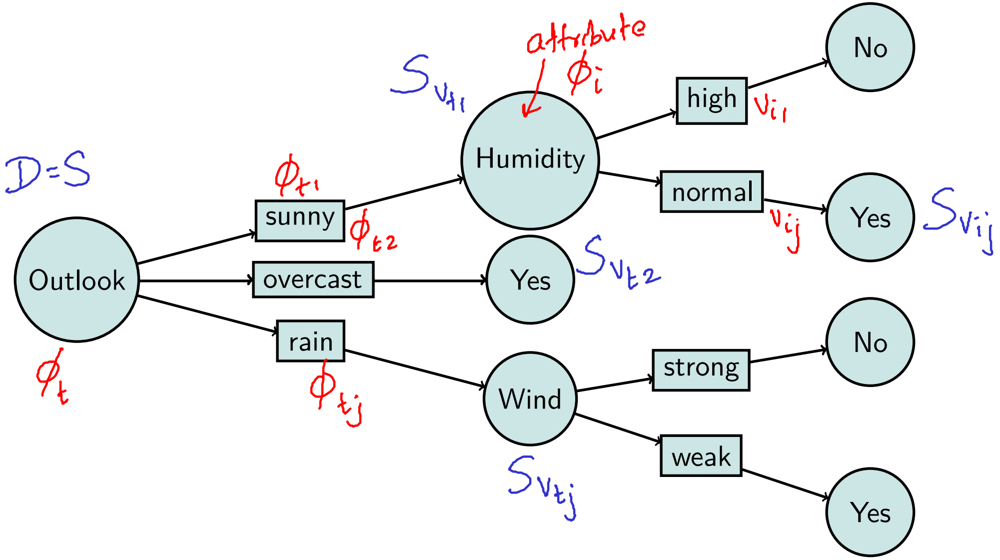
Lecture 24: Other (Non-linear) Classifiers: Decision Tree Learning, Boosting, and Support Vector Classification

Instructor: Prof. Ganesh Ramakrishnan

Decision Trees: Cascade of step functions on individual features



The Canonical Playtennis Dataset

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Decision tree representation

- Each internal node tests an attribute
- Each branch corresponds to attribute value
- Each leaf node assigns a classification

How would we represent:

- $\wedge, \vee$, XOR
- $(A \wedge B) \vee (C \wedge \neg D \wedge E)$
- M of N

How to learn

Top-Down Induction of Decision Trees (Greedy algo)

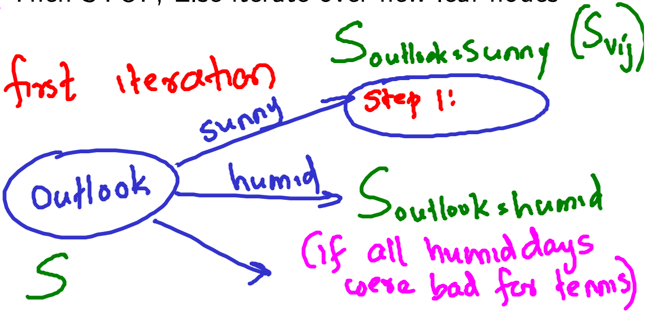
Main loop:

1. $\phi_i \leftarrow$ the "best" decision attribute for next *node*
2. Assign ϕ_i as decision attribute for *node*
3. For each value of ϕ_i, create new descendant of *node*
4. Sort training examples to leaf nodes
5. If training examples perfectly classified, Then STOP, Else iterate over new leaf nodes

Which attribute is best?

Eg: ϕ_i = Outlook in first iteration

$V_i = \{\text{Sunny}, \text{Humid}, .\}$



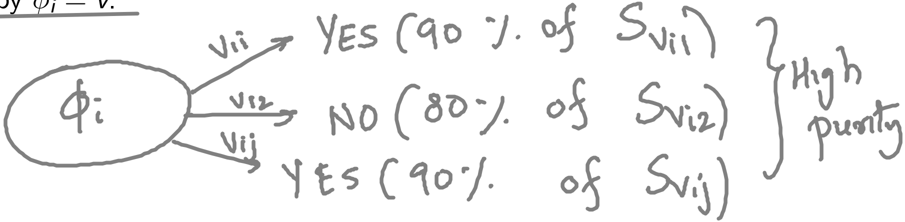
Top-Down Induction of Decision Trees

Main loop:

1. $\phi_i \leftarrow$ the "best" decision attribute for next *node*
2. Assign ϕ_i as decision attribute for *node*
3. For each value of ϕ_i, create new descendant of *node*
4. Sort training examples to leaf nodes
5. If training examples perfectly classified, Then STOP, Else iterate over new leaf nodes

Which attribute is best?

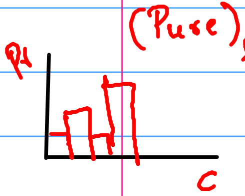
Answer: That which brings about maximum reduction in impurity $\mathbf{Imp}(S_v)$ of the data subset $S_v \subseteq \mathcal{D}$ induced by $\phi_i = v$.

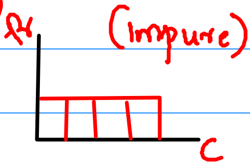


Measures of impurity

(1) Ratio of incorrectly classified (for each class) normalized by size of split

(2) Entropy : $-\sum_{c_i} P_{\phi_j}(c_i) \log P_{\phi_j}(c_i)$





$P_{\phi_j}(c_i)$ = Prob of c_i as view based on splits of S base on ϕ_j

Top-Down Induction of Decision Trees

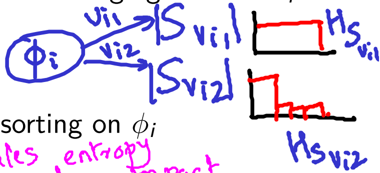
Main loop:

1. $\phi_i \leftarrow$ the "best" decision attribute for next *node*
2. Assign ϕ_i as decision attribute for *node*
3. For each value of ϕ_i, create new descendant of *node*
4. Sort training examples to leaf nodes
5. If training examples perfectly classified, Then STOP, Else iterate over new leaf nodes

Which attribute is best?

Answer: That which brings about maximum reduction in impurity $\mathbf{Imp}(S_v)$ of the data subset $S_v \subseteq \mathcal{D}$ induced by $\phi_i = v$.

- S is a sample of training examples, p_{C_i} is proportion of examples belonging to class C_i in S
- Entropy measures impurity of S: $H(S) \equiv \sum_{i=1}^{K} -p_{C_i} \log_2 p_{C_i}$
- $Gain(S, \phi_i)$ = expected reduction in entropy due to splitting/sorting on ϕ_i
  $Gain(S, \phi_i) \equiv H(S) \; - \sum_{v \in Values(\phi_i)} \frac{|S_v|}{|S|} H(S_v)$



$|S_v| \rightarrow$ scales entropy based on impact

Common Impurity Measures (Tutorial 9)

Impurity before split using ϕ_i

$$\phi_s = \arg \max_{V(\phi_i),\phi_i} \left(Imp(S) - \underbrace{\sum_{v_{ij} \in V(\phi_i)} \frac{|S_{v_{ij}}|}{|S|} Imp(S_{v_{ij}})}_{} \right)$$

Average impurity after split using ϕ_i

where $S_{ij} \subseteq \mathcal{D}$ is a subset of dataset such that each instance x has attribute value $\phi_i(x) = v_{ij}$.

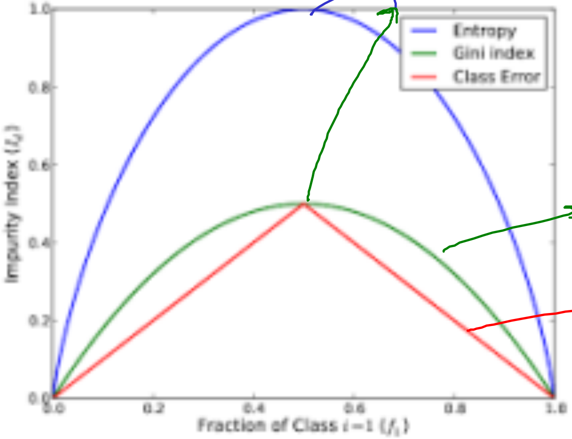
Name	$Imp(S)$
Entropy	$-\sum_{i=1}^{K} Pr(C_i) \bullet \log(Pr(C_i))$
Gini Index	$\sum_{i=1}^{K} Pr(C_i)(1 - Pr(C_i))$
Class (Min Prob) Error	$\underset{i}{\operatorname{argmin}}(1 - Pr(C_i))$

Table: Decision Tree: Impurity measurues

These measure the extent of spread /confusion of the probabilities over the classes

Alternative impurity measures (Tutorial 9)

peak of impurity at uniform distribution.



$p_{c_i}(1-p_{c_i})$

if C = true class
$1-p(C)$

Figure: Plot of Entropy, Gini Index and Misclassification Accuracy. Source:
`https://inspirehep.net/record/1225852/files/TPZ_Figures_impurity.png`

Regularization in Decision Tree Learning → Get simple model (penalize complex models) for generalizability.

- Premise: Split data into *train* and *validation* set[1]

[1] *Note:* The *test set* still remains separate
[2] Like we discussed in the case of Convolutional Neural Networks
[3] Prefer the shortest hypothesis that fits the data

Regularization in Decision Tree Learning

- Premise: Split data into *train* and *validation* set[1]
- Structural Regularization[2] based on Occam's razor[3]
  1. stop growing when data split not statistically significant
     - ★ Use parametric/non-parametric hypothesis tests

eg: χ^2

Few data pts with almost uniform distr

2 C_1
1 C_2
1 C_3

[1] *Note:* The *test set* still remains separate

[2] Like we discussed in the case of Convolutional Neural Networks

[3] Prefer the shortest hypothesis that fits the data

Regularization in Decision Tree Learning

- Premise: Split data into *train* and *validation* set[1]
- Structural Regularization[2] based on Occam's razor[3]
  1. stop growing when data split not statistically significant
     - ★ Use parametric/non-parametric hypothesis tests
  2. grow full tree, then post-prune tree
     - ★ *Minimum Description Length* (MDL): minimize $size(tree) + size(misclassifications_{val}(tree))$
     - ★ Achieved as follows: Do until further pruning is harmful

       (1) Evaluate impact on *validation* set of pruning each possible node (plus those below it)

       (2) Greedily remove the one that most improves *validation* set accuracy

[1] *Note:* The *test set* still remains separate

[2] Like we discussed in the case of Convolutional Neural Networks

[3] Prefer the shortest hypothesis that fits the data

Regularization in Decision Tree Learning

- Premise: Split data into *train* and *validation* set[1]
- Structural Regularization[2] based on Occam's razor[3]
  1. stop growing when data split not statistically significant
     - ★ Use parametric/non-parametric hypothesis tests
  2. grow full tree, then post-prune tree
     - ★ *Minimum Description Length* (MDL): minimize $size(tree) + size(misclassifications_{val}(tree))$
     - ★ Achieved as follows: Do until further pruning is harmful

       (1) Evaluate impact on *validation* set of pruning each possible node (plus those below it)

       (2) Greedily remove the one that most improves *validation* set accuracy
  3. convert tree into a set of rules and post-prune each rule independently (C4.5 Decision Tree Learner)

(if outlook=sunny) & (temp = high) - - --

then play

[1] *Note:* The *test set* still remains separate

[2] Like we discussed in the case of Convolutional Neural Networks

[3] Prefer the shortest hypothesis that fits the data

General Minimum Description Length

Eg: unpruned decision tree (or partially pruned)

Eg: Error

- Data is D and theory about the data is T.
- MDL principle: Define $I(D|T)$ and $I(T)$ and choose T such that it minimizes $I(D|T) + I(T)$. size of tree
- Also aligned with the *Occam Razor* principle.
- *Bayes Estimation*: $I(D|T) = \log P(D|T)$ and $I(T) = \log P(T)$