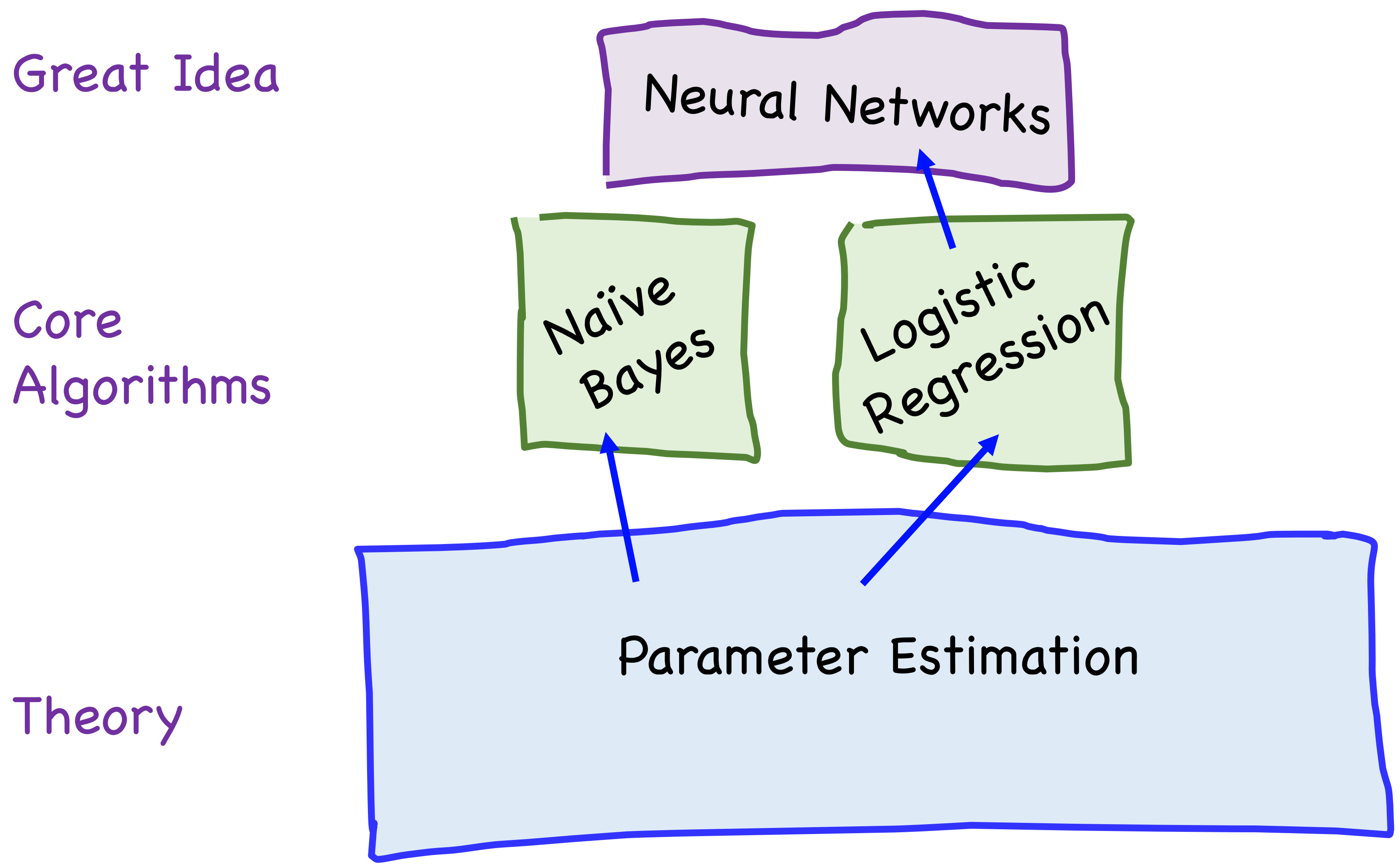
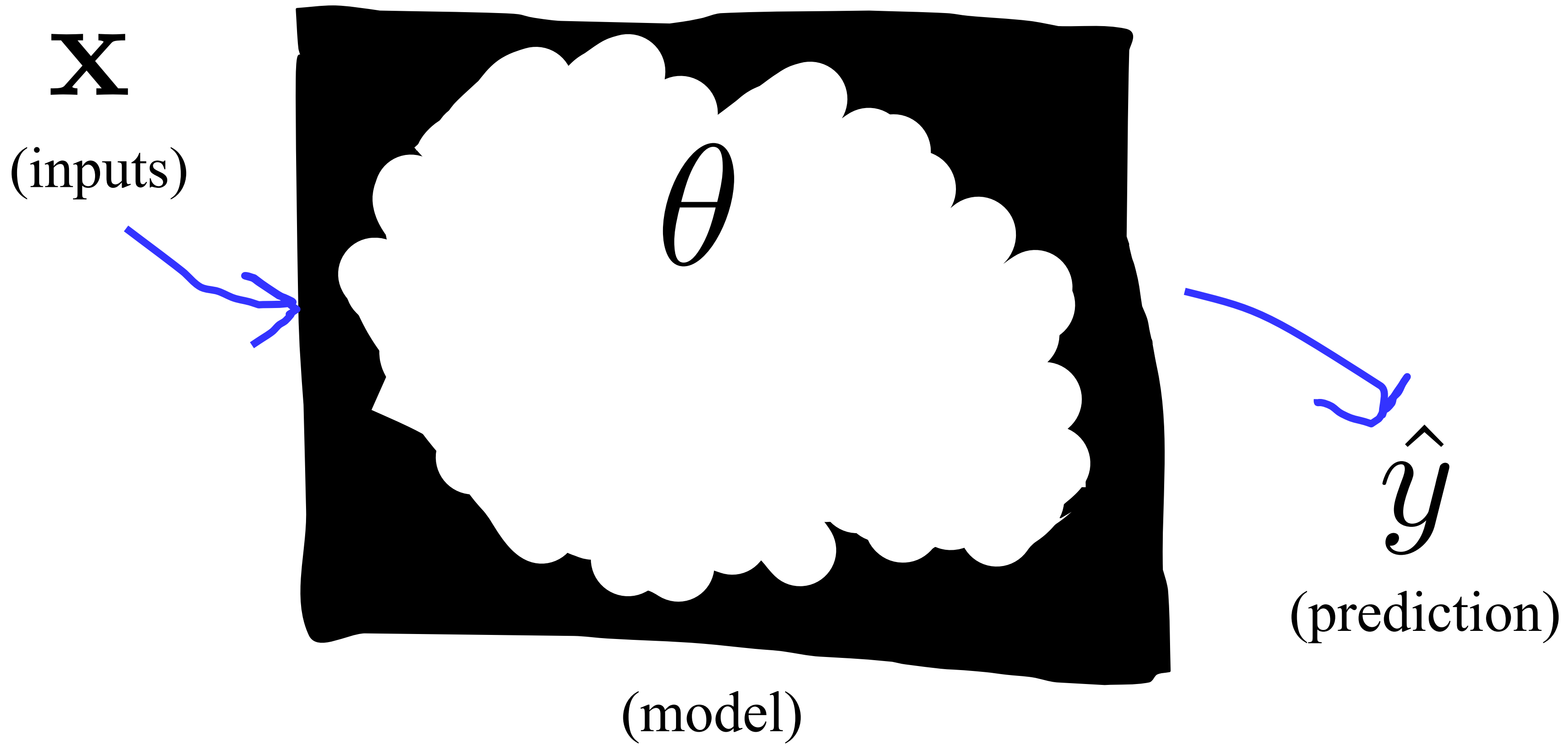
Naïve Bayes

Chris Piech
CS109, Stanford University

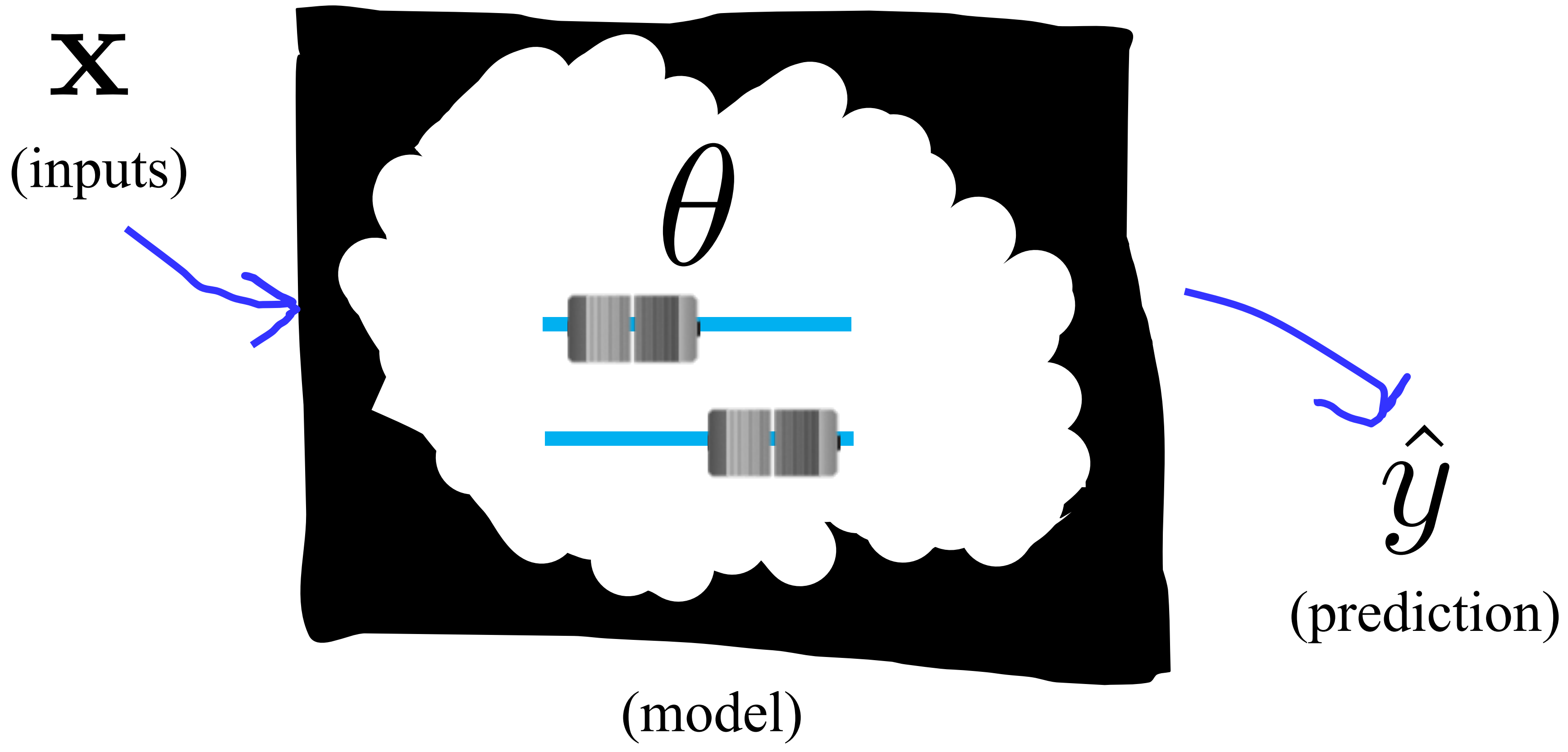
Machine Learning in CS109



Machine Learning



Machine Learning



MLE vs MAP

Data: $x^{(1)}, \ldots, x^{(n)}$

Maximum Likelihood Estimation

$$\hat{\theta}_{MLE} = \underset{\theta}{\operatorname{argmax}}\ f(x^{(1)}, \ldots, x^{(n)} | \theta)$$
$$= \underset{\theta}{\operatorname{argmax}} \left(\sum_i \log f(x^{(i)} | \theta) \right)$$

Maximum A Posteriori

$$\hat{\theta}_{MAP} = \underset{\theta}{\operatorname{argmax}}\ f(\theta | x^{(1)}, \ldots, x^{(n)})$$
$$= \underset{\theta}{\operatorname{argmax}} \left(\log(g(\theta)) + \sum_{i=1}^{n} \log(f(x^{(i)} | \theta)) \right)$$

Event Shorthand

MAP, without shorthand

$$\hat{\theta}_{MAP} = \underset{\theta}{\operatorname{argmax}}\ f(\Theta = \theta | X^{(1)} = x^{(1)}, \ldots, X^{(n)} = x^{(n)})$$

Our shorthand notation

$\boxed{\theta}$ is shorthand for the event: $\boxed{\Theta = \theta}$

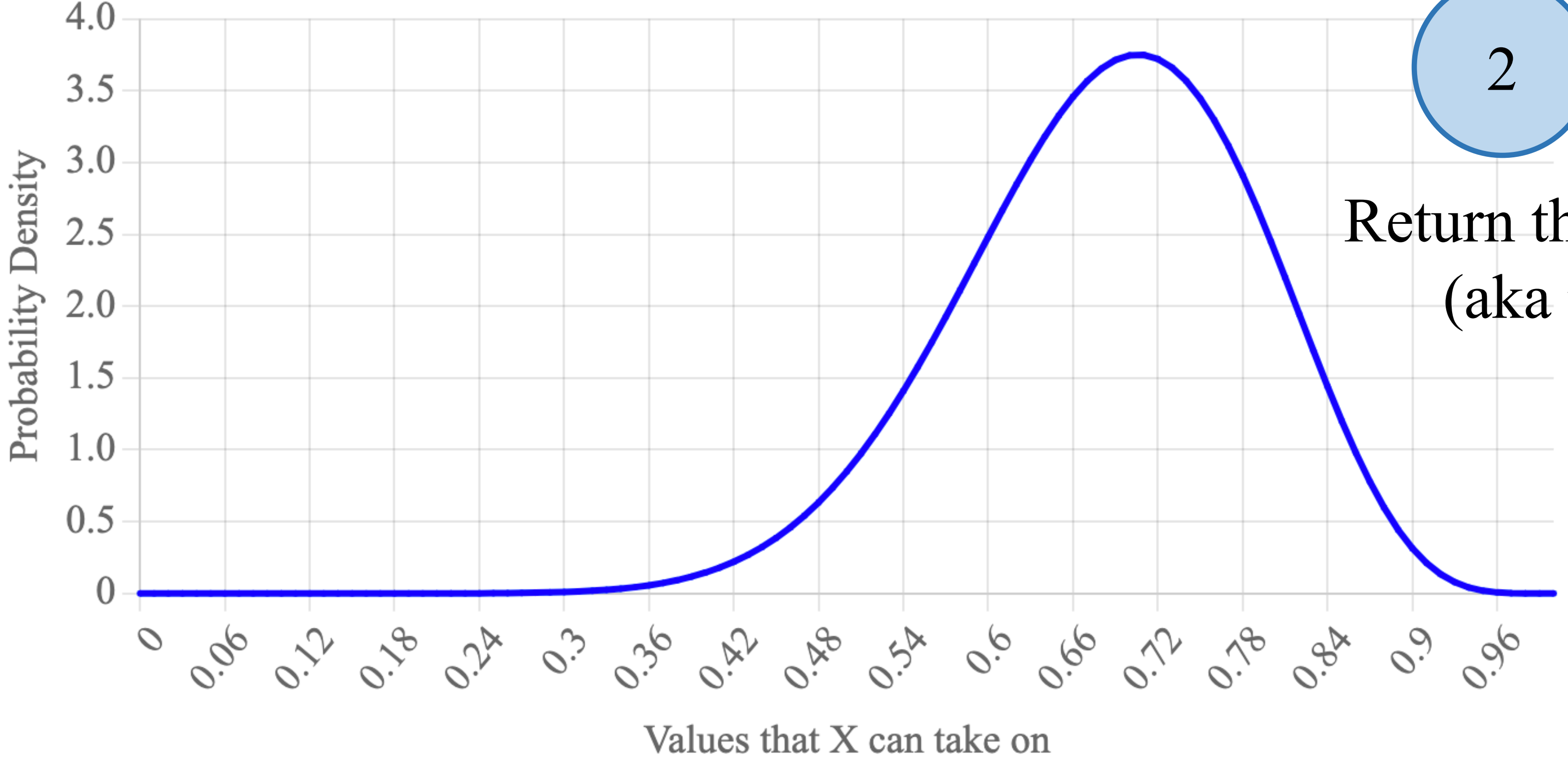
$\boxed{x^{(i)}}$ is shorthand for the event: $\boxed{X^{(i)} = x^{(i)}}$

MAP, now with shorthand

$$\hat{\theta}_{MAP} = \underset{\theta}{\operatorname{argmax}}\ f(\theta | x^{(1)}, \ldots, x^{(n)})$$

MAP For Bernoulli

1 Infer the posterior belief in params



2 Return the argmax (aka mode)

Quick MAP for Bernoulli

Beta(a, b) is a conjugate prior for the probability of success in Bernoulli and Binomial distributions.

$$f(x) = \frac{1}{B(a,b)} x^{a-1}(1-x)^{b-1}$$

Prior Beta(a, b)
Saw $a + b - 2$ imaginary trials: $a - 1$ successes, $b - 1$ failures

Experiment Observe $n + m$ new trials: n successes, m failures

Posterior Beta$(a + n, b + m)$

MAP:
$$p = \frac{a + n - 1}{a + b + n + m - 2}$$

Quick MAP for Bernoulli with Laplace

Beta(a, b) is a conjugate prior for the probability of success in Bernoulli and Binomial distributions.

$$f(x) = \frac{1}{B(a,b)} x^{a-1}(1-x)^{b-1}$$

Prior Beta$(a = 2, b = 2)$
Saw 2 imaginary trials: 1 successes, 1 failures

Experiment Observe $n + m$ new trials: n successes, m failures

Posterior Beta$(2 + n, 2 + m)$

MAP:

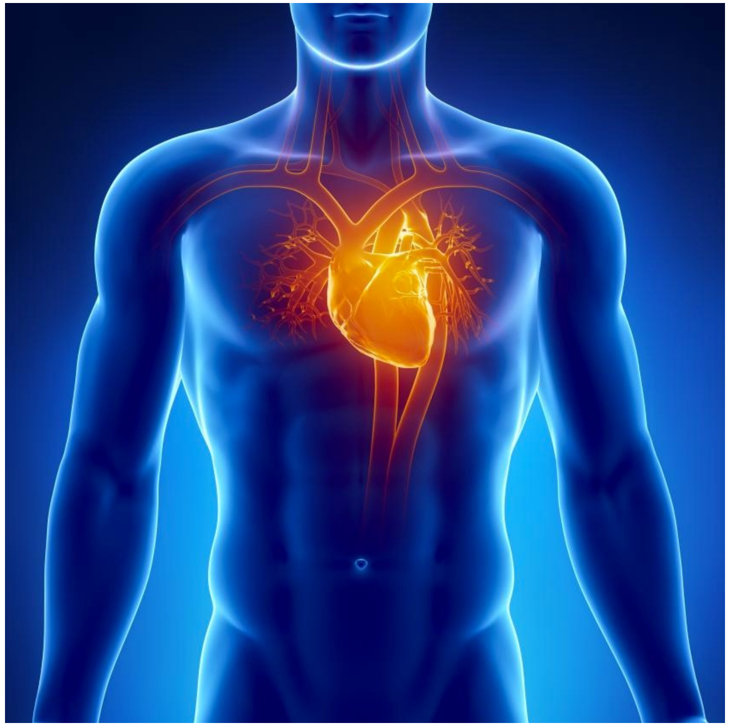
$$p = \frac{n+1}{n+m+2}$$

End Review

The last estimator has risen…

Example Datasets

Heart



Ancestry

Netflix

Training Data

Training Data: assignments all random variables X and Y

Assume IID data:

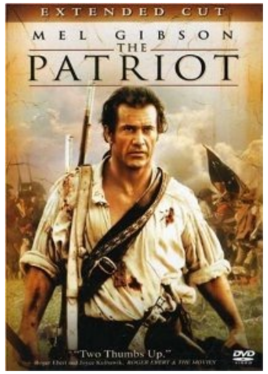
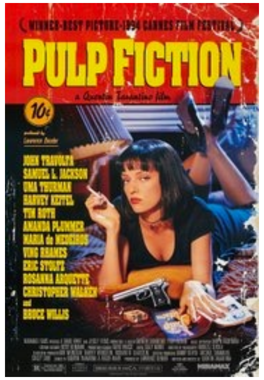
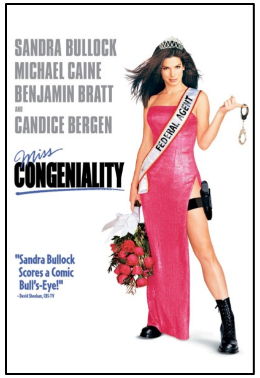
n training datapoints

$$(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \dots (\mathbf{x}^{(n)}, y^{(n)})$$

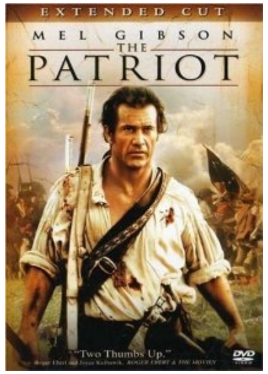
$$m = |\mathbf{x}^{(i)}|$$

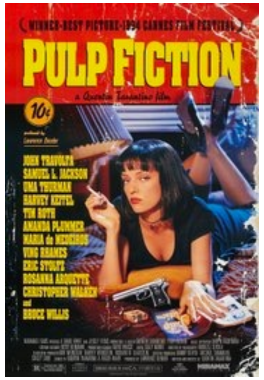
Each datapoint has m features and a single output

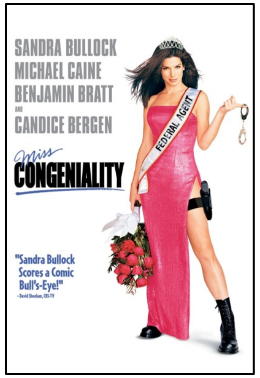
Target Movie “Like” Classification

	Movie 1	Movie 2	…	Movie m	Output
			…		
User 1	1	0		1	1
User 2	1	1		0	0
⋮			⋮		⋮
User n	0	0		1	1

Single Instance



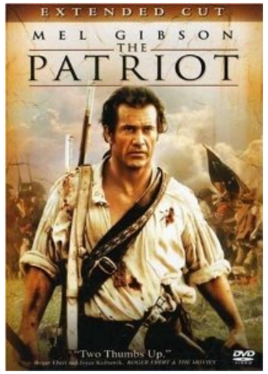


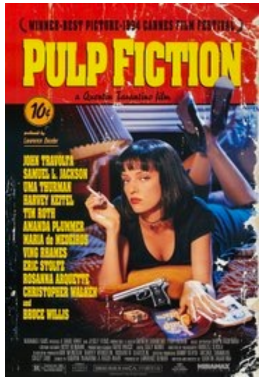


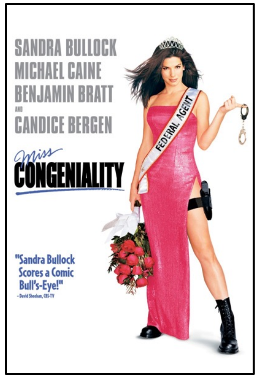
	Movie 1	Movie 2	...	Movie m	Output
User 1	1	0		1	1
User 2	$\boxed{1}$	1		0	0
⋮		⋮			⋮
User n	0	0		1	1

$$(\mathbf{x}^{(i)}, y^{(i)}) \text{ such that } 1 \leq i \leq n$$

Feature Vector



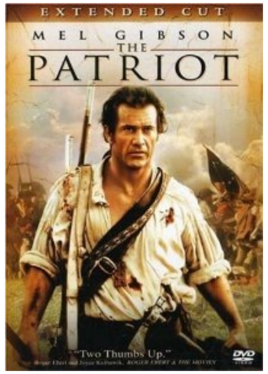


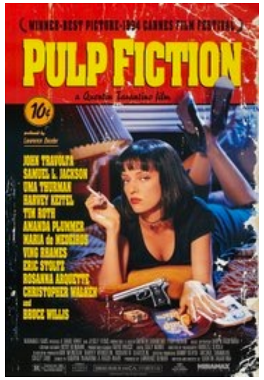


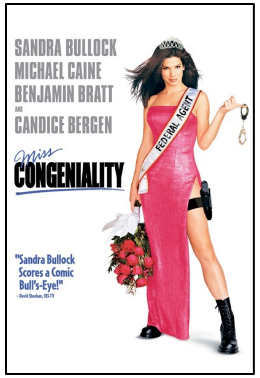
	Movie 1	Movie 2	...	Movie m	Output
User 1	1	0		1	1
User 2	1	1		0	0
		⋮			⋮
User n	0	0		1	1

$$(\boxed{\mathbf{x}^{(i)}}, y^{(i)}) \text{ such that } 1 \le i \le n$$

Output Value



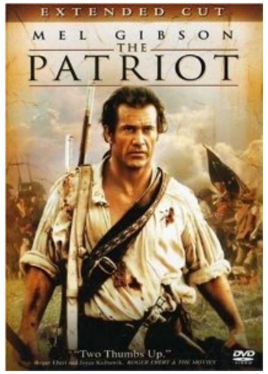


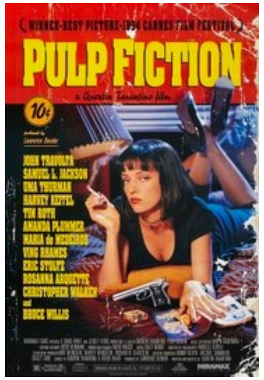


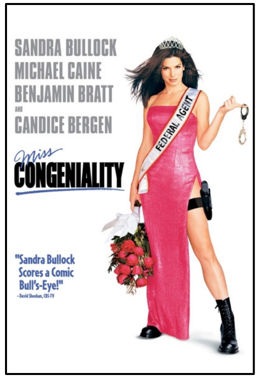
	Movie 1	Movie 2	...	Movie m	Output
User 1	1	0		1	1
User 2	1	1		0	$\boxed{0}$
⋮			⋮		⋮
User n	0	0		1	1

$$(\mathbf{x}^{(i)}, \boxed{y^{(i)}}) \text{ such that } 1 \leq i \leq n$$

Single Feature Value



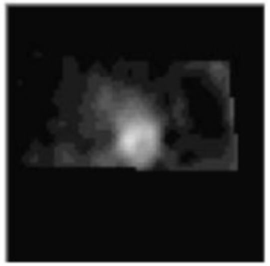
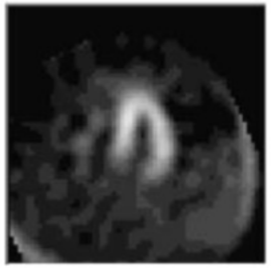




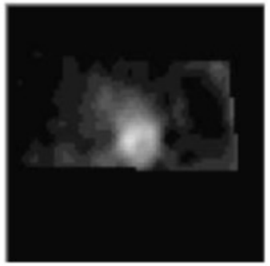
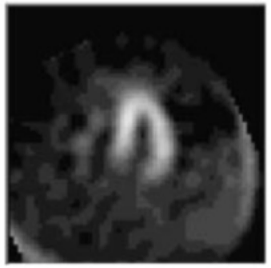
	Movie 1	Movie 2		Movie m	Output
			...		
User 1	1	0		1	1
User 2	1	1		$\boxed{0}$	0
			⋮		⋮
User n	0	0		1	1

In general: $\mathbf{x}_j^{(i)}$ In this case: $\mathbf{x}_m^{(2)}$

Healthy Heart Classifier

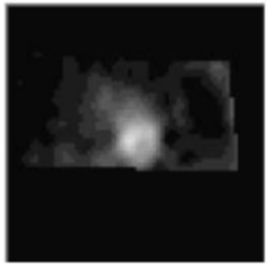
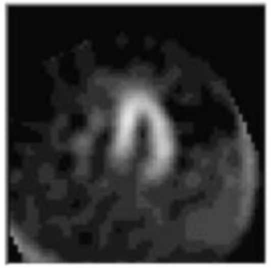
	ROI 1	ROI 2	...	ROI *m*	Output
			...		
Heart 1	0	1		1	0
Heart 2	1	1		1	0
			⋮		⋮
Heart *n*	0	0		0	1

Healthy Heart Classifier

	ROI 1	ROI 2		ROI m	Output
			...		
Heart 1	0	$\boxed{1}$		1	0
Heart 2	1	1		1	0
			⋮		⋮
Heart n	0	0		0	1

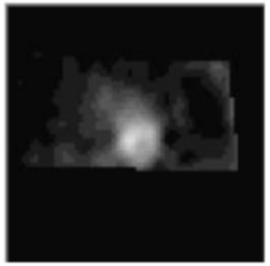
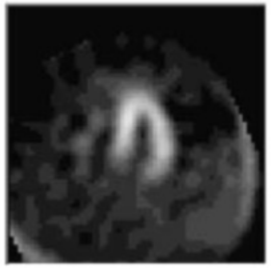
$$x_2^{(1)}$$

Healthy Heart Classifier

	ROI 1	ROI 2		ROI m	Output
			...		
Heart 1	0	1		1	0
Heart 2	1	1		1	0
		$\vdots$			$\vdots$
Heart n	0	0		0	1

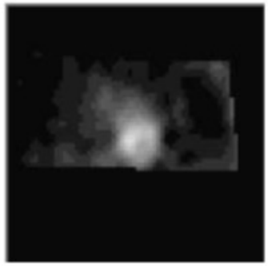
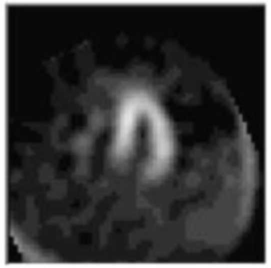
$$(\mathbf{x}^{(2)}, y^{(2)})$$

Healthy Heart Classifier

	ROI 1	ROI 2		ROI *m*	Output
			...		
Heart 1	0	1		1	0
Heart 2	1	1		1	0
		⋮			⋮
Heart *n*	0	0		0	1

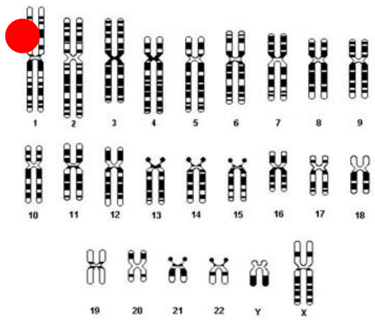
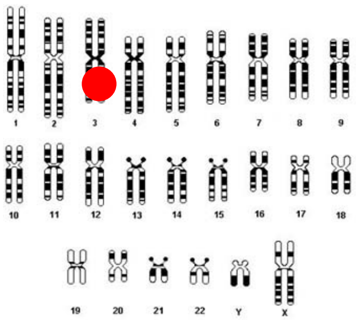
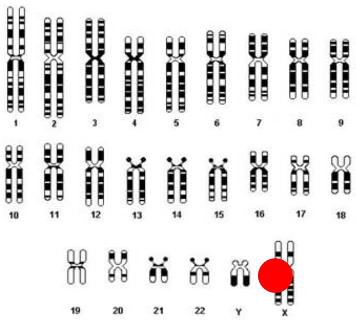
$\mathbf{x}^{(2)}$

Healthy Heart Classifier

	ROI 1	ROI 2		ROI m	Output
			...		
Heart 1	0	1		1	0
Heart 2	1	1		1	$\boxed{0}$
			⋮		⋮
Heart n	0	0		0	1

$$y^{(2)}$$

Ancestry Classifier



...



	SNP 1	SNP 2	SNP *m*	Output
User 1	1	0	1	0
User 2	0	0	1	1
		⋮		⋮
User *n*	1	1	0	1

Regression: Predicting Real Numbers

	Opposing team ELO	Points in last game	…	At Home?	Output # Points
Game 1	84	105		1	120
Game 2	90	102		0	95
⋮			⋮		⋮
Game *n*	74	120		0	115

Training Data

Training Data: assignments all random variables **X** and Y

Assume IID data:

n training datapoints

$$(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \ldots (\mathbf{x}^{(n)}, y^{(n)})$$

$$m = |\mathbf{x}^{(i)}|$$

Each datapoint has m features and a single output

ML is ubiquitous

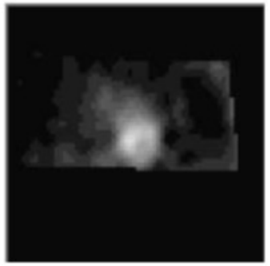
Classification

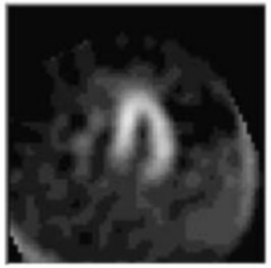
Classification is Building a Harry Potter Hat

$$\hat{y} = 1$$

$$\mathbf{x} = [0, 1, \ldots, 1]$$

Healthy Heart Classifier

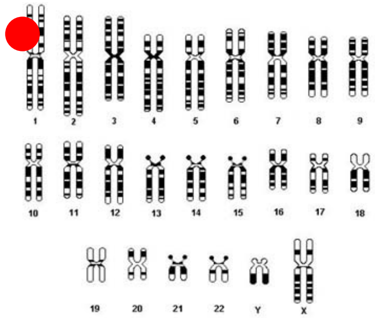
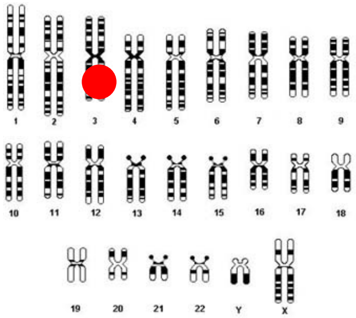
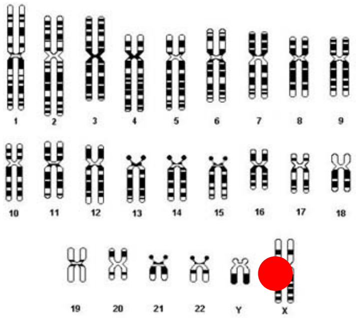




…

	ROI 1	ROI 2	…	ROI *m*	Output
Heart 1	0	1		1	0
Heart 2	1	1		1	0
			⋮		⋮
Heart *n*	0	0		0	1

Ancestry Classifier

	SNP 1	SNP 2	…	SNP m	Output
			...		
User 1	1	0		1	0
User 2	0	0		1	1
			⋮		⋮
User n	1	1		0	1

NETFLIX

And Learn

Target Movie "Like" Classification

Feature 1

Output

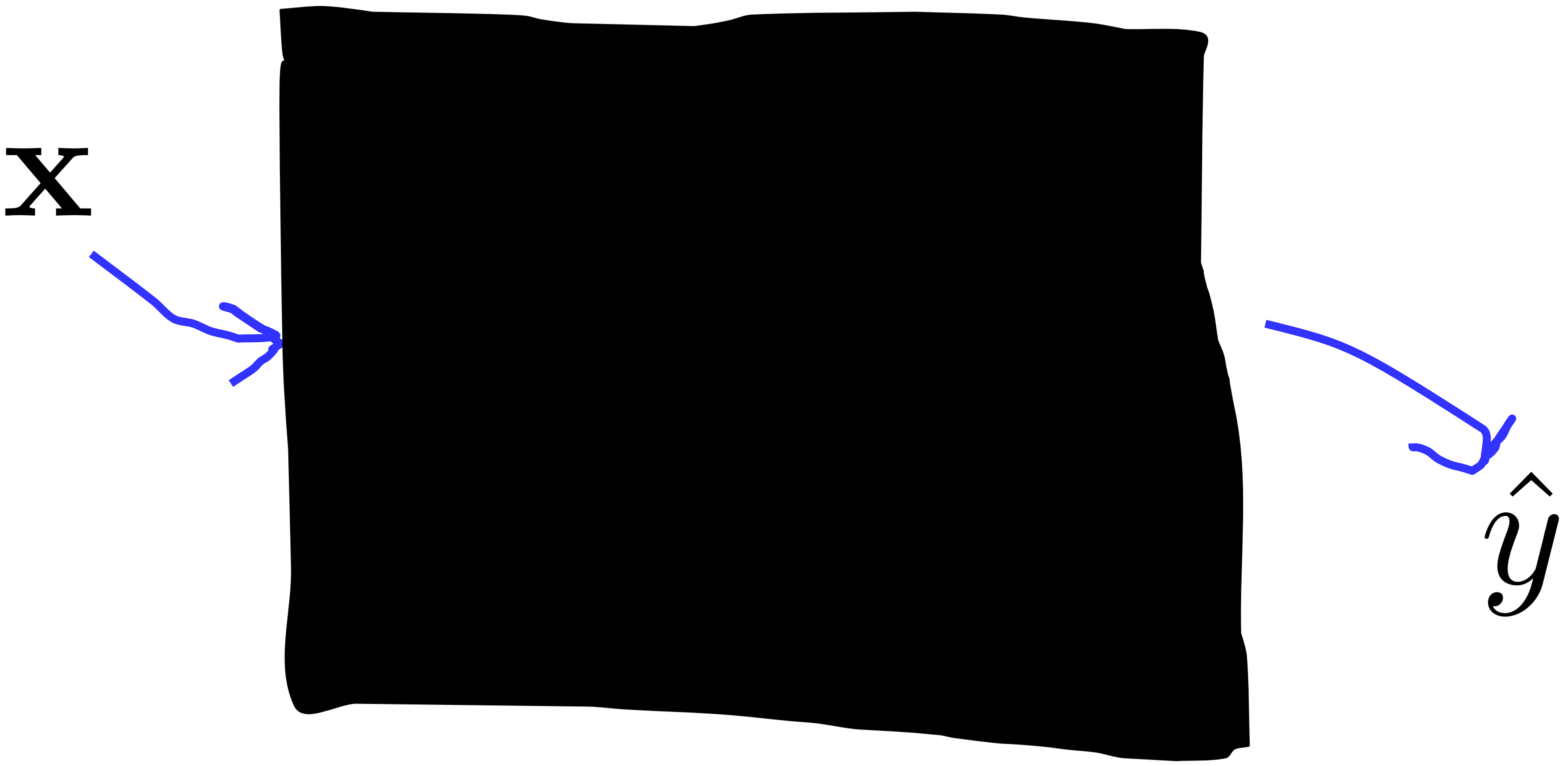
	Feature 1	Output
User 1	1	1
User 2	1	0
⋮		⋮
User *n*	0	1

$$x_j^{(i)} \in \{0, 1\} \qquad y^{(i)} \in \{0, 1\}$$

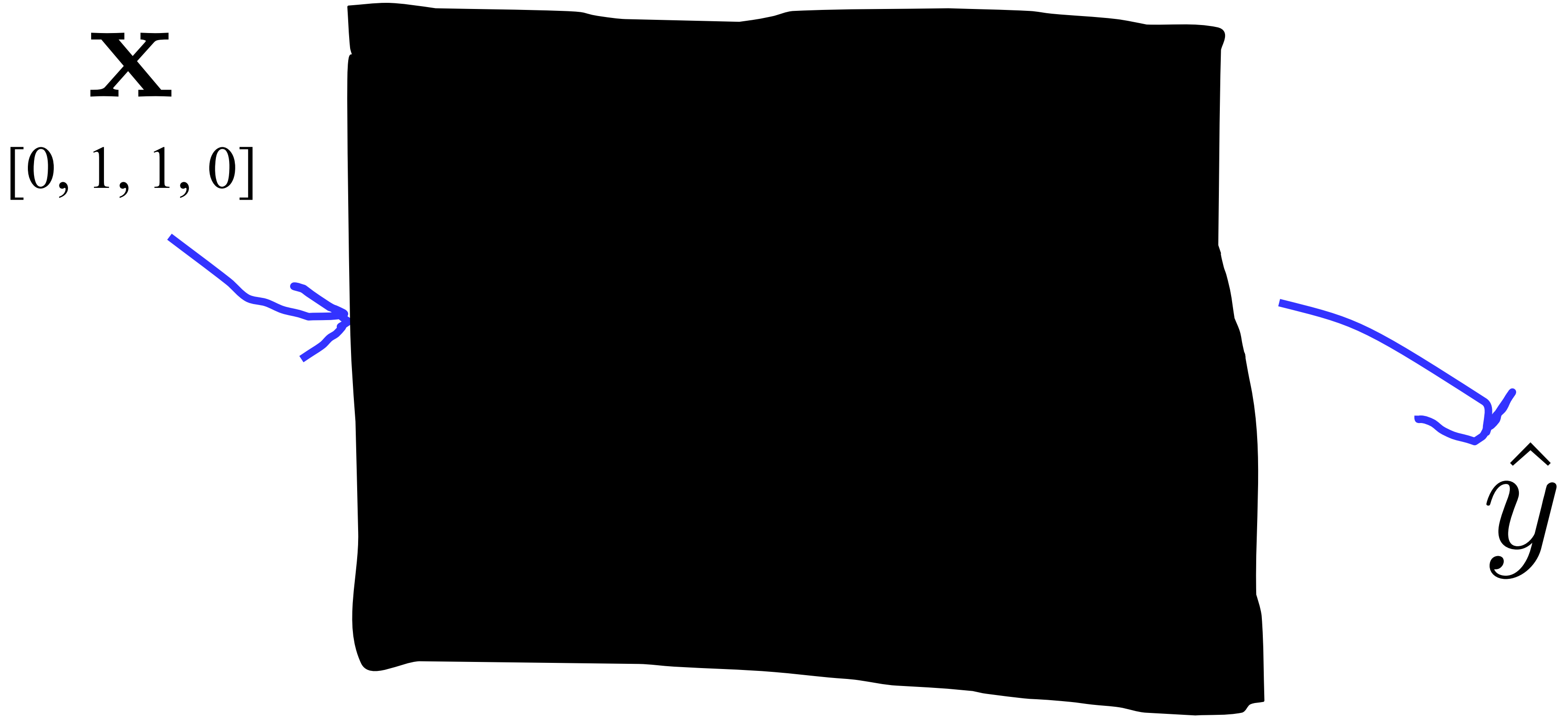
How could we predict the class label:
will the user like life is beautiful?

Fake Algorithm:
Brute Bayes Classifier

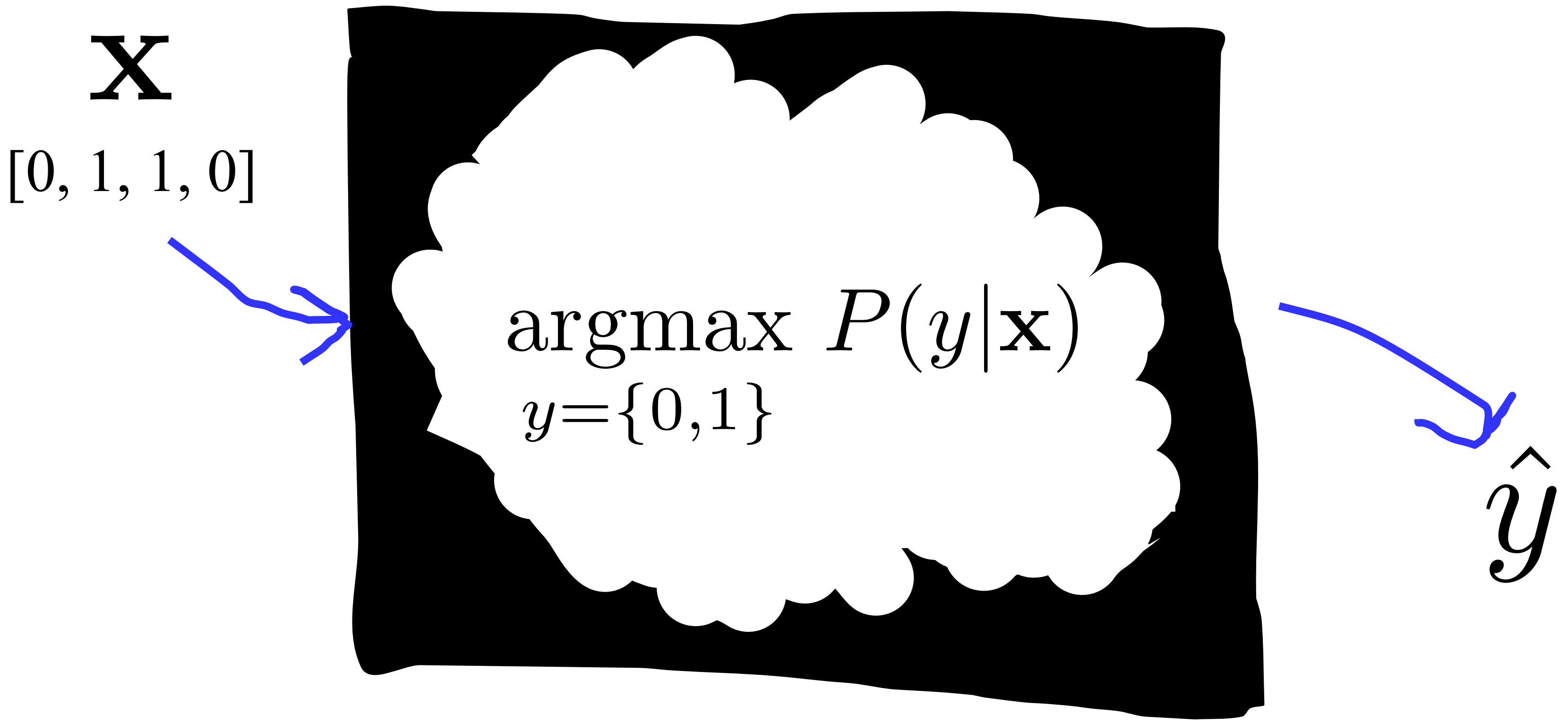
Brute Force Bayes



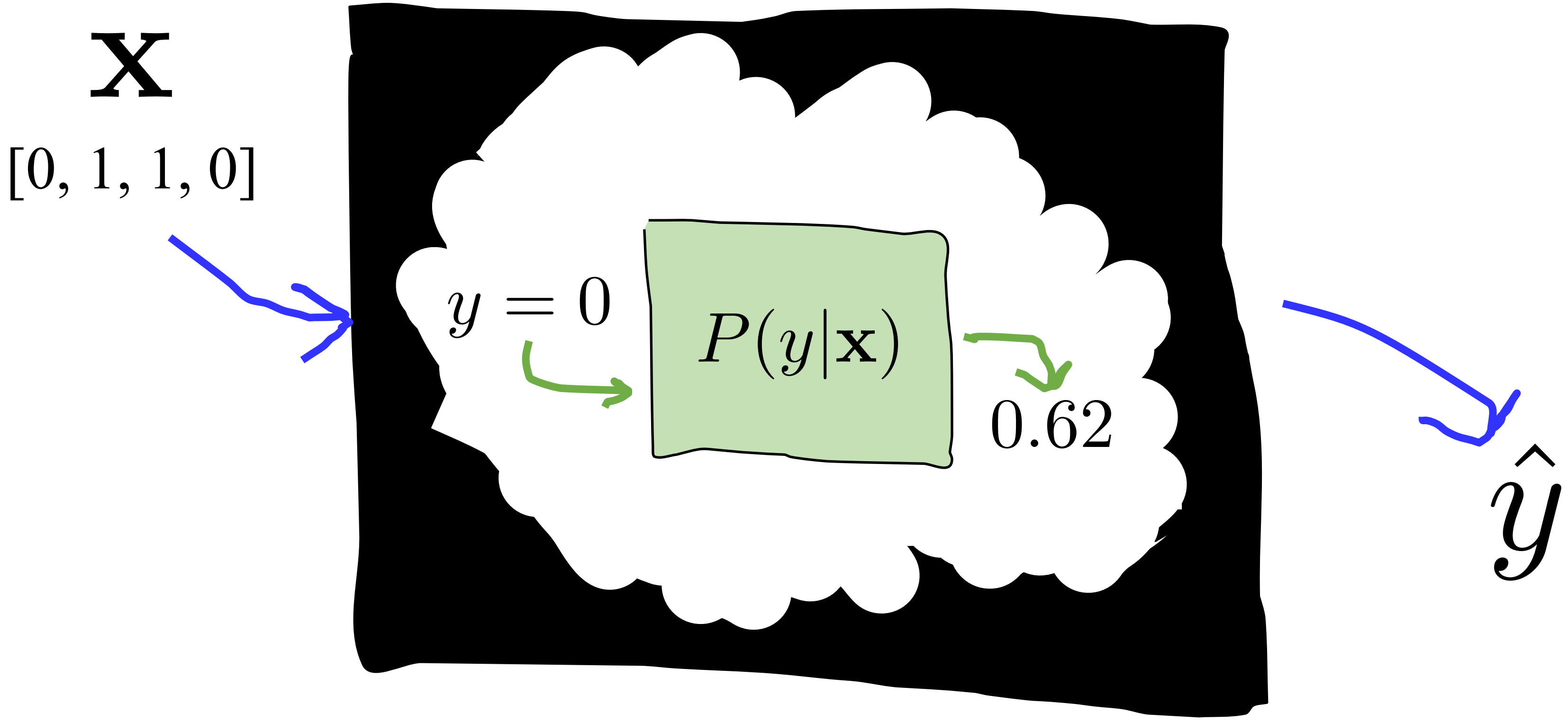
Brute Force Bayes



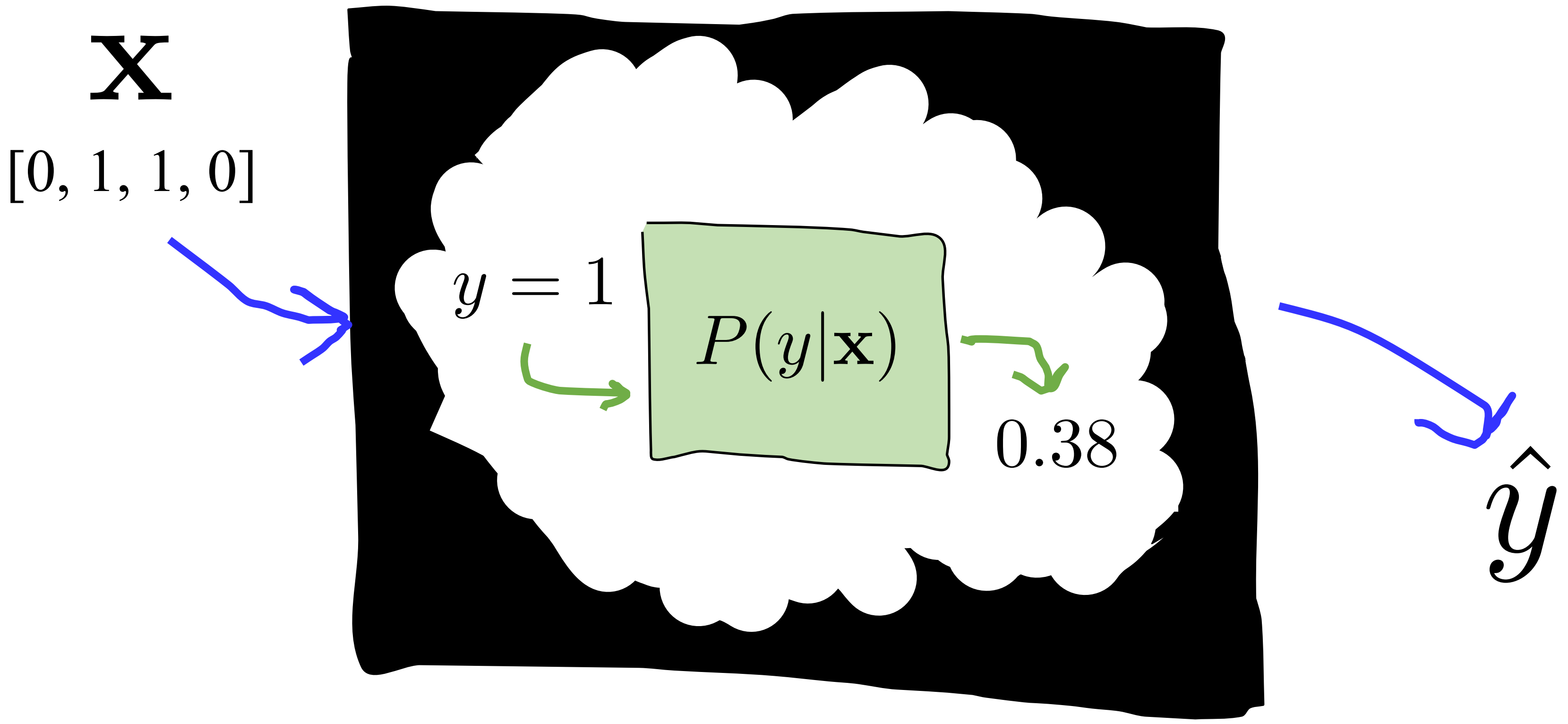
Brute Force Bayes



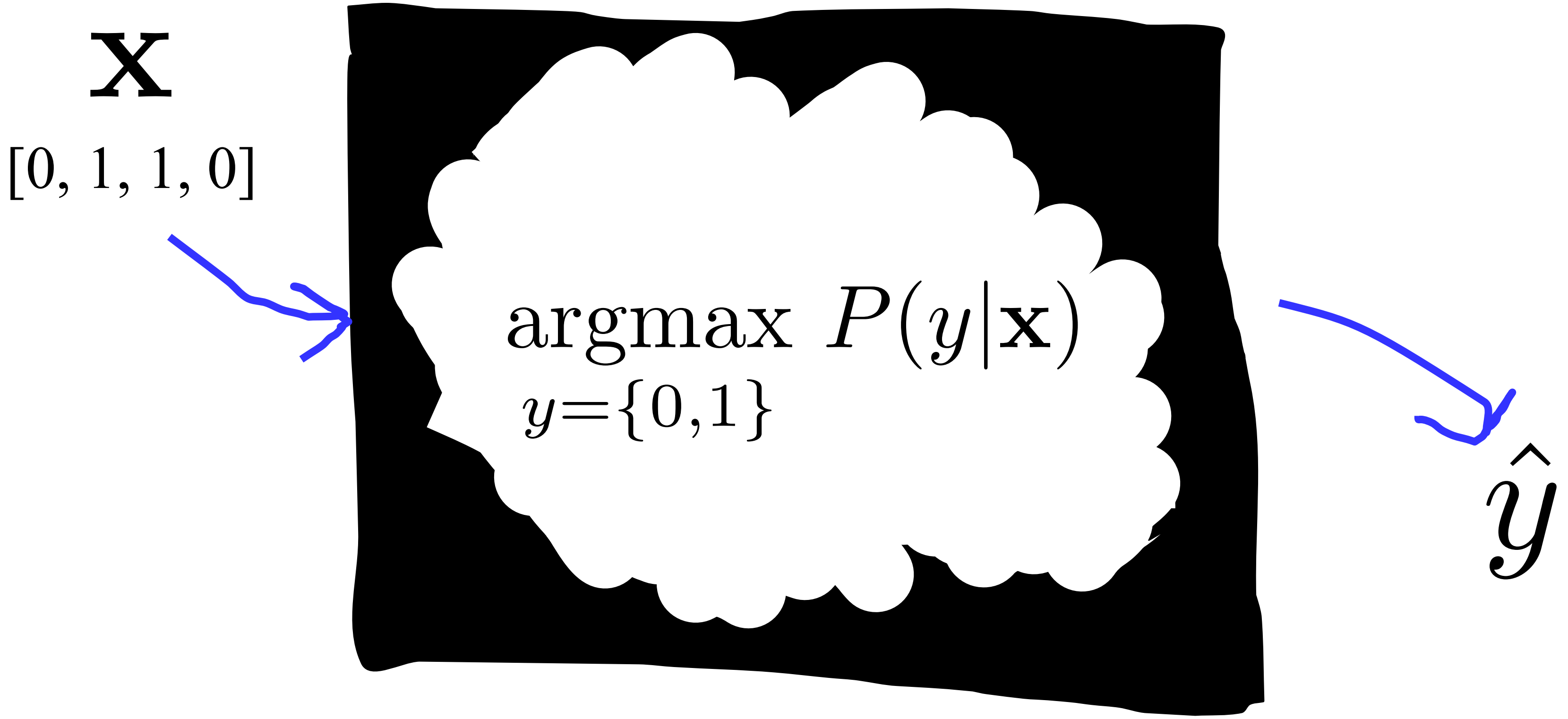
Brute Force Bayes



Brute Force Bayes



Brute Force Bayes



Brute Force Bayes

Prediction: will they like L.I.B.?

$$\hat{y} = \underset{y=\{0,1\}}{\text{argmax}}\ P(y|\mathbf{x})$$

Whether or not they liked Independence day

If y = 1, they like L.I.B.?

Simply chose the class label that is the most likely given the data

This is for one user

Brute Force Bayes

$$\hat{y} = \underset{y=\{0,1\}}{\text{argmax}} \; P(y|\mathbf{x})$$

Simply chose the class label that is the most likely given the data

This is for one user

Brute Force Bayes

$$\hat{y} = \underset{y=\{0,1\}}{\text{argmax}} \; P(y|\mathbf{x})$$

$$= \underset{y=\{0,1\}}{\text{argmax}} \; \frac{P(\mathbf{x}|y)P(y)}{P(\mathbf{x})}$$

$$= \underset{y=\{0,1\}}{\text{argmax}} \; P(\mathbf{x}|y)P(y)$$

Simply chose the class label that is the most likely given the data

This is for one user

* Note how similar this is to Hamilton example ☺

What are the Parameters?

Brute Force Bayes

$$\hat{y} = \underset{y=\{0,1\}}{\operatorname{argmax}} \; \underline{P(\mathbf{x}|y)}\,\underline{P(y)}$$

Conditional probability table

Y = 0

$X_1 = 0$	θ_0
$X_1 = 1$	θ_1

Y = 1

$X_1 = 0$	θ_2
$X_1 = 1$	θ_3

Y = 0	θ_4
Y = 1	θ_5

Learn these during training

Brute Force Bayes

$$\hat{y} = \underset{y=\{0,1\}}{\operatorname{argmax}} \; \underline{P(\mathbf{x}|y)}\,\underline{P(y)}$$

Conditional probability table

X_1 \ Y	0	1
0	θ_0	θ_2
1	θ_1	θ_3

Y = 0	θ_4
Y = 1	θ_5

Learn these during training

Training

	x_1	y
		
User 1	1	1
User 2	0	0
		⋮
User *n*	0	1

$$P(\mathbf{x}|y)$$

X_1 \ Y	0	1
0	θ_0	θ_2
1	θ_1	θ_3

What is $P(X_1 | Y = 0)$?
What is $P(X_1 | Y = 1)$?
Both multinomials with two outcomes

MLE Estimate

	x_1	y
		
User 1	1	1
User 2	0	0
⋮		⋮
User n	0	1

$P(\mathbf{x}|y)$

X_1 \ Y	0	1
0	0.0	0.4
1	1.0	0.6

MLE: Just count

MAP Estimate

	x_1	y
		
User 1	1	1
User 2	0	0
		⋮
User n	0	1

$P(\mathbf{x}|y)$

X_1 \ Y	0	1
0	0.01	0.42
1	0.99	0.58

MAP: Just count and add imaginary trials

Testing

$$\hat{y} = \underset{y=\{0,1\}}{\text{argmax}} \; P(\mathbf{x}|y)P(y)$$

X_1 \ Y	0	1
0	0.01	0.42
1	0.99	0.58

Y = 0	0.21
Y = 1	0.79

Test user: Likes independence day

$$P(x_1 = 1|y = 0)P(y = 0)$$

vs

$$P(x_1 = 1|y = 1)P(y = 1)$$

Testing

$$\hat{y} = \underset{y=\{0,1\}}{\text{argmax}}\ P(\mathbf{x}|y)P(y)$$

Y \ X_1	0	1
0	0.01	0.42
1	$\boxed{0.99}$	0.58

Y = 0	$\boxed{0.21}$
Y = 1	0.79

Test user: Likes independence day

$$\boxed{P(x_1 = 1|y = 0)}\boxed{P(y = 0)}$$

0.208

vs

$$P(x_1 = 1|y = 1)P(y = 1)$$

Testing

$$\hat{y} = \underset{y=\{0,1\}}{\operatorname{argmax}} \; P(\mathbf{x}|y)P(y)$$

X_1 \ Y	0	1
0	0.01	0.42
1	0.99	$\boxed{0.58}$

Y = 0	0.21
Y = 1	$\boxed{0.79}$

Test user: Likes independence day

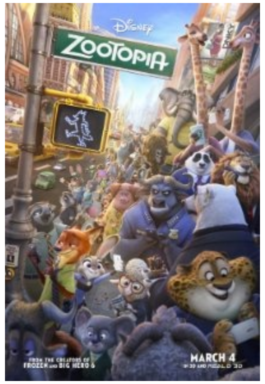
$$P(x_1 = 1|y = 0)P(y = 0)$$ 0.208

vs

$$\boxed{P(x_1 = 1|y = 1)}\boxed{P(y = 1)}$$ 0.458

That was pretty good!

Brute Force Bayes m = 2



	x_1	x_2	y
User 1	1	0	1
User 2	1	0	0
			⋮
User *n*	0	1	1

Brute Force Bayes m = 2

Simply chose the class label that is the most likely given the data

$$\hat{y} = \underset{y=\{0,1\}}{\text{argmax}}\ P(y|\mathbf{x})$$

$$= \underset{y=\{0,1\}}{\text{argmax}}\ \frac{P(\mathbf{x}|y)P(y)}{P(\mathbf{x})}$$

$$= \underset{y=\{0,1\}}{\text{argmax}}\ P(\mathbf{x}|y)P(y)$$

$$P(x_1, x_2|y)$$

Brute Force Bayes

$$\hat{y} = \underset{y=\{0,1\}}{\operatorname{argmax}} \boxed{P(\mathbf{x}|y)} P(y)$$

Y = 0

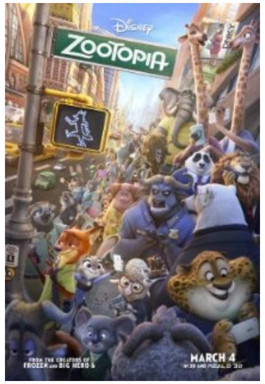
X_2 \ X_1	0	1
0	θ_0	θ_1
1	θ_2	θ_3

Y = 1

X_2 \ X_1	0	1
0	θ_4	θ_5
1	θ_6	θ_7

x_1

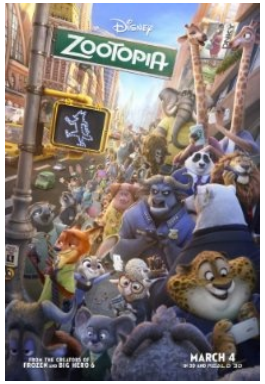
x_2

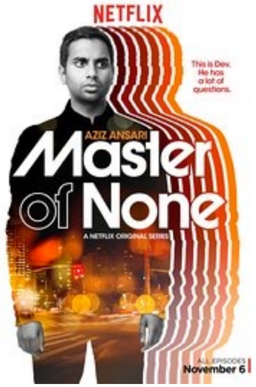


y

Fine

Brute Force Bayes m = 3





	x_1	x_2	x_3	y
User 1	1	0	1	1
User 2	1	0	1	0
				⋮
User *n*	0	1	1	1

Brute Force Bayes m = 3

Simply chose the class label that is the most likely given the data

$$\hat{y} = \underset{y=\{0,1\}}{\text{argmax}} \; P(y|\mathbf{x})$$

$$= \underset{y=\{0,1\}}{\text{argmax}} \; \frac{P(\mathbf{x}|y)P(y)}{P(\mathbf{x})}$$

$$= \underset{y=\{0,1\}}{\text{argmax}} \; P(\mathbf{x}|y)P(y)$$

$$P(x_1, x_2, x_3|y)$$

Brute Force Bayes

$$\hat{y} = \underset{y=\{0,1\}}{\operatorname{argmax}} \boxed{P(\mathbf{x}|y)} P(y)$$

$X_3 = 0$

Y = 0

X_2 \ X_1	0	1
0	θ_0	θ_1
1	θ_2	θ_3

Y = 1

X_2 \ X_1	0	1
0	θ_8	θ_9
1	θ_{10}	θ_{11}

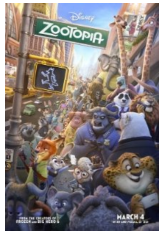
$X_3 = 1$

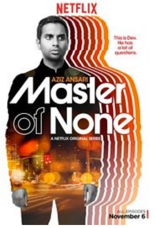
$X_2 = 0$

X_2 \ X_1	0	1
0	θ_4	θ_5
1	θ_6	θ_7

$X_2 = 1$

X_2 \ X_1	0	1
0	θ_{12}	θ_{13}
1	θ_{14}	θ_{15}





And if $m=100$?

Brute Force Bayes m = 100

Simply chose the class label that is the most likely given the data

$$\hat{y} = \underset{y=\{0,1\}}{\text{argmax}}\ P(y|\mathbf{x})$$

$$= \underset{y=\{0,1\}}{\text{argmax}}\ \frac{P(\mathbf{x}|y)P(y)}{P(\mathbf{x})}$$

$$= \underset{y=\{0,1\}}{\text{argmax}}\ P(\mathbf{x}|y)P(y)$$

$$P(x_1, x_2, x_3, \ldots, x_{100}|y)$$

Oops… Number of atoms in the univserse

What is the big O for # parameters?
m = # features.

Big O of Brute Force Joint

What is the big O for # parameters?
m = # features.

$$O(2^m)$$

Assuming each feature is binary...

Not going to cut it!

Pedagogical Pause

What is the problem here?

$$\hat{y} = \underset{y=\{0,1\}}{\text{argmax}}\ P(y|\mathbf{x})$$

$$= \underset{y=\{0,1\}}{\text{argmax}}\ \frac{P(\mathbf{x}|y)P(y)}{P(\mathbf{x})}$$

$$= \underset{y=\{0,1\}}{\text{argmax}}\ P(\mathbf{x}|y)P(y)$$

$$P(\mathbf{x}|y) = P(x_1, x_2, \ldots, x_m|y)$$

Naïve Bayes Assumption

$$\hat{y} = \underset{y=\{0,1\}}{\operatorname{argmax}} \; P(y|\mathbf{x})$$

$$= \underset{y=\{0,1\}}{\operatorname{argmax}} \; \frac{P(\mathbf{x}|y)P(y)}{P(\mathbf{x})}$$

$$= \underset{y=\{0,1\}}{\operatorname{argmax}} \; P(\mathbf{x}|y)P(y)$$

$$P(\mathbf{x}|y) = P(x_1, x_2, \ldots, x_m|y)$$

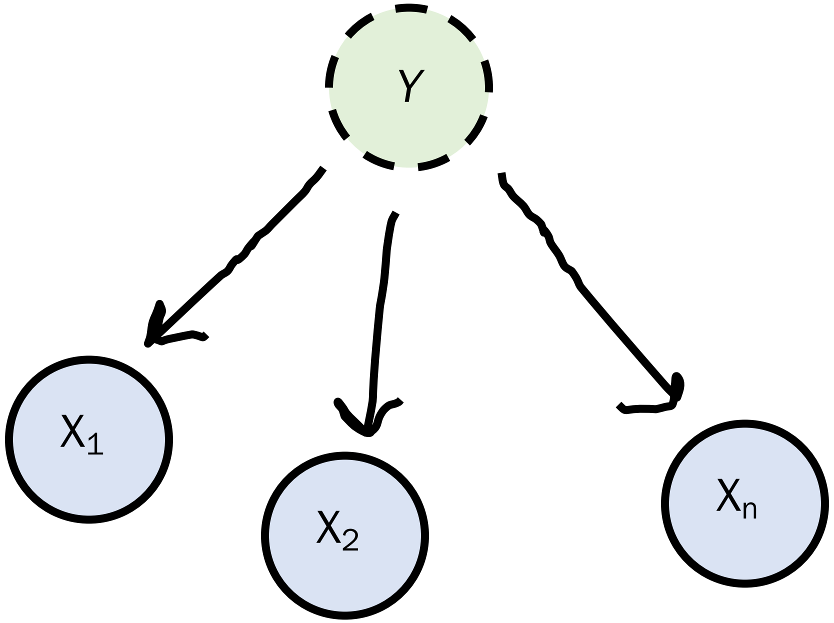
$$= \prod_i P(x_i|y)$$

The Naïve Bayes assumption

Naïve Bayes Assumption:

$$P(\mathbf{x}|y) = \prod_i P(x_i|y)$$

Naïve Bayes as a Model



Class Label

$$Y \sim \text{Bern}$$

Data distribution

$$X_i | Y \sim \text{Bern}$$

$$P(\mathbf{x}, y) = P(y) \prod_i P(x_i | y)$$

Naïve Bayes Classifier

Naïve Bayes

Our prediction for y

Is a function of x

That choses the best value of y given x

$$\hat{y} = g(\mathbf{x}) = \underset{y \in \{0,1\}}{\operatorname{argmax}} \ \hat{P}(y|\mathbf{x})$$

$$= \underset{y \in \{0,1\}}{\operatorname{argmax}} \ \hat{P}(\mathbf{x}|y)\hat{P}(y)$$

Bayes rule!

$$= \underset{y}{\operatorname{argmax}} \ \left(\prod_{i=1}^{n} \hat{P}(x_i|y)\right)\hat{P}(y)$$

Naïve Bayes Assumption

$$= \underset{y}{\operatorname{argmax}} \ \log \hat{P}(y) + \sum_{i=1}^{m} \log \hat{P}(x_i|y)$$

This log version is useful for numerical stability

Computing Probabilities from Data

Various probabilities you will need to compute for Naive Bayesian Classifier (using MLE here):

$$\hat{p}(X_i = 1 | Y = 0) = \frac{(\text{\# training examples where } X_i = 1 \text{ and } Y = 0)}{(\text{\# training examples where } Y = 0)}$$

$$\hat{p}(Y = 1) = \frac{(\text{\# training examples where } Y = 1)}{(\text{\# training examples})}$$

Computing Probabilities from Data With Laplace

Various probabilities you will need to compute for Naive Bayesian Classifier (using MAP with Laplace):

$$\hat{p}(X_i = 1 | Y = 0) = \frac{(\#\text{ training examples where } X_i = 1 \text{ and } Y = 0) \; +1}{(\#\text{ training examples where } Y = 0) \; +2}$$

$$\hat{p}(Y = 1) = \frac{(\#\text{ training examples where } Y = 1) \; +1}{(\#\text{ training examples}) \; +2}$$

Naïve Bayes Example

Predict Y based on observing variables X_1 and X_2

- X_1 and X_2 are both indicator variables
  - X_1 denotes "likes Star Wars", X_2 denotes "likes Harry Potter"
- Y is indicator variable: "likes Lord of the Rings"
  - Use training data to estimate params: $\hat{P}(x_i|y)$ $\hat{P}(y)$

Y \ X_1	0	1	MLE estimates	
0	3	10	0.23	0.77
1	4	13	0.24	0.76

Y \ X_2	0	1	MLE estimates	
0	5	8	0.38	0.62
1	7	10	0.41	0.59

Y	#	MLE est.
0	13	0.43
1	17	0.57

- Say someone likes **Star Wars (X_1 = 1),** but not **Harry Potter (X_2 = 0)**
- Will they like "Lord of the Rings"? Need to predict Y:

$$\hat{y} = \underset{y \in \{0,1\}}{\operatorname{argmax}}\ \hat{P}(\mathbf{x}|y)\hat{P}(y) = \underset{y \in \{0,1\}}{\operatorname{argmax}}\ \hat{P}(x_1|y)\hat{P}(x_2|y)\hat{P}(y)$$

Naïve Bayes Example

Predict Y based on observing variables X_1 and X_2

- X_1 and X_2 are both indicator variables
  - X_1 denotes "likes Star Wars", X_2 denotes "likes Harry Potter"
- Y is indicator variable: "likes Lord of the Rings"
  - Use training data to estimate params: $\hat{P}(x_i|y)$ $\hat{P}(y)$

Y \ X_1	0	1	MLE estimates	
0	3	10	0.23	0.77
1	4	13	0.24	0.76

Y \ X_2	0	1	MLE estimates	
0	5	8	0.38	0.62
1	7	10	0.41	0.59

Y	#	MLE est.
0	13	0.43
1	17	0.57

- Say someone likes **Star Wars (X_1 = 1),** but not **Harry Potter (X_2 = 0)**
- Will they like "Lord of the Rings"? Need to predict Y:

$$\hat{y} = \underset{y \in \{0,1\}}{\text{argmax}} \ \hat{P}(X_1 = x_1|Y = y)\hat{P}(X_2 = x_2|Y = y)\hat{P}(Y = y)$$

Naïve Bayes Example

Predict Y based on observing variables X_1 and X_2

- X_1 and X_2 are both indicator variables
  - X_1 denotes "likes Star Wars", X_2 denotes "likes Harry Potter"
- Y is indicator variable: "likes Lord of the Rings"
  - Use training data to estimate params: $\hat{P}(x_i|y)$ $\hat{P}(y)$

Y \ X_1	0	1	MLE estimates	
0	3	10	0.23	0.77
1	4	13	0.24	0.76

Y \ X_2	0	1	MLE estimates	
0	5	8	0.38	0.62
1	7	10	0.41	0.59

Y	#	MLE est.
0	13	0.43
1	17	0.57

- Say someone likes **Star Wars (X_1 = 1),** but not **Harry Potter (X_2 = 0)**
- Will they like "Lord of the Rings"? Need to predict Y:

$$\hat{y} = \underset{y \in \{0,1\}}{\operatorname{argmax}} \hat{P}(X_1 = 1|Y = y)\hat{P}(X_2 = 0|Y = y)\hat{P}(Y = y)$$

One Slide to Rule them All

Y \ X_1	0	1	MLE estimates	
0	3	10	0.23	0.77
1	4	13	0.24	0.76

Y \ X_2	0	1	MLE estimates	
0	5	8	0.38	0.62
1	7	10	0.41	0.59

Y	#	MLE est.
0	13	0.43
1	17	0.57

$$\hat{y} = \underset{y \in \{0,1\}}{\operatorname{argmax}} \ \hat{P}(X_1 = 1|Y = y)\hat{P}(X_2 = 0|Y = y)\hat{P}(Y = y)$$

- Let Y = 0

$$\hat{P}(X_1 = 1|Y = 0)\hat{P}(X_2 = 0|Y = 0)\hat{P}(Y = 0)$$
$$= (0.77)(0.38)(0.43) = 0.126$$

- Let Y = 1

$$\hat{P}(X_1 = 1|Y = 1)\hat{P}(X_2 = 0|Y = 1)\hat{P}(Y = 1)$$
$$= (0.76)(0.41)(0.57) = 0.178$$

Since term is greatest when Y = 1, we predict Ŷ = 1

$$P(Y = 1) = K \cdot 0.178 \qquad P(Y = 0) = K \cdot 0.126 \qquad K = \frac{1}{0.126 + 0.178}$$

MAP Naïve Bayes

Predict Y based on observing variables X_1 and X_2

- X_1 and X_2 are both indicator variables
  - X_1 denotes "likes Star Wars", X_2 denotes "likes Harry Potter"
- Y is indicator variable: "likes Lord of the Rings"
  - Use training data to estimate PMFs: $\hat{P}(x_i|y)$ $\hat{P}(y)$

Y \ X_1	0	1	MAP estimates
0	3	10	
1	4	13	

Y \ X_2	0	1	MAP estimates
0	5	8	
1	7	10	

Y	#	MAP est.
0	13	
1	17	

What prior?

MAP Naïve Bayes

Predict Y based on observing variables X_1 and X_2

- X_1 and X_2 are both indicator variables
  - X_1 denotes "likes Star Wars", X_2 denotes "likes Harry Potter"
- Y is indicator variable: "likes Lord of the Rings"
  - Use training data to estimate PMFs: $\hat{P}(x_i|y)$ $\hat{P}(y)$

Y \ X_1	0	1	MAP estimates
0	3	10	
1	4	13	

Y \ X_2	0	1	MAP estimates
0	5	8	
1	7	10	

Y	#	MAP est.
0	13	
1	17	

Laplace!

$$p_i = \frac{n_i + 1}{n + 2}$$

MAP Naïve Bayes

Predict Y based on observing variables X_1 and X_2

- X_1 and X_2 are both indicator variables
  - X_1 denotes "likes Star Wars", X_2 denotes "likes Harry Potter"
- Y is indicator variable: "likes Lord of the Rings"
  - Use training data to estimate PMFs: $\hat{P}(x_i|y)$ $\hat{P}(y)$

Y \ X_1	0	1	MAP estimates	
0	3	10	0.27	0.73
1	4	13	0.26	0.74

Y \ X_2	0	1	MAP estimates	
0	5	8	0.4	0.6
1	7	10	0.42	0.58

Y	#	MAP est.
0	13	0.45
1	17	0.55

Laplace!

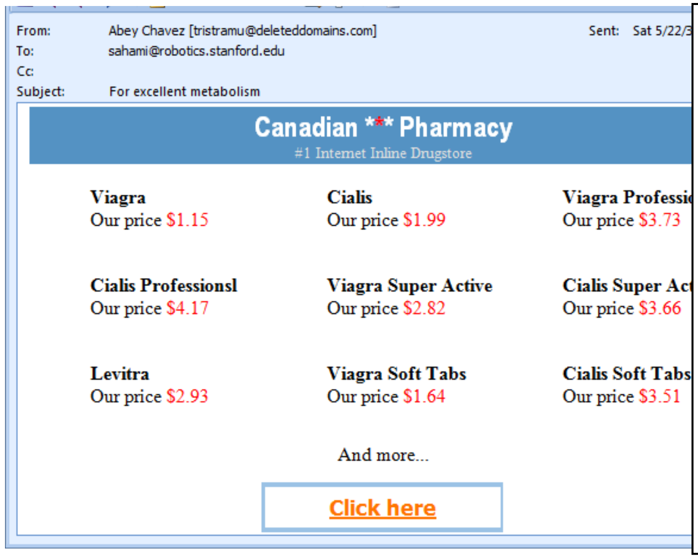
$$p_i = \frac{n_i + 1}{n + 2}$$

Training Naïve Bayes, is estimating parameters for a multinomial (or bernoulli).

Thus training is just counting.

What is Bayes Doing in my Mail Server

This is spam:



Let's get Bayesian on your spam:

```
Content analysis details:   (49.5 hits, 7.0 required)
 0.9 RCVD_IN_PBL            RBL: Received via a relay in Spamhaus PBL
                            [93.40.189.29 listed in zen.spamhaus.org]
 1.5 URIBL_WS_SURBL         Contains an URL listed in the WS SURBL blocklist
                            [URIs: recragas.cn]
 5.0 URIBL_JP_SURBL         Contains an URL listed in the JP SURBL blocklist
                            [URIs: recragas.cn]
 5.0 URIBL_OB_SURBL         Contains an URL listed in the OB SURBL blocklist
                            [URIs: recragas.cn]
 5.0 URIBL_SC_SURBL         Contains an URL listed in the SC SURBL blocklist
                            [URIs: recragas.cn]
 2.0 URIBL_BLACK            Contains an URL listed in the URIBL blacklist
                            [URIs: recragas.cn]
 8.0 BAYES_99               BODY: Bayesian spam probability is 99 to 100%
                            [score: 1.0000]
```

A Bayesian Approach to Filtering Junk E-Mail

Mehran Sahami* Susan Dumais† David Heckerman† Eric Horvitz†

*Gates Building 1A
Computer Science Department
Stanford University
Stanford, CA 94305-9010
sahami@cs.stanford.edu

†Microsoft Research
Redmond, WA 98052-6399
{sdumais, heckerma, horvitz}@microsoft.com

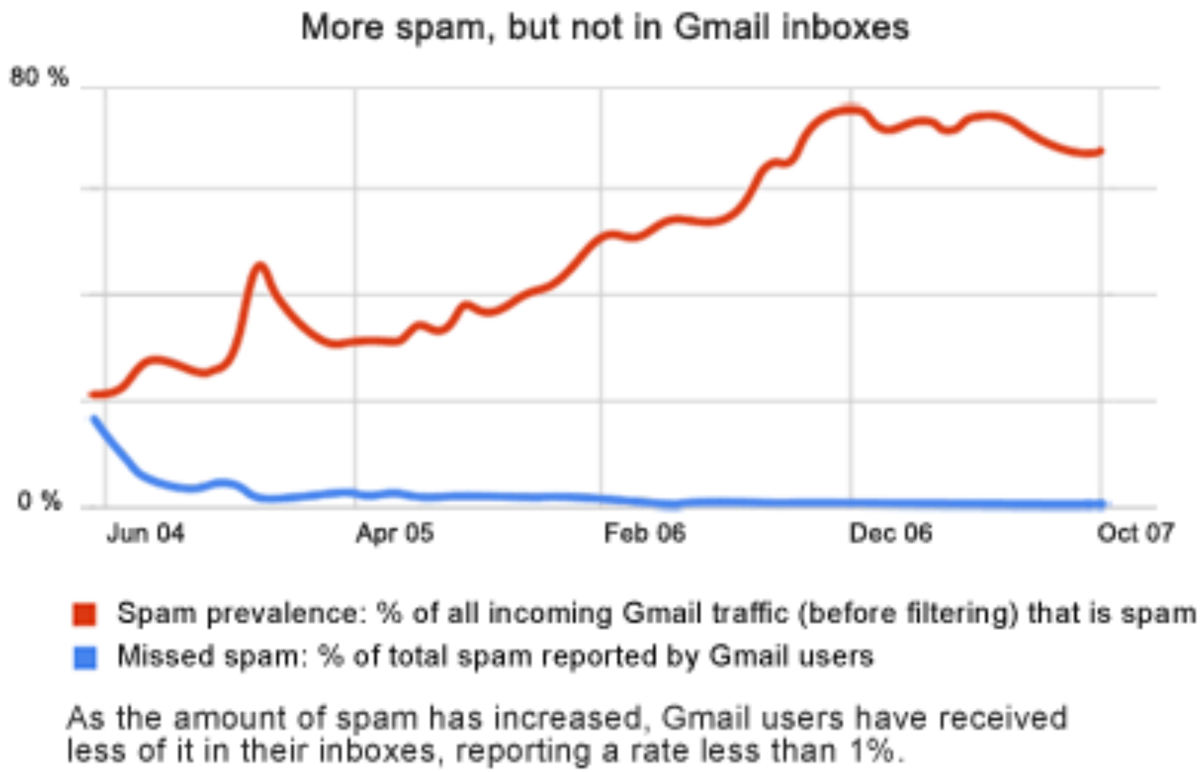
Abstract

In addressing the growing problem of junk E-mail on the Internet, we examine methods for the automated

contain offensive material (such as graphic pornography), there is often a higher cost to users of actually viewing this mail than simply the time to sort out the junk.

Spam, Spam… Go Away!

The constant battle with spam



"And machine-learning algorithms developed to merge and rank large sets of Google search results allow us to combine hundreds of factors to classify spam."

Source: `http://www.google.com/mail/help/fightspam/spamexplained.html`

Email Classification

Want to predict if an email is spam or not

- Start with the input data
  - Consider a lexicon of m words (Note: in English $m \approx 100{,}000$)
  - Define m indicator variables $\mathbf{X} = \langle X_1, X_2, \ldots, X_m \rangle$
  - Each variable X_i denotes if word i appeared in a document or not
  - Note: m is huge, so make "Naive Bayes" assumption
- Define output classes Y to be: {spam, non-spam}
- Given training set of N previous emails
  - For each email message, we have a training instance: $\mathbf{X} = \langle X_1, X_2, \ldots, X_m \rangle$ noting for each word, if it appeared in email
  - Each email message is also marked as spam or not (value of Y)

Training the Classifier

Given N training pairs:

$(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \ldots, (\mathbf{x}^{(n)}, y^{(n)})$

Learning

- Estimate probabilities $\mathrm{P}(y)$ and $\mathrm{P}(x_i \mid y)$ for all i
  - Many words are likely to not appear at all in given set of email
- Laplace estimate: $\hat{p}(X_i = 1 \mid Y = spam)_{Laplace} = \dfrac{(\#\,\text{spam emails with word } i) + 1}{\text{total}\,\#\,\text{spam emails} + 2}$

Classification

- For a new email, generate $\mathbf{X} = \langle X_1, X_2, \ldots, X_m \rangle$
- Classify as spam or not using: $\hat{y} = \underset{y \in \{0,1\}}{\operatorname{argmax}}\ \hat{P}(\mathbf{x}|y)\hat{P}(y)$
- Employ Naive Bayes assumption: $P(\mathbf{x}|y) = \prod_i P(x_i|y)$

Training Naïve Bayes, is estimating parameters for Bernoullis.

Thus it is just counting.

How Does This Do?

After training, can test with another set of data

- "Testing" set also has known values for Y, so we can see how often we were right/wrong in predictions for Y
- Spam data
  - Email data set: 1789 emails (1578 spam, 211 non-spam)
  - First, 1538 email messages (by time) used for training
  - Next 251 messages used to test learned classifier
- Criteria:
  - <u>Precision</u> = # *correctly* predicted class Y/ # predicted class Y
  - <u>Recall</u> = # *correctly* predicted class Y / # real class Y messages

	Spam		Non-spam	
	Precision	**Recall**	**Precision**	**Recall**
Words only	**97.1%**	**94.3%**	**87.7%**	**93.4%**
Words + add'l features	**100%**	**98.3%**	**96.2%**	**100%**

Naïve Bayes Classification

Training Naïve Bayes With MLE

Various probabilities you will need to compute for Naive Bayesian Classifier (using MLE here):

$$\hat{p}(X_i = 1 | Y = 0) = \frac{(\#\text{ training examples where } X_i = 1 \text{ and } Y = 0)}{(\#\text{ training examples where } Y = 0)}$$

$$\hat{p}(Y = 1) = \frac{(\#\text{ training examples where } Y = 1)}{(\#\text{ training examples})}$$

Training Naïve Bayes With Laplace

Various probabilities you will need to compute for Naive Bayesian Classifier (using MAP with Laplace):

$$\hat{p}(X_i = 1 | Y = 0) = \frac{(\#\text{ training examples where } X_i = 1 \text{ and } Y = 0) + 1}{(\#\text{ training examples where } Y = 0) + 2}$$

$$\hat{p}(Y = 1) = \frac{(\#\text{ training examples where } Y = 1) + 1}{(\#\text{ training examples}) + 2}$$

Naïve Bayes Prediction

That choses the best
value of y given x

$$\hat{y} = \underset{y}{\operatorname{argmax}} \; \log \hat{P}(y) + \sum_{i=1}^{m} \log \hat{P}(x_i|y)$$