Abstract

The abstract goes here.

1 Introduction

I wish you the best of success.

mds
February 28, 2015

Problem Definition

Given a dataset $D_{m \times n}$ with a set of features $F = \{f_1, f_2, \ldots, f_n\}$, come up with a subset of features $\hat{F} = \{\hat{f}_1, \hat{f}_2, \ldots, \hat{f}_k\}$ (top k features, with $k < n$), such that

1. The structure of the data is maximally preserved (in terms of nearest neighborhood, clustering) in the reduced feature space.
2. The redundancy is minimal, i.e., the features selected are as independent as possible (in terms of mutual information, correlation)

Motivation

1. Dimensionality reduction (avoid the curse of dimensionality)
2. Interpretability of Features in the reduced set (without transforming the feature space)

Approach

In the unsupervised feature selection technique (*MCFS*) for multicluster data Cai et. al. [2, 3, 1], they addressed the structure preserving in reduced feature set (by using nearest neighbors) in their optimization problem itself. In this paper, we address it differently, by assigning importance scores to the features according to their contribution in capturing the overall structure in the data. Then we use simple greedy / spectral clustering based approach to ensure the minimal overlap between features selected.

In this paper we propose a few algorithms for feature selection. We start with normalized mutual information matrix as the similarity matrix that will be used to capture the overall structure in the data (in terms of the total mutual information) as well as for finding the redundancy / overlap of the selected features in terms of mutual information.

Algorithm **Variable Importance with Mutual Information Component Analysis** (*VIMICA*) - assigns **importance scores** to each of the features based on the contribution of that particular feature in capturing the overall mutual information (needed to retain the overall structure of the data in the reduced feature space).
Since we have obtained the feature importance scores, we are now ready to come up with algorithms for feature selection. We propose the following 4 different algorithms:

1. *NaiveFS*: Being naive, this algorithm straightaway selects features based on the importance scores only (it does not aim at minimizing mutual information / maximizing uncertainty between the features selected).
2. *NaiveRSPFS*: Selects features (leaf nodes) that are nearest to the root of the tree generated with recursive spectral partitioning. The intuition is that the features that

Algorithm: COMPUTENMI (Computes Normalized Mutual Information Matrix)

Input: Dataset D with feature set $F^n = \{f_1, f_2, \ldots, f_n\}$
Output: Normalized Mutual Information Matrix M

begin
  `/* Discretize */`
  foreach *feature* f *in* F^n **do**
    if f *is not categorical* **then**
      use some binning technique (e.g., equal-depth, chi-square-merge etc.) to discretize f
  `/* Compute Normalized Mutual Information matrix` $M_{n \times n}$ `*/`
  foreach i, j **do**
    $M[f_i, f_j] = \frac{I(f_i, f_j)}{\sqrt{(H(f_i)H(f_j))}}$ (where $I(f_i, f_j) = H(f_i) - H(f_i|f_j) = H(f_j) - H(f_j|f_i) \leq \min(f_i, f_j) \Rightarrow 0 \leq M[f_i, f_j] \leq 1$)
  return M

Algorithm: VIMICA Assigns importance scores to features based on their contribution to capturing the overall mutual information

Input: Dataset D with feature set $F^n = \{f_1, f_2, \ldots, f_n\}$
Output: Importance Scores assigned to the features

begin
  $MIthreshold \leftarrow 90\%$
  $M \leftarrow computeNMI(D)$
  Do eigen-analysis of M
  Discard all but the k orthonormal Principal Components that capture at least $MIthreshold$ overall mutual information, find a k s.t. $\frac{\lambda_1+\lambda_2+\ldots+\lambda_k}{\lambda_1+\lambda_2+\ldots+\lambda_n} >$ MIthreshold
  From the loadings matrix compute the contribution of each of the features for k principal components, e.g., compute the contribution $c_{i,r}$ of a feature f_i for r^{th} principal component $PC_r = \sum_{j=1}^{n} w_j.f_j$ as $c_{i,r} = \frac{|w_i|}{\sum_{j=1}^{n} |w_j|}$
  Finally compute the importance score for each feature by taking an weighted average of its contribution to each of the top k principal components, weighted by corresponding eigenvalues (how much total mutual information each component capture), e.g., $score_i = \sum_{r=1}^{k} \lambda_r.c_{i,r}$
  return *scores*

are very discriminative (in terms of mutual information) may get separated out into a different partition with recursive spectral partitioning on the normalized mutual information matrix. Selecting the nearest leaf nodes from the root of the spectral partitioning tree will ensure that we select more of these discriminative features first. This technique is again naive because, it does not consider the importance of features towards capturing the overall mutual information (and the overall structure) in the dataset, although it generally ensures that the features selected have less overlap in between in terms of mutual information.

3. *GreedyFS*: Uses the feature importance scores along with a greedy heuristic (based on average mutual information) to avoid feature redundancy / overlap (in terms of mutual information). Ensures that the next feature selected with the algorithm has high importance score and at the same time has minimum average mutual information with already selected features.

4. *RSPFS*: Uses the feature importance scores along with a different (greedy) heuristic (using feature distance in the tree generated with recursive spectral partitioning) to avoid feature redundancy / overlap (in terms of mutual information). The intuition is that in the spectral partitioning tree, the feature leaf nodes with higher distance in between will generally have least mutual information (the reduction in uncertainty of one feature in presence of the other will be lowest). While selecting the next feature it tries to select a feature which has a higher importance score and at the same time the feature has largest average distance with the features already selected.

We have done multiple experiments on different datasets to compare the quality of the features selected by our algorithm

Algorithm: NaiveFS - Selects features based on importance scores only

Input: Dataset D with feature set $\{f_1, f_2, \ldots, f_n\}$, Number of features to select k
Output: Reduced feature set $\{\hat{f}_1, \hat{f}_2, \ldots, \hat{f}_k\}$

begin
  $scores \leftarrow VIMICA(D)$
  Sort features in decreasing order by the importance scores
  return *top k features*

Algorithm: GreedyFS - Selects features based on importance scores weighted by inverse average mutual information

Input: Dataset D with feature set $F^n = \{f_1, f_2, \ldots, f_n\}$, Number of features to select k
Output: Reduced feature set $\hat{F^k} = \{\hat{f}_1, \hat{f}_2, \ldots, \hat{f}_k\}$

begin
  $scores \leftarrow VIMICA(D)$
  Sort features in decreasing order by the importance *scores*
  Select the most important feature as $\hat{f}_1$
  $\hat{F^k} \leftarrow \{\hat{f}_1\}$ `/* selected features */`
  $F^n \leftarrow F^n - \hat{F^k}$ `/* unexplored features */`
  $i \leftarrow 2$
  while $i <= k$ **do**
    $\hat{f}_i \leftarrow \arg\max_{f_j \in F^n} \frac{score_j}{\frac{1}{|F^k|} \cdot \sum_{r=1}^{i-1} I_{norm}(\hat{f^r}, f_j)}$
    $\hat{F^k} \leftarrow \hat{F^k} \bigcup \{\hat{f}_i\}$
    $F^n \leftarrow F^n - \{\hat{f^i}\}$
  return *top k features*

Algorithm: GetRSPTree - Creates the characteristic features tree by *recursive spectral partitioning* and thresholding on the first nontrivial eigenvector

Input: Normalized Mutual Information Matrix $M_{n \times n}$
Output: Spectral Partitioning Tree T_{sp}

begin
  `/* Compute the diagonal Degree matrix` D `*/`
  foreach i **do**
    $D_{ii} = \sum_j M_{ij}$
  $L = D - M$ `/* Compute the Laplacian matrix` L `*/`

  Use the first nontrivial eigenvector of L (e.g., the 2^{nd} smallest eigenvector) to partition the feature set F into two feature subsets F_1 and F_2 with minimum mutual information (i.e., maximum uncertainty) in between them.
  Recursively partition L_{F_1} and L_{F_2} to generate the spectral characteristic tree T_{sp}. (The root node represents all the features and the other nodes represent feature subsets, with the leaf nodes representing each individual feature.)
  return T_{sp}

Algorithm: NaiveRSPFS - Selects features corresponding to the k nearest leaf nodes in the tree obtained using recursive spectral partitioning

Input: Dataset D with feature set $F^n = \{f_1, f_2, \ldots, f_n\}$, Number of features to select k
Output: Reduced feature set $\hat{F^k} = \{\hat{f}_1, \hat{f}_2, \ldots, \hat{f}_k\}$

begin
  $T_{sp} \leftarrow GetRSPTree(M)$
  Select k nearest leaf nodes (represetning individual features) from the root node (representing all features) in T_{sp}
  return *top k features*

with the features selected with the existing methods. Below are some results.

Result of Experiments

In order to test the quality of the reduced feature set (in terms of retaining the structure of the data, retaining the neighbourhood, preserving the distances etc.) obtained with our unsupervised feature selection algorithms (*NaiveFS*, *GreedyFS* and *RSPFS*), we performed the following 3 experiments on several datasets. Also, we compared the results obtained with existing unsupervised feature selection algorithms (*MCFS*, *LSFS* and *FSFS*). Our algorithms can work straightaway on categorical data, where the existing methods need transformation to numeric.

1. Run *KMeans Clustering* on the datasets with original feature set and with the reduced feature set separately and compare the results.

   (a) First use KMeans *Elbow* method / *Silhouette* opti-

Algorithm: RSPFS - Selects features based on importance scores weighted by the feature distance in the tree obtained using recursive spectral partitioning

Input: Dataset D with feature set $F^n = \{f_1, f_2, \ldots, f_n\}$, Number of features to select k
Output: Reduced feature set $\hat{F^k} = \{\hat{f}_1, \hat{f}_2, \ldots, \hat{f}_k\}$

begin
- $scores \leftarrow VIMICA(D)$
- $T_{sp} \leftarrow GetRSPTree(M)$
- Sort features in decreasing order by the importance scores
- Select the most important feature as $\hat{f}_1$
- $\hat{F^k} \leftarrow \{\hat{f}_1\}$ `/* selected features */`
- $F^n \leftarrow F^n - \hat{F^k}$ `/* unexplored features */`
- $i \leftarrow 2$
- **while** $i <= k$ **do**
  - $\hat{f}_i \leftarrow \arg\max_{f_j \in F^n} \frac{score_j}{\frac{1}{|F^k|} \cdot \sum_{r=1}^{i-1} Dist_{T_{sp}}(\hat{f^r}, f_j)}$
  - $\hat{F^k} \leftarrow \hat{F^k} \bigcup \{\hat{f}_i\}$
  - $F^n \leftarrow F^n - \{\hat{f^i}\}$
- **return** *top k features*

mal width / *Calinsky* criterion / *BIC* EM to find the appropriate number (K) of clusters in the original dataset.

(b) Create K clusters in the dataset using first the original and then the reduced feature set.

(c) Finally compute a score (similar to randIndex) defined by the matches and mismatches as described below.

For each of the pair of points in a dataset, a match occurs *iff* that pair of points belong to

(a) the same cluster in both the original and the reduced feature space.

(b) different clusters in both the original and the reduced feature space.

Otherwise, a mismatch occurs for that pair of points (they are in different clusters in one case but in the same cluster in the other case).The comparison score is defined as $\frac{\#matches}{\#matches+\#mismatches}$, so that it's in between 0 (low score) and 1 (high score) always. The intuition is that if the features in the reduced feature set approximately retain the structure of the data and preserve the distances, the clustering in the original and reduced feature set should be similar, resulting in a high score.

The same experiment was repeated as ground truth validation with class labels provided.

2. Find *K-nearest neighbors* for each point in the dataset both with the original and the reduced feature set. Compare the *Jaccard similarity* score for K-nearest neighbors (for different K, e.g., $K = 3, 5, 10$) for a point p as $\frac{|KNN^p_{original} \bigcap KNN^p_{reduced}|}{|KNN^p_{original} \bigcup KNN^p_{reduced}|}$. Then compute the Average Jaccard Similarity for all the points in the dataset.

Again, the intuition being that if the features in the reduced feature set approximately preserve the neighborhood, the Jaccard similarity should be high.

3. Use *PAM* clustering with different initial clusters (K) both in the original and the reduced feature space. Then compare the average *silhouette* width of the clusters in both the spaces.

Results on different datasets

	numSelFeatures	RSPFS	NaiveRSPFS	NaiveFS	GreedyFS
Sonar (60 features)	2	0.654728725	0.580546265	0.62337421	0.6233742
	3	0.650269417	0.582822371	0.619100706	0.61910070
	4	0.676932367	0.607441472	0.606884058	0.60688405
	5	0.710330732	0.60367893	0.642047566	0.63173541
	6	0.724637681	0.621144556	0.625139353	0.71437198
	7	0.727842809	0.628112226	0.701923077	0.69031029
	8	0.689149015	0.637959866	0.652731327	0.70642883
	9	0.727982163	0.657004831	0.647157191	0.69109996
	10	0.710562988	0.667781494	0.672519509	0.67577108
	11	0.714836492	0.67131178	0.664111854	0.69941471
	12	0.718227425	0.680694909	0.682506503	0.69337606
	13	0.724219621	0.664065403	0.685525827	0.78051839
	14	0.726820884	0.659234485	0.703827573	0.714650687
	15	0.720039019	0.671125975	0.785860275	0.73072278
	16	0.722222222	0.673727239	0.700854701	0.813359346
	17	0.714279078	0.687198068	0.687430323	0.851077666
	18	0.717298402	0.711910071	0.678836864	0.822835377
	19	0.788322185	0.74015236	0.771924935	0.838628763
	20	0.831010777	0.707218506	0.669082126	0.876532885
	21	0.815310294	0.673448532	0.784513192	0.865849127
	22	0.804533631	0.668246005	0.799052397	0.875789669
	23	0.798541434	0.679672984	0.818236715	0.864827202
	24	0.791341509	0.680509104	0.826876626	0.841973244
	25	0.840858417	0.662857674	0.829431438	0.846850613
	26	0.834076551	0.658491267	0.852564103	0.842995169
	27	0.800213675	0.651802304	0.843366778	0.869425864
	28	0.80778521	0.654960981	0.840951319	0.857998885
	29	0.824739874	0.633407655	0.854375697	0.83769974
	30	0.864641397	0.633454106	0.841694537	0.821767001
	average	0.75281	0.65689	0.72799	0.76915

0.67744333	0.722686734	0.623931624	
0.660860275	0.719806763	0.623931624	
0.676793014	0.735974753	0.621144556	
0.667781494	0.751207729	0.621144556	
0.742474916	0.765282423	0.66722408	
0.727610554	0.766118543	0.601356373	
0.745169082	0.782980305	0.604422148	
0.76147343	0.773968785	0.604422148	
0.76885916	0.741824601	0.630527685	
0.804998142	0.740059457	0.632896693	
0.801746563	0.776291342	0.633779264	
0.807460052	0.770717205	0.623560015	
0.816933482	0.774619101	0.640746934	
	0.772551	0.6142	

Figure 1: Sonar Dataset (208x61)

KMeans.png

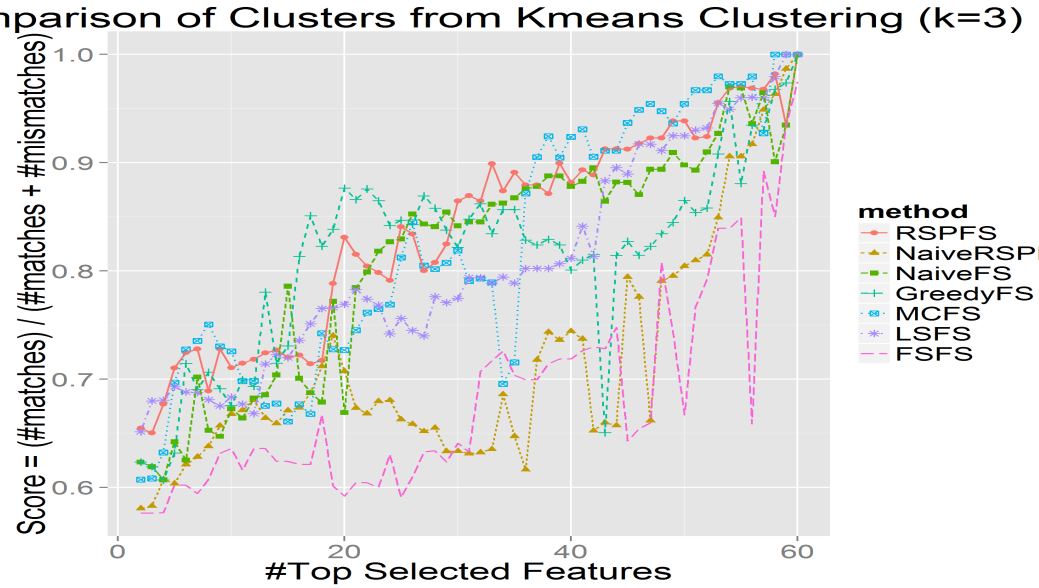

Figure 2: Sonar Dataset (208x61)

Conclusion

As can be seen from the above results, one of the techniques *GreedyFS* and *RSPFS* is performing quite well, sometimes better than all the existing techniques.

References

[1] Sanghamitra Bandyopadhyay, Tapas Bhadra, Pabitra Mitra, and Ujjwal Maulik. Integration of dense subgraph finding with feature clustering for unsupervised feature selection. *Pattern Recognition Letters*, 40:104–112, 2014.

Dataset		FSFS			GreedyFS			LSFS			MCFS			NaiveFS			NaiveRSPFS			RSPFS		
	NumSel	#Nearest Nbrs			#Nearest Nbrs			#Nearest Nbrs			#Nearest Nbrs			#Nearest Nbrs			#Nearest Nbrs			#Nearest Nbrs		
	Features	3	5	10	3	5	10	3	5	10	3	5	10	3	5	10	3	5	10	3	5	10
Sonar (60 features)	1	0.02	0.02	0.04	0.02	0.02	0.05	0.02	0.03	0.05	0.02	0.03	0.04	0.02	0.02	0.05	0.02	0.02	0.04	0.02	0.02	0.05
	2	0.08	0.08	0.09	0.08	0.10	0.12	0.06	0.08	0.09	0.09	0.13	0.13	0.08	0.10	0.12	0.09	0.10	0.09	0.07	0.06	0.10
	3	0.08	0.08	0.09	0.21	0.15	0.13	0.18	0.15	0.15	0.23	0.20	0.18	0.21	0.15	0.13	0.20	0.14	0.12	0.14	0.13	0.13
	4	0.17	0.13	0.10	0.23	0.17	0.14	0.23	0.19	0.16	0.25	0.20	0.18	0.23	0.17	0.14	0.23	0.15	0.12	0.19	0.16	0.14
	5	0.21	0.14	0.11	0.24	0.17	0.16	0.25	0.21	0.19	0.31	0.26	0.22	0.24	0.18	0.15	0.23	0.16	0.12	0.25	0.20	0.18
	6	0.21	0.14	0.11	0.26	0.20	0.19	0.25	0.20	0.18	0.33	0.30	0.26	0.24	0.18	0.16	0.23	0.16	0.13	0.27	0.21	0.19
	7	0.23	0.15	0.14	0.31	0.25	0.22	0.29	0.25	0.21	0.36	0.33	0.29	0.27	0.21	0.18	0.24	0.16	0.13	0.29	0.25	0.23
	8	0.23	0.16	0.14	0.31	0.27	0.22	0.33	0.30	0.26	0.38	0.36	0.32	0.29	0.22	0.20	0.25	0.17	0.16	0.32	0.27	0.25
	9	0.25	0.18	0.16	0.35	0.29	0.25	0.34	0.33	0.28	0.38	0.37	0.33	0.32	0.26	0.23	0.26	0.19	0.17	0.32	0.29	0.25
	10	0.26	0.20	0.17	0.34	0.29	0.25	0.34	0.34	0.29	0.41	0.40	0.35	0.35	0.29	0.27	0.28	0.19	0.18	0.32	0.29	0.25
	11	0.24	0.19	0.17	0.34	0.29	0.25	0.34	0.35	0.30	0.42	0.41	0.35	0.35	0.32	0.28	0.28	0.22	0.20	0.34	0.32	0.27
	12	0.26	0.20	0.17	0.35	0.31	0.28	0.36	0.36	0.31	0.42	0.42	0.35	0.3								
	13	0.26	0.20	0.17	0.38	0.35	0.30	0.38	0.37	0.32	0.44	0.44	0.36	0.3								
	14	0.25	0.22	0.20	0.38	0.33	0.31	0.38	0.39	0.32	0.44	0.46	0.36	0.3								
	15	0.25	0.22	0.20	0.40	0.36	0.33	0.39	0.40	0.33	0.44	0.46	0.36	0.4								
	16	0.29	0.28	0.23	0.42	0.39	0.35	0.41	0.41	0.33	0.45	0.47	0.37	0.4								
	17	0.29	0.28	0.23	0.45	0.41	0.38	0.44	0.41	0.35	0.44	0.47	0.38	0.4								
	18	0.31	0.27	0.24	0.45	0.44	0.40	0.43	0.43	0.36	0.44	0.47	0.38	0.4								
	19	0.35	0.29	0.26	0.46	0.43	0.41	0.45	0.46	0.38	0.45	0.48	0.40	0.4								
	20	0.34	0.31	0.25	0.48	0.46	0.43	0.46	0.47	0.39	0.48	0.50	0.41	0.4								
	21	0.35	0.28	0.25	0.49	0.50	0.45	0.46	0.48	0.39	0.49	0.51	0.42	0.4								
	22	0.35	0.28	0.25	0.51	0.51	0.46	0.45	0.49	0.41	0.49	0.52	0.44	0.4								
	23	0.35	0.28	0.26	0.53	0.51	0.47	0.48	0.51	0.44	0.50	0.52	0.45	0.4								
	24	0.37	0.30	0.26	0.53	0.52	0.49	0.50	0.53	0.45	0.50	0.53	0.46	0.4								
	25	0.34	0.29	0.25	0.53	0.51	0.49	0.49	0.53	0.45	0.52	0.53	0.47	0.4								
	26	0.35	0.31	0.28	0.55	0.52	0.50	0.50	0.54	0.45	0.52	0.54	0.47	0.4								
	27	0.41	0.33	0.29	0.55	0.53	0.50	0.50	0.55	0.46	0.52	0.54	0.47	0.5								
	28	0.40	0.37	0.33	0.58	0.54	0.51	0.51	0.54	0.46	0.54	0.55	0.48	0.5								
	29	0.41	0.38	0.34	0.59	0.56	0.53	0.51	0.54	0.47	0.54	0.55	0.48	0.5								
	30	0.40	0.37	0.34	0.59	0.55	0.54	0.52	0.55	0.47	0.54	0.55	0.49	0.5								
	Average	0.28	0.23	0.20	0.40	0.36	0.34	0.37	0.38	0.32	0.41	0.42	0.36	0.3								

Figure 3: Sonar Dataset (208x61)

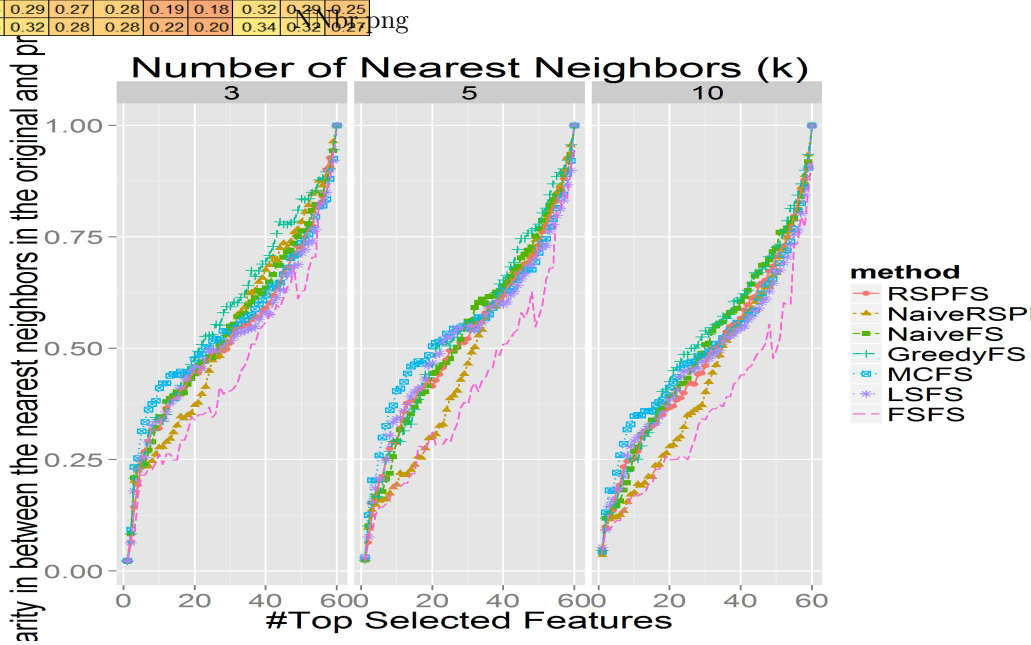

Figure 4: Sonar Dataset (208x61)

[2] Deng Cai, Chiyuan Zhang, and Xiaofei He. Unsupervised feature selection for multi-cluster data. In *16th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'10)*, 2010.

[3] Xiaofei He, Deng Cai, and Partha Niyogi. Laplacian score for feature selection. In *Advances in Neural Information Processing Systems 18*, 2005.

PAM.png

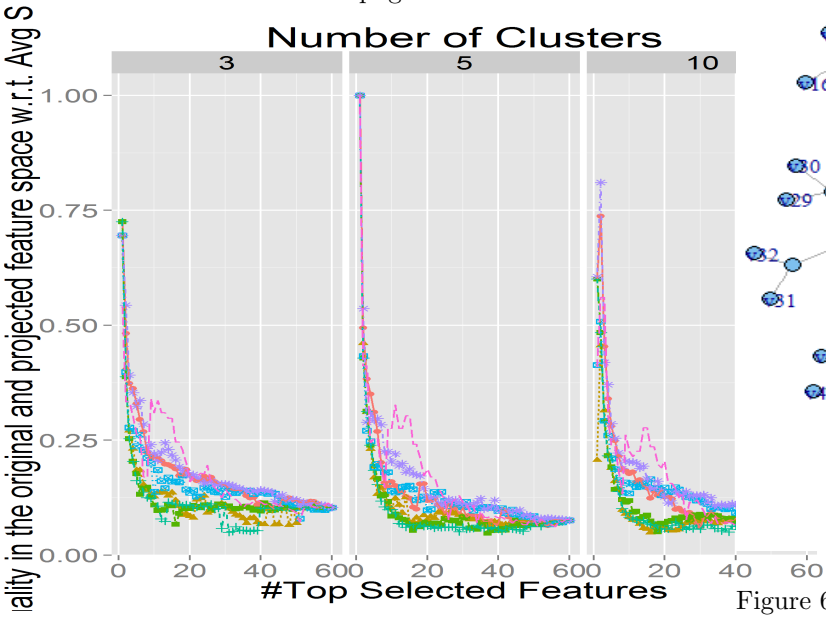

Figure 5: Sonar Dataset (208x61)

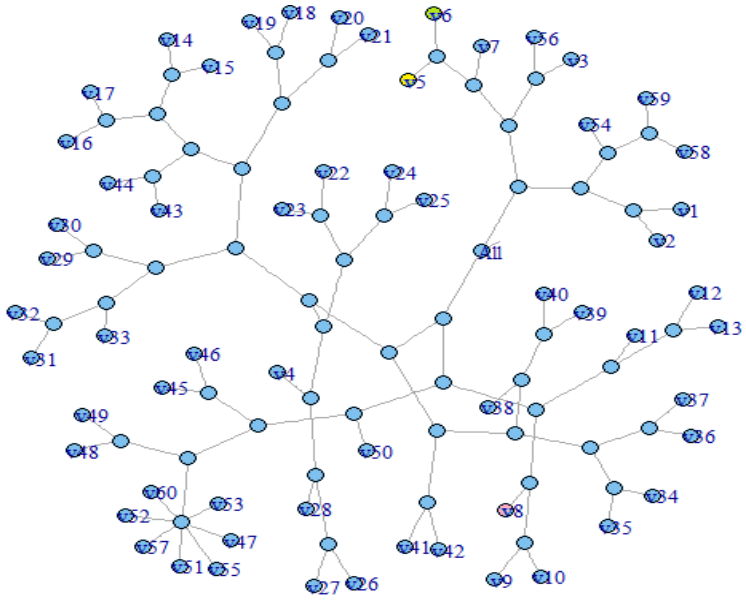

Figure 6: Tree constructed using Recursive Spectral Partitioning with Sonar Dataset: The features $v5$ and $v6$ have very small distance, where $v5$ and $v8$ have large distance in between them $\Rightarrow$ feature-pairs $v5$ and $v8$ are less redundant

KMeans.png

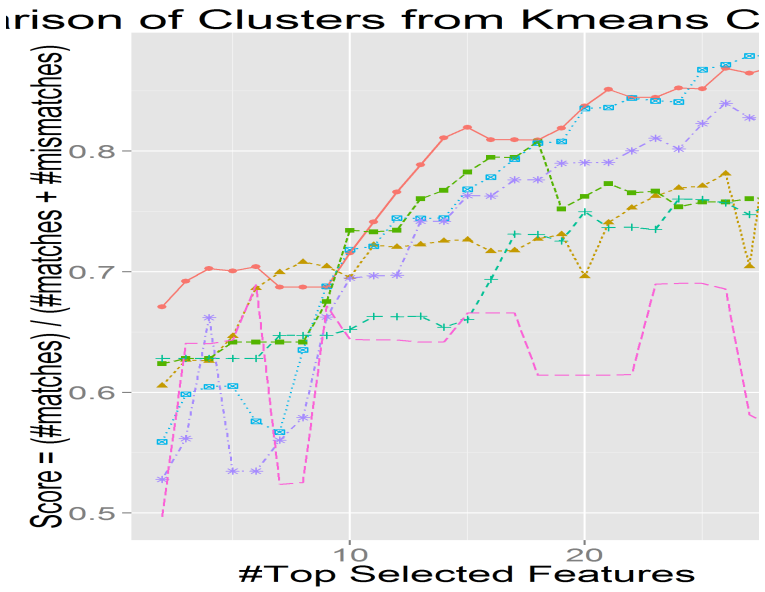

Figure 7: Spambase Dataset (4601x57)

NNbr.png

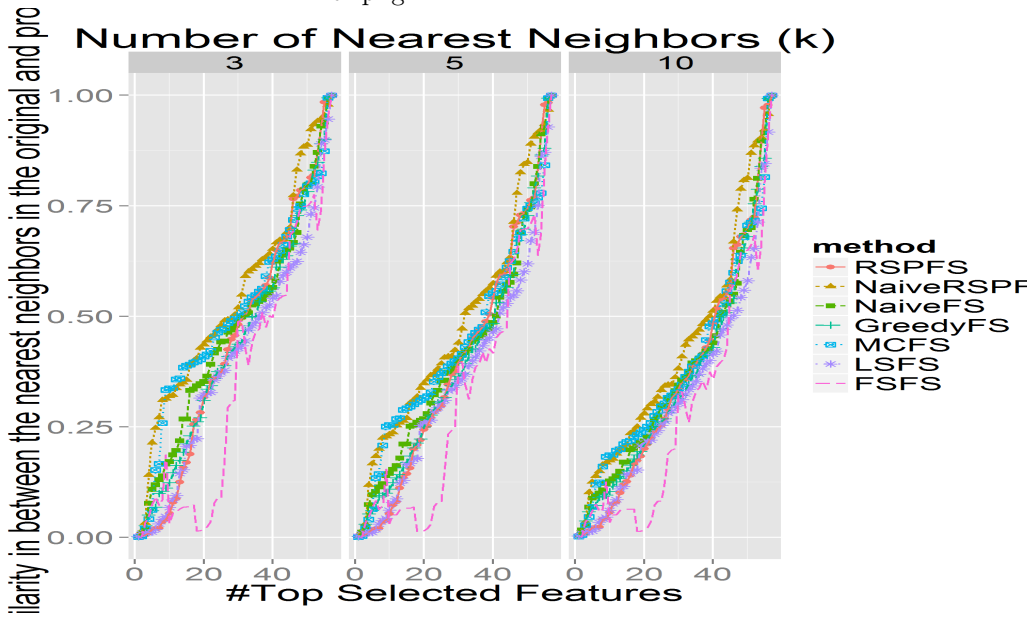

Figure 8: Spambase Dataset (4601x57)

PAM.png

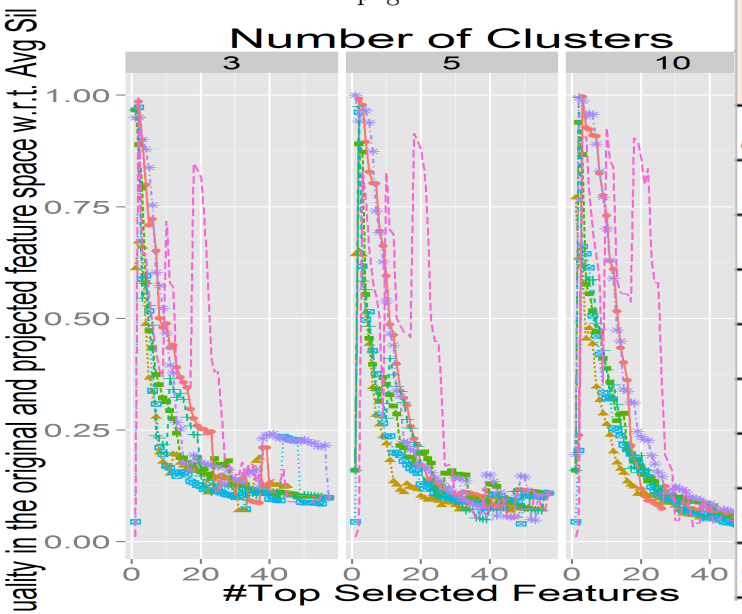

Figure 9: Spambase Dataset (4601x57)

Dataset	Dimension		Average Cluster randIndex Score with at most half of the feat						
	Rows	Cols	RSPFS	NaiveRSPFS	NaiveFS	GreedyFS	MCFS	LSFS	
Abalone	4177	8	0.91557	0.90883	0.8855	0.8796	0.89713	0.92999	(
Sonar	208	60	0.75281	0.65689	0.72799	0.76915	0.73314	0.72551	
Robot	135	90	0.58214	0.59327	0.61589	0.59575	0.60786	0.63346	(
SPECTF	80	44	0.62281	0.60433	0.61641	0.61811	0.62602	0.64572	(
LRS	531	101	0.70566	0.62288	0.66521	0.75181	0.89901	0.884	(
ionosphere	351	33	0.58465	0.68851	0.62953	0.72225	0.76215	0.73149	(
Chess	3196	36	0.49641	0.58833	0.56102	0.62619	0.63828	0.73026	(
splice	3190	60	0.54662	0.55257	0.5797	0.55405	0.56857	0.77664	(
spambase	4601	57	0.78173	0.71567	0.72758	0.69114	0.74651	0.71741	

Figure 10: Combined Results

Dataset	Average Jaccard Similarity with k-Nearest Nbrs with at most half of the features																						
	Dimension		FSFS			GreedyFS			LSFS			MCFS			NaiveFS			NaiveRSPFS			RSPFS		
	Rows	Cols	3	5	10	3	5	10	3	5	10	3	5	10	3	5	10	3	5	10	3	5	10
Abalone	4177	8	0.13	0.13	0.13	0.14	0.14	0.14	0.09	0.09	0.09	0.14	0.12	0.13	0.13	0.13	0.12	0.15	0.15	0.15	0.13	0.13	0.13
Sonar	208	60	0.28	0.23	0.20	0.40	0.36	0.34	0.37	0.38	0.32	0.41	0.42	0.36	0.37	0.35	0.32	0.33	0.26	0.23	0.37	0.35	0.31
Robot	135	90	0.33	0.30	0.30	0.50	0.47	0.47	0.38	0.34	0.37	0.43	0.43	0.45	0.50	0.46	0.45	0.43	0.41	0.39	0.43	0.41	0.39
SPECTF	80	44	0.22	0.19	0.21	0.37	0.33	0.35	0.32	0.31	0.32	0.32	0.28	0.32	0.37	0.33	0.35	0.32	0.27	0.30	0.34	0.31	0.33
LRS	531	101	0.35	0.30	0.29	0.50	0.45	0.45	0.25	0.22	0.24	0.42	0.41	0.43	0.39	0.36	0.38	0.41	0.36	0.35	0.35	0.31	0.32
ionosphere	351	33	0.24	0.22	0.22	0.34	0.32	0.35	0.24	0.21	0.21	0.34	0.32	0.36	0.31	0.27	0.28	0.33	0.30	0.32	0.32	0.29	0.28
Chess	3196	36	0.02	0.02	0.03	0.12	0.13	0.14	0.08	0.09	0.12	0.05	0.06	0.07	0.06	0.07	0.08	0.08	0.08	0.10	0.04	0.04	0.05
spambase	4601	57	0.08	0.07	0.06	0.21	0.17	0.15	0.18	0.15	0.13	0.32	0.25	0.19	0.26	0.21	0.17	0.34	0.26	0.21	0.18	0.15	0.13

Figure 11: Combined Results