




# NNDE : a python package for Nearest Neighbor Density Estimation

Yuheng Ma <sup>1</sup>

<sup>1</sup> School of Statistics, Renmin University of China

DOI: [10.xxxxxx/draft](https://doi.org/10.xxxxxx/draft)

## Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: [Open Journals](#) 

## Reviewers:

- [@openjournals](#)

Submitted: 01 January 1970

Published: unpublished

## License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#)).

## Summary

Nearest Neighbor based Density Estimation is a class of density estimation method which improve the traditional kernel density estimation by allowing the estimation have varying bandwidth depending on nearest neighbor distances. Several advantages are possessed by NN based density estimations. They are lazy learning methods that require no training stage. They utilize local information for bandwidth selection ([Orava, 2011](#)). Their straightforward logic naturally satisfies the requirements of trustworthy AI ([Göpfert et al., 2022](#); [Papernot & McDaniel, 2018](#)). The NNDE package provide an efficient implementation of six NN based density methods that users can directly apply in related studies. The package are presented in class-based manner for extensibility and is compatible with *scikit-learn* based parameter tuning functions such as *sklearn.GridSearchCV*. NNDE's built-in cython implemented adaptive KD tree, which is modidied from *sklearn.neighbors.KDTree*, provides convinient local neighbor selection scheme and is extensible for more adaptive selection functions. Moreover, we provide efficient tools for complex distributions generation and density estimation visualization.

## Statement of Need

There are barely any implementation of nearest neighbor based density estimation methods since the ordinal algorithm is, if equipped with well developed KD tree structure, straight forward and leaves no space for further optimizing. However, as ([Kovacs et al., 2017](#); [Zhao & Lai, 2021](#)) illustrated, performance of estimation, such as accuracy and robustness, is improved if the estimator is chosen from a large functional space, for instance when NN density estimation is weighted or adaptive. These evolutions of NN bring challenge to algorithm implementation as well as parameter tuning. NNDE for the first time provides user friendly functions for six NN based density methods, namely KNN ([Loftsgaarden & Quesenberry, 1965](#)), WKNN ([Biau et al., 2011](#)), TKNN ([Zhao & Lai, 2021](#)), BKNN ([Kovacs et al., 2017](#)) and newly proposed AKNN and AWNN. For parameter tuning, NNDE is compatible with cross validation methods in *scikit-learn* and is extensible for further development. Under research framework of density estimation, researchers in this area often deal with complex distributions. NNDE include efficient functions for generating complex distributions such as densities with different marginals and mixture of densities. Also, basic visualization tools that exhibit behavior of estimation in arbitray dimensions are provided.

A key component for NN based methods is the KD tree structure. A great many packages provided different implementation schemes Virtanen et al. ([2020](#)) for KD tree. BodoBookhagen ([2013](#)) has done a thourough comparison from perspectives such as dimension restriction, query speed and parallelizability. However, consider if we want to search the largest  $k$  such that  $k \cdot R_k(x) < C$ , where  $R_k(x)$  is the  $k$ th nearest neighbor distance for  $x$ . For common KD tree implementations, we would have to query  $(R_1(x), \dots, R_n(x))$  and search iteratively. This guarantees the correct solution when  $k = n - 1$  but causes much waste of computation

if  $k = 2$ . Thus, NNDE choose to modify sklearn, which is based on cython, to implement a KD tree structure with built in adaptive neighbor search algorithm. The algorithm halts the searching of neighbors when nearest neighbor distances exceed some thresholds, which is a much more efficient approach than static searching after querying all the distances.

## Methods

In this section, we introduce the nearest neighbor estimation methods included in NNDE. We denote  $R_k(x)$  as the distance between  $x$  and its  $k$ -th nearest neighbor. Given  $n$  independent identically distributed dataset  $\{X_1, \dots, X_n\} =: D_n \in \mathbb{R}^d$ , the standard  $k$ -NN density estimator (**KNN**) (Dasgupta & Kpotufe, 2014; Loftsgaarden & Quesenberry, 1965) for each point  $x \in \mathbb{R}^d$  is defined as

$$f_k(x) = \frac{k/n}{V_d R_k^d(x)}$$

where  $V_d$  be the volume of the unit ball in  $\mathbb{R}^d$ . An intuitive explanation of this estimator is that from order statistics.  $P(B(x, R_k(x)))$  follows Beta Binomial distribution  $Beta(k, n - k + 1)$ . As a result, we have  $\mathbb{E}[P(B(x, R_k(x)))] = k/(n + 1)$ . Therefore, we have the approximation  $k/n \approx P(B(x, R_k(x))) \approx f(x)V_d R_k^d(x)^d$ . Biau et al. (2011) intended to smooth the standard KNN by aligned weighting and proposed **WKNN** defined as

$$f_W(x) = \frac{\sum_{i=1}^k w_i i/n}{V_d \sum_{i=1}^k w_i R_i^d(x)}$$

where  $w_1, \dots, w_k$  are fixed positive weights summing to 1. In practice,  $w_i$  is usually set to  $1/k$ . KNN and WKNN both utilize fixed weights for all  $x$ . Zhao & Lai (2021) proposed truncated method **TKNN** to fix the potential problem at the tail of distribution. They perform a pre-estimation using uniform kernel and substitute the pre-estimation with standard KNN when pre-estimation is large, namely

$$\hat{f}(x) = \begin{cases} \frac{k}{nV_d R_k^d(x)} & \text{if } \hat{f}(x) \leq a \\ \hat{f}(x) & \text{if } \hat{f}(x) > a \end{cases}$$

where  $\hat{f}(x)$  is uniform kernel density estimator. Their method is still non-adaptive, i.e. using fixed number of neighbors, in most regions. Recently, Kovacs et al. (2017) argued that adaptive choice of weights (number of neighbors) brings advantages to estimation when encountering densities with drastically varying value. Their **BKNN** demonstrate that by selecting largest  $k(x)$  such that

$$k(x) \cdot R_{k(x)}^d(x) < C$$

for some constant  $C$  and dimension  $d$ , the prediction  $k(x)/(nV_d R_{k(x)}^d(x))$  performs better. However, choice of the statistic lacks theoretical support. Also, BKNN provide a plug-in parameter selection scheme for  $C$ , which is quit empirical and fails when dimension is higher than 2. A work in progress, called **AKNN**, argues that the suitable choice is instead

$$k(x) \cdot R_{k(x)}^2(x) < C.$$

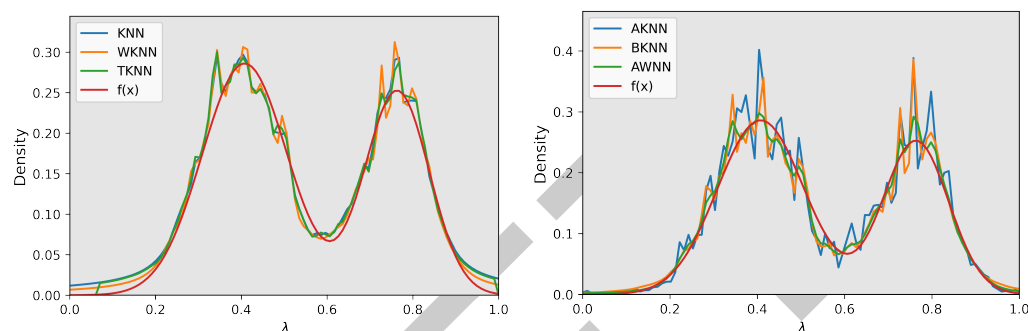
while the selection of  $C$  is done by cross validation. Moreover, another work in progress consider adaptive weighted nearest neighbor estimation. Weighted estimation for each  $x$  is formalized as

$$f_A(x) = \frac{\sum_{i=1}^{k(x)} w_i(x) i/n}{V_d \sum_{i=1}^{k(x)} w_i(x) R_i^d(x)}$$

where  $w_1(x), \dots, w_{k(x)}(x)$  are local weights at  $x$ . Motivated by the bias-variance decomposition for the pointwise error, an efficient optimization approach is provided to select the weights

of nearest neighbors for each  $x$  adaptively. In practical, We apply KD tree from *scikit-learn* to KNN, WKNN, TKNN and AWNN. Both AKNN and BKNN are implemented through the built-in function in adaptive KD tree and achieve roughly the same computational cost as normal KD tree query. The optimization algorithm of AWNN for weight selection is accelerated via *numba*.

In what follows, we use a toy example to exhibits functionality of NNDE. We first generate 1000 samples from a mixture of Gaussian distribution. The following figure shows the estimation results of methods in NNDE.



## Dependencies

*NNDE* utilizes tools and functionality from *numpy* (Harris et al., 2020), *matplotlib* (Hunter, 2007), *scipy* (Virtanen et al., 2020), *jupyter notebooks* [ipython-2007], *scikit-learn* (Buitinck et al., 2013), *cython* (Behnel et al., 2011) and *numba* (Lam et al., 2015).

## Reference

- Behnel, S., Bradshaw, R., Citro, C., Dalcin, L., Seljebotn, D. S., & Smith, K. (2011). Cython: The best of both worlds. *Computing in Science & Engineering*, 13(2), 31–39.
- Biau, G., Chazal, F., Cohen-Steiner, D., Devroye, L., & Rodriguez, C. (2011). A weighted k-nearest neighbor density estimate for geometric inference. *Electronic Journal of Statistics*, 5, 204–237.
- BodoBookhagen. (2013). LidarPC-KDTree. In *GitHub repository*. <https://github.com/UP-RS-ESP/LidarPC-KDTree>; GitHub.
- Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., Niculae, V., Prettenhofer, P., Gramfort, A., Grobler, J., Layton, R., VanderPlas, J., Joly, A., Holt, B., & Varoquaux, G. (2013). API design for machine learning software: Experiences from the scikit-learn project. *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, 108–122.
- Dasgupta, S., & Kpotufe, S. (2014). Optimal rates for k-nn density and mode estimation. *Advances in Neural Information Processing Systems*, 27.
- Göpfert, J. P., Wersing, H., & Hammer, B. (2022). Interpretable locally adaptive nearest neighbors. *Neurocomputing*, 470, 344–351.
- Harris, C. R., Millman, K. J., Walt, S. J. van der, Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., Kerkwijk, M. H. van, Brett, M., Haldane, A., R'io, J. F. del, Wiebe, M., Peterson, P., ... Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, 585(7825), 357–362. <https://doi.org/10.1038/s41586-020-2649-2>

- 111 Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing In Science &*  
112 *Engineering*, 9(3), 90–95. <https://doi.org/10.1109/mcse.2007.55>
- 113 Kovacs, J. A., Helmick, C., & Wriggers, W. (2017). A balanced approach to adaptive probability  
114 density estimation. *Frontiers in Molecular Biosciences*, 4, 25.
- 115 Lam, S. K., Pitrou, A., & Seibert, S. (2015). Numba: A llvm-based python jit compiler.  
116 *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*, 1–6.
- 117 Loftsgaarden, D. O., & Quesenberry, C. P. (1965). A nonparametric estimate of a multivariate  
118 density function. *The Annals of Mathematical Statistics*, 36(3), 1049–1051.
- 119 Orava, J. (2011). K-nearest neighbour kernel density estimation, the choice of optimal k.  
120 *Tatra Mountains Mathematical Publications*, 50(1), 39–50.
- 121 Papernot, N., & McDaniel, P. (2018). Deep k-nearest neighbors: Towards confident, inter-  
122 pretable and robust deep learning. *arXiv Preprint arXiv:1803.04765*.
- 123 Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D.,  
124 Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson,  
125 J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., ... SciPy  
126 1.0 Contributors. (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in  
127 Python. *Nature Methods*, 17, 261–272. <https://doi.org/10.1038/s41592-019-0686-2>
- 128 Zhao, P., & Lai, L. (2021). On the convergence rates of KNN density estimation. *2021 IEEE*  
129 *International Symposium on Information Theory (ISIT)*, 2840–2845.