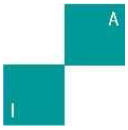
Inteligencia Artificial 18(56) (2015), 14-30
doi: 10.4114/ia.v18i56.1113



INTELIGENCIA ARTIFICIAL

http://journal.iberamia.org/

Naive Bayes vs Logistic Regression: Theory, Implementation and Experimental Validation

Tapan Kumar Bhowmik
LORIA UMR-7503, Université de Lorraine, France
tapan.bhowmik@loria.fr

Abstract This article presents the theoretical derivation as well as practical steps for implementing Naive Bayes (NB) and Logistic Regression (LR) classifiers. A generative learning under Gaussian Naive Bayes assumption and two discriminative learning techniques based on gradient ascent and Newton-Raphson methods are described to estimate the parameters of LR. Some limitations of these learning techniques and implementation issues are discussed as well. A set of experiments is performed for both the classifiers under different learning circumstances and their performances are compared. Statistical tests are performed in this regards to measure the significance of comparison. From the experiments, it is observed that LR learning with gradient ascent technique outperforms general NB classifier. However, under Gaussian Naive Bayes assumption, both classifiers NB and LR perform similar.

Keywords: Naive Bayes, Logistic Regression, Generative Learning, Discriminative Learning, Paired t-Test, McNemar Test.

1 Introduction

Naive Bayes (NB) and Logistic Regression (LR) are two popular classifiers in machine learning [5]. In fact, NB is one of the fastest generative learning classifier for large-scale prediction and classification tasks on complex and incomplete datasets and at the same time it can handle both categorical (discrete) and non-categorical (continuous) data. On the other hand, LR is usually a discriminative learning classifier [7] which is ideally developed for two class problems and can also handle both categorical and non-categorical data. Though, in general, LR is a discriminative classifier, it can be built as a generative classifier using generative learning techniques.
Many learning techniques are available for both classifiers in the literature [6, 3]. In fact, most of the theoretical derivations in this study are inspired by Tom Mitchell's [6] article. However, the aim of this work is to bring forth a complete study describing from the basic theoretical concepts of learning techniques to their implementation details which have not been studied yet. Here, we provide the implementation details for building a generative NB classifier and three different learning techniques: i) Gaussian Naive Bayes assumption (generative), ii) gradient ascent (discriminative) and iii) Newton Rapshon method using Hessian matrix (discriminative) for LR classifier. Some implementation issues are discussed as well. Finally, a set of experiments is performed for both classifiers under different learning techniques to compare their performances.
The rest of the article is organized as follows: In sections 2 and 3, we describe the theory of Naive Bayes and Logistic regression classifiers including implementation and some related issues. The experimental results are given in section 4. Finally, some conclusions are made from the experiments in section 5.

ISSN: 1988-3064(on-line)

2 Naive Bayes

A Naive Bayes is a simple and well known probabilistic classifier based on Bayesian decision theory. It is "naive" in the sense that an attribute value on a given class is assumed to be independent of the values of the other attributes.

2.1 Bayes' Theorem

If $A_1, A_2, \ldots, A_n$ be a given set of n pairwise mutually exclusive events, one of which certainly occurs, i.e. A_iA_j=O (null or impossible event), ($i \neq j$; i, $j = 1, 2, \ldots n$) and $A_1 + A_2 + \ldots + A_n = S$ (a certain event), then for any arbitrary event X

$$\begin{aligned} P(X) &= P(A_1)P(X|A_1) + P(A_2)P(X|A_2) + \ldots \\ &\quad +P(A_n)P(X|A_n) \\ &= \sum_{i=1}^{n} P(A_i)P(X|A_i) \end{aligned}$$

and if $P(X) \neq 0$

$$P(A_i|X) = \frac{P(A_i)P(X|A_i)}{\sum_{i=1}^{n} P(A_i)P(X|A_i)}, i = 1, 2, \ldots n \quad (1)$$

Proof:

We have $X = SX = (A_1 + A_2 + \ldots + A_n)X = A_1X + A_2X + \ldots + A_nX$ (see in Figure 1). Since $(A_iX)(A_jX) = A_iA_jX = OX = O$, ($i \neq j$; i, $j = 1, 2, \ldots n$), $A_1X, A_2X, \ldots, A_nX$ are pairwise mutually exclusive events, and hence
$P(X) = P(A_1X) + P(A_2X) + \ldots + P(A_nX)$
Now from the conditional probability we know that
$P(A_iX) = P(A_i)P(X|A_i) = P(X)P(A_i|X)$.
Hence if $P(X) \neq 0$,

$$\begin{aligned} P(A_i|X) &= \frac{P(A_i)P(X|A_i)}{P(X)} \\ &= \frac{P(A_i)P(X|A_i)}{\sum_{i=1}^{n} P(A_iX)} \\ &= \frac{P(A_i)P(X|A_i)}{\sum_{i=1}^{n} P(A_i)P(X|A_i)} \end{aligned}$$

2.2 Naive Bayes' Classifier

Let $X = (X_1, X_2, \ldots, X_n)$ be a random variable and $A_1, A_2, \ldots, A_n$ be the attributes of X associated with the n components $X_1, X_2, \ldots X_n$ respectively (see in Figure 2). Let $T = \{\mathbf{x} = (X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n)\}$ be the set of training samples drawn from the population of X. Let us assume that there are c classes, $C = \{y_1, y_2, \ldots, y_c\}$ and each and every samples having a certain class labels $Y = y_j \in C$. The task of the classifier is to predict the class label Y for a given sample $\mathbf{x}$. In order to predict the class label of $\mathbf{x}$, the naive Bayes calculates $P(Y = y_j|\mathbf{x})$ for each class $y_j, j = 1, 2, \ldots, c$ and the sample $\mathbf{x}$ is classified in that class whose probability shows the highest value. In mathematical terms

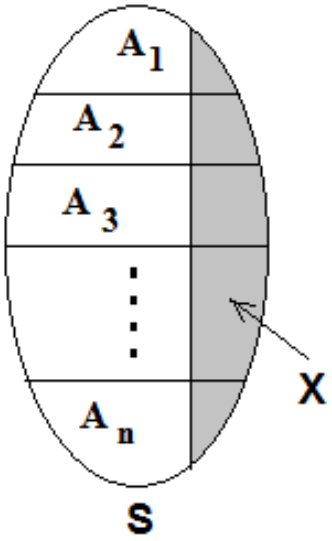


Figure 1: Mutual exclusive events $A_1, A_2, \ldots A_n$ and any arbitrary event X

$$
\begin{aligned}
C_{MAP} &= \arg\max_{y_j \in C} P(Y = y_j|\mathbf{x}) \\
&= \arg\max_{y_j \in C} \frac{P(\mathbf{x}|Y = y_j)P(Y = y_j)}{\sum_j P(\mathbf{x}|Y = y_j)P(Y = y_j)} \\
&\quad \text{(by Bayes theorem in Equation 1)} \\
&= \arg\max_{y_j \in C} P(\mathbf{x}|Y = y_j)P(Y = y_j) \\
&= \arg\max_{y_j \in C} P((x_1, x_2, \ldots, x_n)|Y = y_j)P(Y = y_j) \\
&= \arg\max_{y_j \in C} \prod_{i=1}^{n} P(x_i|Y = y_j)P(Y = y_j) \qquad (2) \\
&\quad \text{(since, the attribute values of } \mathbf{x} \text{ are condition-} \\
&\quad \text{ally independent according to naive bayes} \\
&\quad \text{assumption)}
\end{aligned}
$$

2.2.1 Learning Phase:

In the learning phase, the probabilities $P(x_1|Y = y_j), P(x_2|Y = y_j), \ldots, P(x_n|Y = y_j)$ for each class y_j $(j = 1, 2, \ldots c)$ are estimated from the training samples (see also [8]). For estimating the probabilities of $Y = y_j$, it simply uses the frequencies of $Y = y_j$ in the training samples. So,

$$P(Y = y_j) = \frac{\mathcal{N}(Y = y_j, T)}{\mathcal{N}}, \qquad (3)$$

where $\mathcal{N}(Y = y_j, T)$ is the number of sample of class y_j in T and $\mathcal{N}$ is the number of training samples in T.
(a) If A_i is categorical or alternatively if X_i having discrete observations, then

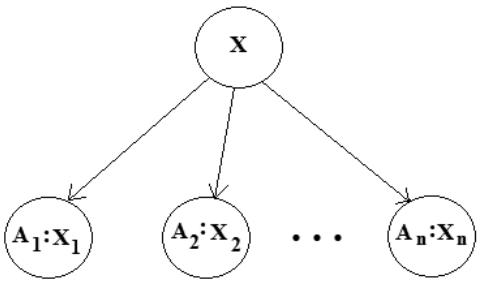


Figure 2: A random variable X is described with n attributes $A_1, A_2, \ldots A_n$

$$P(X_i = x_k|Y = y_j) = \frac{\mathcal{N}(X_i = x_k, Y = y_j, T)}{\mathcal{N}(Y = y_j, T)}, \tag{4}$$

where $\mathcal{N}(X_i = x_k, Y = y_j, T)$ is the number of samples of class y_j in T having the value x_k for attribute A_i.
(b) If A_i is continuous-valued, then we typically assume that the values have a Gaussian distribution with a mean μ_j and standard deviation σ_j and then

$$P(X_i = x_k|Y = y_j) = \frac{1}{\sqrt{2\pi}\sigma_j} \exp\left(-\frac{(x_k - \mu_j)^2}{2{\sigma_j}^2}\right) \tag{5}$$

So, from the training samples we need to compute μ_j and σ_j, which are the mean and standard deviation of values of attribute A_i for training samples of class y_j.

2.2.2 Zero conditional probability problem (Laplacian Correction):

Based on the class independence assumption in naive bayes it can be written as

$$P(x_1, x_2, \ldots, x_n|Y = y_j) = P(x_1|Y = y_j)P(x_2|Y = y_j) \ldots P(x_n|Y = y_j).$$

Now, let there is a class y_j, and X has an attribute value, x_k, such that none of the samples in $Y = y_j$ has that attribute value. In that case, $P(x_k|Y = y_j) = 0$ which results in $P(x_1, x_2, \ldots, x_n|Y = y_j) = 0$ even though $P(x_k|Y = y_j)$ for all other attributes in X may be large.
There is a simple trick to avoid this problem. We can assume that our training set is so large that adding one to each count would only make a negligible difference in the estimated probabilities. This would also avoid the case of zero probability values. This technique is know as Laplacian correction (or Laplace estimator). If there are q counts to which we each add one, then we need to add q also to the corresponding denominator in the probability calculation. So,

$$P(X_i = x_k|Y = y_j) = \frac{\mathcal{N}(X_i = x_k, Y = y_j, T) + 1}{\mathcal{N}(Y = y_j, T) + q}, \tag{6}$$

where q is the number of distinct possible values of X_i.

3 Logistic Regression

Logistic Regression is a well known technique that efficiently used for modeling categorical outcomes as a function of both continuous and categorical variables in various applications. It is commonly used for predicting the probability of occurrence of an event, based on several predictor variables that may either be numerical or categorical.

Let us consider the functions of the form $Y = f(X)$ or $f : X \rightarrow Y$ or $P(Y|X)$ in the case where Y is discrete-valued, and $X = (X_1, X_2, \ldots, X_n)$ is any vector containing discrete or continuous random variables. In this description, we consider the case only where Y is a boolean variable (say, either 0 or 1), in order to simplify notation. But in general Y can be any finite number of discrete values.

Logistic Regression assumes a parametric form for the distribution $P(Y|X)$, then directly estimates its parameters from the training data. The parametric model assumed by Logistic Regression in the case where Y is boolean is:

$$\begin{aligned} P(Y=1|X) &= \frac{1}{1+\exp -(w_0 + \sum_{i=1}^{n} w_i X_i)} \\ &= \frac{\exp(w_0 + \sum_{i=1}^{n} w_i X_i)}{1+\exp(w_0 + \sum_{i=1}^{n} w_i X_i)} \end{aligned} \tag{7}$$

and

$$\begin{aligned} P(Y=0|X) &= 1 - P(Y=1|X) \\ &= \frac{1}{1+\exp(w_0 + \sum_{i=1}^{n} w_i X_i)} \end{aligned} \tag{8}$$

Here, $P(Y=1|X)$ assumes to follow the form $Y = \frac{1}{1+exp(-X)}$ which is shown in Figure 3. One highly convenient property of this form for $P(Y|X)$ is that it leads to a simple linear expression for classification. To classify any given X we generally want to assign the value y_j that maximizes $P(Y = y_j|X)$. Put another way, we assign the label $Y = 1$ if the following condition holds:

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

substituting from equations 8 and 7, this becomes

$$\exp(w_0 + \sum_{i=1}^{n} w_i X_i) > 1$$

and taking the natural log of both sides we have a linear classification rule that assign the label Y if X satisfies as

$$Y = \begin{cases} 1 & \text{if } (w_0 + \sum_{i=1}^{n} w_i X_i) > 0 \\ 0 & \text{otherwise.} \end{cases}$$

3.1 Learning Logistic Regression

The learning of LR deals with the estimation of $n+1$ parameters say, $W = (w_0, w_1, w_2, \ldots, w_n)$ for which the probability of the observed data is the maximum on the training data set. Several techniques are used to estimate these parameters. Among them, three widely used learning techniques are described in the following. The first technique is based on generative learning and other two are on discriminative learning.

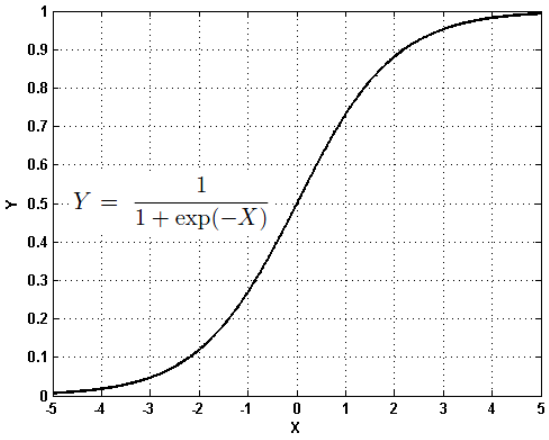


Figure 3: The logistic curve $Y = \frac{1}{1+exp(-X)}$.

3.1.1 Estimating parameters under Gaussian Naive Bayes assumptions:

One of the simplest way to estimate the parameters of LR is under the Gaussian Naive Bayes assumptions where the model is built under the following constraint:

1. Assume that the random variable Y can take only two values say, $\{0, 1\}$.

2. $X_i \sim N(\mu_{ij}, \sigma_i)$, for all i. Here, μ_{ij} is the mean of X_i associated with the class $Y = y_j$ and σ_i varies from attribute to attribute not depending on Y.

3. For all i and $k \neq i$, X_i and X_k are conditionally independent given Y.

Now according to bayes theorem we can write,

$$\begin{aligned}
P(Y=1|X) &= \frac{P(Y=1)P(X|Y=1)}{P(Y)} \\
&= \frac{P(Y=1)P(X|Y=1)}{P(Y=0)P(X|Y=0)+P(Y=1)P(X|Y=1)} \\
&= \frac{1}{1+\frac{P(Y=0)P(X|Y=0)}{P(Y=1)P(X|Y=1)}} \\
&= \frac{1}{1+\exp(\ln\frac{P(Y=0)P(X|Y=0)}{P(Y=1)P(X|Y=1)})} \\
&= \frac{1}{1+\exp(\ln\frac{P(Y=0)}{P(Y=1)}+\sum_i \ln\frac{P(X_i|Y=0)}{P(X_i|Y=1)})} \text{(By assumption 3)} \\
&= \frac{1}{1+\exp(\ln\frac{1-\pi}{\pi}+\sum_i \ln\frac{P(X_i|Y=0)}{P(X_i|Y=1)})} \text{(Assume that } \pi = P(Y=1))
\end{aligned}$$

Now,

$$\sum_i \ln \frac{P(X_i|Y=0)}{P(X_i|Y=1)} = \sum_i \ln \frac{\frac{1}{\sqrt{2\pi\sigma_i^2}} \exp(\frac{-(X_i-\mu_{i0})^2}{2\sigma_i^2})}{\frac{1}{\sqrt{2\pi\sigma_i^2}} \exp(\frac{-(X_i-\mu_{i1})^2}{2\sigma_i^2})}$$
$$= \sum_i \ln \exp\left(\frac{(X_i-\mu_{i1})^2 - (X_i-\mu_{i0})^2}{2\sigma_i^2}\right)$$
$$= \sum_i \left(\frac{2X_i(\mu_{i0}-\mu_{i1}) + (\mu_{i1}^2 - \mu_{i0}^2)}{2\sigma_i^2}\right)$$
$$= \sum_i \left(\frac{(\mu_{i0}-\mu_{i1})}{\sigma_i^2} X_i + \frac{(\mu_{i1}^2 - \mu_{i0}^2)}{2\sigma_i^2}\right)$$

Now substituting this value in the above equation we get

$$P(Y=1|X) = \frac{1}{1+\exp(\ln\frac{1-\pi}{\pi} + \sum_i \frac{(\mu_{i0}-\mu_{i1})}{\sigma_i^2} X_i + \frac{(\mu_{i1}^2-\mu_{i0}^2)}{2\sigma_i^2})}$$
$$= \frac{1}{1+\exp\{-(\ln\frac{\pi}{1-\pi} + \sum_i \frac{(\mu_{i1}-\mu_{i0})}{\sigma_i^2} X_i + \frac{(\mu_{i0}^2-\mu_{i1}^2)}{2\sigma_i^2})\}}$$

Or equivalently,

$$P(Y=1|X) = \frac{1}{1+\exp -(w_0 + \sum_{i=1}^{n} w_i X_i)} \tag{9}$$

where the weights $w_1, w_2, \ldots, w_n$, are given by

$$w_i = \frac{(\mu_{i1}-\mu_{i0})}{\sigma_i^2}$$

and

$$w_0 = \ln\frac{\pi}{1-\pi} + \sum_i \frac{(\mu_{i0}^2 - \mu_{i1}^2)}{2\sigma_i^2}$$

Similarly,

$$P(Y=0|X) = 1 - P(Y=1|X)$$
$$= \frac{1}{1+\exp(w_0 + \sum_{i=1}^{n} w_i X_i)} \tag{10}$$

3.1.2 Estimating parameters with gradient ascent:

Another reasonable approach is to choose the parameter values that maximize the conditional data likelihood or alternatively maximize the conditional data log likelihood. The conditional data likelihood is the probability of the observed Y values in the training data, conditioned on their corresponding X values. Let Y^l be the observed data of Y in the l^{th} training sample. Now if the conditional data log likelihood is denoted by $L(W)$, then we can write

$$L(W) = \log \prod_l P(Y^l|X^l, W)$$

For better understanding the following derivation, let us assume that Y is a random variable can take only two values $\{0,1\}$ and $P(Y=y_l) = p^{y_l}(1-p)^{(1-y_l)}$. So, $P(Y=1)=p$ and $P(Y=0)=1-p$. Now

if there is $\mathcal{N}$ training samples, then total likelihood over all training samples

$$\prod_l P(Y = y_l) = p^{y_1+y_2+\ldots+y_{\mathcal{N}}}(1-p)^{\mathcal{N}-(y_1+y_2+\ldots+y_{\mathcal{N}})}$$

and the log likelihood

$$\begin{aligned} L = \log \prod_l P(Y = y_l) &= (y_1 + y_2 + \ldots + y_{\mathcal{N}}) \log p \\ &\quad + \{\mathcal{N} - (y_1 + y_2 + \ldots + y_{\mathcal{N}})\} \log (1-p) \\ &= \sum_l \{y_l \log P(Y=1) + (1-y_l) \log P(Y=0)\} \end{aligned}$$

Similarly, we can write

$$\begin{aligned} L(W) &= \sum_l \big\{ Y^l \log P(Y^l = 1 | X^l, W) \\ &\quad + (1 - Y^l) \log P(Y^l = 0 | X^l, W) \big\} \\ &= \sum_l Y^l \left\{ \log \frac{P(Y^l = 1 | X^l, W)}{P(Y^l = 0 | X^l, W)} \right. \\ &\quad \left. + \log P(Y^l = 0 | X^l, W) \big\} \right. \end{aligned}$$

Therefore,

$$\begin{aligned} L(W) &= \sum_l \left\{ Y^l (w_0 + \sum_{i=1}^{n} w_i X_i^l) \right. \\ &\quad \left. - \log \left(1 + \exp(w_0 + \sum_{i=1}^{n} w_i X_i^l)\right) \right\} \\ &\quad (\texttt{By the Equations } 8 \texttt{ and } 7) \end{aligned}$$

where X_i^l denotes the value of X_i for the l^{th} training sample. Unfortunately, there is no closed form solution to maximizing $L(W)$ with respect to W. So the alternative approach is to use gradient ascent technique to optimize the weights W by calculating the partial derivative with respect to each w_i say, $\frac{\partial L(W)}{\partial w_i}$.

Now,

$$\begin{aligned} \frac{\partial L(W)}{\partial w_i} &= \sum_l (Y^l X_i^l - X_i^l \frac{\exp(w_0 + \sum_{i=1}^{n} w_i X_i^l)}{1 + \exp(w_0 + \sum_{i=1}^{n} w_i X_i^l)}) \\ &= \sum_l X_i^l (Y^l - P(Y^l = 1 | X^l, W)). \end{aligned}$$

According to gradient ascent technique, the updating rule for weights w_i is given by

$$w_i(t+1) = w_i(t) + \eta \frac{\partial L(W)}{\partial w_i}$$

or,

$$w_i(t+1) = w_i(t) + \eta \sum_l X_i^l (Y^l - P(Y^l = 1 | X^l, W)), \tag{11}$$

where η is a small constant (say, $\eta = .001$), called the learning rate. An algorithm for estimating parameters of logistic regression for two class problem with gradient ascent technique is described in

Algorithm 1 where the term $P(Y = 1|X)$ is calculated as:

$$P(Y = 1|X) = \frac{\exp(w_0 + \sum_{i=1}^{n} w_i X_i)}{1 + \exp(w_0 + \sum_{i=1}^{n} w_i X_i)} = \frac{1}{1 + \exp -(w_0 + \sum_{i=1}^{n} w_i X_i)}$$

Two convergent criteria are used in Algorithm 1. The first one is the fixed number of iterations and other is the prospective parameter change (PPC) measure. PPC measures the maximum relative change in the parameters implied by the parameter-change vector computed for the next iteration and it is defined as

$$\frac{|w_i(t+1) - w_i(t)|}{w_i(t) + \epsilon},$$

where $\epsilon > 0$ is a small positive constant and $w_i(t)$ is the current value of the parameter w_i and $w_i(t+1)$ is the prospective value of this parameter after adding the change vector computed for the next iteration. The iteration procedure is repeated unless PPC is closer to zero.

For classification, we simply calculate $R = (w_0 + \sum_{i=1}^{n} w_i X_i)$. If $R > 0$ assign class label $Y = 1$, otherwise $Y = 0$. An algorithm for classification is described in Algorithm 2.

3.1.3 Logistic Regression for Functions with Many Discrete Values:

In the above description, we consider Y having only two values $\{0, 1\}$, but in general case Y can take on any of the finite discrete values. Let us assume that $Y \in \{y_0, y_1, \ldots, y_c\}$. One of the simplest ways we can realize is a multinomial logistic regression model with $c+1$ discrete outcomes as a set of independent binary logistic regression models in which one outcome say, $Y = y_0$ can be considered as a pivot and then the other c outcomes say, $Y = y_j (j = 1, 2 \ldots c)$ are separately regressed against the pivot outcome. Let us assume that for any $Y = y_j (j = 1, 2, \ldots, c)$

$$\frac{P(Y = y_j|X)}{P(Y = y_0|X)} = \exp(w_{j0} + \sum_{i=1}^{n} w_{ji} X_i), j = 1, 2, \ldots, c.$$

Since,

$$P(Y = y_0) + P(Y = y_1) + P(Y = y_2) + \ldots + P(Y = y_c) = 1$$

Therefore,

$$P(Y = y_0|X) + P(Y = y_0|X)(\sum_{j=1}^{c} \exp(w_{j0} + \sum_{i=1}^{n} w_{ji} X_i)) = 1$$

or,

$$P(Y = y_0|X) = \frac{1}{1 + \sum_{j=1}^{c} \exp(w_{j0} + \sum_{i=1}^{n} w_{ji} X_i)} \tag{12}$$

and for $Y = y_j; j = 1, 2, \ldots c$,

$$P(Y = y_j|X) = \frac{\exp(w_{j0} + \sum_{i=1}^{n} w_{ji} X_i)}{1 + \sum_{j=1}^{c} \exp(w_{j0} + \sum_{i=1}^{n} w_{ji} X_i)} \tag{13}$$

Algorithm 1: TWO-CLASS LOGISTIC REGRESSION: Learning parameters with gradient ascent technique

Input: A set of training samples $T = \{X^l = (X_1^l, X_2^l, \dots X_n^l)\}$
: A corresponding set of labels $\{Y^l \in \{0, 1\}\}$
: $convergeThreshold$
: $maxIteration$
Output: Learning parameters $W = (w_0, w_1, w_2, \dots, w_n)$

Set $\eta \leftarrow 0.001$, $\epsilon \leftarrow 0.000001$
Set $W = (w_0, w_1, w_2, \dots, w_n) \leftarrow 0.0$
Set $GW = (gw_0, gw_1, gw_2, \dots, gw_n) \leftarrow 0.0$
Set $iteration \leftarrow 0$
Set $converge \leftarrow false$
while $converge == false$ **do**
  for $l \leftarrow 1$ ***to*** $\mathcal{N}(T)$ **do**
    Set $sprob \leftarrow w_0$
    for $i \leftarrow 1$ ***to*** n **do**
      $sprob \leftarrow sprob + (w_i * X_i^l)$
    $sprob \leftarrow \frac{1}{1+\exp(-sprob)}$
    Set $eterm \leftarrow sprob$
    if $Y^l == 1$ **then**
      $eterm \leftarrow 1.0 - eterm$
    else
      $eterm \leftarrow 0.0 - eterm$
    $gw_0 \leftarrow gw_0 + eterm$
    for $i \leftarrow 1$ ***to*** n **do**
      $gw_i \leftarrow gw_i + (eterm * X_i^l)$
  Set $ppc \leftarrow 0.0$
  Set $ppc_{deno} \leftarrow 0.0$
  for $i \leftarrow 0$ ***to*** n **do**
    Set $pw \leftarrow w_i$
    $ppc_{deno} \leftarrow ppc_{deno} + \mid pw \mid + \epsilon$
    $w_i \leftarrow w_i + \eta * gw_i$
    $ppc \leftarrow ppc + \mid w_i - pw \mid$
  $ppc \leftarrow \frac{ppc}{ppc_{deno}}$
  $iteration \leftarrow iteration + 1$
  if $(ppc \leq convergeThreshold)$ *or* $(iteration \geq maxIteration)$ **then**
    $converge \leftarrow true$
return $W = (w_0, w_1, w_2, \dots, w_n)$

Here w_{ji} denotes the weight associated with the j^{th} class $Y = y_j$ and with input X_i. In this case, the gradient ascent rule for updating weights becomes:

$$\begin{aligned} w_{ji}(t+1) &= w_{ji}(t) + \eta \sum_l \left\{ X_i^l (\delta(Y^l = y_j) \right. \\ &\quad \left. - P(Y^l = y_j | X^l, W)) \right\}, \end{aligned} \tag{14}$$

where $\delta(Y^l = y_j) = 1$ if the l^{th} training value, Y^l, is equal to y_j and $\delta(Y^l = y_j) = 0$ otherwise. In the learning stage, we need to construct c different linear expressions (for $c+1$ different outcomes) to capture the distributions for the different values of $Y = y_1, y_2, \dots y_c$. The distribution for $Y = y_0$, the value of Y is simply one minus the probabilities of the c values.

Algorithm 2: Two-class Logistic Regression: Classification

Input: A set of samples $T = \{X^l = (X_1^l, X_2^l, \ldots X_n^l)\}$
: A set of learned parameters $W = (w_0, w_1, w_2, \ldots, w_n)$
Output: : A set of assign labels $\{Y^l \in \{0, 1\}\}$

```
for l ← 1 to 𝒩(T) do
    Set R ← w_0
    for i ← 1 to n do
        R ← R + (w_i * X_i^l)
    if R > 0 then
        Set Y^l ← 1
    else
        Set Y^l ← 0
return {Y^l}
```

However, in order to make a simple and straight forward algorithm for implementation, instead of estimating the coefficients for c regressions, we estimate the coefficients for $c + 1$ regressions where one outcome is regressed against the other outcomes. The weights w_{ji} associate with the j^{th} class is updated by computing the probabilities (as in Algorithm 1)

$$\begin{aligned} P(Y = y_j|X) &= \frac{1}{1 + \exp -(w_{j0} + \sum_{i=1}^{n} w_{ji}X_i)} \\ &= \frac{\exp(w_{j0} + \sum_{i=1}^{n} w_{ji}X_i)}{1 + \exp(w_{j0} + \sum_{i=1}^{n} w_{ji}X_i)} \end{aligned} \tag{15}$$

and for

$$P(Y \neq y_j|X) = \frac{1}{1 + \exp(w_{j0} + \sum_{i=1}^{n} w_{ji}X_i)} \tag{16}$$

For the classification, we only compute the value of $(w_{j0} + \sum_{i=1}^{n} w_{ji}X_i)$ for each class j and the sample is classified into that class which shows the maximum value. The algorithms for learning and classification of multiclass logistic regression model is described in Algorithm 3 and Algorithm 4 respectively.

3.1.4 Regularization in Logistic regression:

A common problem with LR is overfitting the training data, especially when the training data are high dimensional and/or sparse. One approach to reduce overfitting is regularization, where we create a modified "penalized log likelihood function", which penalizes large values of W. The penalized log likelihood function can be done by adding a penalty proportional to the squared magnitude of W. In this case, the update rules becomes

$$\begin{aligned} w_i(t+1) &= w_i(t) + \eta \sum_l X_i^l (Y^l - P(Y^l = 1|X^l, W)) \\ &\quad -\eta\lambda w_i(t) \end{aligned} \tag{17}$$

Algorithm 3: Multi-class Logistic Regression: Learning parameters with gradient ascent technique

Input: A set of training samples $T = \{X^l = (X_1^l, X_2^l, \dots X_n^l)\}$
: A corresponding set of labels $\{Y^l \in \{y_0, y_1, \dots y_c\}\}$
: $convergeThreshold$
: $maxIteration$
Output: Learning parameters $W = [w_{ji}]_{(c+1)\times(n+1)}$

Set $\eta \leftarrow 0.001$
Set $\epsilon \leftarrow 0.000001$
Set $W = [w_{ji}]_{(c+1)\times(n+1)} \leftarrow 0.0$
Set $GW = [gw_{ji}]_{(c+1)\times(n+1)} \leftarrow 0.0$
Set $iteration \leftarrow 0$
Set $converge \leftarrow false$
while $converge == false$ **do**
  Set $ppc \leftarrow 0.0$
  Set $ppc_{deno} \leftarrow 0.0$
  for $j \leftarrow 0$ ***to*** c **do**
    for $l \leftarrow 1$ ***to*** $\mathcal{N}(T)$ **do**
      Set $sprob \leftarrow w_{j0}$
      for $i \leftarrow 1$ ***to*** n **do**
        $sprob \leftarrow sprob + (w_{ji} * X_i^l)$
      $sprob \leftarrow \frac{1}{1+\exp(-sprob)}$
      Set $eterm \leftarrow sprob$
      if $Y^l == j$ **then**
        $eterm \leftarrow 1.0 - eterm$
      else
        $eterm \leftarrow 0.0 - eterm$
      $gw_{j0} \leftarrow gw_{j0} + eterm$
      for $i \leftarrow 1$ ***to*** n **do**
        $gw_{ji} \leftarrow gw_{ji} + (eterm * X_i^l)$
    for $i \leftarrow 0$ ***to*** n **do**
      Set $pw \leftarrow w_{ji}$
      $ppc_{deno} \leftarrow ppc_{deno} + \mid pw \mid + \epsilon$
      $w_{ji} \leftarrow w_{ji} + \eta * gw_{ji}$
      $ppc \leftarrow ppc + \mid w_{ji} - pw \mid$
  $ppc \leftarrow \frac{ppc}{ppc_{deno}}$
  $iteration \leftarrow iteration + 1$
  if $(ppc \leq convergeThreshold)$ *or* $(iteration \geq maxIteration)$ **then**
    $converge \leftarrow true$
return $W = [w_{ji}]_{(c+1)\times(n+1)}$

Algorithm 4: Multi-class Logistic Regression: Classification

Input: A set of samples $T = \{X^l = (X_1^l, X_2^l, \ldots X_n^l)\}$
: A set of learned parameters $W = [w_{ji}]_{(c+1)\times(n+1)}$
Output: : A set of assign labels $\{Y^l \in \{y_0, y_1, \ldots y_c\}\}$

```
for l ← 1 to 𝒩(T) do
    for j ← 0 to c do
        R_j ← w_j0
        for i ← 1 to n do
            R_j ← R_j + (w_ji * X_i^l)
    j_max = arg max_j(R_j)
    Set Y^l ← j_max
return {Y^l}
```

when Y is boolean and for general cases it becomes

$$\begin{aligned} w_{ji}(t+1) &= w_{ji}(t) + \eta \sum_l \left\{ X_i^l \left(\delta(Y^l = y_j) \right.\right. \\ &\quad \left.\left. - P(Y^l = y_j | X^l, W) \right) \right\} - \eta \lambda w_{ji}(t) \end{aligned} \tag{18}$$

Here, λ is a constant that determines the strength of the penalty term in "penalized log likelihood function".

3.1.5 Estimating parameters with Hessian matrix:

The ultimate aim of logistic regression learning is to estimate the parameters $W = [w_{ji}]_{(c+1)\times(n+1)}$ for which the probability of the observed data is greatest on the training data set. Here, the weights of j^{th} row in W are associated with the j^{th} class $Y = y_j$. In order to understand simply, it is assumed that Y can have only two discrete values say, $\{0, 1\}$. In this case, W contains two rows. However, for the classification it is sufficient to have only a single row parameters $W = [w_0, w_1, \ldots, w_n]$ as discussed earlier. The estimation process is done with maximum likelihood estimation which entails finding the set of parameters for which the probability of the observed data is greatest. To maximize the likelihood value or specifically the log-likelihood value $L(W)$, a gradient ascent technique is discussed earlier. For each w_i, the partial derivative $\frac{\partial L(W)}{\partial w_i}$ is calculated as:

$$\begin{aligned} \frac{\partial L(W)}{\partial w_i} &= \sum_l (Y^l X_i^l - X_i^l \frac{\exp(w_0 + \sum_{i=1}^n w_i X_i^l)}{1 + \exp(w_0 + \sum_{i=1}^n w_i X_i^l)}) \\ &= \sum_l X_i^l (Y^l - P(Y^l = 1 | X^l, W)). \end{aligned}$$

Another straight forward way in which $L(W)$ can be maximized is by setting the above partial derivatives to zero. So in this case, there is a set of $n+1$ non-linear equations $\frac{\partial L(W)}{\partial w_i} = 0, i = 1, 2, \ldots n$, each of them has $n+1$ unknown variables. However, solving a system of non-linear equations is not easy and straight forward. An iterative Newton-Raphson algorithm is used commonly to find the solution where the procedure uses the partial second derivatives of the parameters in the forms $\frac{\partial^2 L(W)}{\partial {w_i}^2}$ and $\frac{\partial^2 L(W)}{\partial w_i \partial w_j}$.

Here,

$$\begin{aligned}\frac{\partial^2 L(W)}{\partial {w_i}^2} &= -\sum_l X_i^l \frac{(1+\exp(w_0+\sum_{i=1}^n w_i X_i^l)).\exp(w_0+\sum_{i=1}^n w_i X_i^l).X_i^l - (\exp(w_0+\sum_{i=1}^n w_i X_i^l))^2.X_i^l}{(1+\exp(w_0+\sum_{i=1}^n w_i X_i^l))^2} \\ &= -\sum_l X_i^l.X_i^l \frac{\exp(w_0+\sum_{i=1}^n w_i X_i^l)}{(1+\exp(w_0+\sum_{i=1}^n w_i X_i^l))^2} \\ &= -\sum_l X_i^l.X_i^l\{P(Y^l=1|X^l,W)(1-P(Y^l=1|X^l,W))\}\end{aligned}$$

Similarly,

$$\frac{\partial^2 L(W)}{\partial w_i \partial w_j} = -\sum_l X_i^l.X_j^l\{P(Y^l=1|X^l,W)(1-P(Y^l=1|X^l,W))\}$$

In this case, a single Newton-Raphson updated rule is

$$w_i(t+1) = w_i(t) + \left(\frac{\partial^2 L(W)}{\partial w_i \partial w_i}\right)^{-1} \frac{\partial L(W)}{\partial w_i} \tag{19}$$

However, in practice the algorithm uses the partial second derivatives of the parameters in the Hessian matrix (H) to guide incremental parameter changes in an effort to maximize the log likelihood value for the likelihood function. The updated rule can be expressed compactly in matrix form as

$$W(t+1) = W(t) + H^{-1}.\mathbf{X}^T(\mathbf{Y}-\mathbf{P}) \tag{20}$$

where $H = [\mathbf{X}^T\mathbf{D}\mathbf{X}]$, $\mathbf{X}$ is the input data matrix of order $(m \times (n+1))$, $\mathbf{D}$ is the diagonal matrix of order $m \times m$ with l^{th} element $P(Y^l|X^l,W)(1-P(Y^l|X^l,W))$, $(\mathbf{Y}-\mathbf{P})$ is the matrix of order m with l^{th} element $(Y^l - P(Y^l=1|X^l,W))$. Here, the total number of training samples is assumed to be m and each row of $\mathbf{X}$ is a sample of size $(n+1)$ of the form $X = (1, X_1, X_2, \ldots X_n)$. A constant term 1 is introduced to the original X as a first component in order to handle matrix operation. For multi-class logistic regression model the updated rule for j^{th} class can be written in similar way as

$$W_j(t+1) = W_j(t) + H^{-1}.\mathbf{X}^T(\mathbf{Y}-\mathbf{P}) \tag{21}$$

where l^{th} element of $(\mathbf{Y}-\mathbf{P})$ is $(\delta(Y^l=y_j) - P(Y^l=y_j|X^l,W))$, $\delta(Y^l=y_j)=1$ if the l^{th} training value, Y^l, is equal to y_j and $\delta(Y^l=y_j)=0$ otherwise.

Here, the update rule from Newton Raphson method is similar with the gradient ascent method but, in addition, the update rule considers the second derivative for the weight update. The advantage of Newton Raphson is that it converges much faster towards the required solution. However, computing the Hessian for a large dataset is tedious and one of the shortcomings of this method.

4 Experiments

Experiments are performed with NB and LR classifiers on 21 datasets from the UCI [2] and KEEL [1] repositories. The datasets are chosen with both numerical and categorical attributes. A brief description of the experimental datasets is shown in Table 1. The following abbreviations are used to denote the classifiers with different learning circumstances: i) NB: general Naive Bayes, ii) NB-GNB: Naive Bayes under Gaussian assumption (variance (σ_i^2) depends on attribute (X_i) but not on class labels (Y)), iii) LR-GNB:

Logistic regression under Gaussian Naive Bayes assumption, iv) LR-GNB(OVO): Logistic Regression under Gaussian Naive Bayes assumption with one vs one (OVO) strategy, v) LR-GRAD: Logistic Regression with gradient ascent technique, vi) LR-GRAD(OVO): Logistic Regression with gradient ascent technique and OVO strategy and vii) LR-HESS: Logistic Regression with hessian matrix. In one vs one (OVO) strategy, a multiclass classifier is realized by combining several two-class classifiers. For a c-class problem, c(c - 1)/2 binary classifiers are constructed in this strategy. Here, the classifiers NB, NB-GNB, LR-GNB and LR-GNB(OVO) are built with generative learning technique where as LR-GRAD, LR-GRAD(OVO) and LR-HESS are all based on discriminative learning. The experimental results of all the classifiers on 21 datasets are shown in Table 2, where average classification accuracies are reported with 10-fold cross-validation. It is importantly noted that some preprocessing steps are applied to some datasets in order to make them compatible for the classifiers. From the experimental results (see, in Table 2), the following observations are pointed out: i) the average performance of LR classifiers is better than NB, ii) the discriminative learning classifiers produce better classification accuracies than generative learning classifiers, iii) under the same learning circumstances, OVO based methods perform better than without OVO (e.g., LR-GRAD vs LR-GRAD(OVO)), iv) the performance of NB-GNB and LR-GNB classifiers are almost similar. The statistical tests [4] are further performed to see if the above observations on our experimental datasets are significant or not.

4.1 Statistical tests for comparing classifiers

To compare the classifiers: LR-GRAD vs NB, LR-GRAD(OVO) vs LR-GRAD and LR-GNB(OVO) vs LR-GNB (for observations (i)-(iii)) over all datasets, Paired t-Test has been performed. The statistic for Paired t-Test is computed as

$$t = \frac{\sqrt{K}(\bar{D}_K - 0)}{s}, \tag{22}$$

where t follows the t-distribution with degree of freedom $K-1$, $\bar{D}_K = \frac{1}{K}\sum_{k=1}^{K} D_k$ is the sample mean (consistent and unbiased estimator of population mean), $s^2 = \frac{K}{K-1}\frac{1}{K}\sum_{k=1}^{K}(D_k - \bar{D}_K)^2$ is sample variance (consistent and as well as an unbiased estimate of population variance), $D_k = D_k^1 - D_k^2$ $(k = 1, 2, \ldots K)$. Here, D_k^1 and D_k^2 are the classification accuracies (10-fold cross validation) of two classifiers for k^{th} dataset and $K = 21$ is the total number of datsets. If the accuracies are same for two classifiers, then the expected mean $(\bar{D}_K)$ for all datasets will be zero. Therefore, our null hypothesis is that the distribution of D_k has 0 mean i.e., $H_0 : \mu = 0$ against the alternative, $H_1 : \mu \neq 0$. In particular, for $K = 21$, we accept the null hypothesis at 95% confidence level if $|t| < 2.086$ whereas at 90% confidence level if $|t| < 1.725$. For 21 datasets, the t values for the pair of classifiers (LR-GRAD, NB), (LR-GRAD(OVO), LR-GRAD) and (LR-GNB(OVO), LR-GNB) are obtained 1.7731, 2.1226 and $t = 0.7226$ respectively. These imply that LR-GRAD performs better than NB with confidence level 90%, LR-GRAD(OVO) performs better than LR-GRAD with confidence level 95%, however, the better performance of LR-GNB(OVO) than LR-GNB is not statistically significant.

In order to test whether two classifiers NB-GNB and LR-GNB are similar or not we perform McNemar's test (for observation (iv)). This statistic is computed as

$$\chi^2 = \frac{(|N_{01} - N_{10}| - 1)^2}{N_{01} + N_{10}}, \tag{23}$$

where χ^2 follows χ^2-distribution with degree of freedom 1, N_{10} is the number of samples that the first classifier classifies correctly but the second one classifies incorrectly, similarly, N_{01} is the number of samples that the second classifier classifies correctly but the first one classifies incorrectly. In this test, if the null hypothesis (i.e., the two classifiers are same) is correct for a given level of significance, say, 0.05, then the χ^2 (Equation 23) is less than or equal to $\chi^2_{1,0.95} = 3.841459$. For 21 datasets the χ^2 values are obtained as {0, 0, 0, 2.500000, 1.388889, 2.400000, 0, 20.275862, 110.035211, 0.307692, 0.410256, 0, 0.941176, 48.840074, 0, 0, 0.062500, 33.306931, 0.018519, 2.630435, 0.571429}. From the χ^2 statistic it is found that the performances of both classifiers are same for all datasets except four (#{8,9,14,18}).

Table 1: Description of the datasets

#	Data Set	# Classes	# Attrib.	Attrib. Types	# Instances	Source
1	Diabetes	2	8	Numeric	768	UCI
2	Fertility	2	9	Numeric	100	UCI
3	Breast Cancer	2	30	Numeric	569	UCI
4	Spam e-mails	2	57	Numeric	4601	UCI
5	Iris	3	4	Numeric	150	UCI
6	Seeds	3	7	Numeric	210	UCI
7	Wine	3	13	Numeric	178	UCI
8	Cardiotocography	3	21	Numeric	2126	UCI
9	Waveform	3	21	Numeric	5000	UCI
10	Thyroid	3	21	Numeric	7200	UCI
11	Vehicle	4	18	Numeric	846	KEEL
12	Page Blocks	5	10	Numeric	5473	UCI
13	Glass	6	9	Numeric	214	KEEL
14	SatImage	6	36	Numeric	6435	UCI
15	Zoo	7	16	Numeric	101	KEEL
16	Image Segmentation	7	19	Numeric	2310	UCI
17	Ecoli	8	7	Numeric	336	UCI
18	Pen-Based	10	16	Numeric	10992	KEEL
19	Handwritten Digits	10	64	Numeric	5620	UCI
20	English Vowel (Speaker)	11	10	Numeric	990	KEEL
21	Soybean	19	35	Categorical	683	UCI

Table 2: Experimental results with Naive Bayes and Logistic Regression

#	Data Set	NB	NB-GNB	LR-GNB	LR-GNB(OVO)	LR-GRAD	LR-GRAD(OVO)	LR-HESS
1	Diabetes	75.92	74.87	74.87	74.87	76.05	76.05	77.24
2	Fertility	87.78	88.89	88.89	88.89	90.00	90.00	86.81
3	Breast Cancer	92.86	92.85	92.97	92.97	93.02	93.02	93.42
4	Spam e-mails	81.46	87.32	87.45	87.45	88.43	88.43	74.40
5	Iris	94.67	86.67	84.67	92.00	95.33	96.00	95.33
6	Seeds	94.18	96.30	91.53	96.30	96.30	94.71	97.14
7	Wine	96.88	95.63	95.63	93.75	97.02	97.02	95.83
8	Cardiotocography	81.41	81.80	83.84	81.66	83.48	84.38	86.93
9	Waveform	80.96	81.08	78.55	80.66	80.04	81.79	86.79
10	Thyroid	92.76	93.95	94.04	95.21	94.44	97.34	95.36
11	Vehicle	46.22	45.00	44.39	44.20	75.69	76.46	81.71
12	Page Blocks	88.44	93.03	92.61	93.42	94.41	94.43	90.86
13	Glass	50.53	57.37	54.74	55.21	52.55	59.69	44.44
14	SatImage	79.14	78.75	76.33	76.16	75.62	80.20	81.58
15	Zoo	100.00	100.00	100.00	100.00	98.18	98.18	94.44
16	Image Segmentation	79.83	85.11	84.85	79.74	95.63	95.20	73.90
17	Ecoli	59.64	76.18	77.06	76.29	64.29	76.19	76.77
18	Pen-Based	85.74	82.29	81.75	81.28	90.19	94.52	94.42
19	Handwritten Digits	90.88	89.77	89.73	90.27	94.79	96.34	94.89
20	English Vowel (Speaker)	65.91	46.77	45.56	57.98	51.49	65.25	61.62
21	Soybean	88.23	86.69	86.23	80.23	92.38	86.79	77.46
Overall		81.59	81.92	81.22	81.84	84.73	86.76	83.87

5 Conclusion

In this paper, we describe the basic theoretical concepts of learning techniques for Naive Bayes and Logistic Regression classifiers. We also describe the algorithms in a simple way in order to minimize the gap between the theoretical concepts and practical implementations. We build several classifiers under different learning circumstances and their performances are reported in Table 2. It is observed that the

performances of all the classifiers based on discriminative learning are better than the classifiers based on generative learning. In general, discriminative learning algorithm uses the training data to create a decision boundary for separating any class from other corresponding competitive classes. On the other hand, generative learning creates the model on the basis of the distribution of individual class. Moreover, in this study, the parameters in discriminative learning are learnt directly from input to the class label without any prior assumption where as in generative learning the parameters are learnt under some certain prior assumptions. These are the key reasons for which LR-GRAD, LR-GRAD(OVO) and LR-HESS perform better than NB, NB-GNB, LR-GNB and LR-GNB(OVO). Another important observation is noted that NB-GNB has given less accuracy than NB, this is due to the choice of class independent variance (σ_i^2) in NB-GNB. So, the variance in NB plays an important role in classification. Here, we also see that in different datasets, NB-GNB and LR-GNB produce almost the same results which indicate that both classifiers are likely to be identical. In fact, in literature [7], it has been proven that both NB (under Gaussian assumption) and LR-GNB are asymptotically (as the number of training examples grows toward infinity) identical classifiers. Finally, we observe that the classification accuracy with the models based on OVO are better than without OVO. This is induced by the fact that the decision boundary which is constructed based on OVO has more discriminating power than one vs others (OVA) strategy which is implicitly used here for discriminating learning. Now, if we consider LR-GRAD and LR-HESS classifiers, the advantage of LR-HESS learning is that since it is used second order information of parameters, it leads to meet directly towards the convergent point and unlikely to stuck at local extreme points. However, the second order derivative is more expensive to compute and there exists a well known problem in computing inverse of hessian matrix. Because of the problem in computing inverse sometimes LR-HESS performs unexpectedly the worst (marked with gray color in Table 2). Finally, a question can arise which classifier would be chosen for a real application. This is probably difficult to suggest as it completely depends on the data and the objective of the application as well. In general, between LR-GRAD and LR-HESS, LR-GRAD is a better choice to avoid an extra risk of computing inverse. Where as if the data is high dimensional, NB is a better choice because of estimating the parameters in LR (GRAD or HESS) is impracticable.

References

[1] J Alcalá, A Fernández, J Luengo, J Derrac, S García, L Sánchez, and F Herrera. Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework. *Journal of Multiple-Valued Logic and Soft Computing*, 17(2-3):255–287, 2010.

[2] K. Bache and M. Lichman. Uci machine learning repository, 2013.

[3] Scott A. Czepiel. Maximum likelihood estimation of logistic regression models: Theory and implementation. http://czep.net/stat/mlelr.pdf.

[4] Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7:1–30, 2006.

[5] Tom M. Mitchell. *Machine Learning*. McGraw Hill, 1997.

[6] Tom M. Mitchell. Generative and discriminative classifiers: naive bayes and logistic regression, 2005., 2010.

[7] Andrew Y. Ng and Michael I. Jordon. On discriminative vs. generative classifiers: a comparison of logistic regression and naive bayes. In *Advances in Neural Information Processing Systems (NIPS) 14*, 2002.

[8] H. Zhang. The optimality of naive Bayes. In *Proceedings of the 17th FLAIRS conference, AAAI Press*, 2004.