XGB-EN Ensemble Auto-Annotation of Bio Cell Types

Machine Learning Final Project – STAT454-556

University of Victoria

Dayten Sheffar, Hanan Abousaleh, Leno Rocha
Department of Mathematics and Statistics

Keywords: Cell Type Auto-Annotation, Ensemble, XGBoost, Elastic Net 16th April, 2021

Abstract

In the context of single-cell RNA-sequencing data, we develop a method to classify cell types within a particular sample, which is called 'annotation'. We first feature-reduced a given dataset using gene expression profiles and a disease status marker of patients as predictors. We tuned an XGBoost model, feeding these hyperparameters into a cross-validated classification model using XGBoost as a first round guess. If the probability of the cell type prediction was not an order of magnitude larger than the next best guess for cell type, these low certainty classifications were fed into an elastic net model to enhance performance. A benchmark elastic net classifier with a F1-score of 0.98 was outperformed by our ensemble method with an F1-score of 0.986, found to be statistically significant. Results are tabulated and visualized.

1 Introduction

The development of statistical methods and software for analyzing large volumes of data generated from single biological cell sequencing technology is a topic receiving a lot of attention. It is crucial to determine which cell-types exist within a particular sample. This is called 'annotating' the cell-types, and this process is the first step of data analysis for many single-cell RNA-sequencing data. Doing so manually is tedious and prone to error, hence automation for this problem has been a hot research area. For this project, a machine learning classification method is developed to annotate cell-types, using the gene expression profiles and the disease statuses of patients as predictors.

The aim of the project is to build an algorithm that achieves a better performance than a benchmark elastic net (EN) model. We conduct ten replicates of 5-fold cross validation using the same fold indices as the benchmark, so that comparability is ensured. We have achieved this goal using an ensemble of XGBoost and elastic net methods (XGB-EN). A criterion is used to distinguish high-certainty classifications from low ones for the purpose of re-evaluating those observations and either confirming or correcting them through EN predictions. This paper will present our data processing for feature selection, a description of our algorithm and the threshold criteria mentioned above, as well as our comparisons to the benchmark EN results. Our discussions and conclusions follow after, along with references.

2 Method

Feature Reduction

We took the full dataset provided to us, with 4524 observations and 1212 columns (including the response), and performed feature selection to reduce it down to 897 columns. To perform the feature selection, we considered each of the five cell-types one at a time, calling the type "class 1" and assigning the class the value of 1. Each of the remaining cell-types were also selected one at a time, named "class 2" and assigned the value 0. This yielded the $\binom{5}{2} = 10$ combinations: {(1,2), (1,3), (1,4), (1,5), (2,3), (2,4), (2,5), (3,4), (3,5), (4,5)}. For each row-subset of the dataframe corresponding to the pairs of cell-types, we performed a Mann-Whitney U test with continuity correction between each predictor (column) and the provided cell-types (now labelled 1 and 0). This yields 1211 p-values per such comparison. The result is a table with 1211 rows

for the predictors, and 10 columns for the cell-type comparisons. The table is filled with p-values indicating how well a given predictor predicts a given combination of cell-types. Sometimes this method of subsetting resulted in a p-value of zero if a given column in the row-subset by two cell-types had no gene expressions (a constant column of zeroes). To skirt this issue, we assigned any zero p-values a value greater than 1, so that they would never be selected as a minimum value (corresponding to the best performance of a predictor).

Namely, by taking the minimum per row of this p-values table, this allowed us to find the best performance of a given predictor across cell-types. We then ranked all these 1211 columns by their row-wise minimum p-values (i.e., best performance overall), and selected the top 896, which corresponded to a p-value cutoff of exactly 0.05. Of interest is the fact that the condition type column 'condt' is precisely the last predictor included at a 0.05 cutoff. Therefore, all predictors included in the reduced data set are as good as or better predictors than the condition associated with the underlying cell being tested (0 for control or 1 for diseased). A chart of the p-values is seen in Figure 1, where the left side shows all the non-zero p-values (with zeroes being re-assigned as greater than 1), and the right graph shows only those below the 0.05 threshold.

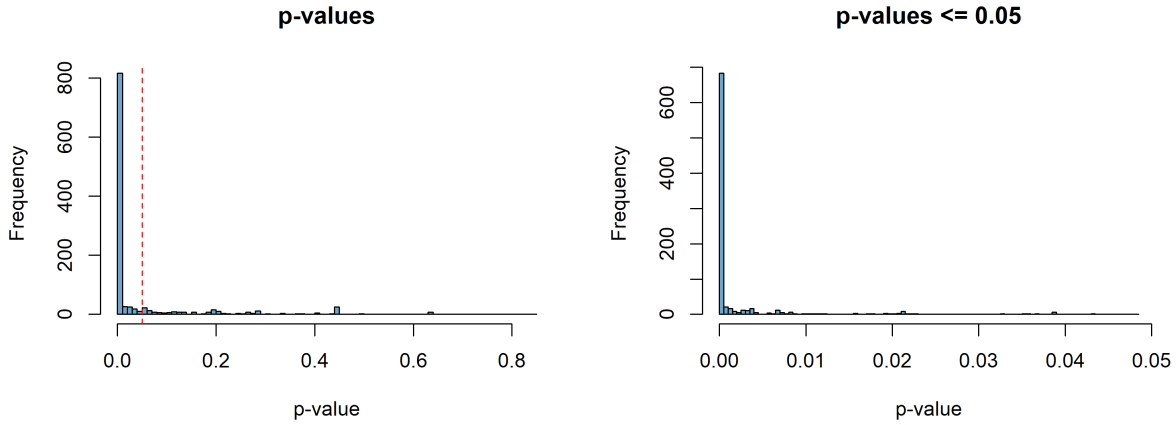

Fig. 1. Mann-Whitney U-test resulting distribution for feature selection. (Left: All p-values included. Right: only p-values ≤ 0.05 included). Note the amount of p-values significantly greater than 0.05.

The XGBoost approach

XGBoost, or simply XGB, is an optimized distributed gradient boosting library in R. It implements machine learning algorithms under the Gradient Boosting framework[1,2] and provides tree computations in parallel. The XGB package was created in 2014 and gained popularity and attention as the algorithm of choice for many (at least 25) winning teams of machine learning competitions[3]. The Boosting Gradient method has three[1] types of tuning parameters (general, booster, and task parameters) which account for two dozen hyper-parameters in total. However, James, G. et al.[1] explain that there are only three main hyper-parameters that are primarily optimized for boosting techniques. In the next sub-section, we discuss these three tuning parameters and describe how the tuning was performed.

Hyper-parameter tuning for the XGBoost

The three hyper-parameters of interest are `nrounds`, `max_depth`, and `eta`. `nrounds` describes the number of trees that are fit to the data in parallel. Unlike random forest, the number of trees in XGBoost is a 'goldilocks' parameter – meaning that there *is* an optimal number of trees that when overshot, leads to over-fitting. `max_depth` describes the number of splits in each tree, which controls the trees' complexities and the interaction order of the boosted model. When this value is 1, each tree is a stump- which often ends

up working quite well[1]. Finally, `eta` is the learning rate, a small positive number also known as the shrinkage parameter[1] which controls the rate at which the boosting learns- i.e. advances its minimization procedure.

The hyper-parameter tuning was carried out via grid-search over a range of hyper-parameter values. This methodology allows a large number of combination of parameters. In order to avoid over-fitting, ten cross validations with 5 folds were employed to select the best set of tuning parameters. From our tuning process, we found that the optimal setting for XGBoost occurs when `nrounds` =82, `max_depth` =3, and `eta` =0.295.

XGB Ensemble with Elastic Net

The XGBoost classification returns, for each observation, the probabilities of belonging to each class (in our case, the five cell-types); the predicted class for the observation, then, is the one with the largest probability. In some cases, the maximum probability is not much larger than one or all of the others. We consider these cases to be *poorly predicted*, even if the prediction is correct. Our criterion to distinguish these observations was checking that the largest probability (which belongs to the predicted class) was *at least one order of magnitude greater* than the probabilities for the four remaining classes. One advantage of this approach is that no subjective cut-off value needed to be defined. Although the order of magnitude is an arbitrary rule, it has an adaptive threshold.

We opted to use an algorithm of a different method class to, in a sense, fix these low-certainty predictions. James, G. et al.[1] discuss the limitations of using tree-based algorithms in ensemble code – like Random Forest – when the main algorithm, XGBoost, is also tree-based. Specifically, when non-orthogonal linear relations are present in the data, as seen in Figure 2 below.

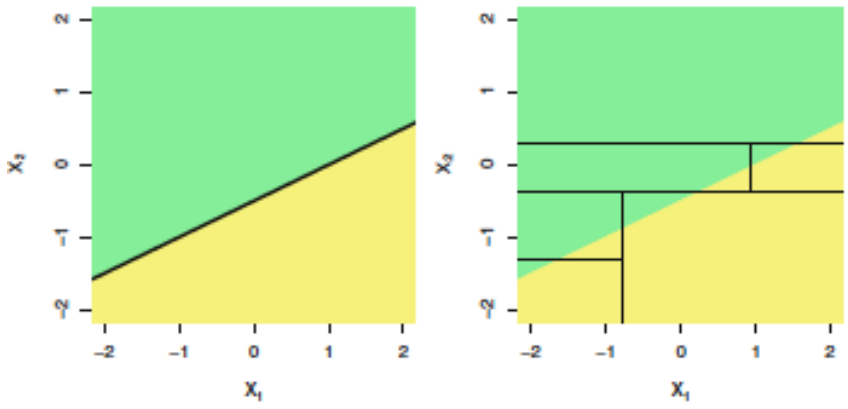

Fig. 2. A 2-dimensional classification example in which the true decision boundary is linear but non-orthogonal, indicated by the shaded regions. Classical linear boundary approach (left) outperforms a decision tree that splits data in parallel to the axes (right). Figure from James, G. et al.[1]

Thus, elastic net modelling was used to correct the poorly predicted observations. From our background knowledge of the midterm project, we knew that the relevant and useful α values would be in the range of 0.1 to 0.3. An odd number of EN models were trained in this range of α's and the predictions stored in a matrix. The poorly-made predictions from XGB were then updated with the modes of the EN predictions. The key assumption here was that the poorly-made-yet correct predictions of XGBoost would, more often than not, be made again in EN. That is, we expected the *incorrect* predictions to be the ones more likely to update through this ensembled (XGB-EN) algorithm.

3 Results

The F1 scores regarding the cross validation repetitions for each type of cell are presented below. Table 1 contains the F1 scores for the benchmark EN model, while Table 2 has the same metric for the proposed

XGB-EN ensemble model. These tables show that the ensemble systematically outperformed the benchmark. In a minor case, 8 out of 10 of the cross-validation F1 scores of the benchmark EN for the B1 cells slightly surpassed the ensemble results for this particular cell type. The mean F1 score of ensemble was just 1 basis-point below the benchmark, though, which makes the superiority of the benchmark questionable regarding B1 cell type. In fact, the p-value of the Wilcoxon Rank-Sum test, shown in Table 3, points to no significance on this result.

On the other hand, for all other kinds of cells, the ensemble surpassed the benchmark EN performance by at least 8 basis-points. Indeed, the Wilcoxon test indicates that the ensemble was statistically significantly better than the baseline EN in classifying 3 out of the 5 cell types, as can be seen in Table 3. In Table 4, other metrics of classification quality of the ensemble method, namely, AUC (area under the curve), precision, and recall are summarized.

Box-plots comparing the F1 scores of the EN baseline and XGB-EN ensemble are presented in Figures 3, containing both scores per cell-type and the scores averaged across the five cell-types. Further analysis of these plots is provided in the next section.

CV	B cells	Mesothelial cells	Myofibroblasts	pDCs	Smooth Muscle cells	Cell averages
1	0.996	0.976	0.9785	0.9727	0.9781	0.9803
2	0.996	0.9783	0.9773	0.9755	0.9767	0.9807
3	0.9965	0.9738	0.9775	0.9755	0.9775	0.9802
4	0.9965	0.9739	0.9761	0.9755	0.9771	0.9798
5	0.9975	0.9782	0.9758	0.9755	0.9761	0.9806
6	0.997	0.9759	0.9767	0.981	0.9756	0.9813
7	0.9965	0.9804	0.9752	0.9783	0.9746	0.981
8	0.9965	0.9783	0.9775	0.9783	0.9771	0.9815
9	0.9955	0.976	0.9764	0.981	0.9757	0.9809
10	0.9945	0.9759	0.974	0.967	0.9739	0.9771
Mean	0.9963	0.9767	0.9765	0.9760	0.9762	**0.9803**

Table 1
F1 scores per cell-type and average for the benchmark EN Model, on 10 repetitions of 5-fold CV's

CV	B cells	Mesothelial cells	Myofibroblasts	pDCs	Smooth Muscle cells	Cell Averages
1	0.9960	0.9935	0.9807	0.9783	0.9767	0.9850
2	0.9955	0.9957	0.9815	0.9755	0.9782	0.9853
3	0.9955	0.9957	0.9807	0.9755	0.9771	0.9849
4	0.9960	0.9978	0.9833	0.9755	0.9797	0.9865
5	0.9960	0.9935	0.9810	0.9838	0.9778	0.9864
6	0.9965	0.9957	0.9818	0.9783	0.9782	0.9861
7	0.9965	0.9935	0.9795	0.9783	0.9757	0.9847
8	0.9955	0.9935	0.9822	0.9755	0.9793	0.9852
9	0.9970	0.9978	0.9804	0.9810	0.9760	0.9865
10	0.9970	0.9957	0.9813	0.9810	0.9782	0.9866
Mean	0.9962	0.9953	0.9812	0.9783	0.9777	**0.9857**

Table 2
F1 scores per cell-type and average for our ensemble XGB-EN method, on 10 repetitions of 5-fold CV's

	B cells	Mesothelial cells	Myofibroblasts	pDCs	Smooth Muscle cells	Average
Mean	0	0.0185	0.0047	0.0022	0.0015	0.0015
Test stat.	22	55	55	37	50	55
P-value	0.6103	0.0059	0.0059	0.3577	0.0249	0.0059

Table 3
Wilcoxon Rank-Sum test results on the differences between the Ensemble XGB-EN and benchmark EN F1 scores

CV	AUC	Precision	Recall
1	0.9879	0.9896	0.9807
2	0.9881	0.9899	0.9809
3	0.9878	0.9895	0.9805
4	0.9891	0.9907	0.9824
5	0.9891	0.9901	0.9829
6	0.9888	0.9903	0.9822
7	0.9876	0.9893	0.9803
8	0.9879	0.9902	0.9804
9	0.9891	0.9899	0.9832
10	0.9892	0.9904	0.983
Mean	0.9885	0.9900	0.9817

Table 4
Averages of AUC, Precision, and Recall across cell-types for the ensemble XGB-EN method, on 10 repetitions of 5-fold CV's

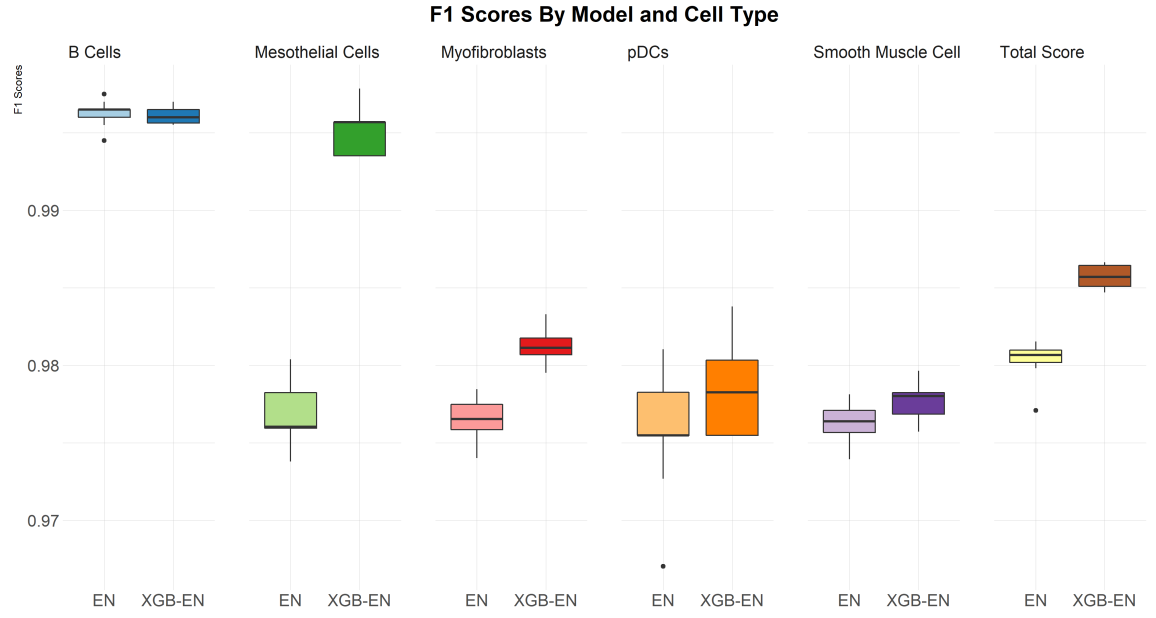

Fig. 3. Box-Plots of F1 scores by cell-type for our ensemble XGB-EN method versus the baseline EN model.

4 Discussion

As presented in Section 3, the ensemble method systematically outperformed the benchmark, with minor cases where the cross-validation F1 scores of the benchmark EN surpassed the ensemble results. Moreover, the Wilcoxon test, shown in Table 3, also attests that the ensemble is statistically significant better, which means that its superior results were unlikely due to chance. That is, there is evidence supporting the en-

semble's explicitly higher quality of classification. The overall average of the ensemble F1 score is 54 basis points above the benchmark average result. Additionally, the AUC, precision and recall metrics were above 98%.

These results are visualized in Figure 3, where we can see that the three cell-types with significant differing F1 scores are those with disjoint box-plots; namely, these are the Mesothelial, Myofibroblast, and Smooth muscle cells. By disjoint, we mean to say that the boxes of the F1 scores- the interquartile ranges- do not overlap. The two models' B cell and PDC F1 scores overlap greatly, however. Overall, both models had their best performances with B cells. Figure 3 also contains the box-plots of the F1 scores averaged across the five cell-types (heading: 'Total Score') for both the ensemble and baseline model. Here we see, once and for all, the significant improvement the ensembled algorithm had on the classifications. The order of difference in their mean F1 value (Tables 1 and 2) is small in magnitude, but the ranges themselves are also quite small; which is what makes that difference so significant. Neither the boxes nor the whiskers of the two models overlap in Figure 3: subplot 'Total Score'.

5 Conclusion

The elastic net model is a powerful tool for machine learning, specifically in the cases of classification of single-cell RNA-sequencing data, and figuring out how to out-perform it in this project was certainly a challenge. We found that feature selection is crucial in this large-data context, both because of the reduction in computation power needed, and in the reduction of multicollinearity within the data. Further work into this project should include a more precise selection of features from the grand, original dataset that was used in the midterm. It is likely possible to find a subset that can achieve F1 scores of approximately 0.99 or greater.

XGBoost has its advantages and it also performs extremely well on its own. However, using elastic net for those poorly-predicted observations enhanced both the sensitivity and specificity. We have found that it is difficult to significantly beat the performance of the elastic net model in the classification of B-cells and PDC's. The upshot is, however, that the ensemble algorithm wasn't significantly worse either. For time and report constraints, we could not include the other models we tried which also beat the benchmark F1 scores. Amongst these include the XGBoost modelling on its own, and two other ensembles: XGBoost with random forest, and random forest with elastic net. Further work into this project would allow us to compare those models to each other.

Overall, in spite of the two cell-type classifications that could not be beaten, the ensemble model is a solid, stable solution for this task. We hope that these results were as interesting to read as they were for us to compute and write about.

6 References

1. Gareth, J., Daniela, W., Trevor, H. & Robert, T. *An Introduction to Statistical Learning* (2017).
2. Trevor, H., Robert, T. & Jerome, F. *The Elements of Statistical Learning* (2017).
3. *Awesome XGBoost* Accessed: 2021-04-14.