METHODS

Spiro Stilianoudakis

August 9, 2018

METHODS

THE DATA

Publicly available topologically associating domain data was obtained from GEO with accession GSE53525 for the GM12878 and K562 cell lines. The domain data was constructed from in situ Hi-C contact matrices at a 5kb and 10 kb resolution for the GM12878 and k562 cell lines respectively. The Arrowhead algorithm was used to identify TAD boundaries for a given contact matrix (Rao et. al). Identified TAD boundaries included chromosomal information as well as the genomic coordinates, with each set of coordinates representing a TAD that was formed along the diagonal of a contact matrix. The coordinates were concatenated into a long vector and were sorted. This vector of points represented each individual TAD boundary. Next, the genome was binned into a series of 1kb intervals (flanked by 500 bases on either side of the boundary point) and an indicator vector Y was created based on whether there was a TAD boundary located in the 1kb interval (Y=1) or not (Y=0).

GENOMIC ANNOTATIONS

Annotation data was obtained from the Encyclopedia of DNA Elements (ENCODE) Consortium. These annotations consisted of genomic coordinates and were used to build the feature space, X = {X1, . . ., Xp}, of the subsequent models. The data consisted of genomic coordinates and chromosomal information. For each genomic feature, an indicator variable (1 for yes; 0 for no) was created based on whether the coordinates of the annotation overlapped with the coordinates representing a flanked TAD boundary. Likewise, a continuous variable representing the distance to the center of the nearest flanked TAD boundary was included. Genomic features that were located inside of a flanked TAD boundary were given a distance of 0. Figure X presents an illustration for how the data was constructed.

DATA PIPELINE

The model pipeline that was proposed aimed to alleviate some of the problems that are typical with genomic data and improve prediction. At each stage in the pipeline, Elastic-Net models were performed and performance metrics such as ROC curves and subsequent AUCs were evaluated. The Elastic-Net model was used here because of its ability to be tuned and its computational speed relative to other machine learning algorithms. For each elastic-net model the mixing parameter, alpha, was tuned on a grid of five values (0.1, 0.325, 0.550, 0.775, and 1.0). A sequence of values for the regularization parameter, lambda, was automatically provided by the CARET package in R. The algorithm then chose the optimal combination of alpha and lambda, and the model was evaluated.

Prior to initiating the pipeline, predictors with near-zero variance were removed. To identify said predictors, the nearZeroVar function in the CARET package was used. The function considers two metrics. First, the ratio of the most frequent value over the second most frequent value was calculated. Ideally, this ratio would be close to one for well-behaved predictors. Next, the number of unique values divided by the total number of samples multiplied by 100 was calculated. Here, if the frequency ratio was greater than 0.99 and the percentage of unique values was less than 0.001, the predictor was flagged as having near-zero variance and removed.

CLASS IMBALANCE

To assess the class imbalance problem due to the sparcity of domains throughout the genome, two methods were used. The first consisted of incorporating the SMOTE command from the DmWR package in R. SMOTE stands for Synthetic Minority Over-sampling Technique. The SMOTE algorithm incorporates both under-sampling (percent under) of the majority class and over-sampling (percent over) of the minority class. The minority class is over-sampled by taking each minority class sample and introducing synthetic examples along the line segments joining any/all of the k minority class nearest neighbors (here, k=5). Then, the majority class is under sampled by taking a random sample that matches some pre-specified percentage of the updated minority class. For example, consider a data set with 700 observations, where there are 35 minority classes and 665 majority classes. Now, when using SMOTE, consider a 200 percent over sample of the minority class and a 100 percent under sample of the majority class. This means that two synthetic observations will be created for every minority class (35+70=105) and 1 majority class will be randomly chosen for every new minority class created (70), resulting in a classification of 70/105 (majority/minority). Now consider, a 200/200 combination. The new minority class is still 105 (70 new synthetic observations created), but now

twice as many majority classes are randomly chosen (70*2=140), resulting in a 140/105 classification. It is important to note that using a combination of 100/200 will always result in perfectly balanced classes. A total of eight different combinations of percent over and undersampling were used which included: 100/200, 200/200, 300/200, 400/200, 100/300, 200/300, 300/300, 400/300. Here, the 100/200 combination creates a completely balanced dataset as mentioned before, while the others create an unbalanced dataset with the "No" class as the majority to reflect the original data. The second method involved iteratively taking 100 bootstrap samples of the majority class to match the number in the minority class. For each sample, an elastic-net model was tuned and evaluated. Then row wise means of the sensitivities, specificities, and AUCs of each of the 100 bootstrap samples were taken to evaluate the models as a whole. The SMOTE combination with the best performance was then compared to the bootstrap method. ROC curves were plotted and corresponding AUCs were presented in a table.

NORMALIZATION

In order to correct for the skewness of the distances, elastic-net models were evaluated based on a combination of log transformed and/or standardized predictors. A total of four elastic net models were performed including: with log transformation and standardization, without log transformation and with standardization, with log transformation and without standardization, and without log transformation and without standardization. One hundred bootstrap samples were taken from the majority class to create a balanced classification data. ROC curves were plotted and corresponding AUCs were presented in a table. Likewise, we looked at the variable importance quality of the models.

VARIABLE SELECTION

Due to the large number of genomic features included in the model, different variable selection techniques were evaluated in order to both improve computational speed and eliminate noise. There were three different wrapper methods that were used here, forward, backward, and stepwise selection. A null model consisting of just an intercept term and a full model, including all features, were specified for each selection technique. For all three, a logistic regression model was performed, and the AIC of the fitted model was used to establish which variables to include. That is, only a reduction in the model's AIC contributed to the inclusion of a feature in the model. Additionally, a recursive feature elimination (RFE) algorithm was performed. In RFE, a full model was created, a measure of variable importance was then computed that ranked the predictors from the most important to the least. Here, the importance of the model was based on the random forest

importance criterion. Then, at each stage of the search, the least important predictors were iteratively eliminated prior to rebuilding the model. The process was continued for a pre-specified subset of predictors.

FINAL MODEL

Once the pipeline produced the appropriate dataset, a random forest algorithm was used to assess which genomic features that were most predictive of the development of TAD boundaries. The algorithm was run iteratively over 100 bootstrap samples, each time producing sensitivities, specificities, AUCs, and variable importances. For each iteration, the model was tuned on the best number of predictors to subset as well as the optimal number of trees to use. At the end of the 100 iterations, row wise means were performed to aggregate the model performances. We then compared the random forest model to both the MLR model and the MLR model with LASSO estimated coefficients that was proposed by Mourad et al.