Generative Learning

INFO-4604, Applied Machine Learning
University of Colorado Boulder

November 29, 2018

Prof. Michael Paul

Generative vs Discriminative

The classification algorithms we have seen so far are called **discriminative** algorithms

- Learn to discriminate (i.e., distinguish/separate) between classes

Generative algorithms learn the characteristics of each class

- Then make a prediction of an instance based on which class it best matches
- Generative models can also be used to randomly generate instances of a class

Generative vs Discriminative

A high-level way to think about the difference: Generative models use *absolute* descriptions of classes and discriminative models use *relative* descriptions

Example: classifying *cats* vs *dogs*

Generative perspective:

- Cats weigh 10 pounds on average
- Dogs weigh 50 pounds on average

Discriminative perspective:

- Dogs weigh 40 pounds more than cats on average

Generative vs Discriminative

The difference between the two is often defined probabilistically:

Generative models:

- Algorithms learn $P(X \mid Y)$
- Then convert to $P(Y \mid X)$ to make prediction

Discriminative models:

- Algorithms learn $P(Y \mid X)$
- Probability can be directly used for prediction

Generative vs Discriminative

While discriminative models are not often probabilistic (but can be, like logistic regression), generative models usually are.

Example

Classify *cat* vs *dog* based on weight

- Cats have a mean weight of 10 pounds (stddev 2)
- Dogs have a mean weight of 50 pounds (stddev 20)

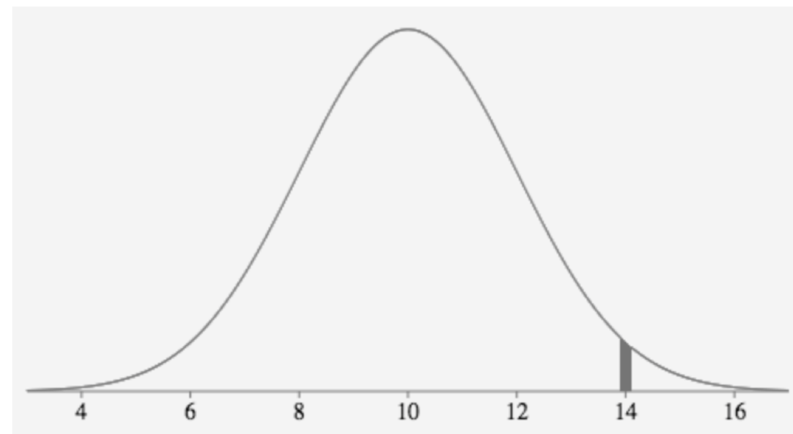
Could model the probability of the weight with a normal distribution

- Normal(10, 2) distribution for cats, Normal(50, 20) for dogs
- This is a distribution of probability *density*, but will refer to this as probability in this lecture

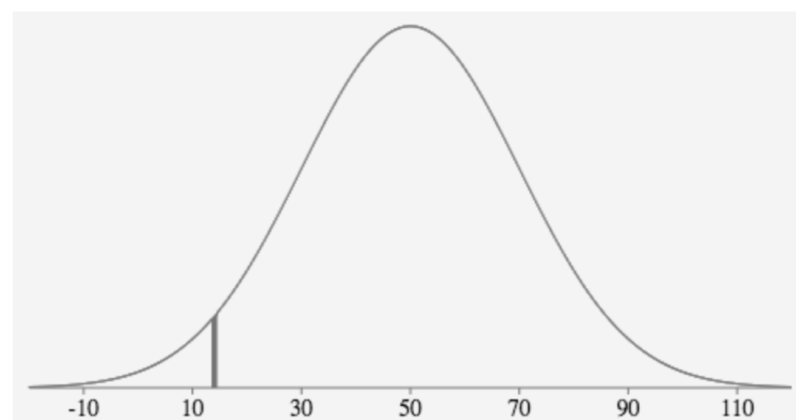
Example

Classify an animal that weighs 14 pounds

P(*weight*=14 | *animal*=cat)
= .027



P(*weight*=14 | *animal*=dog)
= .004



Example

Classify an animal that weighs 14 pounds

P(*weight*=14 | *animal*=cat) = .027

P(*weight*=14 | *animal*=dog) = .004

Choosing the Y that gives the highest P(X | Y) is reasonable… but not quite the right thing to do

- What if dogs were 99 times more common than cats in your dataset? That would affect the probability of being a cat versus a dog.

Bayes' Theorem

We have $P(X \mid Y)$, but we really want $P(Y \mid X)$

Bayes' theorem (or **Bayes' rule**):

$$P(B \mid A) = \frac{P(A \mid B)\, P(B)}{P(A)}$$

Naïve Bayes

Naïve Bayes is a classification algorithm that classifies an instance based on P(Y | X), where P(Y | X) is calculated using Bayes' rule:

$$P(Y \mid X) = \frac{P(X \mid Y)\, P(Y)}{P(X)}$$

Why *naïve*? We'll come back to that.

Naïve Bayes

Naïve Bayes is a classification algorithm that classifies an instance based on P(Y | X), where P(Y | X) is calculated using Bayes' rule:

$$P(Y \mid X) = \frac{P(X \mid Y)\, \mathbf{P(Y)}}{P(X)}$$

- Called the **prior** probability of Y
- Usually just calculated as the percentage of training instances labeled as Y

Naïve Bayes

Naïve Bayes is a classification algorithm that classifies an instance based on P(Y I X), where P(Y I X) is calculated using Bayes' rule:

$$\color{red}{\mathbf{P(Y \mid X)}} = \frac{P(X \mid Y)\, P(Y)}{P(X)}$$

- Called the **posterior** probability of Y
- The **conditional** probability of Y given an instance X

Naïve Bayes

Naïve Bayes is a classification algorithm that classifies an instance based on P(Y | X), where P(Y | X) is calculated using Bayes' rule:

$$P(Y \mid X) = \frac{\color{red}{\mathbf{P(X \mid Y)}}\, P(Y)}{P(X)}$$

- This conditional probability is what needs to be *learned*

Naïve Bayes

Naïve Bayes is a classification algorithm that classifies an instance based on P(Y I X), where P(Y I X) is calculated using Bayes' rule:

$$P(Y \mid X) = \frac{P(X \mid Y)\, P(Y)}{\mathbf{P(X)}}$$

- What about P(X)?
- Probability of observing the data
- *Doesn't actually matter!*
    - P(X) is the same regardless of Y
    - Doesn't change which Y has highest probability

Example

Classify an animal that weighs 14 pounds

Also: dogs are 99 times more common than cats in the data

P(*weight*=14 | *animal*=cat) = .027

P(*animal*=cat | *weight*=14) = ?

Example

Classify an animal that weighs 14 pounds

Also: dogs are 99 times more common than cats in the data

$P(\textit{weight}{=}14 \mid \textit{animal}{=}\text{cat}) = .027$

$P(\textit{animal}{=}\text{cat} \mid \textit{weight}{=}14)$

$\approx P(\textit{weight}{=}14 \mid \textit{animal}{=}\text{cat})\ P(\textit{animal}{=}\text{cat})$

$= 0.027 * 0.01 = 0.00027$

Example

Classify an animal that weighs 14 pounds

Also: dogs are 99 times more common than cats in the data

P(*weight*=14 | *animal*=dog) = .004

P(*animal*=dog | *weight*=14)

≈ P(*weight*=14 | *animal*=dog) P(*animal*=dog)

= 0.004 * 0.99 = 0.00396

Example

Classify an animal that weighs 14 pounds

Also: dogs are 99 times more common than cats in the data

$P(\textit{animal}=\text{dog} \mid \textit{weight}=14) >$
$P(\textit{animal}=\text{cat} \mid \textit{weight}=14)$

You should classify the animal as a dog.

Naïve Bayes

Learning:

- Estimate P(X | Y) from the data
- Estimate P(Y) from the data

Prediction:

- Choose Y that maximizes:

$$P(X \mid Y)\, P(Y)$$

Naïve Bayes

Learning:

- Estimate P(X | Y) from the data
  - ???
- Estimate P(Y) from the data
  - Usually just calculated as the percentage of training instances labeled as Y

Naïve Bayes

Learning:

- Estimate P(X I Y) from the data
  - Requires some decisions (and some math)
- Estimate P(Y) from the data
  - Usually just calculated as the percentage of training instances labeled as Y

Defining P(X | Y)

With continuous features, a normal distribution is a common way to define P(X | Y)

- But keep in mind that this is only an approximation: the true probability might be something different
- Other probability distributions exist that you can use instead (not discussed here)

With discrete features, the observed distribution (i.e., the proportion of instances with each value) is usually used as-is

Defining P(X | Y)

Another complication…
Instances are usually vectors of many features

How do you define the probability of an entire feature vector?

Joint Probability

The probability of multiple variables is called the **joint** probability

Example: if you roll two dice, what’s the probability that they both land 5?

Joint Probability

36 possible outcomes:

1,1	2,1	3,1	4,1	5,1	6,1
1,2	2,2	3,2	4,2	5,2	6,2
1,3	2,3	3,3	4,3	5,3	6,3
1,4	2,4	3,4	4,4	5,4	6,4
1,5	2,5	3,5	4,5	5,5	6,5
1,6	2,6	3,6	4,6	5,6	6,6

Joint Probability

36 possible outcomes:

1,1	2,1	3,1	4,1	5,1	6,1
1,2	2,2	3,2	4,2	5,2	6,2
1,3	2,3	3,3	4,3	5,3	6,3
1,4	2,4	3,4	4,4	5,4	6,4
1,5	2,5	3,5	4,5	**5,5**	6,5
1,6	2,6	3,6	4,6	5,6	6,6

Probability of two 5s:
1/36

Joint Probability

36 possible outcomes:

1,1	2,1	3,1	4,1	5,1	6,1
1,2	2,2	3,2	4,2	5,2	6,2
1,3	2,3	3,3	4,3	5,3	6,3
1,4	2,4	3,4	4,4	5,4	6,4
1,5	2,5	3,5	4,5	5,5	6,5
1,6	2,6	3,6	4,6	5,6	6,6

Joint Probability

36 possible outcomes:

1,1	2,1	3,1	4,1	**5,1**	6,1
1,2	2,2	3,2	4,2	**5,2**	6,2
1,3	2,3	3,3	4,3	**5,3**	6,3
1,4	2,4	3,4	4,4	**5,4**	6,4
1,5	2,5	3,5	4,5	5,5	6,5
1,6	2,6	3,6	4,6	**5,6**	6,6

Probability the first is a 5 and the second is anything but 5:

5/36

Joint Probability

A quicker way to calculate this:

The probability of two variables is the *product* of the probability of each individual variable

- Only true if the two variables are *independent*! (defined on next slide)

Probability of one die landing 5: 1/6

Joint probability of two dice landing 5 and 5:
1/6 * 1/6 = 1/36

Joint Probability

A quicker way to calculate this:

The probability of two variables is the *product* of the probability of each individual variable

- Only true if the two variables are *independent*! (defined on next slide)

Probability of one die landing anything but 5: 5/6

Joint probability of two dice landing 5 and not 5:
1/6 * 5/6 = 5/36

Independence

Multiple variables are **independent** if knowing the outcome of one does not change the probability of another

- If I tell you that the first die landed 5, it shouldn't change your belief about the outcome of the second (every side will still have 1/6 probability)
- Dice rolls are independent

Conditional Independence

Naïve Bayes treats the feature probabilities as independent (conditioned on Y)

$P(<X_1, X_2, \ldots, X_M> \mid Y)$
$= P(X_1 \mid Y) * P(X_2 \mid Y) \ldots * P(X_M \mid Y)$

Features are usually not actually independent!

- Treating them as if they are is considered *naïve*
- But it's often a good enough approximation
- This makes the calculation much easier

Conditional Independence

Important distinction:
the features have **conditional independence:**
the independence assumption only applies to the conditional probabilities $P(X \mid Y)$

Conditional independence:

- $P(X_1, X_2 \mid Y) = P(X_1 \mid Y) * P(X_2 \mid Y)$
- Not necessarily true that
  $P(X_1, X_2) = P(X_1) * P(X_2)$

Conditional Independence

Example: Suppose you are classifying the category of a news article using word features

If you observe the word “baseball”, this would increase the likelihood that the word “homerun” will appear in the same article

- These two features are clearly not independent

But if you already know the article is about baseball (Y=baseball), then observing the word “baseball” doesn’t change the probability of observing other baseball-related words

Defining P(X | Y)

Naïve Bayes is most often used with discrete features

With discrete features, the probability of a particular feature value is usually calculated as:

$$\frac{\text{\# of times the feature has that value}}{\text{total \# of occurrences of the feature}}$$

Document Classification

Naïve Bayes is often used for document classification

- Given the document class, what is the probability of observing the words in the document?

Document Classification

Example:

3 documents:

"the water is cold"

"the pig went home"

"the home is cold"

P("the")	= 3/12
P("is")	= 2/12
P("home")	= 2/12
P("cold")	= 2/12
P("water")	= 1/12
P("went")	= 1/12
P("pig")	= 1/12

P("the water is cold")
= P("the") P("water") P("is") P("cold")

Document Classification

Example:

3 documents:

"the water is cold"

"the pig went home"

"the home is cold"

P("the")	= 3/12
P("is")	= 2/12
P("home")	= 2/12
P("cold")	= 2/12
P("water")	= 1/12
P("went")	= 1/12
P("pig")	= 1/12

P("the water is very cold")
= P("the") P("water") P("is") P("very") P("cold")

Document Classification

Example:

3 documents:

"the water is cold"

"the pig went home"

"the home is cold"

P("the")	= 3/12
P("is")	= 2/12
P("home")	= 2/12
P("cold")	= 2/12
P("water")	= 1/12
P("went")	= 1/12
P("pig")	= 1/12
P("very")	= 0/12

P("the water is very cold")
= P("the") P("water") P("is") P("very") P("cold")
= 0

Document Classification

Example:

3 documents:

"the water is cold"

"the pig went home"

"the home is cold"

P("the")	= 3/12
P("is")	= 2/12
P("home")	= 2/12
P("cold")	= 2/12
P("water")	= 1/12
P("went")	= 1/12
P("pig")	= 1/12
P("very")	= 0/12

One trick: pretend every value occurred one more time than it did

Document Classification

Example:

3 documents:

"the water is cold"

"the pig went home"

"the home is cold"

P("the")	= 4/12
P("is")	= 3/12
P("home")	= 3/12
P("cold")	= 3/12
P("water")	= 2/12
P("went")	= 2/12
P("pig")	= 2/12
P("very")	= 1/12

One trick: pretend every value occurred one more time than it did

Document Classification

Example:

3 documents:

"the water is cold"

"the pig went home"

"the home is cold"

P("the")	= 4/20
P("is")	= 3/20
P("home")	= 3/20
P("cold")	= 3/20
P("water")	= 2/20
P("went")	= 2/20
P("pig")	= 2/20
P("very")	= 1/20

- Need to adjust both numerator and denominator

Smoothing

Adding “pseudocounts” to the observed counts when estimating P(X | Y) is called **smoothing**

Smoothing makes the estimated probabilities less extreme

- It is one way to perform regularization in Naïve Bayes (reduce overfitting)

Generative vs Discriminative

The conventional wisdom is that discriminative models generally perform better because they directly model what you care about, P(Y | X)

When to use generative models?

- Generative models have been shown to need less training data to reach peak performance
- Generative models are more conducive to unsupervised and semi-supervised learning
    - More on that point next week