24: Naïve Bayes

David Varodayan
March 4, 2020
Adapted from slides by Lisa Yan
Today’s plan

Machine Learning

• Inefficient classification: Brute force Bayes
• Naïve Bayes
Our path

Deep Learning

Linear Regression (lite)

Naïve Bayes

Logistic Regression

Unbiased estimators

\( \bar{X}, S^2 \)

\( \theta_{MLE} \)

\( \theta_{MAP} \)

now
Multinomial MLE and MAP

Model: Multinomial with $m$ outcomes:
$p_i$ probability of outcome $i$

Observe:
$n_i = \# \text{ of trials with outcome } i$
Total of $\sum_{i=1}^{m} n_i$ trials

MLE

$$p_i = \frac{n_i}{\sum_{i=1}^{m} n_i}$$

MAP with Laplace smoothing (Laplace estimate)

$$p_i = \frac{n_i + 1}{\sum_{i=1}^{m} n_i + m}$$
Supervised Learning

1. Real World Problem
2. Model the problem
3. Formal Model $\theta$
4. Learning Algorithm
5. Prediction Function $\theta^*$
6. Evaluation score
7. Training Data
8. Testing Data
Modeling

(not the focus of this class)

Real World Problem

Model the problem

Formal Model $\theta$

Learning Algorithm

Testing Data

Prediction Function $\theta^*$

Evaluation score
Real World Problem

Model the problem

Formal Model $\theta$

Learning Algorithm

Prediction Function $\theta^*$

Training Data

Testing Data

Evaluation score
Real World Problem

Model the problem

Formal Model $\theta$

Learning Algorithm

Training Data

Testing Data

Prediction Function $\theta^*$

Evaluation score
Machine Learning (formally)

Many different forms of “Machine Learning”

• We focus on the problem of **prediction** based on observations.

**Goal**

Based on observed $X$, predict unseen $Y$

• **Features**
  Vector $X$ of $m$ observed variables
  $X = (X_1, X_2, ..., X_m)$

• **Output**
  Variable $Y$ (also called **class label**)

**Model**

$\hat{Y} = g(X)$, a function of observations $X$

• **Classification**
  prediction when $Y$ is discrete

• **Regression**
  prediction when $Y$ is continuous
Training data

\[(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(n)}, y^{(n)})\]

\(n\) datapoints, generated i.i.d.

Each datapoint \(i\) is \((x^{(i)}, y^{(i)})\):

- \(m\) features: \(x^{(i)} = (x_1^{(i)}, x_2^{(i)}, \ldots, x_m^{(i)})\)
- A single output \(y^{(i)}\)
- Independent of all other datapoints

Training Goal: Use these \(n\) datapoints to learn a model \(\hat{Y} = g(X)\) that predicts \(Y\)
Example datasets

Heart

Ancestry

Netflix

23andMe
Classification terminology check

Training data: \((x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(n)}, y^{(n)})\)

A. \(x^{(i)}\)  
B. \(y^{(i)}\)  
C. \((x^{(i)}, y^{(i)})\)  
D. \(x_j^{(i)}\)

1: like movie  
0: dislike movie

<table>
<thead>
<tr>
<th>Movie 1</th>
<th>Movie 2</th>
<th>Movie (m)</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Movie 1" /></td>
<td><img src="image2.png" alt="Movie 2" /></td>
<td><img src="image3.png" alt="Movie m" /></td>
<td><img src="image4.png" alt="Output" /></td>
</tr>
</tbody>
</table>

User 1 1. 1 0 … 1 2. 1  
User 2 3. 1 1 … 0 0  
… 4. 0 … 1  
User \(n\) 0 0 … 1
Classification terminology check

Training data: \((x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), ..., (x^{(n)}, y^{(n)})\)

<table>
<thead>
<tr>
<th>User</th>
<th>Movie 1</th>
<th>Movie 2</th>
<th>Movie m</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td></td>
<td>...</td>
</tr>
<tr>
<td>n</td>
<td>0</td>
<td>0</td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

1: like movie  
0: dislike movie

A. \(x^{(i)}\)  
B. \(y^{(i)}\)  
C. \((x^{(i)}, y^{(i)})\)  
D. \(x_j^{(i)}\)

\(i\): \(i\)-th user  
\(j\): movie \(j\)
Regression: Predicting real numbers

Training data: $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(n)}, y^{(n)})$

<table>
<thead>
<tr>
<th>Year</th>
<th>CO2 levels</th>
<th>Sea level</th>
<th>Feature $m$</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Year 1</td>
<td>338.8</td>
<td>0</td>
<td>...</td>
<td>1</td>
</tr>
<tr>
<td>Year 2</td>
<td>340.0</td>
<td>1</td>
<td>...</td>
<td>0</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Year $n$</td>
<td>340.76</td>
<td>0</td>
<td>...</td>
<td>1</td>
</tr>
</tbody>
</table>

Global Land-Ocean temperature
Classification: Harry Potter Sorting Hat

\[ \hat{Y} = 1 \]

\[ X = (1, 1, 1, 0, 0, \ldots, 1) \]
Announcements

Problem Set 6
Due: Wednesday 3/11
Covers: Up to Lecture 25
Extra Python Office Hours: Saturday 3/7, 3-5PM

Regrades
Pset 1 to 5 and Midterm regrades to close on 3/11 at 1pm

Autograded Coding Problems
Run your code in the command line or install Pycharm following directions on Pset 6 webpage

Late Day Reminder
No late days permitted past last day of the quarter, 3/13
Today’s plan

Machine Learning

• Inefficient classification: Brute force Bayes
• Naïve Bayes
Classification: Having a healthy heart

Feature 1: Region of Interest (ROI) is healthy (1) or unhealthy (0)

How can we predict the class label heart is healthy (1) or unhealthy (0)?

One possible solution: Use Bayes.

<table>
<thead>
<tr>
<th>Feature 1</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Patient 1</td>
<td>1</td>
</tr>
<tr>
<td>Patient 2</td>
<td>0</td>
</tr>
<tr>
<td>Patient n</td>
<td>0</td>
</tr>
</tbody>
</table>
Brute force Bayes

Classification (for one patient):

Choose the class label that is most likely given the data.

\[
\hat{Y} = \arg \max_{y=\{0,1\}} \hat{P}(Y \mid X)
\]

\[
= \arg \max_{y=\{0,1\}} \frac{\hat{P}(X \mid Y) \hat{P}(Y)}{\hat{P}(X)}
\]

(Bayes’ Theorem)

\[
= \arg \max_{y=\{0,1\}} \hat{P}(X \mid Y) \hat{P}(Y)
\]

(1/\hat{P}(X) is a positive constant w.r.t \( Y \))

\[\hat{P}(Y = 1 \mid x) : \text{estimated probability a heart is healthy given } x\]

\[x: \text{whether region of interest (ROI) is healthy (1) or unhealthy (0)}\]
Parameters for Brute Force Bayes

\[ \hat{Y} = \arg \max_{y=\{0,1\}} \hat{P}(X|Y)\hat{P}(Y) \]

Parameters:
- \( \hat{P}(X|Y) \) for all \( X \) and \( Y \)
- \( \hat{P}(Y) \) for all \( Y \)

| \( X_1 \) | \( \hat{P}(X|Y = 0) \) | \( \hat{P}(X|Y = 1) \) |
|---|---|---|
| 0 | \( \theta_1 \) | \( \theta_3 \) |
| 1 | \( \theta_2 \) | \( \theta_4 \) |

<table>
<thead>
<tr>
<th>( Y )</th>
<th>( \hat{P}(Y) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>( \theta_5 )</td>
</tr>
<tr>
<td>1</td>
<td>( \theta_6 )</td>
</tr>
</tbody>
</table>

Conditional probability tables \( \hat{P}(X|Y) \)

Marginal probability table \( \hat{P}(Y) \)

Training Goal:
Use \( n \) datapoints to learn \( 2 \cdot 2 + 2 = 6 \) parameters.
Training: Estimate parameters $\hat{P}(X|Y)$

<table>
<thead>
<tr>
<th>Feature 1</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Patient 1</td>
<td>1</td>
</tr>
<tr>
<td>Patient 2</td>
<td>1</td>
</tr>
<tr>
<td>Patient $n$</td>
<td>0</td>
</tr>
</tbody>
</table>

$\hat{P}(X|Y = 0)$ and $\hat{P}(X|Y = 1)$ are both multinomials with 2 outcomes!

Use MLE or Laplace (MAP) estimate for parameters $P(X|Y)$
Training: MLE estimates, $\hat{P}(X|Y)$

<table>
<thead>
<tr>
<th>Feature 1</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Patient 1</td>
<td>1</td>
</tr>
<tr>
<td>Patient 2</td>
<td>1</td>
</tr>
<tr>
<td>Patient $n$</td>
<td>0</td>
</tr>
</tbody>
</table>

| $X_1$ | $\hat{P}(X|Y = 0)$ | $\hat{P}(X|Y = 1)$ |
|-------|--------------------|--------------------|
| 0     | 0.4                | 0.0                |
| 1     | 0.6                | 1.0                |

MLE of $\hat{P}(X_1 = x|Y = y) = \frac{\#(X_1 = x,Y = y)}{\#(Y = y)}$

Just count!
Training: Laplace (MAP) estimates, $\hat{P}(X|Y)$

<table>
<thead>
<tr>
<th>Feature 1</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X_1 = 0$</td>
<td>0.4</td>
</tr>
<tr>
<td>$X_1 = 1$</td>
<td>0.6</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Patient</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Patient 1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Patient 2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Patient $n$</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

MLE of $\hat{P}(X_1 = x|Y = y) = \frac{\#(X_1 = x,Y = y)}{\#(Y = y)}$

Just count!

| $X_1 = 0$ | 0.42 | 0.01 |
| $X_1 = 1$ | 0.58 | 0.99 |

Laplace of $\hat{P}(X_1 = x|Y = y) = \frac{\#(X_1 = x,Y = y) + 1}{\#(Y = y) + 2}$

Just count + add imaginary trials!
\[ \hat{Y} = \text{arg max}_{y=\{0,1\}} \hat{P}(X|Y)\hat{P}(Y) \]

| (MAP) | \( \hat{P}(X|Y = 0) \) | \( \hat{P}(X|Y = 1) \) |
|-------|----------------|----------------|
| \( X_1 = 0 \) | 0.42 | 0.01 |
| \( X_1 = 1 \) | 0.58 | 0.99 |

\[
\begin{align*}
\hat{P}(X_1 = 1|Y = 0)\hat{P}(Y = 0) &= 0.58 \cdot 0.10 \approx 0.058 \\
\hat{P}(X_1 = 1|Y = 1)\hat{P}(Y = 1) &= 0.99 \cdot 0.90 \approx 0.891
\end{align*}
\]

New patient has a healthy ROI (\( X_1 = 1 \)). What is your prediction, \( \hat{Y} \)?

A. \( 0.058 < 0.5 \)  \( \Rightarrow \)  \( \hat{Y} = 1 \)

B. \( 0.891 > 0.5 \)  \( \Rightarrow \)  \( \hat{Y} = 1 \)

C. \( 0.058 < 0.891 \)  \( \Rightarrow \)  \( \hat{Y} = 1 \)
Brute force Bayes: $m = 100$ (# features)

<table>
<thead>
<tr>
<th>Feature 1</th>
<th>Feature 2</th>
<th>Feature 100</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Patient 1</td>
<td>1</td>
<td>0</td>
<td>...</td>
</tr>
<tr>
<td>Patient 2</td>
<td>1</td>
<td>1</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Patient $n$</td>
<td>0</td>
<td>0</td>
<td>...</td>
</tr>
</tbody>
</table>

This won’t be too bad, right?
Brute force Bayes: $m = 100$ (# features)

$$\hat{Y} = \arg \max_{y\in\{0,1\}} \hat{P}(Y | X)$$

$$= \arg \max_{y\in\{0,1\}} \frac{\hat{P}(X|Y)\hat{P}(Y)}{\hat{P}(X)}$$

$$= \arg \max_{y\in\{0,1\}} \hat{P}(X|Y)\hat{P}(Y)$$

$\hat{P}(Y = 1 | x)$: estimated probability a heart is healthy given $x$

$X = (X_1, X_2, ..., X_{100})$: whether 100 regions of interest (ROI) are healthy (1) or unhealthy (0)

How many parameters do we have to learn?

- $\hat{P}(X|Y)$
- $\hat{P}(Y)$

A. $2 \cdot 2 + 2 = 6$
B. $2 \cdot 100 + 2 = 202$
C. $2 \cdot 2^{100} + 2 = \text{a lot}$
The problem with our Brute force Bayes classifier

\[ \hat{Y} = \arg \max_{y=\{0,1\}} P(Y | X) \]

\[ = \arg \max_{y=\{0,1\}} \frac{\hat{P}(X|Y) \hat{P}(Y)}{\hat{P}(X)} \]

\[ = \arg \max_{y=\{0,1\}} \hat{P}(X|Y) \hat{P}(Y) \]

Estimating this joint conditional distribution will require too many parameters.

What if we could make a simplifying (but naïve) assumption—that \( X_1, \ldots, X_m \) are conditionally independent given \( Y \)?
Today’s plan

Machine Learning

• Inefficient classification: Brute force Bayes

• Naïve Bayes
**The Naïve Bayes assumption**

\[ X_1, \ldots, X_m \text{ are conditionally independent given } Y. \]

Our prediction for \( Y \) is a function of \( X \)

\[ \hat{Y} = g(X) = \arg \max_{y=\{0,1\}} \hat{P}(Y | X) = \arg \max_{y=\{0,1\}} \frac{\hat{P}(X|Y)\hat{P}(Y)}{\hat{P}(X)} \]  

(Bayes)

\[ = \arg \max_{y=\{0,1\}} \hat{P}(X|Y)\hat{P}(Y) \]  

(Normalization constant)

\[ = \arg \max_{y=\{0,1\}} \left( \prod_{i=1}^{m} \hat{P}(X_i|Y) \right)\hat{P}(Y) \]

Naïve Bayes Assumption
Naïve Bayes Classifier

\[ \hat{Y} = \arg \max_{y=\{0,1\}} \left( \prod_{i=1}^{m} \hat{P}(X_i|Y) \right) \hat{P}(Y) \]

Training

What is the Big-O of # of parameters we need to learn?

A. \( O(m) \)
B. \( O(2^m) \)
C. other
Naïve Bayes Classifier

\[ \hat{Y} = \arg \max_{y=\{0,1\}} \left( \prod_{i=1}^{m} \hat{P}(X_i|Y) \right) \hat{P}(Y) \]

Training

Use MLE or Laplace (MAP)

for \( i = 1, \ldots, m \):

\[ \hat{P}(X_i|Y = 0), \hat{P}(X_i|Y = 1) \]

\[ \hat{P}(Y = 0), \hat{P}(Y = 1) \]

Testing

\[ \hat{Y} = \arg \max_{y=\{0,1\}} \left( \log \hat{P}(Y) + \sum_{i=1}^{m} \log \hat{P}(X_i|Y) \right) \]

(for numeric stability)
Naïve Bayes for TV shows

Will a user like the Pokémon TV series?

Observe indicator variables $\mathbf{X} = (X_1, X_2)$:

- $X_1 = 1$: “likes Star Wars”
- $X_2 = 1$: “likes Harry Potter”

Output $Y$ indicator:

- $Y = 1$: “likes Pokémon”
Training: Naïve Bayes for TV shows (MLE)

Observe indicator vars. $X = (X_1, X_2)$:
- $X_1$: “likes Star Wars”
- $X_2$: “likes Harry Potter”

Predict $Y$: “likes Pokémon”

1. How many datapoints ($n$) are in our train data?
2. Compute MLE estimates for $\hat{P}(X_1|Y)$:

$$
\hat{Y} = \arg\max_{y=\{0,1\}} \left( \prod_{i=1}^{m} \hat{P}(X_i|Y) \right) \hat{P}(Y)
$$

<table>
<thead>
<tr>
<th>$X_1$</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Y$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>13</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$X_2$</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Y$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>1</td>
<td>7</td>
<td>10</td>
</tr>
</tbody>
</table>

Training data counts
Training: Naïve Bayes for TV shows (MLE)

Observe indicator vars. $X = (X_1, X_2)$:
- $X_1$: “likes Star Wars”
- $X_2$: “likes Harry Potter”

Predict $Y$: “likes Pokémon”

1. How many datapoints ($n$) are in our train data?

2. Compute MLE estimates for $\hat{p}(X_1|Y)$:

$$\hat{Y} = \arg \max_{y=\{0,1\}} \left( \prod_{i=1}^{m} \hat{p}(X_i|Y) \right) \hat{p}(Y)$$

<table>
<thead>
<tr>
<th>$X_1$</th>
<th>$Y$</th>
<th>$X_2$</th>
<th>$Y$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

Training data counts

\[
\begin{array}{c|cc}
Y & 0 & 1 \\
\hline
0 & 3 & 10 \\
1 & 4 & 13 \\
\end{array}
\]

\[
\begin{array}{c|cc}
Y & 0 & 1 \\
\hline
0 & 5 & 8 \\
1 & 7 & 10 \\
\end{array}
\]

\[
n = 30
\]

\[
\begin{array}{c|cc}
Y & 0 & 1 \\
\hline
0 & 3/13 \approx 0.23 & 10/13 \approx 0.77 \\
1 & 4/17 \approx 0.24 & 13/17 \approx 0.76 \\
\end{array}
\]
Training: Naïve Bayes for TV shows (MLE)

Observe indicator vars. \( X = (X_1, X_2) \):
- \( X_1 \): “likes Star Wars”
- \( X_2 \): “likes Harry Potter”

Predict \( Y \): “likes Pokémon”

Training data counts

<table>
<thead>
<tr>
<th>( X_1 )</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>( Y )</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>( Y )</td>
<td>1</td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>( X_2 )</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>( Y )</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>( Y )</td>
<td>1</td>
<td>7</td>
</tr>
</tbody>
</table>

Training MLE estimates: just count.

\[
\hat{P}(X_i = x|Y = y) = \frac{\#(X_i = x, Y = y)}{\#(Y = y)}
\]

\[
\hat{P}(Y = y) = \frac{\#(Y = y)}{n}
\]

\( \hat{y} = \text{arg max}_{y \in \{0, 1\}} \left( \prod_{i=1}^{m} \hat{P}(X_i|Y) \right) \hat{P}(Y) \)
Training: Naïve Bayes for TV shows (MLE)

Observe indicator vars. $X = (X_1, X_2)$:
- $X_1$: “likes Star Wars”
- $X_2$: “likes Harry Potter”

Predict $Y$: “likes Pokémon”

<table>
<thead>
<tr>
<th>$Y$</th>
<th>$X_1$</th>
<th>$X_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.23</td>
<td>0.38</td>
</tr>
<tr>
<td>1</td>
<td>0.24</td>
<td>0.41</td>
</tr>
</tbody>
</table>

$\hat{Y} = \arg \max_{y \in \{0, 1\}} \left( \prod_{i=1}^{m} \hat{P}(X_i | Y) \right) \hat{P}(Y)$

Now that we’ve trained and found parameters, it’s time to classify new users!
Observe indicator vars. $X = (X_1, X_2)$:

- $X_1$: “likes Star Wars”
- $X_2$: “likes Harry Potter”

Predict $Y$: “likes Pokémon”

<table>
<thead>
<tr>
<th>$X_1$</th>
<th>$Y$</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0.23</td>
<td>0.77</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0.24</td>
<td>0.76</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$X_2$</th>
<th>$Y$</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0.38</td>
<td>0.62</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0.41</td>
<td>0.59</td>
</tr>
</tbody>
</table>

Suppose a new person “likes Star Wars” ($X_1 = 1$) but “dislikes Harry Potter” ($X_2 = 0$).

Will they like Pokémon? Need to predict $Y$:

$$\hat{Y} = \arg \max_{y \in \{0,1\}} \hat{P}(X|Y)\hat{P}(Y) = \arg \max_{y \in \{0,1\}} \hat{P}(X_1|Y)\hat{P}(X_2|Y)\hat{P}(Y)$$

If $Y = 0$:

$$\hat{P}(X_1 = 1|Y = 0)\hat{P}(X_2 = 0|Y = 0)\hat{P}(Y = 0) = 0.77 \cdot 0.38 \cdot 0.43 = 0.126$$

If $Y = 1$:

$$\hat{P}(X_1 = 1|Y = 1)\hat{P}(X_2 = 0|Y = 1)\hat{P}(Y = 1) = 0.76 \cdot 0.41 \cdot 0.57 = 0.178$$

Since term is greatest when $Y = 1$, predict $\hat{Y} = 1$
Training: Naïve Bayes for TV shows (MAP)

Observe indicator vars. \( X = (X_1, X_2) \):
- \( X_1 \): “likes Star Wars”
- \( X_2 \): “likes Harry Potter”

Predict \( Y \): “likes Pokémon”

What are our MAP estimates using Laplace smoothing for \( \hat{P}(X_i|Y) \) and \( \hat{P}(Y) \)?

<table>
<thead>
<tr>
<th>( X_1 )</th>
<th>( Y )</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>( Y )</td>
<td>0</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>13</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>( X_2 )</th>
<th>( Y )</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>( Y )</td>
<td>0</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>1</td>
<td>7</td>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>

Training data counts

\( \hat{P}(X_i = x|Y = y) \):
- A. \( \frac{\#(X_i=x,Y=y)}{\#(Y=y)} \)
- B. \( \frac{\#(X_i=x,Y=y)+1}{\#(Y=y)+2} \)
- C. \( \frac{\#(X_i=x,Y=y)+1}{\#(Y=y)+4} \)

\( \hat{P}(Y = y) \):
- A. \( \frac{\#(Y=y)}{\#(Y=y)+2} \)
- B. \( \frac{\#(Y=y)+1}{n} \)
- C. \( \frac{\#(Y=y)+1}{n+2} \)
Training: Naïve Bayes for TV shows (MAP)

Observe indicator vars. \( X = (X_1, X_2) \):
- \( X_1 \): “likes Star Wars”
- \( X_2 \): “likes Harry Potter”

Predict \( Y \): “likes Pokémon”

<table>
<thead>
<tr>
<th>( Y )</th>
<th>( X_1 )</th>
<th>( X_2 )</th>
<th>( Y )</th>
<th>( X_1 )</th>
<th>( X_2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Training data counts

<table>
<thead>
<tr>
<th>( X_1 )</th>
<th>( X_2 )</th>
<th>( Y )</th>
</tr>
</thead>
</table>
| 0 | 0 | 0
| 1 | 1 | 1 |

\[
\hat{P}(X_i = x|Y = y) = \frac{(X_i = x,Y = y) + 1}{(Y = y) + 2}
\]

\[
\hat{P}(Y = y) = \frac{(Y = y) + 1}{n + 2}
\]

Training MAP estimates: just count + imaginary trials.

\[
\hat{y} = \arg \max_{y\in\{0,1\}} \left( \prod_{i=1}^m \hat{P}(X_i|Y) \right) \hat{P}(Y)
\]
Naïve Bayes Model is a Bayesian Network

Naïve Bayes Assumption

\[ P(X|Y) = \prod_{i=1}^{m} P(X_i|Y) \quad \Rightarrow \quad P(X,Y) = P(Y) \prod_{i=1}^{m} P(X_i|Y) \]

Which Bayesian Network encodes this conditional independence?

A.  
\[
\begin{align*}
&\hspace{2cm} \cdots \hspace{2cm} \cdots \hspace{2cm} \cdots \\
&X_1 \quad \quad X_2 \quad \quad \ldots \quad \quad X_n \\
&\quad Y
\end{align*}
\]

B.  
\[
\begin{align*}
&\hspace{2cm} \cdots \hspace{2cm} \cdots \hspace{2cm} \cdots \\
&X_1 \quad \quad X_2 \quad \quad \ldots \quad \quad X_n \\
&\quad Y
\end{align*}
\]
Extra slides

Naïve Bayes with spam classification
What is Bayes doing in my mail server?

Let’s get Bayesian on your spam:

Content analysis details: (49.5 hits, 7.0 required)

0.9 RCVD_IN_PBL           RBL: Received via a relay in Spamhaus PBL
[93.40.189.29 listed in zen.spamhaus.org]

1.5 URIBL_WS_SURBL        Contains an URL listed in the WS SURBL blocklist
[URIs: recargas.cn]

5.0 URIBL_JP_SURBL        Contains an URL listed in the JP SURBL blocklist
[URIs: recargas.cn]

5.0 URIBL_OB_SURBL        Contains an URL listed in the OB SURBL blocklist
[URIs: recargas.cn]

5.0 URIBL_SC_SURBL        Contains an URL listed in the SC SURBL blocklist
[URIs: recargas.cn]

2.0 URIBL_BLACK           Contains an URL listed in the URIBL blacklist
[URIs: recargas.cn]

8.0 BAYES_99             BODY: Bayesian spam probability is 99 to 100%
[score: 1.0000]

A Bayesian Approach to Filtering Junk E-Mail

Mehran Sahami*   Susan Dumais†   David Heckerman†   Eric Horvitz†

*Gates Building 1A
Computer Science Department
Stanford University
Stanford, CA 94305-9010
sahami@cs.stanford.edu

†Microsoft Research
Redmond, WA 98052-6399
{sdumais,heckerman,horvitz}@microsoft.com

Abstract

In addressing the growing problem of junk e-mail on the Internet, we examine methods for the automated
categorization of messages. The approaches we consider are based on different assumptions about
the probability distribution over the content of the messages. We demonstrate the effectiveness of
Bayesian methods by applying them to a large corpus of example e-mail messages. Our experiments
show that Bayesian methods can be very effective in filtering junk e-mail, and they also allow a
Bayesian spam probability to be attached to each message.
Email classification

Goal Based on email content $X$, predict if email is spam or not.

Features Consider a lexicon $m$ words (for English: $m \approx 100,000$).

$$X = (X_1, X_2, \ldots, X_m), \ m \text{ indicator variables}$$

$X_i = 1$ if word $i$ appeared in document

Output $Y = 1$ if email is spam

Note: $m$ is huge. Make Naïve Bayes assumption: $P(X|\text{spam}) = \prod_{i=1}^{m} P(X_i|\text{spam})$

Appearances of words in email are conditionally independent given the email is spam or not
Naïve Bayes Email classification

Train set

\( n \) previous emails \((x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(n)}, y^{(n)})\)

\( x^{(j)} = (x_1^{(j)}, x_2^{(j)}, \ldots, x_m^{(j)}) \) for each word, whether it appears in email \( j \)

\( y^{(j)} = 1 \) if spam, 0 if not spam

Training

Estimate probabilities \( \hat{P}(Y) \) and \( \hat{P}(X_i | Y) \) for all \( i \)

Which estimator should we use?

A. MLE

B. Laplace estimate (MAP)

C. Other MAP estimate

D. Both A and B
Naïve Bayes Email classification

Train set

$n$ previous emails $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(n)}, y^{(n)})$

$x^{(j)} = \left(x^{(1)}_{j}, x^{(2)}_{j}, \ldots, x^{(m)}_{j}\right)$ for each word, whether it appears in email $j$

$y^{(j)} = 1$ if spam, $0$ if not spam

Training

Estimate probabilities $\hat{P}(Y)$ and $\hat{P}(X_{i}|Y)$ for all $i$

Which estimator should we use?

A. MLE
B. Laplace estimate (MAP)
C. Other MAP estimate
D. Both A and B

- Many words are likely to not appear at all in the training set, so we want to avoid $0$ probabilities.
- Laplace estimate is simple.
Naïve Bayes Email classification

Train set

$n$ previous emails $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(n)}, y^{(n)})$

$x^{(j)} = (x_1^{(j)}, x_2^{(j)}, \ldots, x_m^{(j)})$ for each word, whether it appears in email $j$

$y^{(j)} = 1$ if spam, $0$ if not spam

Training

Estimate probabilities $\hat{P}(Y)$ and $\hat{P}(X_i | Y)$ for all $i$

Laplace estimate:

$\hat{P}(X_i = 1 | Y = \text{spam}) = \frac{(\# \text{ spam emails with word } i) + 1}{(\text{total \# spam emails}) + 2}$

Testing (Classification)

For a new email:

• Generate $X = (X_1, X_2, \ldots, X_m)$

• Classify as spam or not using Naïve Bayes assumption

$\hat{Y} = \arg \max_{y=\{0,1\}} \left( \log \hat{P}(Y) + \sum_{i=1}^{m} \log \hat{P}(X_i | Y) \right)$

Use logs for numeric stability
How well does Naïve Bayes perform?

After training, you can test with another set of data, called the **test set**.
- Test set also has known values for $Y$ so we can see how often we were right/wrong in our predictions $\hat{Y}$.

Typical work flow:
- Have a dataset of 1789 emails (1578 spam, 211 ham)
- Train set: First 1538 emails (by time)
- Test set: Next 251 messages

Evaluation criteria on test set:

\[
\text{precision} = \frac{\text{(# correctly predicted class } Y)}{\text{(# predicted class } Y)}
\]

\[
\text{recall} = \frac{\text{(# correctly predicted class } Y)}{\text{(# real class } Y \text{ messages})}
\]

<table>
<thead>
<tr>
<th></th>
<th>Spam Prec.</th>
<th>Spam Recall</th>
<th>Non-spam Prec.</th>
<th>Non-spam Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Words only</td>
<td>97.1%</td>
<td>94.3%</td>
<td>87.7%</td>
<td>93.4%</td>
</tr>
<tr>
<td>Words + addtl features</td>
<td>100%</td>
<td>98.3%</td>
<td>96.2%</td>
<td>100%</td>
</tr>
</tbody>
</table>