A Logistic Model Tree based Approach for eQTL Data Prediction Integration

Stefano Beretta[1,2], Mauro Castelli[3], Ivo Gonçalves[3,4], Ivan Kel[2], and Ivan Merelli[2]

(1) DISCo, Univ. degli Studi di Milano-Bicocca, Milan, Italy.
E-mail: stefano.beretta@disco.unimib.it

(2) Ist. di Tecnologie Biomediche, Consiglio Nazionale delle Ricerche, Segrate, Italy.
E-mail: {ivan.merelli,ivan.kel}@itb.cnr.it

(3) NOVA IMS, Universidade Nova de Lisboa, Lisbon, Portugal.
E-mail: {mcastelli,igoncalves}@novaims.unl.pt

(4) CISUC, Department of Informatics Engineering, University of Coimbra, Coimbra, Portugal.

Keywords: eQTL, Evolutionary Algorithm, Genetic Programming.

Abstract. eQTL analysis is an emerging method to establish the impact of genetic variations (such as Single Nucleotide Polymorphisms) on the expression of genes. We propose an approach based on Logistic Model Trees that integrates the predictions of different eQTL mapping tools. The performance of our approach has been assessed by using data from the DREAM5 challenge.

1 Introduction

Due to the importance of their role in a variety of regulatory processes and also in several diseases, Single Nucleotide Polymorphisms (SNPs) are widely studied, with particular attention to their interactions with genes. To this purpose, one of the most used methods to link the expression of genes/proteins to different genotypes is the Expression Quantitative Trait Loci (eQTL) mapping, which studies the impact of SNPs on differential measurable gene transcript levels. More precisely, the goal of eQTL analysis is to identify genomic regions whose genotypes affect the expression of specific genes.

In the last years, a lot of effort has been dedicated to the definition of methods to perform eQTL analysis. Currently, there are different approaches for eQTL mapping. Among them, one of the most used tools is R/QTL [Broman et al., 2003], which employs Hidden Markov Models (HMM) to deal with missing genotype data, and to map quantitative trait loci. Another widely used approach is MatrixEQTL, which has been proposed in [Shabalin, 2012]. One of the key features of this method is that it tests the associations between each possible pair of SNP and transcript, by using two models: linear and ANOVA. In the former model, the effect of genotype is additive and its significance is evaluated using t-statistic, while in the latter the genotype is modelled as a categorical variable and the significance is evaluated using the ANOVA approach. Recently, a tool called mRMR has been proposed in literature [Huang and Cai, 2013]. This machine learning based method tests all the possible types of dependencies, by taking advantage of the Mutual Information (MI), to assess the association between genotypes and gene expression. More precisely, the key point is to consider not only the interaction between a SNP and a gene, but also the redundancy among genes, which is used to detect the indirect associations.

Although the results obtained in several studies with the state-of-art methods are good and reveal interesting aspects, to compare the performance of such methods for eQTL analysis it is necessary to consider different scenarios. The main problem to

address when performing this task is related to the lack of standard reference datasets. This is mainly due to the fact that evaluating the performance on real data is usually not easy [Michaelson et al., 2010]. As a response to this, the DREAM (Dialogue on Reverse Engineering Assessments and Methods) community, whose aim is to assess the performance of different techniques to solve problems coming from the biology field, proposed a challenge for the eQTL mapping, called DREAM5. The goal of one of these challenges (DREAM5 SYSGEN A) was, starting from synthetic genetic variations and gene expression data, to reverse-engineer the interaction networks. This problem simulates the eQTL mapping and, to this purpose, it can be used to evaluate the accuracy of the existing methods and tools. Moreover, after the conclusion of the DREAM5 challenge, the reference networks were released.

In this work, we present an integrative machine learning approach to combine the results obtained by the previously introduced tools for eQTL analysis, based on a supervised learning process. More precisely, we build a system to aggregate the predictions of existing tools. The main objective is to establish if an interaction between a genotype and a gene is likely to be real or not. By taking advantage of the DREAM5 datasets, for which we have the standard reference (i.e. we know, among all the possible pairs of SNP-gene interactions, those that are true), we trained the proposed classifier by taking into account the scores obtained with R/qtl, MatrixEQTL, and mRMR, which are the most used tools for eQTL analysis. From the results obtained from the 3 tools on the DREAM5 challenge data, we could observe that mRMR was the most stringent, computing the lowest number of mapping, while MatrixEQTL (in both the variants) was the one returning the highest number of predictions. About R/qtl, the number of mappings predicted is in between those of the other two methods, and these results have good overlaps with both the others.

As for most classification problems (especially those related to the biological field), the set of elements of a specific class (e.g. of the positive cases) is usually very small in relation to the overall dataset. This is also true for the considered datasets, which contain the possible interactions between genotypes and genes: the real interactions correspond to a small fraction of all the possible combinations. In this scenario, one of the side effects is that the predictions obtained with most of the classifiers are highly influenced by the fact that the classes are not equally balanced.

To overcome this issue, the method we propose works at data level by undersampling the class having the highest number of elements (majority class) in order to balance the number of elements of the other class (minority class). Then, once the dataset has been pre-processed, we employed a supervised learning algorithm to train the classifier which is based on the so called Logistic Model Trees.

Following the idea proposed in [Ackermann et al., 2012], we have assessed the performance of our approach on the 15 datasets proposed in the "DREAM5 SYMGEN A" challenge, by splitting each matrix in such a way that 70% of the training instances was used for training purpose, while the remaining 30% was used to assess the performance of the classifier on unseen instances.

2 Method

One of the main issues that must be addressed before building a classifier is related to the distribution of the data among the classes. A dataset is unbalanced when at least one class is represented by only a small number of training examples (called the minority class) while the other class(es) make up the majority. In this scenario, classifiers can have good accuracy on the majority class(es), but very poor accuracy on the minority class due to the influence that the larger majority class(es) has on traditional training criteria [Ganganwar, 2012]. In fact, most classification algorithms pursue to minimize the error rate: the percentage of the incorrect predictions of class labels. They ignore

the difference between types of misclassification errors. In particular, they implicitly assume that all misclassification errors cost equally. In several real-world applications this assumption may be not true, and, as a consequence, the classifiers built on these unbalanced data may produce unsatisfactory results.

To overcome the problem related to unbalanced datasets, it is possible to distinguish two different approaches [Ganganwar, 2012]: data level techniques and algorithmic level techniques. At the data level, these solutions include many different forms of resampling, while at the algorithmic level, solutions are related to the particular machine learning technique under investigation. Here, we focus on a data level method, which tries to balance the data by undersampling the majority class, that is to select a number of training instances from the majority class equal to the number of training instances in the minority class. A typical resampling method alters the balance of the data so that the change to the distribution skewness results in an improved predictive performance. Considering the nature of the data, it is fundamental to design an effective sampling methodology. The data considered in this work is composed of two classes, in which the majority class covers the 99.8% of the available data, hence a random sampling will probably result in a poor selection of instances. That is, a random sampling will select, with high probability, instances that are not the main representatives of the majority class. For this reason we did not take into consideration the widely used random majority undersampling (RUS) technique, where instances of the majority class are randomly discarded from the dataset. More advanced techniques have been proposed in the literature to perform a more effective undersampling (e.g., [Kubat et al., 1997, Tomek, 1976]).

In our work, we use the method described in [Kubat et al., 1997], called One Side Selection (OSS), which is an undersampling method resulting from the application of Tomek links [Tomek, 1976] followed by the Condensed Nearest Neighbor (CNN) rule. As explained in [Kubat et al., 1997], the idea of this method is that the learner will select a representative subset of the examples in the majority class. To detect the less reliable instances of this class, it is possible to classify the instances in four different categories (see Figure 1): (i) instances that suffer from the class-label noise (points in the bottom left corner of Figure 1), (ii) borderline examples that are close to the boundary between minority and majority regions (these points are not reliable because even a small amount of noise can push the instance to the wrong region); (iii) redundant instances that can be safely removed (points in the upper right corner of Figure 1), and (iv) safe examples that are worth to maintain for future classification tasks.

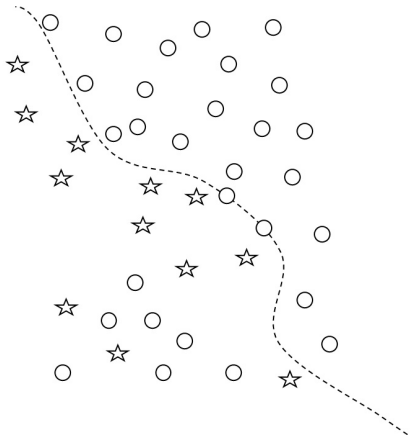

Figure 1: Distribution of instances in a two-classes dataset: stars denote instances in the minority class, circles denote instances in the majority class.

Considering these four cases, an intelligent learner may try to remove the borderline instances as well as the ones suffering from class-noise. Finally, the learner will remove the redundant instances. In particular, Tomek links allow to remove the noisy and bor-

derline examples, while the CNN technique removes examples from the majority class that are far away from the decision border.

Tomek links consist of points that are each others' closest neighbours, but do not share the same class label. More formally, let us assume $\{E_1, \ldots, E_n\} \in [\mathbb{R}]^k$ is a dataset, with each E_i having exactly one of two labels "+" or "-". A pair (E_i, E_j) is called a Tomek link if E_i and E_j have different labels, and there is not an E_l such that $d(E_i, E_l) < d(E_i, E_j)$ or $d(E_j, E_l) < d(E_i, E_j)$, where $d(x, y)$ is the distance between x and y. Regarding the second component of the OSS technique, CNN works by selecting the set S of reference instances (data points that are needed for accurate classification) from the dataset obtained after the Tomek link phase, such that 1-NN with S can classify the examples almost as accurately as 1-NN does with the whole data set.

The complete algorithm and all the related details are outlined in [Kubat et al., 1997].

On the other hand, once the dataset is created by exploiting the previously mentioned technique, a supervised classifier must be trained on it, in order to obtain the final results.

Among the existing methods in the literature to solve supervised learning problems, linear models and tree induction methods have gained popularity in the data mining community, both for the prediction of nominal classes and numeric values. As reported in [Landwehr et al., 2005], the former approaches fit a simple (linear) model to the data, and the process of model fitting is quite stable, resulting in low variance, but potentially high bias. The latter ones exhibit low bias but often high variance: they search a less restricted space of models, allowing it to capture nonlinear patterns in the data, but making it less stable and prone to overfitting. As a consequence, it is not surprising that neither of the two methods is superior in general.

In the last years, work to combine these two schemes into model trees, i.e. trees that contain linear regression functions at the leaves, has been done. More precisely, as reported in [Landwehr et al., 2005], the main idea of model trees is to combine the advantages of tree induction methods and linear models. Hence, model trees rely on simple regression models if only little and/or noisy data is available and add a more complex tree structure if there is enough data to warrant such structure. Using model trees with linear regression functions might not be the best choice when a classification task must be addressed because this approach produces several trees (one per class) and thus makes the final model harder to interpret.

A better solution to tackle classification tasks is to use a combination of a tree structure and logistic regression models resulting in a single tree. Another advantage of using logistic regression is that explicit class probability estimates are produced rather than performing just a classification. Model trees in which the linear regression functions take the form of logistic functions are called Logistic Model Trees (LMT).

In this study, we take into account the LMT introduced in [Landwehr et al., 2005] and the tree induction method described in [Sumner et al., 2005]. The reader is referred to [Landwehr et al., 2005] for a detailed description of the algorithm.

3 Experimental Results

To evaluate the performance obtained by the presented approach, we have used the same data as in [Ackermann et al., 2012]. More precisely, we have considered the 15 datasets from the DREAM5 systems genetics in silico network challenge A, in which the goal is to reconstruct gene-regulatory networks starting from (synthetic) genetic variation and gene expression data. Each of these gene-regulatory networks consists of 1000 genes (corresponding to the nodes), possibly having a different number of edges. Moreover, three different quantity of sample size have been considered: 100, 300 and 999, corresponding to the 3 different sub-challenges SysGenA100, SysGenA300 and SysGenA999, respectively. In the simulation process, the variations were evenly distributed on 20 chromosomes by also considering the possibility of a local linkage be-

tween adjacent positions on the chromosomes. As anticipated, the final goal of the DREAM5 SYSGEN A challenge is to perform an eQTL analysis, that is, to establish the relations in each regulatory network by exploiting the information of the simulated gene expression levels of the 1000 genes and the simulated genotype data.

For each of the 15 datasets taken into account, the OSS sampling technique has been used to produce a balanced dataset. This resulting dataset (called *reduced dataset*) has an equal number of positive and negative instances. Before applying the LMT algorithm, we split the reduced dataset in such a way that 70% of the training instances were used for training purpose, while the remaining 30% was used to assess the performance of the classifier on unseen instances. For each reduced dataset, 30 different partitions between training and test instances have been taken into account. This 30 partitions of the reduced dataset have been created in order to take into account the possible bias introduced by the random sampling.

After the creation of the training and test instances, the LMT algorithm has been executed, considering the implementation provided by the WEKA machine learning tool. In particular, we used the default parameters provided by WEKA, with the exception of the "fastRegression" option that has been used in the experiments we performed. The use of this option includes in the LMT algorithm an heuristic that avoids cross-validating the number of Logit-Boost iterations at every node. When fitting the logistic regression functions at a node, LMT has to determine the number of Logit-Boost iterations to run. Originally, this number was cross-validated at every node in the tree.

Table 1: Classification performance on training and test instances for all the considered datasets. Averages values of positive and negative classes for Precision, Recall and F-measure are reported.

		TRAINING			TEST		
Dataset		Precision	Recall	F-measure	Precision	Recall	F-measure
DREAM5 SysGenA100	Network1	0.8880	0.8705	0.8685	0.8790	0.8660	0.8640
	Network2	0.9760	0.9745	0.9745	0.9760	0.9760	0.9760
	Network3	0.9850	0.9850	0.9850	0.9870	0.9860	0.9860
	Network4	0.9850	0.9850	0.9850	0.9880	0.9880	0.9880
	Network5	0.9810	0.9800	0.9800	0.9830	0.9820	0.9820
DREAM5 SysGenA300	Network1	0.9600	0.9560	0.9560	0.9660	0.9640	0.9640
	Network2	0.9535	0.9485	0.9485	0.9640	0.9620	0.9620
	Network3	0.9580	0.9540	0.9540	0.9650	0.9630	0.9630
	Network4	0.9510	0.9450	0.9450	0.9610	0.9580	0.9580
	Network5	0.9530	0.9475	0.9475	0.9605	0.9580	0.9575
DREAM5 SysGenA999	Network1	0.9750	0.9730	0.9740	0.9770	0.9760	0.9755
	Network2	0.9760	0.9740	0.9740	0.9775	0.9775	0.9775
	Network3	0.9730	0.9720	0.9720	0.9735	0.9730	0.9725
	Network4	0.9345	0.9330	0.9330	0.9150	0.9150	0.9150
	Network5	0.9650	0.9630	0.9630	0.9715	0.9705	0.9700

Results for both training and test instances are obtained considering, for each of the 15 datasets, the median over the 30 runs that have been performed. In particular, we reported, for each of the considered datasets, the following statistics: precision, recall and F-measure. We preferred the median with respect to the average for its higher robustness to outliers. Table 1 summarizes all the results.

As it is possible to see, the proposed system is able to achieve a good classification

performance on both training and test instances, hence showing that the proposed system produces robust classifiers. More in details, all the median values on precision, recall and F-measure obtained on the 15 networks are higher than 0.93 (except for the first network which has values around 0.87). The overall median values on all the computed networks in the training phase are 0.965, 0.963, and 0.963, for precision, recall, and F-measure, respectively. On unseen instances (i.e., test phase) the values are 0.9715, 0.9705, and 0.97, for precision, recall, and F-measure, respectively. These results show the suitability of the proposed method for addressing the problem under exam: in particular, in the large majority of the datasets, training and test performance are comparable. Hence, the method is able not only to extract a model of the data that produces a good classification of the training instances, but it also shows a good generalization ability.

4 Conclusion

In this work we presented an approach, based on logistic model trees, to integrate the predictions of different tools for the eQTL mapping analysis. Results obtained on DREAM5 challenge data are encouraging, showing the suitability of the proposed method for the problem at hand. For this reason, we plan to further extend the analysis on other community shared benchmarks reflecting the complexity of eQTL mapping analysis. Finally, to achieve better tuning of the method, we are also planning to integrate real data into the evaluation procedure.

References

[Ackermann et al., 2012] Ackermann, M., Clment-Ziza, M., Michaelson, J. J., and Beyer, A. (2012). Teamwork: Improved eqtl mapping using combinations of machine learning methods. *PLoS ONE*, 7(7):1–8.

[Broman et al., 2003] Broman, K. W., Wu, H., Sen, a., and Churchill, G. A. (2003). R/qtl: Qtl mapping in experimental crosses. *Bioinformatics*, 19(7):889–890.

[Ganganwar, 2012] Ganganwar, V. (2012). An overview of classification algorithms for imbalanced datasets. *International Journal of Emerging Technology and Advanced Engineering*, 2(4):42–47.

[Huang and Cai, 2013] Huang, T. and Cai, Y.-D. (2013). An information-theoretic machine learning approach to expression qtl analysis. *PLoS ONE*, 8(6):1–9.

[Kubat et al., 1997] Kubat, M., Matwin, S., et al. (1997). Addressing the curse of imbalanced training sets: one-sided selection. In *ICML*, volume 97, pages 179–186. Nashville, USA.

[Landwehr et al., 2005] Landwehr, N., Hall, M., and Frank, E. (2005). Logistic model trees. *Machine Learning*, 59(1):161–205.

[Michaelson et al., 2010] Michaelson, J. J., Alberts, R., Schughart, K., and Beyer, A. (2010). Data-driven assessment of eqtl mapping methods. *BMC Genomics*, 11(1):1–16.

[Shabalin, 2012] Shabalin, A. A. (2012). Matrix eqtl: ultra fast eqtl analysis via large matrix operations. *Bioinformatics*, 28(10):1353–1358.

[Sumner et al., 2005] Sumner, M., Frank, E., and Hall, M. (2005). Speeding up logistic model tree induction. In *Knowledge Discovery in Databases: PKDD 2005*, pages 675–683. Springer.

[Tomek, 1976] Tomek, I. (1976). Two modifications of cnn. *IEEE Transactions on Systems, Man and Cybernetics*, SMC-6(11):769–772.