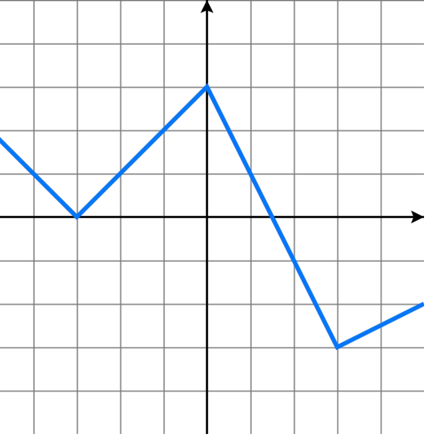
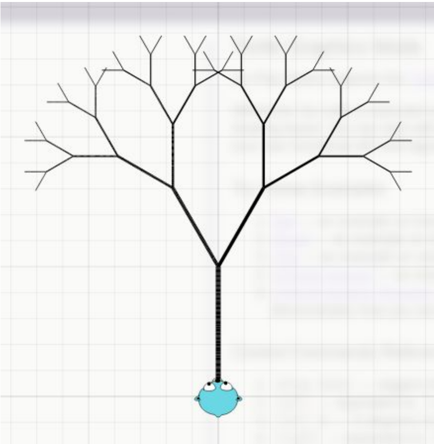


Regression Trees

Saswat Das

ML in Action Talk



Contents

- The CART Algorithm
- Regression Trees
- Model Trees
- Tree-pruning algorithms
- ~~Building a GUI in Python~~

The CART Algorithm

- Short for “Classification and Regression Trees”. (Creative, I know.)
- Helps create trees with respect to some metric, viz. Gini impurity, Shannon entropy.
- For regression trees: uses sum of squared residuals from mean
- Important hyperparameters to be discussed:
    - minimum size of a data partition (tolN)
    - error threshold for a split (tolS)
- Greedy algorithm, can be time consuming to implement
- Binary splits; handles continuous features

Regression Tree Building in a Nutshell

1. Take a dataset.
2. For each feature,
    - Greedily take every possible (binary) split;
    - Calculate the total squared error for each possible split w.r.t. predicted value.
3. Choose the feature with the least sum of squared residuals as the next node with the optimal splitting condition.
4. If a piece of data satisfies the condition, it goes down the left, else right.
5. Take the pertinent subset of the dataset for the next node and repeat steps 2 and 3 on it.
6. Rinse and repeat until a stopping condition is reached.
    - If size of data subset < tolN
    - If error reduction < tolS
7. Terminate with leaf nodes; o/p mean values of the label data of the subset.

Underlying Data Structure

A dictionary for every node.

- Feature to split on
- Value of the feature used to split
- Right Subtree
- Left Subtree

Subtrees can be leaf nodes

Pruning

Taller the tree, higher the variance :/

- Pre-pruning
  - Already done! Done with the training data and via choices of values of tolS and tolN.
- Post-pruning
  - Uses test data
  - Greedily merges splits near leaf nodes and sees if the test error reduces. If yes, keep the merged version.

Pruning prevents overfitting!

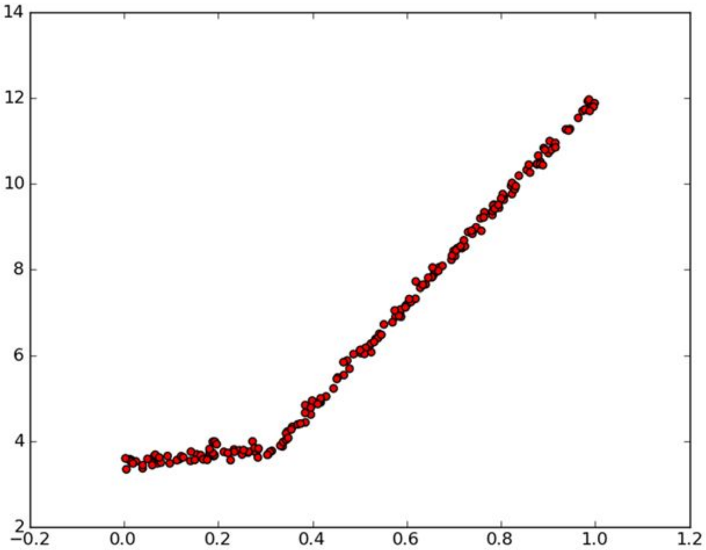
Spoiler: Pre-Pruning>Post-Pruning

Okay but what more can the leaf nodes be?

What if we have a (linear) model at each leaf node?

Voila! Model Tree

Voila! Model Tree



What’s different?

- Has linear models at leaf nodes instead of real values
- Error for splits calculated as total error for training points with respect to linear models at the leaf nodes
- Linear models created using the normal equation,

$$\hat{w} = (X^T X)^{-1} X^T y$$

- Size of data subset at leaf nodes is small. Hence using the normal equation is feasible and much more preferable to gradient descent.

Pros and Cons of Tree Based Regression

- Pros
  - Fits complex, non-linear data
  - Easy to represent and infer predictions quickly from
- Cons
  - Difficult to interpret results
  - Tree building is time consuming
- Works with
  - Numeric values
  - Nominal values

Other Recommended Resources

- "Regression Trees, Clearly Explained!!!" - StatQuest with Josh Starmer (on YouTube)