Lecture 1: From Coins to Machine Learning

Pedro Savarese

TTI

2018

Table of Contents

So far

- Probability:
  - Expectation
  - Conditional Probability
  - Concentration bounds (Markov)
- Coins:
  - How to estimate bias of a coin (Maximum Likelihood Estimation)
  - How to distinguish two coins (Bayes Rule)
  - How to get performance guarantees (Hoeffding and Chernoff bounds)

Now

- Introduction to Machine Learning
- Actually, we have been doing Machine Learning already!
- Distinguishing two coins is similar to:
    - Distinguishing between pictures of dogs and cats (computer vision)
    - Distinguishing between steering a car to the left and right (autonomous cars)
    - Distinguishing between texts written by myself and by someone else (non-intrusive biometry, security)
- We will do exactly that today: distinguish between spam and non-spam e-mail

Table of Contents

Why Machine Learning?

- Problem: spam detection
- Spam e-mail:
  "FREE POLYPHONIC RINGTONE Text SUPER to 87131 to collect FREE POLY TONE of the week now!"
  "WIN: We have a winner! YOU won an iPod!"
- Not spam:
  "pedro why didn't u call on your lunch?"
  "No. Yes please. Been swimming?"
- Goal: design computer program f that tells if e-mail $\mathbf{x}$ is spam or not
  - $f(\mathbf{x}) =$ "spam" if e-mail $\mathbf{x}$ is spam
  - $f(\mathbf{x}) =$ "not spam" if e-mail $\mathbf{x}$ is not spam

Manual Problem Solving

Algorithm 1 Spam Detection Algorithm

Input: e-mail string **x**
if ("free" or "now!" or "collect" or "win" or "won" in **x**), **then**
    return "spam"
else
    return "not spam"
end if

Manual Problem Solving

- How good is the program?
- Manually designing solutions: complex, requires time and effort
- Machine Learning: program **learns** to detect spam from **examples**
- Automated problem solving, little human effort

Table of Contents

Machine Learning

Machine Learning pipeline:

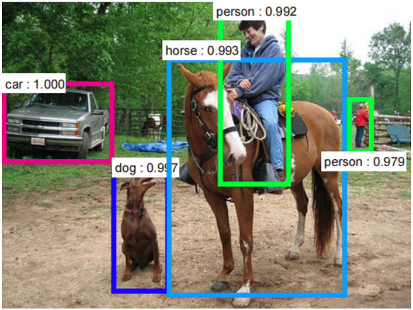
1. Collect examples $\mathbf{X} = (\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}, \dots)$
2. Collect labels $\mathbf{y} = (\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \mathbf{y}^{(3)}, \dots)$
   - For spam problem, $\mathbf{y}^{(i)} = \{\text{spam}, \text{not spam}\}$
3. Use $\mathbf{X}, \mathbf{y}$ to train model f
   - What model do we use? How do we train it? Focus of next lectures
4. Use f to classify new observations: $f(\mathbf{x}) = \hat{y}$

Machine Learning

Machine Translation:

Machine Learning

Object Detection:



Machine Learning

Colorization:

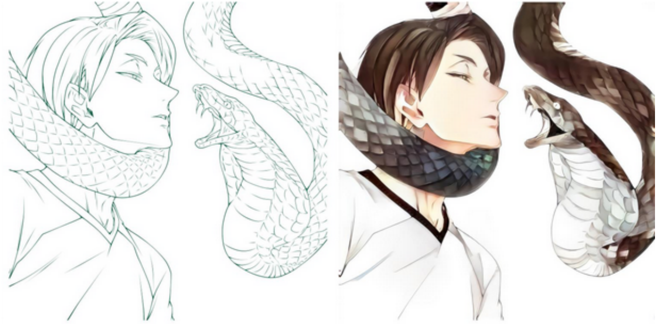


Table of Contents

Bernoulli Language Model

- $\mathbf{X}$ is collection of sentences
- Each sentence $\mathbf{x} \in \mathbf{X}$ is composed of words: $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, \dots\}$
- Goal: learn a language model $p(\mathbf{x})$
- How? Maximum Likelihood Estimation: maximize $\mathcal{L} = \prod_{\mathbf{x} \in \mathbf{X}} p(\mathbf{x})$
- Bernoulli model for $p(\mathbf{x})$
  - Have $|\mathcal{V}|$ **independent** biased coins ($\mathcal{V}$ is set of all words)
  - Coin corresponding to word v has heads probability $p(v)$
  - If heads, then word v occurs in $\mathbf{x}$
  - $p(\mathbf{x}) = \prod_{v \in \mathcal{V}} p(v)^{1\{v \in \mathbf{x}\}} (1 - p(v))^{1\{v \notin \mathbf{x}\}}$
- MLE: $p(v) = \frac{\sum_{\mathbf{x} \in \mathbf{X}} 1\{v \in \mathbf{x}\}}{|\mathbf{X}|}$

Naive Bayes

Spam detection with coins

- Goal: learn $p(\mathbf{x}, y) = p(y)p(\mathbf{x}|y)$
- Here $\mathbf{x}$ is an e-mail and $y \in \{\text{spam, not spam}\}$
- Again, model as collection of coins, but words depend on y
- Intuition:
  - First flip y coin, if heads then $\mathbf{x}$ is spam, if tails then not spam
  - Flip word coins $p(v|y)$ for occurrences: coins are different depending on y

$$p(\mathbf{x}|y) = \prod_{v \in \mathcal{V}} p(v|y)^{1\{v \in \mathbf{x}\}}(1 - p(v|y))^{1\{v \notin \mathbf{x}\}}$$

The joint probability will be:

$$p(\mathbf{x}, y) = p(\text{spam})^{1\{y=\text{spam}\}}(1 - p(\text{spam}))^{1\{y=\text{not spam}\}} \times \prod_{v \in \mathcal{V}} p(v|y)^{1\{v \in \mathbf{x}\}}(1 - p(v|y))^{1\{v \notin \mathbf{x}\}}$$

Naive Bayes

- Final step: learn $p(\text{spam})$ and $p(v|y)$ for each $v \in \mathcal{V}$
- Use **Maximum Likelihood Estimation**
- Likelihood:

$$\mathcal{L} = \prod_{(\mathbf{x},y)\in(\mathbf{X},\mathbf{y})} \Big(p(\text{spam})^{1\{y=\text{spam}\}}(1-p(\text{spam}))^{1\{y=\text{not spam}\}} \times \prod_{v\in\mathcal{V}} p(v|y)^{1\{v\in\mathbf{x}\}}(1-p(v|y))^{1\{v\notin\mathbf{x}\}} \Big)$$

- Log-likelihood:

$$\log\mathcal{L} = \sum_{(\mathbf{x},y)\in(\mathbf{X},\mathbf{y})} \Big(1\{y=\text{spam}\}\log p(\text{spam}) + 1\{y=\text{not spam}\}\log(1-p(\text{spam})) + \sum_{v\in\mathcal{V}} 1\{v\in\mathbf{x}\}\log p(v|y) + 1\{v\notin\mathbf{x}\}\log(1-p(v|y)) \Big)$$

Naive Bayes

- MLE for $p(\text{spam})$:

$$\frac{\partial \log \mathcal{L}}{\partial p(\text{spam})} = \sum_{(\mathbf{x},y)\in(\mathbf{X},\mathbf{y})} \frac{1\{y = \text{spam}\}}{p(\text{spam})} - \frac{1\{y = \text{not spam}\}}{1 - p(\text{spam})} = 0$$

$$p(\text{spam}) = \frac{c(\text{spam})}{N}$$

Naive Bayes

- MLE for $p(v'|y')$:

$$\frac{\partial \log \mathcal{L}}{\partial p(v'|y')} = \sum_{(\mathbf{x},y)\in(\mathbf{X},\mathbf{y})} \frac{1\{v' \in \mathbf{x} \wedge y = y'\}}{p(v'|y')} - \frac{1\{v' \notin \mathbf{x} \wedge y = y'\}}{1 - p(v'|y')} = 0$$

$$p(v'|y') = \frac{c(v' \wedge y')}{c(y')}$$

Naive Bayes

- Solutions:

$$p(\text{spam}) = \frac{c(\text{spam})}{N}$$

$$p(v'|\text{spam}) = \frac{c(v' \wedge \text{spam})}{c(\text{spam})}$$

$$p(v'|\text{not spam}) = \frac{c(v' \wedge \text{not spam})}{c(\text{not spam})}$$

Naive Bayes

- Once we learn $p(\mathbf{x}, y)$, how do we classify e-mails?
- Bayes: $p(\text{spam}|\mathbf{x}) = \frac{p(\text{spam})p(\mathbf{x}|\text{spam})}{p(\mathbf{x})}$
- Predict spam if: $p(\text{spam}|\mathbf{x}) > p(\text{not spam})p(\mathbf{x}|\text{not spam})$
- ...if $\frac{p(\text{spam})p(\mathbf{x}|\text{spam})}{p(\mathbf{x})} > \frac{p(\text{not spam})p(\mathbf{x}|\text{not spam})}{p(\mathbf{x})}$
- ...if $p(\text{spam})p(\mathbf{x}|\text{spam}) > p(\text{not spam})p(\mathbf{x}|\text{not spam})$

Naive Bayes

- Now: try on real data
  - Estimate $p(\text{spam})$
  - Estimate $p(v|\text{spam})$ and $p(v|\text{not spam})$ for each word $v \in \mathcal{V}$
  - Implement prediction: compute $p(\text{spam})p(\mathbf{x}|\text{spam})$ and $p(\text{not spam})p(\mathbf{x}|\text{not spam})$, implement prediction rule
  - Check words v with highest $p(v|\text{spam})$ and $p(v|\text{not spam})$
  - Check words v with highest $\frac{p(v|\text{spam})}{p(v|\text{not spam})}$