Fully Bayesian T-probit Regression with Heavy-tailed Priors for Selection in High-Dimensional Features with Grouping Structure

A Thesis Submitted to the
College of Graduate Studies and Research
in Partial Fulfillment of the Requirements
for the degree of Doctor of Philosophy
in the Department of Mathematics and Statistics
University of Saskatchewan
Saskatoon

By
Lai Jiang

Permission to Use

Abstract

Feature selection is demanded in many modern scientific research problems that use high-dimensional data. A typical example is to find the genes that are most related to a certain disease (e.g., cancer) from high-dimensional gene expression profiles. There are tremendous difficulties in eliminating a large number of useless or redundant features. The expression levels of genes have structure; for example, a group of co-regulated genes that have similar biological functions tend to have similar mRNA expression levels. Many statistical methods have been proposed to take the grouping structure into consideration in feature selection and regression, including Group LASSO, Supervised Group LASSO, and regression on group representatives. In this thesis, we propose to use a sophisticated Markov chain Monte Carlo method (Hamiltonian Monte Carlo with restricted Gibbs sampling) to fit T-probit regression with heavy-tailed priors to make selection in the features with grouping structure. We will refer to this method as fully Bayesian T-probit. The main feature of fully Bayesian T-probit is that it can make feature selection within groups automatically without a pre-specification of the grouping structure and more efficiently discard noise features than LASSO (Least Absolute Shrinkage and Selection Operator). Therefore, the feature subsets selected by fully Bayesian T-probit are significantly more sparse than subsets selected by many other methods in the literature. Such succinct feature subsets are much easier to interpret or understand based on existing biological knowledge and further experimental investigations. In this thesis, we use simulated and real datasets to demonstrate that the predictive performances of the more sparse feature subsets selected by fully Bayesian T-probit are comparable with the much larger feature subsets selected by plain LASSO, Group LASSO, Supervised Group LASSO, random forest, penalized logistic regression and t-test. In addition, we demonstrate that the succinct feature subsets selected by fully Bayesian T-probit have significantly better predictive power than the feature subsets of the same size taken from the top features selected by the aforementioned methods.

Acknowledgements

I would like to acknowledge and express my sincere thanks and gratitude to my supervisor Dr. Longhai Li for his continual guidance, invaluable support, and encouragement during my studies. I am indebted to Dr. Li, who opened my eyes to Markov chain Monte Carlo methods. I would also like to extend my appreciation to the other committee members for their suggestions and comments.

To my parents, Yun and Chuanbin.

Contents

List of Tables

List of Figures

Chapter 1
Introduction

1.1 Feature Selection in High-throughput Data

Today, high-throughput biotechnologies (such as microarrays and RNA sequencing) can easily measure the expression levels of thousands of genes. An important challenge in genomic research is to identify genes that are relevant to a categorical variable (called a "response" hereafter); for example, an indicator of cancer stage. Once such genes are found and verified by biological experiments, they can be used for prognosis or diagnosis of the disease. For this purpose, researchers collect some "training" samples, for which their true class labels are known. Typically, the number of training samples is very small, such as tens or hundreds, but the number of candidate genes is large, such as thousands or tens of thousands depending on the organism being studied and various platforms. Identifying relevant genes for a disease from thousands of candidates (with only hundreds of samples) is still a tremendous challenge to statisticians. Ordinary classification algorithms may not scale well to high-dimensional data, which is known as the curse of dimensionality.

It has been reported that high-throughput biological data has grouping structures. This often has a biological basis; for example, a group of genes relate to the same molecular pathway, or are in close proximity in the genome sequence, or share a similar methylation profile. Biologically speaking, genes in eukaryotic genomes are not completely randomly distributed, and genes located within the same genomic neighborhoods tend to have similar expression patterns (Michalak, 2008). Genes that encode similar polypeptides or proteins are often found within a few thousand bases, and are called gene clusters. Usually, different genes in the same gene cluster tend to produce different kinds of proteins; however, they often share a common function. Gene clusters may range in size from a few genes to hundreds of genes in different organisms (Yi et al., 2007). For example, the homeobox family is a large group of similar genes that direct the formation of many body structures during early embryonic development. As a result, these genes can be viewed as being from the same "gene group", since they share a generalized function with similar expression levels. The grouping structures of genes may also be due to other reasons, such as sharing similar methylation profiles. For example, Jerónimo et al. (2004) identified several genes whose methylation levels progressively increase in prostate carcinogenesis.

The simplest way to detect group structure in gene expression data is to use direct visual inspection. Cho et al. (1998) proposed an approach to group genes with similar expression patterns within particular phases of the cell cycle. This method is appropriate when the patterns of interest are clear with prior knowledge (such as a periodic fluctuation in phase within the cell cycle); however, it is inappropriate for situations without prior knowledge. Many studies have been conducted to identify gene groups related to certain responses. For example, Spellman et al. (1998) applied a hierarchical average linkage clustering algorithm to find groups of co-regulated yeast genes. In practice, many statistical groups, as found by different clustering techniques such as in Spellman et al. (1998), may not match perfectly with functional groups. In addition, such simple grouping structures are probably too artificial to satisfactorily explain complicated gene expression processes. Without causing confusion, we use the phrases "clusters" and "gene groups" interchangeably in this thesis.

1.2 Existing Feature Selection Methods

Univariate Screening Methods

The simplest kind of feature selection methods may be univariate screening methods and model-based inference methods with independence assumptions for genes within classes, such as Student's t-test, Diagonal Linear Discriminant Analysis (DLDA) (Dudoit et al., 2002), and the Prediction Analysis of Microarrays method (PAM) (Tibshirani et al., 2002). Univariate methods ignore the correlations between genes, which are prevalent in gene expression data due to gene co-regulation; see Ma et al. (2007), Clarke et al. (2008) and Tolosi and Lengauer

(2011) for real examples. The consequence is that many differentiated but redundant genes are included in the univariate feature selection results; meanwhile, useful but weakly differentiated genes may be omitted.

Penalized and Bayesian Regression Methods without Considering the Grouping Structure in Features

Methods that work by fitting classification models, which attempt to capture the conditional distribution of the class labels (responses) given the features, can take correlations among features into account. However, when the number of observations is not much larger than the number of features, maximizing the likelihood of a classification model will over-fit the data, with noise rather than signal captured. Therefore, when the number of features is greater than the number of observations, we need to shrink the coefficients (if there are any) in classification models towards 0 to avoid overfitting. An alternative solution to this problem is a logistic regression model with penalized likelihood, known as Penalized Logistic Regression (PLR). Wahba (1990) employed this idea with L_2 norm (Euclidean norm) penalization terms for coefficients to enforce a sparse feature solution. **LASSO** (**Least Absolute Shrinkage and Selection Operator**) replaces the L_2 penalization norm with L_1 absolute shrinkage (the sum of the absolute values) and is a trending topic that has drawn attention from statistics communities in recent years. The major advantage of LASSO is that with strong penalizations (regularizations) some feature coefficients will converge to 0. Therefore, L_1 norm generally provides more sparse feature selection results than L_2 norm, which is very important for high-throughput gene expression analysis.

In the last 15 years, many researchers have worked on proposing priors with tails heavier than the Laplace distribution, called hyper-LASSO priors (Griffin and Brown, 2011) or global-local priors (Polson and Scott, 2010), for high-dimensional regression problems. A common goal of these priors is to shrink each coefficient differently so that large signals can be less penalized while small signals are shrunk towards 0. These priors are often given in the form of a scale mixture of normal or Laplace distributions. Such alternative penalties or priors to Laplace include, but are not limited to: SCAD (Fan and Li, 2001), horseshoe and normal/inverted-beta priors (Gelman, 2006; Carvalho et al., 2009, 2010; Polson and Scott, 2012c; van der Pas et al., 2014), NEG priors (Griffin and Brown, 2011), Normal/Gamma (Caron and Doucet, 2008; Griffin and Brown, 2010), adaptive LASSO (Zou, 2006), generalized double-Pareto (Armagan et al., 2013) and Dirichlet-Laplace and Dirichlet-Gaussian (Bhattacharya et al., 2012). Such methods are also called non-convex learning. For unified reviews of these penalties, one can refer to Kyung et al. (2010) or Polson and Scott (2010, 2012a,b); in addition, Bhattacharya et al. (2012) compared a broad class of shrinkage priors in terms of prior and posterior concentration, and showed that the commonly used Laplace prior is suboptimal in high-dimensional problems.

In the Bayesian community, the most popular choice (of priors) is to explore a posterior based on "spike-and-slab" priors — mixture distributions of a point mass at 0 and a continuous distribution; see Mitchell and Beauchamp (1988), Brown et al. (1998), Sha et al. (2004), Ishwaran and Rao (2005a), Ishwaran and Rao (2005c), Hans et al. (2007), Bottolo and Richardson (2010), Fan and Zhang (2010), Guan and Stephens (2011), Rockova and George (2013), among many others. Such priors may best express our prior belief. However, the inference results using this approach are highly sensitive to the choice of width of the continuous part. If the width is large enough, any features, no matter how large the signals they carry, will be classified as "irrelevant", with data information totally ignored. The sensitivity has been discussed in the context of regression problems theoretically in Denison et al. (2002, p.23) and empirically in Lamnisos et al. (2012); indeed, the problem has long been known as the Jeffreys-Lindley's paradox regarding a Bayesian test of null hypothesis about a point (Lindley, 1957; Shafer, 1982; Robert, 2013).

Penalized Regression Methods Using the Grouping Structure in Features

A recent development of LASSO is Group LASSO (Meier et al., 2008), which considers penalties that enforce similarity for the coefficients of features within groups. Group LASSO achieves better predictive performance than the original LASSO method, because it reduces the dimensionality of features and also consolidates the predictive power of all features within groups. However, since the coefficients of the features within a group are forced to be similar, it will become harder to select features within groups. Another problem of this approach is the so-called correlation bias problem (Tolosi and Lengauer, 2011)—the magnitudes of the coefficients of a group will decrease as the group size increases—which results in ambiguity in comparing features across groups when the group sizes vary greatly. Considering that the genes in a group may not all be associated with the response, Ma et al. (2007) propose Supervised Group LASSO (SGLASSO), which applies LASSO in two steps: 1) applying LASSO to the features of each group separately, and 2) applying LASSO to the features selected from each group in step 1. A drawback of SGLASSO is that it makes selection separately in each group. Therefore, it cannot consider the joint effects of features from different groups. It is possible that a feature is not useful marginally but becomes very predictive if combined with another feature. Another problem of Group LASSO and SGLASSO is that both rely on a pre-specification of the grouping structure, which is often found by a clustering algorithm.

1.3 Contributions of this Thesis

An important property of heavy-tailed priors when used in regression or classification models for features with grouping structures is that they can make selections within groups: either splitting important features into different modes or suppressing less important features. Section 4.1 provides a simple demonstration of this property. Li and Yao (2014) provided a geometric explanation with figures by looking at the "path" of constrained maximum a posterior probabilities (MAPs). This property confers both advantages and disadvantages. The advantage is that the single mode of the posterior provides a feature subset that is much more parsimonious than that given by LASSO. The disadvantage is that the posterior distribution is highly multi-modal. Therefore, optimization algorithms, such as the expectation maximization (EM) algorithm and conjugate gradient method, have great difficulty in reaching a good mode; more discussions of the local mode problem are given by Wang et al. (2014). Therefore, we think that the fully Bayesian approach—using Markov chain Monte Carlo (MCMC) methods to sample from the posterior—is a valuable alternative for the non-convex learning problem, because a well-designed MCMC algorithm has a better chance of traveling across many modes.

In this thesis, we introduce a feature selection method by using a sophisticated MCMC method to explore the posterior of a classification model (called T-probit herein) based on a class of heavy-tailed t priors with moderate degrees of freedom (such as 1, corresponding to the Cauchy distribution) and small scales. Our MCMC algorithm applies the Hamiltonian Monte Carlo method (Neal, 2011) to sample regression coefficients in a restricted Gibbs sampling framework in which the t prior is expressed as a scale-mixture normal. Our examples will show that this algorithm can efficiently travel across a large number of modes. After we obtain the MCMC samples of the coefficients, we further divide them into sub-pools according to the posterior modes. Each sub-pool or posterior mode represents a selected feature subset. Using this method we report a list of selected feature subsets. Furthermore, we use cross-validation to measure the predictive goodness of each feature subset, which provides additional information for selecting from the list of feature subsets.

The main features that distinguish the above feature selection method from other methods in the literature are summarized as follows:

1. Our method makes selections within groups automatically without a pre-specification of the group identities. In addition, the selections within groups are in conjunction with other features in other groups.
2. A single feature subset (posterior mode) is much more parsimonious—only one or very few features are selected from each group, and most noise features are discarded. Such succinct feature subsets are much easier to interpret or understand based on existing biological knowledge and further experimental investigations.
3. The coefficients of each feature subset will not be subject to correlation bias (Tolosi and Lengauer, 2011) because only one or very few features are selected from each group.
4. The multiple feature subsets found by MCMC simulations provide multiple explanations of the associations for scientists to further explore.

Liu (2004) proposed the robit model (with EM algorithm) to find a sparse feature selection results; however, the algorithm seemingly has not been tested in high-dimensional problems. Li and Yao (2014) discussed the choice of degrees of freedom and scale of t priors for logistic regression models, but did not develop a feature selection method by dividing the MCMC samples according to the posterior modes. This thesis uses a t prior in T-probit regression, and empirically demonstrates its within-group selection property.

We test fully Bayesian T-probit on simulated datasets with independent and correlated groups of features, and also compare it with other methods using three real microarray datasets. Our empirical results show that the predictive performances of some feature subsets selected by our method are comparable with feature subsets of much larger size selected by plain LASSO, Group LASSO, Supervised Group LASSO, random forest, logistic regression with non-convex penalization and t-test. In addition, we demonstrate that the succinct feature subsets selected by fully Bayesian T-probit have significantly better predictive power than the feature subsets of the same size taken from the top features selected by the aforementioned methods.

The remainder of this thesis is structured as follows. In Chapter 2, we review a wide range of current methods for feature selection and classification problems. These methods range from univariate filters to penalized regression models and nonparametric methods. In Chapter 3, we describe our fully Bayesian T-probit regression with heavy-tailed t priors in technical detail. In Chapter 4, we present the results of comparing fully Bayesian T-probit with the most commonly used of the feature selection methods using simulated datasets with different grouping structures. We also employ simulation studies to investigate parameter tuning issues and assess the computational efficiency by comparing our algorithm to JAGS (Just Another Gibbs Sampler). In Chapter 5, we compare our method with other methods using three real gene expression datasets related to cancer. The thesis concludes in Chapter 6 with a discussion of future work.

Chapter 2
Literature Review

In this chapter we review some well-known methods for high-dimensional feature selection and prediction. We use $X = (x_{ij}), i = 1, 2, ..., n, j = 1, 2, ..., p$ to denote all explanatory variables, where x_{ij} is the value of the jth feature from the ith observation, n is the number of observations, and p the number of features. $Y = (Y_i, i = 1, 2, ..., n)$ are the values of the response variable, which can only take discrete values from $1, ..., C$. $\{x_{i.} = (x_{i1}, x_{i2}, ..., x_{ip}), i = 1, ...n\}$ is the vector containing all values of features from the ith observation, and $\{x_{.j} = (x_{1j}, x_{2j}, ..., x_{nj}), j = 1, ...p\}$ is the vector containing all values of feature j across all observations. The feature matrix of all observations with response value $y = c$ is given as $x_{y=c} = (x_{ij}, i \in G_c, j = 1, 2, ..., p)$, where G_c is the set of all observations from class c, and $|G_c| = n_c$ is the number of observations in class c. The goal is to select a feature subset containing significant features, and construct a prediction model $f : x_{*.} \to y_*$ for an unknown response y_* of a future case based on covariate values $x_{*.} = (x_{*1}, x_{*2}, ..., x_{*p})$.

2.1 Review of Feature Selection Methods

2.1.1 Univariate Screening Methods

Univariate filtering is a feature selection method. The idea of univariate selection is to construct a ranking index for each feature and then select features according to their ranks. To complete this task we need two quantities: a ranking index and a selection criterion. Available ranking indexes include t-test, F-test and Fisher's Score. The index of the t-test of the jth feature (for binary classes) is given as

$$\frac{\bar{x}_{\cdot j}^{(1)} - \bar{x}_{\cdot j}^{(2)}}{S_j}, \tag{2.1}$$

where

$$S_j = \sqrt{\frac{(s_j^{(1)})^2}{n_1} + \frac{(s_j^{(2)})^2}{n_2}}, \tag{2.2}$$

and $\bar{x}_{\cdot j}^{(c)}, c = 1, 2$ are the mean values of the jth feature across all observations from class c: $\bar{x}_{\cdot j}^{(c)} = \frac{1}{n_c} \sum\limits_{i \in G_c} x_{ij}, c = 1, 2$. G_c is the index of all observations from class c, and $|G_c| = n_c$ is the number of observations in class c. $(s_j^{(c)})^2, c = 1, 2$ are the unbiased estimators of the variances of the corresponding samples:

$$(s_j^{(c)})^2 = \frac{1}{n_c - 1} \sum_{i \in G_c} (x_{ij} - \bar{x}_{\cdot j}^{(c)})^2. \tag{2.3}$$

The Fisher's score is just the square of the t-test score. The F-test is defined as

$$F = \frac{\text{between-class variability}}{\text{within-class variability}}. \tag{2.4}$$

The between-class variability of the jth feature is

$$\sum_{c=1}^{2} n_c (\bar{x}_{\cdot j}^{(c)} - \bar{x})^2 / (C - 1), \tag{2.5}$$

where $\bar{x}_{\cdot j}^{(c)}, c = 1, 2$ are the mean values of the jth feature across all observations from class c, and $\bar{x} = \frac{1}{C} \sum\limits_{c=1}^{C} \bar{x}_{\cdot j}^{(c)}$ is the mean of all $\bar{x}_{\cdot j}^{(c)}$. The within-class variability is

$$\sum_{c=1}^{C} \sum_{i \in G_c} (x_{ij}^{(c)} - \bar{x}_{\cdot j}^{(c)})^2 / (n - C), \tag{2.6}$$

where C is the total number of classes.

Most univariate screening methods focus on univariate relevance ranking of features. For example, the t-test searches for features with significantly different class mean values with respect to the sample variance, and the F-test searches for features with significant between-class variability with respect to within-class variability. After the test statistics for all features are evaluated and ranked, the p-value will be calculated as a criterion to screen features. One disadvantage of univariate screening is the multiple testing problem. That is, with a large number of hypothesis tests of features, the probability of wrongly rejecting a null hypothesis will increase and a substantial number of false significance cases (Type I errors) might occur. Alternatively, a corrected p-value with FDR (False Discovery Rate) correction, known as the q-value, is also available as a feature selection criterion. Another disadvantage of univariate selection methods is that they ignore correlation information, so some marginally "irrelevant" but "useful" features may get a very low score. For example, consider the data points in Figure 2.1. The univariate test is inappropriate here since the T-score will be very small for each axis even though the data points from the two genes are "useful" to construct a linear classifier. This is actually the problem encountered by all univariate screen methods: that some "insignificant" features may still be "useful" when considered in conjunction with others.

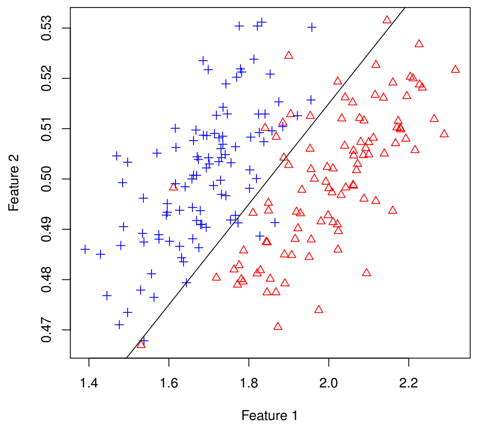


Figure 2.1: A 2-dimensional dataset is illustrated here. Blue points are datapoints from cluster 1 and red points are from cluster 2. The X-axis represents the feature 1 values and the Y-axis represents the feature 2 values.

2.1.2 Penalized and Bayesian Regression Methods without Considering the Grouping Structure in Features

LASSO

Ridge Regression is a commonly used method of regularization on ill-posed problems (because of non-uniqueness of solutions of regression coefficients). The penalized likelihood of ridge regression is

$$LL_{\text{Ridge}}(\beta|X,Y) = ||Y - X\beta||^2 + \lambda||\beta||^2, \tag{2.7}$$

which is equivalent to minimizing

$$||Y - X\beta||^2, \text{subject to } ||\beta||^2 \leq t. \tag{2.8}$$

The solution (of β) to this minimization problem is $(X^TX + \lambda I_p)^{-1}X^TY$. By introducing the regularization parameter λ the problem becomes nonsingular even when X^TX is singular (which is always the case when $n \ll p$), and it is the motivation of Hoerl and Kennard (1970) which leads to regularized linear discriminant analysis. The estimation of λ can be achieved with cross-validation or analytical risk estimates (Donoho et al., 1995). In practice, cross-validation is favored if the data size is large.

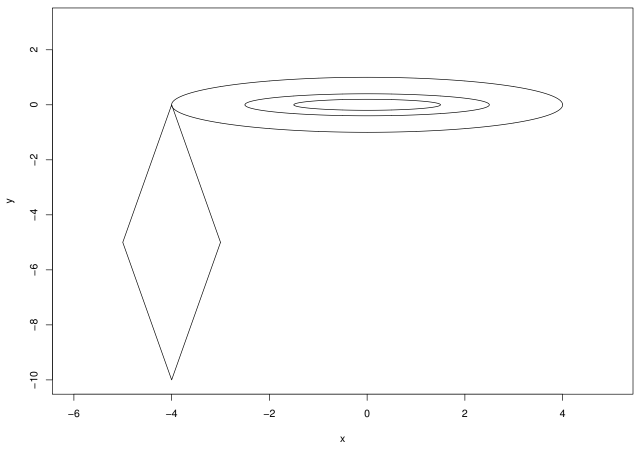
LASSO uses the L_1 penalization norm (absolute shrinkage), which gives the likelihood

$$LL_{\text{LASSO}}(\beta|X,Y) = (Y - X\beta)^T(Y - X\beta) + \lambda\sum_{j=1}^{p}|\beta_j|, \tag{2.9}$$

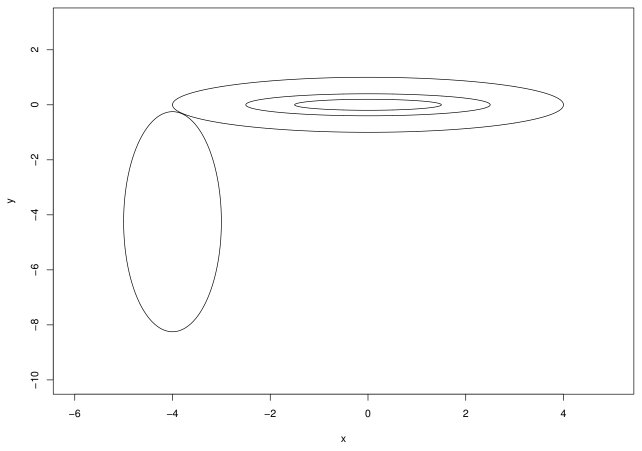
which is equivalent to minimizing

$$||Y - X\beta||^2, \text{subject to } |\beta| \leq t. \tag{2.10}$$

One major advantage of LASSO is that with large values of λ, some coefficients β will



(a) Elliptical Contours with a L_1 constraint.



(b) Elliptical Contours with a L_2 constraint.

Figure 2.2: Elliptical Contours with L_1 and L_2 constraints.

be shrunken exactly to 0. Geometrically, this is because with a L_1 norm the elliptical contours centered at the regularized least square (RLS) estimator will touch the polyhedron (constrained area defined by t) at the LASSO solution, which may happen at a corner, with corresponding zero coefficients of some covariates (Figure 2.2). With the L_2 norm the elliptical contours will touch the polyhedron at a point with nonzero entries of covariates. As a result the L_1 norm will generally provide more sparse feature selection results than the L_2 norm, which is necessary for microarray data analysis since most features are considered “useless”.

The LASSO solution does not have an explicit form and the estimation of β involves quadratic programming. We refer to Tibshirani (1996) for a review of LASSO.

Penalized Logistic Regression with the L_2 norm (Wahba, 1990) is a widely used calcification strategy in medical decisions. The penalized log-likelihood is

$$LL_{\text{PLR}}(\beta|x_{1.}, ..., x_{n.}, y_1, ..., y_n) = \sum_{i=1}^{n} \log(1 + \exp(-y_i x_{i.}^T \beta)) + \lambda||\beta||^2. \tag{2.11}$$

The L_1 norm can also be replaced by the L_2 norm. Then the task is to optimize the log-likelihood

$$LL_{\text{G-LASSO}}(\beta|x_{1.},...,x_{n.},y_1,...,y_n) = \sum_{i=1}^{n} \log(1+\exp(-y_i x_{i.}^T \beta)) + \lambda|\beta|, \tag{2.12}$$

where $x_{i.}$ is a column vector containing feature values of the ith observation, and y_i is the ith response coded as -1 or +1.

An interesting derivation of the LASSO method is the LAR (Least Angle Regression) method, developed by Efron et al. (2004). LAR is an efficient path-computing method and an interesting "bisect" regression strategy, which is an intermediate between classical forward selection and stage-wise regression. It gives almost the same traces as LASSO in many cases.

The Ridge Regression model and LASSO can be cast into the Bayesian framework with negative log-likelihood

$$LL(\beta|X,Y) = (Y - X\beta)^T(Y - X\beta). \tag{2.13}$$

Thus, Ridge Regression adopts an $N(0,1/\lambda)$ Gaussian prior for β while LASSO adopts a double-exponential prior equation $\pi(\beta)$ in (2.14). The prior distribution of λ is usually considered as a non-informative prior. The probability density function of these two distributions are shown in Figure 2.3.

$$\pi(\beta) \propto \exp(-\lambda|\beta|). \tag{2.14}$$

Bayesian LASSO does not enforce β values to be exact zero (as LASSO does) since the posterior distribution of β is continuous on the domain of real values. The double-exponential prior (LASSO) puts more mass on tails (compared with Ridge Regression) to obtain sparse feature selection results. Some Bayesian methods then are motivated to adopt more sparse point-mass priors (Lee et al., 2003).

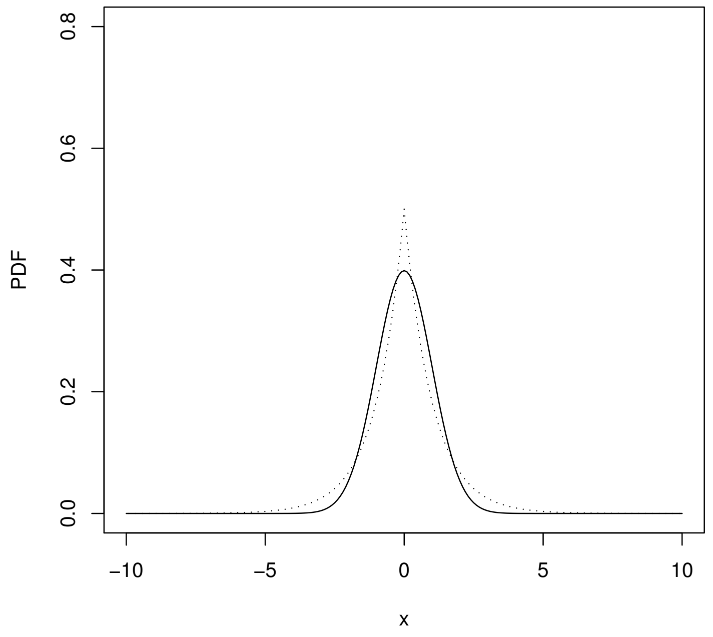


Figure 2.3: Gaussian prior and double-exponential prior

$$\pi(\beta_j) \propto \begin{cases} \text{point mass at 0,} & \text{with probability} \quad p_j \\ N(0, \sigma_j^2), & \text{with probability } 1 - p_j \end{cases} \tag{2.15}$$

This can be interpreted as inserting a covariate indicating vector $\gamma = (\gamma_1, ..., \gamma_p)$, while $\gamma_j, j = 1, ...p$ indicates the presence of feature j in the model:

$$\pi(\beta_j|\gamma_j) \sim \begin{cases} \text{point mass at 0,} & \text{if} \quad \gamma_j = 0 \\ N(0, \sigma_j^2), & \text{if} \quad \gamma_j = 1 \end{cases} \tag{2.16}$$

γ_j is from a Bernoulli distribution with failure probability p_j, which actually corresponds to the prior probability of different models in the model space. This structure emphasizes the sparsity and heavy-tail property of the parameters; thus, it can operate feature selection automatically. However, the underlying assumption of the independence across γ_j is not always appropriate, which means deeper structure for γ may be required (like a transition probability based Markov Model for p_j), which will increase model complexity quickly and cause onerous Markov chain mixing.

One major advantage of the Bayesian formulation is that it enables us to assess the posterior probability of all models, even though the analytical forms of these probabilities are usually complex and difficult to handle (Bayesian Model Averaging (BMA) (Hoeting et al., 1999)). The sampling process, however, may be tricky: the results may fall into several "mode traps", each corresponding to a different model on model space.

Bayesglm: Penalized Logistic Regression with t Priors

The common problems for maximum likelihood estimation of logistic regression are the non-identifiability issue and separation. For fixed data, non-identifiability means that if two parameter estimates give rise to the same log-likelihood, then it will be impossible to distinguish between the two candidate vectors and identify the true parameter values based on the data alone. Separation means a regression model using two or more covariates can separate the classes (of training cases) without misclassification. From an estimation perspective, separation results in infinite coefficients and standard errors as noted in Zorn (2005). In high-dimensional gene expression data, separation always happens since the dimension p is substantially larger than n. However, non-identifiability is a particularly difficult problem because the dimension is high and multiple candidate parameter values will yield same likelihood values.

An approach toward stable logistic regression coefficients is through Bayesian inference. Firth (1993) first proposed the Jeffreys prior distribution to reduce the bias in maximum likelihood estimation. Gelman et al. (2008) generalized the scaled prior distribution to the Student's t-distribution, with a set up for reliable computation of logistic regression parameters. The main procedure incorporates an EM algorithm and the usual iteratively weighted least squares. The algorithm is included in the R package `Bayesglm`.

Consider a logistic regression model with data x for binary classification problem $y \in (0, 1)$. The coefficients $\beta = (\beta_1, ..., \beta_P)'$ have independent Student's t priors with scale parameters $\alpha = (\alpha_1, ..., \alpha_P)$ and $\nu = (\nu_1, ..., \nu_P)$.

$$LL(\beta) = \sum_{i=1}^{n}[\log(\Pr(Y = y_i))] = \sum_{i=1}^{n}[x_{i.}\beta_{y_i} - \log(\exp(x_{i.}\beta))], \tag{2.17}$$

$$\beta_j \sim N(0, \sigma_j^2), \sigma_j^2 \sim IG(\alpha_j, \nu_j). \tag{2.18}$$

where $IG(\alpha_j, \nu_j)$ is the inverse gamma distributions with shape parameter α_j and scale parameter ν_j.

`Bayesglm` implements the algorithm with an initial estimate of β (it could be a guess such as $\beta_j = 0$) and sets each σ_j to the value of ν. The choice of initial point does not change the nature of this algorithm. By default, the values of α_j is set to 1 and ν_j is set to 2.5 for logit model. It proceeds as follows:

1. Suppose $\hat{\beta}$ is the current estimate of parameter β, we construct the auxiliary variable z_i and the auxiliary variances $(\sigma_i^z)^2$

$$z_i = x_{i.}\hat{\beta} + \frac{(1 + e^{x_{i.}\hat{\beta}})^2}{e^{x_{i.}\hat{\beta}}}(y_i - \frac{e^{x_{i.}\hat{\beta}}}{1 + e^{x_{i.}\hat{\beta}}}), \tag{2.19}$$

$$(\sigma_i^Z)^2 = \frac{1}{n}\frac{(1 + e^{x_{i.}\hat{\beta}})^2}{e^{x_{i.}\hat{\beta}}}. \tag{2.20}$$

2. Use the EM algorithm to update the parameters β_j and σ_j. The E step (in the EM algorithm) starts with weighted least squares regression (assigning each observation a different amount of influence over the estimates of β) based on data augmentation techniques. More specifically, we can add pseudo-data points to obtain observations z_*, X_* and weight vector w_*, where

$$z_* = \begin{pmatrix} z \\ 0 \end{pmatrix}, X_* = \begin{pmatrix} X \\ I_p \end{pmatrix}' X, w_* = (\sigma^z, \sigma)^{-2}. \tag{2.21}$$

Weighted linear regression on z_*, X_* and w_* can be performed to obtain an estimate

$\hat{\beta}$. With the augmented X_*, the logistic regression is well defined and the resulting $\hat{\beta}$ has finite variance, even if the original data have non-identifiability and separation problems.

3. The second step in the E step is to approximate the expectation with respect to posterior distribution of β:

$$E(\log p(\beta, \sigma|y)) \approx -\frac{1}{2}\sum_{j=1}^{p}(\frac{1}{(\hat{\sigma}_j)^2}(\hat{\beta}_j)^2 + \log(\hat{\sigma_j^2})) - p(\hat{\sigma}_j|\alpha_j, \nu_j). \tag{2.22}$$

4. The M step is to maximize the (approximate) expected value of the log-posterior function in equation (2.36) to get a new estimate $\tilde{\sigma}_j$,

$$\tilde{\sigma}_j^2 = \frac{(\hat{\beta}_j)^2 + (\hat{\sigma}_j)^2 + \alpha\nu}{1+\alpha}, \tag{2.23}$$

and repeat steps 1-4 above with new estimates of β and σ.

In this approach, the iteratively weighted least squares algorithm (Holland and Welsch, 1977) is used to estimate β with the approximate EM algorithm (Dempster et al., 1977). The method of iteratively weighted least squares is widely applied to solve logistic regression problems. The whole approach above is included in the R package `arm`. In this thesis, we use this package to fit Bayesian logistic regression with Student's t priors. The final converged values β_j are then recorded as the significance score to select features.

In addition to the LASSO methods and the penalized logistic regression methods, there are also techniques for exploring a posterior based on "spike and slab" priors (Mitchell and Beauchamp, 1988; George and McCulloch, 1993; Brown et al., 1998; Sha et al., 2004; Ishwaran and Rao, 2005b; Hans et al., 2007; Bottolo and Richardson, 2010; Li and Zhang, 2010; Guan and Stephens, 2011; Ročková and George, 2014). Spike and slab priors utilize mixture distributions of a point mass at 0 and a continuous distribution. Theoretically, this model design naturally encourages sparse solutions in high-throughput data (Ishwaran and Rao, 2005b). Gibbs sampling is usually considered to identify posterior modes over the parameter spaces with a specified hierarchy of spike and slab priors. However, with increasing numbers of features p, the choice of hierarchy (priors) can be tricky. More specifically, the choices of the width of the continuous distribution often have a great impact on the inference results. With a choice of large width the data information will be ignored and many useful features may be classified as irrelevant. This problem is well addressed theoretically in Denison (2002) and Lamnisos et al. (2012). Theoretically, this problem is also known as the Jeffreys-Lindley's paradox regarding a Bayesian test of a null hypothesis about a point (Lindley, 1957; Shafer, 1982). On the other hand, most of the Gibbs samplers with spike and slab priors are processed using posterior mean values to summarize posterior information (Barbieri and Berger, 2004). The reason is that with a large value of p (in the example of microarrays p can be tens of thousands), it is common to find that none of the posterior modes have high frequency.

2.1.3 Penalized Regression Methods Using the Grouping Structure in Features

Group LASSO

A recent development of LASSO is Group LASSO, which considers group structure among features. Suppose that p covariates can be divided into L groups, with p_l features in group l. Yuan and Lin (2006) proposed Group LASSO to solve the convex optimization problem

$$\min_{\beta \in \Re^p} (||y - x\beta||_2^2 + \lambda \sum_{l=1}^{L} \sqrt{p_l} ||\beta_l||_2), \tag{2.24}$$

where x_l represents the covariates belonging to the lth group with corresponding coefficient β_l, $\sqrt{p_l}$ accounts for the varying group sizes, and $||.||_2$ is the Euclidean norm. The parameter λ controls the amount of penalization since different values of λ will result in different numbers of groups being selected. With $L = 1$ the entire group structure will degenerate to conventional LASSO. This penalty can be viewed as an intermediate between the L_1 and L_2 penalty and can be applied to models other than linear regression such as Cox regression (Tibshirani et al., 1997), logistic regression (Shevade and Keerthi, 2003; Genkin et al., 2007) and multinomial logistic regression (Krishnapuram et al., 2005) by replacing $||y - 1 - \sum_{l=1}^{L} x_l \beta_l||$ with the corresponding negative log-likelihood function. The logistic Group LASSO estimator β_λ is then given by the minimizer of the convex function

$$-l(\beta) + \lambda \sum_{g=1}^{G} \sqrt{p_l} ||\beta_g||_2, \tag{2.25}$$

where $l(.)$ is the log-likelihood function, i.e.,

$$l(\beta) = \sum_{i=1}^{n} [x_i \beta y_i - \log(1 + \exp{(x_i \beta)})]. \tag{2.26}$$

The biostatistics community has begun to apply Group LASSO to microarray studies since it recognizes the importance of group structures (Meier et al., 2008). Analysis of gene expression data is challenging since there may exist natural groups (i.e., gene clusters) that are composed of co-regulated genes with coordinated functions. Meier et al. (2008) applied the Group LASSO penalty with logistic classification to microarray data analysis. Group LASSO is able to select important groups of genes rather than selecting genes individually. However, it is not capable of selecting important individuals within each group, and so it does not yield sparsity. The significances of covariates are generally averaged over each group, so if a group of coefficients is non-zero then they will all be included.

In this thesis, we use the R-package `glmnet` to obtain Group LASSO solutions to microarray classification problems. This package is developed by Lukas Meier and uses the coordinate descent algorithm to solve the resulting convex problem. The least angle regression algorithm (LAR) (Efron et al., 2004) is the default method for linear models with the LASSO penalty. However, with the Group LASSO penalty, some approximate path-following algorithms (Zhao et al., 2009; Park et al., 2007) are not applicable (Tibshirani et al., 2005). Kim et al. (2006) first proposed a gradient descent algorithm to solve the corresponding constrained problem for logistic regression with the Group LASSO penalty. Meier et al. (2008) developed the idea from Genkin et al. (2007) and presented an asymptotic consistency theory for the Group LASSO to deal with high-dimensional problems (where p is fairly large). The key component of this algorithm is a block coordinate gradient descent method from Tseng and Yun (2009), which combines a quadratic approximation of the log-likelihood with an additional linear search. The whole procedure is computationally efficient for large-scale problems (Genkin et al., 2007).

Group LASSO requires a pre-specification of the grouping of features. In most scenarios, we do not expect the experimenters to provide such prior information with the original molecular dataset. Therefore, we need to cluster features into groups. One of the most popular clustering methods, k-means clustering, is easy to implement and also available as an R package. The basic idea of k-means clustering is to partition p features into k groups in which each feature belongs to the group with the nearest mean, serving as a prototype of the group. More specifically, given a set of features $(x_{.1}, x_{.2}, ..., x_{.p})$, where each feature is an n-dimensional vector, k-means clustering aims to partition the p observations into k sets $S = \{s_1, s_2, ..., s_k\}$ so as to minimize the within-group sum of squares

$$\text{argmin} \sum_{i=1}^{k} \sum_{x_{.j} \in s_i} ||x_{.j} - \vec{\mu_i}||^2, \tag{2.27}$$

where $\vec{\mu_i}$ is the mean of the features in s_i. Several algorithms are available for k-means grouping, including the algorithm of Hartigan and Wong (Hartigan and Wong, 1979), the algorithm of MacQueen (MacQueen et al., 1967), and the algorithm of Forgy (Forgy, 1965). By default we apply the algorithm of Hartigan and Wong to obtain k-means grouping results.

An alternative method of clustering analysis is Hierarchical Clustering (Rokach and Maimon, 2005; Johnson, 1967), which seeks to build a hierarchy of clusters. The basic idea is to combine clusters (or divide a cluster) with respect to a measure of dissimilarity between groups. Generally, agglomerative hierarchical clustering (bottom-up method) lets each observation start in its own cluster, and pairs of clusters are merged as one moves up the hierarchy. Divisive hierarchical clustering (top-down method) lets all observations start in one cluster, and splitting is performed recursively as one moves down the hierarchy. Both methods require a linkage method to be selected, which determines the dissimilarity between two groups based on the similarity of the members of those groups. Some commonly used linkage criteria to test the dissimilarity between two groups A and B include maximum linkage

$$\max\{d(a, b) : a \in A, b \in B\} \tag{2.28}$$

and average linkage (also known as Unweighted Pair Group Method with Arithmetic Mean, UPGMA)

$$\frac{1}{|A||B|}\sum_{a\in A}\sum_{b\in B} d(a,b), \tag{2.29}$$

where $d(a,b)$ is the chosen metric, which can be Euclidean distance ($|a-b| = \sqrt{\sum\limits_i (a_i - b_i)^2}$), squared Euclidean distance ($|a-b| = \sum\limits_i (a_i - b_i)^2$), Manhattan distance ($|a-b| = \sum\limits_i |a_i - b_i|$), etc.

In this thesis, we use average linkage (UPGMA) and the squared Euclidean distance metric, even though technically squared Euclidean distance is not really a "metric" as it is defined, and use them as the default configuration for clustering analysis. UPGMA is a simple bottom-up hierarchical clustering method to construct a dendrogram with a pairwise dissimilarity matrix. It is widely used in ecology to classify sampling units and create phenetic trees (phenograms) in bioinformatics. Fionn Murtagh proposed a time-optimal $O(p^2)$ algorithm to construct a standard UPGMA tree (Murtagh, 1984). We use the R package `hclust` to perform hierarchical clustering analysis of the gene expression data.

Both k-means clustering and hierarchical clustering require the number of groups. Remember that k-means grouping aims to find k prototype genes and classify each gene into the neighborhood of these prototypes; thus, the problem of how many prototypes are needed remains. For hierarchical clustering, the dendrogram needs a cut point (how many groups are needed) to select a specific structure. Tibshirani et al. (2001) proposed the gap statistics for estimating the number of groups in a set of features. Suppose that we have grouped the features into k groups $S = \{s_1, s_2, ..., s_k\}$ with $n_k = |s_k|$. Let

$$D_k = \sum_{i,i`\in C_k} d_{ii'} \tag{2.30}$$

be the sum of the pairwise distances for all features in group k, and set

$$W_k = \sum_{K=1}^{K} \frac{1}{2n_k} D_k. \tag{2.31}$$

Then W_k is the pooled within-group sum of squares around the group means if we choose distance d as the squared Euclidean distance. The gap statistic is then defined as

$$\text{Gap}_K = E_n \text{log}(W_K) - \log(W_K), \tag{2.32}$$

where E_n denotes expectation under a sample of size n from the reference distribution. Our estimate $\hat{k}$ will be the value maximizing Gap_K when the sampling distribution is taken into account.

An alternative way to choose the number of groups involves the silhouette value (Struyf et al., 1997). It evaluates how well each feature lies within its group. More specifically, assume the jth feature $x_{\cdot j}$ is clustered into group A. Then

$$a(j) = \frac{1}{|A| - 1} \sum_{i \in A, i \neq j} \triangle(i, j) \tag{2.33}$$

measures the average distance of the jth feature to all other features in group A. For any group $C \neq A$

$$d(j, C) = \frac{1}{|C|} \sum_{i \in C} \triangle(i, j) \tag{2.34}$$

measures the general distance from the jth feature to other groups. The minimum value

$$b(j) = \min_{C \neq A} d(j, C) \tag{2.35}$$

quantifies the distance of the jth feature to the nearest neighbor group. The silhouette value

$s(j)$ is then defined as

$$s(j) = \frac{b(j) - a(j)}{\max\{a(j), b(j)\}}, \tag{2.36}$$

and it indicates how well $x_{.j}$ fits into group A. The average silhouette value is the mean value of all $s(j)$ across all features, and measures the general quality of clustering. In this thesis, we employ both Gap statistics and silhouette values to choose reasonable cutting points, though when p is large the results from both are often similar.

Supervised Group LASSO

Supervised Group LASSO is a recent development of Group LASSO (Ma et al., 2007). Compared with individual gene selection made by LASSO and the group averaging solution of Group LASSO, Supervised Group LASSO employs two levels of shrinkage across groups and within groups. As a result, the within-group gene selections will lead to a more sparse feature solution.

Supervised Group LASSO consists of 2 steps.

1. For a single group $j = 1, ..., m$, compute $\hat{\beta}_j$, the group-wise LASSO estimator of the coefficients in group j, according to

$$\hat{\beta}_j = \text{argmax} \;\; R_n(\beta_j) \tag{2.37}$$

subject to $|\beta_{j,1}| + ... + |\beta_{j,p_j}| \leq u_j$, where u_j is a group-wise tuning parameter and

$$R_n(\beta_j) = \sum_{i=1}^{n} y_i \log\left(\frac{\exp(x_{ij}\beta_j)}{1 + \exp(x_{ij}\beta_j)}\right) + (1 - y_i)\log\left(\frac{1}{1 + \exp(x_{ij}\beta_j)}\right) \tag{2.38}$$

for binary ($y_i = 0$ or 1) classification. x_{ij} here is the covariate vector containing all features from the jth group (from the ith observation).

2. Feature coefficients found in Step 1 for each group will have mostly zero entries. Then

we fit Group LASSO again with

$$\hat{\beta} = \text{argmax} \;\; R_n(\beta) \tag{2.39}$$

subject to $|\beta_1| + ... + |\beta_m| \leq u$, where

$$R_n(\beta) = \sum_{i=1}^{n} y_i \log(\frac{\exp(\beta x_{i.})}{1 + \exp(\beta x_{i.})}) + (1 - y_i) \log(\frac{1}{1 + \exp(\beta x_{i.})}), \tag{2.40}$$

and $x_{i.}$ here is the covariate vector (from the ith observation) only containing features with non-zero coefficients found in Step 1. u is the group regularization parameter. All values of u and u_j are chosen via cross-validation. By default, the values are determined with 3-fold cross validation in the R package `grpLASSO` (Meier, 2009).

2.1.4 Nonparametric Models

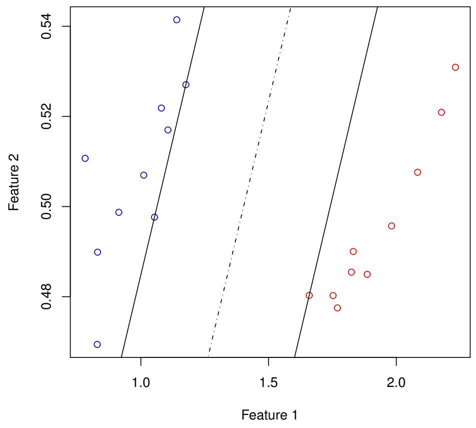


Figure 2.4: SVM classifier on binary cases. Red data points: y=1, blue data points: y=0. X-axis: feature 1 values of data points. Y-axis: feature 2 values of data points.

Recently, the notion of margin for pattern recognition has been popularized by the success of **support vector machines** (SVM). The original motivation of SVM is very intuitive for binary cases as in Figure 2.4. The different colors denote binary responses. With only two features, there are numerous straight lines that can perfectly separate the two classes. The linear SVM method chooses the linear classifier (the dashed line in the picture) that produces the maximum "non-occupied" area between two classes. More specifically, the goal is to find a hyper-plane classifier such that the distances to the nearest points (for each class) are maximized. In Figure 2.4 it means the distance between the two solid parallel lines is maximized among all possible choices.

Although the primary idea is simple (Boser et al., 1992), SVM turns out to have vast potentials for further adjustment and improvement. It can be blended with regularized optimization as we introduced in section 2.1.2. For example, regularized support vector machines (Vapnik, 1995) can be equivalently expressed as to optimize the corresponding log-likelihood

$$LL_{\text{G-SVM}}(\beta|x_{1.}, ..., x_{n.}, y_1, ..., y_n) = \sum_{i=1}^{n}(1 - y_i x_{i.}^T \beta) + \lambda||\beta||^2, \tag{2.41}$$

where $x_{i.}$ is a column vector containing feature values of the ith observation, and y_i is the ith response coded as -1 or +1.

On the other hand, the linear classifier can be modified to non-linear using kernel methods. The multi-class problem can be decomposed into multiple binary problems. Cortes and Vapnik (1995) invented a modified version called "soft margin" to deal with mislabeled cases, by adding a slack variable to measure the degrees of misclassification. Then the optimization problem gets an additional error penalty term. For this improvement and its practical usage Cortes and Vapnik received the 2008 ACM Paris Kanellakis Award. However, the SVM also has limitations: the biggest question may be the choice of the kernel and its parameters; other open questions involve the optimality of multi-class cases and quadratic programming efficiency. Like any non-parametric method, SVM can be used on non-regular data, i.e., the data points have an unknown distribution; however, it cannot give analytical feature selection results. Even though users can have a graphical explanation of the classification, the underlying relationsip between features (genes) and responses can be unclear. For further discussion we refer to Weston et al. (2001) for feature selection techniques and Ben-Hur et al. (2002) for clustering techniques via SVM.

Other non-parametric methods, including NN (Nearest-Neighbor classifier) and Classification And Regression Trees (CART) (Steinberg and Colla, 2009), have been used for classification tasks. The K-nearest-neighbor rule works as follows: first, for each test case, K training cases are identified due to a pre-specification of the distance function (i.e., Euclidean distance). Then the prediction is decided with the most frequently appearing label among the K-neighbors, and the value of K can be chosen by cross-validation. A well-known generation of NN is to assign a label to each test case according to the average weight of each point in the training pool, while the weight is defined by a kernel function

$$\text{Prediction}(y^*|x^*) = \text{sign}(\sum_{i=1}^{n} y_i K(x_{i.}, x^*)), \tag{2.42}$$

where x^* is a new test case and $(x_{i.}, y_i), i = 1, ..., n$ are training samples. This is called Parzen Windows. One advantage of Parzen Windows is the outliers have less influence on weight when it deviates from majority points; however, no guarantees can be made on its predictive performance with small data sizes. The CART method introduces a decision tree to split the root set (starting from the original input data) to descendant nodes, which represents the "finer" subsets of the parental set. After several iterations, which can be decided by cross-validation, each terminal node is given a label. Then this recursive tree is fully explored and can be used for future prediction. There are certain decision strategies that should be made before processing, and the exploration of the optimal tree is known to be NP-complete. In Dudoit's paper (Dudoit et al., 2002), it seems CART is outperformed by simpler learning methods like DLDA and NN when the data size is small.

Non-parametric/semi-parametric methods have been favored in pattern recognition as a "first-pass", since they make fewer assumptions. However, they also have limitations and act more like a "black box". More specifically, direct probability estimates are absent from the results, which will cause difficulty in outlier classification. For example, if two clusters overlap, these overlapped points can be classified into either class and no probabilities are computed according to a naive SVM. On the other hand, because non-parametric/semi-parametric methods are feature space object oriented, some data points may fall into neither of these objects and remain "unclassified". This problem is more acute with high throughput data.

Random Forest

Random forest is a nonlinear and non-parametric method first proposed by Breiman (2001). The name comes from the random decision forests that was first proposed by Ho (1995). The major extension to the original decision methods is Breiman's "bagging" idea and the random selection of feature subsets. Consider a binary classification problem with a training dataset $X = \{x_1, ..., x_n\}$ and responses $Y = \{y_1, ..., y_n\}$, where n is the total number of observations. We repeatedly select a bootstrap sample of the training set and fit trees.
For $b \in (1, 2, ..., B)$:
1. sample n training cases (with replacement) from X, Y; call these X_b, Y_b
2. train the decision tree P_b on X_b, Y_b. For binary classification problems a conventional choice of training models is the logistic regression link

$$P_b(Y_b = 1) = \frac{\exp(\beta_b x_b)}{1 + \exp(\beta_b x_b)}, \tag{2.43}$$

where β_b is the logistic regression parameters in tree b. B is the total number of trees. After training, the prediction of y^* on a test case x^* can be made by averaging all the predictions from the classification trees on x^*

$$P(y^* = 1) = \frac{1}{B} \sum_{b=1}^{B} P_b(y^* = 1). \tag{2.44}$$

In the bagging algorithm, the number of trees B can be tuned according to the size of the training set. With bigger values of B, the variance of the decision trees will decrease while the computational time will grow substantially. An optimal number of trees B can be found using cross-validation to find a balance between precision and computational cost. An ad-hoc candidate list of B usually goes from hundreds to thousands in microarray data analysis.

The original random forest includes a random selection of feature subsets in each split (step 2) in the learning process. The reason for such random selection is to reduce the correlation across the B trees: if a specific set of variables has great predictive power with respect to the response variable, then these variables will be selected repeatedly across B trees, causing them to be highly correlated. An ad-hoc selection of the number of input variables tried at each split will be $\sqrt{p}$, where p is the total number of variables.

Random forest can provide the importance of variables in classification problems. Breiman suggest a permutation-based significance scoring technique to the obtain importance of each feature.

1. Fit the data set with random forest and record the out-of-bag error for each data point. More specifically, for each tree b after we obtain the model p_b we collect the test performance on all out-of-bag cases. All these out-of-bag errors are then recorded.

2. For each feature j and each tree b, the values of the jth feature are permuted among the training data. The corresponding out-of-bag error is recomputed and the difference between it and the results from step 1 (the original error) are recorded as $d_{jb}, b \in (1, ..., B)$. The average value $d_j = \frac{\sum_b d_{jb}}{B\sqrt{\text{Var}_b(d_{jb})}}$ is then recorded as the importance score for feature j.

The random forest method has been applied to various problems in bioinformatics in the last decade, including large-scale association studies for complex genetic diseases (Lunetta et al., 2004) and SNP-SNP interaction detection in a case-control context (Bureau et al., 2005). Díaz-Uriarte and De Andres (2006) compared the performance of random forest with some standard classification methods (including DLDA, KNN and SVM) with microarray data, and recommend random forest as part of a standard toolbox for gene selection and class prediction on microarray data. Unlike many other nonparametric methods, random forest provides a feature importance score for each possible feature selection choice. However, this selection may not be sparse in most cases. In this thesis, we include random forest as a standard non-parametric feature selection method to compare with our proposed full Bayesian T-probit models.

2.2 Review of Bayesian Inference with Markov Chain Monte Carlo Methods

2.2.1 Bayesian Inference

In statistics, Bayesian inference is to update the probability estimate of an event with "prior" information learned earlier with Bayes' rule

$$P(\theta|D) = \frac{L(\theta|D)\pi(\theta)}{\int L(\theta|D)\pi(\theta)d\theta}, \tag{2.45}$$

where θ is the parameter of interest and D is the given data. $\pi(\theta)$ is the *prior probability*, which indicates one's preconceived beliefs about θ before collecting data D. $L(\theta|D)$ is the *likelihood function* of θ given data D, which is usually defined by a probability density function of outcome D given θ values, that is

$$L(\theta|D) = P(D|\theta), \tag{2.46}$$

where $P(D|\theta)$ is the probability density function of outcome D given θ. The likelihood function tells us the compatibility of the parameter value θ with collected data D. $\int L(\theta|D)\pi(\theta)d\theta$ is termed the *marginal likelihood* of θ. It indicates the likelihood of the data D considering all possible parameter values. It is a common factor for each individual value of θ since it does not rely on θ. $P(\theta|D)$ is the *posterior probability.* In words, the posterior probability $P(\theta|D)$ is proportional to the combination of the prior probability of the hypothesis $\pi(\theta)$ and the compatibility of the data with the parameter values θ (likelihood $L(D|\theta)$).

The essential part of Bayesian inference is the usage of the prior information. It includes an independent source of information about the estimation of the parameter θ given the data. It makes sense to consider both the likelihood and prior to make a decision (prediction) according to the posterior probability; if either the data does not likely come from the model (with a specific parameter value) or a value of θ is highly unlikely by nature, then the posterior probability of such values of θ will be low.

Bayes' rule can be incorporated in finding posterior predictive probability. For example, let's consider classification/regression problems with independent variable x and response y. A classification/regression model is determined by the likelihood function $L(\theta|x, y) = P(y|x, \theta)$ with parameter value θ. Consider the prediction problem on a test case x^* with observed data $D = (x, y)$. Then we can integrate over space Θ (of all possible values of θ) to find the posterior predictive probability

$$p(y^*|x^*, x, y) = \int_\Theta p(y^*|x^*, x, \theta)d\theta = \int_\Theta p(y^*|x^*, \theta)p(\theta|x, y)d\theta \tag{2.47}$$

for any possible values of the test response y^*. $p(\theta|x, y)$ here is the posterior weighting factor which can be obtained with Bayes' rule

$$p(\theta|x, y) = \frac{L(\theta|x, y\pi(\theta)}{\int_\Theta L(\theta|x, y)\pi(\theta)d\theta}. \tag{2.48}$$

The posterior distribution (equation (2.48)) is the distribution of the parameter θ after taking into account the observed data $D = (x, y)$.

In practice, we can obtain the posterior predictive distribution in equation (2.47) using sampling techniques. Assume we have a set of samples of $(\theta_1, \theta_2, ..., \theta_m)$, which are independent and identically distributed random samples according to the posterior distribution $p(\theta|x, y)$. Then a natural estimation of the posterior predictive distribution will be

$$p(y^*|x^*, x) = \sum_{i=1}^{m} p(y^*|x^*, \theta_i)p(\theta_i|x, y), \tag{2.49}$$

where the integration of equation (2.47) is approximated with the summation. Apparently the estimation of posterior probability can be improved with more samples.

An alternative set of algorithms to estimate posterior prediction probability uses a single solution of parameter estimation instead of marginalization over the entire distribution $p(\theta|x)$, which is called the maximum a posteriori probability (MAP) estimate.

$$\theta_{MAP} = \underset{\theta \in \Theta}{\operatorname{argmax}}\ p(\theta|x) = \underset{\theta \in \Theta}{\operatorname{argmax}}\ p(x|\theta)\pi(\theta), \tag{2.50}$$

where $\pi(\theta)$ is the prior distribution of θ. Then the simple prediction probability on a new case x^* will be

$$p(y^*|x^*, y, x, \theta) \approx p(y^*|x^*, \theta_{MAP}). \tag{2.51}$$

With the assumption of equal *a priori* probabilities for parameter $\theta \in \Theta$, the MAP in equation (2.50) will lead to the *maximum likelihood* (ML) estimation

$$\theta_{ML} = \underset{\theta \in \Theta}{\operatorname{argmax}}\ L(\theta|x, y) = \underset{\theta \in \Theta}{\operatorname{argmax}}\ p(y|x, \theta). \tag{2.52}$$

2.2.2 Markov Chain Monte Carlo Methods

Starting in the 1980s, there was a substantial growth in applications of Bayesian methodology using Markov chain Monte Carlo Methods (MCMC) to analyze complex problems (Wolpert et al., 2004). Markov chain Monte Carlo Methods construct a Markov chain that has the desired probability distribution as its equilibrium distribution and use that Markov chain to obtain samples. The basic idea is to create a Markov chain whose equilibrium distribution is the target sampling distribution. Then, starting from an initial state θ, a proposed "jump" strategy is employed to obtain the next accepted state. After several iterations, the converged state will be considered as from the desired distribution. More specifically, consider $\theta^t = \{\theta_1^t, ..., \theta_p^t\}$ as the set of variables at step (time) t. Then the chain is defined by giving an initial value (or initial distribution) for θ^0 and transition probabilities $p(\theta^{t-1} \to \theta^t)$. The transition probabilities are carefully chosen so that the distribution of θ^t converges to the desired distribution for θ as $t \to \infty$. Then with a sufficiently long Markov chain, equation (2.49) can be used to approximate the integration.

The theory of Markov chains is well-developed (see Gilks (2005) and Norris (1998)). We will present the essential theory of Markov chains here in developing Monte Carlo methods. Instead of discussing properties of some arbitrary Markov chains, we avoid elaborations of the Markov chain theory and focus on the fact that certain Markov chains converge to a unique invariant distribution, and can be used to estimate expectations of variables of interest.

A series of variables of interest, $\theta^{(0)}, \theta^{(1)}, \theta^{(2)}, \theta^{(3)}, ...$, is called a Markov chain if the value of $\theta^{(n+1)}$ depends only on the value of $\theta^{(n)}$. More formally,

$$P(\theta^{(n+1)}|\theta^{(n)}, \{\theta^{(t)}, t \in \epsilon\}) = P(\theta^{(n+1)}|\theta^{(n)}), \tag{2.53}$$

where ϵ is any subset of $\{0, ..., n-1\}$. The index $t \in \{0, 1, 2, ...\}$ is viewed as successive "time". The possible values of $\theta^{(t)}$ are known as the state space of the Markov chain, which can be either discrete or continuous.

A Markov chain is specified by giving the initial probabilities of the state space (marginal distribution for $\theta^{(0)}$) and the transition probabilities for one state to jump to the next one (conditional distributions for $\theta^{(t+1)}$ given the possible values for $\theta^{(t)}$). The initial probability of state θ is written as $p_0(\theta)$, and the transition probability from state θ (at time t) to state θ' (at time $t+1$) is written as $T_t(\theta, \theta')$. If the transition probability $T_t(\theta, \theta')$ does not depend on the time t, the Markov chain is called homogeneous (or stationary) and $T_t(\theta, \theta')$ can just be $T(\theta, \theta')$.

A distribution $\pi(\theta)$ is called invariant with respect to the Markov chain with transition probabilities $T(\theta, \theta')$ if

$$\pi(\theta) = \int \pi(\tilde{\theta})T(\tilde{\theta}, \theta)d\tilde{\theta}. \tag{2.54}$$

It can be proven that a finite Markov chain (when the state splace is finite) always has at least one invariant distribution.

We are interested in constructing a Markov chain for which the target distribution to sample from, given by π, is invariant. In practice we tend to use time reversible homogeneous

Markov chains that satisfy the (more restrictive) detailed balance equation

$$\pi(\theta)T(\theta,\theta') = \pi(\theta')T(\theta',\theta), \tag{2.55}$$

which implies that π is an invariant distribution.

However, it is not enough to construct a Markov chain with target invariant distribution; we also require the Markov chain to be ergodic – that the probability of state θ occurring at time n, denoted by $p_n(\theta)$, converges to the target invariant distribution as $n \to \infty$, regardless of the choice of initial state θ^0. For an ergodic Markov chain the invariant distribution is unique and usually referred to as the equilibrium distribution.

Two of most important threads of development of MCMC algorithms are the Metropolis-Hasting algorithm and Gibbs sampling. Metropolis-Hasting sampling was first introduced in Metropolis et al. (1953). The method chooses the next new state by adding a small change to the current state, and accepting or rejecting this change based on how the probability of the proposed state compares to that of the current state. More specifically, let $f(\theta)$ be a function that is proportional to the desired probability distribution $p(\theta)$. We choose an arbitrary probability density $Q(\theta|\delta)$ that will suggest a candidate for the next sample value θ, given the previous sample value δ. For the Metropolis-Hasting algorithm, Q must be symmetric ($Q(\theta|\delta) = Q(\delta|\theta)$). A conventional choice for $Q(\theta|\delta)$ is a Gaussian distribution centered at δ. The function Q is referred to as the proposal (jumping) density distribution. The Metropolis-Hasting algorithm then works as follows:

1. Initialization. Choose an initial value θ_0.

2. For each iteration (step) t,

   (a) Generate a candidate value θ^* as a potential sampling from the distribution $Q(\theta^*|\theta_t)$.

   (b) Calculate the acceptance probability $\alpha = \max\{\frac{f(\theta^*)}{f(\theta_t)}, 1\} = \max\{\frac{p(\theta^*)}{p(\theta_t)}, 1\}$, which is used to decide whether to accept or reject the candidate value.

(c) Generate $U \sim$ U(0,1) (uniform distribution from 0 to 1). If $U < \alpha$ we accept the new value with $\theta_{t+1} = \theta^*$; otherwise, we keep the previous sample value $\theta_{t+1} = \theta_t$.

The Metropolis-Hasting algorithm proceeds by randomly attempting to move around the sample space. Note that the acceptance probability α indicates how probable the new proposed sample is with respect to the current sample, according to the distribution value $p(\theta)$. That means if we attempt to move to a more probable point than the existing state with higher density region of $p(\theta)$, we will always accept the move. If we propose to move to a less probable value, we will sometimes reject the move but still be capable of moving outward. Thus, we will tend to stay in (and maintain large numbers of samples from) high-density regions of $p(\theta)$ and only occasionally visit low-density regions.

However, the Metropolis-Hasting algorithm also suffers from low efficiency due to its "local" strategy. In applications, consider θ_t as a vector with length p. Then the naive Metropolis-Hasting algorithm will propose a new value θ_{t+1} from a multivariate normal density function. Then each component is from a conditional univariate normal distribution. When the value p goes large like hundreds or even thousands, the acceptance ratio will be substantially reduced with only one "bad" proposal $\theta_{t,j}$. That is, if we propose θ_t whose jth feature value $\theta_{t,j}$ has low probability $p(\theta_{t,j})$, then the total acceptance probability α will also be substantially reduced. Part of the solution lies in the "local" fashion of exploring the space. In other words, it is more feasible to propose a new value θ_{t+1} with only one different component compared with θ_t - e.g., it may differ with respect to $\theta_{t,i}$, for some i, but have $\theta_{t,j} = \theta_{t+1,j}$ for $j \neq i$. In the Markov chain framework it is possible to guarantee that such step-by-step local methods eventually produce a sample of points from the desired distribution.

A technique based on the "locality" known as Gibbs Sampling has been widely applied to obtain samples from a specified multivariate probability distribution (from the joint probability distribution of two or more random variables). This method is named after the physicist Josiah Willard Gibbs and described by brothers Stuart and Donald Geman in Geman and Geman (1984). Suppose we want to obtain samples of random variables $\theta = (\theta_1, ..., \theta_p)$ from a joint distribution $p(\theta_1, ..., \theta_p)$. We obtain the transition from $\theta^{t-1} = (\theta_1^{t-1}, ..., \theta_p^{t-1})$ to $\theta^t = (\theta_1^t, ..., \theta_p^t)$ with Gibbs sampling as follows:

$$\text{Draw } \theta_1^t \text{ from } p(\theta_1|\theta_2^{t-1}, ..., \theta_p^{t-1})$$

$$\text{Draw } \theta_2^t \text{ from } p(\theta_2|\theta_1^t, \theta_3^{t-1}, ..., \theta_p^{t-1})$$

$$...$$

$$\text{Draw } \theta_i^t \text{ from } p(\theta_i|\theta_1^t, ..., \theta_i^t, \theta_{i+1}^{t-1}, ..., \theta_p^{t-1})$$

$$...$$

$$\text{Draw } \theta_p^t \text{ from } p(\theta_p|\theta_1^t, ..., \theta_{p-1}^t)$$

The samples can then be proved to approximate the joint distribution of all variables. The initial value is not necessary to guarantee the convergence of the Markov chain; however, it does impact the efficiency of the Markov chain mixing speed. That is, if the initial value is far from the high probability density region it may take a very long time for the Markov chain to travel around the sample space since it will take the first few iterations to jump out of the "remote" region whose probability density function value is low. Alternatively, the initial value can be determined by other algorithms such as MLE, MAP, EM algorithm, etc. Their solutions may not be in the high probability density region but it is usually helpful to start the Markov chain from a single parametric estimator.

Another set of sampling techniques called "molecular dynamics" was proposed by Alder and Wainwright (1959) around the same time as when the Metropolis-Hasting sampling algorithm was developed. The primary idea is to find new states by simulating the dynamical evolution of the system. We refer to this technique as the dynamical method, since it can in fact be used to sample from any differential probability distribution, not just distributions for systems of molecules. A combination of the Metropolis-Hasting algorithm and the dynamical method leads to the **Hamiltonian Monte Carlo** method of Duane et al. (1987).

2.2.3 Hamiltonian Monte Carlo Method

The original molecular dynamics approach was first proposed to simulate physical systems with the help of the Metropolis-Hasting algorithm (Alder and Wainwright, 1959). The original dynamical approach only works to sample from the canonical distribution for states under conditions of constant total energy. Andersen (1980) introduced a stochastic element so that the energy and volume of the phase states can fluctuate. More formally, suppose we want to sample from the canonical distribution for a set of real variables, $q = \{q_1, ..., q_n\}$, with respect to the potential energy function, $E(q)$, which is differentiable with respect to the q_i. The canonical distribution is defined as

$$P(q) = \frac{1}{C} \exp\left(-E(q)\right), \tag{2.56}$$

where C is a constant so that the distribution integrates to 1. In applications of molecular dynamics problems, the q_i are the position coordinates to be simulated (with $n = 3$). In Bayesian statistical inference, the q_i are our model parameters of interest. In the context below we use q_i to denote the "position" variables.

We introduce another set of variables, $p = \{p_1, ..., p_n\}$ with kinetic energy function $K(p) = \frac{1}{2}\sum_i p_i^2$. In the context of molecular dynamics, p_i are the components of the momentum of the particles, and in Bayesian inference p_i are auxiliary random variables. We will refer to the p_i as the "momentum" variables.

All possible states of the combination of position and momentum variables are known as *phase space*. In phase space each unique point corresponds to the states of both position variables and momentum variables. The total energy function for all possible points in phase space is known as the *Hamiltonian*,

$$H(q, p) = E(q) + K(p) = E(q) + \frac{1}{2}\sum_i p_i^2. \tag{2.57}$$

The canonical distribution over phase space can then be written as

$$\begin{aligned} P(p, q) &= \frac{1}{C} \exp\left(-H(q, p)\right) \\ &= \frac{1}{C_1} \exp\left(-E(q)\right) \frac{1}{C_2} \exp\left(-K(p)\right) \\ &= P(q)P(p), \end{aligned} \tag{2.58}$$

where C_1 and C_2 are the corresponding integration factors of the marginal distribution for q and p. If we can sample from the canonical distribution over phase space, we can also obtain a sample of values for q from the desired distribution $P(p)$.

We then use the Hamiltonian function to define a dynamics on phase space, and treat both q_i and p_i as functions of a "time" parameter τ. τ is supposed to satisfy the Hamiltonian dynamics equations

$$\frac{dq_i}{d\tau} = +\frac{\partial H}{\partial p_i} = p_i, \tag{2.59}$$

$$\frac{dp_i}{d\tau} = -\frac{\partial H}{\partial q_i} = -\frac{\partial E}{\partial q_i}, \tag{2.60}$$

where τ is the physical time, which is totally arbitrary in statistical inference. It should not be confused with the actual Markov chain discrete time parameter t.

The total energy H is conserved under time evolution since

$$\frac{dH}{d\tau} = \sum_j [\frac{\partial H}{\partial q_j}\frac{dq_j}{d\tau} + \frac{\partial H}{\partial p_j}\frac{dp_j}{d\tau}] = \sum_j [\frac{\partial H}{\partial q_j}\frac{\partial H}{\partial p_j} - \frac{\partial H}{\partial p_j}\frac{\partial H}{\partial q_j}] = 0. \tag{2.61}$$

The Hamiltonian dynamics can also be viewed as a deterministic transformation T_s on the phase space (the space of (p, q)) mapping the state at $(p(t), q(t))$ to the state at $(p(t+s), q(t+s))$. The dynamics preserves volume under all these kinds of transformations. Actually, if we follow these points in some small region with volume V, then the volume of the range of transformation T_s is also V. Then it follows that the total canonical probability, defined by the total energy H, in these two sets under transformation are also equal to each other. This is known as Liouville's theorem

$$\sum_i [\frac{\partial}{\partial q_i}(\frac{dq_i}{d\tau}) + \frac{\partial}{\partial p_i}(\frac{dp_i}{d\tau})] = \sum_i \left[\frac{\partial^2 H}{\partial q_i \partial p_i} - \frac{\partial^2 H}{\partial p_i \partial q_i}\right] = 0. \tag{2.62}$$

In practice we need to use discretization methods to approximate the dynamics equations ((2.59) and (2.60)) with non-zero time evolution steps. The reason is that we can not follow the dynamics exactly, and it can be proven that Liouville's theorem still holds for discretization, while the discretization errors can be corrected using the Metropolis-Hasting acceptance-rejection step.

One common discretization scheme is the leapfrog method. It is time reversible (which is crucial for the Hamiltonian Monte Carlo method, as we are going to introduce), and also preserves phase space volume. In each single iteration during leapfrog we calculate the transformation of p and q from time τ to time $\tau + \epsilon$:

$$\hat{p}_i(\tau + \frac{\epsilon}{2}) = \hat{p}_i(\tau) - \frac{\epsilon}{2}\frac{\partial E}{\partial q_i}(\hat{q}(\tau)) \tag{2.63}$$

$$\hat{q}_i(\tau + \epsilon) = \hat{q}_i(\tau) + \epsilon \hat{p}_i(\tau + \frac{\epsilon}{2}) \tag{2.64}$$

$$\hat{p}_i(\tau + \epsilon) = \hat{p}_i(\tau + \frac{\epsilon}{2}) - \frac{\epsilon}{2}\frac{\partial E}{\partial q_i}(\hat{q}(\tau + \epsilon)). \tag{2.65}$$

The leapfrog method involves a half step updating for p_i, a full step for q_i, and a full step for p_i. It can also be replaced by a half step for q_i, a full step for p_i, and a full step for q_i. But the alternative is slightly less convenient since it is almost always easier to update the momentum variable p. ϵ in equation (2.60) represents a change of time period, which should be small enough to give an acceptably small discretization error. The last step of equation (2.61) and the first equation (2.59) are combined since the updating of p_i in equation (2.61) is immediately followed by the half-step updating of p_i in equation (2.59).

Hamiltonian Monte Carlo (HMC) samples points in phase space ((p, q)) with a Markov chain. The dynamical transition in HMC is similar to that of stochastic dynamical methods, with only 2 modifications – first, during each transition the Markov chain can both jump forward (in time) or backward; second, the final position of the transition will only be a proposal for the Metropolis-Hasting algorithm. The proposed value can be rejected with a rejection probability based on the total energy change as in the Metropolis-Hasting algorithm. If the dynamics functions in equation (2.59) and equation (2.60) are followed exactly, the total energy change and the rejection probability will be zero, and the proposed value will be accepted. This kind of rejection mechanism is crucial to eliminate the dynamical errors introduced by the inexact approximation of the leapfrog method. More formally, the dynamical transitions of the HMC algorithm can be implemented as follows:

1. Randomly choose a direction, $\lambda \in \{1, -1\}$, to define a forward or backward trajectory.
2. Starting from the current phase state $(q, p) = (q_0, p_0)$, implement L leapfrog steps with a stepsize ϵ, and end up with a new state (q^*, p^*).
3. Calculate the acceptance probability $\alpha = \min(1, \exp(-(H(q^*, p^*) - H(q_0, p_0))))$, which is used to decide whether to accept or reject the candidate state (q^*, p^*).
4. Generate $U \sim$ U(0,1) (uniform distribution from 0 to 1). If $U < \alpha$ we accept the new state (q^*, p^*); otherwise, we reject the candidate and keep the previous state (q_0, p_0).

The values of ϵ and L need to be chosen carefully. In practice, it is good for the number of leapfrog steps, L, to be fairly large, so the proposed value can explore the target sampling distribution thoroughly. If the value of ϵ is too large, the acceptance probability will be too low, and if the value is too small, the leapfrog method will jump rather slowly. Both values of ϵ and L should be carefully chosen for a specific situation and we will discuss this later in the thesis.

An important variant of the Hamiltonian Monte Carlo method is the Langevin Monte Carlo Method (LMC). Similar to HMC, Langevin Monte Carlo proposes a new state with leapfrog methods using only one single leapfrog iteration ($L = 1$). As in the case of the Hamiltonian Monte Carlo, LMC can produce exactly unbiased results. It is often viewed as a “simple” version of HMC. In this thesis, we will include both HMC and LMC in the computational efficiency experiments to assess the performance of our proposed sampling methods with the Hamiltonian Monte Carlo method.

Chapter 3
Fully Bayesian T-probit Regression with Heavy-tailed Priors

3.1 T-probit Models with Heavy-tailed Priors

A probit model is a type of regression model in which the response variables $y_i, i = 1, ..., n$ can only take binary values: 0 and 1. The vector of regressors x_i is assumed to influence the response value y_i. Traditionally, probit models can be expressed using a continuous auxiliary variable z so that many tools for linear models can be used (Andrews and Mallows, 1974). More specifically, we have the auxiliary probit model with a normal link function as follows:

$$y_i = \begin{cases} 1, & \text{if} \quad z_i > 0 \\ 0, & \text{if} \quad z_i < 0 \end{cases} \tag{3.1}$$

$$z_i = x_i\beta + \epsilon_i,$$

$$\epsilon_i \sim N(0, 1).$$

where $\beta = (\beta_0, \beta_1, ..., \beta_p)'$ is the coefficient vector, x_i contains the ith observation of all features and z_i is the auxiliary variable.

With z_i integrated out, the probit model can also be expressed with a conditional probability function:

$$P(y_i|x_i, \beta) = \Phi(x_i\beta)^{y_i}(1 - \Phi(x_i\beta))^{1-y_i}, \tag{3.2}$$

where Φ is the cumulative distribution function (CDF) of the standard normal distribution. There are also other binary outcome models with different choices of link functions F (replacing Φ), for example, with the logistic function the model becomes logistic regression.

One advantage of auxiliary variable methods is to bridge the classification model with a familiar linear model. For linear models Gaussian error is common, but may not be appropriate here because of the existence of outliers. Robit regression (Lange et al., 1989) replaces the normal link distribution with Student's t-distribution and the model becomes

$$y_i = \begin{cases} 1, & \text{if} \quad z_i > 0 \\ 0, & \text{if} \quad z_i < 0 \end{cases} \tag{3.3}$$

$$z_i = x_i\beta + \epsilon_i,$$

$$\epsilon_i \sim T(\alpha_0, \omega_0),$$

where $T(\alpha_0, \omega_0)$ is the scaled t-distribution with shape parameter α_0 and scale parameter $\sqrt{\omega_0}$. The probability density function (PDF) value of the scaled t-distribution at ϵ can be given as:

$$t_{\alpha_0,\omega_0}(\epsilon) = \frac{\Gamma(\frac{\alpha_0+1}{2})[1+\frac{\epsilon^2}{\omega_0\alpha_0}]^{-(\frac{\alpha_0+1}{2})}}{\sqrt{\omega_0\alpha\pi}\Gamma(\frac{\alpha_0}{2})}, \tag{3.4}$$

where $\Gamma(.)$ is the Gamma function. As in the probit model, with z_i integrated out, the whole model is equivalent to the following conditional probability:

$$P(y_i|x_i,\beta) = T_{\alpha_0,\omega_0}(x_i\beta)^{y_i}(1 - T_{\alpha_0,\omega_0}(x_i\beta))^{1-y_i}, \tag{3.5}$$

where T_{α_0,ω_0} is the cumulative distribution function of the scaled t-distribution with shape parameter α_0 and scale parameter $\sqrt{\omega_0}$:

$$T_{\alpha_0,\omega_0}(x_i\beta) = \frac{1}{2} + \frac{x_i\beta}{\sqrt{\omega_1}}\Gamma\left(\frac{\alpha_1+1}{2}\right)\frac{{}_2F_1\left(\frac{1}{2},\frac{\alpha_1+1}{2};\frac{3}{2};-\frac{(x_i\beta)^2}{\alpha_1\omega_1}\right)}{\sqrt{\pi\alpha_1}\Gamma\left(\frac{\alpha_1}{2}\right)}, \tag{3.6}$$

where ${}_2F_1$ is the hypergeometric function, which is given as the sum of an infinite series (Abramowitz and Stegun, 1972). This model was also called the robit model by Liu (2004). In this thesis, we refer to it as **T-probit** to emphasize the role of the t distribution and also the similarity with probit models.

In most high-dimensional problems, we want to find the very few features are most relevant to the response y. One of the key components in our method is the assignment of the t distribution with small degrees of freedom, α_1, and small scale, $\sqrt{\omega_1}$, as a prior for β, which is written as follows:

$$\beta_j \sim T(\alpha_1, \omega_1), \quad \text{for} \quad j = 1, \ldots, p. \tag{3.7}$$

For Markov chain Monte Carlo (MCMC) sampling consideration, we express the above t prior for β as a scale-mixture normal by introducing a new variance λ_j for each β_j, which is given as follows:

$$\beta_j|\lambda_j \sim N(0, \lambda_j), \tag{3.8}$$

$$\lambda_j \overset{iid}{\sim} \text{Inverse-Gamma}(\alpha_1/2, \alpha_1\omega_1/2),$$

Hereafter, we will write $\lambda = (\lambda_1, \ldots, \lambda_p)$. The intercept β_0 is assigned with a normal distribution with large variance since β_0 may be centered far from 0.

If the auxiliary variable z_i in equation (3.3) is integrated away, then the **heavy-tailed T-probit model** can be written as:

$$P(y_i|x_i, \beta) = T_{\alpha_0,\omega_0}(x_i\beta)^{y_i}(1 - T_{\alpha_0,\omega_0}(x_i\beta))^{1-y_i}, \tag{3.9}$$

$$\beta = (\beta_0, \beta_1, \ldots, \beta_p)'$$

$$\beta_j|\lambda_j \sim N(0, \lambda_j),$$

$$\lambda_j \overset{iid}{\sim} \text{Inverse-Gamma}(\alpha_1/2, \alpha_1\omega_1/2).$$

The choices of values for α_1 and ω_1 are critical, reflecting our knowledge about the sparseness of β. From our empirical studies in 4.4, an ad-hoc choice for α_1 is 1, which corresponds to the Cauchy distribution; see the empirical investigations in logistic regression (Li and Yao, 2014). The posterior is fairly robust to the choice of ω_1. Theoretically, the choice of a very small value is also crucial. Because when ω_1 is large, the posterior may be influenced by the shape of the t distribution near the origin (which is close to normal) rather than the heavy tails. From our empirical studies, we recommend fixing ω_1 at a value around $\exp(-10)$, i.e., a scale of 0.007 for the t distribution. More discussions about the choices of α_1 and ω_1 are provided by Li and Yao (2014). This choice of scale seems to be significantly smaller than other values we have seen from the literature for non-convex penalization learning with optimization approaches - for example the default value, 2.5, recommended by Gelman et al. (2008). Optimization approaches may have trouble with very small scale. For example, using a small scale like 0.007, the R function `bayesglm` in the R package `ARM` (which implements penalized logistic regression with t priors) will converge to a mode where almost all coefficients are shrunken to a value very close to 0. The MCMC algorithm to be described in this thesis can handle such small scales; hence, the hyper-LASSO sparseness property in the tails of t distribution can be utilized.

3.2 MCMC Sampling for T-probit Models

We propose our MCMC sampling algorithm to sample from the joint posterior of (β, λ), denoted by $f(\beta, \lambda|y, X)$, which is based on the hierarchical models given by equations (3.9), (3.8), and (3.1) with $\alpha_0, \omega_0, \alpha_1, \omega_1$ fixed (so omitted in the following model descriptions). The log posterior can be written as follows:

$$\log(f(\beta, \lambda|y, X)) = \sum_{i=1}^{n} \log(P(y_i|x_i, \beta)) + \sum_{j=0}^{p} \log(f(\beta_j|\lambda_j)) + \sum_{j=1}^{p} \log(f(\lambda_j)) + C, \quad (3.10)$$

where the first three terms come from the models defined by equations (3.9), (3.8), and (3.1). C is the log of the normalization constant unrelated to β and λ. In more detail, the first three terms in equation (3.10) are given as follows:

$$\begin{aligned} \text{lp}(y_i, x_i\beta) &\equiv \log(P(y_i|x_i, \beta)) = y_i \log(T_{\alpha_0,\omega_0}(x_i\beta)) + (1-y_i)\log(T_{\alpha_0,\omega_0}(-x_i\beta)) && (3.11)\\ \log(f(\beta_j|\lambda_j)) &= -\frac{1}{2}\log(\lambda_j) - \frac{\beta_j^2}{2\lambda_j} + C_1, \text{ for } j = 0, \dots, p && (3.12)\\ \log(f(\lambda_j)) &= -\left(\frac{\alpha_1}{2}+1\right)\log(\lambda_j) - \frac{\alpha_1\omega_1}{2\lambda_j} + C_2, \text{ for } j = 1, \dots, p. && (3.13) \end{aligned}$$

where C_1 and C_2 are two constants unrelated to (β, λ). Also note that λ_0 is fixed.

Our MCMC sampling algorithm (referred to as Restricted Gibbs Sampling with HMC) uses Gibbs sampling with the Hamiltonian Monte Carlo (HMC) method for sampling β. The algorithm starts with an initial value for (β, λ). The choice of initial value does not change the nature of our sampling algorithm. In this thesis, we use LASSO solution as a guess of the initial point. Then, given a previous state for (β, λ), we iteratively obtain a new state denoted by $(\hat{\beta}, \hat{\lambda})$ as follows:

Restricted Gibbs Sampling with HMC

step 1: For each j, draw a new $\hat{\lambda}_j$ from the conditional distribution $f(\lambda_j|\beta_j)$ with log PDF equal to the sum of equation (3.12) and equation (3.13). It is well-known that λ_j given β_j has an Inverse-Gamma distribution given as follows:

$$\lambda_j|\beta_j \sim \text{Inverse-Gamma}\left(\frac{\alpha_1+1}{2}, \frac{\alpha_1\omega_1+\beta_j^2}{2}\right). \tag{3.14}$$

step 2: With the new values of $\hat{\lambda}_j$ drawn in step 1, determine a subset, β_U, of β to update in step 3 below. We update β_j if $\hat{\lambda}_j$ is large enough. That is, given a prescribed threshold value η, the subset is defined as $U = \{j|\hat{\lambda}_j > \eta\}$. The β_U is defined as $\{\beta_j|j \in U\}$. The subset of $\beta_F = \{\beta_j|j \in F = \{0, \dots, p\} \setminus U\}$ will be kept unchanged in step 3.

step 3: Update the set of β_j with $j \in U$, denoted by β_U, by applying HMC to the conditional distribution of β_U given as follows:

$$\begin{aligned}&\log(f(\beta_U|\beta_F, \hat{\lambda}, X, y)) \\ &= \sum_{i=1}^{n} \mathsf{lp}(y_i, x_{i,U}\beta_U + x_{i,F}\beta_F) + \sum_{j\in U} \log(f(\beta_j|\hat{\lambda}_j)) + C_3,\end{aligned} \quad (3.15)$$

where the function lp for computing log likelihood is defined in equation (3.11), and $x_{i,U}$ is the subset of x_i with its feature index in U. More details of HMC are given in the Section 2.2.3. After updating $\hat{\beta}_U$, the new value of β is denoted by $\hat{\beta}$ in which β_F does not change. Note that, because HMC is a Metropolis algorithm, the new $\hat{\beta}$ may be equal to β if a rejection occurs.

step 4: Set $(\beta, \lambda) = (\hat{\beta}, \hat{\lambda})$, and go back to step 1 for the next iteration.

We will make additional remarks and supplement some details about the above algorithm:

1. Even if a new proposed β_U is rejected in step 3, the subset U in step 2 will change in the next iteration due to the new values of $\hat{\lambda}$ drawn in step 1.

2. The thresholding with $\hat{\lambda}$ in step 2 chooses only the coefficients with large variance λ_j to update. This is intended to save a great deal of computation because most coefficients are near 0. Updating the coefficients with small λ_j in HMC does not change the likelihood as much as updating the coefficients with large λ_j, but will consume the same computing time. We often choose η so that only 10% of the values in β are updated in step 3.

3. We can save computing time in step 3 by caching values of $x_{i,F}\beta_F$ from the previous iteration. Computing the linear combinations $x_i\beta$ is the major computation in evaluating the log posterior of β. The thresholding in step 2 can save a great deal of computation.

4. The thresholding in step 2 does not change the Markov chain property of the above algorithm. The reason is that the choice of subset U does not depend on the value of β, therefore, the transition in step 3 is reversible with respect to $f(\beta_U, \beta_F|\hat{\lambda}, X, y)$. As

a result, the combination of step 2 and step 3 is a reversible Markov chain transition with respect to the conditional distribution $f(\beta|\hat{\lambda}, X, y)$. Note that the Markov chain is not reversible if we integrate λ out, i.e., using the t density equation (3.7) directly as prior for β, and threshold on β directly in step 2. This is an advantage of using the scale-mixture normal expression (equations (3.8) and (3.1)) instead of using the t density equation (3.7).

5. In applying HMC, we need to compute the gradient of

$$\mathcal{U}(\beta_U) = -\log(f(\beta_U|\beta_F, \hat{\lambda}, X, y)), \tag{3.16}$$

which is called potential energy function; see details in Section 2.2.3. The gradient will be used to define the leapfrog trajectory. The first-order partial derivative is given by this formula:

$$\frac{\partial \mathcal{U}}{\partial \beta_j} = \sum_{i=1}^{n} \frac{\Gamma\left(\frac{\alpha_0+1}{2}\right)}{\sqrt{\alpha_0 \pi}\,\Gamma\left(\frac{\alpha_0}{2}\right)} \times \frac{x_{ij}}{\sqrt{\omega_0}} \times \frac{\left(1 + \frac{(x_{i,U}\beta_U + x_{i,F}\beta_F)^2}{\alpha_0\omega_0}\right)^{-\frac{\alpha_0+1}{2}}}{1 - y_i - T_{\alpha_0,\omega_0}(x_{i,U}\beta_U + x_{i,F}\beta_F)} + \frac{\beta_j}{\hat{\lambda}_j}. \tag{3.17}$$

6. An advantage of HMC is that it can explore a highly correlated posterior quickly without suffering the random walk with a long leapfrog trajectory. This property plays an important role in the sampler's ability to travel quickly between multiple modes of the posterior. This is explained as follows. When $\hat{\lambda}_j$ and $\hat{\lambda}_k$ for two correlated features j and k are large after a draw in step 1, (β_j and β_k) given $\hat{\lambda}_j$ and $\hat{\lambda}_k$ are highly correlated. For such distributions, HMC can move more quickly than Gibbs sampling algorithms along the least constrained direction to avoid random walk, and this move will lead to the change of modes in the joint distribution of (β_j, β_k) with λ marginalized.

When the number of features p is large, such as thousands, it is necessary to implement two-stage sampling for T-probit models. The reason is that the number of modes in the posterior when p is large is be too huge to summarize. Therefore, we first run the restricted Gibbs sampling with HMC using the dataset containing all p features, called Stage 1. Then we calculate the MCMC means of all coefficients β and choose only the top $p^* = 100$ features with the largest absolute values of MCMC means. The choice of p^* does not change the property of our MCMC methods and can be arbitrarily made according to users' need. We then run the MCMC sampling once again with only the selected features (Stage 2). Our feature selections will be based on the MCMC samples obtained from Stage 2.

More details of HMC sampling and a list of parameters with recommended values are given in Section 2.2.3 and Section C.2, respectively.

3.3 Dividing MCMC Samples to Find Feature Subsets

We run the MCMC sampling as described in Section 3.2 to obtain samples from the posterior of β. With the intercept β_i removed, this sample is denoted by a matrix $B = (\beta_{j,i})_{p\times R}$, in which $\beta_{j,i}$ represents the value of β_j in the ith sample and R is the number of MCMC samples. In many cases, the posteriors of β based on T-probit models with heavy-tailed priors are highly multi-modal. For a demonstration, one can look at Figure 4.1, which shows a scatterplot of MCMC samples of two β's for two correlated features from one of our simulation examples. We divide the Markov chain samples B into sub-pools according to the mode that each sample represents. However, the number of such feature subsets may be huge even if the number of features p is small. Therefore, we only consider dividing the Markov chain samples according to the multiple modes for the Markov chain samples obtained in stage 2. In this thesis, we use a scheme that examines the relative magnitude of β_j in comparison to the largest value in all features. Our scheme is described as follows:

step 1: We set $I_{j,i} = 1$ if $|\beta_{j,i}| > 0.1 \times \max\{\beta_{1,i}, \ldots, \beta_{p,i}\}$, and $I_{j,i} = 0$ otherwise. By this way, we obtain a boolean matrix $(I_{j,i})_{p\times R}$ with entry $I_{j,i}$ denoting whether the jth feature is selected or not in the ith sample.

step 2: Discard the features with overall low frequency in step 1. We calculate $f_j = \frac{1}{R}\sum_{i=1}^{R} I_{j,i}$. We will discard a feature j if f_j is smaller than a pre-defined threshold, which is set to be 5% as an ad-hoc choice. Let $D = \{j | f_j < 5\%\}$. For each $j \in D$, we set $I_{j,i} = 0$ for all $i = 1, ..., R$. This step is to remove the features that are selected in step 1 due to MCMC randomness.

step 3: Find a list of feature subsets by looking at the column vectors of I. Each different column in I represents a different feature subset.

The algorithm above may not be the best to accurately divide the MCMC samples according to the posterior modes of β. The reason is that the MCMC simulation introduces small jitters into the samples of the features that are not selected in the mode. This is why we need to use the thresholds 0.1 in step 1 and 5% in step 2 to avoid the selection of features with the jitters. However, the resulting feature subsets selection will depend on the choices of the thresholds. The choices of 0.1 and 5% we use in this thesis are only ad-hoc based on our empirical experiences. A better method may be to find the posterior modes starting from each MCMC sample using a certain optimization algorithm. However, finding such posterior modes for a large number of MCMC samples is time-consuming. In this thesis, we present the results based on the fast algorithm described here for simplicity.

3.4 Feature Subset Evaluation Using Cross-validation

In statistics, there are a number of different ways to evaluate predictive accuracy in binary response models. The predictive performance can be measured using several different criteria, including error rate (ER), average minus log-probability (AMLP) on true response, and area under the ROC (receiver operating characteristic) curve (AUC). Suppose the true responses of n cases are $y_1, \ldots, y_n$. A prediction method provides an estimate of $Pr(y_i = 1|x_i)$ with $\hat{p}_i(1)$ and $Pr(y_i = 0|x_i)$ with $\hat{p}_i(0)$. Note that the predictive probability at the true label y_i can be written as $\hat{p}_i(y_i)$. Then the error rate (ER) is calculated as:

$$\mathbf{ER} = \frac{1}{n}\sum_{i=1}^{n} I(\hat{y}_i \neq y_i), \tag{3.18}$$

where $\hat{y}_i = \arg\max_c \hat{p}_i(c), c = 1, 2$ is our prediction of the true label y_i using a threshold of 0.5. That is, $\hat{y}_i = 1$ if $\hat{p}_i(1) > 0.5$; otherwise, $\hat{y}_i = 0$. The ER criterion is very useful to evaluate the performance of the predictions at the boundary 0.5, but does not punish very poor predictions. For example, suppose that the true response of a test case is 1, for which method A gives a predictive probability 0.49 that it is from class 1, and method B gives 0.2; using 0.5 as threshold, they are both wrong, but we can see that method A is better.

The second criterion is defined as the average of minus log predictive probabilities (AMLP):

$$\text{AMLP} = \frac{1}{n} \sum - \log(\hat{p}_i(y_i)). \tag{3.19}$$

We also include ROC curves to assess the performance of $\hat{p}_i$ as we vary the discrimination threshold. The curve reflects the true positive rate value and the false positive rate value at different threshold settings. The true positive rate is the proportion of positive cases that are correctly identified as such. The false positive value is the number of false (positive) predictions over negative conditions, and the false positive rate is the ratio of false positive value to negative conditions. The true-positive rate is also known as sensitivity in biomedicine (Pepe, 2000), which measures a "hit" for the purpose of prognosis. The false positive rate is also known as the fall-out and can be calculated as 1 - specificity. In this thesis we will collect the corresponding AUC (area under the ROC curve) value as an important measurement to evaluate the performance of different classifiers. Among all three methods mentioned above, the ER is the most intuitive measurement to evaluate prediction accuracy, but it may not be enough to differentiate models. For example, it may be more important for a model to give better predictive performance on "tough" cases (i.e., improve the prediction probability for the true label from 20% to 30%) even though both of them are wrong, rather than improving the prediction probability for a true label from 80% to 90%. Average minus log-probability (AMLP) serves this purpose: for each response value y_i we calculate the predicted probability for true cases. The predictive probability of $P(Y_i = y_i)$ for the true label y_i is $\hat{p}(Y_i = y_i)$, and $\log(\hat{p}(y_i))$ gets smaller when the prediction probability gets smaller. In other words, AMLP puts a greater penalty penalize on the wrong prediction of cases of interest.

For each feature subset found after we divide the MCMC samples we can examine the cross-validated training predictive power of different feature subsets. For example, if we find a top feature subset (x_1, x_2, x_8), we then use (x_1, x_2, x_8) as the new training dataset and find the cross-validated AMLP/AUC/ER of predictive probabilities given by a method that uses only these three features. More specifically, the training dataset is first reduced to 3-

dimensional new data with the selected feature subset $x_{1,2,8}$ and dimensions $n \times 3$. Then we divide the training cases into leave-one-out folds. For each fold f, the new training matrix can be denoted as $x_{,S}$ where x is the matrix containing all p features (columns). S is the subset containing all selected features ($S = \{1, 2, 8\}$ here). Then $x_{,S}$ is a matrix with dimensions $(n-1) \times 3$ since we have $n-1$ observations and 3 features in S. The test case is a vector containing 3 features. We use the R package `arm` to fit a Bayesian generalized logistic model to the new training cases and test the model on the fold-specific test case. The resulting prediction probability is then recorded for each fold. Then over all folds we collect the prediction results and find ER/AMLP/AUC with the true labels. This process is repeated for each feature subset we found from the MC samples. An sample of such a cross-validated evaluation table is given as follows:

Table 3.1: A feature subsets table with cross-validated prediction AMLP/ER/AUC

	fsubsets	freqs	cv_AMLP	cv_ER	cv_AUC	coefs(w/int)
1	1,2,8	0.27	0.37	0.17	0.91	-1.88,6.13,-0.86,6.01
2	1,2,11	0.09	0.38	0.18	0.91	0.48,4.06,-1.93,1.91
3	1,2	0.07	0.47	0.26	0.84	-1.35,10.93,-6.11
4	1,2,7	0.06	0.39	0.17	0.90	-0.74,6.64,-3.56,4.38

3.5 Out-of-Sample Predictions with Fitting Results of Bayesian T-probit Models

In this section, we describe three methods for making predictions for test cases. The conventional prediction is made using the whole Markov chain. In Bayesian analysis, the posterior prediction for a test case with features x^* given a training case with features $X = (x_1, ..., x_n)$ and response $Y = (y_1, ..., y_n)$ is

$$P(y^*|x^*, X, Y) = \int_{\beta} P(y^*|x^*, \beta)P(\beta|X, Y)d\beta, \tag{3.20}$$

where $P(y^*|x^*, \beta)$ is the new likelihood and $P(\beta|X, Y)$ is the posterior probability of parameter β.

Given a Markov chain, we have $(\beta_1, \beta_2, ..., \beta_m)$ as random samples according to the posterior distribution of β. A natural estimation of the posterior predictive probability will be:

$$\hat{p}(y^*|x^*, X, Y) = \sum_{i=1}^{m} P(y^*|x^*, \beta_i)/m. \tag{3.21}$$

We compute the prediction probability $P(y^*|x^*, \beta_i)$ and average all probabilities over the MC samples. The samples will naturally reflect the posterior probability if the MC is mixed well, and then we only need to record $\sum_{i=1}^{m} P(y^*|x^*, \beta_i)/m$ since each single sample accounts for $1/m$ of the entire set of MC samples. We refer to this method as the "Tpro_average" method.

Alternatively, we can find prediction probabilities with specific feature subsets. Two of the most interesting feature subsets are the "Top" feature subset and the "Optimal" feature subset. The "Top" feature subset is the one with the highest frequency (among all subsets) across the Markov chains. That is, given a set of training data (and corresponding training process), we can analyze the Markov chain samples and find the "Top" feature subset with the highest frequency. Then we use this feature subset S to find the predictive probability on the test case y^* with features $x^*_{,S}$ and estimation of β_S. We refer to this prediction method as the "Tpro_top" method. Similarly, we have the "Tpro_optimal" method to find predictive probabilities on a test case using the "optimal" feature subset, which is the feature subset that generates the optimal value of training average minus log-probability (among all feature subsets). In this thesis, we provide predictive performance of both the "Tpro_top" and "Tpro_optimal" methods in each data analysis.

We implement all three prediction methods ("Tpro_average", "Tpro_top" and "Tpro_optimal") and evaluate them as well as other methods in simulation studies and real data analysis. The evaluation process is slightly different for simulations and real data analysis. In simulation studies, we generate a training dataset (X, Y) and a test dataset (X^*, Y^*). The T-probit regression is trained solely on (X, Y) and the resulting Markov chain samples are saved. We first identify the "Top" and "Optimal" feature subsets S from the Markov chain samples, and then we find the prediction probabilities of the test cases Y^* with features $X^*_{,S}$ and estimation of β_S. In real data analysis, we have data (X, Y) where X has limited numbers of observations n (usually hundreds). Thus we evaluate the predictive performance of all methods using cross-validation. The overall prediction probabilities across folds are then recorded and the corresponding ER/AMLP/AUC values are used to evaluate predictive performance of all three methods. An example of the resulting prediction table (both from simulation and real data analysis) is shown in Table 3.2.

Table 3.2: An example table of predictive performance with the T-probit model using Top selected feature subsets, Optimal feature subsets and averaging over all MC samples.

	Tpro_top	Tpro_optimal	Tpro_average
ER	0.19	0.1	0.1
AMLP	0.57	0.38	0.33
AUC	0.86	0.94	0.91
No. of Features	2.85	1.26	

Chapter 4
Simulation Studies

4.1 A Toy Example for Demonstrating the Separation of Correlated Features into Different Feature Subsets

We generate a dataset with only 2 features (p=2) and 100 samples (n=100). The dataset is generated from the following model:

$$P(y_i = c) = \frac{1}{2}, i = 1, 2, ..., n, c = 1, 2, \tag{4.1}$$

$$\mu_{1,j} = 0, \mu_{2,j} = 2, j = 1, 2,$$

$$z_i \sim N(0, 1),$$

$$\epsilon_{i1} \sim N(0, 1), i = 1, 2, ..., n,$$

$$\epsilon_{i2} \sim N(0, 1), i = 1, 2, ..., n,$$

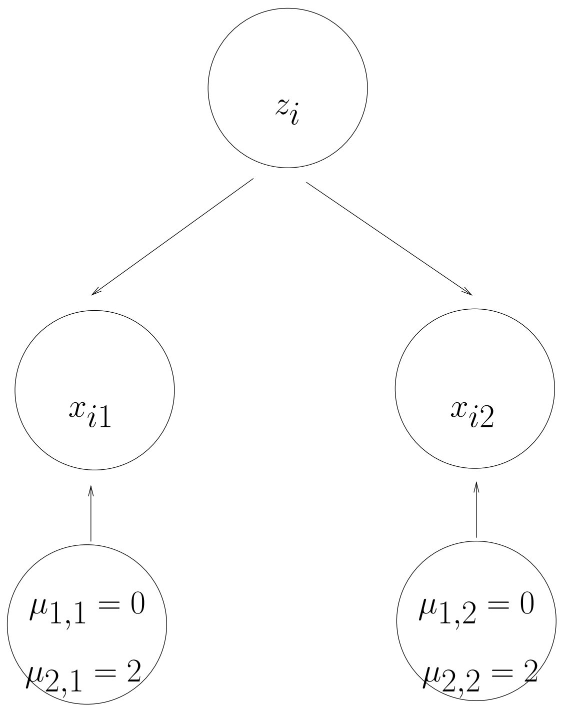
$$x_{i1} = \mu_{y_i,1} + z_i + 0.1\epsilon_{i1}, i = 1, 2, ..., n,$$

$$x_{i2} = \mu_{y_i,2} + z_i + 0.1\epsilon_{i2}, i = 1, 2, ..., n,$$

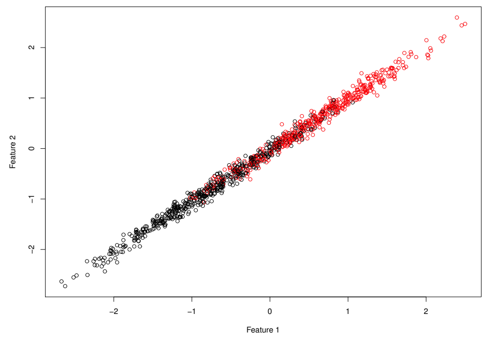
$$\epsilon_{ij} \sim N(0, 1), i = 1, 2, ..., n, j = 1, 2,$$

where y_i can only take values 1 or 2. $\mu_c, c = 1, 2$ are the mean centroids of two features with $\mu_{1,j} = 0$ and $\mu_{2,j} = 2$ for feature $j = 1, 2$. x_{ij} is the ith observation value of feature j

$(x_j, i = 1, 2)$.



(a) Data structure demonstration. x_{i1} and x_{i2} are the values of feature 1 and feature 2 from the ith observation. $\mu_{y_i,j}$ are the sample mean values of feature 1 and feature 2, where y_i is the response value of the ith observation.



(b) Scatterplot of feature 1 and feature 2 values.

Figure 4.1: Data structure demonstration and scatterplot of feature 1 and feature 2 values.

The structure of this data set is shown in Figure 4.1a. From the graph we can see that both feature 1 and feature 2 are relevant to class label y with similar predictive power. They both have a nonzero mean value for $y_i = 1$ and $y_i = 2$ ($\mu_{1,j} = 0$, $\mu_{2,j} = 2$). z_i is the common factor shared by feature 1 and feature 2, while ϵ_{i1} and ϵ_{i2} are independent components of each feature (from the ith observation). As a result, feature 1 and feature 2 are strongly correlated with each other with correlation 99.5%, and they also have similar predictive power with respect to the response y_i. Figure 4.1b shows a scatterplot of feature 1 and feature 2.

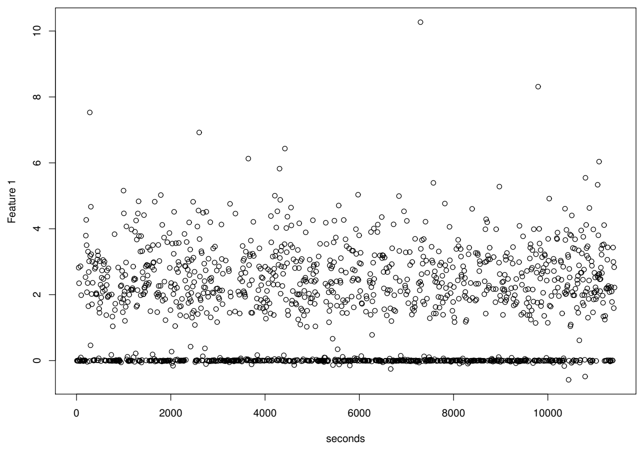
We run MCMC on this dataset with the parameter settings given in Appendix C. The initial values of β and λ are computed with LASSO, and then we fit our T-probit regression with all 100 features. In the end, we record 12,000 iterations of β and λ in 11413 seconds. The Markov chain samples obtained with T-probit regression are demonstrated in Figure 4.2, Figure 4.3, Figure 4.4 and Figure 4.5. Figure 4.2 shows both Markov chain samples of coefficients of feature 1 and feature 2 (separately), and Figure 4.4 shows the results of Markov chain samples after the first 285 seconds. We also provide the scatterplot of the feature 1 and feature 2 Markov chain samples (at 11413 seconds and the first 285 seconds) in Figure 4.3 and Figure 4.5. Both Figure 4.2 and Figure 4.4 show that feature 1 and feature 2 are selected separately into two different modes across Markov chains; that is, in each single Markov chain sample either feature 1 or feature 2 is selected. Figure 4.3b and Figure 4.5 show sample values of the two features connected with solid lines. Both graphs show that T-probit regression can switch between feature 1 and feature 2. In conclusion, the T-probit regression selects the 2 features with similar frequency.

The coefficients from LASSO, Bayesglm and random forest are shown in Table 4.2. The T-probit regression selection results are presented in Table 4.1, which shows that T-probit regression gives similar selection frequency to feature 1 and feature 2, with 56% and 42% frequency among all Markov chain samples, respectively. This means that in most Markov chain samples (98%), either feature 1 or feature 2 will be selected, while in 2% of the Markov chain samples T-probit regression will select feature 1 and feature 2 simultaneously. In conclusion, the T-probit regression is able to provide multiple feature subset selection results, while other methods (including LASSO, Bayesglm and random forest) only provide a single vector of coefficients which signifies the importance of both features.

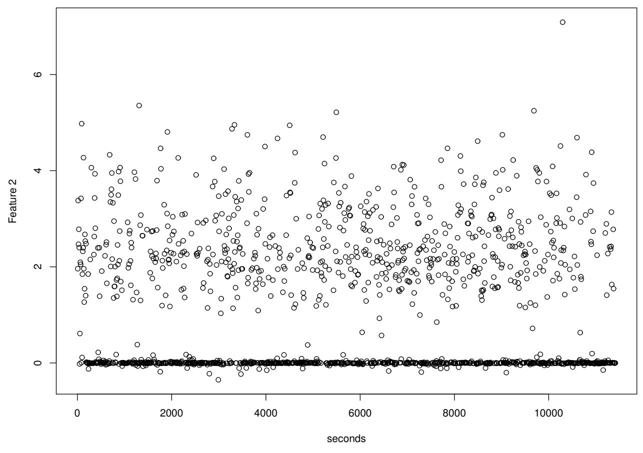
Table 4.1: Feature Subset Selection Table with T-probit regression. fsubsets: feature subsets. freqs: frequency of the corresponding feature subset. coefs: coefficient values (with intercept).

	fsubsets	freqs	coefs(w/int)
1	1	0.56	0.25,2.82
2	2	0.42	0.08,1.96
3	1,2	0.02	0.26,2.35,0.61

We also show the prediction results of all methods on test cases in Table 4.3 and Table 4.4. Table 4.3 shows the prediction results of T-probit regression using the three different feature subsets found in Table 4.1, and Table 4.4 shows the prediction results of coefficient

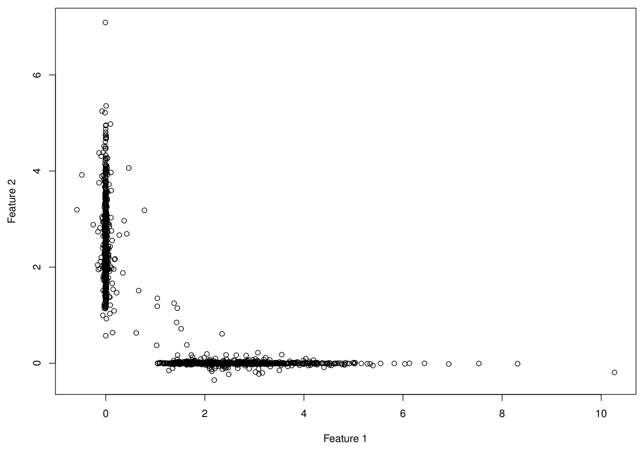


(a) The entire set of Markov chain coefficients of feature 1 after 11413 seconds.

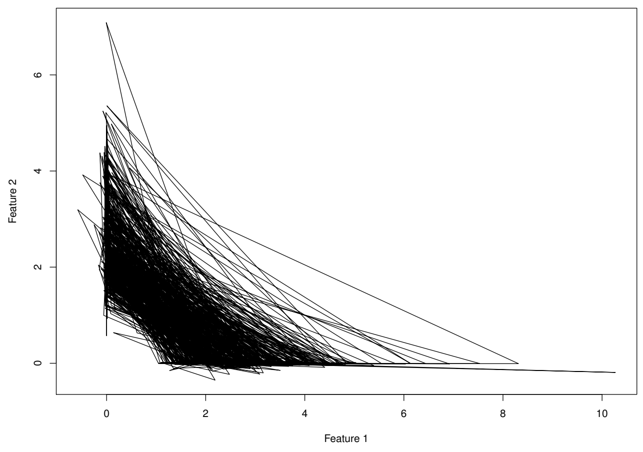


(b) The entire set of Markov chain coefficients of feature 2 after 11413 seconds.

Figure 4.2: The entire set of Markov chain coefficients of feature 1 and feature 2.



(a) Scatterplot of Markov chain coefficients of feature 1 and feature 2.

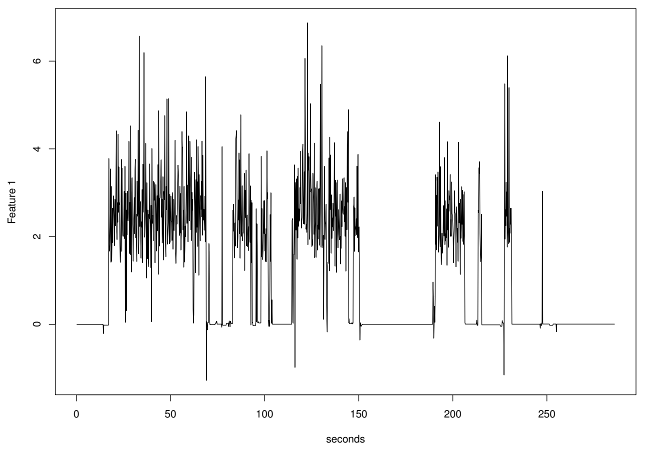


(b) Scatterplot of Markov chain coefficients of feature 1 and feature 2 with line connections.

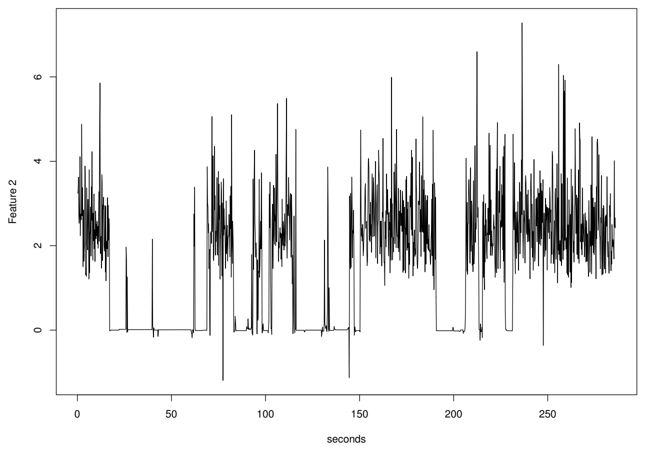
Figure 4.3: Scatterplot of Markov chain coefficients of feature 1 and feature 2.

Table 4.2: Feature Coefficients of LASSO, Bayesglm and Random Forest.

	LASSO	Bayesglm	Random Forest
Feature 1	1.15	24.63	1.26
Feature 2	1.27	24.53	1.26

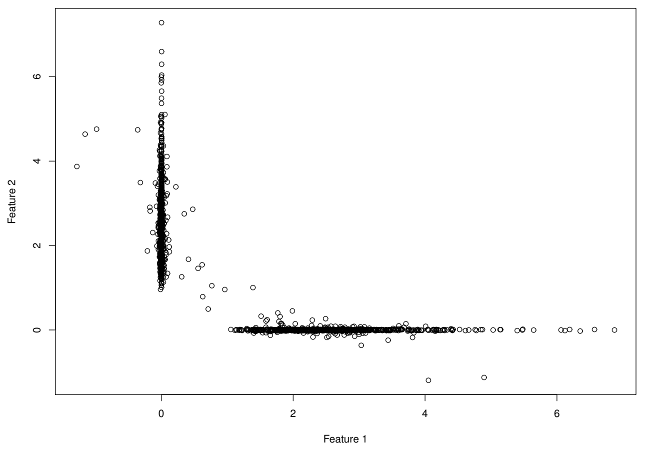


(a) Traceplot of Markov chain coefficients of feature 1 after the first 285 seconds.

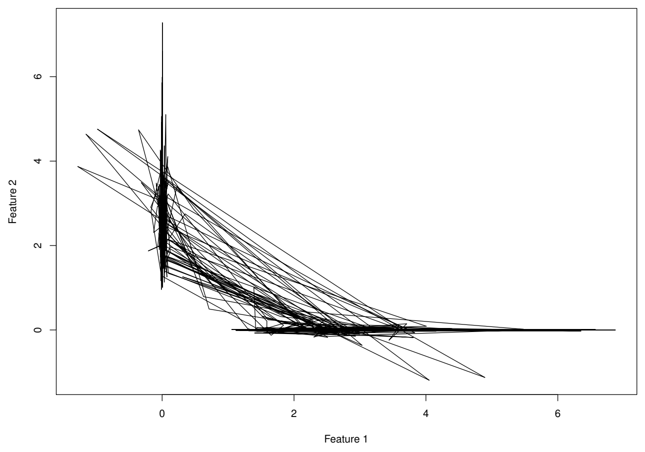


(b) Traceplot of Markov chain coefficients of feature 2 after the first 285 seconds.

Figure 4.4: Traceplot of Markov chain coefficients of feature 1 and feature 2 after the first 285 seconds.



(a) Scatterplot of Markov chain coefficients of feature 1 and feature 2 after the first 285 seconds.



(b) Scatterplot of Markov chain coefficients of feature 1 and feature 2 (after the first 285 seconds) with line connections.

Figure 4.5: Scatterplot of Markov chain coefficients of feature 1 and feature 2 after the first 285 seconds.

vectors from the other methods. From the results, we can tell that all methods provide similar prediction results with AUC close to 0.91, AMLP close to 0.37 and ER close to 180 out of 1000 test cases. The worst result is from random forest, which has relatively weak predictive power with ER 219/1000 and AUC 0.88. The predictive performance of T-probit regression is not superior to other methods on this simple case; however, it is able to provide satisfactory prediction results using only one feature. In conclusion, the T-probit regression is able to detect relevant features with strong correlations, and only select one of them as a representative in a single Markov chain sample.

Table 4.3: Feature Subset Prediction Table of T-probit regression.

	fsubsets	AMLP	ER	AUC
1	1	0.37	0.185	0.91
2	2	0.37	0.180	0.91
3	1,2	0.37	0.178	0.91

Table 4.4: Prediction Table of LASSO, Bayesglm and Random Forest.

Method	AMLP	ER	AUC
LASSO	0.37	0.184	0.91
Random Forest	Inf	0.219	0.88
Bayesglm	0.37	0.184	0.91

4.2 Feature Selection Performance in Datasets with Independent Groups of Features

We generate datasets with dimension $p = 2000$ (the total number of features) and $n = 1200$ (the total number of cases). Among all 1200 cases, 200 are used as training cases and 1000 are test cases. The response vector y and feature values $x_{ij}, i = 1, ..., n, j = 1, ..., p$ are generated as follows:

$$z_{ij} \sim N(0, 1), i = 1, ..., n, j = 1, 2, 3, \tag{4.2}$$

$$\epsilon_{ij} \sim N(0, 1), i = 1, ..., n, j = 1, ..., 150$$

$$\epsilon_i \sim N(0,1), i = 1, ..., n,$$

$$x_{il} = z_{i1} + 0.5\epsilon_{il}, i = 1, ..., n, l = 1, ..., 50, \textbf{(Group 1)}$$

$$x_{im} = z_{i2} + 0.5\epsilon_{im}, i = 1, ..., n, m = 51, ..., 100, \textbf{(Group 2)}$$

$$x_{in} = z_{i3} + 0.5\epsilon_{in}, i = 1, ..., n, n = 101, ..., 150, \textbf{(Group 3)}$$

$$x_{ik} \sim N(0,1), i = 1, ..., n, k = 151, ..., 2000, \textbf{(Group 4)}$$

$$\Phi_i = \frac{z_{i1} + z_{i2} + z_{i3}}{\sqrt{3}} + 0.1\epsilon_i, i = 1, ..., n,$$

$$y_i = I_{\Phi_i > 0}, i = 1, ..., n,$$

where Φ_i is the value of the auxiliary variable from the ith observation. We define Group 1 as the feature group containing the first 50 features, Group 2 contains feature 51 to feature 100 and Group 3 contains feature 101 to feature 150. z_{i1}, z_{i2} and z_{i3} are common factors of Group 1, Group 2 and Group 3 respectively. ϵ_{ij} are the "noises" attached to the jth feature from the ith observation. We construct an auxiliary variable Φ_i and each observation of response y_i takes value 1 when $\Phi_i > 0$ and value 0 otherwise. Data generated in this way can be viewed as from a robit model with nonzero coefficients values (1,1,1) for feature 1 (x_1), feature 51 (x_{51}) and feature 101 (x_{101}).

The data structure is shown in Figure 4.6. From this figure, we can see that all features (x_1 to x_{150}) from three groups (Group 1 to Group 3) are "relevant" to the response since they all share information with feature 1 (x_1), feature 51 (x_{51}) and feature 101 (x_{101}). $x_k, k = 151, ..., 2000$ are all irrelevant and are denoted as Group 4 features. We then implement T-probit regression and other methods including LASSO, Group LASSO (GL), Supervised Group LASSO (SGLASSO), random forest (RF), Penalized Logistic Regression (using Bayesglm) and simple Student's t-test (t-test). The T-probit method is conducted with parameter settings α_0=1, ω_0=0.5, α_1=1, ω_1=exp(-10), R_1=2000, n_1=1000, R_2=3000, n_2=1000, l_1=5, l_2=50, ϵ=0.5, η=0.05. Readers may refer to Appendix C for an explanation of the tuning parameters of the other methods. The goal of this simulation study is to evaluate the feature selection performance of T-probit (as well as the other methods) between

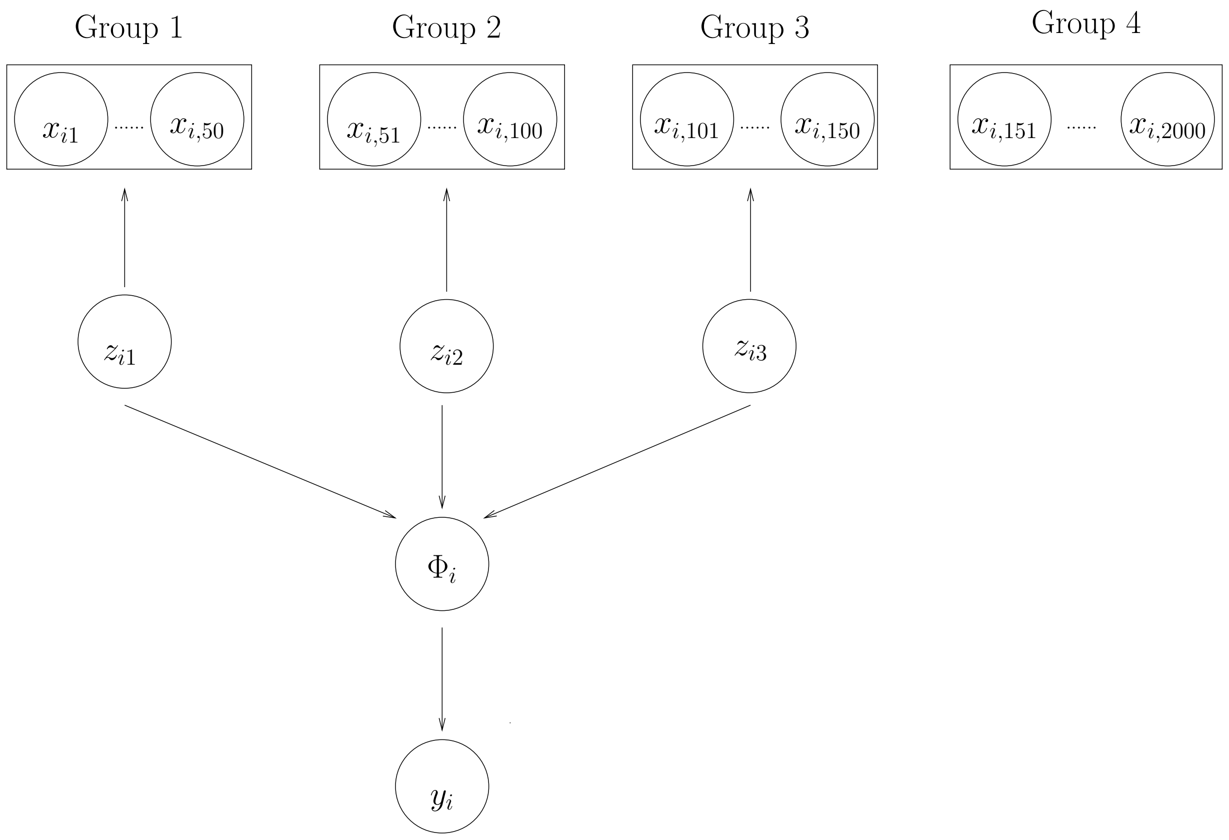


Figure 4.6: Data structure demonstration. $x_{i1}, ..., x_{i,50}$ are the feature values (in the first group) from the ith observation. $x_{i,51}, ..., x_{i,100}$ are the feature values (in the second group) from the ith observation. $x_{i,101}, ..., x_{i,150}$ are the feature values (in the third group) from the ith observation. $x_{i,151}, ..., x_{i,2000}$ are the feature values (in the fourth group) from the ith observation. z_{i1}, z_{i2}, z_{i3} are three independent values. z_{i1}, z_{i2}, z_{i3} determine the value of Φ_i. The response values y_i (of the ith observation) are then decided by Φ_i.

groups (independent groups of features) and within groups.

We show the feature selection results of all methods based on a single dataset simulated with the above method in Table 4.5 and Table 4.6. The numbers of features selected by each method (except Tpro_optimal and Tpro_top) are determined by 0.1 multiplied by the maximum value of the coefficients. That is, for a coefficient vector β, the number of features selected by this method is equal to $\sum_{j=1}^{2000} I_{\{\beta_j > 0.1 \max\{\beta_j\}\}}$. For the Tpro_optimal and Tpro_top methods, this number is just the size of the selected feature subset. Table 4.5 gives the feature subsets extracted from the Markov chain samples returned by T-probit. For each of these feature subsets, we record its coefficient values and frequency across all of the MC samples. In table 4.6, for each method (each column) we show the number of features selected from Group 1 to Group 4. For example, the Tpro_top method (the first column) picks one feature from each of Group 1 to Group 3 and it discards all of the features from Group 4, which are basically random noise. From table 4.6 we see both Group LASSO and Bayesglm successfully select all (150) relevant genes; however, they also select a large number of irrelevant features from Group 4. LASSO and Supervised Group LASSO both provide sparse feature selection results; however, they both select more than 10 irrelevant features from Group 4. The T-probit is the only method that discards all of the irrelevant features. More importantly, all of the top 4 feature subsets from Table 4.5 contain one feature from each of the three different groups. This implies that T-probit is able to select a representative feature from each group. The top feature subset (1,57,140) and optimal feature subset (1,51,140) contain no irrelevant features. In all feature subsets shown in Table 4.5, there are 7 features selected across all the Markov chains, and none of them is from the noise group (Group 4). The detailed feature selection results are shown in Figure D.7a to Figure D.10b.

We also show predictive performance of all methods with respect to AMLP, NMC (number of misclassified cases) and AUC in Table 4.7. The T-probit method makes predictions for test cases in 3 different ways: Tpro_top makes predictions with only the top feature subset found by T-probit, i.e., the subset (1, 57, 140) in Table 4.5. Tpro_optimal uses the optimal (with respect to cross-validated training AMLP) feature subset selected with T-probit regression. Tpro_average makes predictions by averaging the results of each single iteration across the Markov chains. The results show that the values of ER and AUC stay at a similar level for all

	fsubsets	freqs	coefs(w/int)	cv_AMLP	cv_NMC	cv_AUC
1	1,57,140	0.22	0.4,14.93,16.19,15.97	0.13	9.00	0.99
2	1,51,140	0.11	0.02,19.18,14.06,18.85	0.13	8.00	0.99
3	16,57,140	0.10	-0.24,11.97,13.05,13.21	0.14	8.00	0.99
4	1,51,101	0.09	0.75,19.33,19.3,17.03	0.14	8.00	0.99
5	12,57	0.04	-0.6,13.52,4.65	0.41	39.00	0.89
6	57,140	0.04	-0.48,18.33,15.01	0.45	43.00	0.87
7	1,57,101	0.04	-0.14,13.56,12.1,13.15	0.14	12.00	0.99
8	1,57,121	0.03	2.75,19.5,15.72,17.9	0.14	11.00	0.99
9	16,57,101	0.03	1.45,18.63,14.45,15.91	0.14	12.00	0.99
10	16,51,140	0.03	0.94,16.65,17.59,18.44	0.14	9.00	0.99
11	51,140	0.02	1.78,17.48,14.91	0.45	48.00	0.87
12	1,51	0.02	0.03,17.88,18.04	0.42	43.00	0.88
13	12,57,101	0.02	1.21,19.96,20.19,16.93	0.15	12.00	0.99
14	1,57	0.02	-0.96,16.67,13.68	0.40	35.00	0.90
15	1,140	0.02	2.06,16.34,19.06	0.42	33.00	0.89
16	12,51,101	0.01	-2.36,8.76,13.13,11.18	0.15	7.00	0.99
17	16,51,101	0.01	0.85,11.92,20.96,19.32	0.15	8.00	0.99
18	12,57,140	0.01	0.61,2.44,12.46,10.28	0.15	13.00	0.99

Table 4.5: Cross-validated prediction results with top selected feature subsets. For each selected feature subset its predictive performance (AMLP/NMC/AUC) is measured with cross-validation on 1000 cases using Bayesglm.

	Tpro_top	Tpro_optimal	LASSO	GL	SGLASSO	RF	Bayesglm	T
Group1	1.00	1.00	6.00	49.00	7.00	49.00	50.00	43.00
Group2	1.00	1.00	5.00	50.00	10.00	49.00	50.00	44.00
Group3	1.00	1.00	6.00	50.00	6.00	48.00	50.00	44.00
Group4	0.00	0.00	13.00	341.00	12.00	14.00	1305.00	42.00

Table 4.6: The number of features selected within each group from all methods. Group 1, Group 2 and Group 3 are the groups with significant features, while Group 4 is the group containing noise features. GL: Group LASSO. SGLASSO: Supervised Group LASSO. RF: random forest. Bayesglm: bayesian logistic regression. T: *t*-test.

methods. Only the AMLP (average minus log-probability) values are substantially different. Among all methods, the simple t-test has the smallest AMLP values (0.15 and 0.14). LASSO, Group LASSO, and Supervised Group LASSO show competitive results (0.18-0.24), while random forest shows the worst AMLP performance (0.38). Group LASSO is expected to outperform LASSO since it is a LASSO approach embedded with a grouping index; however, both its ER and AMLP performance here are not superior to LASSO. One reason may be that it does not clearly identify useful group information and over-selects features. In the mean time Tpro_top and Tpro_optimal both have satisfactory prediction results using only 3 feature representatives from three groups.

In Table 4.8 we show the predictive performance of feature subsets selected by the T-probit method as well as other methods using the same number of features. From the table we can see the top feature subset and optimal feature subset selected by the T-probit method both contain 3 feature representatives from 3 groups. In the top 3 features selected by the LASSO method, Group LASSO and random forest Group 1 and Group 2 are both selected; however, Group 3 is totally missed by all these methods. Supervised Group LASSO missed features from Group 2 and other methods (Bayesglm and t-test) include irrelevant features (in the top 3 features). The predictive performance of the T-probit methods in Table 4.8 are superior to the others since they include relevant feature representatives, one from each group, and ignore all irrelevant features. In conclusion, among all competitors, T-probit is the most efficient prediction method to achieve satisfactory prediction results using the same number of features.

	Tpro_top	Tpro_optimal	Tpro_average	LASSO	GL	SGLASSO	RF	Bayesglm	T
ER	0.10	0.10	0.06	0.09	0.07	0.10	0.08	0.08	0.07
AMLP	0.22	0.22	0.15	0.21	0.22	0.24	0.38	0.18	0.14
AUC	0.97	0.97	0.99	0.97	0.99	0.97	0.98	0.98	0.99
No. of Features	3	3		30	490	35	160	1455	173

Table 4.7: Prediction results (ER, AMLP and AUC) and the number of features selected from each method.

To validate the consistency of the feature selection and prediction results, we also repeat this simulation study with 100 different experiments (datasets) generated with the mechanism described above. The corresponding average prediction results are given in Table 4.9 and feature selection results are shown in Table 4.10. From the two tables we can see

Method	fsubsets	AMLP	ER	AUC
T-probit top subset	1,57,140	0.22	0.10	0.97
T-probit optimal subset	1,57,140	0.22	0.10	0.97
LASSO	16,57,61	0.46	0.22	0.87
Group LASSO	16,32,57	0.44	0.20	0.88
Supervised Group LASSO	16,138,140	0.47	0.24	0.86
Random Forest	28,50,67	0.46	0.22	0.86
Bayesglm	12,32,218	0.63	0.34	0.72
t-test	57,140,167	0.44	0.21	0.88

Table 4.8: Top feature subsets selected by each method and corresponding predictive performance (AMLP, ER, and AUC) on the test cases.

that Tpro_average is the most powerful prediction method (with AMLP 0.12 and ER 0.04). LASSO, Group LASSO and Supervised Group LASSO have competitive predictive power as well. The feature selection results are similar to the previous single data results. Compared to the other methods, Tpro_top and Tpro_optimal select feature subsets that are much smaller in size and also have better predictive power.

	Tpro_top	Tpro_optimal	Tpro_average	LASSO	GL	SGLASSO	RF	Bayesglm	T
ER	0.05	0.06	0.04	0.08	0.06	0.08	0.10	0.08	0.05
AMLP	0.15	0.16	0.12	0.21	0.20	0.20	0.39	0.17	0.13
AUC	0.99	0.99	0.99	0.98	0.99	0.98	0.97	0.98	0.99
No. of Features	3.00	3.40		32.39	551.17	27.06	147.42	1447.68	169.05

Table 4.9: Prediction results (ER, AMLP and AUC) of each method (averaged over 100 datasets).

	Tpro_top	Tpro_optimal	LASSO	GL	SGLASSO	RF	Bayesglm	T
Group 1	1.00	1.07	6.01	49.95	6.20	47.82	50.00	42.93
Group 2	1.00	1.08	6.00	49.94	5.99	47.48	50.00	42.90
Group 3	1.00	1.06	5.94	49.95	6.04	48.34	50.00	43.13
Group 4	0.00	0.19	14.44	401.33	8.83	3.78	1297.68	40.09

Table 4.10: The number of features selected within each group for each method (averaged over 100 datasets). Group 1, Group 2, and Group 3 are the groups with significant signals, and Group 4 is the group of noise features.

We generated another dataset with larger noise to examine the robustness of T-probit regression (and all the other methods). More specifically, we replace the noise in equation (4.2) from $0.1\epsilon_i$ to $0.5\epsilon_i$ and generate a new dataset. The corresponding feature selection table is shown in Table 4.11. The table shows that T-probit still selects feature representatives from all three groups. However, this time all 3 feature subsets are chosen with low frequency. We also show the corresponding prediction table in Table 4.12. Among all methods Tpro_average has the lowest ER and AMLP with sparse feature selection results (compared with the other methods). Tpro_top and Tpro_optimal both select three features from three groups and ignore all noise features, and with only three features their predictive performance will be competitive to other methods using hundreds (Group LASSO) or even thousands (Bayesglm) of features. The group feature selection (Table 4.13) shows similar results as in Table 4.6. Among all methods, T-probit methods (both Tpro_top and Tpro_optimal) provide the most sparse feature selection results and ignore all irrelevant features. Group LASSO is able to find all relevant features; however, it also includes far more irrelevant ones. Random forest is balanced between the selection of relevant features and irrelevant ones. However, the predictive performance of random forest is unsatisfactory. In summary, compared to other methods, Tpro_top and Tpro_optimal select subsets that are much smaller in size but with slightly worse (but very similar) predictive power.

	fsubsets	freqs	coefs(w/int)	cv_AMLP	cv_ER	cv_AUC
1	20,81,128	0.02	0.22,9.52,6.01,9.57	0.22	14.00	0.97
2	20,55,128	0.02	-0.25,14.84,12.23,8.65	0.20	10.00	0.97
3	50,89	0.01	0.82,15.07,13.11	0.37	38.00	0.91

Table 4.11: Cross-validated prediction results with top selected feature subsets (from the whole dataset with T-probit regression). For each selected feature subset, its prediction values (AMLP/NMC/AUC) are obtained using cross-validation on 1000 cases using Bayesglm.

	Tpro_top	Tpro_optimal	Tpro_average	LASSO	GL	SGLASSO	RF	Bayesglm	T
ER	0.21	0.20	0.16	0.18	0.19	0.20	0.18	0.19	0.19
AMLP	0.45	0.44	0.32	0.42	0.40	0.49	0.44	0.46	0.71
AUC	0.90	0.90	0.92	0.90	0.90	0.90	0.91	0.92	0.91
No. of Features	3.00	3.00		35.00	666.00	30.00	148.00	1449.00	142.00

Table 4.12: Prediction results (ER, AMLP and AUC) and the number of features selected by each method. Tpro_top conducts feature selection and prediction with only the top feature subset selected by T-probit. Tpro_optimal conducts feature selection and prediction with the optimal (with respect to cross-validated training error) feature subset selected with T-probit. Tpro_average conducts feature selection and prediction results averaging over the Markov chains obtained with T-probit regression.

We also repeat these experiments 100 times to verify consistency of T-probit (and all the

	Tpro_top	Tpro_optimal	LASSO	GL	SGLASSO	RF	Bayesglm	T
Group 1	1	1	7	50	9	50	50	31
Group 2	1	1	6	50	6	50	50	38
Group 3	1	1	4	50	6	46	50	33
Group 4	0	0	18	516	9	2	1299	40

Table 4.13: The number of features selected within each group by each method. Group 1, Group 2, and Group 3 are the groups with significant signals, and Group 4 is the group of noise features.

other methods) on this type of data. The averaged prediction and feature selection tables are given in Table 4.14 and Table 4.15. These tables show that T-probit has the best AMLP and ER values. The T-probit method also provides a sparse feature selection with more relevant features and fewer irrelevant features compared with the other methods; Tpro_top selects 0.94, 0.94, 0.94, and 0.73 features from Group 1 - 4, respectively. It means that, on average, the Tpro_top method almost always selects one feature from each group in Groups 1 - 3, and sometimes (with 0.73 frequency) selects features from Group 4. The Tpro_average method employs all feature subsets found in all Markov chain iterations and, on average, selects 3.38, 3.59, 3.17, and 1.56 features from the four groups, respectively, which is also far more sparse than the other methods (e.q., LASSO with 4.14, 4.2, 4.19, and 15.47). Comparing Table 4.15 with Table 4.10, we can see that all methods tend to include more irrelevant features on these difficult datasets. In conclusion, based on these difficult datasets, the T-probit methods are still able to find succinct feature subsets consisting of feature representatives from Group 1 to Group 3, and the Tpro_average method outperforms all other methods with respect to predictive performance.

	Tpro_top	Tpro_optimal	Tpro_average	LASSO	GL	SGLASSO	RF	Bayesglm	T
ER	0.21	0.20	0.15	0.17	0.16	0.19	0.17	0.17	0.19
AMLP	0.46	0.48	0.33	0.41	0.36	0.43	0.47	0.43	0.65
AUC	0.88	0.89	0.92	0.90	0.92	0.90	0.90	0.91	0.90
No. of Features	3.55	4.31		28.00	705.22	27.42	165.58	1488.62	151.09

Table 4.14: The average numbers of features selected and prediction results (ER, AMLP and AUC) of all methods over 100 datasets.

	Tpro_top	Tpro_optimal	LASSO	GL	SGLASSO	RF	Bayesglm	T
Group 1	0.94	0.98	4.14	49.80	5.42	46.48	50.00	37.20
Group 2	0.94	0.97	4.20	49.78	5.48	46.64	50.00	36.86
Group 3	0.94	1.00	4.19	49.92	6.02	47.25	49.98	37.47
Group 4	0.73	1.36	15.47	555.72	10.50	25.20	1338.64	39.56

Table 4.15: The average numbers of features selected within the signal groups and the noise group of all methods over 100 datasets.

4.3 Feature Selection Performance in Datasets with Independent and Correlated Groups

In this section, we simulate data with both independent and correlated group structures. More specifically, we generate multivariate normal data with 3 variables. The first 2 features are correlated and both weakly differentiated, while the third feature is independent with the first 2 features but more differentiated. More specifically, we generate the dataset in this way:

$$P(y_i = c) = \frac{1}{2}, i = 1, 2, ..., n, c = 1, 2, \tag{4.3}$$

$$\mu_{1,1} = -0.3, \mu_{2,1} = 0.3,$$

$$z_{ij} \sim N(0,1), i = 1, ..., n, j = 1, 2, 3,$$

$$\epsilon_{ij} \sim N(0,1), i = 1, ..., n, j = 1, ..., 600,$$

$$x_{ij} = \mu_{y_i,1} + z_{i1} + 0.5\epsilon_{ij}, i = 1, ..., n, j = 1, ..., 200, \textbf{(Group 1)}$$

$$\mu_{1,2} = 0.3, \mu_{2,2} = -0.3,$$

$$x_{ij} = \mu_{y_i,2} + 0.8z_{i1} + 0.6z_{i2} + 0.5\epsilon_{ij}, i = 1, ..., n, j = 201, ..., 400, \textbf{(Group 2)}$$

$$\mu_{1,3} = 1, \mu_{2,3} = -1,$$

$$x_{ij} = \mu_{y_i,3} + z_{i3} + 0.5\epsilon_{ij}, i = 1, ..., n, j = 401, ..., 600, \textbf{(Group 3)}$$

$$x_{ij} \sim N(0,1), i = 1, ..., n, j = 601, ..., 2000, \textbf{(Group 4)}$$

where the centroid matrix ($\mu_{c,1:3}$) for the features in Group 1 to Group 3 can also be written as

$$\mu_{c,1:3} = \begin{pmatrix} -0.3 & 0.3 & 1 \\ 0.3 & -0.3 & -1 \end{pmatrix}$$

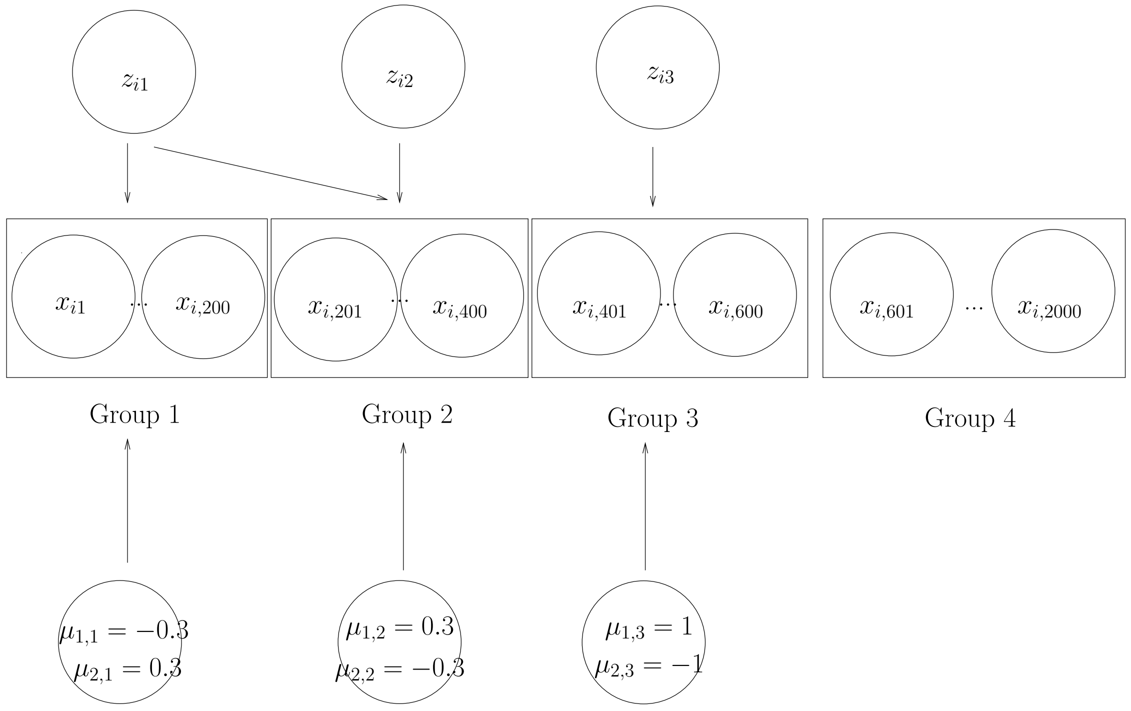


Figure 4.7: Data structure demonstration. $x_{i1}, ..., x_{i,200}$ are the feature values (in the first group) from the ith observation. $x_{i,201}, ..., x_{i,400}$ are the feature values (in the second group) from the ith observation. $x_{i,401}, ..., x_{i,600}$ are the feature values (in the third group) from the ith observation. $x_{i,601}, ..., x_{i,2000}$ are the feature values (in the fourth group) from the ith observation. z_{i1}, z_{i2}, z_{i3} are three individual values. $\mu_{y_i,j}, j = 1, 2, 3$ are sample mean values of features from Group j, where y_i is the response value of the ith observation.

The data structure generated in this way is shown in Figure 4.7. From the figure we can see that there are 3 significant groups and each one contains 200 features. We define Group 1 as the feature group containing the first 200 features, Group 2 contains feature 201 to feature 400 and Group 3 contains feature 401 to feature 600. Group 4 contains all remaining features. The data structure contains Group 1, Group 2 and Group 3 as significant features, with all remaining 1850 features classified as noise. A strong correlation (0.8) exists within each group, and Group 1 - Group 2 have a strong correlation (0.64) between groups. Group

3 is independent from Group 1 and Group 2; however, it has a larger difference in centroids in the two classes ($\mu_1 = -1$, $\mu_2 = 1$) than Group 1 and Group 2 ($\mu_1 = -0.3$, $\mu_2 = 0.3$) .

We run T-probit regression on the training cases with parameter settings α_0=1, ω_0=0.5, α_1=1, ω_1=exp(-10), R_1=2000, n_1=100, R_2 =1500, n_2=100, l_1=5, l_2=50, ϵ=0.5, η=0.05. We compare T-probit with LASSO, Group LASSO, Supervised Group LASSO, random forest, Bayesglm and the simple t-test. To obtain a grouping index for Group LASSO and Supervised Group LASSO, we use K-means correlation and Gap statistics to choose the number of clusters and the corresponding K-means clustering index for each feature. We then run Group LASSO and Supervised Group LASSO method with the same grouping index.

	fsubsets	freqs	coefs(w/int)	cv_AMLP	cv_ER	cv_AUC
1	119,235,451	0.19	-1.59,1.63,-10.12,-11.53	0.15	0.04	0.99
2	235,451	0.18	1.22,-11.57,-13.69	0.27	0.07	0.96
3	189,236,416	0.12	0.75,1.46,-5.63,-7.41	0.22	0.05	0.98
4	14,235,451	0.09	-0.87,6.44,-11.93,-10.62	0.17	0.05	0.98
5	113,235,451	0.07	1.77,2.66,-10.28,-6.24	0.16	0.05	0.99
6	14,416	0.04	0.39,5.45,-9.11	0.37	0.17	0.92
7	141,235,451	0.04	-2.21,-1.38,-7.43,-11.91	0.18	0.06	0.98
8	14,236,451	0.03	-0.29,11.04,-12.83,-8.74	0.19	0.04	0.98
9	235,1298	0.03	3.98,-15.77,-6.26	0.54	0.31	0.79
10	416	0.03	-0.77,-2.33	0.38	0.18	0.91
11	235,451,1298	0.02	0.89,-9.03,-18.34,-4.25	0.25	0.09	0.96
12	14,236,416,451	0.02	2.76,11.37,-7.41,-2.76,-7.46	0.18	0.05	0.98
13	236,416	0.02	0.6,-2.69,-5.15	0.30	0.12	0.95
14	141,235,1298	0.02	3.59,3.37,-14.79,-2.93	0.30	0.14	0.95
15	14,235,236,451	0.02	0.62,4.46,-3.11,-8.31,-7.79	0.16	0.04	0.99
16	141,235,451,1298	0.02	1.02,7.46,-15.97,-8.07,-2.99	0.15	0.06	0.99
17	14,236,416	0.01	3.6,4.97,-4.7,-6.6	0.22	0.07	0.97

Table 4.16: Cross-validated prediction table (on training cases) with top selected feature subsets from T-probit.

For each method, we fit the model with the training cases (n_1 = 100) and then make predictions on the test cases (n_2 = 1000). The results are shown in Table 4.17. More specifically, we obtain cross-validated feature subset selection results as in Table 4.16. The optimal feature subset (with respect to the performance measures AMLP and ER) is feature subset (119,235,451). With Tpro_optimal method we then use this optimal feature subset to make predictions on the test cases and record the corresponding prediction results. The same approach applies to Tpro_top, Tpro_average and all the other methods. From the prediction table, we can see that Tpro_average shows competitive results with Group LASSO and Bayesglm. LASSO and random forest have higher but still satisfactory error rate/AUC. Supervised Group LASSO does not work as well as Group LASSO. Bayesglm is working well with all 2000 features. The surprising part of this table is that Tpro_top and Tpro_optimal methods do not work well, with a rather high error rate (0.16). The reason is that both of them only use three features to make the predictions. For example, the optimal feature subset (119,235,451) includes only one feature from each group (Group 1 - Group 3). This is a succinct demonstration of representative feature selection from groups and shows that the excellent predictive performance of Group LASSO and Tpro_average is may be another kind of overfitting using redundant features.

	Tpro_top	**Tpro_optimal**	**Tpro_average**	LASSO	GL	SGLASSO	RF	Bayesglm	T
ER	0.17	0.17	0.15	0.15	0.10	0.20	0.14	0.10	0.18
AMLP	0.41	0.41	0.32	0.34	0.25	0.51	0.37	0.24	0.51
AUC	0.91	0.91	0.94	0.93	0.97	0.90	0.93	0.97	0.90
No. of features	**3**	**3**		21	983	60	149	1539	134

Table 4.17: Prediction results (ER, AMLP and AUC) and the number of features selected by each method. Tpro_top conducts feature selection and prediction with only the top feature subset selected by T-probit. Tpro_optimal conducts feature selection and prediction with the optimal (with respect to cross-validated training error) feature subset selected by T-probit. Tpro_average conducts feature selection and prediction results averaging over the Markov chains obtained with T-probit regression.

Method	fsubsets	AMLP	ER	AUC
T-probit top subset	119,235,451	0.41	0.17	0.91
T-probit optimal subset	119,235,451	0.41	0.17	0.91
LASSO	416,235,324	0.49	0.20	0.88
Group LASSO	1532,1407,1461	0.77	0.49	0.51
Supervised Group LASSO	1532,324,14	0.61	0.27	0.79
Random Forest	587,595,527	0.41	0.18	0.90
Bayesglm	1532,1298,1407	0.78	0.49	0.52
t-test	54,159,147	0.67	0.40	0.64

Table 4.18: Top feature subsets selected by each method and corresponding predictive performance (AMLP, ER, and AUC) on 1000 test cases.

To compare the efficiency of each method, we also fit other models with the same number of features T-probit is using (3 features). Table 4.18 shows that the Tpro_optimal method provided the most accurate predictive performance across all methods. Tpro_top is also better than any other methods when fitting with only 3 features. From the table we can see that T-probit is the only method that selects one feature from each of the three groups. LASSO detected the significance of Group 3; however, it misses Group 1 since it is less differential.

	Tpro_top	Tpro_optimal	LASSO	GL	SGLASSO	RF	Bayesglm	T
Group 1	1	1	3	154	5	1	158	0
Group 2	1	1	3	176	7	16	186	26
Group 3	1	1	4	170	9	120	180	82
Group 4	0	0	11	483	39	12	1015	26

Table 4.19: The number of features selected within each group for each method. Group 1, Group 2, and Group 3 are the groups with significant signals, and Group 4 is the group of noise features.

To directly look at the feature selection performance of each model, we collect the number of features selected from all 3 groups. For each method (other than Tpro_top and Tpro_optimal), the global feature selection decision is measured with the 0.1 criterion on coefficient scores. If one feature has an absolute coefficient β_j larger than $0.1 * \max_{p \in 1,\ldots,P} \beta_p$, it is then selected. Table 4.19 shows the group identities for all feature selection results. It is clear that Tpro_top and Tpro_optimal (actually the same feature subset as top, by chance, selects only one representative from each of groups). The features selected by Supervised Group LASSO and random forest are mainly from Group 3, which have relatively strong signals. The detailed feature selection results are shown in Figure D.11a to Figure D.14b.

We repeat the same process with 100 simulated datasets. The predictive performance (ER, AMLP, AUC) for each simulated dataset with each method is presented in Table 4.20. Group LASSO exhibits the best results, while Tpro_average and Bayesglm have the worst results. This verifies that Group LASSO always exhibits better performance when there is a clear grouping structure. The feature subsets selected by Tpro_top and Tpro_optimal have less accurate prediction results than Tpro_average and Group LASSO. However, this does not imply worse performance of T-probit; Table 4.21 shows that Tpro_optimal almost guarantees feature representative selection from each group from Group 1 to Group 3 while it filters most of the noise features in Group 4. The Tpro_top method also detects signals from Group 1 to Group 3, but gives fewer chances of identifying signals compared with Tpro_optimal method.

In a word, the T-probit methods (with top and optimal feature subsets) give clearer feature representative selections from Group 1, 2 and 3 while Tpro_average gives more accurate predictive performance.

	Tpro_top	Tpro_optimal	Tpro_average	LASSO	GL	SGLASSO	RF	Bayesglm	T
ER	0.18	0.17	0.12	0.14	0.10	0.16	0.15	0.10	0.19
AMLP	0.47	0.48	0.33	0.34	0.25	0.46	0.37	0.26	0.54
AUC	0.90	0.91	0.95	0.93	0.97	0.92	0.93	0.97	0.89

Table 4.20: Prediction results (ER, AMLP and AUC) from each method averaged over 100 simulation datasets.

	Tpro_top	Tpro_optimal	LASSO	GL	SGLASSO	RF	Bayesglm	T
Group 1	0.74	1.01	2.98	151.79	4.30	2.58	175.44	0.03
Group 2	0.81	1.09	3.16	149.41	5.29	4.63	177.12	49.28
Group 3	0.95	1.24	7.24	171.61	11.63	114.34	191.79	45.76
Group 4	0.13	0.47	9.40	237.77	21.61	2.85	1019.91	40.15

Table 4.21: The number of features selected within each group for each method averaging on 100 simulated datasets.

4.4 An example for Investigating the Choice of Degrees of Freedom of t priors

We generate a dataset with p=2000, n=1100 and implement T-probit regression with different choices of the heaviness index α_1. Among all 1100 cases, 100 cases are used to fit models and 1000 cases are used to evaluate model performance. The response vector $y = (y_1, ..., y_n)$ and the first 12 features are generated as follows:

$$P(y_i = c) = \frac{1}{2}, i = 1, 2, ..., n, c = 1, 2, \tag{4.4}$$

$$\mu_{1,1} = 0, \mu_{2,1} = 2,$$

$$x_{i1} = \mu_{y_i,1} + z_{i1} + \epsilon_{i1}, i = 1, ..., n, \textbf{(Group 1)}$$

$$x_{i2} = 5z_{i1} + z_{i2} + 0.5\epsilon_{i2}, i = 1, ..., n, \textbf{(Group 2)}$$

$$\mu_{1,3} = 0, \mu_{2,3} = 1,$$

$$x_{ij} = \mu_{y_i,3} + z_{i3} + 0.5\epsilon_{i3}, j = 3, ..., 12, i = 1, ..., n, \textbf{(Group 3)}$$

$$x_{ij} \sim N(0, 1), j = 13, ..., 2000, i = 1, ..., n, \textbf{(Group 4)}$$

$$z_{ij} \sim N(0, 1), i = 1, 2, ..., n, j = 1, 2, 3,$$

$$\epsilon_{ij} \sim N(0, 1), i = 1, 2, ..., n, j = 1, 2, 3,$$

where the centroids matrix ($\mu_{y_i,1:12}$) for the features in Group 1 to Group 3 can also be written as

$$\mu_{c,1:12} = \begin{pmatrix} 2 & 0 & 1 & \cdots & 1 \\ 0 & 0 & 0 & \cdots & 0 \end{pmatrix},$$

z_{i1} is the common factor of x_{i1} and x_{i2}. z_{i2} is the unique component of x_{i2}, and z_{i3} is the common factor of features x_3, x_4,..., x_{12}. All other feature values $x_j, j = 13, ..., 2000$ are random noise.

The total data structure is shown in Figure 4.8. From this figure, we can see that all features in Group 1, Group 2 and Group 3 are relevant to the response vector y for this dataset. Features in Group 2 have mean values equal to 0, but they are strongly correlated ($\rho = 0.69$) with features in Group 1 and then they are all relevant to the response vector y. Group 3 features are relevant features sharing the same information (z_{i3}) with a strong correlation ($\rho = 0.8$). The signal of features in Group 3 is relatively weak ($\mu_{1,3} = 0, \mu_{2,3} = 1$) compared to Group 1 ($\mu_{1,1} = 0, \mu_{2,1} = 2$). The motivation of this study is to run T-probit regressions (with different priors) on this dataset, and evaluate feature selection results of different methods between groups (Group 1 and Group 3) and within Group 3.

We first implement T-probit regression on the training cases. The parameters are set as α_0=1, ω_0=0.5, R_1=2000, n_1=100, R_2=3000, n_2=100, l_1=5, l_2=50, ϵ=0.5, η=0.04. α_0 and $\sqrt{\omega_0}$ are the shape and scale parameters for the model, while α_1 and $\sqrt{\omega_1}$ are the shape and scale parameters for the coefficients β. To test the sensitivity of different priors, we take values of α_1 from (0.2,0.5,1,5,10) and ω_1 from $(\exp(-5), \exp(-10), \exp(-20))$. Each

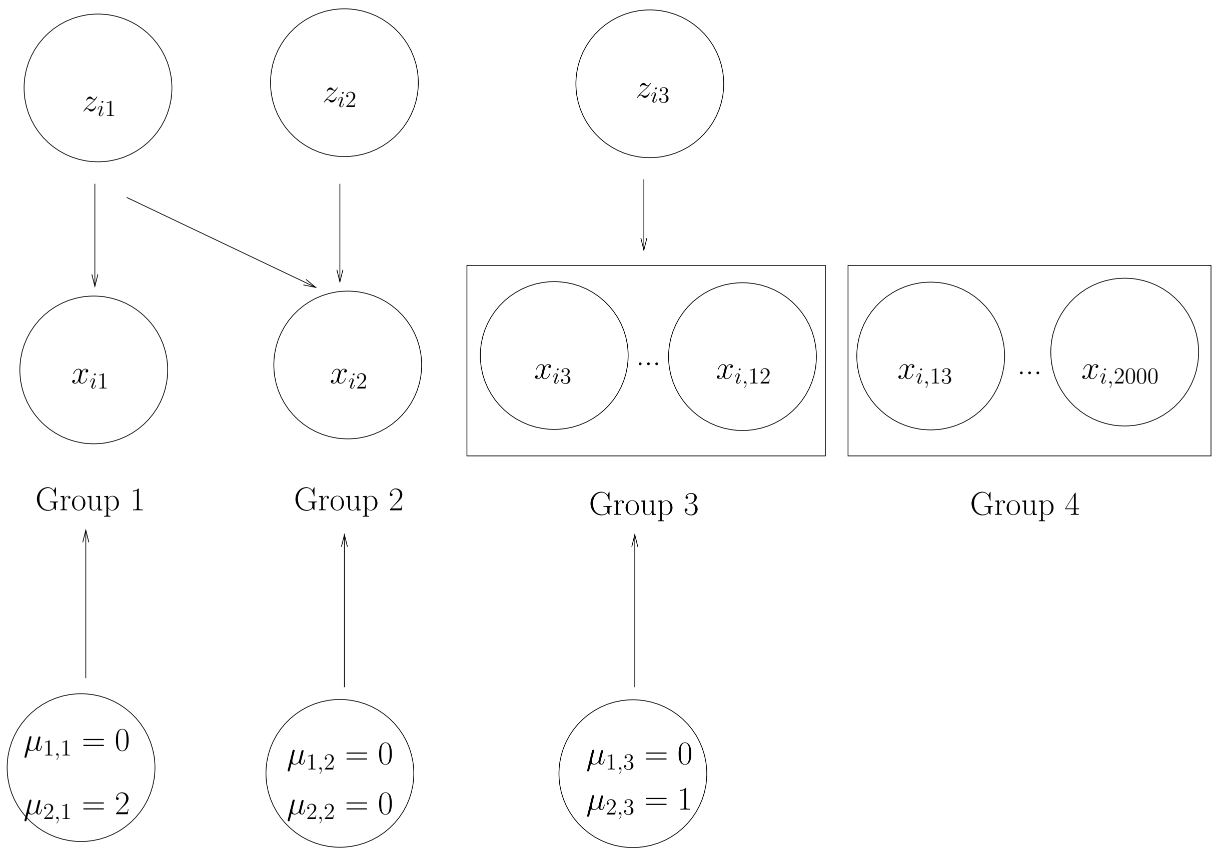


Figure 4.8: Data structure demonstration. x_{i1} and x_{i2} are the first and second feature values from the ith observation. $x_{i3}, ..., x_{i,12}$ are the feature values in the third group from the ith observation. $x_{i,13}, ..., x_{i,2000}$ are the feature values in the fourth group from the ith observation. z_{i1}, z_{i2}, z_{i3} are three individual values. $\mu_{y_i,j}.j = 1, 2, 3$ are sample mean values of features from group j, where y_i is the response value of the ith observation.

configuration (out of 15 different configurations) is tried with different criteria η to make sure the number of features updated within each Markov chain sample is around 2.5% for computational convenience (the coefficients of about 50 features are updated each round among all 2000 features). Other methods including LASSO, Group LASSO, Supervised Group LASSO, random forest, Penalized Logistic Regression and simple Student's t-test are also implemented. The Group LASSO and Supervised Group LASSO are implemented with R package `gglasso` using the original group structure.

	fsubsets	freqs	cv_AMLP	cv_ER	cv_AUC	coefs (w/int)
1	1,2,8	0.27	0.37	0.17	0.91	-1.88,6.13,-0.86,6. 01
2	1,2,11	0.09	0.38	0.18	0.91	0.48,4.06,-1.93, 1.91
3	1,8	0.07	0.47	0.26	0.84	-1.35,10.93,-6.11
4	1,2,7	0.06	0.39	0.17	0.90	-0.74,6.64,-3.56, 4.38
5	1,2,4	0.05	0.39	0.18	0.91	-0.97,6.9,-2.53,4 .26
6	1,2	0.05	0.46	0.19	0.87	-0.63,7.11,3.2
7	1,2,3	0.04	0.39	0.17	0.91	-1.7,5.66,-4.39,1 .75
8	1,2,8,621	0.02	0.32	0.12	0.94	-0.07,4.75,-3 .12,5.45,-0.55
9	1,2,8,210	0.02	0.34	0.15	0.93	-4.45,19.68,- 11.26,11.37,-7.32
10	1,11	0.02	0.47	0.20	0.86	0.67,6.94,5.63
11	1,2,8,15	0.02	0.35	0.15	0.93	-0.34,9.43,-5 .54,5.64,-1.59
12	1,8,48	0.02	0.41	0.22	0.89	-0.5,4.16,4.9,- 4.61
13	1,7,483	0.01	0.43	0.20	0.88	0.31,7.84,8.66 ,5.27
14	1,2,7,483	0.01	0.34	0.15	0.92	-0.52,14.78, -7.45,8.36,2.53
15	1,8,15	0.01	0.42	0.19	0.89	-1.9,19.94,19.1 8,-11.18
16	1,2,4,210	0.01	0.34	0.13	0.92	-0.38,5.97,- 3.61,2.85,-2.42
17	1,2,621	0.01	0.46	0.23	0.86	-0.64,13.28,-8 .32,-2.21

Table 4.22: Cross-validated prediction results with top selected feature subsets (from the entire dataset with T-probit). The prediction accuracy values (ER/AMLP/AUC) are obtained with cross-validation using Bayesglm based on all of the data.

We show the feature subset selection table of T-probit regression with default parameter settings α_1=1, $\log(\omega_1)$= -10. The corresponding predictive power (with cross-validation) and coefficients of all selected feature subsets are shown in Table 4.22. From the table we can see that the feature subset (1,2,8) has the highest frequency 27%, while T-probit also provides alternative feature subsets such as (1,2,3), (1,2,4), (1,2,7) and (1,2,11).

Table 4.23 shows the prediction results (using test cases) of the top 3 feature subsets from all methods. For the other methods, the top 3 feature subsets are the subsets containing the top 3 features with the largest coefficients. The results show that all methods except T-

Method	fsubsets	AMLP	NMC	AUC
T-probit Optimal Subset	1,2,11	0.33	0.14	0.93
T-probit Top Subset	1,2,8	0.35	0.15	0.92
LASSO	1,8,483	0.48	0.24	0.85
Group LASSO	1,383,483	0.64	0.34	0.73
Supervised Group LASSO	1,8,1568	0.47	0.23	0.86
Random Forest	1,4,12	0.42	0.20	0.89
Bayesglm	1,8,48	0.52	0.25	0.84
t-test	1,26,66	0.52	0.26	0.82

Table 4.23: Prediction results on test cases with selected top 3 feature subsets (with T-probit, LASSO, Group LASSO, Supervised Group LASSO, random forest, Bayesglm, simple *t*-test). The prediction values (AMLP/NMC/AUC) are obtained with test cases (with each selected feature subset) using Bayesglm.

probit and random forest will select irrelevant features. In conclusion, the T-probit with $df = 1$ and random forest avoid all irrelevant features. Group LASSO and Supervised Group LASSO select many irrelevant features. LASSO selection is rather sparse (though not as much as T-probit); however, it ignores x_2. Bayesglm also selects many irrelevant features. In conclusion, the T-probit shows the best feature screening performance among all methods for this specific case.

We also show the coefficients score of different methods in Figure D.1 to Figure D.6. With different parameter settings, we fit T-probit regression and obtain Markov chain samples of the coefficients β. The median values of each feature coefficient across the Markov chains is then recorded as the feature score for T-probit regressions. For all other methods, the feature score (Figure D.3 to Figure D.6) is the corresponding coefficient vector of different methods. The results show that T-probit regression with α=1 and log(w)= -10 has great feature selection performance. We highlight the following observations. 1) T-probit identifies feature 2 (x_2), which is ignored by LASSO and random forest. Both Group LASSO and Supervised Group LASSO successfully identify feature 2. 2) T-probit selects representatives (feature 8) from Group 3. 3) T-probit ignores all other irrelevant features. 4) LASSO selects features 1, 8, 48, 383 and 483, which means LASSO identifies representatives from Group 1 and Group 2 while still including two irrelevant features and is missing a representative from Group 3. 5) Group LASSO only selects features from the first 700 features and it has the over-selection problem. 6) Supervised Group LASSO identifies feature 1, 2 and 8 while including dozens of irrelevant feature selections. 6) Random forest and Bayesglm both have severe over-selection problems. In conclusion, the T-probit regression is the only method that selects representatives from all 3 groups and ignores all irrelevant features.

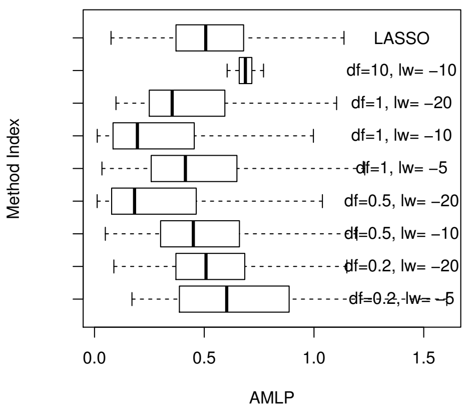
The results of T-probit regression with different parameter settings show how prior information affects feature selection. The T-probit regression with α=1 has optimal feature selection results (and better than any other models) across two different choices of log(w): $\exp(-10)$ and $\exp(-20)$. With larger α values such as (5,10), T-probit underperforms LASSO by selecting more irrelevant features. With a smaller scale (α=(0.2,0.5)), the model adopts heavy tails and results in feature selection with much a larger magnitude because the flat tails of the prior distribution will allow coefficients to diverge to large values. Therefore, we recommend $\alpha = 1$ as the default value for all experiments with T-probit regression.

The predictive performance of all methods is given in Table 4.24. From this table, we see that the predictive performance of the sparse feature selection results (from T-probit and LASSO) are much better than the other methods, which select many irrelevant features. The T-probit also outperforms LASSO because it incorporates feature 2 (x_2) (the correlated but non-differential feature) into the model. To look further into the consistency of the predictive performance, we also show the boxplots of the negative log-probability on each test case for all methods in Figure 4.9. With α=1, T-probit has significantly lower AMLP than LASSO, and only the extreme values of hyper-priors (α=10, log(w)=0.2) lead to a T-probit with bad performance. All other methods underperform both T-probit and LASSO. Bayesglm has a very high AMLP because Bayesglm makes “extremely” wrong predictions on some difficult cases.

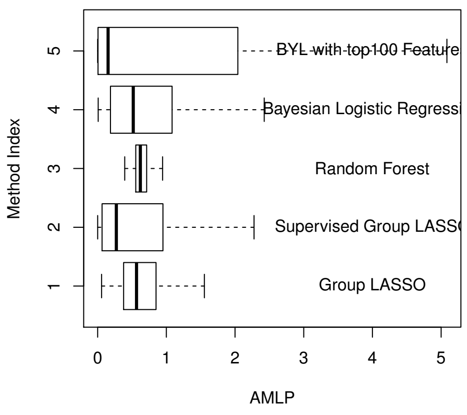
In conclusion, for these simulated datasets, T-probit with hyper-prior α=1 has superior feature selection and predictive performance compared to other methods. With proper parameter settings, T-probit regression is able to provide a feature subset table with multiple selected feature subsets. The T-probit method also has better predictive performance (with respect to ER/AMLP/AUC) compared with LASSO and other methods. However, inappropriate choices of α and ω in T-probit may result in very poor performance. In practice, we recommend default parameter settings (α=1, log(w)=-10) to implement T-probit regression.

	Tpro_top	Tpro_optimal	Tpro_average	LASSO	GL	SGLASSO	RF	Bayesglm	T
No. of features	**3**	**4**		**5**	**451**	**23**	**73**	**1238**	**55**
ER	**0.14**	0.16	0.14	0.23	0.38	0.32	0.29	0.40	0.37
AMLP	**0.33**	0.35	0.37	0.54	0.65	0.68	0.64	0.75	1.54
AUC	**0.93**	0.93	0.93	0.86	0.68	0.76	0.76	0.65	0.68

Table 4.24: Prediction (ER, AMLP and AUC) results and the number of features selected from all methods. Tpro_top conducts feature selection and prediction with only the top feature subset selected with T-probit regression. Tpro_optimal conducts feature selection and prediction with the optimal (with respect to cross-validated training error) feature subset selected with T-probit regression. Tpro_average conducts feature selection and prediction results averaging over the Markov chain obtained with T-probit regression.



(a) AMLP Boxplots of T-probit regressions.



(b) AMLP Boxplots for other methods.

Figure 4.9: AMLP Boxplots of T-probit, LASSO, Group LASSO, Supervised Group LASSO, random forest, and Bayesglm.

4.5 Investigation of Computational Efficiency of T-probit Methods

The aim of this section is to assess the computational efficiency of the HMC approach (with leapfrog method) we used to implement the T-probit model. In HMC, the Hamiltonian dynamics cannot be simulated exactly because of the problem of time discretization, and the leap-frog method is an alternative solution to preserve the volume in 3 steps:

$$\hat{M}_j(\tau + \frac{\epsilon}{2}) = \hat{M}_j(\tau) - \frac{\epsilon}{2}\frac{\partial E}{\partial \beta_j}(\hat{\beta}(\tau)), \tag{4.5}$$

$$\hat{\beta}_j(\tau + \epsilon) = \hat{\beta}_j(\tau) + \epsilon \hat{\beta}_j(\tau + \frac{\epsilon}{2}), \tag{4.6}$$

$$\hat{M}_j(\tau + \epsilon) = \hat{M}_j(\tau + \frac{\epsilon}{2}) - \frac{\epsilon}{2}\frac{\partial E}{\partial \beta_j}(\hat{\beta}(\tau + \epsilon)). \tag{4.7}$$

These 3 steps are repeated L times with stepsize ϵ. The number of leapfrog steps L is a parameter that demands careful tuning. HMC with $L = 1$ reduces to Langevin Monte Carlo (LMC), also known as the Metropolis-Adjusted Langevin Algorithm (MALA) (Rossky et al., 1978). Specific suggestions for tuning ϵ and L can be found in Neal (2011). The primary measurement to tune these parameters is the acceptance rate. If ϵ and L are too small, the MCMC acceptance rate may be too high (rejection rate too low) and the Markov chain will waste time in a small neighborhood. On the contrary, with large ϵ and L the MCMC samples may have a rather high rejection rate. Fortunately, the tuning of L and ϵ can be independent and a satisfactory acceptance rate is recommended to be between 60% to 80% for HMC (Neal, 2011). According to our experience, $L = 50$ is reasonable in most cases.

To evaluate the computational efficiency of our exact approach, we compare three different methods: Leapfrog with $L = 1$, Leapfrog with $L = 50$, and naive Gibbs sampling. Gibbs sampling is implemented with JAGS (Just Another Gibbs Sampler) (Plummer et al., 2003). The BUGS project (Lunn et al., 2009) is designed for MCMC sampling of complex statistical models using Gibbs sampling. Since JAGS needs an object representing a Bayesian graphical model, we describe the T-probit model in the JAGS dialect of the BUGS language in the form of auxiliary variables:

$$y_i = \begin{cases} 1, & \text{if} \quad z_i > 0 \\ 0, & \text{if} \quad z_i < 0 \end{cases} \tag{4.8}$$

$$\begin{aligned} z_i &= x_{i.}\beta + \epsilon_i, \\ \epsilon_i &\sim N(0, \sigma_i^2), \\ \sigma_i^2 &\overset{iid}{\sim} \text{Inverse-Gamma}(\alpha_0/2, \alpha_1\omega_0/2), \\ \beta_j &\sim N(0, \lambda_j^2), \\ \lambda_j^2 &\overset{iid}{\sim} \text{Inverse-Gamma}(\alpha_1/2, \alpha_1\omega_1/2), \end{aligned} \tag{4.9}$$

Thus we trace the Markov chains of parameters β_j, λ_j, σ_i, ϵ_i.

The main focus of computational efficiency is the Markov chain convergence speed. With a specific MCMC sampling techniques application, we expect our chains to eventually converge to our target stationary distribution; however, there is no guarantee that the Markov chain obtained really converged. To measure the quality of Markov chain samples, we mainly focus on two different aspect: the convergence time of the Markov chain and the switching frequency between the converged values. More specifically, we expect the Markov chain to explore the posterior fully and reflect the whole picture of the target distribution. However, there is no way to tell if the MC samples really reflect the true stationary distribution if the target distribution is rather complicated or even unknown to users.

An alternative way to interpret the MC mixing speed is to determine how well the Markov chain will propose a new sample independent from the current state. If ideally the MC can propose states independently from the stationary distribution, then given enough time the MC will have enough samples from all possible values in the sample space to reflect the desired posterior distribution. Then the only question left is if we have enough computational resources to run the Markov chain long enough. This aspect of mixing speed can be visualized in terms of a trace plot (density plot) of Markov chain samples, running means plot (running magnitude plot), autocorrelations, etc. Analytical diagnosis can also be made based on the rejection rate, the Gelman and Rubin multiple sequence diagnostic (Gelman and Rubin, 1992), the Geweke Diagnostic (Geweke et al., 1991), the Raftery and Lewis Diagnostic (Raftery and Lewis, 1992), etc. In this thesis, we performed Markov chain diagnosis using the traceplots and autocorrelations.

Another important aspect of Markov chain mixing speed is the switching speed of the Markov chain between multiple modes of posterior distribution, if they exist. The switching frequency between modes is just a natural realization of the MC traveling speed. If MC successfully visits the whole sample space with respect to the invariant distribution, the switching frequency will reflect the real probability densities. However, even though the Markov chain proposes states independently from previous states, it is still unknown if the Markov chain is long enough to cover all states. On the other hand, even if the Markov chain successfully explores all states, we expect the Markov chain will reflect the real probability density function in terms of sample frequency counts. For T-probit, this is especially important since we are interested in exploring them and interpreting them as feature subsets containing small numbers of features.

We compare three different methods on a simulated dataset. The dataset is generated with grouping structure. More specifically, we generate multivariate normal data with three variables. The first two are correlated and both weakly differentiated (with centroid (-0.3,0.3) to (0,0) across clusters), while the third feature is independent of the first two but more differentiated (with centroid (-1,1) to (0,0) across clusters). More specifically, we generate a dataset X as follows:

$$P(y_i = c) = \frac{1}{2}, i = 1, 2, ..., n, c = 1, 2, \tag{4.10}$$

$$\mu_{1,1} = -0.3, \mu_{2,1} = 0.3,$$

$$z_{ij} \sim N(0, 1), i = 1, ..., n, j = 1, 2, 3,$$

$$\epsilon_{ij} \sim N(0, 1), i = 1, ..., n, j = 1, ..., 30,$$

$$x_{ij} = \mu_{y_i,1} + z_{i1} + 0.5\epsilon_{ij}, i = 1, ..., n, j = 1, ..., 10, \textbf{(Group 1)}$$

$$\mu_{1,2} = 0.3, \mu_{2,2} = -0.3,$$

$$x_{ij} = \mu_{y_i,2} + 0.8z_{i1} + 0.6z_{i2} + 0.5\epsilon_{ij}, i = 1, ..., n, j = 11, ..., 20, \textbf{(Group 2)}$$

$$\mu_{1,3} = 1, \mu_{2,3} = -1,$$

$$x_{ij} = \mu_{y_i,3} + z_{i3} + 0.5\epsilon_{ij}, i = 1, ..., n, j = 21, ..., 30, \textbf{(Group 3)}$$

$$x_{ij} \sim N(0, 1), i = 1, ..., n, j = 31, ..., 100, \textbf{(Group 4)}$$

where n is the total number of cases ($n = 100$) and the centroid matrix ($\mu_{c,1:3}$) for group 1 to group 3 features can also be written as

$$\mu_{c,1:3} = \begin{pmatrix} -0.3 & 0.3 & 1 \\ 0.3 & -0.3 & -1 \end{pmatrix}$$

The data generated in this way has three significant groups, each containing 10 features. All other features (70) are noise. The data structure is also shown in Figure 4.10.

We run T-probit regression on the training cases with parameter settings α_0=1, ω_0=0.5, α_1=1, ω_1=exp(-10), R_1 =5000, n_1=1000, R_2=6000, n_2=2000, ϵ=0.5, η=0.05. To test the efficiency of the leapfrog methods, we also include two different approaches: $l_1 = 5$, $l_2 = 50$ and $l_1 = 1$, $l_2 = 1$. The latter breaks down to Langevin Monte Carlo (LMC). The naive Gibbs sampling method is also tested on JAGS and the corresponding results are recorded as T-probit_jags. After obtaining these three Markov chains, we reorganize them in terms of the computational time. More specifically, for each Markov chain ($m = (1, 2, 3)$) obtained, we have the number of samples $I(m)$ and its computational time $T(m)$. Then, for an arbitrary number of new Markov chain (n=1000 for example), we shrink the three Markov chains with respect to the minimal computational time $T_M = \min T(m), m = 1, 2, 3$. A new Markov chain M is constructed based on the old one. For the jth sample on the new Markov chain M, its recording time is $j \times T_M/n$, which corresponds to the sample (on old Markov chain

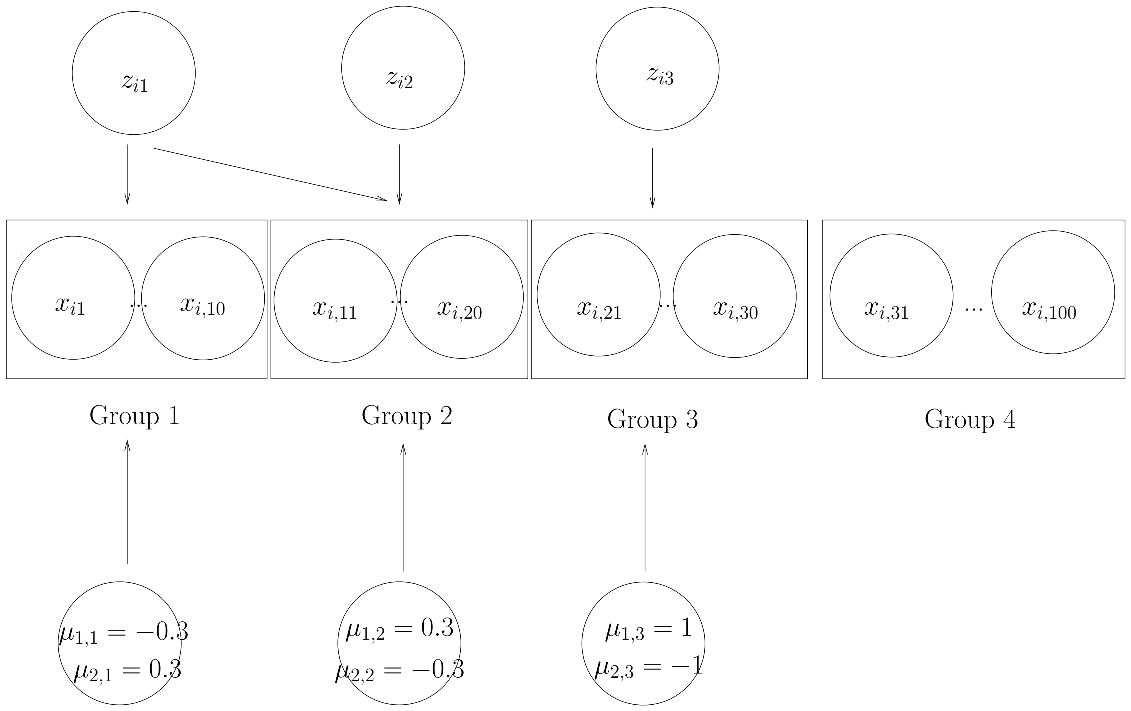


Figure 4.10: Data structure demonstration. $x_{i1}, ..., x_{i,10}$ are the feature values (in the first group) from the ith observation. $x_{i,11}, ..., x_{i,20}$ are the feature values (in the second group) from the ith observation. $x_{i,21}, ..., x_{i,30}$ are the feature values (in the third group) from the ith observation. $x_{i,31}, ..., x_{i,100}$ are the feature values (in the fourth group) from the ith observation. z_{i1}, z_{i2}, z_{i3} are three individual values. $\mu_{y_i,j}, j = 1, 2, 3$ are the sample mean values of features from Group j, where y_i is the response value of the ith observation.

$m)$ $j\frac{T_M I(m)}{T_m n}$. This number may not be an integer; if this is the case, we then use the rounded number as the exact sample we record on the new Markov chain. After this transformation, we obtain three new Markov chains M, which have the same number of Markov chain samples (1000) and same consuming time (16835 seconds). Each sample from the three Markov chains are synchronized at approximately the same moment.

Figure 4.11, Figure 4.12, and Figure 4.13 show the coefficients of features 21 and 28 with corresponding auto-correlations using different approaches. Clearly, T-probit_jags has much stronger auto-correlations, which means the mixing speed of T-probit_jags is slower than LMC and HMC. It should also be noted that T-probit_jags selects feature 23 within the first 100 samples, while it switches to feature 21 after that. In other words, the T-probit_jags switches from a feature subset that includes feature 21 to a feature subset that includes feature 23 only once and stops switching after that, while LMC and HMC frequently switch between modes.

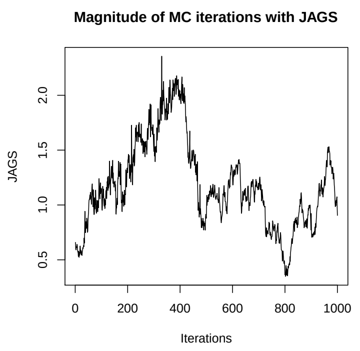
	Computational Time (s)	CT	**NMS**	Number of Modes	Features
JAGS	16835.206	16835.206	**28**	9	1,11,21,28,29,88
LMC	55975.444	16835.206	**798**	79	1,4,8,9,11,12,16,21,23,28
HMC	55644.778	16835.206	**899**	70	1,4,8,9,11,12,16,21,28

Table 4.25: CT: Computation time that are used from each Markov chain. NMS: Numbers of iterations in which mode switching happens. Features: Features selected (across all modes).

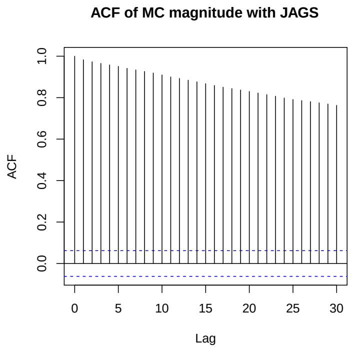
	fsubsets	freqs	coefs (w/int)
1	1,11,21	0.42	1.21,4.32,-4.86,-3.68
2	1,11,28	0.41	1.52,6.75,-8.2,-2.21
3	1,11,21,88	0.05	0.05,8.41,-10.17,-5.94,-1.07
4	1,11,29	0.05	-0.13,4.01,-3.7,-0.56
5	1,11,28,29	0.03	1.16,4.13,-4.51,-3.24,-0.76
6	1,11	0.03	0.71,4,-4.49
7	1,11,21,28	0.01	1.44,5.91,-7.4,-3.64,-1.13

Table 4.26: Feature subsets selected by T-probit_jags.

Table 4.26, Table 4.27 and Table 4.28 are the feature subset selection results extracted from the three Markov chains. The T-probit_jags (ordinary Gibbs sampling) only detected 7 different modes while LMC and HMC detect more feature subsets. This indicates that

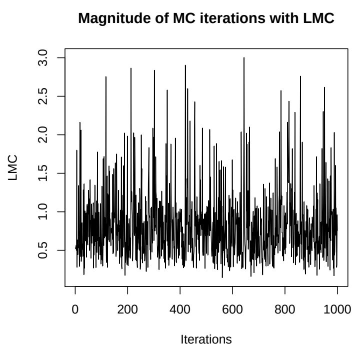


(a) Magnitude of Markov chain samples with T-probit_jags.

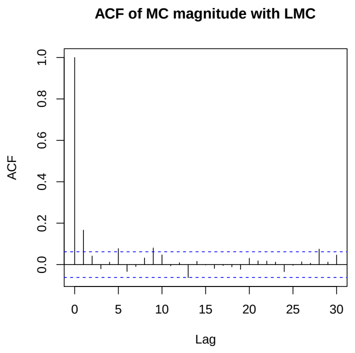


(b) Autocorrelation in Markov chain samples with T-probit_jags.

Figure 4.11: Magnitude and autocorrelation in Markov chains with T-probit_jags.

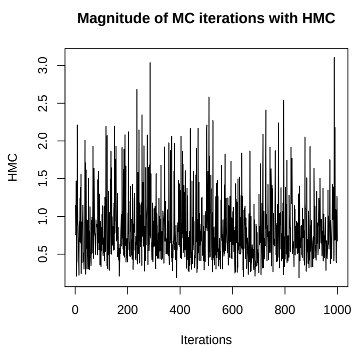


(a) Magnitude of Markov chain samples with LMC.

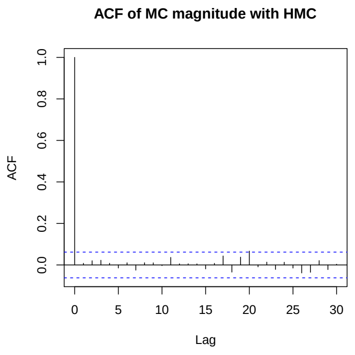


(b) Autocorrelation in Markov chain samples with LMC.

Figure 4.12: Magnitude and autocorrelation in Markov chains with LMC.

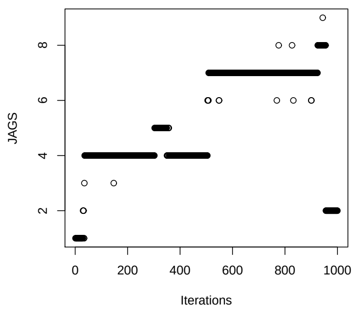


(a) Magnitude of Markov chain samples with HMC.

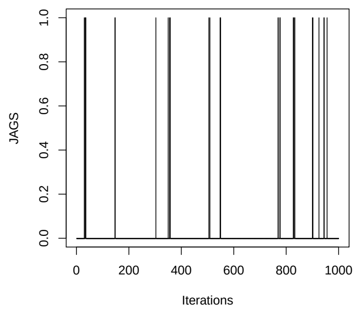


(b) Autocorrelation in Markov chain samples with HMC.

Figure 4.13: Magnitude and autocorrelation in Markov chains with HMC.

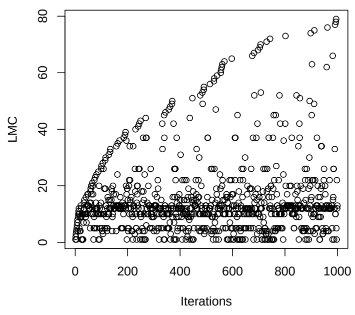


(a) Changing cluster ID with T-probit_jags.

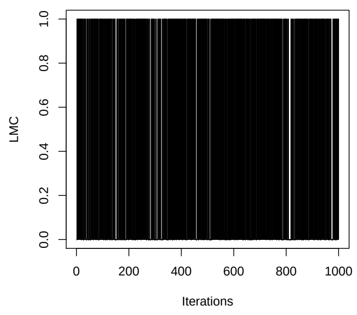


(b) Mode switching status with T-probit_jags. 1: change, 0: no change.

Figure 4.14: Traceplot of mode switching status with T-probit_jags.

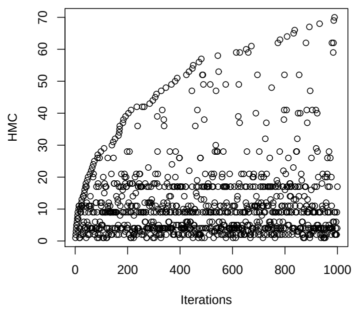


(a) Changing cluster ID with LMC.

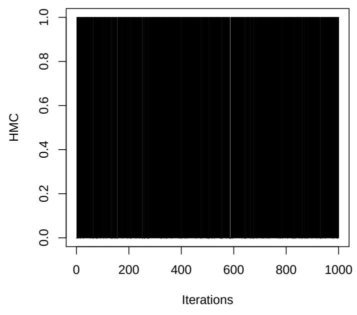


(b) Mode switching status with LMC. 1: change, 0: no change.

Figure 4.15: Traceplot of mode switching status with LMC.



(a) Changing cluster ID with HMC.



(b) Mode switching status with HMC. 1: change, 0: no change.

Figure 4.16: Traceplot of mode switching status with HMC.

	fsubsets	freqs	coefs (w/int)
1	1,11,21	0.20	-0.12,4.93,-7.43,-4.63
2	4,11,21	0.14	-0.57,1.76,-3.23,-4.02
3	1,11,28	0.11	-0.35,6.19,-8.18,-4.37
4	8,11,21	0.09	-1.53,3.11,-3.85,-5.79
5	21	0.05	-0.98,-3.81
6	16,21	0.04	-0.58,-1.88,-7.93
7	11,21	0.03	-1.41,-3,-7.54
8	4,16,21	0.03	-0.17,3.66,-6.15,-9.88
9	9,11,21	0.03	-0.6,1.92,-2.89,-4.36
10	1,11	0.03	0.07,4.2,-4.95
11	1,11,23	0.02	-0.83,2.88,-3.79,-1.83
12	8,16,21	0.02	-1.68,5.61,-7.49,-14.48
13	4,21	0.02	-0.25,1.16,-4.44
14	12,21	0.02	-1.73,-1.49,-8.28
15	1,16,21	0.01	-1.28,1.98,-2.89,-5.1
16	8,12,21	0.01	-1.06,1.69,-2.93,-6.09
17	4,12,21	0.01	-2.16,3.23,-4.42,-5.51
18	8,21	0.01	-0.65,0.23,-2.18

Table 4.27: Feature subsets selected by LMC.

	fsubsets	freqs	coefs (w/int)
1	1,11,21	0.17	-1.05,6.44,-10.56,-6.61
2	4,11,21	0.15	-0.78,2.42,-4.32,-4.99
3	8,11,21	0.10	-0.13,1.44,-1.84,-4
4	1,11,28	0.10	-0.53,6.89,-8.63,-3.12
5	21	0.05	-0.48,-3.15
6	16,21	0.04	-1.2,-0.9,-7.16
7	11,21	0.03	-2.27,-8.49,-7.41
8	8,16,21	0.03	-2.61,3.23,-6.62,-10.92
9	4,16,21	0.03	0.3,1.8,-3.12,-7.83
10	1,11	0.03	0.1,2.65,-3.68
11	9,11,21	0.02	-1.38,3.21,-4.14,-3.21
12	4,21	0.02	-1.18,2.6,-8.19
13	8,21	0.02	-0.28,0.53,-2.41
14	8,12,21	0.02	-0.29,1.69,-1.43,-8.24
15	12,21	0.02	-1.31,-0.6,-5.79
16	9,16,21	0.01	-1.21,2.54,-6.67,-9.17
17	1,16,21	0.01	-0.37,3.61,-7.17,-10.68
18	4,12,21	0.01	-0.09,1.16,-1.67,-2.09
19	1,12,21	0.01	-0.48,2.67,-4.16,-8.92
20	1,11,23	0.01	-0.74,3.18,-3.75,-2.62

Table 4.28: Feature subsets selected by HMC.

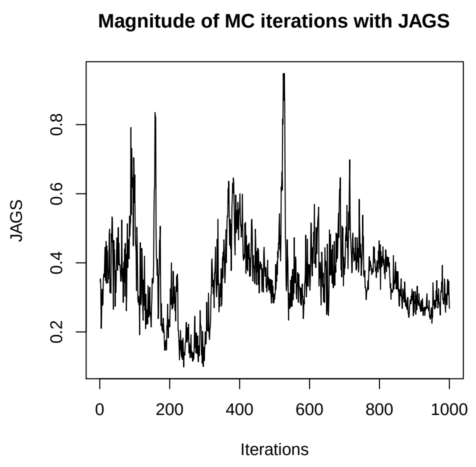
LMC and HMC can switch between modes more frequently. For each sample of new Markov chains, we label its feature subset index and draw the changing plots. Figure 4.16 shows the cluster ID change across 1000 transitions (or 16835 seconds). We use the changing index (CI) to denote whether the Markov chain switches to a different feature subset or not. CI = 1 means the Markov chain switches to a different feature subset while 0 means staying in the same subset. Figure 4.14 and Figure 4.15 show the changing index of LMC and T-probit_jags (naive Gibbs Sampling method). From these plots, we can tell HMC is frequently switching between multiple modes while T-probit_jags stays in the same feature subset most of the time. Table 4.25 shows the quantitative transition information. From the table, we can see that the mode switching rates for different methods are: T-probit_jags 28/1000, LMC 798/1000, HMC 899/1000. Apparently HMC has the highest switching frequency.

We also run experiments with p=1000 to examine the efficiency of the three methods in high-dimensional problems. Table 4.29 shows the general Markov chain information for all three methods. For each of them, the experiment is run for over 10 hours and the switching rates are 16.9%, 47.1% and 92.8% for T-probit_jags, LMC and HMC, respectively. It is worth noting that HMC has almost double the mode switching frequency compared with LMC.

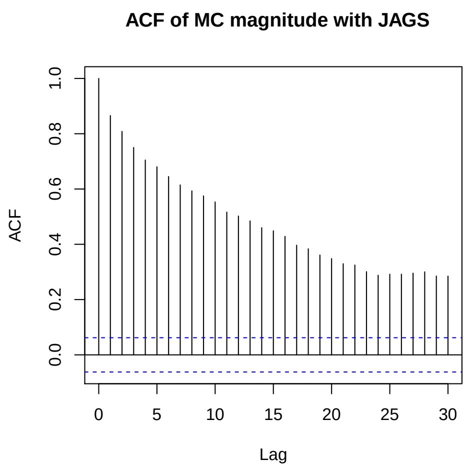
	Total Time	CT	**Switching Modes**	Modes	Features
JAGS	42230.661	42230.661	169	52	42,65,86,105,124,156,201,237,266,367
LMC	52191.94	42230.661	471	57	65,111,153,156,201,237,277,291,658
HMC	59118.072	42230.661	928	72	65,111,153,156,201,237,291,296,297,658

Table 4.29: CT: Computation time that are used for each method. Switching Modes: The number of iterations in which mode switching happens. Modes: The number of modes across Markov chains. Features: Features appeared in all modes.

To examine the convergence of the Markov chains, we plot the magnitude and its autocorrelation for all three Markov chains. Figure 4.17 shows that the T-probit_jags method leads to a Markov chain with poorer convergence speed compared with LMC and HMC. HMC have lower autocorrelation compared with LMC. Another important aspect of Markov chain mixing speed is the switching frequency between different modes, which is shown in Figure 4.20, Figure 4.21 and Figure 4.22. The first plot is the cluster ID for each sample in the Markov chains and the second plot is the switching status. If the Markov chain jumped to a new state of cluster ID (no matter how close this new state is to the previous state), we will then record this node as “switched” and mark it as 1. However, this measurement is not accurate since MC may only travels between two close states and still leaves a dense color. But combined with the traceplots it still implies a frequently mode switching Markov chain using a plot with dense color. Both plots show that HMC outperformed T-probit_jags and LMC, with higher mode switching frequency.

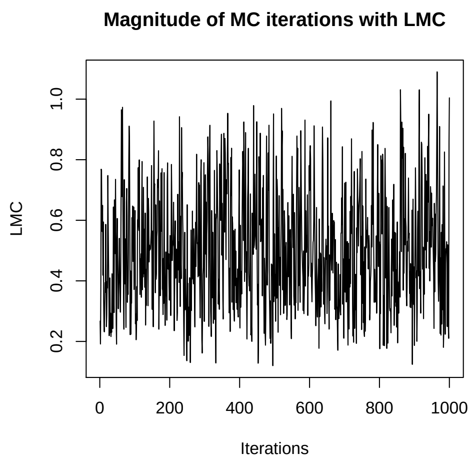


(a) Magnitude of Markov chain samples with T-probit_jags.

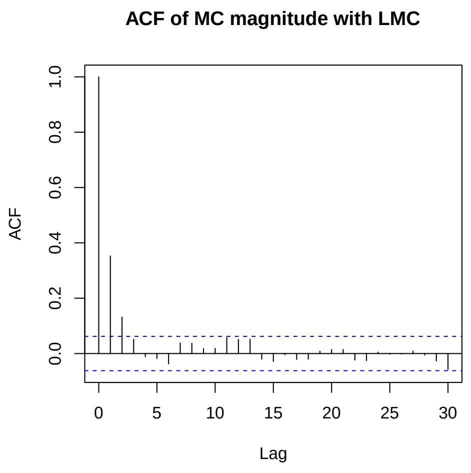


(b) Autocorrelation in Markov chain samples with T-probit_jags.

Figure 4.17: Magnitude and autocorrelation in Markov chains with T-probit_jags.

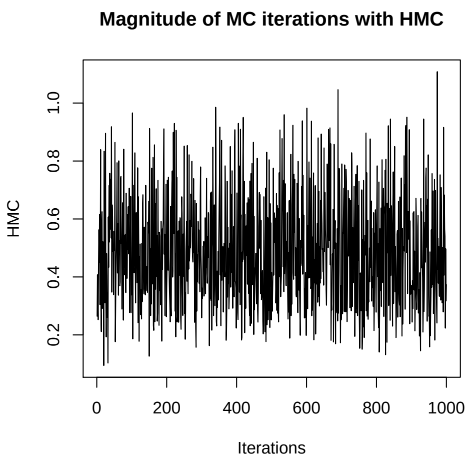


(a) Magnitude of MC samples with LMC.

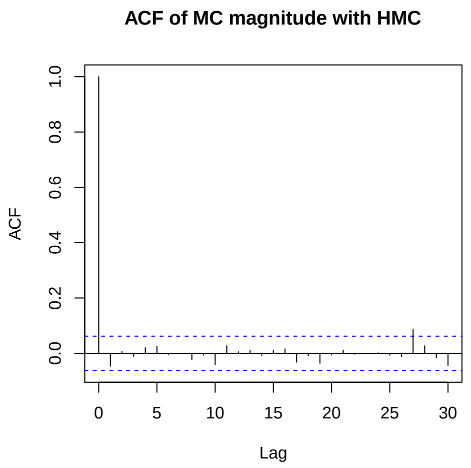


(b) Autocorrelation in Markov chain samples with LMC.

Figure 4.18: Magnitude and autocorrelation in Markov chains with LMC.

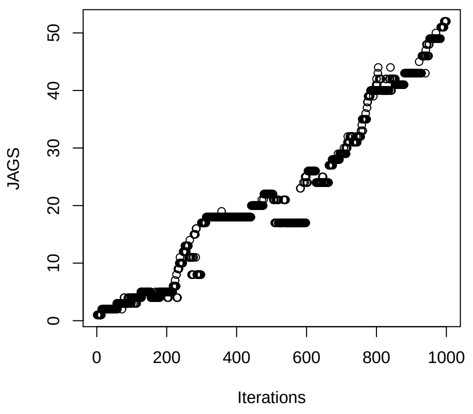


(a) Magnitude of MC samples with HMC.

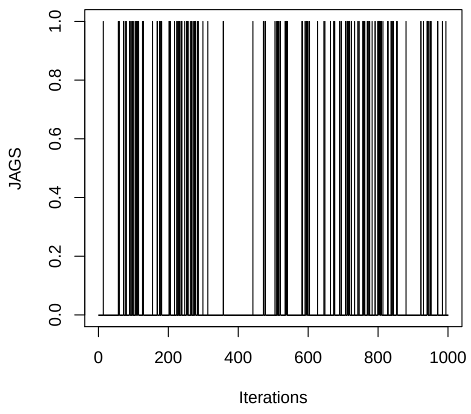


(b) Autocorrelation in Markov chain samples with HMC.

Figure 4.19: Magnitude and autocorrelation in Markov chains with HMC.

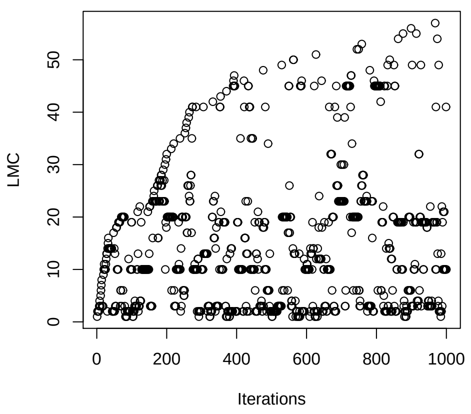


(a) Changing cluster ID with T-probit_jags.

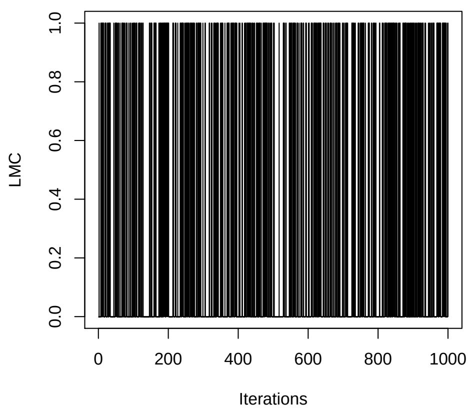


(b) Mode switching status with T-probit_jags. 1: change, 0: no change.

Figure 4.20: Traceplot of mode switching status with T-probit_jags.

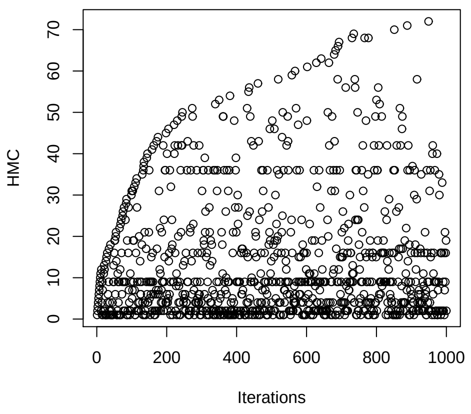


(a) Changing cluster ID with LMC.

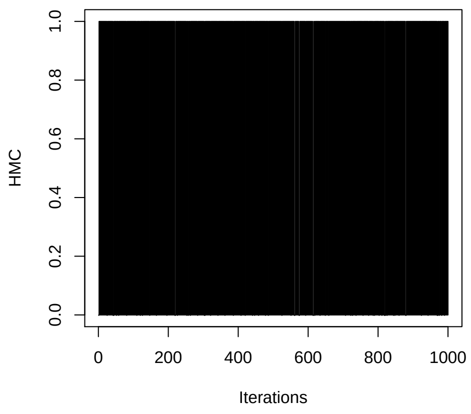


(b) Mode switching status with LMC. 1: change, 0: no change.

Figure 4.21: Traceplot of mode switching status with LMC.



(a) Changing cluster ID with HMC.



(b) Mode switching status with HMC. 1: change, 0: no change.

Figure 4.22: Traceplot of mode switching status with HMC.

4.6 An example for Investigating the Choice of Regularization of Coefficients

We generate a dataset with 200 features (p=200) and 1100 samples (n=1100). 100 samples are used as training cases ($n_{tr} = 100$) and 1000 samples are used as test cases ($n_{ts} = 1000$). The whole dataset is generated from the following multivariate Gaussian model:

$$P(y_i = c) = \frac{1}{2}, i = 1, 2, ..., n, c = 1, 2, \tag{4.11}$$

$$\mu_{1,1} = 0, \mu_{2,1} = 2,$$

$$\mu_{1,2} = \mu_{2,2} = 0,$$

$$z_{ij} \sim N(0, 1), i = 1, 2, ..., n, j = 1, 2,$$

$$x_{i1} = \mu_{y_i,1} + \frac{10}{7} z_{i1}, i = 1, 2, ..., n, \textbf{(Group 1)}$$

$$x_{i2} = \frac{10}{7} z_{i1} + 2 z_{i2}, i = 1, 2, ..., n, \textbf{(Group 2)}$$

$$x_{ij} \sim N(0, 1), i = 1, 2, ..., n, j = 3, ..., 200, \textbf{(Group 3)}$$

where x_{ij} is the ith observation of feature j. The data structure is also shown in Figure 4.23.

With data generated in this way, we have both feature 1 and feature 2 relevant to the response vector $y = (y_1, ..., y_i, ..., y_n)$. Feature 1 ($x_1$) has nonzero mean value difference ($\mu_{2,1}$ - $\mu_{1,1}$ = 2), while feature 2 (x_2) has zero mean value difference ($\mu_{2,2} = \mu_{1,2} = 0$), but it is strongly correlated with feature 1. Data generated in this way can be considered from a logistic regression model:

$$P(y_i = 1) = \frac{\exp(\beta_0 + x_{i.}\beta)}{1 + \exp(\beta_0 + x_{i.}\beta)}, i = 1, 2, ..., n, \tag{4.12}$$

where intercept value $\beta_0 = 0$ and the coefficient values β are ($\beta = (2.6, -1.22, 0, ..., 0)$) with

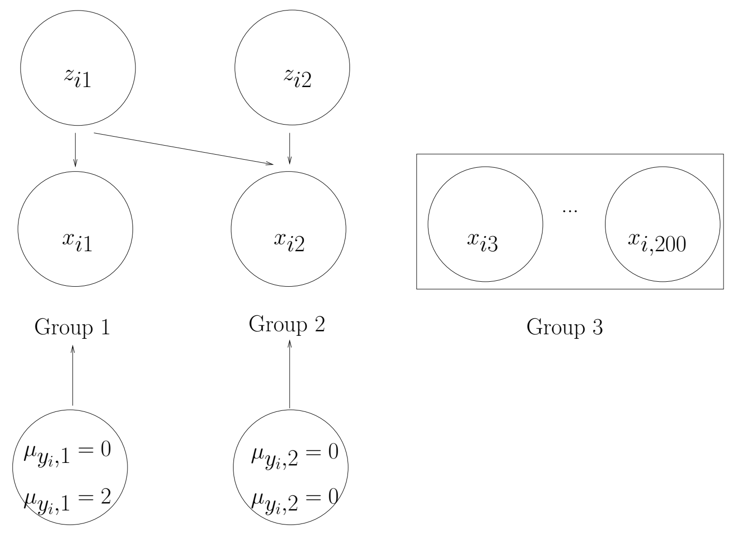
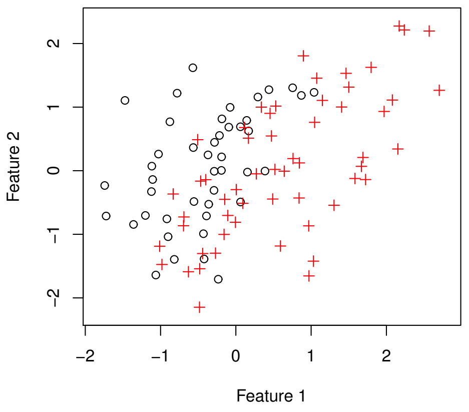


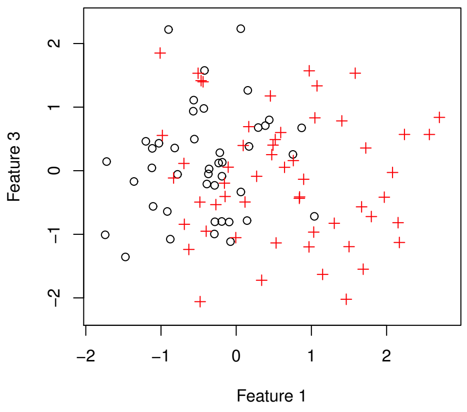
Figure 4.23: Data structure demonstration. x_{i1} and x_{i2} are the first and second feature values from the ith observation. $x_{i3}, ..., x_{i,200}$ are the feature values (in the same group) from the ith observation. z_{i1}, z_{i2}, z_{i3} are three individual values. $\mu_{y_i,1}$ and $\mu_{y_i,2}$ are the sample mean values of feature 1 and feature 2, where y_i is the response value of the ith observation.

only nonzero entries on feature 1 and feature 2. Thus, among all 200 features, only the first two features are relevant to the response vector y and the others are irrelevant. The scatterplots of feature 1 - feature 2 and feature 1 - feature 3 are shown in Figure 4.24.

We implement both T-probit regression and the LASSO methods under varying regularization parameter values. The T-probit regression is implemented with parameter settings α_0=1, ω_0=0.5, α_1=1, R_1=2000, n_1=30, R_2=3000, n_2=15, l_1=5, l_2=50, ϵ=0.5, η=0.04. In total, we obtain 45000 Markov chain samples after a 60000 burn-in period. In the end, we only record 3000 samples of β and λ. During the burn-in we use a small number of leapfrogs (L_1=5) and after the burn-in the number of leapfrogs is set as L_2=50. The stepsize factor for each iteration is the same (ϵ=0.05). η=0.04 is the cutting off value used in the restricted Gibbs sampling. α_0 and $\sqrt{\omega_0}$ are the shape and scale parameters for the model, while α_1 and $\sqrt{\omega_1}$ are the shape and scale parameters for the coefficient vector β. The values of the regularization parameter $\log(\omega_1)$ are chosen evenly from -10 to -4. The entire Markov chain only takes 10 minutes to create and it converges quickly to equilibrium. For different values of $\log(\omega_1)$, we record the values of all coefficients. We also conduct LASSO path solution study using the LARS algorithm for different values of the regularization parameter λ. The



(a) Scatterplot of feature 1 and feature 2.



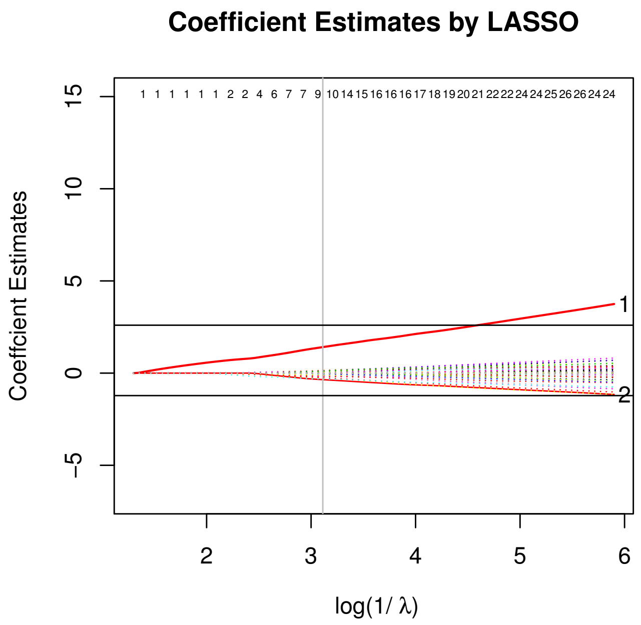
(b) Scatterplot of feature 1 and feature 3.

Figure 4.24: Scatterplots of feature 1 - feature 2 and feature 1 - feature 3.

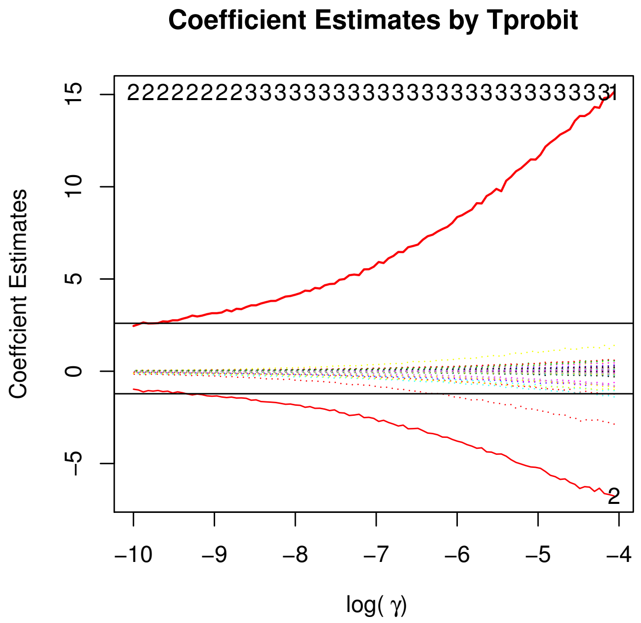
coefficients from LASSO are also recorded with respect to the regularization parameter values $\log(1/\lambda)$.

In Figure 4.25, we show the solution paths of the T-probit model and LASSO with respect to the regularization parameters. The true values of the first two coefficients are shown as two horizontal lines. The red curves denote changing coefficient values of feature 1 (x_1) and feature 2 (x_2) with respect to the regularization parameter $\log(\omega_1)$. For each chosen value of $\log(\omega_1)$, we also show the number of features selected (by two methods) on the top of each figure. That is, with a specific regularization parameter and a chosen criterion (0.1), we identify how many features are significant ($0.1 \times \max\{\beta\}$) and print the number on the top.

In Figure 4.25 we see that T-probit consistently selects the first 2 features while LASSO has the over-selection problem with large $\log(1/\lambda)$. For T-probit, the coefficients of feature 1 and feature 2 (β_1 and β_2) are close to the true value while $\log(\omega)$ ranges from -10 to -8. However, when the regularization is increased the magnitudes of β_1 and β_2 grow larger. In comparison, LASSO only correctly identifies the first two features within a narrow range of regularization parameter values (2.09, 2.28). If the regularization parameter is smaller (less than 2.09), then only feature 1 is selected. The reason is that feature 2 is not relevant alone, so it is easy to be omitted by models when regularization is tight. When $1/\lambda$ is larger than



(a) Regularization path for LASSO. λ is the regularization parameter.

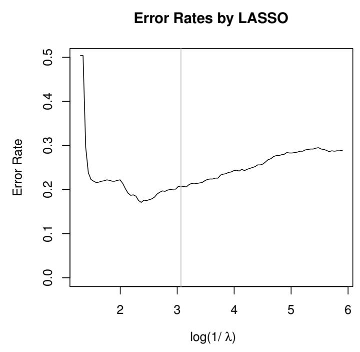


(b) Regularization path for T-probit Model. γ is the regularization parameter.

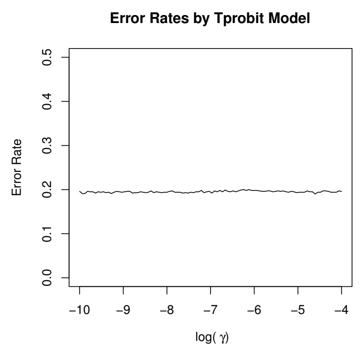
Figure 4.25: Regularization path for LASSO and T-probit model. Red curve denotes the changing coefficients values of feature 1 and feature 2 with respect to changing regularization parameter. Numbers on the top: the numbers of features selected respectively.

2.09, more than two features (some of which are irrelevant) will be selected by LASSO. With the default optimal λ value (with respect to cross-validated training error), there are as many as seven features selected. In conclusion, the feature selection results of T-probit regression are more consistent compared to LASSO with respect to regularization, while LASSO suffers from the over-selection problem with loose regularization.

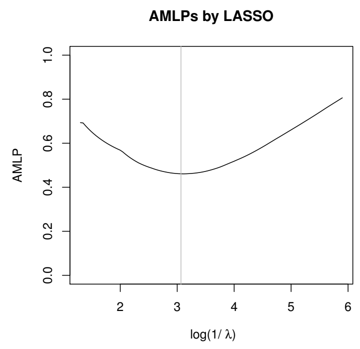
The predictive performance of LASSO and T-probit regression with respect to changing regularization parameters shows that T-probit is better than LASSO on this simple logistic regression case. Figure 4.26 shows the error rate, AUC and AMLP of both methods. The error rates of T-probit are around 20% while the LASSO results are less stable and substantially higher than T-probit. The T-probit also has consistently better AMLP and AUC values than LASSO. Most importantly, T-probit has robust prediction results to the choice of λ, while LASSO fails to provide stable prediction results since it omits the relevant but weakly differentiated (small fold change) feature 2 with stronger regularization. On the other hand, with looser regularization, too many irrelevant features are included in the selection pool and the predictive performance will be reduced. In conclusion, with varying regularization parameters, T-probit provides more stable feature selection results compared to the LASSO method; as a result, the predictive performance of T-probit is also superior to the LASSO method.



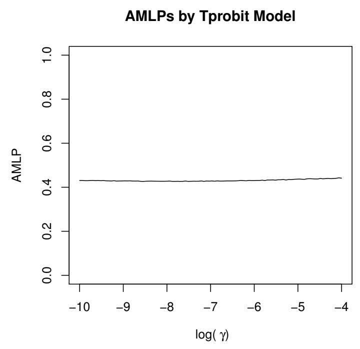
(a)



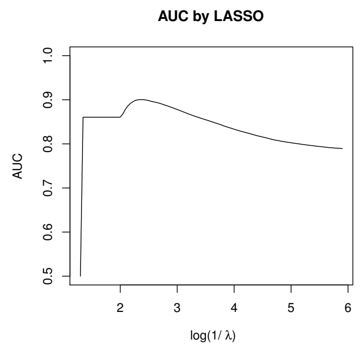
(b)



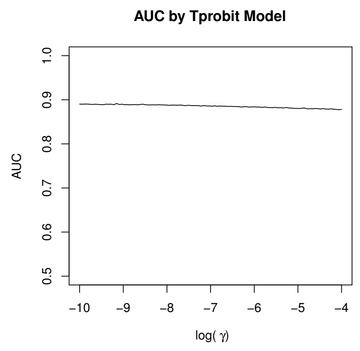
(c)



(d)



(e)



(f)

Figure 4.26: The ER (error rate), AMLP (average minus log-probability) and AUC values of LASSO and T-probit regression with respect to different regularization parameter values. For LASSO, the regularization parameter (λ) value is chosen from 1 to 6, and for T-probit regression the regularization parameter (ω) is chosen from exp(-10) to exp(-4).

Chapter 5
Comparison of Different Methods Using Real Datasets

5.1 Analysis of Breast Cancer Data

Breast cancer is a type of cancer that begins in breast tissue, which is composed of lobules (lobular carcinoma in situ) and ducts (ductal carcinoma in situ) that connect the lobules to the nipple. In many cases, it eventually grows to be a mass or lump called a tumor. Most tumors are benign and do not spread out of control, and some of them are called in situ since they are confined within the lobulesor milk ducts. Another type of cancer is called invasive since it does not confine itself within the initial tissue compartment (lobules or ducts) and tends to break the walls to invade neighboring tissues of the breast. In practice, patients are classified using the “TNM system” based on the size of their tumors (T), whether or not it spreads to lymph nodes (N), and whether or not the tumor has metastasized (M).

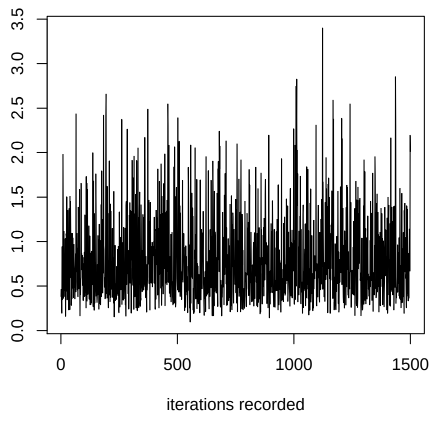
In 2013, there were 23,800 new cases of breast cancer diagnosed among women in Canada, which represents 26% of all new cancer cases in women in this year, and around 5,000 women will die of their metastatic disease (Canadian Cancer Society). Cancer cells may or may not have three different types of receptors on their surface and in their nucleus: Human epidermal growth factor receptor 2 (HER2), estrogen receptor (ER) and progesterone receptor (PR). HER2+ cancers (whose cells have HER2 receptors) are generally more aggressive than HER2- cancers (Rubins and Cotran pathologic basis of disease). Compared with ER- cancers, ER+ cancers depends on estrogen to grow and so there are treatments developed to block estrogen (e.q., tamoxifen), which can result in a substantial improvement in prognosis.

ER- cancers are usually recognized as having a high risk for recurrence, even though there are ER- patients who do well without adjuvant chemotherapy or remain disease free after local therapy alone. For ER+ patients, there have been RT-PCR (Reverse Transcription Polymerase Chain Reaction) prognostic tests developed to estimate the recurrence rate and suggest possible choices of chemotherapy (Oncotype DX). However, such tests have limited predictive power of recurrence on ER- cases (Paik et al., 2004). Therefore, it is of interest to conduct genome-wide studies to discover genes associated with ER status.

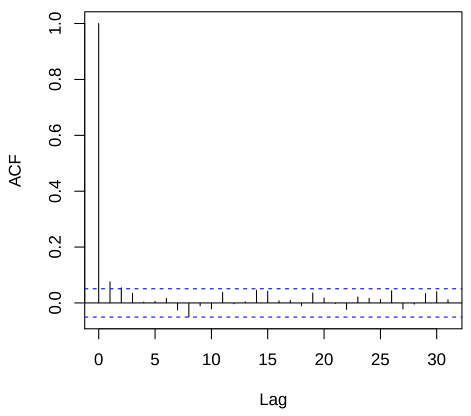
In this chapter, we use the data collected from Surgical Pathology at Johns Hopkins Hospital (Baltimore, Maryland) in 2011. This dataset includes 124 samples, with 21 normal and 103 primary invasive cancer samples. The invasive samples include two strongly estrogen receptor-positive (ER++), 48 moderately estrogen receptor-positive (ER+), and 53 receptor-negative (ER-). The methylation was reported as a β-value, and the values of β range from 0 to 1. The samples were analyzed using the Human Methylation27 DNA Analysis BeadChip (GPL8490) platform, which includes 27,578 probes.

We downloaded the data from the Gene Expression Omnibus (GEO) repository, and then preprocessed it with `Bioconductor` in R (http://www.bioconductor.org). We use the R package `samr` to pre-process the data before conducting different classification algorithms. After analysis using Significance Analysis of Microarrays method (SAM) (Tusher et al., 2001), the top ranking 5000 genes were selected according to their t-test statistics and they are all normal distributed. We save the feature values in a data matrix X and responses in vector Y. Each feature $x_j, j = 1, ..., p$ (p=5000) is then standardized with its median value and standard deviation. We conduct classification analysis for the ER status $Y_i, i = 1, ..., n$ (n=101) with different methods, including T-probit.

Our experiments includes six different methods we want to compare with T-probit: LASSO, Group LASSO, Supervised Group LASSO, random forest, Penalized Logistic Regression with t priors. We run T-probit as discussed in Appendix C. In total, we obtained 3.9 million Markov chain iterations (after 600K burn-in period) in 29131.16 seconds. In the end, we record 1500 Markov chain samples for further analysis. Figure 5.1 illustrates the magnitude of Markov chain iterations obtained. The magnitude of an iteration $\beta = (\beta_1, ..., \beta_p)$ is calculated as $\sqrt{\sum_{j=1}^{p} {\beta_j}^2}$. We show the magnitude of 1500 iterations in Figure 5.1a and corresponding correlogram in Figure 5.1b. The autocorrelations for Markov chain magnitudes at



(a) Magnitude of each Markov chain iteration.



(b) Auto-correlation of the Markov chain.

Figure 5.1: Magnitude and auto-correlation of Markov chain samples.

varying time lags is demonstrated in Figure 5.1b with the R function `acf`. The autocorrelation here is the cross-correlation of the magnitudes with itself at a specific time lag across the Markov chain. From the plot we see the autocorrelation quickly converges to 0 with time lag $l \geq 1$. This ascertains randomness of Markov chain transitions and thus we are assured that the Markov chain is mixing well. Figure 5.1a also confirms this conclusion since the magnitude of the iterations stabilize quickly after the first few hundred iterations.

We first implement all seven methods on the whole data (101 observations) and obtain gene feature selection results. The feature significance of all methods (except T-probit) is measured with coefficient vectors. For T-probit we can extract a set of sparse feature subsets from Markov chain and report them as feature selection results. Table 5.1 reports all feature subsets and their relative frequency among all Markov chains. The "coefficient" column shows the feature coefficients that signify the importance of each gene. For each feature subset as shown in Table 5.1, we use leave-one-out cross-validation (LOOCV) to evaluate its predictive performance by applying Bayesglm with the default choice of prior (Cauchy) distribution. The last three columns ("AMLP", "NMC" and "AUC") show respectively the AMLP (average minus log-probability), the number of misclassified cases and area under the ROC (Receiver operating characteristic) curve with cross-validation. For example, given a fixed feature subset (23, 77), we train Bayesglm on features (x_{23}, x_{77}) with LOOCV, and collect predictive probabilities on validation cases across 101 splits. The LOOCV evaluations of predictive performance in Table 5.1 and Table 5.2 are optimistically biased because the feature subsets were pre-selected with the whole data set that contains all features in the beginning, and we test these feature subsets on the exact same data set again with LOOCV. Nevertheless, these evaluations can still be used to compare the predictive power of different feature subsets.

	fsubsets	freqs	coefs(w/int)	cv_AMLP	cv_NMC	cv_AUC
1	23,77	0.05	2.68,-1.76,-5.4	0.21	9.00	0.98
2	77,554	0.03	3.43,-6.9, 1.81	0.25	11.00	0.96
3	1,366,1795	0.02	0.76,-3.27,-3.43,4.01	0.11	4.00	0.99
4	23,77,1587	0.02	2.84,-2.33,-5.59,1.21	0.16	6.00	0.99
5	1,1526	0.02	0.57,-2.58,1.87	0.23	12.00	0.96
6	3	0.01	0.66,-2.94	0.36	12.00	0.91
7	77	0.01	2.17,-5.43	0.38	19.00	0.89

Table 5.1: Top selected feature subsets (from whole data with T-probit) and corresponding cross-validated predictive performance. The prediction is measured with cross-validation using Bayesglm. AMLP: average minus log-probability. NMC: number of misclassified cases. AUC: area under ROC curve.

Method	fsubsets	cv_AMLP	cv_NMC	cv_AUC
T-probit_subset1 (T-probit_top)	23,77	0.21	9.00	0.98
T-probit_subset2 (T-probit_optimal)	1,366,1795	0.11	4.00	0.99
T-probit_subset3	23,77,1587	0.16	6.00	0.99
LASSO	25,266,614	0.27	10.00	0.95
Group LASSO	2256,1795,266	0.52	21.00	0.82
Supervised Group LASSO	266,2256,1756	0.51	25.00	0.83
Random Forest	10,8,103	0.32	13.00	0.93
Bayesglm	1,2256,4832	0.27	12.00	0.95
t-test	1,2,3	0.29	11.00	0.93

Table 5.2: Cross-validated prediction results with selected 3 feature subsets (with T-probit, LASSO, Group LASSO, Supervised Group LASSO, random forest, Bayesglm, simple *t*-test). The prediction is measured with cross-validation using Bayesglm.

Table 5.2 shows the cross-validated prediction evaluation of all feature subsets containing three features for all other methods. Table 5.3 shows the corresponding prediction evaluation of 10 feature subsets. From Table 5.2, we can see that the feature subsets selected by T-probit have better predictive performance than other subsets of the same size selected by the other methods. In Table 5.2, the optimal feature subset (with respect to AMLP/NMC/AUC) selected by all methods is (1,366,1795) from T-probit, with significantly better AMLP/AUC than that of the other methods. Among all other methods LASSO and Bayesglm achieve the best predictive performance with respect to AMLP/NMC. In Table 5.3, we expand the feature pools into the top 10 features and the results show that even with the top 10 features, all methods underperform T-probit with the top feature subset (1,366,1795). The only comparable results are that of Bayesglm, which has AMLP 0.16, NMC 4.00, and AUC 0.98. It is only slightly worse than (1,366,1795) since Bayesglm also selects gene 1 and gene 1795. More feature selection results of other methods can be found in Appendix E.

Method	fsubsets	AMLP	NMC	AUC
LASSO	25,266,614,67,1980,2081,2256,3009,49,69	0.08	5.00	1.00
Group LASSO	2256,1795,266,3009,2762,2081,2819,2539,4832,1980	0.21	10.00	0.97
SGLASSO	266,2256,1756,23,1980,19,2855,67,413,49	0.12	7.00	0.99
Random Forest	10,8,103,3,31,26,30,1223,18,57	0.32	10.00	0.93
Bayesglm	1,2256,4832,3009,1980,2855,266,2819,4955,1795	0.16	4.00	0.98
t-test	1,2,3,4,5,6,7,8,9,10	0.26	8.00	0.95

Table 5.3: Cross-validated prediction results with selected 10 feature subsets (with T-probit, LASSO, Group LASSO, Supervised Group LASSO, random forest, Bayesglm, simple *t*-test). The prediction is measured with cross-validation using Bayesglm.

In conclusion, the T-probit is able to select multiple feature subsets, with each containing few genes. More importantly, these feature subsets have better predictive performance compared with top feature subsets selected by the other methods. That is, the feature subsets selected by T-probit are able to achieve better prediction accuracy using the same number of features compared with feature subsets from other methods. The detailed annotation information of these feature subsets provided by the original experiment platform and HGNC (HUGO Gene Nomenclature Committee) is given in Table 5.4. Further evidence is needed to validate the biological relevance of these genes to breast cancer hormone receptor status.

We also draw the scatterplots of the values (X) of these feature subsets selected in Table 5.1. From Figure 5.2, 5.3, and 5.4 we see that these feature subsets seem to separate ER+ and ER- very well. This may not be enough to draw the conclusion that these feature subsets are optimal for the purpose of ER+/ER- classification, but such results show the necessity

Feature ID	gene ID	gene Symbol	Synonym	Annotation	Gene Product
1	100	ADA	adenosine aminohydrolase	Go function hydrolase activity, adenosine deaminase activity. go process nucleotide metabolism, purine ribonucleoside monophosphate biosynthesis, antimicrobial humoral response sensu Vertebrata.	adenosine deaminase
366	196472	FAM71C	MGC39520	synonym: MGC39520	hypothetical protein LOC196472
1795	25946	ZNF385	HZF; RZF; ZFP385; DKFZP586G1122	retinal zinc finger, go component: nucleus; go function: DNA binding; go function: zinc ion binding; go function: metal ion binding; go process: transcription; go process: regulation of transcription; DNA-dependent	zinc finger protein 385
23	10164	CHST4	LSST	N-acetylglucosamine 6-O-sulfotransferase; HEC-GLCNAC-6-ST; go component: membrane; go component: Golgi stack; go component: Golgi trans face; go component: integral to membrane; go component: intrinsic to Golgi membrane; go function: transferase activity; go function: N-acetylglucosamine 6-O-sulfotransferase activity; go process: inflammatory response; go process: sulfur metabolism; go process: carbohydrate metabolism; go process: N-acetylglucosamine metabolism	carbohydrate (N-acetylglucosamine 6-O) sulfotransferase 4
77	84816	RTN4IP1	NIMP; MGC12934	NOGO-interacting mitochondrial protein; go function: zinc ion binding; go function: oxidoreductase activity	reticulon 4 interacting protein 1.
1587	9274	BCL7C	BCL7C	B-cell CLL/lymphoma 7C	B-cell CLL/lymphoma 7C

Table 5.4: Annotation table for genes selected by T-probit.

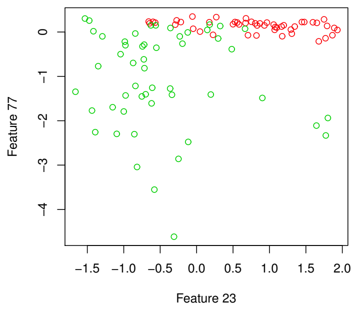
of exploring generic interactions among these genes and their roles in the ER symptoms of patients.

The plots in Figure 5.5 to Figure 5.8 show all feature scores from all methods. The intercepts (if they exist) are omitted from the feature scores for all methods. For T-probit, we show the coefficients of selected feature subsets separately, and for the other methods we only need to show all the coefficient values. To extract feature subsets from the Markov chain obtained with T-probit, we use the 0.1 criterion to find significant genes. We also use the 0.1 criterion on the feature score plots of the other methods to denote significant features. Specifically, with a coefficient vector $\beta = (\beta_1, ..., \beta_p)$, the 0.1 criterion is determined as the value $0.1 \max_j |\beta_j|$. We plot this 0.1 criterion (as well as the 0.01 criterion) on the feature scores plots.

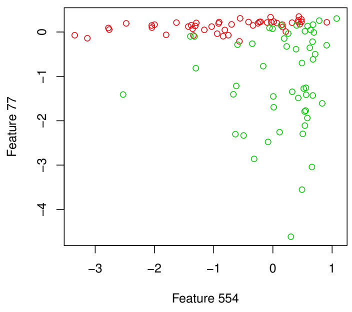
Figure 5.5 provides us a better view of the sparseness of feature selections from the different methods. Group LASSO and random forest both suffer from over-selection problems. Bayesglm selects hundreds of features with a lack of sparseness and most coefficients were not shrunk to zero while the selected features (beyond the 0.1 criterion) are similar to the T-probit results. For example, feature 266, feature 1795, feature 2256, and feature 3009 are also selected by T-probit. This makes sense since the model of Bayesglm is similar to our T-probit. In conclusion, the T-probit provides sparse feature subset selection results, as does the LASSO method, while other methods suffer from over-selection problems.

At last, we explore the predictive performances of the eight methods using leave-one-out cross-validation. We make 101 folds of the entire datasets, each of which leaves one case out as a test case, and use the remaining 100 cases to train/fit a model. In each fold, we fit T-probit using MCMC in two stages as described previously. We obtain a feature subset similar to Table 5.1 for each fold. To make the prediction for the test case, we have three methods: "Tpro_top", "Tpro_optimal", and "Tpro_average" (as we introduced in Section 3.5). For "Tpro_top" and "Tpro_optimal", we also record the number of features used in each method. For "Tpro_optimal" and "Tpro_top" the number of features used is the number of features in the "Optimal" feature subset and "Top" feature subset.

From Table 5.5, we can see "Tpro_optimal" has comparable predictive performance with LASSO, Group LASSO and Supervised Group LASSO using fewer features. For example,

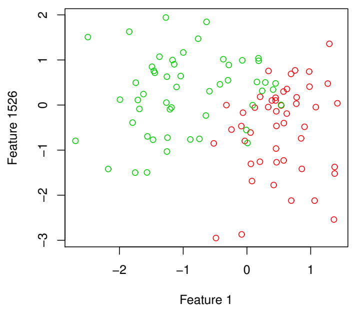


(a) Scatterplot of gene subset (23, 77).

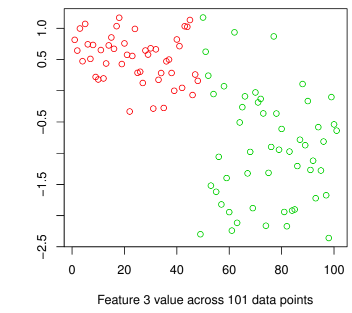


(b) Scatterplot of gene subset (77, 554).

Figure 5.2: Scatterplots of feature subsets (23, 77) and (554, 77). Colors: green y=1, red y=0.

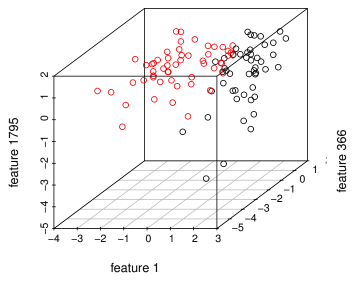


(a) Scatterplot of gene subset (1, 1526).

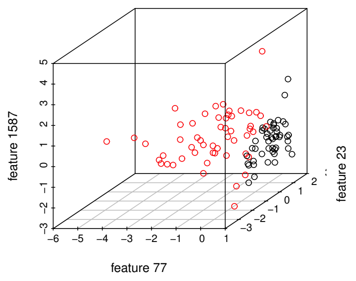


(b) Scatterplot of feature 3 values.

Figure 5.3: Scatterplots of feature subsets (1, 1526) and (3). Colors: green y=1, red y=0.

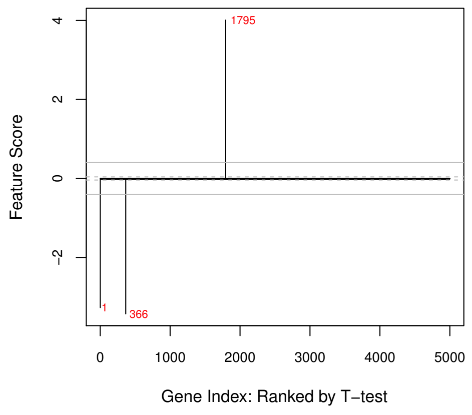


(a) Scatterplot of gene subset (1, 366, 1795).

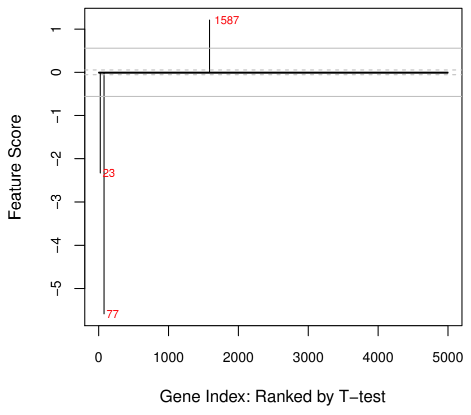


(b) Scatterplot of gene subset (23, 77, 1587).

Figure 5.4: Scatterplots of feature subsets (1, 366, 1795) and (23, 77, 1587). Colors: red y=1, black y=0.

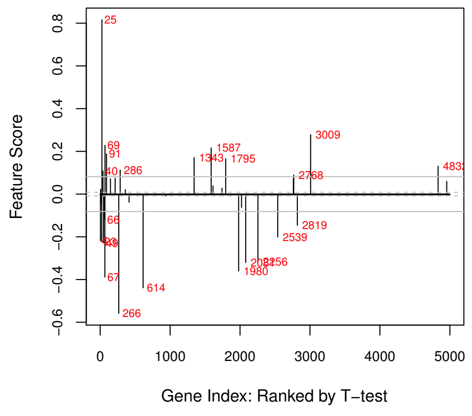


(a) Feature scores of subset (1,366,1795).

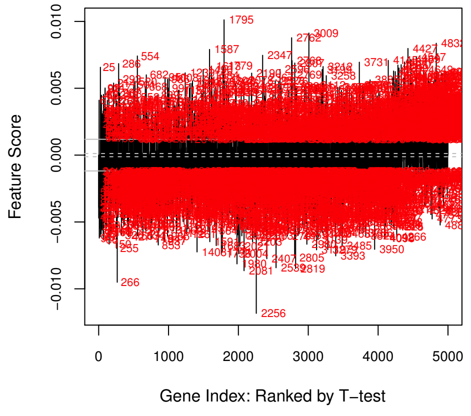


(b) Feature scores of subset (23,77,1587).

Figure 5.5: Feature scores of 2 feature subsets (1,366,1795) and (23,77,1587).

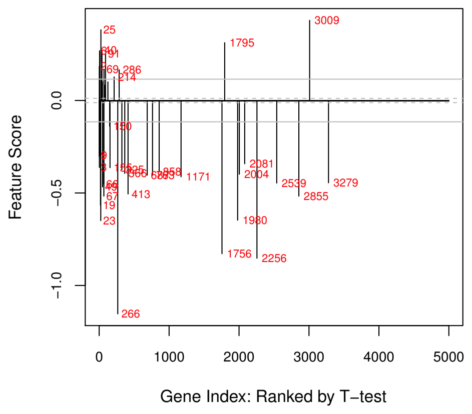


(a) LASSO feature scores.

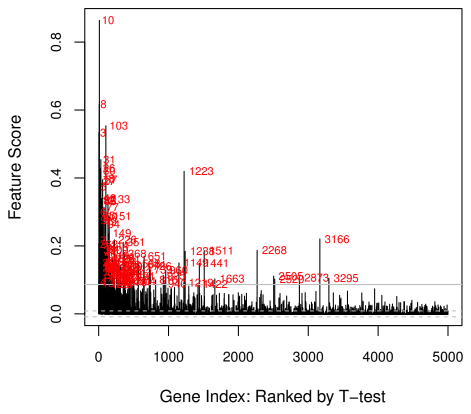


(b) Group LASSO feature scores.

Figure 5.6: LASSO and Group LASSO feature scores.

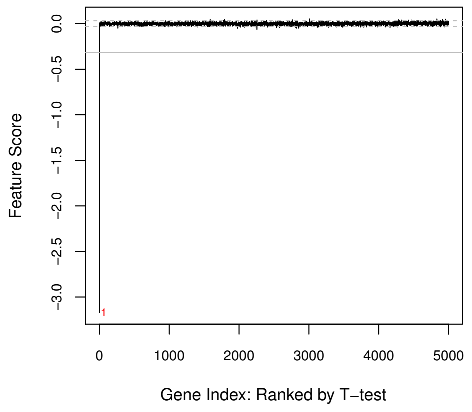


(a) Supervised Group LASSO feature scores.

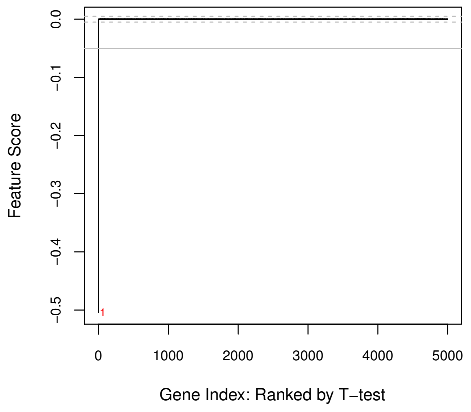


(b) Random Forest feature scores.

Figure 5.7: Supervised Group LASSO and Random Forest feature scores.



(a) Bayesglm (with Gaussian prior) feature scores.



(b) Bayesglm (with a Student's t prior) feature scores.

Figure 5.8: Bayesglm (with a Gaussian prior) and Bayesglm (with a Student's t prior) feature scores.

the "Tpro_optimal" algorithm has an error rate of 9/101 and an AUC of 0.96 (which is best among all the methods) using only 2.98 features on average in each fold. From this table, we see that only LASSO and Supervised Group LASSO have sparse feature selection results, with 39.57 and 36.62 features used (on average). Group LASSO and Bayesglm have acceptable prediction results, but they do give little relevant information on the role of particular genes in the model since they both have over-selection problems. In conclusion, the T-probit is able to provide better predictive performance with sparse feature selection results, which is even more important since it shows the necessity of exploring further the roles of these gene subsets in breast cancer.

	Tpro_optimal	Tpro_top	Tpro_average	LASSO	GL	SGLASSO	RF	Bayesglm	T
No. of features	**2.98**	2.02		39.57	2209.73	36.62	187.63	2667.47	100.00
NMC	**9.00**	21.00	10.00	8.00	9.00	10.00	10.00	10.00	12.00
AMLP	**0.33**	0.51	0.33	0.28	0.27	0.42	0.34	0.45	0.33
AUC	**0.96**	0.88	0.91	0.94	0.94	0.95	0.93	0.95	0.94

Table 5.5: Cross-validated prediction results and the number of features selected with each method (averaging over all folds).

5.2 Analysis of Childhood Acute Leukemia Data

Leukemia is a group of cancers that usually starts in early blood-forming cells and results in large numbers of white blood cells in the bone marrow. The exact cause of this disease is believed to be different for different types of leukemia, and may include both inherited and environmental factors. There are 4 primary types of leukemia: acute lymphoblastic leukemia (ALL), acute myeloid leukemia (AML), chronic lymphocytic leukemia (CLL), and chronic myeloid leukemia (CML), as well as other rare types. Leukemia is the most common type of cancer in children, and the most common types of leukemia cases in children are ALL and AML.

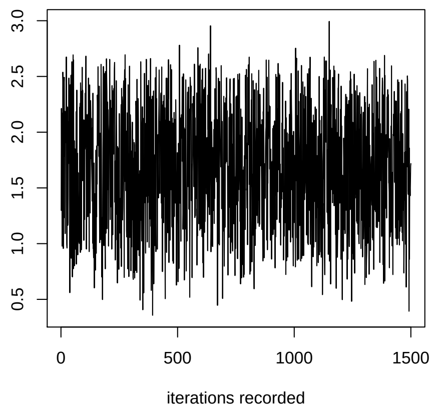
Acute lymphoblastic/lymphoid leukemia (ALL) is an acute form of cancer of the white blood cells. It starts from the early version of the white blood cells called lymphocytes in the bone marrow, and causes over-production of immature white blood cells (known as lymphoblasts). Lymphoblasts usually are overproduced quickly in the bone marrow and invade into the blood. They are able to spread to other organs, including the liver, lungs, spleen, lymph nodes, brain, kidneys, etc. Acute lymphoblastic leukemia is the most common leukemia in children 2-5 years of age. With early detection, it is possible to induce a lasting remission, defined as the absence of detectable blast cells in the marrow. In people with 5 years of leukemia remission, the cancer is unlikely to return.

Acute myeloid leukemia (AML) is a cancer of the myeloid line of blood cells. The malignant cell in AML is known as a myeloblast, which may go through genetic changes and freeze in the immature stage. The rapid growth of these immature white blood cells in the bone marrow will hinder production of normal blood cells. The cancer then causes replacement of normal bone marrow with leukemic cells, and results in a drop in red blood cells, platelets, and normal white blood cells. Compared to ALL, myeloid leukemia is relatively rare, and also has a lower survival rate. The 5-year survival rate for children with ALL has been improved to 85%, while the 5-year survival rate for children with AML is now in the range of 60% to 70%. Distinguishing AML from ALL is critical for successful treatment since chemotherapy regimens for AML and ALL are different (Bishop, 1999).

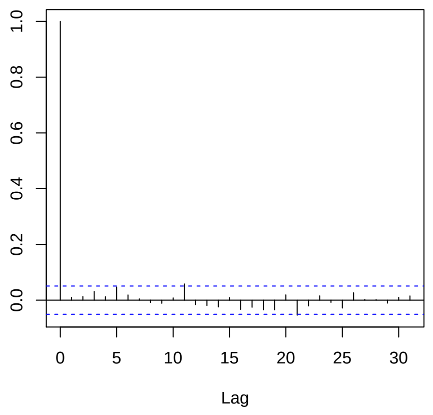
We applied T-probit methods to classify ALL and AML on a childhood leukemia dataset with 121 cases based on the Swegene Human DNA microarray platform. The experiment was initiated by Andersson et al. (2007). 98 acute lymphoblastic leukemias (87 B-lineage ALLs, 11 T-cell ALLs) and 23 AMLs were traced between 1997-2004 at Lund University Hospital and Linkoping University Hospital. In summary, RNA extraction, labeling, hybridization, scanning and post hybridization washing were performed on 121 samples from childen with ALL or AML. As a result, they obtained 27K microarrays containing 25,648 clones corresponding to 13,616 Unigene clusters and 11,645 Entrez gene entries (Unigene build 186). For experimental details, the reader can browse the entry GSE7186 in the GEO repository. The original purpose of their study is to build up a supervised classification system to differentiate ALL and AML. As a result, a K-nearest neighbor (KNN) classifier is presented with a high prediction accuracy.

We pre-process the data using similar techniques as we did in 5.1. First we label each patient with AML as 0 and ALL as 1 ($y_i \in 0, 1, i = 1, 2, ..., n, n$=121). To eliminate irrelevant information, we eliminate probes/features that have too many missing entries across observations. Specifically, for a single probe, if it has over 20% missing values (i.e., more than 25 observations missing across 121 samples), we do not include it in our study. After reducing the data scale from 27498 to 18529, we conduct a significance study using the significant analysis of microarrays method developed by Tusher et al. (2001). As a result, we have 14546 significant features, including 7383 positively correlated significant genes and 7163 negatively correlated significant genes. In practice, we select the top ranking 6000 genes (3000 positive significant genes and 3000 negative significant genes) with high significance scores. The data matrix (with dimensions 121 $\times$ 6000) is then standardized with median and standard deviation. More specifically, each gene value $x_{ij}, i = 1, ..., n, j = 1, ..., p$ is replaced with $\hat{x_{ij}}=\frac{x_{ij}-\mu_j}{SD_j}$, where μ_j is the median value of gene ID j and SD_j is the standard deviation of gene ID j. The resulting data matrix x and response vector y is now suitable for analysis.

We implement T-probit regression with our method (as we discussed in Appendix C), and in total we obtained 5.85 million Markov chain iterations (after 400K burn-in period) in 33860 seconds. In the end, we record 1500 samples of coefficients β and variance of coefficients λ. We provide the magnitude of Markov chain samples and the corresponding auto-correlations in Figure 5.9. The results show that the selected Markov chain samples have low auto-correlations.



(a) Magnitude of each Markov chain samples.



(b) Auto-correlation of the Markov chain.

Figure 5.9: Magnitude and auto-correlation of Markov chain samples.

We first implement all methods on the whole data (121 observations) and evaluate the feature selection results. Except for T-probit, the feature significance of all methods are measured with coefficient scores. For T-probit, we extract a set of sparse feature subsets from the Markov chain samples as described in Section 5.1. The full table of significant feature subsets selected by T-probit is shown in Table 5.6. The counting frequency of each feature subset is listed under the "freqs" column. The "coefficient" column shows the coefficients of each gene within each gene subset averaged over all Markov chain samples containing the corresponding subset (including intercept). Their corresponding predictive power and relative frequency among all Markov chains are also included in the table. The AMLP/NMC/AUC values of the gene subsets are evaluated on the whole data set using Bayesglm with leave-one-out cross-validation. From Table 5.6, we see that T-probit selects several succinct subsets, including (32,35), (30,35), (30,31), and (32,36) with excellent predictive performance. In total, the T-probit method selects 11 feature subsets, and the most significant genes are gene 30 and gene 32, both with frequencies over 0.1. Except for gene30 and gene32, there are still 4 subsets including these two genes: (30,35), (32,35), (30,31), and (32,36). These 6 subsets account for 62% of all samples and all have satisfactory predictive power. This confirms that T-probit can select multiple succinct feature subsets with better predictive performance. We also draw the 2-dimensional scatterplots of selected features in Figure 5.10 and Figure 5.11. These plots show that the feature subsets (30,35), (30,31), (32,36), (32,35) found by T-probit can be used to detect certain patterns in classification, and may lead to further study in the corresponding gene interactions.

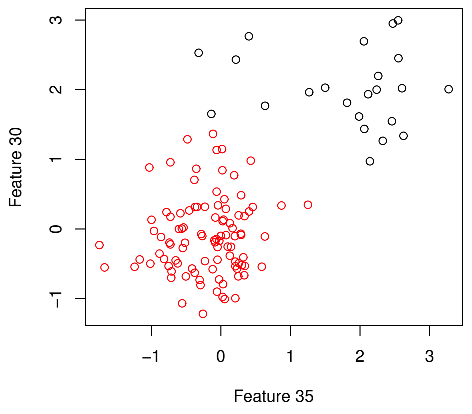
Table 5.7 shows the cross-validated feature selection results of all methods. Feature subsets containing the top 2 features are given under the column "fsubsets", and the prediction AMLP/NMC/AUC of these feature subsets are measured with LOOCV using Bayesglm. It should be noted that the predictive performance obtained in this way is biased since we use the whole dataset as the training set to obtain feature subsets, and test these feature subsets on all cases again with LOOCV. Nevertheless, it still reflects the difference in the predictive power of feature subsets selected by the different methods. Table 5.7 and Table 5.6 show that T-probit selects multiple succinct feature subsets with better predictive performance (than other subsets of the same size selected by the other methods). For example, the feature subsets (30,35) and (32,35) both have the highest predictive power (compared with other

	fsubsets	freqs	coefs(w/int)	cv_AMLP	cv_NMC	cv_AUC
1	32	0.38	17.03,-14.27	0.06	2.00	1.00
2	30	0.18	10.74,-6.97	0.07	4.00	0.99
3	36	0.09	15.06,-8.88	0.09	2.00	0.99
4	37	0.05	14.98,-8.15	0.10	4.00	0.99
5	31	0.05	12.13,-6.71	0.11	4.00	0.99
6	28	0.03	17.88,-8.12	0.14	2.00	0.95
7	35,30	0.02	8.19,-2.64,-5	0.03	1.00	1.00
8	32,35	0.02	15.94,-12.31,-8.38	0.03	1.00	1.00
9	35	0.01	17.9,-13.97	0.18	6.00	0.93
10	32,36	0.01	8.48,-4.54,-2.62	0.04	2.00	1.00
11	30,31	0.01	14.44,-9.33,-2.06	0.04	2.00	1.00

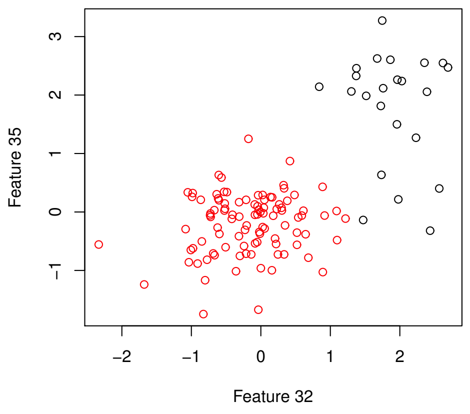
Table 5.6: Cross-validated prediction results with top selected feature subsets (from whole data with T-probit). The prediction is measured with cross-validation using Bayesglm. AMLP: average minus log-probability. NMC: number of misclassified cases. AUC: area under ROC curve.

methods in Table 5.7) with AMLP 0.03, NMC 1 and AUC 1. The LASSO method is the only comparable method with respect to AMLP/NMC/AUC since it selects (32,35) as the top 2 features. In other words, the feature subsets (32,35) and (30,35) may have genetic similarity, which results in similar predictive power. Such a phenomenon can be observed by looking at the scatterplot in Figure 5.10. One strength of T-probit is that it is able to detect such multiple gene subsets, while other methods are forced to deliver a single coefficient vector. Group LASSO, Supervised Group LASSO and random forest select single gene 35 or gene 36, which leads to slightly worse prediction results (NMC 4). The results of Bayesglm and simple *t*-test both ignore genes selected by T-probit, and as a result they both have worse predictions, with NMC 12 and 8. In conclusion, with the top 2 selected features, T-probit outperforms all other methods by providing multiple feature subsets with great predictive performance. More feature selection results of other methods can be found in Appendix F.

The plots in Figure 5.12 to Figure 5.15 show the feature scores of these features selected by all methods. For T-probit, we plot the coefficients for the two significant feature subsets we found in Table 5.6, and for the other methods we use the whole coefficient vector to show the significance score. Both the 0.1 and the 0.01 criterion are shown on the feature coefficients plots to denote significant genes. Under the 0.1 criterion, Bayesglm only selects the most significant gene. LASSO selects 16 genes and Supervised Group LASSO selects 48 genes.

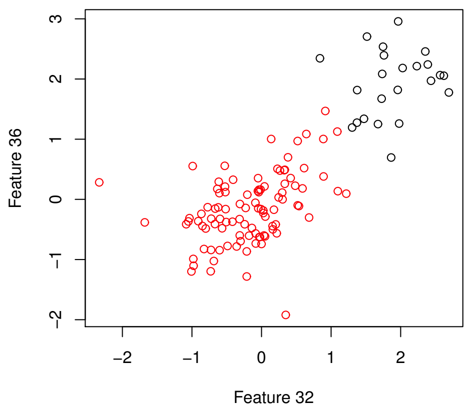


(a) Scatterplot of gene subset (30, 35).

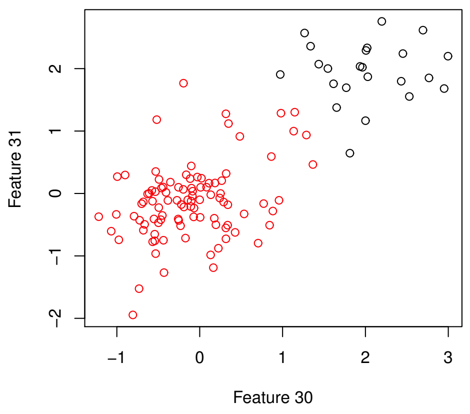


(b) Scatterplot of gene subset (32, 35).

Figure 5.10: Scatterplots of two feature subsets: (35,30) and (32,35). Colors: red y=1, black y=0.



(a) Scatterplot of gene subset (32, 36).



(b) Scatterplot of gene subset (30, 31).

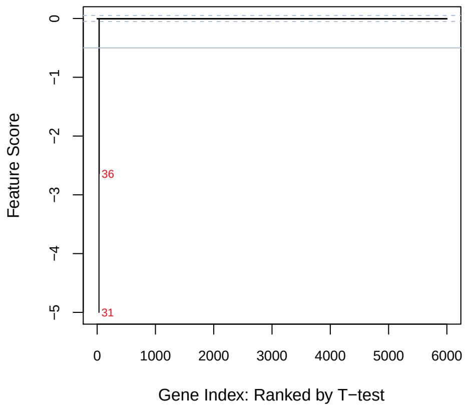
Figure 5.11: Scatterplots of two feature subsets: (32,36) and (30,31). Colors: red y=1, black y=0.

Method	fsubsets	cv_AMLP	cv_NMC	cv_AUC
T-probit_subset1 (T-probit_optimal)	35,30	0.03	1.00	1.00
T-probit_subset2 (T-probit_optimal)	32,35	0.03	1.00	1.00
T-probit_subset3	32,36	0.04	2.00	1.00
T-probit_subset4	30,31	0.04	2.00	1.00
LASSO	32,35	0.03	1.00	1.00
Group LASSO	35,115	0.15	4.00	0.95
Supervised Group LASSO	115,35	0.13	4.00	0.96
Random Forest	36,28	0.07	4.00	1.00
Bayesglm	1,5794	0.20	12.00	0.96
t-test	1,2	0.18	8.00	0.97

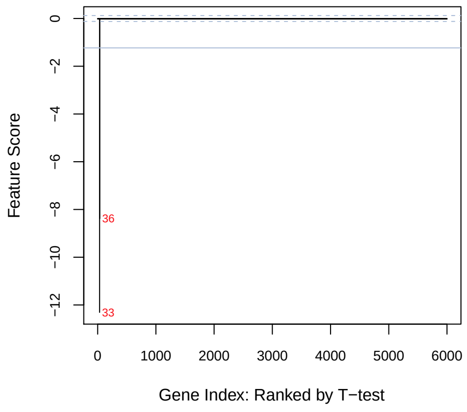
Table 5.7: Cross-validated prediction results with selected 2-way feature subsets (with T-probit, LASSO, Group LASSO, Supervised Group LASSO, random forest, Bayesglm, simple *t*-test). The prediction results are obtained with cross-validation using Bayesglm.

Remember that in the implementation details we cluster all genes into two groups using hierarchical clustering. With this grouping structure, Group LASSO has a very messy result, selecting thousands of genes in one group and nothing in the other group. This is reasonable since LASSO is expected to deliver sparse and stable feature selection results across different data structures, while Group LASSO largely depends on the group structure of features. In comparison, T-probit shows the most sparse feature selection results by selecting feature subsets (30,35) and (32,35) simultaneously.

We also implement all methods and record their prediction results as well as the average number of features used across folds with LOOCV. The approach is the same as we did for the breast cancer dataset in Section 5.1. With T-probit, we extract significant feature subsets (similar to Table 5.6) and select the top and optimal feature subsets. Their predictive powers are then measured across folds. We also include the predictive power of T-probit by averaging predictive probabilities over all Markov chain samples. Across folds the predictive performance of "Tpro_top", "Tpro_optimal", "Tpro_average" are then collected and compared in Table 5.8. The prediction of other methods are conducted as usual, with the same implementation procedure described in Section 5.1. We also record the number of features used for each method to evaluate the sparseness. To choose the number of features used for each coefficient vector, we use the 0.1 as the criterion for "Tpro_average" and all other methods, including LASSO, Group LASSO, etc.

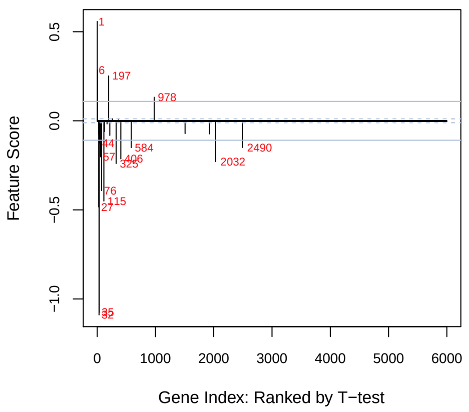


(a) feature scores of subset (30,35).

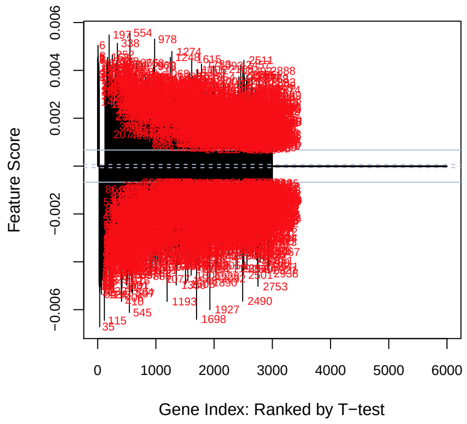


(b) feature scores of subset (32,35).

Figure 5.12: T-probit feature scores of the optimal feature subsets (30,35) and (32,35).

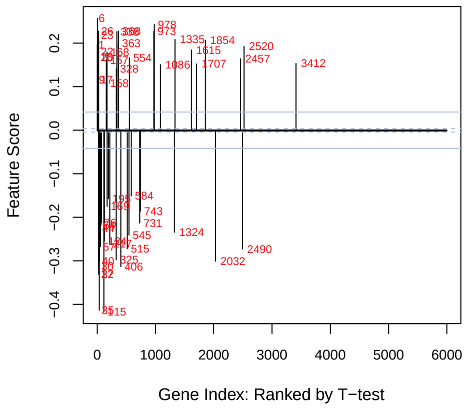


(a) LASSO feature scores.

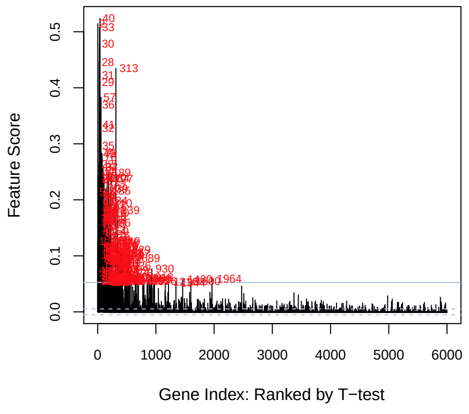


(b) Group LASSO feature scores.

Figure 5.13: LASSO and Group LASSO feature scores.

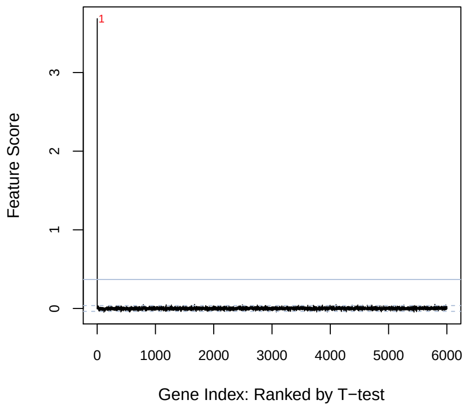


(a) Supervised Group LASSO feature scores.

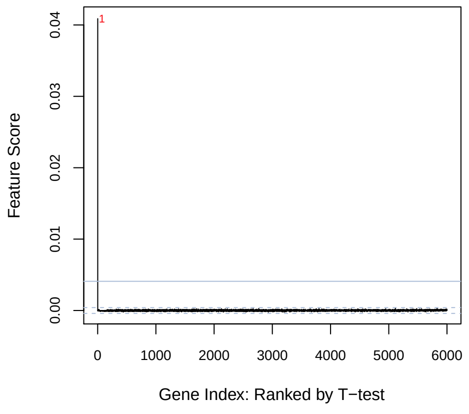


(b) Random forest feature scores.

Figure 5.14: Supervised Group LASSO and random forest feature scores.



(a) Bayesglm (with Gaussian prior) feature scores.



(b) Bayesglm (with a Student's t prior) feature scores.

Figure 5.15: Bayesglm (with a Gaussian prior) and Bayesglm (with a Student's t prior) feature scores.

Table 5.8 shows the unbiased prediction results of all T-probit methods and other models. It is unbiased since the data training and prediction are conducted separately on different cases (training and validation). The results in the table show that Group LASSO has the best cross-validated error rate (0/121), the simple t-test has best AMLP (0.02), and all methods have satisfactory AUC results. "Tpro_top" and "Tpro_average" both achieve satisfactory predictive power (NMC 3/2, AMLP 0.07/0.09, AUC 1) using far fewer features. In conclusion, this data is relatively simple and all methods have satisfactory predictive performance (except Bayesglm). T-probit provides comparable prediction results to the other methods using fewer features.

	Tpro_top	**Tpro_optimal**	**Tpro_average**	LASSO	GL	SGLASSO	RF	Bayesglm	T
No. of features	**1.00**	**1.95**		26.43	2783.26	50.34	149.33	3484.88	100.00
NMC	**3.00**	**5.00**	**2.00**	1.00	0.00	2.00	2.00	10.00	1.00
AMLP	**0.07**	**0.09**	**0.09**	0.04	0.05	0.03	0.12	0.34	0.02
AUC	**1.00**	**0.99**	**1.00**	1.00	1.00	1.00	1.00	1.00	1.00

Table 5.8: Cross-validated prediction results with the number of features used with each method (averaged over all folds).

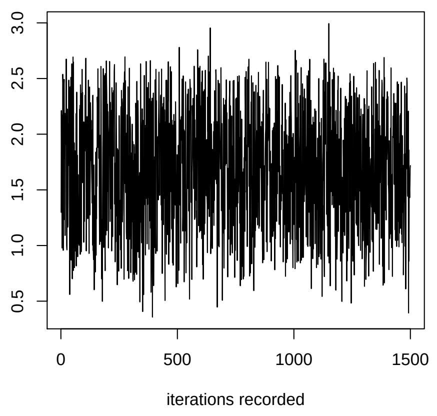
5.3 Analysis of Colon Cancer Status Data

Colon cancer is a type of cancer that develops in the colon or rectum (parts of the large intestine). It usually starts from a growth of polyp (abnormal tissue) on the inner lining of the colon or rectum, and by chance some polyps may change into a malignant tumor (cancer). Eventually, the cancer cells can grow into blood vessels or lymph nodes or even spread to distant parts of the body. The causes of colon cancer are mostly lifestyle and age, with only a small portion of cases being attributed to inherited genetic factors. Globally, colon cancer is the third most common type of cancer, and accounts for around 10% of all cases (Organization, 2014). Colon cancer is highly curable with a 90% 5-year survival rate when detected early (before it has spread). The survival rate drops when the cancer spreads outside the colon. The process of finding out if the cancer has spread to other organs of the body is called staging. The linear progression from normal colon cell to invasive metastatic carcinoma during tumorigenesis is well-studied since the beginning of global gene expression profiling (Alon et al., 1999). A direct delineation of the genetic changes that take place during carcinogenesis can be obtained by comparing differences in gene expression between colon cancer and adjacent normal cells.

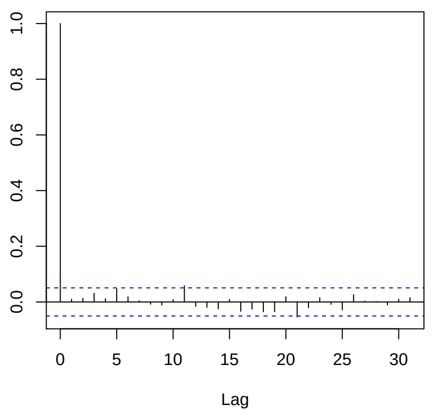
We conduct gene expression classification between mucosa samples and tumor tissue to find distinctive molecular factors related to colon cancer. The original experiment is designed and conducted by Grade et al. (2007), who proposed a gene expression profiling with 103 samples using oligonucleotide microarrays. Among 103 samples there are 73 tumors (33 stage II and 40 stage III) and 30 normal mucosa samples from matched patients. The original study identified a set of 1,950 genes with significant fold changes. The results also include classification systems with LOOCV (100% prediction accuracy), support vector machines, diagonal linear discriminant and compound covariates predictor (prediction accuracies ranging from 96% to 98%). We apply T-probit as well as other methods to this dataset (103 samples, 21732 features) to find relevant genetic factors to distinguish between colon cancer and normal samples.

Before implementing any algorithm, we preprocess the data with the `Bioconductor` package in R as we did before in 5.1 and 5.2. The original arrays contain 21732 genes (103 samples) and no significant missing entries. The correlation study tells us 8784 of the genes have significantly positive correlations with response and 10089 of the genes have significantly negative correlations with response. To eliminate irrelevant information, we only choose the top ranking 6000 significant genes. The 103×6000 data matrix is then standardized with median value and standard deviation. We implement T-probit regression on the whole data with all $p = 6000$ features and $n = 103$ observations. In total we obtained 2.88 million Markov chain samples (after 500K burn-in period) in 28752 seconds. In the end, we record 3000 samples of β and λ. Figure 5.16 shows the magnitude of Markov chain iterations and the corresponding auto-correlations.

We first implement all methods on the entire dataset (103 observations) and evaluate the feature selection results. With T-probit, we extract a set of sparse feature subsets from the Markov chain in the same way as we did in Section 5.1, and record the corresponding predictive power (with cross-validation) and coefficients in Table 5.9. For other methods we obtain vectors of coefficient scores. To compare their performance under the same criterion we show the top 2 features selected by other methods in Table 5.11 as well as their cross-



(a) Magnitude of each Markov chain iteration.



(b) Auto-correlation of the Markov chain.

Figure 5.16: Magnitude and auto-correlation of Markov chain samples.

validated predictive performance. More feature selection results of other methods can also be found in Appendix G. Remember that the corresponding predictive performance is biased; nevertheless, it still reflects the predictive power of the selected feature subsets.

Table 5.9 shows the predictive performance of all significant feature subsets by T-probit. From the table we see that some feature subsets have great predictive power: (14,26), (24,45), (18,21) all give zero test error, (4,24) has two errors, (26,31) and (2,21) have one test error. In Table 5.9, only feature 2, feature 4 and feature 26 have frequency over 10% across all Markov chains, and in total they account for 51% of Markov chain samples. This table also includes feature subsets ((14,26), (24,45), (24,4), (21,18), (2,26), (26,31), (2,21)). This discovery may lead to further study of these genes (annotation of these genes is included in Table 5.10).

In Table 5.11 we show the cross-validated predictive performance of feature subsets selected by all methods. The results show that all of them can achieve 90% accuracy with only two features. T-probit selects optimal feature subsets with 0/103 misclassified cases, while LASSO does well with 2/103 misclassified cases and random forest with 4/103. Group LASSO and Supervised Group LASSO are worse but still give satisfactory numbers of 8/103 and 9/103. This confirms that this colon dataset is rather simple, with high classification success rate of all methods. To illustrate this, we also draw the scatterplots of the features

	fsubsets	freqs	coefs(w/int)	cv_AMLP	cv_NMC	cv_AUC
1	2	0.22	7.38,4.2	0.15	7.00	0.98
2	4	0.18	7.26,7.85	0.13	5.00	0.99
3	26	0.11	11.18,-15.6	0.20	9.00	0.96
4	31	0.05	13.79,-10.49	0.24	10.00	0.95
5	14,26	0.05	13.5,12.47,-14.25	0.03	0.00	1.00
6	14	0.03	2.89,2.97	0.21	9.00	0.96
7	45	0.02	10.08,-11.58	0.29	10.00	0.93
8	24,45	0.02	18.52,-16.38,-20.52	0.03	0.00	1.00
9	18	0.02	14.32,-9.57	0.24	10.00	0.96
10	24,4	0.02	19.27,-10.3,15.1	0.04	2.00	1.00
11	19	0.02	13.83,-13.32	0.23	13.00	0.96
12	21,18	0.02	20.44,-14.02,-15.07	0.04	0.00	1.00
13	2,26	0.02	11.46,8.32,-0.92	0.06	3.00	1.00
14	21	0.02	7.11,-0.6	0.27	14.00	0.95
15	31,26	0.01	13.83,-19.12,-12.34	0.06	1.00	1.00
16	24	0.01	15.32,-7.08	0.27	15.00	0.95
17	2,21	0.01	13.26,5.89,-6.04	0.05	1.00	1.00

Table 5.9: Cross-validated prediction results with selected feature subsets (from whole data with T-probit). The prediction is measured with cross-validation using Bayesglm. AMLP: average minus log-probability. NMC: number of misclassified cases. AUC: area under ROC curve.

Feature ID	MADB-WELL-ID	OLIGO-ID	GENE	Annotation	GB-LIST
14	1190336	H200016672	TBL1XR1	Transducin (beta)-like 1X-linked receptor 1	AK022268
26	1194102	H200020399	STX16	syntaxin 16 (STX16), transcript variant 2, mRNA	AF008937, NM003763
24	1193261	H200019567	C7orf27	Chromosome 7 open reading frame 27	AK024482
45	1180755	H200007191	PTD004	GTP-binding protein PTD004 (PTD004), transcript variant 2, mRNA	NM013341
4	1190065	H200016404	S100A11	S100 calcium binding protein A11 (calgizzarin) (S100A11), mRNA	NM005620
21	1179584	H200006032	ODC1	ornithine decarboxylase 1 (ODC1), mRNA	NM002539
18	1184479	H200010876	CYP2S1	cytochrome P450, family 2, subfamily S, polypeptide 1 (CYP2S1), mRNA	NM030622
2	1181602	H200008029	SLC12A2	solute carrier family 12 (sodium/potassium/chloride transporters), member 2 (SLC12A2), mRNA	AK025062, NM001046
31	1173694	H200000203	CXCL1	chemokine (C-X-C motif) ligand 1 (melanoma growth stimulating activity, alpha) (CXCL1), mRNA	NM001511

Table 5.10: Annotation table for genes selected by T-probit.

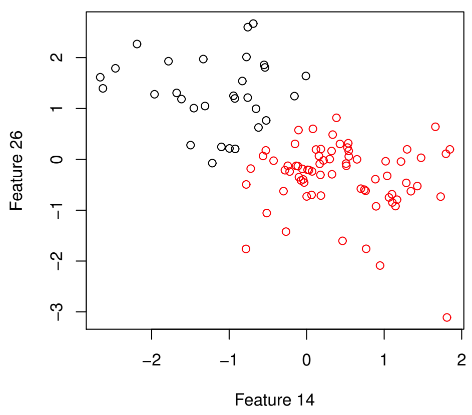
selected (by T-probit) in Figure 5.17 to Figure 5.20. These plots also show that feature subsets selected by T-probit can provide predictive patterns in the dataset.

Method	fsubsets	cv_AMLP	cv_NMC	cv_AUC
T-probit_subset1 (T-probit_optimal)	14,26	0.03	0.00	1.00
T-probit_subset2 (T-probit_optimal)	24,45	0.03	0.00	1.00
T-probit_subset3	21,18	0.04	0.00	1.00
LASSO	70,31	0.05	2.00	1.00
Group LASSO	58,665	0.27	9.00	0.95
Supervised Group LASSO	58,1	0.21	8.00	0.97
Random Forest	4,54	0.11	4.00	0.99
Bayesglm	1,3743	0.26	10.00	0.95
t-test	1,2	0.14	4.00	0.98

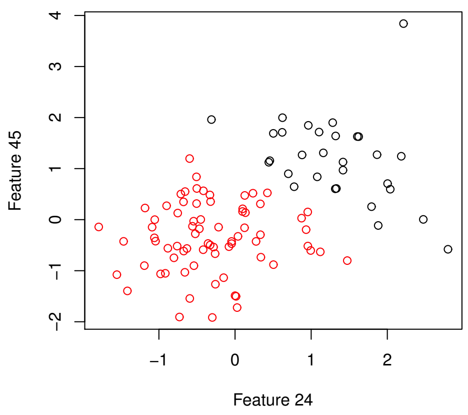
Table 5.11: Cross-validated prediction results with selected 2-way feature subsets (with T-probit, LASSO, Group LASSO, Supervised Group LASSO, random forest, Bayesglm, simple *t*-test). The prediction is measured with cross-validation using Bayesglm.

To see the feature selection results, we provide the coefficients/score from each method with plots in Figure 5.21 to Figure 5.25. For T-probit, we plot the coefficients for the top 2 significant feature subsets given in Table 5.9, and for the other methods we use the corresponding coefficients vectors to show the significance score. The 0.1 and 0.01 criterion are both drawn on the plot for all methods to denote significant features. Plots in Figure 5.21 to Figure 5.25 show that the T-probit methods provide sparse feature selections. The only comparable method is LASSO, which selects 38 features. Supervised Group LASSO selects hundreds of genes, while Group LASSO and random forest both select thousands of features. Bayesglm failed to capture any meaningful signals, selecting only one feature. In conclusion, only LASSO and T-probit generate sparse signals, while T-probit is better than LASSO because it provides more sparse feature subsets.

We also compare the unbiased predictive performance of all methods in Table 5.12. The approach is the same as in Section 5.1: For T-probit we record the predictive performance of "Tpro_top", "Tpro_optimal", "Tpro_average" and all the other methods, as well as the number of features used in Table 5.12. Tpro_top and Tpro_optimal both have worse predictive performances compared with the LASSO family (but still attain 92% - 94% accuracy). The Tpro_optimal method consistently selects 2.00 features, which means in every fold the "op-

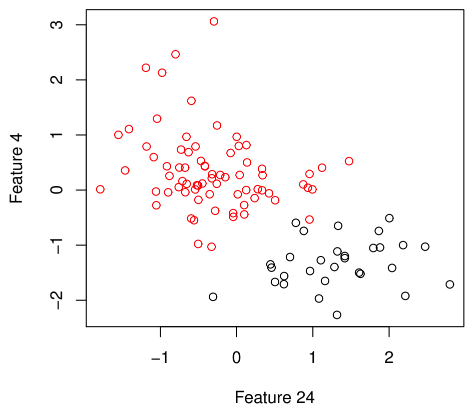


(a) Scatterplot of gene subset (14, 26).

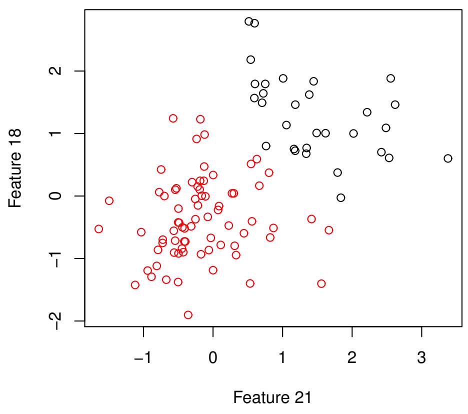


(b) Scatterplot of gene subset (24, 45).

Figure 5.17: Scatterplots of two feature subsets: (14,26) and (24,45). Colors: red y=1, black y=0.

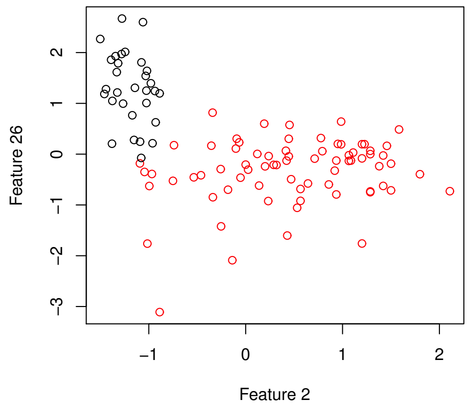


(a) Scatterplot of gene subset (4, 24).

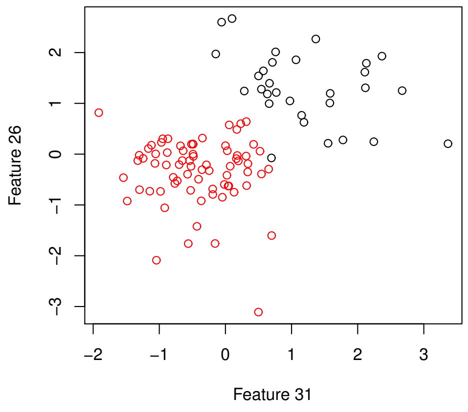


(b) Scatterplot of gene subset (18, 21).

Figure 5.18: Scatterplots of two feature subsets: (4,24) and (18,21). Colors: red y=1, black y=0.

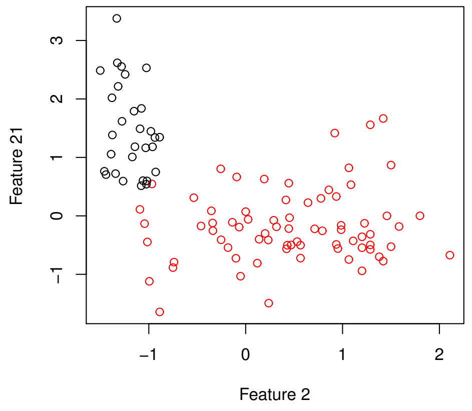


(a) Scatterplot of gene subset (2, 26).

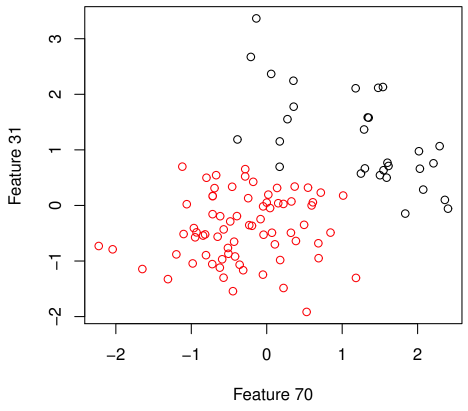


(b) Scatterplot of gene subset (26, 31).

Figure 5.19: Scatterplots of two feature subsets: (2,26) and (26,31). Colors: red y=1, black y=0.

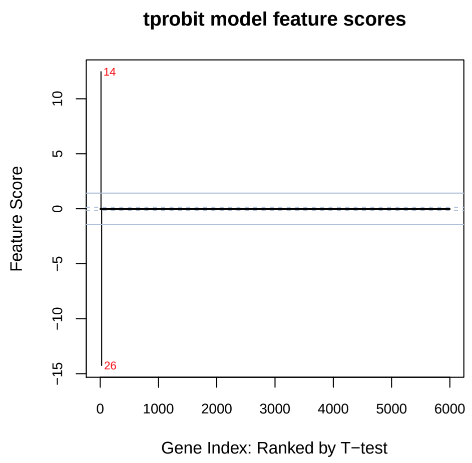


(a) Scatterplot of gene subset (2, 21).

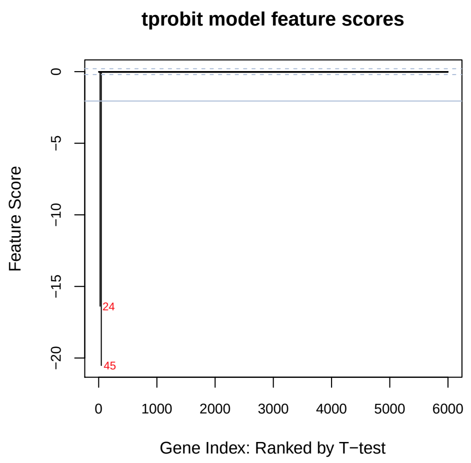


(b) Scatterplot of gene subset (31, 70).

Figure 5.20: Scatterplots of two feature subsets: (2,21) and (31,70). Colors: red y=1, black y=0.

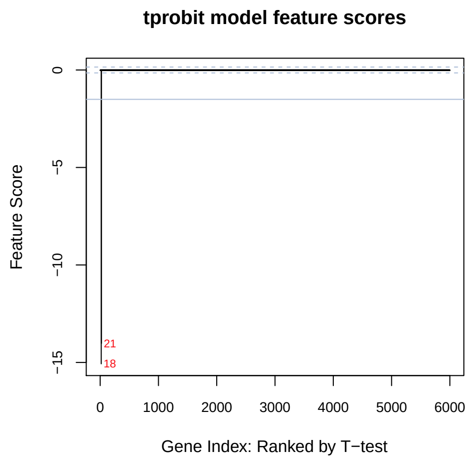


(a) The T-probit feature scores of the optimal feature subset (14,26).

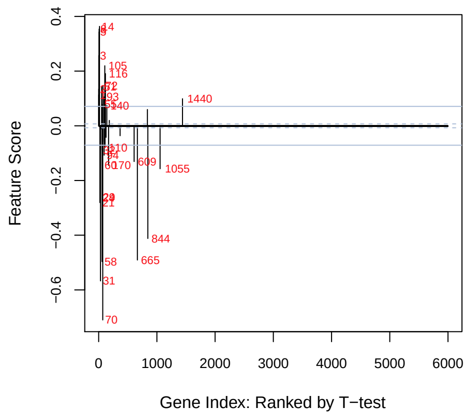


(b) The T-probit feature scores of the optimal feature subset (24,45).

Figure 5.21: T-probit feature scores of the two optimal feature subsets (14,26) and (24,45).

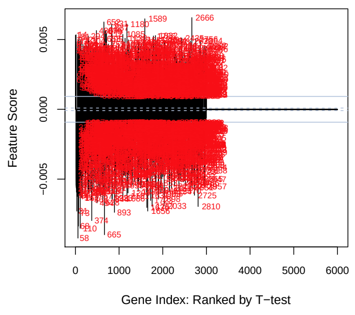


(a) The T-probit feature scores of the optimal feature subset (18,21).

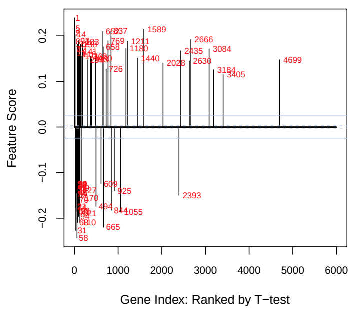


(b) LASSO feature scores.

Figure 5.22: LASSO feature scores and another T-probit optimal feature subset (18,21).

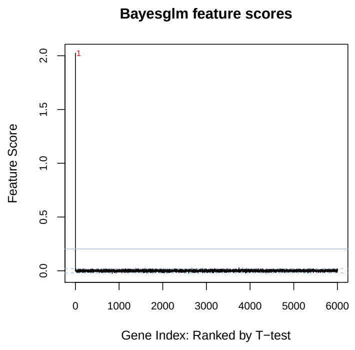


(a) Group LASSO feature scores.

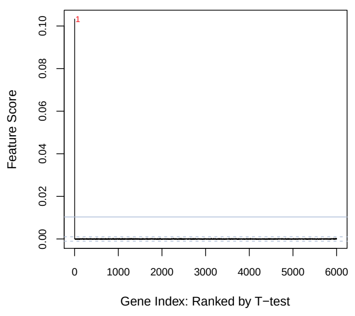


(b) Supervised Group LASSO feature scores.

Figure 5.23: Group LASSO and Supervised Group LASSO feature scores.

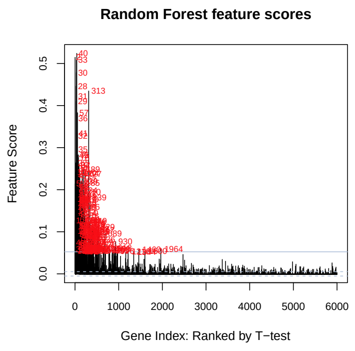


(a) Bayesglm (with a Gaussian prior) feature scores.



(b) Bayesglm (with a Student's t prior) feature scores.

Figure 5.24: Bayesglm (with a Gaussian prior and a Student's t prior) feature scores.



(a) Random forest feature scores

Figure 5.25: Random forest feature scores.

timal" feature subset always contains two features. The Tpro_average method shows better predictive performance with NMC 2 and AMLP 0.11. The best predictive performance comes from the Supervised Group LASSO and the simple t-test. Nevertheless, the predictive power of all methods are not significantly different. The most interesting part of this table is that the simple t-test also performs well with its top 100 selected features. This strengthens our statement that this dataset has a relatively simple structure, since no low p-value features contribute to the predictive power. The "No. of features" row shows that Group LASSO, random forest, and Bayesglm all suffer from over-selection problems by selecting hundreds or even thousands of features. In conclusion, all methods have good predictive performance on this dataset, but only T-probit and LASSO have sparse feature selection results.

	Tpro_top	**Tpro_optimal**	**Tpro_average**	LASSO	GL	SGLASSO	RF	T	Bayesglm
No. of features	**1.50**	**2.00**		39.24	2345.98	70.45	193.17	100.00	2996.74
NMC	**10.00**	**8.00**	**2.00**	1.00	1.00	1.00	3.00	2.00	3.00
AMLP	**0.24**	**0.24**	**0.11**	0.04	0.06	0.02	0.17	0.03	0.10
AUC	**0.96**	**0.98**	**1.00**	1.00	1.00	1.00	1.00	1.00	0.99

Table 5.12: Cross-validated prediction results with the number of features used with each method (averaged over all folds).

Chapter 6
Conclusions and Discussion

6.1 Conclusions

There are tremendous difficulties in selecting relevant features/genes in high throughput data, as we discussed in Chapter 2. One problem is that the number of noise features is much larger than the number of signals. Another problem facing gene expression analysis is that gene expression data also have grouping structures, since co-regulated genes that share the same molecular pathway tend to have similar mRNA expression levels.

In this thesis, we have proposed Bayesian T-probit models with heavy-tailed t priors for high-dimensional feature selection problems. In Chapter 3, we described our Bayesian T-probit model, which incorporates a heavy-tailed t distribution both for noise and for the regression parameters β. We use the Hamiltonian Monte Carlo method (HMC) with restricted Gibbs sampling to update β according to a marginalized posterior distribution. We implement 2-stage sampling for the T-probit model for datasets with very high dimension. In stage I we run the algorithm with all features and obtain Markov chain samples. In stage II we screen features by looking at the posterior means of the coefficients; we then run MCMC sampling again with only the features selected from stage I. For feature selection, we analyze the MCMC samples that we obtain in Stage II: dividing the samples into sub-pools according to their modes, from which we find a list of feature subsets. Furthermore, for each one of these feature subsets, we can assess its corresponding cross-validated predictive performance. We have investigated the computational efficiency of our MCMC sampling method by comparing it to ordinary Gibbs sampling implemented by JAGS and Langevin Monte Carlo in Section 4.5. We have empirically shown that our MCMC sampling algorithm can explore the multi-modal posterior more efficiently than JAGS and LMC.

In Chapter 4 we used simulated datasets with different group structures (independent group structure, correlated group structure and mixed group structure) to compare the T-probit method with many other methods in the literature, including t-test, LASSO, Group LASSO, Supervised Group LASSO, Bayesglm and random forest. In Section 4.1, we compared these methods on a simple dataset with only two strongly correlated features. The results showed that the T-probit method can split these two features into two feature subsets. In Section 4.6, we generated a dataset with 2 features: feature 1 with significant sample mean difference and feature 2 with zero mean difference but strongly correlated with feature 1. Feature 2 is marginally insignificant but useful for predicting the response. The results showed that T-probit can better separate significant features (including feature 2) from noise features, while LASSO includes many more noise features. In Section 4.4, we generated datasets with four groups of features. Features in Group 1 are relevant to the response vector y, Group 2 features have 0 mean differences across two classes, but they are strongly correlated with Group 1 and relevant to the response. Group 3 features are relevant features carrying weaker signals (compared with Group 1). We also generated datasets with independent groups of features in Section 4.2, and both independent and correlated groups in Section 4.3. The results from these simulation studies confirmed that the T-probit method selects only one or very few features from each relevant group and omits most noise features. Therefore, the feature subsets selected by T-probit are much more parsimonious. More importantly, we showed that the predictive performances of the top and optimal feature subsets selected by our method are comparable with the feature subsets of much larger size selected by the aforementioned methods in the literature. Our empirical results also showed that the succinct feature subsets selected by fully Bayesian T-probit have significantly better predictive power than the feature subsets of the same size taken from the top features selected by these methods.

In Chapter 5, we compared our T-probit regression method to other methods on three real gene expression datasets, including a breast cancer dataset, a childhood acute leukemia dataset and a colon cancer dataset. The results showed that Group LASSO and random forest both suffer from the over-selection problem and Bayesglm also lacks sparseness. The T-probit method is able to select the sparse feature subsets with good predictive performance. We also provided the annotations of the features selected by T-probit which will facilitate biological interpretation and provide guidelines for further experimental investigation, possibly with more accurate experimental methods than microarrays.

6.2 Directions for Future Work

In this thesis we have proposed our T-probit method to conduct feature selection and classification for gene expression datasets. There are also some directions in which the method could be extended. First of all, the implementation of the T-probit method can be accelerated by replacing the stage I T-probit fitting with LASSO or other methods. This will save a lot of computational resources, since LASSO is much faster than our T-probit method. Secondly, we can improve the method for clustering Markov chain samples into subpools. The essential part of our method is to extract feature subsets from Markov chain samples. To do that, we first convert the whole Markov chain into a sparse matrix with only 0 and 1 entries. This could be improved with advanced clustering techniques. Finally, this method can be applied to multiclass problems. In this thesis, we have addressed binary classification problems using the T-probit method. With a proper strategy, we believe we can apply the T-probit method to multiclass classification problems, such as multiclass cancer diagnosis.

Appendix A

Gene Expression Measurement Technologies

In genetics, gene expression is the process by which genetic information is used to synthesize gene products. All existent forms of life, including prokaryotes (bacteria), eukaryotes (complex structured organism such as animals and plants), and virus utilize this process to interpret the genetic codes and generate the macromolecular machinery for life. On the macromolecular level, this process can be represented as a generic information flow from DNA to RNA to protein. DNA is first copied into RNA by RNA polymerase, then the messenger RNA (mRNA) is produced and decoded by the ribosome to produce a specific amino acid chain, which can be folded into an active protein. (Although some specific expression information only results in with RNA production). The process by which DNA is copied to RNA is called "transcription", and the process in which cellular ribosomes create proteins is called "translation".

With advances in gene expression technologies, researchers may be interested in measuring the expression levels of certain genes in a cell or organism because it helps us to "reverse engineering" the expression process and find answers to questions involving certain genes of interest. For example, the susceptibility of patients to certain complex diseases (cancer, etc.) can be evaluated based on the expression levels of a few oncogenes. The severity of viral infection in a cell/organ can be assessed based on the levels of viral protein expression.

In general, there are two ways to infer gene expressions: either with RNA quantification or protein quantification. It is always easier/cheaper to perform RNA quantification. However, proteome analysis can provide more stable and direct information of the pathways and the biological system by focusing on the actually molecules of interest (Gygil et al., 1999), while taking possible post-translational modifications into account (Anderson and Anderson, 2005).

The microarray is one of the most well-known technologies in the field of high-throughput gene expression profiling. The word "microarray" refers to a 2-dimensional array (glass or silicon chip) containing the gene expression information of genes. The target genes are usually attached to specific probes on microarray spots and then detected with a high-throughput screening method. It is argued that the basic methodology of microarrays was first introduced in antibody microarrays (Chang, 1983). Some "gene chip" manufacturers such as Affymetrix and Illumina started to establish series of protocols since the 1990s, and now DNA microarrays have become one of the most widely used technologies for measuring gene expression. In this section, we give a brief introduction to the microarray technique and its applications.

The basic idea behind most microarrays is to use complementary base-paring to hybridize a cDNA or cRNA target sample with a specific DNA sequence. Each of the spots on the solid microarray surface contains a specific probe (single stranded DNA oligonucleotide), which is complementary to a specific mRNA molecule that corresponds to a gene. The target molecule is then "hybridized" with its complementary probe (as in Figure A.1), which means these complementary nucleic acid sequences pair with each other by forming hydrogen bonds between nucleotide base pairs. The target sequences (mRNA molecules) are usually labeled with fluorescent dye. The higher the amount of target sample hybridized to a probe, the higher the fluorescence intensity of the dye. The gene expression levels of specific genes are then detected by measuring the intensity of fluorescence of spots on a microarray slide with a scanner system.

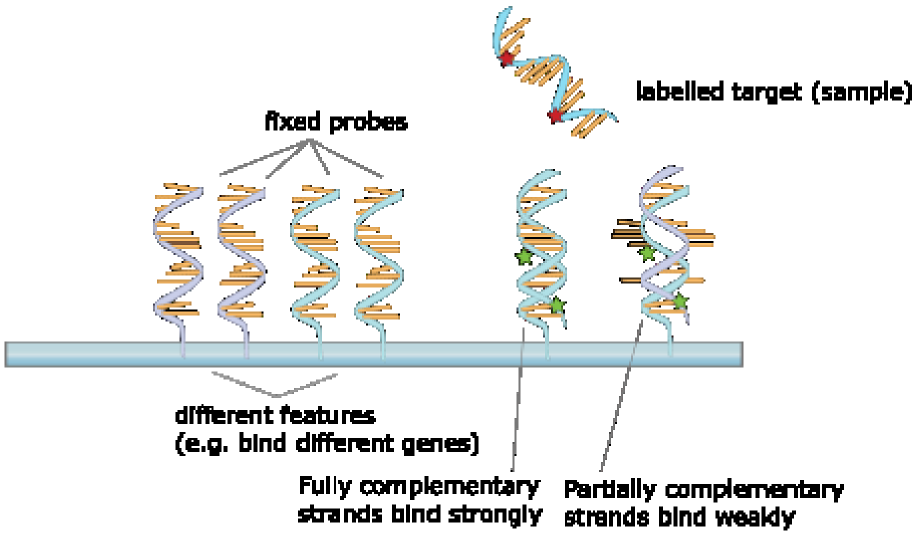


Figure A.1: Hybridization with probes.

There are various commercial gene-chip processing systems available. One of the most well-known gene-chip manufacturers is Affymetrix, which provides oligonucleotide arrays and corresponding analysis tools. Despite the differences in protocols and platforms, each oligonucleotide array usually contains hundreds of thousands of probe spots, each containing millions of copies of a specific 25 base long DNA oligonucleotide. Affymetrix uses a collection of probes to interrogate a given sequence, which is typically known as the 11 probe set. This set of probes consists of perfect match (PM) probes as well as mismatch (MM) probes. Each perfect match probe has a sequence exactly complementary to the particular gene and represents the expression of the gene, while the mismatch probe contains the same 25 base long sequence except that the 13th base in the chain is substituted with a different base that disturbs the binding of the target gene transcript. The signals generated by PM and MM are recognized as the "signal" and "noise" (background information). It helps to determine the specific intensity value for each probe set.

Other than polynucleotide arrays, cDNA arrays are also common in microarrays experiments. It is named as such because it measures complementary DNA (cDNA) levels instead of mRNA levels. The length of probe bases from cDNA microarrays is usually longer than mRNA microarrays, which contain hundreds of bases in many cases. Thus each spot corresponds entirely to a specific gene. It is usually more accurate to measure the expression levels of a specific gene using its cDNA levels since each of the probes on cDNA arrays is complementary to a cDNA molecule and a target gene.

Appendix B

Preprocessing Gene Expression Data

Data pre-processing procedures, including standardization and normalization are often required for data mining purposes. Several steps in the pre-processing procedure may be modulated, including cleansing, standardization, normalization, and feature extraction/selection if necessary. Standardization is usually recommended since the "raw" features may have different scales and cannot be compared directly. One common method can be written as $x_j^{'} = (x_j - \mu_j)/\sigma_j, j = 1, 2, ..., p$, where μ_j and σ_j are the mean and standard deviation of the jth variable values. In data preprocessing, it is also possible to apply signal processing methods to enhance signal-to-noise ratio, like the Fourier transformation and wavelet transform (Wang et al., 2003). In some cases, the expansion of input dimensionality is also considered. For example, a recent study shows that certain protein interactions are functionally related (Marcotte et al., 1999).

Data pre-processing is the first and one of the most important steps in gene expression data analysis. The results of large scale gene expression data-gathering often have loosely controlled entries with missing values, out-of-range values (e.g., -1000), etc. Such "bad data" sometimes is a technical phenomenon, just like missing values or malformed records. For gene expression data, the reason can be poor hybridization, incomplete documentation, etc. The high dimension of gene expression data makes it necessary to identify and correct the misleading entries so we can obtain consistently pre-processed data matrices. Such steps are usually called "quality assessment". The main point of data assessment is to differentiate between good and bad data. Sometimes in practice there is often no choice but to completely discard certain entries from further analysis. It is also crucial to comply with the data provider to preserve useful information within the data according to the original experiment design.

In our study we mainly collect gene expression data from Gene Expression Omnibus (GEO). GEO is a public genomics data repository supporting array and sequence-based data. It also provides basic tools to help researchers to examine experiments and platforms. In practice we work with Bioconductor (using the R programming language) to extract information from GEO repository. With the help of the R package `GEOquery`, we download a given data series using its GEO ID. The format of the files can be chosen as either GSE Series Matrix or GSE SOFT. GSE SOFT files contain full information of the experiments; however, the parsing of the GSE Series Matrix is faster than parsing the GSE SOFT files. In practice we save GSE Series Matrix as the default data format. The expression entries are saved within the GSEMatrix as ExpressionSet. Other important information such as characteristics and protocols are also included in the GSE objects. Features of interest (such as cancer stage label, tumor grading, etc.) can then be extracted from the GSE object.

After we obtain the intact expression matrix and corresponding response, the first quality measurement issue is the treatment of missing values. Troyanskaya et al. (2001) conducted a comparative study of several missing value imputation methods on gene microarray data and showed that the K-nearest neighbor (KNN) imputation method provides robust results for missing value estimation when the amount of missing data ranges from $1 - 20\%$. Suppose there is a missing entry present in gene A and sample 1. We will first identify K other genes which satisfy two conditions: 1) having values in sample 1; 2) having the most similar expression values (according to the Euclidean distance) to A in samples 2-N (N is the total number of samples). The values of these K genes in sample 1 are then weighted by its similarity to gene A and used as the imputed value of the missing entry (x_{A1}). In our real data analysis, we first discard genes that have over 20% entries missing, and then use package `SAMR` to conduct the KNN imputation. The resulting data production is then an expression matrix with a response vector.

Dimension reduction sometimes is also necessary since the original arrays usually have large scales even after deleting some missing entries. To address this problem, Tusher et al. (2001) have proposed Significance Analysis of Microarrays (SAM) to assign scores to genes with respect to changes in gene expression values relative to the standard deviation of repeated measurement. In our study we employ SAM as the default feature screening method in data preprocessing to obtain a set of gene scores. We then rank them and select the significant genes. The number of selections can be arbitrary with respect to the model, and in many cases researchers only select hundreds or even dozens of features for further study. Since our proposed T-probit model is capable of handing large amounts of data, we select 5000 to 6000 genes with top ranked relative difference (from SAM) in most cases.

Another important step we consider in data pre-processing is normalization. Normalization is the process of reducing unwanted variation either within or between features. The typical assumptions of most major normalization methods are 1) Only a small subset of genes are expected to have substantial differences between conditions. 2) Any expression is as likely to be up-regulated as down-regulated (about as many genes going up in expression as are going down between conditions). Many methods are proposed to normalize microarray data according to different platforms. Two well known normalization methods are scaling and quantile normalization. Scaling involves choosing a baseline feature and scaling all other features to have the same mean intensity as the chosen one. Quantile normalization is to calculate an empirical distribution $G(x)$ of intensities from all arrays first, then transform each value with $F^{-1}[G(x_i)]$ where F is the empirical distribution of the averaged sample quantiles. In this thesis, we use the R package `samr` and `Bioconductor` to conduct normalization for microarray datasets. More information about normalization can be found in Gentleman et al. (2006) and Tibshirani et al. (2011).

Appendix C

Implementation Details

C.1 Implementation of Existing Feature Selection Methods

In our simulation studies, we do data analysis with six different methods to compare with our T-probit regression: `LASSO` (**Least Absolute Shrinkage and Selection Operator**), **Group LASSO** (`GL`), **Supervised Group LASSO** (`Sup`), **Random Forest** (`RF`), **Penalized Logistic Regression** with t priors (denoted by `Bayesglm`) and simple t-test ranking. For a specific dataset we apply these methods to obtain feature selection results on training datasets as well as prediction results on test datasets (if there are any). Then we look into the feature selection efficiency and prediction accuracy of various classifiers corresponding to different methods. The implementation details of all six methods are illustrated in this section.

We use the R package `arm` to fit Penalized Logistic Regression with the iteratively reweighted least squares method (IWLS). More specifically, with data $(x_{i.}, y_i), i = 1, ..., n$ (from the ith observation) we use the R function `bayesglm` to train the logistic regression with Student's t-distribution for intercept β_0 and coefficients $\beta = \{\beta_j, j = 1, 2, ..., p\}$ as

$$P(Y_i = y_i | x_{i.}, \beta_0, \beta) = \left((\frac{1}{1 + e^{\beta_0 - x_{i.}\beta}})^{y_i} (1 - \frac{1}{1 + e^{\beta_0 - x_{i.}\beta}})^{1-y_i} \right), \tag{C.1}$$

$$\beta_0 \sim T(\alpha^*, \omega^*),$$

$$\beta_j \sim T(\alpha_1, \omega_1), j \in \{1, 2, ..., p\},$$

where $T(\alpha, \omega)$ is the scaled t-distribution with shape parameter α and scale parameter $\sqrt{\omega}$. We use the default value set by `bayesglm` for specifying priors for coefficients β with shape (degrees of freedom) parameter α_1 (usually 1) and scale parameter $\sqrt{\omega_1}$ (usually $\sqrt{2.5}$). For intercept β_0 we set the shape (degrees of freedom) parameter α_1 (usually 1) and the scale parameter $\sqrt{\omega_1}$ (usually $\sqrt{10}$). To compare with T-probit regression, we also implement this algorithm with the same shape and scale parameters α_1, $\sqrt{\omega_1}$ as we used for T-probit regression. As a result of model fitting, we have a coefficient vector $\{\beta_j, j = 1, 2, ..., p\}$. Feature selection can be conducted and discussed later based on this result. The coefficient solution is also used to find predictive probabilities for a new test case x^* using equation (C.1). For convenience we use "Bayesglm" to denote the method using Penalized Logistic Regression model in the thesis.

LASSO is implemented using the R package `glmnet`. For training data (x, y) we first train LASSO with the R function `glmnet` with a set of regularization parameters $\lambda = \{\lambda_m, m = 1, 2, ..., M\}$. By default, we start with minimum λ_1 value $\lambda_1 = 0.01$ and choose $M = 100$ candidate values with $\lambda_m = 0.01m, m = 1, 2, ..., M$. To find an optimal LASSO solution, we conduct cross-validation with respect to average minus log-probability over all candidate λ_m values. More specifically, for dataset (x, y) with n observations, we first split the whole data into n folds with leave-one-out cross-validation (LOOCV). Within each fold of data, we have $n - 1$ training cases and one test case. Then we train LASSO on $n - 1$ training cases with each single value of regularization parameter λ_m, and use the resulting solution to find predictive probabilities on the single test case. The predictive performance AMLP (average minus log-probability) averaging over all folds is then collected and recorded for each single value of λ_m. The value of λ_m with smallest AMLP is then chosen as the optimal λ value with respect to the optimal LASSO solution. At last, we fit `glmnet` on the whole data (x, y) again with only the optimal λ on all n observations. The fitting LASSO solution contains a set of parameter coefficients β, based on which we can perform feature selection. For a test case x^* the predictive probability can be calculated with the optimal coefficients vector β using R function `predict.glmnet`.

We implement Group LASSO with prior group structure determined by hierarchical clustering (HC). With data set x we first conduct hierarchical clustering with the `hclust` function in the R package `clust`. More specifically, for a given number of groups C, we first use the R function `hclust` to construct a tree with UPGMA (Unweighted Pair Group Method with Arithmetic Mean), and then the tree is cut into several groups by specifying the desired number of groups C. For a list of candidate values of C (and corresponding group structure), silhouette values are calculated. In practice, we usually set the candidate values from 2 to 50, which means we divide the features into at most 50 groups. The optimal value of C is chosen using the maximum silhouette value and the corresponding group structure of features in x is then saved in the form of an "index". That is, for each feature j, we assign an index label I_j to denote the group origin of this feature. With such a group index, we run Group LASSO (using the R function `gglasso`) on different values of the regularization parameter λ. An optimal λ is then chosen to minimize the cross-validated AMLP (average minus log-probability). At last we fit Group LASSO again with this optimal λ and the optimal group index. The resulting Group LASSO solution contains the coefficient value of all β and can be used to find the predictive probability for the test cases x^*.

Supervised Group LASSO is implemented with a two-stage strategy. 1) We borrow the same group structure used in Group LASSO. For each feature group we then implement the LASSO algorithm with a reduced dataset and use the LASSO solution to extract significant features. More specifically, we combine all features in the kth group into x_k and fit LASSO (as we introduced before) on $(x_k\ ,\ y)$. These features with nonzero coefficients in the resulting LASSO solution will be retained and used as representatives of group k. 2) All group representative features are then combined into a consolidated training dataset, while their group indices are retained. We fit Group LASSO on this consolidated dataset with the consolidated group structure. The resulting Group LASSO solution is assigned to the corresponding features and all other features abandoned from stage II are labeled with zero coefficients. The new coefficients and prediction results are then saved as Supervised Group LASSO results.

We implement Breimans random forest algorithm with the `RandomForest` R package (based on Breiman and Cutlers original Fortran code). Two important parameters in random forest are the number of trees (`ntree`) to grow and the number of variables randomly sampled as candidates at each split in the forest (`mtry`). With two arbitrary sets of candidate values for them, we fit randomForest with cross-validation. By default we use the candidate values of `mtry` ranging from $\sqrt{p}$ to n if $\sqrt{p} < n$, or n to $\sqrt{p}$ if $\sqrt{p} > n$. The candidate values of `ntree` are chosen from 250 to 500. For each pair value of `mtry` and `ntree` we run the random forest algorithm with the R function `randomForest` with cross-validation. The optimal pair values of `mtry` and `ntree` are then selected with respect to minimum AMLP. We then fit the whole data again with the optimal value of `mtry` and `ntree`. A coefficient vector is obtained as the feature scores and can be used to find predictive probabilities for test cases. Another simple feature selection method included is the simple t-test. With the t-test, we simply rank features and implement feature selection based on t-scores, and then perform classification on test cases with only the top ranked 100 features. It will be used as a simple benchmark to compare with other classification algorithms.

C.2 List of Setting Parameters for T-probit Implementation

We run MCMC sampling of T-probit regression in two stages. In stage 1, we run MCMC sampling with all p features. Then we use MCMC means of p coefficients to choose the top p^* features. In stage 2, we run MCMC sampling with a reduced dataset with only the p^* selected features from stage 1.

In each stage we use our C code to implement T-probit regression. To use our C function, we need to decide an initial status for the coefficients β and Markov chain parameter values for α_0, ω_0, α_1, ω_1, R_1, n_1, R_2, n_2, l_1, l_2, ϵ, η. The initial status for the coefficients β^0 is obtained using the LASSO approach. The initial value for λ is then sampled from its posterior distribution given β^0. We then implement HMC sampling with leapfrog numbers L (default value 50) and stepsize ϵ (default value 0.5). In the burn-in period, we set l_1 and run $R_1 \times n_1$ samples but only record R_1 (usually thousands) of them. After the burn-in period, we record R_2 iterations out of a total of $R_2 \times n_2$ samples. In circumstances with large values of p (usually thousands), we implement two-stage fitting of T-probit regression. That is, we first fit T-probit regression with all p features (as we discussed above) and record the resulting Markov chain samples of β and λ. The mean sample coefficients of β are calculated by averaging over all R_1 samples. The top ranking 100 features are then selected by ranking the mean values of the sample coefficients. The original dataset is then reduced to a smaller one with dimension n and $p = 100$. We then conduct stage II T-probit fitting with the same parameters α_0, ω_0, α_1, ω_1, R_1, n_1, R_2, n_2, l_1, l_2, ϵ, η. The final Markov chain of β and λ are then recorded and analyzed for the purpose of feature subset selection. In other cases, if we have a dataset with a small value of p (like hundreds), we then only implement proibt regression once.

In summary, we use the same MCMC sampling settings in two stages as listed below:

1. α_0, $\sqrt{\omega_0}$: shape and scale parameters of t priors for residual errors. They are all fixed at $\alpha_0 = 1, \omega_0 = 0.5$.

2. α_1, $\sqrt{\omega_1}$: shape and scale parameters of t priors for coefficients β. They are all fixed at $\alpha_1 = 1, \omega_1 = \exp(-10)$ in most experiments except Section 4.4 and Section 4.6 when their effects are investigated.

3. η: the cutting off value used in the restricted Gibbs sampling. In step 3 of the "Restricted Gibbs sampling with HMC" presented in Section 3.2, we only choose β_j with $j \in U = \{j|\hat{\lambda}_j > \eta\}$ to update with HMC. We often choose η so that 10% of β are updated.

4. ϵ: stepsize modifier for leapfrog steps in HMC sampling. There are two critical tuning parameters for HMC: the stepsize of each leapfrog step and the length of the leapfrog trajectory. Fortunately, they can be tuned independently (Neal, 2011). Following Neal (2011), we set the leapfrog stepsize ϵ_j for β_j with the second order derivative multiplied by a common adjustment factor ϵ:

$$\epsilon_j = \epsilon \left(\frac{\partial^2 \mathcal{U}}{\partial \beta_j^2} \right)^{-1/2}. \tag{C.2}$$

   The ϵ is an adjustment factor usually chosen from 0.1 to 1 such that we obtain the optimal rejection rate 30% for HMC (Neal, 2011). The required second-order derivative of $\mathcal{U}$ with respect to β_j is approximated by:

$$\frac{\partial^2 \mathcal{U}}{\partial \beta_j^2} \approx \sum_{i=1}^{n} \frac{x_{ij}^2}{\hat{\lambda}_j} + \frac{1}{\hat{\lambda}_j}, \tag{C.3}$$

   where x_{ij} is the value of the jth feature in the ith case.

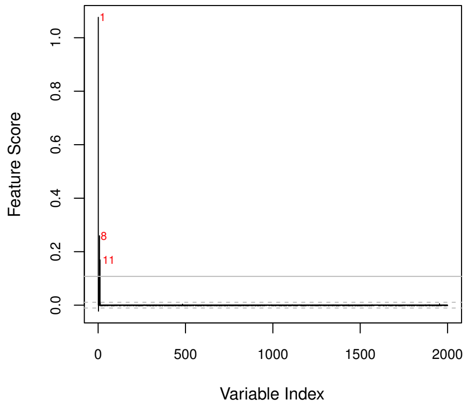
   The choice of length of trajectory is complicated. Neal (1995) recommended running HMC in two phases: initial (burn-in) phase and sampling phase. In the initial phase, one uses a leapfrog trajectory of short length so that the log likelihood can be changed more quickly and the Markov chain can more quickly reach equilibrium or a local mode for our problems. In the sampling phase, one should use a leapfrog trajectory of longer length to make full use of the ability of HMC to reach a distant point from the starting point. However, the optimal choice of L is difficult to determine since it depends on specific problems. In addition, for our problems, the posterior is highly multi-modal. Therefore, the optimal choice of L may vary for different modes. An automatic scheme for choosing L, called NUTS, is proposed by Homan and Gelman (2014). In our empirical studies, for simplicity, we use $L = 50$ which appears to be sufficiently long for our problems.

5. l_1, R_1: R_1 is the total number of recorded samples in the burn-in period. l_1 is the number of transition states (not recorded) between two recorded samples in the burn-in period.

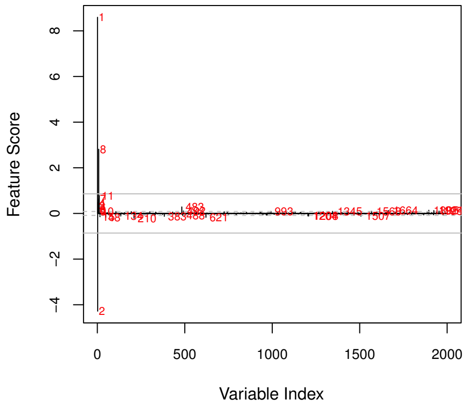
6. l_2, R_2: R_2 is the total number of recorded samples after the burn-in period. l_2 is the number of transition states (not recorded) between two recorded samples after the burn-in period.

Appendix D

Supplementary Figures for Simulations

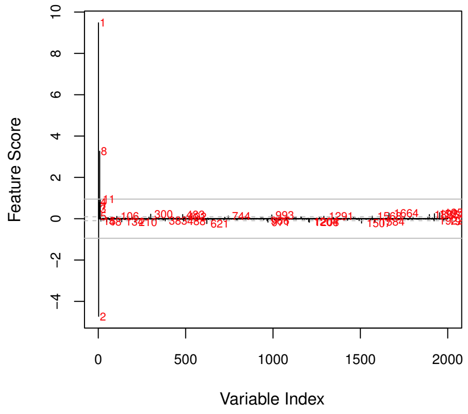


(a) The feature scores of T-probit regression (with df=1, log(w)=-20).

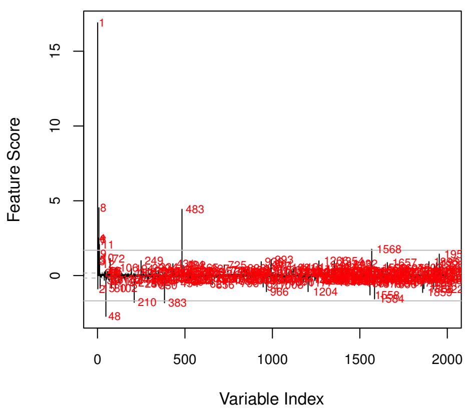


(b) The feature scores of T-probit regression (with df=1, log(w)=-10).

Figure D.1: The feature scores of T-probit regression (with df=1, and log(w)=-10 or log(w)=-20) on dataset for investigating the choice of heaviness of heavy-tailed priors in Section 4.4. The red features are the significant features selected by T-probit regression.

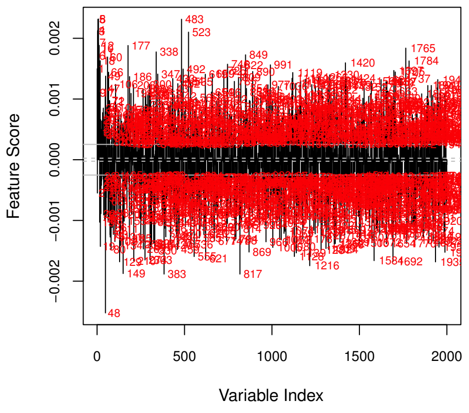


(a) The feature scores of T-probit regression (with df=0.5, log(w)=-20).

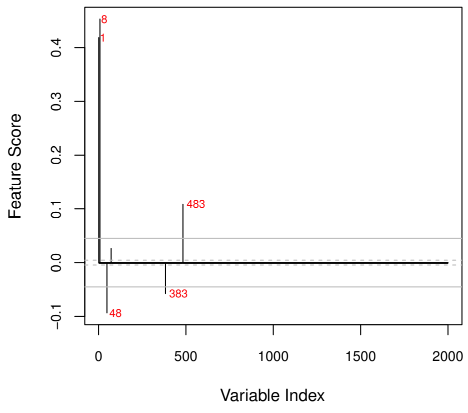


(b) The feature scores of T-probit regression (with df=0.5, log(w)=-10).

Figure D.2: The feature scores of T-probit regression (with df=0.5, and log(w)=-10 or log(w)=-20) on dataset for investigating the choice of heaviness of heavy-tailed priors in Section 4.4. The red features are the significant features selected by T-probit regression.

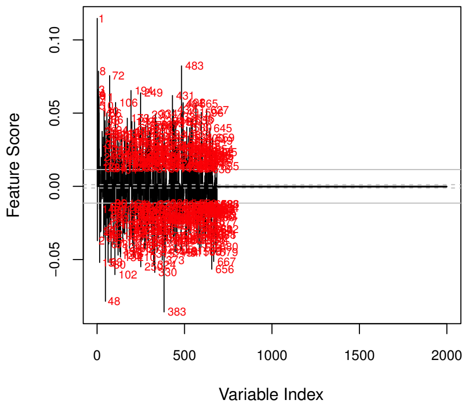


(a) The feature scores of T-probit regression (with df=10, log(w)=-10).

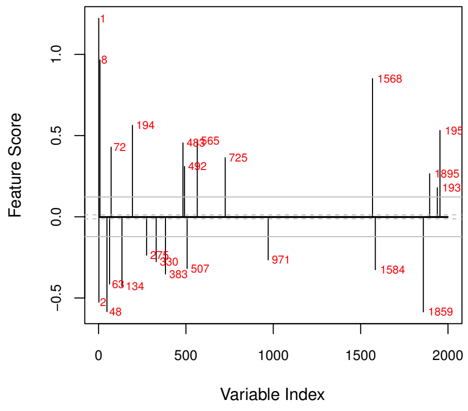


(b) LASSO feature scores.

Figure D.3: The feature scores of T-probit regression (with df=10, log(w)=-10) and LASSO feature scores on dataset for investigating the choice of heaviness of heavy-tailed priors in Section 4.4. The red features are the significant features selected by T-probit regression and LASSO.

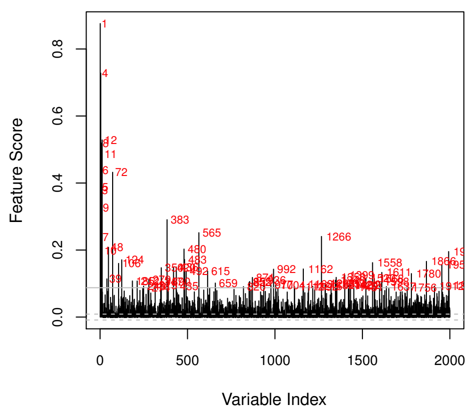


(a) Group LASSO feature scores.

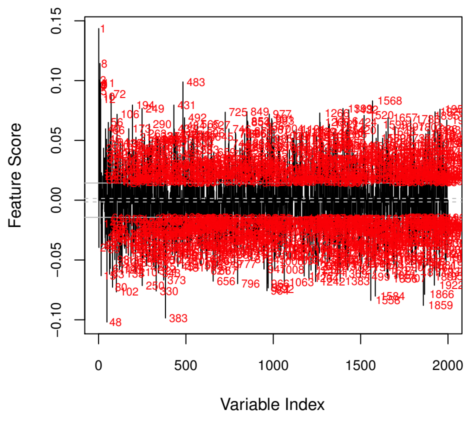


(b) Supervised Group LASSO feature scores.

Figure D.4: Group LASSO and Supervised Group LASSO feature scores on dataset for investigating the choice of heaviness of heavy-tailed priors in Section 4.4. The red features are the significant features selected by Group LASSO and Supervised Group LASSO.

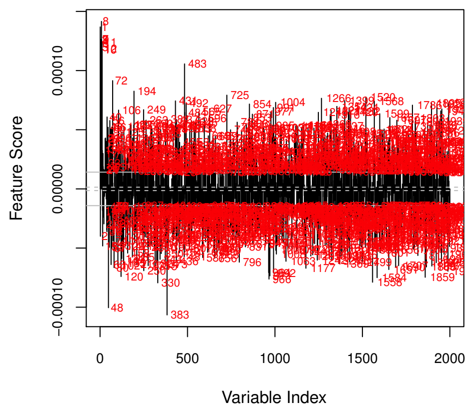


(a) Random forest Feature scores.

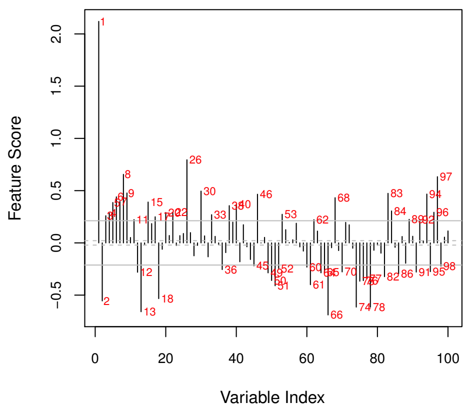


(b) Bayesglm (with Gaussian prior) feature scores.

Figure D.5: Random forest and Bayesglm (with Gaussian prior) feature scores on dataset for investigating the choice of heaviness of heavy-tailed priors in Section 4.4. The red features are the significant features selected by Random forest and Bayesglm.

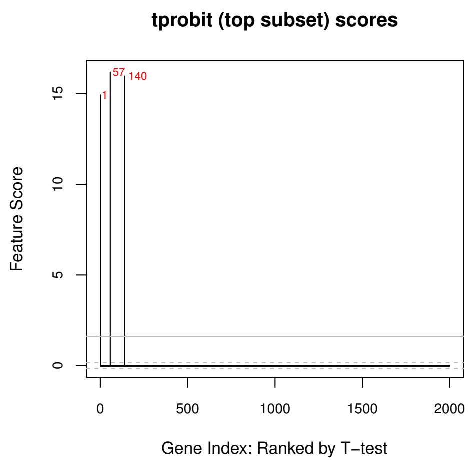


(a) Bayesglm (with a Student's *t* prior) feature scores.

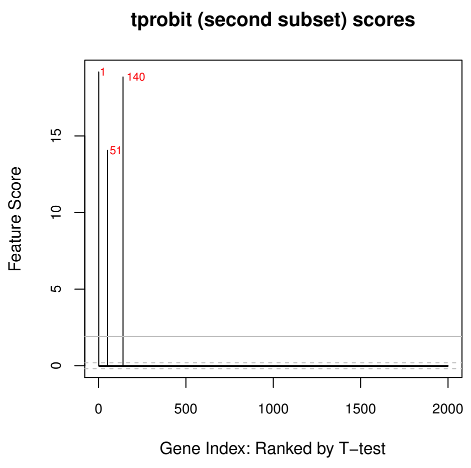


(b) Bayesglm (with Top 100 Features) feature scores.

Figure D.6: Bayesglm (with a Student's *t* prior) and Bayesglm (with Top 100 Features) feature scores on dataset for investigating the choice of heaviness of heavy-tailed priors in Section 4.4. The red features are the significant features selected by Bayesglm.

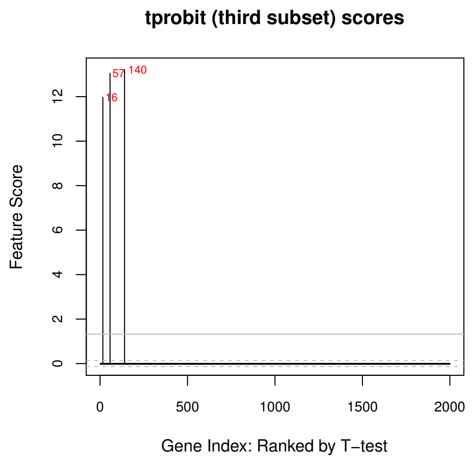


(a) Feature scores for subset (1,57,140)

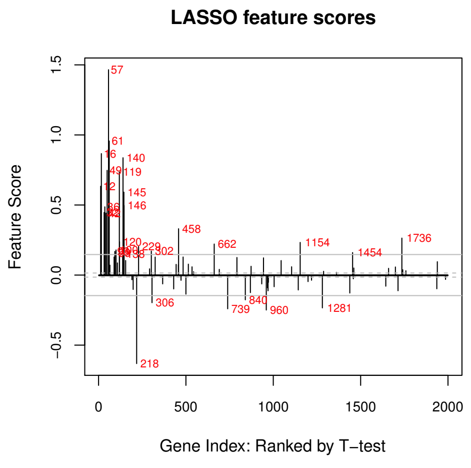


(b) Feature scores for subset (1,51,140)

Figure D.7: T-probit regression feature scores for the top feature subset (1,57,140) and second (1,51,140) based on datasets with independent groups of features in Section 4.2.

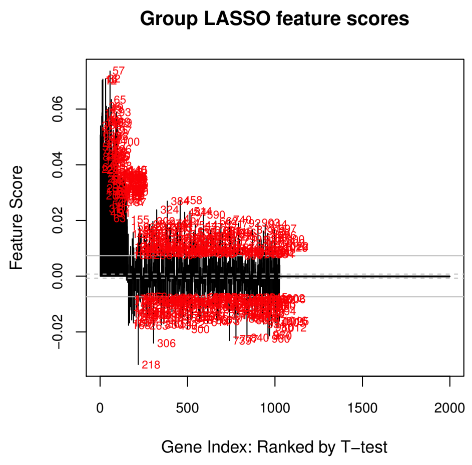


(a) Feature scores for subset (16,57,140)

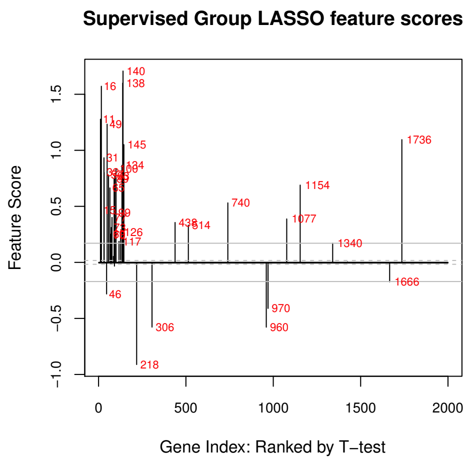


(b) LASSO feature scores

Figure D.8: Third feature subset (16,57,140) and LASSO feature scores based on datasets with independent groups of features in Section 4.2.

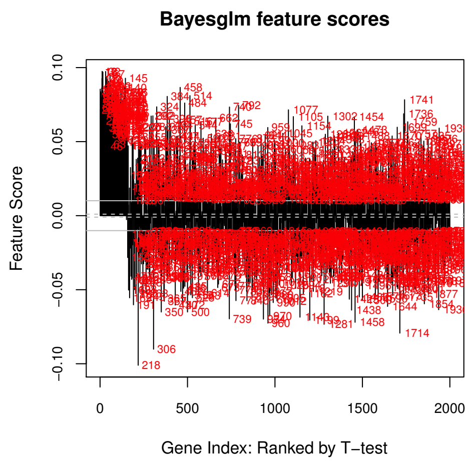


(a) Group LASSO feature scores

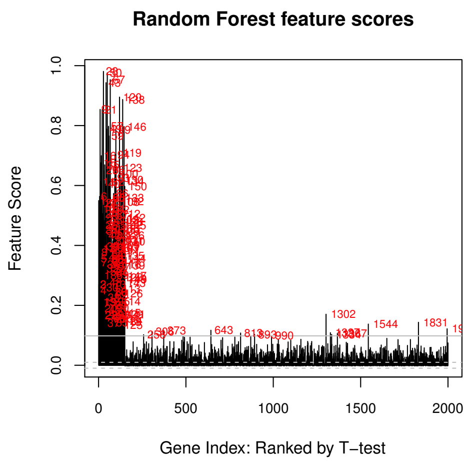


(b) Supervised Group LASSO feature scores

Figure D.9: Group LASSO and Supervised Group LASSO feature scores based on datasets with independent groups of features in Section 4.2.

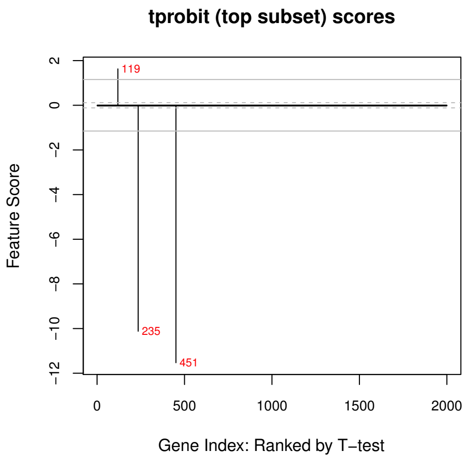


(a) Bayesglm feature scores

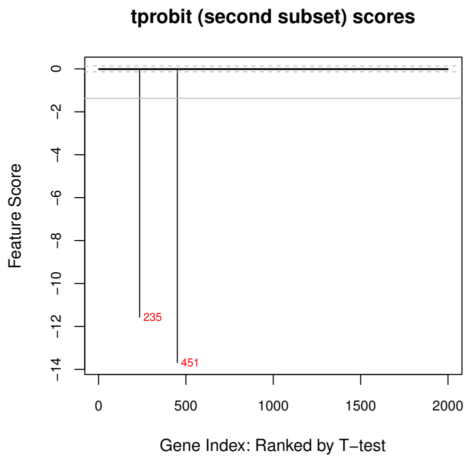


(b) Random forest feature scores

Figure D.10: Bayesglm and random forest feature scores based on datasets with independent groups of features in Section 4.2.

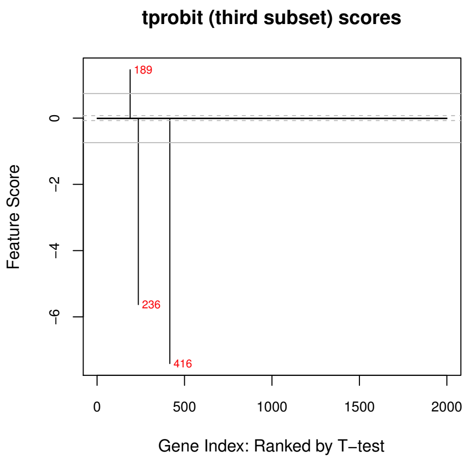


(a) Feature scores for subset (119,235,451)

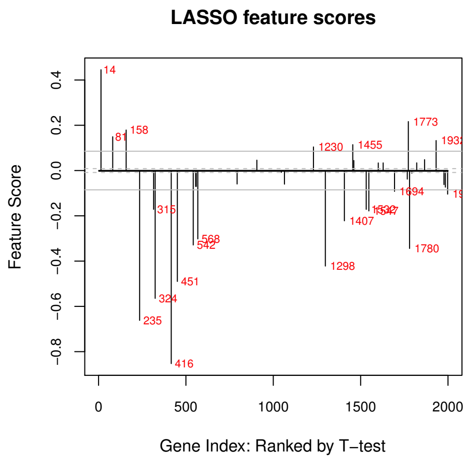


(b) Feature scores for subset (235,451)

Figure D.11: T-probit regression feature scores for the top feature subset (119,235,451) and second (235,451) based on datasets with independent and correlated groups in Section 4.3.

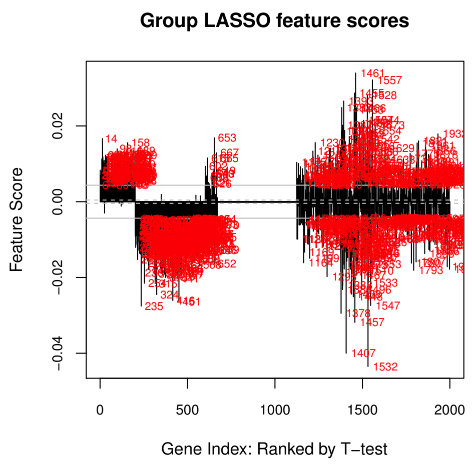


(a) Feature scores for subset (189,236,416)

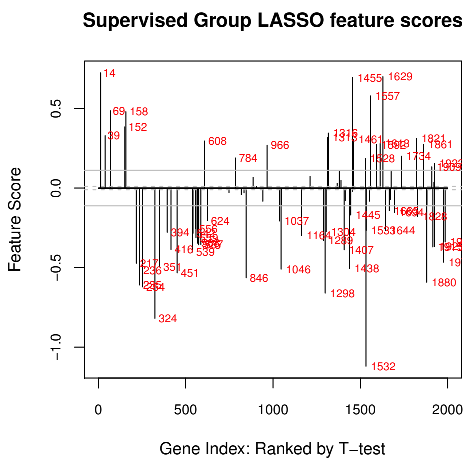


(b) LASSO feature scores

Figure D.12: Third feature subset (189,236,416) and LASSO feature scores based on datasets with independent and correlated groups in Section 4.3.

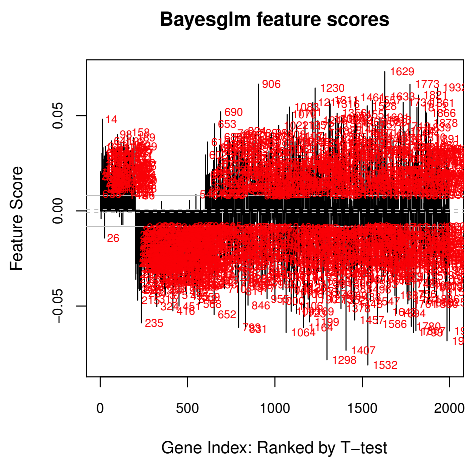


(a) Group LASSO feature scores

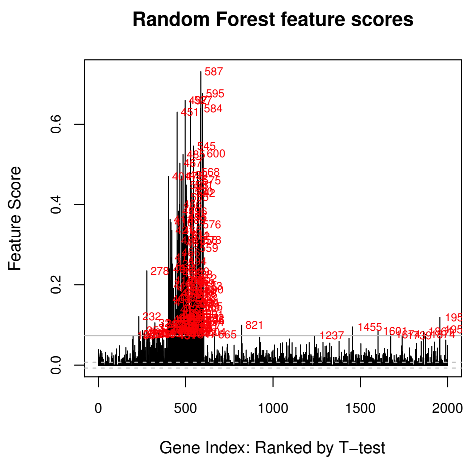


(b) Supervised Group LASSO feature scores

Figure D.13: Group LASSO and Supervised Group LASSO feature scores based on datasets with independent and correlated groups in Section 4.3.



(a) Bayesglm feature scores



(b) Random forest feature scores

Figure D.14: Bayesglm and random forest feature scores based on datasets with independent and correlated groups in Section 4.3.

Appendix E

Supplementary Tables and Figures for Breast Cancer Data

	fsubsets	amlp	er	auc
1	25	0.46	21.00	0.87
2	25,266	0.35	12.00	0.93
3	25,266,614	0.27	10.00	0.95
4	25,266,614,67	0.18	6.00	0.98
5	25,266,614,67,1980	0.15	5.00	0.98
6	25,266,614,67,1980,2081	0.12	6.00	0.99
7	25,266,614,67,1980,2081,2256	0.08	2.00	1.00
8	25,266,614,67,1980,2081,2256,3009	0.08	3.00	1.00
9	25,266,614,67,1980,2081,2256,3009,49	0.08	3.00	1.00
10	25,266,614,67,1980,2081,2256,3009,49,69	0.08	5.00	1.00

Table E.1: LASSO: feature subset and corresponding predictive performance based on breast cancer data in Section 5.1.

	fsubsets	amlp	er	auc
1	2256	0.63	33.00	0.70
2	2256,1795	0.58	35.00	0.75
3	2256,1795,266	0.52	21.00	0.82
4	2256,1795,266,3009	0.46	17.00	0.87
5	2256,1795,266,3009,2762	0.41	17.00	0.89
6	2256,1795,266,3009,2762,2081	0.29	13.00	0.95
7	2256,1795,266,3009,2762,2081,2819	0.28	16.00	0.95
8	2256,1795,266,3009,2762,2081,2819,2539	0.22	12.00	0.97
9	2256,1795,266,3009,2762,2081,2819,2539,4832	0.21	12.00	0.97
10	2256,1795,266,3009,2762,2081,2819,2539,4832,1980	0.21	10.00	0.97

Table E.2: Group LASSO: feature subset and corresponding predictive performance based on breast cancer data in Section 5.1.

	fsubsets	amlp	er	auc
1	266	0.56	25.00	0.78
2	266,2256	0.55	21.00	0.81
3	266,2256,1756	0.51	25.00	0.83
4	266,2256,1756,23	0.41	18.00	0.89
5	266,2256,1756,23,1980	0.36	14.00	0.92
6	266,2256,1756,23,1980,19	0.22	10.00	0.97
7	266,2256,1756,23,1980,19,2855	0.28	10.00	0.96
8	266,2256,1756,23,1980,19,2855,67	0.23	8.00	0.97
9	266,2256,1756,23,1980,19,2855,67,413	0.14	6.00	0.99
10	266,2256,1756,23,1980,19,2855,67,413,49	0.12	7.00	0.99

Table E.3: Supervised Group LASSO: feature subset and corresponding predictive performance based on breast cancer data in Section 5.1.

	fsubsets	amlp	er	auc
1	10	0.37	16.00	0.90
2	10,8	0.34	11.00	0.91
3	10,8,103	0.32	13.00	0.93
4	10,8,103,3	0.32	11.00	0.93
5	10,8,103,3,31	0.32	10.00	0.92
6	10,8,103,3,31,26	0.33	10.00	0.93
7	10,8,103,3,31,26,30	0.32	10.00	0.93
8	10,8,103,3,31,26,30,1223	0.33	10.00	0.92
9	10,8,103,3,31,26,30,1223,18	0.30	9.00	0.93
10	10,8,103,3,31,26,30,1223,18,57	0.32	10.00	0.93

Table E.4: Random Forest: feature subset and corresponding predictive performance based on breast cancer data in Section 5.1.

	fsubsets	amlp	er	auc
1	1	0.33	18.00	0.93
2	1,2256	0.30	14.00	0.94
3	1,2256,4832	0.27	12.00	0.95
4	1,2256,4832,3009	0.27	12.00	0.96
5	1,2256,4832,3009,1980	0.26	12.00	0.96
6	1,2256,4832,3009,1980,2855	0.24	12.00	0.96
7	1,2256,4832,3009,1980,2855,266	0.17	7.00	0.98
8	1,2256,4832,3009,1980,2855,266,2819	0.18	6.00	0.98
9	1,2256,4832,3009,1980,2855,266,2819,4955	0.17	6.00	0.98
10	1,2256,4832,3009,1980,2855,266,2819,4955,1795	0.16	4.00	0.98

Table E.5: Bayesglm: feature subset and corresponding predictive performance based on breast cancer data in Section 5.1.

	fsubsets	amlp	er	auc
1	1	0.33	18.00	0.93
2	1,2	0.29	13.00	0.94
3	1,2,3	0.29	11.00	0.93
4	1,2,3,4	0.27	12.00	0.94
5	1,2,3,4,5	0.27	12.00	0.95
6	1,2,3,4,5,6	0.26	10.00	0.95
7	1,2,3,4,5,6,7	0.26	11.00	0.95
8	1,2,3,4,5,6,7,8	0.27	11.00	0.95
9	1,2,3,4,5,6,7,8,9	0.26	9.00	0.95
10	1,2,3,4,5,6,7,8,9,10	0.26	8.00	0.95

Table E.6: *t*-test ranking: feature subset and corresponding predictive performance based on breast cancer data in Section 5.1.

Appendix F

Supplementary Tables and Figures for Childhood Acute Leukemia Data

	fsubsets	amlp	er	auc
1	32	0.06	2.00	1.00
2	32,35	0.03	1.00	1.00
3	32,35,1	0.02	0.00	1.00
4	32,35,1,27	0.02	0.00	1.00
5	32,35,1,27,115	0.02	0.00	1.00
6	32,35,1,27,115,76	0.02	0.00	1.00
7	32,35,1,27,115,76,6	0.01	0.00	1.00
8	32,35,1,27,115,76,6,197	0.01	0.00	1.00
9	32,35,1,27,115,76,6,197,325	0.01	0.00	1.00
10	32,35,1,27,115,76,6,197,325,2032	0.01	0.00	1.00

Table F.1: LASSO: feature subset and corresponding predictive performance based on childhood acute leukemia data in Section 5.2.

	fsubsets	amlp	er	auc
1	35	0.19	6.00	0.93
2	35,115	0.15	4.00	0.95
3	35,115,1698	0.16	5.00	0.96
4	35,115,545,1698	Inf	4.00	0.95
5	35,115,545,1698,1927	Inf	4.00	0.95
6	35,115,410,545,1698,1927	Inf	4.00	0.95
7	35,115,410,545,1193,1698,1927	Inf	6.00	0.94
8	35,115,410,545,1193,1698,1927,2490	Inf	7.00	0.97
9	35,115,410,545,554,1193,1698,1927,2490	Inf	5.00	0.97
10	35,115,406,410,545,554,1193,1698,1927,2490	1.91	8.00	0.99

Table F.2: Group LASSO: feature subset and corresponding predictive performance based on childhood acute leukemia data in Section 5.2.

	fsubsets	amlp	er	auc
1	115	0.19	6.00	0.95
2	115,35	0.13	4.00	0.96
3	115,35,27	0.07	3.00	1.00
4	115,35,27,32	0.02	0.00	1.00
5	115,35,27,32,406	0.02	0.00	1.00
6	115,35,27,32,406,30	0.02	0.00	1.00
7	115,35,27,32,406,30,40	0.02	0.00	1.00
8	115,35,27,32,406,30,40,2032	0.01	0.00	1.00
9	115,35,27,32,406,30,40,2032,325	0.01	0.00	1.00
10	115,35,27,32,406,30,40,2032,325,2490	0.01	0.00	1.00

Table F.3: Supervised Group LASSO: feature subset and corresponding predictive performance based on childhood acute leukemia data in Section 5.2.

	fsubsets	amlp	er	auc
1	36	0.09	2.00	0.99
2	36,28	0.07	4.00	1.00
3	36,28,30	0.04	2.00	1.00
4	36,28,30,32	0.04	1.00	1.00
5	36,28,30,32,29	0.04	2.00	1.00
6	36,28,30,32,29,37	0.04	2.00	1.00
7	36,28,30,32,29,37,34	0.04	3.00	1.00
8	36,28,30,32,29,37,34,227	0.03	1.00	1.00
9	36,28,30,32,29,37,34,227,115	0.03	1.00	1.00
10	36,28,30,32,29,37,34,227,115,47	0.03	1.00	1.00

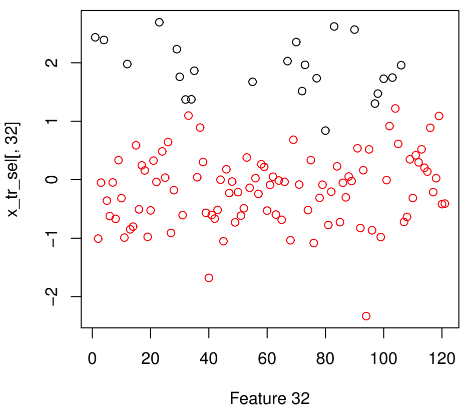
Table F.4: Random Forest: feature subset and corresponding predictive performance based on childhood acute leukemia data in Section 5.2.

	fsubsets	amlp	er	auc
1	1	0.21	11.00	0.96
2	1,5794	0.20	12.00	0.96
3	1,5794,1671	0.14	8.00	0.98
4	1,5794,1671,4099	0.15	10.00	0.98
5	1,5794,1671,4099,792	0.13	8.00	0.97
6	1,5794,1671,4099,792,115	0.06	3.00	1.00
7	1,5794,1671,4099,792,115,1184	0.06	3.00	1.00
8	1,5794,1671,4099,792,115,1184,2492	0.06	3.00	1.00
9	1,5794,1671,4099,792,115,1184,2492,5437	0.05	3.00	1.00
10	1,5794,1671,4099,792,115,1184,2492,5437,555	0.03	1.00	1.00

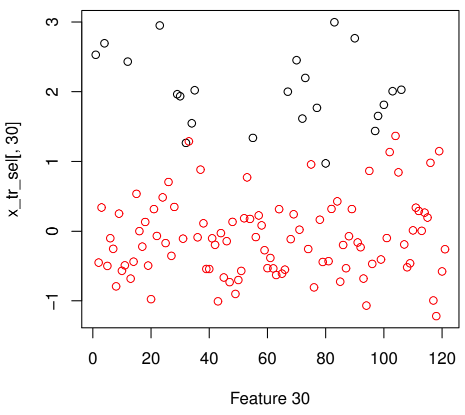
Table F.5: Bayesglm: feature subset and corresponding predictive performance based on childhood acute leukemia data in Section 5.2.

	fsubsets	amlp	er	auc
1	1	0.21	11.00	0.96
2	1,2	0.18	8.00	0.97
3	1,2,3	0.14	7.00	0.98
4	1,2,3,4	0.14	8.00	0.98
5	1,2,3,4,5	0.14	9.00	0.98
6	1,2,3,4,5,6	0.10	6.00	0.99
7	1,2,3,4,5,6,7	0.10	6.00	0.99
8	1,2,3,4,5,6,7,8	0.10	6.00	0.99
9	1,2,3,4,5,6,7,8,9	0.08	4.00	0.99
10	1,2,3,4,5,6,7,8,9,10	0.07	4.00	1.00

Table F.6: *t*-test ranking: feature subset and corresponding predictive performance based on childhood acute leukemia data in Section 5.2.

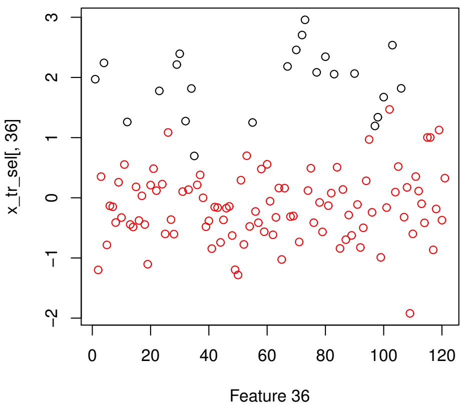


(a) Scatterplot of feature 32 between groups

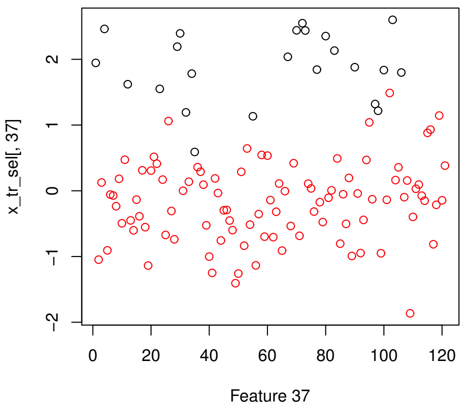


(b) Scatterplot of feature 30 between groups

Figure F.1: Scatterplot of values of feature 32 and feature 30 based on childhood acute leukemia data in Section 5.2.

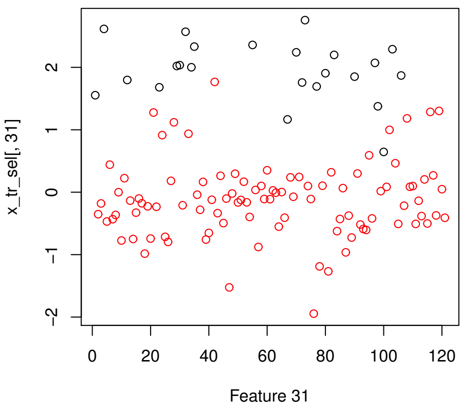


(a) Scatterplot of feature 36 between groups

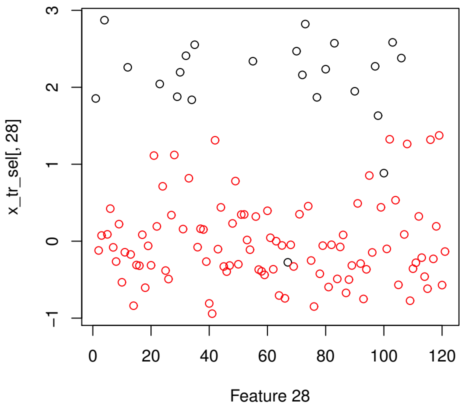


(b) Scatterplot of feature 37 between groups

Figure F.2: Scatterplot of values of feature 36 and feature 37 based on childhood acute leukemia data in Section 5.2.

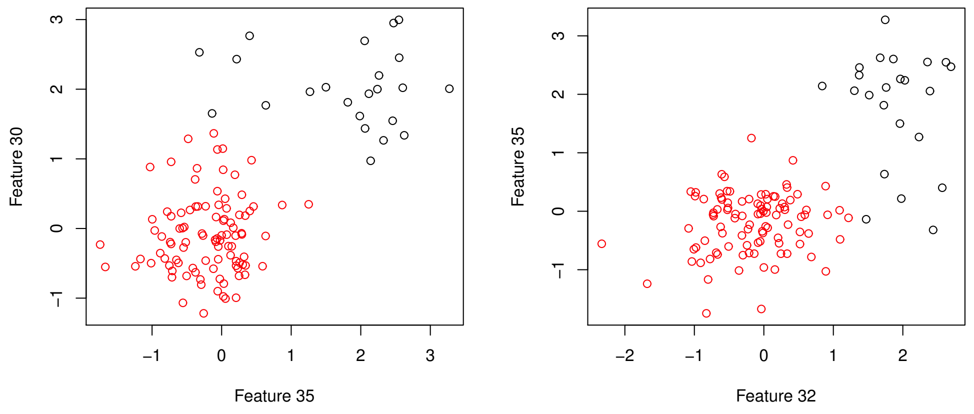


(a) Scatterplot of feature 31 between groups



(b) Scatterplot of feature 28 between groups

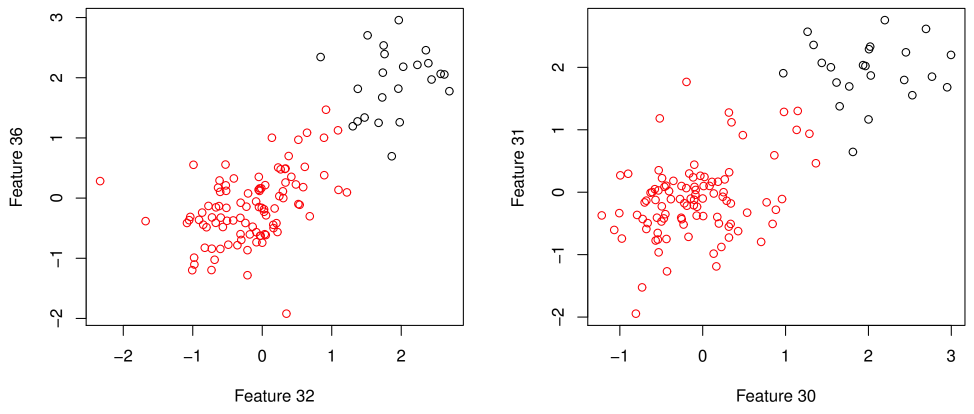
Figure F.3: Scatterplot of values of feature 31 and feature 28 based on childhood acute leukemia data in Section 5.2.



(a) Scatterplot of feature 35 and feature 30 between groups

(b) Scatterplot of feature 32 and feature 35 between groups

Figure F.4: Scatterplot of feature group (35,30) and feature group (32,35) based on childhood acute leukemia data in Section 5.2.



(a) Scatterplot of feature 32 and feature 36 between groups

(b) Scatterplot of feature 30 and feature 31 between groups

Figure F.5: Scatterplot of feature group (32,36) and feature group (30,31) based on childhood acute leukemia data in Section 5.2.

Appendix G

Supplementary Tables and Figures for Colon Cancer Status Data

	fsubsets	amlp	er	auc
1	70	0.33	12.00	0.90
2	70,31	0.05	2.00	1.00
3	70,31,58	0.03	0.00	1.00
4	70,31,58,665	0.03	0.00	1.00
5	70,31,58,665,844	0.02	0.00	1.00
6	70,31,58,665,844,14	0.02	0.00	1.00
7	70,31,58,665,844,14,8	0.02	0.00	1.00
8	70,31,58,665,844,14,8,5	0.01	0.00	1.00
9	70,31,58,665,844,14,8,5,21	0.01	0.00	1.00
10	70,31,58,665,844,14,8,5,21,24	0.01	0.00	1.00

Table G.1: LASSO: feature subset and predictive performance based on colon cancer status data in Section 5.3.

	fsubsets	amlp	er	auc
1	58	0.32	11.00	0.94
2	58,665	0.27	9.00	0.95
3	58,665,110	0.27	9.00	0.96
4	58,665,110,68	0.30	10.00	0.95
5	58,665,110,68,374	0.32	9.00	0.95
6	58,665,110,68,374,73	0.25	6.00	0.95
7	58,665,110,68,374,73,893	0.27	9.00	0.95
8	58,665,110,68,374,73,893,31	0.24	8.00	0.96
9	58,665,110,68,374,73,893,31,1656	0.23	8.00	0.96
10	58,665,110,68,374,73,893,31,1656,1636	0.22	7.00	0.97

Table G.2: Group LASSO: feature subset and predictive performance based on colon cancer status data in Section 5.3.

	fsubsets	amlp	er	auc
1	58	0.32	11.00	0.94
2	58,1	0.21	8.00	0.97
3	58,1,31	0.18	6.00	0.98
4	58,1,31,665	0.17	6.00	0.98
5	58,1,31,665,5	0.07	3.00	1.00
6	58,1,31,665,5,1589	0.06	3.00	1.00
7	58,1,31,665,5,1589,3	0.07	2.00	0.99
8	58,1,31,665,5,1589,3,837	0.08	3.00	0.99
9	58,1,31,665,5,1589,3,837,68	0.12	3.00	0.99
10	58,1,31,665,5,1589,3,837,68,652	0.11	3.00	0.99

Table G.3: Supervised Group LASSO: feature subset and predictive performance based on colon cancer status data in Section 5.3.

	fsubsets	amlp	er	auc
1	4	0.13	5.00	0.99
2	4,54	0.11	4.00	0.99
3	4,54,67	0.09	2.00	0.99
4	4,54,67,31	0.06	2.00	1.00
5	4,54,67,31,100	0.06	2.00	1.00
6	4,54,67,31,100,344	0.05	2.00	1.00
7	4,54,67,31,100,344,61	0.03	0.00	1.00
8	4,54,67,31,100,344,61,35	0.03	0.00	1.00
9	4,54,67,31,100,344,61,35,20	0.04	0.00	1.00
10	4,54,67,31,100,344,61,35,20,188	0.04	0.00	1.00

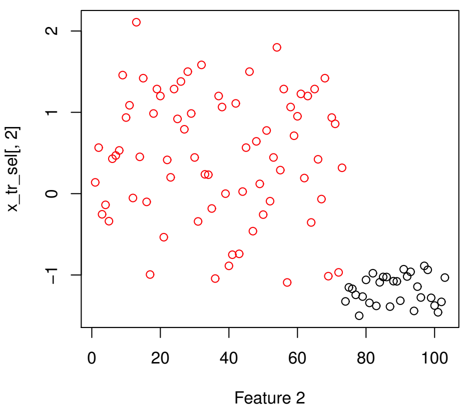
Table G.4: Random Forest: feature subset and predictive performance based on colon cancer status data in Section 5.3.

	fsubsets	amlp	er	auc
1	1	0.25	11.00	0.95
2	1,3743	0.26	10.00	0.95
3	1,3743,844	0.22	10.00	0.97
4	1,3743,844,2666	0.21	8.00	0.96
5	1,3743,844,2666,1636	0.10	3.00	0.99
6	1,3743,844,2666,1636,3899	0.05	1.00	1.00
7	1,3743,844,2666,1636,3899,2810	0.04	1.00	1.00
8	1,3743,844,2666,1636,3899,2810,925	0.04	1.00	1.00
9	1,3743,844,2666,1636,3899,2810,925,3831	0.04	1.00	1.00
10	1,3743,844,2666,1636,3899,2810,925,3831,2499	0.03	1.00	1.00

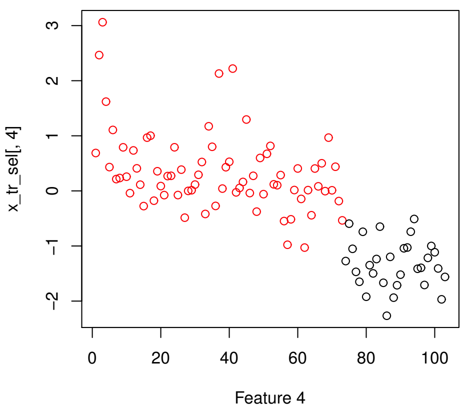
Table G.5: Bayesglm: feature subset and predictive performance based on colon cancer status data in Section 5.3.

	fsubsets	amlp	er	auc
1	1	0.25	11.00	0.95
2	1,2	0.14	4.00	0.98
3	1,2,3	0.15	5.00	0.98
4	1,2,3,4	0.10	4.00	0.99
5	1,2,3,4,5	0.11	2.00	0.99
6	1,2,3,4,5,6	0.12	3.00	0.99
7	1,2,3,4,5,6,7	0.11	3.00	0.99
8	1,2,3,4,5,6,7,8	0.10	2.00	0.99
9	1,2,3,4,5,6,7,8,9	0.10	3.00	0.99
10	1,2,3,4,5,6,7,8,9,10	0.10	3.00	0.99

Table G.6: *t*-test ranking: feature subset and predictive performance based on colon cancer status data in Section 5.3.

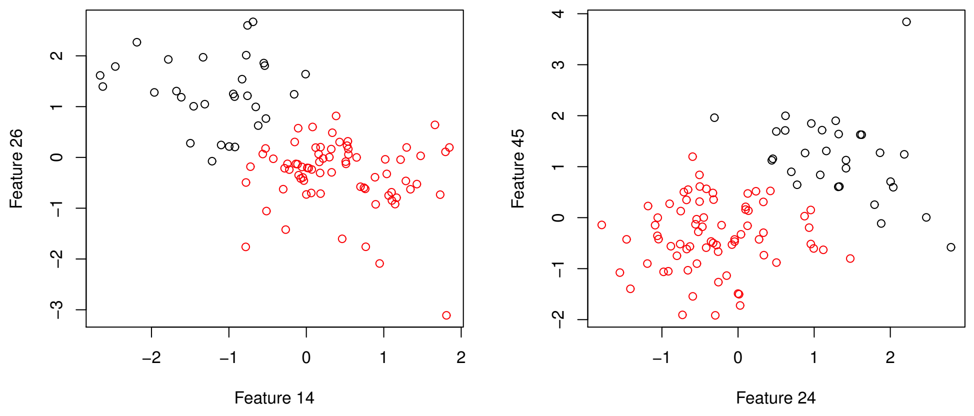


(a) Scatterplot of feature 2 between groups



(b) Scatterplot of feature 4 between groups

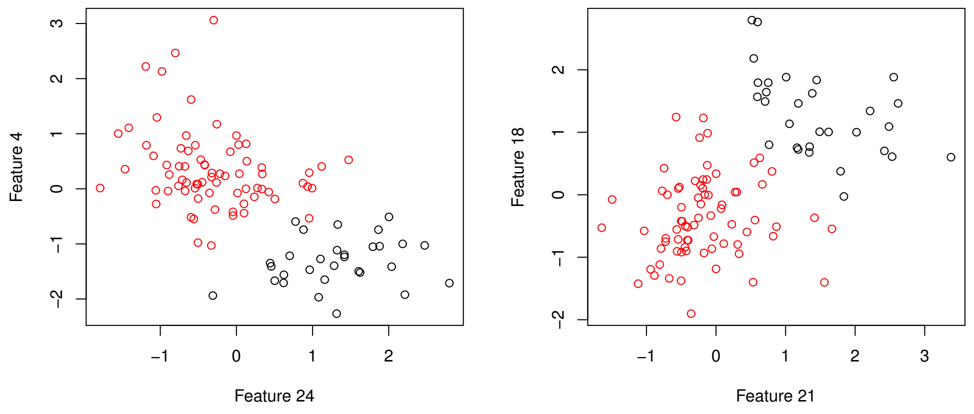
Figure G.1: Scatterplot of values of feature 2 and feature 4 based on colon cancer status data in Section 5.3.



(a) Scatterplot of feature 14 and feature 26 between groups

(b) Scatterplot of feature 24 and feature 45 between groups

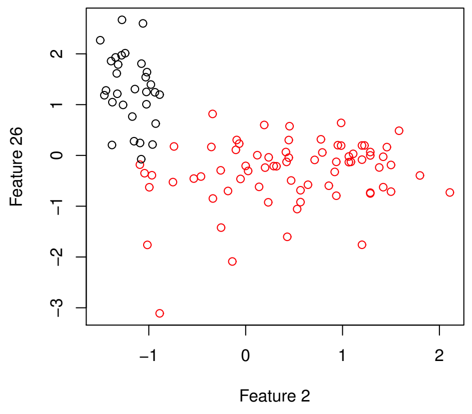
Figure G.2: Scatterplot of feature group (14,26) and feature group (24,45) based on colon cancer status data in Section 5.3.



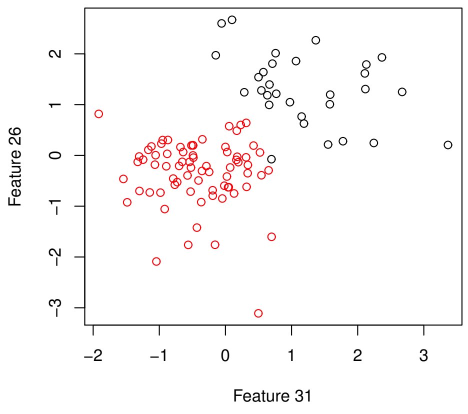
(a) Scatterplot of feature 24 and feature 4 between groups

(b) Scatterplot of feature 21 and feature 18 between groups

Figure G.3: Scatterplot of feature group (4,24) and feature group (18,21) based on colon cancer status data in Section 5.3.

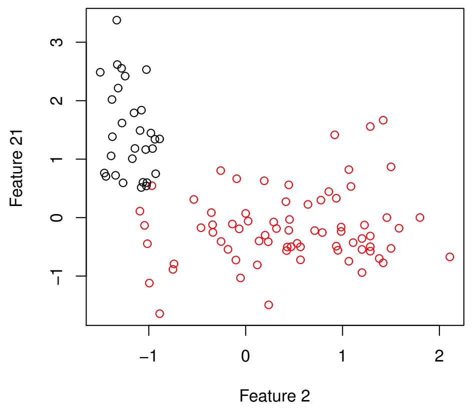


(a) Scatterplot of feature 2 and feature 26 between groups

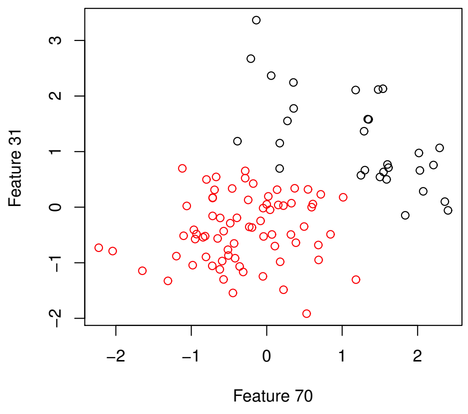


(b) Scatterplot of feature 31 and feature 26 between groups

Figure G.4: Scatterplot of feature group (2,26) and feature group (26,31) based on colon cancer status data in Section 5.3.



(a) Scatterplot of feature 2 and feature 21 between groups



(b) Scatterplot of feature 31 and feature 70 between groups

Figure G.5: Scatterplot of feature group (2,21) and feature group (31,70) based on colon cancer status data in Section 5.3.

References

Abramowitz, M. and Stegun, I. A. (1972), *Handbook of Mathematical Functions*, Dover publications.

Alder, B. J. and Wainwright, T. (1959), "Studies in molecular dynamics. I. General method," *The Journal of Chemical Physics*, 31, 459–466.

Alon, U., Barkai, N., Notterman, D. A., Gish, K., Ybarra, S., Mack, D., and Levine, A. J. (1999), "Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays," *Proceedings of the National Academy of Sciences*, 96, 6745–6750.

Andersen, H. C. (1980), "Molecular dynamics simulations at constant pressure and/or temperature," *The Journal of Chemical Physics*, 72, 2384–2393.

Anderson, N. L. and Anderson, N. G. (2005), "Proteome and proteomics: new technologies, new concepts, and new words," *Electrophoresis*, 19, 1853–1861.

Andersson, A., Ritz, C., Lindgren, D., Edén, P., Lassen, C., Heldrup, J., Olofsson, T., Råde, J., Fontes, M., Porwit-Macdonald, A., et al. (2007), "Microarray-based classification of a consecutive series of 121 childhood acute leukemias: prediction of leukemic and genetic subtype as well as of minimal residual disease status," *Leukemia*, 21, 1198–1203.

Andrews, D. and Mallows, C. (1974), "Scale mixtures of normal distributions," *Journal of the Royal Statistical Society. Series B (Methodological)*, 36, 99–102.

Armagan, A., Dunson, D. B., and Lee, J. (2013), "Generalized double Pareto shrinkage," *Statistica Sinica*, 23, 119.

Barbieri, M. M. and Berger, J. O. (2004), "Optimal predictive model selection," *Annals of Statistics*, 870–897.

Ben-Hur, A., Horn, D., Siegelmann, H., and Vapnik, V. (2002), "Support vector clustering," *The Journal of Machine Learning Research*, 2, 125–137.

Bhattacharya, A., Pati, D., Pillai, N. S., and Dunson, D. B. (2012), "Bayesian shrinkage," *arXiv preprint arXiv:1212.6088*.

Bishop, J. F. (1999), "Adult acute myeloid leukaemia: update on treatment." *The Medical Journal of Australia*, 170, 39–43.

Boser, B., Guyon, I., and Vapnik, V. (1992), "A training algorithm for optimal margin classifiers," in *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, ACM, pp. 144–152.

Bottolo, L. and Richardson, S. (2010), "Evolutionary stochastic search for Bayesian model exploration," *Bayesian Analysis*, 5, 583–618.

Breiman, L. (2001), "Random forests," *Machine Learning*, 45, 5–32.

Brown, P. J., Vannucci, M., and Fearn, T. (1998), "Multivariate Bayesian variable selection and prediction," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 60, 627641.

Bureau, A., Dupuis, J., Falls, K., Lunetta, K. L., Hayward, B., Keith, T. P., and Van Eerdewegh, P. (2005), "Identifying SNPs predictive of phenotype using random forests," *Genetic Epidemiology*, 28, 171–182.

Caron, F. and Doucet, A. (2008), "Sparse Bayesian nonparametric regression," in *Proceedings of the 25th international conference on Machine learning*, ACM, p. 8895.

Carvalho, C. M., Polson, N. G., and Scott, J. G. (2009), "Handling sparsity via the horseshoe," *Journal of Machine Learning Research*, 5, 73–80.

— (2010), "The horseshoe estimator for sparse signals," *Biometrika*, 97, 465–465.

Chang, T.-W. (1983), "Binding of cells to matrixes of distinct antibodies coated on solid surface," *Journal of Immunological Methods*, 65, 217–223.

Cho, R. J., Campbell, M. J., Winzeler, E. A., Steinmetz, L., Conway, A., Wodicka, L., Wolfsberg, T. G., Gabrielian, A. E., Landsman, D., Lockhart, D. J., et al. (1998), "A genome-wide transcriptional analysis of the mitotic cell cycle," *Molecular Cell*, 2, 65–73.

Clarke, R., Ressom, H. W., Wang, A., Xuan, J., Liu, M. C., Gehan, E. A., and Wang, Y. (2008), "The properties of high-dimensional data spaces: implications for exploring gene and protein expression data," *Nature Reviews Cancer*, 8, 37–49.

Cortes, C. and Vapnik, V. (1995), "Support-vector networks," *Machine Learning*, 20, 273–297.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977), "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, 1–38.

Denison, D. G. (2002), *Bayesian methods for nonlinear classification and regression*, vol. 386, John Wiley & Sons.

Denison, D. G. T., Holmes, C. C., Mallick, B. K., and Smith, A. F. M. (2002), *Bayesian methods for nonlinear classification and regression*, Wiley Series in Probability and Statistics.

Díaz-Uriarte, R. and De Andres, S. A. (2006), "Gene selection and classification of microarray data using random forest," *BMC Bioinformatics*, 7, 3.

Donoho, D., Johnstone, I., Kerkyacharian, G., and Picard, D. (1995), "Wavelet shrinkage: asymptopia?" *Journal of the Royal Statistical Society. Series B (Methodological)*, 57, 301–369.

Duane, S., Kennedy, A. D., Pendleton, B. J., and Roweth, D. (1987), "Hybrid monte carlo," *Physics Letters B*, 195, 216–222.

Dudoit, S., Fridlyand, J., and Speed, T. (2002), "Comparison of discrimination methods for the classification of tumors using gene expression data," *Journal of the American Statistical Association*, 97, 77–87.

Efron, B., Hastie, T., Johnstone, I., Tibshirani, R., et al. (2004), "Least angle regression," *The Annals of Statistics*, 32, 407–499.

Fan, J. and Li, R. (2001), "Variable Selection via Nonconcave Penalized Likelihood and its Oracle Properties," *Journal of the American Statistical Association*, 96, 1348–1360.

Fan, L. I. and Zhang, N. R. (2010), "Bayesian variable selection in structured high-dimensional covariate spaces with applications in genomics," *JASA Theory and Methods*, 105, 1202–1214.

Firth, D. (1993), "Bias reduction of maximum likelihood estimates," *Biometrika*, 80, 27–38.

Forgy, E. W. (1965), "Cluster analysis of multivariate data: efficiency versus interpretability of classifications," *Biometrics*, 21, 768–769.

Gelman, A. (2006), "Prior distributions for variance parameters in hierarchical models," *Bayesian Analysis*, 1, 515–533.

Gelman, A., Jakulin, A., Pittau, M. G., and Su, Y. (2008), "A weakly informative default prior distribution for logistic and other regression models," *The Annals of Applied Statistics*, 2, 1360–1383.

Gelman, A. and Rubin, D. B. (1992), "Inference from iterative simulation using multiple sequences," *Statistical Science*, 457–472.

Geman, S. and Geman, D. (1984), "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 721–741.

Genkin, A., Lewis, D. D., and Madigan, D. (2007), "Large-scale Bayesian logistic regression for text categorization," *Technometrics*, 49, 291–304.

Gentleman, R., Carey, V., Huber, W., Irizarry, R., and Dudoit, S. (2006), *Bioinformatics and computational biology solutions using R and Bioconductor*, Springer Science & Business Media.

George, E. I. and McCulloch, R. E. (1993), "Variable selection via Gibbs sampling," *Journal of the American Statistical Association*, 88, 881–889.

Geweke, J. et al. (1991), *Evaluating the accuracy of sampling-based approaches to the calculation of posterior moments*, vol. 196, Federal Reserve Bank of Minneapolis, Research Department.

Gilks, W. R. (2005), *Markov chain monte carlo*, Wiley Online Library.

Grade, M., Hörmann, P., Becker, S., Hummon, A. B., Wangsa, D., Varma, S., Simon, R., Liersch, T., Becker, H., Difilippantonio, M. J., et al. (2007), "Gene expression profiling reveals a massive, aneuploidy-dependent transcriptional deregulation and distinct differences between lymph node–negative and lymph node–positive colon carcinomas," *Cancer Research*, 67, 41–56.

Griffin, J. E. and Brown, P. J. (2010), "Inference with Normal-Gamma prior distributions in regression problems," *Bayesian Analysis*, 5, 171–188.

— (2011), "Bayesian Hyper-Lassos with Non-Convex Penalization," *Australian & New Zealand Journal of Statistics*, 53, 423–442.

Guan, Y. and Stephens, M. (2011), "Bayesian variable selection regression for genome-wide association studies and other large-scale problems," *The Annals of Applied Statistics*, 5, 1780–1815.

Gygil, S. P., Rist, B., Gerber, S. A., Turecek, F., Gelb, M. H., and Aebersold, R. (1999), "Quantitative analysis of complex protein mixtures using isotope-coded affinity tags," *Nature Biotechnology*, 17, 994–999.

Hans, C., Dobra, A., and West, M. (2007), "Shotgun stochastic search for large p regression," *Journal of the American Statistical Association*, 102, 507–516.

Hartigan, J. A. and Wong, M. A. (1979), "Algorithm AS 136: A k-means clustering algorithm," *Applied Statistics*, 100–108.

Ho, T. K. (1995), "Random decision forests," in *Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on*, IEEE, vol. 1, pp. 278–282.

Hoerl, A. and Kennard, R. (1970), "Ridge regression: applications to nonorthogonal problems," *Technometrics*, 12, 69–82.

Hoeting, J., Madigan, D., Raftery, A., and Volinsky, C. (1999), "Bayesian model averaging: a tutorial (with comments by M. Clyde, David Draper and EI George, and a rejoinder by the authors," *Statistical Science*, 14, 382–417.

Holland, P. W. and Welsch, R. E. (1977), "Robust regression using iteratively reweighted least-squares," *Communications in Statistics-Theory and Methods*, 6, 813–827.

Homan, M. D. and Gelman, A. (2014), "The no-U-turn sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo," *The Journal of Machine Learning Research*, 15, 1593–1623.

Ishwaran, H. and Rao, J. S. (2005a), "Spike and slab gene selection for multigroup microarray data," *Journal of the American Statistical Association*, 100, 764–780.

— (2005b), "Spike and slab gene selection for multigroup microarray data," *Journal of the American Statistical Association*, 100, 764–780.

— (2005c), "Spike and slab variable selection: frequentist and Bayesian strategies," *The Annals of Statistics*, 33, 730–773.

Jerónimo, C., Henrique, R., Hoque, M. O., Mambo, E., Ribeiro, F. R., Varzim, G., Oliveira, J., Teixeira, M. R., Lopes, C., and Sidransky, D. (2004), "A quantitative promoter methylation profile of prostate cancer," *Clinical Cancer Research*, 10, 8472–8478.

Johnson, S. C. (1967), "Hierarchical clustering schemes," *Psychometrika*, 32, 241–254.

Kim, Y., Kim, J., and Kim, Y. (2006), "Blockwise sparse regression," *Statistica Sinica*, 16, 375.

Krishnapuram, B., Carin, L., Figueiredo, M. A., and Hartemink, A. J. (2005), "Sparse multinomial logistic regression: Fast algorithms and generalization bounds," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27, 957–968.

Kyung, M., Gill, J., Ghosh, M., and Casella, G. (2010), "Penalized Regression, Standard Errors, and Bayesian Lassos," *Bayesian Analysis*, 5, 369–412.

Lamnisos, D., Griffin, J. E., and Steel, M. F. (2012), "Cross-validation prior choice in Bayesian probit regression with many covariates," *Statistics and Computing*, 22, 359–373.

Lange, K. L., Little, R. J., and Taylor, J. M. (1989), "Robust statistical modeling using the t distribution," *Journal of the American Statistical Association*, 84, 881–896.

Lee, K., Sha, N., Dougherty, E., Vannucci, M., and Mallick, B. (2003), "Gene selection: a Bayesian variable selection approach," *Bioinformatics*, 19, 90.

Li, F. and Zhang, N. R. (2010), "Bayesian variable selection in structured high-dimensional covariate spaces with applications in genomics," *Journal of the American Statistical Association*, 105.

Li, L. and Yao, W. (2014), "Fully Bayesian Logistic Regression with Hyper-Lasso Priors for High-dimensional Feature Selection," *arXiv*, 1405.3319 [stat].

Lindley, D. V. (1957), "A statistical paradox," *Biometrika*, 44, 187–187.

Liu, C. (2004), "Robit regression: A simple robust alternative to logistic and probit regression," in *Applied Bayesian Modeling and Causal Inference from Incomplete-Data Perspectives*, John Wiley & Sons, Ltd, pp. 227–238.

Lunetta, K. L., Hayward, L. B., Segal, J., and Van Eerdewegh, P. (2004), "Screening large-scale association study data: exploiting interactions using random forests," *BMC Genetics*, 5, 32.

Lunn, D., Spiegelhalter, D., Thomas, A., and Best, N. (2009), "The BUGS project: Evolution, critique and future directions," *Statistics in Medicine*, 28, 3049–3067.

Ma, S., Song, X., and Huang, J. (2007), "Supervised group Lasso with applications to microarray data analysis," *BMC Bioinformatics*, 8, 60.

MacQueen, J. et al. (1967), "Some methods for classification and analysis of multivariate observations," in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, California, USA, vol. 1, pp. 281–297.

Marcotte, E., Pellegrini, M., Ng, H., Rice, D., Yeates, T., and Eisenberg, D. (1999), "Detecting protein function and protein-protein interactions from genome sequences," *Science*, 285, 751.

Meier, L. (2009), "grpLASSO: Fitting user specified models with Group Lasso penalty (2009)," *R Package Version 0.4-2*.

Meier, L., Van De Geer, S., and Bühlmann, P. (2008), "The group LASSO for logistic regression," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70, 53–71.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953), "Equation of State Calculations by Fast Computing Machines," *The Journal of Chemical Physics*, 21, 1087–1092.

Michalak, P. (2008), "Coexpression, coregulation, and cofunctionality of neighboring genes in eukaryotic genomes," *Genomics*, 91, 243–248.

Mitchell, T. J. and Beauchamp, J. J. (1988), "Bayesian variable selection in linear regression," *Journal of the American Statistical Association*, 1023–1032.

Murtagh, F. (1984), "Complexities of hierarchic clustering algorithms: State of the art," *Computational Statistics Quarterly*, 1, 101–113.

Neal, R. (2011), "MCMC Using Hamiltonian Dynamics," in *Handbook of Markov Chain Monte Carlo*, CRC Press, pp. 113–162.

Neal, R. M. (1995), "Bayesian learning for neural networks," Ph.D. thesis, University of Toronto.

Norris, J. R. (1998), *Markov chains*, Cambridge University Press.

Organization, W. H. (2014), "World Cancer Report 2014." *World Cancer Report*, 1.1.

Paik, S., Shak, S., Tang, G., Kim, C., Baker, J., Cronin, M., Baehner, F. L., Walker, M. G., Watson, D., Park, T., et al. (2004), "A multigene assay to predict recurrence of tamoxifen-treated, node-negative breast cancer," *New England Journal of Medicine*, 351, 2817–2826.

Park, M. Y., Hastie, T., and Tibshirani, R. (2007), "Averaged gene expressions for regression," *Biostatistics*, 8, 212–227.

Pepe, M. S. (2000), "Receiver operating characteristic methodology," *Journal of the American Statistical Association*, 95, 308–311.

Plummer, M. et al. (2003), "JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling," in *Proceedings of the 3rd International Workshop on Distributed Statistical Computing*, Vienna, vol. 124, p. 125.

Polson, N. G. and Scott, J. G. (2010), "Shrink globally, act locally: Sparse Bayesian regularization and prediction," *Bayesian Statistics*, 9, 501538.

— (2012a), "Good, great, or lucky? Screening for firms with sustained superior performance using heavy-tailed priors," *The Annals of Applied Statistics*, 6, 161–185, zentralblatt MATH identifier1235.91144, Mathematical Reviews number (MathSciNet) MR2951533.

— (2012b), "Local shrinkage rules, Levy processes and regularized regression," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 74, 287311.

— (2012c), "On the half-Cauchy prior for a global scale parameter," *Bayesian Analysis*, 7, 887902.

Raftery, A. E. and Lewis, S. M. (1992), "[Practical Markov Chain Monte Carlo]: comment: one long run with diagnostics: implementation strategies for Markov Chain Monte Carlo," *Statistical Science*, 493–497.

Robert, C. P. (2013), "On the Jeffreys-Lindley's paradox," *ArXiv*, 1303.5973v, 1–13.

Rockova, V. and George, E. I. (2013), "EMVS: The EM Approach to Bayesian Variable Selection," *Journal of the American Statistical Association*, forthcoming.

Ročková, V. and George, E. I. (2014), "Emvs: The em approach to bayesian variable selection," *Journal of the American Statistical Association*, 109, 828–846.

Rokach, L. and Maimon, O. (2005), "Clustering methods," in *Data Mining and Knowledge Discovery Handbook*, Springer, pp. 321–352.

Rossky, P., Doll, J., and Friedman, H. (1978), "Brownian dynamics as smart Monte Carlo simulation," *The Journal of Chemical Physics*, 69, 4628–4633.

Sha, N., Vannucci, M., Tadesse, M. G., Brown, P. J., Dragoni, I., Davies, N., Roberts, T. C., Contestabile, A., Salmon, M., and Buckley, C. (2004), "Bayesian variable selection in multinomial probit models to identify molecular signatures of disease stage," *Biometrics*, 60, 812–819.

Shafer, G. (1982), "Lindley's paradox," *Journal of the American Statistical Association*, 77, 325–334.

Shevade, S. K. and Keerthi, S. S. (2003), "A simple and efficient algorithm for gene selection using sparse logistic regression," *Bioinformatics*, 19, 2246–2253.

Spellman, P. T., Sherlock, G., Zhang, M. Q., Iyer, V. R., Anders, K., Eisen, M. B., Brown, P. O., Botstein, D., and Futcher, B. (1998), "Comprehensive identification of cell cycle–regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization," *Molecular Biology of the Cell*, 9, 3273–3297.

Steinberg, D. and Colla, P. (2009), "CART: classification and regression trees," *The Top Ten Algorithms in Data Mining*, 9, 179.

Struyf, A., Hubert, M., and Rousseeuw, P. (1997), "Clustering in an object-oriented environment," *Journal of Statistical Software*, 1, 1–30.

Tibshirani, R. (1996), "Regression shrinkage and selection via the LASSO," *Journal of the Royal Statistical Society. Series B (Methodological)*, 58, 267–288.

Tibshirani, R., Chu, G., Narasimhan, B., and Li, J. (2011), "SAMR: Significance Analysis of Microarrays," *R Package Version*, 2.

Tibshirani, R., Hastie, T., Narasimhan, B., and Chu, G. (2002), "Diagnosis of multiple cancer types by shrunken centroids of gene expression," *Proceedings of the National Academy of Sciences of the United states of America*, 99, 6567.

Tibshirani, R., Saunders, M., Rosset, S., Zhu, J., and Knight, K. (2005), "Sparsity and smoothness via the fused LASSO," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67, 91–108.

Tibshirani, R., Walther, G., and Hastie, T. (2001), "Estimating the number of clusters in a data set via the gap statistic," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63, 411–423.

Tibshirani, R. et al. (1997), "The LASSO method for variable selection in the Cox model," *Statistics in Medicine*, 16, 385–395.

Tolosi, L. and Lengauer, T. (2011), "Classification with correlated features: unreliability of feature ranking and solution," *Bioinformatics*, 27, 1986–1994.

Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., Botstein, D., and Altman, R. B. (2001), "Missing value estimation methods for DNA microarrays," *Bioinformatics*, 17, 520–525.

Tseng, P. and Yun, S. (2009), "A coordinate gradient descent method for nonsmooth separable minimization," *Mathematical Programming*, 117, 387–423.

Tusher, V. G., Tibshirani, R., and Chu, G. (2001), "Significance analysis of microarrays applied to the ionizing radiation response," *Proceedings of the National Academy of Sciences*, 98, 5116–5121.

van der Pas, S. L., Kleijn, B. J. K., and van der Vaart, A. W. (2014), "The Horseshoe Estimator: Posterior Concentration around Nearly Black Vectors," *arXiv:1404.0202 [math, stat]*.

Vapnik, V. (1995), "The nature of statistical learning," .

Wahba, G. (1990), *Spline models for observational data*, Society for Industrial Mathematics.

Wang, X., Istepanian, R., and Song, Y. (2003), "Microarray image enhancement by denoising using stationary wavelet transform," *NanoBioscience*, 2, 184–189.

Wang, Z., Liu, H., and Zhang, T. (2014), "Optimal computational and statistical rates of convergence for sparse nonconvex learning problems," *Annals of Statistics*, 42, 2164.

Weston, J., Mukherjee, S., Chapelle, O., Pontil, M., Poggio, T., and Vapnik, V. (2001), "Feature selection for SVMs," *Advances in Neural Information Processing Systems*, 668–674.

Wolpert, R. L. et al. (2004), "A Conversation with James O. Berger," *Statistical Science*, 19, 205–218.

Yi, G., Sze, S.-H., and Thon, M. R. (2007), "Identifying clusters of functionally related genes in genomes," *Bioinformatics*, 23, 1053–1060.

Yuan, M. and Lin, Y. (2006), "Model selection and estimation in regression with grouped variables," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68, 49–67.

Zhao, P., Rocha, G., and Yu, B. (2009), "The composite absolute penalties family for grouped and hierarchical variable selection," *The Annals of Statistics*, 3468–3497.

Zorn, C. (2005), "A solution to separation in binary response models," *Political Analysis*, 13, 157–170.

Zou, H. (2006), "The Adaptive LASSO and Its Oracle Properties," *Journal of the American Statistical Association*, 101, 1418–1429.