8-10-2017

Rank Aggregation of Feature Scoring Methods for Unsupervised Learning

Tara N. Yankee
University of Connecticut, tara.yankee@uconn.edu

Rank Aggregation of Feature Scoring Methods for Unsupervised Learning

Tara N. Yankee

B.S. Mechanical Engineering, University of Connecticut, 2011

A Thesis

Submitted in Partial Fulfillment of the

Requirements for the Degree of

Master of Science

at the

University of Connecticut

2017

2017

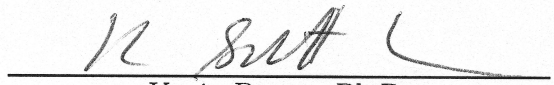
APPROVAL PAGE

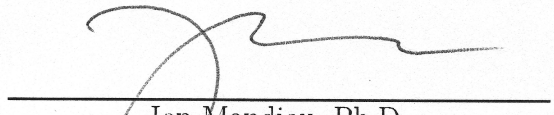
Master of Science Thesis

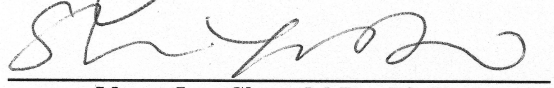
Rank Aggregation of Feature Scoring Methods for Unsupervised Learning

Presented by

Tara N. Yankee, B.S.

Major Advisor 
Kevin Brown, Ph.D.

Associate Advisor 
Ion Mandiou, Ph.D.

Associate Advisor 
Yong-Jun Shin, M.D., Ph.D.

University of Connecticut

2017

ACKNOWLEDGMENTS

Thank you to Dr. Kevin Brown who so graciously agreed to mentor me. This Master's program was a vehicle for me to change careers from corporate to academia. Even though I had no prior experience in research, Dr. Brown took a chance on me and I will forever be grateful. I also want to thank my parents, who although may not have understood my choices, nevertheless always supported my elusive dream of entering academia at the expense of my 401k. Thank you for your patience and abiding faith. Thank you to my boyfriend who as a solider of the United States Army is a hero to me and this country. I thank him for always seeing the best in me, and believing in my abilities even at times when I didn't. My time as a masters student at UConn was life changing. I discovered passions I never knew I had. I feel incredibly lucky for this opportunity and without it would never have been able to pursue my dream of obtaining a doctorate.

Contents

Approval Page iii

Acknowledgements iv

List of Tables vi

List of Figures vii

Abstract viii

1 Introduction 1

2 Methods 5

2.1 Pipeline 6

2.2 Feature Scoring and Selection 7

2.2.1 Filter Methods 8

2.2.2 Wrapper Methods 12

2.3 Rank Aggregation 16

2.4 Clustering 18

2.5 Validation 19

3 Results 22

3.1 Benchmark Data 22

3.2 Aggregate Ranking of Features 23

3.3 Inter-Ranker Agreement 26

3.3.1 IRA $W+$ Results 28

3.3.2 IRA $W-$ Results 29

4 Discussion 38

4.1 Results -Highlights/Interpretations 38

4.2 Voting Methods 40

4.3 Future Directions/Extensions 41

Bibliography 42

List of Tables

2.1.1 No. of Features to Cluster per Data-set 6
2.1.2 No. of Features to Cluster Iris Example 7
2.2.1 Feature Selection Methods 16
2.3.1 Borda Example 17

3.1.1 Data-set List 23

List of Figures

2.0.1 Pipeline 5
2.2.1 Feature Scoring 8
2.2.2 Pseudo-Labels for Wrapper Methods within Pipeline 13
2.4.1 Fisher Score v. Cluster Number 19

3.2.1 Heatmaps 31
3.2.2 Results- SSD Bar Graphs 32
3.2.3 Results-NMI Bargraphs 33
3.2.4 Results-NMI Overall 34
3.3.1 Kendall's $W+$ Results 34
3.3.2 Iter-Ranker Agreement $W+$ Results 35
3.3.3 Kendall's $W-$ Results 36
3.3.4 Iter-Ranker Agreement $W-$ Results 37

ABSTRACT

The ability to collect and store large amounts of data is transforming data-driven discovery; recent technological advances in biology allow systematic data production and storage at a previously unattainable scale. It is common for biological Big Data to have an order of magnitude or more features than samples. Feature scoring with selection is therefore an essential pre-processing step to finding meaningful clusters in these data. Many feature scoring algorithms have been proposed; they are based on dramatically different ideas about what constitutes a "good" or "important" feature. Motivated by studies in data classification, we use a rank aggregation (RANKAGG) method to combine estimates of feature importance from multiple sources and use a subset of the highest scoring features for subsequent clustering. We demonstrate the performance of RANKAGG on five real-world biological data-sets, and compare the clustering performance of RANKAGG to the thirteen individual feature scoring methods comprising RANKAGG. The rank aggregated features have a mean performance across the five data-sets equal to the best individual feature scoring method but with lower variance, indicating robust performance across a variety of data. We carefully consider if there is any systematic way to remove rankers from RANKAGG to improve clustering performance. We demonstrate that rank aggregated feature selection yields excellent performance in clustering problems and possibly more importantly, greatly limits the risk of choosing a method that is sub-optimal for a given data-set.

1

Introduction

The ability to collect large-scale data, particularly in biology, has posed unique challenges for data analysts. In what Donoho (15) calls the "classical world" of data, the dominant paradigm underpinning traditional analytic approaches relied on the assumption that the number of observations far outnumbered the measured variables. However, due to unprecedented technological advances of systematic data production and storage, we now live in a "post-classical," "Big Data" world. The well-known *curse of dimensionality* (6) is most keenly felt in these modern data-sets: it is not uncommon for a given data-set to have orders of magnitude more variables than samples. This is particularly true in genetics, where next generation sequencing technology can gather expression data for tens of thousands of genes in thousands of cells (20). Along with the unfavorable ratio of variables to samples, these data in particular have a lack of characterizing information. For example attempting to identify rare cell types from single-cell sequencing data (19, 22, 26), there is no training set of known labels to employ. Hence the need for powerful unsupervised learning (clustering) methods to make sense of these data.

Feature selection is a powerful and sometimes absolutely necessary work around for the dimensionality challenge of Big Data (21). Feature selection acts as filter to reduce noise in a data-set by discarding irrelevant variables (features) while also identifying those that best represent the data. With respect to clustering, there are three categories of feature selection methods: filters, wrappers, and hybrid methods (3). Filters are efficient because they use a specific criterion with which to score features independent of any classification. Wrappers need to interact with some kind

of classifier since they generally use class labels to score the features. Hybrid methods combine some elements of both filters and wrappers.

There are a dizzying array of feature selection methods that have been proposed but prescriptions for their usage are vague. Each of these methods measure different aspects of the data and hence produce very different definitions of a "good" feature. These choices can result in substantially varied performance across an array of datasets. Confronted with some new set of data for which feature selection is either desirable or essential, a question that many researchers encounter in their work is how does one choose which feature selection method is best?

To mitigate this issue we take a cue from ensemble learning (44). Succinctly put, ensemble learning is the process of employing multiple models and then synthesizing the results. Ensemble methods have a wide range of applications in many fields such as machine learning, pattern recognition, neural networks and statistics. Subsequently, ensemble learning has been used extensively in classification- a cousin of the clustering problem. It is known that an ensemble classifier is often more stable and better performing that a given single classifier (12). Ensemble learning has also been applied to feature selection for classification problems. Previous studies have shown that ensemble feature selection successfully produces a more robust feature subset for classification than single feature selection techniques (34, 35, 36). Ensemble techniques have also been used to combine multiple clusterings into a consensus set of clusterings, which can allow a simple clustering method like K-means to uncover arbitrarily shaped clusters (18).

In this paper we develop an ensemble feature selection/scoring algorithm and compare it to a suite of individual feature selection methods. We deal exclusively with unsupervised learning problems. Whereas supervised learning allows for a function to

be inferred from a set of training data, unsupervised learning does not contain labels and hence no training data is available. Unsupervised learning is an understudied sub genre within machine learning, primarily because of the difficulty in extracting patterns from this elusive data. To our knowledge this is the first study of rank aggregation of feature scoring methods with respect to unsupervised learning. Clustering is a common machine learning tool used in unsupervised learning to group data without preexisting labels; categorizing data samples into clusters based solely on their feature characteristics. The clustering performance of features selected by feature methods, including the ensemble feature method through rank aggregation will be a gauge to evaluate their effectiveness. To produce a set of voted features we use rank aggregation by Borda scoring (7); this method has been used for feature selection in classification (34, 35, 36) and is also one of the simplest ways of combining the results from multiple rankers (16). We also evaluate a large suite (thirteen) feature selection methods on five data-sets with varying numbers of features and samples; this gives a much more comprehensive view of the performance of these methods than has heretofore been available. We show that voted features yield a clustering performance equal to the best single feature selection method, but with lower variance (more stability) across the five data-sets we consider. We also carefully analyze a proposal for feature selection in classification that tries to improve performance by selectively removing rankers from the voting subset (36). We find inconsistent results which indicate that manipulating the voter rolls can be at best minimally helpful and at worst deleterious to the final clustering result.

The paper is organized as follows. In section 2.1 we describe our processing and analysis pipeline in detail. Sections 2.2 gives a detailed description of the thirteen feature scoring methods we consider in this study. We describe the process of rank

aggregation and our clustering methods in sections 2.3 and 2.4 respectively. We then turn to our methods for validating clustering performance in section 2.5. Results for both the rank aggregate feature scoring algorithm and attempts to improve results by removing voters are in sections 3.2 and 3.3. Finally, we discuss the implications of our results and point out some future directions.

2

Methods

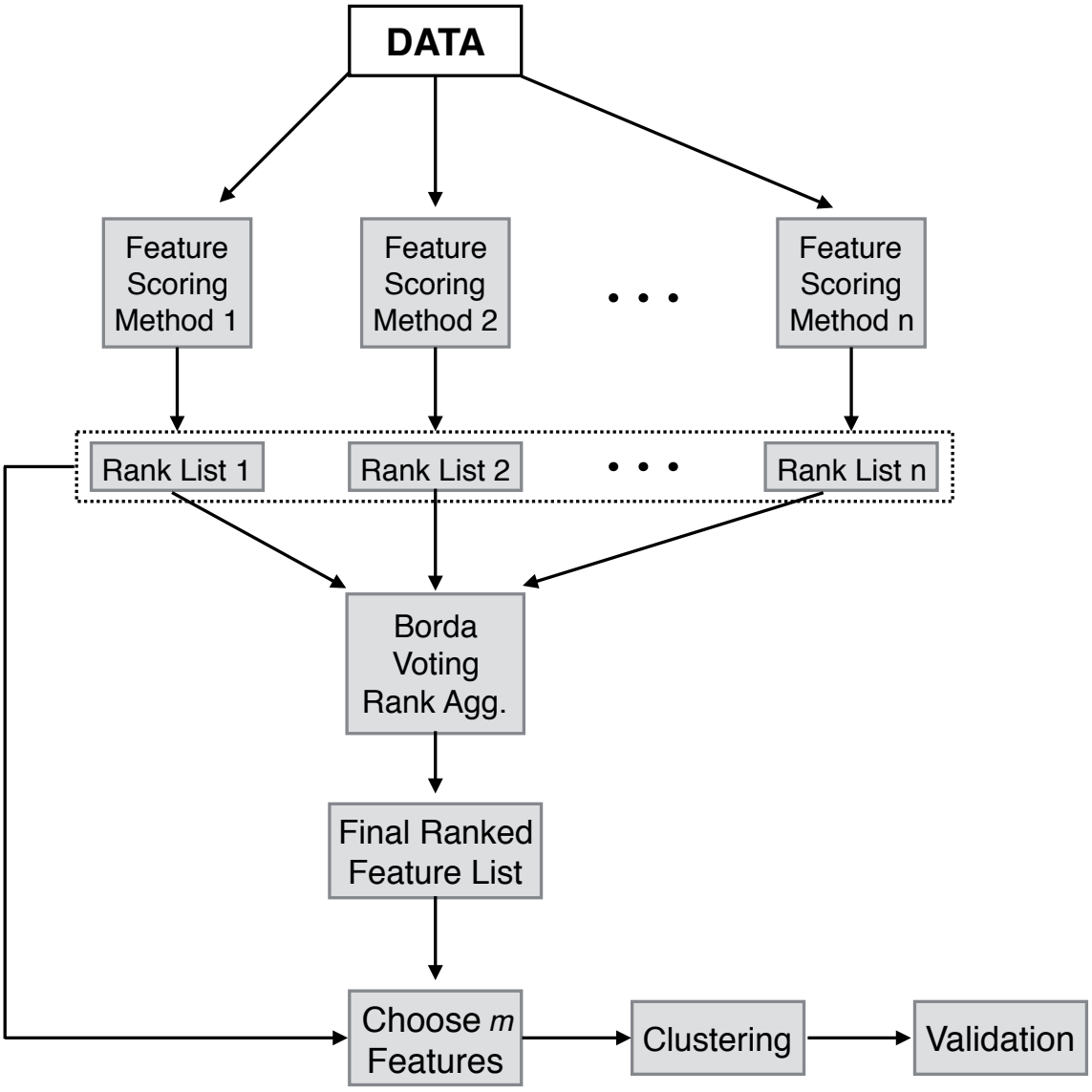


FIGURE 2.0.1: Pipeline used for test data. n is the number of feature methods. The rank list from each feature method was itself clustered and validated.

2.1 Pipeline

Figure 2.0.1 shows our feature selection, clustering, and validation pipeline. Python was the primary language used to create this pipeline (1). We assume the data is an $N \times p$ matrix consisting of N samples, each with p features. A set of feature scores is computed for each of the feature selection methods under consideration; these methods are explained in detail in Section 2.2. After score computation, each of the n vectors of feature scores are transformed to a vector of ranks. Depending on the scoring method, a "good" feature can be a feature with either a high or low score.

From each set of ranked features, we choose $m = \sqrt{p}$ of the highest-ranked features. See Table 2.1.1 for the number of features clustered from each data-set. Table 2.1.2 shows an example of how the top features were chosen for the Iris data-set for the LAPL feature method (see Section sec:wrapper for more details). LAPL ranked features 4 and 3 as the first and second best and so these features were input to be clustered. This ensures that clustering is performed using the same number of features regardless of how the features were scored, and the number of selected features scales with the total number of features p in the data. From the n vectors of ranks we produce a consensus set of feature ranks via voting (see Section 2.3) and also select the m highest-ranked features from the consensus ranks for subsequent clustering.

TABLE 2.1.1: The number of features that were clustered from each data-set. $m = \sqrt{p}$

Data-set	**Features**	m
Iris	4	2
Ecoli	7	3
Yeast	8	3
Tumor	9	3
Mouse	77	9

TABLE 2.1.2: Iris Data-set Feature Rank using Laplacian (LAPL) Feature Method. The highlighted columns in the second row indicate the feature numbers chosen to be clustered according to rank.

Rank	1st	2nd	3rd	4th
Feature	4	3	1	2

All of these feature sets (individual and voted) are then used to cluster the data (details in Section 2.4). Performance for each set of selected features and the rank aggregated features was then assessed by comparison to known cluster labels present in each benchmark data-set. For details on the data used in this study see Section 3.1, and for details of the validation methods refer to Section 2.5. We emphasize here that the known cluster labels are not used by any of the scoring methods we consider; they are only employed in the *post hoc* validation process.

2.2 Feature Scoring and Selection

Feature scoring attempts to score features based on their ability to separate clusters and feature selection will select features based on these scores. As an example, suppose one had a data-set that contained two clusters, A& B. A helpful feature method would give distinct and separate values to samples that belonged to either cluster A or B, while an unhelpful feature method would not be able to discriminate scores between the two clusters. This scenario is depicted in Figure 2.2.1. In plot (a) the data samples form Gaussian distributions centered over two separate values, while in plot (b) the feature scores of the samples do not discriminate well between the two clusters and end up in a uniform distribution.

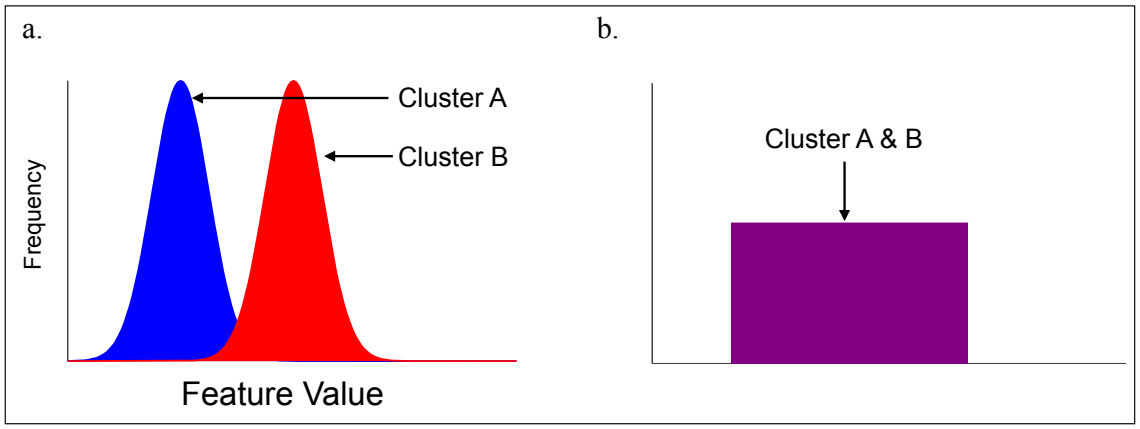


FIGURE 2.2.1: Simple visualization of Feature Scoring. Plot (a) shows a hypothetical distribution of a data-set whos feature scores discriminate between clusters well, while plot (b) is of a feature method that does not well separate data from its clusters.

2.2.1 Filter Methods

Filter methods need no information for calculation other than the $N \times p$ data matrix itself.

- **Linear Predictability (LP)**. Linear Predictability posits that using sets of correlated features to cluster data will always result in better clusters than if uncorrelated features are used (2). To assign an LP score to feature i, we build a linear regression model to predict the value of the i^{th} feature from the remaining $p-1$ features. We compute the root mean square error (RMSE) of the predicted feature value, and this forms the LP score. Features with smaller scores (lower prediction error) are better features.

- **Spectral Feature Scoring 1 & 3 (SP1,SP3)**. Both of these methods construct a similarity matrix for all pairs of data instances. The spectrum of the graph induced by this similarity matrix can then be used to features which tend

to assign similar values to data instances that are nearby in the graph (43). These features are the ones which will better separate the data into its constituent classes. The Laplacian score (23), which we discuss in detail below, is another variant of a spectral feature score. For SP1, smaller values indicate better features while for S3 larger values are better.

- **Laplacian Score (LAPL)**. The Laplacian score (23) is based on the assumption that data from the same class are often "neighboring" one another and useful features will preserve the nearness of samples in the same class. Additional details can be found in (23). Let f_{ri} be the value of the r^{th} feature for the i^{th} sample, where $r = 1, \ldots, p$ and $i = 1, \ldots, N$.

  1. Build a k-nearest neighbor graph for the samples. For each sample, compute its k nearest neighbor samples. An edge between samples x_i and x_j exists if x_j is among the k nearest neighbors of x_i. We used $k = 3$ in this study.

  2. Build a similarity matrix or weight matrix S for the graph. $S_{ij} = e^{-\|x_i - x_j\|^2/t}$ if x_i and x_j are connected in the nearest neighbor graph, otherwise $S_{ij} = 0$. We used $t = 2$ to construct the similarity matrix.

  3. If we write the values of the r^{th} feature as a column vector $\mathbf{f}_r = (f_{r1}, \ldots, f_{rN})^T$, and then define $\mathbf{1} = (1, \ldots, 1)^T$, $D = \text{diag}(S\mathbf{1})$, and the graph Laplacian $L = D - S$, the Laplacian Score for feature r can be computed as

$$L_r = \frac{\tilde{\mathbf{f}}_r^T L \tilde{\mathbf{f}}_r}{\tilde{\mathbf{f}}_r^T D \tilde{\mathbf{f}}_r}, \tag{2.2.1}$$

where $\tilde{\mathbf{f}}_r$ is defined as

$$\tilde{\mathbf{f}}_r = \mathbf{f}_r - \frac{\mathbf{f}_r^T D\mathbf{1}}{\mathbf{1}^T D\mathbf{1}}\mathbf{1}. \tag{2.2.2}$$

The smaller the LAPL, the better the feature.

- **Relevance (RELF)**. "Relevance" is an oft-used word in feature selection; we refer here to a specific notion of relevance (42). RELF can be calculated for an arbitrary subset of features; we compute a relevance score for each single feature, one at a time. Relevance is based on the spectral properties of the affinity matrix A_i, which is constructed as follows. If we take the columns of the $N \times p$ data matrix as vectors and standardize them to obtain $\mathbf{f}_1, \ldots, \mathbf{f}_p$, then $A_i = \mathbf{f}_i \mathbf{f}_i^T$. If we further arrange the leading k eigenvectors of A_i as columns in a matrix Q, the relevance score is given by

$$R_i = \mathrm{Tr}\left\{Q^T A^T A Q\right\}. \tag{2.2.3}$$

We use $k = 3$ for our studies. Larger relevance scores indicate better features. The extension of the score to feature subsets is straightforward; more details can be found in (42).

- **Variance (VAR)**. Variance scores are simple, easy to calculate, and widely used. Any feature which is constant across the entire set of samples is not helpful for clustering. Variance scores postulate that the broader the distribution of sample values within a feature, the more useful that feature will be for clustering. Hence larger feature variance indicates a better feature.

- **Leverage (LEV)**. Leverage scores form part of a feature selection method

specifically designed for k-means clustering. The feature selection method is provably accurate in the case where one knows the number of clusters in advance (9). For unsupervised data this parameter is often not available however we can still compute a leverage score for each feature. Let V be a $p \times k$ matrix of the k leading singular vectors of the $N \times p$ data matrix, and $\mathbf{v}_i$ be the i^{th} row of V. The leverage score for the i^{th} feature is defined as

$$p_i = \frac{1}{k} \|\mathbf{v}_i\|_2^2 \tag{2.2.4}$$

We set k by keeping enough singular vectors to account for 99% of the variance in the data matrix. Larger leverage scores indicate better features.

- **Redundancy (RED)**. Generally, removing features redundant with other existing features should allow the same degree of classification or clustering quality using a reduced number of features. Redundancy (13, 33) defines feature redundancy using mutual information between two (discrete) variables x and y as $I[x, y] = -\sum_{x,y} p_{xy} \log_2 p_{xy}$. To compute the redundancy of a given feature, we first discretize all features as in GINI (see below) in order to compute the joint probability distribution of any two features. The redundancy of feature α is the average mutual information between that feature and the other $p - 1$ features:

$$\text{Rd}\,(\alpha) = \frac{1}{p-1} \sum_{\beta \neq \alpha} I\left[v(\alpha), v(\beta)\right]. \tag{2.2.5}$$

Lower values of redundancy indicate more independent features.

2.2.2 Wrapper Methods

Wrapper methods require a set of class labels for calculation; this means that once labels are available, most feature selection methods usable for classification can also be used for clustering. Common practice is to simply obtain some set of labels via an initial clustering, and use those pseudo-labels as the "true" labels for feature scoring. For all wrapper methods we performed an initial k-means clustering with 3 classes (see Section 2.4 for additional details). Refer to Figure 2.2.2 to see where in the pipeline these extra steps reside. We experimented with values of k between two and ten for the feature method Fisher Score, applied to Fisher's Iris data (see Section 3.1 for details on data used). Feature ranking in this example was invariant to k (see Figure 2.4.1).

- **Gini Index (GINI)**. The Gini index (2, 10) measures the power a feature has in discriminating between classes. The Gini index is typically defined for categorical features, but can be extended to continuous features by discretization. We begin by binning each feature into b bins; in all cases we used Sturges' rule (39) to calculate $b = \lceil \log_2 N \rceil + 1$. Continuous feature values are thus converted to categorical variables; for the α^{th} feature denote these values $v_1(\alpha), \ldots, v_b(\alpha)$. We then compute a matrix $p_{ij}(\alpha)$, which is defined as the number of data samples with pseudo-label (class) j that have discretized value $v_i(\alpha)$ with respect to feature α. The Gini index for the value $v_i(\alpha)$ of the categorical variable is defined as

$$G\left(v_i(\alpha)\right) = 1 - \sum_{j=1}^{k} p_{ij}(\alpha). \tag{2.2.6}$$

  There are two extremes to G. If all the samples having attribute value $v_i(\alpha)$

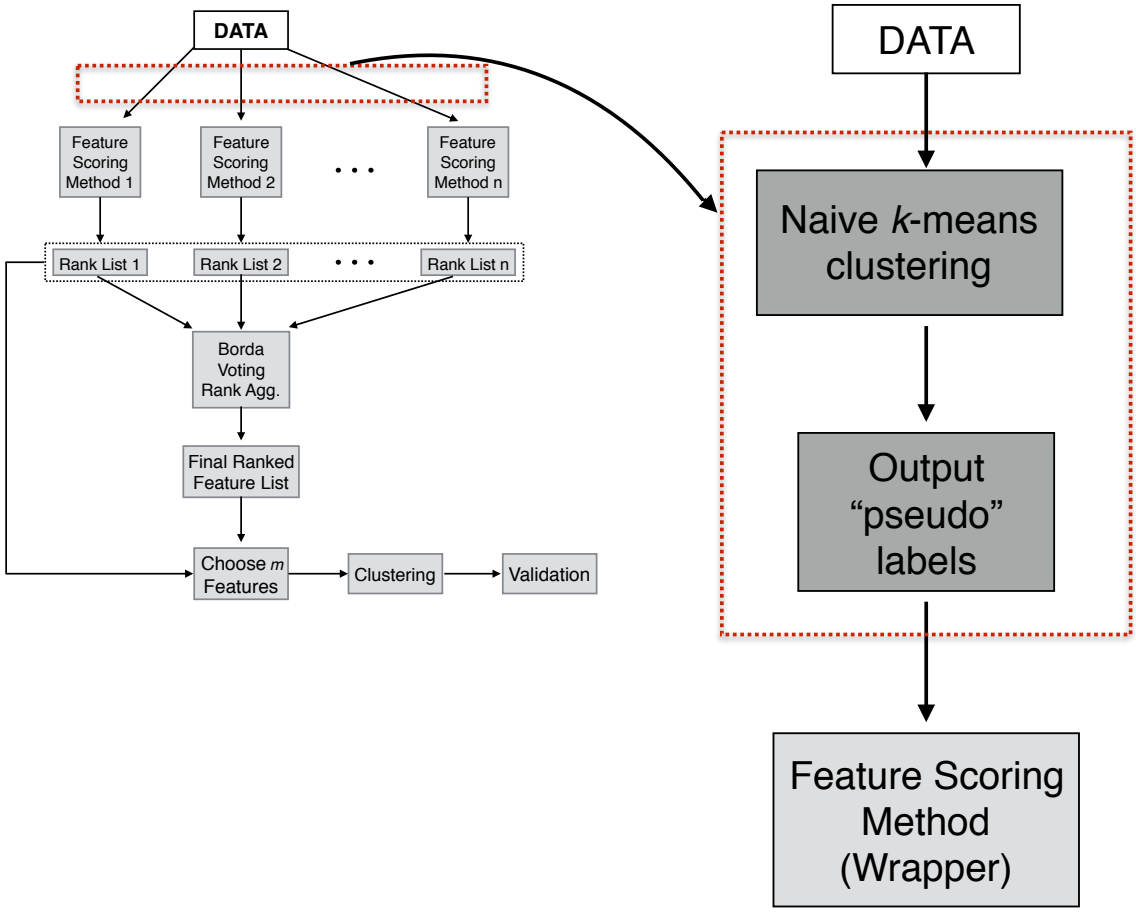


FIGURE 2.2.2: Creating pseudo-labels for wrapper feature methods requires extra steps in the pipeline

belong to the same class, then $G = 0$. On the other hand, if class membership is divided equally for attribute value $v_i(\alpha)$, then $G = 1 - 1/k$. Lower Gini index values therefore indicate better discriminatory power. To assign a Gini value to an entire feature, and not just one of that feature's categorical values $v_i(\alpha)$, we form a weighted average of $G\left(v_i(\alpha)\right)$. Let n_i be the number of samples with discretized feature value $v_i(\alpha)$, and $n = \sum_i n_i$. Then the Gini index for feature

α is given by

$$G(\alpha) = \frac{1}{n} \sum_{i=1}^{b} n_i G\left(v_i(\alpha)\right). \tag{2.2.7}$$

Again, lower values of $G(\alpha)$ indicate better features.

- **Entropy (ENTR)**. Class-based entropy (2, 32) has a similar goal to the Gini index — to measure the amount of mixing between classes — but has a more rigorous theoretical foundation in information theory. Using the same notation and quantities introduced for the Gini index, the entropy for categorical value $v_i(\alpha)$ is

$$E\left(v_i(\alpha)\right) = -\sum_{j=1}^{k} p_{ij}(\alpha) \log_2 p_{ij}(\alpha), \tag{2.2.8}$$

and the Entropy for feature α is computed as the weighted average

$$E(\alpha) = \frac{1}{n} \sum_{i=1}^{b} n_i E\left(v_i(\alpha)\right). \tag{2.2.9}$$

As with GINI, a smaller value of $E(\alpha)$ indicates a better feature.

- **Fisher Score(FISH)**. Fisher Score (FISH) (11, 17) measures the ratio of the average inter-class separation to the average intraclass separation. Larger scores indicate features that contribute more to producing cohesive clusters that are well separated. Suppose the wrapper method has been used to split the data into K classes. Define p_i as the fraction of samples that belong to class i, and let $\mu_i(\alpha)$ and $\sigma_i(\alpha)$ be respectively the mean and standard deviation of the α^{th} feature restricted to class i. Then the Fisher Score for feature α is calculated as

$$F_\alpha = \frac{\sum_{i=1}^{K} p_i \left(\mu_i(\alpha) - \mu(\alpha)\right)^2}{\sum_{i=1}^{K} p_i \sigma_i(\alpha)^2}, \tag{2.2.10}$$

where $\mu(\alpha)$ is the mean over all classes for that feature.

- **Relevance (RELW)**. As we noted in the section on Filter Methods above, the word "Relevance" has been repeatedly re-used in machine learning for a wide variety of different criteria. Here, we refer to a specific notion of Relevance (13, 33), which is the amount of dependency between a given feature and (in this case) the pseudo-labels necessary for wrapper methods. If we call the set of N pseudo-labels c, then the relevance of a feature is its mutual information with c:

$$\mathrm{Rl}(\alpha) = I[v(\alpha), c]. \tag{2.2.11}$$

  Larger values of RELW indicate features that carry more information about the pseudo-class labels.

- **Minimum Redundancy Maximum Relevance (mRMR)**. Feature selection based on either RELW or RED alone may not be ideal- without considering redundancy, unnecessary, noisy features may be retained; yet in some cases eliminating certain redundant features could result in a sub-optimal data-set, causing degradation in performance (21). Therefore the "best" features should theoretically have low redundancy and high relevance (13, 33). mRMR simply combines these two measures using

$$\mathrm{mRMR}(\alpha) = \mathrm{Rl}(\alpha) - \mathrm{Rd}(\alpha). \tag{2.2.12}$$

  Previous work used mRMR as a feature selection method for classification (33) and we use the same definition to compute a score for each feature for clustering. Larger values of mRMR indicate more desirable features.

For a listing of all the feature scoring methods we use and their abbreviations, see Table 2.2.1.

TABLE 2.2.1: Feature selection methods used in this study, left column contains feature methods categorized as Filters and the right column for Wrappers

Filter (abbr.)	**Wrapper (abbr.)**
Linear Predictability (LP)	Gini (GINI)
Spectral 1 (SP1)	Entropy (ENTR)
Laplacian (LAPL)	Fisher (FISH)
Relevance (RELF)	Relevance (RELW)
Variance (VAR)	Max Relevance, Min Redundancy (mRMR)
Leverage (LEV)	
Spectral 3 (SP3)	
Redundancy (RED)	

2.3 Rank Aggregation

To combine feature scores from multiple scoring methods, one cannot simply average scores since the scores produced by different feature scoring methods are on incommensurate scales. The mathematically sound procedure is to first convert the scores to ranks, and then compute a set of aggregate ranks. There are a wide variety of voting methods available to compute aggregate ranks (8), each of which differ in the number of criteria they satisfy that are considered desirable by Social Choice theorists. We use a simple, classical method originally due to the French political scientist and mathematician Jean-Charles de Borda (7). Table 2.3.1 shows a simple example of using the borda count method to determine the winner of a set of voters. There are more sophisticated voting methods: however, many of them scale well with the number of electors but poorly with the number of candidates — for example, the computational cost of the Schulze method (37) increases as n^3, where n is the number

of ranked candidates. In contrast to most voting applications, aggregating the results of feature scoring algorithms will generally feature vastly more candidates (features) than voters (methods). Rank aggregation methods, particularly Borda scoring, have proven useful outside of social choice. For example to aggregate web search results or ratings from multiple raters (16).

TABLE 2.3.1: Toy voting example using Borda Scoring. α_2 is the Borda winner.

Number of Voters	**6**	**4**	**1**
1st Choice	α_1	α_2	α_3
2nd Choice	α_2	α_3	α_1
3rd Choice	α_3	α_1	α_2

Candidate	**Borda Score**
α_1	13
α_2	14
α_3	6

We produce a set of consensus ranks as follows. We begin with a set of M scores, one for each feature scoring method. Each of the M score sets contains p scores, one for each feature:

$$\left\{S_i^{(1)}\right\}_{i=1}^{p}, \ldots, \left\{S_i^{(M)}\right\}_{i=1}^{p}. \tag{2.3.1}$$

The set of feature scores for each method are then converted into ranks $1, ..., p$ where 1 is the highest (best) rank and p the lowest (worst):

$$\left\{R_i^{(1)}\right\}_{i=1}^{p}, \ldots, \left\{R_i^{(M)}\right\}_{i=1}^{p}. \tag{2.3.2}$$

The Borda count or score for a given feature is equal to the number of features ranked lower than they are. This is computed in a two-step process, whereby we first form a

set of partial counts for each of the M scoring methods using

$$B_k(c) = \sum_{j=1}^{p} \left[R_j^{(i)} < c \right], \tag{2.3.3}$$

where the notation $[P]$ (the Iverson bracket) is equal to one if P is satisfied and zero otherwise. Finally, Borda scores for each feature are computed via

$$B(c) = \sum_{k} B_k(c). \tag{2.3.4}$$

To rank the features from best to worst, sort the Borda scores $B(c)$ in descending order.

2.4 Clustering

We performed all clustering experiments with k-means (sometimes called Lloyd's method) (30). The k-means clustering objective is to split the samples into a specified number of clusters k so that the total sum of the squared Euclidean distances of each point to its cluster center is minimized. We use k-means for two purposes in this study: (1) to generate pseudo-labels for feature scoring methods which are Wrappers and (2) to cluster the data based on the best subset of features judged by each method. For (1), k was set to 3; Figure 2.4.1 demonstrates insensitivity of feature ranking to this choice. For (2), we set k to be the number of *a priori* known classes in the data. For example, for the Iris data (17) we set $k = 3$. In both cases we used the k-means++ algorithm (4) to set the initial cluster centers. Pseudo-labels were generated using a single k-means clustering; cluster assignments for the feature subsets were obtained

by running k-means ten times with different random k-means++ initialization and choosing the clustering with the best value of the k-means objective function.

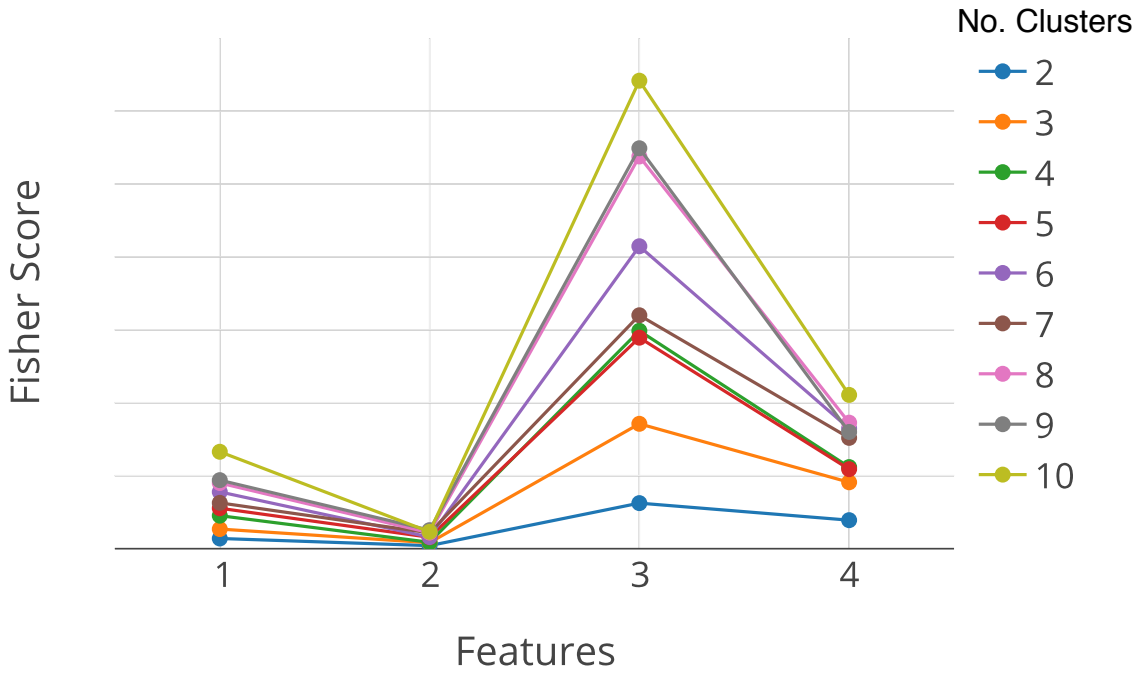


FIGURE 2.4.1: The effect of changing the number of pseudo-classes k on FISH scores for the Iris data. The larger the score, the higher ranking the feature. Absolute values of the score change with k, but all values of k yielded an identical feature ranking.

2.5 Validation

To validate the performance of each feature scoring method and the rank aggregation method, we used both internal and external criteria. Internal criteria require no information about the data beyond the clustering itself, while external criteria score the calculated cluster assignment against a set of known cluster labels (2). It is possible to achieve good results on internal criteria with cluster assignments that are

inaccurate because internal validation measures essentially measure how well the data fits the model, where the model is the algorithm assigning clusters. Just because the data seems "clumpy" or "clusterable" does not mean that one has found the correct cluster assignments! However, internal criteria are still important, particularly in unsupervised learning problems, since in a real clustering situation the real labels are unknown and internal criteria are the only way to choose among different cluster assignments.

For internal validation, we used the sum of the squared (Euclidean) distances (SSD) between each data point and its corresponding cluster center:

$$\text{SSD} = \sum_{k=1}^{K} \sum_{l=1}^{N_k} \|\mathbf{C}_k - \mathbf{d}_{lk}\|^2 . \tag{2.5.1}$$

Here, K is the number of assigned clusters and N_k is the number of data points assigned to each cluster. $\mathbf{C}_k$ is the center of cluster k, and $\mathbf{d}_{lk}$ is one of the N_k rows of the data matrix assigned to cluster k. The smaller the SSD, the better the clustering achieved. There are several features of the SSD which are appropriate for our study but would make SSD an unwise choice in other situations. First, the SSD is the objective function for k-means clustering using Euclidean distance. Clustering via other methods or using other distance measures would be better served by another criterion. Secondly, the SSD always decreases if more clusters are added, making it inappropriate to compare cluster assignments with different numbers of clusters. Third, the SSD scales with the size of the data matrix, so values obtained for different size data-sets are only comparable in a relative sense.

When true class labels are available, there are many methods designed to compute a similarity score between two partitions of objects (41). We use an information

theoretic measure from Strehl and Ghosh (38) that we abbreviate to NMI in this paper. If we denote the set of true cluster labels as T and the estimated labels derived from clustering as E, the measure is the following normalized mutual information between the two sets of labels

$$\mathrm{NMI}(T, E) = \frac{I(T, E)}{\sqrt{H(T)H(E)}}. \tag{2.5.2}$$

For any set of labels L (true or estimated) $H(L)$ is the entropy of the set. If we define $P_k = N_k/N$ as the frequency of label k in L, then

$$H(L) = -\sum_k P_k \log P_k. \tag{2.5.3}$$

Similarly, using P_{kl} as the joint probability distribution of two sets of labels K and L gives

$$I(K, L) = -\sum_k P_{kl} \log P_{kl}. \tag{2.5.4}$$

$\mathrm{NMI}(T, E)$ varies between zero (no correspondence between the labels) and unity (perfect association).

3

Results

3.1 Benchmark Data

We tested each feature scoring method and our rank aggregate method on five publically available data-sets from the UCI machine learning database (29). Our primary application interest is biological data, so we chose biological calibration data. These five data-sets varied in the number of features, samples, and number of true classes. We chose multiple calibration data to assess the data-dependence of each feature scoring method, and to determine if rank aggregation smooths out performance fluctuations over different data. A brief description of each data-set follows; a summary of the number of features, samples, and true classes for each data-set in Table 3.1.1.

- **Iris**. The iris flower data-set is from the famous paper by R. A. Fisher (17). It is a commonly used data-set to test clustering and classification algorithms because it is small, with only 150 instances and 4 features known to cluster well on two of these features. The features are the measured length and width of the flower sepal and petals. The classes are the three species of iris.

- **Ecoli**. Protein localization data from the *E. coli* (*Escherichia coli*) bacterium (25). The classes to be predicted are the cellular localization sites of the proteins, such as cytoplasm, perisplasm, etc. Seven features were constructed from the the amino acid sequence information of 336 *E. Coli* proteins.

- **Yeast**. Protein localization data from the yeast *S. cerevisiae* (25). Locations

within the cell including the mitochondria and endoplasmic reticulum are the classes. Eight features were constructed from the amino acid sequence information of 1484 yeast proteins.

- **Tumor**. Clinically obtained breast tumor data (31). The two classes are benign and malignant tumors. The nine features are the cellular attributes of excised tumors such as clump thickness and the extent of abnormal nucleoli. Data were collected from 683 tumors.

- **Mouse**. Protein expression levels in genetically modified mice (24). The data were collected to identify marker proteins critical for learning in a mouse model of Down Syndrome. 1080 measurements were taken from 77 proteins; the eight classes represent eight forms of genetic modification.

TABLE 3.1.1: Data-sets used in this study. Each column contains the number of Features, Instances and Classes of the data-set which are named in the first column.

Name	**Features**	**Instances**	**Classes**
Iris	4	150	3
Ecoli	7	336	8
Yeast	8	1484	10
Tumor	9	683	2
Mouse	77	1080	8

3.2 Aggregate Ranking of Features

In Figure 3.2.1 we show heatmaps for feature ranks, converted from scores for all but RANKAGG, which naturally generates ranks. Each panel is for a different one of the five data-sets and we show feature ranks for each individual feature scoring

method along with RANKAGG. One can see some agreement on ranks; for example in the Mouse data (Fig. 3.2.1e) methods seem to split into one of two categories, based on whether they rank the first twenty or so features near the bottom (LP, SP1, RELF, RED) or near the top (GINI,ENTR,FISH,LAPL,VAR). When we say "top" we mean a rank near the *lowest* rank, which is unity. Despite some general agreement, there is clearly a tremendous amount of variability. There are many columns (features) in each of the panels that contain both dark blue and bright yellow, indicating that these features are ranked both the best and worst by different methods. These irregularities illustrate why choosing a single feature selection method for a given data-set can be difficult. The degree of inter-method agreement is greatest in the Iris data (Fig. 3.2.1a); roughly 70% of methods ranked features three and four either one or two, and RANKAGG also places these two features at the top. These are known to be the features that most accurately cluster the data. However, the Iris data is also a relatively easy problem; even naive methods like VAR are able to determine that feature four is important.

We then proceeded to select features as described in Section 2.1 and cluster the data as detailed in Section 2.4. We performed the clustering for the same number of features selected from each individual method as well as RANKAGG. We then calculated two metrics to assess clustering quality, as described in detail in Section 2.5: sum-squared Euclidean distance (SSD) and normalized mutual information (NMI). Figure 3.2.2 shows a bar graph of the SSD for each of the thirteen feature selection methods and RANKAGG, for each of the five data-sets. SSD does not compare to known class labels, and a lower SSD score indicates better clustering performance. From this figure, three observations are apparent. One, clustering performance within a given data-set varies widely with the chosen method. This is particularly seen for the

more complex data-sets(compare Iris to Mouse, for example); SSD differs by more than a factor of two in some cases. Two, a given method's performance can vary substantially across different data-sets; LEV is one of the best performing methods in the Iris data-set but among the worst in the Ecoli data. Three, RANKAGG is always among the top performers, and, importantly, is never the worst performing method.

Given that NMI uses the true class labels to assess clustering quality, it is likely a truer measure of performance. Figure 3.2.3 shows the NMI score for all methods and RANKAGG, for all five data-sets. The same patterns we saw in SSD are present here. There is inter- and intra-data-set variability. For example SP1 is one of the best methods in the Yeast and Mouse data, but one of the worst in the other three. LEV is the worst performing method in the Mouse and Ecoli data but the best for Yeast. Once again, RANKAGG is a highly consistent performer. It is close to or equal to the best method in all data-sets but Mouse, and always far from the worst method. Most methods (saving SP1 and LAPL) perform pretty poorly on Mouse, which probably explains RANKAGG's performance on that data.

A more comprehensive view of each method's performance across the five data-sets is shown in Figure 3.2.4. For each of the thirteen feature selection methods and RANKAGG, we computed the mean and the variance in NMI across all five data-sets. When plotting these against each other, the best methods should cluster towards the upper left of the figure. One can see that RANKAGG is tied with ENTR and FISH for the best mean performance, but with a bit less variance. RANKAGG therefore makes a good general-purpose feature selection algorithm with excellent worst-case performance. We note that this same kind of analysis was not performed for SSD, since it has no consistent scale across data-sets with different numbers of features and samples.

3.3 Inter-Ranker Agreement

We now turn to a different question: is it possible to improve the results of RANKAGG by selectively removing voters before tallying the votes? This was proposed in a recent paper on classification (36), but those authors only used five feature selection methods, so the opportunities for testing are severely limited. There might be two different (and opposite) rationales for purging the voter rolls before aggregation. The classification study assumed that most rankers will choose helpful features, so dropping methods that show the least agreement would remove weight for unhelpful features. In this case, one would want to drop voters in order to *increase* the inter-group similarity of the voters. If, however, one was concerned that only a few of the voters choose helpful features (for example in the Mouse data, see Figure 3.2.3e), then one might want to remove voters to *decrease* consensus. We consider this question in detail for unsupervised learning.

To quantify inter-ranker agreement, we used Kendall's W (27). Kendall's W is defined as:

$$W = \frac{12S}{m^2\left(n^3 - n^2\right)}, \tag{3.3.1}$$

where n is the number of items (features) ranked, m is the number of voters/rankers, and S is the following sum of squared deviations

$$S = \sum_{i=1}^{n} \left(R_i - \bar{R}\right)^2. \tag{3.3.2}$$

Here, R_i is the sum of the ranks given to item i by all the rankers, and $\bar{R}$ is the mean

of the total ranks:

$$\bar{R} = \frac{1}{n} \sum_{i=1}^{n} R_i \tag{3.3.3}$$

W takes values in $[0, 1]$, in which zero indicates complete lack of consensus among the rankers and one indicates perfect concordance.

We used a greedy backward elimination algorithm to determine which feature selection methods to drop. We begin by computing W for the entire set of thirteen methods. We then remove each method in turn from the set of rankers and recompute W; the selection method that, when dropped, yielded the greatest *increase* in W was removed, and we continued in like manner until only two feature selection methods remained. This algorithm was repeated for when the dropped method would produce the greatest *decrease* in W. The values of W at each iteration for each data-set are shown in Figures 3.3.1 & 3.3.3. For the case where we wished to *increase* consensus we label the Kendall's W as $W+$, for *decrease* in consensus $W-$. The $W+$ values in Figure 3.3.1 show a monotonic increase in W, often saturating for data-sets with fewer features. And Figure 3.3.3 showing $W-$ values monotonically decreasing to a minima and then increasing. This inflection point occurs because as the feature sets get smaller, the less opportunity for non-consensus. The minima suggests that there is a point at which the dropping disparate feature methods is no longer effective.

We clustered a set of RANKAGG features produced using the remaining feature selection methods at each step of the backward elimination algorithm and computed both SSD and NMI. We did this for all five data-sets. These results are shown in Figures 3.3.2 & 3.3.4. In the figures, we have scaled all SSD and NMI values relative to their data-set-specific values at iteration zero. In addition, even though a *decrease* in SSD is indicative of better performance, we have plotted the results so that values

above one represent an improvement in validation score for both SSD and NMI. Notice that the Iris data shows no or little change in clustering quality no matter how many voters we drop; however this again merely indicates the Iris data is a relatively simple problem with few features. Even for random ranking, there are only 24 possible rankings for the four Iris features and we use 13 voters.

3.3.1 IRA $W+$ Results

For $W+$, Figure 3.3.2 three of the remaining four data-sets (Ecoli, Yeast, Mouse) showed improved values of either SSD or NMI or both for reduced sets of rankers. However, recall that in a real unsupervised learning problem, we don't have access to true class labels. Therefore, in these circumstances we would not use NMI scores to decide how many rankers to keep. The only way in which increasing inter-ranker agreement could be a viable preprocessing step is if the reduced set of rankers yielding the largest SSD increase (which can be computed without labels) also yields the largest NMI increase, or at the very least an improved NMI. Furthermore, the only way to recommend this as a general algorithm is if these correlations between internal and external validation criteria are seen across many calibration data-sets.

A careful study of Figure 3.3.2 show that these criteria are not met. For the tumor data, trying to increase agreement has no positive benefits, but that does not necessarily rule out the algorithm in general. For the Mouse data we get a 20% change in SSD which (iteration 3) *does* correspond to the largest NMI increase, but the magnitude of the change in NMI is minuscule, making it not worth the extra computational time of additional classifications. Preprocessing for agreement is most promising in the Ecoli data; the largest change in SSD is in iterations 4-8, and NMI

increases modestly (about 5%) for those iterations. However, the Yeast data removes any hope that the (slight) improvements we see in Mouse and Ecoli will generalize. It is clear that we can obtain a much better RANKAGG clustering of this data by backward eliminating to iteration 6; the NMI for that set of selection methods is roughly 30% larger. However, this same iteration corresponds to the *worst* value for SSD that we see, meaning in a real scenario where only SSD values would be available, we would never select this improved set of rankers. Again, even when the algorithm performs consistently, the gains are very small. Due to these results do not recommend trying to systematically remove voters to increase consensus before RANKAGG for unsupervised learning.

3.3.2 IRA $W-$ Results

In order for the $W-$ IRA algorithm to be useful, we would expect to see a correlation between the minima of the W value and the validation measures. This would indicate that there is an ideal number of voters that should be dropped for the case where one believes that most voters are incorrect. The SSD and NMI in Figure 3.3.4 would also show a worst value at these iterations, with an increase or constant value for the remaining iterations. However this desired result was not observed. Firstly, like the $W+$ results, patterns of inconsistency between SSD and NMI values for $W-$ were observed which is problematic if one wanted to use $W-$ as a preprocessing step. Also there does not seem to be any correlation between the $W-$ (Figure 3.3.3) minimas and the validation minimas. For instance the Yeast data-set experienced a minima of $W-$ at iteration 9 yet the most decrease in NMI improvement was observed at iteration 6, with the remaining iterations oscillating between the minima and 1. Curiously, the

Tumor data-set validation measures remained relatively constant at 1 throughout all iterations. So even as the consensus between feature sets decreased it had no effected on the clustering performance of these features. This could be because most feature sets picked the same top m features but in differing order. Overall the results for *decreasing consensus* were erratic and lead us to the conclusion that removing voters in this manner is not recommended for unsupervised learning.

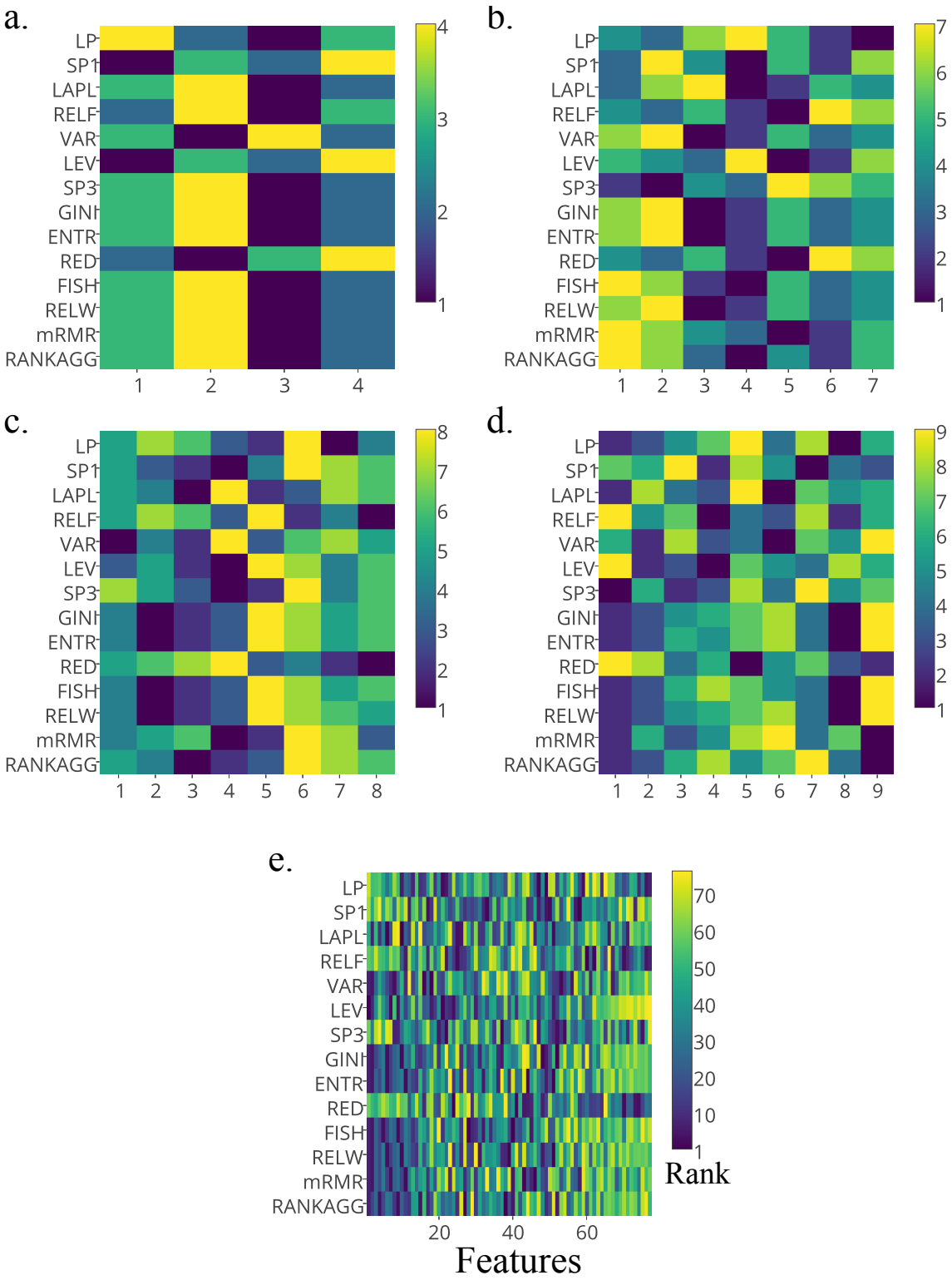


FIGURE 3.2.1: Feature Ranking. The heatmaps show feature ranks, converted from feature scores for all methods (see 2.2) except RANKAGG, which naturally generates ranks. Each panel is a different test data-set (see 3.1): a. Iris, b. Ecoli, c.Yeast, d. Tumor, e. Mouse. The colorbar to the right of each plot indicate the ranking. The best (top ranked) features are in dark blue.

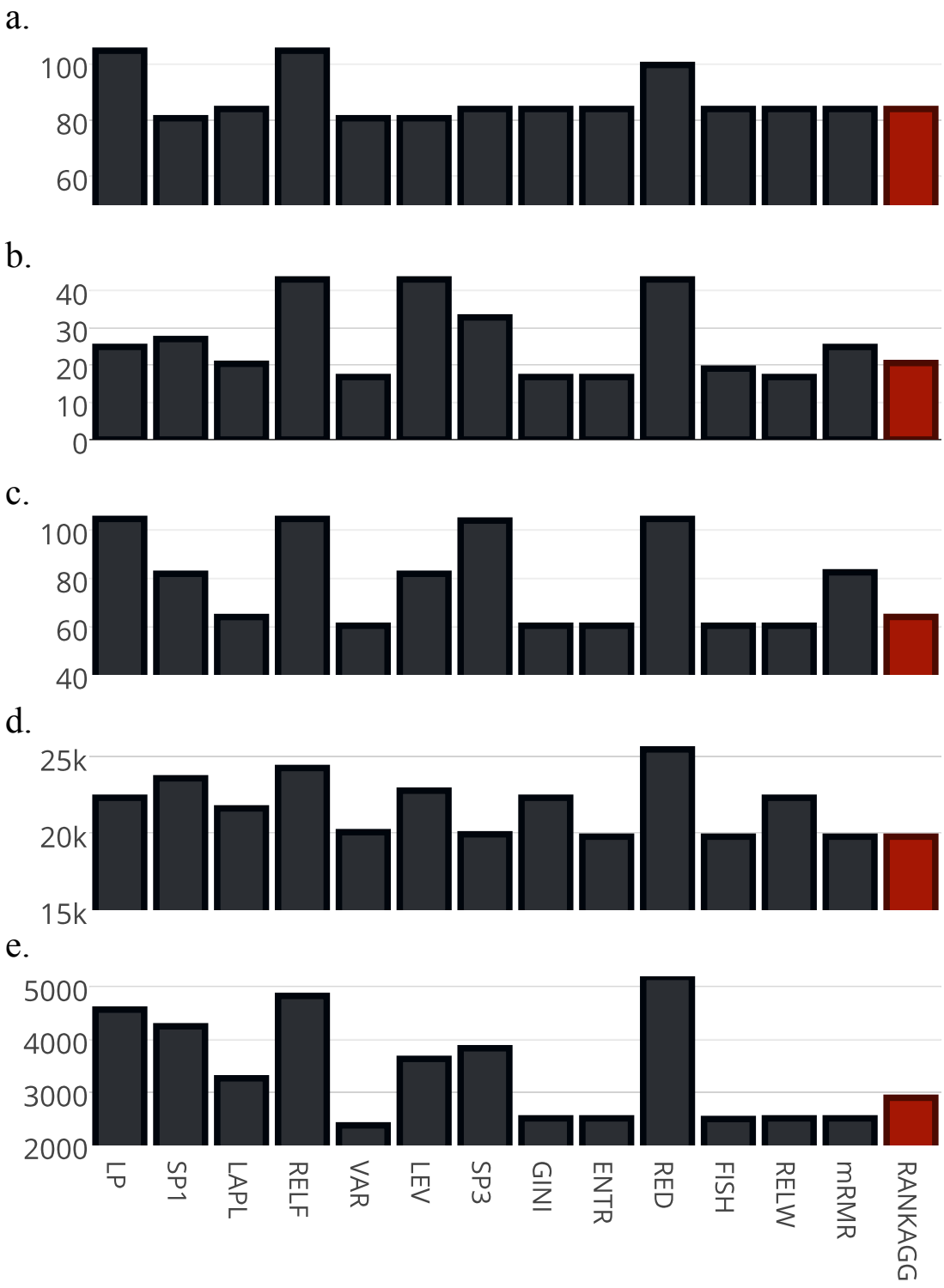


FIGURE 3.2.2: SSD bar graphs for a. Iris, b. Ecoli, c.Yeast, d. Tumor, e. Mouse. A lower SSD score means the samples were clustered closer to their respective cluster centers which may indicate that the clustering algorithm had better performance. The red bar is the results from RANKAGG and the black bars are the results from each feature method independently.

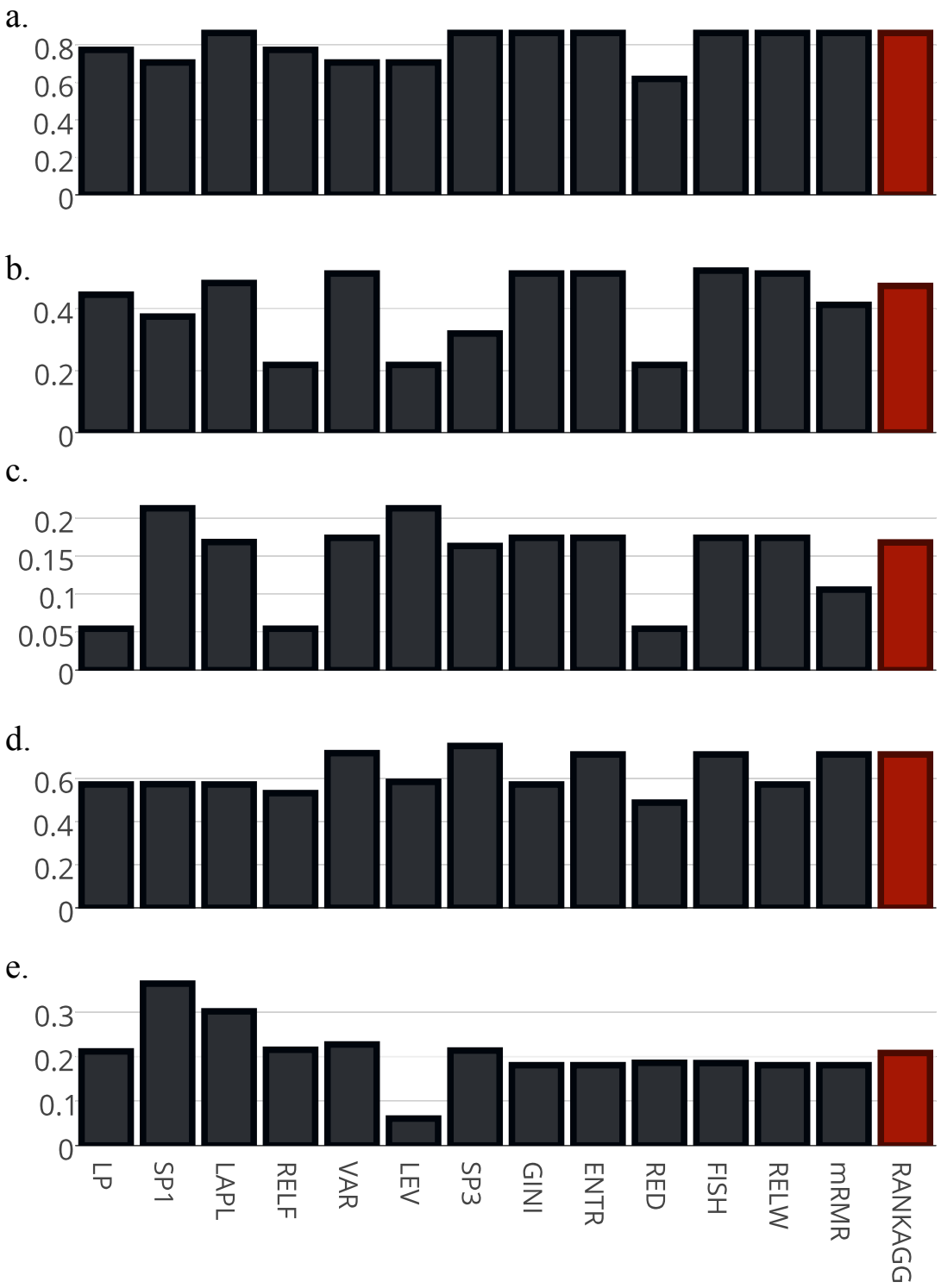


FIGURE 3.2.3: NMI bar graphs for a. Iris, b. Ecoli, c.Yeast, d. Tumor, e. Mouse. NMI is bounded below by zero (complete disagreement) and above by one (perfect correlation). The red bar is the results from RANKAGG and the black bars are the results from each feature method independently.

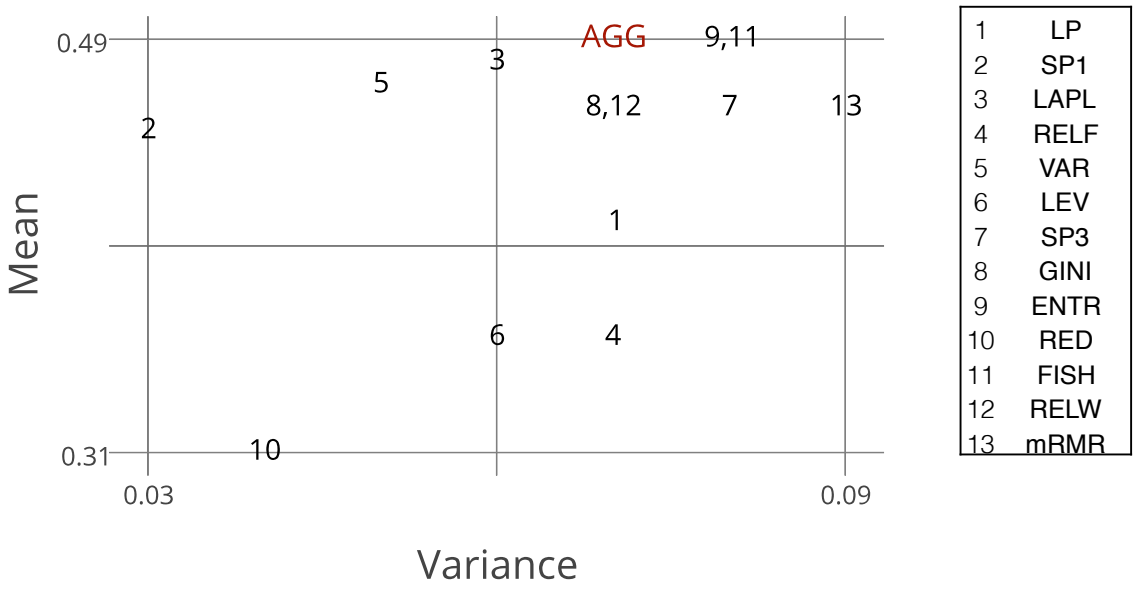


FIGURE 3.2.4: Mean and variance of NMI scores of each FS method and the rank aggregate across all data-sets. The lines mark the minimum, middle and maximum values calculated.

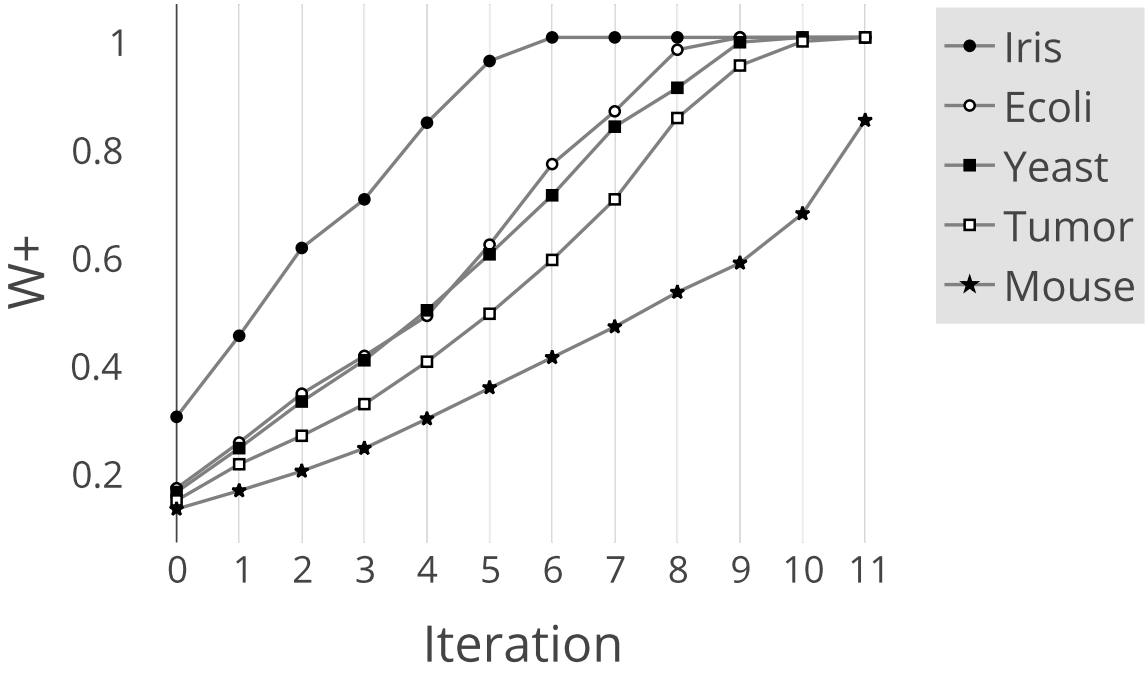


FIGURE 3.3.1: Value of Kendall's W at each step of our backward elimination algorithm that removes voters to increase inter-ranker agreement.

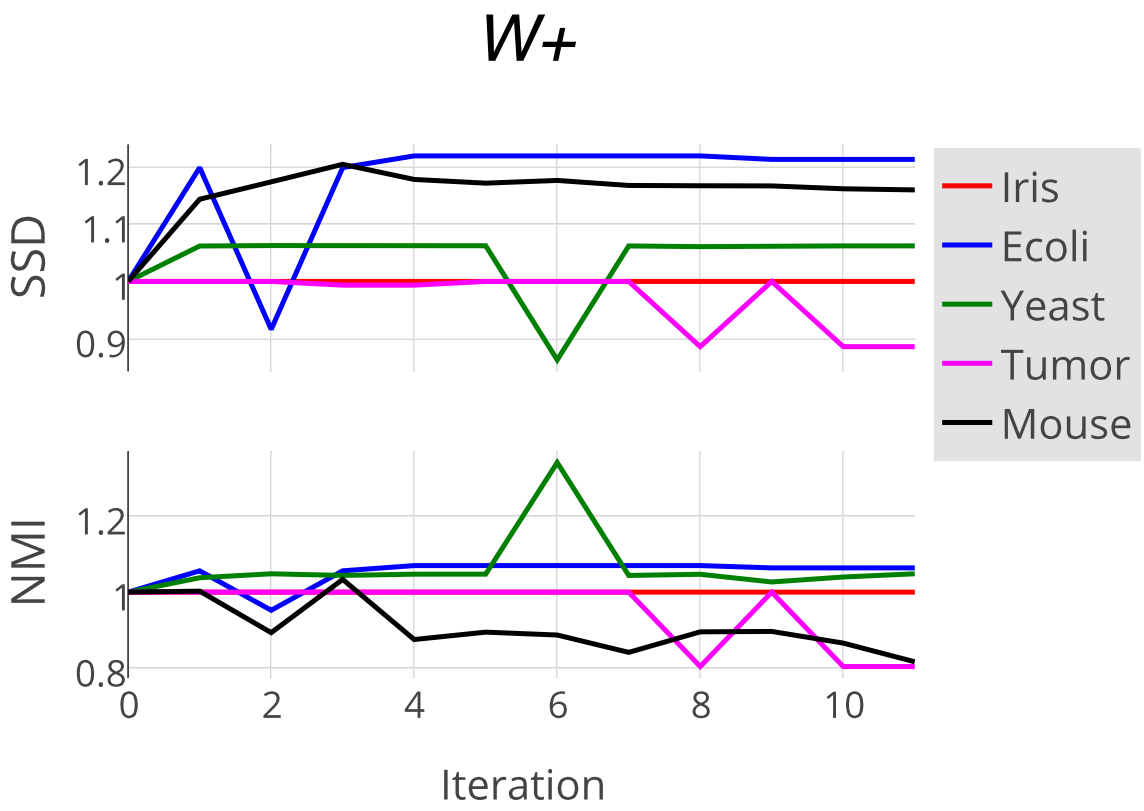


FIGURE 3.3.2: Effect of removing voters on RANKAGG clustering quality. Iteration zero represents the full, unaltered set of feature selection methods. Each iteration of the backward selection algorithm dropped the method that led to the largest increase in W. Values are normalized so that SSD and NMI have a value of unity for the full feature set. A value of either SSD or NMI greater than one indicates an improvement, when compared to including the full suite of methods in RANKAGG.

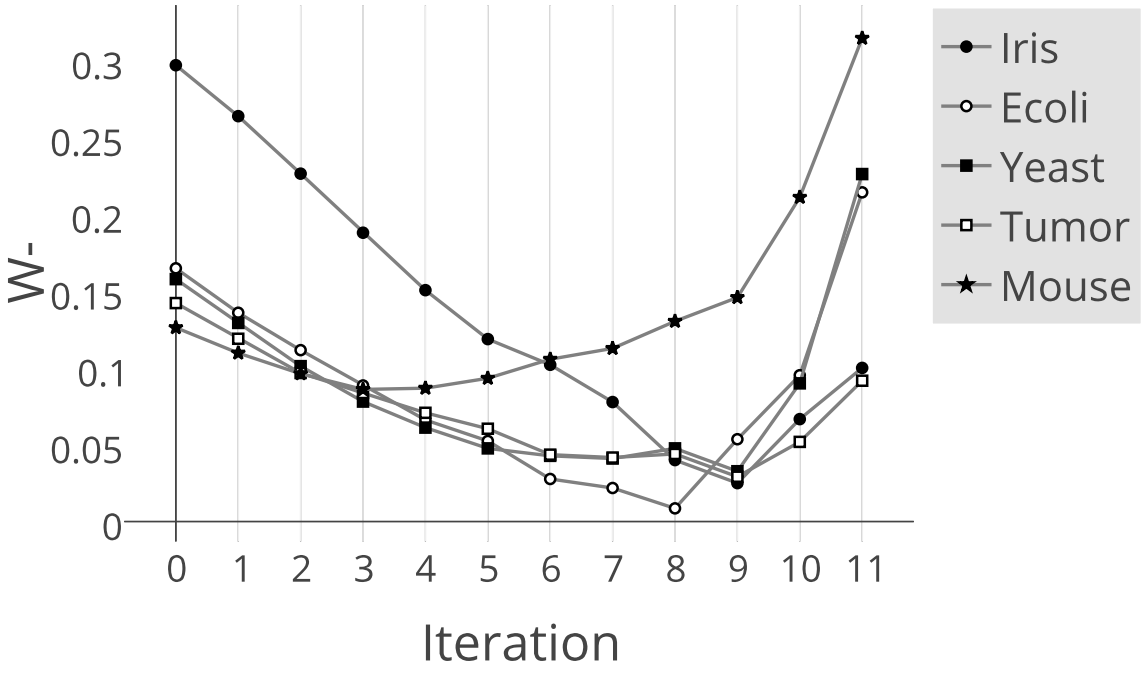


FIGURE 3.3.3: Value of Kendall's W at each step of our backward elimination algorithm that removes voters to increase inter-ranker agreement.

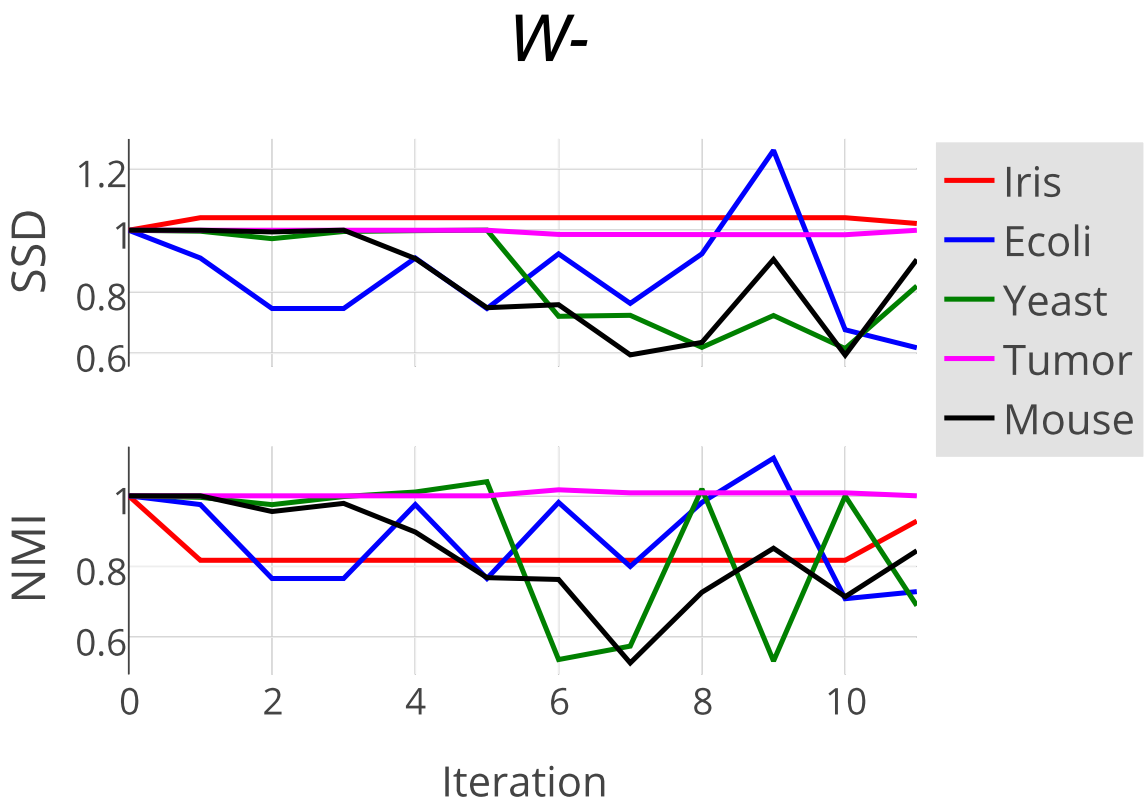


FIGURE 3.3.4: Effect of removing voters on RANKAGG clustering quality. Iteration zero represents the full, unaltered set of feature selection methods. Each iteration of the backward selection algorithm dropped the method that led to the largest decrease in W. Values are normalized so that SSD and NMI have a value of unity for the full feature set. A value of either SSD or NMI less than one indicates a decrease of improvement, when compared to including the full suite of methods in RANKAGG.

4

Discussion

4.1 Results -Highlights/Interpretations

This study clustered on feature subsets chosen from thirteen feature selection methods and an ensemble set (RANKAGG) of features selected from these methods. Our goal was to use RANKAGG to mitigate the known stability problems with most individual feature selection methods (5). Previous studies using ensemble features for classification ((12, 34, 35, 36)) showed that the ensemble features had greater stability/robustness with respect to data perturbations and outperformed single methods for subsequent classification performance. We studied this issue in unsupervised learning, in which we cannot rely on known data labels of a training set to guide the selection algorithms. We mention here that for some of the data we used for testing, especially those with very small numbers of features, feature selection may not be necessary and the overall clustering performance may be better when keeping all the features. However, our study was focused on relative performance and with the specific intent that RANKAGG will be especially useful for data in which feature selection is a necessity, as in genomic data when features outnumber samples by factors of ten or more.

RANKAGG did indeed show very promising performance. When applied to five different data-sets, RANKAGG had average performance equaling the best single feature selection method but with reduced variance. In the process of demonstrating the superiority of RANKAGG, we were also able to obtain a rather comprehensive

view of the performance of a wide variety of commonly used feature selection methods. Our results in Figure 3.2.4 showed that wrapper methods tended to perform better than filter methods. This is despite the fact that we used a very simple, uniform clustering scheme to produce the pseudo-labels for each data-set. No matter how many true clusters exist in the data, we used three clusters to generate the pseudo-labels. Once pseudo-labels are generated, many feature selection algorithms designed for classification (where one needs to know the labels of a training set) can be adapted to clustering. As far as filter method performance goes, the two best methods with respect to mean NMI were LAPL and VAR. LAPL is known to be a good method (23); what is notable is that its performance is quite different from both SP1 and SP3, despite the fact they are mathematically related. Given its simplicity, the performance of VAR is quite surprising, but also encouraging: VAR is probably the easiest and least computationally expensive feature selection method, making it convenient even for huge data-sets.

Finally, we carefully evaluated a previous claim that selectively removing voters before aggregation can improve ensemble performance (36). We developed a backwards elimination algorithm to prune the list of rankers, and a consistent internal criterion for picking the optimal subset of rankers. The Iris data was too simple to benefit from this procedure. Of the remaining four, in one data-set (Tumor) the full suite of rankers was optimal, in two more (Ecoli, Mouse) optimal subsets yielded only modest gains in clustering performance, and in one (Yeast) the *least* optimal feature set yielded the largest gains in classification performance. Given both these inconsistent results across data-sets and the extremely small gains in cluster quality when reduced sets of rankers were optimal, the computational burden of performing many additional clusterings and rank aggregations is not justified and can even be

detrimental.

4.2 Voting Methods

We aggregated ranks using Borda scoring, one of the simplest methods available. There are many voting methods available that differ in the degree to which they exhibit mathematically desirable properties (8). Some of these techniques (like Schulze (37)) become intractable for large numbers of candidates; others (Round Robin, Exponential Weighting, Stability Selection) do not. However, there is already some evidence from the classification literature that the extra computational burden of these more sophisticated methods is not reflected in better aggregate rankings (14). This study compared the effect of nine rank aggregation techniques for ensemble feature selection on classification performance, and they concluded that besides one exceptionally low performer, different voting methods were statistically indistinguishable from one another. It would be useful to revisit this question in the context of unsupervised learning, since it would be valuable if a simple technique like Borda scoring is not only sufficient but just as potent as more computationally expensive methods. Although it was shown that different voting methods did not make a big difference in that paper, they didn't explore the differences between voting methods that follow the condorcet criterion or not. Voters that follow condorcet criterion are often seen as the "true" winner, unfortunately there exists the condorcet paradox where it does not exist. Borda scoring does not obey condorcet criterion. It may be that it does not matter for our purposes but still interesting.

4.3 Future Directions/Extensions

There are several extensions to our current study that would be worth pursuing. One obvious direction is performance comparison on many more data-sets of varied composition. This would extend the robustness results for RANKAGG, provide a compelling, controlled comparison of a wide variety of individual feature selection methods, and give a more comprehensive view of the pitfalls of trying to manipulate the voter rolls. Another extension would be to have an even larger suite of methods. Given the relative dominance of wrapper over filter methods, adding more wrapper methods would likely produce the greatest gain for RANKAGG. RELIEFF (28) is one example of a wrapper method designed for classification problems that would be easy to adapt to the unsupervised case, given the extra step of producing pseudo-labels. Finally, for ease of comparison, we always performed the final clustering using the known number of clusters in the test data-set. In new problems, this number is not available and some method, like the gap statistic (40), must be used to try to estimate the number of clusters. It would be interesting to compare performance when the number of clusters is not prespecified but instead estimated from the data.

Bibliography

[1] Python software foundation. python language reference, version 2.7.

[2] C. C. Aggarwal. *Data Mining*. Springer, 2015.

[3] S. Alelyani, J. Tang, and H. Liu. Feature selection for clustering: a review. In C. C. Aggrawal and C. K. Reddy, editors, *Data Clustering: Algorithms and Applications*, chapter 2, pages 29–66. Chapman & Hall/CRC, Boca Raton, FL, 2014.

[4] D. Arthur and S. Vassilvitskii. K-means++: The advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1027–1035, Philadelphia, PA, USA, 2007. SIAM.

[5] W. Awada, T. M. Khoshgoftaar, D. Dittman, R. Wald, and A. Napolitano. A review of the stability of feature selection techniques for bioinformatics data. In *2012 IEEE 13th International Conference on Information Reuse Integration (IRI)*, pages 356–363, Aug 2012.

[6] R. Bellman. Adaptive control processes: A guided tour. *Princeton University Press*, 1961.

[7] J.-C. Borda. Mémoire sur les élections au scrutin. *Histoire de l'Académie Royale des Sciences*, 1781.

[8] C. Börger. *Mathematics of Social Choice: Voting, Compensation, and Division*. SIAM, 2010.

[9] C. Boutsidis, P. Drineas, and M. W. Mahoney. Unsupervised feature selection for

the k-means clustering problem. In *Advances in Neural Information Processing Systesm 22 (NIPS 2009)*, pages 1–9, 2009.

[10] L. Breiman, J. Friedman, R. Olshen, and C. J. Stone. *Classification and Regression Trees*. CRC Press, 1984.

[11] J. G. Bryan. The generalized discriminant function: mathematical foundations and computational routine. *Harvard Educational Review*, 21:90–95, 1951.

[12] T. G. Dietterich. Ensemble methods in machine learning. In *Multiple Classifier Systems. MCS 2000. Lecture Notes in Computer Science*, volume 1857, Berlin, Heidelberg, 2000. Springer.

[13] C. Ding and H. Peng. Minimum redundancy feature selection from microarray gene expression data. *Journal of Bioinformatics and Computational Biology*, 03(02):185–205, 2005.

[14] D. J. Dittman, T. M. Khoshgoftaar, R. Wald, and A. Napolitano. Classification performance of rank aggregation techniques for ensemble gene selection. In *Proceedings of the Twenty-Sixth International Florida Arificial Intelligence Research Society Conference*, 2013.

[15] D. L. Donoho. High-dimensional data analysis: The curses and blessings of dimensionality. In *AMS Conference on Math Challenges of the 21st Century*, 2000.

[16] C. Dwork, R. Kumar, M. Naorand, and D. Sivakumar. Rank aggregation methods for the web. In *Proceedings of the 10th International Conference on World Wide Web*, WWW '01, pages 613–622, New York, NY, USA, 2001. ACM.

[17] R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(2):179–188, 1936.

[18] A. L. N. Fred and A. K. Jain. Data clustering using evidence accumulation. In *Proceedings of the 16th International Conference on Pattern Recognition*, pages 276–280, 2002.

[19] D. Grün, A. Lyubimova, L. Kester, K. Wiebrands, O. Basak, N. Sasaki, H. Clevers, and A. van Oudenaarden. Single-cell messenger rna sequencing reveals rare intestinal cell types. *Nature*, 525:251–255, 2015.

[20] D. Grün and A. van Oudenaarden. Design and analysis of single-cell sequencing experiments. *Cell*, 163(4):799—810, 2015.

[21] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

[22] N. Habib, Y. Li, M. Heidenreich, L. Swiech, I. Avraham-Davidi, J. J. Trombetta, C. Hession, F. Zhang, and A. Regev. Div-seq: Single-nucleus rna-seq reveals dynamics of rare adult newborn neurons. *Science*, 353(6302):925–928, 2016.

[23] X. He, D. Cai, and P. Niyogi. Laplacian score for feature selection. In *Proceedings of the 18th International Conference on Neural Information Processing Systems*, NIPS'05, pages 507–514, Cambridge, MA, USA, 2005. MIT Press.

[24] C. Higuera, K. J. Gardiner, and K. J. Cios. Self-organizing feature maps identify proteins critical to learning in a mouse model of down syndrome. *PLoS ONE*, 10(6):1–28, 2015.

[25] P. Horton and K. Nakai. A probablistic classification system for predicting the cellular localization sites of proteins. *Intelligent Systems in Molecular Biology*, pages 109–115, 1996.

[26] L. Jiang, H. Chen, L. Pinello, and G.-C. Yuan. Giniclust: detecting rare cell types from single-cell gene expression data with gini index. *Genome Biology*, 17(1):144, 2016.

[27] M. G. Kendall and B. B. Smith. The problem of m rankings. *Annals of Mathematical Statistics*, pages 275–287, 1939.

[28] I. Kononenko, E. Šimec, and M. Robnik-Šikonja. Overcoming the myopia of inductive learning algorithms with relieff. *Applied Intelligence*, 7(1):39–55, 1997.

[29] M. Lichman. UCI Machine Learning Repository, 2013.

[30] S. Lloyd. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.

[31] O. L. Mangasarian and H. W. Wolberg. Cancer diagnosis via linear programming. *SIAM News*, 23(5), 1990.

[32] T. Mitchell. *Machine Learning*. McGraw-Hill, 1996.

[33] H. Peng, F. Long, and C. Ding. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1226–1238, 2005.

[34] R. C. Prati. Combining feature ranking algorithms through rank aggregation. In *The 2012 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2012.

[35] Y. Saeys, T. Abeel, and Y. Van de Peer. Robust feature selection using ensemble feature selection techniques. In M. K. Daelemans W., Goethals B., editor, *Machine Learning and Knowledge Discovery in Databases*, volume 5212, Berlin, Heidelberg, 2008. Springer.

[36] C. Sarkar, S. Cooley, and J. Srivastava. Robust feature selection technique using rank aggregation. *Applied Artificial Intelligence*, 28(3):243–257, 2014.

[37] M. Schulze. A new monotonic, clone-independent, reversal symmetric, and condorcet-consistent single-winner election method. *Social Choice and Welfare*, 36:267–303, 2011.

[38] A. Strehl and J. Ghosh. A knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, 3:583–617, 2002.

[39] H. A. Sturges. The choice of a class interval. *Journal of the American Statistical Association*, 21(153):65–66, 1926.

[40] R. Tibshirani, G. Walther, and T. Hastie. Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(2):411–423, 2001.

[41] S. Wagner and D. Wagner. Comparing Clusterings – An Overview. Technical Report 2006-04, Universität Karlsruhe (TH), 2007.

[42] L. Wolf and A. Shashua. Feature selection for unsupervised and supervised inference: The emergence of sparsity in a weight-based approach. *Journal of Machine Learning Research*, 6, 2005.

[43] Z. Zhao and H. Liu. Spectral feature selection for supervised and unsupervised learning. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, pages 1151–1157, New York, NY, USA, 2007. ACM.

[44] Z.-H. Zhou. *Ensemble Methods: Foundations and Algorithms*. Chapman & Hall/CRC, Boca Raton, FL, 2012.