# Gradient Boosting Models

Using GBMs in Pricing Models

Greg Sollenberger, FCAS

# Tree-Based Models

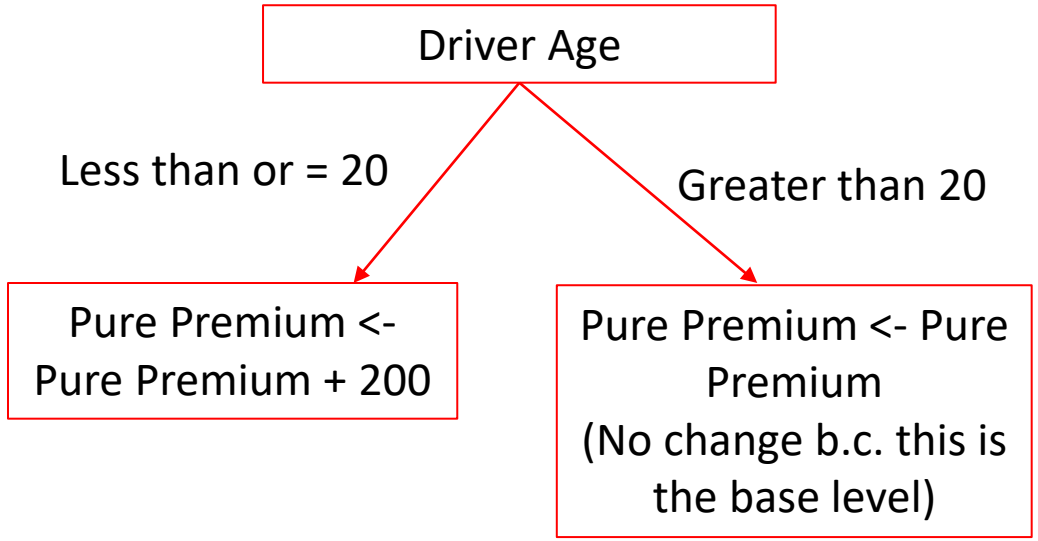
- Single decision trees
  - CART (classification and regression trees)
- Random Forests
  - Averages the output of many trees (hundreds or thousands)
  - Separate trees are non-sequential
- Boosting
  - Uses “weak learners”, trees that individually predict small differences in target
  - Learners ***are*** sequential: result of prior trees gets passed to the next tree
  - Leaners can also be GLMs, but I am focusing here on tree-based methods

# Brief Outline of the GBM Algorithm

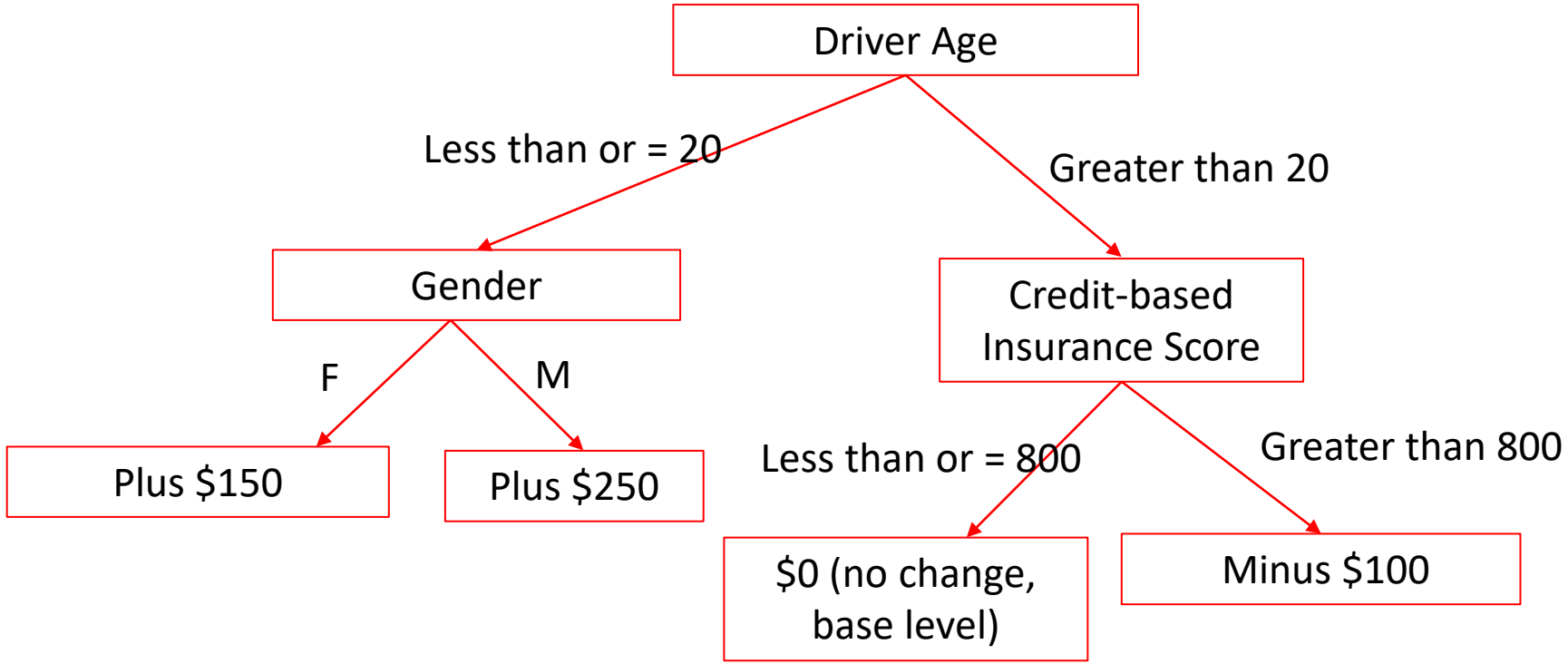
1. Specify the number and depth of trees used to build the model
2. Find the best “split” (e.g. “Driver Age greater than and less than 20”)
3. Find the magnitude of that split (e.g. “Drivers older than 20 have a pure premium $200 lower than drivers younger than 20”)
4. Apply a “shrinkage factor” to the magnitude (e.g. reduce the magnitude of each tree by a factor of 0.1. So use $20 instead of $200.)
5. Apply the tree’s implied credit/debit to each record in the dataset (everyone above 20 gets a $20 credit – model keeps a running tally of each record’s model prediction based on all prior trees).
6. Repeat for N trees (find the next split, find it’s magnitude, discount by the shrinkage factor, debit/credit each record, back to step 2)

# Example of a Simple Decision Tree

- Splits on Driver Age
- Splits at Driver Age = 20
- Determines magnitude of the difference between classes at this split
- Not shown: the GBM will not use the full $200 difference, but will cut it down by some factor (typically in the 0.01 to 0.1 range)
- We are incrementing the debits/credits at each iteration, so the next split "knows about" this previous split.
- The algorithm passes the output of this to the next tree, determines the next split, and so on.



# Example of a Slightly Deeper Decision Tree




- Here is a depth 2 tree
- Each split has an implied increment to pure premium based on the observed differences on either side of the split
- Note that this is building in an interaction term
- Depth 2 trees support 2-way interactions, depth 3 supports 3-way, and so on.
- Depth 1 implies “simple factor” tables, with no interaction terms

# Parameters to a GBM

- Learn rate (aka shrinkage)
  - No need to tune
  - Pick a “low enough” learn rate, say 1-5%
- Number of trees
  - Don’t need to tune
  - If specified correctly, early stopping will halt the model when enough trees are included
  - No need to say “Show me a model with 500 trees, 1000 trees, …”
- Maximum Depth
  - May wish to tune
  - *Elements* text says not to go higher than 8 (8-way interactions?)
  - May wish to set at 1 for practical purposes (run-time, interpretability)
- Bag fraction
  - What fraction of training data to use at each tree?
  - Can sometimes get better performance by setting to other than 1
- Column sample rate
  - How many candidate columns to try in each tree/ at each split?
- Minimum observations at each node
- Chapter 10 of *Elements of Statistical Learning* has more details on how these parameters work
  - Available here: https://web.stanford.edu/~hastie/ElemStatLearn/

# Why Use a GBM?

- Simple answer: they perform better.
  - “I tried a GLM and a GBM, and the GBM gave me a better test metric.”
- They automate the search for curvature.
  - Linear models assume a linear relationship, or a polynomial relationship or some other curvature the user specifies.
  - Tree-based models will automate the search for thresholds and curvature (examples to follow)
  - Linear models automatically extrapolate beyond the range found in data, tree-based models automatically impose a step-function-like structure
- They automate the search for interaction terms
  - Slightly dangerous – can find a lot of spurious interaction terms
- Best for when you don’t care about the exact structure of the model
  - Searching for “reversals”, where the model goes the opposite of the expected direction, is impossible
  - Sheer predictive success in a relatively unregulated market (e.g. click-through rates for prospective customers)

# Why Not Use a GBM?

- You place a premium on model interpretability
  - It is very clear what the parameters of a GLM mean
  - GBMs are hundreds or thousands of trees in combination –harder to just “read the parameters”
  - But we can still get around this by outputting partial dependence plots!
- Need to implement as factor tables
  - “Implement” includes “file with Departments of Insurance”
  - Company’s rating plan is historically based on factor tables, not decision trees
- Need to impose discipline on the interaction structure
  - This is related to the top two bullet points
  - It’s harder to specify a model with mostly simple factors but just one or two interaction terms
- Longer Run-time (hours or days, versus minutes for a GLM)
- Need to tune hyper-parameters (tree depth, number trees, sample rate…)
  - Easy to over-fit if you aren’t careful!

# What Model Support Would Be Most Helpful?

- Partial dependence tables or plots
  - These summarize the marginal effect of each variable on the target
  - For depth-1 models these can be converted to factor tables that *exactly* encode the model
  - No need to provide the trees themselves (Thousands of decision trees? Is this useful for reviewing the model?)
- Observed vs. Modeled ("Univariate") tables or plots
  - Generate modeled losses
  - Compare the average modeled pure premium to the observed pure premium
  - Can see whether the model gives a decent fit (or under-fits or over-fits)
- Deviance/logloss/R-squared?
  - Probably not.
  - These stats are not meaningful without context
  - Some of these are only useful for comparing different models on the exact same dataset.
  - "I just now built an xgboost model with a loglik statistic of 164.997." Is that helpful?
- Variable Importance Plot?
  - Shows how relatively frequently the  model splits on a given variable
  - A formulaic way of answering what's really a subjective question
  - Kind of perfunctory/traditional to include these, but not necessary