18-661: Introduction to ML for Engineers

Decision Trees, Ensemble Methods, and Neural Networks

Spring 2025

ECE – Carnegie Mellon University

Decision Tree Motivation

Motivation:

Simple yet powerful supervised learning method.

Easy to interpret, visualize, and explain decisions.

Handles both categorical and numerical data naturally.

Decision Tree Concept:

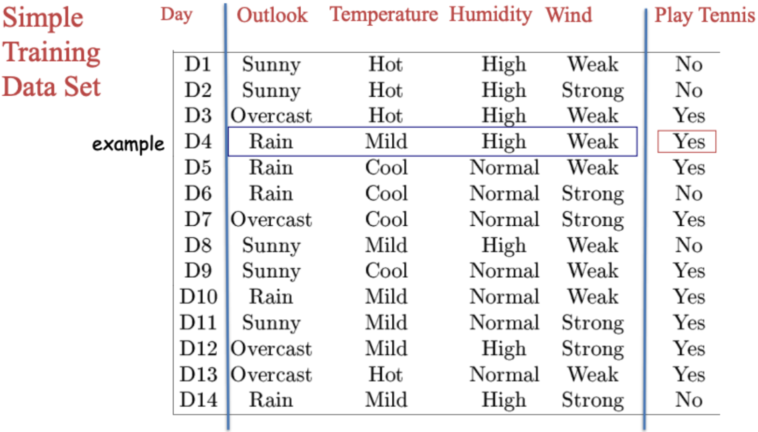
Root node: Represents the entire dataset.

Internal nodes: Represent test conditions on attributes.

Branches: Outcomes of test conditions.

Leaf nodes: Final decisions or class labels.

Decision Tree Example



Simple Training Data Set

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
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

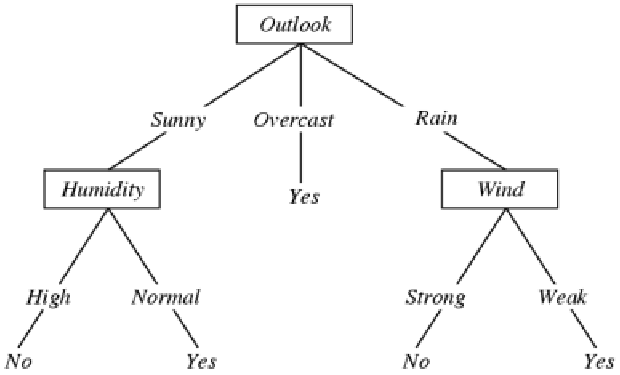
example

label

* Example taken from 10-701

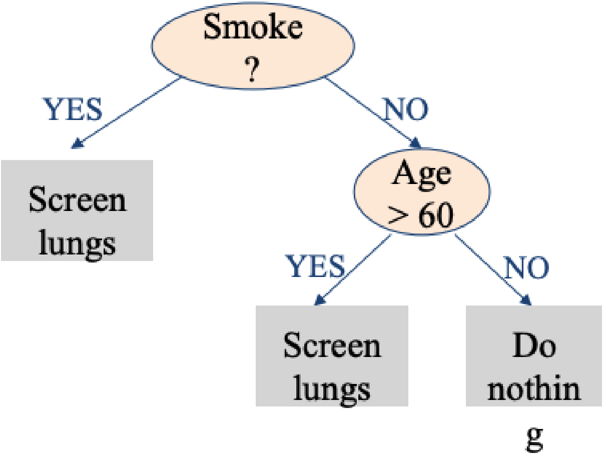
Decision Tree Example

Learned decision tree -



Decision Tree Numerical Features

Decision Trees can handle numerical (continuous) features as well:



Decision Tree T/F

Decision Trees can be used for both classification and regression problems

A) True

B) False

Decision Tree T/F

Decision Trees can be used for both classification and regression problems

A) True

B) False

Solution: True Can average the value of all the training examples that fall into each leaf node

Decision Tree Splitting Criterion

Entropy Definition -

Binary Data - $H(S) = -p_+ \log_2(p_+) - p_- \log_2(p_-)$

Non-Boolean Data - $H(S) = \sum -p_i \log_2(p_i)$ *Remains base 2

Information Gain Definition -

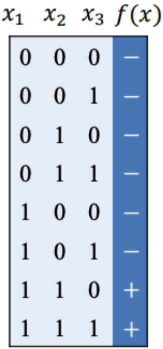
Gain$(S, A) =$ Entropy(S) - Entropy$(S|A) =$
Entropy$(S) - \sum\limits_{v \in values(A)} \frac{S_v}{S}$ Entropy(S_v)

*Intuition - want to determine how effective a particular attribute is at classifying the training examples. Higher information gain indicates a greater reduction in uncertainty, making the attribute more useful for splitting.

Decision Tree Splitting on Error Rate

Why not just split on the error rate?

$$f(x) = x_1 \wedge x_2$$

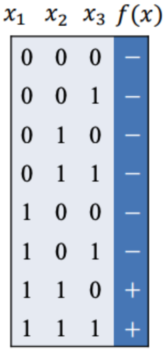


x_1	x_2	x_3	$f(x)$
0	0	0	−
0	0	1	−
0	1	0	−
0	1	1	−
1	0	0	−
1	0	1	−
1	1	0	+
1	1	1	+

Decision Tree Splitting on Error Rate

Let's split on information gain instead

$$f(x) = x_1 \wedge x_2$$

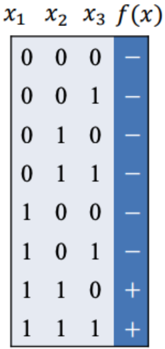


x_1	x_2	x_3	$f(x)$
0	0	0	−
0	0	1	−
0	1	0	−
0	1	1	−
1	0	0	−
1	0	1	−
1	1	0	+
1	1	1	+

Decision Tree Splitting on Error Rate

Let's split on information gain instead

$$f(x) = x_1 \wedge x_2$$



x_1	x_2	x_3	$f(x)$
0	0	0	−
0	0	1	−
0	1	0	−
0	1	1	−
1	0	0	−
1	0	1	−
1	1	0	+
1	1	1	+

Decision Tree Question

Draw a decision tree that correctly classifies the function $f(x) = x_1 \wedge x_2$ and another decision tree that correctly classifies the function $f(x) = x_1 \vee x_2$

Decision Tree Question

Consider a binary decision problem with the following validation dataset -

x_1	x_2	x_3	x_4	y
0	0	1	1	0
1	0	1	1	1
0	1	0	1	1
0	1	0	0	0
0	0	1	0	0
1	1	0	0	1
1	1	0	0	0
1	0	0	0	0

Can a decision tree correctly classify this entire validation set? If yes, draw the tree

Decision Tree Overfitting

Decision Trees are prone to overfitting (can become very complicated with a few layers of depth)

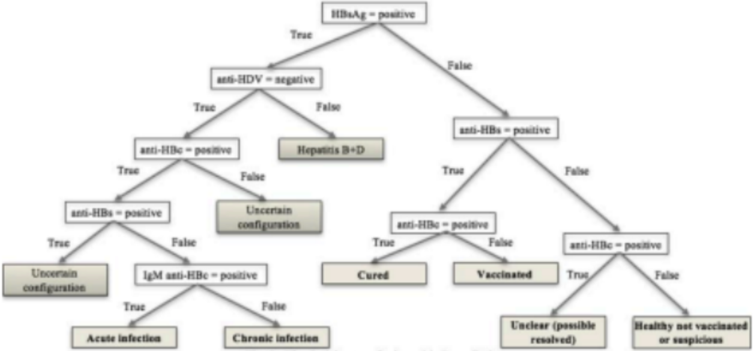


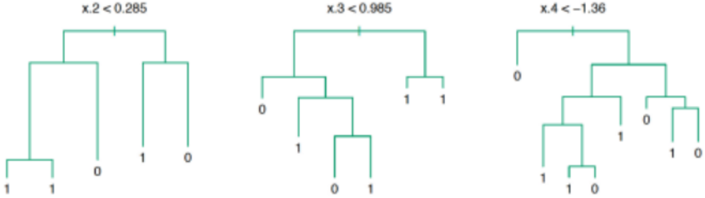
Fig. 2. The decision tree for hepatitis B predictions

https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7995362

Ensemble Methods Motivation

Single decision trees are either shallow (high bias, low variance) or deep (low bias, high variance)

Could try to combine multiple decision trees to try and achieve the best of both worlds



Bagging

Bagging incorporates the decision from multiple trees in an effort to prevent overfitting

- ◦ Learn B decision trees from subsets of the training data
- ◦ For test examples, aggregate the predictions from each tree

Random Forests

Random Forests also incorporates multiple trees in the prediction but each tree uses only a subset of the features before each split

- If the splits across the decision trees are very similar, redundancy makes having multiple trees less useful
- Adding randomness to the splits makes the trees less correlated

Random Forests MCQ

Which of the following statements about Random Forests is not true?

A) Random Forests are an ensemble learning method that are used for both classification and regression problems

B) Random Forests operates by constructing a multitude of decision trees at test time

C) For binary classification problems, random forests outputs the mode of the predictions of all the trees

D) For regressions problems, random forests outputs the mean, or weighted mean, of the predictions of all the trees

Random Forests MCQ

Which of the following statements about Random Forests is not true?

A) Random Forests are an ensemble learning method that are used for both classification and regression problems

B) Random Forests operates by constructing a multitude of decision trees at test time

C) For binary classification problems, random forests outputs the mode of the predictions of all the trees

D) For regressions problems, random forests outputs the mean, or weighted mean, of the predictions of all the trees

Solution: B Random Forests operates constructs a multitude of decision trees at **training** time

Boosting

Boosting incorporates the decision from multiple **weak learners** in an effort to prevent improve performance

- ◦ Weak learners are simple decision functions (e.g. decision stumps) that tend not to be very accurate on their own
- ◦ Using multiple weak learners together can make the prediction more accurate

Boosting vs. Bagging MCQ

Which of the following statements is correct regarding the fundamental differences between bagging and boosting?

A) In bagging, multiple base learners are trained in parallel on different bootstrap samples, whereas in boosting, base learners are added sequentially, each focusing on the errors of the previous learner.

B) In bagging, the primary focus is on reducing the bias of the model, whereas boosting is designed primarily to reduce variance.

C) In bagging, the final prediction is determined by selecting the best single base learner, whereas in boosting, the final prediction is typically the average of all learners.

D) Bagging is inherently more prone to overfitting than boosting because it trains multiple models on overlapping subsets of the data.

Boosting vs. Bagging MCQ

Which of the following statements is correct regarding the fundamental differences between bagging and boosting?

A) In bagging, multiple base learners are trained in parallel on different bootstrap samples, whereas in boosting, base learners are added sequentially, each focusing on the errors of the previous learner.

B) In bagging, the primary focus is on reducing the bias of the model, whereas boosting is designed primarily to reduce variance.

C) In bagging, the final prediction is determined by selecting the best single base learner, whereas in boosting, the final prediction is typically the average of all learners.

D) Bagging is inherently more prone to overfitting than boosting because it trains multiple models on overlapping subsets of the data.

Solution: A

Adaboost

- Given: N samples $\{\boldsymbol{x}_n, y_n\}$, where $y_n \in \{+1, -1\}$, and some way of constructing weak (or base) classifiers
- Initialize weights $w_1(n) = \frac{1}{N}$ for every training sample n
- For $t = 1$ to T
  1. Train a weak classifier $h_t(\boldsymbol{x})$ using weights $w_t(n)$, by minimizing
     $$\epsilon_t = \sum_n w_t(n)\mathbb{I}[y_n \neq h_t(\boldsymbol{x}_n)] \quad \text{(the weighted classification error)}$$
  2. Compute contribution for this classifier: $\beta_t = \frac{1}{2}\log\frac{1-\epsilon_t}{\epsilon_t}$
     (smaller $\epsilon_t \Rightarrow$ larger β_t)
  3. Update weights on each training sample n
     $$w_{t+1}(n) \propto w_t(n)e^{-\beta_t y_n h_t(\boldsymbol{x}_n)}$$
     ($w_t(n)$ decreased if $y_n = h_t(\boldsymbol{x}_n)$; increased if $y_n \neq h_t(\boldsymbol{x}_n)$)
     and normalize them such that $\sum_n w_{t+1}(n) = 1$.
- Output the final classifier
  $$h[\boldsymbol{x}] = \text{sign}\left[\sum_{t=1}^{T} \beta_t h_t(\boldsymbol{x})\right]$$

Adaboost MCQ

In each round of AdaBoost, the misclassification penalty for a particular training observation is increased going from round t to round $t+1$ if the observation was

A) classified incorrectly by the weak learner trained in round t

B) classified incorrectly by the strong learner trained up to round t

C) classified incorrectly by a majority of the weak learners trained up to round t.

Adaboost MCQ

In each round of AdaBoost, the misclassification penalty for a particular training observation is increased going from round t to round $t+1$ if the observation was

A) classified incorrectly by the weak learner trained in round t

B) classified incorrectly by the strong learner trained up to round t

C) classified incorrectly by a majority of the weak learners trained up to round t.

Solution: A Change from round t to round $t+1$ is entirely dependent on the classification of the classifier at round t

Adaboost T/F

In a given round t, the weak learner misclassifies examples m_1 and m_2. The weights of m_1 and m_2 increase by the same multiplicative factor

A) True

B) False

Adaboost T/F

In a given round t, the weak learner misclassifies examples m_1 and m_2. The weights of m_1 and m_2 increase by the same multiplicative factor

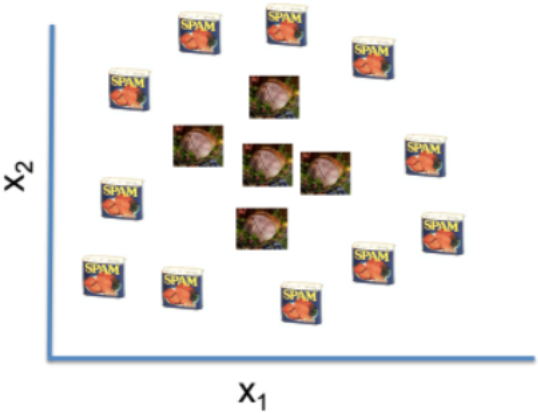
A) True

B) False

Solution: True the scaling of all misclassified examples in a given round will be the exact same

Neural Networks Motivation

Many problems will have a complicated non-linear decision boundary



Neural Networks Motivation

Suppose you have some classifier represented as follows -

◦ $f(x) = \mathbf{W}_1 x$ where x is your input and $\mathbf{W}_1$ is your weight vector.

◦ Intuition might be to try to make the model more complex by adding another weight vector, for example:

$$f(x) = \mathbf{W}_2\big(\mathbf{W}_1 x\big).$$

◦ However, this is just another linear model!

◦ Solution - add non-linearities between weight vectors

Neural Networks

Introducing Nonlinearities:

By applying a nonlinear activation function (e.g., ReLU, sigmoid) after each linear transform, we can learn more complex mappings beyond simple linear separation.

A typical two-layer neural network can be written as

$$f(x) = \mathbf{W}_2\,\sigma\big(\mathbf{W}_1 x + \mathbf{b}_1\big) + \mathbf{b}_2,$$

where $\sigma(\cdot)$ is a nonlinear activation.

Practical Motivation:

Nonlinear layers enable the model to capture complex relationships in data.

This leads to superior performance on tasks like image recognition, natural language processing, and more.

Neural Networks MCQ

Suppose you train a neural network on a training dataset D_T using a validation dataset D_V to determine the optimal value of the model hyperparameters. However, you observe that while the trained neural network has low training error, it has high testing error on the test dataset. Which of the following mistakes in your training procedure might explain these results?

A) You accidentally omitted half of the training data

B) You accidentally used the same training and validation datasets

C) You used a neural network with too many layers

D) All of the above.

Neural Networks MCQ

Suppose you train a neural network on a training dataset D_T using a validation dataset D_V to determine the optimal value of the model hyperparameters. However, you observe that while the trained neural network has low training error, it has high testing error on the test dataset. Which of the following mistakes in your training procedure might explain these results?

A) You accidentally omitted half of the training data

B) You accidentally used the same training and validation datasets

C) You used a neural network with too many layers

D) All of the above.

Solution D Omitting half of the training data makes the model more likely to overfit to the training dataset. Using the training data as validation also makes the model prone to overfitting. Using a neural network that is too deep can also lead to overfitting

Conclusion

Key Takeaways:

Make sure to review: Decision Trees, Entropy/Information Gain, Ensemble Methods, Neural Networks.

Understand the motivation for each method and their key advantages/disadvantages.

These concepts are all included in Homework 3

These concepts will also be tested on Mini-Exam 2 (03/26)