STA 235H - Prediction: Classification and Regression Trees (CART)

Fall 2021

McCombs School of Business, UT Austin

Announcements

No class next week

- Homework 5 is due on Thursday
- Homework 6 will be posted on Thursday (<u>not due until Dec. 2nd</u>)
- I will still hold Office Hours on Tuesday 10/23
- Great resources for the project uploaded to the course website ("Resources --> Machine Learning")
- Start the project early

Where we've been...

- Talking about **bias vs variance** trade-off.
- **Model selection and regularization**: Stepwise selection, Ridge and Lasso regression.
- **K-nearest neighbors**: Simple non-parametric approach. Importance of finding the optimal K.

… and where we're going.

- Continue on our prediction journey:
  - Decision Trees: Classification and Regression Trees (CART)
- Participation: Activity in R.

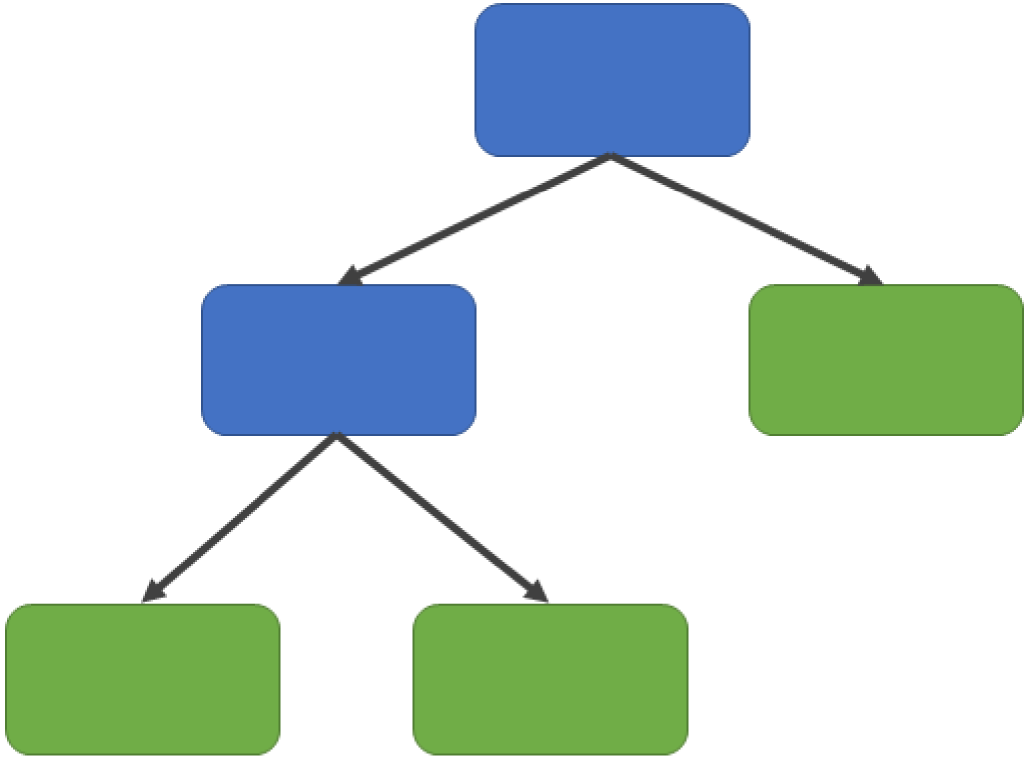
Trees, trees everywhere!

Idea behind Decision Trees

- Create a flow chart for making decisions
  - *How do we classify an individual?*
- … But there are many decisions!
  - How many variables do we use?
  - How do we sort them? In what order do we place them?
  - How do we split them?
  - How deep do we go?

What is the main disadvantage of a shallower tree?

Structure of Decision Trees



Structure:

- Root node
- Internal nodes
- Leaves

Why do we like/not like Decision Trees?

Main advantages

- Simple interpretation
- Mirror human decision-making
- Graphic displays!
- Handle categorical variables

Main disadvantages

- Overfitting
- Not very accurate/not very robust

Let's start with a simple example

Remember our Disney+ example?

Predict who will cancel their subscription

We have some information:

- `city`: Whether the customer lives in a big city or not
- `female`: Whether the customer is female or not
- `age`: Customer's age (in years)
- `logins`: Number of logins to the platform in the past week.
- `mandalorian`: Whether the person has watched the Mandalorian or not.
- `unsubscribe`: Whether they canceled their subscription or not.

The prediction task: Classification

- Our outcome is binary, so this is a classification task.
- Let's start looking at two variables:

City & Mandalorian

- Which one do you think should be at the top of the tree?

Let's look at the data!

City vs. Mandalorian

```
disney <- read.csv("https://raw.githubusercontent.com/maibennett/sta235/main/exampleSite/content/Cla

#Subscribers
disney.train %>% filter(unsubscribe==0) %>% select(city, mandalorian) %>% table

#Unsubscribers
disney.train %>% filter(unsubscribe==1) %>% select(city, mandalorian) %>% table
```

Subscribers

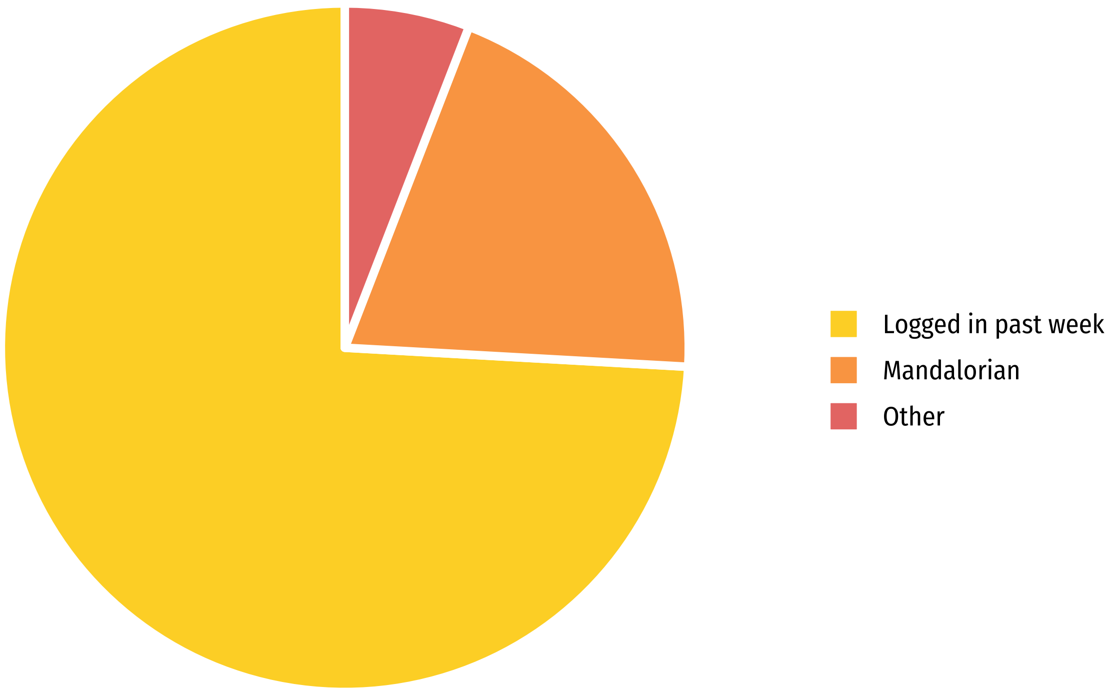
	Mandalorian	
City	0	1
0	173	155
1	1067	1505

Unsubscribers

	Mandalorian	
City	0	1
0	39	345
1	186	1530

Let's talk about the JITT

Answers (up to Monday morning)



How do we decide?

- Recursive Binary Splitting:
  - Divide regions of covariates in two (recursively).
  - This works both for continuous and categorical variables
- We test out every covariate and see which one reduces the error the most in our predictions
  - In regression tasks, we can use RMSE (or MSE).
  - In classification tasks, we can use accuracy/classification error rate, Gini Index, or entropy

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$$

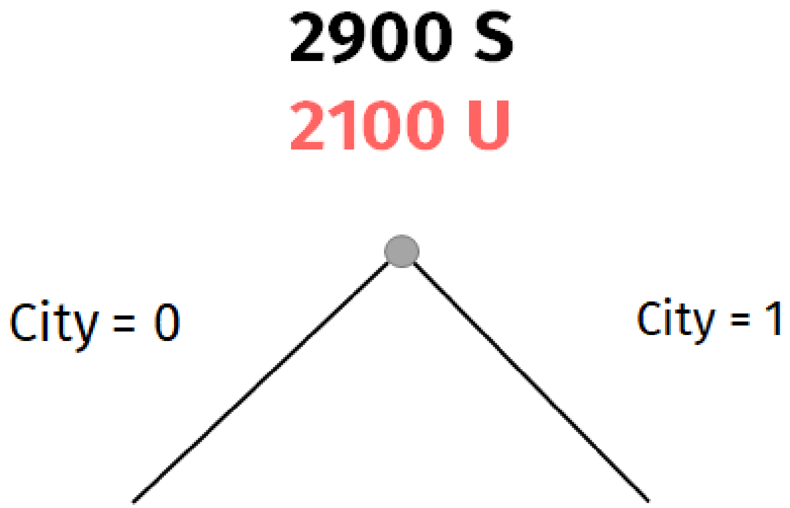
where $\hat{p}_{mk}$ is the proportion of obs. in the m region for class k.

According to the Gini Index, is it better or worse to have a high p_{mk}?

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$$

Let's split by city first...

	Subscribers			Unsubscribers	
	Mandalorian			Mandalorian	
City	0	1	City	0	1
0	173	155	0	39	345
1	1067	1505	1	186	1530



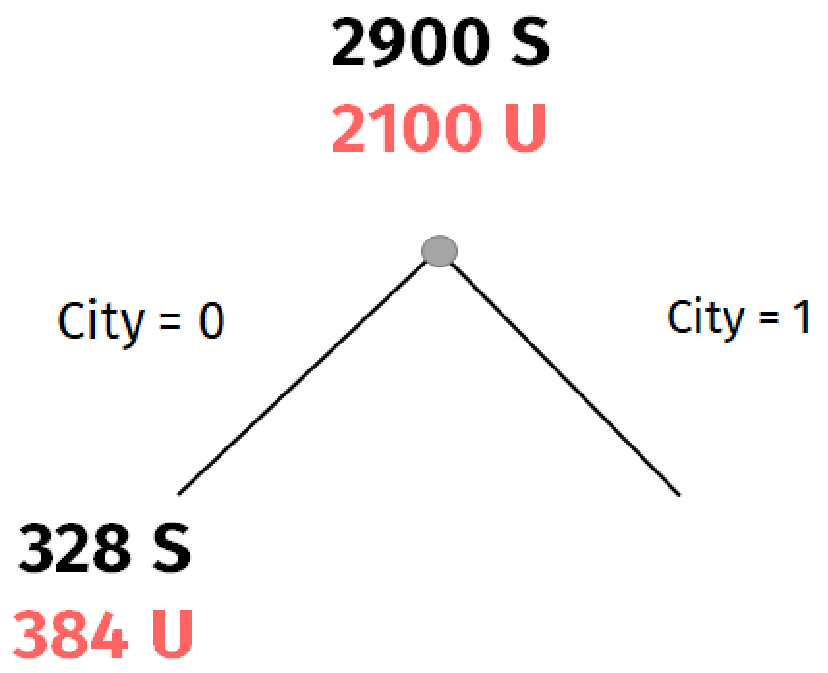
Let's split by city first...

Subscribers

	Mandalorian	
City	0	1
0	173	155
1	1067	1505

Unsubscribers

	Mandalorian	
City	0	1
0	39	345
1	186	1530



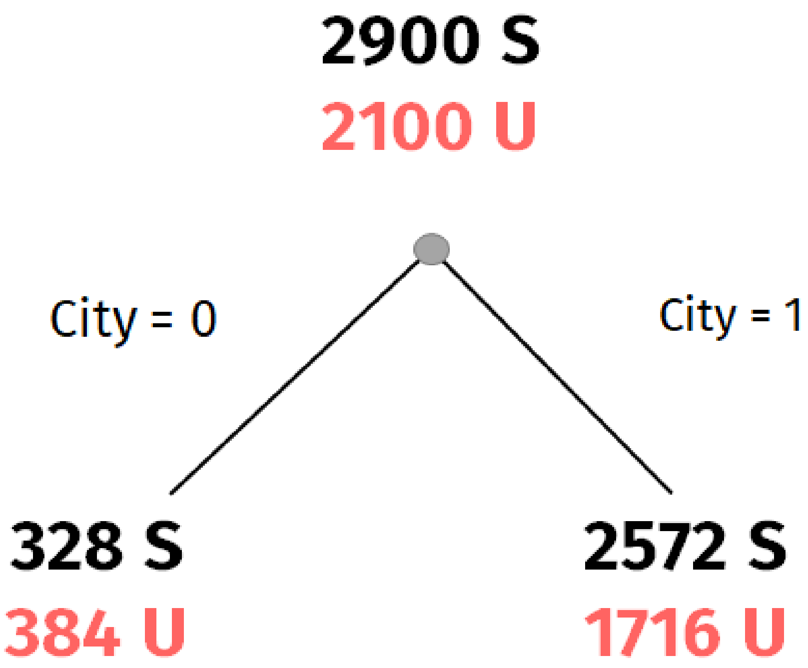
Let's split by city first...

Subscribers

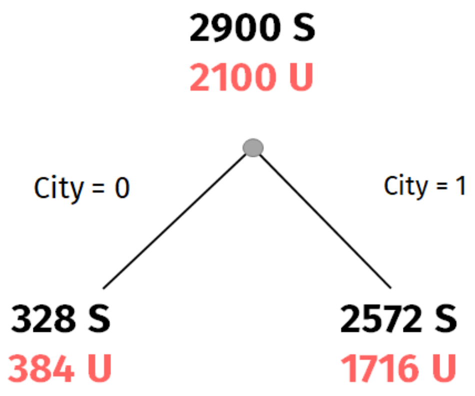
	Mandalorian	
City	0	1
0	173	155
1	1067	1505

Unsubscribers

	Mandalorian	
City	0	1
0	39	345
1	186	1530

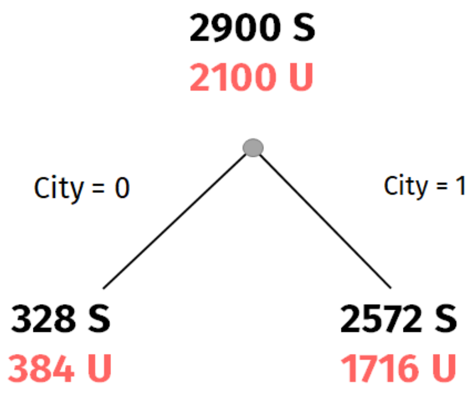


Calculate Gini for city



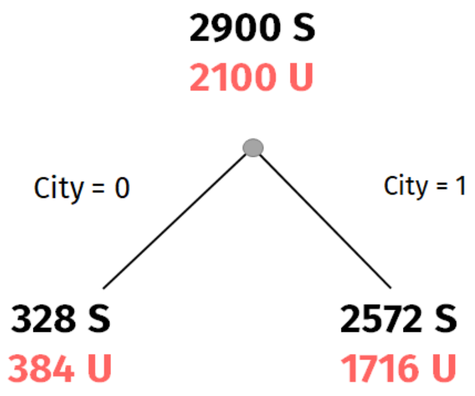
$$G_0 = \sum_{k=1}^{K} p_{mk}(1 - p_{mk}) = ?$$

Calculate Gini for city



$$G_0 = \sum_{k=0}^{1} p_{0k}(1 - p_{0k}) = \frac{384}{328+384} \cdot (1 - \frac{384}{328+384}) + \frac{328}{328+384} \cdot (1 - \frac{328}{328+384}) = 0.497$$

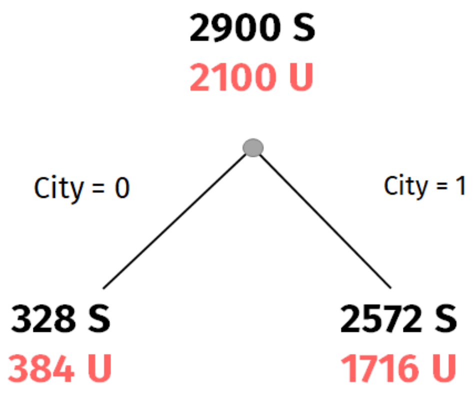
Calculate Gini for city



$$G_1 = \sum_{k=0}^{1} p_{1k}(1 - p_{1k}) = \frac{1716}{1716 + 2572} \cdot (1 - \frac{1716}{1716 + 2572}) + \frac{2572}{1716 + 2572} \cdot (1 - \frac{2572}{1716 + 2572}) =$$

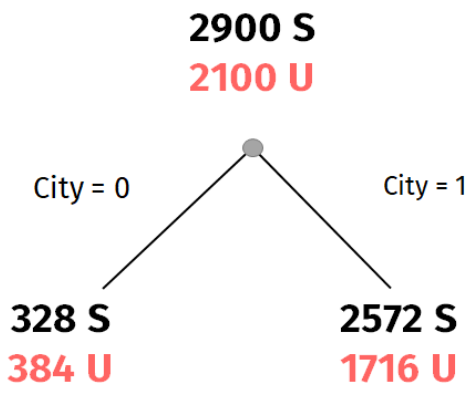
$$= 0.48$$

Calculate Gini for city



$$G = \omega_0 G_0 + \omega_1 G_1$$

Calculate Gini for city

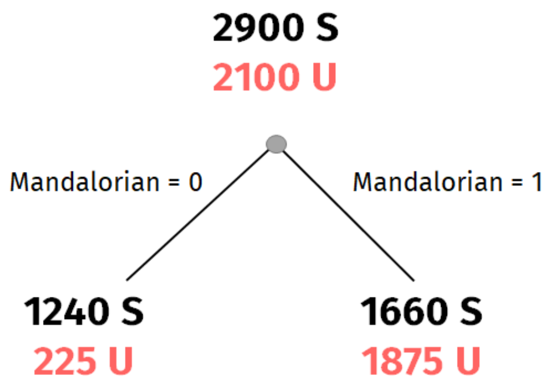


$$G = \omega_0 G_0 + \omega_1 G_1$$

$$G = \frac{384 + 328}{384 + 328 + 1716 + 2572} G_0 + \frac{1716 + 2572}{384 + 328 + 1716 + 2572} G_1$$

$$G = 0.482$$

And we can do the same for mandalorian



$$
\begin{aligned}
G =& \omega_0\left(\frac{225}{1465}\cdot\left(1-\frac{225}{1465}\right)+\frac{1240}{1465}\cdot\left(1-\frac{1240}{1465}\right)\right)+\\
& \omega_1\left(\frac{1875}{3535}\cdot\left(1-\frac{1875}{3535}\right)+\frac{1660}{3535}\cdot\left(1-\frac{1660}{3535}\right)\right)\\
=& 0.428
\end{aligned}
$$

Which variable would you choose for prediction?

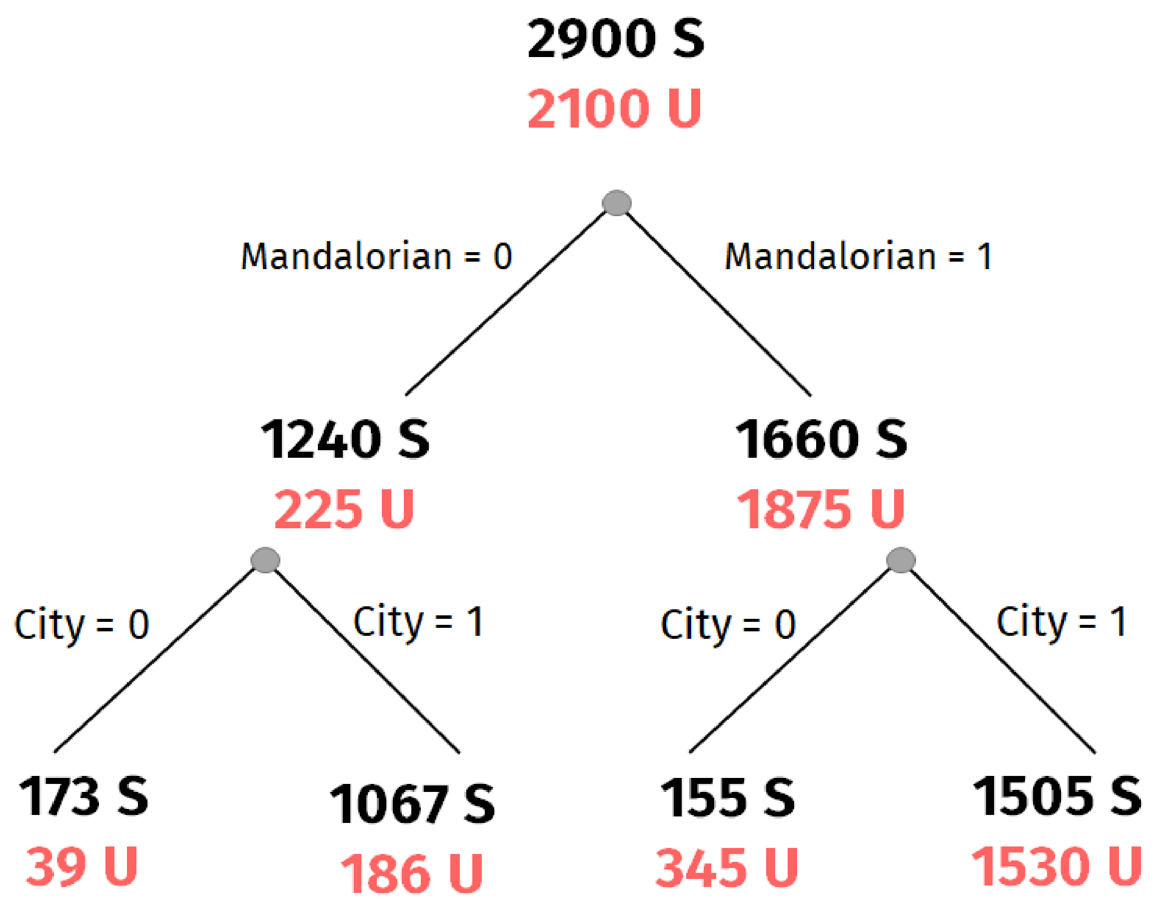
Choosing predictors

- From the previous exercise, we can see that using `mandalorian` yields a lower Gini compared to `city` (0.428 vs. 0.482)

But we have more variables

How do we choose?

Ok, but how about including other variables?



Basic Algorithm

1) Start at the root node

2) Split the parent node at covariate x_i to minimize the sum of child node impurities

3) Stop if leaves are pure or early stopping criteria is satisfied, else repeat step (1) and (2) for each new child nodes

4) Prune your tree according to a complexity parameter (cp)

5) Assign the average outcome (regression) or the majority (classification) in each leaf.

Adapted from "Machine Learning FAQs" (Raschka, 2021)

Hyper-parameter: Complexity parameter

- Measure of how much a split should improve prediction for it to be worth it.

$$\sum_{m=1}^{|T|} \sum_{i:i \in R_m} (y_i - \hat{y}_i)^2 + \alpha |T|$$

- $|T|$: Number of terminal nodes or leaves (e.g. size of the tree)
- R_m: Predictor space of the mth leaf
- α: Tuning parameter

What happens if $\alpha = 0$?

Let's see how to do it in R!

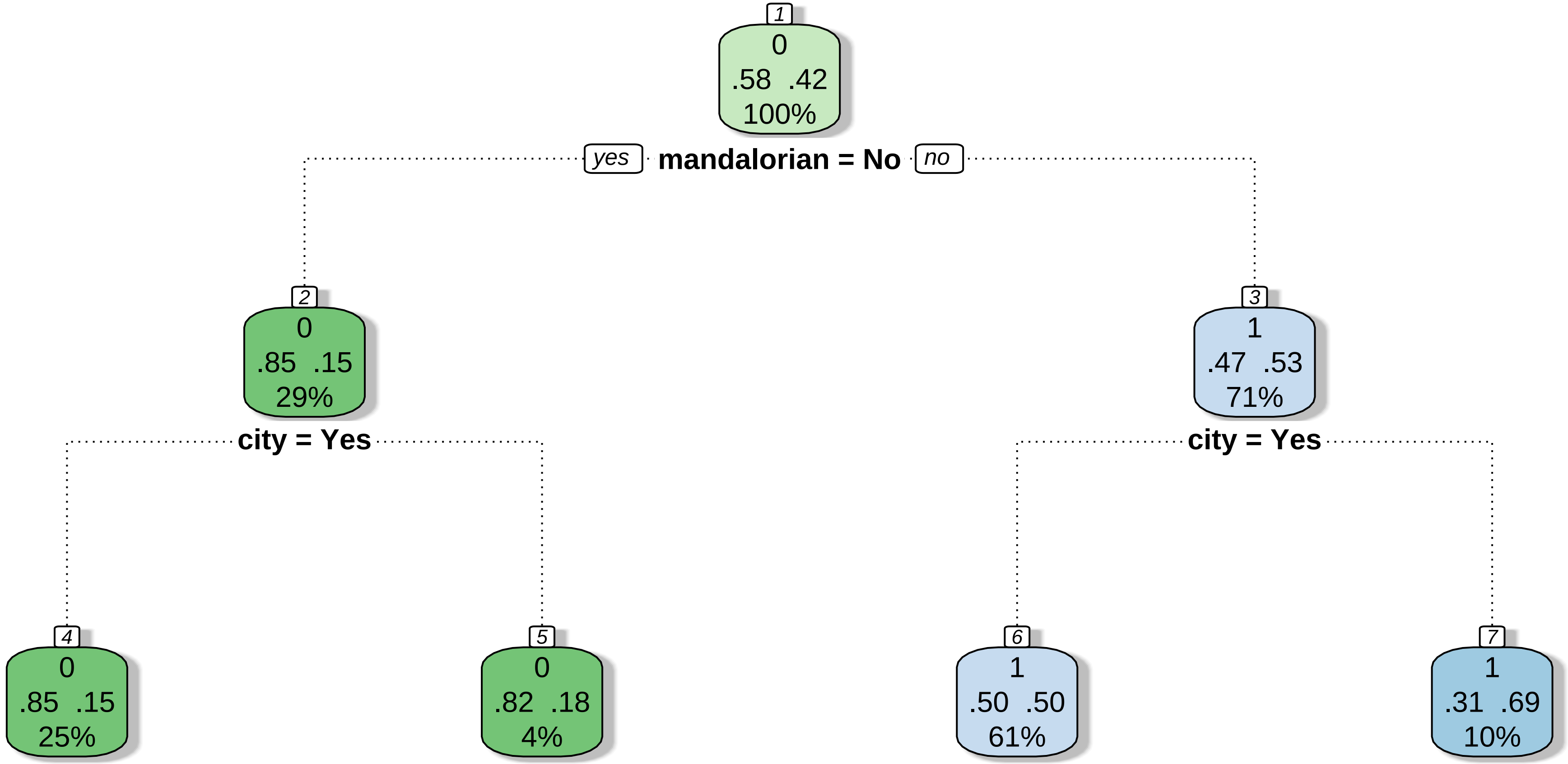
```r
library(caret)
library(rpart)

set.seed(100)

ct <- train(
  unsubscribe ~ ., data = d.train, #remember your outcome needs to be a factor!
  method = "rpart", # The method is called rpart
  trControl = trainControl("cv", number = 10),
  tuneLength = 15
)
```

Let's go to R

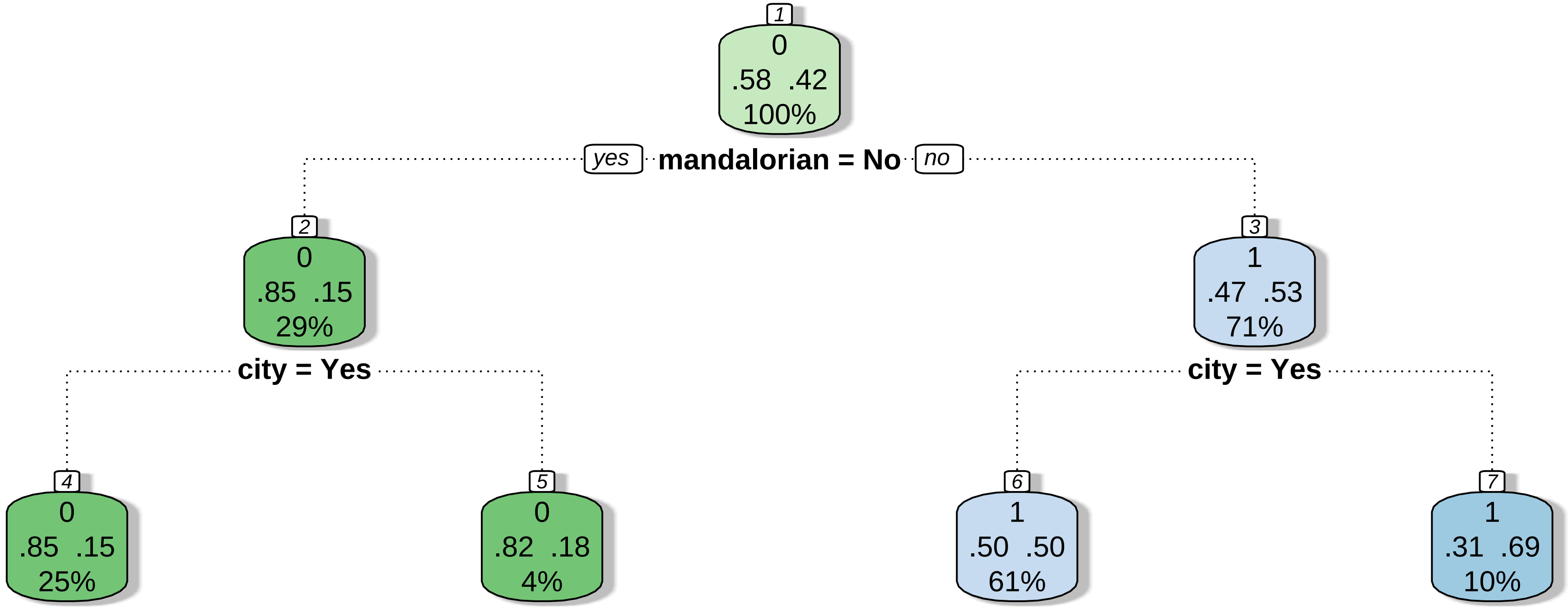
We could also provide a specific complexity parameter

```
library(rpart)

d.train <- disney.train %>% select(mandalorian, city, unsubscribe)

set.seed(100)

m1 <- rpart(unsubscribe ~., data = d.train, method = "class",
            control = rpart.control(minsplit = 1, cp=-1))
```

Tree using all the covariates



What do you think the percentages in the leaves represent?



Some parameters that might be important

```
ct <- train(
  unsubscribe ~ ., data = d.train,
  method = "rpart",
  trControl = trainControl("cv", number = 10),
  tuneGrid = expand.grid(cp = seq(0,2, by = 0.01)),
  control = rpart.control(minsplit = 20)
)
```

- `cp`: Complexity parameter
  - Split must decrease the overall lack of fit by a factor of `cp`, or is not attempted.
  - Parameter for **pruning the tree**.
  - Higher `cp`, smaller the tree!

Some parameters that might be important

```
ct <- train(
  unsubscribe ~ ., data = d.train,
  method = "rpart",
  trControl = trainControl("cv", number = 10),
  tuneGrid = expand.grid(cp = seq(0,2, by = 0.01)),
  control = rpart.control(minsplit = 20)
)
```

- `minsplit`: Min. number of obs in a node to attempt a split.

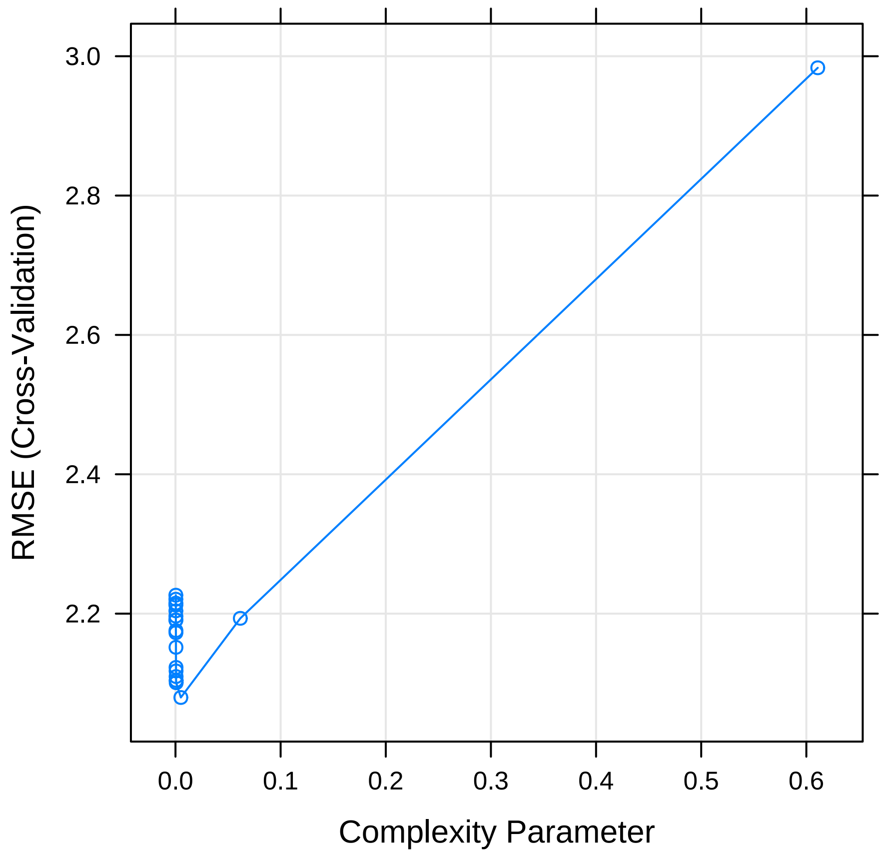
Regression Trees

Regression Trees

- Outcome is continuous
- Very similar to what we have seen with classification trees:
  - Predicted outcome is the mean outcome for the leaf/region.

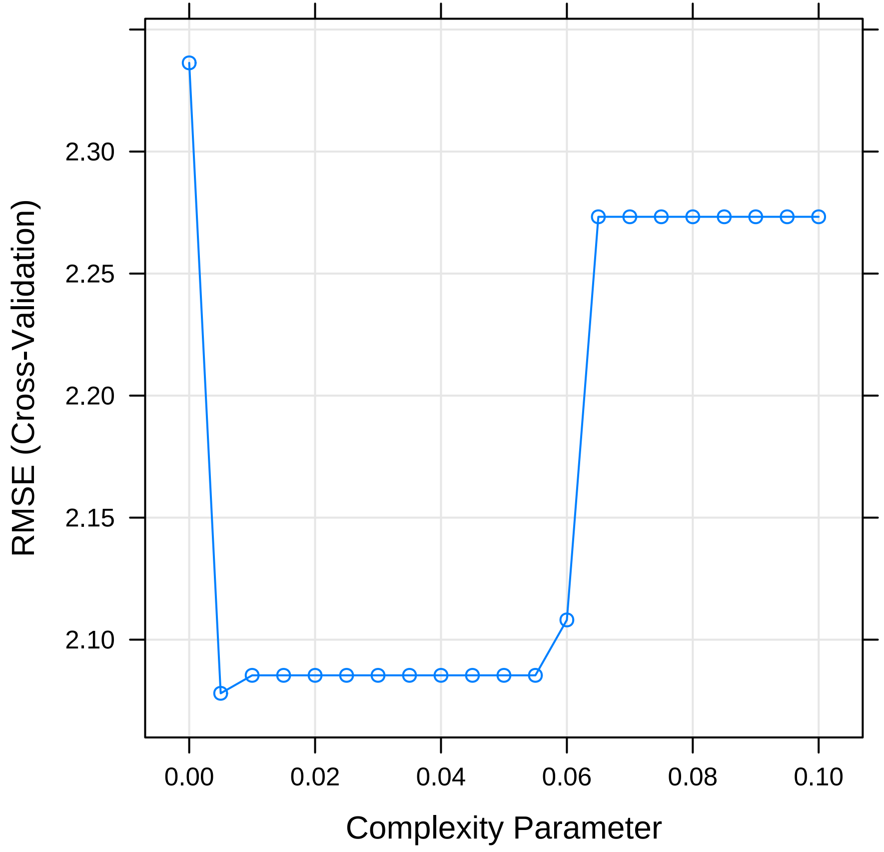
In R is basically the same

```
library(caret)
set.seed(100)

mcv <- train(
  logins ~. - unsubscribe, data = disney.train
  method = "rpart",
  trControl = trainControl("cv", number = 10),
  tuneLength = 20
  )

plot(mcv)
```

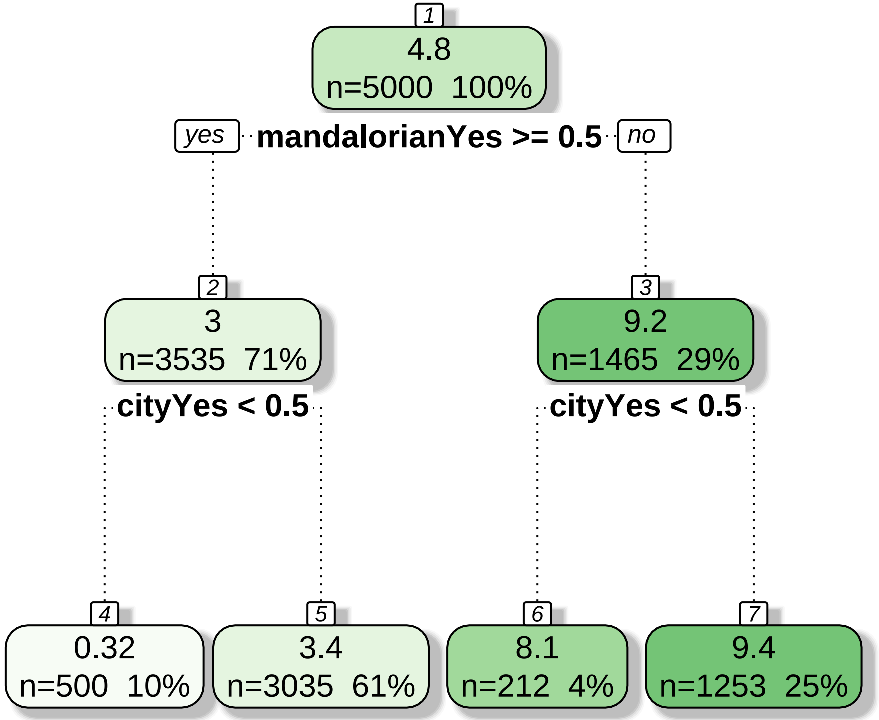


Providing a specific grid for cp

```r
library(caret)
set.seed(100)

tuneGrid <- expand.grid(cp = seq(0, 0.1, 0.005

mcv <- train(
  logins ~. - unsubscribe, data = disney.train
  method = "rpart",
  trControl = trainControl("cv", number = 10),
  tuneGrid = tuneGrid
  )

plot(mcv)
```



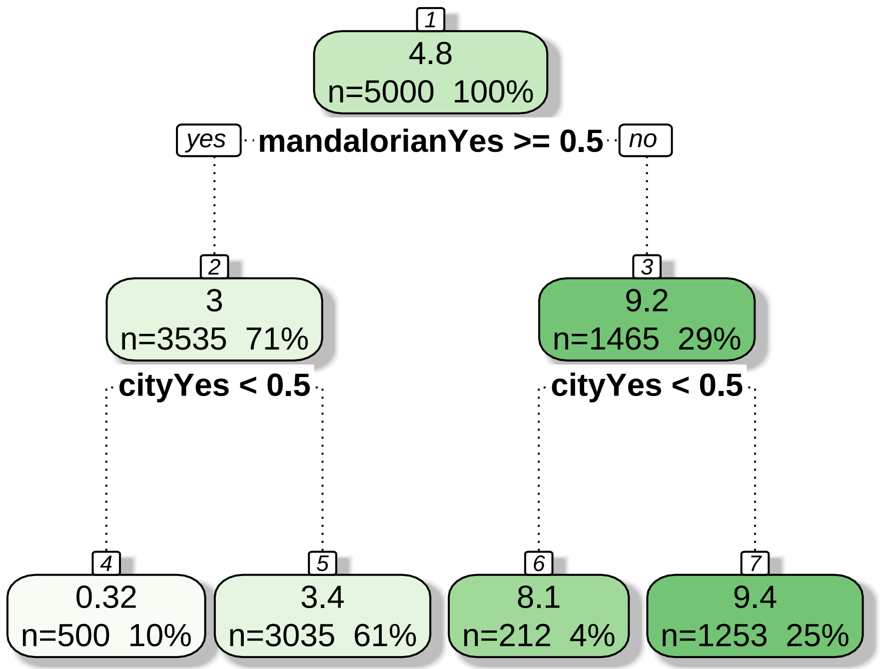
Plot the tree




```
mcv$finalModel
```

```
## n= 5000
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
## 1) root 5000 66387.3700 4.806800
##   2) mandalorianYes>=0.5 3535 24633.5000 2.973409
##     4) cityYes< 0.5 500   517.1580 0.322000 *
##     5) cityYes>=0.5 3035 20022.2800 3.410214 *
##   3) mandalorianYes< 0.5 1465  1200.0180 9.230717
##     6) cityYes< 0.5 212   132.2028 8.061321 *
##     7) cityYes>=0.5 1253   728.8571 9.428571 *
```

What would the predicted value be for a customer who hasn't watched The Mandalorian and lives in a city?




```
mcv$finalModel
```

```
## n= 5000
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
## 1) root 5000 66387.3700 4.806800
##   2) mandalorianYes>=0.5 3535 24633.5000 2.973409
##     4) cityYes< 0.5 500   517.1580 0.322000 *
##     5) cityYes>=0.5 3035 20022.2800 3.410214 *
##   3) mandalorianYes< 0.5 1465  1200.0180 9.230717
##     6) cityYes< 0.5 212   132.2028 8.061321 *
##     7) cityYes>=0.5 1253   728.8571 9.428571 *
```

Main takeaways of decision trees

Main advantages:

- Easy to interpret and explain (you can plot them!)
- Mirrors human decision-making.
- Can handle qualitative predictors (without need for dummies).

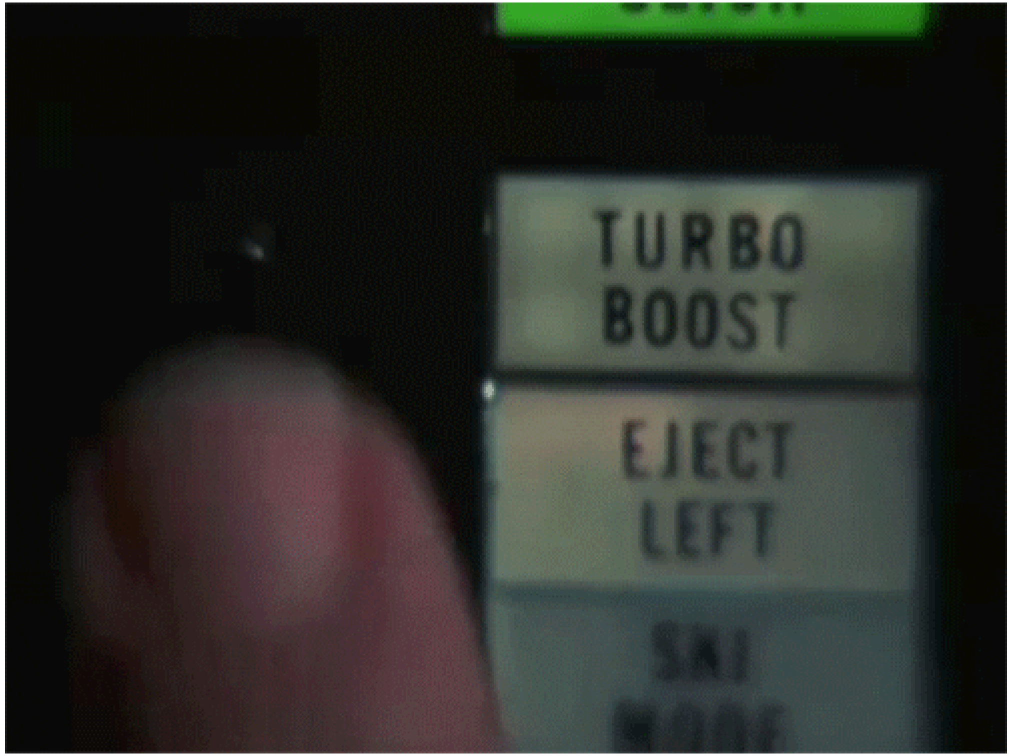
Main disadvantages:

- Accuracy not as high as other methods
- Very sensitive to training data (e.g. overfitting)

Next class

Use of decision trees as building blocks for **more powerful prediction methods!**

- Bagging
- Random Forests
- Boosting



References

- James, G. et al. (2021). "Introduction to Statistical Learning with Applications in R". *Springer. Chapter 8*
- Starmer, J.. (2018). "Decision Trees". *Video materials from StatQuest (YouTube).*
- STDHA. (2018). "CART Model: Decision Tree Essentials"