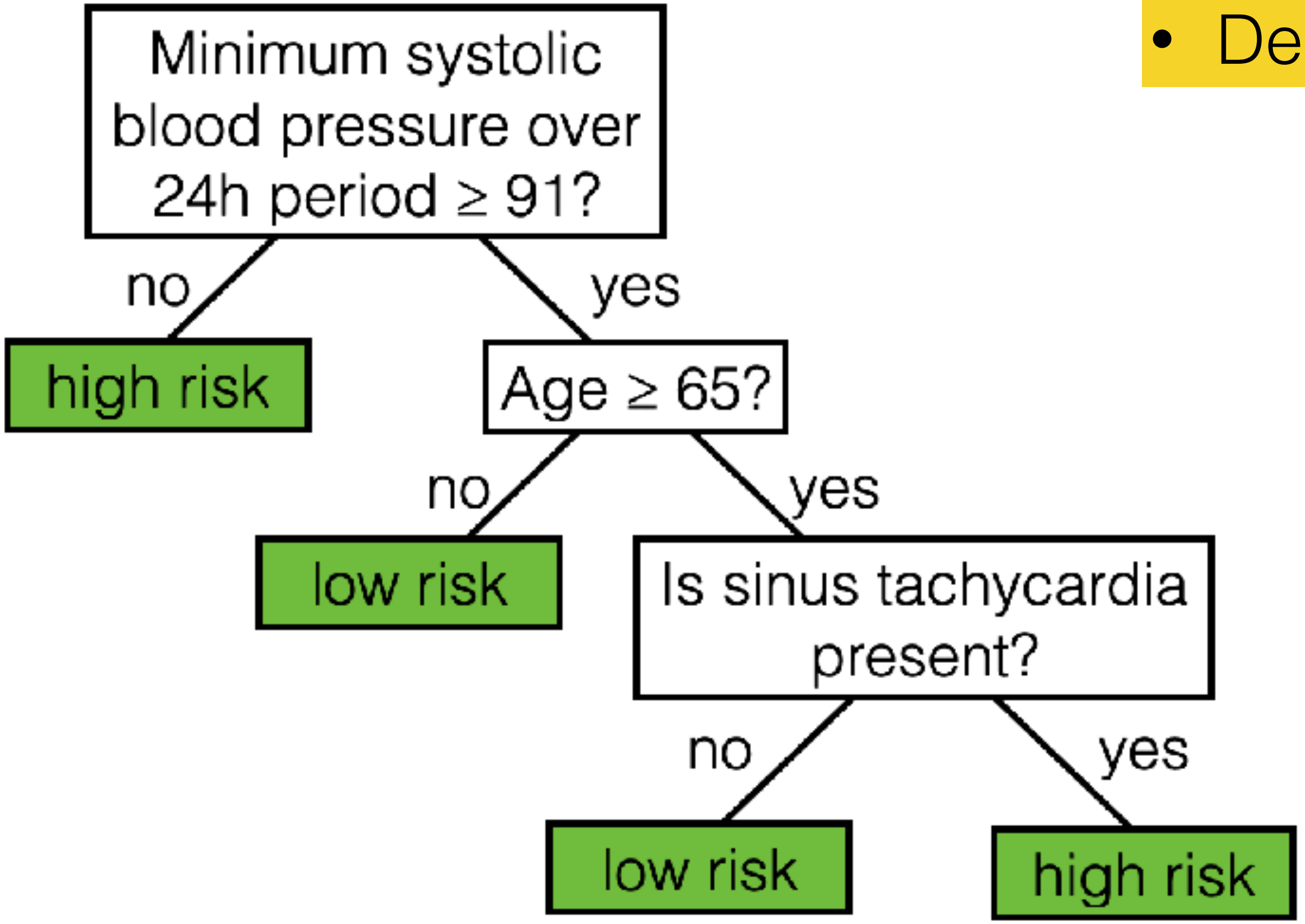
no

low risk

yes

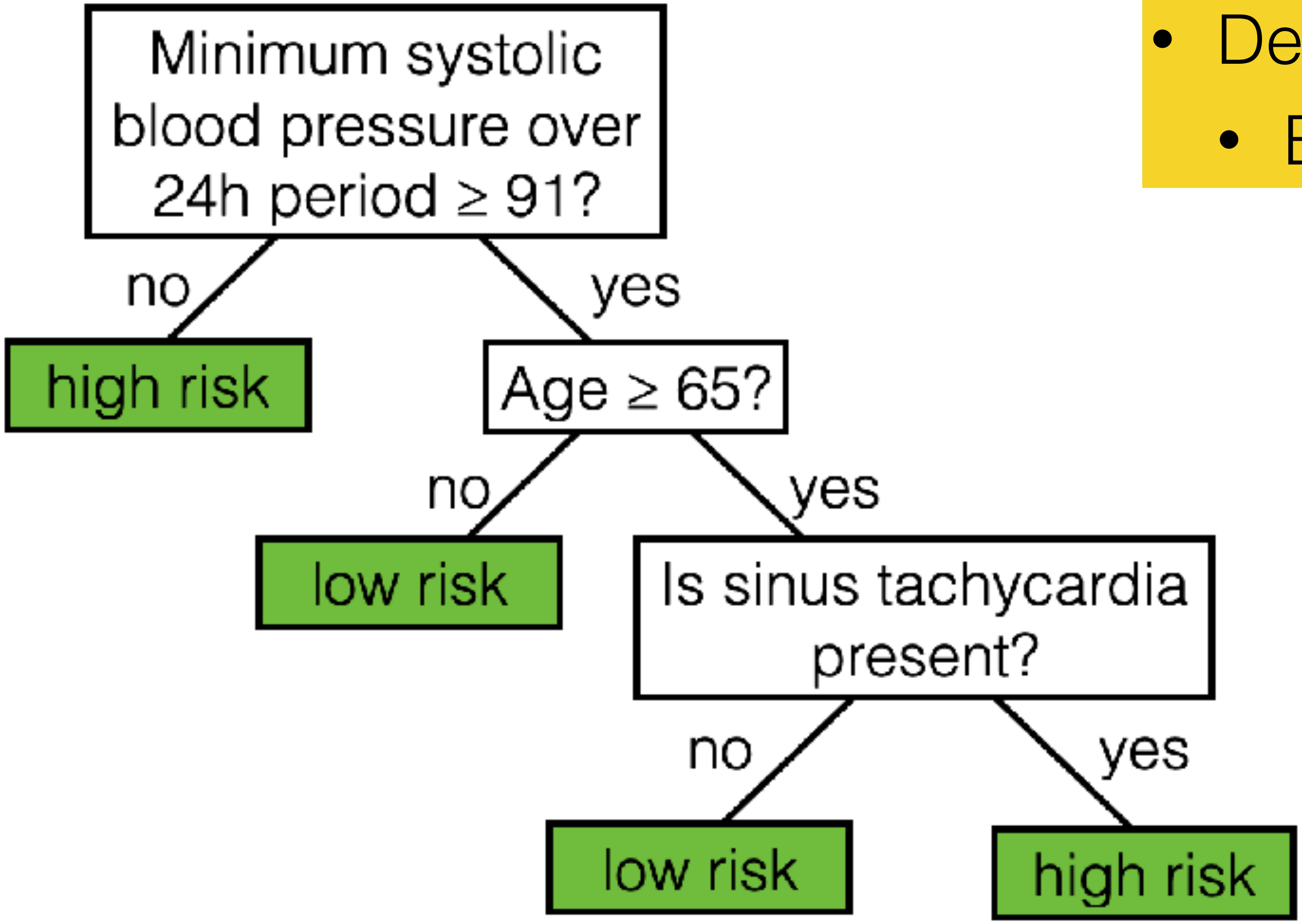
high risk

# Decision trees & random forests



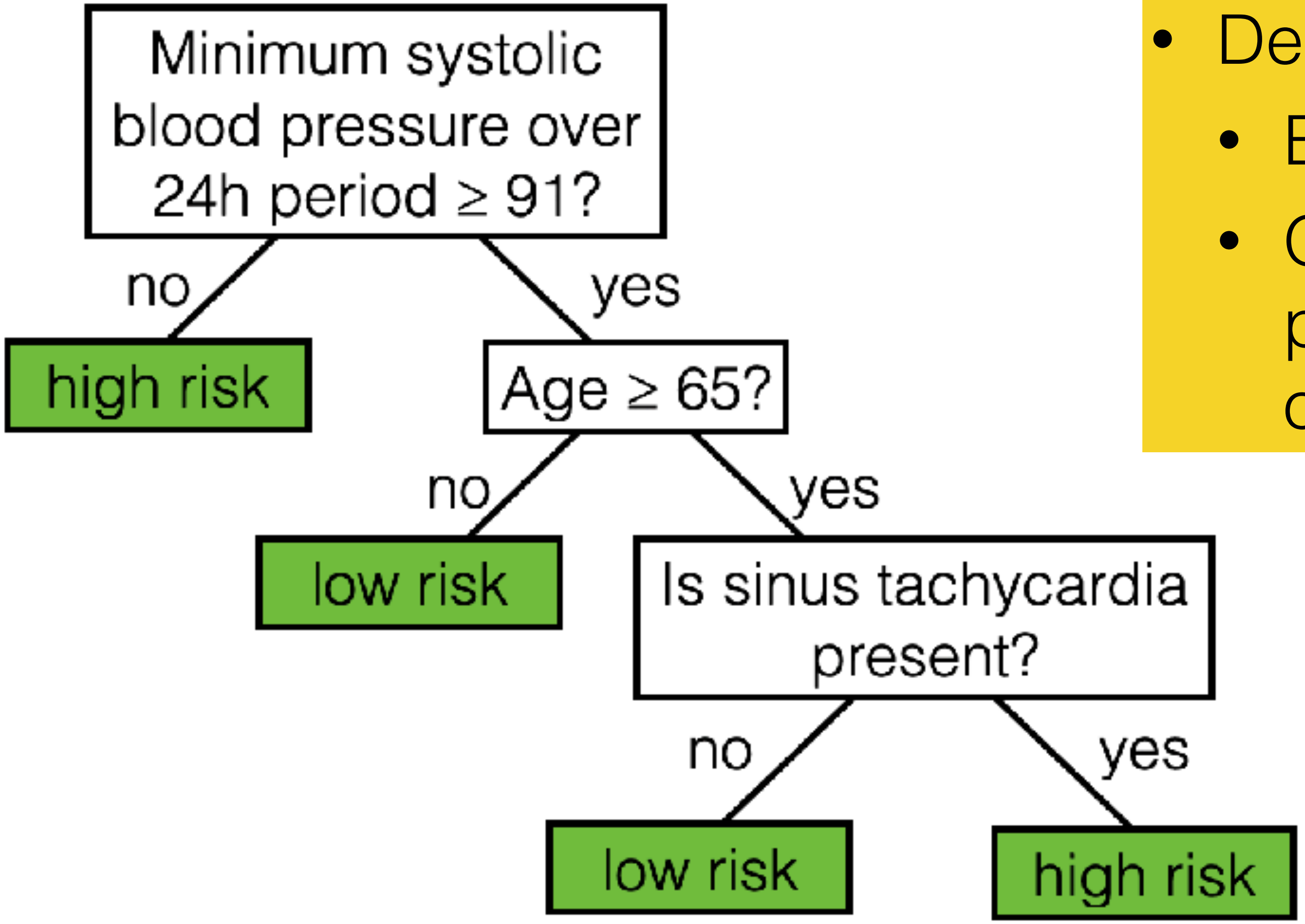
- Decision trees

# Decision trees & random forests



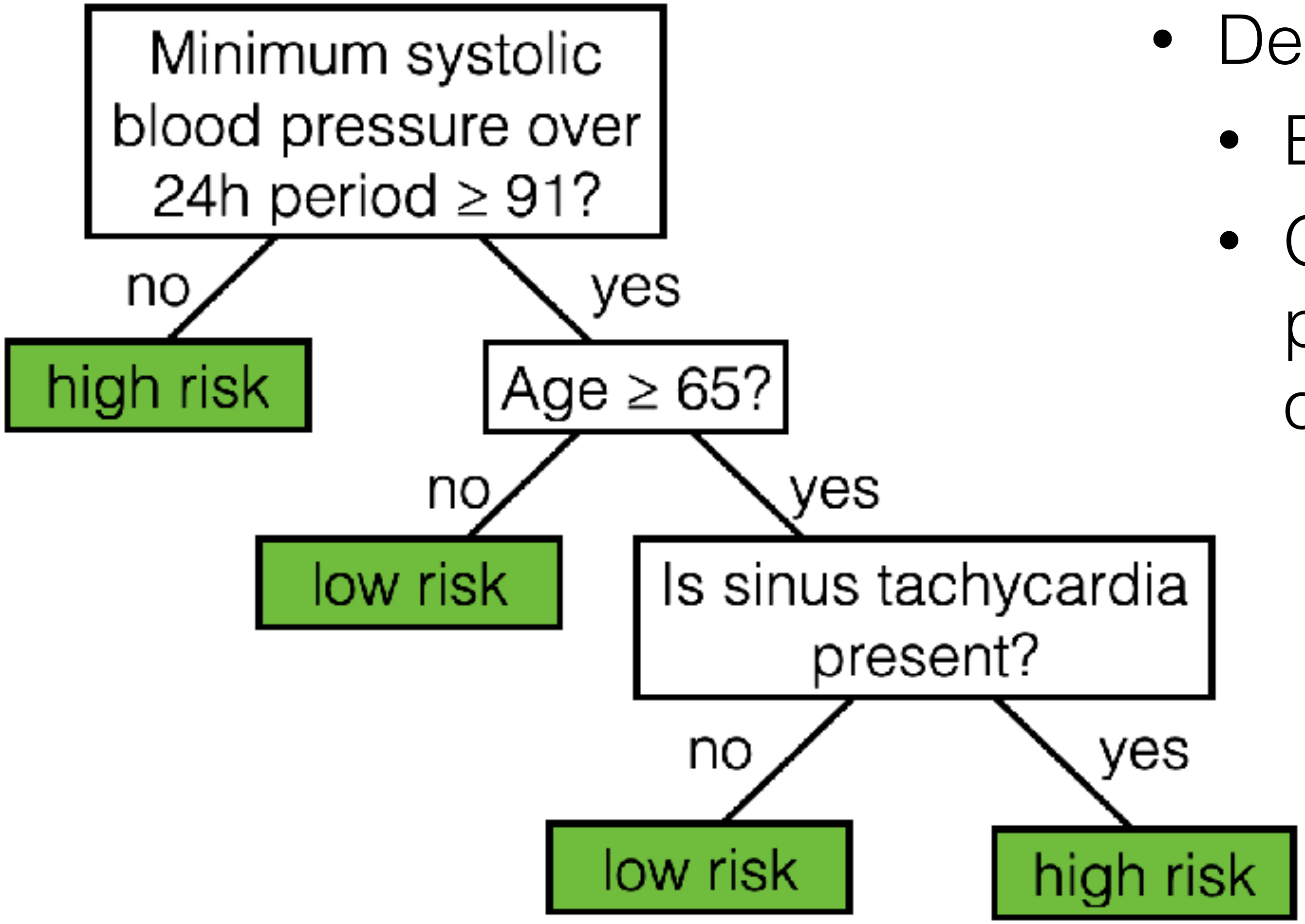
- Decision trees
    - Easy to interpret

# Decision trees & random forests



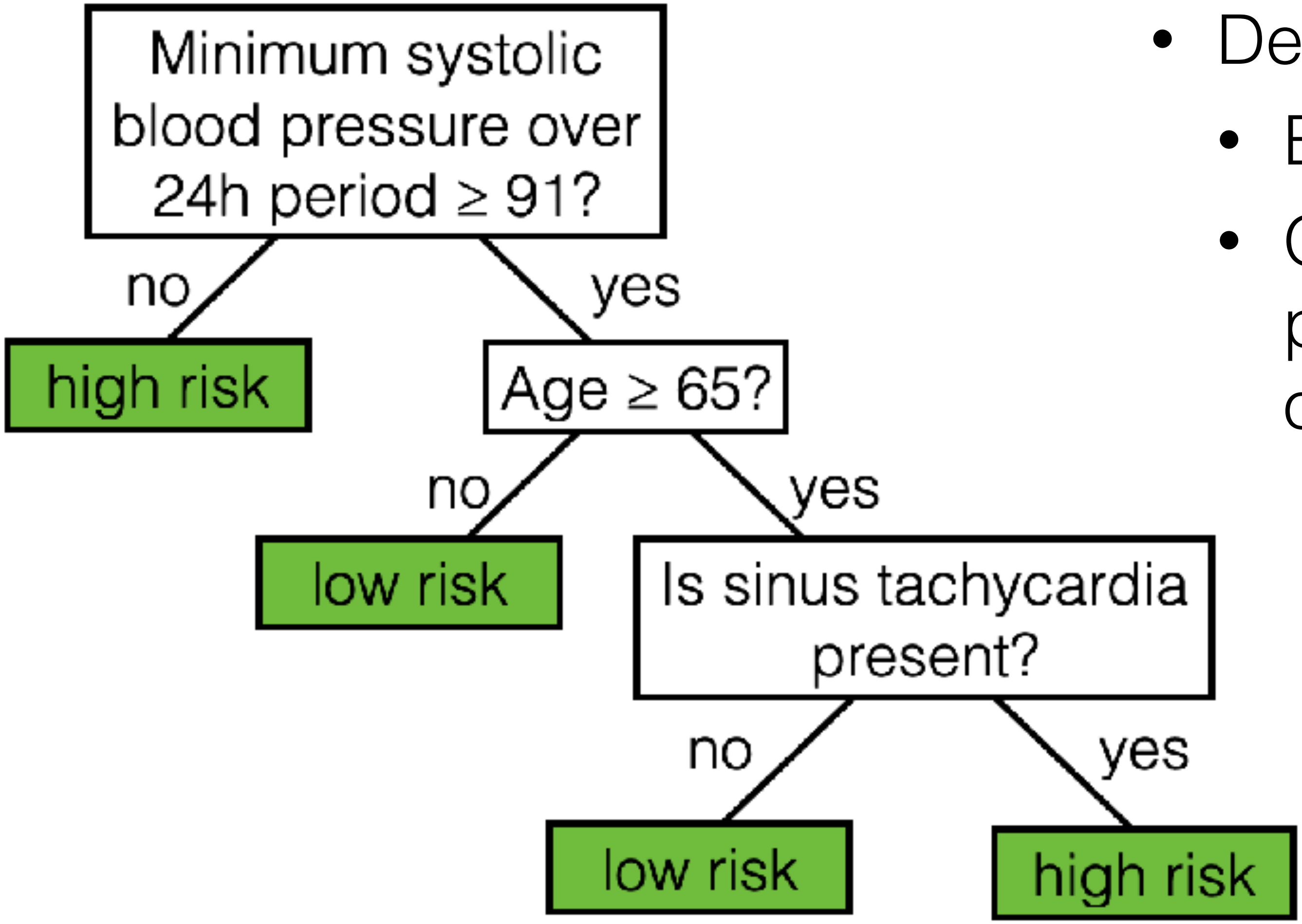
- Decision trees
  - Easy to interpret
  - Often not the best predictions (error on new data)

# Decision trees & random forests



- Decision trees
  - Easy to interpret
  - Often not the best predictions (error on new data)
- Random forests/ensembling

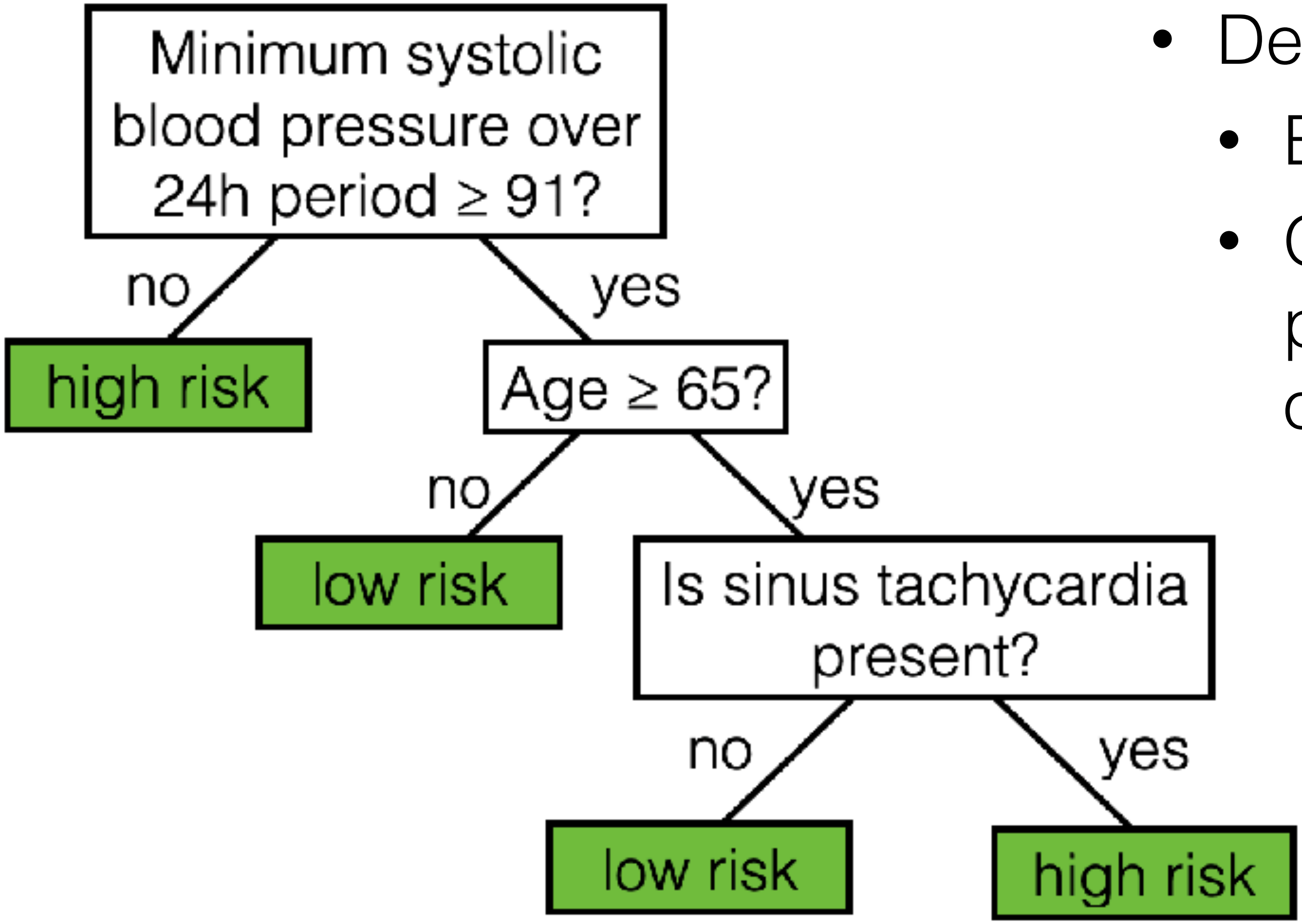
# Decision trees & random forests



- Decision trees
  - Easy to interpret
  - Often not the best predictions (error on new data)

- Random forests/ensembling
  - Harder to interpret

# Decision trees & random forests



Minimum systolic blood pressure over 24h period ≥ 91?

no

high risk

yes

Age ≥ 65?

no

low risk

yes

Is sinus tachycardia present?

no

low risk

yes

high risk

- Decision trees
  - Easy to interpret
  - Often not the best predictions (error on new data)

- Random forests/ensembling
  - Harder to interpret
  - Often much better predictions

Hvala na pažnji.

Molim vas pitajte sve šta vas interesuje.