CS 188: Artificial Intelligence
Naïve Bayes

Agent Testing Today!

Instructors: Anca Dragan--- University of California, Berkeley

[These slides were created by Dan Klein, Pieter Abbeel, Sergey Levine, with some materials from A. Farhadi. All CS188 materials are at http://ai.berkeley.edu.]
Machine Learning

- Up until now: how use a model to make optimal decisions

- Machine learning: how to acquire a model from data / experience
  - Learning parameters (e.g. probabilities)
  - Learning structure (e.g. BN graphs)
  - Learning hidden concepts (e.g. clustering)

- Today: model-based classification with Naive Bayes
Classification
Example: Spam Filter

- **Input:** an email
- **Output:** spam/ham

**Setup:**
- Get a large collection of example emails, each labeled “spam” or “ham”
- Note: someone has to hand label all this data!
- Want to learn to predict labels of new, future emails

**Features:** The attributes used to make the ham / spam decision
- Words: FREE!
- Text Patterns: $dd, CAPS
- Non-text: SenderInContacts
- ...
Example: Digit Recognition

- **Input:** images / pixel grids
- **Output:** a digit 0-9

**Setup:**
- Get a large collection of example images, each labeled with a digit
- Note: someone has to hand label all this data!
- Want to learn to predict labels of new, future digit images

**Features:** The attributes used to make the digit decision
- Pixels: (6,8)=ON
- Shape Patterns: NumComponents, AspectRatio, NumLoops
- …

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>??</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Other Classification Tasks

- Classification: given inputs $x$, predict labels (classes) $y$

- Examples:
  - Spam detection (input: document, classes: spam / ham)
  - OCR (input: images, classes: characters)
  - Medical diagnosis (input: symptoms, classes: diseases)
  - Automatic essay grading (input: document, classes: grades)
  - Fraud detection (input: account activity, classes: fraud / no fraud)
  - Customer service email routing
  - ... many more

- Classification is an important commercial technology!
Model-Based Classification
Model-Based Classification

- **Model-based approach**
  - Build a model (e.g. Bayes’ net) where both the label and features are random variables
  - Instantiate any observed features
  - Query for the distribution of the label conditioned on the features

- **Challenges**
  - What structure should the BN have?
  - How should we learn its parameters?
Naïve Bayes for Digits

- Naïve Bayes: Assume all features are independent effects of the label

- Simple digit recognition version:
  - One feature (variable) $F_{ij}$ for each grid position $<i,j>$
  - Feature values are on / off, based on whether intensity is more or less than 0.5 in underlying image
  - Each input maps to a feature vector, e.g.
    \[
    \{F_{0,0} = 0 \quad F_{0,1} = 0 \quad F_{0,2} = 1 \quad F_{0,3} = 1 \quad F_{0,4} = 0 \quad \ldots \quad F_{15,15} = 0\}
    \]
  - Here: lots of features, each is binary valued

- Naïve Bayes model:
  \[
P(Y|F_{0,0} \ldots F_{15,15}) \propto P(Y) \prod_{i,j} P(F_{i,j}|Y)
\]

- What do we need to learn?
A general Naïve Bayes model:

\[ P(Y, F_1 \ldots F_n) = P(Y) \prod_i P(F_i | Y) \]

- We only have to specify how each feature depends on the class.
- Total number of parameters is \textit{linear} in \( n \).
- Model is very simplistic, but often works anyway.
Inference for Naïve Bayes

- **Goal**: compute posterior distribution over label variable $Y$
  - Step 1: get joint probability of label and evidence for each label
  $$P(Y, f_1 \ldots f_n) = \begin{bmatrix} P(y_1, f_1 \ldots f_n) \\ P(y_2, f_1 \ldots f_n) \\ \vdots \\ P(y_k, f_1 \ldots f_n) \end{bmatrix}$$

  $$= \begin{bmatrix} \frac{P(y_1) \prod_i P(f_i|y_1)}{P(f_1 \ldots f_n)} \\ \frac{P(y_2) \prod_i P(f_i|y_2)}{P(f_1 \ldots f_n)} \\ \vdots \\ \frac{P(y_k) \prod_i P(f_i|y_k)}{P(f_1 \ldots f_n)} \end{bmatrix}$$

- Step 2: sum to get probability of evidence
- Step 3: normalize by dividing Step 1 by Step 2

$$P(Y|f_1 \ldots f_n) = \frac{P(Y, f_1 \ldots f_n)}{P(f_1 \ldots f_n)}$$
Naïve Bayes spam filter

Data:
- Collection of emails, labeled spam or ham
- Note: someone has to hand label all this data!
- Split into training, held-out, test sets

Classifiers
- Learn on the training set
- (Tune it on a held-out set)
- Test it on new emails

---

Dear Sir.
First, I must solicit your confidence in this transaction, this is by virtue of its nature as being utterly confidencial and top secret. …

TO BE REMOVED FROM FUTURE MAILINGS, SIMPLY REPLY TO THIS MESSAGE AND PUT "REMOVE" IN THE SUBJECT.

99 MILLION EMAIL ADDRESSES FOR ONLY $99

Ok, I know this is blatantly OT but I'm beginning to go insane. Had an old Dell Dimension XPS sitting in the corner and decided to put it to use, I know it was working pre being stuck in the corner, but when I plugged it in, hit the power nothing happened.
Naïve Bayes for Text

- **Bag-of-words Naïve Bayes:**
  - Features: $W_i$ is the word at position $i$
  - As before: predict label conditioned on feature variables (spam vs. ham)
  - As before: assume features are conditionally independent given label
  - New: each $W_i$ is identically distributed

- **Generative model:**
  \[ P(Y, W_1 \ldots W_n) = P(Y) \prod_i P(W_i|Y) \]

- **“Tied” distributions and bag-of-words**
  - Usually, each variable gets its own conditional probability distribution $P(F|Y)$
  - In a bag-of-words model:
    - Each position is identically distributed
    - All positions share the same conditions
    - Why make this assumption?
  - Called “bag-of-words” because model is insensitive to word order or reordering

When the lecture is over, remember to wake up the person sitting next to you in the lecture room.
Example: Spam Filtering

- **Model:** 
  \[ P(Y, W_1 \ldots W_n) = P(Y) \prod P(W_i|Y) \]

- **What are the parameters?**

| P(Y) | P(W|spam) | P(W|ham) |
|------|-----------|----------|
| ham: 0.66 | the: 0.0156 | the: 0.0210 |
| spam: 0.33 | to: 0.0153 | to: 0.0133 |
| | and: 0.0115 | of: 0.0119 |
| | of: 0.0095 | 2002: 0.0110 |
| | you: 0.0093 | with: 0.0110 |
| | a: 0.0086 | from: 0.0108 |
| | with: 0.0080 | and: 0.0107 |
| | from: 0.0075 | a: 0.0105 |
| | ... | ... |
Spam Example

\[
\begin{array}{|c|c|c|c|}
\hline
\text{Word} & P(w|\text{spam}) & P(w|\text{ham}) & \text{Tot Spam} & \text{Tot Ham} \\
\hline
\text{(prior)} & 0.33333 & 0.66666 & -1.1 & -0.4 \\
\hline
\end{array}
\]

Gary

would

you

like

to

lose

weight

while

you

sleep
What do we need in order to use Naïve Bayes?

- Inference method (we just saw this part)
  - Start with a bunch of probabilities: $P(Y)$ and the $P(F_i|Y)$ tables
  - Use standard inference to compute $P(Y|F_1...F_n)$
  - Nothing new here

- Estimates of local conditional probability tables
  - $P(Y)$, the prior over labels
  - $P(F_i|Y)$ for each feature (evidence variable)
  - These probabilities are collectively called the *parameters* of the model and denoted by $\Theta$
  - Up until now, we assumed these appeared by magic, but...
  - ...they typically come from training data counts
Parameter estimation (Naïve version)

- How do we estimate the conditional probability tables?
  - Naïve version: just count!
  - $P(Y)$ – what ratio of the data is the digit 1?
  - $P(F_i|Y)$ – what ratio of the digit 1 has this pixel on?

- Need to be careful though ... let’s see what goes wrong..
Important Concepts

- **Data**: labeled instances, e.g. emails marked spam/ham
  - Training set
  - Held out set
  - Test set

- **Features**: attribute-value pairs which characterize each x

- **Experimentation cycle**
  - Learn parameters (e.g. model probabilities) on training set
  - (Tune hyperparameters on held-out set)
  - Compute accuracy of test set
  - Very important: never “peek” at the test set!

- **Evaluation**
  - Accuracy: fraction of instances predicted correctly

- **Overfitting and generalization**
  - Want a classifier which does well on test data
  - Overfitting: fitting the training data very closely, but not generalizing well
  - Underfitting: fits the training set poorly
Underfitting and Overfitting
Overfitting

Degree 15 polynomial
Example: Overfitting

\[ P(\text{features}, C = 2) \]
\[ P(C = 2) = 0.1 \]
\[ P(\text{on}|C = 2) = 0.8 \]
\[ P(\text{on}|C = 2) = 0.1 \]
\[ P(\text{off}|C = 2) = 0.1 \]
\[ P(\text{on}|C = 2) = 0.01 \]

\[ P(\text{features}, C = 3) \]
\[ P(C = 3) = 0.1 \]
\[ P(\text{on}|C = 3) = 0.8 \]
\[ P(\text{on}|C = 3) = 0.9 \]
\[ P(\text{off}|C = 3) = 0.7 \]
\[ P(\text{on}|C = 3) = 0.0 \]

2 wins!!
Example: Overfitting

- \textit{relative} probabilities (odds ratios):

\[
\frac{P(W|\text{ham})}{P(W|\text{spam})} \quad \frac{P(W|\text{spam})}{P(W|\text{ham})}
\]

| south-west | : inf |
| nation     | : inf |
| morally    | : inf |
| nicely     | : inf |
| extent     | : inf |
| seriously  | : inf |
| ...        |      |

| screens    | : inf |
| minute     | : inf |
| guaranteed | : inf |
| $205.00    | : inf |
| delivery   | : inf |
| signature  | : inf |
| ...        |      |

What went wrong here?
Relative frequency parameters will overfit the training data!
- Just because we never saw a 3 with pixel (15,15) on during training doesn’t mean we won’t see it at test time
- Unlikely that every occurrence of “minute” is 100% spam
- Unlikely that every occurrence of “seriously” is 100% ham
- What about all the words that don’t occur in the training set at all?
- In general, we can’t go around giving unseen events zero probability

As an extreme case, imagine using the entire email as the only feature
- Would get the training data perfect (if deterministic labeling)
- Wouldn’t generalize at all
- Just making the bag-of-words assumption gives us some generalization, but isn’t enough

To generalize better: we need to smooth or regularize the estimates
Parameter Estimation
Parameter Estimation

- Estimating the distribution of a random variable
- **Elicitation**: ask a human (why is this hard?)
- **Empirically**: use training data (learning!)
  - E.g.: for each outcome $x$, look at the empirical rate of that value:
    \[
    P_{\text{ML}}(x) = \frac{\text{count}(x)}{\text{total samples}}
    \]
    \[
    P_{\text{ML}}(r) = \frac{2}{3}
    \]
  - This is the estimate that maximizes the likelihood of the data

\[
L(x, \theta) = \prod_i P_\theta(x_i) = \theta \cdot \theta \cdot (1 - \theta)
\]

\[
P_\theta(x = \text{red}) = \theta
\]
\[
P_\theta(x = \text{blue}) = 1 - \theta
\]
A billionaire tech entrepreneur asks you a question:

He says: I have a thumbtack, if I flip it, what’s the probability it will fall with the nail up?

You say: Please flip it a few times:

You say: The probability is:

P(H) = 3/5

He says: Why???

You say: Because...
Your First Consulting Job

- $P(\text{Heads}) = \theta$, $P(\text{Tails}) = 1-\theta$

- **Flips are i.i.d.**: $D = \{x_i | i = 1 \ldots n\}$, $P(D | \theta) = \Pi_i P(x_i | \theta)$
  - Independent events
  - Identically distributed according to unknown distribution

- **Sequence $D$ of $\alpha_H$ Heads and $\alpha_T$ Tails**

$$P(D | \theta) = \theta^{\alpha_H} (1 - \theta)^{\alpha_T}$$
Maximum Likelihood Estimation

- **Data**: Observed set $D$ of $\alpha_H$ Heads and $\alpha_T$ Tails
- **Hypothesis space**: Binomial distributions
- **Learning**: finding $\theta$ is an optimization problem
  - What’s the objective function?
    \[
P(D \mid \theta) = \theta^{\alpha_H}(1 - \theta)^{\alpha_T}
    \]
- **MLE**: Choose $\theta$ to maximize probability of $D$
  \[
  \hat{\theta} = \arg \max_{\theta} P(D \mid \theta) = \arg \max_{\theta} \ln P(D \mid \theta)
  \]
Maximum Likelihood Estimation

\[ \hat{\theta} = \arg \max_{\theta} \ln P(D | \theta) \]

\[ = \arg \max_{\theta} \ln \theta^{\alpha_H} (1 - \theta)^{\alpha_T} \]

- Set derivative to zero, and solve!

\[ \frac{d}{d\theta} \ln P(D | \theta) = \frac{d}{d\theta} [\ln \theta^{\alpha_H} (1 - \theta)^{\alpha_T}] \]

\[ = \frac{d}{d\theta} [\alpha_H \ln \theta + \alpha_T \ln(1 - \theta)] \]

\[ = \alpha_H \frac{d}{d\theta} \ln \theta + \alpha_T \frac{d}{d\theta} \ln(1 - \theta) \]

\[ = \frac{\alpha_H}{\theta} - \frac{\alpha_T}{1 - \theta} = 0 \]

\[ \hat{\theta}_{MLE} = \frac{\alpha_H}{\alpha_H + \alpha_T} \]
Maximum Likelihood?

- Relative frequencies are the maximum likelihood estimates

\[ \theta_{ML} = \arg \max_{\theta} P(X|\theta) \]
\[ = \arg \max_{\theta} \prod_i P_\theta(X_i) \]

\[ P_{ML}(x) = \frac{\text{count}(x)}{\text{total samples}} \]

- Another option is to consider the most likely parameter value given the data

\[ \theta_{MAP} = \arg \max_{\theta} P(\theta|X) \]
\[ = \arg \max_{\theta} P(X|\theta)P(\theta)/P(X) \]

???
Unseen Events
Laplace Smoothing

- Laplace’s estimate:
  - Pretend you saw every outcome once more than you actually did

\[
P_{LAP}(x) = \frac{c(x) + 1}{\sum_x [c(x) + 1]}
\]

\[
= \frac{c(x) + 1}{N + |X|}
\]

- Can derive this estimate with \textit{Dirichlet priors} (see cs281a)

\[
P_{ML}(X) =
\]

\[
P_{LAP}(X) =
\]
Laplace Smoothing

- Laplace’s estimate (extended):
  - Pretend you saw every outcome \( k \) extra times

\[
P_{LAP,k}(x) = \frac{c(x) + k}{N + k|X|}
\]

- What’s Laplace with \( k = 0 \)?
- \( k \) is the strength of the prior

- Laplace for conditionals:
  - Smooth each condition independently:

\[
P_{LAP,k}(x|y) = \frac{c(x, y) + k}{c(y) + k|X|}
\]

\[
P_{LAP,0}(X) =
\]

\[
P_{LAP,1}(X) =
\]

\[
P_{LAP,100}(X) =
\]
Estimation: Linear Interpolation*

- In practice, Laplace can perform poorly for $P(X|Y)$:
  - When $|X|$ is very large
  - When $|Y|$ is very large

- Another option: linear interpolation
  - Also get the empirical $P(X)$ from the data
  - Make sure the estimate of $P(X|Y)$ isn’t too different from the empirical $P(X)$

$$P_{LIN}(x|y) = \alpha \hat{P}(x|y) + (1.0 - \alpha) \hat{P}(x)$$

- What if $\alpha$ is 0? 1?

- For even better ways to estimate parameters, as well as details of the math, see cs281a, cs288
Real NB: Smoothing

- For real classification problems, smoothing is critical
- New odds ratios:

\[
\frac{P(W|\text{ham})}{P(W|\text{spam})} \quad \frac{P(W|\text{spam})}{P(W|\text{ham})}
\]

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>helvetica</td>
<td>11.4</td>
<td></td>
</tr>
<tr>
<td>seems</td>
<td>10.8</td>
<td></td>
</tr>
<tr>
<td>group</td>
<td>10.2</td>
<td></td>
</tr>
<tr>
<td>ago</td>
<td>8.4</td>
<td></td>
</tr>
<tr>
<td>areas</td>
<td>8.3</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>...</td>
</tr>
</tbody>
</table>

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>verdana</td>
<td>28.8</td>
<td></td>
</tr>
<tr>
<td>Credit</td>
<td>28.4</td>
<td></td>
</tr>
<tr>
<td>ORDER</td>
<td>27.2</td>
<td></td>
</tr>
<tr>
<td>&lt;FONT&gt;</td>
<td>26.9</td>
<td></td>
</tr>
<tr>
<td>money</td>
<td>26.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>...</td>
</tr>
</tbody>
</table>

Do these make more sense?
Tuning
Tuning on Held-Out Data

- Now we’ve got two kinds of unknowns
  - Parameters: the probabilities $P(X|Y)$, $P(Y)$
  - Hyperparameters: e.g. the amount / type of smoothing to do, $k$, $\alpha$

- What should we learn where?
  - Learn parameters from training data
  - Tune hyperparameters on different data
    - Why?
  - For each value of the hyperparameters, train and test on the held-out data
  - Choose the best value and do a final test on the test data
First step: get a baseline
- Baselines are very simple “straw man” procedures
- Help determine how hard the task is
- Help know what a “good” accuracy is

Weak baseline: most frequent label classifier
- Gives all test instances whatever label was most common in the training set
- E.g. for spam filtering, might label everything as ham
- Accuracy might be very high if the problem is skewed
- E.g. calling everything “ham” gets 66%, so a classifier that gets 70% isn’t very good...

For real research, usually use previous work as a (strong) baseline
The confidence of a probabilistic classifier:
- Posterior over the top label
  \[ \text{confidence}(x) = \max_y P(y|x) \]
- Represents how sure the classifier is of the classification
- Any probabilistic model will have confidences
- No guarantee confidence is correct

Calibration
- Weak calibration: higher confidences mean higher accuracy
- Strong calibration: confidence predicts accuracy rate
- What’s the value of calibration?
Summary

- Bayes rule lets us do diagnostic queries with causal probabilities
- The naïve Bayes assumption takes all features to be independent given the class label
- We can build classifiers out of a naïve Bayes model using training data
- Smoothing estimates is important in real systems
- Classifier confidences are useful, when you can get them