Classification Problem
Naïve Bayes and Logistic Regression

CSE512 – Machine Learning, Spring 2018, Stony Brook University

Instructor: Minh Hoai Nguyen (minhhoai@cs.stonybrook.edu)

Date: 12 Feb 2018

Classification Problem

Learn to predict class label *Y* from features **X:**

- Y = {1, 2, …, k}

Suppose we know P(Y|X), for all Y = 1,…,k. We can use Bayes classifier.

Bayes classifier: $y^*(x) = \operatorname{argmax}_y P(Y = y|X = x)$

Bayes Classifier is Optimal

Theorem: The Bayes classifier h_{bayes} is optimal

$$h_{bayes}(x) = y^*(x) = \operatorname*{argmax}_y P(Y = y|X = x)$$

It means:

$$error_{true}(h_{bayes}) \leq error_{true}(h)\ \forall h(x)$$

Proof:

$$\begin{aligned} error_{true}(h) &= \int_x \int_y \delta(h(x) \neq y) p(x, y) dy dx \\ &= \int_x \left(\sum_{y=1}^{k} \delta(h(x) \neq y) p(y|x) \right) dx \\ &\vdots \end{aligned}$$

Why not using the Bayes classifier?

If the Bayes classifier is optimal, why don't we use it for classification?

Answer: it's very hard to learn the Bayes classifier.

It's even very hard just to represent the distribution P(Y|X) exactly.

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)} \propto P(X|Y)P(Y)$$

Data

GRE	GPA	Letter	Job Offered
High	High	Good	Yes
Low	Low	Bad	No
Low	Medium	Good	Yes
Low	Medium	Good	No

Representing the Conditional Probabilities

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)} \propto P(X|Y)P(Y)$$

Suppose there are:

- k classes
- d attributes
- each attribute has *m* possible values

To represent P(Y), we need:

To represent P(X|Y), we need:

Learning the Bayes classifier is hard

- It takes exponentially many parameters to represent the conditional probabilities exactly
- It requires exponentially many training examples!

Conditional Independence

A is conditionally independent of B given C if:
The probability distribution of A is independent of the probability distribution of B given the value of C.

$$P(A|B,C) = P(A|C)$$

Which is short-hand for:

$$P(A = a|B = b, C = c) = P(A = a|C = c)\ \forall(a, b, c)$$

E.g., P(Thunder|Rain, Lightning) = P(Thunder|Lightning)

Equivalent definition of conditional independence:

$$P(A, B|C) = P(A|C)P(B|C)$$

Conditionally Independent Features

Predicting Job Offered (J) from:

- GRE score
- GPA
- Letter of Reference (LOR)

$$P(J|GRE, GPA, LOR) \propto P(J)P(GRE|J)P(GPA|J)P(LOR|J)$$

The Naïve Bayes Assumption

Given X = (X_1, X_2,..., X_d)

The Naïve Bayes assumption: features are independent given the class

$$P(X_i, X_j|Y) = P(X_i|Y)P(X_j|Y)$$

We can show now:

$$P(X_1, \cdots \mathbf{X}_d|Y) = \prod_{i=1}^{d} P(X_i|Y)$$

How many parameters do we need to represent this distribution?

The Naïve Bayes Classifier

Assumption: features are conditionally independent given class label

Classifier's parameters:

- Prior P(Y)
- Conditional likelihood: $P(X_i|Y)$

Decision rule:

$$y^*(X) = \operatorname*{argmax}_y P(Y = y) \prod_{i=1}^{d} P(X_i|Y = y)$$

If the assumption holds, NB is the optimal classifier

MLE for NB

Count and build a probability tables

$$P(A=a|B=b) = Count(A=a,B=b)/Count(B=b)$$

Prior probability

$$P(Y=y) = Count(Y=y)/nTrain$$

P(JobOffer = Yes)	P(JobOffer = No)
0.3	0.7

Conditional probability

$$P(X_i = x_i | \ Y=y)$$

	P(JobOffer = Yes)	P(JobOffer = No)
Letter=Good	0.9	0.5
Letter=Bad	0.1	0.5

	P(JobOffer = Yes)	P(JobOffer = No)
GPA=High	0.7	0.3
GPA=Med	0.2	0.3
GPA=Low	0.1	0.4

Notes about NB

Normally, the assumption does not hold

$$P(X_1, \cdots \mathbf{X}_d|Y) \neq \prod_{i=1}^{d} P(X_i|Y)$$

However, NB often performs well in practice

"All models are wrong, but some are useful" – George Box

NB for Text Classification

Email classification: classes = {Spam, NotSpam}

Article classification: classes = {Politics, Sports, Fashion, …}

Webpage classification: classes = {Professor, Student, Course …}

Input: text from a document

NB for Text Classification

Features for a document:

+ $X = (X_1, X_2, \ldots, X_d)$
+ d: the length of the document
+ X_i is the word at position i^{th}. The domain is the vocabulary.

NB assumption is important: The exact probability distribution $P(X|Y)$ is huge because the document might be very long (d can be > 10000)

NB assumption:

+ $P(X_i = x_i | Y = \text{topic})$ is the probability of observing word x_i in a document of topic y
+ E.g., $P(X_i = \text{weapon} \mid Y = \text{politics})$, $P(X_i = \text{trendy} \mid Y = \text{politics})$, $P(X_i = \text{weapon} \mid Y = \text{fashion})$,

Bag-of-Words assumption

In a document, the positions of the words do not matter

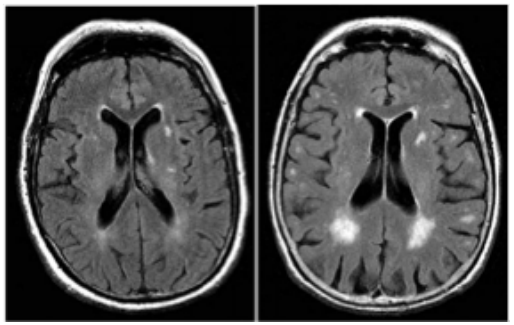
+ It's a Bag of Words
+ Not a (Ordered) Sequence of Words.
+ $P(X_1 = \text{weapon} \mid \text{politics}) = P(X_{42} = \text{weapon} \mid \text{politics})$
+ The assumption doesn't hold, but it works well in practice

Classifying 20 news groups: 89% accuracy

NB for Continuous Variables

What if we have continuous variables?

+ E.g., MRI brain classification:
pixel values are from 0 to 255



Gaussian Naïve Bayes (GNB):

$$P(X_i = x|Y = k) = \frac{1}{\sigma_{ik}\sqrt{2\pi}} e^{-\frac{(x-\mu_{ik})^2}{2\sigma_{ik}^2}}$$

Sometimes we assume the variance:

+ is independent of class: $\sigma_{ik} = \sigma_i \ \forall k$
+ is independent of features: $\sigma_{ik} = \sigma_k \ \forall i$
+ or both: $\sigma_{ik} = \sigma \ \forall i, k$

MLE for Gaussian NB

For each class Y = *k*, find all training examples of class k

$\hat{\mu}_{ik}$ is the mean of X_i in all training examples of class k

$\hat{\sigma}_{ik}$ is the variance of X_i in all training examples of class k

Things You Need to Know

- Bayes Classifier is Optimal
- Learning Bayes Classifier is hard or impossible
- Naïve Bayes classifier:
  - Learn the Bayes Classifier with an assumption
  - What's the assumption
  - How to learn the NB parameters
- Bag-of-words model
- Gaussian NB

NB is a Generative Classifier

Our task: learning a function h: **X** -> Y

Bayes Optimal Classifier: P(Y|**X**)

Naïve Bayes:

+ Use chain rule to compute P(Y|**X**) from P(**X**|Y) and P(Y)
+ Assume some functional form of P(**X**|Y) and P(Y)
+ Use training data to estimate P(**X**|Y) and P(Y)

Naïve Bayes is a generative classifier:

+ We learn P(**X**|Y).
+ Given this conditional probability. We can "generate" **X** given Y.

But we are ultimately interested in P(Y|**X**):

+ This is indirectly estimated through P(**X**|Y) and P(Y) using chain rule

Question: can we estimate P(Y|**X**) directly?

Discriminative Classifier

Yes, we can learn P(Y|**X**) directly. This is called Discriminative approach.

Approach of a Discriminative Classifier:

+ Assume some functional form of P(Y|**X**)
+ Estimate the parameters of the function using training data
+ We learn P(Y|**X**) directly, which is what we care about.
+ We don't care about P(**X**|Y).
+ But, we cannot generate or sample **X** from Y as in the case of a generative classifier.

Logistic Regression

Logistic Regression is a discriminative classifier:

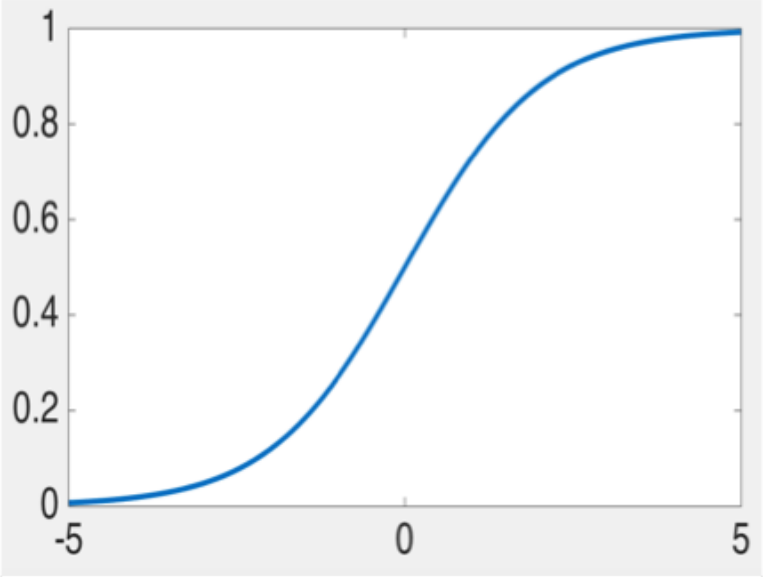
+ Learn P(Y|**X**) directly
+ Assume a functional form of P(Y|**X**)

Assume a probability as Sigmoid function

$$P(Y = 1|\mathbf{X}) = sigmoid(\boldsymbol{\theta}^T \mathbf{X})$$

$$sigmoid(z) = \frac{1}{1 + \exp(-z)}$$

$$P(Y = 0|\mathbf{X}) = 1 - P(Y = 1|\mathbf{X})$$

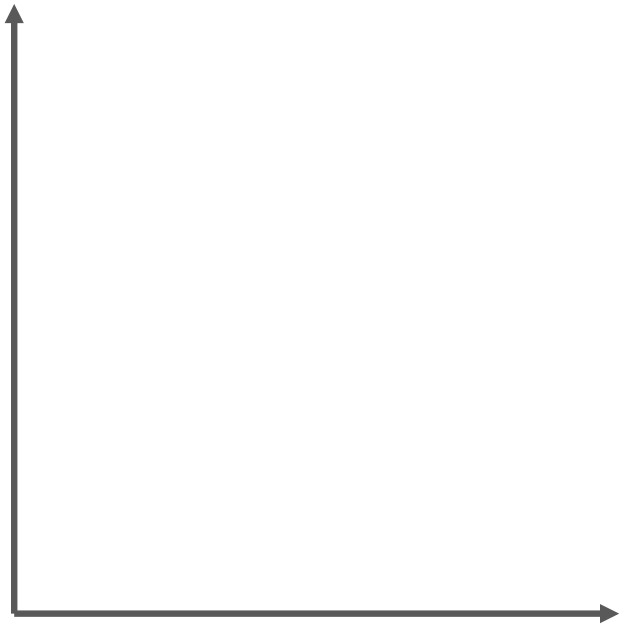


Understand the Sigmoid

Logistic Regression – Linear Classifier

Classification decision: $P(Y = 1|\mathbf{X}) \geq 0.5$

This happens when $\boldsymbol{\theta}^T \mathbf{X} \geq 0$



Learning the Parameters

Maximize the Conditional Log-likelihood

$$L(\boldsymbol{\theta}) = \sum_j \log(P(Y^j|\mathbf{X}^j, \boldsymbol{\theta}))$$

$$L(\boldsymbol{\theta}) = \sum_j Y^j \log(P(Y=1|\mathbf{X}^j, \boldsymbol{\theta})) + (1-Y^j) \log(P(Y=0|\mathbf{X}^j, \boldsymbol{\theta}))$$

Learning the Parameters

Maximize the Conditional Log-likelihood

$$L(\boldsymbol{\theta}) = \sum_j \log(P(Y^j|\mathbf{X}^j, \boldsymbol{\theta}))$$

$$L(\boldsymbol{\theta}) = \sum_j Y^j \log(P(Y=1|\mathbf{X}^j, \boldsymbol{\theta})) + (1-Y^j)\log(P(Y=0|\mathbf{X}^j, \boldsymbol{\theta}))$$

$$= \sum_j [Y^j(\boldsymbol{\theta}^T\mathbf{X}^j) - \log(1+\exp(\boldsymbol{\theta}^T\mathbf{X}^j))]$$

Bad News: No closed-form solution

Good News: The function is concave => Easy to optimize

Optimization with Gradient Ascent

Iterative optimization:

$$\boldsymbol{\theta}^{(t+1)} := \boldsymbol{\theta}^t + \eta \left. \frac{\partial L}{\partial \boldsymbol{\theta}} \right|_{\boldsymbol{\theta}=\boldsymbol{\theta}^t}$$

$$\frac{\partial L}{\partial \boldsymbol{\theta}} =$$

Logistic Regression – Summary

Objective: maximize the log-likelihood

$$L(\boldsymbol{\theta}) = \sum_j [Y^j(\boldsymbol{\theta}^T \mathbf{X}^j) - \log(1 + \exp(\boldsymbol{\theta}^T \mathbf{X}^j))]$$

Optimization with gradient ascent

$$\boldsymbol{\theta}^{(t+1)} := \boldsymbol{\theta}^t + \eta \left. \frac{\partial L}{\partial \boldsymbol{\theta}} \right|_{\boldsymbol{\theta}=\boldsymbol{\theta}^t}$$

The derivative

$$\frac{\partial L}{\partial \boldsymbol{\theta}} = \sum_j [Y^j \mathbf{X}^j - \frac{\exp(\boldsymbol{\theta}^T \mathbf{X}^j)}{1 + \exp(\boldsymbol{\theta}^T \mathbf{X}^j)} \mathbf{X}^j]$$

$$= \sum_j [Y^j - P(Y = 1 | \mathbf{X}^j)] \mathbf{X}^j$$

Understand the gradient update

Optimization with gradient ascent $\boldsymbol{\theta}^{(t+1)} := \boldsymbol{\theta}^t + \eta \left.\frac{\partial L}{\partial \boldsymbol{\theta}}\right|_{\boldsymbol{\theta}=\boldsymbol{\theta}^t}$

$$\frac{\partial L}{\partial \boldsymbol{\theta}} = \sum_j [Y^j - P(Y=1|\mathbf{X}^j)]\mathbf{X}^j$$

Case 1: $Y^j = 1$

+ If $P(Y=1|\mathbf{X}^j)$ is big, $\mathbf{X}^j$ induces a weak pull

+ If $P(Y=1|\mathbf{X}^j)$ is small, $\mathbf{X}^j$ induces a strong pull

Case 2: $Y^j = 0$

+ If $P(Y=1|\mathbf{X}^j)$ is big, $\mathbf{X}^j$ induces a strong push

+ If $P(Y=1|\mathbf{X}^j)$ is small, $\mathbf{X}^j$ induces a weak push

That's MLE. How's about MAP?

A common approach:

- Assume normal distribution, zero mean, identity covariance
- Push parameters towards zero

This corresponds to Regularization

- Helps avoid large weights and overfitting

MAP estimate:

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \sum_j \log(P(Y^j|\mathbf{X}^j, \boldsymbol{\theta})) + \log(P(\boldsymbol{\theta}))$$

Logistic Regression for *k* classes

$$P(Y = 1|\mathbf{X}) = \frac{\exp(\boldsymbol{\theta}_1^T \mathbf{X})}{1 + \sum_{i=1}^{k-1} \exp(\boldsymbol{\theta}_i^T \mathbf{X})}$$

$$\vdots$$

$$P(Y = k-1|\mathbf{X}) = \frac{\exp(\boldsymbol{\theta}_{k-1}^T \mathbf{X})}{1 + \sum_{i=1}^{k-1} \exp(\boldsymbol{\theta}_i^T \mathbf{X})}$$

$$P(Y = k|\mathbf{X}) = \frac{1}{1 + \sum_{i=1}^{k-1} \exp(\boldsymbol{\theta}_i^T \mathbf{X})}$$

Logistic Regression vs. Naïve Bayes

Consider the binary classification task with real-values features:

+ Y is binary
+ X_i is real-value

Could use Gaussian Naïve Bayes classifier

+ Assume all X_i are conditionally independent given Y
+ Assume $P(X_i|Y=k) = \mathcal{N}(\mu_{ik}, \sigma_i)$
+ Note the variance is same for all classes
+ Model $P(Y) \sim Bernoulli(\gamma, 1-\gamma)$

Then P(Y|**X**) is:

$$P(Y=1|\mathbf{X}=<X_1,\cdots,X_d>) = \frac{1}{1+\exp(-\sum_i \theta_i X_i - \theta_{d+1})}$$

Let's Derive P(Y=1|**X**)

$$P(Y=1|X) = \frac{P(Y=1)P(\mathbf{X}|Y=1)}{P(Y=1)P(\mathbf{X}|Y=1)+P(Y=0)P(\mathbf{X}|Y=0)}$$

$$= \frac{1}{1+\frac{P(Y=0)P(\mathbf{X}|Y=0)}{P(Y=1)P(\mathbf{X}|Y=1)}}$$

$$= \frac{1}{1+\exp(\log(\frac{P(Y=0)P(\mathbf{X}|Y=0)}{P(Y=1)P(\mathbf{X}|Y=1)}))}$$

$$= \frac{1}{1+\exp(\log(\frac{1-\gamma}{\gamma})+\sum_i \log(\frac{P(X_i|Y=0)}{P(X_i|Y=1)}))}$$

$$\log(\frac{P(X_i|Y=0)}{P(X_i|Y=1)}) =$$

Logistic Regression and Gaussian NB

Equivalent representation for a special case

Logistic Regression

+ Assume conditional probability has as a Sigmoid functional form.
+ A set of logistic regression parameters

Special Gaussian NB

+ Assume features are independent given class label
+ Assume feature variance independent of class label
+ A set of Gaussian parameters

Differences:

+ Logistic Regression makes no assumption about $P(\mathbf{X}|Y)$ in learning
+ The loss functions are different!!!
+ Will lead to different solutions

Loss Functions:

Data Likelihood vs. Conditional Data Likelihood

Naive Bayes maximizes the likelihood
While a Logistic Regression maximizes the conditional likelihood

Generative loss (Naïve Bayes) function: $\max_{\boldsymbol{\theta}}. P(\mathcal{D}_Y, \mathcal{D}_X|\boldsymbol{\theta})$

$$\log(P(\mathcal{D}_Y, \mathcal{D}_X|\boldsymbol{\theta})) = \sum_j \log(P(\mathbf{X}^j, Y^j|\boldsymbol{\theta}))$$

$$= \sum_j \log(P(\mathbf{Y}^j|\mathbf{X}^j, \boldsymbol{\theta})) + \sum_j \log(P(\mathbf{X}^j|\boldsymbol{\theta}))$$

Discriminative loss (Logistic Regression) function: $\max_{\boldsymbol{\theta}}. P(\mathcal{D}_Y|\mathcal{D}_X, \boldsymbol{\theta})$

$$\log(P(\mathcal{D}_Y|\mathcal{D}_X, \boldsymbol{\theta})) = \sum_j \log(P(\mathbf{Y}^j|\mathbf{X}^j, \boldsymbol{\theta}))$$

- Focuses the all learning effort on P(Y|**X**), which is all we need for classification

Naïve Bayes vs Logistic Regression

Consider binary classification with *d* continuous features

	Naïve Bayes	Logistic Regression
Number of parameters	4d + 1	d + 1
Loss function in learning	Data likelihood	Conditional Data Likelihood
Parameter Estimation	uncoupled	coupled

Generative vs Discriminative Classifiers

(Ng & Jordan, NIPS 2002)

Consider Asymptotic Comparison (# training examples -> infinity)

When assumptions correct:

+ GNB, LR produce identical classifiers

When assumptions incorrect:

+ LR is less biased – does not assume conditional independence

+ LR is expected to be better than GNB (as training examples -> infinity)

Generative vs Discriminative Classifiers

(Ng & Jordan, NIPS 2002)

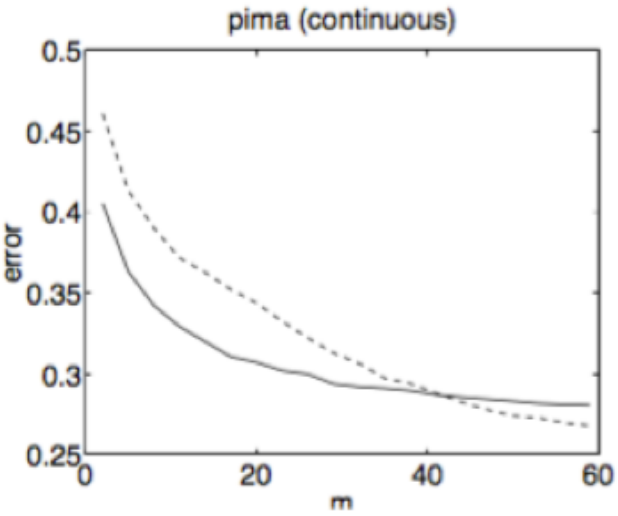
Consider Non-Asymptotic Comparison

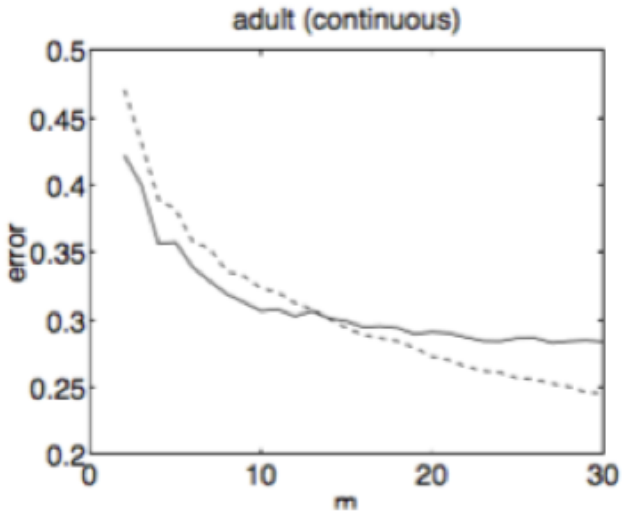
Convergence rate of parameter estimates

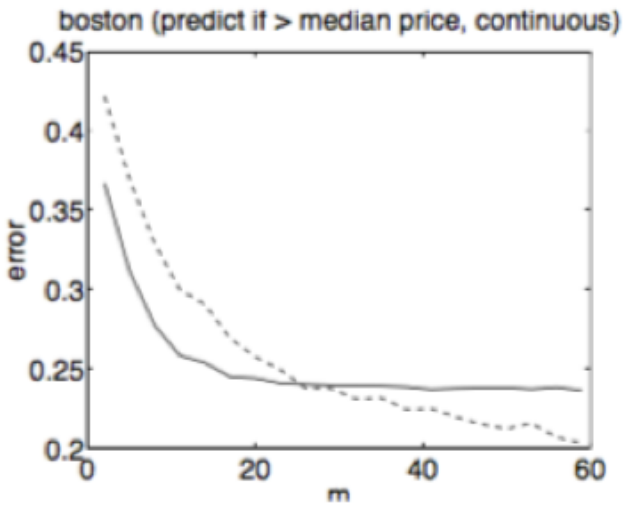
+ Size of training data to get closes to the solution with infinite data
+ d: number of attributes of **X**
+ GNB needs O(log(d)) samples
+ LR needs O(d) samples

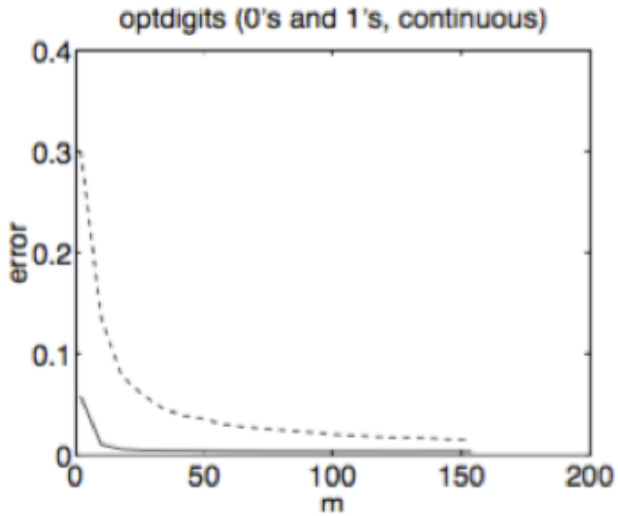
GNB converges more quickly to its asymptotic estimates

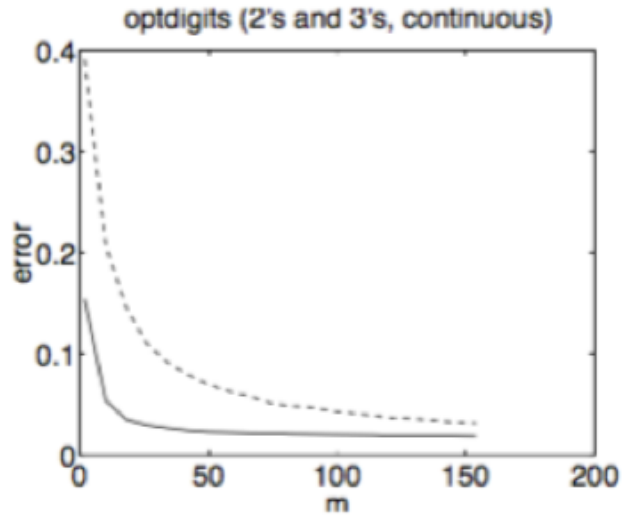
Some Experiments from UCI datasets

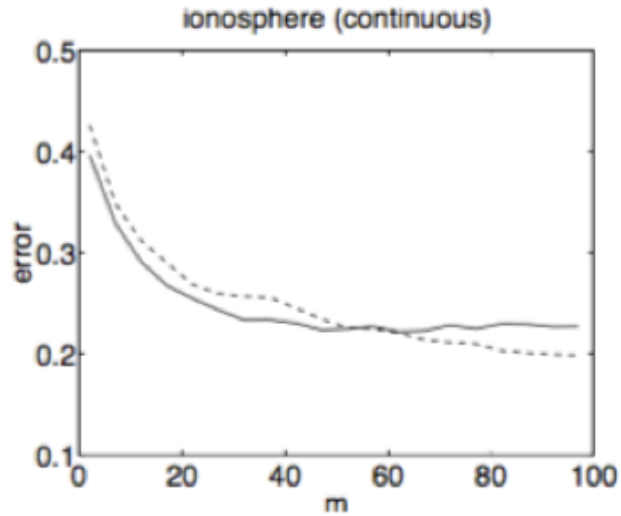












Logistic Regression: Dashed line. NB: Dash line.

Things to know about Logistic Regression

LR is a linear classifier. The decision is a hyperplane.

LR optimizes conditional likelihood

+ No closed-form solution
+ Concave -> optimize with gradient ascent

In general, LR and NB makes different assumptions.

+ NB: features are independent given class => assume $P(X|Y)$
+ LR: direct estimate of $P(Y|X)$, no assumption about $P(X|Y)$
+ NB optimizes the data likelihood
+ LR optimizes the conditional data likelihood

Convergence Rate

+ NB usually needs less data
+ LR usually gets to better solutions given more data