Journal of Machine Learning Research 1 (2020) 1-10 Submitted 04/24; Published 04/24

Comparing Lowest Probability Mass Neighbors with Condensed and Edited Nearest Neighbors for Overcoming the Density Problem

Joshua Ryan Steenson JOSHUA.RYAN.STEENSON@GMAIL.COM
Gianforte School of Computing
Montana State University
Bozeman, MT, 59715, USA

Logan Caraway LACARAWAY@OUTLOOK.COM
Gianforte School of Computing
Montana State University
Bozeman, MT, 59715, USA

Editor: None

Abstract

Ting et al. (2018) stated that nearest neighbor algorithms tend to have difficulties in classification where the classes have different densities and implemented a mass-based dissimilarity in order to attempt to overcome this density problem. This mass-based dissimilarity was utilized in place of the distance metric to create the Lowest Probability Mass Neighbors (LMN) algorithm. However, the algorithm was not without its own shortcomings. Therefore, an experiment was run to determine if existing algorithms could overcome the density issue without these same shortcomings. The experiment compared k Nearest Neighbors (k-NN), LMN, Condensed Nearest Neighbors (C-KNN), Condensed Lowest Probability Mass Neighbors (C-LMN), Edited Nearest Neighbors (E-KNN), and Edited Lowest Probability Mass Neighbors (E-LMN) to determine whether C-KNN and E-KNN could overcome the difference in densities problem in lieu of LMN. We tested the hypothesis that LMN does not tend to have a significantly different accuracy than reduced nearest neighbor methods (C-KNN and E-KNN) on data sets where LMN does have a significantly higher accuracy than k-NN. The experiment was run on four datasets utilized by Ting et al. (2018) in which Ting et al. (2018) used default parameters and $k = 5$ with LMN outperforming k-NN in three of the four datasets. It was found that, after tuning the algorithms, LMN only had significantly better performance that k-NN on one of the data sets (one-third of the datasets expected). This raises the question of how well LMN itself handles the density problem. In the data set where LMN significantly outperformed k-NN, it also outperformed C-KNN and E-KNN. C-KNN and E-KNN also tended to perform worse than standard k-NN, likely because removing enough points to even out the densities tended to cause over-fitting of the decision boundary.

1. Introduction

For many machine learning tasks, such as classification, clustering, and density estimation, a nearest neighbor algorithm (Kularni and Harman (2011)) is commonly utilized. Nearest neighbor algorithms have widespread applications, but, due to the utilization of a distance metric, they also have various shortcomings. Distance measures do not consider two points in a dense region to have a lower similarity than two points of the same interpoint distance in a less dense region, which is a key property of dissimilarity for humans (Krumhansl (1978); Tversky (1977)). This failure to capture the local density in the similarity measure causes an increased misclassification rate in many situations where the classes have different densities (Ting et al. (2018)). Ting et al. (2018) proposed the utilization of a mass-based dissimilarity to replace the distance metric in nearest neighbor algorithms, claiming that the resulting Lowest Probability Mass Neighbors (LMN) algorithms overcome key shortcomings of nearest neighbor algorithms in classification and clustering tasks. Through simulations and empirical evaluation, Ting et al. (2018) demonstrated that LMN algorithms overcame some density-based shortcomings of nearest neighbor algorithms. Unfortunately, for supervised tasks, LMN has a runtime that renders it impractical for most real time applications (Ting et al. (2018)). Furthermore, in applications where classes have wildly different densities, one must consider whether the LMN algorithm is an improvement over existing algorithms. Ting et al. (2018) demonstrated that LMN overcame the density issue compared to standard k Nearest Neighbors (k-NN). Condensed Nearest Neighbors (C-KNN) (Alpaydin (2014)) and Edited Nearest Neighbors (E-KNN) are nearest neighbor algorithms that reduce the data set while maintaining the decision boundary. Since C-KNN and E-KNN reduce the data set to only the points required to maintain the decision boundary, one must consider whether these algorithms could be utilized in place of LMN for overcoming the difference in densities problem. We ran an experiment comparing k-NN, LMN, C-KNN, Condensed Lowest Probability Mass Neighbors (C-LMN), E-KNN, and Edited Lowest Probability Mass Neighbors (E-LMN) for classification to test the hypothesis that LMN does not tend to have a significantly different accuracy than reduced nearest neighbor methods on data sets where LMN does have a significantly higher accuracy than k-NN.

2. Related Literature

The utilization of a mass-based dissimilarity as studied in Ting et al. (2018) is part of the wider field of mass estimation. LMN is actually not the first algorithm that replaces the distance metric with a mass-based similarity for nearest neighbor algorithms; A Shared Nearest Neighbors (SNN) similarity measure has successfully been utilized in DBSCAN to find clusters with varying densities (Jarvis and Patrick (1973)). SNN similarity has been demonstrated to be a mass-based similarity (Ting et al. (2018)). It has been demonstrated that another well-known algorithm, iForest (Liu et al. (2008)), utilizes mass estimation, where the anomaly score of a query point through each iTree is a proxy for mass (Ting et al. (2013)). Mass estimation has been utilized for numerous applications including, but not limited to: anomaly detection (Liu et al. 2008), classification (Ting et al. (2018)), and clustering (Ting et al. (2018); Jarvis and Patrick (1973)).

3. Problem Statement and Hypothesis

It has been demonstrated that k-NN has reduced accuracy when the classes have different densities (Ting et al. (2018)). Ting et al. (2018) proposed a mass-based similarity measure to replace the distance metric in nearest neighbors in order to convert it to LMN. LMN has demonstrated better accuracy than k-NN in cases where the classes have different densities (Ting et al. (2018)), but LMN can only deal with dimensionality of a small or medium size due to its utilization of iForest since each iTree utilizes a small subset of features. The utilization of a mass-based similarity measure by LMN enables it to overcome nearest neighbors density problems, but one must determine whether the utilization of a mass-based similarity, especially the computationally expensive utilization of iForest as in the current implementation of LMN, is truly necessary or whether the density problem could be overcome by utilizing existing, and computationally simpler, methods. C-KNN and E-KNN are algorithms that receive a data set and only retain the data points necessary to maintain the decision boundary. Since these reduced nearest neighbor methods only retain the points needed for the decision boundary, thereby helping to even out the densities of the classes assuming unweighted neighbors, one must determine whether it could overcome the density problem similar to LMN. We ran an experiment comparing k-NN, LMN, C-KNN), C-LMN, E-KNN, and E-LMN for classification on a subset of the same datasets utilized by Ting et al. (2018) to test the hypothesis that LMN does not tend to have a significantly different accuracy than reduced nearest neighbor methods on data sets where LMN does have a significantly higher accuracy than k-NN.

4. Algorithms Utilized

4.1 *K* Nearest Neighbors (*k*-NN)

k-NN is a lazy algorithm that utilizes distance to formulate predictions. Nearest neighbors calculates the distance between a new observation and all examples in its data set and selects the k nearest as the new observation's "neighbors". These neighbors then vote for their class and the class with the most votes is returned as the predicted class.

4.2 Condensed Nearest Neighbors (C-KNN)

C-KNN works identically to k Nearest Neighbor (k-NN) but holds a reduced example set. Condensed nearest neighbors takes the example set and retains only the points necessary to maintain the decision boundary. C-KNN begins by initializing a new example set with one example. It then loops through the original example set classifying each point using the new example set. Each point that is misclassified is added to the new example set. This process continues until it can iterate through the original example set without misclassifying any points.

4.3 Edited Nearest Neighbors (E-KNN)

E-KNN also works identically to k Nearest Neighbor (k-NN) but holds a reduced example set. E-KNN starts with the entire original example set. It classifies each point in the example set using the other points. If the point is correctly classified and its removal does not degrade performance on an external validation set, it is removed from the example set. This process continues until it can iterate through the held example set without removing any points.

4.4 Lowest Probability Mass Neighbors (LMN)

LMN is an algorithm created by Ting et al. (2018) to overcome key shortcomings of nearest neighbor algorithms. LMN is simply nearest neighbors with the distance metric replaced with a mass-based similarity. In Ting et al. (2018), probability mass is determined by utilizing a modified version of iForest (Liu et al. (2008)) directly using mass (Aryal et al. (2014)). A hierarchical partitioning method is utilized to define the regions for mass estimation. The first step is to build an iForest containing t iTrees. Each iTree is constructed by sampling without replacement from the data set to build a subset D of size ϕ. The set is then split randomly along an axis and the two resulting sets become children with each child containing the elements of the parent that fall to one side of the (axis parallel) split. This process is repeated until either every leaf contains a solitary point, or the maximum tree height h (determined by $h = \log_2 \phi$ in the paper) is reached. The initial mass of each node is the number of points from D in that region (so the mass of the root node is $\|D\|$). The entire data set is then filtered through each of the t iTrees until all points reach a leaf (moving to the left or right child from each parent node depending on whether the value of a specific attribute is less than or greater than the given axis parallel split for that parent) and every point increases the mass of every node it passes through by 1. The second step is to define the mass-based dissimilarity. To determine the dissimilarity between points x and y, Ting et al. (2018) takes the average mass of the deepest node in each iTree containing both x and y and then normalizes by the size of the data set.

5. Experimental Methods

For classification, Ting et al. (2018) compared LMN, k-NN, Extended Nearest Neighbors (Tang and He (2015)), Large Margin Nearest Neighbors (Weinberger and Saul (2009)), and Geometric Mean Metric Learning (Zadeh et al. (2016)) on thirty data sets from the UCI Machine Learning Repository (Lichman (2013)) using $k = 5$ (arbitrarily chosen) and default parameters for the modified iForest ($t = 100$ and $\phi = 256$). For our experiment, we used four of the same data sets (German, Australian, Ionosphere, and Vowel-Context) to compare k-NN, C-KNN, E-KNN, and the lowest probability mass variant of each of those three algorithms. Ting et al. (2018) used the Vowel data set, whereas this experiment utilized Vowel-Context. Vowel-Context is a reformatted and expanded version of the Vowel data set with more examples. The goal of this experiment was to test the hypothesis that LMN does not tend to have a significantly different accuracy than reduced nearest neighbor methods on data sets where LMN does have a significantly higher accuracy than k nearest neighbors. We compared the accuracy of the algorithms using a 10-fold cross validation paired t-test. A 10-fold cross validation paired t-test has elevated probability of type 1 error but has high power. Data was normalized prior to the experiment. Algorithms were compared pairwise using $\alpha = 0.05$. Code for these algorithms is available at https://github.com/LoganCaraway/Lowest-Probability-Mass-and-reduced-Nearest-Neighbors-Research.

Algorithm parameters were tuned in a 10-fold cross validation paired t-test for each data set. Algorithms with multiple parameters had all parameters set to default values, then the parameters were iterated through and tuned one at a time.

The number k of neighbors was tuned by setting $k = 1$ and increasing k in steps of one until increasing k resulted in consistently worsening performance. The k resulting in the best performance for each algorithm was then used for that algorithm.

The number of iTrees in lowest probability mass neighbors, t, was set to its default value of 100 and changed in steps of five in both directions until changing t resulted in consistently worsening performance. The t resulting in the best performance was then utilized.

The subsample size, ϕ, in lowest probability mass neighbors was set to its default value of 256 and changed in steps of 16 in both directions until changing ϕ resulted in consistently worsening performance. The ϕ resulting in the best performance was then utilized.

The max iTree height, h, was calculated using the internal formula $h = log_2\phi$.

The datasets were converted to CSV prior to execution. Properties of the data sets that were utilized are summarized in table 1.

Data Set	[#] **Elements**	[#] **Features**	[#] **Classes**
Australian	690	14	2
German	1000	24	2
Ionosphere	351	34	2
Vowel-Context	990	10	11

Table 1: Data Set Properties

6. Results

Algorithm	k	t	ϕ
k**-NN**	15	-	-
LMN	7	145	336
C-KNN	15	-	-
C-LMN	12	155	208
E-KNN	13	-	-
E-LMN	7	110	224

Table 2: Australian Parameter Values

Algorithm	**Accuracy**
k**-NN**	0.855
LMN	0.858
C-KNN	0.854
C-LMN	0.829
E-KNN	0.849
E-LMN	0.867

Table 3: Australian Results

The parameters for the Australian data set are displayed in table 2. The accuracy is displayed in table 3. C-LMN performed significantly worse than LMN and E-LMN. With the exception of C-LMN performing worse, none of the algorithms performed significantly different. In Ting et al. (2018), t and ϕ were left at default values (100 and 256) and k was arbitrarily set to 5; Ting et al. (2018) found that their LMN performed significantly better than k-NN. After tuning, however, no significant difference was found.

Algorithm	k	t	ϕ
k-NN	16	-	-
LMN	16	105	368
C-KNN	14	-	-
C-LMN	13	100	256
E-KNN	12	-	-
E-LMN	17	70	368

Table 4: German Parameter Values

Algorithm	**Accuracy**
k-NN	0.731
LMN	0.732
C-KNN	0.698
C-LMN	0.698
E-KNN	0.705
E-LMN	0.731

Table 5: German Results

The parameters for the German data set are displayed in table 4. The accuracy is shown in table 5. k-NN performed significantly better than C-KNN and E-KNN. LMN performed significantly better than C-KNN, E-KNN, and C-LMN. E-KNN performed significantly better than C-KNN and E-LMN performed significantly better than C-KNN and E-KNN. In Ting et al. (2018), t and ϕ were left at default values (100 and 256) and k was arbitrarily set to 5; Ting et al. (2018) found that their LMN performed significantly better than k-NN. After tuning, however, no significant difference was found.

Algorithm	k	t	ϕ
k-NN	1	-	-
LMN	1	110	304
C-KNN	8	-	-
C-LMN	2	135	288
E-KNN	1	-	-
E-LMN	6	95	336

Table 6: Ionosphere Parameter Values

Algorithm	**Accuracy**
k-NN	0.875
LMN	0.914
C-KNN	0.877
C-LMN	0.886
E-KNN	0.877
E-LMN	0.903

Table 7: Ionosphere Results

The parameters for the Ionosphere data set are displayed in table 6. The accuracy is displayed in table 7. LMN performed significantly better than k-NN, C-KNN, and E-KNN in pairwise comparisons. E-LMN performed significantly better than E-KNN. Ting et al. (2018) also found that LMN performed significantly better than k-NN. For the Ionosphere data set, LMN performed better than k-NN as well as the reduced nearest neighbor methods such as C-KNN and E-KNN.

Algorithm	k	t	ϕ
k-NN	2	-	-
LMN	2	180	272
C-KNN	2	-	-
C-LMN	1	195	304
E-KNN	2	-	-
E-LMN	1	165	320

Table 8: Vowel-context Parameter Values

Algorithm	**Accuracy**
k-NN	0.988
LMN	0.982
C-KNN	0.942
C-LMN	0.928
E-KNN	0.964
E-LMN	0.953

Table 9: Vowel-context Results

The parameters for the Vowel-Context data set are displayed in table 8. The accuracy is displayed in table 9. k-NN and LMN both had significantly better accuracy than the reduced nearest neighbor methods and the reduced LMN methods. C-KNN had significantly better performance than C-LMN. E-KNN had a significantly better performance than both reduced LMN methods. Similar to Ting et al. (2018), we found no significant difference between k-NN and LMN.

As mentioned previously, Ting et al. (2018) arbitrarily set k to 5 for all the algorithms and left t and ϕ at default values. Using these settings, Ting et al. (2018) found that LMN performed better than k-NN on three of the four datasets that we used (Australian, German, and Ionosphere), yet after tuning the algorithms, we found that LMN only performed better than k-NN on one of these three datasets. One possible direction for future research is running the experiment from Ting et al. (2018) again and tuning all the algorithms to determine how effective LMN actually is at overcoming the density problem. Additionally, future research can be conducted to compare k-NN, LMN, and variants of the reduced nearest neighbor methods designed to fight over-fitting.

Despite the fact that LMN only performed better than k-NN on one dataset (Ionosphere) instead of the expected three data sets, on the data set where LMN performed better than k-NN, LMN also performed better than both reduced nearest neighbor methods.

One possible reason that C-KNN and E-KNN performed poorly on all data sets except the Australian data set is the tendency for these algorithms to undergo over-fitting. These algorithms remove points that are not necessary in order to maintain the decision boundary, but for them to remove enough points to even out the densities, they will tend to subject the training data to over-fitting.

7. Conclusion

Nearest neighbors has been shown to have an increased misclassification rate in cases where the classes have different densities (Ting et al. (2018)). Ting et al. (2018) replaced the distance metric in nearest neighbors with a mass-based similarity measure to create LMN and showed that this new algorithm performs significantly better than nearest neighbors in many cases where the classes have differing densities in an experiment without tuning parameters. Once parameters were tuned, we found that LMN only performed significantly better than k-NN one one-third of the expected datasets tested. Unfortunately, the current implementation of LMN can only deal with dimensionalities with small or medium sizes and is computationally expensive. We explored whether condensed nearest neighbors and edited nearest neighbors, due to only retaining the data points necessary to maintain the decision boundary, has similar performance to LMN in cases with the density problem. We ran an experiment comparing k-NN, LMN, C-KNN, C-LMN, Edited Nearest Neighbors, and E-LMN for classification to test the hypothesis that LMN does not tend to have a significantly different accuracy than reduced nearest neighbor methods on data sets where LMN does have a significantly higher accuracy than k-NN. Due to LMN not performing significantly better than k-NN in two-thirds of the data sets where Ting et al. (2018) found that it would once the algorithms were tuned, the ability of LMN to overcome the density issue is brought into question. It should be noted that, while LMN only performed better than k-NN on one data set, C-KNN and E-KNN performed significantly worse than LMN on all data sets except the Australian data set, where there was no significant difference. It is believed that this discrepancy is due to C-KNN and E-KNN over-fitting the training data.

References

Ethem Alpaydin. *Introduction to Machine Learning*, chapter 8. MIT Press, 2014.

Sunil Aryal, Kai Ming Ting, Jonathon R. Wells, and Takashi Washio. Improving iforest with relative mass. *Advances in knowledge discovery and data mining*, pages 510–521, 2014.

Ray A. Jarvis and Edward A. Patrick. Clustering using a similarity measure based on shared near neighbors. *IEEE Transactions on Computers*, 100(11):1025–1034, 1973.

Carol L. Krumhansl. Concerning the applicability of geometric models to similarity data: The interrelationship between similarity and spatial density. *Psychological Review*, 85(5):445–463, 1978. doi: https://doi.org/10.1037/0033-295X.85.5.445.

Sanjeev Kularni and Gilbert Harman. *An Elementary Introduction to Statistical Learning Theory*, chapter 7. John Wiley & Sons, 2011.

M. Lichman. UCI machine learning repository, 2013. URL http://archive.ics.uci.edu/ml.

Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. *Advances in knowledge discovery and data mining*, pages 413–422, 2008.

Bo Tang and Haibo He. ENN: Extended nearest neighbor method for pattern recognition. *IEEE Computational Intelligence Magazine*, 10(3):52–60, 2015.

Kai Ming Ting, Guang-Tong Zhou, Fei Tony Liu, and Swee Chuan Tan. Mass estimation. *Machine Learning*, 90(1):127–160, 2013.

Kai Ming Ting, Ye Zhu, Mark Carman, Yue Zhu, and Takashi Washio. Lowest probability mass neighbour algorithms: relaxing the metric constraint in distance-based neighbourhood algorithms. *Machine Learning*, 108:331–376, 2018.

A. Tversky. Features of similarity. *Psychological Review*, 84(4):327–352, 1977. doi: https://doi.org/10.1037/0033-295X.84.4.327.

Kilian Q. Weinberger and Lawrence K. Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10(2):207–244, 2009.

Pourya Habib Zadeh, Reshad Hosseini, and Suvrit Sra. Geometric means metric learning. *Proceedings of the 33rd international conference on machine learning*, pages 2464–2471, 2016.