Last Time
- Learning Gaussians
- Naïve Bayes

Today
- Gaussians Naïve Bayes
- Logistic Regression

Text Classification
Bag of Words Representation

Bayesian Learning
Use Bayes rule:
\[ P(Y \mid X) = \frac{P(X \mid Y) P(Y)}{P(X)} \]
Or equivalently:
\[ P(Y \mid X) \propto P(X \mid Y) P(Y) \]

Naïve Bayes
- Naïve Bayes assumption:
  - Features are independent given class:
    \[ P(X_1, X_2 \mid Y) = P(X_1 \mid X_2, Y) P(X_2 \mid Y) \]
  - More generally:
    \[ P(X_1 \ldots X_n \mid Y) = \prod P(X_i \mid Y) \]
- How many parameters now?
  - Suppose X is composed of n binary features

The Distributions We Love

<table>
<thead>
<tr>
<th></th>
<th>Discrete</th>
<th>Continuous</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Event</td>
<td>Bernoulli</td>
<td>k Values</td>
</tr>
<tr>
<td>Sequence</td>
<td>Binomial</td>
<td>Multinomial</td>
</tr>
<tr>
<td>(N trials)</td>
<td>Beta</td>
<td>Dirichlet</td>
</tr>
<tr>
<td>Conjugate Prior</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
NB with Bag of Words for Text Classification

- **Learning phase:**
  - Prior $P(Y_m)$
  - Count how many documents from topic $m$ / total # docs
  - $P(X | Y_m)$
    - Let $B_m$ be a bag of words formed from all the docs in topic $m$
    - Let $\#(i, B_m)$ be the number of times word $i$ is in bag $B$
    - $P(X | Y_m) = (\#(i, B_m)+1) / (W+\sum\#(i, B_m))$ where $W$=#unique words

- **Test phase:**
  - For each document
    - Use naive Bayes decision rule

$$h_NB(x) = \arg \max_y P(y)^{\text{LengthDoc}} \prod_{i=1} P(x_i | y)$$

### Probabilities: Important Detail!

- $P(\text{spam} | X_1 \ldots X_n) = \prod P(\text{spam} | X_i)$

**Any more potential problems here?**

- We are multiplying lots of small numbers
  - Danger of underflow!
    - $0.557 = 7 \times 10^{-18}$

- Solution? Use logs and add!
  - $p_1 \times p_2 = e^{\log(p_1)+\log(p_2)}$
  - Always keep in log form

### Twenty News Groups results

Given 2009 training documents from each group
Learn to classify new documents according to which newsgroup it came from

- comp.graphics
- comp.os.ms-windows.misc
- comp.sys.mac.hardware
- comp.windows.x
- rec.autos
- rec.sport.baseball
- rec.sport.hockey
- sci.crypt
- sci.med
- sci.space
- soc.religion.christian

Naive Bayes: 96% classification accuracy

### Learning curve for Twenty News Groups

**Easy to Implement**

- But...

- If you do... it probably won’t work...

### Naive Bayes Posterior Probabilities

- Classification results of naive Bayes
  - i.e. the class with maximum posterior probability...
  - Usually fairly accurate (?!?!)

- However, due to the inadequacy of the conditional independence assumption...
  - Actual posterior probability estimates not accurate.
  - Output probabilities generally very close to 0 or 1.
Bayesian Learning
What if Features are Continuous?

Eg., Character Recognition: $X_i$ is $i^{th}$ pixel

Prior

Posterior

Data Likelihood

$P(Y | X) \propto P(X | Y) P(Y)$

$P(X_i = x | Y = y_k) = \mathcal{N}(\mu_{ik}, \sigma_{ik})$

$N(\mu_{ik}, \sigma_{ik}) = \frac{1}{\sigma_{ik}\sqrt{2\pi}} e^{-\frac{(x-\mu_{ik})^2}{2\sigma_{ik}^2}}$

Gaussian Naïve Bayes

Sometimes Assume Variance

- is independent of $Y$ (i.e., $\sigma_{ik}$),
- or independent of $X_i$ (i.e., $\sigma_i$)
- or both (i.e., $\sigma$)

$P(Y | X) \propto P(X | Y) P(Y)$

$P(X_i = x | Y = y_k) = \mathcal{N}(\mu_{ik}, \sigma_{ik})$

$N(\mu_{ik}, \sigma_{ik}) = \frac{1}{\sigma_{ik}\sqrt{2\pi}} e^{-\frac{(x-\mu_{ik})^2}{2\sigma_{ik}^2}}$

Learning Gaussian Parameters

Maximum Likelihood Estimates:

- Mean:
  \[ \hat{\mu}_{MLE} = \frac{1}{N} \sum_{i=1}^{N} x_i \]

- Variance:
  \[ \hat{\sigma}^2_{MLE} = \frac{1}{N} \sum_{i=1}^{N} (x_i - \hat{\mu})^2 \]

Maximum Likelihood Estimates:

- Mean:
  \[ \hat{\mu}_{MLE} = \frac{1}{\sum_j \delta(Y_j = y_k)} \sum_j X_j \delta(Y_j = y_k) \]

- Variance:
  \[ \hat{\sigma}^2_{MLE} = \frac{1}{\sum_j \delta(Y_j = y_k)} \sum_j (X_j - \hat{\mu}_{MLE})^2 \delta(Y_j = y_k) \]
Example: GNB for classifying mental states

-1 mm resolution
-2 images per sec.
15,000 voxels/image
non-invasive, safe
measures Blood Oxygen Level Dependent (BOLD) response

Brain scans can track activation with precision and sensitivity

Typical impulse response

Gaussian Naive Bayes: Learned $\mu_{\text{voxel,word}}$
P(BrainActivity $|$ WordCategory = \{People,Animal\})

Pairwise classification accuracy: 85%

People words

Animal words

What You Need to Know about Naïve Bayes

- Optimal Decision using Bayes Classifier
- Naïve Bayes Classifier
  - What’s the assumption
  - Why we use it
  - How do we learn it
- Text Classification
  - Bag of words model
- Gaussian NB
  - Features still conditionally independent
  - Features have Gaussian distribution given class

What’s (supervised) learning more formally

- Given:
  - Dataset: Instances $\{(x_i, t(x_i))\}_{i=1}^{N}$
    - e.g., $(x_i, t(x_i)) = (\text{GPA}=3.9, \text{IQ}=120, \text{MLscore}=99, 150K)$
  - Hypothesis space: $H$
    - e.g., polynomials of degree 8
  - Loss function: measures quality of hypothesis $h \in H$
    - e.g., squared error for regression
- Obtain:
  - Learning algorithm: obtain $h \in H$ that minimizes loss function
    - e.g., using closed form solution if available
    - Or greedy search if not
    - Want to minimize prediction error, but can only minimize error in dataset
Types of (supervised) learning problems, revisited

- **Decision Trees**, e.g.,
  - dataset: (votes; party)
  - hypothesis space:
  - Loss function:

- **NB Classification**, e.g.,
  - dataset: (brain image; verb | noun)
  - hypothesis space:
  - Loss function:

- **Density estimation**, e.g.,
  - dataset: (grades)
  - hypothesis space:
  - Loss function:

Learning is (simply) function approximation!

- The general (supervised) learning problem:
  - Given some data (including features), hypothesis space, loss function
  - Learning is no magic!
  - Simply trying to find a function that fits the data

- **Regression**
- **Density estimation**
- **Classification**

(Not surprisingly) Seemly different problem, very similar solutions...

---

What you need to know about supervised learning

- Learning is function approximation
- What functions are being optimized?

---

Generative vs. Discriminative Classifiers

- **Want to Learn**: h:X → Y
  - X - features
  - Y - target classes
- **Bayes optimal classifier** – P(Y|X)
- **Generative classifier**, e.g., Naive Bayes:
  - Assume some functional form for P(X|Y), P(Y)
  - Estimate parameters of P(X|Y), P(Y) directly from training data
  - Use Bayes rule to calculate P(Y|X = x)
  - This is a ‘generative’ model
    - Indirect computation of P(Y|X) through Bayes rule
    - As a result, can also generate a sample of the data, P(X) = ∑y P(y) P(X|y)
- **Discriminative classifiers**, e.g., Logistic Regression:
  - Assume some functional form for P(Y|X)
  - Estimate parameters of P(Y|X) directly from training data
  - This is the ‘discrimination’ model
    - Directly learn P(Y|X)
    - But cannot obtain a sample of the data, because P(X) is not available

---

Logistic Regression

Learn P(Y|X) directly!

- Assume a particular functional form
- Not differentiable...

![Logistic Regression Diagram](image)

![Logistic Regression Graph](image)
Logistic Function in n Dimensions

\[ P(Y = 1|X) = \frac{1}{1 + e^{w_0 + \sum_{i=1}^{n} w_i X_i}} \]

Sigmoid applied to a linear function of the data:

Features can be discrete or continuous!

Very convenient!

\[ P(Y = 1|X = < X_1, \ldots, X_n >) = \frac{1}{1 + e^{w_0 + \sum_{i=1}^{n} w_i X_i}} \]
implies

\[ P(Y = 0|X = < X_1, \ldots, X_n >) = \frac{e^{w_0 + \sum_{i=1}^{n} w_i X_i}}{1 + e^{w_0 + \sum_{i=1}^{n} w_i X_i}} \]
implies

\[ \frac{P(Y = 0|X)}{P(Y = 1|X)} = e^{w_0 + \sum_{i=1}^{n} w_i X_i} \]
implies

\[ \ln \frac{P(Y = 0|X)}{P(Y = 1|X)} = w_0 + \sum_{i=1}^{n} w_i X_i \]

linear classification rule!

Expressing Conditional Log Likelihood

\[ l(w) = \sum_j \ln p(y_j|x^j, w) \]

\[ P(Y = 0|X, w) = \frac{1}{1 + e^{w_0 + \sum_{i=1}^{n} w_i X_i}} \]

\[ P(Y = 1|X, w) = \frac{e^{w_0 + \sum_{i=1}^{n} w_i X_i}}{1 + e^{w_0 + \sum_{i=1}^{n} w_i X_i}} \]

\[ l(w) = \sum_j y_j \ln p(y_j = 1|x^j, w) + (1 - y_j) \ln (1 - p(y_j = 1|x^j, w)) \]

Maximizing Conditional Log Likelihood

\[ l(w) = \ln \prod_j p(y_j|x^j, w) = \sum_j y_j(w_0 + \sum_i w_i x_i) - \ln(1 + e^{w_0 + \sum_i w_i x_i}) \]

Good news: \( l(w) \) is concave function of \( w \) no locally optimal solutions

Bad news: no closed-form solution to maximize \( l(w) \)

Good news: concave functions easy to optimize
Optimizing concave function – Gradient ascent

• Conditional likelihood for Logistic Regression is concave! Find optimum with gradient ascent

Gradient: $\nabla \ell(w) = \left[ \frac{\partial \ell(w)}{\partial w_0}, \ldots, \frac{\partial \ell(w)}{\partial w_n} \right]$

Update rule: $\Delta w = \eta \nabla \ell(w)$

$w_i^{(t+1)} = w_i^{(t)} + \eta \frac{\partial \ell(w)}{\partial w_i}$

• Gradient ascent is simplest of optimization approaches
  – e.g., Conjugate gradient ascent much better (see reading)

Maximize Conditional Log Likelihood: Gradient ascent

$\ell(w) = \sum_j \left[ \frac{y_j w_{j0}}{\sum_i w_{j0}} - \ln(1 + \exp(\sum_i w_{j0} x_i)) \right]$

That’s all M(C)LE. How about MAP?

$p(w \mid Y, X) \propto P(Y \mid X, w) p(w)$

• One common approach is to define priors on $w$
  – Normal distribution, zero mean, identity covariance
  – “Pushes” parameters towards zero
• Corresponds to Regularization
  – Helps avoid very large weights and overfitting
  – More on this later in the semester

• MAP estimate

$w^* = \arg \max_w \left\{ p(w) \prod_{j=1}^N P(y_j \mid x_j, w) \right\}$

M(C)AP as Regularization

$\ln \left[ p(w) \prod_{j=1}^N P(y_j \mid x_j, w) \right] = \frac{1}{2} \sum_{j=1}^N \frac{x_j^2}{2 \sigma^2}$

Penalizes high weights, also applicable in linear regression

Large parameters $\rightarrow$ Overfitting

• If data is linearly separable, weights go to infinity
• Leads to overfitting:

  • Penalizing high weights can prevent overfitting...
    – again, more on this later in the semester
Gradient of M(C)AP

\[
\frac{\partial}{\partial w_i} \ln \left( p(w) \prod_{j=1}^{N} P(y^j | x^j, w) \right)
= \prod_{j=1}^{N} \frac{1}{n_j} \frac{1}{\sqrt{2\pi} \sigma_{ij}} e^{-\frac{x_j^2}{2\sigma_{ij}^2}}
\]

MLE vs MAP

\[ w^* = \arg \max_{w} \ln \left( \prod_{j=1}^{N} P(y^j | x^j, w) \right) \]

\[ w^* = \arg \max_{w} \ln \left( p(w) \prod_{j=1}^{N} P(y^j | x^j, w) \right) \]

Logistic regression v. Naïve Bayes

- Consider learning \( f: X \rightarrow Y \), where
  - \( X \) is a vector of real-valued features, \( <X_1 \ldots X_n> \)
  - \( Y \) is boolean
- Could use a Gaussian Naïve Bayes classifier
  - assume all \( X_i \) are conditionally independent given \( Y \)
  - model \( P(X_i | Y = y_k) \) as Gaussian \( N(\mu_{ik}, \sigma_i^2) \)
  - model \( P(Y) \) as Bernoulli(\( \theta \), 1 - \( \theta \))
- What does that imply about the form of \( P(Y|X) \)?
  \( P(Y = 1|X = <X_1 \ldots X_n>) = \frac{1}{1 + \exp(w_0 + \sum_{i=1}^{n} w_i X_i)} \)
  Cool!!!!

Derive form for \( P(Y|X) \) for continuous \( X_i \)

\[ P(Y = 1|X) = \frac{P(Y = 1)P(X|Y = 1)}{P(Y = 1)P(X|Y = 1) + P(Y = 0)P(X|Y = 0)} \]

\[ = \frac{1 + \exp(\ln \frac{P(Y = 1)}{P(Y = 0)} + \sum_{i=1}^{n} \ln \frac{P(X_i|Y = 1)}{P(X_i|Y = 0)})}{1 + \exp(\ln \frac{P(Y = 1)}{P(Y = 0)} + \sum_{i=1}^{n} \ln \frac{P(X_i|Y = 1)}{P(X_i|Y = 0)})} \]

Ratio of class-conditional probabilities

\[ \ln \frac{P(X_i|Y = 0)}{P(X_i|Y = 1)} \]

\[ P(X_i = x_i | Y = y_k) = \frac{1}{\sqrt{2\pi} \sigma_{ik}} e^{-\frac{(x_i - \mu_{ik})^2}{2\sigma_{ik}^2}} \]

Derive form for \( P(Y|X) \) for continuous \( X_i \)

\[ P(Y = 1|X) = \frac{P(Y = 1)P(X|Y = 1)}{P(Y = 1)P(X|Y = 1) + P(Y = 0)P(X|Y = 0)} \]

\[ = \frac{1 + \exp(\ln \frac{P(Y = 1)}{P(Y = 0)} + \sum_{i=1}^{n} \ln \frac{P(X_i|Y = 1)}{P(X_i|Y = 0)})}{1 + \exp(\ln \frac{P(Y = 1)}{P(Y = 0)} + \sum_{i=1}^{n} \ln \frac{P(X_i|Y = 1)}{P(X_i|Y = 0)})} \]

\[ P(Y = 1|X) = \frac{1}{1 + \exp(\sum_{i=1}^{n} \ln \frac{P(X_i|Y = 1)}{P(X_i|Y = 0)})} \]
Gaussian Naïve Bayes v. Logistic Regression

Set of Gaussian Naïve Bayes parameters (feature variance independent of class label)

Set of Logistic Regression parameters

- Representation equivalence
  - But only in a special case!!! (GNB with class-independent variances)
- LR makes no assumptions about P(X|Y) in learning!!!
- Loss function!!!
  - Optimize different functions! Obtain different solutions

Naïve Bayes vs Logistic Regression

Consider Y boolean, X continuous, X=<X₁ ... Xₙ>

Number of parameters:
- NB: 4n +1
- LR: n+1

Estimation method:
- NB parameter estimates are uncoupled
- LR parameter estimates are coupled

G. Naïve Bayes vs. Logistic Regression 1
[Ng & Jordan, 2002]

- Generative and Discriminative classifiers
- Asymptotic comparison (# training examples → infinity)
  - when model correct
    - GNB, LR produce identical classifiers
  - when model incorrect
    - LR is less biased – does not assume conditional independence
      - therefore LR expected to outperform GNB

G. Naïve Bayes vs. Logistic Regression 2
[Ng & Jordan, 2002]

- Generative and Discriminative classifiers
- Non-asymptotic analysis
  - convergence rate of parameter estimates, n = # of attributes in X
    - GNB needs O(log n) samples
    - LR needs O(n) samples
  - GNB converges more quickly to its (perhaps less helpful) asymptotic estimates

What you should know about Logistic Regression (LR)

- Gaussian Naïve Bayes with class-independent variances representationally equivalent to LR
  - Solution differs because of objective (loss) function
- In general, NB and LR make different assumptions
  - NB: Features independent given class I assumption on P(X|Y)
  - LR: Functional form of P(Y|X), no assumption on P(X|Y)
- LR is a linear classifier
  - decision rule is a hyperplane
- LR optimized by conditional likelihood
  - no closed-form solution
  - concave I global optimum with gradient ascent
  - Maximum conditional a posteriori corresponds to regularization
- Convergence rates
  - GNB (usually) needs less data
  - LR (usually) gets to better solutions in the limit

Some experiments from UCI data sets