# PROBABILISTIC GRAPHICAL MODELS IN PYTHON
# **A CRASH COURSE**

Aileen Nielsen

Aileen.A.Nielsen@gmail.com

# BAYESIAN NETWORKS

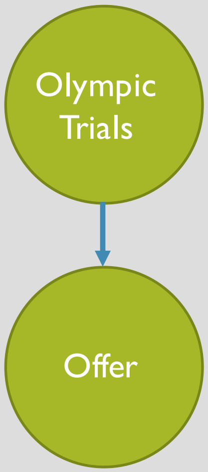
But really… **A CRASH COURSE**

# WHAT ARE THEY AND WHY SHOULD I USE THEM?

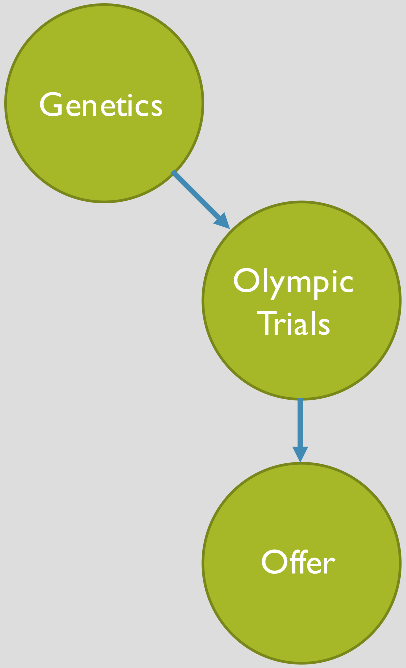
# WHAT THEY ARE

Offer

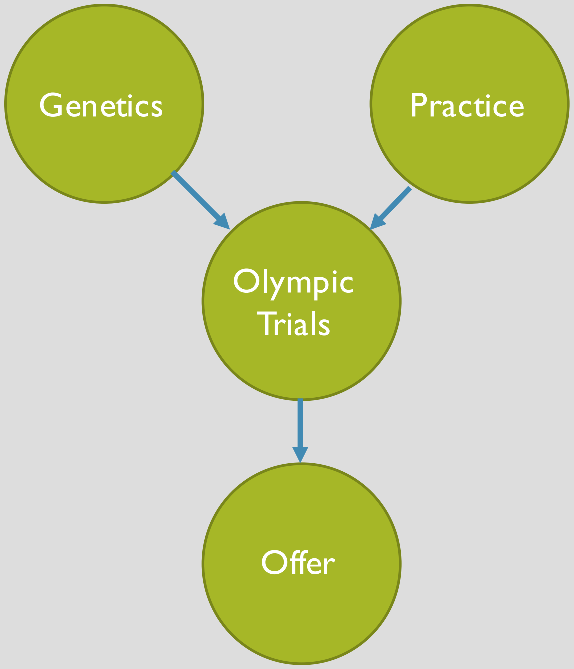
# WHAT THEY ARE



# WHAT THEY ARE

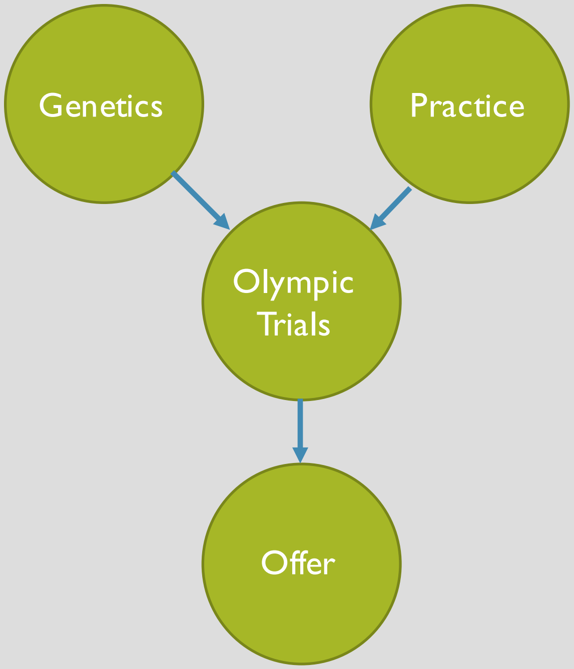


# WHAT THEY ARE



# WHAT THEY ARE

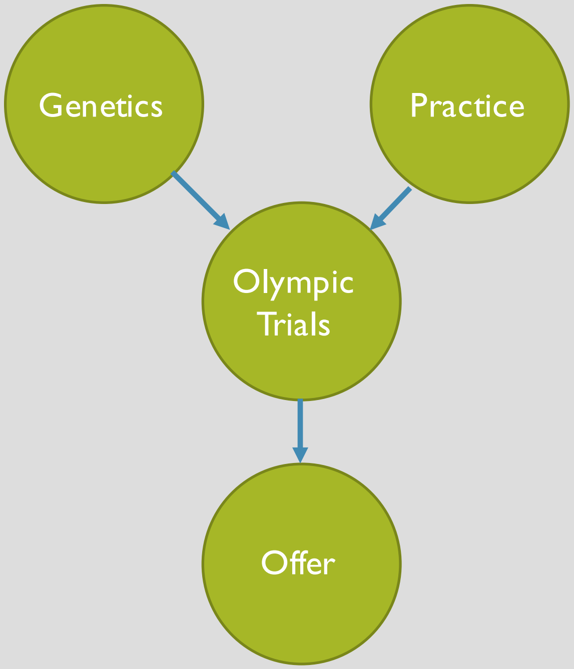
- Use what you know about direct relationships between variables to get information about more complicated relationships



# WHAT THEY ARE

- Use what you know about direct relationships between variables to get information about more complicated relationships

- Store your data/probabilities in a form where it's easy to input evidence and get out updated information



# WHAT THEY'RE NOT

- This is not your intro-CS algorithms graph traversal class all over again

# WHAT THEY'RE NOT

- This is not your intro-CS algorithms graph traversal class all over again
- This is not the same as a network

# WHAT THEY'RE NOT

- This is not your intro-CS algorithms graph traversal class all over again
- This is not the same as a network
- Key word is **probabilistic** graphical models

# WHAT THEY'RE NOT

- This is not your intro-CS algorithms graph traversal class all over again
- This is not the same as a network
- Key word is **probabilistic** graphical models
- Think Markov

# COMMON APPLICATIONS

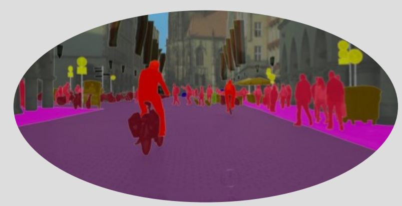
- Medical diagnostics – model how a physician thinks

http://www.almdiagnostics.com/
http://www.markus-enzweiler.de/docs/datasets.html
http://www.slideshare.net/priberam/introducing-priberam-labs-machine-learning-and-natural-language-processing

# COMMON APPLICATIONS

- Medical diagnostics – model how a physician thinks

- Image processing – labeling pixels with information about their neighbors

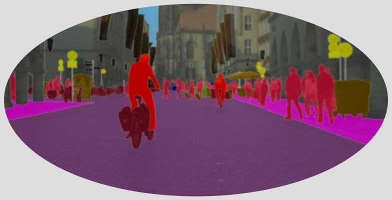


http://www.almdiagnostics.com/
http://www.markus-enzweiler.de/docs/datasets.html
http://www.slideshare.net/priberam/introducing-priberam-labs-machine-learning-and-natural-language-processing

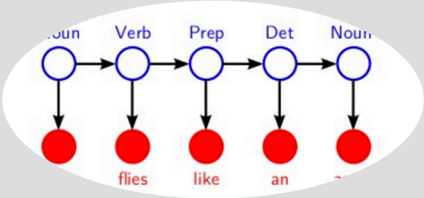
# COMMON APPLICATIONS

- Medical diagnostics – model how a physician thinks

- Image processing – labeling pixels with information about their neighbors



- Natural language processing

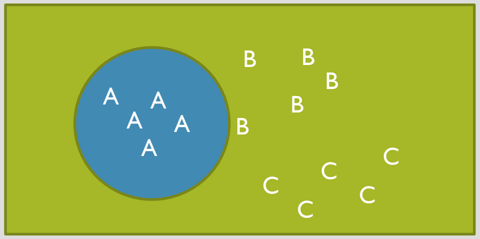



http://www.almdiagnostics.com/
http://www.markus-enzweiler.de/docs/datasets.html
http://www.slideshare.net/priberam/introducing-priberam-labs-machine-learning-and-natural-language-processing

# BASIC CONCEPTS

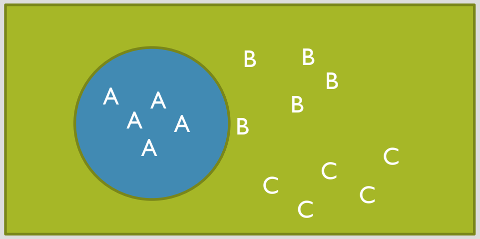
# PROBABILITY

- What is the probability of event A happening?



# PROBABILITY

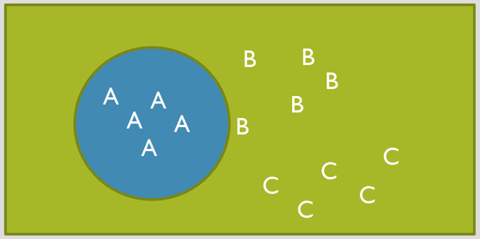
- What is the probability of event A happening?




- Basic idea: Count up all the A's and divide that by the total number of possibilities:

# PROBABILITY

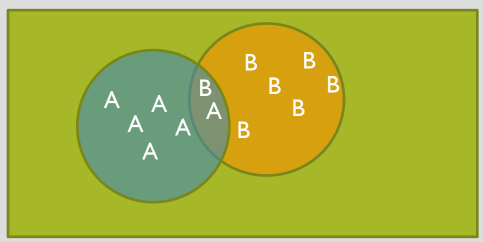
- What is the probability of event A happening?




- Basic idea: Count up all the A's and divide that by the total number of possibilities:
- P(A) = (# A)/(#A + #B + #C)

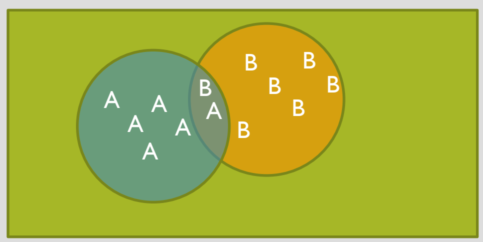
# CONDITIONAL PROBABILITY

- The real world is not so discrete. Multiple things of interest can happen simultaneously.



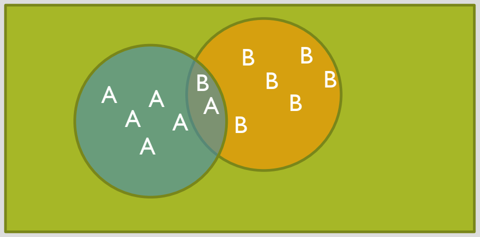
# CONDITIONAL PROBABILITY

- What is the probability of A & B?



# CONDITIONAL PROBABILITY

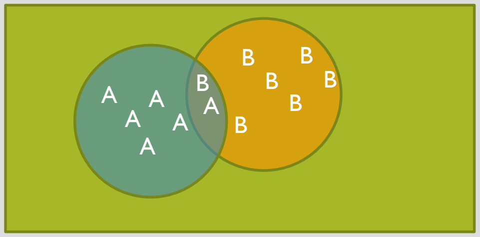
- What is the probability of A & B?




- What needs to happen for A & B to happen? Let's say first A needs to happen, and then with A already happening, B needs to happen

# CONDITIONAL PROBABILITY

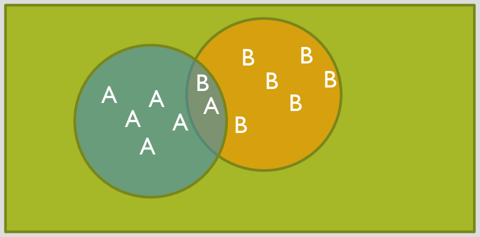
- What is the probability of A & B?




- What needs to happen for A & B to happen? Let's say first A needs to happen, and then with A already happening, B needs to happen
- We write this as P(A) * P(B|A)

# CONDITIONAL PROBABILITY

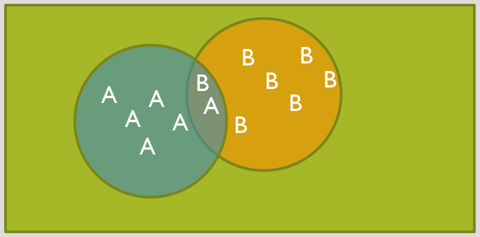
- What is the probability of A & B?




- What needs to happen for A & B to happen? Let's say first A needs to happen, and then with A already happening, B needs to happen
- We write this as P(A) * P(B|A)
- We can also write this as P(B) * P(A|B)

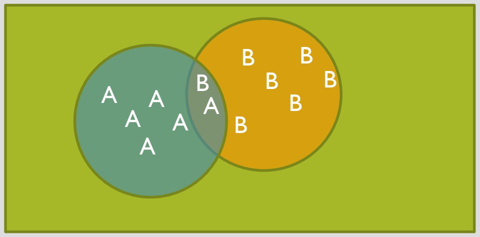
# BAYESIAN PROBABILITY

- What we have discovered is a rule for finding what is ordinarily more interesting: P(One Event | Another Event)



# BAYESIAN PROBABILITY

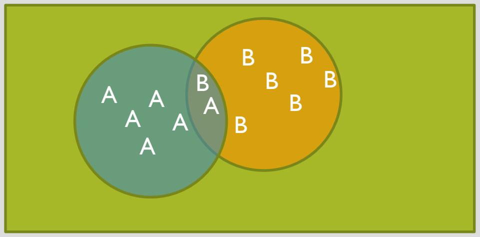
- What we have discovered is a rule for finding what is ordinarily more interesting: P(One Event | Another Event)




- We said P(A & B) = P(A) * P(B|A)

# BAYESIAN PROBABILITY

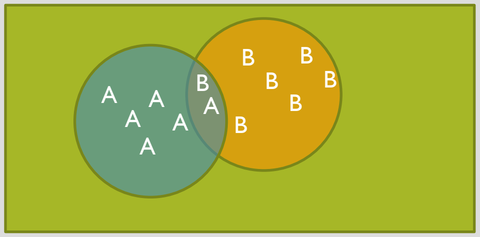
- What we have discovered is a rule for finding what is ordinarily more interesting: P(One Event | Another Event)




- We said P(A & B) = P(A) * P(B|A)
- We also said P(A&B) = P(B) * P(A|B)

# BAYESIAN PROBABILITY

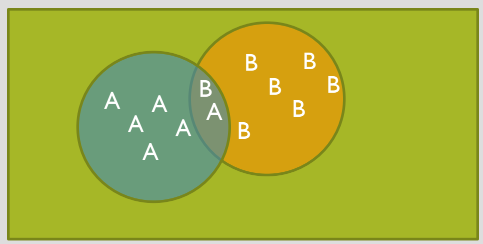
- What we have discovered is a rule for finding what is ordinarily more interesting: P(One Event | Another Event)




- We said P(A & B) = P(A) * P(B|A)
- We also said P(A&B) = P(B) * P(A|B)
- We can isolate either P(A|B) or P(B|A) and compute from simpler probabilities

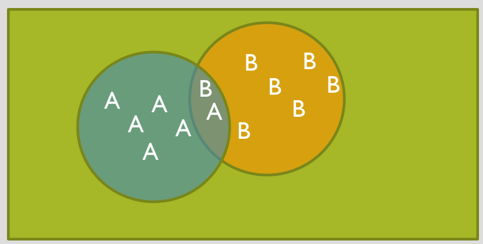
# BAYES' THEOREM

$$P(A|B) = P(B|A) * P(A)/P(B)$$



# BAYES' THEOREM

$$P(A|B) = P(B|A) * P(A)/P(B)$$




$$P(B|A) = P(A|B) * P(B)/P(A)$$

# JOINT, CONDITIONAL, & MARGINAL DISTRIBUTIONS

- The **joint probability distribution** describes how two or more variables are distributed *simultaneously*. To get a probability from the joint distribution of A and B, you would consider P(A = a and B = b)

# JOINT, CONDITIONAL, & MARGINAL DISTRIBUTIONS

- The **joint probability distribution** describes how two or more variables are distributed *simultaneously*. To get a probability from the joint distribution of A and B, you would consider P(A = a and B = b)

- The **conditional probability distribution** looks at how the probabilities of A are distributed given a certain value, say, for B: P(A = a| B = b)

# JOINT, CONDITIONAL, & MARGINAL DISTRIBUTIONS

- The **joint probability distribution** describes how two or more variables are distributed *simultaneously*. To get a probability from the joint distribution of A and B, you would consider P(A = a and B = b)

- The **conditional probability distribution** looks at how the probabilities of A are distributed given a certain value, say, for B: P(A = a| B = b)

- The **marginal probability distribution** is a distribution that results from averaging over one variable to get the probability distribution of the other. For example, the marginal probability distribution of A when A and B are related would be given by $P_A(a) = \int_B P(a|b)P(b)db$

# BUT WHERE ARE THE GRAPHICAL MODELS?

# BAYES NETWORK

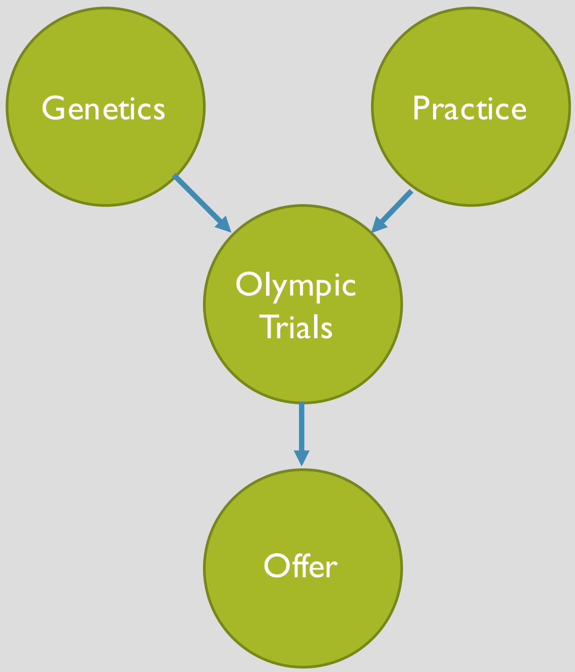
- A Bayes Network is a structure that can be represented as a **directed**, **acyclic** graph.

# BAYES NETWORK

- A Bayes Network is a structure that can be represented as a **directed**, **acyclic** graph.
- The advantages are two-fold:

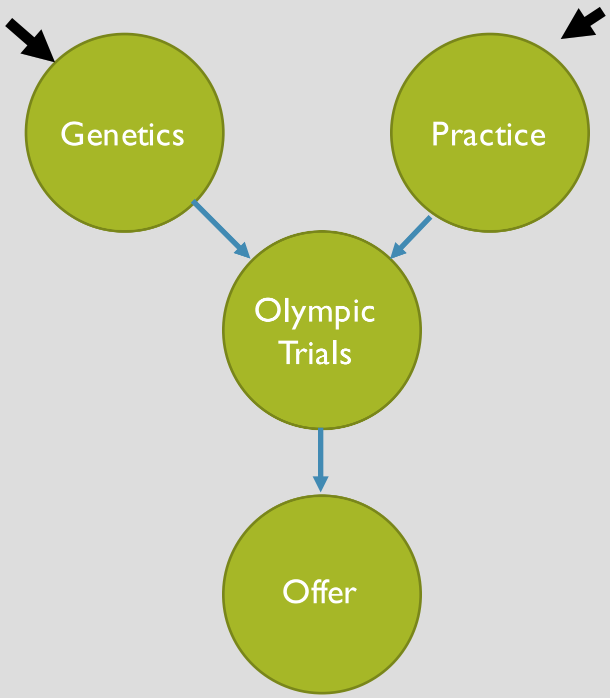
1. Allow a compact representation of the joint distribution from the chain rule for Bayes networks
2. Observe conditional independence relationships between vertices/random variables

# EXAMPLE BAYES NETWORK



# EXAMPLE BAYES NETWORK

| Genes | P(Genes) |
|---|---|
| Good | .2 |
| Bad | .8 |

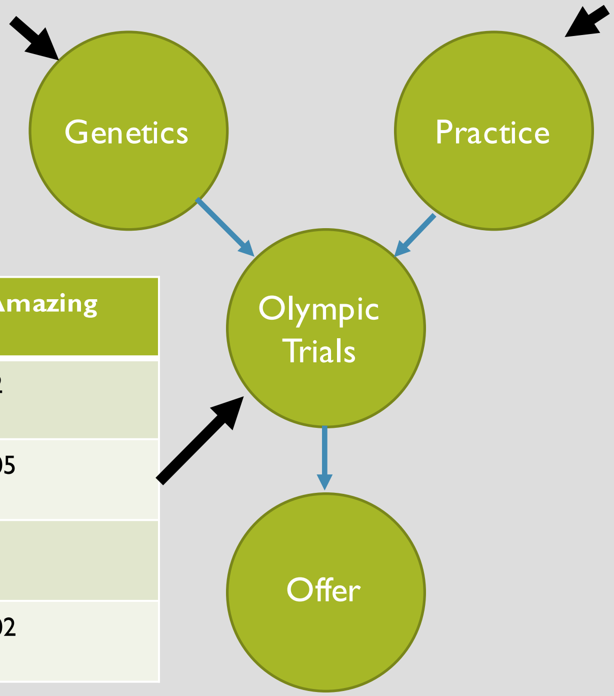



| Practice | P(Practice) |
|---|---|
| Yes | .7 |
| No | .3 |

# EXAMPLE BAYES NETWORK

| Genes | P(Genes) |
|---|---|
| Good | .2 |
| Bad | .8 |

| Practice | P(Practice) |
|---|---|
| Yes | .7 |
| No | .3 |

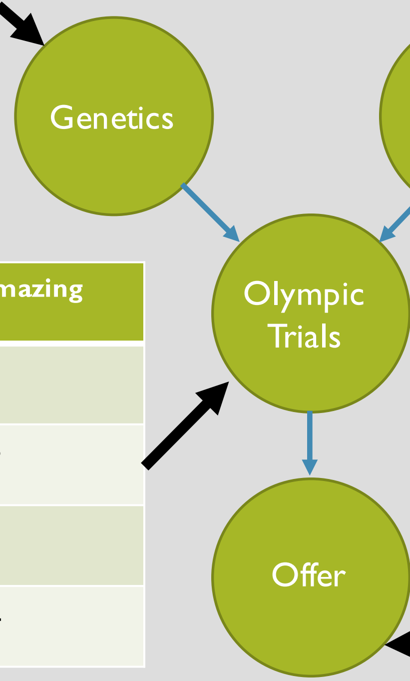



| | Bad | Border -line | Amazing |
|---|---|---|---|
| Good Genes, Did Practice | .5 | .3 | .2 |
| Good Genes, Didn't Practice | .8 | .15 | .05 |
| Bad Genes, Did Practice | .8 | .1 | .1 |
| Bad Genes, Didn't Practice | .9 | .08 | .02 |

# EXAMPLE BAYES NETWORK

| Genes | P(Genes) |
|---|---|
| Good | .2 |
| Bad | .8 |

| Practice | P(Practice) |
|---|---|
| Yes | .7 |
| No | .3 |

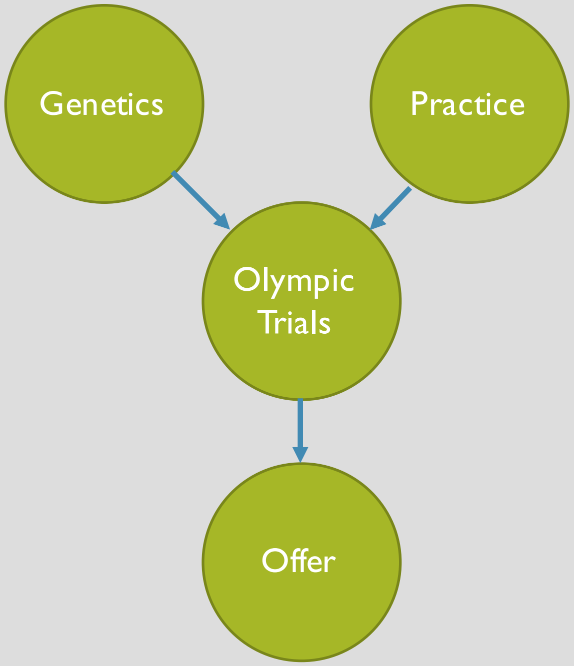



| | Bad | Border -line | Amazing |
|---|---|---|---|
| Good Genes, Did Practice | .5 | .3 | .2 |
| Good Genes, Didn't Practice | .8 | .15 | .05 |
| Bad Genes, Did Practice | .8 | .1 | .1 |
| Bad Genes, Didn't Practice | .9 | .08 | .02 |

| | No Offer | Got Offer |
|---|---|---|
| Bad Olympic Trials | .95 | .05 |
| Borderline Olympic Trials | .8 | .2 |
| Amazing Olympic Trials | .5 | .5 |

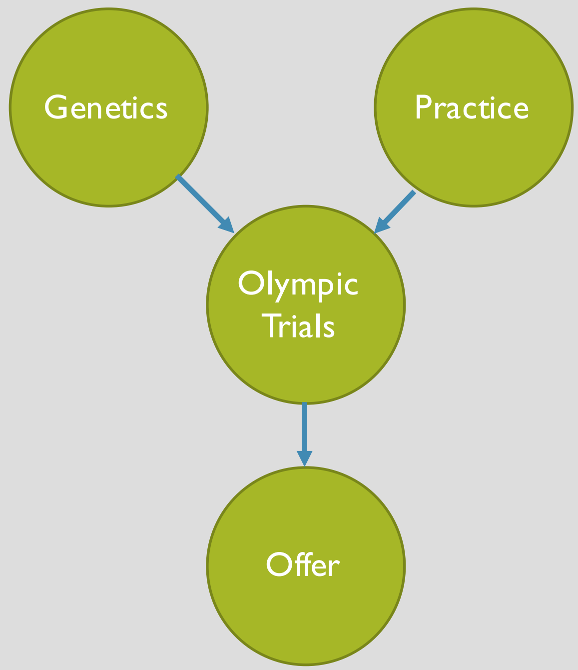
# THINK ABOUT IT

- Does an Offer depend on Genetics?



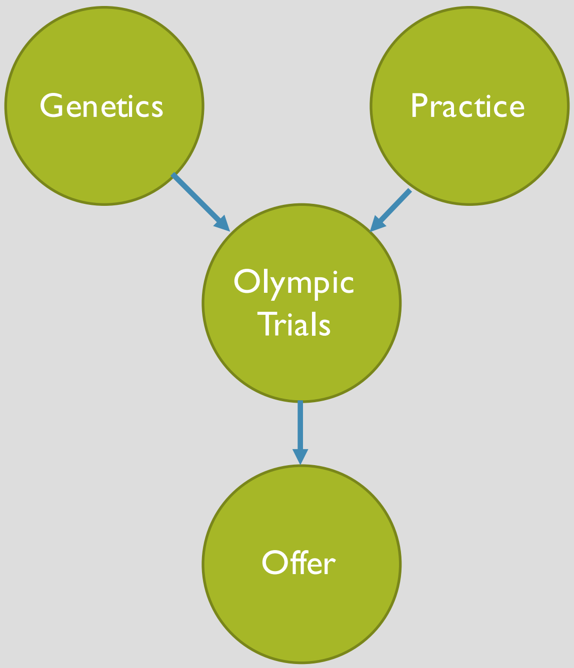
# THINK ABOUT IT

- Does an Offer depend on Genetics?
- Does an Offer depend on Genetics if you know Practice?



# THINK ABOUT IT

- Does an Offer depend on Genetics?
- Does an Offer depend on Genetics if you know Practice?
- Does an Offer depend on Genetics if you know Olympic Trials performance?



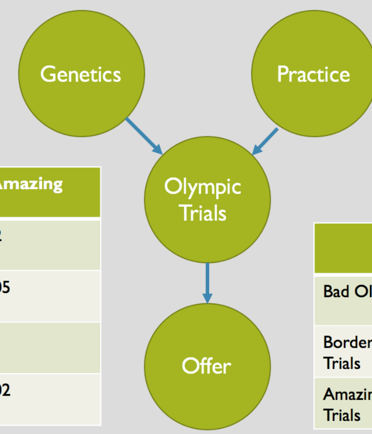
# LET'S SEE HOW IT WORKS…

# CONDITIONAL PROBABILITY DISTRIBUTION (CPD)

- Each node (random variable) in your Bayesian Nework has a CPD associated with it.

| Genes | P(Genes) |
|---|---|
| Good | .2 |
| Bad | .8 |

| Practice | P(Practice) |
|---|---|
| Yes | .7 |
| No | .3 |



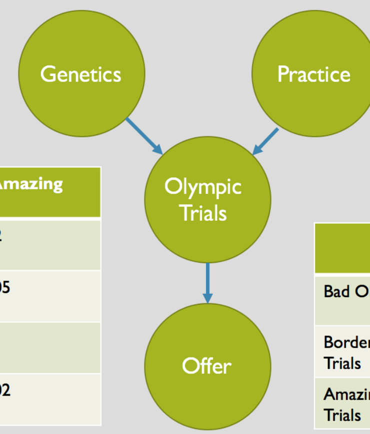

| | Bad | Border -line | Amazing |
|---|---|---|---|
| Good Genes, Did Practice | .5 | .3 | .2 |
| Good Genes, Didn't Practice | .8 | .15 | .05 |
| Bad Genes, Did Practice | .8 | .1 | .1 |
| Bad Genes, Didn't Practice | .9 | .08 | .02 |

| | No Offer | Got Offer |
|---|---|---|
| Bad Olympic Trials | .95 | .05 |
| Borderline Olympic Trials | .8 | .2 |
| Amazing Olympic Trials | .5 | .5 |

# CONDITIONAL PROBABILITY DISTRIBUTION (CPD)

- Each node (random variable) in your Bayesian Nework has a CPD associated with it.
- If a node has parents, the associated CPD represent P(value | parent's value)

| Genes | P(Genes) |
|---|---|
| Good | .2 |
| Bad | .8 |




| Practice | P(Practice) |
|---|---|
| Yes | .7 |
| No | .3 |

| | Bad | Border -line | Amazing |
|---|---|---|---|
| Good Genes, Did Practice | .5 | .3 | .2 |
| Good Genes, Didn't Practice | .8 | .15 | .05 |
| Bad Genes, Did Practice | .8 | .1 | .1 |
| Bad Genes, Didn't Practice | .9 | .08 | .02 |

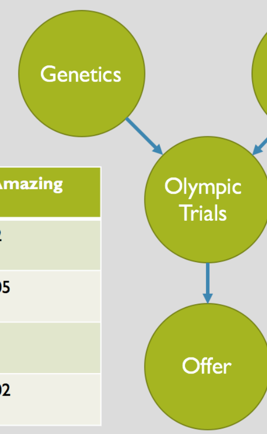
| | No Offer | Got Offer |
|---|---|---|
| Bad Olympic Trials | .95 | .05 |
| Borderline Olympic Trials | .8 | .2 |
| Amazing Olympic Trials | .5 | .5 |

# CONDITIONAL PROBABILITY DISTRIBUTION (CPD)

- Each node (random variable) in your Bayesian Nework has a CPD associated with it.
- If a node has parents, the associated CPD represent P(value | parent's value)
- If a node has no parents, the CPD represents P(value), the unconditional probability of that value

| Genes | P(Genes) |
|---|---|
| Good | .2 |
| Bad | .8 |

| Practice | P(Practice) |
|---|---|
| Yes | .7 |
| No | .3 |




| | Bad | Border-line | Amazing |
|---|---|---|---|
| Good Genes, Did Practice | .5 | .3 | .2 |
| Good Genes, Didn't Practice | .8 | .15 | .05 |
| Bad Genes, Did Practice | .8 | .1 | .1 |
| Bad Genes, Didn't Practice | .9 | .08 | .02 |

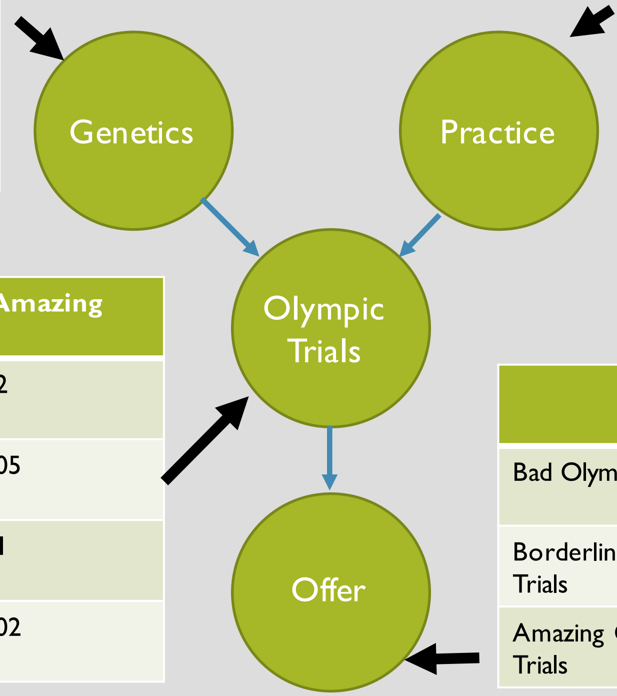
| | No Offer | Got Offer |
|---|---|---|
| Bad Olympic Trials | .95 | .05 |
| Borderline Olympic Trials | .8 | .2 |
| Amazing Olympic Trials | .5 | .5 |

# LET'S LOOK AT HOW TO DO THIS WITH **PGMPY**

1. Define network structure
2. Add network parameters
3. Go!

# FOR REFERENCE

| Genes | P(Genes) |
|---|---|
| Good | .2 |
| Bad | .8 |




| Practice | P(Practice) |
|---|---|
| Yes | .7 |
| No | .3 |

| | Bad | Border-line | Amazing |
|---|---|---|---|
| Good Genes, Did Practice | .5 | .3 | .2 |
| Good Genes, Didn't Practice | .8 | .15 | .05 |
| Bad Genes, Did Practice | .8 | .1 | .1 |
| Bad Genes, Didn't Practice | .9 | .08 | .02 |

| | No Offer | Got Offer |
|---|---|---|
| Bad Olympic Trials | .95 | .05 |
| Borderline Olympic Trials | .8 | .2 |
| Amazing Olympic Trials | .5 | .5 |

# LET'S COMPARE SYNTAXES

# LIBPGM
## (PYTHON 2, CTS DISTRIBUTIONS)

**Input structure:**

```
{ "V": ["Letter", "Grade",
"Intelligence", "SAT", "Difficulty"],
"E": [["Intelligence", "Grade"],
["Difficulty", "Grade"],
["Intelligence", "SAT"], ["Grade",
"Letter"]], "Vdata": { "Letter": {
"ord": 4, "numoutcomes": 2, "vals":
["weak", "strong"], "parents":
["Grade"], "children": None, "cprob":
{ "['A']": [.1, .9], "['B']": [.4,
.6], "['C']": [.99, .01] } },
```

***Sample from graph:***

```
# load nodedata and graphskeleton
nd = NodeData()
skel = GraphSkeleton()
nd.load("../tests/unittestdict.txt")
# any input file
skel.load("../tests/unittestdict.txt")
 # topologically order graphskeleton
skel.toporder()
# load bayesian network
bn = DiscreteBayesianNetwork(skel, nd)
# sample result = bn.randomsample(10)
```

**Get conditional probability given data:**

```
bn.specificquery(dict(Offer='1'),
dict=(Grades='0'))
```

**Learn structure:**

```
learner = PGMLearner()
data = bn.randomsample(200)
result =
learner.discrete_consraint_estimates
tructure(data)
```

# POMEGRANATE
## (PYTHON 3, DISCRETE DISTRIBUTIONS)

**Input structure part 1:**

```
# The guests initial door selection is completely random
guest = DiscreteDistribution( { 'A': 1./3, 'B': 1./3,
'C': 1./3 } )
# The door the prize is behind is also completely random
prize = DiscreteDistribution( { 'A': 1./3, 'B': 1./3,
'C': 1./3 } )
# Monty is dependent on both the guest and the prize.
monty = ConditionalProbabilityTable(
[[ 'A', 'A', 'A', 0.0 ],
[ 'A', 'A', 'B', 0.5 ],
[ 'A', 'A', 'C', 0.5 ],
…
```

**Input structure part 2:**

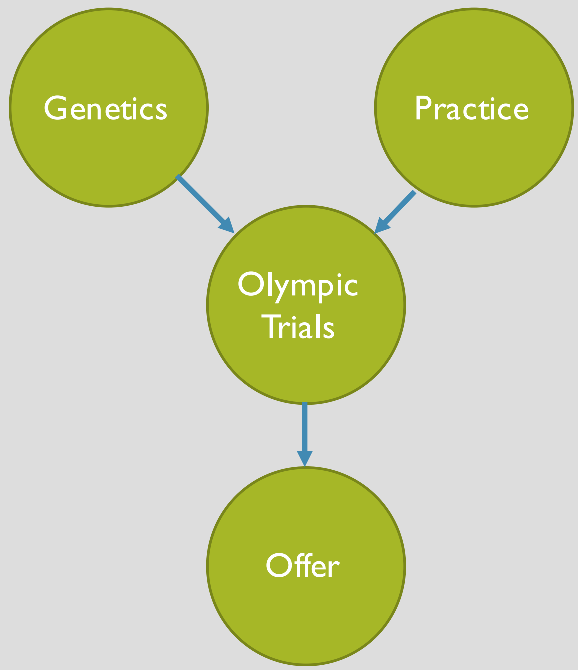
```
# State objects hold both the distribution, and a high
level name.
s1 = State( guest, name="guest" )
s2 = State( prize, name="prize" )
s3 = State( monty, name="monty" )


# Create the Bayesian network object with a useful name
network = BayesianNetwork( "Monty Hall Problem" )
# Add the three states to the network
network.add_states( [ s1, s2, s3 ] )
# Add transitions which represent conditional
dependencies, where the second node is conditionally
dependent on the first node (Monty is dependent on both
guest and prize) network.add_transition( s1, s3 )
network.add_transition( s2, s3 ) network.bake()
```

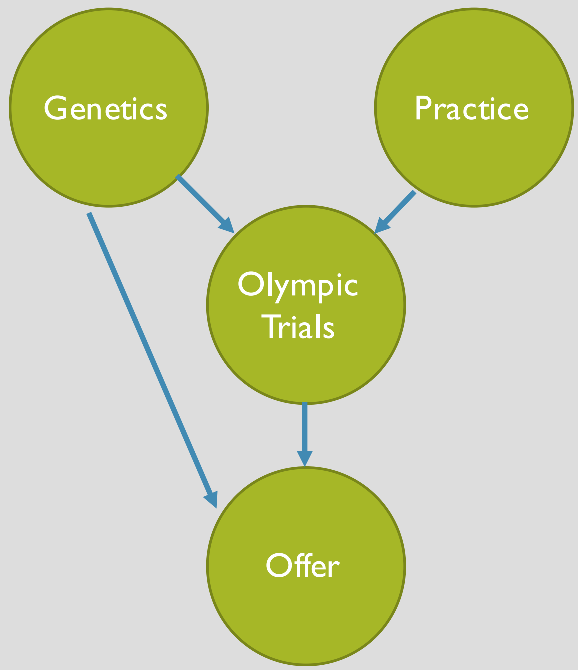
**Map out conditional probabilities:**

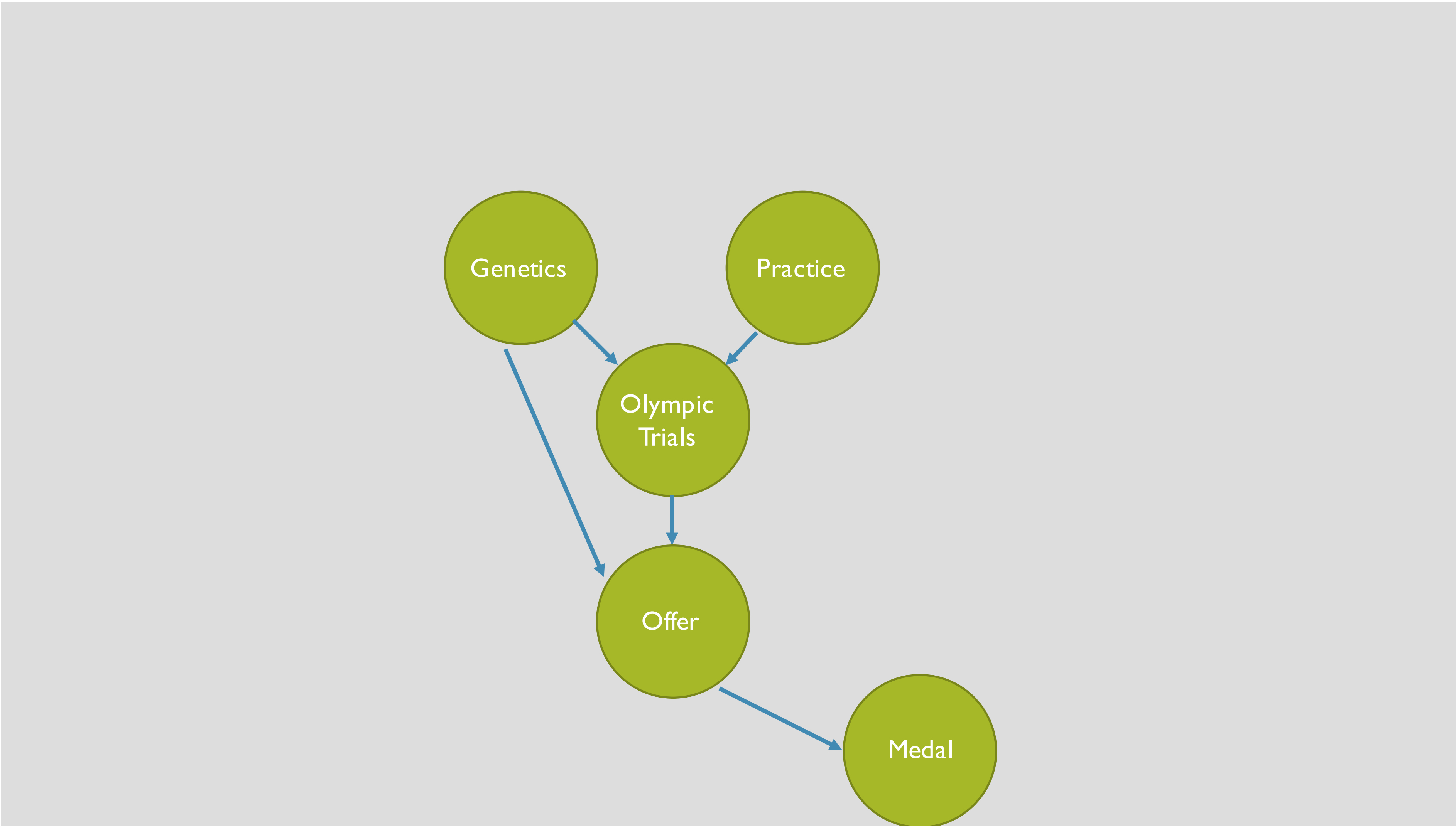
```
observations = { 'guest' : 'A' }

# beliefs will be an array of
posterior distributions or clamped
values for each state, indexed
corresponding to the order
# in self.states.
beliefs = network.forward_backward(
observations )
# Convert the beliefs into a more
readable format
beliefs = map( str, beliefs ) #
Print out the state name and belief
for each state on individual lines
print "\n".join( "{}\t{}".format(
state.name, belief ) for state,
belief in zip( network.states,
beliefs ) )
```

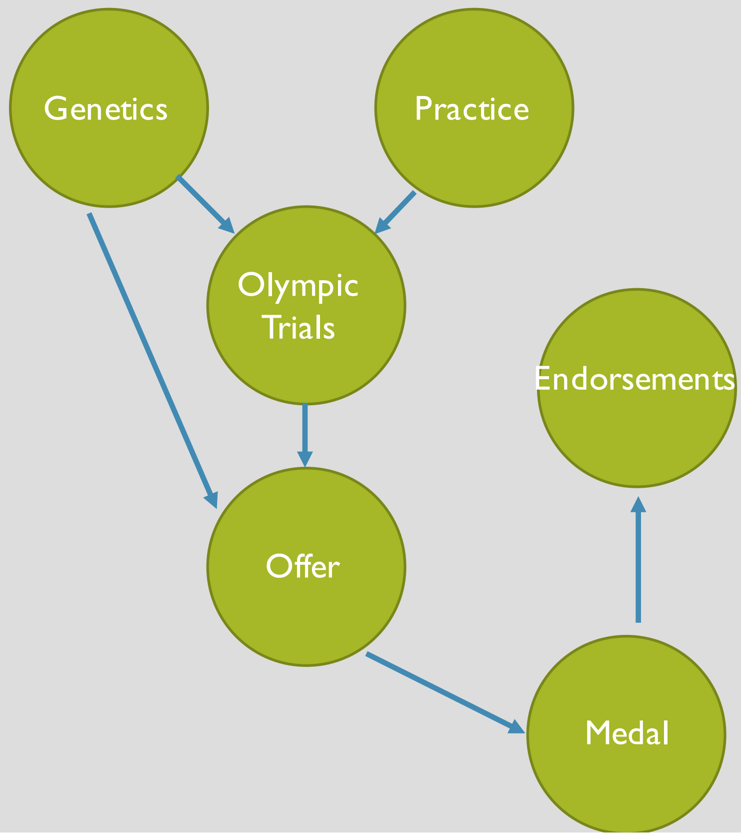
https://github.com/jmschrei/pomegranate

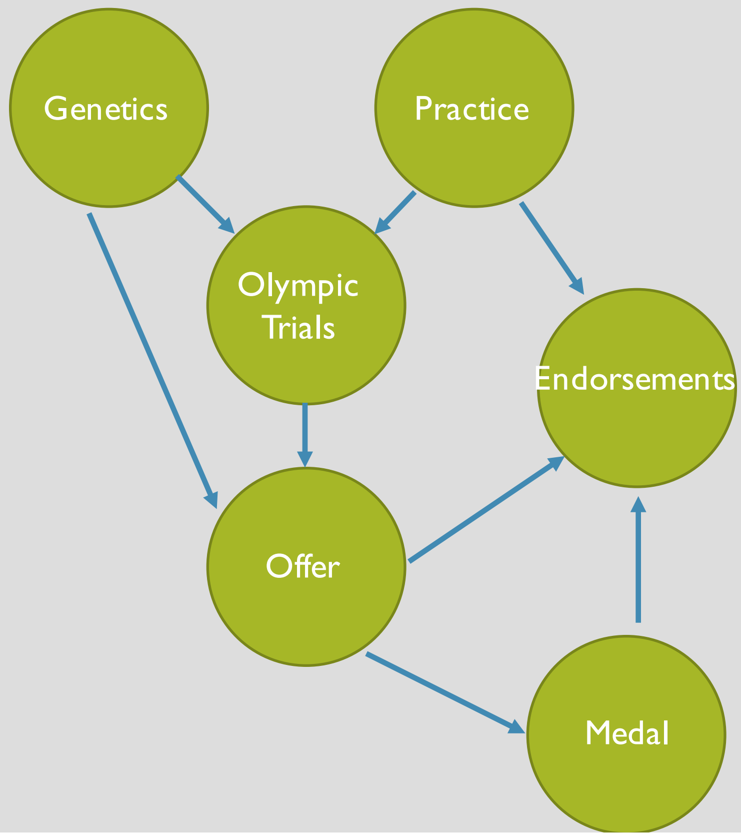
# WHY USE THEM?

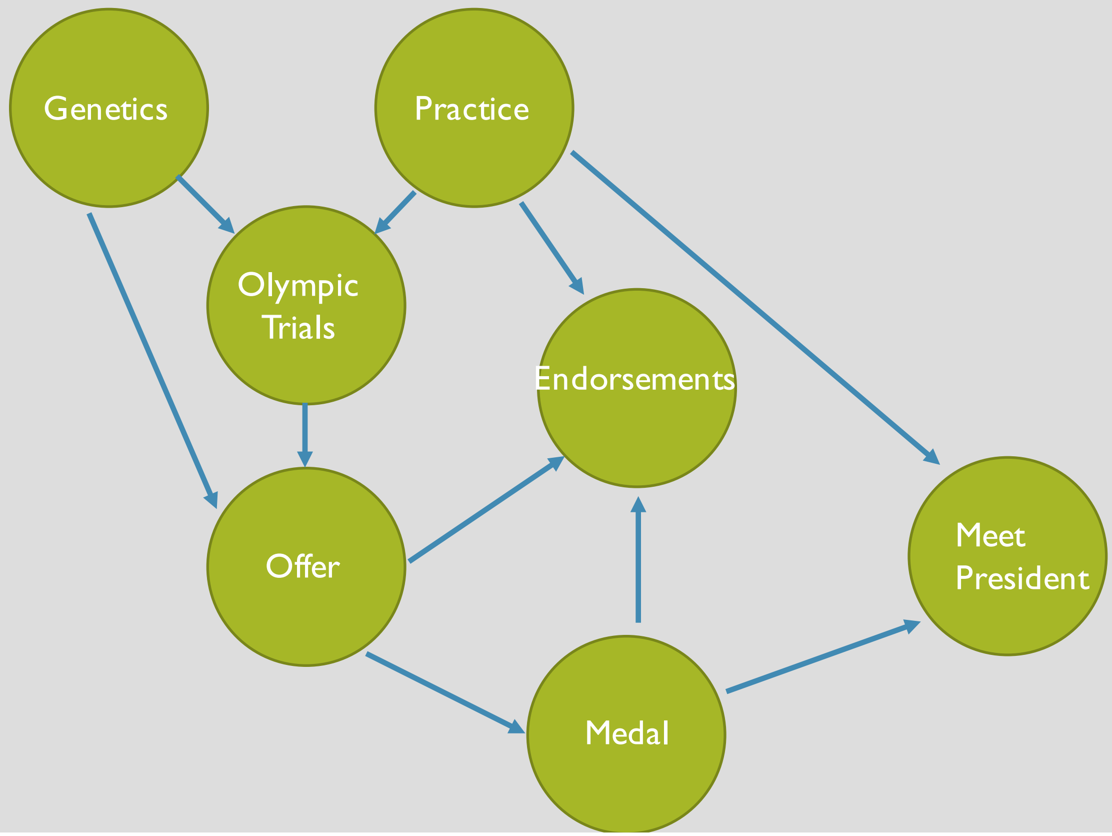

Genetics
Practice
Olympic Trials
Offer

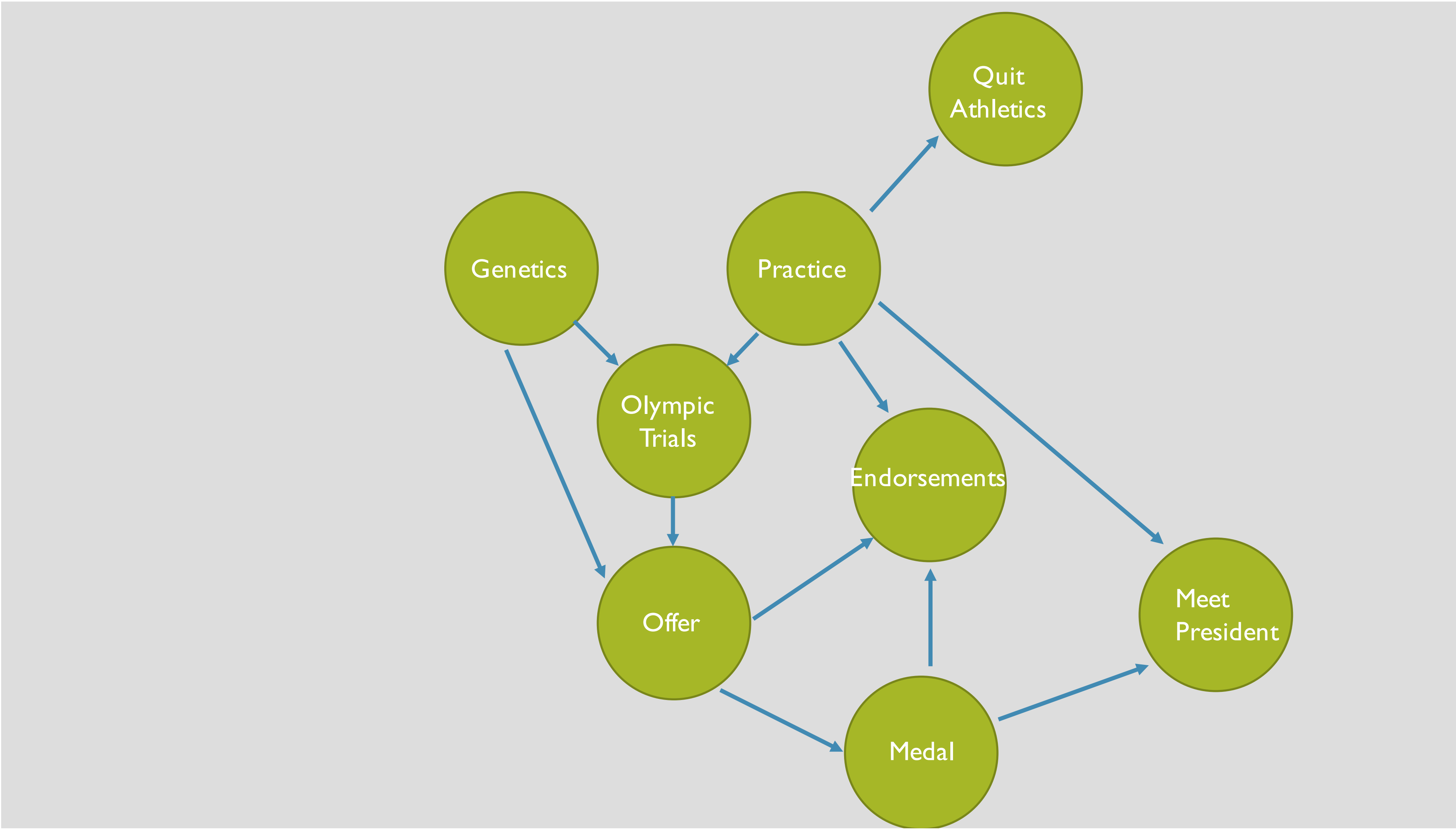

Genetics
Practice
Olympic Trials
Offer


Genetics
Practice
Olympic Trials
Offer
Medal

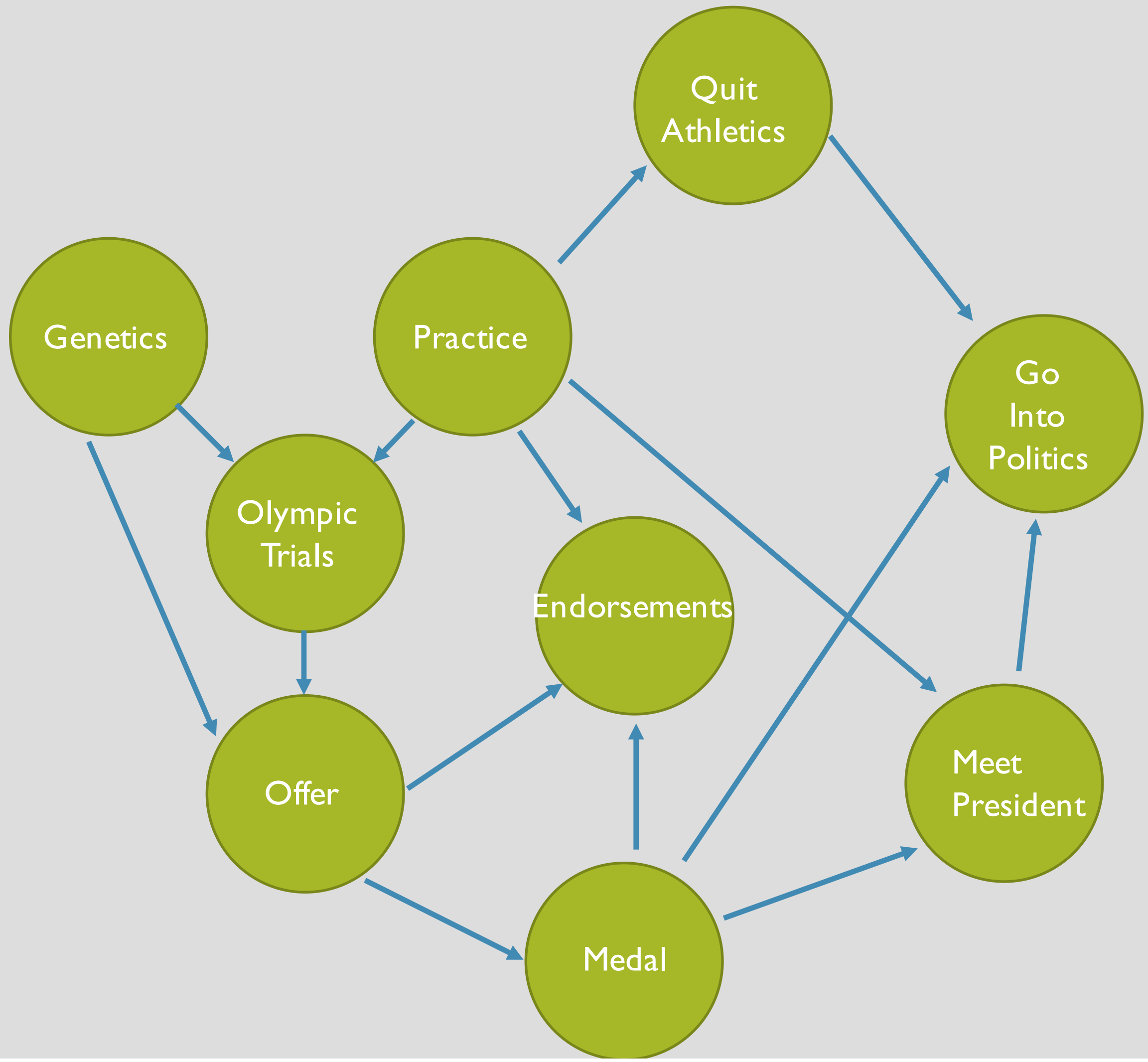

Genetics
Practice
Olympic Trials
Endorsements
Offer
Medal


Genetics
Practice
Olympic Trials
Endorsements
Offer
Medal


Genetics
Practice
Olympic Trials
Endorsements
Offer
Meet President
Medal



Quit Athletics
Genetics
Practice
Olympic Trials
Endorsements
Offer
Meet President
Medal


Quit Athletics
Genetics
Practice
Go Into Politics
Olympic Trials
Endorsements
Offer
Meet President
Medal

# WHY?

- You can input what you know, or intuit, and let the models tell you what you don't know…or don't feel like calculating

# WHY?

- You can input what you know, or intuit, and let the models tell you what you don't know…or don't feel like calculating
- You can extract probabilities from your data if you have an underlying intuition for the domain structure

# WHY?

- You can input what you know, or intuit, and let the models tell you what you don't know…or don't feel like calculating
- You can extract probabilities from your data if you have an underlying intuition for the domain structure
- You can extract structure and your probabilities given lots of computing power, though this is suboptimal

# LEARN MORE

- Markov Networks (undirected)

# LEARN MORE

- Markov Networks (undirected)

- Graphical Models are increasingly used by the Python data analysis community. Many young open source projects are actively seeking contributors and adding new features now.

# LEARN MORE

- Markov Networks (undirected)

- Graphical Models are increasingly used by the Python data analysis community. Many young open source projects are actively seeking contributors and adding new features now.

- There's a whole vocabulary and set of symbolic logical rules to better understand how graphs work – it's generally a good idea to learn these if you want to work more with python graphical models

THANK YOU