# Lecture 17: Naive Bayes (Mitchell Chapter 6)

CS 167: Machine Learning

## Motivating Example

A lab test for a certain kind of cancer is known to have the following properties

- when the patient actually has this cancer, test correctly identifies it 98% of the time
- when the patient does not actually have this cancer, test correctly gives negative result 97% of the time
- 0.8% of population has this cancer

Let's say you take the test and it comes back positive. **What is the probability that you actually have this cancer?**

- **A**: around 98%
- **B**: around 20%
- **C**: around 2%

PollEv.com/manley
or Text MANLEY
and response to 37607



## Some Probability Notation

$P(x)$: the probability that $x$ holds without knowing anything else

$P(x|y)$: the probability of $x$ given $y$

### Applied to Machine Learning

$P(h)$: **Prior Probability** of hypothesis $h$ - may reflect background knowledge we have that $h$ is a correct hypothesis

$P(D)$: probability that training data $D$ will be observed

$P(D|h)$: probability of observing $D$ given some world where hypothesis $h$ holds

$P(h|D)$: **Posterior Probability** that $h$ holds given observed training data $D$ (*this is what we really want to know!*)



## Bayes Theorem

**Bayes Theorem:**

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

Gives us a way to combine prior information with observations

Rev. Thomas Bayes

## Back to our example

A lab test for a certain kind of cancer is known to have the following properties

- when the patient actually has this cancer, test correctly identifies it 98% of the time
- when the patient does not actually have this cancer, test correctly gives negative result 97% of the time
- 0.8% of population has this cancer

$$P(cancer) = \qquad P(\neg cancer) =$$
$$P(+|cancer) = \qquad P(-|cancer) =$$
$$P(+|\neg cancer) = \qquad P(-|\neg cancer) =$$

**Which hypothesis has the higher likelihood of being true?**
$h = cancer$ **or** $h = \neg cancer$**?**



## Using Bayes Theorem

**Bayes Theorem:**

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

Gives us a way to combine prior information with observations

We've been using it more like in this form:

$$P(h|D) = \frac{P(D|h)P(h)}{P(D|h)P(h) + P(D|\neg h)P(\neg h)}$$

But really, to find the most likely hypothesis, you don't need to worry about normalizing by $P(D)$ or $P(D|h)P(h) + P(D|\neg h)P(\neg h)$



## Maximum a posterior hypothesis

From a set of possible hypotheses, the hypothesis with the greatest

$$P(h|D) = P(D|h)P(h)$$

is the **maximum a posterior** (MAP) hypothesis

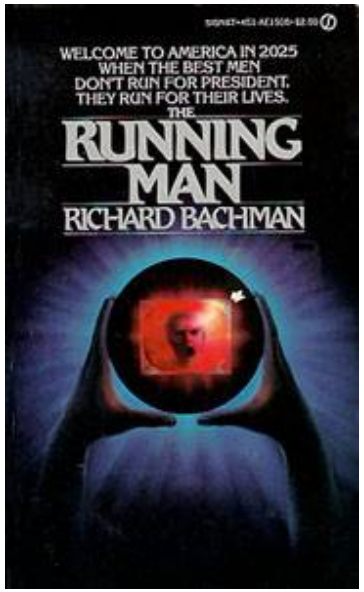
sometimes denoted $h_{MAP}$

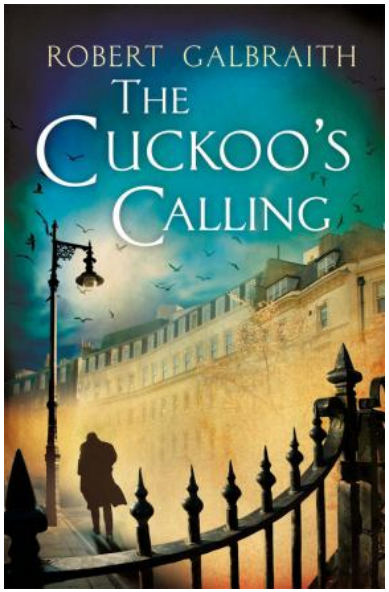
Brute Force MAP Learning algorithm is

1. For each possible hypothesis $h$, compute $P(h|D)$
2. output $h$ with greatest $P(h|D)$



## Problem: Authorship Detection



## Problem: Authorship Detection

Relative frequency of using these three words by each author

| | J. K. Rowling | Stephen King |
|---|---|---|
| began | 0.2 | 0.7 |
| suddenly | 0.3 | 0.2 |
| quite | 0.5 | 0.1 |

**Who is more likely to have written the text: *began suddenly*?**

- **A**: J. K. Rowling
- **B**: Stephen King

PollEv.com/manley
or Text MANLEY
and response to 37607

**Exercise:** *How about quite began?*



## Interesting Observation

**if** the following are true

- all priors are the same (i.e., $P(h_i) = P(h_j)$ for any pair of hypotheses)
- deterministic, noise-free data

**then**
*every consistent learner outputs a MAP hypothesis*

even if they don't explicitly compute probabilities



## Bayes Theorem applied to classification

**Classification problem**: attribute values $\langle a_1, a_2 \dots a_n \rangle$, target function is some class among $\{v_1, v_2, \dots, v_c\}$
Most probable class value for new instance is the $v_j$ resulting in the largest value of

$$P(v_j|a_1, a_2 \dots a_n) = \frac{P(a_1, a_2 \dots a_n|v_j)P(v_j)}{P(a_1, a_2 \dots a_n)}$$

which is the same as the $v_j$ with largest

$$P(a_1, a_2 \dots a_n|v_j)P(v_j)$$



## Discussion Questions

Attribute values $a_1, a_2 \dots a_n$, looking for class $v_j$ with largest

$$P(a_1, a_2 \dots a_n|v_j)P(v_j)$$

**How can we estimate $P(v_j)$ for a given $v_j$?**
(Hint: thinking of credit data, what is the probability of *1 (creditable)* vs *2 (not creditable)*?)

**How can we estimate $P(a_1, a_2 \dots a_n|v_j)$?**
(Hint: When *1 (creditable)* is the class value, what is the probability of seeing Status of account=A11, Credit history = A34, Purpose = A43? When When *2 (not creditable)* is the class value, what is the probability of seeing Status of account=A11, Credit history = A34, Purpose = A43? )

## More Discussion

Estimating $P(a_1, a_2 \ldots a_n | v_j)$ is not feasible

- we'd need to see every possible combination of attributes, several times each for good estimates

A simpler task: **How can we estimate $P(a_i|v_j)$ for some attribute $i$?**

Simplifying assumption for the **Naive Bayes** algorithm: assume attribute values are *conditionally independent*given the target value. That is, assume

$$P(a_1, a_2 \ldots a_n | v_j) = P(a_1|v_j) \cdot P(a_2|v_j) \cdot \cdots \cdot P(a_n|v_j)$$



## Naive Bayes Algorithm

Naive_Bayes_Learn(*examples*)
  For each target value $v_j$
    $\hat{P}(v_j) \leftarrow$ estimate $P(v_j)$
    For each attribute value $a_i$ of each attribute $a$
      $\hat{P}(a_i|v_j) \leftarrow$ estimate $P(a_i|v_j)$

estimate $P(v_j) = \#(\text{examples with } v_j)/\#(\text{examples})$
estimate $P(a_i|v_j) = \#(\text{examples with } v_j \text{ and } a_i)$ / $\#(\text{examples with } v_j)$

Classify_New_Instance($x$)

$$v_{NB} = \text{argmax}_{v_j \in V} \hat{P}(v_j) \prod_{a_i \in x} \hat{P}(a_i|v_j)$$



## Naive Bayes Subtleties

Obviously, the independence assumption doesn't always hold:

$$P(a_1, a_2 \ldots a_n | v_j) = P(a_1|v_j) \cdot P(a_2|v_j) \cdot \cdots \cdot P(a_n|v_j)$$

But, it usually works ok anyway.

**Group Discussion Question:** What does this simplifying assumption mean for the authorship detection problem?



## Example

| Day | Outlook | Temperature | Humidity | Wind | PlayTennis |
|---|---|---|---|---|---|
| D1 | Sunny | Hot | High | Weak | No |
| D2 | Sunny | Hot | High | Strong | No |
| D3 | Overcast | Hot | High | Weak | Yes |
| D4 | Rain | Mild | High | Weak | Yes |
| D5 | Rain | Cool | Normal | Weak | Yes |
| D6 | Rain | Cool | Normal | Strong | No |
| D7 | Overcast | Cool | Normal | Strong | Yes |
| D8 | Sunny | Mild | High | Weak | No |
| D9 | Sunny | Cool | Normal | Weak | Yes |
| D10 | Rain | Mild | Normal | Weak | Yes |
| D11 | Sunny | Mild | Normal | Strong | Yes |
| D12 | Overcast | Mild | High | Strong | Yes |
| D13 | Overcast | Hot | Normal | Weak | Yes |
| D14 | Rain | Mild | High | Strong | No |

new instance:
$x = \langle Outlk = sun, Temp = cool, Humid = high, Wind = strong \rangle$

## Example Continued...

new instance:
$x = \langle Outlk = sun, Temp = cool, Humid = high, Wind = strong \rangle$

$P(\text{y}) = 9/14$ $P(\text{n}) = ?$
$P(\text{sun}|\text{y}) = 2/9$ $P(\text{sun}|\text{n}) = ?$
$P(\text{cool}|\text{y}) = 3/9$ $P(\text{cool}|\text{n}) = ?$
$P(\text{high}|\text{y}) = 3/9$ $P(\text{high}|\text{n}) = ?$
$P(\text{strong}|\text{y}) = 3/9$ $P(\text{strong}|\text{n}) = ?$

$P(\text{y}|x) =$
$P(\text{y})P(\text{sun}|\text{y})P(\text{cool}|\text{y})P(\text{high}|\text{y})P(\text{strong}|\text{y}) = \frac{9}{14} \cdot \frac{2}{9} \cdot \frac{3}{9} \cdot \frac{3}{9} \cdot \frac{3}{9} \approx 0.005$

$P(\text{n}|x) = ?$ **Exercise:** complete the example



## Issues

what if none of the training instances with target value $v_j$ have attribute value $a_i$? Then

$$\hat{P}(a_i|v_j) = 0, \text{ and...}$$
$$\hat{P}(v_j) \prod_i \hat{P}(a_i|v_j) = 0$$

We usually estimate $\hat{P}(a_i|v_j)$ as $\frac{n_c}{n}$ where

- $n$ is number of training examples for which $v = v_j$,
- $n_c$ number of examples for which $v = v_j$ and $a = a_i$

Instead, use *m-estimate*: $\frac{n_c + mp}{n+m}$ where

- $p$ is prior estimate for $\hat{P}(a_i|v_j)$ (e.g., $\frac{1}{2}$ for Y/N)
- $m$ is weight given to prior (i.e., number of "virtual" examples)



## Lab Exercise

`scikit-learn` has implementations for
3 common variants of **Naive Bayes**

Using default parameters, which one does the best on the *iris* data set?

Look at the documentation (including
`http://scikit-learn.org/stable/modules/naive_bayes.html` )

**Discuss:** Explain why the good algorithm(s) did well and why the bad one(s) didn't