Preconditioning an Artificial Neural Network Using Naive Bayes

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

Faculty of Information Technology, Monash University, VIC 3800, Australia.
{nayyar.zaidi,francois.petitjean,geoff.webb}@monash.edu

Abstract. Logistic Regression (LR) is a workhorse of the statistics community and a state-of-the-art machine learning classifier. It learns a linear model from inputs to outputs trained by optimizing the Conditional Log-Likelihood (CLL) of the data. Recently, it has been shown that preconditioning LR using a Naive Bayes (NB) model speeds up LR learning many-fold. One can, however, train a linear model by optimizing the mean-square-error (MSE) instead of CLL. This leads to an Artificial Neural Network (ANN) with no hidden layer. In this work, we study the effect of NB preconditioning on such an ANN classifier. Optimizing MSE instead of CLL may lead to a lower bias classifier and hence result in better performance on big datasets. We show that this NB preconditioning can speed-up convergence significantly. We also show that optimizing a linear model with MSE leads to a lower bias classifier than optimizing with CLL. We also compare the performance to state-of-the-art classifier Random Forest.

Key words: Logistic Regression, Preconditioning, Conditional Log-Likelihood, Mean-square-error, WANBIA-C, Artificial Neural Networks.

1 Introduction

Logistic Regression (LR) is a state-of-the-art machine learning classifier and is widely used by statisticians [1, 2]. It has been shown recently that LR training converges more rapidly when each axis is scaled by the log of the naive Bayes estimates of the conditional probabilities [3, 4]. Such rescaling leads to an alternative parameterization with both naive Bayes parameters (learned generatively) and LR parameters (learned discriminatively). The resulting parameterization of LR is known as $\text{WANBIA}^{\text{C}}_{\text{CLL}}$ and has been shown to be effective for both online and batch gradient based optimization for logistic regression[1]. LR optimizes the conditional log-likelihood (CLL) of the data given the model. We conjecture that optimizing the mean square error (MSE) should lead to more accurate (low-biased) models, especially for bigger datasets because, it is mainly the bias that contributes to the error on the bigger datasets [5, 6]. Note, that

[1] Note, we add CLL as subscript to WANBIA-C to show explicitly the objective function that it optimizes.

a linear model optimizing MSE is an Artificial Neural Network (ANN) with no hidden layer (the structure constitutes only an input layer with multiple nodes and an output layer with multiple nodes).

This paper investigates the performance of linear classification models that optimize MSE relative to those that optimize CLL and whether NB regularization is as effective with the MSE objective function as it is with CLL. One can view $\text{WANBIA}^{\text{C}}_{\text{CLL}}$ from two perspectives.

1. From the NB perspective, the parameters learned with discriminative training are only alleviating NB's independence assumption. It is irrelevant whether the weights are optimized by the CLL or by the MSE objective function.
2. From the LR perspective, $\text{WANBIA}^{\text{C}}_{\text{CLL}}$ introduces NB weights that precondition the search space. For CLL, which is a convex objective function, this leads to faster convergence. A natural question is: will the same trend hold for other objective functions which are not convex, such as MSE?

The contributions of this paper are two-fold:

1. We show that NB preconditioning is applicable and equally useful for learning a linear classification model optimizing the MSE objective function.
2. Optimizing MSE leads to a lower bias classifier than LR optimizing CLL. This leads to lower 0-1 loss and RMSE on big datasets.

The rest of this paper is organized as follows. We discuss LR and $\text{WANBIA}^{\text{C}}_{\text{CLL}}$ in section 2. We will derive NB preconditioning of a linear classification model optimizing MSE in section 3. Empirical analysis is given in Section 4. We conclude in Section 5 with some pointers to future work.

2 $\text{WANBIA}^{\text{C}}_{\text{CLL}}$

Let us start by explaining $\text{WANBIA}^{\text{C}}_{\text{CLL}}$. Typically, an LR optimizes the following objective function:

$$\text{CLL}(\beta) = \sum_{i=1}^{N} \log \text{P}_{LR}(y^{(i)}|\mathbf{x}^{(i)}), \tag{1}$$

where N is the number of data points. Note, we are constraining ourselves to categorical attributes and multi-class problems only. We write P_{LR} for categorical features and multiple classes as:

$$\begin{aligned}\text{P}_{LR}(y|\mathbf{x}) &= \frac{\exp(\beta_y + \sum_{i=1}^{a} \beta_{y,i,x_i})}{\sum_{c\in\Omega_Y} \exp\Big(\beta_c + \sum_{j=1}^{a} \beta_{c,j,x_j}\Big)}, \\ &= \exp\Big(\beta_y + \sum_{i=1}^{a} \beta_{y,i,x_i} - \log \sum_{c\in\Omega_Y} \exp\Big(\beta_c + \sum_{j=1}^{a} \beta_{c,j,x_j}\Big)\Big), \end{aligned} \tag{2}$$

where a is the number of attributes and β_{y,i,x_i} denotes the parameter associated with class y, and attribute i taking value x_i. On the other hand, naive Bayes is defined as:

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

One can add weights to NB to alleviate the attribute independence assumption, resulting in the $\mathrm{WANBIA}^{\mathrm{C}}_{\mathrm{CLL}}$ formulation, that can be written as:

$$\begin{aligned}\mathrm{P}_W(y\,|\,\mathbf{x}) &= \frac{\mathrm{P}(y)^{w_y}\prod_{i=1}^{a}\mathrm{P}(x_i\,|\,y)^{w_{y,i,x_i}}}{\sum_{c\in\Omega_Y}\mathrm{P}(c)^{w_c}\prod_{j=1}^{a}\mathrm{P}(x_j\,|\,c)^{w_{c,j,x_j}}}\\ &= \exp\Big(w_y\log\mathrm{P}(y)+\sum_{i=1}^{a}w_{y,i,x_i}\log\mathrm{P}(x_i\,|\,y)-\\ &\log\sum_{c\in\Omega_Y}\exp\Big(w_c\log\mathrm{P}(c)+\sum_{j=1}^{a}w_{c,j,x_j}\log\mathrm{P}(x_j|\,c)\Big)\Big). \qquad (3)\end{aligned}$$

When conditional log likelihood (CLL) is maximized for LR and weighted NB using Equation 2 and 3 respectively, we get an equivalence such that $\beta_c \propto w_c\log\mathrm{P}(c)$ and $\beta_{c,i,x_i} \propto w_{c,i,x_i}\log\mathrm{P}(x_i\,|\,c)$. Thus, $\mathrm{WANBIA}^{\mathrm{C}}_{\mathrm{CLL}}$ and LR generate equivalent models. While it might seem less efficient to use $\mathrm{WANBIA}^{\mathrm{C}}_{\mathrm{CLL}}$ which has twice the number of parameters of LR, the probability estimates are learned very efficiently using maximum likelihood estimation, and provide useful information about the classification task that in practice serve to effectively precondition the search for the parameterization of weights to maximize conditional log likelihood.

3 Method

In this section, we will derive a variant of $\mathrm{WANBIA}^{\mathrm{C}}_{\mathrm{CLL}}$ that is optimized to minimize MSE. But before doing that, we will first derive a variant of LR using the MSE objective function — an ANN with no hidden layer.

ANN Instead of optimizing the objective function in Equation 1, one can optimize the following MSE objective function: $\mathrm{MSE}(\beta) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{C}\sum_{c=1}^{C}(\mathrm{P}(y|\mathbf{x}^{(i)})-\hat{\mathrm{P}}(c|\mathbf{x}^{(i)}))^2$, where y is the true label and $C = |\,\Omega_Y\,|$. Let us simplify the above equation slightly:

$$\mathrm{MSE}(\beta) = \frac{1}{2}\sum_{i=1}^{N}\sum_{c=1}^{C}\Big(\delta(y=c)-\mathrm{P}(c|\mathbf{x}^{(i)})\Big)^2, \qquad (4)$$

where $\delta(.)$ is an indicator function which is 1 if its input parameter condition holds and 0 otherwise. Note that unlike the CLL objective function in Equation 1, the above objective function (Equation 4) is not convex. It is likely that one will

be stuck in local minimum and, therefore, local minimum avoidance techniques may be required. We will show in Section 4 that in practice one can obtain good results with simple gradient descent based (such as quasi-Newton) optimization algorithms without requiring specific mechanisms to avoid local minima.

In the following, we will drop the superscript (j) for simplicity. Optimizing Equation 4 requires us to compute its derivative with respect the parameters β. We have the following:

$$\frac{\partial \text{MSE}(\beta)}{\partial \beta_{k,i,x_i}} = -\sum_{i=1}^{N}\sum_{c}^{C}(\delta(y=c) - \text{P}(c|\mathbf{x}))\frac{\partial \text{P}(c|\mathbf{x})}{\partial \beta_{k,i,x_i}}, \tag{5}$$

where,

$$\begin{aligned}
\frac{\partial \text{P}(c|\mathbf{x})}{\partial \beta_{k,i,x_i}} &= \frac{\partial}{\partial \beta_{k,i,x_i}}\left(\frac{\exp(\beta_c + \sum \beta_{c,i,x_i})}{\sum_{c'}\exp(\beta_{c'} + \sum \beta_{c',i,x_i})}\right), \\
&= \frac{\partial}{\partial \beta_{k,i,x_i}}\exp\left((\beta_c + \sum \beta_{c,i,x_i}) - \log(\sum_{c'}\exp(\beta_{c'} + \sum \beta_{c',i,x_i}))\right), \\
&= \text{P}(c|\mathbf{x})\left(\delta(c=k)\delta(x_i) - \left(\frac{\beta_k + \sum \beta_{k,i,x_i}}{\sum_{c'}\exp(\beta_{c'} + \sum \beta_{c',i,x_i})}\right)\delta(x_i)\right), \\
&= \text{P}(c|\mathbf{x})\Big(\delta(c=k)\delta(x_i) - \text{P}(k|\mathbf{x})\delta(x_i)\Big), \\
&= \text{P}(c|\mathbf{x})(\delta(c=k) - \text{P}(k|\mathbf{x}))\delta(x_i),
\end{aligned} \tag{6}$$

where, $\delta(x_i)$ is an indicator function if value of x_i is same to the value with which we are differentiating. Plugging in Equation 5, we get:

$$\frac{\partial \text{MSE}(\mathbf{w})}{\partial \beta_{k,i,x_i}} = -\sum_{i=1}^{N}\sum_{c}^{C}(\delta(y=c) - \text{P}(c|\mathbf{x}))\text{P}(c|\mathbf{x})(\delta(c=k) - \text{P}(k|\mathbf{x}))\delta(x_i). \tag{7}$$

Note, the gradients with respect to the class parameters can be calculated similarly. The gradients in Equation 7 is the same as optimized by ANN with back-propagation training algorithm. In the following, we will formulate $\text{WANBIA}^{\text{C}}_{\text{CLL}}$ with MSE objective function.

$\text{WANBIA}^{\mathbf{C}}_{\mathbf{MSE}}$ Given Equation 3, assuming a Dirichlet prior, a MAP estimate of $\text{P}(y)$ is π_y which equals: $\frac{\#_y + m/C}{N+m}$, where $\#_y$ is the number of instances in the dataset with class y and N is the total number of instances, and m is the smoothing parameter. We will set $m = 1$ in this work. Similarly, a MAP estimate of $\text{P}(x_i|y)$ is $\theta_{x_i|c}$ which equals: $\frac{\#_{x_i,y} + m/|x_i|}{\#_y + m}$, where $\#_{x_i,y}$ is the number of instances in the dataset with class y and attribute values x_i. Now, we have:

$$\text{P}(y|\mathbf{x}) = \frac{\pi_y^{w_y}\prod_{i=1}^{a}\theta_{x_i|y}^{w_{y,i,x_i}}}{\sum_{c\in\Omega_Y}\pi_c^{w_c}\prod_{j=1}^{a}\theta_{x_i|y}^{w_{c,j,x_j}}}.$$

Using the above equation, let us optimize the MSE objective function by taking gradients with respect to the parameters w. We write:

$$\frac{\partial \text{MSE}(\mathbf{w})}{\partial w_{k,i,x_i}} = -\sum_{i=1}^{N}\sum_{c}^{C}(\delta(y=c) - \text{P}(c|\mathbf{x}))\frac{\partial \text{P}(c|\mathbf{x})}{\partial w_{k,i,x_i}}, \tag{8}$$

where w_{k,i,x_i} denotes parameter associated with attribute i taking value x_i and class attribute k. Let us expand $\frac{\partial \text{P}(c|\mathbf{x})}{\partial w_{k,i,x_i}}$ in the following way:

$$\begin{aligned}
\frac{\partial \text{P}(c|\mathbf{x})}{\partial w_{k,i,x_i}} &= \frac{\partial}{\partial w_{k,i,x_i}} \exp\Big(w_c \log \pi_c + \sum_{i=1}^{a} w_{c,i,x_i} \log \theta_{x_i|c} - \\
&\quad \log \sum_{c' \in \Omega_Y} \exp\Big(w_{c'} \log \pi_{c'} + \sum_{j=1}^{a} w_{c',j,x_j} \log \theta_{x_j|c'}\Big)\Big), \\
&= \text{P}(c|\mathbf{x})\Big(\delta(c=k)\delta(x_i)\log \theta_{x_i|k} - \\
&\quad \frac{\exp(w_k \log \pi_k + \sum_{j=1}^{a} w_{k,j,x_j} \log \theta_{x_j|k})}{\log \sum_{c' \in \Omega_Y} \exp\Big(w_{c'} \log \pi_{c'} + \sum_{j=1}^{a} w_{c',j,x_j} \log \theta_{x_j|c'}\Big)} \delta(x_i) \log \theta_{x_i|k}\Big), \\
&= \text{P}(c|\mathbf{x})(\delta(c=k) - \text{P}(k|\mathbf{x}))\delta(x_i)\log \theta_{x_i|k},
\end{aligned}$$

and plug it in Equation 8:

$$\frac{\partial \text{MSE}(\mathbf{w})}{\partial w_{k,i,x_i}} = -\sum_{i=1}^{N}\sum_{c}^{C}\Big(\delta(y=c) - \hat{\text{P}}(c|\mathbf{x})\Big)\text{P}(y|\mathbf{x})(\delta(y=k) - \text{P}(k|\mathbf{x}))\log \theta_{x_i|k}\delta(x_i). \tag{9}$$

The gradients for weights associated with class y (w_y) can be computed similarly. Comparing Equation 7 and 9, the following holds:

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

This shows that when optimizing MSE, just like CLL, naive Bayes preconditioning has the effect of scaling the gradients of a linear classification model by the log of the NB probability estimates. Such scaling leads to faster convergence, as is shown in the next section.

4 Experimental Results

In this section, we compare the performance of a linear model optimized with the MSE objective function with and without NB preconditioning in terms of 0-1 loss, RMSE, bias, variance, training time and the number of iterations it takes each algorithm to converge on 73 natural domains from the UCI repository (Table 1). We will also compare performance with LR and $\text{WANBIA}^{\text{C}}_{\text{CLL}}$ optimized with the CLL objective function.

Domain	Case	Att	Class	Domain	Case	Att	Class
Poker-hand	1175067	11	10	Annealing	898	39	6
Covertype	581012	55	7	Vehicle	846	19	4
Census-Income(KDD)	299285	40	2	PimaIndiansDiabetes	768	9	2
Localization	164860	7	3	BreastCancer(Wisconsin)	699	10	2
Connect-4Opening	67557	43	3	CreditScreening	690	16	2
Statlog(Shuttle)	58000	10	7	BalanceScale	625	5	3
Adult	48842	15	2	Syncon	600	61	6
LetterRecognition	20000	17	26	Chess	551	40	2
MAGICGammaTelescope	19020	11	2	Cylinder	540	40	2
Nursery	12960	9	5	Musk1	476	167	2
Sign	12546	9	3	HouseVotes84	435	17	2
PenDigits	10992	17	10	HorseColic	368	22	2
Thyroid	9169	30	20	Dermatology	366	35	6
Pioneer	9150	37	57	Ionosphere	351	35	2
Mushrooms	8124	23	2	LiverDisorders(Bupa)	345	7	2
Musk2	6598	167	2	PrimaryTumor	339	18	22
Satellite	6435	37	6	Haberman'sSurvival	306	4	2
OpticalDigits	5620	49	10	HeartDisease(Cleveland)	303	14	2
PageBlocksClassification	5473	11	5	Hungarian	294	14	2
Wall-following	5456	25	4	Audiology	226	70	24
Nettalk(Phoneme)	5438	8	52	New-Thyroid	215	6	3
Waveform-5000	5000	41	3	GlassIdentification	214	10	3
Spambase	4601	58	2	SonarClassification	208	61	2
Abalone	4177	9	3	AutoImports	205	26	7
Hypothyroid(Garavan)	3772	30	4	WineRecognition	178	14	3
Sick-euthyroid	3772	30	2	Hepatitis	155	20	2
King-rook-vs-king-pawn	3196	37	2	TeachingAssistantEvaluation	151	6	3
Splice-junctionGeneSequences	3190	62	3	IrisClassification	150	5	3
Segment	2310	20	7	Lymphography	148	19	4
CarEvaluation	1728	8	4	Echocardiogram	131	7	2
Volcanoes	1520	4	4	PromoterGeneSequences	106	58	2
Yeast	1484	9	10	Zoo	101	17	7
ContraceptiveMethodChoice	1473	10	3	PostoperativePatient	90	9	3
German	1000	21	2	LaborNegotiations	57	17	2
LED	1000	8	10	LungCancer	32	57	3
Vowel	990	14	11	Contact-lenses	24	5	3
Tic-Tac-ToeEndgame	958	10	2				

Table 1. Details of Datasets (UCI Domains)

In this work, we use the bias and variance definitions of [7] together with the repeated cross-validation bias-variance estimation method proposed by [8]. The reason for performing bias/variance estimation is that it provides insights into how the learning algorithm will perform with varying amount of data. We expect low variance algorithms to have relatively low error for small data and low bias algorithms to have relatively low error for large data [9].

The experiments are conducted on the datasets described in Table 1. There are a total of 73 datasets, 40 datasets with less than 1000 instances, 21 datasets with instances between 1000 and 10000, and 12 datasets with more than 10000 instances. The datasets with more than 10000 are shown in bold font in Table 1.

Each algorithm is tested on each dataset using 5 rounds of 2-fold cross validation. We report Win-Draw-Loss (W-D-L) results when comparing the 0-1 loss, RMSE, bias and variance of two models. A two-tail binomial sign test is used to determine the significance of the results. Results are considered significant if $p \leq 0.05$.

The datasets in Table 1 are divided into two categories. The first category constitutes all the datasets. The category is denoted by *All* in the results. The second category constitutes only datasets with more than 10000 instances. This is denoted by *Big* in the results When comparing average results across *All* and *Big* datasets, we normalize the results with respect to one of the comparative technique and present the geometric mean.

Numeric attributes are discretized by using the Minimum Description Length (MDL) discretization method [10]. A missing value is treated as a separate attribute value and taken into account exactly like other values.

We employed L-BFGS quasi-Newton methods [11] for solving the optimization[2].

We used a Random Forest that is an ensemble of 100 decision trees [13].

We will denote a linear model optimized with the MSE objective function with or without NB preconditioning as $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ and ANN respectively.

4.1 MSE vs. CLL

A win-draw-loss (W-D-L) comparison of bias, variance, 0-1 loss and RMSE of $\text{WANBIA}^{\text{C}}_{\text{CLL}}$ and LR versus $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ and ANN is given Table 2. It can be seen that $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ achieves significantly lower bias than $\text{WANBIA}^{\text{C}}_{\text{CLL}}$, whereas ANN has lower bias than LR but this difference does not achieve statistical significance. Both $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ and ANN exhibit higher variance, but this is statistically significant in the case of ANN vs. LR only. This suggests that both $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ and ANN are well suited for bigger datasets for which lower bias is preferable [14]. This is also evident from Table 2 where $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ has significantly lower 0-1 loss than $\text{WANBIA}^{\text{C}}_{\text{CLL}}$ on *Big* datasets. Similarly, the ANN results (with 9 wins, 1 draw and 2 losses), though not significantly different, are better than LR.

	$\textbf{WANBIA}^{\textbf{C}}_{\textbf{MSE}}$ **vs.** $\textbf{WANBIA}^{\textbf{C}}_{\textbf{CLL}}$		**ANN vs. LR**	
	W-D-L	p	W-D-L	p
		All Datasets		
Bias	45/7/20	**0.002**	38/5/28	0.276
Variance	19/6/47	**<0.001**	21/4/47	**0.002**
0-1 Loss	34/6/32	0.902	31/5/36	0.625
RMSE	29/4/39	0.275	31/3/38	0.470
		Big Datasets		
0-1 Loss	10/1/1	**0.011**	9/1/2	0.065
RMSE	8/0/4	0.387	8/0/4	0.387

Table 2. Win-Draw-Loss: $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ vs. $\text{WANBIA}^{\text{C}}_{\text{CLL}}$ and ANN vs. LR. p is two-tail binomial sign test. Results are significant if $p \leq 0.05$.

[2] The original L-BFGS implementation of [12] from `http://users.eecs.northwestern.edu/~nocedal/lbfgsb.html` is used.

In Figure 1, we show the geometric average of the results. It can be seen that $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ and ANN are lower-bias and higher-variance models as compared to $\text{WANBIA}^{\text{C}}_{\text{CLL}}$ and LR. The superior performance of $\text{WANBIA}^{\text{C}}_{\text{MSE}}$, however,

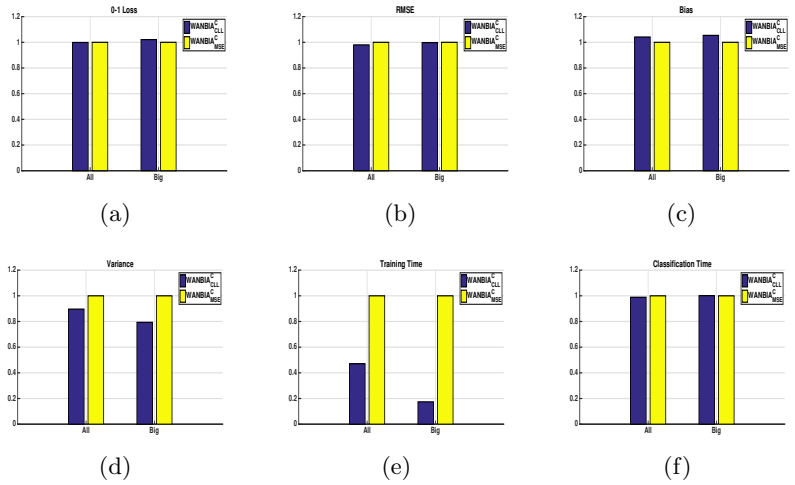

Fig. 1. An (geometric) average comparison of the 0-1 loss, RMSE, Bias and Variance of $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ and WANBIA-C on *All* and *Big* datasets.

comes at an extra cost. A comparison of the training and classification time of $\text{WANBIA}^{\text{C}}_{\text{CLL}}$ and $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ is shown in Figure 1(e) and Figure 1(f) respectively. It can be seen that optimizing the MSE objective function, though low biased, is a magnitude of order slower than optimizing the CLL objective function.

4.2 $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ vs. ANN

Now that we have established that optimizing the MSE for LR leads to a lower bias model than that by CLL, in this section, we will compare $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ and ANN to see the effects of scaling and whether NB preconditioning is as effective with the MSE as with the CLL objective function. We compare the scatter of 0-1 loss and RMSE values in Figures 2 and 3 respectively. It can be seen that both parameterizations lead to a similar scatter of 0-1 loss and RMSE. This suggests the equivalence of two models (same model, different parameterizations).

The training time and number of iterations to convergence for ANN and $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ is shown in Figures 4 and 5 respectively. It can be seen that $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ greatly improves the training time of ANN. Note, the plots are on the log scale. It can be seen that $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ on some datasets is an order of magnitude faster than ANN. Similarly, the number of iterations it takes $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ to converge are an order of magnitude less than for ANN.

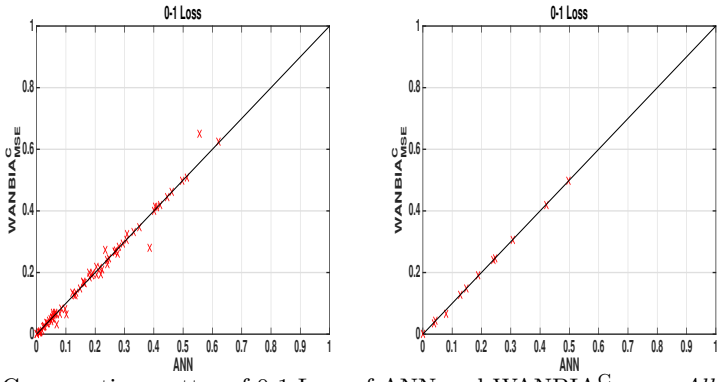

Fig. 2. Comparative scatter of 0-1 Loss of ANN and WANBIA$^{\mathrm{C}}_{\mathrm{MSE}}$ on *All* (Left) and *Big* (Right) datasets.

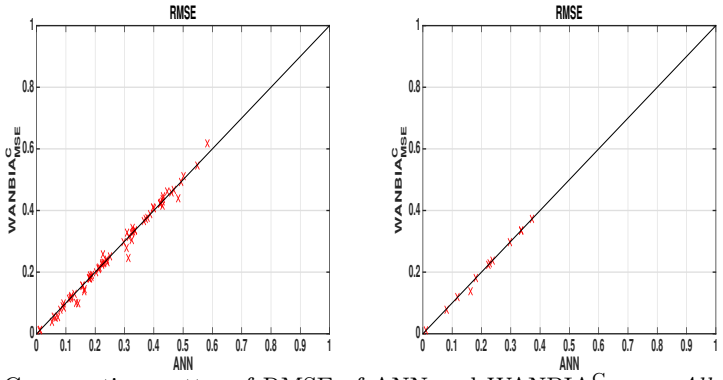

Fig. 3. Comparative scatter of RMSE of ANN and WANBIA$^{\mathrm{C}}_{\mathrm{MSE}}$ on *All* (Left) and *Big* (Right) datasets.

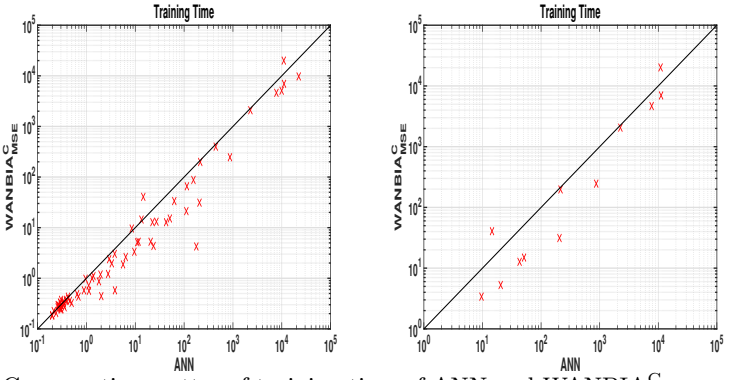

Fig. 4. Comparative scatter of training time of ANN and WANBIA$^{\mathrm{C}}_{\mathrm{MSE}}$ on *All* (Left) and *Big* (Right) datasets.

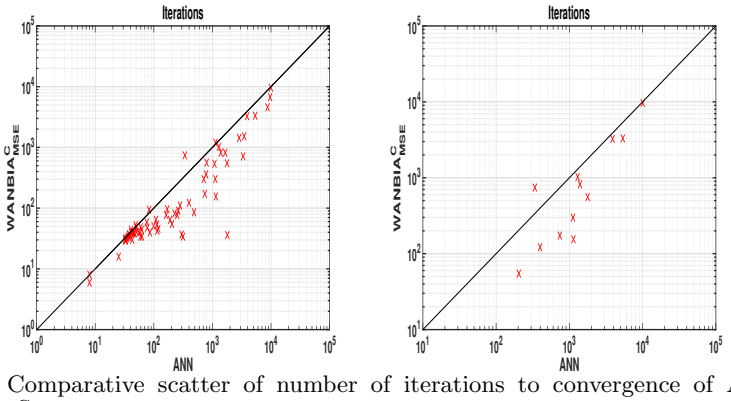

Fig. 5. Comparative scatter of number of iterations to convergence of ANN and $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ on *All* (Left) and *Big* (Right) datasets.

Finally, let us have a look at the convergence plots of ANN and $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ in Figure 6 on some sample datasets. The variation in mean-square-error is plotted with varying number of iterations until convergence. It can be seen that $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ has a much better convergence profile than ANN. It is not only converging in far fewer iterations but asymptoting far more quickly than ANN. This is extremely desirable when learning from few passes through the data.

4.3 $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ vs. Random Forest

In Table 3, we compare the performance of $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ with Random Forest. It can be seen that though not significantly better, bias of $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ is smaller than that of Random Forest. The variance of RF is slightly lower than that of $\text{WANBIA}^{\text{C}}_{\text{MSE}}$. On bigger datasets, RF has lower error than $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ slightly more often than $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ (winning on seven and losing on five datasets). Note that none of the results in the table are significant. A comparison of the training and classification time of $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ and RF is shown in Figure 7. It can be seen that $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ is an order of magnitude slower than RF on *Big* datasets at training time but at classification time, it is many order of magnitude faster than Random Forest.

5 Conclusion

In this paper, we showed that a linear classifier optimizing MSE has lower bias than vanilla LR optimizing CLL. We also showed that NB preconditioning, which is very effective for LR, is equally effective for a linear model optimized with MSE. We showed that NB preconditioning can speed-up convergence by many orders of magnitude resulting in convergence in far fewer iterations. The low-bias classification of a linear classifier optimized with MSE is competitive to state-of-the-art Random Forest classifier with an added advantage of faster training time. There are many interesting directions following from this work:

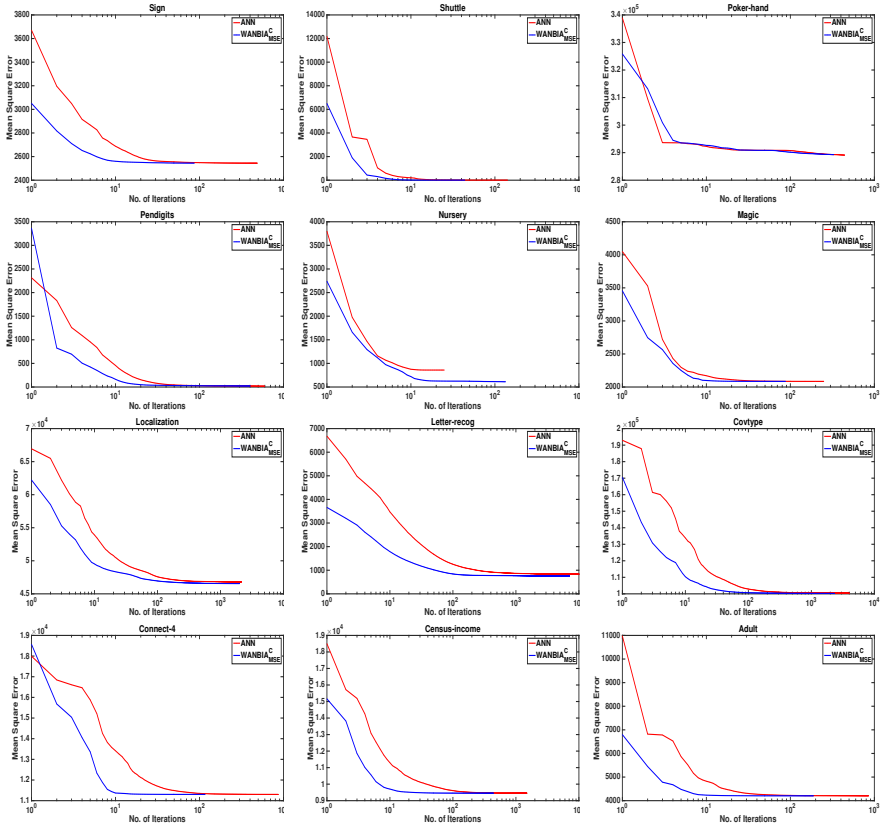

Fig. 6. Comparison of rate of convergence of WANBIA$^{C}_{MSE}$ and ANN on several datasets. The X-axis (No. of iterations) is on log scale.

– This paper shows that NB preconditioning is effective for an ANN with no hidden layers. It will be interesting to formulate similar preconditioning for ANNs with hidden layers. WANBIA$^{C}_{CLL}$ provides scaling for the nodes in the input layer, however, for nodes in the hidden layer, what weights one should use is an open question that needs investigation.
– It will be interesting to run WANBIA$^{C}_{MSE}$ with MSE with stochastic gradient descent (SGD) on very large datasets and compare the performance with WANBIA$^{C}_{CLL}$. We anticipate that WANBIA$^{C}_{MSE}$ will lead to lower error in fewer iterations.

Acknowledgements

This research has been supported by the Australian Research Council under grants DP120100553 and DP140100087, and Asian Office of Aerospace Research and Development, Air Force Office of Scientific Research under contracts FA2386-15-1-4007 and FA2386-15-1-4017.

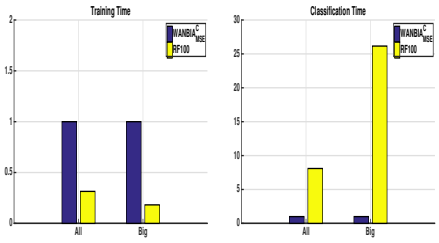

Fig. 7. Comparison of the (geometric) average of the training and classification time of RF and $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ on *All* and *Big* datasets.

	$\textbf{WANBIA}^{\textbf{C}}_{\textbf{MSE}}$ **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 3. Win-Draw-Loss: $\text{WANBIA}^{\text{C}}_{\text{MSE}}$ vs. Random Forest. p is two-tail binomial sign test. Results are significant if $p \leq 0.05$.

References

1. R. Duda, P. Hart, and D. Stork, *Pattern Classification*. John Wiley and Sons, 2006.
2. T. P. Minka, "A comparison of numerical optimizers for logistic regression," 2003.
3. 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.
4. 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*, 2014.
5. A. Martinez, S. Chen, G. I. Webb, and N. A. Zaidi, "Scalable learning of Bayesian network classifiers," *Journal of Machine Learning Research*, 2015.
6. N. A. Zaidi, G. I. Webb, M. J. Carman, and F. Petitjean, "Deep broad learning - Big models for Big data," *arXiv:1509.01346*, 2015.
7. R. Kohavi and D. Wolpert, "Bias plus variance decomposition for zero-one loss functions," in *ICML*, 1996, pp. 275–283.
8. G. I. Webb, "Multiboosting: A technique for combining boosting and wagging," *Machine Learning*, vol. 40, no. 2, pp. 159–196, 2000.
9. D. Brain and G. I. Webb, "The need for low bias algorithms in classification learning from small data sets," in *PKDD*, 2002, pp. 62–73.
10. U. M. Fayyad and K. B. Irani, "On the handling of continuous-valued attributes in decision tree generation," *Machine Learning*, vol. 8, no. 1, pp. 87–102, 1992.
11. C. Zhu, R. H. Byrd, and J. Nocedal, "LBFGSB, fortran routines for large scale bound constrained optimization," *ACM Transactions on Mathematical Software*, vol. 23, no. 4, pp. 550–560, 1997.
12. R. Byrd, P. Lu, and J. Nocedal, "A limited memory algorithm for bound constrained optimization," *SIAM Journal on Scientific and Statistical Computing*, vol. 16, no. 5, pp. 1190–1208, 1995.
13. L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 2001.
14. D. Brain and G. Webb, "On the effect of data set size on bias and variance in classification learning," in *Proceedings of the Fourth Australian Knowledge Acquisition Workshop*. University of New South Wales, 1999, pp. 117–128.