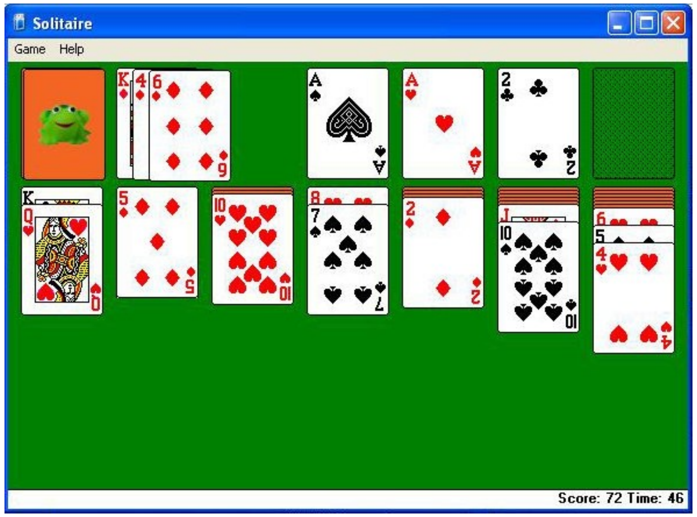
Lecture 12

Sampling and re-sampling

Why Sampling?

Problem: What is the chance of winning at patience?



Analytical solution

- too difficult for mortals!

Enumerate all possibilities by computer

- the world won't last long enough!

Play 100 games and see how many you win

- a practical possibility

Monte Carlo Methods

Sampling solutions to problems like estimating the probability of winning at cards are called Monte Carlo methods:

Every year punters lose large sums of money by sampling in the casinos at Monte Carlo.

(and they don't even bother to calculate the probability of winning)

Monte Carlo in general

Given a set of discrete variables, and the ability to make random samples from them can we infer the probability distribution over the data.

We could:

- fit a distribution to the data
- train a classifier (neural net, Bayesian net etc) with the data
- retain the samples as the data model
- etc.

Example: Sampling from a Bayesian Network

Given a Bayesian Network with variables $X=\{X_1, X_2 .. X_N\}$, we can draw a sample from the joint probability distribution as follows:

Instantiate randomly all but one of the variables, say Xi
Compute the posterior probability over the states of Xi
Select a state of Xi at random, based on the distribution

We can always do this even if the network is multiply connected.

Markov Blanket - an implementation detail

If all nodes in a network except Xi are instantiated then only a small set of nodes are needed to compute the posterior distribution over Xi.

The Children of Xi

The Parents of Xi

The parents of the children of Xi

These variables are termed the Markov Blanket of Xi

Monte Carlo methods in Bayesian Inference

Given a Bayesian Network with some nodes instantiated in cases where propagation is not feasible we can estimate the posterior probabilities of the un-instantiated variables as follows:

1. Draw *n* samples from the Bayesian network with the instantiated variables fixed to their values
2. Estimate the posterior distributions from the frequencies

Problem: The state space is big, so we need a very large number of samples.

Markov Chain

Random sampling from a Bayesian Network may not be the best strategy. For efficiency it would be desirable to try to pick the most representative samples.

One way of doing this is to create a 'Markov Chain' in which each sample is selected using the previous sample.

This is the basis of Monte Carlo Markov Chain (MCMC) methods.

Gibbs Sampling in Bayesian Networks

Here is a simple MCMC strategy:

Given a Bayesian Network with N unknown variables choose an initial state for each variable at random, then sample as follows:

- Select one variable at random, say Xi
- Compute the posterior distribution over the states of Xi
- Select a state of Xi from this distribution
- Replace the value of Xi with the selected state

Loop

Intuition on Gibbs Sampling

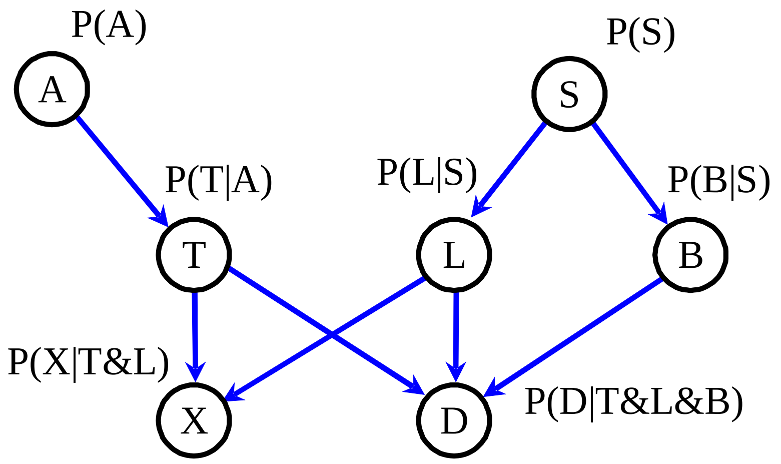
At every iteration we weight our selection towards the the most probable sample. Hence our samples should follow the most common states accurately.

Moreover, the process is

- Ergodic: it is aperiodic and it is possible to reach every state;
- Balanced: it will converge to a stable distribution given enough samples.

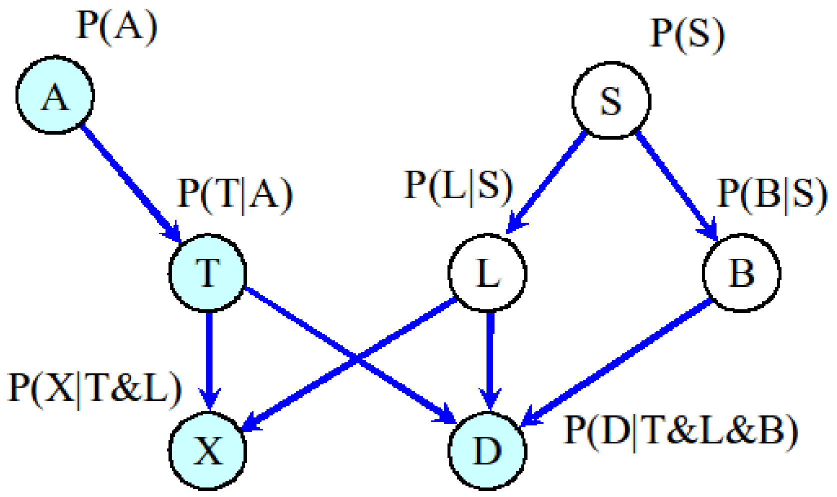
Example of Gibbs Sampling

Let’s look at how we can use Gibbs Sampling to solve the Asia Network. Remember that for certain instantiations Pearl’s propagation algorithm will fail due to loops



Example of Gibbs Sampling

The first step is to instantiate any known variables to their measured state. In the example below we assume that A, T, X and D are instantiated. This is an example where Pearl’s algorithm fails.

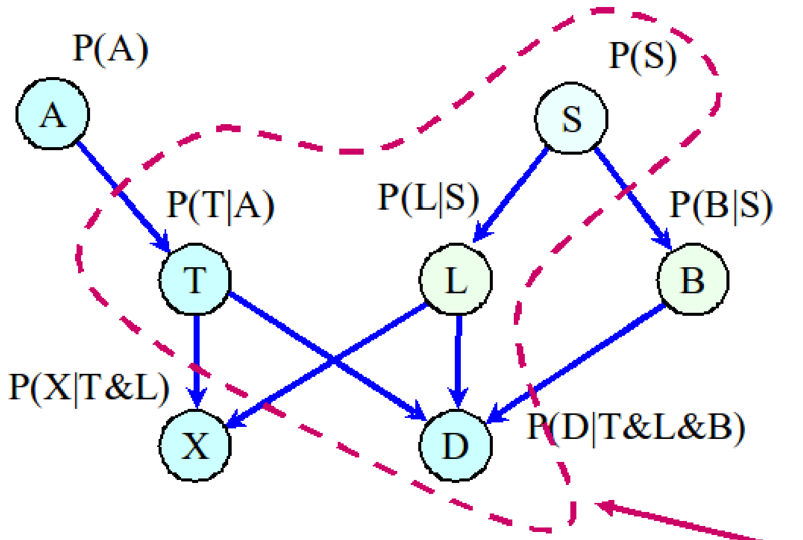


A	Asia	A=a1
B	Bronchitis	Unknown
D	Dyspnea	D=d1
L	Lung Cancer	Unknown
S	Smoking	Unknown
T	Tuberculosis	T=t0
X	XRay	X=x0

Example of Gibbs Sampling

Choose a random instantiation for all the unknown variables, in this case S, L and B

Select one unknown variable at random, for example B, and identify its Markov blanket



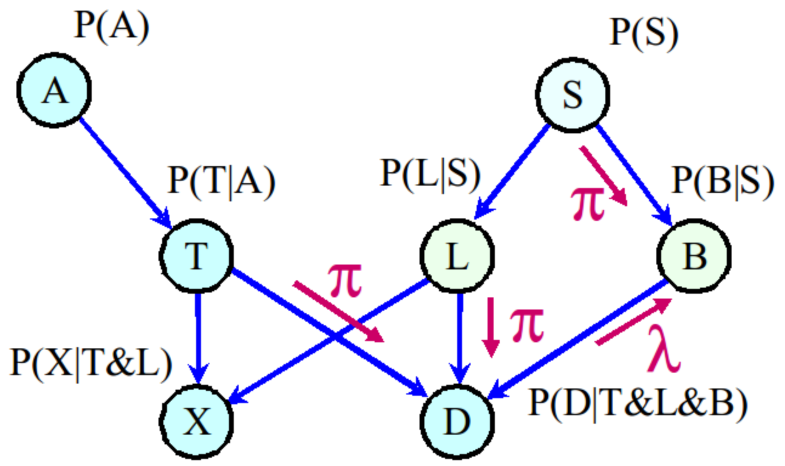
A	Asia	A=a1
B	Bronchitis	**B=b0**
D	Dyspnea	D=d1
L	Lung Cancer	**L=l1**
S	Smoking	**S=s1**
T	Tuberculosis	T=t0
X	XRay	X=x0

Markov blanket of B

Example of Gibbs Sampling

Re-compute B from its Markov Blanket, suppose that we obtain P’(B) = (0.7, 0.3)

Select a state of B weighted by its distribution (select b0 with probability 0.7 and b1 with probability 0.3)

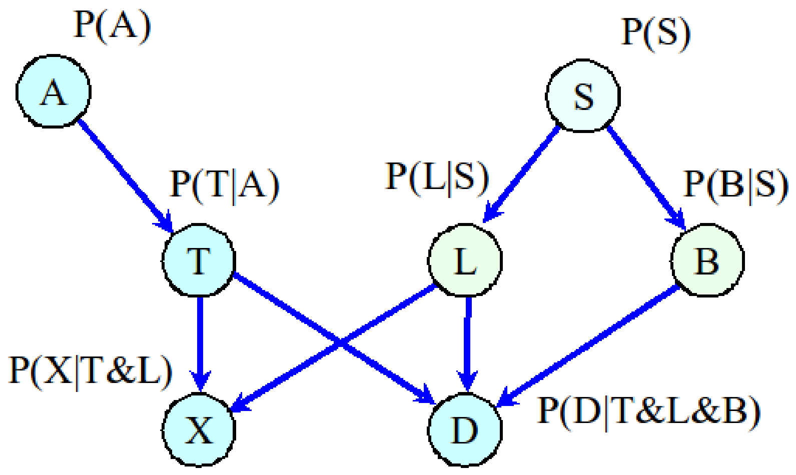


A	Asia	A=a1
B	Bronchitis	**B=b0**
D	Dyspnea	D=d1
L	Lung Cancer	**L=l1**
S	Smoking	**S=s1**
T	Tuberculosis	T=t0
X	XRay	X=x0

Example of Gibbs Sampling

Suppose we selected b0, the state of the network is our first sample - {a1,b0,d1,l1,s1,t0,x0}

We are only interested in the unknown states so our sample is really just {b0,l1,s1}

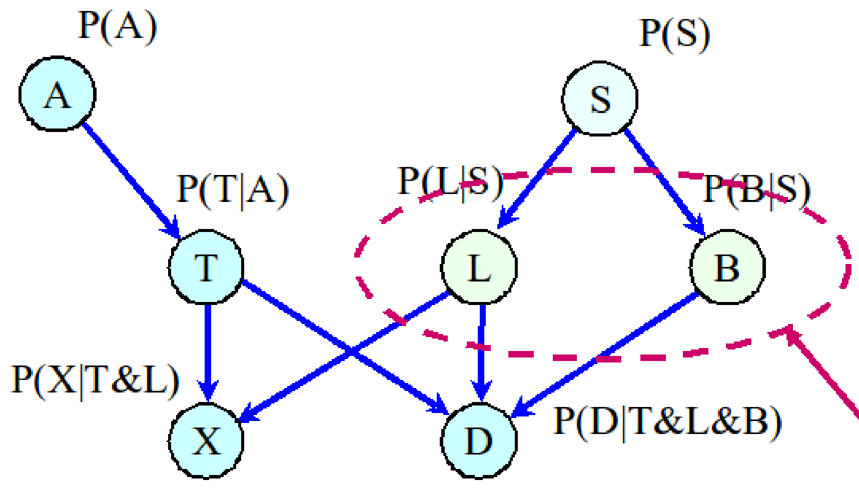


A	Asia	A=a1
B	Bronchitis	**B=b0**
D	Dyspnea	D=d1
L	Lung Cancer	**L=l1**
S	Smoking	**S=s1**
T	Tuberculosis	T=t0
X	XRay	X=x0

Example of Gibbs Sampling

Now just repeat the process

Select one unknown variable at random, for example S, and identify its Markov blanket



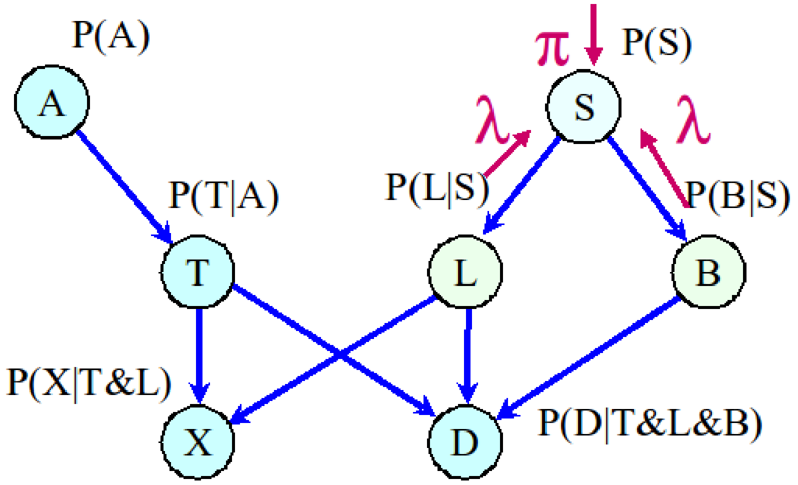
A	Asia	A=a1
B	Bronchitis	**B=b0**
D	Dyspnea	D=d1
L	Lung Cancer	**L=l1**
S	Smoking	**S=s1**
T	Tuberculosis	T=t0
X	XRay	X=x0

Markov blanket of S

Example of Gibbs Sampling

Re-compute S from its Markov Blanket, suppose that we obtain P’(S) = (0.4, 0.6)

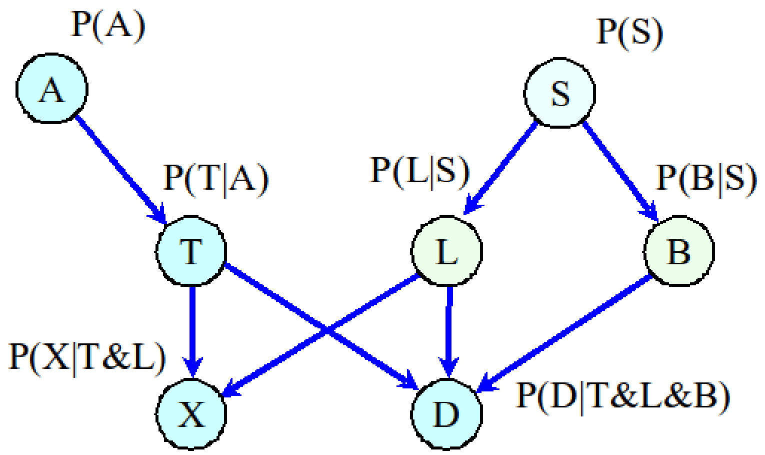
Select a state of S weighted by its distribution (select s0 with probability 0.4 and s1 with probability 0.6)



A	Asia	A=a1
B	Bronchitis	**B=b0**
D	Dyspnea	D=d1
L	Lung Cancer	**L=l1**
S	Smoking	**S=s1**
T	Tuberculosis	T=t0
X	XRay	X=x0

Example of Gibbs Sampling

Suppose we selected s0, the state of the network is our second sample - {b0,l1,s0}



A	Asia	A=a1
B	Bronchitis	**B=b0**
D	Dyspnea	D=d1
L	Lung Cancer	**L=l1**
S	Smoking	**S=s0**
T	Tuberculosis	T=t0
X	XRay	X=x0

Example of Gibbs Sampling

We keep repeating to gather more samples

{b0,l1,s1}

{b0,l1,s0}

{b0,l1,s0}

{b1,l1,s0}

{b1,l0,s0}

{b1,l0,s0}

{b1,l0,s0}

{b1,l1,s0}

We compute the probabilities from the sample frequencies.

After drawing eight samples we have:

P’(B) = (3/8, 5/8)

P’(L) = (3/8, 5/8)

P’(S) = (7/8, 1/8)

Metropolis-Hastings Algorithm

The Metropolis-Hastings algorithm provides a general criterion for accepting new samples such that the sample chain is both ergodic and balanced.

Gibbs sampling as described above is a special case of the Metropolis Hastings algorithm, but it may require more samples to reach convergence.

Original Metropolis Algorithm (1953)

(N. Metropolis, M. Rosenbluth, A. Rosenbluth, E. Teller and A. Teller)

Given a chain of samples X^0, X^1, X^2 . . X^t , and a method to compute sample X^{t+1} from X^t:

If a sample has a probability $P(X^{t+1})$

A probability ratio is taken to be:

$$\alpha = P(X^{t+1}) / P(X^t)$$

Calculate a probability of acceptance $p_t = \min(\alpha,1)$

Add X^{t+1} to the chain with probability p_t

Hastings' Generalisation

Compute the “proposal density” $Q(X^{t+1}| X^{t})$ which is the probability of sampling X^{t+1} from X^{t}

The probability ratio is taken to be:

$$\alpha = (P(X^{t+1})/Q(X^{t+1} | X^{t})) \;/\; (P(X^{t})/ Q(X^{t} | X^{t+1}))$$

$$\alpha = P(X^{t+1})\, Q(X^{t} | X^{t+1}) \;/\; P(X^{t})\, Q(X^{t+1} | X^{t})$$

Calculate a probability of acceptance $p_t = \min(\alpha,1)$

Add X^{t+1} to the chain with probability p_t

Metropolis-Hastings Algorithm

For the Gibbs sampling example we can easily calculate the proposal density $Q(X^{t+1} | X^{t})$

eg: for sample [b0, d1, s1] what is the probability of the next sample being [b0, d0, s1]?

We have a probability of 1/3 of selecting variable D for re-calculation.

After selecting D and recalculating its distribution we have the probability of selecting d0 (as opposed to d1).

Re-sampling

Re-sampling gives us a way of estimating statistical properties from a finite data set.

Instead of using the data once we use samples from it several times over to estimate properties like model accuracy and parameter variance

Hold out methods

Hold out methods are the most useful techniques involving re-sampling.

Typically they involve holding back a proportion of the data to use in testing a model.

The leave one out method

Computing model accuracy

For each data point *Dj*:

- Compute the model parameters with all the other data points
- Calculate the prediction accuracy for *Dj*

The average prediction accuracy is used as an estimate of the accuracy of the model trained on all the data.

Cross validation

Leave one out is computationally expensive. Cross validation reduces the computation costs.

Divide the data into *k* similarly sized subsets

For each subset

- Compute the model parameters using all the other subsets
- Find the average prediction accuracy for the subset

Hold out methods can be used to choose between competing models.

Bootstrapping

Bootstrapping is a method that can be used to estimate statistical properties from a finite data set.

For example consider a data set in two variables.

One statistic we may be interested in is the mutual information:

$$Dep(X,Y) = P(X\&Y) \log_2(P(X\&Y)/(P(X)P(Y)))$$

Computing the Mutual Entropy (revision)

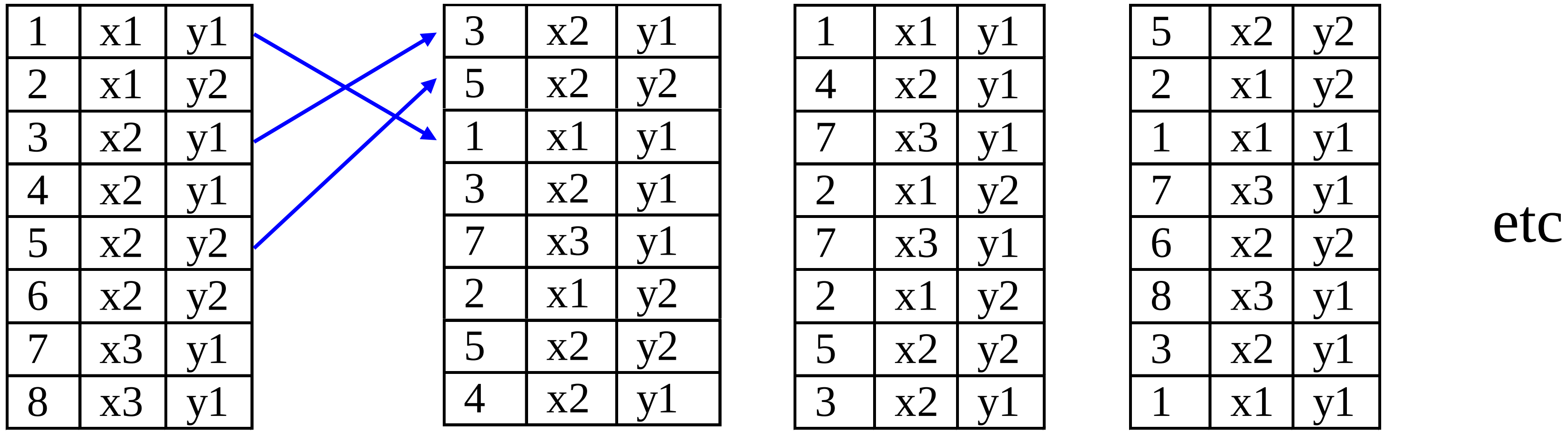
We used mutual entropy to find the maximum weighted spanning tree as follows:

- Compute the X-Y co-occurrence matrix
- Normalise the matrix to form the joint probability matrix P(X&Y).
- Marginalise P(X&Y) to find P(X) and P(Y)
- Calculate the mutual entropy of X and Y

But this only gives us one value.

Bootstrap data sets

Given a data set with *m* data points, a bootstrap data set is a data set of *m* points chosen at random with replacement from the original data set.



1	x1	y1
2	x1	y2
3	x2	y1
4	x2	y1
5	x2	y2
6	x2	y2
7	x3	y1
8	x3	y1

Original Data Set

3	x2	y1
5	x2	y2
1	x1	y1
3	x2	y1
7	x3	y1
2	x1	y2
5	x2	y2
4	x2	y1

Bootstrap 1

1	x1	y1
4	x2	y1
7	x3	y1
2	x1	y2
7	x3	y1
2	x1	y2
5	x2	y2
3	x2	y1

Bootstrap 2

5	x2	y2
2	x1	y2
1	x1	y1
7	x3	y1
6	x2	y2
8	x3	y1
3	x2	y1
1	x1	y1

Bootstrap 3

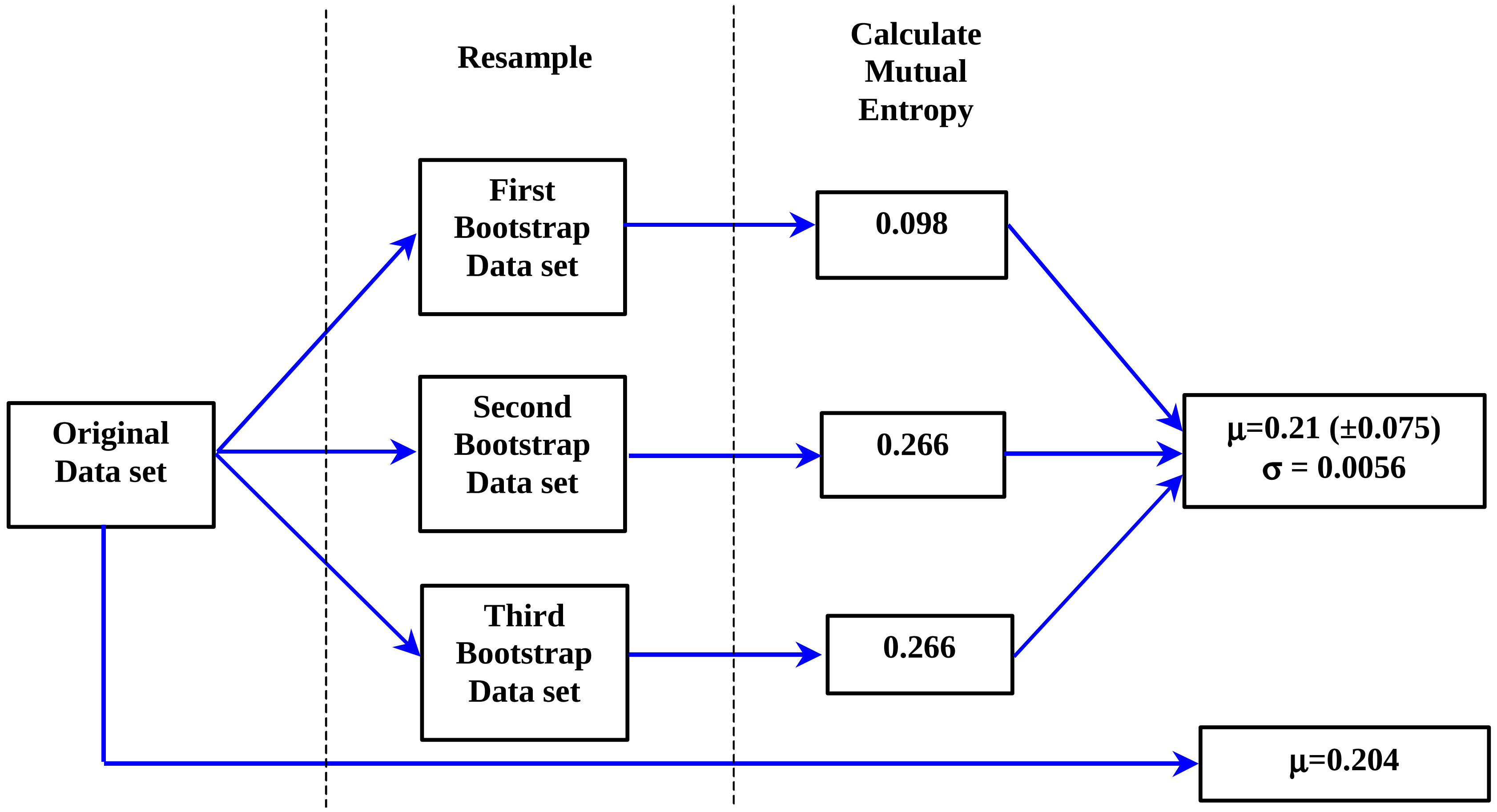
etc

Bootstrapping to find variance

Given a data set D in X-Y

- Select *n* equi-probable bootstrap data sets from D
- Calculate the statistic of interest (Dep(X,Y)) from each bootstrap set
- Find the mean and variance of the estimate

Estimating the variance from the Bootstraps



Bagging: Bootstrap-aggregating

Objective - to reduce the variance component of prediction error.

Method - Create a set of predictors using bootstrap samples of the data set. Aggregate (average) the predictions to get a better estimate.

Aggregating clearly does not affect bias, but does reduce variance.

Boosting:

Bagging creates bootstrap data sets by sampling the original with equal probability.

Boosting changes the probability of selection, eg:

- Sample a bootstrap data set Ti
- Test the data set on Ti
- Increase the probability of selection for misclassified points

Boosting is sometimes called Arcing (Adaptive Resampling and Combining)

Aggregating in General

Bagging and Boosting are both found to reduce the variance of prediction error in simulation studies.

They are therefore proposed as good techniques for building classifiers, particularly with models that suffer from high variance (neural networks).