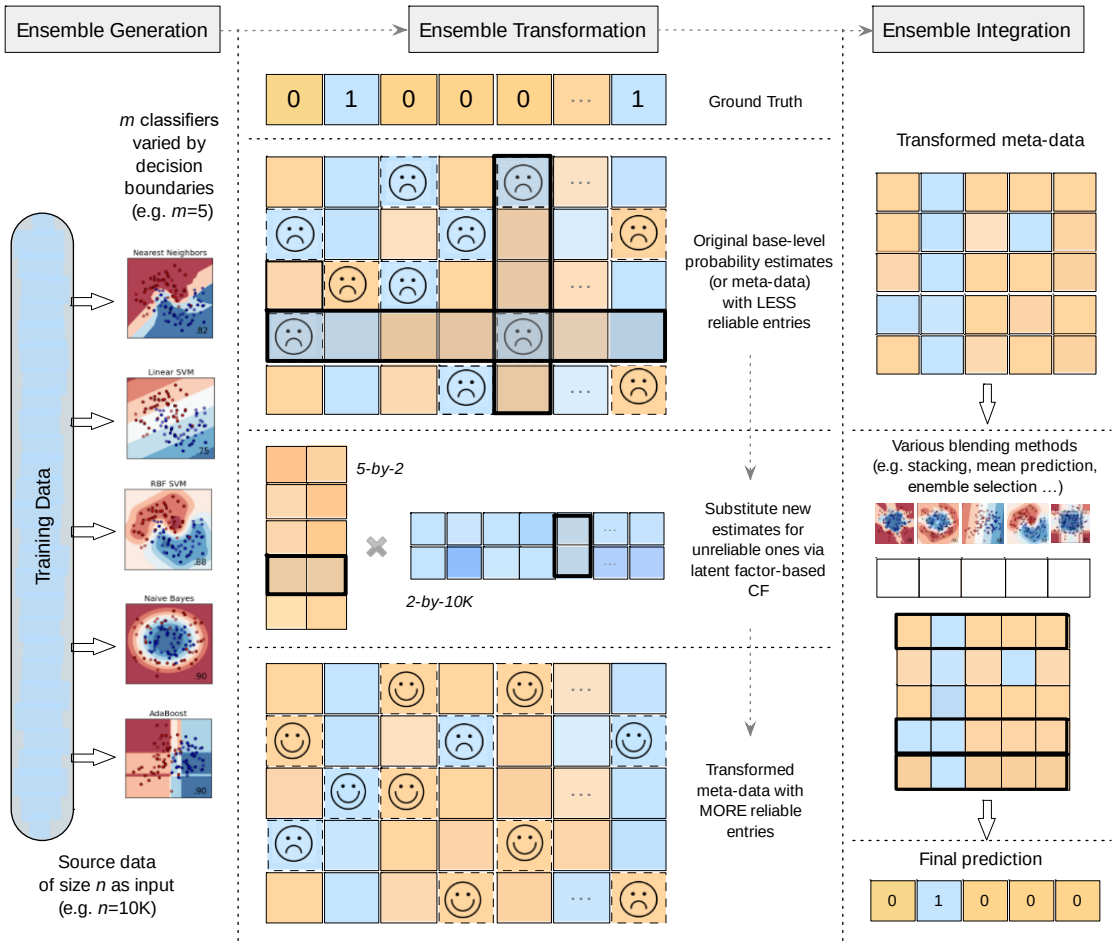

Figure 1. An example ensemble system with *m* classifiers (*m*=5) and *n* data points (e.g. *n*=10K) in the ensemble generation stage, producing meta-data (level-1 data) in the form of an *m-by-n* probability matrix, which is then processed by the CF-based ensemble transformation, leading to an re-estimated matrix of probabilities (level-2 data) that potentially benefits the ensemble integration stage.

Classification problems in biomedical domains are challenging due to the complexity inherent in the relationship between biological concepts and explanatory variables as well as the lack of consensus in best classifiers, being problem dependent. From data science perspective, the contributing factors to the modeling difficulties largely arise from prevalence of class imbalance, missing values, imprecise measurements introducing noises, and far-from-ideal predictor variables due to incomplete knowledge of the underlying processes per se.

Ensemble classifiers are meta-algorithms that combine several classification algorithms into one predictive model in order to decrease variance (e.g., bagging) and generalization errors (e.g. stacking). Heterogeneous ensembles in particular make the algorithmic diversity explicit by introducing classifiers that generate decision boundaries with varying properties that, when combined, can be empirically shown to adapt better to the biological datasets in terms of the predictive performance[1-3]. For instance, simple linear models such as logistic regression can perform better in regions of the data exhibiting simpler relationships between dependent variables and class labels while complex, non-linear models such as SVM with non-linear kernels, deep neural networks, among others, can progressively cover data regions that would require more complex decision boundaries in order to separate the classes well.

Without loss of generality, the ensemble learning process can be divided into two main stages: i) ensemble generation, where multiple base models are generated, and ii) ensemble integration, where base-level predictions are pruned and combined, leading to a multitude of methodologies – ensemble selections[4], stacked generalization[5, 6], and Bayesian model combination[7, 8], being among the prime examples. In this study, we will introduce an intermediate stage, *ensemble transformation*, that precedes the ensemble integration stage. More precisely, this is an additional layer of meta-learning through transforming the probability predictions at the base level toward increasing the reliability of their class conditional probability estimates, which then potentially benefit the ensemble integration stage as well as providing additional features such as model interpretation, often desirable in the context of biomedical data analytics. For instance, identifying group structures in the training data in contexts of how they are being classified by the ensemble helps to identify the homogeneous subsets that share similar biomedical properties as well as training instances that would potentially pose challenges to the ensemble. To achieve both predictive performance improvements and model interpretability, we propose a latent factor-based collaborative filtering (CF) approach[9-12] to realizing the ensemble transformation stage, as illustrated in Fig. 1.

CF is a methodology commonly used in recommender systems in which active users provide ratings among many forms of feedback signals, explicit or implicit, to the items of interest. CF predicts user ratings on a set of commercial items, being constantly evaluated by the active users (collaborative), through the feedback signals coming from like-minded users (filtering). The collaborative aspect of forming a consensus from diversified ensemble classifiers bears a conceptual resemblance to the CF paradigm in the sense that base models play the role of interested users assessing a set of consumer items, the collection of which is analogous to the training data instances in a supervised learning setting. Specifically, a high user preference over a consumer item can be inferred from the user behavior including implicit feedback, where users exhibit their interests implicitly through high frequencies of online interactions; and explicit feedback, where users provide their personal ratings on items directly. In a similar fashion, ensemble classifiers generating high class conditional probabilities often signifies high confidence levels in the label prediction when these classifiers are probabilistic and well-calibrated[13, 14].

Using both the "collaborative" and "filtering" properties, the latent factor modeling approach is brought to bear upon the ensemble transformation where each base classifier and data instance are to be represented by their respective latent factors such that they can be combined (e.g., in terms of inner products) to provide a means to re-estimate class conditional probabilities. Specifically, we wish to tease out unreliable probability outputs at the base level and subsequently re-estimate these probability values through the classifier-data interactions with higher degrees of reliability via a selective subset of the classifiers and training data.

Similar to the rating matrix in the CF setting (i.e. user ratings organized in a matrix form[15]), one can organize the base-level class conditional probabilities into a "probabilistic rating matrix" where each row references a prediction vector associated with a base classifier, representing its probabilistic predictions on the training data (indexed by columns). Parallel to the probabilistic rating matrix is a confidence matrix where each entry represents a confidence score that measures the reliability of a class conditional probability associated with its corresponding base classifier and training instance. The probability rating matrix is then embedded within a weighted matrix factorization[11, 16, 17] as an iterative optimization process in the form of alternating least square (ALS)[18] for the purpose of i) deriving the latent factor representation for both the classifiers and the training data, and ii) using the learned latent factors to re-estimate the portion of the probabilistic outcomes deemed unreliable by the confidence matrix. In other words, the confidence matrix acts as a weighting mechanism in the optimization procedure that assigns higher weights on reliable entries in the probability matrix such that the resulting latent factors better reflect reliable class conditional probabilities from the classifier ensemble upon filtering those deemed unreliable.

Formally, consider a classifier ensemble of size m and a training set of size n such that the class conditional probability resulted from the ensemble generation stage can be organized in an $m \times n$ matrix R with entries r_{ui}, where we have adopted the notational convention from the CF literature by indexing classifiers by u (user), data by i (item) while denoting probability values in R by r_{ui} as having a semantic of ratings. Assuming that each probability value comes in tandem with a confidence score, we have a $m \times n$ confidence matrix C (or by index notation c_{ui}) assuming the same dimensionality as the probability rating matrix R.

Suppose further that a *d*-dimensional factor representation is chosen for both classifiers and training data. We then seek to find $x_u \in \mathbb{R}^d$ for each classifier, and $y_i \in \mathbb{R}^d$ for each training instance such that the following loss function is minimized:

$$L(X,Y) = \sum_{u,i} c_{ui}(r_{ui} - x_u^T \cdot y_i)^2 + \lambda(\sum_u \|x_u\|^2 + \sum_i \|y_i\|^2) \quad (1)$$

The optimization objective is thus to find the factor representation (i.e. appropriate configurations as *d*-dimensional vectors for x_u and y_i respectively) such that the inner product between x_u and y_i best approximates r_{ui} in a (weighted) least-squares sense, where each square-error term is weighted by the corresponding confidence score c_{ui}. In particular, given a confidence measure (e.g. Brier score[19]), the higher the confidence score it is assigned to a class conditional probability, the smaller the deviation from the inner product $x_u^T \cdot y_i$ to the target r_{ui} it is necessary in order to keep the loss *L* small. Additionally, the $\ell 2$-regularization with parameter λ is used to prevent overfitting during the training phase.

For notational convenience, let $X \in \mathbb{R}^{m \times d}$ be a matrix formed by stacking *m* classifier vectors $\{x_u \mid u = 1 \ldots m\}$, each of which as a row vector, and let $Y \in \mathbb{R}^{n \times d}$ be the corresponding data matrix comprising *n* data instance vectors $\{y_i \mid i = 1 \ldots n\}$. We use ALS to iteratively minimize the loss by alternating the following two steps: i) hold *Y* fixed and solve for x_u such that *L* is minimized as a quadratic function of x_u and ii) by symmetry, hold *X* fixed and solve for y_i that minimizes *L* as a quadratic function of y_i. Note that in the optimization step i) and ii), an analytic expression can be found for the classifier vector x_u in terms of the data matrix *Y* by solving $\partial L / \partial\, x_u = 0$ to get

$$x_u = (Y^T C^u Y + \lambda I)^{-1} Y^T C^u r_u \quad (2)$$

Similarly, data instance vector y_i can be expressed analytically in *X* by solving $\partial L / \partial\, y_i = 0$ to get

$$y_i = (X^T C^i X + \lambda I)^{-1} X^T C^i r_i \quad (3)$$

In (2), we have defined C^u, for each classifier *u*, as an $n \times n$ diagonal matrix with c_{ui} in the diagonal, and r_u as a *n*-dimensional prediction vector with components r_{ui}; that is, C^u holds in the diagonal the confidence scores associated with classifier *u*, representing its class conditional probability estimates on the training data indexed by *i*. Similarly, C^i is defined for each data instance *i* as an $m \times m$ diagonal matrix with c_{ui} along the diagonal, representing the confidence scores associated with training instance *i* being predicted/rated by ensemble classifiers indexed by *u*; and r_i is a *m*-dimensional vector with components r_{ui}, representing the class conditional probabilities on a training instance *i* coming from across ensemble classifiers indexed by *u*.

Using (2) and (3), and careful matrix operations for additional speedup[17], we can then use linear system solvers in SciPy[20] (e.g. scipy.sparse.linalg.spsolve) to iteratively re-compute x_u and y_i through the following linear system, starting from their respective initialized values:

$$(Y^T C^u Y + \lambda I) x_u = Y^T C^u p_u$$

$$(X^T C^i X + \lambda I) y_i = X^T C^i p_i$$

The advantage of the latent factor approach to ensemble transformation mainly comes in three-fold: i) capability of coping with missing values ii) dimensionality reduction, and iii) interpretability.

Since the optimal ensemble is difficult to determine a priori (and at times infeasible when the target dataset is unknown), contemporary ensemble learning methods inevitably experience some level of performance bottleneck, stemming from poor probability estimates from at least a subset of the classifier ensemble over a subset of the data wherein they do not generalize well. These bad probabilities/ratings that drove down the predictive performance can thus be regarded as "missing values" that are to be estimated collaboratively by the classifier ensemble using reliable probability values.

Further, consider the case where the probability rating matrix came from an unknown source, as is common in a data integration setting[21, 22]. Without a full transparency, we may not have access to certain details of these base classifiers, which can include classification algorithm specifics, how models were trained, classifier properties such as generalization capacity – on top of the data sample characteristics such as the raw feature set and its dimensionality. Thus, similar to the role of users and items in the CF setting, classifiers and sample characteristics ought to be regarded as unknown. Nonetheless, latent factor models attempt to reasonably approximate classifier and sample properties through fixed dimensional vectors, regardless their respective intrinsic dimensions. Additionally, the learned latent factors can be shown useful in constructing similarity measures for users and items in the CF setting[17, 23, 24] and analogously, we hope to establish this property as well for ensemble learning toward better model interpretability.

We illustrate the utility of the ensemble transformation by contrasting the predictive performance before and after the transformation on several slice site prediction datasets[25, 26] formulated as binary classification problems. In the ensemble generation stage, we pre-train 12 different classifiers[a], as a demonstration, to obtain the probability matrix as meta-data (or level-1 data). This is followed by the ensemble transformation stage using the aforementioned CF-based latent factor model (with latent feature dimension d=100) to arrive at the transformed meta-data (or level-2 data) comprising potentially better class conditional probability estimates to be observed empirically. In the subsequent ensemble integration stage, we then observe the performance difference by employing the following two methods on both the original dataset and the transformed dataset: i) stacked generalization using multiple classifiers (n=8 in this study), and ii) mean aggregation by averaging the prediction vectors from the given stackers. Doing so gives us 6 sets of pairwise performance comparison bar charts as illustrated in Fig 2, one for each splice-site prediction dataset. In particular, for the simplicity of comparison, the 8 stacking methods and the mean aggregation are averaged for both level-1 data (i.e., original meta-data) and level-2 data (i.e. transformed meta-data).

References

1. Whalen S, Pandey OP, Pandey G. Predicting protein function and other biomedical characteristics with heterogeneous ensembles. Methods. 2016;93:92-102. Epub 2015/09/08. doi: 10.1016/j.ymeth.2015.08.016. PubMed PMID: 26342255; PMCID: PMC4718788.
2. Wang L, Law J, Kale SD, Murali TM, Pandey G. Large-scale protein function prediction using heterogeneous ensembles. F1000Res. 2018;7:ISCB Comm J-1577. doi: 10.12688/f1000research.16415.1. PubMed PMID: 30450194.
3. Lertampaiporn S, Thammarongtham C, Nukoolkit C, Kaewkamnerdpong B, Ruengjitchatchawalya M. Heterogeneous ensemble approach with discriminative features and modified-SMOTE bagging for pre-miRNA classification. Nucleic acids research. 2013;41(1):e21-e. Epub 2012/09/24. doi: 10.1093/nar/gks878. PubMed PMID: 23012261.

[a] We use 12 diverse base-level classifiers from WEKA machine learning suite including: Naïve Bayes, logistic regression, multilayer perceptron, linear SVM, K-nearest neighbors, Adaboost, classification via regression, two rule-based classifiers (RIPPER, PART), and 3 tree-based classifiers (decision tree, random forest, random tree).

4. Caruana R, Munson A, Niculescu-Mizil A, editors. Getting the most out of ensemble selection. Sixth International Conference on Data Mining (ICDM'06); 2006: IEEE.
5. Džeroski S, Ženko B. Is combining classifiers with stacking better than selecting the best one? Machine learning. 2004;54(3):255-73.
6. Ting KM, Witten IH. Stacked Generalization: when does it work?1997.
7. Simpson E, Roberts S, Psorakis I, Smith A. Dynamic bayesian combination of multiple imperfect classifiers. Decision making and imperfection: Springer; 2013. p. 1-35.
8. Kim H-C, Ghahramani Z, editors. Bayesian classifier combination. Artificial Intelligence and Statistics; 2012.
9. Mnih A, Salakhutdinov RR, editors. Probabilistic matrix factorization. Advances in neural information processing systems; 2008.
10. Koren Y, Bell R, Volinsky C. Matrix factorization techniques for recommender systems. Computer. 2009(8):30-7.
11. Gu Q, Zhou J, Ding C, editors. Collaborative filtering: Weighted nonnegative matrix factorization incorporating user and item graphs. Proceedings of the 2010 SIAM international conference on data mining; 2010: SIAM.
12. Koren Y, Bell R. Advances in collaborative filtering. Recommender systems handbook: Springer; 2015. p. 77-118.
13. Niculescu-Mizil A, Caruana R, editors. Predicting good probabilities with supervised learning. Proceedings of the 22nd international conference on Machine learning; 2005: ACM.
14. Zadrozny B, Elkan C, editors. Obtaining calibrated probability estimates from decision trees and naive Bayesian classifiers. Icml; 2001: Citeseer.
15. Li B, Yang Q, Xue X. Transfer learning for collaborative filtering via a rating-matrix generative model. Proceedings of the 26th Annual International Conference on Machine Learning; Montreal, Quebec, Canada. 1553454: ACM; 2009. p. 617-24.
16. Wang K, Peng H, Jin Y, Sha C, Wang X. Local weighted matrix factorization for top-n recommendation with implicit feedback. Data Science and Engineering. 2016;1(4):252-64. doi: 10.1007/s41019-017-0032-6.
17. Hu Y, Koren Y, Volinsky C, editors. Collaborative Filtering for Implicit Feedback Datasets. ICDM; 2008: Citeseer.
18. Hastie T, Mazumder R, Lee JD, Zadeh R. Matrix completion and low-rank SVD via fast alternating least squares. The Journal of Machine Learning Research. 2015;16(1):3367-402.
19. Hernández-Orallo J, Flach PA, Ramirez CF, editors. Brier Curves: a New Cost-Based Visualisation of Classifier Performance. ICML; 2011.
20. Jones E, Oliphant T, Peterson P. SciPy: Open Source Scientific Tools for Python 2014-, http://www.scipy.org/ [Online; accessed 2019-06-15].
21. Li Y, Wu F-X, Ngom A. A review on machine learning principles for multi-view biological data integration. Briefings in bioinformatics. 2016;19(2):325-40.
22. Zitnik M, Nguyen F, Wang B, Leskovec J, Goldenberg A, Hoffman MM. Machine learning for integrating data in biology and medicine: Principles, practice, and opportunities. Information Fusion. 2019;50:71-91.

23. Wang X, Xu C, editors. SBMF: similarity-based matrix factorization for collaborative recommendation. 2014 IEEE 26th International Conference on Tools with Artificial Intelligence; 2014: IEEE.
24. Takács G, Pilászy I, Németh B, Tikk D, editors. Matrix factorization and neighbor based algorithms for the netflix prize problem. Proceedings of the 2008 ACM conference on Recommender systems; 2008: ACM.
25. Rätsch G, Sonnenburg S, Srinivasan J, Witte H, Müller K-R, Sommer R-J, Schölkopf B. Improving the Caenorhabditis elegans genome annotation using machine learning. PLoS Computational Biology. 2007;3(2):e20.
26. Schweikert G, Rätsch G, Widmer C, Schölkopf B, editors. An empirical analysis of domain adaptation algorithms for genomic sequence analysis. Advances in Neural Information Processing Systems; 2009.