Graphs for Decision-Making

EN 500.111 Week 8

Presentation Work

- Meet with your groupmates for ~20 minutes to work on coordinating presentation
- I'll be coming around to each group to check in and answer any questions
- https://github.com/mkirsche/TAG2020/blob/master/presentation/guidelines.md

Outline

- Dynamic programming
- Machine Learning
  - Decision trees
  - Neural networks

Counting Paths

Sometimes we're interested in knowing not only the shortest path, but how any routes there are from one node in a graph to another

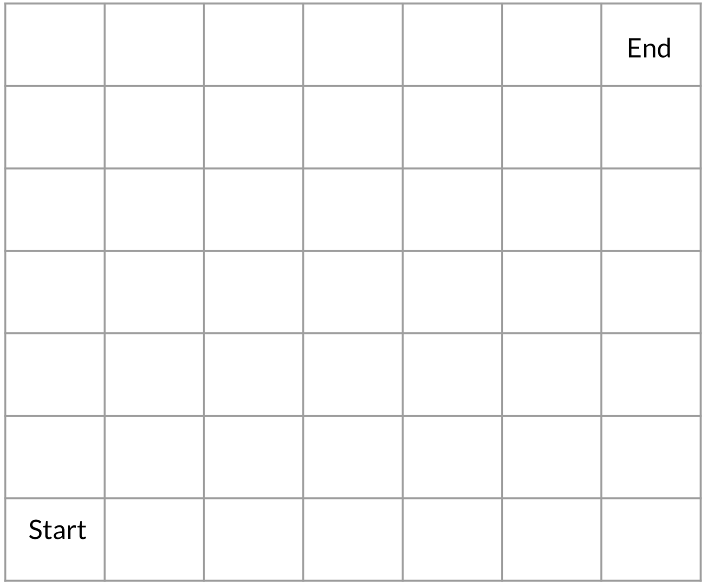
- ➔ Finding traffic bottlenecks for road planning
- ➔ Checking whether people are in the same group within a social network

How many shortest paths are there from start to end in this grid (moving up/down/left/right)?

		End
Start		

Counting Paths

What about this graph?



Counting Paths

What about this graph?

						End
		X				
			X		X	
	X					
				X		
		X	X			
Start						

Counting Paths

What if we want the total number of paths, including those that aren’t shortest paths? Does strategy still work?

						End
		X				
			X		X	
	X					
				X		
		X	X			
Start						

Dynamic Programming

- ➔ A technique introduced in the 1950s for that involves breaking down a problem into smaller versions of the same problem
- ➔ Tons of applications in all different subject areas

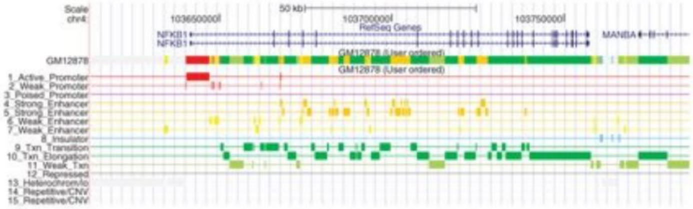
Algorithms that use dynamic programming

- Recurrent solutions to lattice models for protein-DNA binding
- Backward induction as a solution method for finite-horizon discrete-time dynamic optimization problems
- Method of undetermined coefficients can be used to solve the Bellman equation in infinite-horizon, discrete-time, discounted, time-invariant dynamic optimization problems
- Many string algorithms including longest common subsequence, longest increasing subsequence, longest common substring, Levenshtein distance (edit distance)
- Many algorithmic problems on graphs can be solved efficiently for graphs of bounded treewidth or bounded clique-width by using dynamic programming on a tree decomposition of the graph.
- The Cocke–Younger–Kasami (CYK) algorithm which determines whether and how a given string can be generated by a given context-free grammar
- Knuth's word wrapping algorithm that minimizes raggedness when word wrapping text
- The use of transposition tables and refutation tables in computer chess
- The Viterbi algorithm (used for hidden Markov models, and particularly in part of speech tagging)
- The Earley algorithm (a type of chart parser)
- The Needleman–Wunsch algorithm and other algorithms used in bioinformatics, including sequence alignment, structural alignment, RNA structure prediction [11]
- Floyd's all-pairs shortest path algorithm
- Optimizing the order for chain matrix multiplication
- Pseudo-polynomial time algorithms for the subset sum, knapsack and partition problems
- The dynamic time warping algorithm for computing the global distance between two time series
- The Selinger (a.k.a. System R) algorithm for relational database query optimization
- De Boor algorithm for evaluating B-spline curves
- Duckworth–Lewis method for resolving the problem when games of cricket are interrupted
- The value iteration method for solving Markov decision processes
- Some graphic image edge following selection methods such as the "magnet" selection tool in Photoshop
- Some methods for solving interval scheduling problems
- Some methods for solving the travelling salesman problem, either exactly (in exponential time) or approximately (e.g. via the bitonic tour)
- Recursive least squares method
- Beat tracking in music information retrieval
- Adaptive-critic training strategy for artificial neural networks
- Stereo algorithms for solving the correspondence problem used in stereo vision
- Seam carving (content-aware image resizing)
- The Bellman–Ford algorithm for finding the shortest distance in a graph
- Some approximate solution methods for the linear search problem
- Kadane's algorithm for the maximum subarray problem
- Optimization of electric generation expansion plans in the Wein Automatic System Planning (WASP) package

How is it different from brute force?

- Consider some point in the middle of the grid
- There are tons of ways to get there and tons of ways to get to the end from there
- Brute force explores every pair of incoming path and outgoing path since it corresponds to a valid start -> end path
- The dynamic programming algorithm doesn’t care how incoming paths got to the X square and just adds them all to its tally before considering outgoing paths

Another example



ChromHMM

Suppose we are trying to identify important regions of a genome. There are methods which establish different types of annotations (histone modifications, types of gene expression regulators, etc.) and assign each basepair a score for how likely each annotation is to apply to it. This is based on sequence context, similar sequence in already-annotated genomes, etc. We want to identify contiguous segments which contain a lot of bases with a similar annotation.

A	G	C	G	C	T	A	C	A	C	G	T
10	-5	-4	15	10	5	0	20	-100	20	10	-1

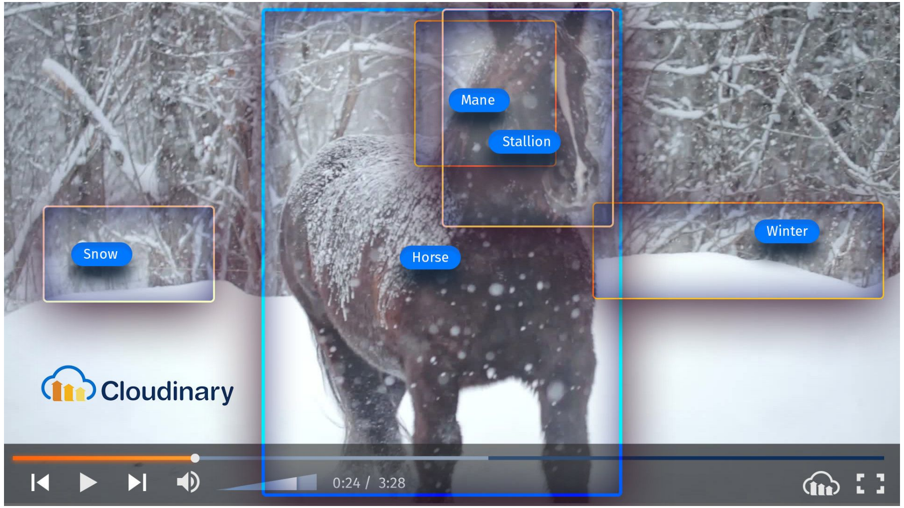
Subarray sum problem

Given a list of numbers, find the contiguous block with the highest sum.

A 10	G -5	C -4	G 15	C 10	T 5	A 0	C 20	A -100	C 20	G 10	T -1

Classification Tasks

→ We commonly want to use algorithms to perform classification, or figuring out what category an object belongs to:
  - Automated image/video tagging
  - Determining what organism a genomic read came from
  - Filtering spam in an email inbox
  - Triaging patients in a hospital
  - Determining whether someone has good enough credit to receive a loan
  - ...and much more!



https://cloudinary.com/blog/new_google_powered_add_on_for_automatic_video_categorization_and_tagging

Decision Trees

- ➔ A tree where each leaf node is a category and each intermediate node is a question/feature which splits the data
- ➔ Uses training data (category already known) to decide on the best features to split on and to label leaf nodes, and then predicts category of new objects

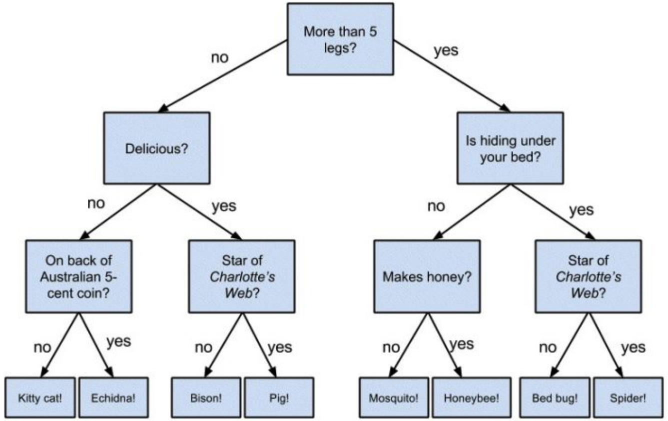
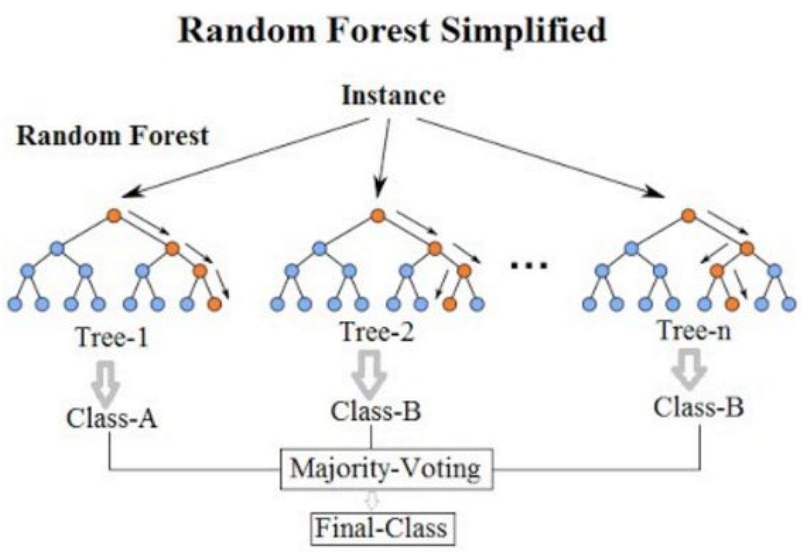


Figure 17-1. A "guess the animal" decision tree

https://medium.com/analytics-vidhya/a-guide-to-machine-learning-in-r-for-beginners-decision-trees-c24dfd490abb

Random Forest

- A collection of decision trees where each one is split on different features and uses different training data
- To perform classification, they vote on the category a given object falls into



(Wikipedia)

Hospital Queue Time Minimization

- ➔ One place random forests have been applied is to predict hospital queue times based on limited resources
- ➔ This can be used to recommend the order in which treatments/tests/etc. Can be done for different patients

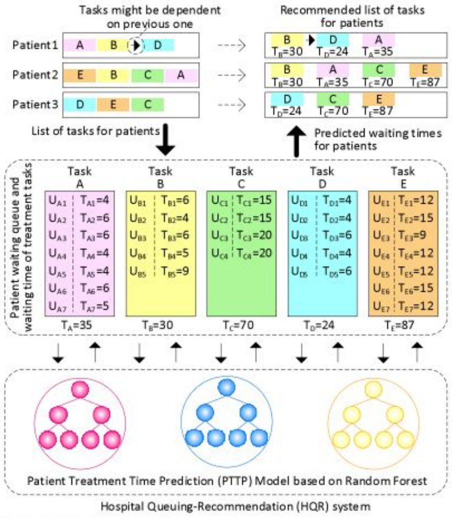
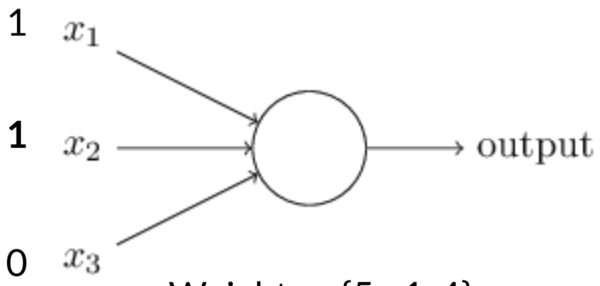


FIGURE 1. Workflow of patient treatment and wait model.

"A Parallel Patient Treatment Time Prediction Algorithm and Its Applications in Hospital Queuing-Recommendation in a Big Data Environment"

Another Classification Method - Perceptrons

- ➔ Takes a series of binary (0 or 1) inputs and classifies as 0 (no) or 1 (yes)
- ➔ The perceptron stores a series of weights and a threshold value
- ➔ It classifies by adding up $weight_i * x_i$ for each of the inputs and checking if it’s greater than (1) or less than (0) the threshold



Weights = {5, -1, 4}
Threshold = 4.5

Sum = 1*5 + 1*-1 + 0*4 = 4
Sum < Threshold, so output 0.

Artificial Neural Networks

- ➔ (Very) loosely based on neurons in the human brain
- ➔ The idea is to perform classification tasks by combining a large number of perception-like nodes
- ➔ Nodes are arranged in layers, with edges between nodes in consecutive layers
- ➔ Weights of edges are learned from training data and reflect how much an input from one layer affects the outputs of the next layer

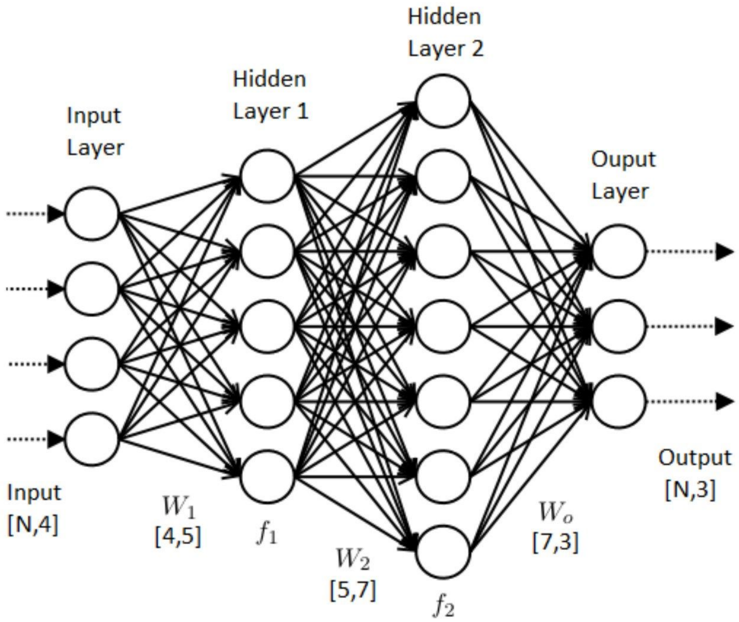
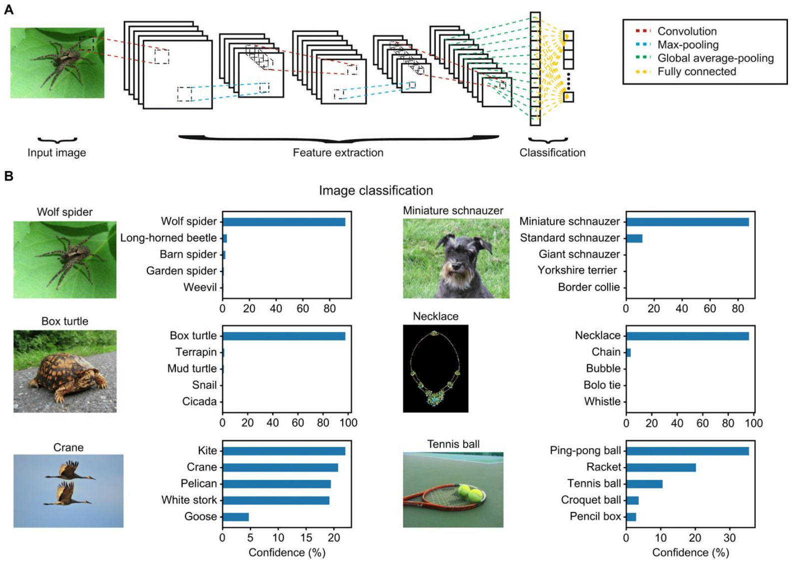


Image Classification

- ➔ Neural networks commonly used to classify images
- ➔ Takes images as a series of pixel intensities and is able to infer even very complex patterns
- ➔ Lots of ongoing research for how neural networks should be arranged and what types of functions should be used



Conclusions

- ➔ Dynamic programming gives us faster algorithms than brute force when we can divide a task into smaller versions of the same task
- ➔ Graphs are used for the underlying structure of many “machine learning” classificiation techniques, including random forests and neural networks