Machine Learning

- Up until now: how to reason in a model and how to make optimal decisions

- Machine learning: how to acquire 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)

This slide deck courtesy of Dan Klein at UC Berkeley
Example: Spam Filter

- Input: 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
  - ...

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.
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
  - ...

??
Other Classification Tasks

- In classification, we predict labels $y$ (classes) for inputs $x$

- 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!
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
Bayes Nets for Classification

- One method of classification:
  - Use a probabilistic model!
  - Features are observed random variables $F_i$
  - $Y$ 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

\[ P(m|s, f) \text{ direct estimate} \]

\[ P(m|s, f) = \frac{P(s, f|m)P(m)}{P(s, f)} \]

\[ P(m|s, f) = \frac{P(s|m)P(f|m)P(m)}{P(s, f)} \text{ Bayes estimate (no assumptions)} \]

\[ + \]

\[ \begin{aligned}
P(+m, s, f) &= P(s|+m)P(f|+m)P(+m) \\
P(-m, s, f) &= P(s|-m)P(f|-m)P(-m)
\end{aligned} \]
A Digit Recognizer

- Input: pixel grids

- Output: a digit 0-9
Naïve Bayes for Digits

- **Simple version:**
  - One feature $F_{ij}$ for each grid position $<i,j>$
  - Possible 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.
    \[ \rightarrow (F_{0,0} = 0 \ F_{0,1} = 0 \ F_{0,2} = 1 \ F_{0,3} = 1 \ F_{0,4} = 0 \ldots F_{15,15} = 0) \]
  - 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?
General Naïve Bayes

- A general *naive Bayes* model:

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

- We only specify how each feature depends on the class
- Total number of parameters is *linear* in $n$
Inference for Naïve Bayes

- **Goal:** compute posterior over causes
  - Step 1: get joint probability of causes and evidence

\[
P(Y, f_1 \ldots f_n) = \begin{bmatrix}
P(y_1, f_1 \ldots f_n) \\
P(y_2, f_1 \ldots f_n) \\
\vdots \\
P(y_k, f_1 \ldots f_n)
\end{bmatrix} \rightarrow \begin{bmatrix}
P(f_1) \prod_i P(f_i | c_1) \\
P(f_2) \prod_i P(f_i | c_2) \\
\vdots \\
P(f_k) \prod_i P(f_i | c_k)
\end{bmatrix}
\]

- Step 2: get probability of evidence
- Step 3: renormalize

\[
P(Y|f_1 \ldots f_n) = \frac{P(f_1 \ldots f_n)}{P(f_1 \ldots f_n)}
\]
General Naïve Bayes

What do we need in order to use naïve Bayes?

- Inference (you know this part)
  - Start with a bunch of conditionals, $P(Y)$ 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: we’ll look at this now
Examples: CPTs

\[
P(Y)
\]

\[
\begin{array}{|c|c|}
\hline
1 & 0.1 \\
2 & 0.1 \\
3 & 0.1 \\
4 & 0.1 \\
5 & 0.1 \\
6 & 0.1 \\
7 & 0.1 \\
8 & 0.1 \\
9 & 0.1 \\
0 & 0.1 \\
\hline
\end{array}
\]

\[
P(F_{3,1} = on|Y) \quad P(F_{5,5} = on|Y)
\]

\[
\begin{array}{|c|c|}
\hline
1 & 0.01 \\
2 & 0.05 \\
3 & 0.05 \\
4 & 0.30 \\
5 & 0.80 \\
6 & 0.90 \\
7 & 0.05 \\
8 & 0.60 \\
9 & 0.50 \\
0 & 0.80 \\
\hline
\end{array}
\]

\[
\begin{array}{|c|c|}
\hline
1 & 0.05 \\
2 & 0.01 \\
3 & 0.90 \\
4 & 0.80 \\
5 & 0.90 \\
6 & 0.90 \\
7 & 0.25 \\
8 & 0.85 \\
9 & 0.60 \\
0 & 0.80 \\
\hline
\end{array}
\]
Parameter Estimation

- Estimating distribution of random variables like X or X | Y

**Empirically:** use training data
- For each outcome x, look at the *empirical rate* of that value:

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

- This is the estimate that maximizes the *likelihood of the data*

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

**Elicitation:** ask a human!
- Usually need domain experts, and sophisticated ways of eliciting probabilities (e.g. betting games)
- Trouble calibrating
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

---

**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:**
  - Predict unknown class label (spam vs. ham)
  - Assume evidence features (e.g. the words) are independent
  - Warning: subtly different assumptions than before!

- **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 \(P(F|Y)\)
  - In a bag-of-words model
    - Each position is identically distributed
    - All positions share the same conditional probs \(P(W|C)\)
    - Why make this assumption?

*Word at position \(i\), not \(i^{th}\) word in the dictionary!
Example: Spam Filtering

- Model: \( P(C, W_1 \ldots W_n) = P(C) \prod_i P(W_i|C) \)

- What are the parameters?

| \( P(C) \) | \( P(W|\text{spam}) \) | \( P(W|\text{ham}) \) |
|---|---|---|
| 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: 0.0108 |
| | a: 0.0086 | from: 0.0107 |
| | with: 0.0080 | and: 0.0105 |
| | from: 0.0075 | a: 0.0100 |
| | ... | ... |

- Where do these tables come from?
## Spam Example

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

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

- Posteriors determined by *relative* probabilities (odds ratios):

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

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

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

- Posteriors:
  - 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
Estimation: Smoothing

- Maximum likelihood estimates:

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

- 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

\[ P_{ML}(r) = \frac{1}{3} \]