### Machine Learning

- Up till now: how to reason or make decisions using a model
- Machine learning: how to select a model on the basis of data / experience
  - Learning parameters (e.g. probabilities)
  - Learning structure (e.g. BN graphs)
  - Learning hidden concepts (e.g. clustering)

### Classification

- In classification, we learn to predict labels (classes) for inputs
- Examples:
  - Spam detection (input: document, classes: spam / ham)
  - OCR (input: images, classes: characters)
  - Medical diagnosis (input: symptoms, classes: diseases)
  - Automatic essay grader (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!

### Bayes Nets for Classification

- One method of classification:
  - Features are observed variables
  - Y is the query variable
  - Use probabilistic inference to compute most likely Y
  
  \[ y = \arg \max_y \ P(y|f_1 \ldots f_n) \]
  
- You already know how to do this inference

---

### Simple Classification

- Simple example: two binary features
  - This is a naïve Bayes model

\[
\begin{align*}
P(m|s, f) & = \frac{P(s, f|m)P(m)}{P(s, f)} & \text{direct estimate} \\
P(m|s, f) & = \frac{P(s|m)P(f|m)P(m)}{P(s, f)} & \text{Bayes estimate (no assumptions)} \\
\end{align*}
\]

\[
\begin{align*}
P(m, s, f) & = P(s|m)P(f|m)P(m) \\
P(\bar{m}, s, f) & = P(s|\bar{m})P(f|\bar{m})P(\bar{m}) \\
\end{align*}
\]
General Naïve Bayes

- A general naïve Bayes model:

\[ P(\text{Cause, Effect}_1 \ldots \text{Effect}_n) = \frac{P(\text{Cause}) \prod P(\text{Effect}_i | \text{Cause})}{P(\text{Effect}_1 | \text{Cause}) \ldots P(\text{Effect}_n)} \]

\[ |C| \times |E|^n \text{ parameters} \]

- We only specify how each feature depends on the class
- Total number of parameters is linear in n

General Naïve Bayes

- What do we need in order to use naïve Bayes?
  - Inference (you know this part)
    - For fixed evidence, build \( P(C, e) \), that is, \( P(c, e) \) for each \( c \)
    - Sum out \( C \) to get \( P(e) \)
    - Divide to get \( P(C | e) \)
  - Estimates of local conditional probability tables
    - \( P(C) \), the prior over causes
    - \( P(E | C) \) for each evidence variable
    - These probabilities are collectively called the parameters of the model and denoted by \( \theta \)
    - These typically come from observed data: we'll look at this now

Inference for Naïve Bayes

- Goal: compute posterior over causes
  - Step 1: get joint probability of causes and evidence
    \[ P(C, e_1 \ldots e_n) = P(c_1, e_1 \ldots e_n) \cdot \frac{P(c_2)}{P(c_1)} \cdot \frac{P(e_2 | c_2)}{P(e_2 | c_1)} \cdot \ldots \cdot \frac{P(e_n | c_n)}{P(e_n | c_{n-1})} \]
    \[ \frac{P(c_n)}{P(e_1 \ldots e_n)} \]
  - Step 2: get probability of evidence
  - Step 3: renormalize

A Digit Recognizer

- Input: pixel grids
- Output: a digit 0-9

Naïve Bayes for Digits

- Simple version:
  - One feature \( F_{i,j} \) for each grid position \( <i,j> \)
  - Feature values are on/off based on whether intensity is more or less than 0.5
  - Input maps to feature vector, e.g.

\[ \begin{array}{cccc}
0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
\end{array} \]

- Naïve Bayes model:

\[ P(C, F_{0,0} \ldots F_{15,15}) = P(C) \prod_{i,j} P(F_{i,j} | C) \]

- What do we need to learn?

Examples: CPTs
Parameter Estimation

- Estimating distribution of random variables like \( X \) or \( X|Y \)
- Empirically: use training data
  - For each value \( x \), look at the empirical rate of that value:
    \[
    \hat{p}(x) = \frac{\text{count}(x)}{\text{total samples}}
    \]
  - This estimate maximizes the likelihood of the data
    \[
    L(x, \theta) = \prod \hat{p}(x_i^n)
    \]
- Elicitation: ask a human!
  - Usually need domain experts, and sophisticated ways of eliciting probabilities (e.g. betting games)
  - Trouble calibrating

Naive Bayes for Text

- Naive Bayes:
  - Predict unknown cause (spam vs. ham)
  - Assume evidence (e.g. the words) to be independent
- Generative model
  \[
  P(C, W_1 \ldots W_n) = P(C) \prod_i P(W_i|C)
  \]
- Tied distributions and bag-of-words
  - Usually, each variable gets its own conditional probability distribution \( P(E|C) \)
  - In a bag-of-words model
    - Each position is identically distributed
    - All share the same distributions
    - Why make this assumption?

Spam Example

| Word | P(w|spam) | P(w|ham) | Tot Spam | Tot Ham |
|------|----------|----------|----------|---------|
| prior| 0.33333  | 0.66666  | -1.1     | -0.4    |

\[ P(\text{spam} | w) = 98.9 \]

A Spam Filter

- 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

Example: Spam Filtering

- Model:
  \[
  P(C, W_1 \ldots W_n) = P(C) \prod_i P(W_i|C)
  \]
- What are the parameters?
  \[
  P(C) \quad P(W|\text{spam}) \quad P(W|\text{ham})
  \]
  - ham: 0.33
  - spam: 0.66
  - to: 0.0153
  - and: 0.0115
  - you: 0.0093
  - with: 0.0086
  - from: 0.0075
  - a: 0.0100

Example: Overfitting

\[ 2 \text{ wins!!} \]
Example: Spam Filtering

- Raw probabilities don’t affect the posteriors; relative probabilities (odds ratios) do:

\[
\begin{align*}
P(W|\text{ham}) & \quad P(W|\text{spam}) \\
P(W|\text{ham}) & \quad P(W|\text{ham})
\end{align*}
\]

<table>
<thead>
<tr>
<th>Word</th>
<th>ham</th>
<th>spam</th>
</tr>
</thead>
<tbody>
<tr>
<td>southwest</td>
<td>inf</td>
<td>inf</td>
</tr>
<tr>
<td>nation</td>
<td>inf</td>
<td>inf</td>
</tr>
<tr>
<td>morally</td>
<td>inf</td>
<td>inf</td>
</tr>
<tr>
<td>nicely</td>
<td>inf</td>
<td>inf</td>
</tr>
<tr>
<td>seriously</td>
<td>inf</td>
<td>inf</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

What went wrong here?

Generalization and Overfitting

- Relative frequency parameters will overfit the training data:
  - 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?
  - 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

Estimation: Smoothing

- Problems with maximum likelihood estimates:
  - If I flip a coin once, and it’s heads, what’s the estimate for P(heads)?
  - What if I flip 10 times with 8 heads?
  - What if I flip 10M times with 8M heads?

- Basic idea:
  - We have some prior expectation about parameters (here, the probability of heads)
  - Given little evidence, we should skew towards our prior
  - Given a lot of evidence, we should listen to the data

Estimation: Laplace Smoothing

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

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

- Can derive this as a MAP estimate with Dirichlet priors (see cs281a)

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

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

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

- Laplace for conditionals:
  - Smooth each condition independently:

\[
P_{\text{LAP,100}}(x|y) = \frac{c(x,y) + k}{c(y) + k|X|}
\]
**Estimation: Linear Interpolation**

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

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

\[ P_{LIN}(x|y) = \alpha P(x|y) + (1 - \alpha) P(x) \]

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

- For even better ways to estimate parameters, as well as details of the math see cs281a, cs294

---

**Real NB: Smoothing**

- For real classification problems, smoothing is critical

- New odds ratios:

|      | $P(W|\text{ham})$ | $P(W|\text{spam})$ |
|------|-------------------|-------------------|
| helvetica | 11.4              | 28.8              |
| seems   | 10.8              | 28.4              |
| group   | 10.2              | 27.2              |
| ago     | 8.4               | 26.9              |
| areas   | 8.3               | 26.5              |
| ...     | ...               | ...               |

*Do these make more sense?*

---

**Tuning on Held-Out Data**

- Now we’ve got two kinds of unknowns
  - Parameters: the probabilities $P(Y|X)$, $P(Y)$
  - Hyperparameters, like the amount of smoothing to do: $\alpha$, $\nu$

- Where to learn?
  - Learn parameters from training data
  - Must 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

---

**Baselines**

- First task: 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

- For real research, usually use previous work as a (strong) baseline

---

**Confidences from a Classifier**

- The confidence of a probabilistic classifier:
  - Posterior over the top label

\[ \text{confidence}(x) = \arg \max P(y|x) \]

- 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?

---

**Errors, and What to Do**

- Examples of errors

---

---
What to Do About Errors?

- Need more features—words aren’t enough!
  - Have you emailed the sender before?
  - Have 1K other people just gotten the same email?
  - Is the sending information consistent?
  - Is the email in ALL CAPS?
  - Do inline URLs point where they say they point?
  - Does the email address you by (your) name?
- Can add these information sources as new variables in the NB model
- Next class we’ll talk about classifiers which let you easily add arbitrary features more easily

Summary

- Bayes rule lets us do diagnostic queries with causal probabilities
- The naïve Bayes assumption makes all effects independent given the cause
- 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