Machine Learning and Programming in Python

Lecture for Master and PhD students

Chair of Data Science in Economics

Ruhr University Bochum

Summer semester 2024

Lecture 8

Decision Trees

- A **Supervised Learning** approach
- Available for Regression and for Classification

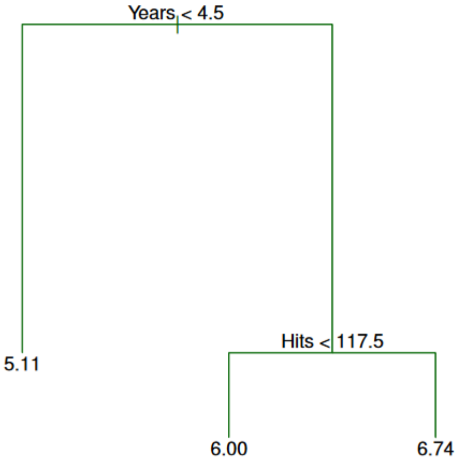
- Decision trees involve stratifying or segmenting the predictor space into a number of simple regions
- Since the set of splitting rules used to segment the predictor space can be summarized in a tree, these types of approaches are known as decision-tree methods

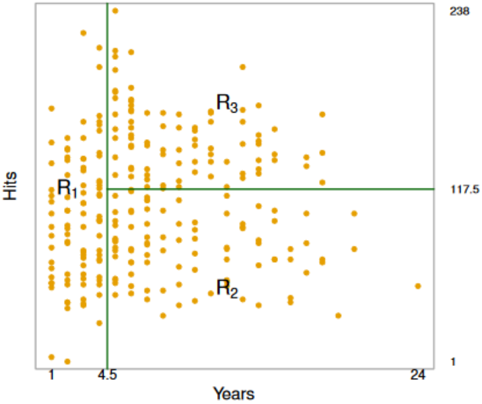
- Tree-based methods are simple and useful for interpretation.
- However they typically are not competitive with the best supervised learning approaches in terms of prediction accuracy.
- Hence we also discuss bagging, random forests, and boosting. These methods grow multiple trees which are then combined to yield a single consensus prediction.
- Combining a large number of trees can often result in dramatic improvements in prediction accuracy, at the expense of some loss interpretation

- Sequential splitting of the predictor space X into more 'homogeneous' regions.
  - Mean or mode of y of training data in a region to predict $\hat{y}$ for a test point in that region
- Decision tree can illustrate splitting rules
  - Internal nodes (predictors/features): where splitting happens
  - Leaf or terminal nodes: final regions
  - Regression or classification
- Implementation: How to split - sequence and splitting points? When to stop?

Regression Trees

- Predict salary of a baseball player based on:
  - Number of years in the major leagues.
  - Number of hits scored in the previous year.
- Build a decision tree.


Years < 4.5
Hits < 117.5
5.11
6.00
6.74


238
117.5
1
R3
R1
R2
Hits
1
4.5
24
Years

- Stratifies into three regions.
  - R1: $years <= 4.5$ (inexperienced).
  - R2: $years >= 4.5$ and $hits <= 117.5$ (experienced but low hit rate recently).
  - R3: $years >= 4.5$ and $hits >= 117.5$ (experienced and high hit rate recently).
- Predicted salaries:
  - R1: $1000 * e^{5.11} \approx 165000$
  - R2: $1000 * e^{6.00} \approx 403000$
  - R3: $1000 * e^{6.74} \approx 845000$
- Experience key to salaries.
  - For inexperienced players, hit rate does not matter much. For experienced players, hit rate is important.
- Easy to interpret and illustrate graphically.

- Split the feature space $X_1, X_2, ..., X_p$ into J distinct exclusive regions $R_1, ..., R_j$
- For every observation x_i in region R_j , the predicted $\hat{y}_i$ is the mean of the y of the training data in R_j

- How to split?
  - Could be any shape but divide into high-dimensional boxes
  - Minimise squared error: $\sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2$
  - $\hat{y}_{R_j}$ is the mean y of the training data in R_j
- **Top-down, greedy** splitting (recursive binary splitting):
  - Start at the tree top and split in two sequentially
  - Greedy because at each step choose the **best** split at that point (instead of a split that may lead to a better tree further down)

1. For each X_j in $X_1, X_2, ..., X_p$
   1. Cutoff s splits the data into two regions: $R_1(j,s) = [X|X_p < s]$ & $R_2(j,s) = [X|X_p \geq s]$
2. Pick j and s that minimises squared error:

$$\sum_{i:x_i \in R_1(j,s)} (y_i - \hat{y}_{R_1})^2 + \sum_{i:x_i \in R_2(j,s)} (y_i - \hat{y}_{R_2})^2$$

3. Repeat the process by scanning across regions, and picking the best region and cutoff for the next split.
4. Stop at a pre-defined criterion e.g. 5 or fewer observations in each region.
5. Predict the test $\hat{y}$ by computing the mean y in each region.

- Top Down, greedy approach usually leads to overfitting.
- Complex trees: Too many small leaves.
  - Intuitively, leaves may differ in y by chance (noise).
  - But the tree has tried to fit this noise.
- Low bias but high variance.
- Poor performance in test data.

- Simpler trees?
- Stop when the decrease in squared error falls below a (high) threshold.
  - Smaller trees but...
  - Possibly bad trees because may stop at a 'poor' split which would have lead to a 'good' split later down the tree.
- Instead, grow a large tree T_0 and prune it back to get a subtree.
- Prune back to subtree that has the lowest test error rate.

- Too many subtrees - how to prune?
- Assign a cost $\alpha >= 0$ to complexity.
- For each α, there is a subtree that $T \subset T_0$ that minimises:

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

- $|T|$ is the number of leaves in T; R_m is the subset corresponding to the mth leaf.
- When $\alpha = 0$, the full tree T_0 is selected.
- For $\alpha > 0$, penalty for having many leaves; smaller subtree is selected.

Algorithm 8.1 *Building a Regression Tree*

1. Use recursive binary splitting to grow a large tree on the training data, stopping only when each terminal node has fewer than some minimum number of observations.

2. Apply cost complexity pruning to the large tree in order to obtain a sequence of best subtrees, as a function of α.

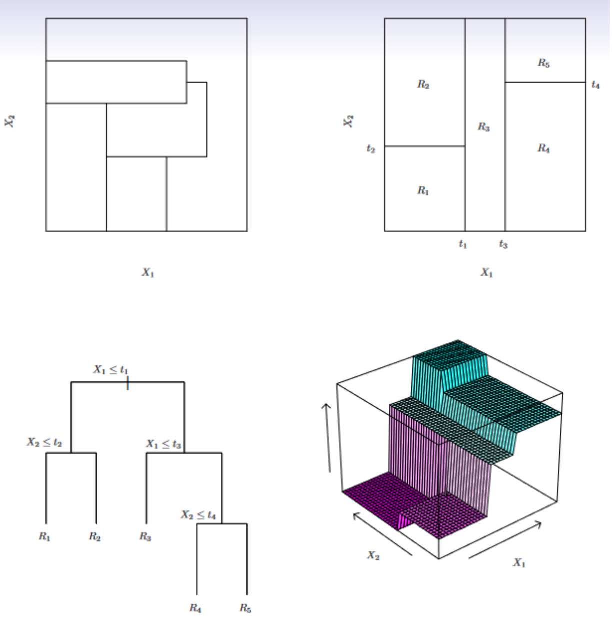
3. Use K-fold cross-validation to choose α. That is, divide the training observations into K folds. For each $k = 1, \ldots, K$:

   (a) Repeat Steps 1 and 2 on all but the kth fold of the training data.

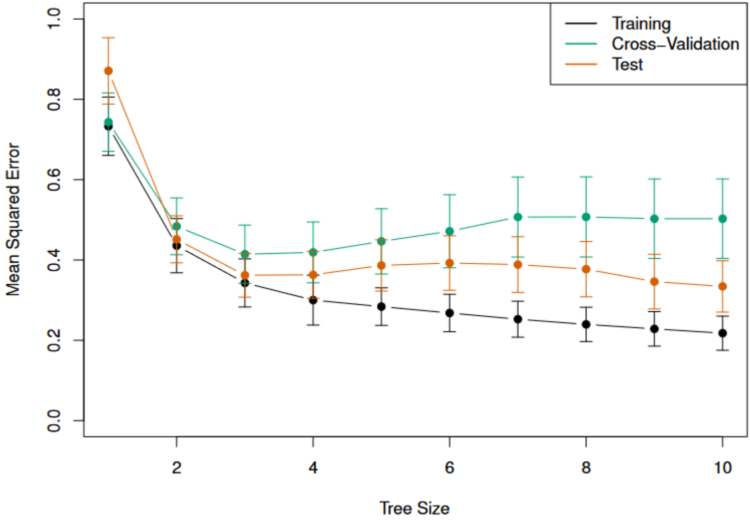
   (b) Evaluate the mean squared prediction error on the data in the left-out kth fold, as a function of α.

   Average the results for each value of α, and pick α to minimize the average error.

4. Return the subtree from Step 2 that corresponds to the chosen value of α.

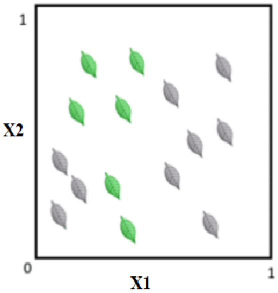


X_2
X_1
R_1
R_2
R_3
R_4
R_5
t_1
t_2
t_3
t_4
$X_1 \leq t_1$
$X_2 \leq t_2$
$X_1 \leq t_3$
$X_2 \leq t_4$

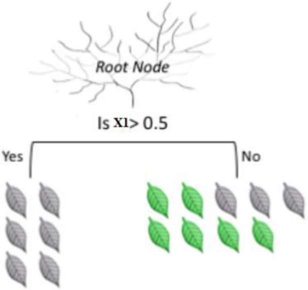

Training
Cross–Validation
Test
Mean Squared Error
0.0
0.2
0.4
0.6
0.8
1.0
2
4
6
8
10
Tree Size

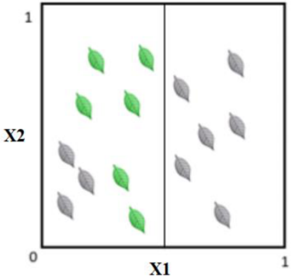
Classification Trees

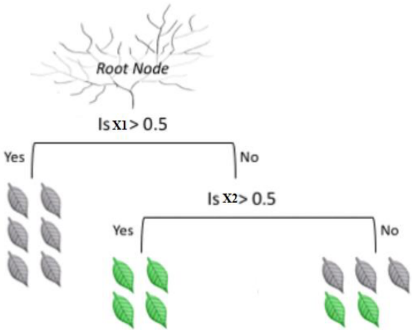
- Classification instead of Regression (qualitative output variable instead of quantitative output variable)
- Prediction in a region is the modal (most frequent) class in a region
- How to do the splitting?
  - Mostly binary splits (a parent node and two child nodes)
  - Aim: Raise purity in the regions

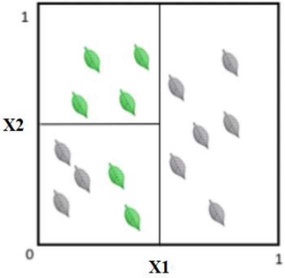


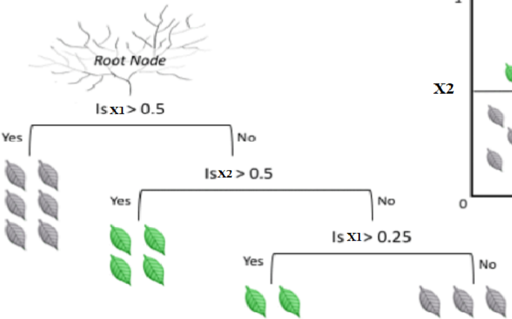
Source: https://towardsdatascience.com/an-exhaustive-guide-to-classification-using-decision-trees-8d472e77223f

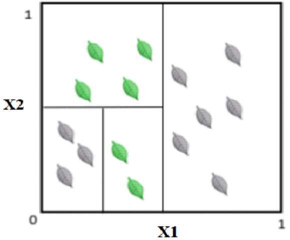

Root Node
Is X1> 0.5
Yes
No


1
X2
0
X1
1


Root Node
Is X1 > 0.5
Yes
No
Is X2 > 0.5
Yes
No


1
X2
0
X1
1


Root Node
Is X1 > 0.5
Yes
No
Is X2 > 0.5
Yes
No
Is X1 > 0.25
Yes
No


1
X2
0
X1
1

Data Project

- Titanic data set
- Fit a decision tree and a random forest to the data
- Visualise the decision tree and the random forest

Literature:

James, Witten, Hastie, Tibshirani, Taylor (2023), An Introduction to Statistical Learning, Springer, Chapter 8, pp. 331-350 .

Hastie, Tibshirani, Friedman (2017), The Elements of Statistical Learning, Springer, Chapter 9, pp. 305-310.