Statistical Learning for Classification

Yi Zhang
University of California, Santa Cruz
Review of Basic Prob. Concepts

- Marginal probability $P(A)$: “the fraction of possible world in which $A$ is true”
  - Examples
    - $A =$ Your paper will be accepted by SIGKDD 2011
    - $A =$ It rains in Santa Cruz
    - $A =$ A document contains the word “mining”

- Joint probability $P(A, B)$

- Conditional probability
  - $P(A|B) = P(A, B)/P(B)$

- Bayes’ rule
  - $P(A|B) = P(B|A)P(A)/P(B)$
Classification as Supervised Statistical Learning

- Supervised learning
  - Given: input and output variables pairs (training data)
    \[\{(x_1,y_1),(x_2,y_2)\ldots(x_N,y_N)\}\]
  - Learning: infer a function \(f(X)\) from the training data
  - Prediction: predict future outcomes \(y = f(x)\)

\[
\text{classification: } R^{|V|} \rightarrow \{0,1\} \quad X \xrightarrow{\text{classifier}} \text{Prediction of category } c
\]

\[
\text{regression: } R^{|V|} \rightarrow R \quad X \xrightarrow{\text{regression}} \text{Prediction of real value output}
\]
How to Learn?

- Given evidence (data) $E$, find hypothesis (model) $h$
- Maximum likelihood (ML) estimation
  \[ h_{ML} = \arg \max_h P(E \mid h_i) \]
- Maximum a posteriori (MAP) estimation
  \[ h_{MAP} = \arg \max_h \frac{P(h \mid E)}{P(E \mid h)P(h)} = \arg \max_h \frac{P(E \mid h)P(h)}{P(E)} \]
Major Steps for Supervised Learning

- Gathering a training set (input objects and corresponding outputs) from human or from other measurements
- Determine the input feature of the learned function (What’s X)
  - Typically, the input object is transformed into a feature vector
  - This step influence the final performance of the system greatly
- Determine the functional form of the learned algorithm
  - Logistic regression? Neural network? Support Vector Machines?
- Determine the corresponding learning algorithm (ML or MAP or else)
- Learn: run the learning algorithm on the gathered training set
  - Optional: adjust the parameter via cross validation
- Test the performance on a test set
Two Statistical Learning Approaches

☐ Generative models

☐ Model the joint probabilistic distribution \( p(c, X) \), and derive the conditional probability

\[
P(c_j \mid X) = \frac{P(X \mid c_j)P(c_j)}{\sum_j P(X \mid c_j)P(c_j)}
\]

☐ Examples: Naive Bayes

☐ Discriminative models

☐ Model the conditional probability \( P(c \mid X) \) directly

☐ Examples: decision tree, neural networks, support vector machines, boosting or bagging, regression (linear, polynomial …), k nearest neighbor
Naïve Bayes for Documents (Multinomial Mixture Models)

- Pick up a class randomly according to a class prior \( P(c_i) \)
- Each document \( X \) in a class \( c \) is generated by a multinomial distribution

\[
P(x | c) = P(x_1 | c)P(x_2 | c)\ldots P(x_T | c)
\]

where

\[
\sum_{k=1}^{\mid V \mid} P(x_k | c) = 1
\]

\[
x = \{x_1, x_2, x_3, \ldots x_j, \ldots x_T \}
\]

\[
x_j \in \{1, 2, \ldots, \mid V \mid\}
\]

\( |V| \) dimensional document space
Naïve Bayes (Multi-Variate Bernoulli Mixture Models)

- Pick up a class randomly according to a class prior $P(c_i)$
- The person runs through the dictionary, deciding whether to include each word $i$ in that document according to the probability $P(X_j=1|C_i)\quad P(X_j=1|C_i)+P(X_j=0|C_i)=1$
- The probability of a document is
  $$P(X, c_i) = P(c_i) \prod_{j=1}^{||V||} P(x_j | c_i)$$
  $$X = \{x_1, x_2, \ldots, x_j, \ldots, x_{||V||}\}, x_j = \{0,1\}$$

$|V|$ dimensional document space
Example of Documents Generation based on Multinomial Mixture Models

\[ p(x_k|c) \]

\[ \text{P(c)} \]

class 1:
Search engine

... search 0.05
google 0.01
yahoo 0.05
... Clinton 0.000001
...

class 2:
Bill Clinton

... Clinton 0.025
Bill 0.01
President 0.009
Lewinsky 0.008
Hillary 0.005
...

(Ponte & Croft 98)
Learning in Naïve Bayes (Multinomial Model)

- \( P(c) = \) the percentage of training example that belong to class \( c \)
- Learning the probability of the \( k \)th feature given the class \( p(x_k \mid c) \):
  - Maximum likelihood estimation
    \[
    P_{MLE}(x_i \mid c) = \frac{\#(x_i, c)}{\sum_{j=1\ldots|V|} \#(x_j, c)}
    \]
  - Maximum a posteriori (MAP) estimation
    \[
    P_{MAP}(x_i \mid c) = \frac{\#(x_i, c) + \mu P(x_i)}{\sum_{j=1\ldots|V|} \#(x_j, c) + \mu}
    \]

Dirichlet Prior: \((\mu P(x_1), \mu P(x_2), \ldots, \mu P(x_{|V|}))\)

Avoid zero probability
Example of ML Learning

\[ p(x_k | c) = \]

100/10000
50/10000
50/10000
70/10000
10/10000

... Google ?
search ?
engine ?
Yahoo ?
...
query ?
...

Google 100
search 50
engine 50
Yahoo 70
algorithm 20
...
query 10
compete 10

(total #words occurrences=10000)

Documents about “search engines”
Binary Classification in Naïve Bayes (Multinomial Model)

- Classification Rule: classify $X$ to class $c$ if:

$$1 < \frac{P(c | X)}{P(\bar{c} | X)} = \frac{P(X | c)P(c) / P(X)}{P(X | \bar{c})P(\bar{c}) / P(X)} = \frac{P(X | c)P(c)}{P(X | \bar{c})P(\bar{c})} = \frac{\prod_{j=1..|V|} P(X_j | c)^{tf_j} P(c)}{\prod_{j=1..|V|} P(X_j | \bar{c})^{tf_j} P(\bar{c})}$$

$$\Leftrightarrow 0 < \sum_{j=1..|V|} \#(X_j) \log \frac{P(X_j | c)}{P(X_j | \bar{c})} + \log \frac{P(c)}{P(\bar{c})} = \left( \log \frac{P(c)}{P(\bar{c})}, \log \frac{P(X_1 | c)}{P(X_1 | \bar{c})}, ..., \log \frac{P(X_{|V|} | c)}{P(X_{|V|} | \bar{c})} \right) \times \begin{pmatrix} 1 \\ \#(x_1) \\ ... \\ \#(X_{|V|}) \end{pmatrix}$$

$$\Leftrightarrow W^T X > 0$$

linear separator
Naïve Bayes Classifier

Pros:
- Clear theoretical foundation
- Relatively effective
- Very simple
- Fast: handles 10,000 attributes easily

Cons
- Wrong assumptions
  - Independence assumptions
  - Multi-Variate Bernoulli or Multinomial models assumption
  - One class per document assumption
- Classification accuracy is usually worse than many other methods, such as logistic regression, linear regression or support vector machines
- Bad probabilistic estimation of $P(c|x)$
Exercise: Naïve Bayes Classifier

A: attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Give Birth</th>
<th>Can Fly</th>
<th>Live in Water</th>
<th>Have Legs</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>human</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>mammals</td>
</tr>
<tr>
<td>python</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>non-mammals</td>
</tr>
<tr>
<td>salmon</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>non-mammals</td>
</tr>
<tr>
<td>whale</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>mammals</td>
</tr>
<tr>
<td>frog</td>
<td>no</td>
<td>no</td>
<td>sometimes</td>
<td>yes</td>
<td>non-mammals</td>
</tr>
<tr>
<td>komodo</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>non-mammals</td>
</tr>
<tr>
<td>bat</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>mammals</td>
</tr>
<tr>
<td>pigeon</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>non-mammals</td>
</tr>
<tr>
<td>cat</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>mammals</td>
</tr>
<tr>
<td>leopard shark</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>non-mammals</td>
</tr>
<tr>
<td>turtle</td>
<td>no</td>
<td>no</td>
<td>sometimes</td>
<td>yes</td>
<td>non-mammals</td>
</tr>
<tr>
<td>penguin</td>
<td>no</td>
<td>no</td>
<td>sometimes</td>
<td>yes</td>
<td>non-mammals</td>
</tr>
<tr>
<td>porcupine</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>mammals</td>
</tr>
<tr>
<td>eel</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>non-mammals</td>
</tr>
<tr>
<td>salamander</td>
<td>no</td>
<td>no</td>
<td>sometimes</td>
<td>yes</td>
<td>non-mammals</td>
</tr>
<tr>
<td>gila monster</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>non-mammals</td>
</tr>
<tr>
<td>platypus</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>mammals</td>
</tr>
<tr>
<td>owl</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>non-mammals</td>
</tr>
<tr>
<td>dolphin</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>mammals</td>
</tr>
<tr>
<td>eagle</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>non-mammals</td>
</tr>
</tbody>
</table>

M: mammals

N: non-mammals

\[
P(A | M) = \frac{6}{7} \times \frac{6}{7} \times \frac{2}{7} = 0.06
\]

\[
P(A | N) = \frac{1}{13} \times \frac{10}{13} \times \frac{3}{13} \times \frac{4}{13} = 0.0042
\]

\[
P(A | M) P(M) = 0.06 \times \frac{7}{20} = 0.021
\]

\[
P(A | N) P(N) = 0.004 \times \frac{13}{20} = 0.0027
\]

\[P(A|M)P(M) > P(A|N)P(N)\]

=> Mammals
Discriminative Models

- Focusing on estimating the decision boundary between classes or $P(y|X)$
- No explicit assumption on how the documents are generated
- For text classification task, usually the decision boundary is a linear separator
  - Assign a document to the positive class if $h(X) = W^T X > 0$
Logistic Regression (Maximum Entropy)

- Modeling the conditional probability \( p(y|X) \) as a logistic function

\[
P(y|X,W) = g(yW^T X) = \frac{1}{1 + e^{-yW^T X}}
\]

input vector \( X \)

Model parameter vector \( W \)

Class label \( \{-1,1\} \)

Logistic function
Learning Logistic Regression Model

- Maximum likelihood estimation

\[ W_{ML} = \arg \max_w \prod_{i=1}^{t} p(y_i | x_i, W) = \arg \max_w \sum_{i=1}^{t} \log( p(y_i | x_i, W)) \]

- Maximum a posteriori (MAP) estimation

\[ W_{MAP} = \arg \max_w \prod_{i=1}^{t} p(y_i | x_i, W)P(W) \]

\[ = \arg \max_w \sum_{i=1}^{t} \log( p(y_i | x_i, W)) + \log( P(W)) \]

Usually a Gaussian prior

Document space (N)

Maximum likelihood estimation

Maximum a posteriori (MAP) estimation
Logistic Regression Classifier

- If the goal is to minimize classification error, classify $X$ to the class if:

$$P(y = \text{yes} \mid X, W) = \frac{1}{1 + e^{-W^T X}} > 0.5 \iff W^T X > 0$$

linear separator
Linear Regression Model

- Modeling the conditional probability $p(y|X)$ as a Normal distribution

$$P(y|X,W) = N(y;W^T X, \sigma^2)$$
Learning Linear Regression Model

- **Maximum likelihood estimation**
  \[ W_{ML} = \arg \max_w \sum_{i=1}^{t} \log(p(y_i \mid x_i, W)) = \arg \max_w - \sum_{i=1}^{t} (y_i - W^T x_i)^2 \]

- **Maximum a posteriori (MAP) estimation**
  \[ W_{MAP} = \arg \max_w \sum_{i=1}^{t} \log(p(y_i \mid x_i, W)) - \log P(W) \]
  \[ = \arg \max_w - \sum_{i=1}^{t} (y_i - W^T x_i)^2 - \lambda(W - U)^2 \]

  - Sum square error on training data
  - Deviation from the prior mean
  - Usual a Gaussian prior of data

 Document space \((N)\)
Linear Least Square Classifier

- If the goal is to minimize classification error, classify a document $X$ to the class if:

$$P(y = yes \mid X, W) = N(y; W^T X, \sigma^2) > 0.5 \iff W^T X > 0$$
The Benefit of Using Prior P(W)

- Controlling model complexity
- Avoiding the pain of zero probability
- Integrating expert/prior knowledge
- Integrating two classification algorithms
- Transfer learning
  - Using one task to help another task
Learning as Empirical Loss Minimization

- Given: input and output variables pairs \( \{(x_1,y_1)(x_2,y_2)\ldots(x_N,y_N)\} \)
- Learning: infer \( W \) from the training data
- How: minimize the loss on training data

\[
\sum_i \text{Loss}(y_i, \hat{y}_i) = \sum_i \text{Loss}(y_i, f(x_i, w))
\]

- Maximize the posterior probability (Logistic regression, linear regression)
- Minimize the mean square error (Least Square Fit)
- Maximizing the margin between two classes (Support Vector Machines)

- Usually no close form solution. Gradient descent algorithms are often used to find the optimal solution \( W^* \)
Loss Function for Different Discriminative Classifiers
Some Empirical Performance

![F1 performance of the classifiers graph]

From Li and Yang
SIGIR03
## Comparison of Generative Models and Discriminative Models

<table>
<thead>
<tr>
<th></th>
<th>Generative Models (Naïve Bayes)</th>
<th>Discriminative Models (LR, LR)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Focus</strong></td>
<td>$P(X, y)$</td>
<td>$P(y</td>
</tr>
<tr>
<td><strong>Training efficiency</strong></td>
<td>😊</td>
<td>😞</td>
</tr>
<tr>
<td><strong>Effectiveness</strong></td>
<td>OK</td>
<td>Good</td>
</tr>
</tbody>
</table>
Improving Simple Classifiers

- **Bagging**: Fit many large trees to bootstrap-resampled versions of the training data, and classify by majority vote. (Variance reduction)

- **Boosting**: Fit many large or small trees to reweighted versions of the training data. Classify by weighted majority vote. (Bias reduction)
MART

- General boosting algorithm that works with a variety of different loss functions. Models include regression, outlier-resistant regression, K-class classification and risk modeling.

- MART uses gradient boosting to build additive tree models, for example, for representing the logits in logistic regression.

- Tree size is a parameter that determines the order of interaction (next slide).

- MART inherits all the good features of trees (variable selection, missing data, mixed predictors), and improves on the weak features, such as prediction performance.
Comparison of Learning Methods

Some characteristics of different learning methods.
Key: ⬤ = good, ⬤ = fair, and ⬤ = poor.

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Neural Nets</th>
<th>SVM</th>
<th>CART</th>
<th>GAM</th>
<th>KNN, kernels</th>
<th>MART</th>
</tr>
</thead>
<tbody>
<tr>
<td>Natural handling of data of “mixed” type</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
</tr>
<tr>
<td>Handling of missing values</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
</tr>
<tr>
<td>Robustness to outliers in input space</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
</tr>
<tr>
<td>Insensitive to monotone transformations of inputs</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
</tr>
<tr>
<td>Computational scalability (large $N$)</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
</tr>
<tr>
<td>Ability to deal with irrelevant inputs</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
</tr>
<tr>
<td>Ability to extract linear combinations of features</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
</tr>
<tr>
<td>Interpretability</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
</tr>
<tr>
<td>Predictive power</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
<td>⬤</td>
</tr>
</tbody>
</table>
Multi-Class Classification

classification: $R^{|V|} \rightarrow \{0,1\}^K$

regression: $R^{|V|} \rightarrow R^K$

$V$: the set of vocabularies

$K$: the number of classes

$X_j \in R^{|V|}$: a document

classification: $R^{|V|} \rightarrow \{0,1\}^K$

regression: $R^{|V|} \rightarrow R^K$

$V$: the set of vocabularies

$K$: the number of classes

$X_j \in R^{|V|}$: a document
Multi-Class Classification Approaches

- Learn one binary classifier for each class
  - Assign a document to all classes that the corresponding classifiers says “yes”; or
  - Assign a document to the “best” class

- Many against many
  - Learn multiple classifiers
  - Each classifier assign a document to a set of classes
  - Assign a document to the class with the biggest votes from classifiers
Other Practical Concerns

- Controlling model complexity
  - Feature selection, smoothing (coefficient shrinkage): Ridge regression, lasso regression

- Over fitting (cross-validation, leave one out)

- Cost sensitive learning
  - Classify a ham as spam is more costly than the other way around

- Unbalanced samples and rare classes: 0.01% positive vs. 99.99% negative samples

- Biased samples
  - User only provides feedback on documents she reads
  - While she may not not read randomly

- Noisy label