Machine Learning and Data Mining

Lecture 3.2: Generative Models and Naïve Bayes

CS 434

RECAP
From Last Lecture

We've covered a few techniques for fitting parameter:

Parameter Estimation:

Frameworks
- Maximum Likelihood Estimation
- Maximum A Posteriori

Optimization Tools
- Gradient Descent

Where are we?

We've covered a few predictive models so far in class:

Predictive Models:

Classifiers (given x, produce discrete y)
- k-Nearest Neighbors Classifier
- Logistic Regression (binary classification only)
- Perceptron (binary classification only)
- Naïve Bayes Classifier (Today)

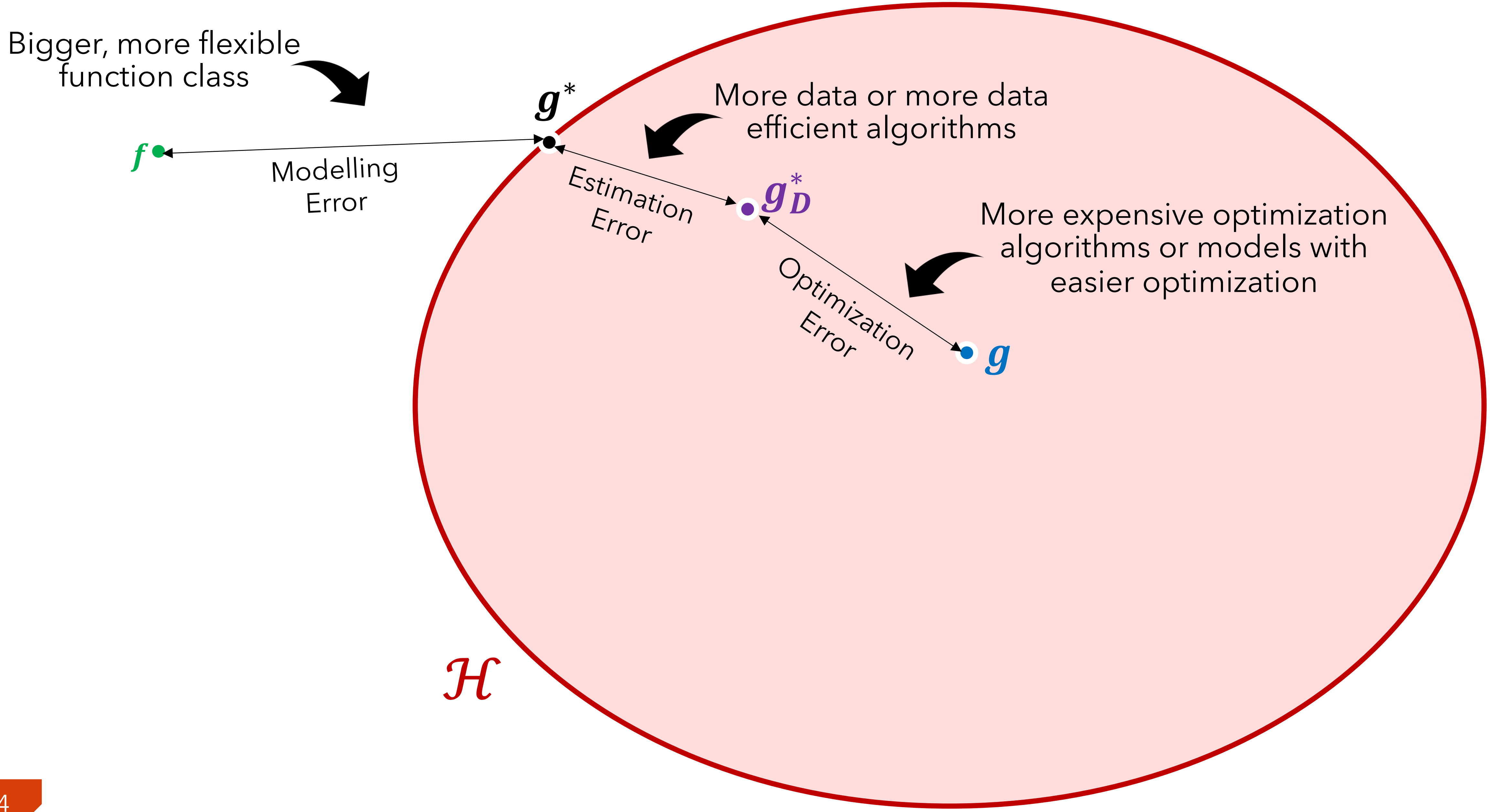
Regressors (given x, produce continuous y)
- Linear Regression
- k-Nearest Neighbors Regressor

Today's Learning Objectives

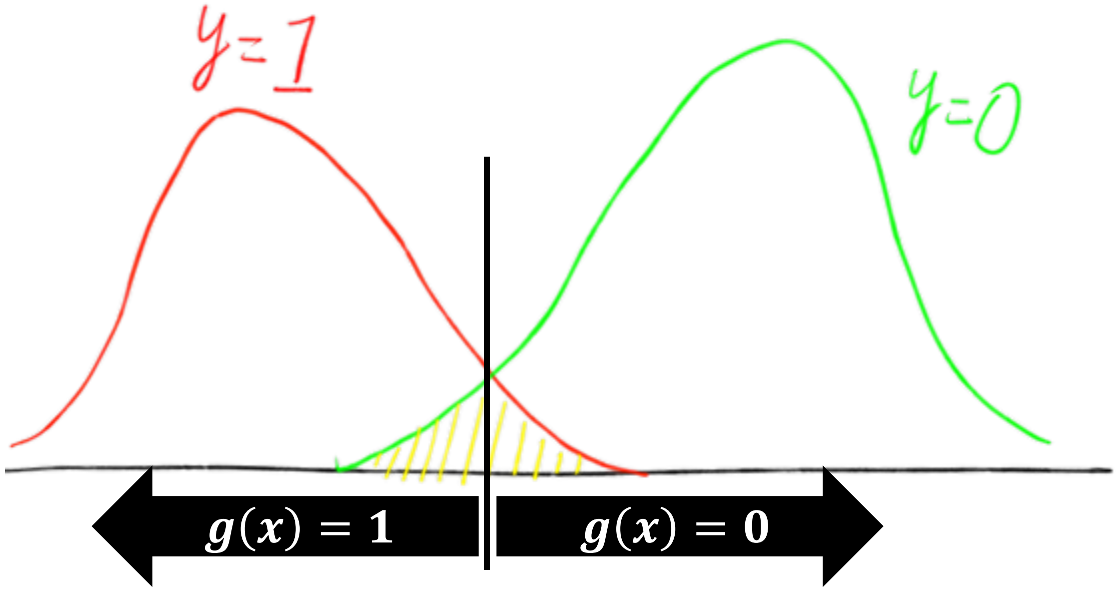
Be able to answer:

- What is a Bayes classifier?
- How do generative and discriminative models differ?
- What is a generative "story" for data?
  - How do we estimate a joint distribution for discrete variables?
- What is the Naïve-Bayes classifier?
  - What assumption makes it "naïve"?
    - What is conditional independence in probability?
    - How do you fit conditional distributions
  - What is Bernoulli Naïve Bayes?
  - What is Categorical Naïve Bayes?
  - What is Gaussian Naïve Bayes?

Recall: Sources of Error



Bayes Error
Irreducible error inherit in the function being approximated – nothing we can fix.



Recall: Sources of Error

Bayes Error
Irreducible error inherit in the function being approximated – nothing we can fix.

Likes Math	Reads ML News	Thinks MLE Is Cool?
0	0	0
1	0	1
1	1	1
1	1	0

Optimal Bayes Classifier:

Suppose we know the true distribution $P^*(Y|\boldsymbol{X})$ and for each $\boldsymbol{x}$ we encounter we predict:

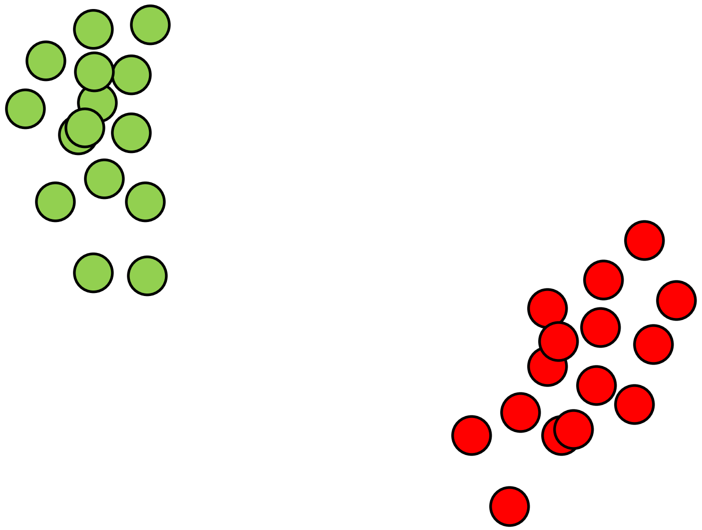
$$\hat{y} = argmax_y\, P^*(Y = y|\boldsymbol{X} = \boldsymbol{x})$$

- If we know the true $P^*(Y|\boldsymbol{X})$, this is optimal.
  https://en.wikipedia.org/wiki/Bayes_classifier

Problem: We don't know the true P(Y|X). How to learn it?

Problem: We don't know the true $P(y|x)$. How to learn it?

Consider the following binary classification problem (colors for labels):

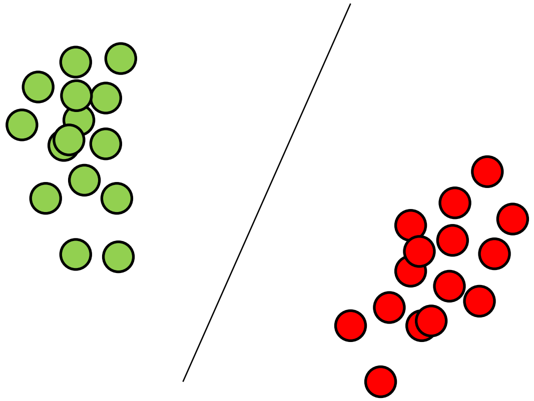


In logistic regression, we directly tried to model $P(y|\boldsymbol{x})$ -- assuming it was $\sigma(w^T\boldsymbol{x})$ and learning $\boldsymbol{w}$ with MLE.

An alternative approach would be to model each cluster separately – i.e. modelling $P(\boldsymbol{x}|y)$.

Discriminative Classifiers:

- Learn $P(y|\boldsymbol{x})$ directly
- Logistic regression is one example
- *Nomenclature note -- people will also refer to algorithms that model no distribution as discriminative (such as kNN).*

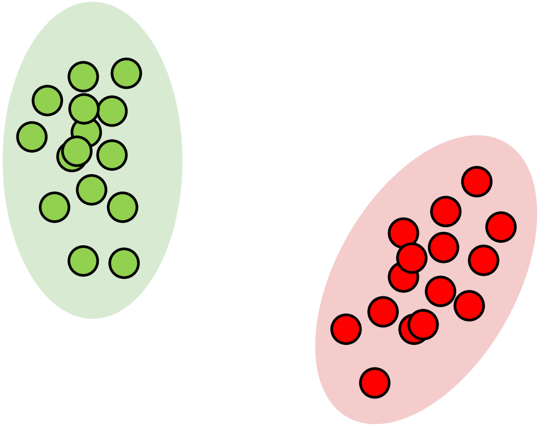


Generative Classifiers:

- Learn $P(\boldsymbol{x}|y)$ and $P(y)$
- Compute $P(y|\boldsymbol{x})$ using Bayes Rule

$$P(y|\boldsymbol{x}) = \frac{P(\boldsymbol{x}|y)P(y)}{P(\boldsymbol{x})} = \frac{P(\boldsymbol{x}|y)P(y)}{\sum_y P(\boldsymbol{x}|y)P(y)}$$

- Naïve Bayes is one example (today)



Both classify according to argmax $P(y|\boldsymbol{x})$. Just learn and represent it differently.

Generative Classifiers:

- Learn $P(\boldsymbol{x}|y)$ and $P(y)$
- Compute $P(y|\boldsymbol{x})$ using Bayes Rule

$$P(y|\boldsymbol{x}) = \frac{P(\boldsymbol{x}|y)P(y)}{P(\boldsymbol{x})} = \frac{P(\boldsymbol{x}|y)P(y)}{\sum_y P(\boldsymbol{x}|y)P(y)}$$

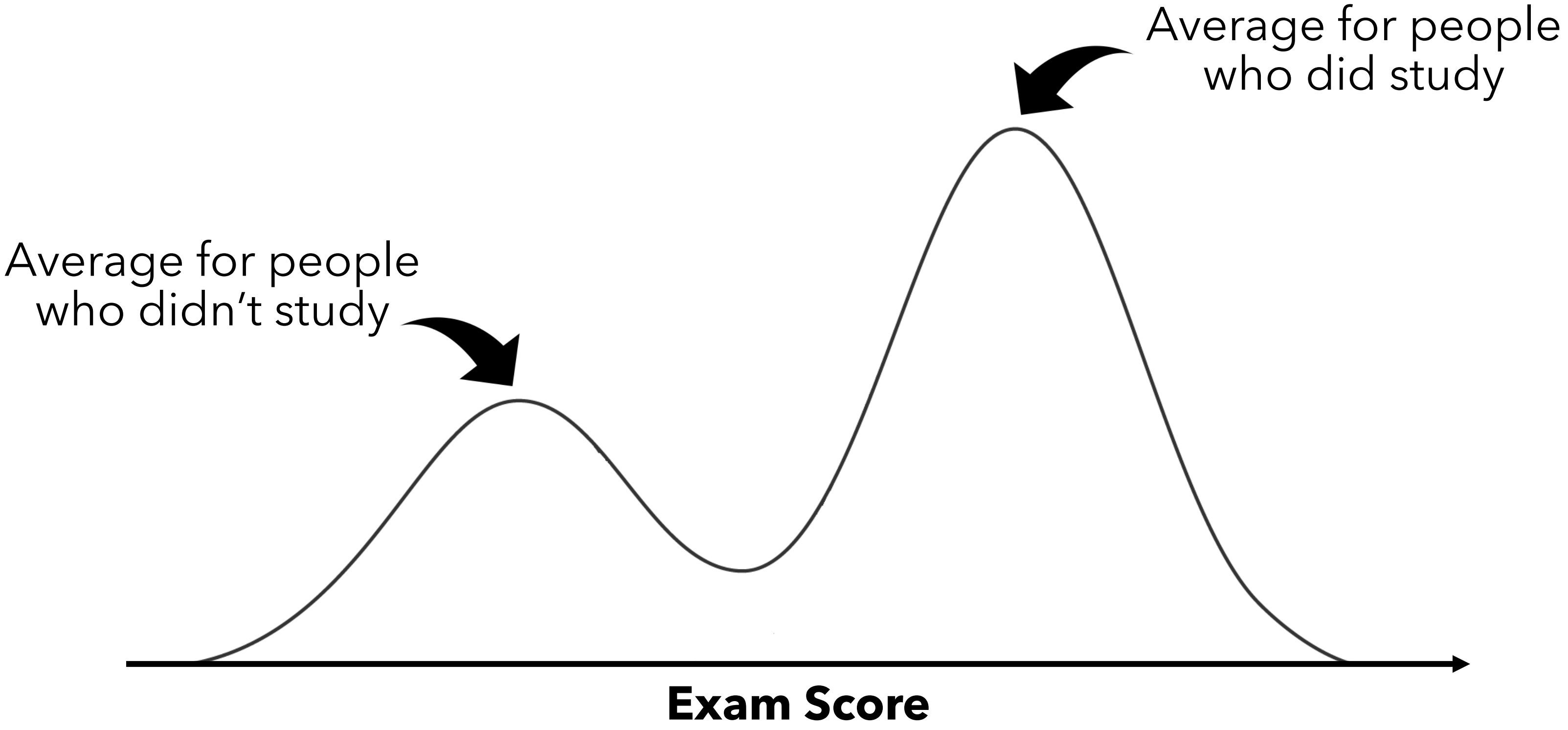
When defining a generative classifier, we are rather explicitly making a *fictional* story about how the dataset we observe was generated – the reading calls these *generative stories*.

Our modeling decisions combined with the data answer:

$P(y)$ How are classes distributed?

$P(\boldsymbol{x}|y)$ Given a class, how are features distributed?

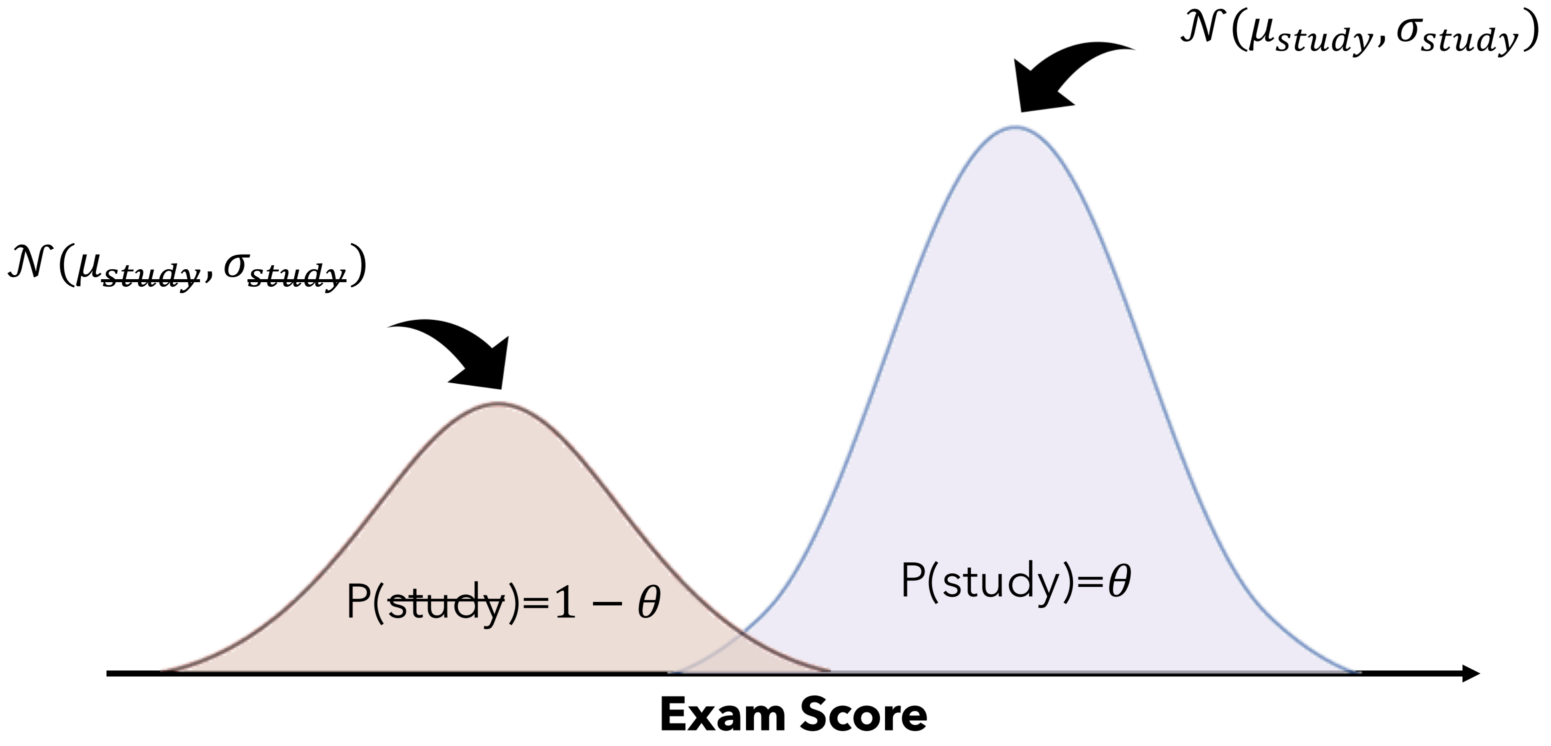
For example, suppose we have a class of students. Each student either studies or doesn't study before taking an exam. Consider the distribution of exam scores below. How might this distribution come to be?



Generative Classifiers – Example 1: Studying Students

A simple generative story for this exam scores:

1. Flip a coin (P(heads)=θ) to decide if someone studies (heads=study, tails=no)
2. If **study**, sample the exam score from a Gaussian distribution $\mathcal{N}(\mu_{study}, \sigma_{study})$
3. If **no study**, sample the exam score from a Gaussian distribution $\mathcal{N}(\mu_{\cancel{study}}, \sigma_{\cancel{study}})$



How to fit these distributions?

- P(study) = Bernoulli(θ_{study})
  - θ_{study} = # studying students / total students
- *P(score | study)* = $\mathcal{N}(score;\ \mu_{study}, \sigma_{study})$
  - $\mu_{study}, \sigma_{study}$ = average and variance of scores of studying students
- P(score | ~~study~~) = $\mathcal{N}(score;\ \mu_{\cancel{study}}, \sigma_{\cancel{study}})$
  - $\mu_{\cancel{study}}, \sigma_{\cancel{study}}$ = average and variance of scores of non-studying

Given an observed exam score, how would this story help us predict whether the student studied? Bayes rule to the rescue again.

$$P(study|score) \propto P(score|study)P(study)$$
$$= \mathcal{N}\left(score;\ \mu_{study}, \sigma_{study}\right)\theta$$

$$P(no\ study|score) \propto P(score|no\ study)P(no\ study)$$
$$= \mathcal{N}\left(score;\ \mu_{\cancel{study}}, \sigma_{\cancel{study}}\right)(1-\theta)$$

If $P(study|score) > P(no\ study|score)$, then predict study. Otherwise predict no study.

Study?	Score
Yes	87
Yes	89
Yes	87
Yes	88
Yes	91
Yes	97
Yes	91
Yes	93
No	80
No	77
No	64
No	75
No	91

$$P(study) = \frac{8}{14} \qquad P(no\ study) = \frac{6}{14}$$

$$P(score|study) = \mathcal{N}(score;\ \mu_{study} = 90.375,\ \ \sigma_{study} = 3.19)$$

$$P(score|\cancel{study}) = \mathcal{N}(score;\ \mu_{\cancel{study}} = 77.4,\ \ \sigma_{\cancel{study}} = 8.68)$$

See a new score of 82, did the student study?

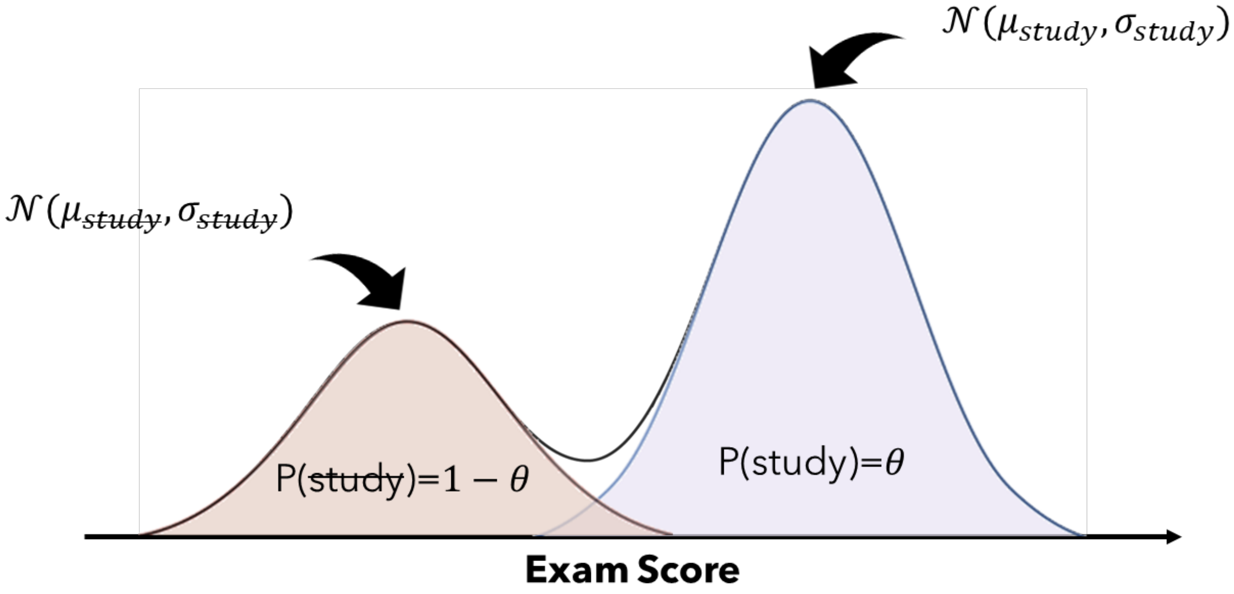
$$P(study|82) \propto P(82|study)P(study)$$

$$= \mathcal{N}(82;\ \mu_{study},\ \ \sigma_{study})\frac{8}{14} = 0.00227$$

$$P(no\ study|82) \propto P(82|no\ study)P(no\ study)$$

$$= \mathcal{N}(82;\ \mu_{\cancel{study}},\ \ \sigma_{\cancel{study}})\frac{6}{14} = 0.0171$$

This example had binary labels (y=study / no study) and only a single continuous feature (x=exam score). But we can do similar things with many more features or classes.



Consider the task of classifying email as either Normal or Spam based on the words in the email.

Output: $y \in \{Normal, Spam\}$ or equivalently $y \in \{0,1\}$

Input: List of M words occurring in an email. How to represent this?

- Popular approach is "bag-of-words"

Option 1: Binary Bag of Words

With a dictionary of size d, an email is represented as a binary vector $x = [x_1, x_2, \ldots, x_d]$ where $x_i \in \{0,1\}$

- $x_i = 1$ if the email contains the ith word in the dictionary at least once
- $x_i = 0$ otherwise

Option 2: Multinomial Bag of Words

With a dictionary of size d, an email is represented as a binary vector $x = [x_1, x_2, \ldots, x_d]$ where $x_i \in \mathbb{N}_0$

- x_i: non-negative integer occurrence count of the ith word in the dictionary

Option 1: Binary Bag of Words
With a dictionary of size d, an email is represented as a binary vector $x = [x_1, x_2, \dots, x_d]$ where $x_i \in \{0,1\}$
- $x_i = 1$ if the email contains the i[th] word in the dictionary at least once
- $x_i = 0$ otherwise

… source for a partner abroad who can accommodate HUGE RESOURCES.

Each document represented by a vector showing which words occurred:

$$\begin{array}{c} \begin{array}{cccc} & \text{partner} & \text{huge} & \text{resources} \end{array} \\ \boldsymbol{x} = [0, 1, 0, 0, 0, 1, \dots, 0, 0, 0, 1, 0] \\ \begin{array}{ccc} \text{assignment} & \text{research} & \text{meeting} \end{array} \end{array}$$

Vector has length equal to the number of words in our vocabulary (d).

A simple generative story for this spam emails:

1. Flip a weighted coin to determine Normal or Spam
2. If **Normal**, sample an **x** vector from P(**x**|normal)
3. If **Spam**, sample an **x** vector from P(**x**|spam)

Would need to learn:

Prior distribution $\boldsymbol{P(y)}$ as the prior probability of being Normal, Spam, or Advertisement

- P(normal) = # normal / total. P(spam) = # spam / total. P(ad) = # ad / total.

Conditional distributions $\boldsymbol{P(x|\, y = normal), P(x \mid y = spam)}$

- P(**x** | y=normal) = (# normal emails where the exact set of words in **x** occur) / (# normal emails)
- P(**x** | y=spam) = (# spam emails where the exact set of words in **x** occur) / (# spam emails)

Problem: How do we fit these distributions?

Joint Distributions over Discrete Variables

Recipe for writing a joint distribution of M discrete variables:

1. Make a table listing all value combinations of the variables. (M Boolean variables → 2^M rows)

2. For each value combination, define how probable it is

3. Due to axioms of probability, these probabilities must sum to 1.

A	*B*	C	Prob
0	0	0	**0.1**
0	0	1	**0.2**
0	1	0	**0.3**
0	1	1	**0.05**
1	0	0	**0.025**
1	0	1	**0.025**
1	1	0	**0.1**
1	1	1	**0.2**

$\mathrm{P}(\mathrm{A}=0, \mathrm{B}=0, \mathrm{C}=0)$

$\mathrm{P}(\mathrm{A}=1, \mathrm{B}=0, \mathrm{C}=1)$

Example: Joint distribution over Boolean random variables A,B,C

Learning Joint Distributions over Discrete Variables

Recipe for estimating a joint distribution of M discrete variables from date:

1. Make a table listing all value combinations of the variables. (M Boolean variables → 2^M rows)

2. For each value combination, make an MLE estimate of the probability of that entry:

$$P(row) = \frac{examples\ matching\ row\ exactly}{total\ number\ of\ examples}$$

(This is the MLE for a categorical distribution)

A	B	C	Prob
0	0	0	?
0	0	1	?
0	1	0	?
0	1	1	?
1	0	0	?
1	0	1	?
1	1	0	?
1	1	1	?

$$\frac{\text{count(1,0,1 examples)}}{total\ examples}$$

Example: Joint distribution over Boolean random variables A,B,C

Example of Estimating Joint Distributions over Discrete Variables

Our Data:

Likes Math	Reads ML News	At Least Junior
0	0	1
1	0	1
1	1	1
0	0	1
1	1	1
0	1	1
0	1	1
1	1	1
1	1	0
1	0	0

Out Joint Distribution:

Likes Math	Reads ML News	At Least Junior	Prob
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

Example of Estimating Joint Distributions over Discrete Variables

Our Data:

Likes Math	Reads ML News	At Least Junior
0	0	1
1	0	1
1	1	1
0	0	1
1	1	1
0	1	1
0	1	1
1	1	1
1	1	0
1	0	0

Out Joint Distribution:

Likes Math	Reads ML News	At Least Junior	Prob
0	0	0	**0**
0	0	1	**0.2**
0	1	0	**0**
0	1	1	**0.2**
1	0	0	**0.1**
1	0	1	**0.1**
1	1	0	**0.1**
1	1	1	**0.3**

Estimating Joint Distributions over Discrete Variables

$$\Bigg)$$

Generative Stories – Example 2: Email Classification

Prior distribution $\boldsymbol{P(y)}$ as the prior probability of being Normal, Spam, or Advertisement

- P(normal) = # normal / total. P(spam) = # spam / total. P(ad) = # ad / total.

Conditional distributions $\boldsymbol{P(x| \, y = normal), P(x \mid y = spam)}$

- P(**x** | y=normal) = (# normal emails where the exact set of words in **x** occur) / (# normal emails)
- P(**x** | y=spam) = (# spam emails where the exact set of words in **x** occur) / (# spam emails)

P(**x** | y=normal)

x_1	x_2	…	x_{d-1}	x_d	Prob
0	0	…	0	0	?
0	0	…	0	1	?
0	0	…	1	0	?
0	0	…	1	1	?
0	0	…	0	0	?
⋮	⋮	⋮	⋮	⋮	?
1	1	…	1	0	?
1	1	…	1	1	?

P(**x** | y=spam)

x_1	x_2	…	x_{d-1}	x_d	Prob
0	0	…	0	0	?
0	0	…	0	1	?
0	0	…	1	0	?
0	0	…	1	1	?
0	0	…	0	0	?
⋮	⋮	⋮	⋮	⋮	?
1	1	…	1	0	?
1	1	…	1	1	?

These are huge! $2 * 2^d$ entires.

In general, if I measure **d** things in **x** with each having **m** options, P(**x**|y) will have $\boldsymbol{c} * \boldsymbol{m}^{\boldsymbol{d}}$ parameters to learn for a **c** class problem. **Yikes.**

The real problem is that we must consider all combinations of features jointly.

The Naïve Bayes Assumption:

Each feature is **conditionally independent** given the class label.

$$P(\boldsymbol{x}|y) = \prod_{i=1}^{d} P(x_i|y)$$

Reminders about (conditional) independence

Independence: If two random variables X and Y are independent, then:

$$P(X,Y) = P(X)P(Y), \qquad P(X|Y) = P(X), \qquad P(Y|X) = P(Y)$$

Conditional Independence: If two random variables X and Y are conditionally independent given Z, then:

$$P(X,Y|Z) = P(X|Z)P(Y|Z), \qquad P(X|Y,Z) = P(X|Z), \qquad P(Y|X,Z) = P(Y|Z)$$

Reminders about (conditional) independence

Example: Define three random variables:

- **H:** A person's height
- **V:** How many words they know
- **A:** A person's age

def children:
Short people with small vocabularies.

Are height and vocabulary size independent?

No. Children tend to have lower vocabularies than adults (on average). So without knowing the person's age: $P(V,H) \neq P(V)P(H)$

However, if I tell you their age… its likely that these are independent. $\mathrm{P}(\mathrm{V},\mathrm{H}|A) = P(V|A)P(H|A)$

Events that are dependent in general, can be made independent given some other observation.

Reminders about (conditional) independence

The Naïve Bayes Assumption:

Each feature is **conditionally independent** given the class label.

$$P(\boldsymbol{x}|y) = \prod_{i=1}^{d} P(x_i|y)$$

How does this help?

$$P(y|\boldsymbol{x}) \propto P(\boldsymbol{x}|y)P(y)$$

Bayes Theorem

$$= P(y)\prod_{i=1}^{d} P(x_i|y)$$

Naïve Bayes Assumption

Can make predictions this way:

$$\underset{c=1,2,3,\ldots,k}{\operatorname{argmax}} \; P(y=c)\prod_{i=1}^{d} P(x_i|y=c)$$

A "naïve" generative story for this spam emails:

1. Flip a weighted coin to determine Normal or Spam.
2. For each word i, flip a weighted coin to determine if it is included in the email. Let the weight of the coin depend on whether the email is normal or spam.

Would need to learn:

Prior distribution $\boldsymbol{P}(\boldsymbol{y})$ as the prior probability of being Normal, Spam, or Advertisement

- P(normal) = # normal / total. P(spam) = # spam / total. P(ad) = # ad / total.

Conditional distributions $\boldsymbol{P}(\boldsymbol{x}|\, y = normal), \boldsymbol{P}(\boldsymbol{x} \mid y = spam)$

- P(x_i | y=normal) = (# normal emails with word i) / (# normal emails)
- P(x_i| y=spam) = (# spam emails with word i) / (# spam emails)

Problem: How do we fit these distributions?

Updating our Generative Story for Spam Emails to be "Naïve"

Would need to learn:

Prior distribution $P(y)$ as the prior probability of being Normal, Spam, or Advertisement

- P(normal) = # normal / total. P(spam) = # spam / total. P(ad) = # ad / total.

Conditional distributions $P(x|\, y = normal), P(x \mid y = spam)$

- P(x_i | y=normal) = (# normal emails with word i) / (# normal emails)
- P(x_i | y=spam) = (# spam emails with word i) / (# spam emails)

P(x_i | y=spam)

x_i	Prob
0	?
1	?

x d

P(x_i | y=normal)

x_i	Prob
0	?
1	?

x d

Now we only need to learn 2*2d values. (2*d if you take advantage of axioms)

Parameter Cost of Learning P(x|y):

In general, if I measure **d** things in **x** with each having **m** options, P(**x**|y) will have $\boldsymbol{c * m^d}$ free parameters to learn for a **c** class problem. **Yikes.**

With the Naïve Bayes assumption, we only need $\mathbf{c} * \boldsymbol{md}$ for the $P(x_i|y)$ distributions and $\boldsymbol{c-1}$ for the class prior $P(y)$ for this setting.

The Naïve Bayes Model Steps:

1. **Learn the conditional** $P(x_i|y=c)$ **for each feature** $\boldsymbol{x_i}$ **and class c** (training)
2. **Estimate** $P(y=c)$ **as a fraction of records with** $\boldsymbol{y=c}$ **for each class c** (training)
3. **For a new example** $\boldsymbol{x}=[x_1,\ldots,x_m]^T$**, predict:** (testing)

$$\underset{c=1,2,3,\ldots,k}{\operatorname{argmax}}\; P(y=c)\prod_{i=1}^{d} P(x_i|y=c)$$

An example: Suppose I measure three things about each of you:

- **UpperLowerClass** {Lower, Upper}
- **LikesMath** {Yes, No}
- **ReadsML** {Yes, No}

And want to predict whether you think the class is easy/moderate/hard:

- **CS434** {Easy, Moderate, Hard}

What do examples look like?

([Lower, Yes, No], Moderate), ([Upper, Yes, Yes], Easy),
([Lower, No, Yes], Difficult]), ([Upper, Yes, Yes], Moderate)

Naïve Bayes Model Idea

Now need to learn the following distributions:

$P(\text{UpperLower}|\text{CS434})$

P(lower | easy)	P(lower | moderate)	P(lower | hard)
P(upper | easy)	P(upper | moderate)	P(upper | hard)

4 free parameters

$P(\text{CS434})$

P(CS434=easy)
P(CS434=moderate)
P(CS434=hard)

2 free parameters

$P(LikesMath|\text{CS434})$

P(0 | easy)	P(0 | moderate)	P(0| hard)
P(1 | easy)	P(1 | moderate)	P(1 | hard)

3 free parameters

$P(ReadsML|\text{CS434})$

P(0 | easy)	P(0 | moderate)	P(0| hard)
P(1 | easy)	P(1 | moderate)	P(1 | hard)

3 free parameters

Let's try fitting one of these to the class:

$P(\text{UpperLower}|\text{CS434})$

P(lower | easy)	P(lower | moderate)	P(lower | hard)
P(upper | easy)	P(upper | moderate)	P(upper | hard)

$P(\text{CS434})$

P(CS434=easy)
P(CS434=moderate)
P(CS434=hard)

$P(LikesMath|\text{CS434})$

P(0 | easy)	P(0 | moderate)	P(0| hard)
P(1 | easy)	P(1 | moderate)	P(1 | hard)

$P(ReadsML|\text{CS434})$

P(0 | easy)	P(0 | moderate)	P(0| hard)
P(1 | easy)	P(1 | moderate)	P(1 | hard)

Series of polls to fill tables: A=Easy B=Moderate C=Hard

The zero-probability problem:

$$\underset{c=0,1}{\operatorname{argmax}} P(y=c)\prod_{i=1}^{d} P(w_i|y=c)$$

If any one of these terms is 0 for an instance, whole thing is 0.

Why might that happen?

What if a new email contains a word we never saw in the training emails?

$$P(w|spam) = 0 \; and \; P(w|not\ spam) = 0$$

Laplace Smoothing for Binary Variables

For binary variable x_i, add a small prior to $p(x_i|y=c)$:

- Bernoulli

$$p(x_i|y=c) = \frac{(\#\ of\ times\ x_i\ is\ true\ and\ y=c)+1}{(\#\ times\ y=c)+2}$$

Taking our P(LowerUpper | CS434=easy) example:

$$p(lower|CS434=easy) = \frac{(\#\ of fresh\&soph\ who\ think\ cs434\ is\ easy)+1}{(\#fresh\&soph)+2}$$

This is just adding a prior to the estimated conditional distributions.

Specifically, a Beta(1,1) prior.

Laplace Smoothing for Categorical Variables

For categorical variable x_i, add a small prior to $p(x_i|y=c)$:

- Categorical

$$p(x_i = a|y = c) = \frac{(\#\ of\ times\ x_i\ is\ a\ and\ y = c) + 1}{(\#\ times\ y = c) + \#classes}$$

This is just adding a prior to the estimated conditional distributions.

Specifically, a Dirichlet prior.

Taking the product of a bunch of probabilities is prone to underflow errors:

$$\operatorname*{argmax}_{c=0,1} P(y=c)\prod_{i=1}^{d} P(x_i|y=c)$$

0.0001*0.001*0.02*0.01*...

Let's do this in log space:

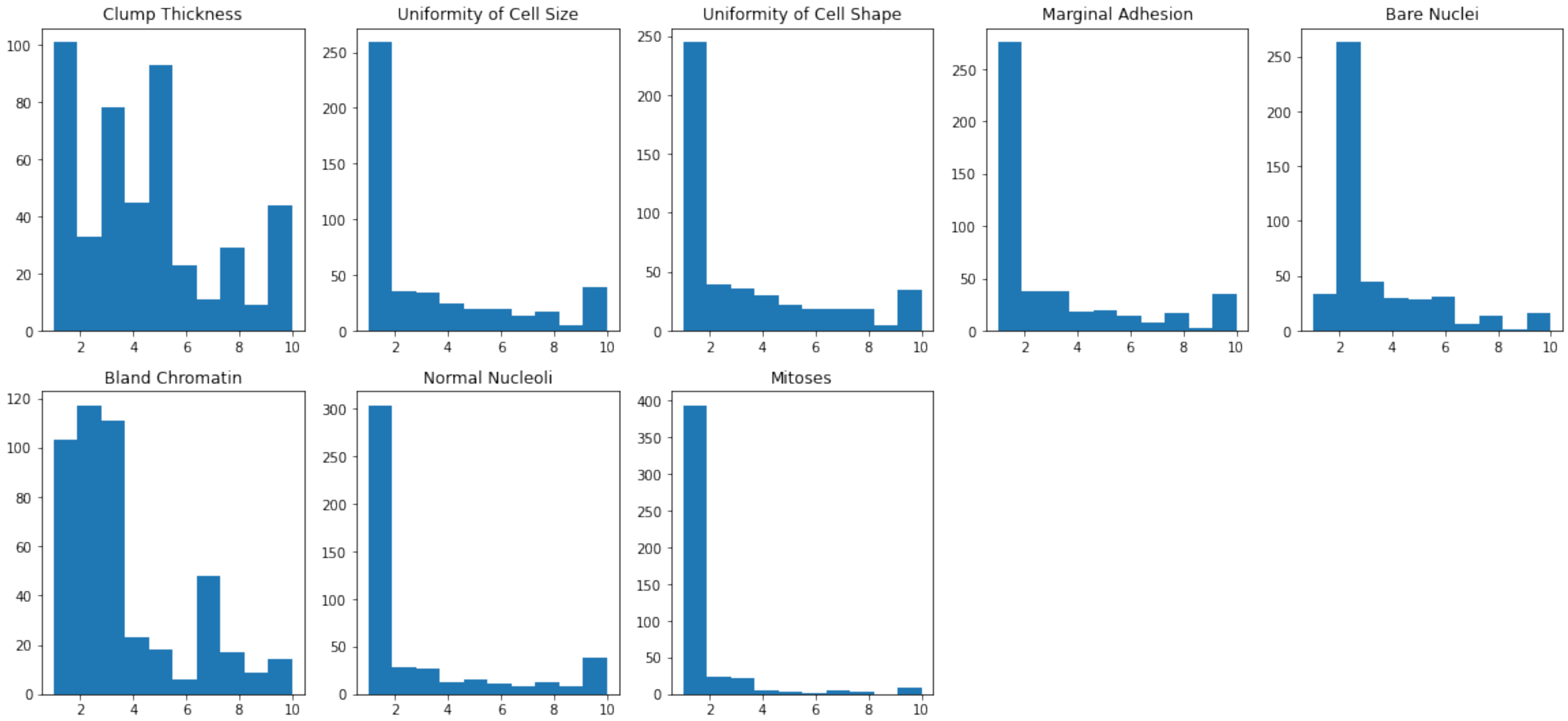
$$\operatorname*{argmax}_{c=0,1} logP(y=c) + \sum_{i=1}^{d} logP(x_i|y=c)$$

Summary of Naïve Bayes Classifier

- **Generative Model:** Estimate $P(y|\boldsymbol{x})$ by learning $P(\boldsymbol{x}|y)$ and $P(y)$. However $P(\boldsymbol{x}|y)$ can have way too many parameters to be fit effectively.
- **Naïve Bayes Assumption:** Assume features are conditionally independent given the class labels: $P(\boldsymbol{x}|y) = \prod P(x_i|y)$
- Training a Naïve Bayes classifier comes down to fitting distributions $P(x_i|y)$ and $P(y)$ either with MLE or MAP (MAP is more robust to data sparsity)
- Naïve Bayes is cheap and survives tens of thousands of attributes easily. Also does okay even when conditional independent doesn't hold.
- Any density estimator can be plugged in to estimate $p(x_i|y)$ for Naïve Bayes
  - Real valued attributes can be discretized or directly modeled using simple continuous distributions such as Gaussian (Normal) distribution

In-Class Coded Example 1:
Categorical Naïve Bayes on Tumor Classification

Categorical Naïve Bayes on Tumor Classification



Categorical Naïve Bayes on Tumor Classification

Let $y \in \{0,1\}$ map to benign / malignant respectively (assume Bernoulli)

- Two parameters P(y=Benign) and P(y=Malignant)

For each feature i, assume:

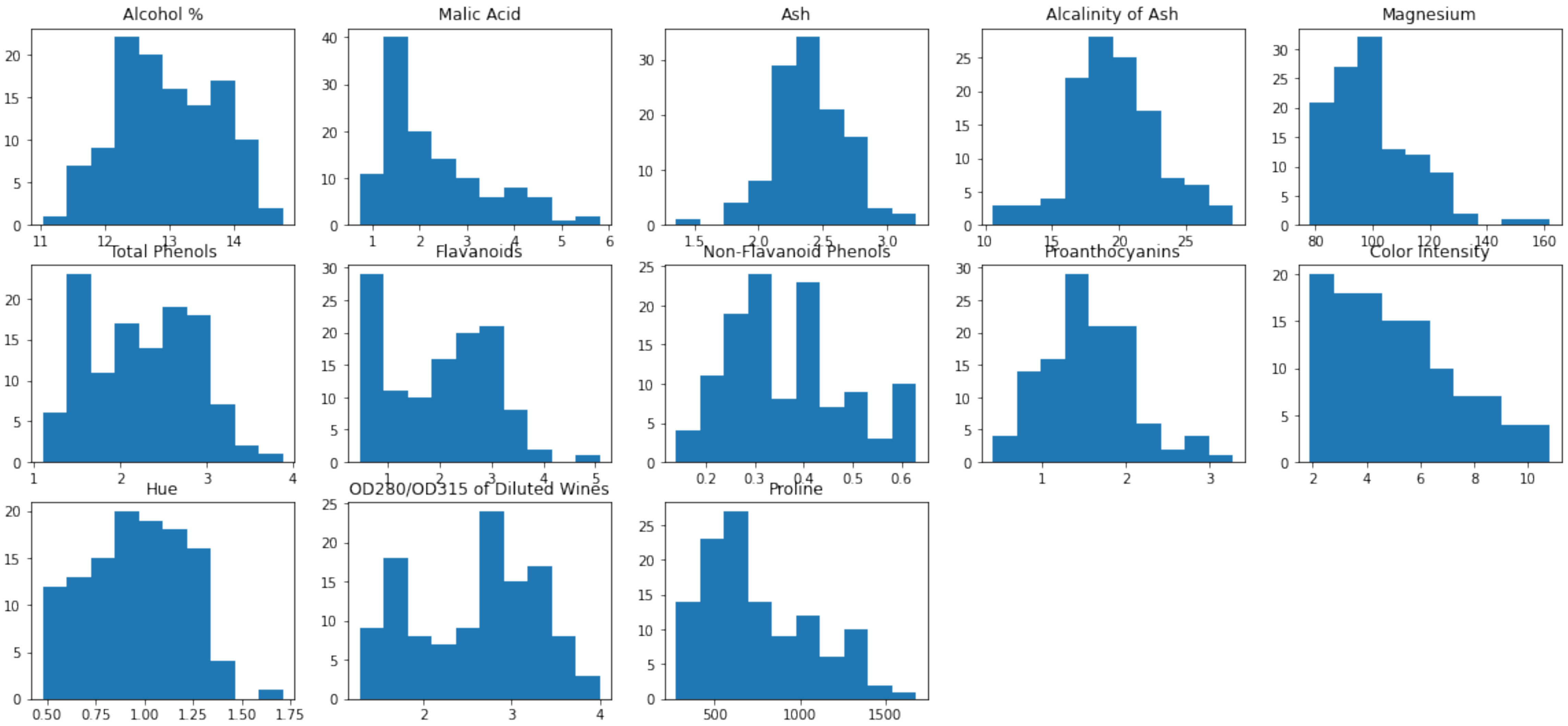
Let $x_i|y \in \{1,2,3,4,5,6,7,8,9,10\}$ be Categorical

- 10*2 parameter $P(x_i = 0 \mid y = 0), P(x_i = 0|y = 1), P(x_i = 1|y = 0), \ldots$

Need to learn the distributions:

- $P(y)$ and $P(x_i|y_i)\forall i$

In-Class Coded Example 2:
Gaussian Naïve Bayes on Winegrape Classification


Alcohol %
Malic Acid
Ash
Alcalinity of Ash
Magnesium
Total Phenols
Flavanoids
Non-Flavanoid Phenols
Proanthocyanins
Color Intensity
Hue
OD280/OD315 of Diluted Wines
Proline

Let $y \in \{0,1,2\}$ map to three different types of grapes (assume Categorical)

- Two parameters P(y=G1), P(y=G2), and P(y=G3)

For each feature i, assume:

Let $x_i|y \in \{-\infty, \infty\}$ be Gaussian

- 3*2 parameters -- $\mu_{G1}, \sigma_{G1}, \mu_{G2}, \sigma_{G2}, \mu_{G3}, \sigma_{G3}$

Need to learn the distributions:

- $P(y)$ and $P(x_i|y_i) \forall i$

You can check out the Colab notebook here:
https://colab.research.google.com/drive/1_7306psnPOFgrgUgAGQUm8XEL_3QZ6FO?usp=sharing

Next Time: We'll talk about maximum margin classifiers – specifically SVMs!