Machine learning and decision trees

Name some applications where we would want to use decision trees as our machine learning model.

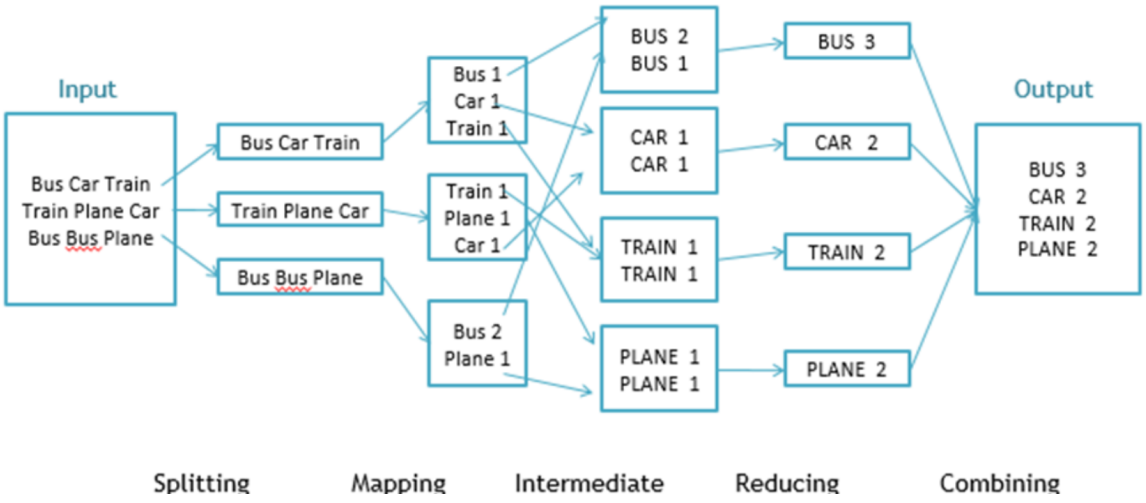
What is the classic criteria we use for 'splitting' the decision tree? Why is it expected to lead to a 'good' decision tree?

How do we avoid overfitting in the decision tree?

Imagine that you have a point (X,y) in the training data, where X are the observed features and y is the prediction. After learning the decision tree using the splitting criterion above, is it possible to predict a different prediction y' for X in the 'test' phase (i.e., when using the tree)? Why isn't this necessarily a bad thing (hint: one of the previous questions gives you plausible reasons)?

Web applications of MapReduce

Let's consider wordcount

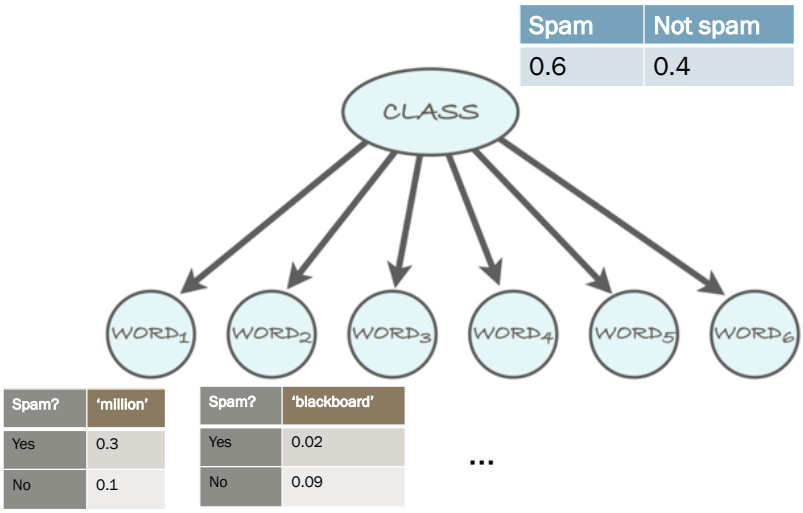


https://dzone.com/articles/word-count-hello-word-program-in-mapreduce

Naïve Bayes

Example: Spam classification (see below).

--Describe in depth how you would use Naïve Bayes for spam classification? If possible, try to be as quantitative as possible



Supervised machine learning workflow

What are the key steps in a supervised machine learning workflow?

Answer:

When would hyperparameter not be considered important or necessary?

When is it absolutely necessary? What would happen if you did not do it but only used a 'default' value of hyperparameters?

The word 'model' is used several times in the figure above. What is a model in this context? Give some examples.

Name some mechanisms for exploring data. Why would you want to explore data?

What is the dominant technique for 'gathering' data on the Web? Why has it become hard in modern times?

What does it mean to ‘deploy’ a model?

Information Retrieval and metrics to measure ‘goodness’ of IR

What is a definition of information retrieval?

Why do we tend not to use a term like ‘unstructured’ data, instead of ‘text’ or ‘natural language’ data?

What is the dominant mechanism in IR for fast retrieval?

Must IR involve text? What are some applications of text that involve little text or even no text?

List two definitions of precision (probabilistic and non-probabilistic). Why are they equivalent?

List two definitions of recall (probabilistic and non-probabilistic). Why are they equivalent?

How do we capture the tradeoff between precision and recall? What is one issue with this method? What is another mechanism that addresses this issue?

What is a contingency table? How does it help you compute precision and recall?

Let’s use a real example

You are given the following set of ‘documents’ :

D1: The Getty Villa is modeled after a Roman villa
D2: The Getty museum is worth a visit
D3: I can’t believe she missed a trip to the Getty Villa
D4: make sure to not miss the Getty villa while you’re in los angeles
D5: we can’t make time to visit museums this trip to los angeles
D6: why not visit the Getty on Wednesday?

You are given the following sets of queries:

Q1: villa los angeles

Q2: museums los angeles

For the purposes of all questions, you can ignore case i.e. ‘The’ is the same word as ‘the’.

You are told that the ground truth for Q1 is the set {D1} and the ground truth for Q2 is {D1, D2}.

Question 1: What is the *size* of the vocabulary?

Question 2: You design a simple retrieval system as a baseline. Given a query, you tokenize it into words (ignoring case) and if *all* words in the query are in a document, you put the document in an answer-set. [For example, 'villa los angeles' is tokenized into {villa, los, angeles}]

Given this set-based mode of retrieval, what is the output of the system for Q1 and Q2?

What is the precision, recall and F-measure of this system for both Q1 and Q2? Show your working.