Unsupervised Ensembles for Outlier Detection

Guilherme O. Campos
Supervised by Prof. Dr. Wagner Meira Jr.
Department of Computer Science
Federal University of Minas Gerais
Belo Horizonte, Brazil
gocampos@dcc.ufmg.br

ABSTRACT

Ensemble techniques have been applied to the unsupervised outlier detection problem in some scenarios. Challenges are the generation of diverse ensemble members and the combination of individual results into an ensemble. For the latter challenge, some methods tried to design smaller ensembles out of a wealth of possible ensemble members, to improve the diversity and accuracy of the ensemble (relating to the ensemble selection problem in classification). In this paper, We propose a boosting strategy to solve the ensemble selection problem, called BoostSelect. We evaluate BoostSelect over a large benchmark of datasets for outlier detection, showing improvements over baseline approaches.

1. INTRODUCTION

The identification of outliers (i.e., data objects that do not fit well to the general data distribution) is very important in many practical applications. Application examples are the detection of credit card fraud in financial transactions data, the identification of measurement errors in scientific data, or the analysis of sports statistics data.

Recent research on the unsupervised problem of outlier detection advanced the area by applying ensemble techniques [31]. Ensemble methods, i.e., combining the findings or results of individual learners to an integrated, typically more reliable and better result, are well established in the supervised context of classification or regression [20]. In unsupervised learning, the theoretical underpinnings are less clear but can be drawn in analogy to the supervised context as it has been done for clustering ensembles [30].

The focus of my PhD thesis is on ensemble selection, which has been well studied in supervised scenarios [4] (also called selective ensembles [29], or ensemble pruning [15, 27, 30]). Ensemble selection is also related to boosting [22], which is often used to change training conditions for additionally sought, yet to be trained ensemble members or to select the most suitable additional ensemble members from a pool of solutions.

Proceedings of the VLDB 2018 PhD Workshop, August 27, 2018. Rio de Janeiro, Brazil.Copyright (C) 2018 for this paper by its authors. Copying permitted for private and academic purposes..

Given many outlier detection results, how do we select which members we are going to include into the ensemble? How do we combine these selected members into a single and more robust result? These challenges mixed to an unsupervised environment are still pertinent and more research is need to increase our knowledge over this topic in question.

In this paper we propose an initial approach towards a more robust ensemble method by transferring the supervised boosting technique to the unsupervised scenario of outlier detection ensembles. The proposed outlier ensemble selection technique is called BoostSelect.

2. RELATED WORK

The ensemble approach to learning has been studied in outlier detection several times. In analogy to supervised learning, an ensemble can be expected to improve over its components if these components deliver results with a reasonable accuracy while being diverse [31]. The two main challenges for creating good ensembles are, therefore, (i) the generation of diverse (potential) ensemble members, and (ii) the combination (or selection) of members to an ensemble.

Some strategies to achieve diversity among ensemble members have been explored, such as feature bagging (i.e., combining outlier scores learned on different subsets of attributes) [13], different parameter choices for some base method [5], the combination of actually different base methods [16, 10, 23], the introduction of a random component in a given learner [14], the use of different subsamples of the data objects (parallel [33] or sequential [21, 19]), adding some random noise component on the data ("perturbation") [32], or using approximate neighborhoods for density estimates [8]. Likewise, different combination procedures have been proposed based on outlier scores or on outlier ranks [13, 5, 10, 31].

Some methods have also been proposed to select the more diverse or the more accurate ensemble members [23, 18]. These unsupervised methods construct a target result vector from unfiltered results and then sequentially select individual results that somehow fit to the target vector while being different from already selected solutions. The "Greedy ensemble" approach [23] fails in generating a good target vector by selecting a percentage of top instances for each ensemble candidate to compose the vector. This procedure normally selects many inliers to fit the target vector due to the imbalance between few outliers and many inliers. To solve this problem, "SelectV" approach [18] generate the target vector by taking the average over all outlier detection results (possible ensemble members). The main problem with

"SelectV" is that the algorithm does not consider diversity as an important factor in selecting ensemble members.

3. BOOSTING FOR ENSEMBLE SELECTION

Starting from the ideas discussed for the "Greedy ensemble" [23] and for "SelectV" [18], we propose here an improved outlier ensemble selection method that is amenable to the application of boosting techniques. Boosting is well studied in supervised contexts [22]. We design and apply an equivalent technique in the unsupervised setting, to select good components for an ensemble of outlier detectors, resulting in our method BoostSelect.

3.1 Construction of the Target Vector

As a prerequisite for the combination of different outlier score lists (i.e., individual results, potential ensemble members), we normalize the scores following established procedures [10]. The target vector is constructed by combining the scores of all available results. Different combination methods could be used here, without further assumptions taking the average score is the most natural approach [31], i.e., the target vector lists the average scores of all individual results for each data object. From this target vector, we preliminarily assume the top $\lfloor n \cdot t \rfloor$ objects (ranked by their combined score) to be outliers, where n is the dataset size and $0 < t \ll 1$ is a parameter capturing the expected percentage of outliers in the dataset (i.e., there are $K = \lfloor n \cdot t \rfloor$ outliers assumed to be present). The target vector thus becomes a binary vector, listing 1 for an (alleged) outlier and 0 for an (alleged) inlier and serves as pseudo ground truth for the boosting approach to ensemble selection.

3.2 Weights and Ensemble Diversity

Weighted Pearson correlation has been proposed as a similarity measure for outlier rankings [23]. We follow the procedure of Schubert et al. [23], setting weights for Pearson correlation to outliers and inliers. Different from previous approaches, though, these values are only the initial weights. The weights will be updated by the boosting procedure.

The potential ensemble members are sorted according to their weighted Pearson correlation to the target vector. The candidate that is most similar to the target vector is chosen as the first ensemble member.

Remaining potential ensemble members are iteratively re-sorted in ascending order according to their similarity to the current prediction of the ensemble, resulting in a preference for the most different (i.e., most complementary) additional ensemble members. Potential members are included if their inclusion would increase the similarity of the ensemble prediction to the target vector, otherwise they are discarded. If the correlation improves, the ensemble is updated and the remaining lists are re-sorted by their weighted Pearson correlation to the updated prediction.

3.3 Boosting Procedure

The boosting is performed upon the inclusion of a new member into the ensemble. The idea is to reduce the weights for those outliers that have already been identified by any ensemble member. The weights are reduced by some specified parameter $0 < d < 1$ (drop rate).

The boosting effect is that the selection will prefer to include such additional ensemble members that detect those outliers that have not yet been detected by any ensemble member, while very easy outliers that have been detected by many ensemble members already will get assigned smaller and smaller weights.

Algorithm 1 BoostSelect

Input: P := set of normalized outlier score lists, d := drop rate (percentage), t := threshold (percentage), $combination$:= combination technique
Output: E := ensemble members

```
 1: $W := [n], E := \emptyset$
 2: $target := combination(P)$    ▷ Generating the target vector
 3: $target := convertBinary(target, t)$    ▷ Top $K = \lfloor n \cdot t \rfloor$ scores ← 1, others ← 0
 4: $W := \left[out = \frac{1}{2K}, in = \frac{1}{2(n-K)}\right]$    ▷ $K$ = number of outliers, $n$ = size
 5: Sort $P$ by weighted Pearson Correlation ($wPC$) to $target$    ▷ Descending order
 6: $f := getFirst(P)$    ▷ Remove $f$ from $P$
 7: $E := E \cup f$
 8: while $P \neq \emptyset$ do
 9:     $curr := combination(E)$    ▷ Current prediction
10:     sort $P$ by $wPC$ to $curr$    ▷ Ascending order
11:     $f := getFirst(P)$    ▷ Remove $f$ from $P$
12:     if $wPC(combination(E \cup f), target) > wPC(curr, target)$ then
13:         $E := E \cup f$    ▷ Include into ensemble
14:         $Boosting(W, target, f, t, d)$    ▷ Adapt the weights
15:     end if
16: end while
```

Algorithm 2 Boosting

Input: W := weight vector, $target$:= target vector, f := new ensemble member, t := threshold (percentage), d := drop rate (percentage)
Output: W := Updated weights

```
1: $outliers := convertBinary(f, t)$
2: for $i \in 1 : size(target)$ do
3:     if $target(i) == 1$ & $outliers(i) == 1$ then
4:         $W(i) := W(i) * d$
5:     end if
6: end for
```

Algorithm 1 lists the steps of the overall framework BoostSelect in pseudo code. The boosting procedure is detailed in Algorithm 2.

4. EXPERIMENTS

4.1 Datasets

For evaluation, we use a benchmark data repository for outlier detection [3]. The repository is based on 23 basic datasets, processed in different ways mainly to provide variants with different percentage of outliers and with different handling of dataset characteristics such as duplicates, attribute normalization, and categorical values. As suggested for analysis [3], we focus on the normalized datasets without duplicates, which leaves us with 422 dataset variants.

4.2 Ensemble Members

As basic outlier detection results (i.e., potential ensemble members) we use the results provided along with the datasets [3], testing 12 neighborhood-based outlier detection algorithms changing the neighborhood size k from 1 to 100. The algorithms are: KNN [17], KNNW [1], LOF [2], SimplifiedLOF [25], LoOP [9], LDOF [28], ODIN [6], FastABOD [11], KDEOS [24], LDF [12], INFLO [7], and COF [26]. For LDOF and KDEOS, k must be larger than 1, for FastABOD, k must be larger than 2, resulting in 1196 results per dataset (less on some small datasets where k cannot reach 100). These results compose the set of potential ensemble members.

The outlier scores of these results are processed (following Kriegel et al. [10]) by applying an inverse logarithmic scaling on FastABOD results and an inverse linear scaling on ODIN results, since FastABOD and ODIN give inverse score results (i.e., the lower the scores, the higher is the chance of an observation to be an outlier). Then a simple linear scaling from 0 to 1 is applied to transform all scores into the same range.

4.3 Competitors and Settings

We compare BoostSelect against the Greedy [23] and SelectV [18] ensembles. We also generate a "Naïve" ensemble and Random ensembles as baselines. The "Naïve" ensemble is a combination of all individual outlier results (i.e., a full ensemble without selection procedure).

For each instance of an ensemble selection strategy (Greedy, SelectV, and BoostSelect, respectively, on each dataset), we generate 1000 "Random" ensembles consisting of the same number of members as the corresponding selective ensemble, where the ensemble members are randomly selected.

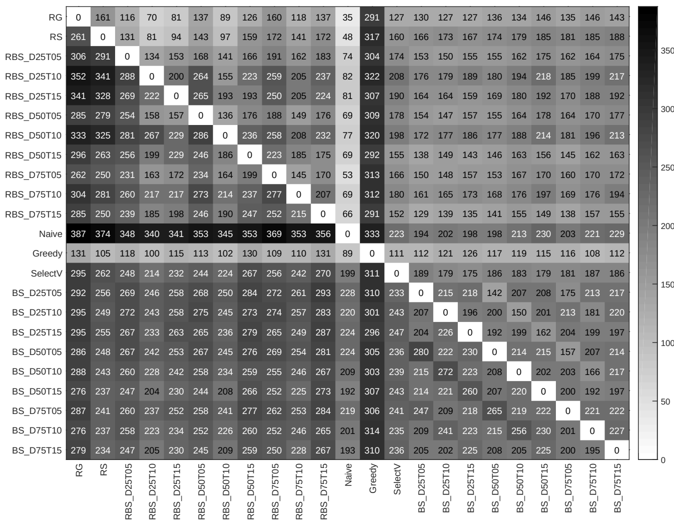
We used the Greedy ensemble rate parameter as 0.01, as suggested by the authors of the Greedy ensemble [23]. We test a range of parameters for BoostSelect: $d = [0.25, 0.5, 0.75]$ and $t = [0.05, 0.1, 0.15]$.

As combination technique for ensembles we use the average score.

4.4 Results

Figure 1 shows pairwise comparisons between all ensembles over all datasets, considering the ROC AUC evaluation measure (area under the curve of the receiver operating characteristic). We compare the ensemble selection techniques "Naïve", "Greedy", "SelectV", and "BS" (BoostSelect). We include random ensembles for each ensemble selection strategy and for each parametrization of BoostSelect: RG (Random Greedy), RS (Random SelectV), RBS (Random BoostSelect). The numbers represent on how many datasets the ensemble listed in the row has performed better than the ensemble listed in the column. Numbers representing the majority (more than 50%) of the datasets are white, smaller numbers black. The larger the number, the darker is its background. For the random ensemble, we take the average performance over the 1000 instances.

The best overall method is BoostSelect with $d = 0.75$ and $t = 0.05$, which has only more losses than wins when competing against BoostSelect with $d = 0.25$ and $t = 0.1$. The Greedy ensemble does not perform well in general, having more losses than wins against every other competitor. SelectV is better than all random variants and Greedy, but worse than Naïve and worse than all BoostSelect results. The Naïve ensemble behaves very consistently, as it beats by



	RG	RS	RBS_D25T05	RBS_D25T10	RBS_D25T15	RBS_D50T05	RBS_D50T10	RBS_D50T15	RBS_D75T05	RBS_D75T10	RBS_D75T15	Naive	Greedy	SelectV	BS_D25T05	BS_D25T10	BS_D25T15	BS_D50T05	BS_D50T10	BS_D50T15	BS_D75T05	BS_D75T10	BS_D75T15
RG	0	161	116	70	81	137	89	126	160	118	137	35	291	127	130	127	127	136	134	146	135	146	143
RS	261	0	131	81	94	143	97	159	172	141	172	48	317	160	166	173	167	174	179	185	181	185	188
RBS_D25T05	306	291	0	134	153	168	141	166	191	162	183	74	304	174	153	150	155	155	162	175	162	164	175
RBS_D25T10	352	341	288	0	200	264	155	223	259	205	237	82	322	208	176	179	189	180	194	218	185	199	217
RBS_D25T15	341	328	269	222	0	265	193	193	250	205	224	81	307	190	164	164	159	169	180	192	170	188	192
RBS_D50T05	285	279	254	158	157	0	136	176	188	149	176	69	309	178	154	147	157	155	164	178	164	170	177
RBS_D50T10	333	325	281	267	229	286	0	236	258	208	232	77	320	198	172	177	186	177	188	214	181	196	213
RBS_D50T15	296	263	256	199	229	246	186	0	223	185	175	69	292	155	138	149	143	146	163	156	145	162	163
RBS_D75T05	262	250	231	163	172	234	164	199	0	145	170	53	313	166	150	148	157	153	167	170	160	170	172
RBS_D75T10	304	281	260	217	217	273	214	237	277	0	207	69	312	180	161	165	173	168	176	197	169	176	194
RBS_D75T15	285	250	239	185	198	246	190	247	252	215	0	66	291	152	129	139	135	141	155	149	138	157	155
Naive	387	374	348	340	341	353	345	353	369	353	356	0	333	223	194	202	198	198	213	230	203	221	229
Greedy	131	105	118	100	115	113	102	130	109	110	131	89	0	111	112	121	126	117	119	115	116	108	112
SelectV	295	262	248	214	232	244	224	267	256	242	270	199	311	0	189	179	175	186	183	179	181	187	186
BS_D25T05	292	256	269	246	258	268	250	284	272	261	293	228	310	233	0	215	218	142	207	208	175	213	217
BS_D25T10	295	249	272	243	258	275	245	273	274	257	283	220	301	243	207	0	196	200	150	201	213	181	220
BS_D25T15	295	255	267	233	263	265	236	279	265	249	287	224	296	247	204	226	0	192	199	162	204	199	197
BS_D50T05	286	248	267	242	253	267	245	276	269	254	281	224	305	236	280	222	230	0	214	215	157	207	214
BS_D50T10	288	243	260	228	242	258	234	259	255	246	267	209	303	239	215	272	223	208	0	202	203	166	217
BS_D50T15	276	237	247	204	230	244	208	266	252	225	273	192	307	243	214	221	260	207	220	0	200	192	197
BS_D75T05	287	241	260	237	252	258	241	277	262	253	284	219	306	241	247	209	218	265	219	222	0	221	222
BS_D75T10	276	237	258	223	234	252	226	260	252	246	265	201	314	235	209	241	223	215	256	230	201	0	227
BS_D75T15	279	234	247	205	230	245	209	259	250	228	267	193	310	236	205	202	225	208	205	225	200	195	0

Figure 1: Summarization of pairwise comparisons over all 422 datasets. The number counts the wins in term of ROC AUC (average ROC AUC in case of random ensembles) of the ensemble listed in the row against the ensemble listed in the column.

a large margin all random ensemble approaches, but still has more losses when compared to BoostSelect. Even though neither the threshold t nor the drop rate d has a strong impact on wins, setting a relatively large drop rate and a relatively small threshold overall seems to be a good choice of parameters for BoostSelect, although the optimal parameter choice differs from dataset to dataset.

Looking at the top left quadrant of the heat map (Figure 1), where the random ensembles compete against themselves, we also see a broad dominance by the random ensembles based on BoostSelect. This suggests that the number of ensemble members selected by BoostSelect is a better choice than those selected by the other strategies.

5. CONCLUSION AND FUTURE DIRECTIONS

We proposed a new ensemble selection strategy for unsupervised outlier detection ensembles, using the unsupervised equivalent to a boosting strategy for ensemble selection. Experiments show the favorable behavior of the new ensemble selection strategy compared to existing methods (Greedy and SelectV) on a large set of benchmark datasets. Main differences between our method BoostSelect, the Greedy ensemble, and SelectV can be attributed to a different way of focusing on diversity and accuracy of ensemble members. Greedy goes all out for diversity and mostly disregards accuracy, while SelectV ignores diversity and maximizes accuracy of the ensemble members. Our new method BoostSelect considers both, diversity and accuracy, in a balanced manner and performs competitively on average over a large selection of benchmark datasets with strong improvements on many of the benchmark datasets.

The behavior of BoostSelect is robust to the parameters on many datasets but depends strongly on the choice of parameters on some datasets. As future work, we are specially interested on this behavior and potential relation to properties of the datasets. We are also interested to improve the target vector generation step and to include a complete study over the combination step of outlier detection ensemble.

Acknowledgments This work was partially supported by CAPES - Brazil, Fapemig, CNPq, and by projects InWeb, MASWeb, EUBra-BIGSEA, INCT-Cyber, and Atmosphere.

6. REFERENCES

[1] F. Angiulli and C. Pizzuti. Fast outlier detection in high dimensional spaces. In *Proc. PKDD*, pages 15–26, 2002.

[2] M. M. Breunig, H.-P. Kriegel, R. Ng, and J. Sander. LOF: Identifying density-based local outliers. In *Proc. SIGMOD*, pages 93–104, 2000.

[3] G. O. Campos, A. Zimek, J. Sander, R. J. G. B. Campello, B. Micenkov, E. Schubert, I. Assent, and M. E. Houle. On the evaluation of unsupervised outlier detection: Measures, datasets, and an empirical study. *Data Min. Knowl. Disc.*, 30:891–927, 2016.

[4] R. Caruana, A. Niculescu-Mizil, G. Crew, and A. Ksikes. Ensemble selection from libraries of models. In *Proc. ICML*, 2004.

[5] J. Gao and P.-N. Tan. Converting output scores from outlier detection algorithms into probability estimates. In *Proc. ICDM*, pages 212–221, 2006.

[6] V. Hautamki, I. Krkkinen, and P. Frnti. Outlier detection using k-nearest neighbor graph. In *Proc. ICPR*, pages 430–433, 2004.

[7] W. Jin, A. K. H. Tung, J. Han, and W. Wang. Ranking outliers using symmetric neighborhood relationship. In *Proc. PAKDD*, pages 577–593, 2006.

[8] E. Kirner, E. Schubert, and A. Zimek. Good and bad neighborhood approximations for outlier detection ensembles. In *Proc. SISAP*, pages 173–187, 2017.

[9] H.-P. Kriegel, P. Krger, E. Schubert, and A. Zimek. LoOP: local outlier probabilities. In *Proc. CIKM*, pages 1649–1652, 2009.

[10] H.-P. Kriegel, P. Krger, E. Schubert, and A. Zimek. Interpreting and unifying outlier scores. In *Proc. SDM*, pages 13–24, 2011.

[11] H.-P. Kriegel, M. Schubert, and A. Zimek. Angle-based outlier detection in high-dimensional data. In *Proc. KDD*, pages 444–452, 2008.

[12] L. J. Latecki, A. Lazarevic, and D. Pokrajac. Outlier detection with kernel density functions. In *Proc. MLDM*, pages 61–75, 2007.

[13] A. Lazarevic and V. Kumar. Feature bagging for outlier detection. In *Proc. KDD*, pages 157–166, 2005.

[14] F. T. Liu, K. M. Ting, and Z.-H. Zhou. Isolation-based anomaly detection. *ACM TKDD*, 6(1):3:1–39, 2012.

[15] D. D. Margineantu and T. G. Dietterich. Pruning adaptive boosting. In *Proc. ICML*, pages 211–218, 1997.

[16] H. V. Nguyen, H. H. Ang, and V. Gopalkrishnan. Mining outliers with ensemble of heterogeneous detectors on random subspaces. In *Proc. DASFAA*, pages 368–383, 2010.

[17] S. Ramaswamy, R. Rastogi, and K. Shim. Efficient algorithms for mining outliers from large data sets. In *Proc. SIGMOD*, pages 427–438, 2000.

[18] S. Rayana and L. Akoglu. Less is more: Building selective anomaly ensembles. *ACM TKDD*, 10(4):42:1–42:33, 2016.

[19] S. Rayana, W. Zhong, and L. Akoglu. Sequential ensemble learning for outlier detection: A bias-variance perspective. In *Proc. ICDM*, pages 1167–1172, 2016.

[20] L. Rokach. Ensemble-based classifiers. *Artif. Intell. Rev.*, 33:1–39, 2010.

[21] M. Salehi, X. Zhang, J. C. Bezdek, and C. Leckie. Smart sampling: A novel unsupervised boosting approach for outlier detection. In *AI 2016: Advances in Artificial Intelligence - 29th Australasian Joint Conference, Hobart, TAS, Australia, December 5-8, 2016, Proceedings*, pages 469–481, 2016.

[22] R. E. Schapire and Y. Freund. *Boosting. Foundations and Algorithms.* MIT Press, Cambridge, MA, 2012.

[23] E. Schubert, R. Wojdanowski, A. Zimek, and H.-P. Kriegel. On evaluation of outlier rankings and outlier scores. In *Proc. SDM*, pages 1047–1058, 2012.

[24] E. Schubert, A. Zimek, and H.-P. Kriegel. Generalized outlier detection with flexible kernel density estimates. In *Proc. SDM*, pages 542–550, 2014.

[25] E. Schubert, A. Zimek, and H.-P. Kriegel. Local outlier detection reconsidered: a generalized view on locality with applications to spatial, video, and network outlier detection. *Data Min. Knowl. Disc.*, 28(1):190–237, 2014.

[26] J. Tang, Z. Chen, A. W.-C. Fu, and D. W. Cheung. Enhancing effectiveness of outlier detections for low density patterns. In *Proc. PAKDD*, pages 535–548, 2002.

[27] G. Tsoumakas, I. Partalas, and I. Vlahavas. An ensemble pruning primer. In O. Okun and G. Valentini, editors, *Applications of Supervised and Unsupervised Ensemble Methods*, pages 1–13. Springer, 2009.

[28] K. Zhang, M. Hutter, and H. Jin. A new local distance-based outlier detection approach for scattered real-world data. In *Proc. PAKDD*, pages 813–822, 2009.

[29] Z. Zhou, J. Wu, and W. Tang. Ensembling neural networks: Many could be better than all. *Artif. Intell.*, 137(1-2):239–263, 2002.

[30] Z.-H. Zhou. *Ensemble Methods. Foundations and Algorithms.* CRC Press, 2012.

[31] A. Zimek, R. J. G. B. Campello, and J. Sander. Ensembles for unsupervised outlier detection: Challenges and research questions. *SIGKDD Explor.*, 15(1):11–22, 2013.

[32] A. Zimek, R. J. G. B. Campello, and J. Sander. Data perturbation for outlier detection ensembles. In *Proc. SSDBM*, pages 13:1–12, 2014.

[33] A. Zimek, M. Gaudet, R. J. G. B. Campello, and J. Sander. Subsampling for efficient and effective unsupervised outlier detection ensembles. In *Proc. KDD*, pages 428–436, 2013.