JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue

JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)

An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Application of K-nearest Neighbor in Medicine by Isolation Forest Approach

Blessa Binolin Pepsi M[1], Deepika J[2], Nandhini R[3], Viruthika M[4]

[1]Assistant Professor (Senior Grade), [2,3,4]UG Degree

[1,2,3,4] Department of Information Technology

[1,2,3,4]Mepco Schlenk Engineering College

Sivakasi, Tamil Nadu.

Abstract : K-nearest neighbor (kNN) search is a significant issue in knowledge discovery and data mining. We suggest a novel approach for kNN search called random projection forests, which is motivated by the enormous success of tree-based techniques and ensemble approaches over the past decades (rpForests). In order to locate the closest neighbor's rpForests combines several kNN-sensitive trees, with each through a sequence of arbitrary projections in recursive fashion. Experiments on a large number of real datasets have shown that our strategy is effective. Produces a remarkable accuracy in terms of the disparity in the k^{th} nearest neighbor and the fast-declining missing rate of kNNs. As a tree-based approach, rpForests has a very low computational complexity. The rpForests ensemble structure makes on clustered or multicore machines, is readily parallelized to run, the running. By demonstrating the exponential decline of nearby points as a theoretical insight into IsolationForests when the ensemble size grows, they are divided by ensemble random projection trees.

IndexTerms **- Component,formatting,style,styling,insert. RpForests, K-Nearest Neighbor, ensemble, IsolationForests**

I. INTRODUCTION

To compute kNNs, a variety of techniques have been developed [1], it is outside the scope of this paper to undertake a comprehensive examination ofliterature. References to and their contents are provided for the users. Emerging applications of kNN search are of interest to us. The large amount of data involved in current apps is one of their most notable characteristics. When it comes to the number of records that must be processed, the data volume can easily approach a million. For instance, transaction logs or click streams at a big e-commerce website, data from wearables, sensors in moving vehicles (for road condition monitoring), messages or photos strewn across the internet, and data generated from portable devices like the IPhone. The fact that the underlying data is highly dimensional is another characteristic of developing applications. Gene expression data, for instance, frequently is another characteristic of developing applications. Gene expression data, for instance, frequently contains data dimensions greater than 10k, but a typical photograph from the internet has a size . It is commonly acknowledged that the 5Vs—volume, velocity, variety, veracity, and value—are the characteristics of contemporary "big data."

Our focus is on scalability and dimensionality, but it turns out that because our approach is tree-based, it can also successfully handle fast-arriving stream data. As a result, pertinent data structures may be readily and quickly updated when new data is received. Emerging applications also frequently require more precise kNN search, which is another characteristic of them[3]. For instance, inaccuracy in a kNN search may result in an incorrect route or a failure to avoid obstacles in robotic route planning.

When using face identification or unsupervised image labeling, surveillance systems often identify a person by matching their face to.Inspired by the success of Random Forests (RF), we investigate an ensemble of tree-based approaches for kNN the idea of ensemble to improve algorithmic performance is well-established in statistics and machine learning and has been the key component of some of the most successful machine learning algorithms[4]. Therefore even for more accurate prediction we use IsolationForests(IF) approach. More recently, a group of 106 classifiers came out on top of the well-known Netflix Challenge. Our method derives a number of desirable traits from tree-based methods because trees serve as its fundamental building component. It is well known that trees are insensitive to monotonic data modifications. Once the trees are formed, tree-based approaches are often relatively efficient with computational complexity at the order of the tree heights.The following are our work's significant contributions. First, we provide a technique that combines the adaptability of tree-based techniques with the strength of ensemble techniques approach is easily implemented, very scalable, and available adapt to the underlying data's shape. As the since the approach is ensemble-based[5], parallel execution is simple. The likelihood of adjacent points being separated by ensemble random projection is the second concept we construct. Trees, such likelihood would explain why a tree was constructed. Through random projections might be appropriate for KNN search, and it's true that errors decline exponentially when trees are growing in number. Third, our hypothesis has an intriguing implication that it can be utilized to inform decision-making projections at random, those corresponding to directions along preferred which the data stretches more[6,7].

In fact, nearly prior randomized tree techniques have concentrated on the selection rather than the direction, of the bifurcation point, IsolationForests focuses on the dividing direction, and our research indicates that this approach is effective. This approach may be used in a broader context. The remainder of this paper is organized as follows. In Section 3, we give a detailed description of our algorithm, along with a toy example to illustrate IsolationForest. This is followed by a little theory on the probability of a miss in kNN search by IsolationForests[9-15] in Section 4. Section 5 contains the list of various datasets.In Section 6, we present our experimental results on a wide collection of real datasets. Finally, we conclude in Section 6.

II. OBJECTIVE

Finding the closest neighbors for a given data point on an interesting distance metric is referred to as Nearest Neighbor Search. In many domains, including data mining, machine learning, statistics, anomaly detection, etc., it is a crucial task. A similarity search in a large dataset can typically be formulated in data mining. Applications in K Nearest Neighbor(KNN) which includes, Medicine, Finance etc. This paper focuses on how, K Nearest Neighbor Algorithm can be applied in Medicine Field using Isolation Forest Technique. In machine learning, such as kernel techniques, one computes the Gram matrix frequently. The computational complexity for a simple implementation would be $O(n^2)$, where n is the total number of data points. A crucial group of techniques would sparsify the Gram matrix. First, and a common method, is to create a k-nearest kNN neighbor graph, followed by collapse

III. RELATED WORKS

In [21-27] it describes a novel approach in this part termed the Minimal Spanning Tree-Based Isolation Forest (MSTBIF). It significantly affects Isolation Forest's fundamental strategy. The improvements happen in two ways. The first is to get betteremploying a minimum spanning tree to build search trees [16]. The second one is the modification of the metric for assessing anomalies.Let's run through the entire tree training procedure.The algorithm begins in the same manner as the basic IF instance[28]. The components are selected at random from the complete set being taken into account. All coordinates are normalized using the following formula in order to maintain the accurate distances between the elements.

$$z = x - {}^{\mu}/_{\sigma}$$

If x is a non-standardized variable, and zis is a standardized variable, and where and are the average and standard deviation of the standardized dimension for the previously chosen items, respectively.The modification of the isolation tree building technique is then introduced. The fundamental strategy was to divide apart the tree's parts. In contrast, our proposal proposes to form a tree by suitably merging the pieces rather than dividing them. The Euclidean distances between each element of the isolation tree are determined. The list of edges with weights matching to the previously determined distances is then completed, connecting them. Ascending value order is used to arrange this set of edges. The preparation of the isolation tree comes next. We choose components at random from the complete set, just like in the fundamental technique.

A distinct single-element tree represents each initial choice of element. The trees are combined in the following phases using an algorithm based on the least spanning tree and the separation between elements in the edge table. We combine trees using the Kruskal's algorithm. We can get an isolation tree from a minimal spanning tree thanks to the merges. The isolation tree is updated by adding a new merge node for the relevant trees each time any two minimal spanning trees are linked. It shows the process's illustration.If we obtain a single tree that contains every member of the subset drawn in the initial phase of tree building, the process is complete.The height of the tree, just like in the base technique, influences whether an analyzed element traversing the tree is a member of the anomalous set.There are various processes involved in building an isolation tree. Before anything else, we set up data converters for the scalability of dimension values. Converters apply the formula to convert attributes to standardizedvalues[17-18].The enhanced version of the Isolation Forest building technique is identical to the standard Isolation Forest technique despite differences in the construction technique. Whereas, in our desired approach the creation of each tree is based on After the construction mentioned above, it is added to the forest using randomly chosen points from the complete set.In[29-34], extending isolation forest for anomaly detection in

Big data via k-means, In the paper, it presented an intrusion detection method for huge network traffic data to identify anomalies. We begin by addressing the problems that make it difficult for the Isolation Forest algorithm to be trained on sizable industrial datasets. Then we show how we get over these issues so that studies on such huge datasets can be repeated in the future.By combining Isolation with our suggested intrusion detection system, we present a revolutionary machine learning approach.Additionally addressing the difficulties in training the Isolation Forest method in big data environments is Forest with K-Means. We find that our proposed intrusion detection system can be trained on a large industrial dataset with impressive performance for anomaly detection while using fewer parameters than the original Isolation Forest model, according to our experiments in real-world large network traffic data at iSecurity. On 12 academic datasets, we compare the performance of our suggested strategy with that of other cutting-edge models [35]. Finally, we demonstrate that our system is successful for real-time detection of anomalies on the live streaming data at iSecurity, demonstrating the utility of our proposed intrusion detection system in the industrial arena.Our suggested IForest-KMeans's overall time complexity can be expressed as:

$O(n)+O(n\log n) \approx O(n\log n)$

The Overall Complexity of our proposed system is less than the above model [19].

IV. PROPOSED SYSTEM DESIGN

4.1 RPFORESTS

Our suggested approach makes use of a random projection tree, or its fundamental component is rpTree. A randomized tree is the rpTree version of the kd-tree, a popular data structure in data mining software. A tree-based approach uses recursive space partitioning, which operates as follows. It first divides the root node into two child nodes in accordance with a splitting rule, starting with the complete data set as the root node of a tree. It recursively applies the two child nodes' respective the same process until a stopping requirement, for as, the node is not big enough. The selection of a split direction and a split point are the two components of the node split. The heading might be a linear combination of numerous coordinates or a coordinate. In order to maximize a splitting measure, such as the Gini index [10], the sum of squared distances between data points in different partitions, etc., a split point is chosen. For rpTree, the tree node split functions a little differently. A node, let's say R, will split in the direction d, which will be chosen at random. A node R can be randomly split into its left and right children, $R = R_L \cup R_R$, in a variety of ways. One option that is frequently used is to uniformly choose a point, let's say s, at random from the range defined by the smallest and largest members of the projection onto d for all points in R. The projection of a point x ϵ R onto d for all points in W. For a point x € R, its projection onto d is given by

$$\frac{d.x}{|d|^2}$$

where . indicates dot product. Define the projection coefficient of all points in along direction d as $R_R = \{.x : x\epsilon R\}$. Let the projection coefficient at the splitting point be denoted by a. Then the left child is given by $R_L=\{x € R: d.x<a\}$, techniques like the kd-tree have the advantage of being able to adjust to the geometry of the underlying data and easily escape the dimensionality curse. As a main data structure for quick calculation, rpTree has been utilized frequently. It is clear that approximation near neighbor search using tree-based algorithms has a very low computing complexity.

The average computational difficulty of growing the tree for n data points is O (nlogn), while the computational complexity of traversing a data point from the root node to a leaf node during a search is O (logn). A flaw with such procedures is the rptree.It is clear that rpTree, in particular, has a very low computational complexity for approximate near neighbor search when using tree-based approaches. A search entails moving a data point from the root node all the way down to a leaf node, which has a computational complexity of O(log n), whereas the growth of the tree for n data points has an average computational complexity of O(n log n).All of the data points are initially contained in the root node. Consider the case where data point A is the only one we are interested in, and all of its kNNs are coloured blue (additional points are not represented for clarity of presentation). One of point A's kNNs is now located at a different child node than it would have otherwise due to the split of the root node. Possibly more kNNs of A will be split off as the rpTree expands. Point A would eventually land in the same leaf node as the majority of its kNNs, with a handful of its kNNs might be an exception in various leaf nodes, which would result in a kNN error search.

4.2ISOLATION FORESTS (IF)

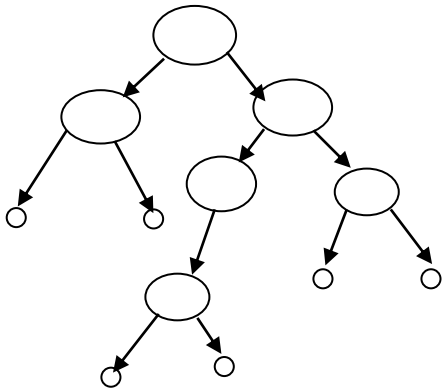

Fig. 1 An example of Isolation Forest randomly splitted.

Similar to Random Forests, Isolation Forests (IF) are constructed using decision trees. Additionally, this model is unsupervised because there are no predefined labels present. The foundation of IsolationForests is the idea that anomalies are the "few and different" data items. Randomly subsampled data is processed in an isolation forest using a tree structure based on randomly chosen attributes. Since it took more cuts to isolate the samples that travelled further into the tree, they are less likely to be abnormalities. Similar to the last example, data that end up on shorter branches tend to be anomalies since the likelihood of missing the best matches is significantly decreased when the child nodes of the same node in a k-d tree are permitted to overlap. As an alternative to the spill tree, the virtual spill tree, is introduced in which each data point is assigned to a single leaf node, but an overlapping split would send a data point to several leaf nodes, and the kNN search is then carried out on the union of these leaf nodes. However, because it entails choosing whether splits or nodes should overlap, the implementation of spill trees or its variants is challenging.An unsupervised machine learning approach for anomaly identification is isolation forest. Isolation Forest is an ensemble approach, as its name suggests. In other words, while determining the final anomaly score for a particular data point, it takes the average of the predictions made by a number of decision trees. Isolation Forest seeks to separate abnormal data points from the start, in contrast to other anomaly detection algorithms that first establish what is "normal" and then label everything else as anomalous [20].

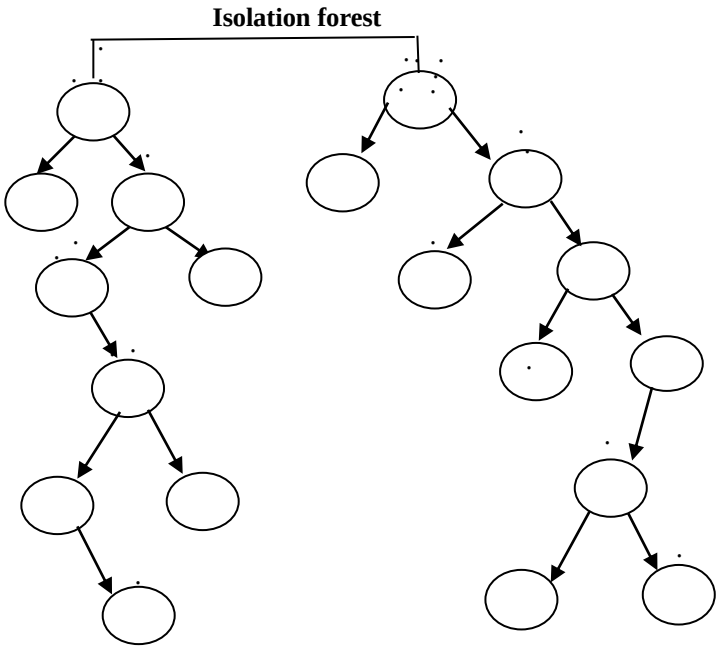

Fig. 2: Overall Workflow of Proposed System

The child nodes of the same node in a k-d tree are permitted to overlap in a spill tree, effectively increasing efficiency, lessen the likelihood of skipping the top matches. Each data point is assigned to a single leaf node in the virtual spill tree that Dasgupta and Sinha [14] suggested. However, an overlapping split would route a data point to several leaf nodes, and the kNN search is then carried out on the union of these leaf nodes. The decision of whether to overlap splits or nodes makes the implementation of the spill tree or its variants challenging. In order to lower the possibility of mismatches in kNNs search, we suggest using an ensemble of rpTree. The kNN search will be directed to one leaf node for each rpTree. The union of all the routed leaf nodes in the ensemble will now be the subject of the kNN search. Because each rpTree grows independently, the union of these leaf nodes will make up for mismatches in one node with some other nodes, lowering. Data points that require fewer splits to be isolated are given higher anomaly scores by isolation forest, which divided the data space using orthogonal to the origin lines. Isolation Forest was applied to the intervals between eruptions

In a large amount of cases, anomalies in a dataset may follow extremely complex patterns that are challenging to see visually. This is why machine learning techniques are highly suited for use in the field of anomaly identification.The most widely used methods for finding anomalies are based on creating a profile of what the likelihood of a mismatch. We refer to the resulting technique as random projection forests or rpForests because it is similar to RF and employs rpTree as a building block (with the splitting direction modified). Here, we provide a description of rpForests for kNN search.

V. IMPLEMENTATION

The Proposed System is implemented with the following modules:

1. The rpTreealgorithm(Algorithm 1)
2. Choose a splitting direction (Algorithm 2)
3. Using rpTree to locate kNNs (Algorithm 3)
4. Using Isolation Forest to locate kNNs(Algorithm 4).

Algorithm 1. RpTree (C)

1. Let C be the root node of tree t;
2. Initialize the set of working nodes R <-{C};
3. While R is not empty do
4. Randomly pick R ∈ R and set R <-R{R};
5. if |R| < n_8 then
6. Skip to the next round of the while loop;
7. End if
8. Generate a random direction $\vec{d}$;
9. Project points in R onto $\vec{d}$, $R_{\vec{d}}$={.x:x∈ R};
10. Let y=min(R_r) and z = max($R_{\vec{r}}$);
11. Generate a splitting point a ~run if|y,z|;
12. Split node R by R_L = {x: $\bar{P}_r$ (x < a} and R_R= {x:$\bar{p}_r$(x ≥ a};
13. R.Left<- R_L and R.right<-R_R;
14. Update the working set by R <- R ∪ {R_L, R_R};
15. End while
16. Return(t);

where, C -> The available data collection.
t ->rpTree will be constructed from C.
R -> collection of active nodes.The smallest number of data points in a tree node for which we shall further divide is represented by the specified constant Ns.P_r(x) is the coefficient of projection of the point x onto the line.Each component of the set of neighborhoods. N is a collection of nearby points in the set of neighborhoods C.

Algorithm 2. Projection (R, nTry)

1. Initialize display<-0,direction<-NULL;
2. For i=1 to nTry do
3. Generate a random projection d ;
4. Project R onto d and let R_R={.x:x ϵ R};
5. Let tDisplay<- SD($R_{r)}$;
6. if tDisplay> display then
7. Set display<- tDisplay and direction<-T;
8. End if
9. End for
10. Return (direction);

An elementary implementation of picking the splitting direction the output of our program would just produce a random heading, and after that decide where to split off in that direction. A To create a more careful design, create nTry random choose one of the projections so that the projected data the longest stretches. The statistical measure of the stretch of the variation of data or the spread (dispersion) a data. Thus, to calculate the amount of uncertainty in the anticipated data, we will utilize a straightforward statistic called the standard deviation. Spread of data as our theory will demonstrate, such a decision the split will be guided by the direction in which the data stretches. The "most," preventing the possibility of the node splitting.

Algorithm 3 kNNrpForests (Q, C)

1. For i=1 to T do
2. Build the i–thrpTree by t_i<-rpTree(C);
3. End for
4. For each data point q ϵ Q do
5. For i=1 to T do
6. Let q fall through $t_i$and arrive at leaf node N^i
7. Set the Union of neighbor sets of q by N_q<- $C^T{}_{i=1}$ $N^{i;}$
8. Compute distance between q and each point in N_q;
9. The k-points in N_q with smallest distance to q are its KNNSs
10. End for

Finds the k-nearest neighbors using the ensemble of rpTree. Let Q ϵC be the set of data points for which we wish to find their k-nearest neighbors in C. Q can be the set Citself.

Both are decision tree-based ensemble approaches namely Isolation Forest and RRCF (Robust Random Cut Forest) that seek to isolate every single point. The likelihood of the point being an inlier increases with the number of isolation stages, but the inverse is also true.Even though, features are sampled in the following ways at each recursive isolation: RRCF(Robust Random Cut Forest) offers more weight to dimensions with higher variance, but random samples from isolation forests are preferable.The way anomaly scores are assigned varies as well. The score for Isolation Forest is determined by how far it is from the root node. Based on how much a new point alters the tree structure, RRCF is calculated. Because of this, RRCF is less sensitive to sample size.

Algorithm 4 IsolationForests (Q, C)

1. Btree<- C { C ϵ Q}
2. Select a_i branch based onΘ.
3. For i=1 to D do
3. if D_i< Θ then R.Left<- R_L
4. else R.Right<- R_R
5. Update the working set by R <- R ∪ {R_L, R_R};
6. End for
7. For i=1 to T do
8. Build the i–thITree by t_i<- ITree(C);
9. End for
10. For each data point q ϵ Q do
11. For i=1 to T do
12. Let q fall through ti and arrive at leaf nodeNi
13. Set the Union of neighbor sets of q by Nq<- CTi=1 Ni;
14. Compute distance between q and eachpoint in Nq;
15. The k-points in Nq with smallest distance to q are its KNNSs
16. End for

It starts with training of data C, then by generating Isolation Trees. Where, Btree denotes a binary tree with two splits as Left & Right. A be the set of Attributes, {a_i ϵ A}. Θ the threshold, it lies between the maximum and minimum values of the feature {val_{min}< Θ <val_{max}}. ITree be the IsolationTrees ensembled to create a model. Finally, by Ensembling of IsolationForests , The k Nearest Neighbors are obtained.

VI. Performance Metrics

We establish two indicators to rate the algorithm's effectiveness. The first is represented by b, or the average missing rate per point. Pretend there are n data points, each represented by $X_1, X_2 \ldots X_n$. Let the number of those be denoted by b_i. For each point X_i, some of its kNNs might not be in the list of kNNs generated by the procedure. Then,

$$b = \frac{1}{nk}\sum_{i=1}^{n} b_i$$

Where, b is the average missing rate, n is no. of data points, k is the nearest neighbor, b_i is the missing rate. Assume we are interested in the five neighbors that are closest to us, or K= 5. The average missing rate as the number of rpTree rises is depicted in Fig. 3. The second metric, represented by d_k, is the average kNN distance calculated across all points. The separation between each point x_i and its k-th closest neighbouris denoted by the term $d_k(i)$, which stands for the kNN distance for x_i.

$$d_k = \frac{1}{n}\sum_{i=1}^{n} d_k(i)$$

Where, d_k is the average distance, n is the number of data points ,$d_k(i)$ is the distance. It is obvious that for any point, let's say X_i, if any of its kNNs are missed, the estimated kNN distance, $d_k(i)$, computed by an algorithm, satisfies $d_k(i)$. Consequently, the typical kNN distance will also increase. Therefore, for d_k, the smaller is preferable (or, equivalently, the closer to that estimated by brute force) the average kNN distance as the number of rpTree rises. The tendency is sharply dropping and similar. The average kNN distance is almost identical to that determined via brute force using approximately 5 rpTree.

VII. Results And Discussions

The Datasets used for our proposed system includes Medical Diagnosis Data. A set of real datasets which includes Fetal Heart Rate (FHR), Wisconsin Breast Cancer (WDBC), COVID 19, Dermatology are taken for analysis. We run tests on a large number of real-world datasets. All these datasets are from the UC Irvine Machine Learning Repository (UCI)[2], most are obtained from Computer Laboratory at Cambridge University. Even for more accurate predictions various other datasets had been used

namely, Smartphone Activity, Parkinsons, Arcene Dataset. Running time are taken into consideration when evaluating the algorithms.

TABLE 1 A list of the datasets used to evaluate performance

Dataset	Features	#Instances
Fetal Heart Rate	22	2126
Wisconsin Breast Cancer (WDBC)	30	569
COVID 19	16	209
Dermatology	35	366
Diabetics	50	101767
Hungarian	8	2940

We calculate the two measures while varying the set's tree count from 10 to 20 to 60 to 80 to 100 for each dataset.Averaging the outcomes across 100 runs is done. We choose K = 5 and K = 10 because real-world applications frequently show the greatest interest in these two values. We limit the size of the leaf nodes (i.e., the node capacity) to no more than 20 for K = 5 and 30 for K = 10 when growing the IsolationTrees. For each dataset, we normalise the average discrepancy in kNN distance before showing the results.

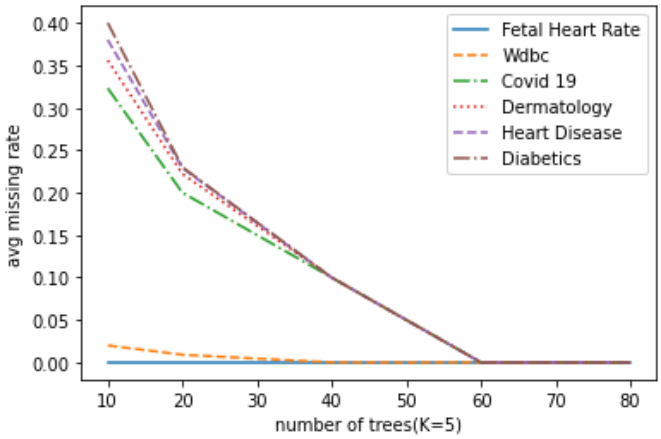

Fig. 3. The Average Missing rates by varying the no. of trees

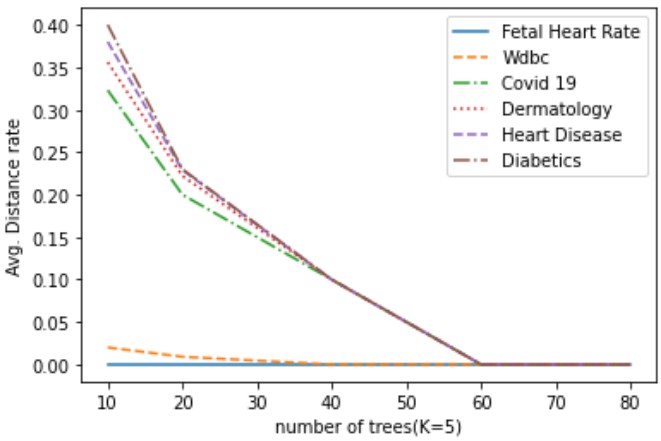

Fig. 4. The Average Discrepeancy rate by varying the no. of trees

Both of the figures show that as the number of trees rises, errors in kNN search by IsolationForests, as indicated by the average missing rates and the missing kNN distance, reduce relatively quickly. In fact, as anticipated by our hypothesis, this deterioration appears to occur at an exponentially high rate. For the majority of the datasets we examine, the errors abruptly disappear to zero with 20–40 trees. With the exception of the data from smartphones, for which 60 trees were used, IsolationForests accuracy was constructed with 40 IsolationTrees. There is no dominant algorithm, and all of the error rates are extremely low. With additional trees, we anticipate that the error rate for isolation forests will progressively decline.

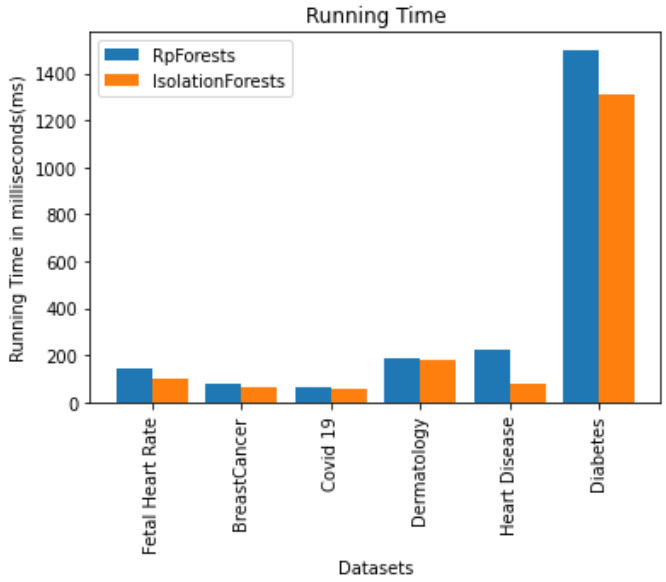

Fig. 5. Duration in milliseconds when K is equal to 5.

Running time is also another crucial component of our assessment.In Fig. 5, we compare the running time of our proposed system model (Isolation Forest) with that to the RpForests.

VIII. CONCLUSION

We have put forth a powerful kNN search method (Isolation Forest) that is straightforward to use, incredibly scalable, and easily adaptable to the geometry of the underlying data. The method is ensemble-based, making it simple to parallelize so that it can operate on clustered or multicore computers. As more cores or computers are added to the computation, the running time gets down. The versatility of tree-based approaches is offered by rpForests, which is also quick in terms of tree growth and search and permits data modification (as long as it is monotonic). Due to the frequent application of these qualities in feature engineering, they are desired qualities. We create a hypothesis for the rapidly decreasing likelihood that ensemble random projection trees will divide two neighboring points. This would account for the fact that a tree created using Isolation may be utilised for kNN search and that the inaccuracy decreases exponentially as the number of trees rises. In comparison to previous techniques, we can discover anomalies using Isolation Forest faster and with reduced memory usage. Instead of profiling typical data points, Isolation Forest isolates anomalies in the data points. Hence, this approach is widely used in various medical datasets for accurate prediction.

REFERENCES

[1] D. Yan, Y. Wang, J. Wang, H. Wang and Z. Li, "K-Nearest Neighbor Search by Random Projection Forests," in IEEE Transactions on Big Data, vol. 7, no. 1, pp. 147-157, 1 March 2021
[2] M. Lichman, UC Irvine Machine Learning Repository, 2013. [Online]. Available: http://archive.ics.uci.edu/ml
[3] P. N. Yianilos, “Data structures and algorithms for nearest neighbor search in general metric spaces,” in Proc. ACM-SIAM Symp.Discrete Algorithms, 1993, pp. 311–321.
[4] M. Penrose and J. Yukiko, “Laws of large numbers and nearest neighbor distances,” in Proc. Advances Directional Linear Statist., 2010, pp. 189–199.
[5] S. Ramaswamy, R. Rastogi, and K. Shim, “Efficient algorithms for mining outliers from large data sets,” in Proc. ACM SIGMOD Int.Conf. Manage. Data, 2000, pp. 427–438.
[6] M. Lucinska and S. Wierzchon, “Spectral clustering based on K-Nearest neighbor graph,” in Proc. Conf. Comput. Inf. Syst. Ind.Manag., Sep. 2012, pp. 254–265.
[7] B. Leibe, K. Mikolajczyk, and B. Schiele, “Efficient clustering and matching for object class recognition,” in Proc. Brit. Mach. Vis.Conf., Sep. 2006.
[8] P. Belanovi´c and M. Leeser, “A library of parameterized floating-point modules and their use,” in Proc. Int. Conf. Field Programm. Logic Appl. Berlin, Germany: Springer, 2002
[9] F. T. Liu, K. M. Ting and Z. -H. Zhou, "Isolation Forest," 2008 Eighth IEEE International Conference on Data Mining, Pisa, Italy, 2008, pp. 413-422
[10] Z. He, X. Xu, and S. Deng. Discovering cluster-based local outliers. Pattern Recogn. Lett., 24(9-10):1641-1650, 2003.
[11]F. A. Mazarbhuiya, M. Y. AlZahrani, and L. Georgieva, ‘‘Anomaly detection using agglomerative hierarchical clustering algorithm,’’ in Proc. Int. Conf. Inf. Sci. Appl., 2019, pp. 475–484.
[12] H. Liu, J. Li, Y. Wu, and Y. Fu, ‘‘Clustering with outlier removal,’’ IEEE Trans. Knowl. Data Eng., vol. 33, no. 6, pp. 2369–2379, Jun. 2021.
[13] Z. Li, Z. Xiang, W. Gong, and H. Wang, ‘‘Unified model for collective and point anomaly detection using stacked temporal convolution networks,’’ Appl. Intell., vol. 52, no. 3, pp. 1–14, 2021.
[14] C. Miao, Q. Dong, M. Hao, C. Wang, and J. Cao, ‘‘Magnetic anomaly detection based on fast convergence wavelet artificial neural network in the aeromagnetic field,’’ Measurement, vol. 176, May 2021, Art. no. 109097.

[15] H. Wang, W. Yu, J. You, R. Ma, W. Wang, and B. Li, ‘‘A unified framework for anomaly detection of satellite images based on well-designed features and an artificial neural network,’’ Remote Sens., vol. 13, no. 8, p. 1506, Apr. 2021.
[16] P. Ayegba, J. Ayoola, E. Asani, and A. Okeyinka, ‘‘A comparative study of minimal spanning tree algorithms,’’ in Proc. Int. Conf. Math., Comput. Eng. Comput. Sci. (ICMCECS), Mar. 2020, pp. 1–4.
[17] D. Lee, M.-H. Yang and S. Oh, "Fast and accurate head pose estimation via random projection forests", Proc. IEEE Int. Conf. Comput. Vis., pp. 1958-1966, 2015.
[18] R. Hartley, "Optimised kd-trees for fast image descriptor matching", Proc. IEEE Conf. Comput. Vis. Pattern Recognit., pp. 1-8, 2008
[19] Md Tahmid Rahman Laskar Information Retrieval & Knowledge Management Research Lab, York University, Canada “Extending isolation forest for anomaly detection in big data via k-means”
[20] T. Cannings and R. Samworth, "Random-projection ensemble classification", J. Roy. Statistical Soc. Series B, vol. 79, no. 4, pp. 959-1035, 2017
[21] N. Abe, B. Zadrozny, and J. Langford. Outlier detection by active learning. In Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 504–509. ACM Press, 2006.
[22] A. Asuncion and D. Newman. UCI machine learning repository, 2007.
[23] S. D. Bay and M. Schwabacher. Mining distance-based outliers in near linear time with randomization and a simple pruning rule. In Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, pages 29–38. ACM Press, 2003.
[24] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. LOF: identifying density-based local outliers. ACM SIGMOD Record, 29(2):93–104, 2000.
[25] Z. He, X. Xu, and S. Deng. Discovering cluster-based local outliers. Pattern Recogn. Lett., 24(9-10):1641–1650, 2003.
[26] E. M. Knorr and R. T. Ng. Algorithms for mining distance based outliers in large datasets. In VLDB ’98: Proceedings of the 24rd International Conference on Very Large Data Bases, pages 392–403, San Francisco, CA, USA, 1998. Morgan Kaufmann.
[27] D. E. Knuth. Art of Computer Programming, Volume 3:Sorting and Searching (2nd Edition). Addison-Wesley Professional, April 1998.
[28] R. B. Murphy. On Tests for Outlying Observations. PhDthesis, Princeton University, 1951.
[29] B. R. Preiss. Data Structures and Algorithms with Object Oriented Design Patterns in Java. Wiley, 1999.
[30] D. M. Rocke and D. L. Woodruff. Identification of outliers in multivariate data. Journal of the American Statistical Association, 91(435):1047–1061, 1996.
[31] P. J. Rousseeuw and K. V. Driessen. A fast algorithm for the minimum covariance determinant estimator Technometrics,
[32] T. Shi and S. Horvath. Unsupervised learning with random forest predictors. Journal of Computational and Graphical Statistics, 15(1):118–138, March 2006.
[33] M. Wu and C. Jermaine. Outlier detection by sampling with accuracy guarantees. In Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 767–772, New York, NY, USA, 2006. ACM.
[34] K. Yamanishi, J.-I. Takeuchi, G. Williams, and P. Milne. Online unsupervised outlier detection using finite mixtures with discounting learning algorithms. In Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining, pages 320–324. ACM Press, 2000.
[35] Shi Ying, Bingming Wang, Lu Wang, Qingshan Li, Yishi Zhao, Jianga Shang, Hao Huang, Guoli Cheng, Zhe Yang, and JiangyiGeng. An improved knn-based efficient log anomaly detection method with automatically labeled samples. ACM Transactions on Knowledge Discovery from Data (TKDD), 15(3):1–22, 2021.