Neighborhood Random Classification

Djamel A. Zighed[1], Diala Ezzeddine[2], Fabien Rico[2]

[1] abdelkader.zighed@ish-lyon.cnrs.fr
Institut des Sciences de l'Homme (ISH - USR 3385)
Université de Lyon,
14, avenue Berthelot, 69007 Lyon
[2] diala.ezzeddine@univ-lyon2.fr
fabien.rico@univ-lyon1.fr
Laboratoire Eric,
Université de Lyon,
5, avenue Pierre Mendès France, 69676 Bron Cedex, France

Abstract. Ensemble methods (EMs) have proved their efficiency in data mining, especially in supervised machine learning (ML). An EM generates a set of classifiers using one or several machine learning algorithms (MLAs) and aggregates them into a single classifier (Meta-Classifier, MC) using, for example, a majority rule vote. Instance Based (IB) MLAs , such as k-Nearest Neighbors (kNN), are very popular because of their straightforwardness. To implement these, it is simply necessary to define a dissimilarity measure based on the set of observations and fix the value of k. Thus, use of the kNN principle, as an EM algorithm is immediate. However, handling the parameter k might be difficult for some users. To simplify this problem, we can use approaches based on neighborhood graphs as alternatives. For example, relative neighborhood graphs (RNGs) or Gabriel graphs (GGs) are good candidates. Like kNN, for an unlabeled observation, the classifier assigns a label based on neighborhood graphs according to the labels in the neighborhood. For example, we can simply use the majority rule vote in the neighborhood of the unlabeled observation. While many studies using kNN in the context of EMs have been done, we found no studies that assess the interestingness of neighborhood graphs in EM approaches. In this paper, we provide comparisons with many EM approaches based either on IB learning or on other methods such as kSVM, decision tree (random forest) etc.

Keywords: Ensemble methods, neighborhood graphs, relative neighborhood Graphs, Gabriel graphs, k-Nearest Neighbors

Neighborhood Random Classification

No Author Given

No Institute Given

Abstract. Ensemble methods (EMs) have proved their efficiency in data mining, especially in supervised machine learning (ML). An EM generates a set of classifiers using one or several machine learning algorithms (MLAs) and aggregates them into a single classifier (Meta-Classifier, MC) using, for example, a majority rule vote. Instance Based (IB) MLAs , such as k-Nearest Neighbors (kNN), are very popular because of their straightforwardness. To implement these, it is simply necessary to define a dissimilarity measure based on the set of observations and fix the value of k. Thus, use of the kNN principle, as an EM algorithm is immediate. However, handling the parameter k might be difficult for some users. To simplify this problem, we can use approaches based on neighborhood graphs as alternatives. For example, relative neighborhood graphs (RNGs) or Gabriel graphs (GGs) are good candidates. Like kNN, for an unlabeled observation, the classifier assigns a label based on neighborhood graphs according to the labels in the neighborhood. For example, we can simply use the majority rule vote in the neighborhood of the unlabeled observation. While many studies using kNN in the context of EMs have been done, we found no studies that assess the interestingness of neighborhood graphs in EM approaches. In this paper, we provide comparisons with many EM approaches based either on IB learning or on other methods such as kSVM, decision tree (random forest) etc.

Keywords: Ensemble methods, neighborhood graphs, relative neighborhood Graphs, Gabriel graphs, k-Nearest Neighbors

1 Introduction

Ensemble methods (EMs) have proved their efficiency in data mining, especially in supervised machine learning (ML). An EM generates a set of classifiers using one or several machine learning algorithms (MLA) and aggregates them into a single classifier (meta-classifier, MC) using, for example, a majority rule vote. Many papers [3,18,2,14] have shown that a set of classifiers produces a better prediction than the best among them, regardless of the MLA. Theoretical and experimental results have encouraged the implementation of EM techniques in many application fields such as physics [6], face recognition [17], ecology [12], recommender systems [9] and many others too numerous to mention here. The efficiency of EMs lies in the fact that aggregating different and independent classifiers reduces the bias and the variance of the MC [8,1,5,3], which are two key concepts for effective classifiers.

Instance based (IB) MLAs such as k-Nearest Neighbors (kNN) are very popular because of their straightforwardness. Indeed, to implement these, it is simply necessary to define a dissimilarity measure on the set of observations and fix the value of k. Thus, use of the kNN principle, as an EM algorithm is immediate. However, handling the parameter k might be difficult for some users. To simplify this problem, we can use approaches based on neighborhood graphs as alternatives. For example, Relative Neighborhood Graphs (RNG) or Gabriel Graphs (GG) are "good" candidates. Like kNN, for an unlabeled observation, the classifier, based on neighborhood graphs, assigns a label according to the labels in the neighborhood. For example, we can simply use the majority rule vote in the neighborhood of the unlabeled observation. While many studies using kNN in the context of EM have been carried out, we did not find any study that assesses the interestingness of such neighborhood graphs, based more particularly on RNGs, in EM approaches. In this paper, we propose an EM approach based on neighborhood graphs. We provide comparisons with many EM approaches based on kSVM, Decision Tree (Random Forest), kNN etc. We have performed our experiments on an R platform.

This paper is organized as follows. In section 2, we introduce and recall some notations and definitions. In section 3, we introduce the EMs based on neighborhoods. Besides the classic kNN neighborhood, we will present the RNG and GG neighborhoods. Section 4 is devoted to evaluations and comparisons. Section 5 gives the main conclusions of this study.

2 Basic concepts

2.1 Notations

Let us consider a training sample E_l of n individuals $(\omega_i)_{1...n}$, described by p attributes $X^j, j = 1, \ldots, p$ and a membership class $Y \in \{y_1, \ldots, y_k, \ldots\}$. For an individual ω_i, we denote by X_i the vector $(X^j(\omega_i); j = 1, \ldots, p$ and Y_i its membership class by $Y(\omega_i)$. Below, we will refer to an individual ω or to a point $X(\omega)$ with the same meaning, indistinguishably.

For the sake of illustration, we will use the toy example of Table 1. This is a two-class data set of 17 individuals mapped into a two-dimensional space $\mathbb{R}^2$. In the figure, the class $y_1 = 1$ is indicated by a bold dot and the class $y_2 = 2$ by an empty dot.

The goal of any machine learning algorithm is to lead to a classifier capable of predicting, with high accuracy, the membership class $Y(\omega)$ for any individual ω knowing its attribute values in $X(\omega)$. Basically, the prediction is based on the knowledge we can get about the probability distribution:

$$P(Y/X) = (p(Y = y_k/X); k = 1, \ldots, K).$$

Generally, a classifier ϕ helps, somehow, for all individuals ω, to estimate $\hat{P}$ which is the membership probability vector for all classes. Thanks to the learning

E_l	X^1	X^2	Y	E_l	X^1	X^2	Y
X_1	2.13	2.33	2	X_{10}	0	2.33	2
X_2	2.13	4.11	2	X_{11}	5.64	5.17	2
X_3	2.22	1.76	2	X_{12}	7.87	2.33	1
X_4	3.37	6.88	1	X_{13}	5.64	7.5	1
X_5	6.77	0.67	1	X_{14}	4.53	8.1	1
X_6	4.53	1.16	1	X_{15}	3.37	4.31	1
X_7	3.37	0	1	X_{16}	5.64	4.11	2
X_8	1.8	6.47	2	X_{17}	7.87	4.11	2
X_9	0	5.77	2				

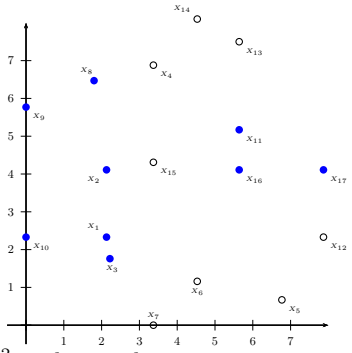

Table 1. set of points in $\mathbb{R}^2$ with two classes

sample E_l, the predicted membership class is $\hat{y_k}$ the most likely, determined as follows:

$$\hat{y_k} \text{ is such that } \hat{p}(Y = \hat{y_k}/X) = \max_{k=1,\ldots,K} \hat{p}(Y = y_k/X)$$

By thresholding at the maximum value of this vector, the membership class can be represented by a zero vector except for the most likely class by a value 1 at the corresponding rank: $\hat{P} = (0, \ldots, 0, 1, 0, \ldots, 0)$. If the classifier ϕ is considered as being quite reliable, then the prediction of the membership class for an individual ω is $\phi(X(\omega)) \in \{y_1, \ldots, y_k, \ldots\}$.

2.2 Neighborhood structure

There are many types of neighborhood that could be used to build a classifier. Among the most well known are:

- The well-known k-nearest neighbors;
- The ε-neighbors, which are defined by the subset of E_l that are in the ball of radius ε, centered on the individual, i.e. a point in Euclidean space;
- The neighborhood regions brought about by a decision tree where each leaf defines a subregion of the space. An individual that falls in a specific leaf has, as neighbors, those of the learning sample located in the same leaf.
- Parzens window neighbors;
- The neighbors in random spaces. For example, we can cite the weak models approach [7] where neighbors are obtained after a random projection along axes.
- The neighbors in the sense of a specific property. For example, the Gabriel Graphs (GG) neighbors are given by the subset of individuals of the learning sample that fulfill a certain condition. Likewise, we can define the relative neighbors (RN), the minimum spanning tree's (MST) neighbors or the Delaunay's polyhedron neighbors and so forth [13];

2.3 Neighborhood classifiers

The neighborhood classifiers depend on three components :

1. ***Neighbourhood set*** $\mathcal{P}$: the set of all subsets of E_l . This is the set of all possible neighbors to which each individual will be connected.
2. ***The neighborhood function*** $\mathcal{V}$: this defines the way in which an individual is linked to an element of the neighborhood set: $\mathcal{V} : \Re \longrightarrow \mathcal{P}$
$X \longmapsto \mathcal{V}(X)$
This function links any point X to a subset of E_l which contains its neighbors.
3. ***The decision rule*** **C**: this leads to the probability distribution of the classes
$C : \quad \Re \times \mathcal{P} \longrightarrow S_K$
$X, \mathcal{V}(X) \longmapsto y_p(X) = (p_1, p_2, \ldots, p_K)$

Hence, we can define a neighborhood classifier ϕ as based on a combination of the triplet $(\mathcal{P}, \mathcal{V}, C)$:

$$\phi(X) = y_{\mathcal{V}(X)}(X)$$

2.4 Partition by neighborhood graphs

Here we are focusing on geometrical graphs, therefore we build $\mathcal{P}$ using neighborhood graphs, such as Voronoi diagrams [13] or their dual of the Delaunay polyhedral, Gabriel graphs [10], relative neighbors graphs [16] or the minimum spanning tree [11]. In such graphs, points are linked according to a specific property. Below we give the properties that define RNGs and GGs: For a given distance measure d, a learning sample E_l and a set of individuals $x, y, z, \ldots$, any two points x and y are linked by the following rules :

– Gabriel graph (GG) :

$$y \in \mathcal{V}_{GG}(x) \Longleftrightarrow \forall z \in E_l - \{x, y\} \quad d(x, y) \leq \sqrt{d^2(x, z) + d^2(z, y)};$$

– Relative neighbours graph (RNG) :

$$y \in \mathcal{V}_{RNG}(x) \Longleftrightarrow \forall z \in E_l - \{x, y\} \quad d(x, y) \leq \max\left(d(x, z), d(y, z)\right);$$

All these geometric structures induce a related neighborhood graph with a symmetric neighborhood relationship. Figures 1 and 2 show the neighbor structures of the relative neighbors graph and the Gabriel graph, using the dataset introduced above (cf 2.1).

3 Ensemble method classifier based on neighborhood

We call this framework "Random Neighborhood Classifier (RNC)". The principle of EMs is to generate M classifiers and then aggregate them into one. To do so, M randomized iterations are performed. At iteration m, RNC:

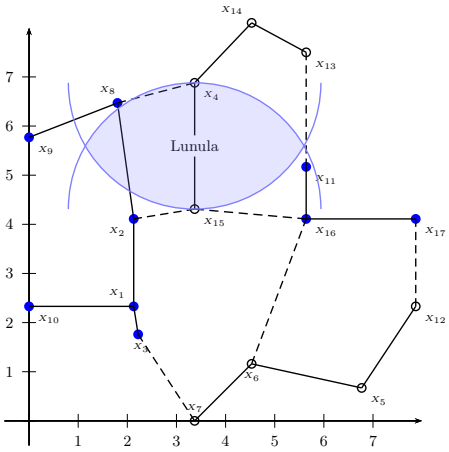

Fig. 1. Graph of relative neighbours

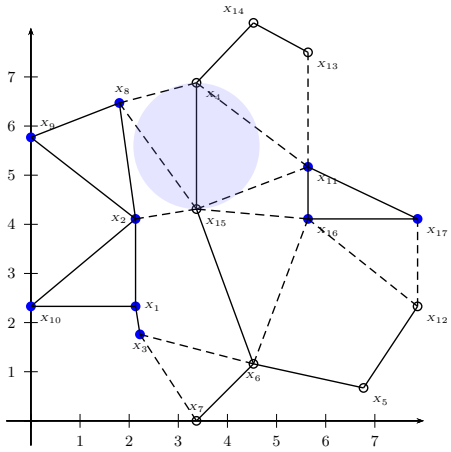

Fig. 2. Gabriel graph

1. generates a new learning set E_l^m with a given size;
2. generates a new classifier $\phi^m = (\mathcal{P}^m, \mathcal{V}^m, C^m)$;
3. uses the generated classifier to determine membership class of unclassified individual X.

Following these steps, RNC then aggregates the M predicted values related to an unclassified individual to determine its final membership class. The two key points of this procedure are the sampling procedure for generating the M classifiers and the procedure for combining the M predictions. Below, we give some details about the two key points:

1. **Sampling procedures** : From the training data set E_l which is an $n \times p$ table of values, we carry out M random samples. The sampling can be achieved in different ways:
   - Sampling on rows with or without replacement;
   - Sampling on columns;
   - Building new columns by a linear combination of some existing ones (oblique projection);
   - Generating new individuals by a linear combination of columns;
   - Adding randomly x% of rows and/or columns.

   Each sample produced leads to a specific classifier.
2. **Aggregating function** : Generally, the aggregating function is based on the majority rule vote. However, many other possibilities could be used [15]. Among these, we may cite:
   - Vote of classifiers, which aggregate the responses of each classifier and normalize them. The majority rule vote is a particular case of this one.
   - Average vector where the score for each class is the mean of the answers for all the classifiers.
   - Weighted version (majority or mean)

- Maximum Likelihood calculated as the product of the answers for all the classifiers, for each class. The winner class is the one that has the highest value.
- Naive Bayes [15].
- Decision Templates [15]. This method is based on the concept of a decision template, which is the average vector over the individual of a test sample belonging to each class, and a decision profile, which is the set of responses of all classifiers. The membership class is determined according to the Euclidean distance between the decision profile and the decision template. The winner class is the one that minimizes this distance.
- Linear regression. In this method, we assume that the probability of a class is the linear combination of the probabilities of class for each classifier.

4 Evaluation

To assess the performance of RNC, we carried out many experiments on different data sets taken from the UCI Irvine repository. For this, we made some distinctions according to the type of neighborhood used. Indeed, as our work was motivated by the absence of studies on EMs based on geometrical graphs such as RNGs, we designed two separate experiments for RNC. One was based on RNGs and the other on kNN with k = 1, 2, 3. The comparison was also extended to random forests (RFs), K support vector machines (KSVMs), Adaboost, discriminant analysis (DA), logistic regression (RegLog) and C4.5. All was done with R software.

The implementation of RF is the `randomForest` library using 500 trees. We used the R `kernlab` library to apply the KSVM algorithm with classification type C-svc. For AD and RegLog, we used `lda` and `glm` functions of the R `MASS` library and for C4.5, the `J48` function of the R `RWeka` library using the control of the Weka learner.

For RNC based on RNGs, for building the graph, we have used the MAHALANOBIS distance between individuals. For aggregating the classifiers we used the Decision Templates (DT). 100 iterations were carried that provided 100 classifiers. For kNN, we just replaced the neighborhood graph by the k-nearest neighbors one using the same distances and the same number of classifiers. The chosen aggregation method was majority voting.

We used 14 quantitative data sets. We ran the same protocol over all the methods mentioned above. For each experiment, we applied 10-Cross Validations to obtain an estimation of the error rates. For each dataset, we used the WILCOXON test [4] to evaluate the results.

The results are reported in Table 2. For each dataset, we computed the average error rate, the rank of each method among the others and the p-values detected by WILCOXON test.

As can be seen in Table 2, RNC based on RNGs performs well in comparison to kNN and also in comparison to the other methods. Indeed, from the 14 data

	Glass			Image			Ionosphere			Iris			Letter (RvsB)		
	% Err.	Rank	Wilcox	% Err.	Rank	Wilcox	% Err.	Rank	Wilcox	% Err.	Rank	Wilcox	% Err.	Rank	Wilcox
RNC	21.4	2		2.52	3		4	1		4.66	3	=	1.05	2	=
Random Forest	18.6	1	=	1.83	1	$< 5\%$	6.29	4	=	5.33	4.5	=	1.78	5	=
KSVM	31.4	7	$< 5\%$	5.57	7	$< 1\%$	5.71	3	=	5.33	4.5	=	1.38	4	=
adaBoost	-	na		-	na		7.14	6	=	-	na	=	2.03	6	$< 5\%$
AD	38.5	8	$< 5\%$	8.34	8	$< 1\%$	12.3	10	$< 1\%$	2	1	=	6.84	10	$< 1\%$
Log. Reg.	-	na		-	na		12	9	$< 1\%$	-	na	=	6.25	9	$< 1\%$
C4.5	29.5	6	$< 5\%$	3.17	5	=	8.86	7.5	$< 5\%$	5.99	6.5	=	5	8	$< 1\%$
kNN1	25.7	3	=	2.17	2	=	6.58	5	$< 5\%$	5.99	6.5	=	0.92	1	=
kNN2	27.6	5	=	3.34	6	$< 5\%$	5.43	2	$< 5\%$	8	8	=	3.48	7	$< 5\%$
kNN3	29.5	4	$< 5\%$	3.13	4	$< 5\%$	8.86	7.5	$< 1\%$	4	2	=	1.32	3	=

	Musk			Diabete (Pima)			Ringnorm			Sat			Sonar		
	% Err.	Rank	Wilcox	% Err.	Rank	Wilcox	% Err.	Rank	Wilcox	% Err.	Rank	Wilcox	% Err.	Rank	Wilcox
RNC	3.02	3		24.9	6	=	2.26	2		10.8	4		12.5	2	
Random Forest	2	1.5	$< 1\%$	23.	2	=	5.24	4	$< 1\%$	7.79	1	$< 1\%$	14.5	4.5	=
Ksvm	3.46	5	=	24.1	5	=	1.56	1	=	9.49	2	$< 5\%$	17	7	=
adaBoostv	2	1.5	$< 1\%$	23.8	4	=	3.4	3	$< 1\%$	-	na		16	6	=
AD	5.58	10	$< 1\%$	22.2	1	=	38.1	10	$< 1\%$	16.	8	$< 1\%$	26	8	$< 5\%$
Log. Reg.	4.84	8	$< 1\%$	23.7	3	=	34.8	9	$< 1\%$	-	na		28	9	$< 5\%$
C4.5	3.13	4	=	25.7	9	=	13.9	7	$< 1\%$	13.7	7	$< 1\%$	31	10	$< 1\%$
kNN1	3.61	6	$< 5\%$	25	7	=	6.37	5	$< 1\%$	10.6	3	=	11	1	=
kNN2	5.24	9	$< 1\%$	30.1	10	=	24.2	8	$< 1\%$	12.3	6	$< 5\%$	14.5	4.5	=
kNN3	4.49	7	$< 1\%$	25.4	8	=	8.82	6	$< 1\%$	11.9	5	$< 5\%$	12.5	3	=

	Threenorm			Twonorm			Waveform			Wisc. Breast Cancer		
	% Err.	Rank	Wilcox	% Err.	Rank	Wilcox	% Err.	Rank	Wilcox	% Err.	Rank	Wilcox
RNC	13.3	2	=	2.64	3.5		14.8	4		2.65	2.5	
Random Forest	13.6	3	=	3.4	7	$< 5\%$	14.4	3	=	2.5	1	=
Ksvm	12.4	1	=	2.69	5	=	13.3	1	$< 5\%$	4.12	9	$< 5\%$
adaBoostv	15.1	6	=	4.48	8	$< 1\%$	-	na		3.09	5	=
AD	16.6	7.5	$< 1\%$	2.56	2	=	13.8	2	$< 5\%$	3.82	8	=
Log. Reg.	16.6	7.5	$< 1\%$	2.83	6	=	-	na		3.24	6.5	=
C4.5	28.5	10	$< 1\%$	16.7	10	$< 1\%$	24.1	7	$< 1\%$	4.41	10	$< 5\%$
kNN1	14.6	5	=	2.64	3.5	=	16.9	6	$< 1\%$	3.08	4	=
kNN2	25.7	9	$< 1\%$	10	9	$< 5\%$	29.6	8	$< 1\%$	2.65	2.5	=
kNN3	14.4	4	=	2.4	1	=	15.9	5	$< 1\%$	3.24	6.5	=

Table 2. Comparison of RNC with several classification algorithms

sets, RNC is first once, second 6 times and third 4 times. Thus, mostly RNC is one of the 3 first methods.

We also computed the mean rank. This is done 2 times, using or not the classifiers adaBoost and Logistic Regression (which could not give an answer for more than 2 classes). The result is shown in Table 3 where RNC comes out first. To see if the difference is significant, we applied the simple Friedman test [4], which showed a difference with a p-value of 6.849×10^{-6} and the post-hoc test (comparing each classifier by pair) gave the result shown in Table 4.

	RNC	Random Forest	Ksvm	adaBoost	AD	Log. Reg.	C4.5	KNN1	KNN2	KNN3
All methods	2.88	3.19	4,04	5,06	6,58	7,56	7,46	4,15	7,04	4,58
Without adaBoost & LogReg	2.64	2.86	3.96		5.79		6.64	3.86	6.00	4.25

Table 3. Mean rank of the methods

	RNC	Random Forest	Ksvm	AD	C4.5	KNN1	KNN2	KNN3
RNC		1	0.84	0.015	0.0004	0.89	0.0067	0.66
Random Forest			0.93	0.032	0.0011	0.96	0.015	0.80
Ksvm				0.50	0.072	1	0.35	1
AD					0.98	0.42	1	0.71
C4.5						0.052	1	0.16
KNN1							0.28	1
KNN2								0.55
KNN3								

Table 4. p-value for the difference

These results are very encouraging, because we think that they could be improved by varying some parameters such as:

- The choice of neighborhood structure, especially since we know that the neighborhood graphs are particularly sensitive to the dimension of the representation space.
- The type of base classifier. Should we use the closest connected homogeneous component? How to define precisely this notion of proximity ? Should we take into consideration the size of the database or other characteristics of the neighborhood such as density, etc...?
- The selection methods to improve the quality of the data sets or the classifiers.

All these issues are under study and should produce significant improvements for RNCs based on geometrical graphs.

5 Conclusion and further work

Here we have provided a new approach for using neighborhood structures in ensemble methods. The results obtained show that they are challenging the most powerful techniques such as random forests and kSVM. Methods based on geometrical neighborhood graphs outperform the classic methods such as kNN. Many possibilities are underway in order to improve the RNC based on RNG. A library containing all the functionalities that have been achieved is available by emailing the authors.

References

1. Breiman, L.: Bias, variance, and arcing classifiers. Statistics (1996)
2. Breiman, L.: Random forests. Machine learning 45(1), 5–32 (2001)
3. Brown, G., Wyatt, J., Harris, R., Yao, X.: Diversity creation methods: a survey and categorisation. Information Fusion 6(1), 5–20 (2005)
4. Demsar, J.: Statistical comparisons of classifiers over multiple data sets (2006)
5. Domingos, P.: A unified bias-variance decomposition and its applications. In: ICML. pp. 231–238. Citeseer (2000)
6. Ham, J., Chen, Y., Crawford, M., Ghosh, J.: Investigation of the random forest framework for classification of hyperspectral data. IEEE Transactions on Geoscience and Remote Sensing 43(3) (2005)
7. Ho, T., Kleinberg, E.: Building projectable classifiers of arbitrary complexity. In: International Conference on Pattern Recognition. vol. 13, pp. 880–885 (1996)
8. Kohavi, R., Wolpert, D.: Bias plus variance decomposition for zero-one loss functions. In: Machine Learning-International Workshop. pp. 275–283. Citeseer (1996)
9. O'Mahony, M.P., Cunningham, P., Smyth, B.: An assessment of machine learning techniques for review recommendation. In: Proceedings of the 20th Irish conference on Artificial intelligence and cognitive science. pp. 241–250. AICS'09, Springer-Verlag, Berlin, Heidelberg (2010), `http://portal.acm.org/citation.cfm?id=1939047.1939075`
10. Park, J., Shin, H., Choi, B.: Elliptic Gabriel graph for finding neighbors in a point set and its application to normal vector estimation. Computer-Aided Design 38(6), 619–626 (2006)
11. Planeta, D.S.: Linear time algorithms based on multilevel prefix tree for finding shortest path with positive weights and minimum spanning tree in a networks. CoRR
12. Prasad, A., Iverson, L., Liaw, A.: Newer classification and regression tree techniques: bagging and random forests for ecological prediction. Ecosystems 9(2), 181–199 (2006)
13. Preparata, F., Shamos, M.: Computational geometry: an introduction. Springer (1985)
14. Schapire, R.: The boosting approach to machine learning: An overview. Lecture Notes In Statistics-Springer Verlag pp. 149–172 (2003)
15. Shipp, C., Kuncheva, L.: Relationships between combination methods and measures of diversity in combining classifiers. Information Fusion 3(2), 135–148 (2002)
16. Toussaint, G.: The relative neighbourhood graph of a finite planar set. Pattern recognition 12(4), 261–268 (1980)

17. Wang, X., Tang, X.: Random sampling lda for face recognition pp. 259–267 (2004), http://portal.acm.org/citation.cfm?id=1896300.1896337
18. Zhou, Z., Wu, J., Tang, W.: Ensembling neural networks: Many could be better than all* 1. Artificial intelligence 137(1-2), 239–263 (2002)