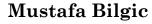
CS 584 - MACHINE LEARNING

TOPIC: NAÏVE BAYES





♦ http://www.cs.iit.edu/~mbilgic



https://twitter.com/bilgicm

CLASSIFICATION

- Input: $\vec{X} = \langle X_1, X_2, \dots, X_n \rangle$
- Output: Y

- We have seen
 - Candidate elimination to find the full version space
 - Decision trees

BAYES CLASSIFIER

$$P(Y \mid \vec{X}) = \frac{P(\vec{X} \mid Y)P(Y)}{P(\vec{X})} = \frac{P(Y)P(X_1, X_2, ..., X_n \mid Y)}{P(X_1, X_2, ..., X_n)}$$

$$P(X_1, X_2, ..., X_n) = \sum_{y} P(Y = y) P(X_1, X_2, ..., X_n \mid Y = y)$$

Assuming all variables are binary, how many independent parameters are needed for the Bayes classifier?

BAYES CLASSIFIER

AYES CLASSIFIER

$$P(Y \mid \vec{X}) = \frac{P(\vec{X} \mid Y)P(Y)}{P(\vec{X})} = \frac{P(Y)P(X_1, X_2, ..., X_n \mid Y)}{P(X_1, X_2, ..., X_n)}$$

P(A/B): P(B/A) P(A)

$$P(X_1, X_2, ..., X_n) = \sum_{y} P(Y = y) P(X_1, X_2, ..., X_n \mid Y = y)$$

Assuming all variables are binary, how many independent parameters

are needed for the Bayes classifier?

$$P(Y) \rightarrow 1 \quad \text{in} \quad P(X_1, -X_1 \mid Y) \quad \text{in} \quad \text{in} \quad P(X_1, -X_1 \mid Y) \quad \text{in} \quad \text{in}$$

Naïve Bayes Assumption

$$X_i \perp X_j \mid Y$$

Naïve Bayes

Bayes rule:

$$P(Y \mid X_1, X_2, ..., X_n) = \frac{P(Y)P(X_1, X_2, ..., X_n \mid Y)}{\sum_{y} P(y)P(X_1, X_2, ..., X_n \mid y)}$$

Assuming $X_i \perp X_j \mid Y$, naïve Bayes:

$$P(Y \mid X_1, X_2, ..., X_n) = \frac{P(Y) \prod P(X_i \mid Y)}{\sum_{y} P(y) \prod P(X_i \mid y)}$$

Assuming all variables are binary, how many independent parameters are needed for the naive Bayes classifier?

PARAMETER ESTIMATION

- Given a dataset $\mathcal{D} = \{\langle \vec{X}[m], Y[m] \rangle\}$, how can we estimate
 - P(Y)
 - $P(X_i \mid Y)$
- Intuitive idea: count and normalize
 - But, why is this the right idea? Or, is it even the right idea?

TOPIC SWITCH

• Probability estimation from data

ZERO PROBABILITIES

Assume

- $P(X_i = T | yes) = 0$ and $P(X_i = T | no) > 0$; and
- $P(X_j \mid yes) > 0$ and $P(X_j \mid no) > 0$ for all other features
- What is $P(yes \mid \vec{X})$ if $X_i = T$?

Assume

- $P(X_i = T | yes) = 0$ and $P(X_i = T | no) = 0$; and
- $P(X_j \mid yes) > 0$ and $P(X_j \mid no) > 0$ for all other features
- What is $P(yes \mid \vec{X})$ if $X_i = T$?
- Are these common?
- What can we do?

ZERO PROBABILITIES

Assume

- $P(X_i = T | yes) = 0$ and $P(X_i = T | no) > 0$; and
- $P(X_j \mid yes) > 0$ and $P(X_j \mid no) > 0$ for all other features
- What is $P(yes \mid \vec{X})$ if $X_i = T$?

Assume

- $P(X_i = T | yes) = 0$ and $P(X_i = T | no) = 0$; and
- $P(X_j \mid yes) > 0$ and $P(X_j \mid no) > 0$ for all other features
- What is $P(yes \mid \vec{X})$ if $X_i = T$?
- Are these common?
- What can we do?

LAPLACE SMOOTHING

- For parameter estimates, instead of MLE, use Bayesian
- Assume uniform prior
- Use the mean of the posterior as the parameter

Multiplying Several Probability Values

- Assume we have 1,000 features
- What is the product of 1,001 probability values?
 - p = np.random.random(1001)
 - p c = np.clip(p, 0.01, 0.99)
 - print(np.product(p c))
- o In naïve Bayes,
 - $a = P(Y = T) \prod P(X_i | Y = T)$
 - $b = P(Y = F) \prod P(X_i | Y = F)$
 - $P\left(Y = T \middle| \vec{X}\right) = \frac{a}{a+b}$
 - If a = b = 0 in your code, then what?

Converting LogJoint to Cond Probs

$$\circ \log \left(P\left(Y = yes, \vec{X} \right) \right) =$$

• $\log(P(Y = yes)) + \sum \log(P(X_i \mid Y = yes))$

$$\circ \log \left(P\left(Y = no, \vec{X} \right) \right) =$$

- $\log(P(Y = no)) + \sum \log(P(X_i \mid Y = no))$
- What is $P(Y = yes \mid \vec{X})$?
- Calculate it without causing an underflow.
 - You cannot use np.exp $(\log(P(Y = yes, \vec{X})))$
 - For example, try np.exp(-1000)

Converting LogJoint to Cond Probs

$$P(Y = yes \mid \vec{X}) = ?$$

Naïve Bayes Implementations

- o Bernoulli / categorical naïve Bayes
 - Features are assumed to be binary / categorical
- Multinomial naïve Bayes
 - $P(\vec{X} \mid y)$ is a multinomial distribution
- o Gaussian naïve Bayes
 - Each $p(x_i \mid y)$ is a Gaussian distribution