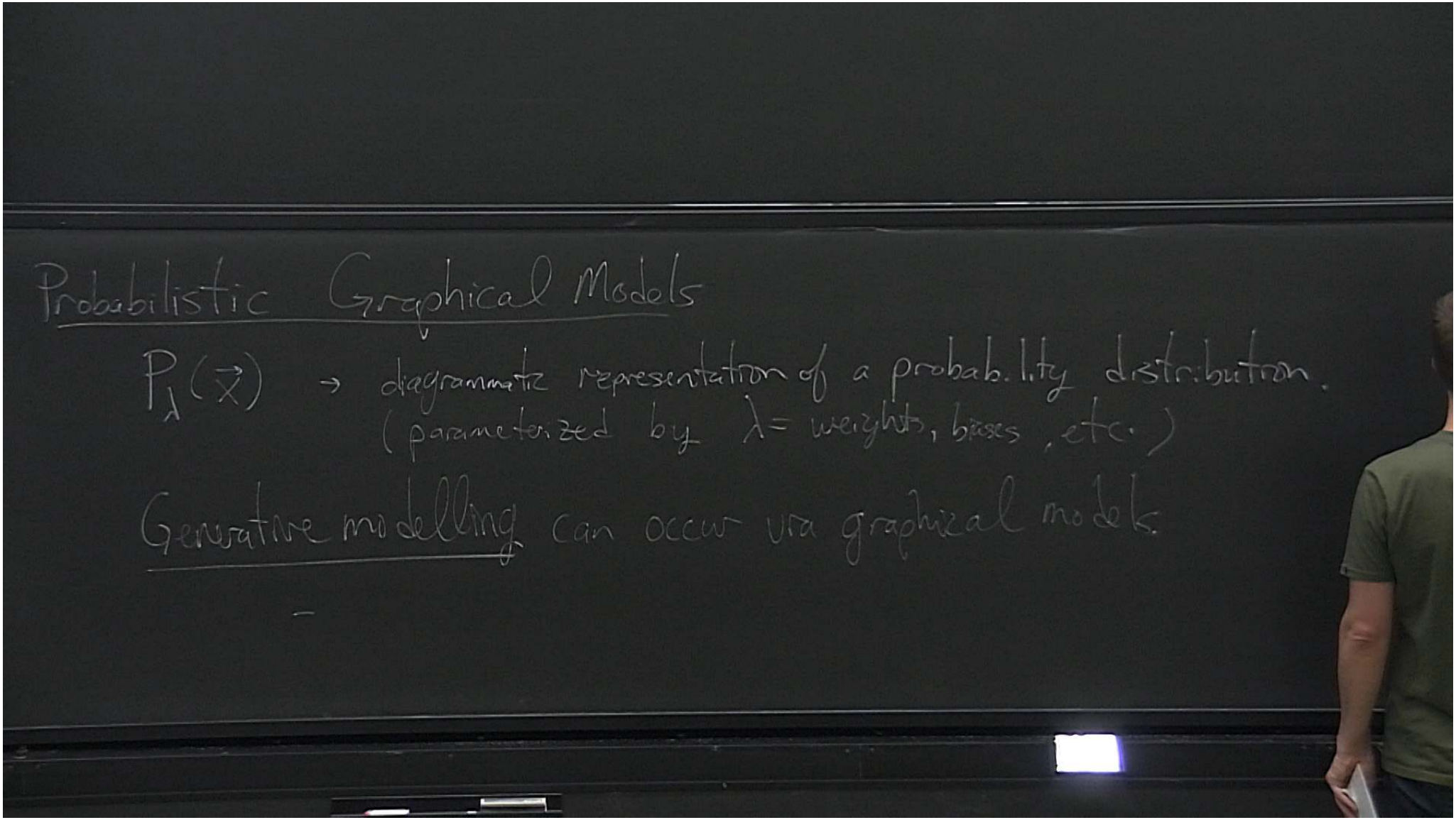
Title: PSI 2017/2018 - Machine Learning for Many Body Physics - Lecture 10

Date: Apr 19, 2018 03:45 PM

URL: http://pirsa.org/18040071

Abstract:

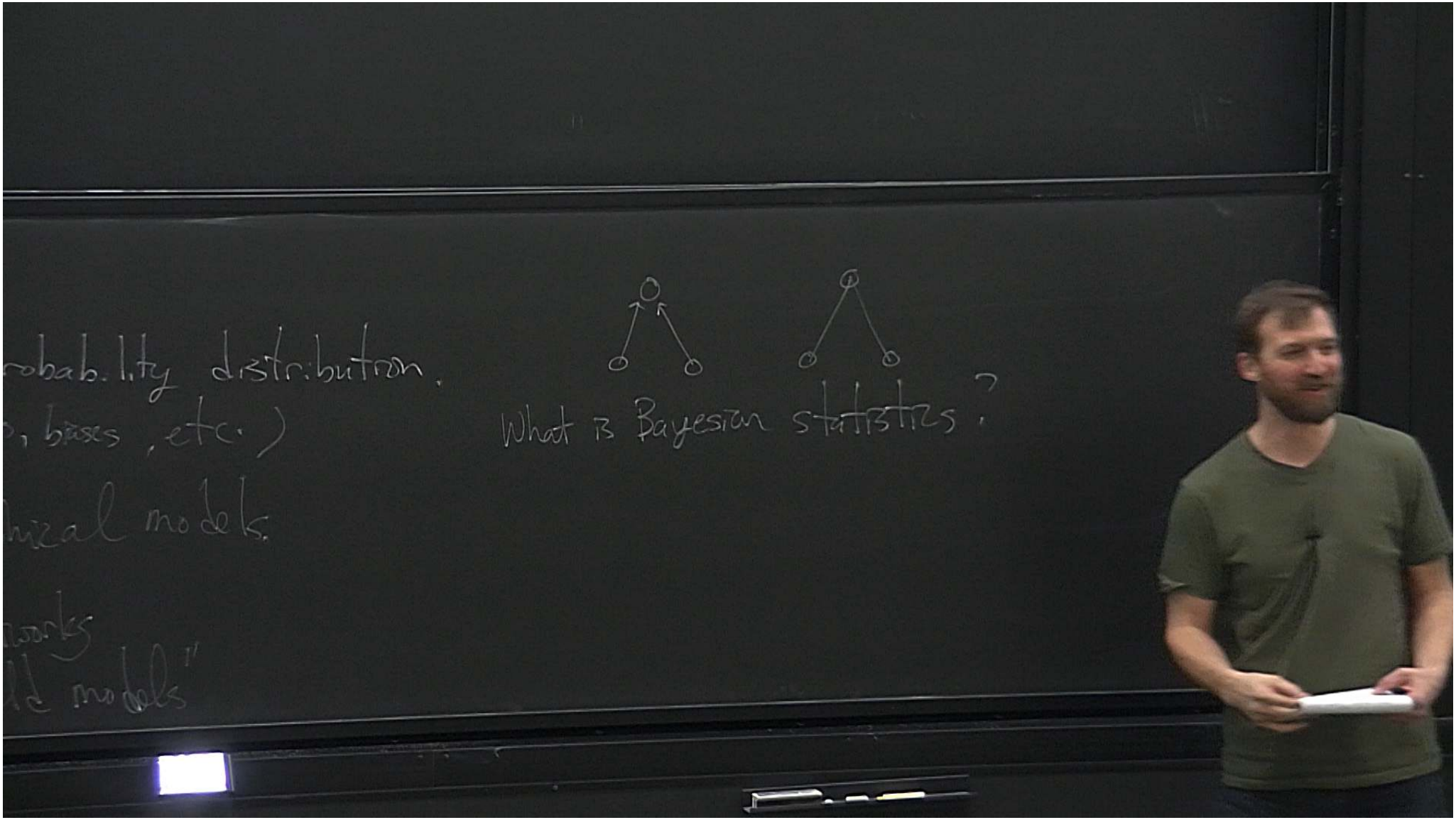

Probabilistic Graphical Models
$P_\lambda(\vec{x})$ → diagrammatic representation of a probability distribution.
(parameterized by λ = weights, biases, etc.)
Generative modelling can occur via graphical models.

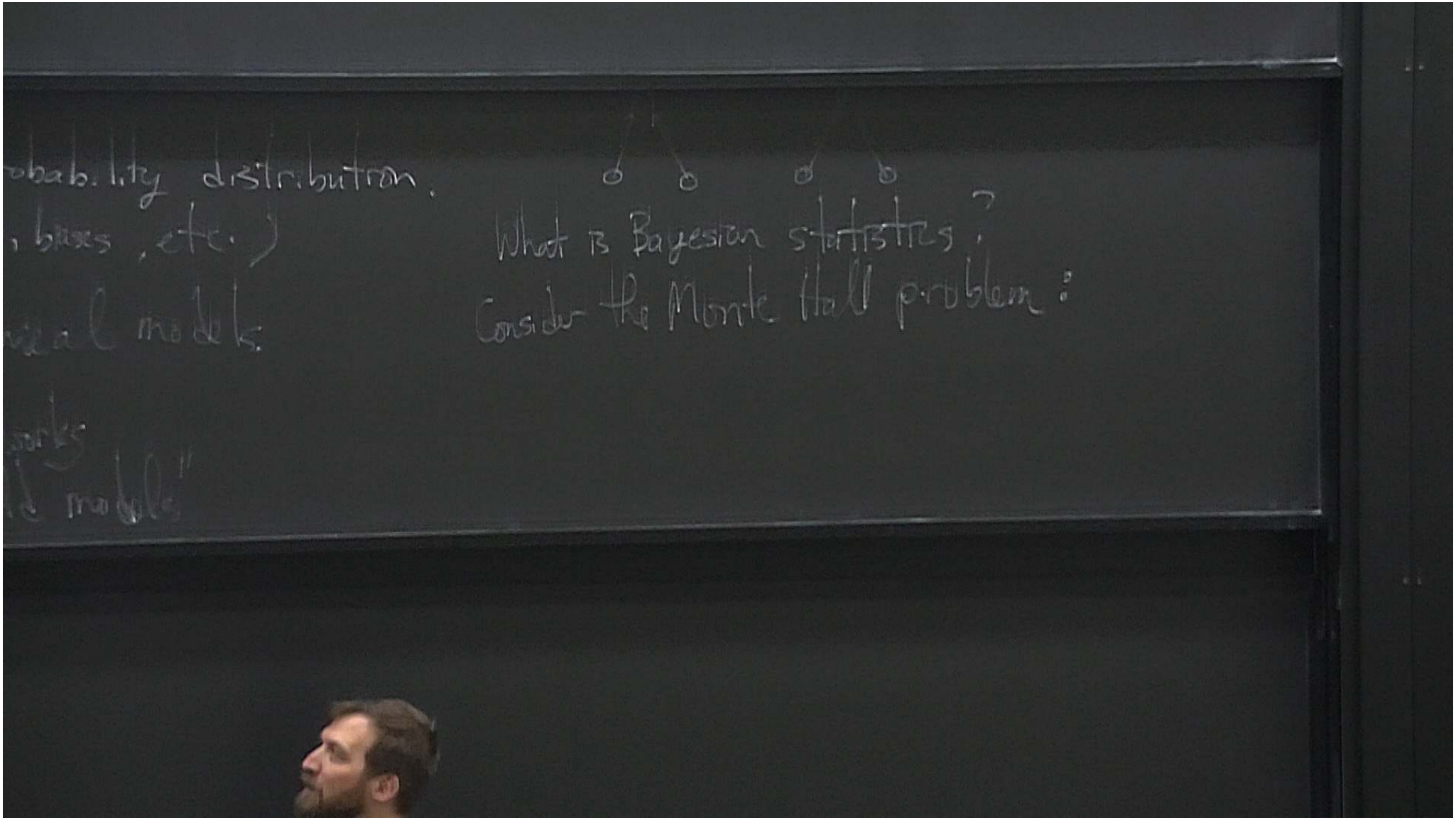
Probabilistic Graphical Models

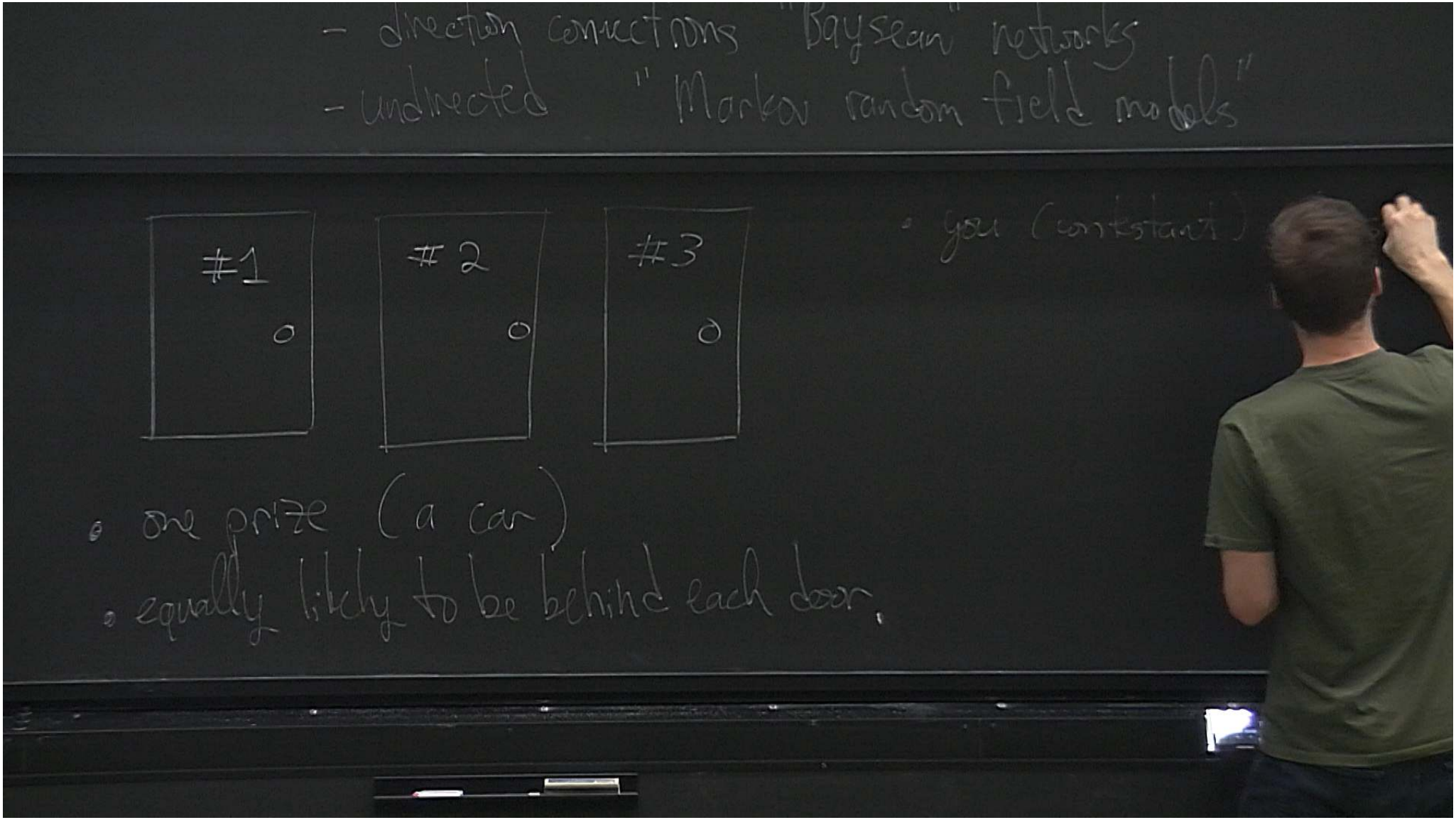
$P_\lambda(\vec{x})$ → diagrammatic representation of a probability distribution.
(parameterized by λ = weights, biases, etc.)

Generative modelling can occur via graphical models:

- directed connections "Bayesian" networks
- undirected "Markov random field models"

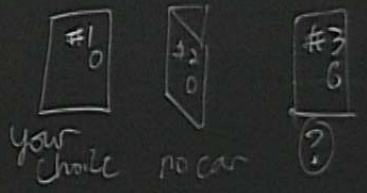

robability distribution.
, biases, etc.)
hical models.
works
ld models"
What is Bayesian statistics?


obab. lity distr. butron.
, bases, etc.)
What is Bayesian statistics?
Consider the Monte Hall problem:


- directed connections "Bayesian" networks
- undirected "Markov random field models"
#1
#2
#3
• you (contestant)
• one prize (a car)
• equally likely to be behind each door.

ysean networks"
random field models"

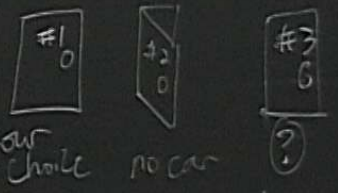
- you (contestant) chooses a door (say door #1)
- Me (the host) reveals a non-selected door that does not contain the car. (say door #2)



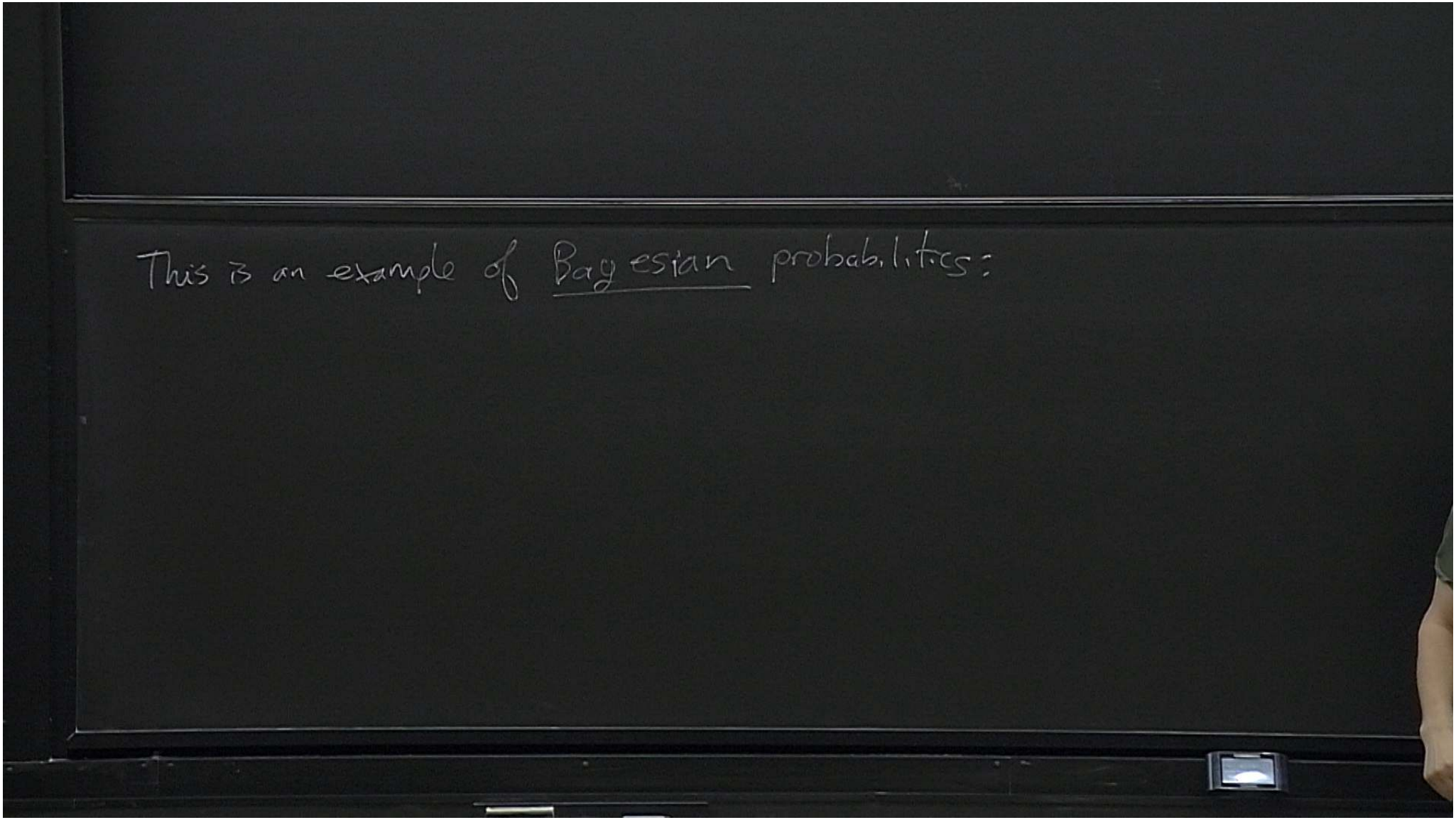
or.

ysean networks
ndom field models"

- you (contestant) chooses a door (say door #1)
- Me (the host) reveals a non-selected door that does not contain the car. (say door #2)

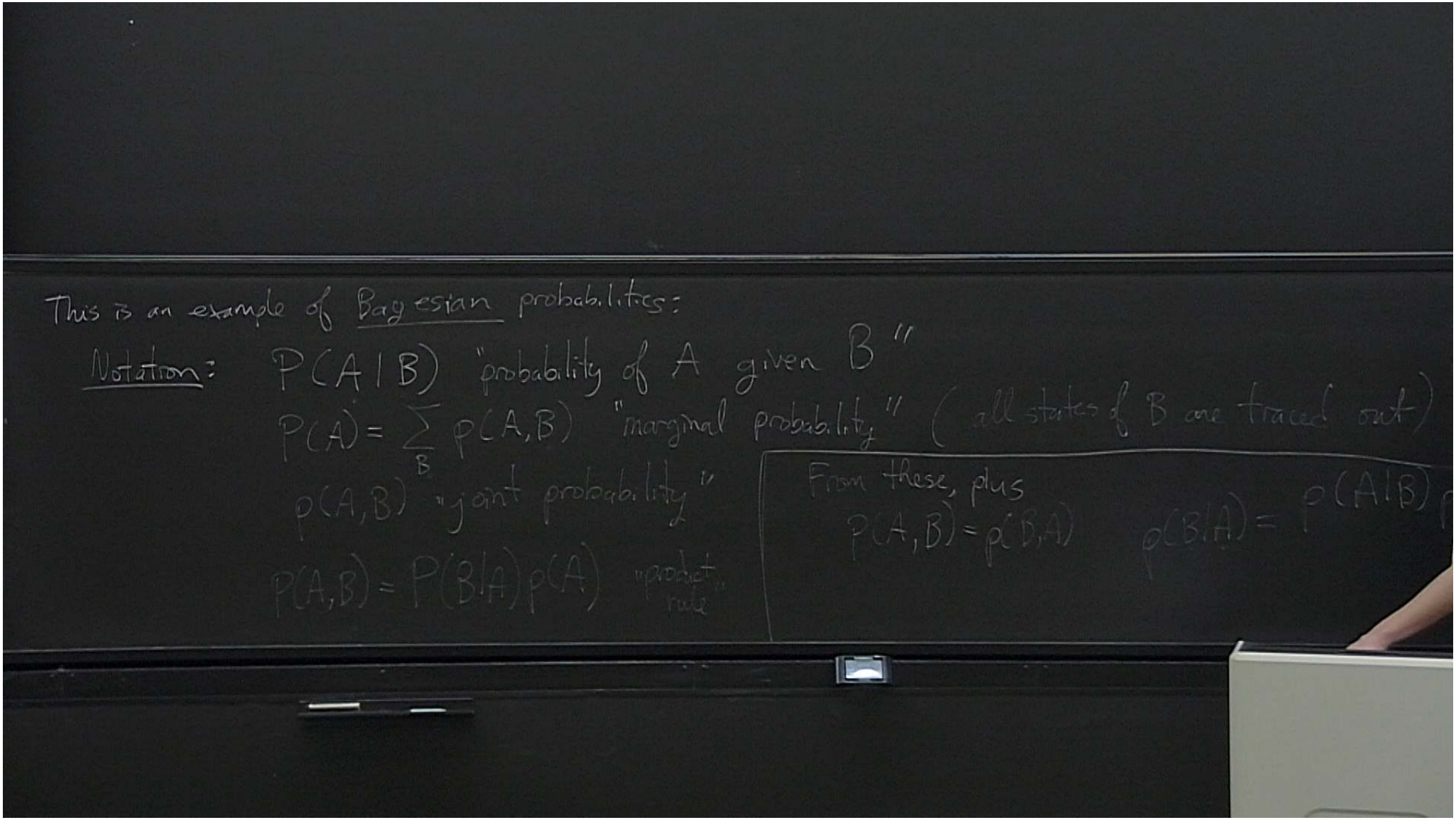


Q: Do you stick with your original choice (#1) or switch to door #3?


This is an example of Bayesian probabilities:

This is an example of Bayesian probabilities:

Notation: $P(A|B)$ "probability of A given B"


This is an example of Bayesian probabilities:
Notation: P(A|B) "probability of A given B"
P(A) = Σ_B p(A,B) "marginal probability" (all states of B are traced out)
p(A,B) "joint probability"
P(A,B) = P(B|A) p(A) "product rule"
From these, plus
p(A,B) = p(B,A)
p(B|A) = p(A|B)

given B"

probability" (all states of B are traced out)

From these, plus

$$p(A,B) = p(B,A) \qquad p(B|A) = \frac{p(A|B)\,p(B)}{p(A)}$$

Bayes' Theorem

$P(A,B) = P(B|A)P(A)$ "product rule"

$P(A,B) = P(B,A)$

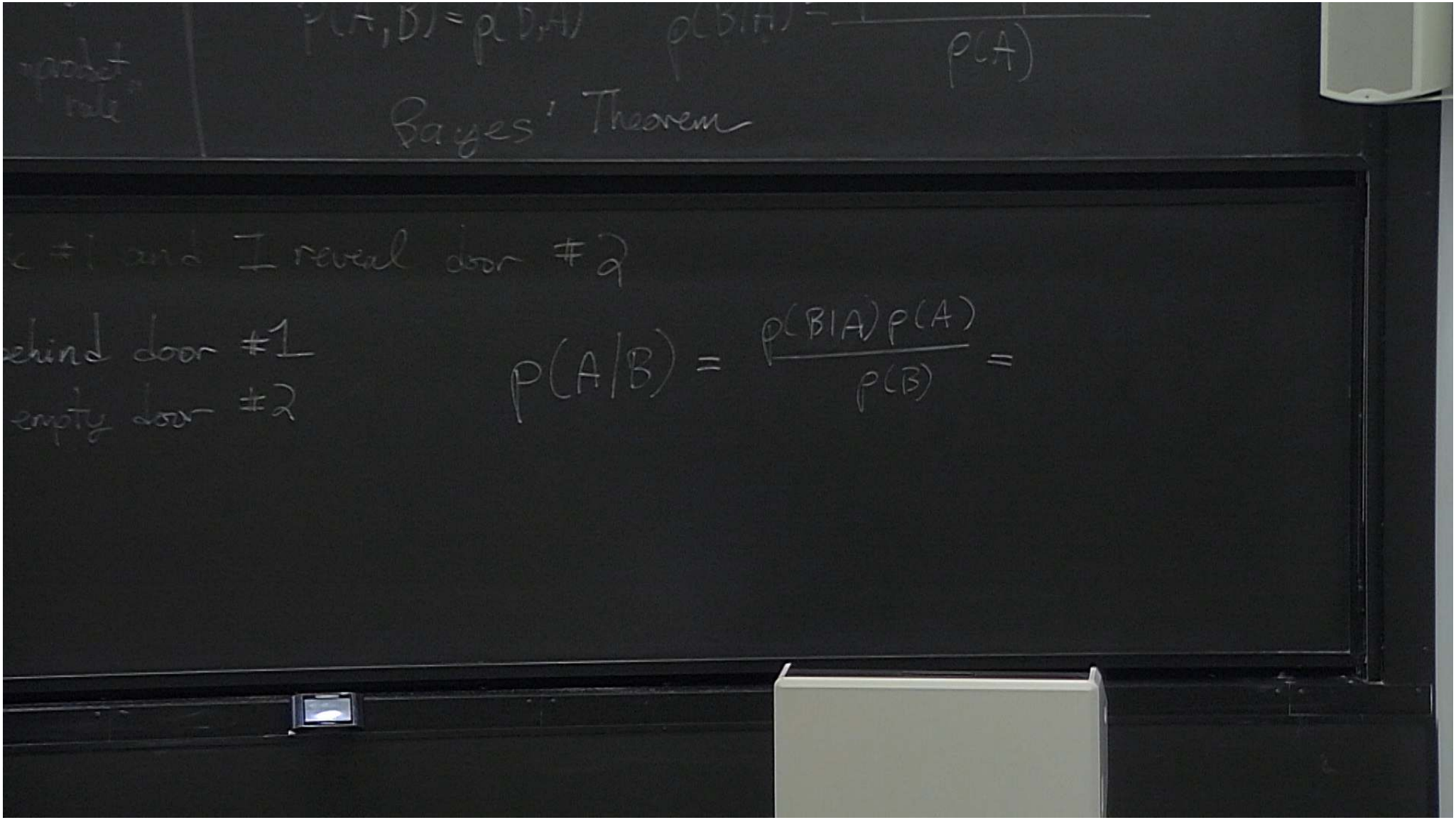
Bayes' Theorem

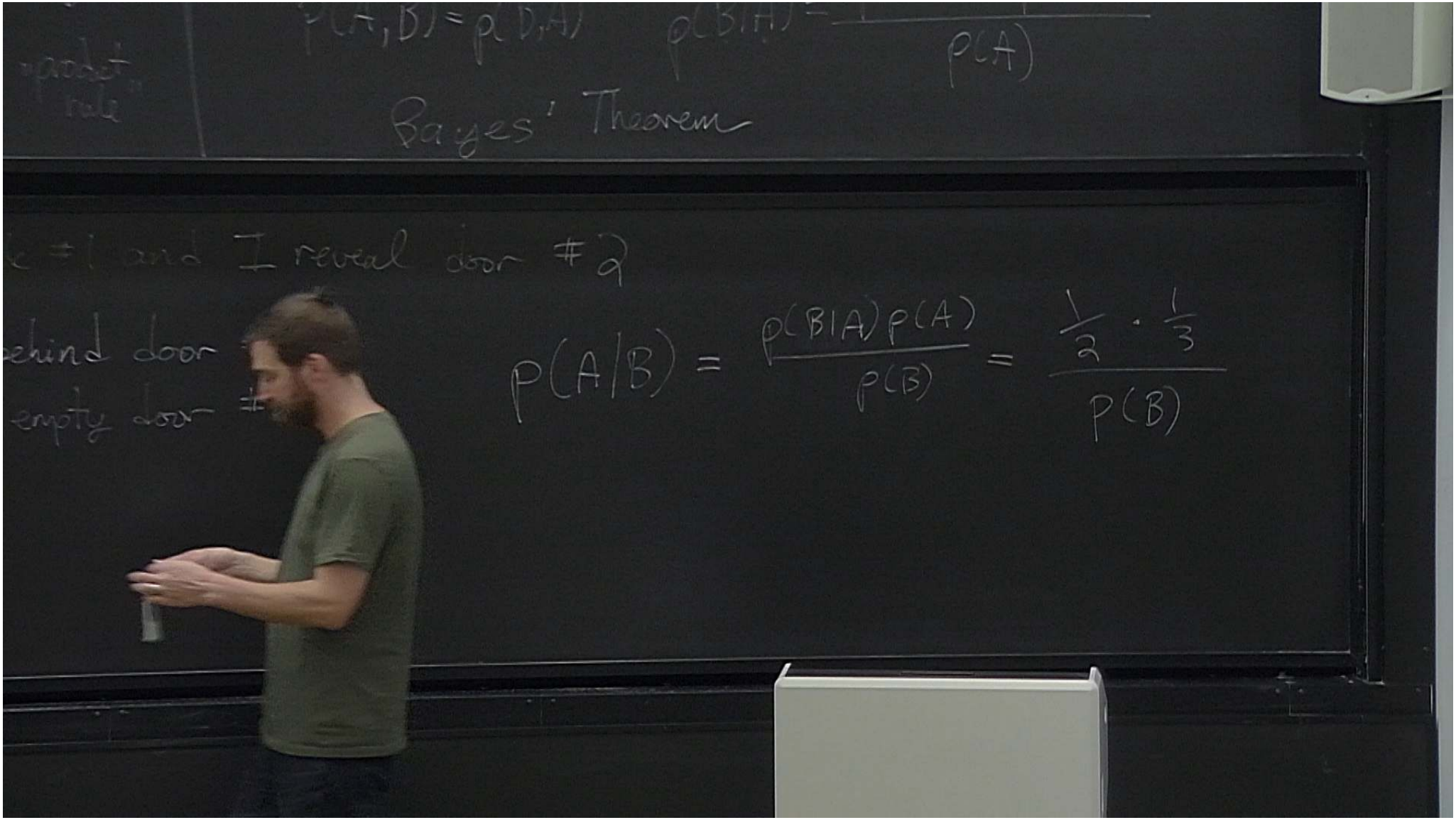
Monte Hall: Assume that you pick #1 and I reveal door #2

A: the event that the car is behind door #1

B: the event that I show an empty door #2

$P(A|B)$


P(A,B) = P(B,A)
P(A)
Bayes' Theorem
I reveal door #2
behind door #1
empty door #2
P(A|B) = P(B|A)P(A) / P(B) =


"product rule"
Bayes' Theorem
and I reveal door #2
behind door
empty door
$P(A|B) = \frac{P(B|A)P(A)}{P(B)} = \frac{\frac{1}{2} \cdot \frac{1}{3}}{P(B)}$

$P(A,B) = P(B|A)P(A)$ "product rule"

$P(A,B) = P(B,A)$

Bayes' Theo

Monte Hall: Assume that you pick #1 and I reveal door #2

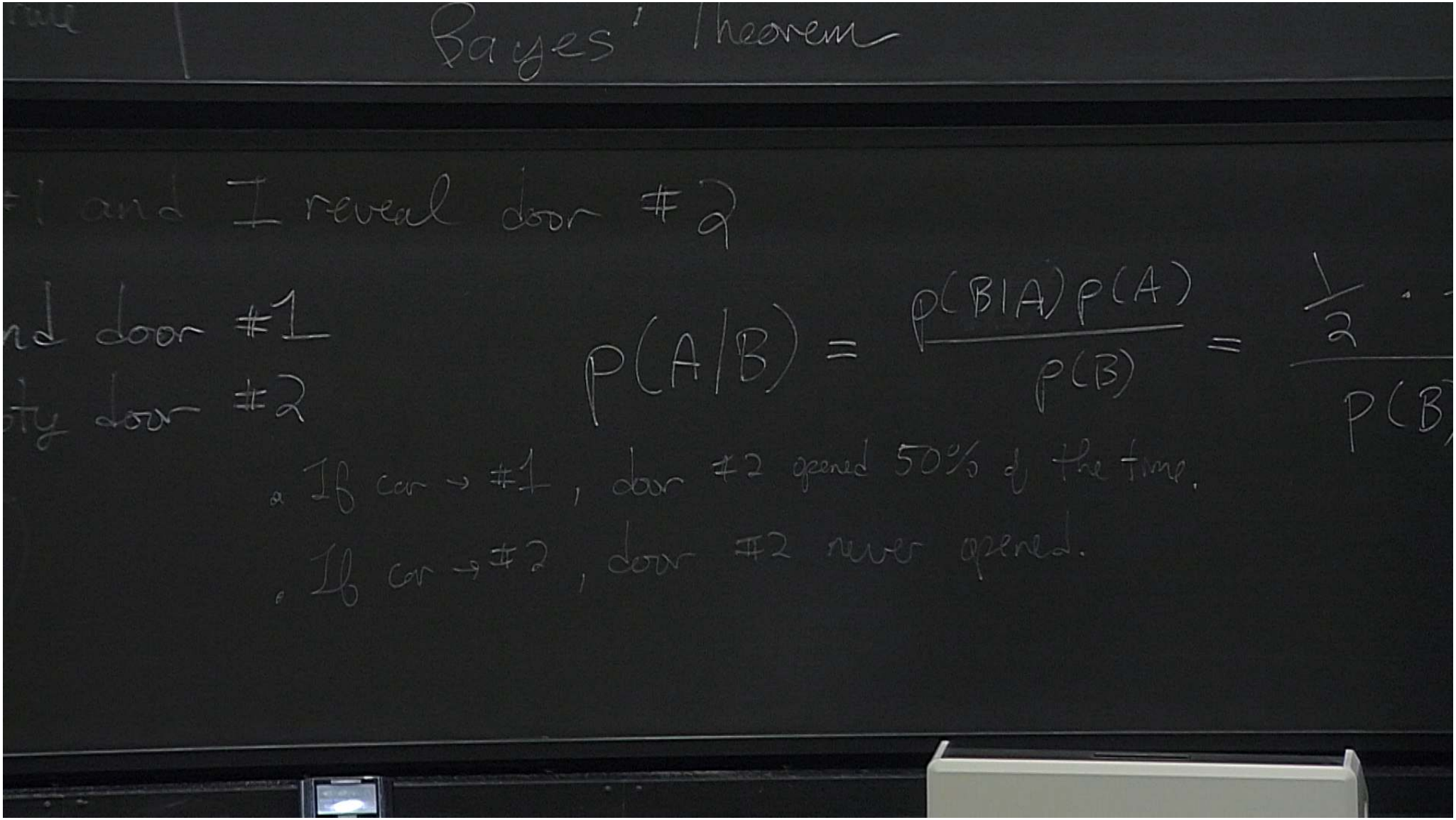
A: the event that the car is behind door #1

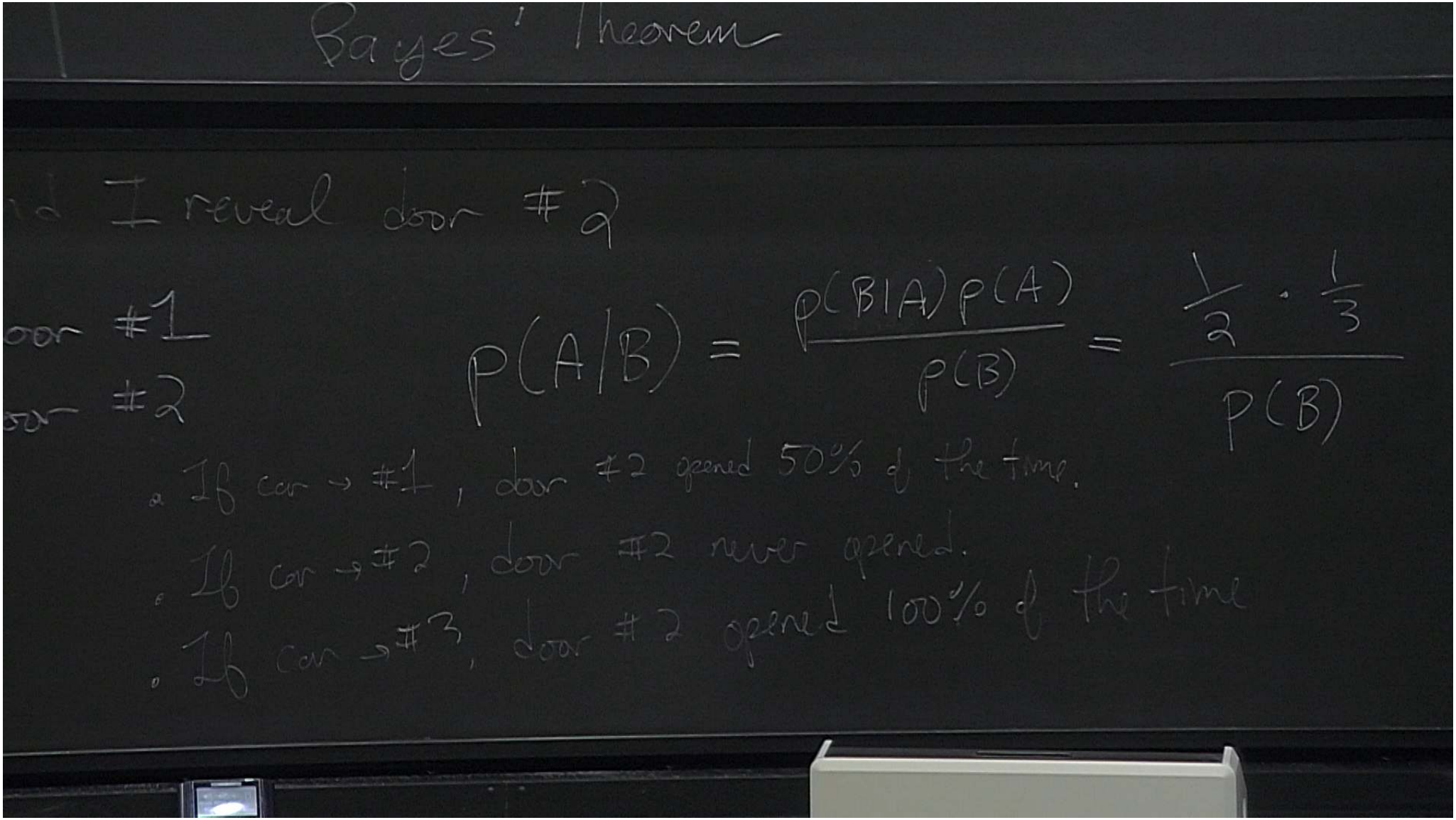
B: the event that I show an empty door #2

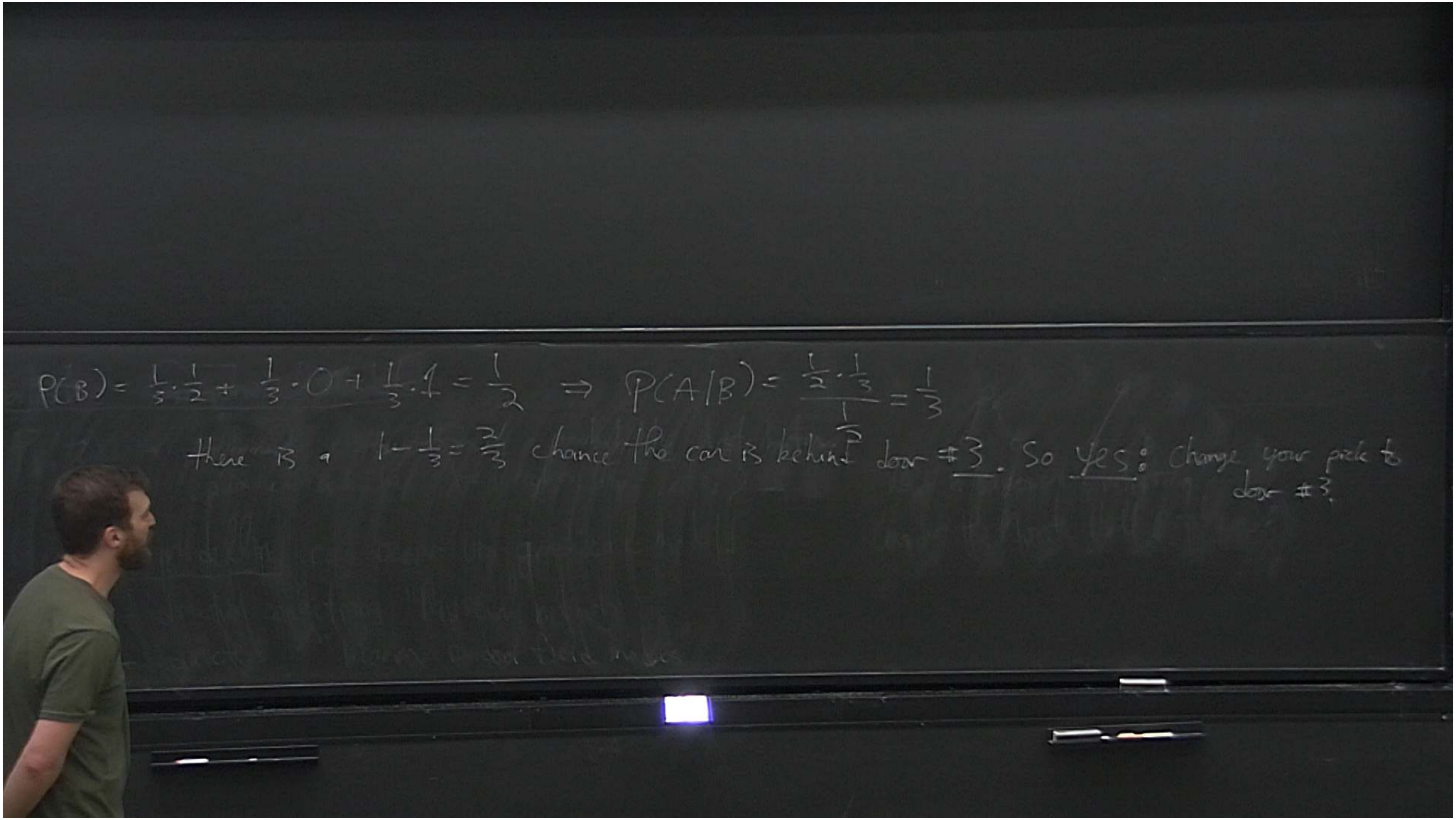
$P(A|B)$

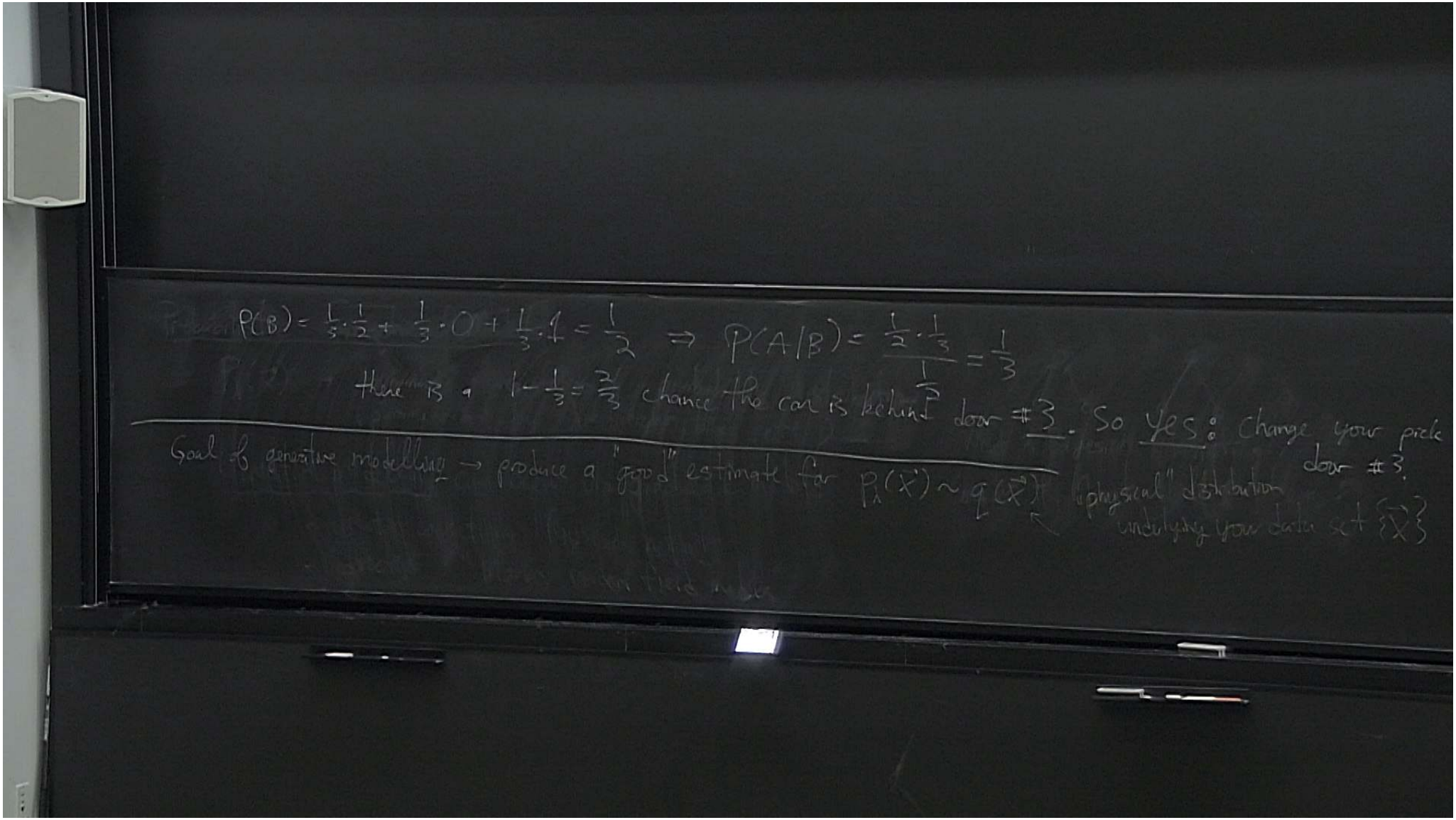
SO: $P(B) = \sum_A P(B|A)P(A)$

$\frac{1}{3}$


Bayes' Theorem
and I reveal door #2
nd door #1
ty door #2
$P(A/B) = \frac{P(B/A)P(A)}{P(B)} = \frac{1}{2} \cdot$
$P(B)$
If car → #1, door #2 opened 50% of the time.
If car → #2, door #2 never opened.


Bayes' Theorem
I reveal door #2
oor #1
or #2
$P(A|B) = \frac{P(B|A)P(A)}{P(B)} = \frac{\frac{1}{2} \cdot \frac{1}{3}}{P(B)}$
If car → #1, door #2 opened 50% of the time.
If car → #2, door #2 never opened.
If car → #3, door #2 opened 100% of the time


$P(B) = \frac{1}{3}\cdot\frac{1}{2} + \frac{1}{3}\cdot 0 + \frac{1}{3}\cdot 1 = \frac{1}{2} \Rightarrow P(A|B) = \frac{\frac{1}{2}\cdot\frac{1}{3}}{\frac{1}{2}} = \frac{1}{3}$
there is a $1-\frac{1}{3}=\frac{2}{3}$ chance the car is behind door #3. So yes: change your pick to door #3.

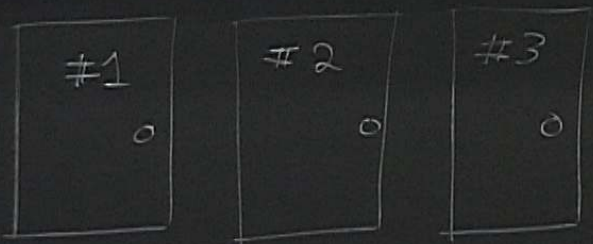

$P(B) = \frac{1}{3}\cdot\frac{1}{2} + \frac{1}{3}\cdot 0 + \frac{1}{3}\cdot 1 = \frac{1}{2} \Rightarrow P(A|B) = \frac{\frac{1}{2}\cdot\frac{1}{3}}{\frac{1}{2}} = \frac{1}{3}$
There is a $1 - \frac{1}{3} = \frac{2}{3}$ chance the car is behind door #3. So yes: change your pick door #3.
Goal of generative modelling → produce a "good" estimate for $P_{\text{data}}(\vec{x}) \sim q(\vec{x})$
"physical" distribution underlying your data set $\{\vec{x}\}$

$P(B) = \frac{1}{3}\cdot\frac{1}{2} + \frac{1}{3}\cdot 0 + \frac{1}{3}\cdot 1 = \frac{1}{2} \Rightarrow P(A|B) = \frac{\frac{1}{2}\cdot\frac{1}{3}}{\frac{1}{2}} = \frac{1}{3}$

there is a $1 - \frac{1}{3} = \frac{2}{3}$ chance the car is behind door #3. So yes: change

Goal of generative modelling → produce a "good" estimate for $p_\lambda(\vec{x}) \sim q(\vec{x})$ "physical" distribution underlying your data

→ generate new instances of $\vec{x}$ (possibly not contained in the original data set)



- one prize (a car)

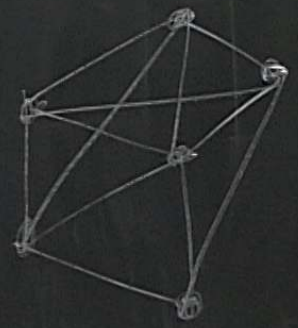
- you (contestant) chooses a door (say door #1)
- Me (the host) reveals a non-selected door containing the car. (say door #2)

Q: Do you stick with your original choice or switch to door #3?

Goal of generative modelling → produce a "good" estimate for $p_\lambda(\vec{x})$

→ generate new instances of $\vec{x}$ (possibly

Hopfield network (1982)



→ produce a good estimate for $p_\lambda(\vec{x}) \approx q(\vec{x})$ "physical" underlying

→ generate new instances of $\vec{x}$ (possibly not contained in the original data

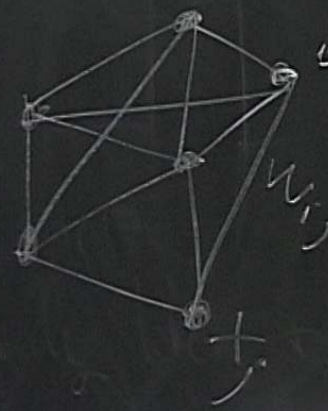
Like an Ising model with "Energy"

$$E(\vec{x}) = -\frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} W_{ij} x_i x_j + \sum_{i=1}^{n} h_i x_i$$

Goal of generative modelling → produce a good estimate for $P_\lambda(\vec{x}) \sim q$

→ generate new instances of $\vec{x}$ (possibly not in

Hopfield network (1982)



Like an Ising model with "Energy"

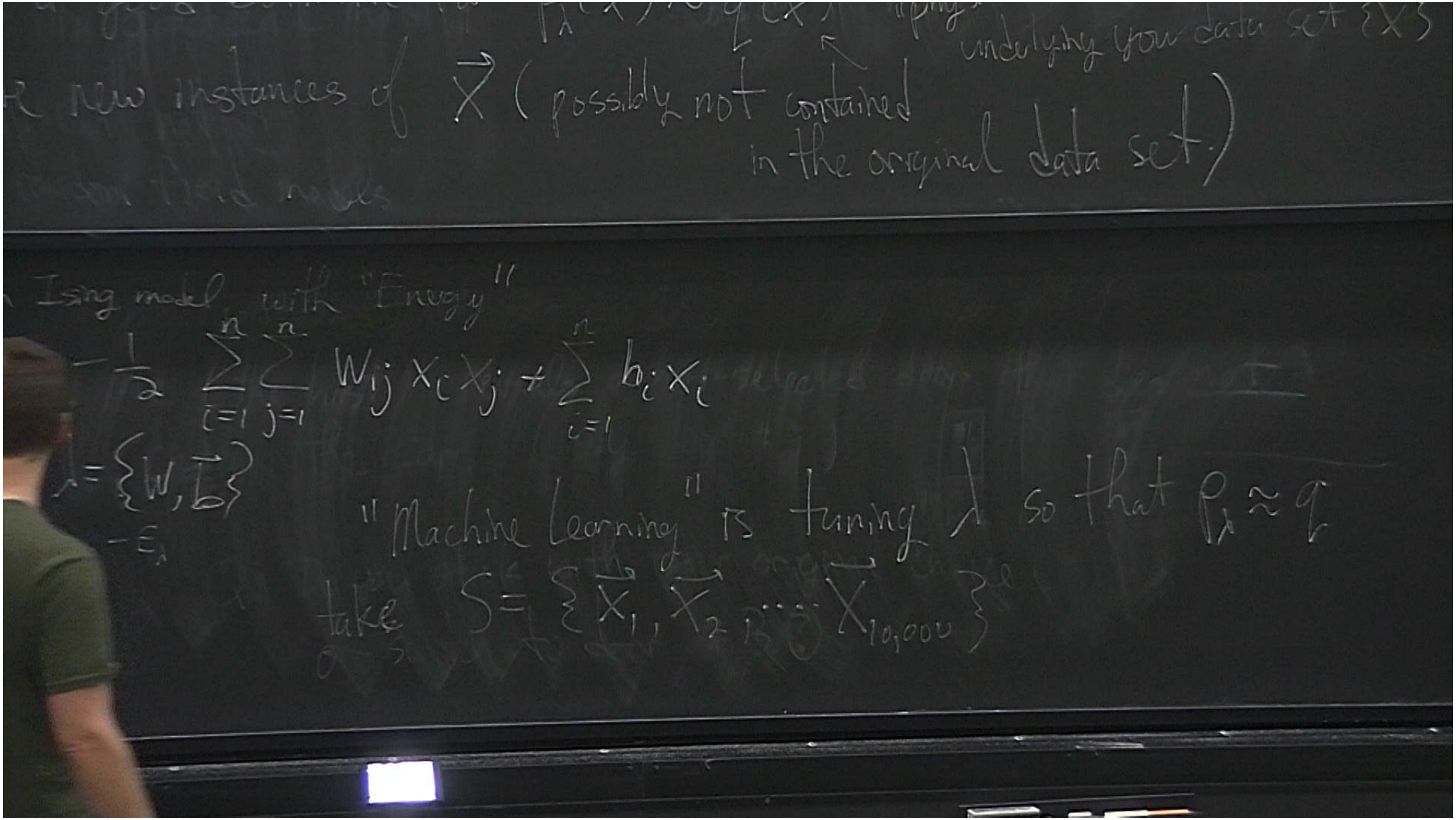
$$E_\lambda(\vec{x}) = -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} x_i x_j + \sum_{i=1}^{n} b_i x_i$$

where $\lambda = \{W, \vec{b}\}$

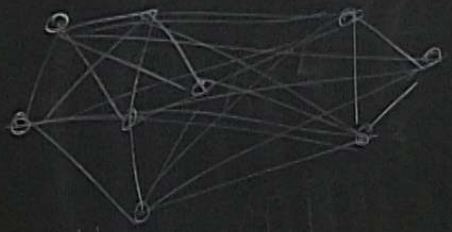
and $p_\lambda = \frac{1}{Z_\lambda} e^{-E_\lambda}$

"Machine learning" is

take $S = \{\vec{x}_1, \vec{x}_2, \ldots$


new instances of $\vec{X}$ (possibly not contained in the original data set!)
underlying your data set $\{X\}$
Ising model with "Energy"
$-\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} W_{ij} x_i x_j + \sum_{i=1}^{n} b_i x_i$
$\lambda = \{W, \vec{b}\}$
$-E_\lambda$
"Machine learning" is tuning λ so that $P_\lambda \approx q$
take $S = \{\vec{X}_1, \vec{X}_2, \ldots \vec{X}_{10,000}\}$

Boltzmann Machine Like an extended Hopfield network.

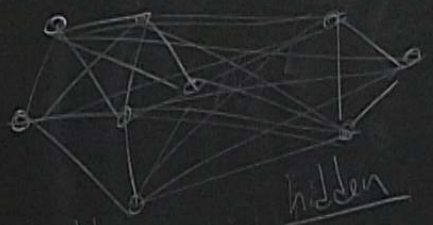


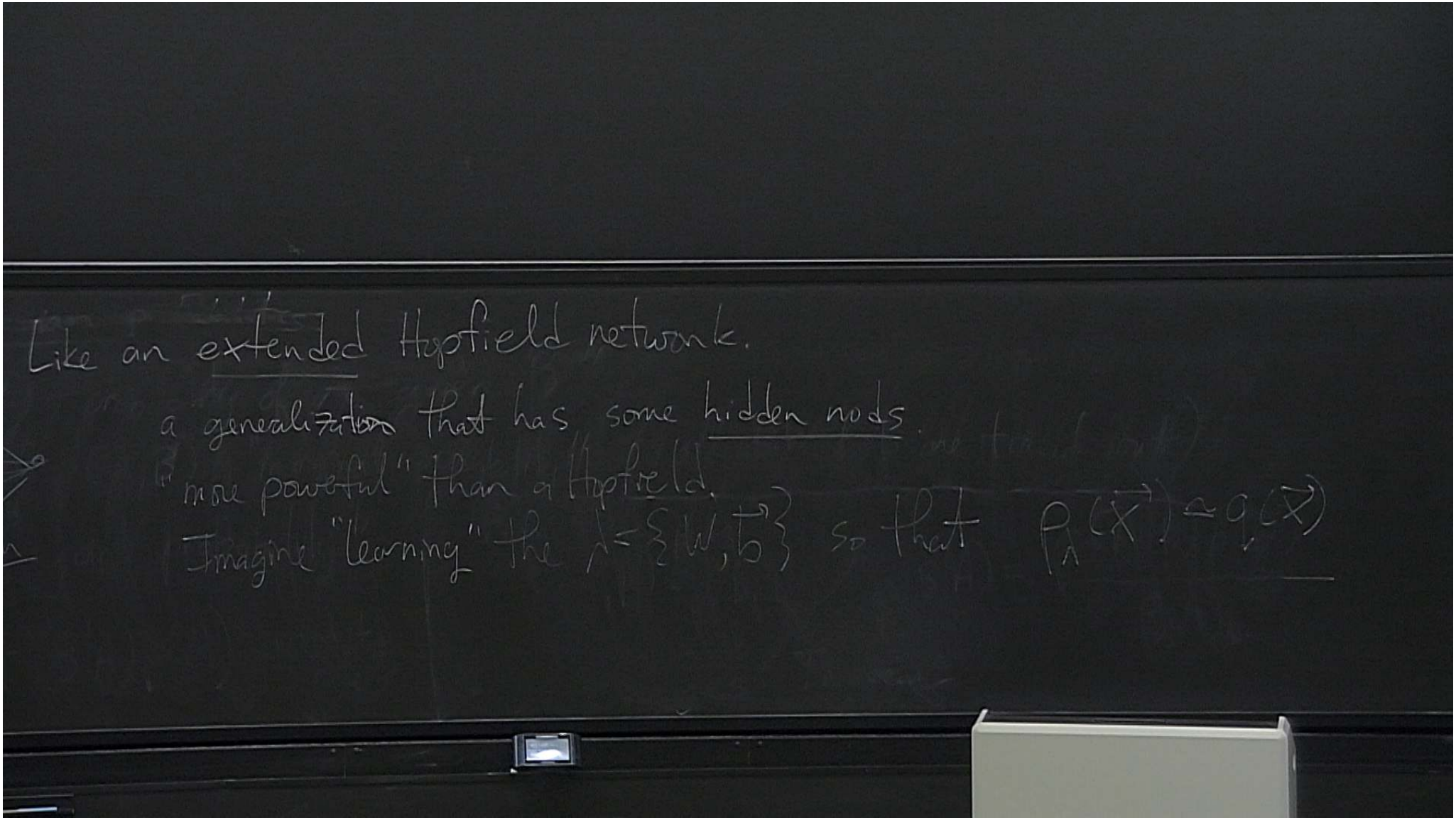
Visible

Boltzmann Machine Like an extended Hopfield network.

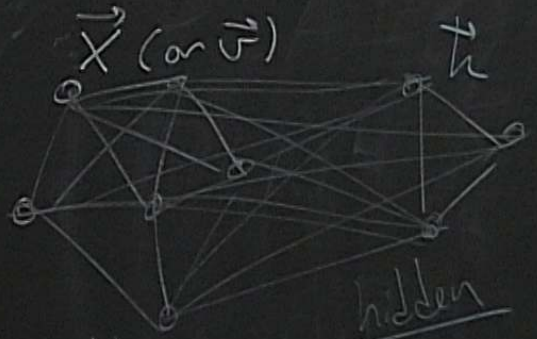
a generalization that has some hidden nods.

"more powerful" than a Hopfield.

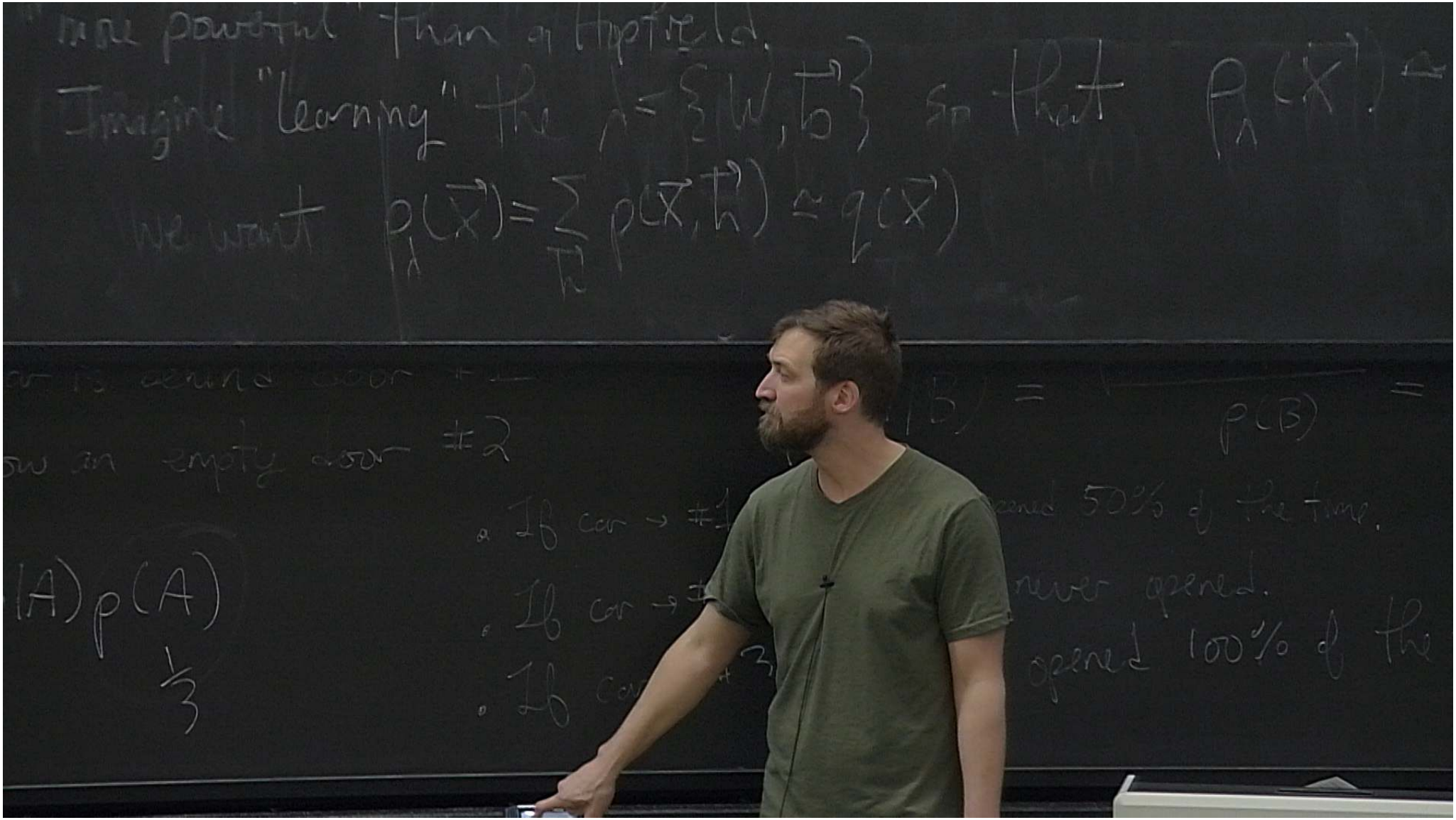



Like an extended Hopfield network.
a generalization that has some hidden nods
"more powerful" than a Hopfield.
Imagine "learning" the $\lambda = \{W, \vec{b}\}$ so that $P_\lambda(\vec{x}) \approx q(\vec{x})$

Boltzmann Machine Like an extended Hopfield net



a generalization that has so
"more powerful" than a Hopf
Imagine "learning" the $\vec{x}$=


"more powerful" than a Hopfield.
Imagine "learning" the λ = {W, b⃗} so that ρ_λ(x⃗) ≃
We want ρ_λ(x⃗) = Σ_h⃗ p(x⃗, h⃗) ≃ q(x⃗)
ow an empty door #2
p(B)
If car → #1
opened 50% of the time.
If car → #
never opened.
If car
opened 100% of the
(A) p(A)
1/3

an extended Hopfield network.

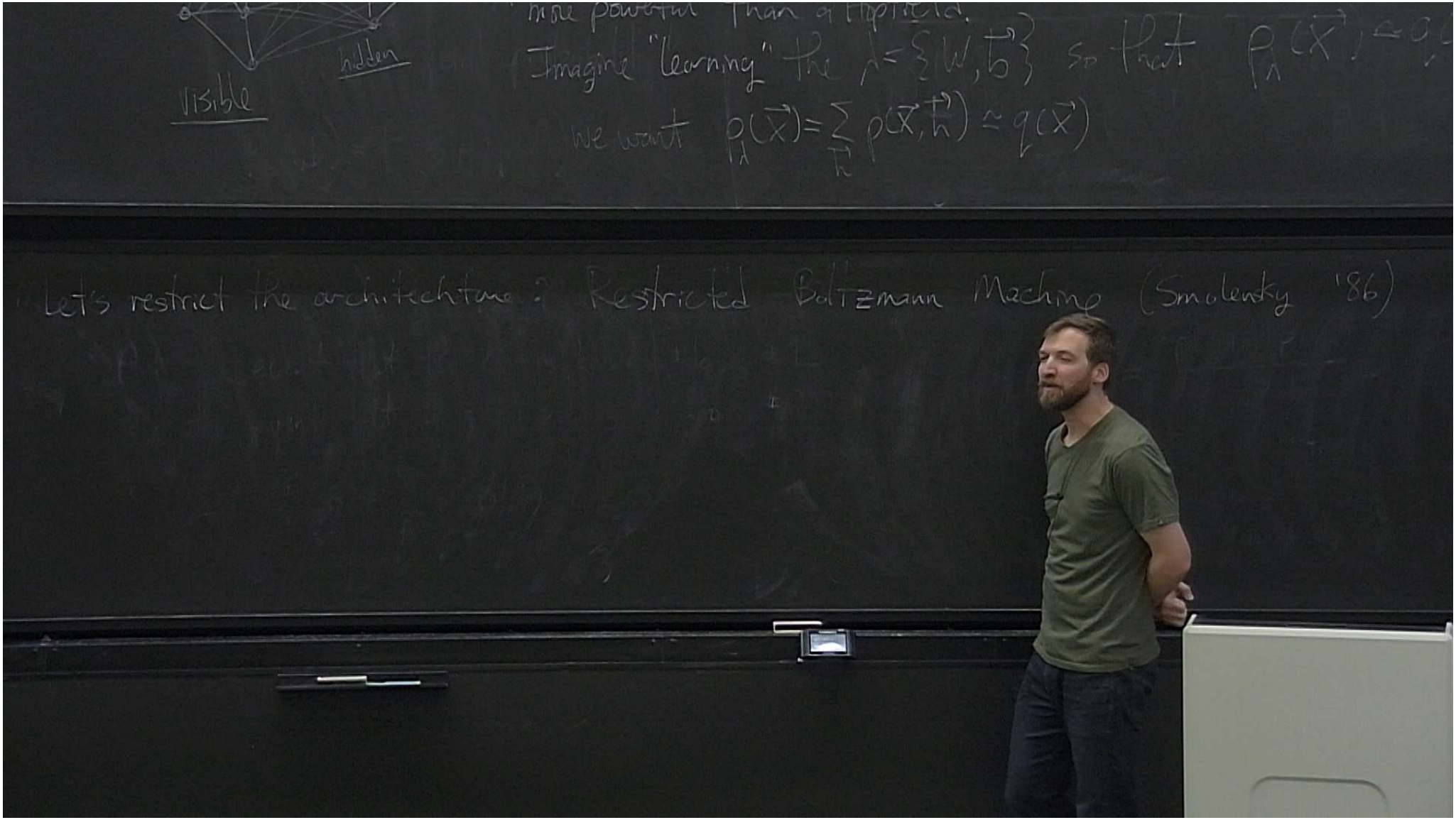
a generalization that has some hidden nods.

"more powerful" than a Hopfield.

Imagine "learning" the $\lambda = \{W, \vec{b}\}$ so that $P_\lambda(\vec{x}) \simeq$

we want $P_\lambda(\vec{x}) = \sum_{\vec{h}} p(\vec{x}, \vec{h}) \simeq q(\vec{x})$

you pick #1 and I reveal door #2


hidden
visible
more powerful than a Hopfield.
Imagine "learning" the $\lambda = \{W, \vec{b}\}$ so that $p_\lambda(\vec{x}) \approx q$
we want $p_\lambda(\vec{x}) = \sum_{\vec{h}} p(\vec{x}, \vec{h}) \approx q(\vec{x})$
Let's restrict the architechture: Restricted Boltzmann Machine (Smolensky '86)

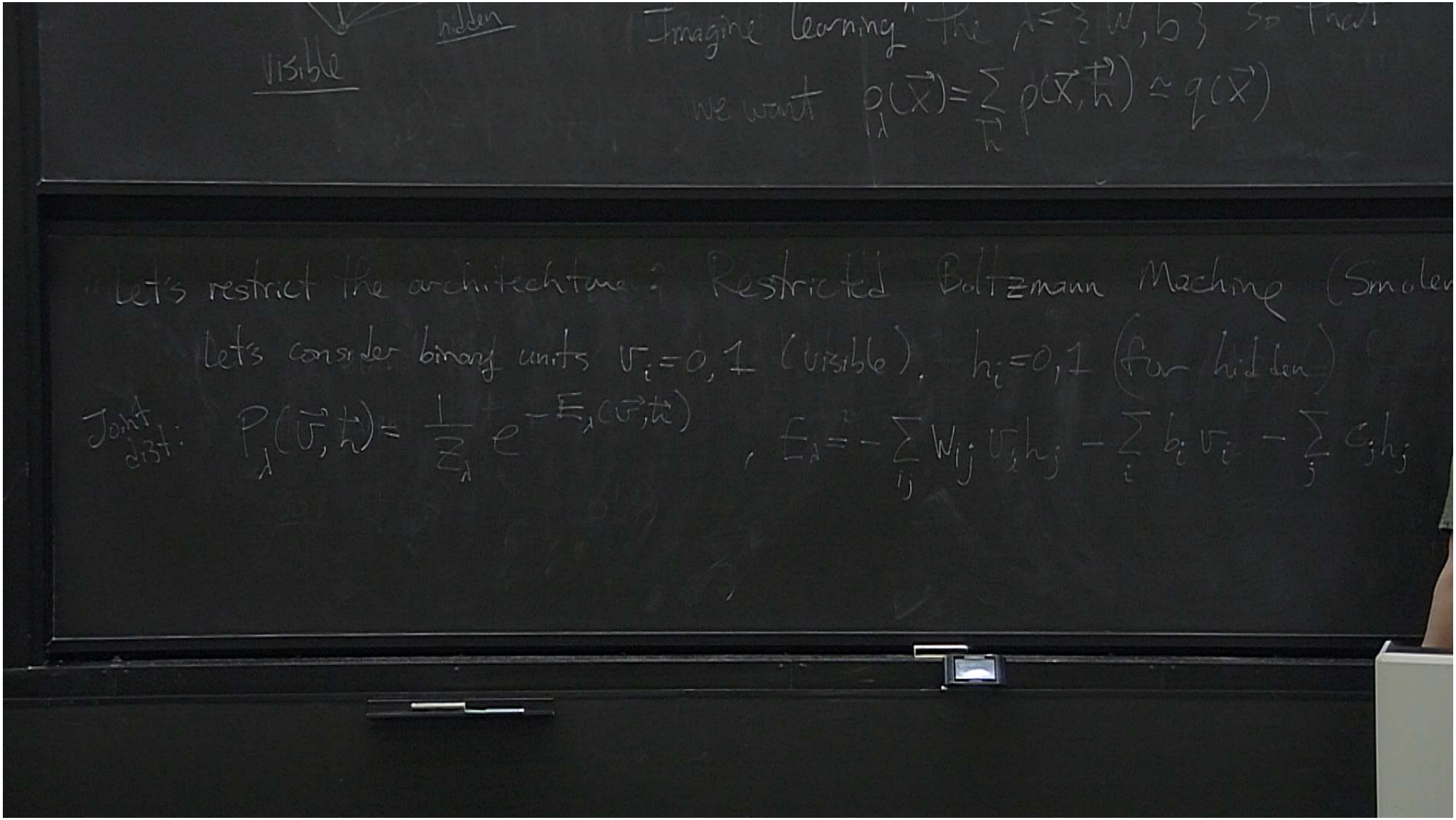
visible

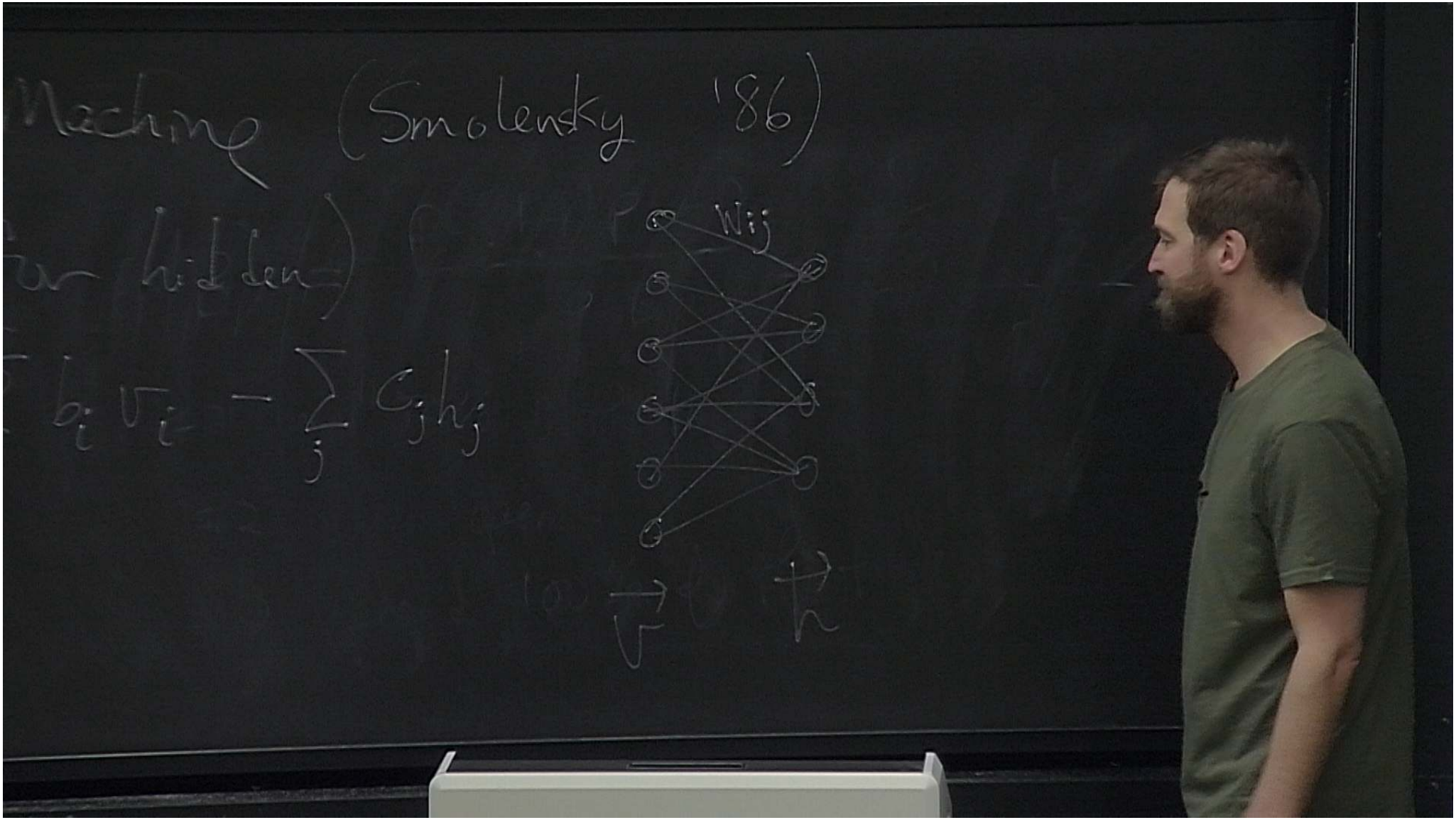
we want $P_\lambda(\vec{v}) = \sum_{\vec{h}} p(\vec{v}, \vec{h}) \simeq q(\vec{v})$

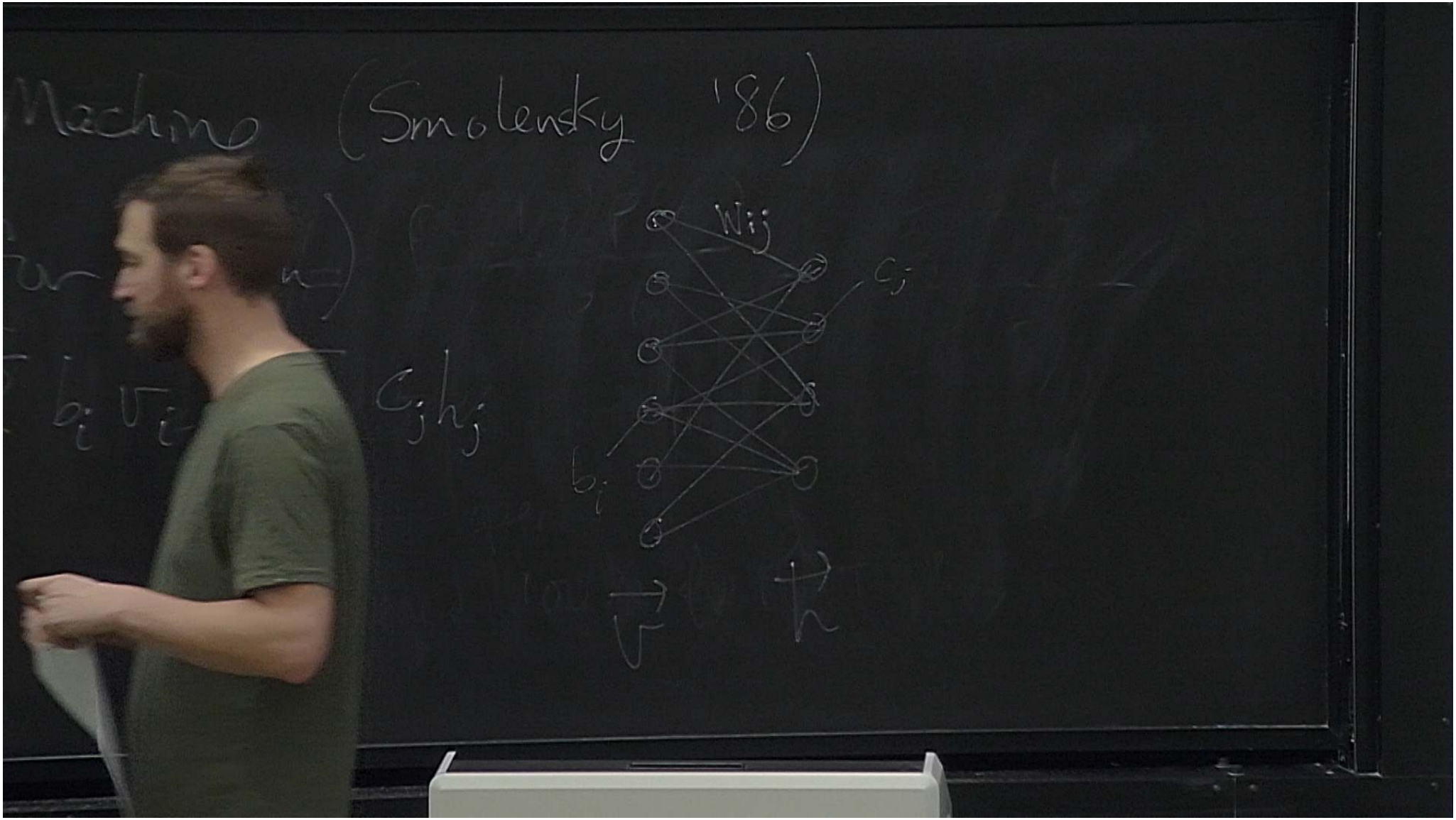
Let's restrict the architecture: Restricted Boltzmann Machine (Sm

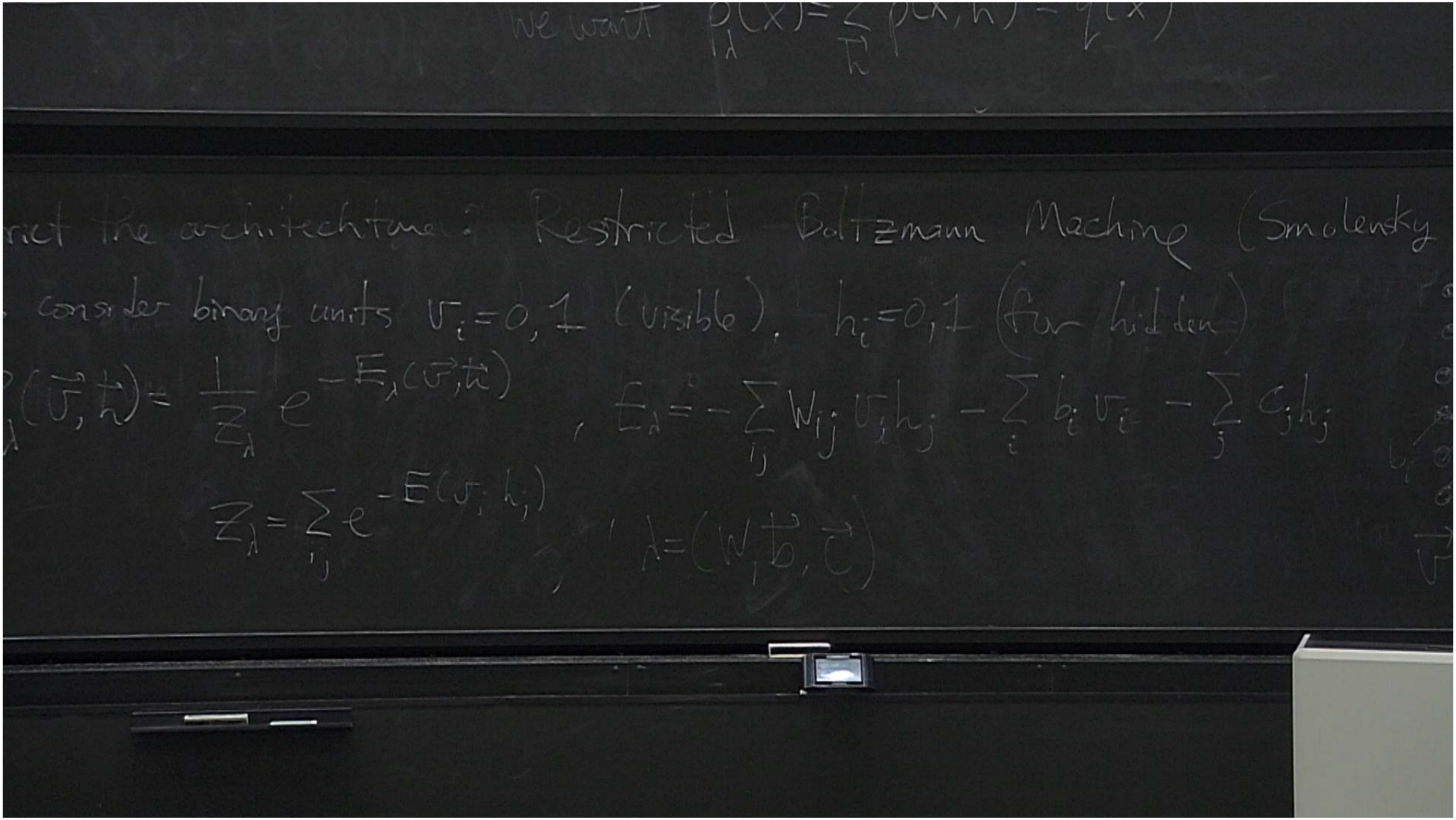
Let's consider binary units $v_i = 0, 1$ (visible), $h_i = 0, 1$ (for hidden)

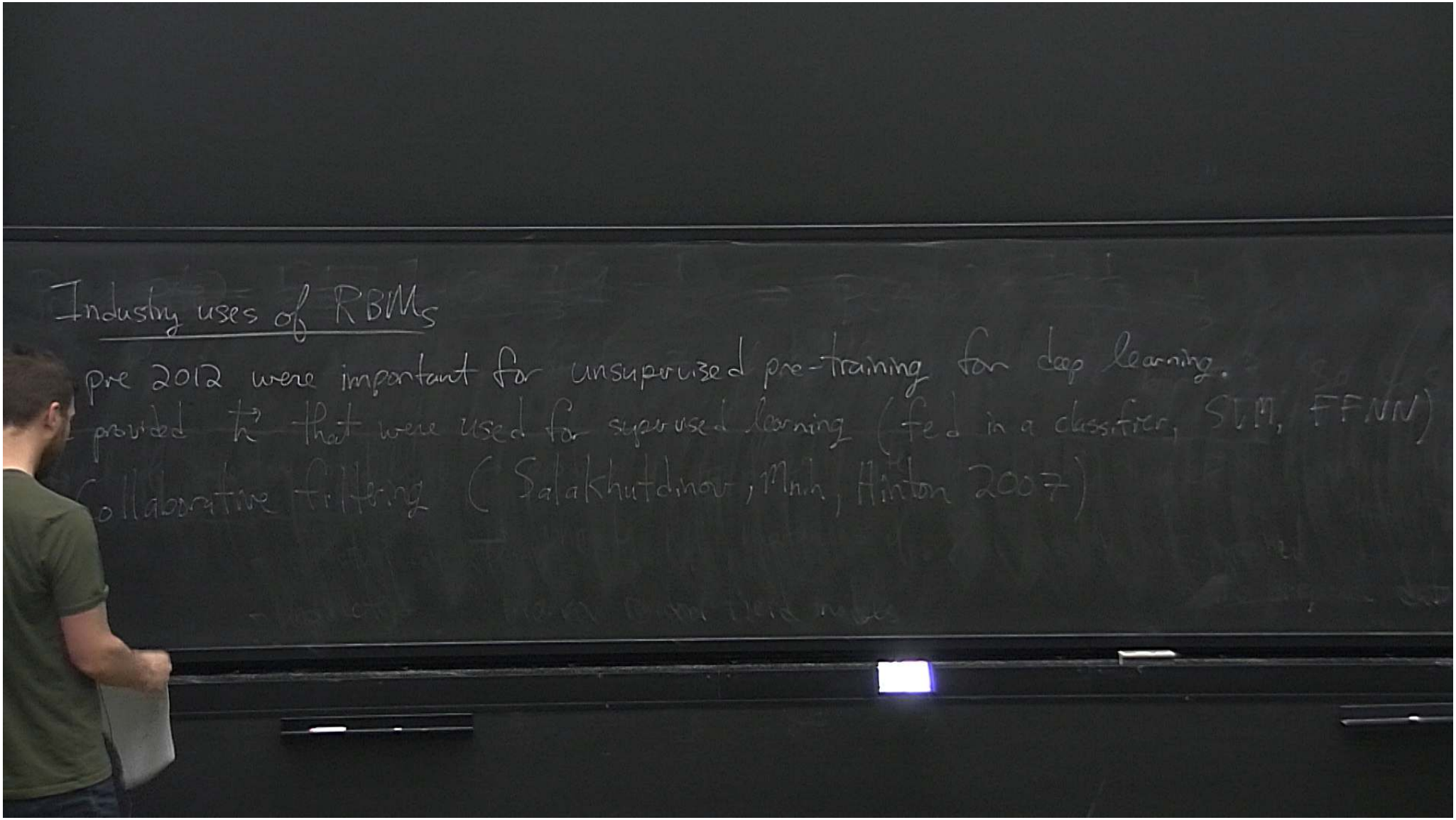
Joint dist: $P_\lambda(\vec{v}, \vec{h}) = \frac{1}{Z_\lambda} e^{-E_\lambda(\vec{v}, \vec{h})}$, $E_\lambda = -\sum_{ij} W_{ij} v_i h_j$

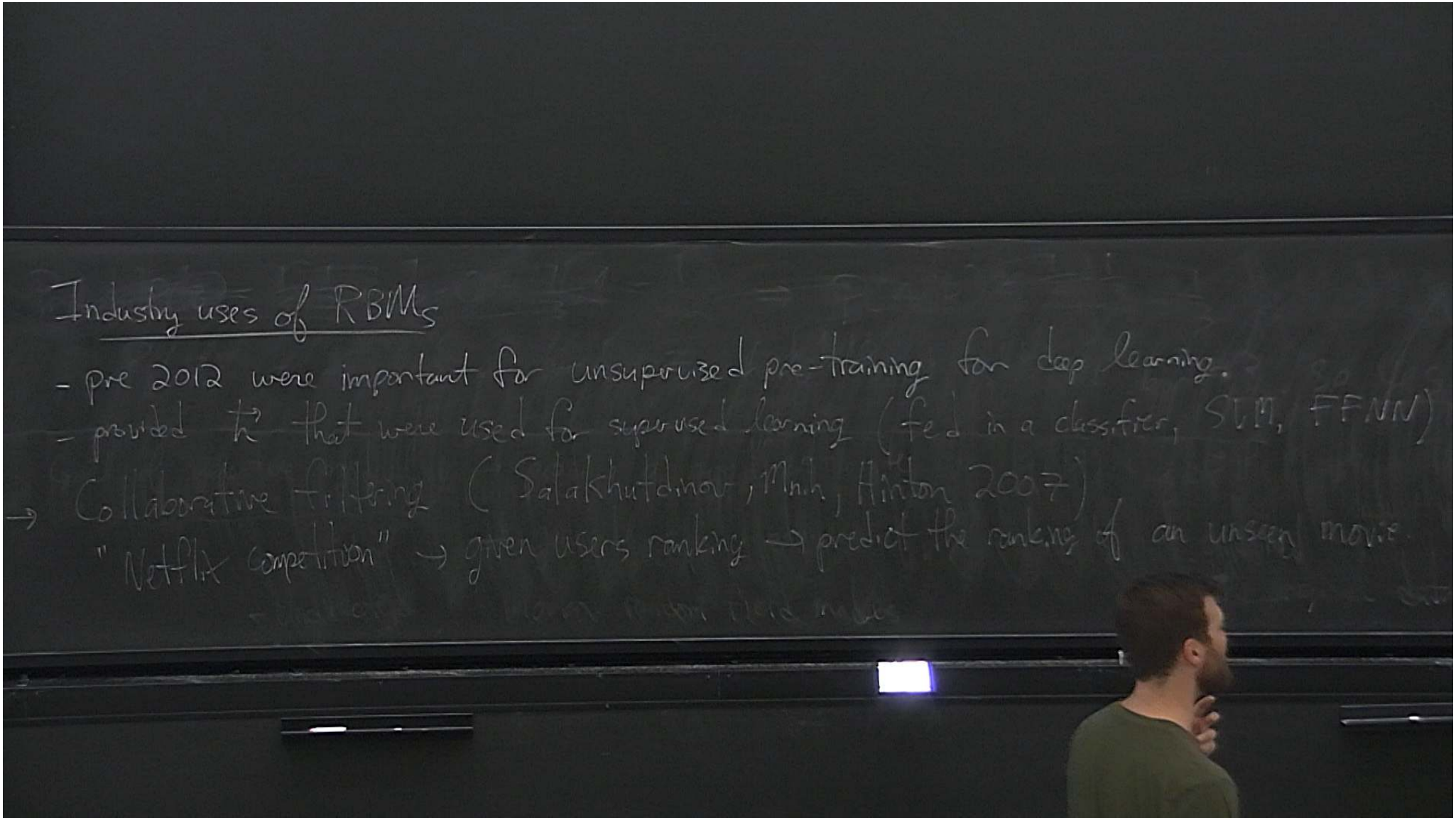

visible
hidden
Imagine learning" the $\lambda = \{W, b\}$ so that
we want $p_\lambda(\vec{x}) = \sum_{\vec{h}} p(\vec{x}, \vec{h}) \simeq q(\vec{x})$
Let's restrict the architechture: Restricted Boltzmann Machine (Smolen
Let's consider binary units $v_i = 0, 1$ (visible), $h_i = 0, 1$ (for hidden)
Joint dist: $P_\lambda(\vec{v}, \vec{h}) = \frac{1}{Z_\lambda} e^{-E_\lambda(\vec{v}, \vec{h})}$, $E_\lambda = -\sum_{ij} W_{ij} v_i h_j - \sum_i b_i v_i - \sum_j c_j h_j$

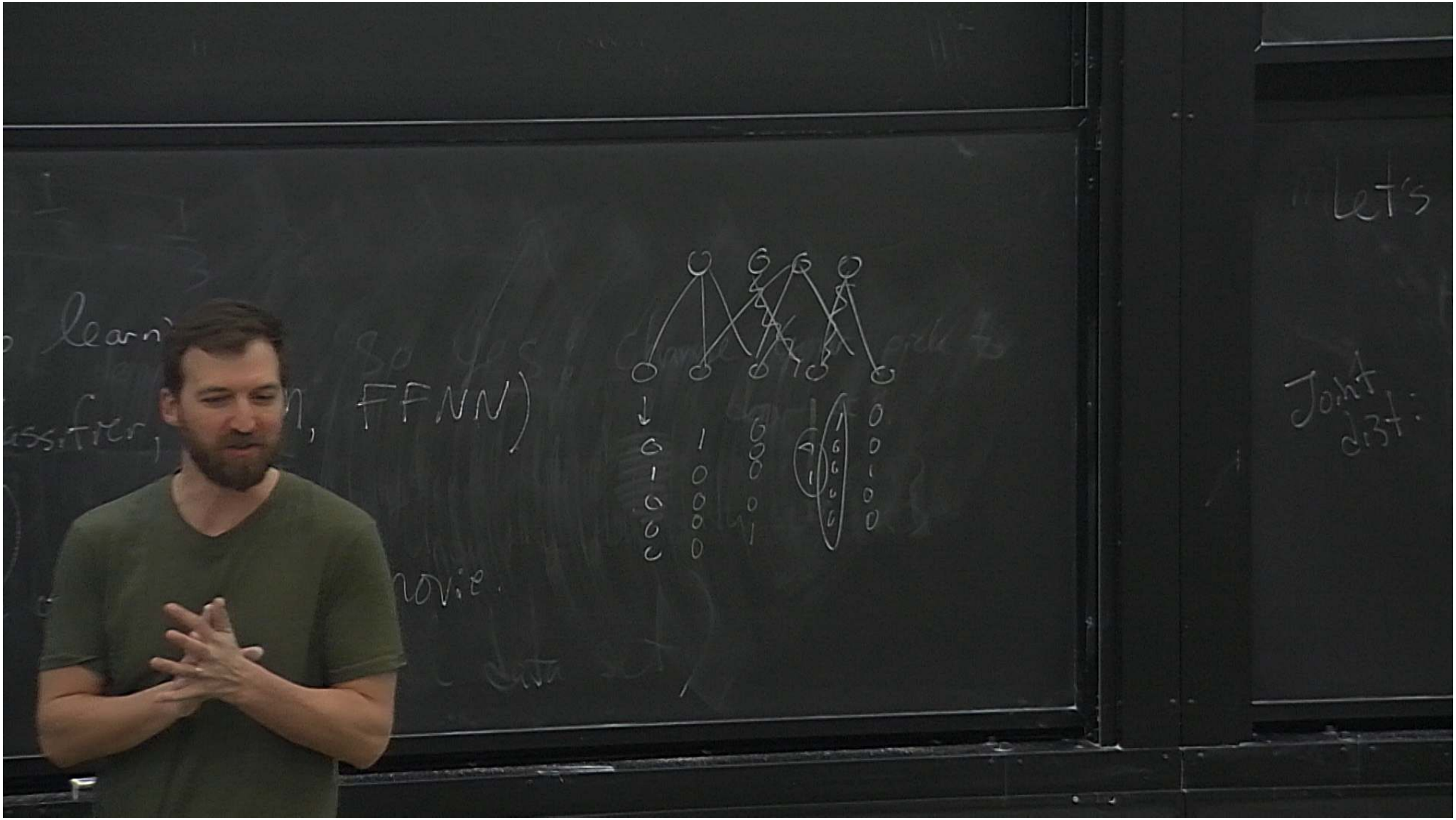

Machine (Smolensky '86)
Wij
$\vec{v}$
$\vec{h}$


Machine (Smolensky '86)
Wij
c_j
b_i
$c_j h_j$
$\vec{v}$
$\vec{h}$


we want
rict the architechture: Restricted Boltzmann Machine (Smolensky
consider binary units $v_i = 0,1$ (visible), $h_i = 0,1$ (for hidden)
$p_\lambda(\vec{v},\vec{h}) = \frac{1}{Z_\lambda} e^{-E_\lambda(\vec{v},\vec{h})}$
$E_\lambda = -\sum_{ij} W_{ij} v_i h_j - \sum_i b_i v_i - \sum_j c_j h_j$
$Z_\lambda = \sum_{ij} e^{-E(v_i, h_j)}$
$\lambda = (W, \vec{b}, \vec{c})$


Industry uses of RBMs
pre 2012 were important for unsupervised pre-training for deep learning.
provided h^2 that were used for supervised learning (fed in a classifier, SVM, FFNN)
Collaborative filtering (Salakhutdinov, Mnih, Hinton 2007)


Industry uses of RBMs
- pre 2012 were important for unsupervised pre-training for deep learning.
- provided $\vec{h}$ that were used for supervised learning (fed in a classifier, SVM, FFNN)
→ Collaborative filtering (Salakhutdinov, Mnih, Hinton 2007)
"Netflix competition" → given users ranking → predict the ranking of an unseen movie


FFNN)
movie.
Let's
Joint