# Decision Trees

Blackbox Machine learning

**Q** Estimate in-game win probability for American Football

↳ as a function of game-state

* Why?
  - Betting
  - Player valuation
  - strategic decision making

  make the decision which maximizes our win probability.

$\mathbb{P}(Td) = 0.2$
$\mathbb{P}(fg) = 0.2$
$\mathbb{P}(\text{turnover}) = 0.1$
$\mathbb{P}(\text{punt}) = 0.5$

* Mathematical Models:
  - state-space models & dynamic Programming
    - States: {# possessions remaining, score of team A, score of team B}
    - transition probabilities → $\mathbb{P}(\text{next state is } S_1 \mid \text{current state is } S_0)$

$\mathbb{P}(2,7,0 \mid 3,0,0) = \mathbb{P}(td) = 0.2$

* Statistical models:
  - learn entirely from historical data
  - across "similar" situations over the entire (recent) history of football, what proportion of times did the team with possession win the game?
  - play by play NFL data is easily accessible today (since 2017) NFLFastR
  - easy to fit your favorite ML model today

– tho today everyone uses statistical models for WP

* We want to fit a ML WP model from historical data.

as a function of game-state

## Variables

outcome variable – whether the team with possession won or lost the game 1/0

game state vars:

- yardline
- down
- distance
- score differential
- game seconds remaining
- off. and def. team quality → point spread
- Receive 2nd half kickoff
- Timeouts

$\beta_{11} \cdot ydl + \beta_{12} \cdot ydl^2 + \beta_{13} \cdot ydl^3 + \ldots$

categorical

$\beta_{21}$ down1 +
$\beta_{22}$ down2 +
$\beta_{23}$ down3 +
$\beta_{24}$ down4

* Last week: Logistic Regression

$$\mathbb{P}(\text{Win} = 1 \mid \underset{X}{\text{game-state}}) = \frac{1}{1 + \exp\left(-\left(\beta_1 \cdot ydl + \beta_2 \,\text{down} + \beta_3 \cdot \text{dist} + \beta_4 \cdot \text{time rem.} + \beta_5 \cdot \text{game sec rem} + \ldots\right)\right)}$$

What's wrong with this?

* These variables are all <u>interacting</u> but regression models are fundamentally additive (non-interacting)

<u>ex</u> scorediff and time Remaining are def. interacting

$$\beta_1 \cdot \text{score diff} + \beta_2 \cdot \text{time rem} + \beta_3 \cdot \underbrace{\left(\text{score diff}\right) \cdot \left(\text{time Rem}\right)}_{\beta_3 \cdot \text{sign}[\text{score diff}] \cdot e^{-(\text{score diff}) \cdot (\text{time Rem})}}$$

* You can model interactions in additive regression settings but this is extremely difficult with 2 numeric/continuous vars and it is easy with indicator / binary vars

linear: $\beta \cdot X$

<u>Shane</u> $\beta_1 \cdot D_1 + \beta_2 \cdot D_2 + \beta_3 \cdot D_1 \cdot D_2$

$D_1$ = indicator

$$P(\text{out}) = \frac{1}{1 + \exp(-( \qquad ))}$$

All of these variables are interacting and nonlinear.

Logistic regression is a ~~terrible~~ bad idea.

$\longrightarrow$ Machine Learning!

Learn an arbitrarily complex relationship between variables, given enough data.

* Focus on using WP for strategic decision making.

| |
|---|
| Q What fourth down decision should I make in {Go, FG, Punt} given the game state? |

* The entire 4th down decision process can be defined in terms of the win probability if you have a 1st down and 10 yards to go

def $V_1(x)$ = value (win probability) of a $1^{st}$ and 10 at game-state $x$

* value of kicking a FG on $4^{th}$ down:

$$V_{FG}(x) = \underbrace{\mathbb{P}\begin{pmatrix}\text{make}\\ \text{FG}\end{pmatrix}}_{\text{logistic regression}} \cdot \underbrace{V\begin{pmatrix}\text{make}\\ \text{FG}\end{pmatrix}}_{1 - V_1\begin{pmatrix}x \\ \text{except} \\ \text{75 yardline} \\ \text{scorediff} + 3\end{pmatrix}} + \underbrace{\mathbb{P}\begin{pmatrix}\text{miss}\\ \text{FG}\end{pmatrix}}_{1 - \mathbb{P}\begin{pmatrix}\text{make}\\ \text{FG}\end{pmatrix}} \cdot \underbrace{V\begin{pmatrix}\text{miss}\\ \text{FG}\end{pmatrix}}_{1 - V_1\begin{pmatrix}x \\ \text{except} \\ \text{flip} \\ \text{ydl}\end{pmatrix}}$$

logistic Regression
outcome 1/0 made/miss FG
Yardline, Kicker quality,
weather, time rem.
HW

$$\mathbb{P}\left(\begin{matrix}\text{make}\\ \text{FG}\end{matrix} \,\middle|\, \text{ydl}, \text{Kq}\right)$$

$$\text{Kq} = \text{Kicks made over expected}$$

$$= \sum_{\substack{\text{his prev}\\ \text{kicks } i}} \left[ \begin{cases} 1 & \text{if kick } i \text{ made} \\ 0 & \text{if not} \end{cases} - \underbrace{\mathbb{P}\left(\begin{matrix}\text{kick}\\ i\\ \text{made}\end{matrix} \,\middle|\, \text{ydl}\right)}_{\mathbb{P}\left(\begin{matrix}\text{make}\\ \text{FG}\end{matrix} \,\middle|\, \text{ydl}\right)} \right]$$

$$\text{punter qual} = \text{punt yardlines added over expected}$$

$$= \sum_{\substack{\text{his prev}\\ \text{punts } i}} \left[ \left(\begin{matrix}\text{ydl } y_i'\\ \text{after}\\ \text{punt } i\end{matrix}\right) - \underbrace{\mathbb{E}\left(\begin{matrix}\text{ydl } y_i'\\ \text{after}\\ \text{punt } i\end{matrix} \,\middle|\, \text{ydl}\right)}_{\text{ignore punter quality}} \right]$$

* Value of Punting on $4^{th}$ down:

$$V_{punt}(x) = \sum_{y'} P\begin{pmatrix} \text{ydl after} \\ \text{punting} \\ \text{is } y' \end{pmatrix} \cdot \left(1 - V_1\begin{pmatrix} x \\ \text{except} \\ \text{ydl} & y' \end{pmatrix}\right)$$

$$= 1 - \mathbb{E}_{\substack{punt \\ y'}} V_1\begin{pmatrix} x \\ \text{ydl}, y' \end{pmatrix}$$

$$\approx 1 - V_1\begin{pmatrix} x \\ \text{ydl}, \underbrace{\mathbb{E}_{punt}(y')}_{\text{expected next yardline after punting}} \end{pmatrix}$$

- bin by ydl and take avg next ydl
- linear regression on ydl + punter quality → HW

$\mathbb{E} f(Y) \neq f \mathbb{E} Y$ in general

but in this specific case $=$ is close enough because $f$, $y \mapsto V_1(y)$ is nearly <u>Linear</u>

and so $\mathbb{E} f(Y) = f \mathbb{E} Y$

* Value of going for it on $4^{th}$ down:

$4^{th}$ down and $z$ yards to go
at yardline $y$
game-state $x$

$$V_{go}(x) \approx \underbrace{\mathbb{P}(\text{convert})}_{} \cdot V_1\begin{pmatrix} x \\ \text{except} \\ \text{new ydl} \end{pmatrix} + \left(1 - \mathbb{P}(\text{convert})\right) \cdot \left(1 - V_1\begin{pmatrix} x \\ \text{except} \\ \text{opp. ydl} \end{pmatrix}\right)$$

new ydl $\rightarrow$ $y - z$

opp. ydl $\rightarrow$ $100 - y$

ydl. $y$ = $y$ yards from td

$\mathbb{P}(\text{convert}) \rightarrow \mathbb{P}(\text{gain} \geq z \text{ yards})$

off. def. team quality
dist
yardline
outcome var: 1/0, 1 = conversion

$\longrightarrow$ Logistic Regression

HW

* $V_{go}$, $V_{fg}$, $V_{punt}$ on $4^{th}$ down as a function of game-state $x$ can be written in terms of

$$\underbrace{V_1(x)}_{\text{Machine Learning}} = \text{WP if a team has a } 1^{st} \text{ and } 10 \text{ at } x$$

and $\underbrace{\mathbb{P}(\text{make fg} \mid x),\ \mathbb{P}(\text{convert} \mid x),\ \mathbb{E}_{punt}[y' \mid x]}_{\text{Regression (HW)}}$

**Task** Estimate $V_1(x) = WP[x \mid 1^{st} \text{ and } 10)$ using Machine Learning.

**Game-State x** yardline, score diff, timeouts, game sec. rem., point spread, receive $2^{nd}$ half kickoff

## Dataset

$i$ = index of 1st and 10 play $i$ in dataset of plays

$x_i$ = game-state vector of play $i$

$y_i = 1$ if team with possession on play $i$ wins the game else 0

want $\hat{y}_i = \hat{f}(x_i)$

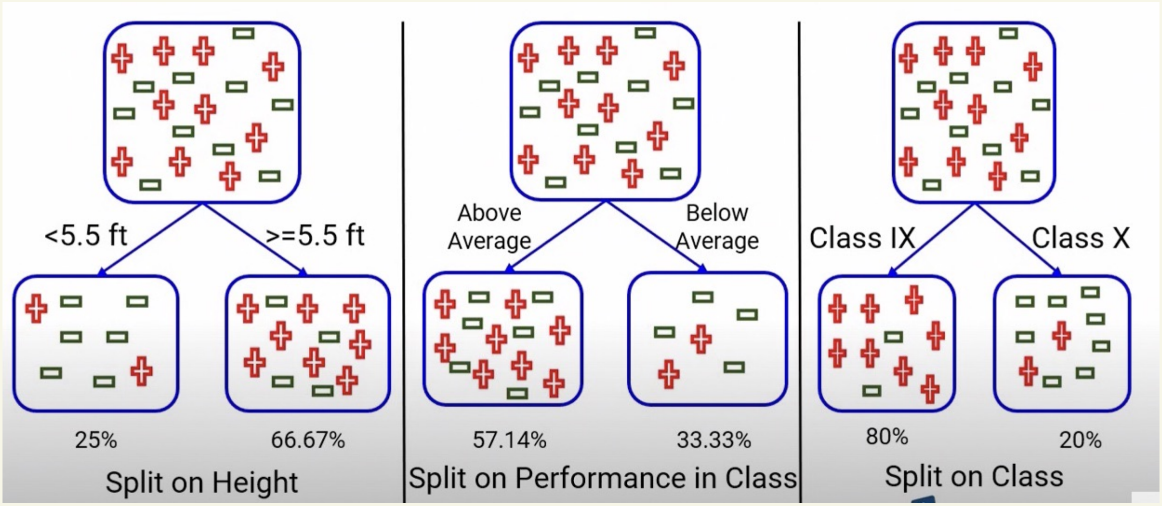
* Will focus on tree ML models easy ~~to~~ fit in R/ widely used.

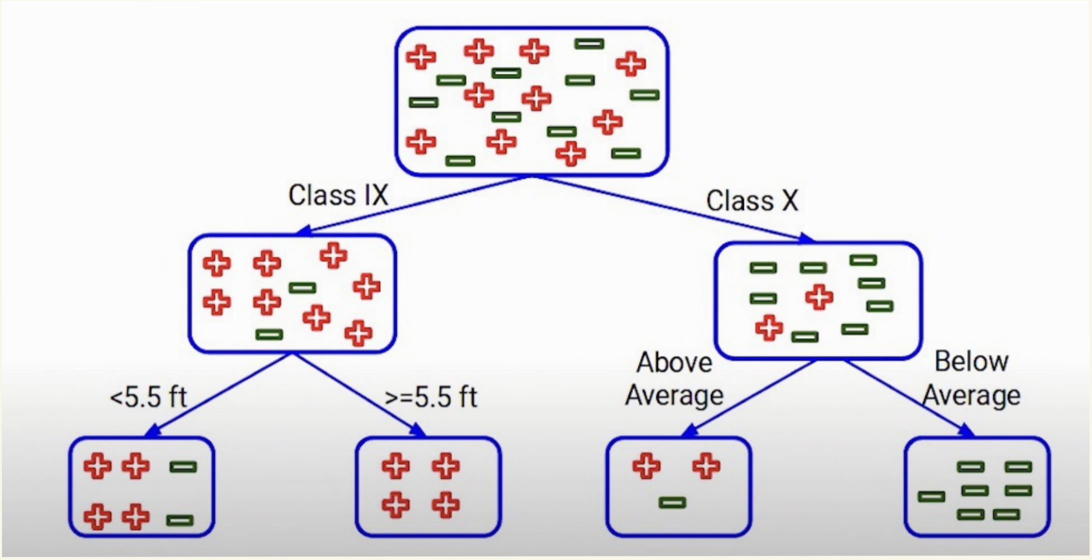
ML: select $\hat{y}_i$ to be as "close" to $y_i$ as possible

→ logloss

$$\min_{\hat{y}} \sum_{i=1}^{n} \left( y_i \log \hat{y}_i + (1-y_i) \log (1-\hat{y}_i) \right)$$

# Decision Trees



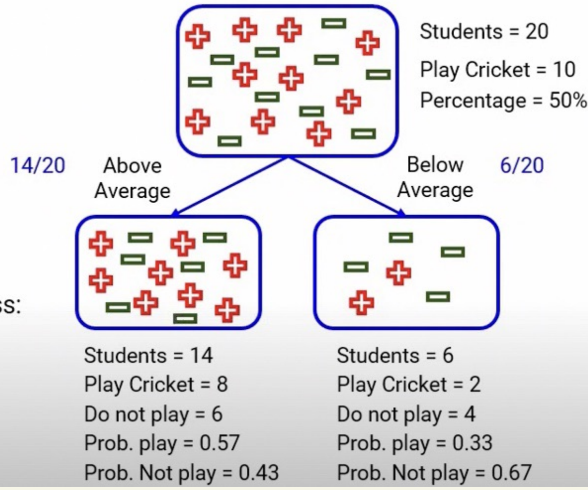

Class IX
Class X
<5.5 ft
>=5.5 ft
Above
Average
Below
Average

## Split on Performance in Class

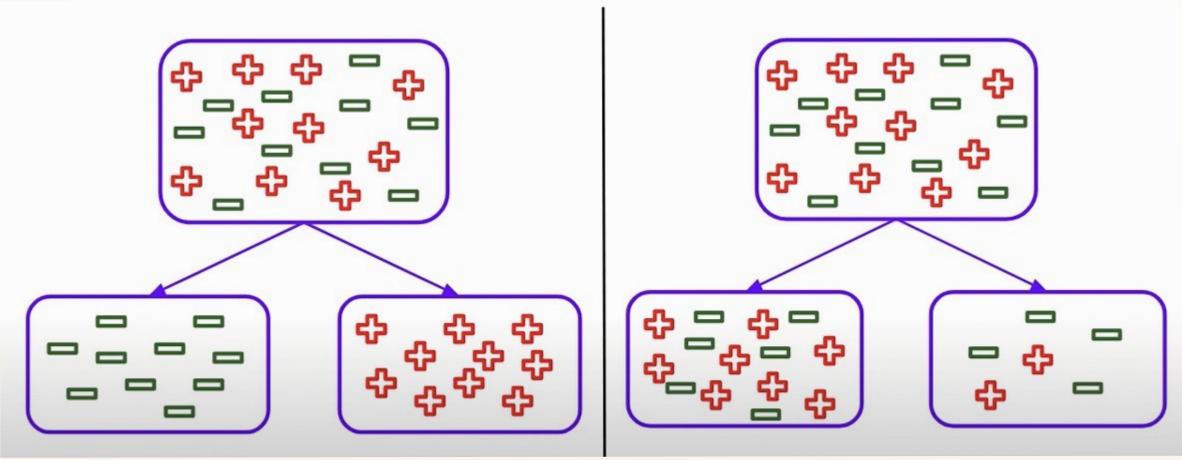
- Gini Impurity: sub-node Above Average:
  1 - [(0.57)*(0.57) + (0.43)*(0.43)] = 0.49

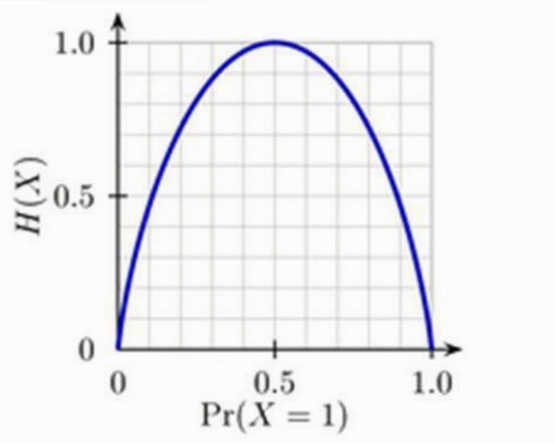
- Gini Impurity: sub-node Below Average:
  1 - [(0.33)*(0.33) + (0.67)*(0.67)] = 0.44

- Weighted Gini Impurity: Performance in Class:
  (14/20)*0.49 + (6/20)*0.44 = 0.475

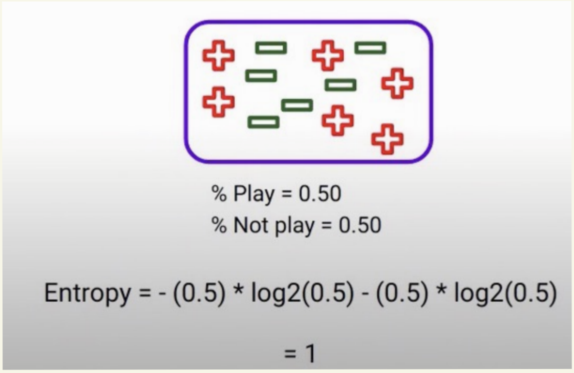


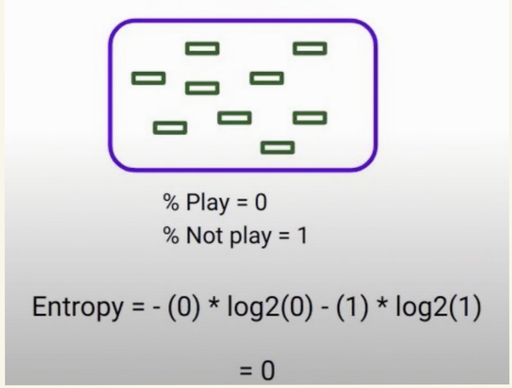

| Split | Weighted Gini Impurity |
|---|---|
| Performance in Class | 0.475 |
| Class | 0.32 |

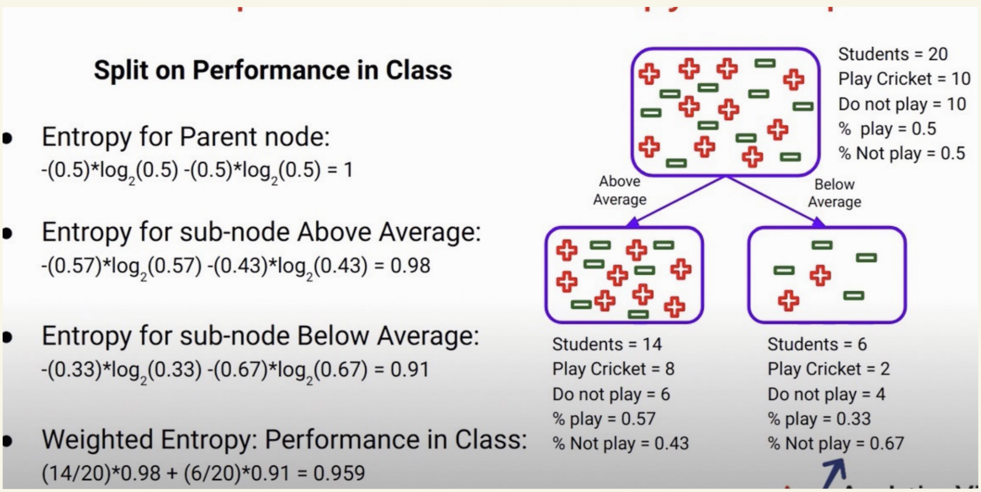




1.0
0.5
0
H(X)
0
0.5
1.0
Pr(X = 1)

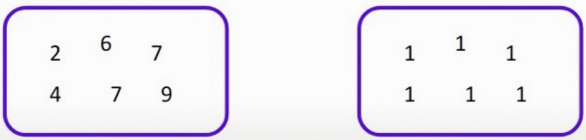

% Play = 0.50
% Not play = 0.50
Entropy = - (0.5) * log2(0.5) - (0.5) * log2(0.5)
= 1


% Play = 0
% Not play = 1
Entropy = - (0) * log2(0) - (1) * log2(1)
= 0


Split on Performance in Class
Entropy for Parent node:
-(0.5)*log2(0.5) -(0.5)*log2(0.5) = 1
Entropy for sub-node Above Average:
-(0.57)*log2(0.57) -(0.43)*log2(0.43) = 0.98
Entropy for sub-node Below Average:
-(0.33)*log2(0.33) -(0.67)*log2(0.67) = 0.91
Weighted Entropy: Performance in Class:
(14/20)*0.98 + (6/20)*0.91 = 0.959
Students = 20
Play Cricket = 10
Do not play = 10
% play = 0.5
% Not play = 0.5
Above Average
Below Average
Students = 14
Play Cricket = 8
Do not play = 6
% play = 0.57
% Not play = 0.43
Students = 6
Play Cricket = 2
Do not play = 4
% play = 0.33
% Not play = 0.67

| Split | Entropy | Information Gain |
|---|---|---|
| Performance in Class | 0.959 | 0.041 |
| Class | 0.722 | 0.278 |

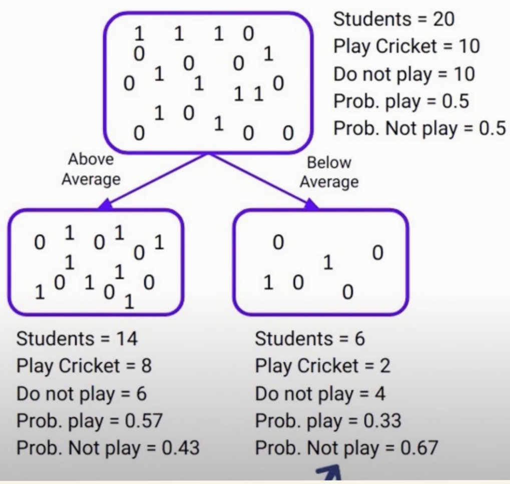
$$\text{Variance} = \Sigma\,[(X - \mu)^2] / n$$




Variance ~ 6

Variance = 0

- Above Average node:
  - Mean = (8*1 + 6*0) / 14 = 0.57
  - Variance = $[8*(1-0.57)^2 + 6*(0-0.57)^2] / 14 = 0.245$
- Below Average node:
  - Mean = (2*1 + 4*0) / 6 = 0.33
  - Variance = $[2*(1-0.33)^2 + 4*(0-0.33)^2] / 6 = 0.222$
- Variance: Performance in Class:
  (14/20)*0.245 + (6/20)*0.222 = 0.238



| Split | Variance |
|---|---|
| Performance in Class | 0.238 |
| Class | 0.16 |