Preconditioning an Artificial Neural Network Using Naive Bayes

Nayyar A. Zaidi, Francois Petitjean, Geoffrey I. Webb

MONASH University
Information Technology

Introduction

- Maximum likelihood estimates of naive Bayes probabilities can be used to greatly speed-up logistic regression
- This talk demonstrates that this speed-up can also be attained for Artificial Neural Networks
- Talk outline
  - Introduction (2 minutes)
  - Proposed Approach (6 minutes)
  - Experimental Analysis (6 minutes)
  - Future Research, Q & A (3 minutes)

Contributions of the Paper

- We show that:

1. Preconditioning based on naive Bayes is applicable and equally useful for Artificial Neural Networks (ANN) as it is for Logistic Regression (LR)
2. Optimizing MSE objective function leads to lower bias than optimzing CLL, this leads to lower 0-1 Loss and RMSE on big datasets

Logistic Regression

- One of the state-of-the-art classifier
- Maximizes the Conditional Log-Likelihood (NLL)

$$\mathrm{CLL}(\beta) \quad = \quad \sum_{i=1}^{N} \log \mathrm{P}_{LR}(y^{(i)}|\mathbf{x}^{(i)}) \qquad (1)$$

- If constrained to categorical attributes and multi-class problems, it leads to:

$$\mathrm{P}_{LR}(y \mid \mathbf{x}) \quad = \quad \frac{\exp\left(\beta_y + \sum_{i=1}^{a} \beta_{y,i,x_i}\right)}{\sum_{c\in\Omega_Y} \exp\left(\beta_c + \sum_{j=1}^{a} \beta_{c,j,x_j}\right)} \qquad (2)$$

- and

$$\exp\left(\beta_y + \sum_{i=1}^{a} \beta_{y,i,x_i} - \log \sum_{c\in\Omega_Y} \exp\left(\beta_c + \sum_{j=1}^{a} \beta_{c,j,x_j}\right)\right) \quad (3)$$

Naive Bayes and Weighted Naive Bayes

- Naive Bayes can be written as:

$$\mathrm{P}_{NB}(y \mid \mathbf{x}) = \frac{\mathrm{P}(y) \prod_{i=1}^{a} \mathrm{P}(x_i \mid y)}{\sum_{c \in \Omega_Y} \mathrm{P}(c) \prod_{j=1}^{a} \mathrm{P}(x_j \mid c)} \tag{4}$$

- Adding weights in naive Bayes:

$$\mathrm{P}_{W}(y \mid \mathbf{x}) = \frac{\mathrm{P}(y)^{w_y} \prod_{i=1}^{a} \mathrm{P}(x_i \mid y)^{w_{y,i,x_i}}}{\sum_{c \in \Omega_Y} \mathrm{P}(c)^{w_c} \prod_{j=1}^{a} \mathrm{P}(x_j \mid c)^{w_{c,j,x_j}}} \tag{5}$$

$$= \exp\Big(w_y \log \mathrm{P}(y) + \sum_{i=1}^{a} w_{y,i,x_i} \log \mathrm{P}(x_i \mid y) - \log \sum_{c \in \Omega_Y} \exp\Big(w_c \log \mathrm{P}(c) + \sum_{j=1}^{a} w_{c,j,x_j} \log \mathrm{P}(x_j \mid c) \Big)\Big). \tag{6}$$

WANBIA-C

- LR

$$\exp\left(\beta_y + \sum_{i=1}^{a} \beta_{y,i,x_i} - \log \sum_{c \in \Omega_Y} \exp\left(\beta_c + \sum_{j=1}^{a} \beta_{c,j,x_j}\right)\right)$$

- Weighted Naive Bayes

$$\exp\Big(w_y \log \mathrm{P}(y) + \textstyle\sum_{i=1}^{a} w_{y,i,x_i} \log \mathrm{P}(x_i|y) -$$

$$\log \textstyle\sum_{c \in \Omega_Y} \exp\Big(w_c \log \mathrm{P}(c) + \textstyle\sum_{j=1}^{a} w_{c,j,x_j} \log \mathrm{P}(x_j \mid c)\Big)\Big)$$

WANBIA-C

- LR

$$\exp\left(\beta_y + \sum_{i=1}^{a} \beta_{y,i,x_i} - \log \sum_{c \in \Omega_Y} \exp\left(\beta_c + \sum_{j=1}^{a} \beta_{c,j,x_j}\right)\right)$$

- Weighted Naive Bayes

$$\exp\Big(w_y \log \mathrm{P}(y) + \textstyle\sum_{i=1}^{a} w_{y,i,x_i} \log \mathrm{P}(x_i \mid y) -$$
$$\log \textstyle\sum_{c \in \Omega_Y} \exp\Big(w_c \log \mathrm{P}(c) + \textstyle\sum_{j=1}^{a} w_{c,j,x_j} \log \mathrm{P}(x_j \mid c)\Big)\Big)$$

WANBIA-C

- LR

$$\exp\left(\beta_y + \sum_{i=1}^{a} \beta_{y,i,x_i} - \log \sum_{c \in \Omega_Y} \exp\left(\beta_c + \sum_{j=1}^{a} \beta_{c,j,x_j}\right)\right)$$

- Weighted Naive Bayes

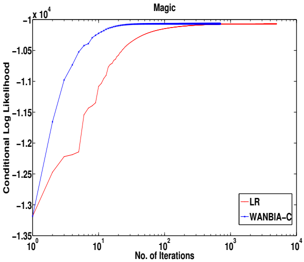
$$\exp\Big(w_y \log \mathrm{P}(y) + \textstyle\sum_{i=1}^{a} w_{y,i,x_i} \log \mathrm{P}(x_i \mid y) -$$
$$\log \textstyle\sum_{c \in \Omega_Y} \exp\Big(w_c \log \mathrm{P}(c) + \textstyle\sum_{j=1}^{a} w_{c,j,x_j} \log \mathrm{P}(x_j \mid c)\Big)\Big)$$

- $\beta_c \propto w_c \log \mathrm{P}(c)$ and $\beta_{c,i,x_i} \propto w_{c,i,x_i} \log \mathrm{P}(x_i \mid c)$
- WANBIA-C: Proposed in [1] shows an equivalence between LR and weighted naive Bayes
- For sake of clarity - we denote it by: $\mathrm{WANBIA}^{\mathrm{C}}_{\mathrm{CLL}}$

WANBIA-C

- View 1: Learn weights by optimizing CLL to alleviate naive Bayes independence assumption
- View 2: $\mathrm{WANBIA}^{\mathrm{C}}_{\mathrm{CLL}}$ uses generative estimates of the probabilities to speed-up the convergence
- View 3: Way of combining generative and discriminative models

- $\mathrm{WANBIA}^{\mathrm{C}}_{\mathrm{CLL}}$ and LR generate equivalent models
- But have different convergence profiles



Artificial Neural Networks (LR)

- Minimizes MSE Objective Function instead of NLL
- We begin by writing an objective function:

$$\mathrm{MSE}(\beta) = \frac{1}{2}\sum_{i=1}^{N}\sum_{c=1}^{C}\left(\delta(y=c) - \mathrm{P}(c|\mathbf{x}^{(i)})\right)^2$$

- where

$$\mathrm{P}(c \mid \mathbf{x}) \quad = \quad \frac{\exp\left(\beta_c + \sum_{i=1}^{a} \beta_{c,i,x_i}\right)}{\sum_{c' \in \Omega_Y} \exp\left(\beta_{c'} + \sum_{j=1}^{a} \beta_{c',j,x_j}\right)}$$

Artificial Neural Networks (WANBIA-C)

- Minimizes MSE Objective Function instead of NLL
- We begin by writing an objective function:

$$\mathrm{MSE}(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}\sum_{c=1}^{C}\left(\delta(y=c) - \mathrm{P}(c|\mathbf{x}^{(i)})\right)^2$$

- where

$$\mathrm{P}(c \mid \mathbf{x}) \quad = \quad \frac{\mathrm{P}(y)^{w_y}\prod_{i=1}^{a}\mathrm{P}(x_i \mid y)^{w_{y,i,x_i}}}{\sum_{c\in\Omega_Y}\mathrm{P}(c)^{w_c}\prod_{j=1}^{a}\mathrm{P}(x_j \mid c)^{w_{c,j,x_j}}}$$

Proposed Method – $\mathrm{WANBIA}^{\mathrm{C}}_{\mathrm{MSE}}$

- Step 1:
  - Calculate class-probabilities as $\mathrm{P}(y) = \pi_y = \frac{\#_y + m/C}{N+m}$
  - Calculate other probabilities as $\mathrm{P}(x_i \mid y) = \theta_{x_i|c} = \frac{\#_{x_i,y} + m/|x_i|}{\#_y + m}$
- Step 2:
  - Optimize MSE based on weighted naive Bayes
  - Use gradient-based iterative optimization algorithm
  - Calculate the gradient:

$$\frac{\partial \mathrm{MSE}(\mathbf{w})}{\partial w_{k,i,x_i}} = -\sum_{i=1}^{N} \sum_{c}^{C} (\delta(y=c) - \mathrm{P}(c|\mathbf{x})) \frac{\partial \mathrm{P}(c|\mathbf{x})}{\partial w_{k,i,x_i}},$$

  - where

$$\frac{\partial \mathrm{P}(c|\mathbf{x})}{\partial w_{k,i,x_i}} = \mathrm{P}(c|\mathbf{x}) (\delta(c=k) - \mathrm{P}(k|\mathbf{x})) \, \delta(x_i) \log \theta_{x_i|k},$$

- Use L-BFGS to get parameter vector w

$\text{WANBIA}^{\text{C}}_{\text{MSE}}$ vs. ANN

- Gradient of parameters can be defined as:

$$\frac{\partial \text{MSE}(\mathbf{w})}{\partial w_{k,i,x_i}} = -\sum_{i=1}^{N}\sum_{c}^{C}\left(\delta(y=c) - \hat{\text{P}}(c|\mathbf{x})\right)\text{P}(y|\mathbf{x}) \\ (\delta(y=k) - \text{P}(k|\mathbf{x}))\log\theta_{x_i|k}\delta(x_i)$$

- Note for ANN, we have:

$$\frac{\partial \text{MSE}(\mathbf{w})}{\partial \beta_{k,i,x_i}} = -\sum_{i=1}^{N}\sum_{c}^{C}(\delta(y=c) - \text{P}(c|\mathbf{x}))\,\text{P}(y|\mathbf{x}) \\ (\delta(y=k) - \text{P}(k|\mathbf{x}))\,\delta(x_i)$$

- $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ has the effect of re-scaling the gradients of ANN

$$\frac{\partial \text{MSE}(\mathbf{w})}{\partial w_{k,i,x_i}} = \frac{\partial \text{MSE}(\beta)}{\partial \beta_{k,i,x_i}}\log\theta_{x_i|k}, \quad \frac{\partial \text{MSE}(\mathbf{w})}{\partial w_k} = \frac{\partial \text{MSE}(\beta)}{\partial \beta_k}\log\pi_k$$

Experimental Results

- 73 standard UCI datasets
- Algorithms evaluated in terms of bias, variance, 0-1 Loss and RMSE
- 40 datasets with < 1000 instances
- 21 between 1000 and 10000 instances
- 12 datasets with > 10000 instances
- Datasets are divided into two categories *All* and *Big*
- MDL discretization is used to discretize numeric attributes
- L-BFGS solver is used

MSE vs. CLL

	$\text{WANBIA}^{C}_{\text{MSE}}$ **vs.** $\text{WANBIA}^{C}_{\text{CLL}}$		**ANN vs. LR**	
	W-D-L	p	W-D-L	p
		All Datasets		
Bias	39/15/18	**0.007**	36/14/22	0.086
Variance	21/8/42	0.011	26/7/38	0.168
0-1 Loss	33/12/27	0.519	34/9/29	0.614
RMSE	30/5/37	0.463	28/4/40	0.181
		Big Datasets		
0-1 Loss	10/1/1	**0.011**	8/2/2	0.109
RMSE	8/1/3	0.226	8/0/4	0.387

Table: Win-Draw-Loss: $\text{WANBIA}^{C}_{\text{MSE}}$ vs. $\text{WANBIA}^{C}_{\text{CLL}}$ and ANN vs. LR. p is two-tail binomial sign test. Results are significant if $p \leq 0.05$.

$\mathrm{WANBIA}^{\mathrm{C}}_{\mathrm{MSE}}$ vs. ANN

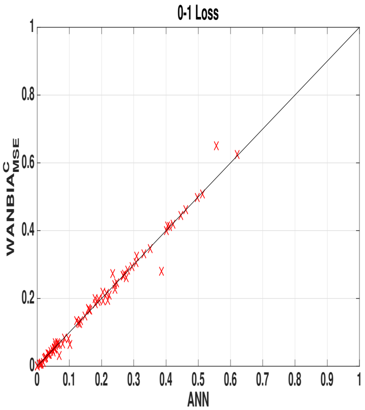


Figure: Comparative scatter of 0-1 Loss of ANN and $\mathrm{WANBIA}^{\mathrm{C}}_{\mathrm{MSE}}$ on *All* (Left) and *Big* (Right) datasets.

$\mathrm{WANBIA}^{\mathrm{C}}_{\mathrm{MSE}}$ vs. ANN

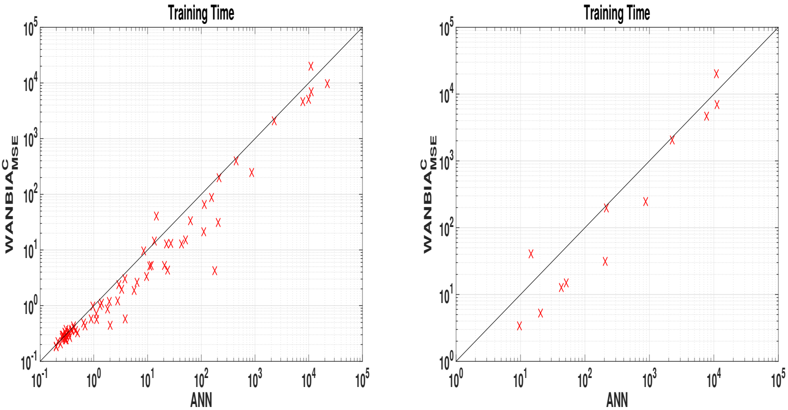


Figure: Comparative scatter of training time of ANN and $\mathrm{WANBIA}^{\mathrm{C}}_{\mathrm{MSE}}$ on *All* (Left) and *Big* (Right) datasets.

$\text{WANBIA}^{\text{C}}_{\text{MSE}}$ vs. ANN

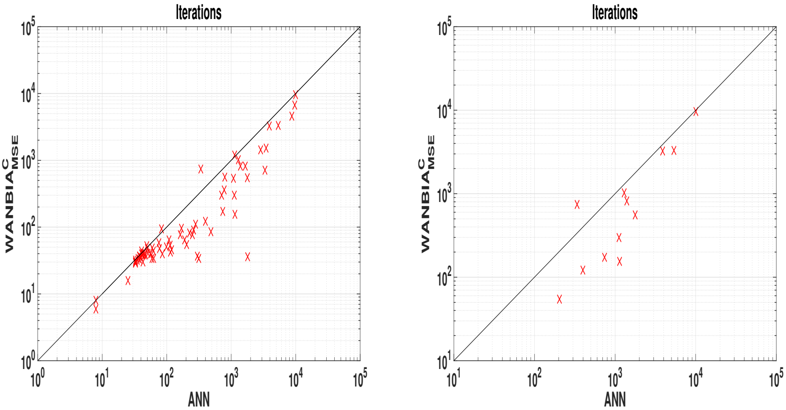


Figure: Comparative scatter of number of iterations to convergence of ANN and $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ on *All* (Left) and *Big* (Right) datasets.

Convergence Curves

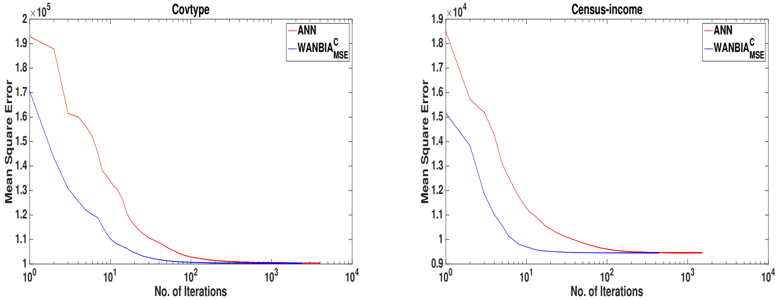


Figure: Comparative convergence profiles of ANN and $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ on `Covtype` (Left) and `Census-income` (Right) datasets.

Convergence Curves

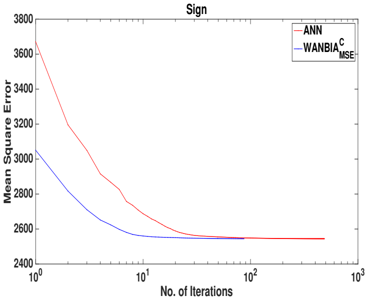
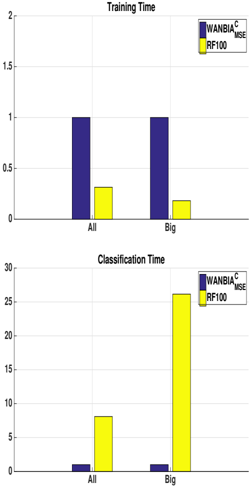


Figure: Comparative convergence profiles of ANN and $\mathrm{WANBIA}^{\mathrm{C}}_{\mathrm{MSE}}$ on `Sign` (Left) and `Shuttle` (Right) datasets.

$\mathrm{WANBIA}^{C}_{\mathrm{MSE}}$ vs. Random Forest



	$\mathrm{WANBIA}^{C}_{\mathrm{MSE}}$ vs. RF100	
	W-D-L	p
	All Datasets	
Bias	41/5/26	0.086
Variance	32/2/38	0.550
0-1 Loss	30/2/40	0.282
RMSE	27/0/45	0.044
	Big Datasets	
0-1 Loss	5/0/7	0.774
RMSE	5/0/7	0.774

Table: Win-Draw-Loss: $\mathrm{WANBIA}^{C}_{\mathrm{MSE}}$ vs. Random Forest. p is two-tail binomial sign test. Results are significant if $p \leq 0.05$.

Conclusion and Future Work

- Simple (naive Bayes based) preconditioning can speed-up convergence of ANN
- The proposed $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ approach has the desirable property of asymptoting to optimum much quicker than ANN
- We are investigating:

1. Why naive Bayes estimates are such a good pre-conditioner?
2. An out-of-core Stochastic Gradient Descent (SGD) optimization
3. $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ for ANN with hidden layers
4. Applicability of WANBIA-C style pre-conditioning to other objective functions

- Q & A
- Offline Discussions
  - @nayyar_zaidi
  - nayyar.zaidi@monash.edu
  - nayyar_zaidi
  - `http://users.monash.edu.au/~nzaidi`
- For further discussions, contact:

MONASH University
Information Technology

N. A. Zaidi, M. J. Carman, J. Cerquides, and G. I. Webb, "Naive-Bayes inspired effective pre-conditioners for speeding-up logistic regression," in *IEEE International Conference on Data Mining*, pp. 1097–1102, 2014.

N. A. Zaidi, J. Cerquides, M. J. Carman, and G. I. Webb, "Alleviating naive Bayes attribute independence assumption by attribute weighting," *Journal of Machine Learning Research*, vol. 14, pp. 1947–1988, 2013.