Brief Summary of Output from Summer 2019
Yuri Ahuja
9/22/2019

Important New Scripts

1. ensemble_classifer2.R: Implements classifier ensembling LASSO-regularized logistic regression and random forest
2. underflow.R: Workbook for experiments regarding dealing with underflow in MBX data
3. simulateData.R: Simulates datasets assuming a naive correlation structure in X and user-specified correlation structure in beta, used to evaluate LASSO Gauss Copula
4. lgp.py: Implements LASSO Gauss Copula regressor

Summary of Projects

1. LASSO/Random Forest Ensemble
    a. *Motivation:* Averaging several predictors with similar bias and sufficiently non-overlapping decision functions improves variance and by extension MSE.
    b. *Experiments:* Ran LASSO-regularized logistic regression and random forest on a variety of IBD and colorectal cancer datasets, predicting as many classes as possible simultaneously. Evaluated classification accuracy and mean AUC of ensemble predictions computed with a variety of relative weightings of LASSO vs. relative weighting (ensemble = alpha*LASSO + (1-alpha)*RF, alpha = {0.1,0.2,...,1.0}).
    c. *Code:* ensemble_classifier2.R
    d. *Results:* Ensembling almost always outperforms individual predictors, on average performing best for alpha = 0.5.
        i. See “ensemble_{disease}_{evaluation metric}.pdf” files for plots of performance on {disease} as a function of alpha
        ii. See “ensemble_summary_aucs.pdf” file for a summary of all above plots on a single plot. Note that an alpha of 0.5 (arithmetic mean between classifiers) performs roughly optimal on average.
    e. *Conclusion:* Use LASSO/Random Forest ensemble with alpha = 0.5.
2. Underflow
    a. *Motivation:* Develop a robust method to predict the probability that a sample with a non-zero count of a given bacterium would register as zero were the extraction/sequencing protocol repeated.
    b. *Experiments:* Fitted sequencing- and sampling-level replicate counts data to Multinomials as well as overdispersed/more flexible derivatives thereof, including Dirichlet-Multinomials, Negative Binomials, and pairwise Beta-Binomials. Evaluated relative fits of first three by likelihood ratio test, AIC, and BIC. Evaluated whether held-out predictions of P(0) concur with actual proportions of replicates that = 0.
    c. *Code:* underflow.R

d. *Results:* Dirichlet-Multinomial fits significantly better than Multinomial or Negative Binomial models as assessed by LRT or AIC/BIC. Still does not seem to allow enough flexibility to reflect differences in overdispersion between bacteria.
    i. See “DM_alpha_vs_propNonZero.png” for plot of the fitted Dirichlet-Multinomial alpha parameter for a given bacterium as a function of the proportion of its reads not equal to 0. Note that the fitted alpha seems to increase at a rate of approx. 0.2 x Prob(Read =/= 0). This suggests that an alpha on the order of 0.1 is an appropriate default for bacteria with observed values of 0 (though in practice this finding is irrelevant since we only care about fitting and potentially underflowing non-zero-count bacteria.
    ii. See “MN_expected_vs_observed_variance.png” and “DM_expected_vs_observed_variance.png” for plots of observed variance of different bacteria versus what would be expected under the Multinomial and Dirichlet-Multinomial models respectively (note that both axes are log-scaled). Note that the Multinomial is underdispersed, and that the degree of underdispersion increases with prevalence of the bacterium. Moreover, note that the Dirichlet-Multinomial is overdispersed, and the degree of overdispersion decreases with prevalence of the bacterium. Thus, whereas the Multinomial significantly underestimates variance for more-prevalent bacteria (and would therefore underestimate P(0)), the Dirichlet-Multinomial significantly overestimates variance for less-prevalent bacteria (and would therefore overestimate P(0)). This suggests that a more flexible overdispersed model might be warranted to better reflect bacteria-level overdispersion.

e. *Conclusion:* Fit Beta-Binomials to replicate data for each individual bacterium to estimate scale parameters for each bacterium. Fit standard Binomials to non-overdispersed bacteria (Beta-Binomial should fail). Fit Dirichlet-Multinomial to dataset as a whole to estimate scale parameter for bacteria with all zeros in replicate data (cannot be fitted). For new sample, fit bacterium-specific Beta-Binomial (or Binomial if deemed non-overdispersed in replicate fit), keep fitted mean parameter, and set scale parameter to replicate experiment estimator. Predict probability of 0 given fitted model and set observation to 0 (i.e. underflow) if P(0) exceeds some predefined threshold (5%?).

3. <u>LASSO Gauss Copula</u>
    a. *Motivation:* LASSO implicitly assumes independence among beta regression coefficients. We may be able to mitigate bias and thus improve predictive accuracy by incorporating prior knowledge of the expected correlation structure between betas.
    b. *Experiments:* Implemented an inference procedure for the LASSO Gauss Copula model. Evaluated in simulated datasets with high degrees of (pre-specified) inter-beta correlation whether the Copula fit (with intra-beta correlation matrix set to the pre-specified values used to simulate) better approximates the true beta

vector than does the traditional LASSO fit. Finally, evaluated on Crohns vs. Healthy kmer data from the Pascal dataset whether the Copula fit (with intra-beta correlation matrix set to the empirical correlation matrix of the corresponding kmer features (X)) achieves higher AUC than the traditional LASSO fit.

c. *Code:* simulateData.R to simulate data; lgp.py to run copula
d. *Results:* For simulated datasets with high intra-beta correlation (cor(beta_k,beta_{k+1}) = 0.9), the Copula estimator approximates the true beta significantly better than the traditional LASSO estimator (correlation with true beta of ~0.35 for LASSO vs. ~0.70 for Copula). However, this does not translate into a significant improvement in predictive AUC: both models achieve AUCS ~0.90-0.91. As of writing this document (8/22 4pm), the Copula hasn't finished running on the first of 5 lambdas to test after 40 hrs on subway, so it would seem that computational optimization will be necessary to make this practically feasible.
e. *Conclusion:* Need to optimize for computational tractability. Performs iterative matrix operations that may be infeasible for 4096-dimensional feature spaces. Regardless, given the lack of significant improvement in predictive AUC on simulated datasets despite significant improvement in actual beta approximation, it is unlikely that the Copula will improve predictive accuracy over traditional LASSO.

4. <u>Batch Effect (Negative Results)</u>
    a. *Motivation:* Classifiers trained on one data source (i.e. Pascal) are not at all generalizable to another (i.e. Amerigut).
    b. *Experiments:* 1) Tried standardizing feature sets by subtracting off mean feature vectors and dividing individual features by their standard deviations. Did not help with generalizability (i.e. train on one set, evaluate on another) at all. 2) Tried projecting each dataset onto principal component axes identified by PCA on combined datasets. Again did not help with generalizability at all. Hypothesis behind both experiments was that the disease signal (i.e. vector between mean *disease* feature vector and mean *control* feature vector) remains constant across datasets, but that the datasets themselves are just shifted and scaled by constants, thus limiting generalizability. Turns out this is not the case - the disease signal vectors for different datasets (i.e. Pascal vs. Amerigut) are roughly orthogonal, suggesting that there is little to no shared information between datasets.
    c. *Code:* Did not upload to github since its just a mess. Let me know if you want it!
    d. *Results:* All negative.
    e. *Conclusion:* Could not overcome batch effect.
5. <u>Other Classifier Experiments (Negative Results)</u>
    a. *Motivation:* To improve the classifier in any way possible.
    b. *Experiments:* Tried cross-validation-optimized 1) Ridge-regularized logistic regression, 2) random forest, 3) k nearest neighbors, 4) PCA-LASSO regression, and 5) relaxed LASSO using both unregularized and Ridge-regularized logistic

regression for the secondary regression step, on a variety of IBD and colorectal cancer datasets.

c. *Code:* Did not upload to github since its just a mess. Let me know if you want it!
d. *Results:* Random forest performed comparably to LASSO-regularized logistic regression across datasets. See section 1 of this document for discussion of the ensemble LASSO/Random Forest classifier. All other methods performed significantly worse than LASSO. Ensembling k nearest neighbors with LASSO and LASSO/Random Forest also hurt predictive accuracy.
e. *Conclusion:* Go with LASSO/Random Forest Ensemble classifier.