A Bias-Variance Based Heuristic for Constructing a Hybrid Logistic Regression-Naïve Bayes Model for Classification

Yi Tan[1] and Prakash P. Shenoy[1]
vieiraty0113@gmail.com, pshenoy@ku.edu

School of Business, University of Kansas, Lawrence, KS 66045 USA

Abstract. Discriminative classifiers tend to have lower asymptotic classification errors, while generative classifiers can be more accurate when the training set size is small. In this paper, we examine the construction of hybrid models from categorical data, where we use logistic regression (LR) as a discriminative component, and naïve Bayes (NB) as a generative component. We adopt a bias-variance tradeoff based strategy, with the objective of minimizing the sum of these two errors. Specifically, the proposed heuristic consists of functions of training sample size and conditional dependence among features. These functions serve as proxies for model variance and model bias. We implement our method on 25 different classification datasets, and find that the hybrid model does better than pure LR and pure NB. Our proposed method is competitive with random forest. Although the hybrid model fails to beat LASSO in predictive performance, as suggested by the experimental results, the difference appears to be insignificant when the number of features is small. Also, the hybrid model requires less training time than LASSO, which makes it more attractive when the training time is a big concern.

Keywords: logistic regression, naïve Bayes, hybrid discriminative-generative model, bias-variance strategy, model construction heuristic

1 Introduction

For classification problems, people are often faced with a choice between a generative and a discriminative classifier. Generative classifiers learn the joint probability distribution $P(F_1, \ldots, F_n, C)$ of the features $F_1, \ldots, F_n$, and the class C, make their predictions by using Bayes rule to compute $P(C \mid F_1, \ldots, F_n)$, and then predict a label with the highest posterior probability. In contrast, discriminative classifiers directly learn the conditional probability $P(C \mid F_1, \ldots, F_n)$, without assuming anything about the feature distribution, $P(F_1, \ldots, F_n)$. When training data are large, discriminative classifiers often achieve better prediction performance than generative classifiers, and hence are widely preferred. However, generative classifiers often have better performance when the size of training data is small [13]. Also, generative classifiers are more tolerant of missing values than discriminative classifiers.

To take advantage of both worlds, this paper investigates the construction of hybrid models from categorical data where we use logistic regression (LR) as a discriminative component, and naïve Bayes (NB) as a generative component, for datasets in the general domain. Both LR and NB belong to the family of probabilistic classifiers, and form a well known discriminative-generative pair [18]. Because LR and NB models have few parameters, they scale well to high dimensions, and can be trained very efficiently. It has been shown that LR can be modeled as a Bayesian

network [17]. The hybrid LR-NB model, first proposed by Kang and Tian [10], is recognized as a restricted class of Bayesian network classifier that combines LR and NB in a graphical way. The task we are concerned with is *learning Bayesian network structures*, i.e., deciding to which part of the hybrid model a given feature should be assigned. Kang and Tian [10] use a greedy method based on in-sample classification accuracy. We argue that this method may yield a local optimal solution. Kang and Tian provide only one-round cross validation result for each dataset. We conjecture that this resulting hybrid model may not perform well in terms of average out-of-sample classification accuracy (with a suitably large number of trials). Also, their method is time intensive.

In this paper, we propose a more efficient model construction heuristic that balances the tradeoff between model bias and model variance. Specifically, LR produces the lowest prediction error among all linear classifiers by achieving the lowest bias if there are sufficient training data. However, this is not the case when training set size is limited. LR estimates may overfit the data, which makes the prediction less accurate due to high variance. On the other hand, NB overcomes the overfitting issue by making a strong assumption of conditional independence, and thus learns each parameter using the entire training sample. This makes NB work surprisingly well for small datasets. However, as the conditional independence assumptions rarely hold in practice, NB estimates are often suboptimal due to the introduction of bias in comparison to the situation where the conditional independence assumptions are satisfied. Our heuristic consists of functions of training sample size and conditional dependence among features. These functions serve as proxies for model variance and model bias. For each given feature, we estimate the bias and the variance introduced if the feature is assigned to the LR or to the NB part of the hybrid model, respectively. By minimizing the sum of bias and variance errors, we assign the feature to the corresponding part. If all features are assigned to the LR (NB) part, the resulting model is pure LR (NB), otherwise the resulting model is strictly hybrid. Our heuristic can be regarded as a selection mechanism that helps make the choice between pure LR, pure NB and strictly hybrid models.

We conduct experiments on 25 different machine learning datasets from UCI Machine Learning Repository. We select these datasets such that we have a diversity of sample sizes, number of features, and number of classes. We compare the 0-1 loss and root mean square error (RMSE) of hybrid model with pure LR, pure NB, random forest (RF), and LASSO, which are widely recognized as state-of-the-art classifiers, using paired t-tests with 0.05 significance level. Experimental results show that the hybrid model constructed using our heuristic achieves a more accurate classification performance than both pure LR and pure NB models. Specifically, the Win/Draw/Loss (W/D/L) of the hybrid model versus pure LR (pure NB) in terms of 0-1 loss and RMSE are 6/19/0 (18/2/5) and 10/14/1 (18/1/6), respectively. Also, the hybrid model is competitive with RF. It has a higher 0-1 loss (W/D/L=5/7/13), but lower RMSE (W/D/L=14/2/9). LASSO is difficult to beat, in terms of both 0-1 loss (5/10/10) and RMSE (1/14/10), in general. However, experimental results suggest that the difference appears to be insignificant when the number of features is small. For example, the W/D/L for 0-1 loss is 3/7/4, and for RMSE is 0/12/2 on datasets with fewer than 10 features. If we have a large number of features, estimating the parameters of a pure LR model by maximizing the conditional log-likelihood can be computationally intensive. We also compare the training time, showing that the hybrid model is more efficient than pure LR by assigning some of the features to the NB part. Also, the training time for hybrid models is much lower on an average than the corresponding RF and LASSO models.

An outline of the remainder of the paper is as follows. In Section 2, we describe related work on hybrid discriminative/generative classifiers, and state the contributions of our paper. Section 3 sketches and compares the LR and the NB models. In Section 4, we describe the hybrid model.

Section 5 describes our method for constructing a hybrid model. Section 6 shows the empirical results from our experiments using 25 datasets from the UCI Machine Learning Repository. Finally, in Section 7, we summarize and conclude.

2 Related Literature

Our paper is related to the literature on the comparison of discriminative and generative classifiers, which has a long history. Efron [3] presents theoretical and simulation studies showing that linear discriminant analysis (LDA), a generative classifier, is asymptotically (between one third and one half) more efficient than LR if the model is correctly specified. On the other hand, Ng and Jordan [13] do an empirical and theoretical study of LR and NB models for classification. They find that there are two distinct regimes of prediction performance with respect to training set size. Particularly, an LR model has a lower asymptotic error compared to NB, but an NB model approaches its asymptotic error much faster than an LR model. In other words, for large training datasets, LR classifiers have higher accuracies, whereas for small training datasets, NB classifiers may have higher accuracies than LR. In our construction of a hybrid model, we make use of results discussed by Ng and Jordan [13] for penalizing the model complexity. Xue and Titterington [22] conduct empirical and simulation studies, as a complement to Ng and Jordan's work. However, they find no convincing evidence to support Ng and Jordan's claim.

In the last two decades, several researchers have been exploring hybrid models that combine discriminative and generative models into one model [1, 8, 10, 16, 18, 21]. These are discussed in the following paragraphs.

Rubinstein and Hastie [18] are among the earliest to suggest combining discriminative and generative models. They suggest that features that satisfy the assumption of a generative model be retained in the generative part, with the remaining moved to the discriminative part. They compare linear discriminant analysis (LDA), a generative model, with LR, a discriminative model, for three different simulated datasets, and discover that LDA does better than LR when the class densities are Gaussian, and vice-versa. They also compare NB, a generative model, with generalized additive model (GAM), a discriminative model, for a simulated dataset with log-spline density that satisfies the assumptions of the GAM model. Asymptotically, the GAM model achieves a lower error rate than the NB model. However, when the training set is a small subset (25 observations) of the entire dataset, NB model has a lower average error than GAM. While they propose combining the two approaches, they do not describe any experimental results.

Raina *et al.* [16] investigate a hybrid model with LR as the discriminative component, and NB as the generative component, in the context of text classification problems. They run experiments using pairs of newsgroups from a dataset of USENET new postings, where the documents have two disjoint regions—a subject region and a message body region. An NB model treats the two regions in exactly the same way due to the strong conditional independence assumptions of an NB model. A hybrid model treats the two regions differently using different weight parameters for each. As the subject region has fewer words than the message body region, the words in the subject region are weighted higher than the words in the message body region. Depending on how the weight parameters are estimated from a dataset, the hybrid model reduces to an LR model. Experimental results show that hybrid models can provide lower test error than either pure LR or pure NB. Fujino *et al.* [8] investigate hybrid discriminative-generative classifiers similar to [16] for text classification having multiple components (such as titles, hyperlinks, anchor text, images, etc.). They use a generative classifier for each component that are then combined using weights learnt

using the maximum entropy principle. They do an empirical evaluation on four text-classification datasets, and find that hybrid classifiers outperform pure NB and pure LR models.

In an attempt to benefit from the advantages of both generative and discriminative approaches, Bishop and Lasserre [1] propose a heuristic that interpolate between these two extremes by taking a convex combination of the generative and discriminative log-likelihood functions. They apply their approach to object recognition in static images. Each image has two sets of features—observable features, and latent patch labels—in addition to class. They compare the performance of hybrid models with different combination weights and find that the best performance is obtained with a blend of generative and discriminative extremes.

The studies described above all focus on one specific domain. Our paper differs from these prior works in that we examine the construction of the hybrid model that is applicable in general for any domain.

Zaidi *et al.* [24] [23] discuss a weighted variant of NB with the goal of alleviating the feature conditional independence assumption of NB, or using the maximum likelihood parametrization of NB to pre-condition the discriminative search of LR. By exploiting the direct equivalence between a weighted NB and LR, they introduce a hybrid discriminative-generative estimation approach, i.e., minimization of either the negative conditional log-likelihood, or the mean squared error objective functions. As a result, the weighted NB learns models that are exactly equivalent to LR, but computationally much more efficient. Experimental results suggest that the resulting weighted NB is a competitive alternative to state-of-the-art classifiers, such as random forest, LR, and A1DE (Average One-Dependence Estimators). Following the same intuition, Zaidi *et al.* [25] introduce a hybrid discriminative-generative approach, called *accelerated logistic regression* (ALR), to train LR with high-order features. They show that ALR significantly improves the efficiency of LR. Moreover, by incorporating higher order features to reduce the bias, ALR predicts well as compared to state-of-the-art classifiers including random forest and average n-dependence estimators, especially on large datasets. All of these methods are driven by the fact that weighted NB has an equivalent functional form compared to LR. The model approach is hybrid in the sense of estimating a generative model discriminatively. In contrast, our focus is on a restricted Bayesian network classifier, which combines two probabilistic models in a graphical way.

One of the closest to our work is that by Xue and Titterington [21]. They study hybrid discriminative-generative classifiers where the discriminative component is LR, and the generative component is Fisher's linear discriminant analysis (LDA). They test all features for univariate normality, and those that fail the test are assigned to the LR portion of the hybrid model. Xue and Titterington test their algorithm for 6 datasets that have only numeric features. They find that for smaller sizes of the training set, the hybrid model does better than the pure LR and pure LDA models. Our focus is on classification tasks for categorical data. Instead of using LDA, which is applicable only for numeric features, we use NB, which is well suited for categorical features, as the generative component. Although NB can also be used for numeric features, it involves making distributional assumptions for the conditional distribution of a feature given the class, and this makes a formal comparison messy.

Our study contributes to the literature by providing a new method on construction of such a hybrid discriminative-generative classifier where the discriminative component is LR, and the generative component is NB, first introduced by Kang and Tian [10]. Kang and Tian learn a hybrid model by starting with an empty discriminative component, and then greedily add one feature at a time (which results in the maximum in-sample classification accuracy gain) to the discriminative component until the in-sample classification accuracy does not improve. They test their algorithm

for 20 different datasets, which are pre-processed so that there are no missing values, and all features are categorical. They measure classification accuracy using either 10-fold cross-validation (for small datasets) or 3-fold cross-validation (for large datasets). This is done just once, so they get a point estimate of the classification accuracy. The average point estimate of the classification errors for all 20 datasets is lowest for the hybrid LR-NB model. However, the average classification accuracy (with a suitably large number of trials) is not captured, in terms of which the resulting hybrid model may not outperform the benchmark models because it yields a local optimal solution. Also this method is computationally intensive.

The strategy of a recent work by Tan *et al.* [19] for constructing such a hybrid model is based on reducing the conditional dependence of features in the NB part of the hybrid model. They estimate the normalized conditional mutual information ($norMI$) given the class variable for all pairs of features, and find a pair of features with highest $norMI$ (using 0.05 as a cutoff point). Then for each of these two features, the authors compare their second highest $norMI$ and move the one with higher value to the LR part. This strategy deals with the model bias, but ignores the impact of model variance. Consequently, the resulting hybrid model outperforms the pure NB model, but does worse than the pure LR model in the pairwise comparison. In this paper, we construct the hybrid model by balancing the tradeoff between model bias and model variance. The proposed heuristic consists of functions of training sample size and conditional dependence among features. These functions serve as proxies for model variance and model bias,and our objective is to minimize the sum of these two errors.

The contributions of this work are as follows:

- We investigate the strengths of hybrid LR-NB models by reviewing the literature on comparison between LR and NB.
- We propose an efficient heuristic for constructing a hybrid LR-NB model. Experimental results show that the heuristic is effective in improving the classification performance of hybrid model compared to pure LR and pure NB. Also, the resulting hybrid model is comparable in classification accuracy to random forest. Although it fails to beat LASSO in general, experimental results suggest that the difference in predictive performance between hybrid model and LASSO appears to be insignificant when the number of features is small. Also, the training time for hybrid models is less on an average than corresponding LR, RF, and LASSO models.
- We propose a novel idea for balancing the bias-variance tradeoff. Compared to traditional bias-variance techniques, which add a regularizer to a loss function, our method proposes proxies for both bias and variance, and then minimizes the sum of these two errors.

3 LR versus NB

3.1 Logistic Regression

In this subsection, we discuss LR as a classification method for categorical data. Suppose we seek to assign a class label $c \in \Omega_C$ of the class variable C to instances described by a set of n categorical features $\mathbf{F} = (F_1, \ldots F_n)$ defined on the probabilistic space χ. For simplicity of exposition, we assume that $F_1, \ldots, F_n$ are all Boolean. If feature F_j is not Boolean, i.e., has k_j states with $k_j > 2$, we can represent F_j by $k_j - 1$ Boolean features $F_{j2}, \ldots, F_{jk_j}$ where $F_{jt} = 1$ if variable F_j takes state t and $F_{jt} = 0$ otherwise, $t = 2, \ldots, k_j$.

The LR model is a discriminative classifier that directly learns the conditional probability $P(C \mid \mathbf{F})$ by assuming that the log odds for a class c_j is a linear function of the features:

$$\ln\left(\frac{P(C = c_j \mid \mathbf{f})}{1 - P(C = c_j \mid \mathbf{f})}\right) = \beta_{0\,j} + \sum_{i=1}^{n} \beta_{i\,j} f_i \tag{1}$$

where $\mathbf{f} = (f_1, \ldots f_n)$.

We can derive the parametric form for the distribution $P(C \mid \mathbf{F})$ by rewriting Eq. (1) as:

$$P(C = c_j|\mathbf{f}) = \frac{\exp(\beta_{\mathbf{0\,j}} + \sum_{\mathbf{i=1}}^{\mathbf{n}} \beta_{\mathbf{i\,j}} \mathbf{f_i})}{\sum_{\mathbf{k=1}}^{\mathbf{c}} \exp(\beta_{\mathbf{0\,k}} + \sum_{\mathbf{i=1}}^{\mathbf{n}} \beta_{\mathbf{i\,k}} \mathbf{f_i})} \tag{2}$$

Notice that for a dataset that has a class variable with q classes, we have $(q-1)\cdot(n+1)$ parameters. The small number of parameters is one reason for the simplicity and robustness of the LR classifier. Using Eq. (2), we can compute the probability distribution of classes in C. The predicted class is the one with the highest probability.

LR parameters are usually estimated by maximizing the conditional likelihood, i.e., choosing parameters $\mathbf{B}$ that satisfy $\mathbf{B} \leftarrow \arg\max_{\beta} \prod P(C = c_j \mid \mathbf{f}, \beta)$, where $\beta = (\beta_0, \ldots, \beta_n)$. As there is no closed form solution with respect to $\mathbf{B}$, one common approach is to use gradient-based methods. Ng and Jordan [13] show that the prediction performance of LR converges to the best performance of all linear classifiers as the training sample size goes to infinity. However, with a small training sample size, LR estimates may overfit the data, which makes the prediction less accurate. Also, the gradient-based method can be computationally intensive especially for a large number of parameters of interest. For datasets with large number of features, any speed-up to the parameter learning process may be highly desired.

Rijmen [17] models an LR model as a Bayesian network, where Eq. (2) constitutes the conditional probability distribution for C given $\mathbf{F} = (F_1, \ldots F_n)$. LR assumes a parametric form for the distribution $P(C|\mathbf{F})$, and has its model structure as shown in Fig. (1). In this figure, the dotted oval around the features denotes that the Bayesian network structure of the feature variables is not represented, as it is irrelevant to C, assuming that we have observed values of all features.

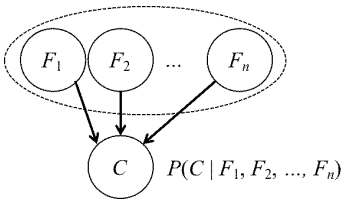

Fig. 1. An LR Model as a Bayesian Network

3.2 Naïve Bayes

In this subsection, we discuss NB model as a classification method for categorical data. Suppose C is the class variable, whose value we wish to predict based on observation of a set of m categorical features $\mathbf{E} = (E_1, \ldots E_m)$.

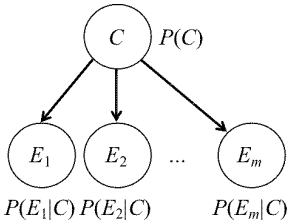

Fig. 2. An NB Model as a Bayesian Network

The NB model is a generative classifier that learns the joint probability distribution $P(C, \mathbf{E})$ by making an assumption that features $\mathbf{E} = (E_1, \ldots E_m)$ are mutually conditionally independent given the class variable C. Fig. 2 is a Bayesian network depiction of an NB classifier. Using Bayes rule, it can be shown that

$$odds(C = c_j \mid \mathbf{e}) = odds(C = c_j) \prod_{i=1}^{m} lr(e_i, C = c_j), \tag{3}$$

where $\mathbf{e} = (e_1, \ldots e_n)$ and $lr(e_i, C = c_j) = \frac{P(e_i|C=c_j)}{P(e_i|C\neq c_j)}$ is the likelihood ratio for $C = c_j$ based on the observed value $E_i = e_i$. In words, the posterior odds of $C = c_j$ is equal to prior odds of $C = c_j$ times the likelihood ratios of observed features for $C = c_j$. If a feature is not observed, we can regard its likelihood ratio as equal to 1.

We can also derive the parametric form of the posterior probability $P(C \mid \mathbf{E})$ from Eq. (3) as

$$P(C = c_j \mid \mathbf{e}) = \frac{P(C = c_j) \prod_{i=1}^{m} P(e_i \mid c_j)}{\sum_{k=1}^{c} P(C = c_k) \prod_{i=1}^{m} P(e_i \mid c_k)} \tag{4}$$

As we are interested in the most probable value of C, we have the classification rule for NB as:

$$C \leftarrow \arg\max_{c_j} \frac{P(C = c_j) \prod_{i=1}^{m} P(e_i \mid c_j)}{\sum_{k=1}^{c} P(C = c_k) \prod_{i=1}^{m} P(e_i \mid c_k)} \tag{5}$$

The conditional independence assumption reduces the complexity of the NB model (number of parameters), which makes it a robust model. Specifically, if the class variable C has q classes, and features $E_1, \ldots, E_m$ are all binary, the number of parameters to be estimated is $q\,m+q-1$. Also, the conditional independence assumption helps overcome the overfitting issue by making NB estimate its parameters using the entire training sample. In spite of its simplicity and strong conditional independence assumption, NB performs surprisingly well, even against other more sophisticated classifiers, especially when the training set size is small.

However, conditional independence assumption rarely holds in practice, and as a consequence an NB model may not predict well. A great amount of literature addresses approaches to improving the prediction performance of NB by either relaxing the conditional independence assumptions between features given the class label [4,7,11,15,19,27] or by weighting the features [6,9,24]. In the next section, we will describe a hybrid LR-NB classifier that relaxes the conditional independence assumptions in NB by assigning mutually conditionally dependent features to the LR part of the hybrid model.

3.3 Comparison of LR and NB

In supervised learning, the prediction error for a given machine learning algorithm can always be broken down into three parts: bias, variance, and irreducible error. Irreducible error is associated with the inherent variability in the data, and there is nothing one can do about it. Thus, an effective learning algorithm should minimize the sum of bias and variance errors.

The *bias* error is the result of erroneous assumptions in the given learning algorithm. A high bias learner is usually less flexible, has a simpler functional form, and can be trained more efficiently (than low bias learners). On the other hand, the *variance* reflects the sensitivity of learning algorithm to the changes in the training dataset. A high variance learner is usually more flexible, has a more complex functional form, and is more likely to overfit the training data (than low variance learners). Reducing the bias usually results in increasing the variance, and vice-versa. As there is no way out of this inverse relationship between bias and variance, we need to balance the trade-off between these two errors. Such bias-variance tradeoff forms the conceptual basis for many regularization methods such as LASSO and ridge regression.

NB is a learning algorithm with lower variance, but higher bias, in comparison to LR. First, we can show that the conditional independence assumption of NB implies the same parameteric form of $P(C \mid \mathbf{E})$ as $P(C \mid \mathbf{F})$ in LR. To maximize the conditional likelihood of LR and NB using Eqs. (2) and (5) respectively, we get a direct equivalence between LR and NB as $e^{\beta_{0,j}} \propto P(C = c_j)$ and $e^{\beta_{i,j}} \propto P(e_i \mid c_j)$. This equivalence suggests that LR and NB have the same hypothesis space, and asymptotically converge toward identical classifiers assuming that the conditional independence assumptions of NB holds. As a result, they tend to produce the same classification error as the number of training examples approach infinity. However, when the conditional independence assumption does not hold, it introduces bias and consequently NB parameter estimates are suboptimal, i.e., the asymptotic classification error for LR is often lower than that of NB.

On the other hand, the conditional independence assumption makes NB estimate the parameters over the entire sample, thus exhibits low variance by preventing it from overfitting. Ng and Jordan [13] also show that, the generative NB converges to its asymptotic error more quickly than the discriminative LR. Accordingly, the low-variance learner NB is expected to achieve lower error compared to LR, when the training set size is small.

4 A Hybrid LR-NB Model

In this section, we discuss a hybrid LR-NB model (hybrid, in short) as a classification method for categorical data. The graphical structure of a hybrid model represented as Bayesian network is shown in Fig. 3. Node C is the class variable, whose value we need to predict based on observation of two sets of features: $\mathbf{F} = (F_1, \ldots F_n)$, the parents of C in Fig. 3, called the LR part, and $\mathbf{E} = (E_1, \ldots E_m)$, the children of C, called the NB part. As in Section 3, we assume that $F_1, \ldots F_n$ are all Boolean.

The conditional independence assumptions of a hybrid model are as follows. First, the features in the LR part of the model are conditionally independent of the features in the NB part given the class variable C. Second, the features in the NB part of the model are mutually conditionally independent given the class variable C.

One implication of the first conditional independence assumption is that to learn the parameters of the conditional distribution of C given the features in the LR part, the features in the NB part are irrelevant for this task. Thus, one can use standard LR parameter estimation methods to learn

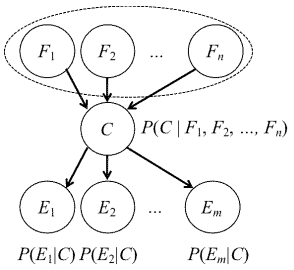

Fig. 3. A Hybrid LR-NB Model as a Bayesian Network

these parameters. Similarly, to learn the parameters of the NB part of the hybrid model, the features in the LR part are irrelevant for this task, and thus, we can use standard NB parameter estimation methods for learning these parameters.

Making inferences in a hybrid model is easy. Using variable elimination [26], after we eliminate the observed features in the LR part, the posterior distribution of the class variable C is given by the LR model:

$$\ln(odds(C = c_j|\mathbf{f})) = \beta_{0,j} + \sum_{i=1}^{n} \beta_{i,j} f_i$$

where $\mathbf{f} = (f_1, \ldots, f_n)$, f_i is the observed state of feature F_i in the LR part, and $\beta_{i,j}$ are the parameters of the LR model.

This gives the posterior odds of $C = c_j$ given $\mathbf{f} = (f_1, \ldots, f_n)$ as:

$$odds(C = c_j|\mathbf{f}) = \exp(\beta_{0,j} + \sum_{i=1}^{n} \beta_{i,j} f_i) \tag{6}$$

After elimination of the features $\mathbf{F}$ in the LR part, what is left is an NB model such that the prior distribution of C (defined as the posterior distribution of C given $\mathbf{F} = \mathbf{f}$) is as given in Eq. (6). Thus, we can now compute the posterior distribution of C given $\mathbf{F} = \mathbf{f}$ and $\mathbf{E} = \mathbf{e}$ using the NB model as follows:

$$odds(C = c_j \mid \mathbf{e}, \mathbf{f}) = \exp\left(\beta_{0,j} + \sum_{i=1}^{n} \beta_{i,j} f_i\right) \prod_{k=1}^{m} lr(e_k, C = c_j) \tag{7}$$

where $\mathbf{e} = (e_1, \ldots, e_m)$, and $lr(e_k, C = c_j) = \frac{P(e_k|C=c_j)}{P(e_k|C\neq c_j)}$ is the likelihood ratio for $C = c_j$ based on the observed state $E_k = e_k$. Similar to NB model, if a feature in the NB part is not observed, we can regard its likelihood ratio as equal to 1.

Eq. (7) is used for making inferences from a hybrid classifier, which estimates the probability that the class variable C will take the value of $c_1, \ldots, c_q \in \Omega_c$ given the observed values of all features.

A hybrid model has an advantage of reducing the prediction error by relaxing the conditional independence assumptions of an NB model. From Fig. 3, the LR part of hybrid model does not require a full model structure, and hence makes no conditional independence assumptions among its features. Given a set of features that are highly conditionally dependent on each other given the class variable, we can assign them to the LR part, as the conditional likelihood maximization

algorithm for LR part can easily adjust its parameters to maximize the conditional likelihood of the data, hence reduce the bias of the hybrid model.

Apart from relaxing the conditional independence assumption of an NB model, hybrid model retains the simplicity of LR and NB models. Assuming all the features in both LR and NB parts of the model are binary-valued, then the number of parameters in the hybrid model is $(q-1)\,(n+1)+q\,m$, where n is the number of features in the LR part of the model, m is the number of features in the NB part of the model, and q is the number of classes of the class variable.

As a combination of LR model and NB model, the hybrid model is considered to be more flexible in terms of training set size. When the training sample is large, we tend to assign features to the LR part to reach the lowest bias among all linear classifiers. However, when the training sample is small, we may sacrifice some bias to achieve a lower variance by assigning features to the NB part. When the training sample is neither too large nor too small, we may assign some features to the LR part, and some features to the NB part by balancing the bias-variance tradeoff with the consideration of their conditional dependence with other features.

Finally, pure LR model can be computationally intensive, as its training time grows exponentially with the number of parameters to be estimated. The hybrid model is capable of reducing the LR's computational complexity by assigning some of the features to the NB part when we have a large number of features. Consequently, there are fewer parameters in the LR part to be estimated. Note that learning NB parameters does not require any optimization, thus offers a much faster training step as compared to LR. On the other hand, because it is just as easy to make inferences from a hybrid model as from pure LR and pure NB models, the hybrid model is computationally efficient at classification time.

Constructing a hybrid model is strictly more general than making the selection between a pure LR and a pure NB model. A hybrid model is identical to LR if all its features are assigned to the LR part, and is identical to NB if all its features are assigned to the NB part. However, other feature assignments can result in classifiers that differ from both pure LR and pure NB. Later, we will show that by implementing our hybrid model construction heuristic, the hybrid model is capable of outperforming both pure LR and pure NB in many cases.

5 A Method for Constructing a Hybrid LR-NB Model

The main focus of this section is on construction of a hybrid model, with LR as the discriminative component, and NB as the generative component, that predicts well. As there are 2^{m+n} possible hybrid model structures, where $m+n$ is the total number of features in the hybrid model, searching the space of all possible hybrid models is computationally intractable for large values of $m+n$.

To the best of our knowledge, not much work has been done to address this issue. Kang and Tian [10] use a greedy method by starting with all features in the NB part, and they move one feature at a time to the LR part until the in-sample classification accuracy does not increase. One problem with this approach is that it may yield a local optimal solution. Also, this method is time intensive. Tan *et al.* [19] select the NB part based on the conditional independence relations of pairs of features. Their strategy tries to control for the model bias, however, it ignores the impact of model variance. As a result, their heuristic fails to outperform pure LR.

In this section, we propose a new heuristic to select the LR and NB parts for a hybrid model. Our heuristic only decides to which part a variable should be assigned. It does not serve as a feature selection process, i.e., we use all features in the datasets to train a hybrid model.

5.1 Feature Evaluation

In this section, we adopt a bias-variance tradeoff based strategy to decide whether we should assign a given feature to the LR part, or to the NB part, of a hybrid model. Traditional bias-variance techniques, such as LASSO and ridge regression, balance the tradeoff by adding a regularizer to a loss function. Our method differs from them in that we use a proxy for relative bias and relative variance of assigning features to either the LR part or to the NB part. Thus, we construct a hybrid model by evaluating each feature to see if it favors high-bias, low-variance NB part or high-variance, low-bias LR part.

At the heart of our model construction method are two proxies for relative bias and relative variance of assigning features to either the LR or the NB part of a hybrid model, respectively, based on the following observations:

1. *Exponential Decrease*: As the size of the training set increases, the prediction error for both LR and NB model decreases exponentially.
2. *Convergence Rate*: NB's strong assumption of conditional independence allows NB to estimate the parameters using the entire training sample. This prevents NB from overfitting to the training data, and makes NB converge to its asymptotic error more quickly than LR [13]. As a result, NB tends to produce lower variance.
3. *Identical Classifier*: Assuming that the NB model's conditional independence assumption holds, LR and NB converge toward identical classifiers as the number of training instances tends to infinity.
4. *Conditional Independence*: A hybrid model assumes that features in the LR part are conditionally independent of the features in the NB part given the class variable, and that all features in the NB part are mutually conditionally independent given the class variable. In other words, any feature in the NB part is assumed to be conditionally independent with all other features in the hybrid model, and thus the violation of such conditional independence assumption will result in a bias.

First, we define $\bar{r}_i$ as the average conditional dependence given class variable C of feature F_i with all other features. We use it as a proxy for relative bias error induced by assigning feature F_i to the NB part, as compared with assigning the feature to the LR part, of a hybrid model: $B(F_i) = \bar{r}_i$. The prediction performance of LR converges to the best performance of all linear classifiers as the training sample size goes to infinity, i.e., LR produce the lowest bias among all linear classifiers; while, as per observations *Identical Classifier* and *Conditional Independence*, if we assign a feature to the NB part of a hybrid model, it induces some additional bias by violating the conditional independence assumptions of the hybrid model. In this paper, we use normalized conditional mutual information as a measure of conditional dependence.

Next, we use the training set size as a proxy for relative variance error induced by assigning feature F_i to the LR part, as compared with assigning the feature to the NB part, of a hybrid model. Specifically, we adopt an exponential decay function relating the total number of instances in the training set, N, to the relative variance of the learning algorithm: $V(F_i) = e^{-\lambda N}$, following *Exponential Decrease*. It helps aviod the high influence of the sample size for large datasets. This function form is also consistent with the observation *Identical Classifier* as it converges to zero when N goes to infinite, and with the observation *Convergence Rate* as we set $\lambda > 0$.

Notice that both LR and NB models can handle numeric features. However, in the case of an NB model, we need to make distributional assumptions for the conditional distributions of numeric

features given the class variable. Thus any violation of such distributional assumptions will result in additional bias. Similarly, regarding missing data, in the case of an LR model, we need to either ignore the corresponding instances, or impute the data, for example, with expectation-maximization algorithm or Markov chain Monte Carlo approaches. Both listwise deletion and data imputation result in additional variance. These add complexity to our heuristic. Therefore, we only focus on categorical data with no missing values in this paper. In addition, we do not apply any regularization in LR part of the hybrid model, because it increases the bias and reduces the variance of the LR part by shrinking its coefficients towards zero. Our approach controls the model variance by assigning features towards the NB part of the hybrid model.

5.2 Model Construction Algorithm

The previous paragraph describes the basic idea of the proposed heuristic in an intuitive manner. Following the bias-variance tradeoff strategy, we define $e(F_i)$ as a proxy for the relative model reducible error if F_i is assigned to the LR part, as compared with assigning the feature to the NB part, of a hybrid model:

$$\begin{aligned} e(F_i) &= V(F_i) - B(F_i) \\ &= e^{-\lambda N} - \bar{r}_i \end{aligned} \tag{8}$$

Our heuristic criterion is described in Algorithm 1. First, we calculate the index of relative reducible errors, $e(F_i)$, for each of the features $F_1, \ldots, F_n$ using Eq. (8). Next, for any given feature F_i, if $e(F_i) \leq 0$, then F_i is assigned to the LR part. Otherwise, F_i is assigned to the NB part. We determine the value of the tuning parameter $\lambda \geq 0$ using cross-validation, which is described in Section 6.1.

Algorithm 1 Find structure of a hybrid model

Input: A set of labelled instances.

Output: A hybrid network structure with identified *LR-part* and *NB-part*.

1: Set *NB-part* $= \emptyset$ and *LR-part* $= \emptyset$.
2: For each $F_i \in \{F_1, \ldots, F_n\}$:
3: Calculate the index of relative errors $e(F_i)$ using Eq. (8).
4: If $e(F_i) \leq 0$, then *LR-part* $=$ *LR-part* $\cup \{F_i\}$
5: If $e(F_i) > 0$, then *NB-part* $=$ *NB-part* $\cup \{F_i\}$
6: end for;
7: end algorithm

6 Experimental Analysis

To evaluate the performance of hybrid models constructed using the heuristic introduced in Section 5, we conduct experiments on 25 different machine learning datasets from the UCI Machine Learning Repository. A summary of these datasets is given in Table 1. The datasets are selected such that we have a diversity of sample sizes, number of features, and binary/non-binary class variables.

Table 1. A Summary of 25 Bench-Mark Datasets

Dataset	*# Features*	*# Instances*	*# Classes*	*Dataset*	*# Features*	*# Instances*	*# Classes*
Abalone	8	4,177	3	Iris	4	150	3
Balance Scale	4	625	3	Liver Disorders	6	345	2
Banknote Authentication	4	1,372	2	Magic Gamma Telescope	10	19,020	2
Qualitative Bankruptcy	6	250	2	Mammographic Mass	5	961	2
Blogger	5	100	2	Mushroom	21	8,124	2
Blood Transfusion Service Center	4	748	2	New Thyroid	5	215	3
Car Evaluation	6	1,728	4	Pima Indians Diabetes	8	768	2
Connectionist Bench	60	208	2	Statlog Vehicle Silhouettes	18	846	4
Credit Approval	15	690	2	Vertebral Column	6	310	2
Hepatitis	19	155	2	Congressional Voting Records	16	435	2
Heart Disease (Hungarian)	13	294	5	Wilt	5	4,839	2
Hypothyroid	17	3,163	2	Wine	13	178	3
ILPD (Indian Liver Patient Dataset)	10	583	2				

In Section 6.1, we start with the description of our experimental setup. In Section 6.2, we compare the performance of hybrid models constructed using our heuristic with that of pure LR and pure NB models. The training time of hybrid models is compared with that of pure LR and pure NB models in Section 6.3. The performance of our proposed method is compared with state-of-the-art classifiers, Random Forest (RF) and regularized LR (LASSO) in Sections 6.4 and 6.5 respectively.

6.1 Experimental Setup

The experiments are conducted using R. Numeric features in the original datasets are discretized using an entropy-based method (minimum description length (MDL) method), proposed by Fayyad and Irani [5]. We carry out the discretization procedure using a filter in WEKA. Besides, missing values of any features are imputed with the conditional probability given the response variable, i.e., $P(E_i \mid C)$.

First, we randomly divide each dataset into two parts, a training set with about 90% of the instances, and a test set with the remaining 10% of the instances. Using the training set, we implement the Algorithm 1 described in Section 5 to identify the hybrid model structure, i.e., to partition the feature set into an LR part and an NB part. Specifically, we select the value of tuning parameter λ using 10-fold cross-validation in the training set by minimizing the root mean square error (RMSE), which is calculated as follows:

$$RMSE = \sqrt{\frac{\sum_{t=1}^{T}\sum_{s=1}^{S}(\hat{P}(y_{ts}) - y_{ts})^2}{T \cdot S}}$$

where y_{ts} is an indicator, which takes the value of 1 if the observed category of class variable y for instance t is s, and 0 otherwise. $\hat{P}(y_{ts})$ is the predicted probability of the class variable y for instance t to take the category s, T is the total number of instances, and S is the total number of categories for the class variable. Note that, the value of λ can also be chosen based on hybrid models' 0-1 loss, posterior likelihood, or AUC (for binary responses).

Next, we learn the parameters of the corresponding hybrid model with the entire training data. The estimation of parameters of the NB part of hybrid model and also those of pure NB model

are conducted using the Laplace correction [14] to prevent the high influence of zero probabilities. Specifically, we assume the training set is large enough that adding one to each count would not make a significant difference in estimated probabilities.

Finally, we predict the class for instances in the test set. The prediction performance is measured using 0-1 loss, and RMSE.

We repeat the entire procedure (division of dataset, identification of model structure, estimation of model parameters using the training set, and computation of prediction accuracies using the test set) 1,000 times. For computational reasons, experiments for datasets with more than 8,000 instances, or 20 features (*Connectionist*, *Magic*, *Mashroon*) are conducted only 200 times. In the remainder of this section, we report Win/Draw/Loss (W/D/L) results when comparing the 0-1 loss and RMSE of two models. A two-tail paired t-test is used and we consider the results to be significant if its p-value is less than 0.05. The detailed results in terms of average structure of the hybrid model, prediction accuracies measured by 0-1 loss (in units of %) and RMSE with their standard errors, and training time are presented in the appendix.

6.2 Hybrid versus LR and NB

We start by examining the average structure of the hybrid model found using our heuristic before making the comparison between hybrid model versus pure LR and pure NB. On an average for the 25 datasets, the hybrid model consists of 77.33% of the features in the LR part, with the remaining 22.67% in the NB part. Notice that if no feature is assigned to the LR (NB) part, then the hybrid model is exactly the same as pure NB (LR) model. The resulting hybrid model is identical to pure LR in 5 datasets, and possibly strictly hybrid in the remaining 20 datasets.

Table 2. Win/Draw/Loss: Hybrid versus LR, Hybrid versus NB

W/D/L	*Hybrid vs. LR*	*Hybrid vs. NB*
0-1 Loss	6/19/0	18/2/5
RMSE	10/14/1	18/1/6

The Win/Draw/Loss (W/D/L) results of hybrid model against pure LR and pure NB are given in Table 2. It can be seen that hybrid model has significantly better 0-1 loss and RMSE than both pure LR and pure NB, indicating that our heuristic decribed in Section 5 is effective in improving the classification performance of hybrid model compared to pure LR and pure NB.

It is also worth noting that for those 20 datasets where the resulting model is possibly strictly hybrid, the hybrid model significantly outperforms both LR and NB in terms of 0-1 loss in 2 of them, and in terms of RMSE in 4 of them. This suggests that the hybrid model can be more powerful than making the selection between a pure LR and a pure NB model by providing the additional model structure fexibility.

6.3 Training Time Comparison

LR is computationally expensive at training time. Given a large number of features, estimating the parameters of an LR model by maximizing the conditional log-likelihood is a computationally intensive task. In this section, we examine the effectiveness of hybrid model on reducing the

LR's training time by assigning some of the features to the NB part, and consequently achieving dimension reduction in the LR part.

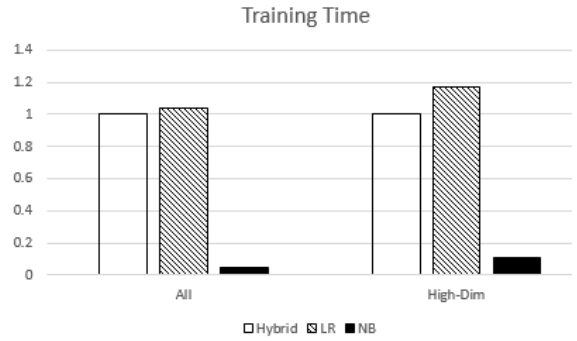

Fig. 4. Average training time of hybrid model, LR and NB on all 25 or high-dimensional datasets. Results are normalized with respect to hybrid model.

A comparison of the training time of hybrid model versus LR and NB is given in Fig. 4. The results shown are consistent with our expectation that hybrid model requires less training time than pure LR. While over the entire 25 datasets, the hybrid model exhibits a slightly faster training speed than pure LR, the difference is more significant over the high-dimensional datasets (7 datasets with at least 15 features). Notice that we ignore the issue of classification time in this paper as hybrid model, LR and NB are all quite efficient in terms of classification time.

6.4 Hybrid versus Random Forest

Random forest (RF) [2] is recognized as a state-of-the-art classification technique. It basically consists of many classification trees, where each tree is trained using randomly selected (with replacement) instances from the training set. The RF makes a classification by choosing the most frequently selected category over all trees in the forest. We use 100 decision trees in this work.

Table 3. Win/Draw/Loss: Hybrid versus RF

W/D/L	0-1 Loss	RMSE
Hybrid vs. RF	5/7/13	14/2/9

The hybrid model is compared with RF in terms of W/D/L of 0-1 loss and RMSE in Table 3. It can be seen that hybrid model has higher 0-1 loss, but lower RMSE than RF, indicating that our heuristic results in a hybrid model that is competitive with the well known random forest. Also, Figure 5 suggests that RF requires much more training time than the hybrid model.

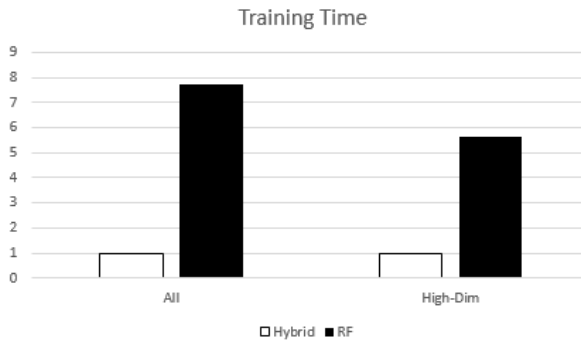

Fig. 5. Average training time of hybrid model and RF on all 25 or high-dimensional datasets. Results are normalized with respect to hybrid model.

6.5 Hybrid versus LASSO

L_1 regularized logistic regression, also known as LASSO [20], is a bias-variance technique that performs both feature selection and shrinkage in order to improve the algorithm prediction power. It reduces the variance at the expense of increasing its bias, by shrinking the coefficients towards zero. By assuming sparsity, which reduces the model complexity and ensures the identifiability of the true underlying sparse model given limited sample size, LASSO is particularly useful in high-dimensional cases. We select the value of penalty parameter using 10-fold cross validation in the training set by minimizing the MSE in this work.

Table 4. Win/Draw/Loss: Hybrid versus LASSO

W/D/L	0-1 Loss	RMSE
Hybrid vs. LASSO	5/10/10	1/14/10

The hybrid model is compared with LASSO in terms of W/D/L of 0-1 loss and RMSE in Table 4. Our method performs worse than LASSO over the entire 25 datasets in terms of both 0-1 loss and RMSE. However, when the number of predictors is small, LASSO's sparsity assumption is more likely to be violated. As a result, the difference between hybrid model and LASSO appears to be insignificant. For example, the W/D/L for 0-1 loss is 3/7/4, and for RMSE is 0/12/2 on datasets with fewer than 10 features. Also, Figure 6 suggests that LASSO requires much more training time than the hybrid model.

7 Summary and Conclusions

In this paper, we describe a new hybrid LR-NB model construction method that follows the strategy of balancing the tradeoff between model bias and model variance, with the objective of minimizing the sum of these two errors. Our approach is primarily motivated by the intuition of taking advantage of the strengths of both LR and NB, and takes into account the training sample size and

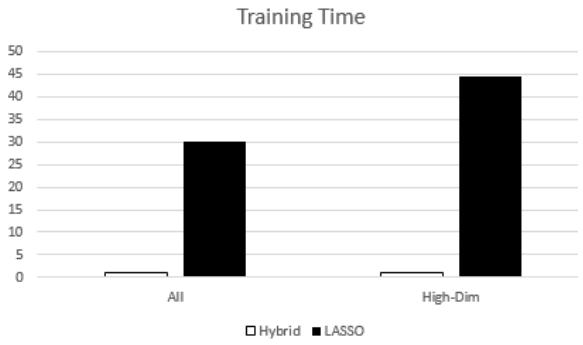

Fig. 6. Average training time of hybrid model and LASSO on all 25 or high-dimensional datasets. Results are normalized with respect to hybrid model.

the conditional dependence among features. Experimental results are presented showing that the hybrid model constructed using our heuristic can generally outperform both pure LR and pure NB models. Hybrid model offers an improvement over pure LR in terms of training time by optimizing fewer parameters in the LR part. Also its prediction performance is comparable to random forest. Although the hybrid model fails to beat LASSO, the difference appears to be insignificant when the number of features is small. Also, the hybrid model requires much less training time, which makes it attractive when the training time is a big concern.

Our work adds to the literature that investigates the properties of LR and NB [12, 17], and makes the comparison between them [13, 22]. Particularly, our heuristic posits functions that try to link the training sample size and the conditional dependence among features, to the bias and the variance of assigning a given feature to the LR and the NB part of a hybrid model respectively. By balancing the tradeoff between bias and variance, we provide a selection mechanism that helps in making the choice between pure LR, pure NB, and strictly hybrid models.

8 Acknowledgements

Thanks to Suzanna Emelio for a careful proof-reading of this paper. Also, thanks to three reviewers of IJAR for comments on an earlier draft of this paper.

References

1. Bishop, C. M., and Lasserre, J. Generative or discriminative? Getting the best of both worlds. In *Bayesian Statistics 8*. Oxford Univ. Press, 2007, pp. 3–24.
2. Breiman, L. Random forests. *Machine Learning 45*, 1 (2001), 5–32.
3. Efron, B. The efficiency of logistic regression compared to normal discriminant analysis. *Journal of the American Statistical Association 70*, 352 (1975), 892–898.
4. Ezawa, K. J., and Schuermann, T. Fraud/uncollectible debt detection using a Bayesian network based learning system: A rare binary outcome with mixed data structures. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence* (1995), Morgan Kaufmann Publishers Inc., pp. 157–166.

5. Fayyad, U. M., and Irani, K. B. Multi-interval discretization of continuous-valued attributes for classification learning. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence* (1993), Morgan Kaufmann, pp. 1022–1027.
6. Ferreira, J., Denison, D., and Hand, D. Weighted naive Bayes modelling for data mining. Technical report, Dept. of Mathematics, Imperial College, London, UK, 2001.
7. Friedman, N., Geiger, D., and Goldszmidt, M. Bayesian network classifiers. *Machine Learning 29*, 2–3 (1997), 131–163.
8. Fujino, A., Ueda, N., and Saito, K. A hybrid generative/discriminative approach to text classification with additional information. *Information Processing and Management 43* (2007), 379–392.
9. Hall, M. A decision tree-based attribute weighting filter for naive Bayes. *Knowledge-Based Systems 20*, 2 (2007), 120–126.
10. Kang, C., and Tian, J. A hybrid generative/discriminative Bayesian classifier. In *Proceedings of the 19th International Florida Artificial Intelligence Research Society Conference* (2006), pp. 562–567.
11. Kononenko, I. Semi-naive Bayesian classifier. In *European Working Session on Learning* (1991), Springer, pp. 206–219.
12. Mitchell, T. M. *Machine Learning*. WCB/McGraw-Hill, Boston, MA, 1997.
13. Ng, A. Y., and Jordan, M. I. On discriminative vs. generative classifiers: A comparison of logistic regression and naïve Bayes. In *Advances in Neural Information Processing Systems* (2001), T. G. Dietterich, S. Becker, and Z. Ghahramani, Eds., vol. 14, pp. 841–848.
14. Niblett, T. Constructing decision trees in noisy domains. In *Proceedings of the Second European Working Session on Learning* (Bled, Yugoslavia, 1987), Sigma, pp. 67–78.
15. Pazzani, M. J. Searching for dependencies in Bayesian classifiers. In *Learning from Data*. Springer, 1996, pp. 239–248.
16. Raina, R., Shen, Y., Ng, A. Y., and McCallum, A. Classification with hybrid generative/discriminative models. In *Advances in Neural Information Processing Systems* (2003), S. Thrun, L. K. Saul, and B. Schölkopf, Eds., vol. 16, pp. 545–552.
17. Rijmen, F. Bayesian networks with a logistic regression model for the conditional probabilities. *International Journal of Approximate Reasoning 48*, 2 (2008), 659–666.
18. Rubinstein, Y., and Hastie, T. Discriminative vs. informative learning. In *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining* (1997), AAAI Press, pp. 49–53.
19. Tan, Y., Shenoy, P. P., Chan, M. W., and Romberg, P. M. On construction of hybrid logistic regression-naïve Bayes model for classification. In *Proceedings of Eighth International Conference on Probabilistic Graphical Models* (Lugano, Switzerland, 2016), PMLR, pp. 523–534.
20. Tibshirani, R. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological) 58*, 1 (1996), 267–288.
21. Xue, J.-E., and Titterington, D. M. Joint discriminative-generative modelling based on statistical tests for classification. *Pattern Recognition Letters 31*, 9 (2010), 1048–1055.
22. Xue, J.-H., and Titterington, D. M. Comment on "on discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes". *Neural processing letters 28*, 3 (2008), 169.
23. Zaidi, N. A., Carman, M. J., Cerquides, J., and Webb, G. I. Naive-Bayes inspired effective pre-conditioner for speeding-up logistic regression. In *Data Mining (ICDM), 2014 IEEE International Conference on* (2014), IEEE, pp. 1097–1102.
24. Zaidi, N. A., Cerquides, J., Carman, M. J., and Webb, G. I. Alleviating naive Bayes attribute independence assumption by attribute weighting. *Journal of Machine Learning Research 14*, 1 (2013), 1947–1988.
25. Zaidi, N. A., Webb, G. I., Carman, M. J., Petitjean, F., and Cerquides, J. ALR^n: accelerated higher-order logistic regression. *Machine Learning 104*, 2-3 (2016), 151–194.
26. Zhang, N. L., and Poole, D. A simple approach to Bayesian network computations. In *Proc. of the Tenth Canadian Conference on Artificial Intelligence* (1994).
27. Zheng, F., Webb, G. I., Suraweera, P., and Zhu, L. Subsumption resolution: An efficient and effective technique for semi-naive Bayesian learning. *Machine learning 87*, 1 (2012), 93–125.

Table 5. Average Structure of the Hybrid Model

Dataset	*# Features*	*# LR-part*	*# NB-part*	*Dataset*	*# Features*	*# LR-part*	*# NB-part*
Abalone	8	8	0	Iris	4	1.133	2.867
Balance Scale	4	4	0	Liver Disorders	6	5.858	0.142
Banknote Authentication	4	4	0	Magic Gamma Telescope	10	10	0
Qualitative Bankruptcy	6	4.872	1.128	Mammographic Mass	5	4.74	0.26
Blogger	5	1.402	3.598	Mushroom	21	15.376	5.624
Blood Transfusion Service Center	4	3.89	0.11	New Thyroid	5	2.302	2.698
Car Evaluation	6	6	0	Pima Indians Diabetes	8	7.313	0.687
Connectionist Bench	60	47.505	12.495	Statlog Vehicle Silhouettes	18	15.585	2.415
Credit Approval	15	7.109	7.891	Vertebral Column	6	4.656	1.344
Hepatitis	19	13.747	5.253	Congressional Voting Records	16	15.4	0.6
Heart Disease (Hungarian)	13	5.555	7.445	Wilt	5	4.489	0.511
Hypothyroid	17	16.019	0.981	Wine	13	1.314	11.686
ILPD (Indian Liver Patient Dataset)	10	9.926	0.074				

A Detailed Results

This appendix presents the detailed results for average structure of the hybrid model, prediction accuracies measured by 0-1 loss (in units of %) and RMSE, and training time.

Table 6. Summary of Average 0-1 Loss of Hybrid Model, LR, NB, RF, and LASSO in units of % (SE in parentheses).

Dataset	*Hybrid*	*LR*	*NB*	*RF*	*LASSO*
Abalone	36.34 (0.14)	36.34 (0.14)	41.26 (0.16)	36.11 (0.14)	36.26 (0.14)
Balance Scale	1.84 (0.06)	1.84 (0.06)	8.59 (0.11)	16.21 (0.14)	1.47 (0.04)
Banknote Authentication	5.39 (0.06)	5.39 (0.06)	7.18 (0.06)	5.41 (0.06)	5.38 (0.06)
Qualitative Bankruptcy	0.86 (0.06)	0.77 (0.06)	0.76 (0.05)	0.00 (0.00)	0.58 (0.05)
Blogger	27.68 (0.41)	27.00 (0.42)	28.54 (0.41)	15.71 (0.37)	28.66 (0.43)
Blood Transfusion Service Center	22.30 (0.14)	22.30 (0.14)	24.70 (0.15)	22.61 (0.15)	22.34 (0.15)
Car Evaluation	6.59 (0.06)	6.59 (0.06)	14.73 (0.09)	2.78 (0.04)	6.71 (0.06)
Connectionist Bench	18.16 (0.29)	20.48 (0.28)	19.77 (0.30)	10.31 (0.24)	12.81 (1.00)
Credit Approval	13.54 (0.12)	14.37 (0.13)	13.25 (0.12)	13.71 (0.12)	13.93 (0.13)
Hepatitis	11.38 (0.26)	13.20 (0.28)	11.72 (0.25)	9.07 (0.22)	9.81 (0.23)
Heart Disease (Hungarian)	33.13 (0.26)	33.02 (0.26)	34.21 (0.26)	33.29 (0.25)	34.29 (0.25)
Hypothyroid	0.81 (0.02)	0.79 (0.02)	1.29 (0.02)	0.84 (0.02)	0.76 (0.02)
ILPD (Indian Liver Patient Dataset)	27.53 (0.16)	27.52 (0.16)	31.29 (0.18)	29.51 (0.17)	28.51 (0.17)
Iris	5.97 (0.18)	5.82 (0.18)	5.53 (0.17)	5.03 (0.17)	6.13 (0.18)
Liver Disorders	33.33 (0.24)	33.27 (0.24)	35.59 (0.24)	31.23 (0.23)	33.06 (0.24)
Magic Gamma Telescope	15.12 (0.07)	15.12 (0.07)	21.63 (0.08)	14.29 (0.06)	15.13 (3.16)
Mammographic Mass	17.21 (0.11)	17.19 (0.11)	16.20 (0.11)	17.10 (0.11)	16.88 (0.11)
Mushroom	0.00 (0.00)	0.00 (0.00)	5.51 (0.10)	0.00 (0.00)	0.00 (0.00)
New Thyroid	5.07 (0.15)	5.60 (0.16)	3.81 (0.12)	3.35 (0.12)	4.11 (0.13)
Pima Indians Diabetes	18.74 (0.13)	18.74 (0.13)	19.82 (0.14)	20.29 (0.14)	18.71 (0.13)
Statlog Vehicle Silhouettes	25.48 (0.15)	25.31 (0.16)	34.92 (0.16)	22.92 (0.14)	24.65 (0.15)
Vertebral Column	15.27 (0.19)	15.84 (0.20)	21.14 (0.22)	15.21 (0.19)	15.84 (0.20)
Congressional Voting Records	3.80 (0.09)	3.77 (0.09)	8.90 (0.13)	3.05 (0.08)	3.25 (0.08)
Wilt	2.78 (0.02)	2.78 (0.02)	5.38 (0.03)	2.91 (0.02)	2.78 (0.02)
Wine	2.34 (0.11)	3.09 (0.14)	1.16 (0.08)	1.18 (0.08)	1.64 (0.10)

Table 7. Summary of Average RMSE of Hybrid Model, LR, NB, RF, and LASSO (SE in parentheses).

Dataset	*Hybrid*	*LR*	*NB*	*RF*	*LASSO*
Abalone	0.3899 (0.0005)	0.3899 (0.0005)	0.4513 (0.0009)	0.4100 (0.0007)	0.3895 (0.0005)
Balance Scale	0.0841 (0.0021)	0.0841 (0.0021)	0.2837 (0.0007)	0.2807 (0.0009)	0.0651 (0.0012)
Banknote Authentication	0.2018 (0.0009)	0.2018 (0.0009)	0.2264 (0.0007)	0.2169 (0.0012)	0.2016 (0.0009)
Qualitative Bankruptcy	0.0411 (0.0024)	0.0359 (0.0025)	0.0361 (0.0015)	0.0448 (0.0010)	0.0419 (0.0017)
Blogger	0.4331 (0.0029)	0.4440 (0.0030)	0.4272 (0.0027)	0.3315 (0.0034)	0.4322 (0.0023)
Blood Transfusion Service Center	0.3962 (0.0009)	0.3962 (0.0009)	0.4103 (0.0010)	0.4406 (0.0015)	0.3963 (0.0009)
Car Evaluation	0.1498 (0.0006)	0.1498 (0.0006)	0.2272 (0.0004)	0.1326 (0.0003)	0.1500 (0.0006)
Connectionist Bench	0.4070 (0.0038)	0.4369 (0.0034)	0.3866 (0.0032)	0.3218 (0.0013)	0.2896 (0.0022)
Credit Approval	0.3179 (0.0014)	0.3188 (0.0013)	0.3237 (0.0015)	0.3173 (0.0012)	0.3159 (0.0012)
Hepatitis	0.2792 (0.0039)	0.3157 (0.0044)	0.2831 (0.0040)	0.2677 (0.0023)	0.2703 (0.0028)
Heart Disease (Hungarian)	0.2959 (0.0010)	0.2985 (0.0010)	0.3022 (0.0011)	0.2999 (0.0011)	0.2921 (0.0009)
Hypothyroid	0.0840 (0.0008)	0.0832 (0.0008)	0.1017 (0.0009)	0.0812 (0.0007)	0.0809 (0.0008)
ILPD (Indian Liver Patient Dataset)	0.4162 (0.0009)	0.4161 (0.0009)	0.4601 (0.0013)	0.4348 (0.0011)	0.4137 (0.0008)
Iris	0.1351 (0.0032)	0.1457 (0.0031)	0.1258 (0.0031)	0.1271 (0.0029)	0.1373 (0.0025)
Liver Disorders	0.4537 (0.0010)	0.4534 (0.0010)	0.4655 (0.0008)	0.4723 (0.0016)	0.4538 (0.0010)
Magic Gamma Telescope	0.3347 (0.0005)	0.3347 (0.0005)	0.3903 (0.0007)	0.3309 (0.0006)	0.3348 (0.0006)
Mammographic Mass	0.3424 (0.0010)	0.3422 (0.0010)	0.3565 (0.0012)	0.3592 (0.0012)	0.3421 (0.0009)
Mushroom	0.0000 (00000)	0.0002 (0.0001)	0.2065 (0.0062)	0.0061 (0.0002)	0.0035 (0.0001)
New Thyroid	0.1308 (0.0027)	0.1416 (0.0033)	0.1130 (0.0022)	0.1264 (0.0018)	0.1214 (0.0025)
Pima Indians Diabetes	0.3689 (0.0010)	0.3691 (0.0010)	0.3784 (0.0011)	0.3771 (0.0011)	0.3694 (0.0010)
Statlog Vehicle Silhouettes	0.2910 (0.0008)	0.2942 (0.0009)	0.3695 (0.0009)	0.2750 (0.0007)	0.2803 (0.0007)
Vertebral Column	0.3435 (0.0017)	0.3461 (0.0016)	0.3879 (0.0021)	0.3481 (0.0023)	0.3448 (0.0014)
Congressional Voting Records	0.1640 (0.0024)	0.1635 (0.0025)	0.2761 (0.0022)	0.1572 (0.0015)	0.1481 (0.0018)
Wilt	0.1584 (0.0006)	0.1584 (0.0006)	0.1911 (0.0006)	0.1630 (0.0006)	0.1584 (0.0006)
Wine	0.0751 (0.0029)	0.0909 (0.0033)	0.0431 (0.0022)	0.0994 (0.0011)	0.0690 (0.0022)

Table 8. Summary of Average Train Time of Hybrid Model, LR, NB and RF in seconds.

Dataset	*Hybrid*	*LR*	*NB*	*RF*	*LASSO*
Abalone	0.5224	0.5204	0.0090	2.5621	27.9373
Balance Scale	0.0206	0.0211	0.0023	0.1575	5.1914
Banknote Authentication	0.0339	0.0330	0.0030	0.4136	2.1155
Qualitative Bankruptcy	0.0050	0.0049	0.0026	0.0229	0.3455
Blogger	0.0056	0.0055	0.0021	0.0240	0.2145
Blood Transfusion Service Center	0.0084	0.0083	0.0024	0.1022	0.4382
Car Evaluation	0.1188	0.1185	0.0038	0.5156	10.5964
Connectionist Bench	0.0222	0.0226	0.0169	0.0781	0.6644
Credit Approval	0.0183	0.0204	0.0056	0.1418	1.2105
Hepatitis	0.0134	0.0132	0.0070	0.0434	1.1598
Heart Disease (Hungarian)	0.0130	0.0200	0.0043	0.0750	1.2388
Hypothyroid	0.0715	0.0722	0.0116	0.9972	3.7391
ILPD (Indian Liver Patient Dataset)	0.0087	0.0088	0.0038	0.0956	0.4695
Iris	0.0045	0.0074	0.0020	0.0178	0.7708
Liver Disorders	0.0091	0.0092	0.0034	0.0789	0.3703
Magic Gamma Telescope	2.0696	2.0794	0.0291	18.5753	21.2618
Mammographic Mass	0.0118	0.0114	0.0031	0.1633	0.6603
Mushroom	0.4388	0.5057	0.0239	2.1637	12.6604
New Thyroid	0.0065	0.0079	0.0022	0.0281	0.8481
Pima Indians Diabetes	0.0127	0.0132	0.0037	0.1518	0.5654
Statlog Vehicle Silhouettes	0.1260	0.1734	0.0065	0.4541	11.1105
Vertebral Column	0.0054	0.0047	0.0025	0.0346	0.3363
Congressional Voting Records	0.0178	0.0187	0.0065	0.0932	0.8938
Wilt	0.0519	0.0463	0.0062	0.9906	3.1263
Wine	0.0082	0.0134	0.0042	0.0276	0.6744

B R Code

```
#'Experiments on empirical data from UCI with 1000 training-testing splitting. We compare both 0-1 loss and RMSE.
#'Further adjustment is needed to run another dataset.
#'@param delta turning parameter, denoted as lambda in the paper.
#'@param ci conditional mutual information.
#'@param std_ci normalized conditional mutual information.
#'@param mi average normalized conditional mutual information.
#'@param mb vector of predictor names in the data.
#'@param n_mb number of predictors in the data.
#'@param merit index of relative reducible errors, denoted as e(F) in the paper.
#'@param lr vector of names of predictors in the LR part of the hybrid model.
#'@param n_lr number of predictors to be assigned to the LR part of the hybrid model.
#'@param nb vector of names of predictors in the NB part of the hybrid model.
#'@param n_nb number of predictors to be assigned to the NB part of the hybrid model.



score_nb<-rep(0,1000)
score_lr<-rep(0,1000)
rmse_nb<-rep(0,1000)
rmse_lr<-rep(0,1000)
score_hybrid<-rep(0,1000)
rmse_hybrid<-rep(0,1000)
score_lasso<-rep(0,1000)
rmse_lasso<-rep(0,1000)
score_rf<-rep(0,1000)
rmse_rf<-rep(0,1000)
m_nb<-rep(0,1000)
m_lr<-rep(0,1000)
set.seed(1)

for (o in 1:1000){

 # Random sample with about 10% instances as the test set

 data<-read.csv("C:\\hybrid model\\workspace\\pima\\disc_pima.csv",head=T)
 nn<-rownames(data)
 nn_test<-sample(nn,75)
 train<-data[!rownames(data) %in% nn_test, ]
 test<-data[rownames(data) %in% nn_test, ]
 write.csv(train,file="C:\\hybrid model\\workspace\\pima\\analysis\\train.csv")
 write.csv(test,file="C:\\hybrid model\\workspace\\pima\\analysis\\test.csv")
```

```
mb<-names(data)
mb<-mb[-length(mb)]
n_mb<-length(mb)-1
ci<-matrix(0,n_mb,n_mb)
std_ci<-matrix(0,n_mb,n_mb)

# Calculate the average standardized conditional mutual information for predictors in the training set.

data<-train[mb]
for (i in 1:n_mb){
  for (j in 1:n_mb){
    data0<-data[,c(i,j,n_mb+1)]
    ci[i,j]<-condinformation(data0[,1],data0[,2],data0[,3],method="emp")
  }
}
for (i in 1:n_mb){
  for (j in 1:n_mb){
    std_ci[i,j]<-ci[i,j]/sqrt(ci[i,i]*ci[j,j])
  }
}
diag(std_ci)<-0
if (n_mb>1){
  mi<-colSums(std_ci)/(n_mb-1)
}
if (n_mb==1){
  mi<-0
}


#Determine the max value of delta which leads to a pure NB, and the min value which leads to a pure
LR.

delta_max<-(-log(min(mi)))/nrow(train)*1.2
delta_min<-(-log(max(mi)))/nrow(train)*0.8
delta0<-exp(seq(log(delta_min),log(delta_max),length.out=50))
delta0_error<-matrix(0,50,10)

# Use 10-folds cv to find the delta with the smallest RMSE.

flds0 <- createFolds(train[,ncol(train)], k = 10, list = TRUE, returnTrain = FALSE)
for (w in 1:10){
  flds<-flds0
  names(flds)[w] <- "test"
  training<-train[-flds$test,]
```

```
  testing<-train[flds$test,]

 # Calculate the average standardized conditional mutual information for each cv.

  data<-training[mb]
  for (i in 1:n_mb){
   for (j in 1:n_mb){
    data0<-data[,c(i,j,n_mb+1)]
    ci[i,j]<-condinformation(data0[,1],data0[,2],data0[,3],method="emp")
   }
  }
  for (i in 1:n_mb){
   for (j in 1:n_mb){
    std_ci[i,j]<-ci[i,j]/sqrt(ci[i,i]*ci[j,j])
   }
  }
  diag(std_ci)<-0
  if (n_mb>1){
   mi_cv<-colSums(std_ci)/(n_mb-1)
  }
  if (n_mb==1){
   mi_cv<-0
  }

  # Construct the hybrid model using our heuristic with 50 possible values of delta range from
delta_min to delta_max.

  for (u in 1:50){
   delta=delta0[u]
   merit_cv=exp(-delta*nrow(training))-mi_cv
   nb<-c("y_c")
   lr<-c("y_n")
   for (k in 1:n_mb){
    if (merit_cv[k]<0){
     lr<-union(mb[k],lr)
    }
    if (merit_cv[k]>0){
     nb<-union(mb[k],nb)
    }
   }
   n_nb<-length(nb)-1
   n_lr<-length(lr)-1

   training_lr<-training[lr]
```

```
training_nb<-training[nb]
testing_lr<-testing[lr]
testing_nb<-testing[nb]

# Keep the consistency of levels for each variable from sub-sample.

level_lr<-read.csv("C:\\hybrid model\\workspace\\pima\\disc_pima.csv",head=T)
level_lr<-level_lr[lr]
level_nb<-read.csv("C:\\hybrid model\\workspace\\pima\\disc_pima.csv",head=T)
level_nb<-level_nb[nb]
if (ncol(training_nb)!=1){
 for (v in 1:n_nb){
  training_nb[,c(nb[v])]<-factor(training_nb[,c(nb[v])], levels=levels(level_nb[,c(nb[v])]))
 }
 for (v in 1:n_nb){
  testing_nb[,c(nb[v])]<-factor(testing_nb[,c(nb[v])], levels=levels(level_nb[,c(nb[v])]))
 }
}
if (ncol(training_lr)!=1){
 for (v in 1:n_lr){
  training_lr[,c(lr[v])]<-factor(training_lr[,c(lr[v])], levels=levels(level_lr[,c(lr[v])]))
 }
 for (v in 1:n_lr){
  testing_lr[,c(lr[v])]<-factor(testing_lr[,c(lr[v])], levels=levels(level_lr[,c(lr[v])]))
 }
}
training_nb<-as.data.frame(training_nb)
testing_nb<-as.data.frame(testing_nb)
training_lr<-as.data.frame(training_lr)
testing_lr<-as.data.frame(testing_lr)

# Learn the parameters of both LR and NB part of the hybrid model separately, and make the
prediction for RMSE.

if (ncol(training_nb)!=1 & ncol(training_lr)==1){
 fitted<-naiveBayes(y_c ~ ., data = training_nb,laplace=1)
 pred_mse<-predict(fitted, testing_nb,"raw")
 result_mse<-testing_nb
 result_mse$pred_a<-pred_mse[,1]
 result_mse$pred_b<-pred_mse[,2]
 for(l in 1:(nrow(result_mse))){
  if (result_mse$y_c[l]=="a"){
   result_mse$a[l]<-1
   result_mse$b[l]<-0
```

```
      }
      if (result_mse$y_c[l]=="b"){
        result_mse$a[l]<-0
        result_mse$b[l]<-1
      }
    }
    result_mse$mse=((result_mse$pred_a-result_mse$a)^2+(result_mse$pred_b-result_mse$b)^2)/2
    delta0_error[u,w]<-sqrt(mean(result_mse$mse))
  }
  if (ncol(training_nb)==1 & ncol(training_lr)!=1){
    fitted<-multinom(y_n~.,data=training_lr)
    pred_mse<-predict(fitted, testing_lr,"probs")
    pred_mse<-as.data.frame(pred_mse)
    result_mse<-testing_nb
    result_mse$pred_a<-pred_mse[,1]
    result_mse$pred_b<-1-pred_mse[,1]
    for(l in 1:(nrow(result_mse))){
      if (result_mse$y_c[l]=="a"){
        result_mse$a[l]<-1
        result_mse$b[l]<-0
      }
      if (result_mse$y_c[l]=="b"){
        result_mse$a[l]<-0
        result_mse$b[l]<-1
      }
    }
    result_mse$mse=((result_mse$pred_a-result_mse$a)^2+(result_mse$pred_b-result_mse$b)^2)/2
    delta0_error[u,w]<-sqrt(mean(result_mse$mse))
  }
  if (ncol(training_nb)!=1 & ncol(training_lr)!=1){
    fitted1<-naiveBayes(y_c ~ ., data = training_nb,laplace=1)
    fitted2<-multinom(y_n~.,data=training_lr)
    pred2 <- predict(fitted2, testing_lr,"probs")
    pred2<-as.data.frame(pred2)
    pred2$a<-pred2[,1]
    pred2$b<-1-pred2[,1]
    pred_mse<-array(0,dim=c(nrow(testing_nb),2))

  # To make inferences in a hybrid model, we replace the prior of NB part by the posterior from LR for each instance.

    for(j in 1:(nrow(testing_nb))){
      cptY=matrix(c(pred2$a[j],pred2$b[j]),ncol=2,dimnames=list(NULL,c("a","b")))
      fitted1$apriori<-cptY
```

```
      pred_mse[j,]<-predict(fitted1, testing_nb[j,],"raw")
     }
     pred_mse<-as.data.frame(pred_mse)
     pred_mse[is.na(pred_mse)]<-0
     result_mse<-testing_nb
     result_mse$pred_a<-pred_mse[,1]
     result_mse$pred_b<-pred_mse[,2]
     for(l in 1:(nrow(result_mse))){
      if (result_mse$y_c[l]=="a"){
       result_mse$a[l]<-1
       result_mse$b[l]<-0
      }
      if (result_mse$y_c[l]=="b"){
       result_mse$a[l]<-0
       result_mse$b[l]<-1
      }
     }
     result_mse$mse=((result_mse$pred_a-result_mse$a)^2+(result_mse$pred_b-result_mse$b)^2)/2
     delta0_error[u,w]<-sqrt(mean(result_mse$mse))
    }
   }
  }

  delta0_error<-rowMeans(delta0_error)

  #Pick the delta with smallest RMSE. If multiple delta give the same RMSE, we pick the middle point.

  delta<-
0.5*max(delta0[which(delta0_error==min(delta0_error))])+0.5*min(delta0[which(delta0_error==min(de
lta0_error))])

  # Construct the hybrid structure using training data.

  merit=exp(-delta*nrow(train))-mi
  nb<-c("y_c")
  lr<-c("y_n")
  for (k in 1:n_mb){
   if (merit[k]<0){
    lr<-union(mb[k],lr)
   }
   if (merit[k]>0){
    nb<-union(mb[k],nb)
   }
  }
```

```
n_nb<-length(nb)-1
n_lr<-length(lr)-1
m_nb[o]<-length(nb)-1
m_lr[o]<-length(lr)-1

train_lr<-train[lr]
train_nb<-train[nb]
test_lr<-test[lr]
test_nb<-test[nb]
level_lr<-read.csv("C:\\hybrid model\\workspace\\pima\\disc_pima.csv",head=T)
level_lr<-level_lr[lr]
level_nb<-read.csv("C:\\hybrid model\\workspace\\pima\\disc_pima.csv",head=T)
level_nb<-level_nb[nb]
if (ncol(train_nb)!=1){
  for (v in 1:n_nb){
    train_nb[,c(nb[v])]<-factor(train_nb[,c(nb[v])], levels=levels(level_nb[,c(nb[v])]))
  }
  for (v in 1:n_nb){
    test_nb[,c(nb[v])]<-factor(test_nb[,c(nb[v])], levels=levels(level_nb[,c(nb[v])]))
  }
}
if (ncol(train_lr)!=1){
  for (v in 1:n_lr){
    train_lr[,c(lr[v])]<-factor(train_lr[,c(lr[v])], levels=levels(level_lr[,c(lr[v])]))
  }
  for (v in 1:n_lr){
    test_lr[,c(lr[v])]<-factor(test_lr[,c(lr[v])], levels=levels(level_lr[,c(lr[v])]))
  }
}
train_nb<-as.data.frame(train_nb)
test_nb<-as.data.frame(test_nb)
train_lr<-as.data.frame(train_lr)
test_lr<-as.data.frame(test_lr)

# Train the hybrid model using training data and predict on the test set.

if (ncol(train_nb)!=1 & ncol(train_lr)==1){
  fitted<-naiveBayes(y_c ~ ., data = train_nb,laplace=1)
  pred<-predict(fitted, test_nb)
  pred<-as.data.frame(pred)
  result<-cbind(test_nb,pred)
  right<-result[ which(result$y_c==result$pred),]
  score_hybrid[o]<-nrow(right)/nrow(result)
  pred_mse<-predict(fitted, test_nb,"raw")
```

```
  result_mse<-test_nb
  result_mse$pred_a<-pred_mse[,1]
  result_mse$pred_b<-pred_mse[,2]
  for(l in 1:(nrow(result_mse))){
    if (result_mse$y_c[l]=="a"){
      result_mse$a[l]<-1
      result_mse$b[l]<-0
    }
    if (result_mse$y_c[l]=="b"){
      result_mse$a[l]<-0
      result_mse$b[l]<-1
    }
  }
  result_mse$mse=((result_mse$pred_a-result_mse$a)^2+(result_mse$pred_b-result_mse$b)^2)/2
  rmse_hybrid[o]<-sqrt(mean(result_mse$mse))
}
if (ncol(train_nb)==1 & ncol(train_lr)!=1){
  fitted<-multinom(y_n~.,data=train_lr)
  pred<-predict(fitted, test_lr)
  pred<-as.data.frame(pred)
  result<-cbind(test_lr,pred)
  right<-result[ which(result$y_n==result$pred),]
  score_hybrid[o]<-nrow(right)/nrow(result)
  pred_mse<-predict(fitted, test_lr,"probs")
  pred_mse<-as.data.frame(pred_mse)
  result_mse<-test_nb
  result_mse$pred_a<-pred_mse[,1]
  result_mse$pred_b<-1-pred_mse[,1]
  for(l in 1:(nrow(result_mse))){
    if (result_mse$y_c[l]=="a"){
      result_mse$a[l]<-1
      result_mse$b[l]<-0
    }
    if (result_mse$y_c[l]=="b"){
      result_mse$a[l]<-0
      result_mse$b[l]<-1
    }
  }
  result_mse$mse=((result_mse$pred_a-result_mse$a)^2+(result_mse$pred_b-result_mse$b)^2)/2
  rmse_hybrid[o]<-sqrt(mean(result_mse$mse))
}
if (ncol(train_nb)!=1 & ncol(train_lr)!=1){
  fitted1<-naiveBayes(y_c ~ ., data = train_nb,laplace=1)
  fitted2<-multinom(y_n~.,data=train_lr)
```

```
pred2 <- predict(fitted2, test_lr,"probs")
pred2<-as.data.frame(pred2)
pred2$a<-pred2[,1]
pred2$b<-1-pred2[,1]
pred<-rep(0,nrow(test_nb))
for(j in 1:(nrow(test_nb))){
 cptY=matrix(c(pred2$a[j],pred2$b[j]),ncol=2,dimnames=list(NULL,c("a","b")))
 fitted1$apriori<-cptY
 pred[j]<-predict(fitted1, test_nb[j,])
}
pred<-as.data.frame(pred)
for(l in 1:(nrow(pred))){
 if (pred$pred[l]==1){
  pred$yhat[l]<-"a"
 }
 if (pred$pred[l]==2){
  pred$yhat[l]<-"b"
 }
}
result<-cbind(test_nb,pred)
right<-result[ which(result$y_c==result$yhat),]
score_hybrid[o]<-nrow(right)/nrow(result)
pred_mse<-array(0,dim=c(nrow(test_nb),2))
for(j in 1:(nrow(test_nb))){
 cptY=matrix(c(pred2$a[j],pred2$b[j]),ncol=2,dimnames=list(NULL,c("a","b")))
 fitted1$apriori<-cptY
 pred_mse[j,]<-predict(fitted1, test_nb[j,],"raw")
}
pred_mse<-as.data.frame(pred_mse)
pred_mse[is.na(pred_mse)]<-0
result_mse<-test_nb
result_mse$pred_a<-pred_mse[,1]
result_mse$pred_b<-pred_mse[,2]
for(l in 1:(nrow(result_mse))){
 if (result_mse$y_c[l]=="a"){
  result_mse$a[l]<-1
  result_mse$b[l]<-0
 }
 if (result_mse$y_c[l]=="b"){
  result_mse$a[l]<-0
  result_mse$b[l]<-1
 }
}
result_mse$mse=((result_mse$pred_a-result_mse$a)^2+(result_mse$pred_b-result_mse$b)^2)/2
```

```
  rmse_hybrid[o]<-sqrt(mean(result_mse$mse))
}

# Train the NB model using training data and predict on the test set.

train_nb<-read.csv("C:\\hybrid model\\workspace\\pima\\analysis\\train.csv",head=T)
train_nb<-train_nb[mb]
test_nb<-read.csv("C:\\hybrid model\\workspace\\pima\\analysis\\test.csv",head=T)
test_nb<-test_nb[mb]
level<-read.csv("C:\\hybrid model\\workspace\\pima\\disc_pima.csv",head=T)
level<-level[mb]
for (v in 1:(n_mb+1)){
  train_nb[,c(mb[v])]<-factor(train_nb[,c(mb[v])], levels=levels(level[,c(mb[v])]))
}
for (v in 1:(n_mb+1)){
  test_nb[,c(mb[v])]<-factor(test_nb[,c(mb[v])], levels=levels(level[,c(mb[v])]))
}
fitted<-naiveBayes(y_c ~ ., data = train_nb,laplace=1)
pred<-predict(fitted, test_nb)
pred<-as.data.frame(pred)
result<-cbind(test_nb,pred)
right<-result[ which(result$y_c==result$pred),]
score_nb[o]<-nrow(right)/nrow(result)
pred_mse<-predict(fitted, test_nb,"raw")
result_mse<-test_nb
result_mse$pred_a<-pred_mse[,1]
result_mse$pred_b<-pred_mse[,2]
for(l in 1:(nrow(result_mse))){
  if (result_mse$y_c[l]=="a"){
    result_mse$a[l]<-1
    result_mse$b[l]<-0
  }
  if (result_mse$y_c[l]=="b"){
    result_mse$a[l]<-0
    result_mse$b[l]<-1
  }
}
result_mse$mse=((result_mse$pred_a-result_mse$a)^2+(result_mse$pred_b-result_mse$b)^2)/2
rmse_nb[o]<-sqrt(mean(result_mse$mse))

# Train the random forest using training data and predict on the test set.

train_rf<-read.csv("C:\\hybrid model\\workspace\\pima\\analysis\\train.csv",head=T)
train_rf<-train_nb[mb]
```

```
test_rf<-read.csv("C:\\hybrid model\\workspace\\pima\\analysis\\test.csv",head=T)
test_rf<-test_nb[mb]
level<-read.csv("C:\\hybrid model\\workspace\\pima\\disc_pima.csv",head=T)
level<-level[mb]
for (v in 1:(n_mb+1)){
  train_rf[,c(mb[v])]<-factor(train_rf[,c(mb[v])], levels=levels(level[,c(mb[v])]))
}
for (v in 1:(n_mb+1)){
  test_rf[,c(mb[v])]<-factor(test_rf[,c(mb[v])], levels=levels(level[,c(mb[v])]))
}
train_rf<-na.omit(train_rf)
fitted<-randomForest(y_c~.,data=train_rf)
pred=predict(fitted,newdata=test_rf)
pred<-as.data.frame(pred)
result<-cbind(test_rf,pred)
right<-result[ which(result$y_c==result$pred),]
score_rf[o]<-nrow(right)/nrow(result)
pred_mse<-predict(fitted, test_rf,"prob")
pred_mse=as.data.frame(pred_mse)
result_mse<-test_rf
result_mse$pred_a<-pred_mse[,1]
result_mse$pred_b<-pred_mse[,2]
for(l in 1:(nrow(result_mse))){
  if (result_mse$y_c[l]=="a"){
    result_mse$a[l]<-1
    result_mse$b[l]<-0
  }
  if (result_mse$y_c[l]=="b"){
    result_mse$a[l]<-0
    result_mse$b[l]<-1
  }
}
result_mse$mse=((result_mse$pred_a-result_mse$a)^2+(result_mse$pred_b-result_mse$b)^2)/2
rmse_rf[o]<-sqrt(mean(result_mse$mse))
mb<-mb[-length(mb)]
mb<-union(mb,"y_n")

# Train the LR model using training data and predict on the test set.

train_lr<-read.csv("C:\\hybrid model\\workspace\\pima\\analysis\\train.csv",head=T)
train_lr<-train_lr[mb]
test_lr<-read.csv("C:\\hybrid model\\workspace\\pima\\analysis\\test.csv",head=T)
test_lr<-test_lr[mb]
level<-read.csv("C:\\hybrid model\\workspace\\pima\\disc_pima.csv",head=T)
```

```
level<-level[mb]
for (v in 1:n_mb){
  train_lr[,c(mb[v])]<-factor(train_lr[,c(mb[v])], levels=levels(level[,c(mb[v])]))
}
for (v in 1:n_mb){
  test_lr[,c(mb[v])]<-factor(test_lr[,c(mb[v])], levels=levels(level[,c(mb[v])]))
}
fitted<-multinom(y_n~.,data=train_lr)
pred<-predict(fitted, test_lr)
pred<-as.data.frame(pred)
result<-cbind(test_lr,pred)
right<-result[ which(result$y_n==result$pred),]
score_lr[o]<-nrow(right)/nrow(result)
pred_mse<-predict(fitted, test_lr,"probs")
pred_mse<-as.data.frame(pred_mse)
result_mse<-test_nb
result_mse$pred_a<-pred_mse[,1]
result_mse$pred_b<-1-pred_mse[,1]
for(l in 1:(nrow(result_mse))){
  if (result_mse$y_c[l]=="a"){
    result_mse$a[l]<-1
    result_mse$b[l]<-0
  }
  if (result_mse$y_c[l]=="b"){
    result_mse$a[l]<-0
    result_mse$b[l]<-1
  }
}
result_mse$mse=((result_mse$pred_a-result_mse$a)^2+(result_mse$pred_b-result_mse$b)^2)/2
rmse_lr[o]<-sqrt(mean(result_mse$mse))

# Train the l1 penalized LR using training data and predict on the test set.

X=model.matrix(~.,data=train_lr[,-ncol(train_lr)])[,-1]
X_test=model.matrix(~.,data=test_lr[,-ncol(test_lr)])[,-1]
fitted<-cv.glmnet(X,train_lr[,ncol(train_lr)],family="multinomial",type.measure="mse")
pred<-predict(fitted,X_test,s="lambda.min","class")
pred<-as.data.frame(pred)
result<-cbind(test_lr,pred)
right<-result[ which(result$y_n==result$"1"),]
score_lasso[o]<-nrow(right)/nrow(result)
pred_mse<-predict(fitted,X_test,s="lambda.min","response")
pred_mse<-as.data.frame(pred_mse)
result_mse<-test_nb
```

```
  result_mse$pred_a<-pred_mse[,2]
  result_mse$pred_b<-pred_mse[,1]
  for(l in 1:(nrow(result_mse))){
    if (result_mse$y_c[l]=="a"){
      result_mse$a[l]<-1
      result_mse$b[l]<-0
    }
    if (result_mse$y_c[l]=="b"){
      result_mse$a[l]<-0
      result_mse$b[l]<-1
    }
  }
  result_mse$mse=((result_mse$pred_a-result_mse$a)^2+(result_mse$pred_b-result_mse$b)^2)/2
  rmse_lasso[o]<-sqrt(mean(result_mse$mse))
}
```

This is the Pima Indians Diabetes Dataset
==========
X1,X2,X3,X4,X5,X6,X7,X8,y_c,y_n
a,c,b,b,b,b,b,b,a,1
a,a,a,b,a,a,a,b,b,0
b,d,a,b,a,a,b,b,a,1
a,a,a,a,a,b,a,a,b,0
a,c,a,b,b,b,b,b,a,1
a,b,b,a,b,a,a,b,b,0
a,a,a,b,a,b,a,a,a,1
b,b,b,b,b,b,a,b,b,0
a,d,b,b,b,b,a,b,a,1
b,b,b,b,b,b,a,b,a,1
a,b,b,b,b,b,a,b,b,0
b,d,b,b,b,b,b,b,a,1
b,c,b,b,b,a,b,b,b,0
a,d,a,a,b,b,a,b,a,1
a,d,b,a,b,a,b,b,a,1
b,b,b,b,b,b,a,b,a,1
a,b,b,b,b,b,b,b,a,1
b,b,b,a,b,b,a,b,a,1
a,b,a,b,a,b,a,b,b,0
a,b,b,b,a,b,b,b,a,1
a,b,b,b,b,b,b,a,b,0
b,a,b,b,a,b,a,b,b,0
b,d,b,b,b,b,a,b,a,1
b,b,b,b,b,b,a,b,a,1
b,c,b,b,b,b,a,b,a,1
b,b,b,b,b,b,a,b,a,1
b,c,b,b,b,b,a,b,a,1
a,a,a,a,b,a,a,a,b,0
b,c,b,a,b,a,a,b,b,0
a,b,b,a,b,b,a,b,b,0
a,b,b,b,a,b,b,b,b,0
a,d,b,b,b,b,b,a,a,1
a,a,a,a,a,a,a,a,b,0
a,a,b,b,a,a,a,a,b,0
b,b,b,b,a,b,a,b,b,0
a,b,a,b,b,a,b,b,b,0
b,c,b,a,a,b,a,b,b,0
b,b,b,b,a,b,b,b,a,1
a,a,a,b,b,b,a,a,a,1
a,b,b,b,b,b,b,b,a,1
a,d,a,b,a,b,a,a,b,0
b,c,b,b,b,b,b,b,b,0
b,b,b,a,b,a,a,b,b,0
b,d,b,b,b,b,b,b,a,1
b,d,a,b,a,b,a,b,b,0
a,d,a,b,a,b,b,a,a,1
a,c,a,a,a,b,b,b,b,0
a,a,b,b,a,b,b,a,b,0

b,b,a,b,b,b,a,b,a,1
b,b,a,b,b,b,a,a,b,0
a,b,b,a,a,a,a,a,b,0
a,b,a,a,a,a,a,a,b,0
a,a,a,a,a,a,a,b,b,0
b,d,b,b,b,b,a,b,a,1
b,c,a,b,b,b,b,b,b,0
a,a,a,a,b,a,a,a,b,0
b,d,a,b,b,b,a,b,a,1
a,b,b,b,b,b,b,b,b,0
a,c,b,b,a,b,b,b,b,0
a,b,a,b,b,b,a,a,b,0
a,a,a,b,a,a,a,a,b,0
b,c,b,a,b,b,a,b,a,1
a,a,a,a,b,a,b,b,b,0
a,c,a,b,b,a,b,a,b,0
b,b,a,a,b,b,a,b,a,1
a,a,b,b,a,b,a,b,b,0
a,b,b,b,b,b,b,b,a,1
a,b,b,a,a,b,b,b,b,0
a,a,a,a,a,a,a,a,b,0
a,c,b,b,a,b,a,a,b,0
a,b,a,a,a,b,b,a,a,1
a,c,a,b,b,b,a,a,b,0
b,b,b,b,a,b,b,b,a,1
a,c,b,a,b,b,a,a,b,0
a,a,b,b,b,b,a,a,b,0
a,b,a,a,b,a,a,a,b,0
b,a,b,b,a,b,a,b,b,0
a,a,b,b,a,b,a,a,b,0
a,c,a,b,b,b,a,a,a,1
a,b,a,a,a,a,a,a,b,0
a,b,a,a,b,a,a,a,b,0
a,a,b,b,a,a,a,a,b,0
b,a,b,b,a,b,b,b,b,0
a,b,a,b,b,a,a,a,b,0
a,c,b,b,b,b,a,b,a,1
a,b,b,b,b,b,b,a,b,0
b,b,b,b,b,b,a,b,b,0
a,b,a,b,a,b,a,a,b,0
b,c,b,b,b,b,a,b,a,1
a,b,a,a,b,a,a,a,b,0
a,a,a,b,a,a,a,a,b,0
a,b,b,a,b,b,a,b,b,0
b,a,b,b,a,b,a,b,b,0
a,c,b,a,b,a,a,b,a,1
a,c,b,a,a,a,b,a,b,0
a,c,b,b,b,b,a,b,b,0
a,a,a,b,a,b,a,a,b,0
a,a,a,a,a,a,a,a,b,0
a,a,a,b,a,b,a,a,b,0

a,b,b,b,b,b,a,b,a,1
a,d,b,b,b,b,b,b,a,1
a,c,a,b,a,a,a,a,b,0
a,b,b,b,a,a,a,a,b,0
a,a,b,a,a,a,a,a,b,0
a,a,a,a,b,b,b,a,b,0
a,b,a,b,b,b,b,a,b,0
a,a,b,b,a,a,a,a,b,0
a,c,a,b,b,b,a,b,b,0
a,a,a,b,a,b,a,a,b,0
a,a,b,b,a,b,a,a,a,1
a,d,b,b,b,b,a,a,a,1
b,d,a,b,b,b,b,b,a,1
a,a,b,b,a,b,a,a,b,0
a,a,a,a,a,b,a,a,b,0
b,d,a,b,b,b,b,b,a,1
a,c,b,a,b,b,b,b,a,1
a,b,b,a,b,b,a,b,a,1
a,a,a,a,a,b,b,a,b,0
a,a,a,a,b,b,a,a,b,0
a,a,b,a,a,a,a,a,b,0
a,d,b,b,a,b,b,a,a,1
a,b,a,b,a,b,a,a,b,0
a,b,b,b,a,b,a,a,b,0
a,c,b,a,a,a,a,b,b,0
a,b,b,b,b,b,a,a,a,1
a,a,a,b,a,b,a,a,a,1
a,b,b,b,b,b,a,b,b,0
a,b,a,b,a,b,a,a,b,0
a,b,b,b,b,b,a,b,a,1
a,b,b,b,b,b,b,b,a,1
a,d,b,a,b,b,a,b,a,1
b,b,a,b,b,b,b,b,a,1
a,d,a,b,b,b,a,b,a,1
b,a,b,b,a,b,a,b,b,0
a,a,a,a,a,a,b,a,b,0
a,b,a,a,b,b,a,b,b,0
a,b,b,b,a,b,b,a,b,0
a,a,a,b,a,b,b,a,b,0
a,c,b,b,b,b,b,b,b,0
a,b,b,b,b,b,a,a,b,0
a,c,b,a,b,a,a,b,b,0
a,b,b,b,b,b,a,b,b,0
a,b,a,b,a,b,a,a,b,0
b,b,a,b,b,b,a,b,a,1
a,c,a,b,b,b,a,a,b,0
a,b,b,a,a,b,b,a,b,0
b,a,b,b,a,b,a,b,b,0
a,b,a,b,b,b,b,b,b,0
a,c,b,a,b,b,a,b,b,0
a,a,b,a,a,a,a,a,b,0

a,c,b,b,b,b,a,a,b,0
a,b,a,b,a,a,a,b,b,0
b,d,b,b,b,b,b,b,a,1
a,c,b,b,b,b,b,a,b,0
b,d,b,a,a,b,a,b,a,1
b,c,b,b,b,b,a,b,a,1
a,a,a,a,a,a,b,a,b,0
a,b,a,a,b,a,b,a,b,0
a,a,b,a,a,b,a,a,b,0
b,d,b,b,b,b,b,b,a,1
a,c,b,b,a,b,a,b,b,0
b,b,b,b,a,b,a,b,b,0
a,b,b,b,b,b,a,a,b,0
a,b,a,a,a,b,a,a,b,0
a,c,b,b,b,b,b,b,a,1
a,b,b,a,b,b,b,b,a,1
a,c,a,b,a,b,a,a,b,0
a,b,a,b,a,b,b,b,b,0
a,b,a,b,a,b,a,b,b,0
a,b,b,a,a,b,a,b,b,0
a,b,b,b,b,b,a,b,a,1
a,c,b,a,b,b,b,b,a,1
a,a,b,a,a,b,b,a,b,0
a,a,a,b,a,b,b,a,b,0
a,a,a,b,a,b,a,b,b,0
b,d,b,b,b,b,b,b,a,1
a,a,b,b,a,b,a,b,b,0
a,c,b,b,b,b,a,a,a,1
a,c,b,b,b,b,a,b,b,0
a,c,b,b,b,b,b,b,a,1
a,a,b,b,a,a,a,b,b,0
a,b,a,a,a,b,b,a,b,0
a,b,b,a,a,b,a,a,b,0
a,a,a,a,b,a,a,a,b,0
a,c,b,b,b,b,a,b,b,0
b,d,a,b,a,b,b,b,a,1
b,d,a,b,b,b,b,b,a,1
a,c,b,b,a,b,b,b,a,1
b,b,b,b,b,b,b,b,a,1
a,c,b,b,b,b,a,a,a,1
a,b,a,b,a,a,a,a,b,0
b,b,b,b,a,b,a,b,b,0
b,d,a,b,b,b,a,b,a,1
b,c,b,b,b,b,b,b,a,1
b,a,a,a,a,a,a,b,b,0
a,d,b,b,b,b,a,b,a,1
a,b,a,a,b,a,a,a,b,0
a,b,a,a,a,a,b,a,a,1
a,b,a,b,a,b,b,a,a,1
a,c,a,b,b,b,a,b,a,1
a,b,b,a,a,b,b,a,b,0

a,c,b,b,a,b,a,a,b,0
a,b,a,a,a,a,b,b,b,0
a,a,b,a,a,a,a,a,b,0
a,b,b,b,b,b,a,b,b,0
a,b,b,b,a,a,a,a,b,0
b,d,b,b,b,b,b,b,a,1
a,d,b,b,b,b,a,b,a,1
a,a,a,b,a,b,a,a,b,0
b,d,b,b,b,b,a,b,a,1
a,a,a,a,a,b,a,a,b,0
a,c,b,b,a,b,a,a,b,0
b,d,b,b,a,b,a,b,b,0
a,c,a,b,b,b,a,a,a,1
b,b,b,b,b,b,a,b,a,1
b,c,b,b,b,b,b,b,a,1
a,b,a,b,b,b,a,a,a,1
a,b,a,b,b,b,a,b,b,0
a,a,b,a,b,b,b,b,a,1
a,b,a,b,b,b,a,b,a,1
a,d,a,b,b,b,b,a,a,1
a,d,b,b,b,b,b,b,a,1
b,b,b,b,b,a,a,b,b,0
b,c,a,b,b,b,b,b,b,0
a,b,a,a,a,a,b,a,b,0
a,a,b,b,a,b,a,a,b,0
a,b,b,b,b,b,a,a,b,0
a,d,a,b,b,b,b,a,a,1
a,d,b,b,b,b,b,b,b,0
a,b,b,b,a,b,a,a,b,0
a,c,b,b,b,b,b,a,a,1
a,c,b,b,b,b,a,b,a,1
a,a,b,b,a,a,b,a,b,0
a,b,a,a,a,b,a,b,b,0
a,a,a,b,a,b,a,a,b,0
a,d,b,b,b,b,a,a,a,1
b,d,b,a,b,b,b,b,a,1
a,d,b,b,b,b,b,a,a,1
b,d,b,a,b,b,b,b,a,1
a,b,b,b,a,a,b,a,b,0
a,a,a,b,b,b,a,a,b,0
a,a,b,b,a,b,a,a,b,0
a,c,a,b,a,a,a,a,a,1
a,b,a,a,b,a,b,b,a,1
a,c,b,b,b,b,a,b,b,0
b,d,b,a,b,b,b,b,a,1
b,b,a,b,a,b,a,b,b,0
a,d,b,b,b,b,a,a,b,0
b,b,b,b,b,b,a,b,b,0
a,b,b,a,b,b,a,a,b,0
b,b,a,a,a,b,a,b,b,0
a,c,b,a,a,b,a,a,b,0

a,a,b,a,a,a,a,a,b,0
a,a,a,b,a,b,a,a,b,0
b,a,a,a,b,b,b,b,a,1
a,b,a,b,a,b,b,a,a,1
a,b,a,b,b,b,b,b,b,0
a,b,a,a,b,b,a,a,b,0
a,d,a,a,b,a,b,a,b,0
b,d,b,b,b,b,b,b,a,1
a,d,a,a,b,b,a,b,b,0
a,c,b,b,b,b,b,a,a,1
a,a,b,b,b,b,b,a,b,0
a,c,b,a,b,b,a,b,b,0
a,b,a,b,b,b,a,b,a,1
a,a,b,a,a,b,b,b,b,0
a,c,b,a,b,b,b,a,a,1
a,c,a,b,a,b,b,a,b,0
a,b,a,b,a,a,a,a,b,0
a,c,b,b,a,b,a,a,a,1
b,b,b,b,b,b,b,b,a,1
a,b,a,b,a,a,a,a,b,0
a,b,b,a,b,a,a,b,b,0
a,a,b,b,a,b,a,a,b,0
b,b,b,a,a,b,a,b,b,0
a,b,b,b,a,b,b,a,b,0
b,b,a,b,a,a,a,b,a,1
a,b,a,a,b,b,a,a,b,0
a,b,b,a,a,a,b,b,b,0
a,b,a,a,b,a,b,a,b,0
a,c,b,b,b,b,a,a,a,1
b,c,b,b,b,b,a,b,b,0
b,c,b,a,b,b,a,b,b,0
b,d,b,b,b,b,a,b,a,1
a,b,b,b,b,a,a,b,a,1
b,c,b,b,b,a,b,b,b,0
a,d,b,b,b,b,b,b,b,0
a,b,b,b,b,b,b,b,a,1
a,a,a,a,a,a,a,a,b,0
a,b,b,b,a,b,a,b,b,0
a,a,b,b,a,b,a,a,b,0
a,b,a,b,a,b,b,a,a,1
a,c,b,b,b,b,b,b,a,1
a,c,a,b,b,b,b,a,a,1
a,d,a,a,a,a,a,b,b,0
a,c,a,b,b,b,b,a,b,0
a,c,b,b,b,b,a,b,a,1
a,b,b,b,b,b,a,a,b,0
b,b,b,b,b,b,a,b,a,1
b,b,b,b,a,a,b,b,b,0
a,d,b,b,b,b,b,b,a,1
a,c,a,b,b,b,a,a,a,1
a,a,b,b,a,b,a,b,b,0

a,b,b,b,b,b,a,a,a,1
a,c,b,b,a,a,a,b,b,0
a,b,b,b,a,b,a,b,b,0
b,d,a,a,b,a,a,b,a,1
a,c,a,a,b,a,a,a,b,0
a,c,a,a,b,b,b,a,a,1
a,b,a,b,b,b,b,b,a,1
a,a,a,b,a,a,a,b,b,0
a,b,b,b,b,b,b,a,b,0
a,d,b,a,a,a,a,a,a,1
a,b,a,a,a,b,b,a,b,0
b,b,b,b,b,b,b,b,a,1
a,b,a,a,a,b,a,a,b,0
a,a,b,a,a,a,a,b,b,0
a,d,b,b,b,b,a,b,a,1
a,b,a,b,b,b,a,a,b,0
a,d,b,b,b,a,a,b,a,1
a,c,a,a,b,b,a,b,b,0
a,b,b,b,b,b,a,a,a,1
a,b,b,a,a,b,a,b,a,1
b,c,b,b,a,a,b,b,a,1
a,b,b,b,b,b,a,a,b,0
a,d,b,a,b,a,a,a,b,0
a,b,a,b,b,b,b,b,a,1
b,d,b,b,b,b,a,b,b,0
a,b,b,b,b,b,a,a,a,1
a,b,b,b,a,b,a,b,b,0
b,b,b,a,a,a,b,b,b,0
a,a,a,a,a,b,a,a,b,0
a,d,b,b,a,b,a,b,a,1
b,b,b,b,a,a,a,b,b,0
a,a,a,a,a,a,a,a,b,0
a,d,b,b,b,b,a,a,b,0
a,b,b,a,b,b,b,b,b,0
a,b,b,a,b,b,a,b,a,1
b,c,b,b,b,b,b,b,a,1
b,d,b,b,b,b,a,b,a,1
a,c,b,a,a,a,a,a,b,0
a,a,b,a,a,a,b,b,b,0
a,b,a,b,a,b,a,a,b,0
a,b,b,b,b,b,a,b,b,0
b,a,b,a,a,b,a,b,b,0
b,b,b,b,a,b,a,b,b,0
a,c,a,a,a,b,b,a,b,0
a,b,a,a,b,a,a,a,b,0
a,a,a,a,a,a,a,a,b,0
a,c,b,b,b,b,a,b,a,1
a,a,b,a,a,b,a,b,b,0
a,c,b,b,b,b,a,b,b,0
a,a,b,b,a,b,a,b,b,0
a,a,a,a,a,a,b,a,b,0

a,a,b,a,b,b,b,a,b,0
b,d,b,a,b,b,a,b,a,1
a,b,a,b,b,b,b,a,a,1
b,c,a,b,b,b,b,b,a,1
b,a,b,b,a,b,a,b,b,0
a,d,b,b,b,b,b,b,a,1
a,d,a,b,b,b,b,b,a,1
a,d,b,b,b,b,a,b,b,0
a,b,b,b,b,b,a,b,b,0
a,c,b,b,b,b,a,b,a,1
a,c,b,b,b,b,a,b,b,0
a,a,a,b,a,b,a,b,b,0
a,b,b,b,b,b,a,b,a,1
a,b,a,a,a,a,a,a,b,0
a,a,b,a,a,b,a,a,b,0
a,c,b,b,b,b,a,b,a,1
a,d,b,b,b,b,b,a,a,1
a,b,a,a,a,a,b,a,b,0
a,a,a,a,a,b,b,a,b,0
a,b,a,b,a,b,a,a,b,0
a,b,a,b,b,b,b,a,b,0
b,c,b,b,b,b,b,b,a,1
a,a,b,a,a,a,a,a,b,0
a,a,a,b,a,b,a,a,b,0
a,d,b,a,b,b,a,b,a,1
a,a,b,b,a,b,b,b,b,0
a,b,b,b,a,b,b,a,b,0
a,b,a,a,b,a,a,a,b,0
a,b,a,a,b,a,b,a,b,0
a,a,a,a,a,a,b,a,b,0
a,b,b,b,b,a,a,a,b,0
a,b,a,a,a,a,a,a,b,0
a,b,b,b,b,b,b,b,a,1
b,b,b,b,b,b,a,b,a,1
a,c,b,b,b,b,a,b,a,1
a,b,a,a,a,b,b,a,b,0
a,b,a,b,b,b,a,b,b,0
a,d,b,b,b,b,a,a,a,1
a,c,a,a,b,a,a,a,b,0
a,b,b,a,a,a,a,b,b,0
a,d,b,b,a,b,b,b,a,1
a,b,a,b,b,b,b,a,b,0
a,a,a,b,b,a,b,b,b,0
a,c,a,b,b,b,a,a,a,1
a,a,b,a,b,a,a,a,b,0
a,d,b,b,b,b,a,a,a,1
a,a,a,b,b,b,a,b,a,1
a,c,a,b,b,a,a,b,b,0
a,c,b,b,a,b,a,b,a,1
b,a,b,b,a,b,a,b,b,0
a,d,a,b,b,b,a,b,a,1

a,b,a,b,b,b,a,a,b,0
a,b,b,b,a,b,a,b,a,1
a,b,a,b,a,a,a,a,b,0
b,d,b,a,b,a,b,b,a,1
a,d,a,b,b,b,b,a,a,1
a,b,b,b,b,b,b,a,b,0
a,b,b,b,b,b,b,a,b,0
a,c,b,a,b,b,b,a,b,0
a,c,b,a,a,a,a,a,b,0
a,c,a,b,b,b,b,a,a,1
a,d,b,b,b,b,a,a,a,1
a,a,a,a,a,a,b,a,b,0
a,c,b,b,b,b,b,b,a,1
a,a,a,b,a,a,b,a,b,0
a,c,a,b,b,a,a,a,a,1
a,b,b,b,b,b,a,a,b,0
a,a,a,a,a,a,b,a,b,0
a,b,a,b,a,b,a,a,b,0
a,b,a,a,a,b,a,a,b,0
b,c,b,b,b,b,a,b,a,1
a,d,b,b,b,b,a,b,a,1
a,a,b,a,a,b,a,a,b,0
a,d,a,b,b,b,a,b,a,1
a,c,b,b,b,b,a,a,b,0
a,a,b,b,b,b,a,b,a,1
a,a,a,b,a,a,a,a,b,0
a,a,b,a,a,b,b,b,b,0
a,a,b,a,a,b,a,a,b,0
a,c,b,b,b,a,a,b,b,0
a,a,a,a,b,a,b,b,b,0
a,c,b,b,b,b,a,b,a,1
b,c,b,b,a,b,a,b,b,0
a,c,b,a,b,b,a,a,b,0
a,a,b,a,a,a,a,a,b,0
a,b,b,a,a,b,b,b,b,0
a,d,b,b,b,b,a,b,a,1
a,a,a,a,a,b,a,a,b,0
a,b,a,b,b,b,a,a,b,0
b,b,b,b,b,b,b,b,a,1
a,b,a,a,b,b,a,b,a,1
a,d,b,b,a,b,b,a,a,1
a,b,b,a,a,a,b,a,b,0
a,a,b,b,a,b,a,a,b,0
a,b,a,b,a,b,a,a,a,1
a,b,b,a,a,b,a,a,b,0
a,a,a,a,a,a,a,a,b,0
a,c,b,b,b,b,b,a,a,1
a,a,a,b,b,b,a,a,b,0
a,b,b,b,a,a,b,b,b,0
a,b,a,b,a,b,a,a,b,0
b,d,a,b,b,b,a,b,a,1

a,c,a,a,a,a,b,b,b,0
a,a,a,b,a,b,a,a,b,0
b,c,b,b,b,b,b,b,a,1
b,c,b,b,a,a,a,b,b,0
b,b,b,a,a,a,b,b,b,0
a,a,a,b,b,a,a,a,b,0
b,a,b,b,a,b,b,b,b,0
a,a,b,b,b,b,a,b,b,0
b,b,b,a,b,a,b,b,b,0
a,b,a,a,a,a,a,a,b,0
a,a,a,a,a,b,a,a,b,0
a,a,a,b,a,b,b,a,b,0
b,b,b,a,b,b,a,b,a,1
a,c,b,b,b,b,b,a,b,0
a,c,b,b,a,b,b,a,b,0
a,c,b,b,b,b,a,a,b,0
a,b,a,b,a,b,a,a,b,0
b,c,b,b,b,b,a,b,b,0
a,b,a,b,a,b,a,a,b,0
a,c,b,b,b,a,a,b,b,0
a,b,b,b,b,b,b,b,a,1
b,b,b,a,b,a,a,b,b,0
b,b,b,b,a,a,a,b,b,0
a,c,b,b,a,b,a,b,b,0
a,d,b,b,b,b,a,b,a,1
a,b,b,b,a,b,a,b,b,0
a,a,a,a,a,b,a,a,b,0
a,a,b,b,b,b,a,a,b,0
a,c,a,b,b,b,b,b,a,1
a,c,a,b,b,b,a,a,a,1
a,c,a,b,b,b,b,a,b,0
a,d,b,b,b,b,b,b,b,0
a,a,b,a,b,a,a,a,b,0
b,d,b,b,a,a,b,b,b,0
a,a,a,b,a,b,b,a,b,0
a,a,b,b,a,b,a,b,b,0
a,a,a,b,b,b,a,b,b,0
a,b,b,a,b,b,b,b,a,1
a,a,b,a,b,b,a,a,b,0
a,d,b,a,b,a,a,b,b,0
a,b,a,b,b,a,a,b,b,0
a,a,b,a,a,b,b,a,b,0
b,d,b,b,b,a,a,b,a,1
a,c,b,b,b,b,b,b,b,0
a,b,b,a,a,a,a,a,b,0
a,a,b,b,a,b,a,a,b,0
a,d,a,b,b,b,b,b,a,1
b,a,a,b,a,b,b,b,b,0
a,a,b,b,b,b,a,b,b,0
b,a,b,a,a,b,a,b,b,0
a,d,b,b,a,b,a,b,a,1

a,c,a,a,b,b,b,a,b,0
a,a,a,a,a,b,b,a,b,0
b,b,b,a,a,a,a,b,b,0
b,a,b,b,a,b,a,b,a,1
a,c,a,a,b,a,a,a,b,0
b,a,a,b,a,a,a,b,b,0
a,a,a,a,b,a,a,a,b,0
a,a,a,a,a,a,a,a,b,0
a,d,b,a,a,b,a,a,a,1
b,c,b,b,b,b,b,b,a,1
b,b,b,a,a,b,a,b,b,0
b,a,a,b,a,b,a,b,b,0
a,c,b,a,b,a,b,b,b,0
a,a,b,b,a,a,a,a,b,0
a,b,b,b,b,b,a,a,b,0
a,b,a,b,a,b,a,a,b,0
b,c,b,b,a,b,b,b,a,1
a,b,a,b,b,b,a,a,b,0
a,a,a,a,b,a,a,a,b,0
a,a,a,a,a,a,a,a,b,0
a,b,b,a,a,a,a,a,b,0
a,b,a,b,b,b,a,a,b,0
a,b,a,b,b,a,b,b,b,0
a,b,a,a,a,b,b,a,b,0
a,b,b,b,b,b,b,a,b,0
a,a,a,b,a,b,b,b,b,0
a,a,a,b,b,b,a,b,b,0
a,a,a,b,a,b,b,a,b,0
a,c,a,b,b,b,a,a,a,1
a,b,b,a,a,b,a,b,b,0
a,a,a,a,b,a,b,b,b,0
a,b,b,b,b,b,b,a,b,0
a,c,b,b,b,b,b,b,a,1
b,b,b,b,b,b,b,b,a,1
a,c,b,b,b,b,b,a,a,1
b,a,b,b,b,b,b,b,a,1
a,a,b,a,a,b,a,a,b,0
a,a,b,b,a,b,a,b,b,0
b,d,b,b,b,b,a,b,a,1
a,d,b,b,b,b,b,b,a,1
a,c,a,a,b,b,a,a,b,0
a,d,b,b,a,b,a,b,b,0
a,d,b,b,a,b,b,b,b,0
a,b,b,b,a,b,a,a,b,0
a,a,a,b,a,b,b,a,b,0
a,b,b,b,b,b,a,b,b,0
a,a,a,b,a,b,a,a,b,0
a,a,a,a,b,b,a,a,b,0
b,b,b,b,b,a,a,b,b,0
a,a,b,b,a,b,a,b,b,0
b,b,b,a,b,b,a,b,b,0

b,b,a,b,b,b,a,b,b,0
b,a,b,a,b,b,a,b,b,0
a,b,b,a,b,b,a,b,a,1
a,d,a,b,b,b,a,a,a,1
a,a,a,b,a,b,a,a,b,0
a,a,a,a,a,a,a,b,b,0
a,a,b,a,b,b,b,a,b,0
a,a,a,a,a,a,b,a,b,0
a,a,b,b,a,b,a,a,b,0
a,a,a,b,b,b,a,b,b,0
a,c,b,b,b,b,a,b,b,0
a,b,a,b,b,b,a,b,a,1
a,a,b,a,b,b,a,b,b,0
a,c,b,b,a,a,a,a,b,0
a,b,a,b,a,b,a,a,b,0
a,a,a,a,b,b,a,a,b,0
a,c,b,b,b,b,b,a,b,0
a,b,a,b,a,b,a,a,b,0
a,b,a,a,b,a,b,b,b,0
a,b,b,b,a,b,b,a,a,1
b,c,a,b,b,a,a,b,b,0
a,d,b,b,b,b,b,b,a,1
a,c,b,b,b,b,a,a,a,1
a,b,a,b,a,a,a,a,b,0
b,b,b,a,a,a,a,b,b,0
b,b,b,b,a,b,a,b,b,0
b,b,b,b,b,b,b,b,a,1
a,a,a,a,a,a,a,a,b,0
b,c,a,b,b,b,a,b,a,1
a,b,a,b,a,a,a,b,b,0
a,d,b,b,b,b,b,b,a,1
a,a,a,b,b,a,a,a,b,0
b,b,b,b,b,b,b,b,a,1
a,b,b,b,b,b,a,a,b,0
a,c,b,b,b,b,a,b,a,1
a,a,a,a,b,b,b,a,b,0
a,b,b,b,b,b,b,b,b,0
a,d,b,a,b,b,b,a,a,1
a,a,b,b,a,b,a,b,b,0
a,a,a,a,a,b,b,a,b,0
a,d,b,a,b,b,b,a,b,1
a,b,a,a,b,a,a,a,b,0
a,b,b,a,b,a,a,a,b,0
a,a,b,a,b,a,a,a,b,0
a,b,b,b,b,b,a,b,b,0
b,c,b,b,b,b,b,b,a,1
a,d,b,b,b,b,a,b,a,1
a,b,a,b,b,b,a,a,b,0
a,d,b,b,b,b,b,a,a,1
a,a,a,b,a,a,a,a,b,0
a,c,b,b,b,b,a,a,b,0

a,b,a,a,b,a,a,a,b,0
a,b,a,a,b,b,a,a,b,0
a,d,a,a,b,b,b,b,a,1
b,d,b,b,b,b,b,b,a,1
a,b,b,b,b,b,b,a,b,0
b,c,b,b,b,b,b,b,a,1
a,b,b,b,a,a,a,a,b,0
a,b,b,a,a,b,a,b,b,0
a,a,a,a,a,a,a,a,b,0
b,b,b,b,b,b,b,b,a,1
a,b,b,b,b,b,a,a,a,1
a,b,b,b,b,b,a,a,b,0
a,a,b,a,b,a,b,a,b,0
a,d,b,a,a,b,b,b,b,0
a,a,b,b,b,b,a,a,b,0
a,b,a,a,a,b,a,a,b,0
a,a,b,b,a,b,a,b,b,0
a,b,a,b,b,a,a,a,b,0
a,c,b,b,b,b,a,a,b,0
a,c,b,b,a,b,a,b,b,0
a,a,a,a,a,a,a,a,b,0
b,b,a,b,b,b,b,b,a,1
a,b,b,b,a,b,a,a,b,0
a,b,a,b,a,a,a,a,b,0
a,c,b,a,b,b,a,a,b,0
b,a,a,b,b,a,a,b,b,0
b,b,b,b,b,b,a,b,a,1
a,b,b,b,b,b,a,b,b,0
a,a,b,a,a,b,b,a,b,0
b,a,b,b,a,b,b,b,a,1
a,b,b,a,a,a,a,a,b,0
a,b,b,a,a,b,b,a,b,0
a,c,b,a,b,b,a,a,b,0
a,c,b,a,b,b,a,b,a,1
a,a,a,b,a,b,b,b,b,0
a,b,b,b,b,b,b,a,b,0
a,d,b,b,b,b,a,b,b,0
a,d,b,a,b,a,a,b,a,1
a,d,a,b,b,b,a,a,a,1
b,c,b,b,b,b,a,b,a,1
a,b,a,b,a,a,a,a,b,0
a,a,a,b,a,a,a,a,b,0
a,b,a,a,a,b,a,a,b,0
a,b,b,b,a,b,a,a,b,0
a,b,a,a,b,a,a,a,b,0
a,b,b,b,b,b,a,a,b,0
a,d,a,b,b,b,a,a,a,1
a,b,a,b,a,a,a,a,b,0
a,b,b,b,b,b,b,b,b,0
b,b,b,b,a,b,a,b,b,0
a,a,b,b,a,b,b,a,a,1

b,d,b,b,a,b,a,b,b,0
a,d,b,b,b,b,b,a,a,1
b,d,b,b,b,b,a,b,a,1
b,c,b,b,b,b,b,b,a,1
a,b,a,b,b,b,a,b,a,1
a,b,b,b,b,b,a,a,b,0
a,c,b,a,b,b,a,b,a,1
b,b,b,b,b,b,a,b,a,1
a,a,a,b,b,b,a,b,b,0
b,c,b,b,a,b,a,b,b,0
a,d,a,b,b,b,b,b,b,0
a,a,a,a,a,a,b,a,b,0
b,a,b,a,a,b,a,b,b,0
a,b,b,b,b,b,b,a,b,0
b,a,b,a,b,b,b,b,b,0
a,d,b,b,b,b,a,b,a,1
b,d,b,b,b,a,a,b,a,1
a,a,a,a,b,b,a,a,b,0
a,b,a,b,a,b,a,a,a,1
a,b,a,a,b,a,b,a,b,0
a,a,a,b,a,a,a,a,b,0
a,d,b,b,b,b,a,a,a,1
a,a,a,b,a,b,a,a,b,0
a,b,b,a,b,b,b,a,a,1
a,c,b,b,b,a,b,b,b,0
a,c,b,b,b,b,b,a,b,0
a,c,a,a,b,a,a,a,b,0
a,b,a,a,a,b,a,b,b,0
a,c,b,b,b,a,b,a,b,0
a,c,b,b,b,b,a,b,a,1
b,b,b,a,b,a,b,b,b,0
b,d,b,a,b,b,a,b,a,1
a,b,b,b,a,b,b,a,b,0
b,c,a,b,b,b,a,b,a,1
a,a,a,b,a,a,a,a,b,0
b,c,b,b,b,b,a,b,a,1
a,d,b,a,b,b,a,b,a,1
a,a,a,b,a,a,a,a,b,0
a,b,b,a,b,b,b,a,b,0
a,b,b,a,a,b,b,a,b,0
a,b,b,b,b,b,a,a,b,0
a,b,b,b,b,b,b,b,a,1
a,d,b,b,a,b,b,b,a,1
a,c,a,b,a,b,a,b,b,0
a,b,b,a,a,b,a,a,b,0
a,a,b,b,a,b,a,a,b,0
b,b,b,b,b,a,a,b,a,1
a,b,a,a,b,b,a,a,b,0
b,d,b,b,a,b,a,b,a,1
a,a,a,b,b,b,b,a,a,1
a,d,a,a,b,b,a,a,b,0

a,b,b,b,a,b,a,b,b,0
b,c,a,b,a,b,a,b,a,1
a,c,a,a,b,a,a,a,b,0
a,b,b,b,a,b,a,b,b,0
b,d,a,b,b,b,b,b,a,1
a,d,b,b,b,b,b,b,a,1
b,a,b,a,a,a,b,b,b,0
a,b,a,b,b,b,a,a,b,0
a,a,b,b,b,b,a,b,a,1
a,a,b,a,b,b,a,b,b,0
a,b,a,b,b,b,a,a,b,0
a,c,a,b,b,b,a,b,a,1
a,b,b,b,a,b,a,b,b,0
a,b,b,a,b,b,a,b,b,0
a,b,b,b,b,b,a,b,b,0
a,b,b,b,b,b,a,a,b,0
a,c,b,b,a,b,a,a,b,0
a,d,b,b,b,a,a,a,b,0
a,a,a,a,a,b,a,a,b,0
a,c,b,a,a,b,a,b,a,1
b,b,b,b,b,b,a,a,a,1
a,d,b,b,b,b,b,a,a,1
a,b,a,b,b,b,a,a,b,0
a,b,b,b,b,a,b,b,b,0
a,a,a,b,b,b,a,a,b,0
a,b,b,b,b,b,a,a,b,0
b,a,b,a,b,b,b,b,b,0
a,a,a,a,b,b,a,a,b,0
a,b,b,a,a,b,a,b,a,1
b,b,b,b,b,b,b,b,a,1
a,b,a,a,a,b,a,a,b,0
a,b,a,a,b,b,a,a,b,0
b,c,b,b,a,b,b,b,a,1
b,c,b,b,b,b,b,b,b,0
b,b,b,b,a,b,a,b,b,0
a,c,b,b,b,b,a,a,a,1
a,a,b,b,a,b,b,b,b,0
a,d,b,a,b,b,a,b,a,1
a,d,a,a,b,a,a,b,a,1
a,c,b,a,b,b,b,a,a,1
a,b,b,b,a,b,a,a,b,0
a,b,a,b,b,a,a,a,b,0
a,d,b,b,b,b,a,a,a,1
b,c,b,b,b,b,a,b,a,1
a,c,b,b,b,b,b,b,a,1
b,c,b,b,a,b,a,b,b,0
a,b,b,b,b,b,a,b,a,1
a,b,b,b,b,b,a,a,b,0
a,d,b,b,b,b,a,b,a,1
a,a,a,b,a,b,b,a,b,0
b,d,b,b,b,b,a,b,a,1

b,a,a,b,a,a,a,b,b,0
b,b,b,b,b,b,a,b,b,0
a,b,b,b,b,b,a,a,b,0
a,b,b,a,b,a,a,b,b,0
a,b,a,b,b,b,a,b,a,1
a,a,b,b,a,b,a,a,b,0