What is a decision tree

- A set of ordered rules to classify data
- Each node addresses and input variable
- Each leave is a class of the data

Types of decsision trees

- Classification trees have leaves of descrete data values
- Regression trees have leaves of continuous data values

Advantages of decision trees

- Simple and intuitve
- Robust
- good computational performance

Disadvantages of decision trees

- can't model every type of data
  - XOR parity
- Greedy algorithms get stuck in local optimum

Decision tree algorithms

- Information gain
- gini impurity
- variance reduction
- entropy

How are decision trees constructed

- data inputs are recusively split into sub classes based on a single input
- until they can no longer be split that improves prediction of their class
- or all data in a class is the same

How are the sets split

- Identify candidate splits
- Loop through all possibilities
- Apply metric to candidate split
- Choose the split that produces th ebest sub sets

Identifying the candidate split

- Each node in the decision tree describes a rule for splitting the data
- Each rule consists of a variable to be considered and threshold to value to compare
- threshold values between the variable values don't increase prediction

Information gain

- Measures the difference in entropy before and after split
- Is measured to gage the purity of a set

Entropy

- The probability of being in that class
- For this algorithm is based on frequency

Gini impurity

- Measures the diversity of a set
- Is the squared probability of being in a class

Variance reduction

- For regression trees that have leaves of continuous values
- Variance of the set is taken before the split and it used to find the difference between the variance of the two new resulting sets

Split quality - purity metrics

- Quality of a decision tree is based on terms purity
- Purity is measured in terms of probability of being in a class
- Gini purity
  - based off of gini impurity
- Information gain
  - Based off of entropy
- Variance reduction

Improvement

- Improvement is based on the difference in quality between the original sub set and the joint quality of the two new subsets

Evaluating a decision tree

- Error rate
  - The proportion of errors across all instances
- Resubsitition error
  - The error rate in the training data
- Test set error
  - The testing error
- hold out
  - Hold back some data for testing

Repeated testing

- Hold data back for testing
- Repeat testing and average the error across all tests

N-fold cross valdation

- Split data up into n subsets
- need to use one of the subsets for testing and teh rest training
- Do that n times
  - Everytime switching the test subset until all subsets have been used for testing
- Average the error result
- Good because you use all data for testing and training
- wanted to maximise that
- testing for validation
- training for accuracy

The bootstrap

- Randomly select data points from the training data into N bags
  - Can have the same data point for training in same bag as well as others
- Train on these bags
- test each instance with same data
- average the error rate to get the accuracy

Properties of bootstrap

- Can be pessimistic as maybe not all data is chosen
- combine with resubsitiion error to get realistic estimate

Pruning

- Splitting criteria that is too small and specific grow small trees, underfitted
- Splitting criteria that is vague will grow overfitted large trees
  - This can be good
- Can then prune the branches that dont significantly contribute to the acrucacy of prediction for classes
  - Or leaves that are too small single values
- Can be ideal as it is simple to do

Reinforcement learning

- generate actions that affect the environment it is in
- Learning what to do when
  - What actions
  - What states
  - Reward

Representing the problem

- Markov decision tree
- State
- Action
- Transition function
- Reward

Probabilistic actions

- Actions and are represented as probability distribution across the states

Policies

- Tells the agent which actions to take in what states to get somewhere
  - Most likely the goal state
  - This will give you a sequence, a plan or the best optimial states
- Commonly based on a value function and an action selection mechanism

Delayed rewards

- The notion that you will only get a reward at the end
- Utility defined by the fact you take best actions based on expected reward in the future

- Don't know the value of the policy until you have the reward
- Based on how many actions and states you came across you can see how good this policy was compared to others
- Allows you to take the immediate reward which can be good or bad to get the expected reward in the future

Bellman equation

- The police is defined by the action that maximises your expected discounted reward over the policies
- maximises long term expected reward

Finding policies

- start by picking arbitrary values
- Continue these until you start to get rewards
- Can use the bellman equation as an update for each states value
- repeat until convergence
- Possible because of the back propagation of the reward throughout the policy

Value iteration

- Have some estimate of the policy
- So use this to update and make the policy better
- Update every iteration estimate of the state s
  - Recaclulated it to be
  - The immediate reward
  - Plus the discounted rewards of the states already visited in this policy

Value functions

- functions of states
- Tells you the estimated value of being in this state
- How good is based on expected reward of being in that state
- Value functions are defined in respect to particular policies ### State-value functions
- How good it is to be in a state
- The evaluation of this policy
- Defined by expected return

- Expresses the relationship with this state to its successor states

Policy evaluation

- Trying to define the value a policy
- Pick aribtrary values of the policy
- Repeatedly applying the bellman equation as an update rule
- Full backup in place

Policy improvement

- should we change our given policy?

Value iteration

- Have some estimate of the value of function
- Use this to make the policy better
- Every iteration update the value of S
- by recaculating it to be
  - The immediate reward
  - Plus the summed discounted rewards that already encountered

Model free algorithms

- No model available so have to estimate the transition function and value function
- Temporal difference learning
  - The difference in values between two states
- Monte carlo methods
  - Randomly sampled states and actions to get the expected reward to make an estimate of that states value
- Q-learning

Monte carlo methodoly

- Don't know the optimal policy yet or fully
- Try to estimate it by randomly taking a set of states and actions
- Might end up in states more than once
- take the average of the values ending up in that state as the estimate for it
- can use this to back propagate through the policy so far to update it

Monte-carlo policy evaluation

- only consider first visit as estimation
- iterate through forever
- Get all instances of the first occurance of the states
- average the rewards of them to get their values

Monte-carlo properties

- Good because you don't need to go through all the MDP
- you need a lot of samples to get a good estimate
- Planning time is independant of state space

Estimating action values

- Without the model of the environment, T we cannot choose the optimal actions
- Estimate the action values

Temporal difference learning

- Like monte carlo
- estimating based on exipiernce
- update as you go
- Estimates the value of the previous step by taking the next and looking at the difference in reward

Advantages of TD prediction

- Learn as you go
- Computationally unexppensive
- Don't need the model
- Learn from a guess

TD control

- Use the values guess by td to estimate best path to move agent through world

Q-learning

- model free environment
- intiialise random values for all states
- Implements e-greedy algorithm to select actions
- Does updates previous states based on reward
- The reward back propergates through the state values

Episode

- One loop through from random start state to goal

Trial

- many run throughs of an episode
- Memory is transferred from episode to episode

Expierement

- Many run throughs of trails
- Memory isnt shared between trails
- Can get mean and standard deviation performance accross trails

Methods for known transition functions

- Policy evaluation
  - Finds the value of a policy
- Value iteration
  - Finds the optimal policy assuming greedy action selection

Model free methods

- Monte carlo
- Temporal difference
- Q learning

Eligibility traces

- A record of the most recently visted state action and reward tuples
  - Stored SAR in chronological order
  - Updated on each step of the algorithm

- basic mechanism for Temporal credit assignment
  - Spreads values throughout the value function
- Bridge between MC and TD

Forward view

- Look to the future to see rewards to determine current action
- Theoretical and not implementable

Backward view

- Traces correspond closely to short term memory
- Our value function and transition function are long term memory
- Implementable

Models

- Anything that can be used to predict results of actions
  - Simulated expeirence
- distributed

Planning

- Takes a model
- A method of improving the policy or producing a policy

Dyna q

- Added onto q-learning
- Trying to create the transition and value function
- mixture of model based and model free
- For every step
- Simulate lots of next steps
  - Use this to update the q table
  - Get a better prediction of current state

Table based solutions

- Everything is a matrix or vector
- Not always practical
- Can be limited in amount of space they take
- long time of updating them

Discretization

- Taking a continuous value and making a discrete estimate of the same value

Function approximation

- We have inputs and outputs
- We want to find the what the function does
- Then we can use that to map more new inputs to outputs
- State value function approximation
- State-action vallue function approximation
  - These are control function approximation methods
- Evaluated with MSE

Generalisation

- Learning from expierence
- One can assume if in state that has already been visited then that same value applies

Partial observability

- Cannot uniquely identify the state of the world
- Need memory to tell use where we are
- Make observations ofcurrent state

Limited observablility

- Observables
- Observation function
- POMDP model is now ‘

Objective and belief state

- Believed current state from the observations
- A probability distribution across the states we think we could be in

Belief update

- In a belief state b, we take an action and an observation
- the result is b’

Finite horizon

- all the belief states can be contained in a vector and transformed to linear regression
- if you were then to plot all the lines this is peicewise-linear convex
  - Get a cup looking shape and choose the maxmum value line of you x position

POMDP Value iteration

- the value of the belief state given we're in time step t
- is equal to the maximum over all actions
- The reward we get from being in the belief state
- plus the sum of the discounted rewards we got to get their

Instance based solutions

- Nearest sequence memory
  - Keep current in STM
  - N instances of previous in LT<
  - Match STM and LTM on nearest neighbouts
  - Calculate highes value action
- Can learn very quickly

Types of learning

supervised - The machine is given inputs and outputs and its goal is to learn how to reproduce the outputs from the inputs - Used for classifying data

unsupervised - There is no disered output you are given inputs and after some iterations you start to categories the data based on some criteria - For regression - for prediction - Given unlabelled inputs only and it has to classify the inputs to be able to handle new inputs

reinformcement - Agent performs actions that affect the enivornment around it and gets some reward or punishment based on them

The addition law of probability

the probability of two independant events is the addition of probability of both events happen

if they're not mutally exclusive then you have to - addition of them both happening - minus the intersect - whole thing minus 1

Discrete distribution

- finite possibilities of things to happen and all have an equal chance of occouring

Cumulative

- The possibility of current event happening with all of the other events leading to it happening as well

Binomial

- 2 outcomes
- probab of HHTT HHTT HTHT THHT TTHH THTH

4 x (1/2)^2 x (1/2)^2 = 0.25

Uniform distribution

- A distribution has a constant probability

Continuous data distributions

- A continuous random variable is a random variable with a set of possible values that is infinite or uncountable
- a countinus random variable is a random variable with a set of possible values that is infinite

Variance

- how far the random varaibles are from the mean

Expected value

- What is the expected value that a value x falls into in the probability distribution

Covariance - joint probability

- the covariance is the stregnth of linear relationship between two variables

Conditional probability

- how can you tell if values x and y are independant
- if you change the value of x it shouldn't change y

Effect of standard deviation

- square root of variance
- if the std is larger than point x has a greater probability of falling into the area that it covers
- it moves further from the mean value

bayes rule

- Assume indepence of the variables
- the chances of going to beach and getting sunstroke are linked but they're also independant
- if you go to the beach its because its hot
- if you get a sun stroke then it is because it is hot
- can multiple across the probabilities to get the chance of it being a part of that

if A and B are NOT independant events

Interpreting covariance

- if COV(X, Y) $<$ 0 - they're negatively correlated
- if COV(X, Y) $>$ 0 - they're positively correlated
- if COV(X, Y) = 0 - they're independant

Conditional probabilites

- If A and B are not independant events
- then the probability of P(A, B) = P(A) P(B|A)
- if they are independant P(A, B) = P(A) . P(B)
- leads to bayes rule
- P(A|B) = P(A).P(B|A)/P(B)

gradient descent

- want to climb to top
- if we're at top then slopes either side
- slopes is found by differentials

determinates

- det(A) = (ad - bc)

inverse

- 1/det(d -b; -c a)

If we generate a N dimenional Gaussian data distribution

- if you sample from the same data set
- they will have independant covariance

Maximum likelihood (MP)

- Does not assume a prior parameters
- goal is to maximise the probability of it happening
- can run into problems estimating

Maximum a prior likelihood (MAP)

- Assumes a prior
- maximise the prosteriar
- MAP and ML esitmates are identical when the prior is uniformly distributed

Frequentest approach

- Probability is the limit of observed frequency as number of observations goes to infinity
- probability is the limit of observerd frequency as number of observations goes to infinity

Bayesian approach

- Probability is a degree of confidence that one attaches to an uncertain event
- Probability is a degree of confidence that one attaches to an uncertain event

Eigenvectors and values

- Eigenvectors and values scale a given matrix
- Ax = lambdax
- The A could be a matrix and lamdba a scalar
- lambda would be an eigen value if it produces the same product if multiply the lambda by the vector and matrix by vector
- then you can say any multiple of x is a value
- when we transform x by multiplying by a we end up with vector x again but scaled by lambda
- that means direction isn't affected by transformation
- if x is an eigenvector, it means that the product of matrix A . x is the same as lambda . x
- also means that lambda is an eigenvalue of x
- as we transform x by multiplying by A we end up with x scaled by lambda eigenvalue
- means the direction isn't affected by the transformation

Clustering

- Idea of clustering is group patterns together so that
  - Patterns of data that are similar are in the same cluster
  - patterns of data that are disimilar are in different clusters
- Require a way to determine similarity

K-means clustering

- initialise K clusters
- Assign K amount of points to find the centre of the clusters
  - Known as centroids
- Randomly assign there positions
- Iteratively
  - Find the most amount of data that is nearest to the centroid
  - assign that cluster to that centroid and move the centroid towards it
  - recompute the cluster centres as the means of the assigned data points
  - recomput ht ecluster centres ans the means of the data points
  - Repeat until convergence
- an example of hard clustering
- clusters do not overlap

- one data point to one cluster

Mixture of guassians

- a multidemsion mixture of gaussians can be used to represent almost any distribution
- Gaussian FA and PCA are convient ways to reduce dimensionality of high dimensional data sets
- make strong assumptions of the data
- mean and variance taken into consideration
- soft clustering

EM algorithm

- Need to know the gaussian parameters mean and std for each cluster to estimate the data point cluster memebership
- Randomly intialise the parameters
- look at data point and see how likely it is that it came from that data point
  - bayes
- Re-estimate parameters
- repeat until convergence

Pattern classification

- Pattern classifiers partition the input space
- May have multiple input data dimensions
- May have multiple output classes
- Type of deicision boudary depends on the classifier
- Variety of ways to determine boudaries

Information theory

- The probability P(X) encodes uncertainty about the random variable X

Entropy

- We can quantify the information represented in such a random variable
- This is the entropy of the variable X which is averageamounth of the information required to encode x

Naive bayes versus full gaussian classifier

- Naive bayes only estimates and uses marginal gaussian parameters
- Only has non zero values in the covariance matrix along its leading diagonal
- Assumes the variables are independate
  - zero covariance

Limitations of generative models

- Bayes decision rule minimise average probability of error
- Can train generative models by directly estimating paramters
- Bayes classifier is the minimum error classifier only if our model of the data is appropriate
- In particular the form of the class is conditional distribution is correct

Discriminative models

- Generate model classfication requires the class posterior
- don't model the data
- just try to find decision boundary

Linear decision boundaries

- Decision bounders partition the input space into regions
- Each region is associated with a class label

The perceptron

- LEarning means changing weights between the neurons
- Relationship between input and output is important in computational neurosience
- Simple but limited capabilities
- Basic concepts are useful for multi-layer models
- If the data is linearly separable then:
  - Li

Deep learning

- Deep learning multiple layer neural network
- feature detection on each layer

Deep networks

- convolutional neural networks
  - Emply alternating layers of convolutional networks
  - pooling layer
  - output uses MLP
- deep belief
  - Consists of perceptron stackked
  - classification output layer

Auto encoder

- An auto encoder is trained with standard training algorithm
- it learns a map the input back onto itself

Single vs multi layer networks

- Single layer networks implement linear decision boundary
- Multilayer network can implement complex decision boundaries

Limitation of back propagation

- Multiple hidden layers
- gets stuck in local optima
- Slow convergence
- only use labelled data