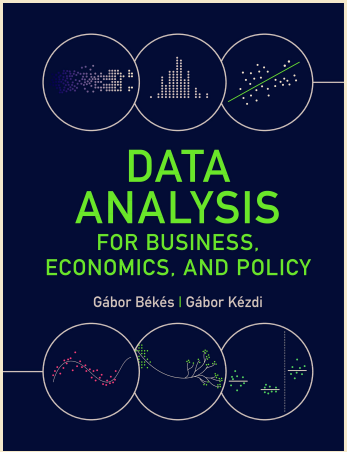
Békés-Kézdi: Data Analysis, Chapter 15: Introduction to Machine Learning



Data Analysis for Business, Economics, and Policy

Gábor Békés (Central European University)
Gábor Kézdi (University of Michigan)

Cambridge University Press, 2021

gabors-data-analysis.com

Central European University

Version: v3.1 License: CC BY-NC 4.0

Any comments or suggestions:
gabors.da.contact@gmail.com

Regression, LASSO, machine learning

- Regression
  - Analyst defines variables via feature engineering, including functional form selection (polynomials, splines, interactions)
  - Analyst defines a set of possible models
  - Analyst selects best model trying out various options (Best model selected by cross-validated RMSE)

Regression, LASSO, machine learning

- Regression
  - Analyst defines variables via feature engineering, including functional form selection (polynomials, splines, interactions)
  - Analyst defines a set of possible models
  - Analyst selects best model trying out various options (Best model selected by cross-validated RMSE)
- Regression with LASSO
  - Analyst defines variables via feature engineering, including functional form selection (polynomials, splines, interactions)
  - Analyst defines broadest model
  - Algorithm selects best model

Regression, LASSO, machine learning

- Regression
  - Analyst defines variables via feature engineering, including functional form selection (polynomials, splines, interactions)
  - Analyst defines a set of possible models
  - Analyst selects best model trying out various options (Best model selected by cross-validated RMSE)
- Regression with LASSO
  - Analyst defines variables via feature engineering, including functional form selection (polynomials, splines, interactions)
  - Analyst defines broadest model
  - Algorithm selects best model
- Machine learning (Random Forest)
  - Analyst defines set of variables. No functional form selection.
  - Algorithm defines a variety of possible models.
  - Algorithm selects best model (Best model selected by cross-validated RMSE)

How does ML work?

- Regression – finds a single solution.
  - You run an OLS regression and it yields one single output (estimated coefficients), calculated by formulae.
  - LASSO a numerical algorithm finds coefficients and the tuning parameter.
- Machine learning is different
  - No single best solution by formulae
  - Search through a set of possible prediction models
    - That best captures the relationship.
- In terms of prediction
  - Both estimate the model on the training set
  - And we avoid overfitting on the test set (and hope for external validity).

What is machine learning?

"It is harder than one might think to come up with an operational definition of machine learning." (Susan Athey)

Typical ML prediction procedure

- We start with the data at hand.
- Create work and holdout set
- In work set: *k*-folds: training and test sets, use cross-validation.
- Pick model by best fit (e.g. RMSE) in test sets
- Best model, re-estimate on work set
- Evaluate on the hold-out sample to estimate the fit we can expect on the live data.

Underfitting and overfitting

- Some ideas are the same as before:
  - Underfitting - you could add more predictors or more complicated functional form to improve predictive power
  - Overfitting - you added too much. The model fits the noise element.
  - Trade-off between risks
- Machine learning techniques may allow for more effective ways to model complex relationships.
- Covered in this course
  - Decision trees: CART
  - Ensemble method 1: Random forest
  - Ensemble method 2: Gradient boosting
  - Not covered: support vector machines, neural nets, deep learning

Roadmap – chapters 15-16

- Growing one tree
  - The idea of regression trees (or CART)
  - Understand ML basics - finding an optimal relationship by trying
  - How to develop (grow) a tree
- How to make it less prone to overfitting
  - Go back to random sampling, and bootstrap
  - Random forest (or RF)
    - as tweaked bagging
  - Boosting: GBM

Machine learning: Regression Trees with CART

- Regression Tree: Basic idea is that relationships are modelled as a series of binary decisions (splits)
- Classification and Regression Trees (CART) is a regression tree algorithm

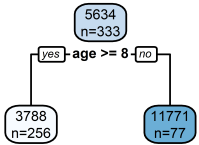
- Decision trees can be applied to both regression and classification problems.
  - Today: target is numerical
  - Classification: target is binary - DA3 Week 5

- Louis Gordon, Richard A. Olshen, "Asymptotically Efficient Solutions to the Classification Problem", The Annals of Statistics, 1978, Vol. 6, No. 3, pp. 515-533
  - Basic machine learning was known.
  - Data + computational power was useful.

Case Study: CART for used cars price prediction

- Case study: used-cars
- The used-cars data includes data on offers of used Toyota Camry cars advertised in the Chicago and Los Angeles areas, in 2018.
- The dataset has N=477 observations.
- Illustration / small dataset + simplicity: Single training–test split.
  - Training set - 70 percent of the original data (N = 333).

Case Study: CART for used cars price prediction



- Relationships between y and an x are modeled as a series of binary decisions (splits)
  - Is the car age below or above 8

Regression tree basics

How are trees and cuts created?

- You can't try out *all* possible segmentation combinations.
  - Why? Because, even if we limit the number of nodes, it is computationally infeasible in most of the cases.
- CART offers a *process* to try out *many* options and pick a set of decision rules
- The outcome of CART is a set of a prediction **rules** as well as predicted values for y

- CART has no formulae
- Not coefficients anymore

Today: Understanding CART in a few steps

- Single predictor
- Multiple predictors
- Looking into the process - which variable matters?
- Dealing with overfitting

- Will show case study along with process

CART intro with single x

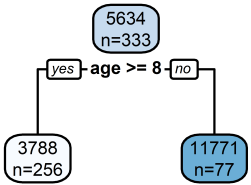
CART with a single predictor

- CART is based on a **binary splitting algorithm**.
- Take a predictor, find a cut-off
- Create two bins - below, above the cutoff
- What is the optimal cut-off?
- The one that creates a separating rule that yields the greatest predicting power
- Predicted value is calculated as mean price at each bin.
  - Why mean? Because it minimizes RMSE given cutoff values.

CART with a single predictor

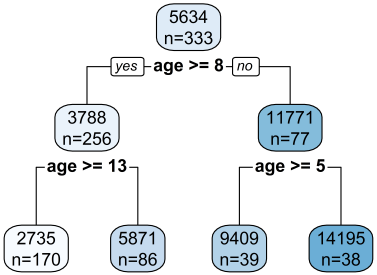
- How we make predictions?
- For each bin, the predicted price will be the simple average of observations in that bin.

CART with a single predictor



- Consider x_1 and pick the value that yields the model with smallest RMSE.
  - Pick the cutoff, that yields the the smallest RMSE.
- Here: check on dozen variables we have
- Pick the 8 yr as cut-off
  - 1-7 – right group, average price is 12 thousand dollars (77 obs).
  - 8 and more - left group, average price is 4 thousand (256 obs).

CART with a single predictor



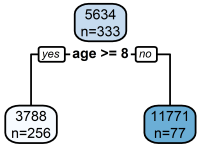
- As with bins, we can have more than 2.
- Once again, how shall we separate? The process
  - Algo picks left branch, and finds the optimal cutoff.
  - Algo picks right branch, and finds the optimal cutoff.
  - Algo compares outcomes, and picks the right branch and the optimal cutoff.
- We now have 3 bins (called nodes).
- We start again, and separate at a node, creating 4 bins (nodes).
- We repeat this, till a stopping point.

CART Overview

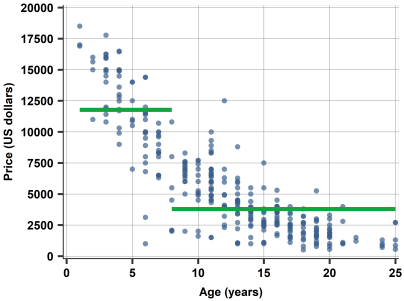
- A regression tree is the result of a series of binary splits of the sample
- Subsamples after splits are called bins (nodes in tree)
- Single binary x, only one split possible; 2 terminal nodes
- Single numerical x, the algorithm starts with one split and makes additional splits
- Splitting continues till a stopping rule
- The result of the algorithm is a set of bins (the terminal nodes) that cover the entire sample.
- The predicted $\hat{y}$ values are the average y value within each bin

CART with a single predictor (age)

CART-decision tree

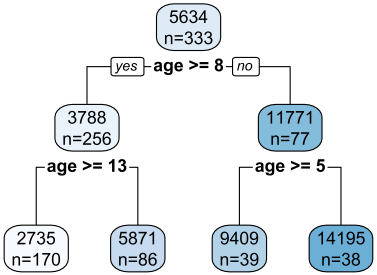


Step function

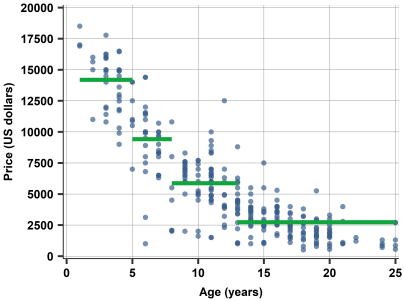


CART with a single predictor (age)

CART-decision tree



Step function



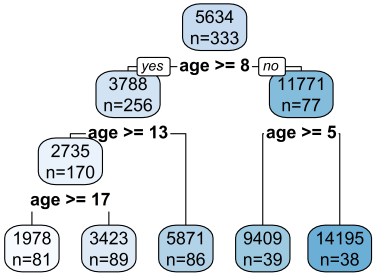
CART M2 output summary

Category	Number of observations	Average_price
Age 0-4	38	14194.84
Age 5-7	39	9408.56
Age 8-12	86	5870.52
Age 13 or more	170	2734.54

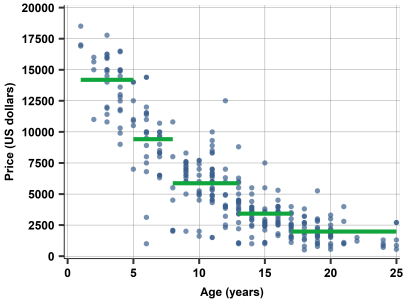
Note: The predicted y values from CART M2, a regression tree grown by CART allowing for three levels.

CART with a single predictor (age)

CART-decision tree

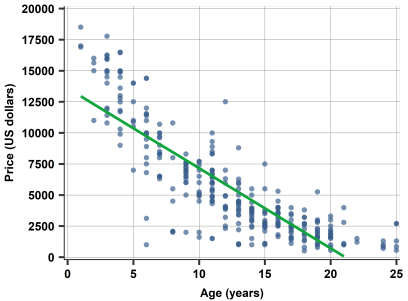


Step function

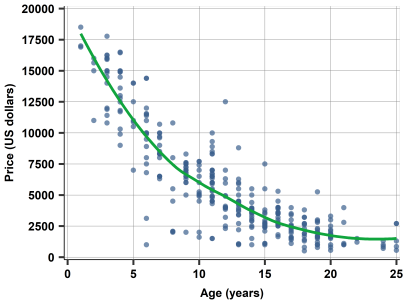


CART with a single predictor (age)

Linear fit



Fit with Loess



Predicting price with age: a comparison of CART and OLS models

Model	Description	RMSE
CART M1	2 term. nodes	2781.06
CART M2	4 term. nodes	2074.46
CART M3	5 term. nodes	1969.52
OLS M1	1 variable only	2357.01

Note: Age is the only predictor. RMSE from single 30% test set.

CART vs Linear Regression 1

- Fitting rule: similar

- There is a basic similarity.
- Both aim at finding a model to minimize a loss function.
- It is the same loss function of sum of squared residuals (MSE).

- Regression finds a global optimum
- CART may not find the best tree (see more later)

CART vs Linear regression 2

- Getting Linear relationships: different
- We are after the relationship between Y and X
- CART can pick up non-linear patterns and do a better job than linear regression.
  - No need to do splines, etc.
- But, if the Y,X relationship *is* linear, it can just approximate it...
  - Regression may be more efficient.

CART vs Linear regression 3

Predictions: different logic

- For regression
  - if $x_1 \neq x_2$
  - then $\hat{y}_1 \neq \hat{y}_2$
- For CART
  - if $x_1 \neq x_2$
  - BUT x_1, x_2 in $R_i x$
  - then $\hat{y}_1 = \hat{y}_2$

- examples for cars
  - age=2, age=3
  - price will be different by 0.5K
    - age=2, age=3
    - both in node age<3.5
    - price is same (14K)

CART process with many x

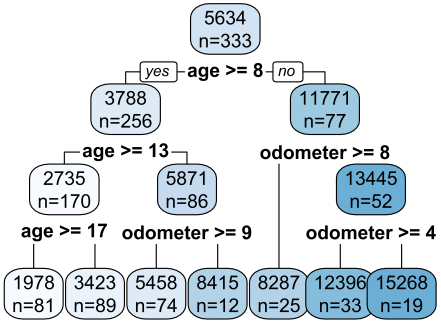
Multiple predictors

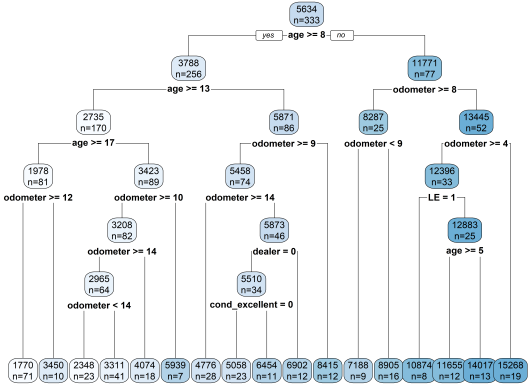
- With multiple predictors, as we grow trees, at each step, we consider
  - all the terminal nodes
  - all predictors and
  - all possible cut-offs for each and every predictor
- Pick the node-predictor-cutoff that leads to the largest drop in MSE.
- Repeat.

CART with many predictor

- Separation process
- We have a set of predictors, $x_1 ... x_p$
- Start with x_1
  - Look through all values.
    - For $x_1 = k$, see calculate RMSE
  - Pick the value that yields the model with smallest RMSE.
- Repeat for all predictors
- Pick the predictor AND the cutoff, that yields the the smallest RMSE.

Multiple predictors





Multiple predictors

- With multiple predictors, we consider all predictors and possible cut-offs step by step and grow trees.
- The hard part is to know when to stop.
- Every split will improve fit.
- But: overfitting...
- So we employ a stopping rule.

Stopping rule and prediction

- The binary splitting process continues until a stopping criterion is reached.
- Minimum number of observations in a bin for further splitting
- The number of observations in any terminal node.
- Minimum fit improvement - a split is made only if it improves the fit of the by a minimum amount = the complexity parameter(cp)
- Example: cp=0.001 split needs to improve the R-squared by at least 0.001.

- Stopping rule strictness will define size of tree.

- *Could* use cross-validation to find optimal tree size. But that turns out to be not the best way to get the best tree... So we will not do that.

Case Study: CART process (multivar, smaller tree)

- Output gives sample mean, and total variance (i.e. Mean SSR, where resid= = $\hat{y}$ - $\overline{y}$
- Output table show how every new step yields a smaller and smaller improvement (drop in relative error).
  - The relative error is 1-R^2, similar to linear regression. So here R^2 is about 85%.
  - First single step (age) already very large.
- And at every step is reduced by highest cp (improvement in fit).
  - Until its less than stopping rule (0.01 here).

]

	cp	N split	Rel error
1	0.62976	0	1.0000
2	0.09376	1	0.3702
3	0.07500	2	0.2764
4	0.01660	3	0.2014
5	0.01507	4	0.1848
6	0.01478	5	0.1697
7	0.01000	6	0.1549

CART process review with some jargon

- The method is called **binary splitting**. It is a **top-down**, **greedy** approach.
- **Top-down** because it begins at the top of the tree and then successively splits the predictor space; each split is indicated via two new branches further down on the tree.
- **Greedy** because at each step of the tree-building process, the best split is made at that particular step, rather than looking ahead and picking a split that will lead to a better tree in some future step.
  - Disadvantage: Myopic–does not take into account how a split(r) will affect a split (r+1, r+2, etc)
  - Advantage: it's (very) fast

CART process

- What is the output of CART?
- A set of rules (decisions rules)

- If age is less than 8, go right, otherwise go left.
- If you went left, check if age is greater than 13, go left if yes, go right if not.
- Etc.

CART vs Linear regression 4

- Output: different but with a similarity

- Output for OLS will be a set of coefficient estimates
- Output for CART will be a set of decision rules.

- Similarity
- We can take output that we get in the training data and use it on the test data.
  - We also use it on the live data.

CART overview

- The advantage of CART is pattern discovery and easy interpretation.
- Sometimes easier than linear regression
- Decision trees (=bins) - easy to explain

- As a prediction method, performance is not so great.
- But CART is building block of some top performing ML tools.

CART overview

- The basic advantage of CART is an automatic pattern detection.
  - Well beyond linear relationships
  - Automatic: search method finds optimality at every step
- Disadvantage
  - CART is very likely overfit the data
  - One also has to make arbitrary decisions
    - the stopping criterion, min bucket size
- Note that advantage-disadvantage comes from same aspect: very strong pattern detection.

What's wrong with trees?

- Two key problems
- Overfitting
  - Remember the price - age step-function with 9 nodes.
- Splitting is not robust.
  - Change a few variables, and a split might alter at another point
  - Leading to a completely different tree!
  - This comes from the fact that a split at t, does note take into account future options.

CART vs Linear regression 5

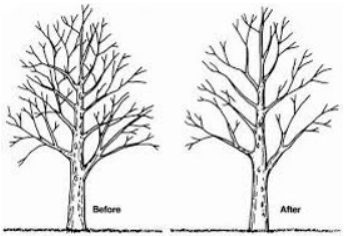
- Robustness: different
- Robustness
  - What happens when we modify a few observations?
- Linear regression is more robust, some coefficients may change a little, but overall result is likely to be similar.
- CART may look very different.

Improving on CART

- A smaller tree with fewer splits might be a better idea
  - More stability
  - Worse in sample fit
- Solution 1: early (tough) stopping rule
  - Stop when drop in RSS is below a threshold (cp)
  - Short-sighted: a split now may be not so important but followed by an important split later
- Solution 2: grow a tree and cut back = pruning
  - Grow a tree as large as possible
  - Cut back
  - Turns out, this is better...

Pruning the tree

- Grow a large tree first, with a stopping rule that lets it grow big
  - and prune it *afterwards* by deleting some of the splits, with the goal of arriving at a better prediction..
- **Cost complexity pruning** (= weakest link pruning) is used to do this.



The idea of pruning

- Pruning is a complicated procedure. We start from the large tree, T_0 with simple stopping rule, such as n>=10
- Deleting splits at the end of the tree, one by one
- Compare a set of possible versions ("subtree") of the tree without a set of nodes.
- Eventually we have smaller version of the large tree we built
- But: this is the tree that has the best out of sample performance
  - Out of all possible trees based on the input features.

- Breiman and co-authors (in 1984): try out many, but not all possible subtrees.

Pruning: how it works: the trade-off

- Key question: what is the optimal amount of pruning?
- Trade-off between a better fit and the overfit risk
- Method is about finding the optimal sub-tree, ie a tree that is nested in T_0, but has fewer nodes
  - More stable / does less overfit
- Try out many subtrees
  - Performance in test sample – select a subtree with lowest test RMSE
  - With a given subtree, we can estimate CV RMSE
- Will do by trying many options

Pruning: how it works: weakest link

- Consider a sequence of trees indexed by tuning parameter α
  - α controls tradeoff fit vs complexity.
- $|T|$ denotes the number of terminal nodes, $m = 1, 2...|T|$
- R_m set of observations in the terminal node m
- $\hat{y}_{Rm}$ predicted value for terminal node m (=mean in training set)

- For each value of α there corresponds a subtree $T \subset T_0$ such that we may find the minimal value of a loss function, L_α, T

$$L_{\alpha,T} = \min_{T,\alpha} \left(\sum_{m=1}^{|T|} \sum_{y_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha |T| \right) \tag{1}$$

Pruning: how it works: loops

Consider two loops

- Define a stopping parameter α, let it go between 0 and 1
  - Compare a set of possible versions (sub-trees) of the tree without a set of nodes
  - Find the best subtree $T(\alpha)$ for a given α, calculate test RMSE for α
- Define another parameter α and repeat
- Do this selection process by CV

Pruning: how it works: best tree for an α

- For a given α, we need to find the best subtree
- To find $T(\alpha)$, ie the best tree for a given α, we successively collapse the internal node that produces the smallest per-node increase in residual error (weakest link pruning)
  - getting rid of one split
  - getting rid of another split
  - Getting rid of 2 splits
  - ...
  - and continue until we produce the single-node (root) tree.
- Pick the subtree with the smallest $L_{\alpha,T}$
- This sequence must contain the best subtree for α: $T(\alpha)$

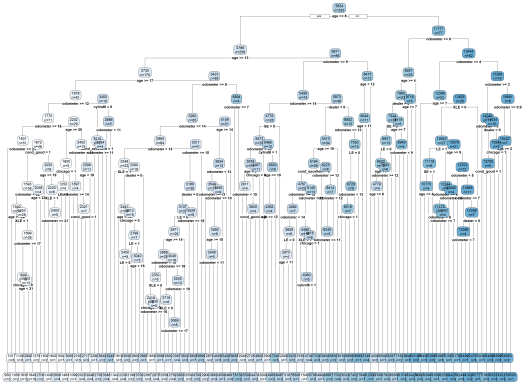
Pruning: Summary

Pruning is the idea that rather than stopping early on, we build a large tree and cut back in smart way.

- Have a loss function $L_{\alpha,T}$
- Have a tuning parameter α
- For each α we try out many trees, one by one getting rid of nodes that add the least
  - Cross validation
- Pick the best subtree for a given α
- End up with a series of subtrees for each α
- Pick the best...

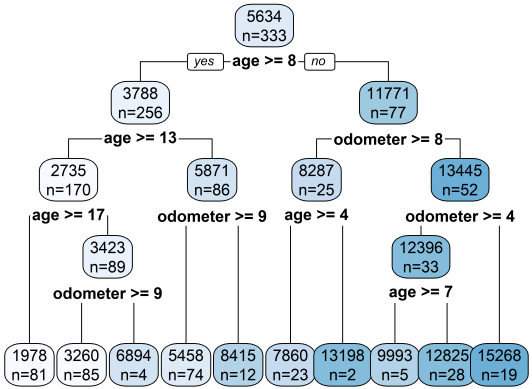
Case study: Pruning

- Grow a very large tree:
- Cp very low and/or stop bucket very low



Case study: Pruning

- Grow a very large tree
- Cp very low and/or stop bucket very low
- Prune: get to fewer nodes.



CART vs Linear Regression 6

- Variable selection process: different
- Key difference is how variables are selected and used.
- Say, we have two predictors: x_1, x_2
- In linear regression, we would typically have $x_1, x_2, x_1 * x_2$
  - Maybe splines, polynomials
  - Eventually, a list of variables
- In tree, we can have many nodes based on cutoffs, capturing
  - functional form
  - interactions
  - Not a list

Variable importance

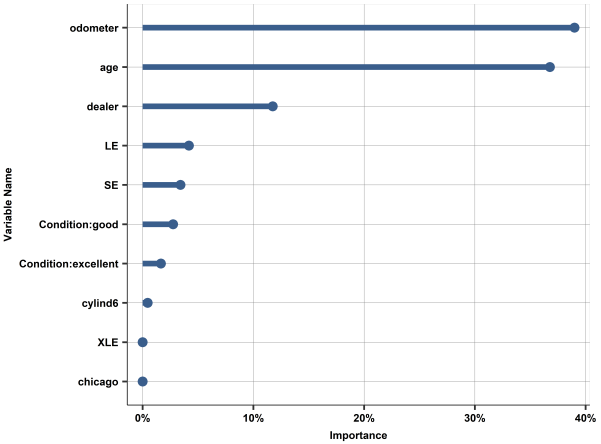
- Learning how predictions are made is rather difficult.
- Unlike for a regression model, we do not have coefficient estimates.
- A variable appear at many splits.
- **Variable Importance:** measures how much fit improved when a particular x variable is used for splitting, summed across all splits in which that variable occurs.

Variable importance

- **Variable Importance:** measures how much fit improved when a particular x variable is used for splitting, summed across all splits in which that variable occurs.
  - Expressed as the share of fit improvement (MSE reduction) due to the particular x variable
  - relative to the overall improvement of fit achieved by the regression tree
  - as opposed to a prediction that does not use any x variables
- The measures how important a variable is in terms of helping improve prediction
- High variable importance means that given variable plays an important role in prediction.

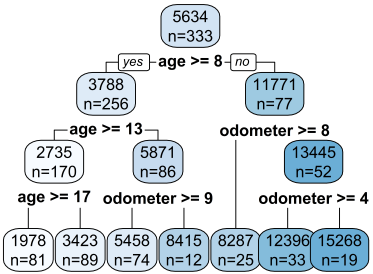
Case Study: Variable importance plot

- Variable importance for a regression tree on the holdout set.
- Odometer and age are about equally important predictors while other variables help substantially less



Variables used

- Important variable are used early and frequently
- Here: **age:odometer**



Questions and answers

- https://gabors-data-analysis.com/part3-qanda
- Q: The Variable importance plot for the CART has small values for variables that are not in the tree shown. How is it possible?
- A: Caret package documentation: "The reduction in the loss function (e.g. mean squared error) attributed to each variable at each split is tabulated and the sum is returned. Also, since there may be candidate variables that are important but are not used in a split, the top competing variables are also tabulated at each split"

CART summary

- Simple version is useful to show some basic correlations
- Helps focus on important variables
- Automatic functional form building
- If wanna use one tree: should grow a large one and prune back
  - Better out-of-sample performance
- For prediction, it turns out, not great....

Pros and Cons of Using a Regression Tree for Prediction

Feature	Linear regression (OLS)	Regression tree (CART)
Solution method	Formula	Algorithm without a formula
Solution goal	Minimize loss (MSE)	Minimize loss (MSE)
Solution optimality	Finds best possible linear regression, but only given the included x variables	Greedy algorithm; does not find best possible tree
Variable selection	Pre-defined list of x variables	No pre-defined list
Main results	Set of coefficient estimates; prediction by plugging into formula	Set of terminal nodes; prediction by specifying values of x variables
Predicted y value and x values	Different $\hat{y}$ for different x values	May have same $\hat{y}$ for different x values if in the same bin
Linear relationship between x and average y	Captures a linear relationship	Approximates linearity by step function
Nonlinear relationship between x and average y	Need to pre-specify functional form to approximate nonlinearity	Approximates nonlinearity by step function

Case Study: Regression tree summary and discussion

- Predictive performance of all regression tree models by the test set RMSE.
- Recall single training–test split.
- OLS M2 linearly includes all seven variables we used for the CART, OLS M3 then adds quadratic terms for age and odometer reading.

Model	Number of variables	Describe	RMSE
CART M1	1	2 levels	2781.06
CART M2	1	3 levels	2074.46
CART M3	1	4 levels	1969.52
CART M4	7	cp = 0.01	1892.96
CART M5	7	cp = 0.002	1892.35
CART M6	7	cp = 0.0001	2072.60
CART M7	7	pruned	1818.09
OLS M2	7	linear	1905.85
OLS M3	7	w/ polynomial terms	1636.50

Case Study: Regression tree summary and discussion

- Pruned CART or small CART are best, pruned is marginally best
- OLS with some feature engineering (functional form for key vars) are better than any CART
- Advantage of CART is that it is automatic, no need to look for functional form. But worse performance than OLS *with* feature engineering.

Main takeaways

- A regression tree is a method for predicting y based on x variables using an automated algorithm that approximates any functional form and any interaction between the x variables
  - A regression tree splits the sample into many bins according to values of the x variables and predicts y as the average within those bins
  - Regression trees can be thought of as non-parametric regressions
  - A regression tree is prone to overfitting the data even after pruning. For this reason, it is rarely used for prediction in itself