Naïve Bayes

Chris Piech
CS109, Stanford University

Review

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

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

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

MLE for Multinomial

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

$$= \underset{\theta}{\text{argmax}} \left(\sum_i \log f(x^{(i)}|\theta) \right)$$

MLE estimate of the probability of outcome i

number of observed outcomes of type i

$$p_i = \frac{n_i}{n}$$

number of observations

MAP for Multinomial, Leplace Prior

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

$$= \underset{\theta}{\text{argmax}} \left(\log(g(\theta)) + \sum_{i=1}^{n} \log(f(x^{(i)}|\theta)) \right)$$

MAP estimate of the probability of outcome i

number of observed outcomes of type i

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

number of observations

number of outcome types

Parent's Club

The last estimator has risen…

Machine Learning

Machine Learning: Formally

- Many different forms of "Machine Learning"
  - We focus on the problem of *prediction*
- Want to make a prediction based on observations
  - Vector $\mathbf{X}$ of *m* observed variables: $\mathbf{X} = [X_1 \ldots X_m]$
  - Based on observed $\mathbf{X}$, want to predict unseen variable Y
    - Y called "output feature/variable" (or the "dependent variable")
  - Seek to "learn" a function $g(\mathbf{X})$ to predict Y:
    - $\hat{Y} = g(\mathbf{X})$
    - When Y is discrete, prediction of Y is called "classification"
    - When Y is continuous, prediction of Y is called "regression"

Training Data

Assume IID data:

n training datapoints

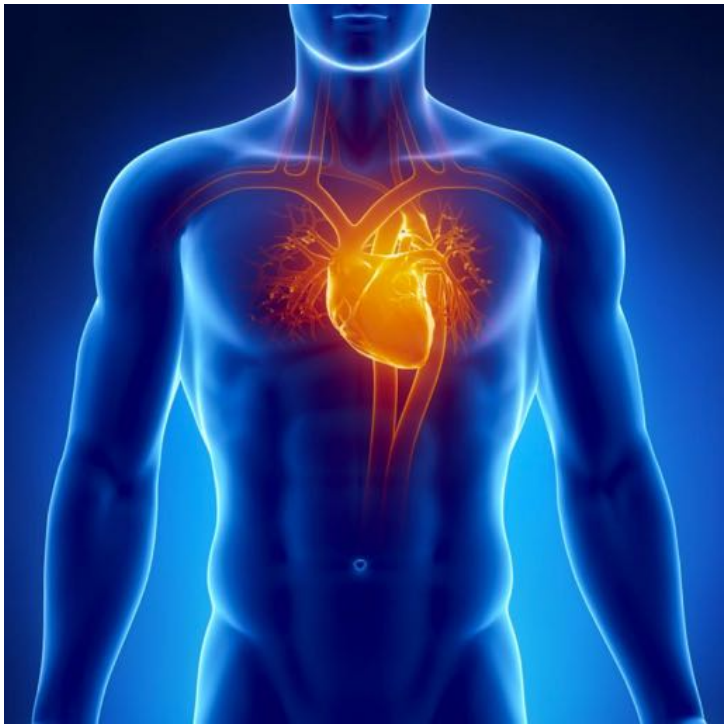
$$(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \ldots (\mathbf{x}^{(n)}, y^{(n)})$$

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

Each datapoint has m features and a single output

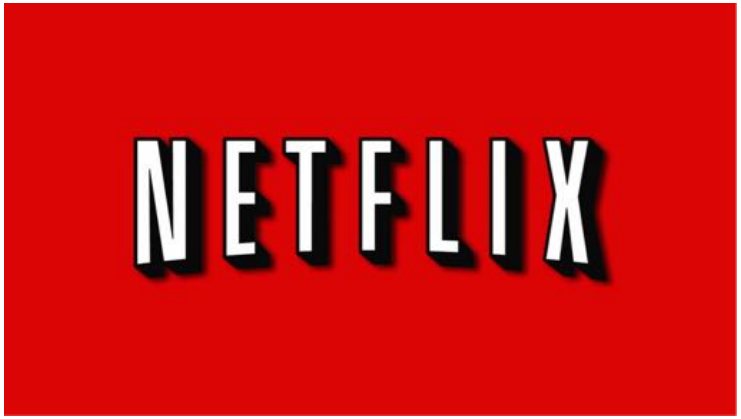
Example Datasets

Heart

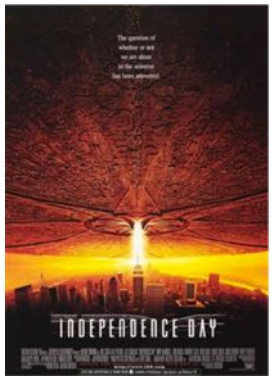
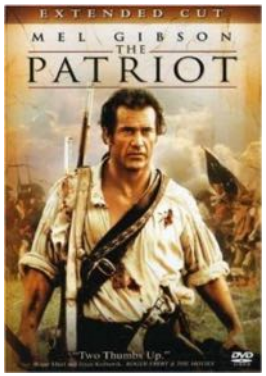
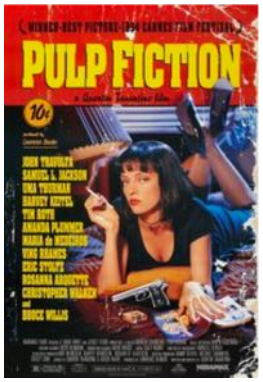
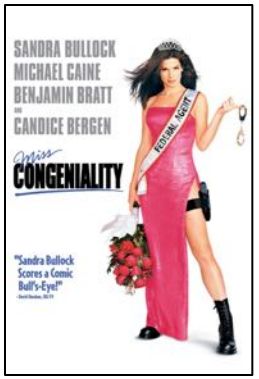


Ancestry

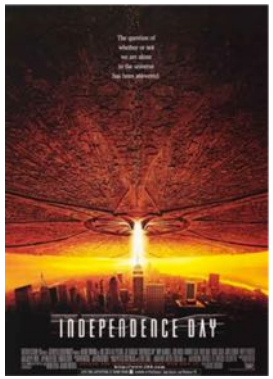
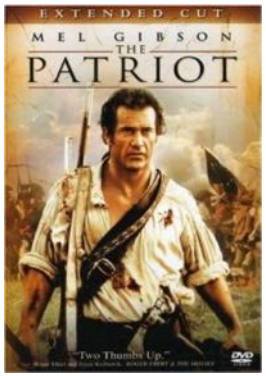
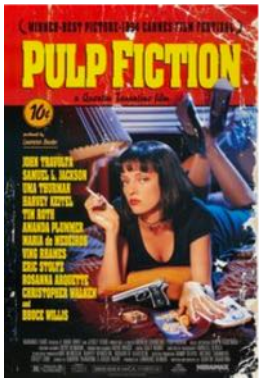
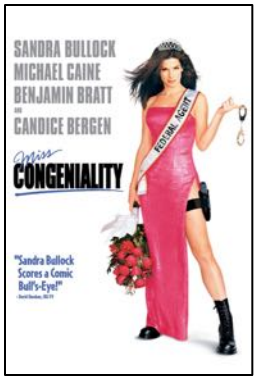
Netflix



Target Movie "Like" Classification

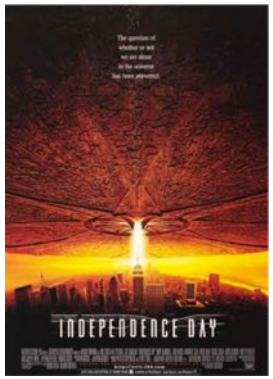
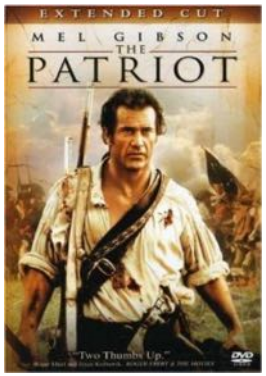
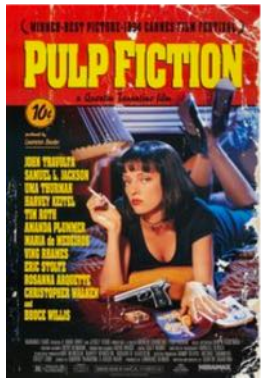
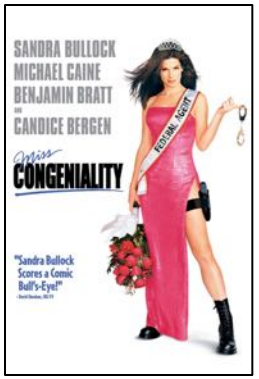
	Movie 1	Movie 2	…	Movie m	Output
					
User 1	1	0		1	1
User 2	1	1		0	0
		⋮			⋮
User n	0	0		1	1

Single Instance

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

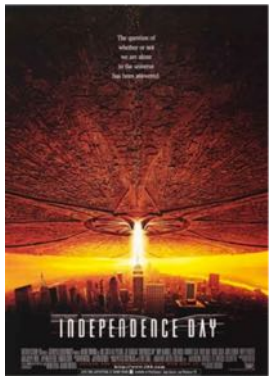
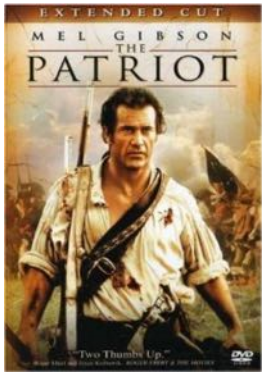
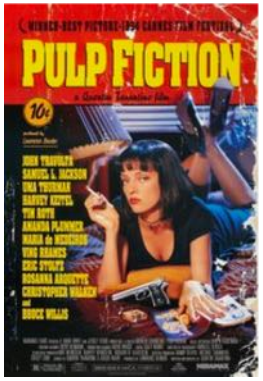
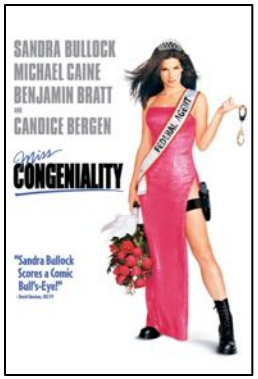
$$(\mathbf{x}^{(i)}, y^{(i)}) \text{ such that } 1 \leq i \leq n$$

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

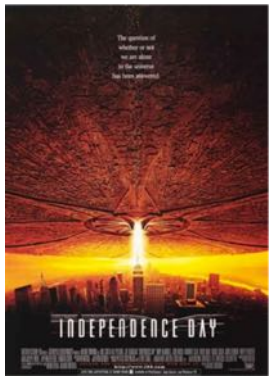
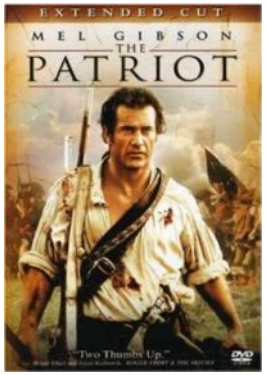
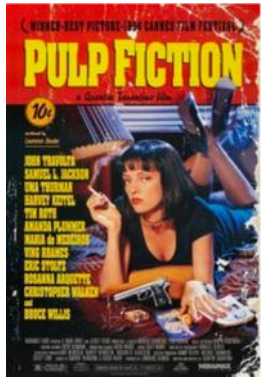
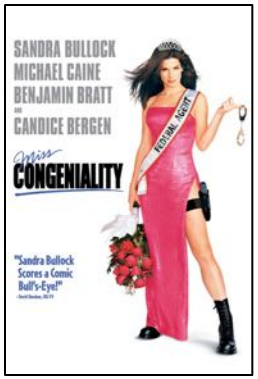
$(\mathbf{x}^{(i)}, y^{(i)})$ such that $1 \leq i \leq n$

Output Value

	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

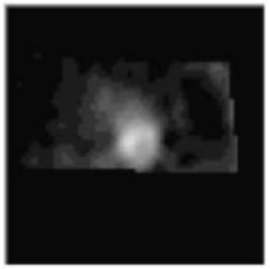
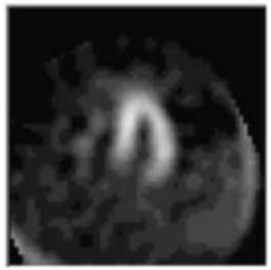
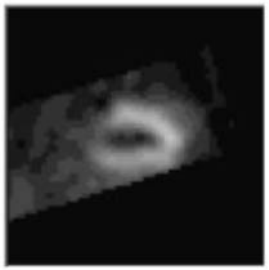
$$(\mathbf{x}^{(i)} \; y^{(i)}) \text{ such that } 1 \leq i \leq n$$

Single Feature Value

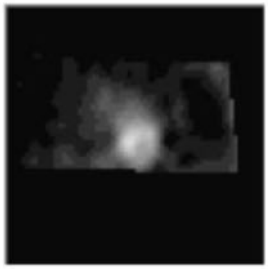
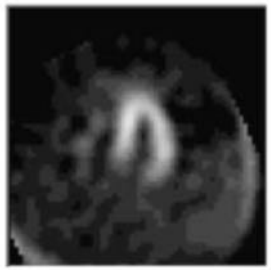
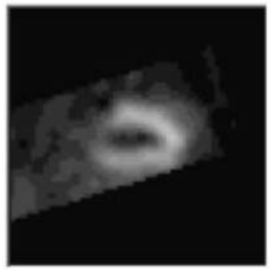
	Movie 1	Movie 2	…	Movie m	Output
					
User 1	1	0		1	1
User 2	1	1		0	0
⋮		⋮			⋮
User n	0	0		1	1

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

Healthy Heart Classifier

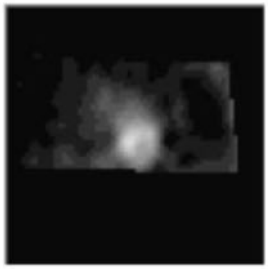
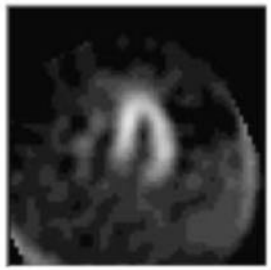
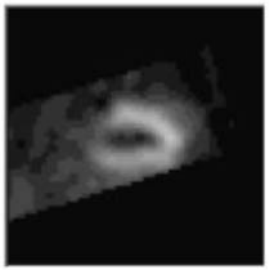
	ROI 1	ROI 2	ROI m	Output
				
Heart 1	0	1	1	0
Heart 2	1	1	1	0
		⋮		⋮
Heart n	0	0	0	1

Healthy Heart Classifier

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

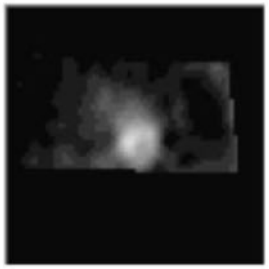
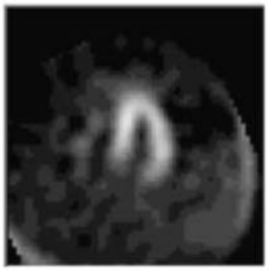
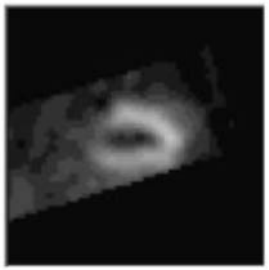
$x_2^{(1)}$

Healthy Heart Classifier

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

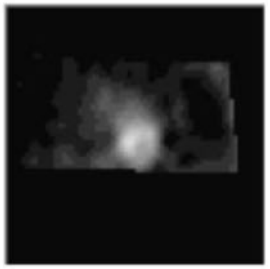
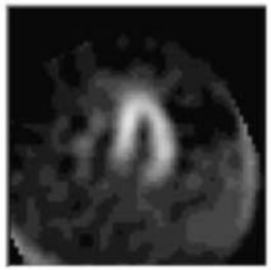
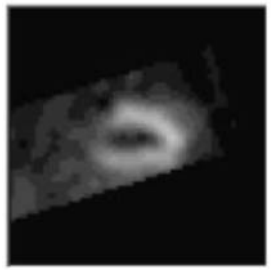
$(\mathbf{x}^{(2)}, y^{(2)})$

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

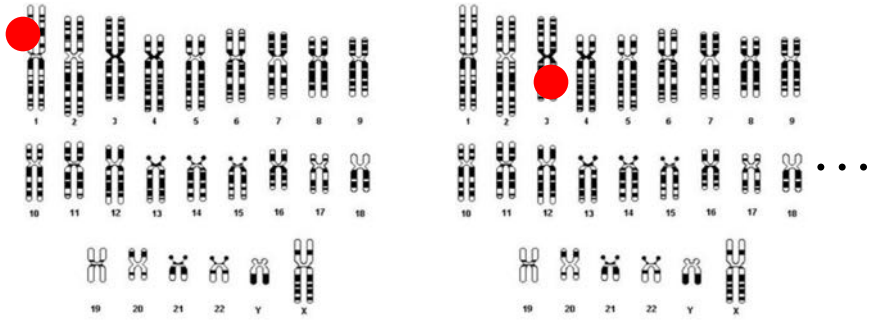
$\mathbf{x}^{(2)}$

Healthy Heart Classifier

	ROI 1	ROI 2	...	ROI m	Output
					
Heart 1	0	1		1	0
Heart 2	1	1		1	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
					GOLDEN STATE WARRIORS # Points
Game 1	84	105		1	120
Game 2	90	102		0	95
		⋮			⋮
Game n	74	120		0	115

Training Data

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

Linear Regression

A Grounding Example: Linear Regression

Problem: Predict real value Y based on observing variable X

N training pairs: $(\boldsymbol{x}^{(1)}, y^{(1)})$, $(\boldsymbol{x}^{(2)}, y^{(2)})$, …, $(\boldsymbol{x}^{(n)}, y^{(n)})$

Model: Linear weight for every feature

$$\begin{aligned}\hat{Y} &= \theta_1 X_1 + \theta_2 X_2 + \ldots \theta_{n-1} X_{n-1} + \theta_n 1 \\ &= \boldsymbol{\theta}^T \mathbf{X}\end{aligned}$$

Training: Chose the best thetas to describe your data

$$\hat{\theta}_{MLE} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} - \sum_{i=1}^{n} (y^{(i)} - \boldsymbol{\theta}^T \mathbf{x}^{(i)})^2$$

Use Gradient Ascent to optimize

MLE based on assumption

$$Y = \hat{Y} + Z$$

$$Z \sim N(0, \sigma^2)$$

Predicting Warriors

X_1 = Opposing team ELO

X_2 = Points in last game

X_3 = Curry playing?

X_4 = Playing at home?

Y = Warriors points

Predicting CO_2

X_1 = Temperature

X_2 = Elevation

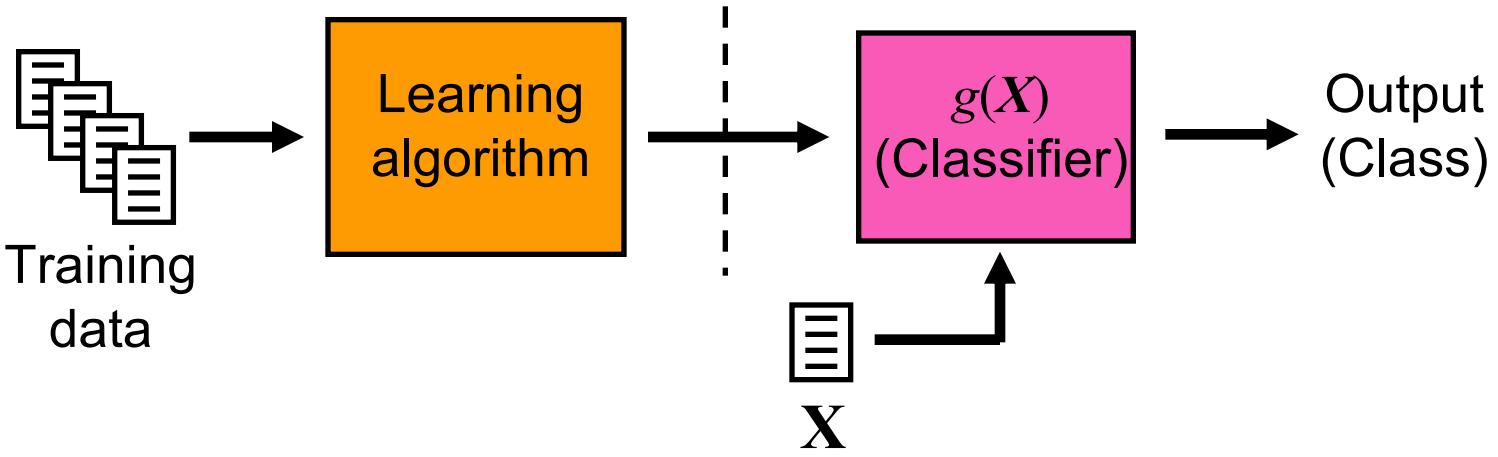
X_3 = CO_2 level yesterday

X_4 = GDP of region

X_5 = Acres of forest growth

Y = CO_2 levels

The Machine Learning Process



- Training data: set of *n* pre-classified data instances
  - *n* training pairs: $(\boldsymbol{x}^{(1)}, y^{(1)})$, $(\boldsymbol{x}^{(2)}, y^{(2)})$, …, $(\boldsymbol{x}^{(n)}, y^{(n)})$
    - Use superscripts to denote *i*-th training instance
- Learning algorithm: method for determining $g(\boldsymbol{X})$
  - Given a new input observation of $\boldsymbol{x} = x_1, x_2, \ldots, x_m$
  - Use $g(\boldsymbol{x})$ to compute a corresponding output (prediction)

Predicting Warriors

Y = Warriors points

$$\hat{Y} = \theta_1 X_1 + \theta_2 X_2 + \ldots \theta_{n-1} X_{n-1} + \theta_n 1$$
$$= \theta^T \mathbf{X}$$

X_1 = Opposing team ELO

X_2 = Points in last game

X_3 = Curry playing?

X_4 = Playing at home?

$\theta_1 = -2.3$

$\theta_2 = +1.2$

$\theta_3 = +10.2$

$\theta_4 = +3.3$

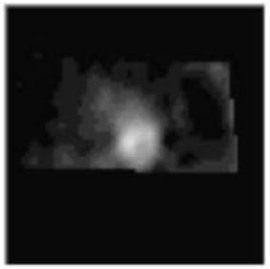
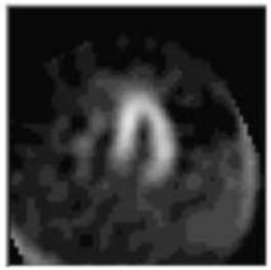
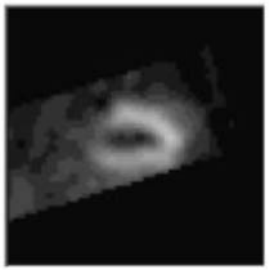
$\theta_5 = +95.4$

Classification

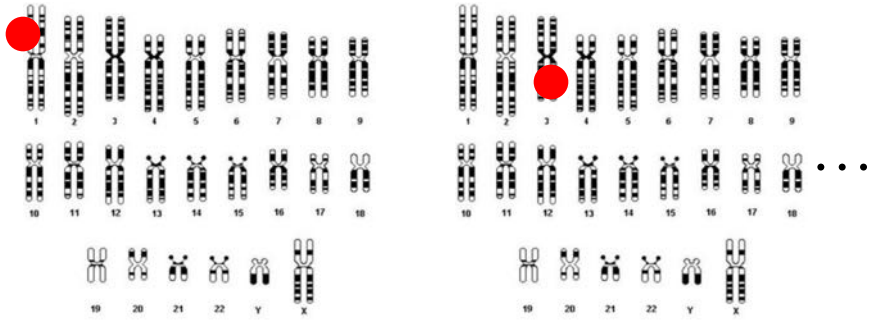
Classification is Building a Harry Potter Hat

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

Healthy Heart Classifier

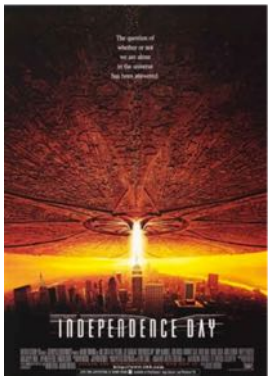
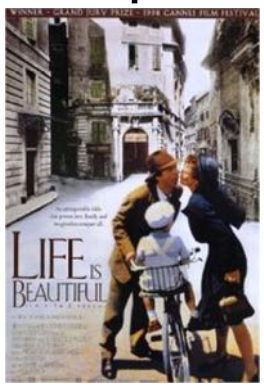
	ROI 1	ROI 2	ROI m	Output
				
Heart 1	0	1	1	0
Heart 2	1	1	1	0
		⋮		⋮
Heart n	0	0	0	1

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

And Learn

Target Movie "Like" Classification

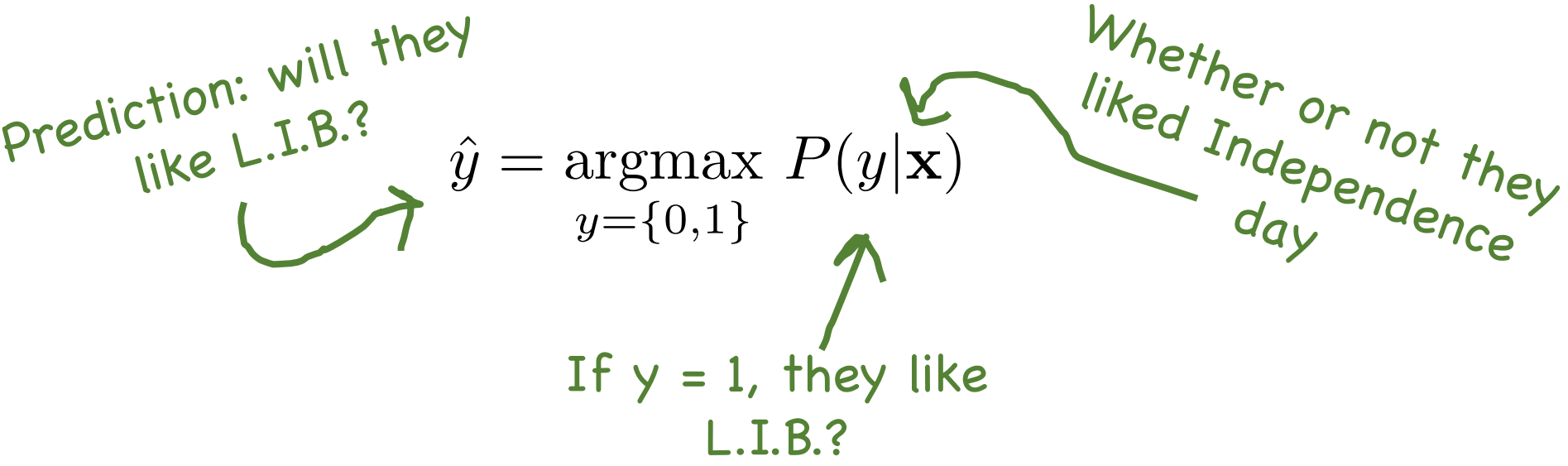
	Feature 1	Output
		
User 1	1	1
User 2	1	0
		⋮
User n	0	1

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

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

Fake Algorithm:
Brute Bayes Classifier

Brute Force Bayes



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

This is for one user

Brute Force Bayes

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

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

This is for one user

Brute Force Bayes

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

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

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

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

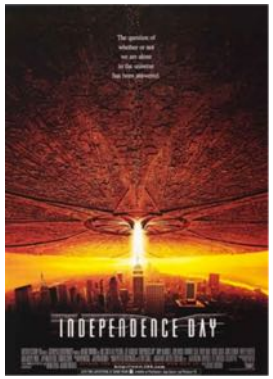
This is for one user

* Note how similar this is to Hamilton example ☺

What are the Parameters?

Brute Force Bayes

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



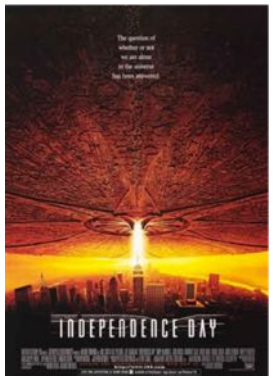
Joint probability table

Y \ X_1	0	1
0	θ_0	θ_1
1	θ_2	θ_3

y
θ_4
θ_5

Learn these during training

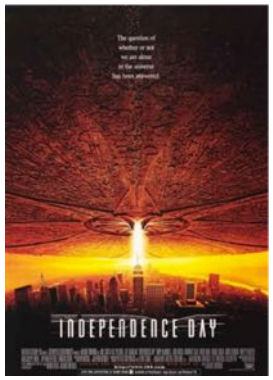
Training

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

Y \ X_1	0	1
0	θ_0	θ_1
1	θ_2	θ_3

Let (x_1, y) be a multinomial with four outcomes

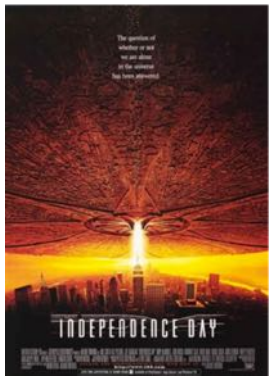
MLE Estimate

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

Y \ X_1	0	1
0	0.2	0.0
1	0.3	0.5

MLE: Just count

MAP Estimate

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

Y \ X_1	0	1
0	0.20	0.01
1	0.30	0.49

Add Laplace smoothing

Testing

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

Y \ X_1	0	1
0	0.20	0.01
1	0.30	0.49

y
0.21
0.79

Test user: Likes independence day

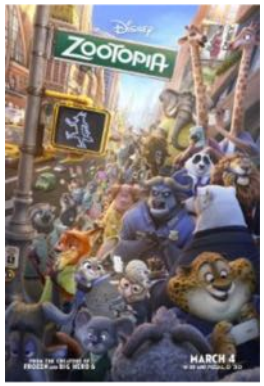
$$P(x_1 = 1|y = 0)P(y = 0)$$

vs

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

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

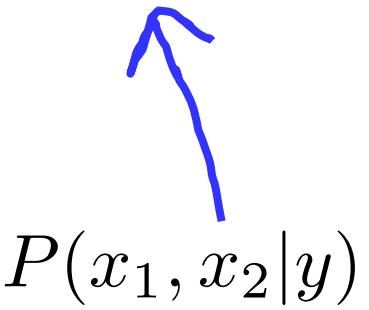
Brute Force Bayes m = 2

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

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

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

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



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

Brute Force Bayes

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

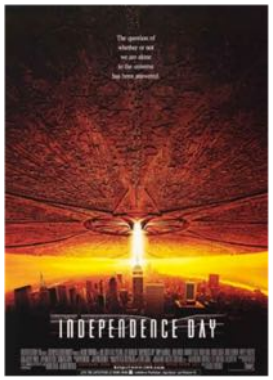
$X_2 = 0$

Y \ X_1	0	1
0	θ_0	θ_1
1	θ_2	θ_3

$X_2 = 1$

Y \ X_1	0	1
0	θ_4	θ_5
1	θ_6	θ_7

x_1

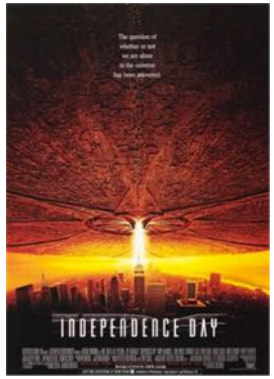
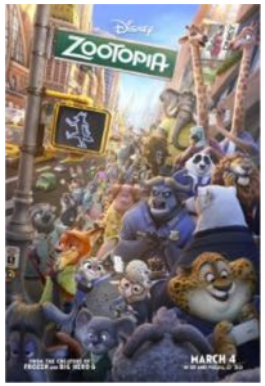
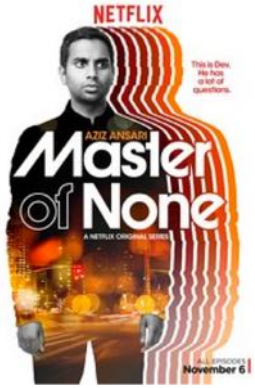
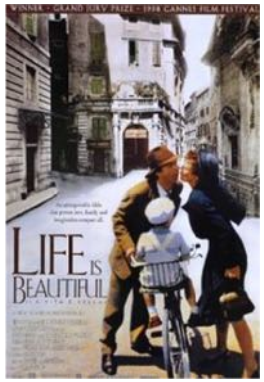


x_2

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

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

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

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

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

Brute Force Bayes

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

$X_3 = 0$

$X_2 = 0$

Y \ X_1	0	1
0	θ_0	θ_1
1	θ_2	θ_3

$X_2 = 1$

Y \ X_1	0	1
0	θ_8	θ_9
1	θ_{10}	θ_{11}

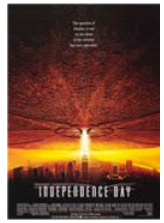
$X_3 = 1$

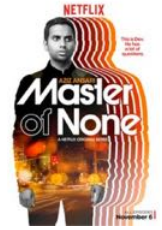
$X_2 = 0$

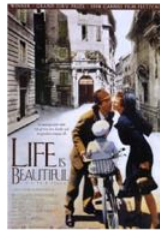
Y \ X_1	0	1
0	θ_4	θ_5
1	θ_6	θ_7

$X_2 = 1$

Y \ 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

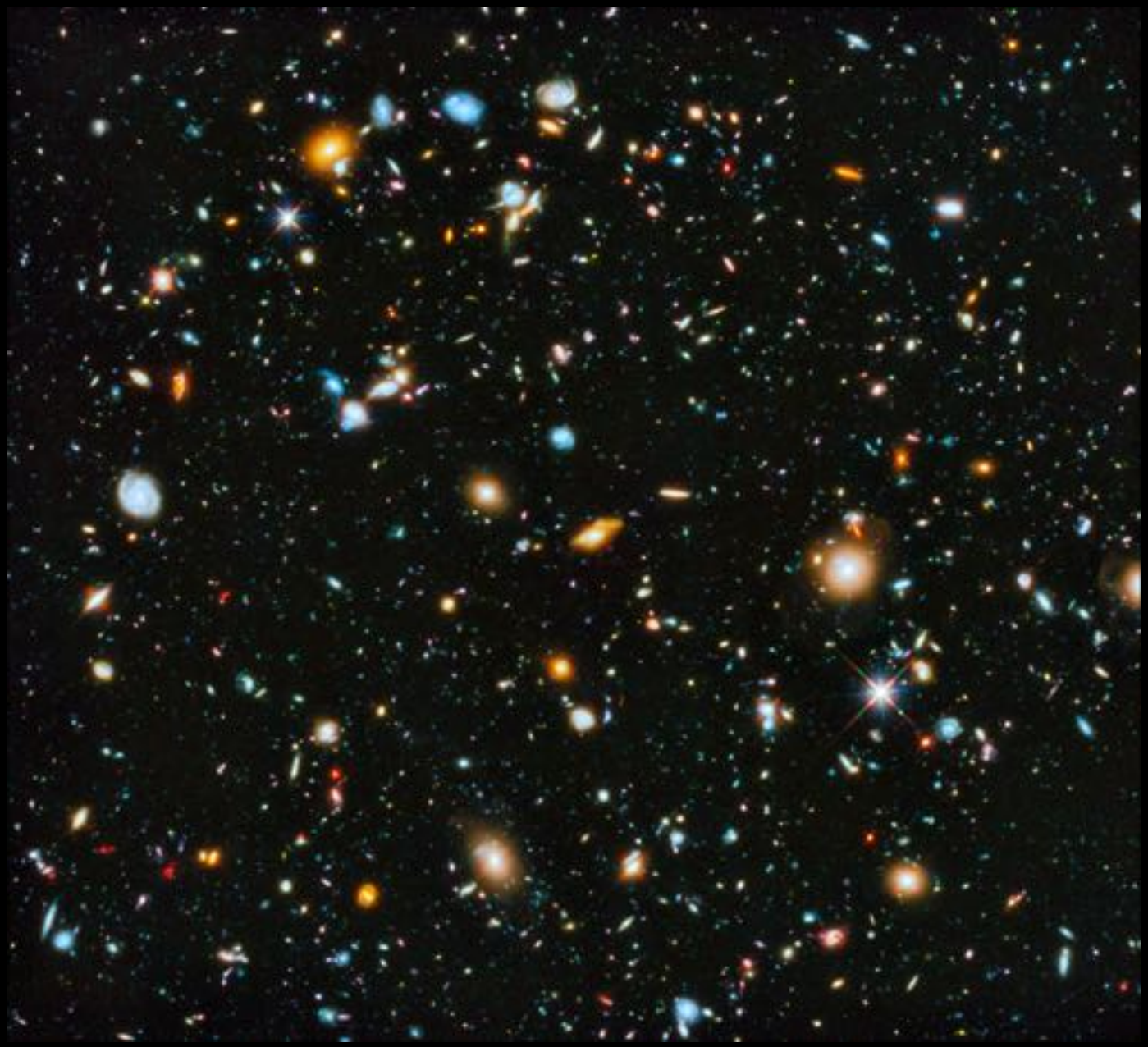
$$\hat{y} = \underset{y=\{0,1\}}{\operatorname{argmax}} \; P(y|\mathbf{x})$$

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

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

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

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!

What is the problem here?

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

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

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

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

Naïve Bayes Assumption

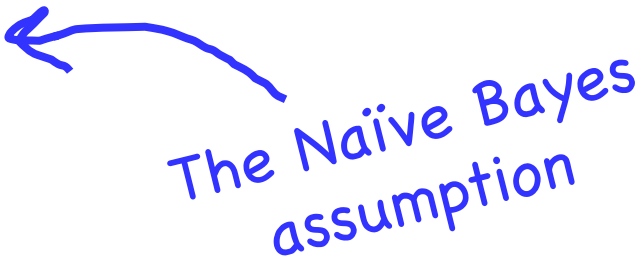
$$\hat{y} = \underset{y=\{0,1\}}{\operatorname{argmax}} P(y|\mathbf{x})$$

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

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

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

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



Naïve Bayes Assumption:

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

Naïve Bayes Classifier

Naïve Bayes

Our prediction for y

If 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

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

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 PMFs: $\hat{P}(x_i|y) \quad \hat{P}(y)$

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 PMFs: $\hat{P}(x_i|y) \quad \hat{P}(y)$

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.

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

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 PMFs: $\hat{P}(x_i|y) \quad \hat{P}(y)$

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 SciFi/Fantasy 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 $\hat{Y} = 1$

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

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: $\hat{P}(x_i|y) \quad \hat{P}(y)$

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: $\hat{P}(x_i|y) \quad \hat{P}(y)$

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

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

Y	#	MAP est.
0	13	
1	17	

Laplace! $$p_i = \frac{n_i + 1}{n + m} \qquad p_i = \frac{n_i + 1}{n + 2}$$

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: $\hat{P}(x_i|y) \quad \hat{P}(y)$

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

Computing Probabilities from Data

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

$$\hat{P}(Y=0)=\frac{\#\text{instances in class}=0}{\text{total}\,\#\,\text{instances}}$$

$$\hat{P}(X_i=0,Y=0)=\frac{\#\text{instances where } X_i=0 \text{ and class}=0}{\text{total}\,\#\,\text{instances}}$$

$$\hat{P}(X_i=0\mid Y=0)=\frac{\hat{P}(X_i=0,Y=0)}{\hat{P}(Y=0)} \qquad \hat{P}(X_i=0\mid Y=1)=\frac{\hat{P}(X_i=0,Y=1)}{\hat{P}(Y=1)}$$

$$\hat{P}(X_i=1\mid Y=0)=1-\hat{P}(X_i=0\mid Y=0)$$

Training Naïve Bayes, is estimating parameters for a multinomial.

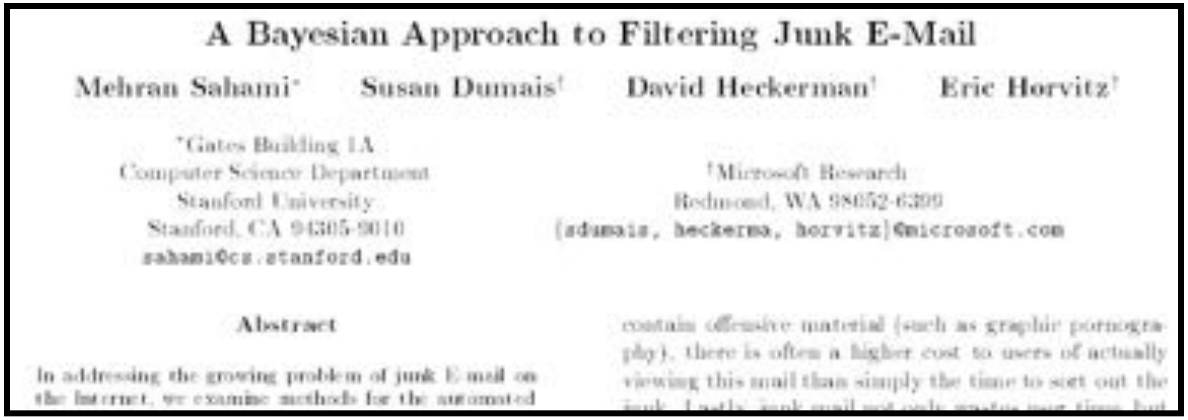
Thus training is just counting.

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

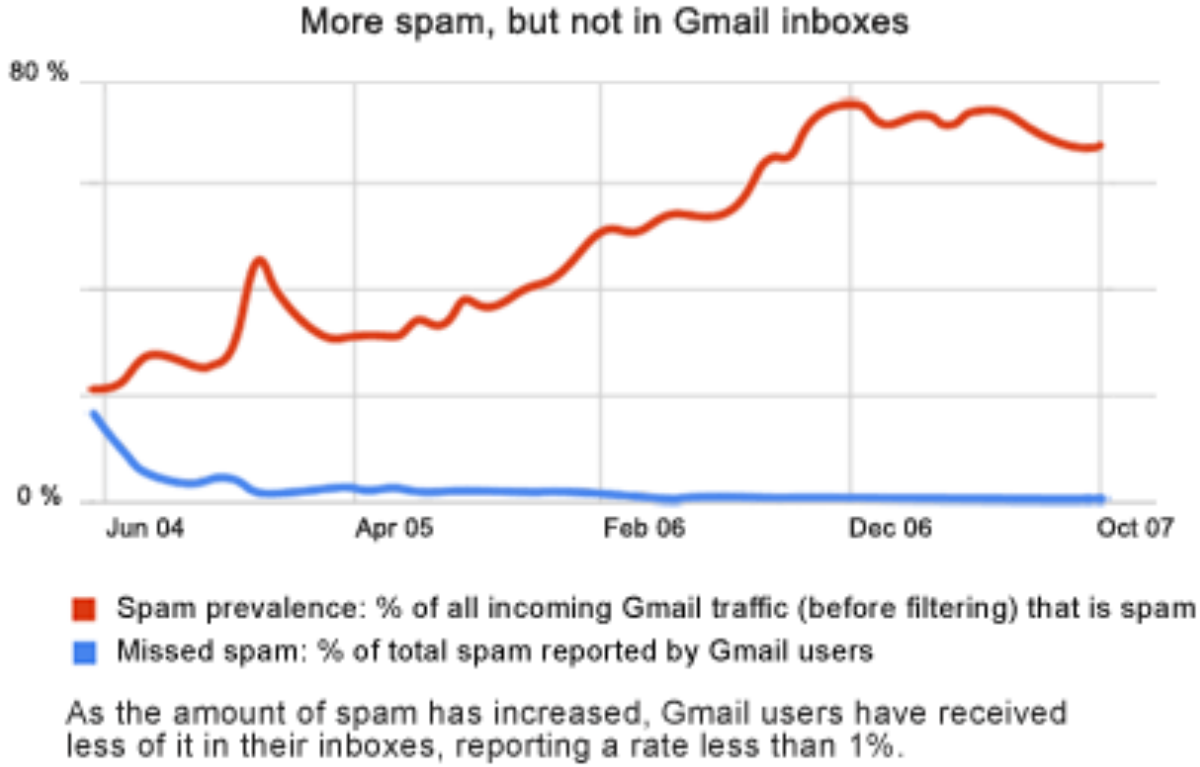
Abstract

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

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:

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

- Learning
  - Estimate probabilities P(Y) and each $P(X_i \mid Y)$ for all *i*
    - Many words are likely to not appear at all in given set of email
  - Laplace estimate: $\hat{p}(X_i = 1 \mid Y = spam)_{Laplace} = \frac{(\#\text{ spam emails with word } i) + 1}{\text{total } \#\text{ spam emails} + 2}$
- Classification
  - For a new email, generate $\mathbf{X} = \langle X_1, X_2, \ldots, X_m \rangle$
  - Classify as spam or not using: $\hat{Y} = \arg\max_y \hat{P}(\boldsymbol{X} \mid Y)\hat{P}(Y)$
  - Employ Naive Bayes assumption: $\hat{P}(\boldsymbol{X} \mid Y) = \prod_{i=1}^{m} \hat{P}(X_i \mid Y)$

Training Naïve Bayes, is estimating parameters for a multinomial.

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:
    - Precision = # *correctly* predicted class Y/ # predicted class Y
    - Recall = # *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%**

On biased datasets

Ethics and Datasets?

Sometimes machine learning feels universally unbiased.

We can even prove our estimators are “unbiased” ☺

Google/Nikon/HP had biased datasets

Ancestry dataset prediction

East Asian
or
Ad Mixed American (Native, European and African Americans)

It is much easier
to write a binary classifier
when learning ML
for the first time

Learn Three Things From This

1. What classification with DNA Single Nucleotide Polymorphisms looks like.

2. That genetic ancestry paints a more realistic picture of how we are mixed in many nuanced ways.

3. The importance of choosing the right data to learn from. Your results will be as biased as your dataset.

Know it so you can beat it!

Ethics in Machine Learning
is a whole new field