D-Trees – continued

AW

Lecture Overview

Recursive Partitioning Algorithmic Scheme

Estabished facts about D-trees

- If all features have finite domains the class of D-tree hypothethis is finite
- Finding smallest D-tree is $\mathbf{NP}$-complete
  - This means finding smallest ERM D-tree is $\mathbf{NP}$ complete
  - We'll see later that finding any ERM D-tree is $\mathbf{NP}$ complete

We need approximation algorithm $\Rightarrow$ we need purity measure M (it is not neccesarily ER), that is optimized in the algorithm and works well for approximating ERM.

Recursive Partitioning Algorithmic Scheme

So, we designed greedy Recursive Partitioning agorithmic scheme that works with any measure M:

Recursive Partitioning

Input: training set S, feature set $A = \{X_1, \ldots, X_n\}$

1. If all examples are labeles 1 return the leaf labeled 1
2. If all examples are labeled 0 return the leaf labeled 0
3. Compute the impurity measure M of the current set S
4. Find the best split $\langle X_j, v_1, \ldots, v_k \rangle$ among all (feature, values) combinations w.r.t. M
5. If the impurity of the split is no better than the impurity of S, then return the leaf labeled with majority label in S
6. Otherwise aplly Recursive Partioning to each branch (X_j, v_1), (X_j, v_2), $\ldots$, (X_j, v_k)

Recursive Partitioning is Greedy

- Putting aside computinng the best split any recursive partitioning algorithm based on this scheme runs in Poly-time: at every iteration we partition the dataset D into 2 parts, each at least one datapoint smaller than the original data, so the total number of vertexes in decision tree (partitioning steps) is at most $|D|$
- Step 3 of any recursive partitioning based algorithm chooses the split that provides ”greatest separation” of the classes (smallest misclassification error) *after this step*. The latter means that any recursive partitioning based algorithm is ”greedy”:
  - To find ”greatest separation” of classes we must show that after split the quality of leafs is best achievable *in this step*. So we need
    - Purity measure for the current node evaluating how good the node is w.r.t. assigned class
    - A way to evaluate purity of the expansion nodes produced by potential split and to evaluate total purity of the split
    - A way to compute gain of purity obtained by a split
    - Computaton of purity to be ”local”, i.e. without looking ahead and evaluating effects of current split on optimality of final solution
  - Each step must maximize gain of the split

Lecture Overview

From DT Recursive Scheme to Learning Algorithm

To turn the recursive learning idea into algorithm we need to specify:

- How do we choose the next node for splitting?
- What class of trees we allow? i.e. do we allow only binary splits (i.e. pair (feature,value)) or do we allow multiway splits (i.e. tuple (feature,value$_1, \ldots,$ value$_{k-1}$)) for some k as well?
- What is the impurity measure by which we evaluate splits?
- How do we compute the best next split (i.e. (feature,value)) w.r.t. gain defined by purity measure?
  - Fix feature X. How dow we compute the best value for the split?
  - We decided on impurity measure for the node. But how do we compute impurity of the split w.r.t this impurity measure.
- When do we stop splitting?

Only the first question is answered uniformly in all algotihms: the tree is build in the breadth–first manner. In other words, the list of leaf nodes is in fact FIFO queue, i.e. newly created leaf is added at the end and the next node for splitting is the head of the queue.

Things to Specify in The Algorithm

To turn the recursive learning scheme into algorithm we need to specify:

- How do we choose the next node for splitting?
  - The answer: natural appropach to expansion of nodes is Breadth First Search (BFS)

Things to Specify in The Algorithm

To turn the recursive learning scheme into algorithm we need to specify:

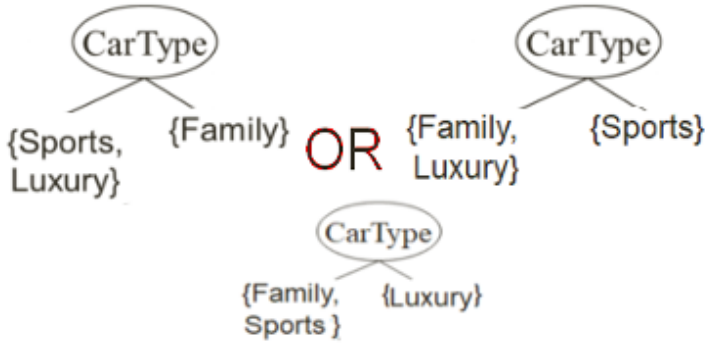
- What kind of splits we allow? i.e. do we allow only binary splits (i.e. pair (feature,value)) or do we allow multiway splits (i.e. tuple (feature,value$_1, \ldots,$ value$_{k-1}$)) for some k as well?
  - The answer: both binary splits and multi-outcome splits are viable decisions but in binary case we need to specify how to deal with nominal fetures while in multi-outcome case we need to specify how to deal with ordinal/continuous features
- What is the impurity measure by which we evaluate splits?
  - The answer: we'll introduce training error Err, gini index $Gini$, entropy H shortly
- How do we compute the best next split (i.e. (feature,value)) w.r.t. gain defined by purity measure?
  - Fix feature X. How dow we compute the best value(s) for the split?
  - How do we compute impurity of the split w.r.t this impurity measure.
- When do we stop splitting?

What kind of splits we allow

The type of split further limits the hypothethis class:

- If only binary splits are allowes (as in the generic scheme) then our class $\mathscr{H}$ is the class of binary trees.

- In this case how do we partition feature domain for values if it is nominal (i.e. no operations on domain values other than comparison for being equal/not equal are allowed)? How do we split these values into 2 groups?



- When we allow multi-outcome splits then the class $\mathscr{H}$ is the class of all finite trees. Each node in a Decision tree predictor can the have its own successor degree (number of children)
  - If we allow multi-outcome splits on nominal features then it is natural to associate each childnode with a domain value. For example if feature ”car” is analysed for a split then 3 values of this feature determine 3 children of current node, each having one value of an attribute
  - If we allow multiway splits on ordinal (=subset of integer numbers) or continuous attributes then the number of splits that need to be compared to find the best split becomes huge/infinite

Lecture Overview

Measures of Node Impurity

Given data set D_t in node t, let $\mathbb{P}_{\mathcal{D}_t}(A)$ be a probability of event A w.r.t. uniform distribution $\mathcal{D}_t$ over D_t. For example $\mathbb{P}_{\mathcal{D}_t}(y = 0)$ is the probability that class assigned to node t is 0.
The following empiric measures of node impurity are used in practice:

- Training error in node t (= Empirical Risk):

$$Err(t) = \min\{\mathbb{P}_{\mathcal{D}_t}(y = 0), 1 - \mathbb{P}_{\mathcal{D}_t}(y = 0)\}$$

  since $\mathbb{P}_{\mathcal{D}_t}(y = 0) = 1 - \mathbb{P}_{\mathcal{D}_t}(y = 1)$
- Gini Index of node t:

$$Gini(t) = 1 - \sum_{j=0}^{1} \left(\mathbb{P}_{\mathcal{D}_t}(y = j)\right)^2$$

  or equivalently $GINI(t) = 2Err(t)\left(1 - Err(t)\right)$
- Entropy of node t:

$$H(t) = -Err(t)\log_2\left(Err(t)\right) - \left(1 - Err(t)\right)\log_2\left(1 - Err(t)\right)$$

Example: Computing Impurity measures

Of course we estimate probabilities by frequences of respective events.
Consider 3 cases of class distribution on 6 data points in a node t:

C_1

$y = 0$	$y = 1$
6	0

C_2

$y = 0$	$y = 1$
2	4

C_3

$y = 0$	$y = 1$
5	1

Example: Computing Impurity measures

Of course we estimate probabilities by frequences of respective events.
Consider 3 cases of class distribution on 6 data points in a node t:

C_1

$y=0$	$y=1$
6	0

C_2

$y=0$	$y=1$
2	4

C_3

$y=0$	$y=1$
5	1

C_1 $Err(t) = \min\{\frac{0}{6}, \frac{6}{6}\} = 0;$

C_2 $Err(t) = \min\{\frac{2}{6}, \frac{4}{6}\} = \frac{1}{3};$

C_3 $Err(t) = \min\{\frac{5}{6}, \frac{1}{6}\} = \frac{1}{6}.$

Example: Computing Impurity measures

Of course we estimate probabilities by frequences of respective events.
Consider 3 cases of class distribution on 6 data points in a node t:

C_1

$y=0$	$y=1$
6	0

C_2

$y=0$	$y=1$
2	4

C_3

$y=0$	$y=1$
5	1

C_1 $Err(t) = \min\{\frac{0}{6}, \frac{6}{6}\} = 0;$

C_2 $Err(t) = \min\{\frac{2}{6}, \frac{4}{6}\} = \frac{1}{3};$

C_3 $Err(t) = \min\{\frac{5}{6}, \frac{1}{6}\} = \frac{1}{6}.$

C_1 $Gini(t) = 1 - 0^2 - 1^2 = 0;$

C_2 $Gini(t) = 1 - \left(\frac{1}{3}\right)^2 - \left(\frac{2}{3}\right)^2 = \frac{4}{9};$

C_3 $Gini(t) = 1 - \left(\frac{5}{6}\right)^2 - \left(\frac{1}{6}\right)^2 = \frac{10}{36}.$

Example: Computing Impurity measures

Of course we estimate probabilities by frequences of respective events.
Consider 3 cases of class distribution on 6 data points in a node t:

C_1

$y=0$	$y=1$
6	0

C_2

$y=0$	$y=1$
2	4

C_3

$y=0$	$y=1$
5	1

C_1 $Err(t) = \min\{\frac{0}{6}, \frac{6}{6}\} = 0;$

C_2 $Err(t) = \min\{\frac{2}{6}, \frac{4}{6}\} = \frac{1}{3};$

C_3 $Err(t) = \min\{\frac{5}{6}, \frac{1}{6}\} = \frac{1}{6}.$

C_1 $Gini(t) = 1 - 0^2 - 1^2 = 0;$

C_2 $Gini(t) = 1 - \left(\frac{1}{3}\right)^2 - \left(\frac{2}{3}\right)^2 = \frac{4}{9};$

C_3 $Gini(t) = 1 - \left(\frac{5}{6}\right)^2 - \left(\frac{1}{6}\right)^2 = \frac{10}{36}.$

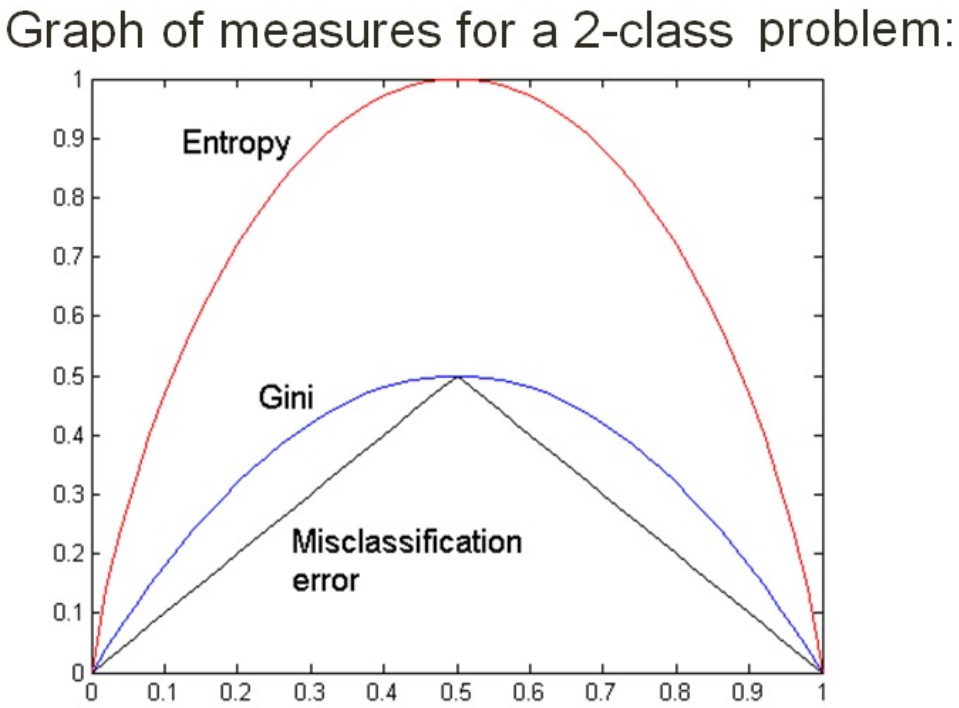
C_1 $H(t) = -0\log_2 0 - 1\log_2 1 = 0$

C_2 $H(t) = -\left(\frac{1}{3}\right)\log_2\left(\frac{1}{3}\right) - \left(\frac{2}{3}\right)\log_2\left(\frac{2}{3}\right) \approx 0.918;$

C_3 $H(t) = -\left(\frac{5}{6}\right)\log_2\left(\frac{5}{6}\right) - \left(\frac{1}{6}\right)\log_2\left(\frac{1}{6}\right) \approx 0.650$

Relationship between Measures

Ther reason why entropy and gini work well in practical implementations of Decision Tree learning is because both measures are smooth upper bounds of misclassification error.



While Err may be skewed because of the small size of training sample, smooth upper bounds are less affected by noise and outliers in small training sample.

Lecture Overview

Measures of a Split and Gain

Recall that $\mathbb{P}_{\mathcal{D}_t}(A)$ be a probability of event A w.r.t. uniform distribution over $\mathcal{D}_t$.
A k-way split partition D_t into k parts $D_{t_1}, \ldots, D_{t_k}$. These subsets are data sets associated with children of t generated by a split, so probability of child node t_i is $\mathbb{P}_{\mathcal{D}_t}(t_i)$ and we estimate it by frequency $\frac{|D_{t_i}|}{|D_t|}$.

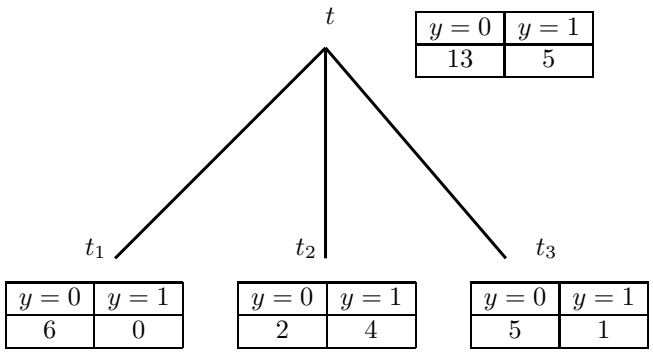
Each measure M is itself a random variable on the corresponding data set. Then we can define an M-vaue of a split as an expected M-value for a child of t produced by a split, i.e.

$$M(X, v_1, \ldots, v_k) \stackrel{\triangle}{=} E(M_{t_{split}}) = \sum_{i=1}^{k} \mathbb{P}_{\mathcal{D}_t}(t_i) \cdot M(t_i)$$

Then the gain of the split can be computed as a measure of a parent minus expected value of its child, i.e.

$$Gain(t, \langle X, v_1, \ldots, v_k \rangle) = M(t) - M(X, v_1, \ldots, v_k)$$

Example of Gain Computation



t: $Err(t) = \min\{\frac{13}{18}, \frac{5}{18}\} = \frac{5}{18}$

t_1: $Err(t_1) = \min\{\frac{0}{6}, \frac{6}{6}\} = 0$

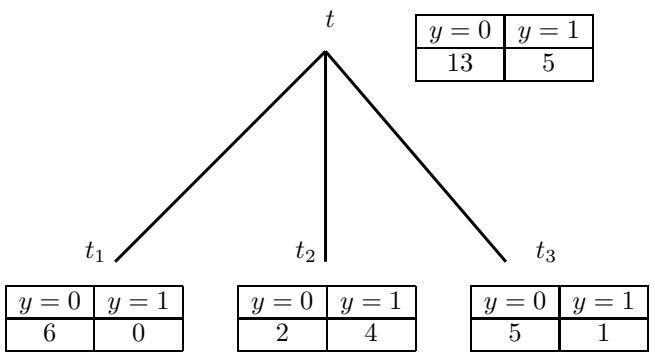
t_2: $Err(t_2) = \min\{\frac{2}{6}, \frac{4}{6}\} = \frac{1}{3}$

t_3: $Err(t_3) = \min\{\frac{5}{6}, \frac{1}{6}\} = \frac{1}{6}$

Split: $Err(split) = \frac{6}{18} \cdot 0 + \frac{6}{18} \cdot \frac{1}{3} + \frac{6}{18} \cdot \frac{1}{6} = \frac{1}{6}$

Gain: $Gain_{Err}(t, \langle t_1, t_2, t_3 \rangle) = \frac{5}{18} - \frac{1}{6} = \frac{1}{9}$

Example of Gain Computation



$$Gain_{Err}(t, \langle t_1, t_2, t_3 \rangle) = \frac{5}{18} - \frac{1}{6} = \frac{1}{9}$$

t: $Gini(t) = 1 - \left(\frac{13}{18}\right)^2 - \left(\frac{5}{18}\right)^2 = \frac{65}{162}$

t_1: $Gini(t_1) = 1 - 0^2 - 1^2 = 0$

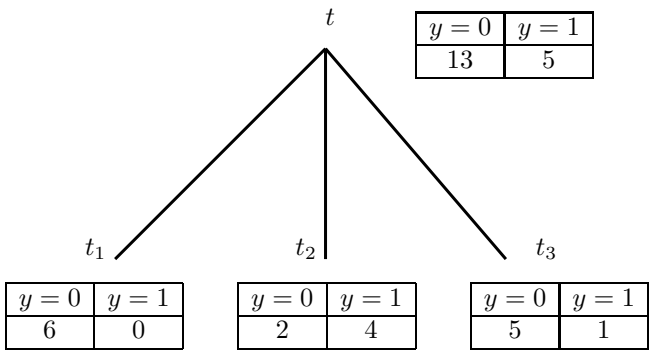
t_2: $Gini(t_2) = 1 - \left(\frac{1}{3}\right)^2 - \left(\frac{2}{3}\right)^2 = \frac{4}{9}$

t_3 $Gini(t_3) = 1 - \left(\frac{5}{6}\right)^2 - \left(\frac{1}{6}\right)^2 = \frac{10}{36}$

Split: $Gini(split) = \frac{6}{18} \cdot 0 + \frac{6}{18} \cdot \frac{4}{9} + \frac{6}{18} \cdot \frac{10}{36} = \frac{13}{54}$

Gain: $Gain_{Gini}(t, \langle t_1, t_2, t_3 \rangle) = \frac{65}{162} - \frac{13}{54} = \frac{13}{81}$

Example of Gain Computation



$$Gain_{Err}(t, \langle t_1, t_2, t_3 \rangle) = \frac{5}{18} - \frac{1}{6} = \frac{1}{9}$$

$$Gain_{Gini}(t, \langle t_1, t_2, t_3 \rangle) = \frac{65}{162} - \frac{13}{54} = \frac{13}{81}$$

t: $H(t) = -\left(\frac{13}{18}\right)\log_2\left(\frac{13}{18}\right) - \left(\frac{5}{18}\right)\log_2\left(\frac{5}{18}\right) = 0.8524$

t_1: $H(t_1) = 1 - 0^2 - 1^2 = 0$

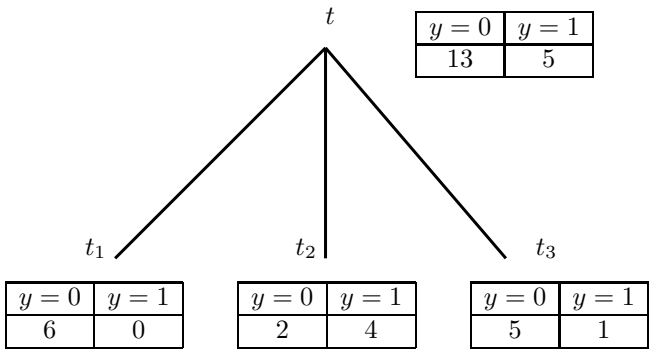
t_2: $H(t_2) = -\left(\frac{1}{3}\right)\log_2\left(\frac{1}{3}\right) - \left(\frac{2}{3}\right)\log_2\left(\frac{2}{3}\right) \approx 0.918$

t_3: $H(t_3) = -\left(\frac{5}{6}\right)\log_2\left(\frac{5}{6}\right) - \left(\frac{1}{6}\right)\log_2\left(\frac{1}{6}\right) \approx 0.650$

Split: $H(split) = \frac{6}{18} \cdot 0 + \frac{6}{18} \cdot o.918 + \frac{6}{18} \cdot 0.650 \approx 0.523$

Gain: $Gain_H(t, \langle t_1, t_2, t_3 \rangle) = 0.852 - 0.523 = 0.329$

Example of Gain Computation



$$Gain_{Err}(t, \langle t_1, t_2, t_3 \rangle) = \frac{5}{18} - \frac{1}{6} = \frac{1}{9} \approx 0.1111$$

$$Gain_{Gini}(t, \langle t_1, t_2, t_3 \rangle) = \frac{65}{162} - \frac{13}{54} = \frac{13}{81} \approx 0.161$$

$$Gain_{H}(t, \langle t_1, t_2, t_3 \rangle) = 0.852 - 0.523 = 0.329$$

Lecture Overview

Split Value for a Feature

Fix a feature X, what is (are) the best value(s) for a split over a feature X? The answer depends on whether all spiits are binary or we allow multi-branch splits.
In binary (balanced) D-tree

- At a split of node t one feature's domain is partitioned into 2 sub-domains that determine how data set D_t is split into data sets D_{t_L} and D_{t_R} assosiated with children t_L and t_R
- One feature can be re-used in several splits, but each time its range is narrowed
- In the resulting decision tree each node is either internal (and has either 2 children) or is a leaf (and has no children)

Binary D-tree – Split Value for Nominal Feature

- To find best split at a node t for a nominal (aka categorical) feature X that has k values $c_1, \ldots, c_k$ we must compare measure values for all splits iduced by possible subsets $S \subset \{1, \ldots, k\}, \quad S \neq \emptyset$. A split (X, C_S) then is a defined by membership of feature X value v at a datapoint x in $C_S = \bigcup_{i \in S} c_i$.
- This exact computation of the best split requires $2^k - 3$ measure computations (one for each C_S, i.e. brute-force enumeration and subsequent comparison of all possible splits). This approach may not be viable if k is big.
- When complete feature enumeration is not viable we can use another algorithm
  1. Compute $r = \arg\min_{r \in \{0,1\}} \mathbb{P}_{D_t}(y = r)$
  2. Make two copies C_r of a set $C = \{c_1, \ldots, c_k\}$, initialize $S_r = D_r = \emptyset$
  3. Order values in C_r in nonincreasing order w.r.t. class probabilities $\mathbb{P}_{D_t}(y = r \mid v = c_j) = \frac{n_{y=r, v=c_j}}{n_{v=c_j}}$ to get the list $\bar{L}_r$. Here probabilities are computed w.r.t. current value of D_t.
  4. Truncate $\bar{L}_r$ to length l to get list $\bar{L}_r|_l$. Note that l is an algorithm's parameter

Binary D-tree – Split Value for Nominal Feature

- When complete feature enumeration is not viable we can use another algorithm
  1. Compute $r = \arg\min_{r \in \{0,1\}} \mathbb{P}_{D_t}(y = r)$
  2. Make two copies C_r of a set $C = \{c_1, \ldots, c_k\}$, initialize $S_r = D_r = \emptyset$
  3. Order values in C_r in nonincreasing order w.r.t. class probabilities $\mathbb{P}_{D_t}(y = r \mid v = c_j) = \frac{n_{y=r, v=c_j}}{n_{v=c_j}}$ to get the list $\overline{L}_r$. Here probabilities are computed w.r.t. current value of D_t.
  4. Truncate $\overline{L}_r$ to length l to get list $\overline{L}_r|_l$. Note that l is an algorithm's parameter
  5. Compute $c_j = \arg\max_{i \in \overline{L}_r|_l} Gain(t, \langle X, S_r \cup \{c_i\}\rangle)$ & $\frac{n_{y=r, v=c_j}}{n_{v=c_j}} \geq 0.5$. Add the data points with $X = c_j$ to D_r removing them from D_t.
  6. Record computed gain. Update $S_r = S_r \cup \{c_j\}$ and $C_r = C_r - \{c_j\}$
  7. Repeat steps 3 - 6 until $C_r = \emptyset$
  8. Choose split that produced highest gain in the previous steps

The algorithm is quadratic in k. It is based on Bayesian approach and it is optimal for Information and Gini. Unfortunately it is suboptimal for Err because it is not smooth

Example of Chooseing Categorical Split - One step

Assume that split measure is Gini, $l = 3$. The car data is

$$D = \begin{array}{c} \\ \text{sports} \\ \text{family} \\ \text{truck} \\ \text{luxury} \end{array} \begin{array}{c} \begin{array}{cc} y=0 & y=1 \end{array} \\ \begin{pmatrix} 40 & 160 \\ 150 & 50 \\ 60 & 40 \\ 50 & 150 \end{pmatrix} \end{array}$$

The probability of classes are $\mathbb{P}(y=0) = \frac{300}{700} = \frac{3}{7}$ and $\mathbb{P}(y=1) = \frac{4}{7}$, so $r = 0$. The list of values ordered according to probabilirites is

$\mathbb{P}(v = c_j | y = 0) = \frac{n_{y=0, v=c_j}}{n_{v=c_j}}$ is $\begin{array}{cccc} \text{family} & \text{truck} & \text{luxury} & \text{sports} \\ \begin{pmatrix} 0.5 & 0.2 & 0.16(6) & 0.06(6) \end{pmatrix} \end{array}$.

The truncated list is (family, truck, luxury).

Example of Chooseing Categorical Split - One step

Assume that split measure is Gini, $l = 3$. The car data is

$$D = \begin{array}{l} \\ \text{sports} \\ \text{family} \\ \text{truck} \\ \text{luxury} \end{array} \begin{array}{c} \begin{array}{cc} y=0 & y=1 \end{array} \\ \begin{pmatrix} 40 & 160 \\ 150 & 50 \\ 60 & 40 \\ 50 & 150 \end{pmatrix} \end{array}$$

The truncated list is (family, truck, luxury).
Gini of the original node is $1 - (3/7)^2 - (4/7)^2 \approx 0.49$. Gini of each split is respectively:

for fam $\frac{2}{7}\left(1 - (3/4)^2 - (1/4)^2\right) + \frac{5}{7}\left(1 - (7/10)^2 - (3/10)^2\right) \approx 0.407$
for fam+tr $\frac{3}{7}\left(1 - (7/10)^2 - (3/10)^2\right) + \frac{3}{10}\left(1 - (9/31)^2 - (22/31)^2\right) \approx 0.303$
for fam+tr+lux $\frac{5}{7}\left(1 - (13/25)^2 - (12/25)^2\right) + \frac{2}{7}\left(1 - (3/4)^2 - (1/4)^2\right) \approx 0.464$

Maximum gain then obtained on family+truck which becomes the choisen split.

Example of Chooseing Categorical Split - One step

Well working in practice heuristic algorithm is based on PCA scoring of values.
I am not covering it. Interested students can read in
Coppersmith, D., S. J. Hong, and J. R. M. Hosking. ”Partitioning Nominal Attributes in Decision Trees.” Data Mining and Knowledge Discovery, Vol. 3, 1999, pp. 197217