Lecture 1: From Coins to Machine Learning

Pedro Savarese

TTI

2018
Table of Contents

1. What have we learned?
2. Why Machine Learning?
3. Pipeline and Practical Examples
4. Naive Bayes
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)
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

1. What have we learned?
2. Why Machine Learning?
3. Pipeline and Practical Examples
4. Naive Bayes
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 $x$ is spam or not
  
  $f(x) = "$spam" if e-mail x is spam
  
  $f(x) = "$not spam" if e-mail x is not spam
Manual Problem Solving

Algorithm 1 Spam Detection Algorithm

1: Input: e-mail string $x$
2: if ("free" or "now!" or "collect" or "win" or "won" in $x$), then
3: return "spam"
4: else
5: return "not spam"
6: 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

1. What have we learned?
2. Why Machine Learning?
3. Pipeline and Practical Examples
4. Naive Bayes
Machine Learning pipeline:

1. Collect examples $\mathbf{X} = (\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}, \ldots)$
2. Collect labels $\mathbf{y} = (\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \mathbf{y}^{(3)}, \ldots)$
   - For spam problem, $\mathbf{y}^{(i)} = \{\text{spam, not spam}\}$
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 Translation:

Automatic translation is typically done with Machine Learning.

Jidō hon'yaku wa, tsūjō, kikai gakushū
Machine Learning

Object Detection:
Machine Learning

Colorization:
Table of Contents

1. What have we learned?
2. Why Machine Learning?
3. Pipeline and Practical Examples
4. Naive Bayes
What have we learned?
Why Machine Learning?
Pipeline and Practical Examples
Naive Bayes

Bernoulli Language Model

- \( \mathbf{X} \) is collection of sentences
- Each sentence \( \mathbf{x} \in \mathbf{X} \) is composed of words: \( \mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots \} \)
- 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: \( p(v) = \frac{\sum_{\mathbf{x} \in \mathbf{X}} 1\{v \in \mathbf{x}\}}{|\mathbf{X}|} \)
Spam detection with coins

- Goal: learn $p(x, y) = p(y)p(x|y)$
- Here $x$ 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 $x$ is spam, if tails then not spam
  - Flip word coins $p(v|y)$ for occurrences: coins are different depending on $y$

$$p(x|y) = \prod_{v \in \mathcal{V}} p(v|y)^{1\{v \in x\}} (1 - p(v|y))^{1\{v \not\in x\}}$$

The joint probability will be:

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

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

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

- Log-likelihood:

$$
\log \mathcal{L} = \sum_{(x,y) \in (X,y)} \left( 1\{y = \text{spam}\} \log p(\text{spam}) + 1\{y = \text{not spam}\} \log (1 - p(\text{spam})) 
+ \sum_{v \in V} \left( 1\{v \in x\} \log p(v|y) + 1\{v \not\in x\} \log (1 - p(v|y)) \right) \right)
$$
Naive Bayes

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

\[
\frac{\partial \log L}{\partial p(\text{spam})} = \sum_{(x,y) \in (X,Y)} \frac{1\{y = \text{spam}\}}{p(\text{spam})} - \frac{1\{y = \text{not spam}\}}{1 - p(\text{spam})} = 0
\]

\[
p(\text{spam}) = \frac{c(\text{spam})}{N}
\]
Naive Bayes

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

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

$$p(v'|y') = \frac{c(v' \land y')}{c(y')}$$
Naive Bayes

- Solutions:

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

\[
p(v'|\text{spam}) = \frac{c(v' \land \text{spam})}{c(\text{spam})}
\]

\[
p(v'|\text{not spam}) = \frac{c(v' \land \text{not spam})}{c(\text{not spam})}
\]
Once we learn $p(x, y)$, how do we classify e-mails?

Bayes: $p(\text{spam} | x) = \frac{p(\text{spam})p(x | \text{spam})}{p(x)}$

Predict spam if: $p(\text{spam} | x) > p(\text{not spam})p(x | \text{not spam})$

...if $\frac{p(\text{spam})p(x | \text{spam})}{p(x)} > \frac{p(\text{not spam})p(x | \text{not spam})}{p(x)}$

...if $p(\text{spam})p(x | \text{spam}) > p(\text{not spam})p(x | \text{not spam})$
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 $p(\text{spam})p(x|\text{spam})$ and $p(\text{not spam})p(x|\text{not spam})$, implement prediction rule
- Check words $v$ with highest $p(v|\text{spam})$ and $p(v|\text{not spam})$
- Check words $v$ with highest $\frac{p(v|\text{spam})}{p(v|\text{not spam})}$