Group Project – COMP 4710 (Group 17)

***Abstract*: This paper describes the working of the combination of two existing Machine learning algorithms to produce a better algorithm to provide better efficiency on the test dataset.**

**In the present day and age, we have an abundance of data all around as everything is slowly moving to the internet. Along with this movement onto the internet, a vast amount of data is constantly being generated and collected. But we can’t take all that data in consideration for our research due to data quality. Because of this, we require good algorithms to generate valuable insights from all this data. In order to accomplish this, techniques like KNN and Random Forest have been utilized to figure out some implicit data that may not be obvious at first glance.**

***Keywords: KNN, Random Forest, Data mining, Predictions, Big Data, Machine Learning***

INTRODUCTION

In this age, big data applications are increasingly becoming the main focus of attention because of the enormous increment of data generation and storage that has taken place in the last years [1]. Big data, a key component of economic growth, is expanding at an incredible rate. These big data are implicit, previously unknown, and potentially useful information [2]. By utilizing big data, economic growth can be monitored efficiently. The retail and eCommerce sector requires special analysis to get a better idea of customer behaviour and their purchasing decisions.

Our algorithms utilize data from items that are for sale and consider factors that may affect the sale of a particular item. In general, our algorithm will help the stakeholders to make predictions and smart market decisions for their businesses. As an example of an application of our algorithm, we have decided to work with a dataset of used vehicles as an attempt to make predictions about the usability class of the dataset.

Our basic motivation was to combine Random Forests and concepts of KNN to produce a hybrid algorithm that acts as an improvement to Random Forests. One of the very fundamental algorithms in Machine learning is Decision trees which work well for almost all types of datasets. Random forest is an improvement of decision trees that boosts the accuracy of decision trees by using the concepts of bootstrapping, random subspace, feature selection and bagging. KNN is another algorithm used in machine learning which makes classifications and solves regression problems by finding the closest neighbors to the given test subject based on the distance calculated using a sort of naive mathematical formula. Random forest almost always outperforms KNN due to more complexity in its work.

Our idea is to insert a KNN like algorithm at the leaf nodes of Random Forest to improve the accuracy rather than relying totally on pure majority voting.

The improved algorithm has almost the usability as the standard random forests and could be used in almost all cases where random forests could be used.

The final algorithm was tested on our chosen dataset and the accuracy was also compared to random forests by plotting a graph.

As the number of internet users grows every day, each user generates valuable data that can be used for a wide variety of purposes. Namely, companies find the data to be useful for targeted ads to boost sales. Dealing with a person’s interests is an effective way to encourage consumers to buy certain products. But another important thing to consider when it comes to sales and peoples’ choices are the specific factors related to the product. A person may be interested in computers, for example. But what type of computers do they like the most? What type of processor? How much memory? What screen size and resolution? These are all important questions to ask when analyzing users’ data. Perhaps we can find some

valuable information regarding the products themselves, rather than working only with the users’ side of things. Combining the two ideas would result in a more efficient business model. Products predicted to sell fewer units can be produced in smaller numbers while higher demand products can be produced in greater numbers. This is good for everyone involved; less waste, greater availability of in-demand products, etc.

BACKGROUND & RELATED WORK

In this section, we investigate the background and related works on three supervised learning algorithms used to solve classification and regression problems. More specifically, we examined:

• Decision Tree

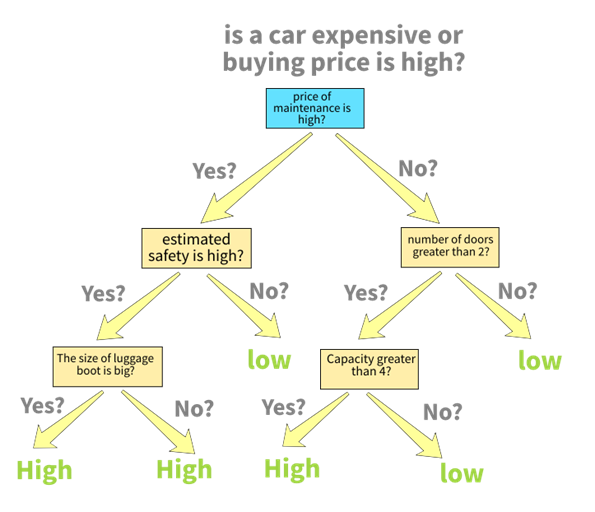
• Random Forest

• K-nearest neighbor (KNN)

A. Decision Tree

A decision tree is a tree where each node represents a test of an attribute, and a leaf node provides classification. Starting from the root, the test example is classified at each node and sorted down to the suitable branch before it reaches the leaf node which delivers classification.

Here’s an illustration of a decision tree (using our given dataset):



**B. Random Forest**

Random forest is a composition of several distinct decision tree classifiers.[3] The combination of each output generated from each decision tree is used to calculate the final result through a voting system.

Here’s an illustration of a random forest:

Diagram

Description automatically generated

The output for test samples is generated according to certain rules. Following steps are involved while generating classification results for the test set using a random forest algorithm.

1. Use Bootstrapping method, that is, a powerful statistical method for estimating a quantity from a data sample. A Bootstrapped training subset is generated by selecting random samples from the training set.
2. From the training subset, several features are randomly selected as feature set for splitting each node of the decision tree.
3. Steps (1) and (2) are repeated until a certain number of the decision trees are generated to form the random forest.
4. The sample from the test set are tested on a random forest and outputs are generated from each decision tree. After calculating the outputs from each decision tree, the results are generated by majority voting for test set sample classification.
5. Step (4) is repeated till all samples from the test set are classified.

**C. K-nearest neighbor (KNN)**

KNN is used to classify unlabeled observations by allocating them to the class of the maximum similar labeled examples [4]. By default, KNN classifiers use the Euclidean metric to determine the variations between examples represented as vector inputs. But due to high dimensionality in the dataset, we are using Manhattan Distance.

where

is the number of features.

are the data points.

is from 1 to n.

The class label is assigned based on the majority vote of its k nearest neighbors. [5]

where

is a test example.

is one of its

k nearest neighbors in the training set,

indicates that whether

belongs to feature

**.**  Above equation (2) return the feature having most members in the k nearest neighbors.

MAINBODY

Random forests and other ensemble techniques work well on most datasets, but their accuracy could be improved further. The shortcomings of random forests include making decisions based on the majority voting whereas the shortcomings of KNN include not taking the feature importance into account. We chose random forest because it is one of the best performing algorithms in the machine learning world and KNN was chosen due to its naïve approach. By improving the naïve approach of KNN and inserting it into random forests to improve random forests we were able to improve an already well-performing algorithm. We did not choose other machine learning algorithms like decision trees because the KNN improvement was hard to put in them and could not get any improvements that were efficient enough to prove the enhancements.

The KNN algorithm is a type of supervised machine learning algorithm that can be used for both classification and regression predictive problems [6].

The main advantage of K-Nearest Neighbor algorithms is their simplicity, robustness, and validity towards noisy data. Due to various pitfalls of KNN like high computational cost, costly memory requirements, and uncertainty deciding the value of K, modified versions of KNN techniques are used. The accuracy of KNN depends on the input parameter K and distance metrics used [7].

Moreover, according to Larijani, “KNN is not bringing time efficiency as it is reconsidering all of the instances every time, the algorithm is at risk of dimensionality and easily affected by the inconsistent and similar characteristics” [7]. Besides, A wrong choice of the distance or the value of k degrades the performance [8].

Also, by default, KNN assumes that the features in training samples have equal importance. Which can create a biased result as one feature can have more importance over others when calculating the distance among the neighbors’ nodes [9].

In general, feature importance is crucial, especially in test samples with many features and variables. It will remove unimportant variables and improve the accuracy as well as the performance of classification. [10]

The basic formula used to calculate the distance to a specific neighbor in a KNN algorithm is from equation (1) with respect to ith feature range.

where

is the range of feature.

There is a total of n features in the dataset which will help us in making the prediction. **x**i is the value of the ith feature for the neighbor and **yi** is the value of the ith feature for the test subject.

The drawback with this formula is that it doesn't take into account the feature importance, i.e., one feature might be more important than the other when calculating the distance among the neighbour nodes. After adding feature importance to equation (3). The improved Manhattan equation formula looks as the following:

where

is the feature importance

Our whole KNN algorithm works this way. First, it calculates the feature ranges which are divided by the difference between the neighbor and testing node to come up with a value between 0 and 1. Then it computes the distance to the neighbors using our improved Manhattan distance formula. Once the distances to the neighbors within the leaf node of the decision tree have been calculated, they are stored and sorted in ascending order (lowest to highest) in a dictionary. The following code snippet will show how we are implementing this in our code:

Text, letter

Description automatically generated

Text, letter

Description automatically generated

**Feature importance function**:

**Chi-Square Test:**

Chi-square test is one of the ways by which we can compute nominal variables, provide information about any found offset between the categories along with the important observed difference in the whole competition [11]. “The chi-square (X2) is a test of significance for categorical variables” [12]. The chi-square can be used as a goodness-of-fit test, in univariate analysis, or as a test of independence, in the bivariate analysis [12]. Where the goodness-of-fit tests is a preceding towards leaving the alternative model unspecified prudently [13]. We use a specific algorithm to calculate the importance of features we use in our improved version of KNN. Since all the columns in our dataset are categorical and so is our predictor variable, we use a chi-square test to calculate the chi scores of the independent columns with regards to the dependent columns. The feature with the lowest chi score is considered to impact the dependant variable the most, and we use a weighted feature impotence score in the improved version of KNN in the following manner:

where

is the importance ratio of

feature,

is the chi-score of

feature.

is from 1 to

Another important change is made to the default KNN algorithm is after distances to the neighbor nodes have been calculated and stored, instead of selecting a fixed number of neighbors from the top sorted distances, we use a 50% threshold to choose the top 50% neighbors to make the majority voting from. This way the lower 50% of the neighbors which are not suitable for making the decision are discarded and we start getting an improvement in the accuracy when we implement the last step of our algorithm which is essentially taking the majority votes from the eligible neighbors.

The following code snippet shows the code written for the KNN-like extension inserted into the code of random forests. This is the extended part of the code snippet which has been pasted before. The above description is what explains the following code:

Text

Description automatically generated

Diagram below shows the position where KNN is inserted to one of Decision trees of Random Forest.

Diagram

Description automatically generated

EXPERIMENTS

We have examined RegularRandomForest with ImrovedRandomForest. In our experiment 2 data sets are used. These set are available in the UCI repository website. To show variation between both set, different feature and data size were chosen.

Type of Dataset:

|  |  |  |
| --- | --- | --- |
| Name | Features | # of Entries |
| Heart Disease | 13(default), 22 (after transformation) | 303 (default)  297 (after transformation) |
| Car Evaluation | 7(default), 6 (after transformation) | 386(both) |

Before testing straight on algorithm, both datasets get transformed by removing missing entries depending on their quantities, removing dependencies on categories of different features and by change their data types.

|  |  |
| --- | --- |
| ***Name*** | ***Description*** |
| price | Overall price |
| buying | Buying price |
| maint | Charges for maintenance |
| tech | Technical characteristics |
| comfort | Comfort level of the car |
| doors | Number of doors |
| persons | Capacity in terms of persons to carry |
| lug\_boot | The size of luggage boot |
| safety | Estimated safety of the car |

**Car-Evaluation Dataset:**

The column to be predicted is Car usability

The columns which we used to predict car usability: buying, maintenance, doors, persons, lug\_boot, safety

All the specific values of the columns were arranged in an ordinal order and an ordinal encoder was used to convert those into a numerical format which is usable by a standard machine learning algorithm.

**Heart-Disease Dataset:**

The column to be predicted is: num (have HD or not) usability

The columns which we used to predict if patient have HD or not: age, sex, cp, trestbps, chol, fbs, restecg, thalach, exang, Oldpeak, Slope, Ca, thal

The datasets contain following attributes:

|  |  |
| --- | --- |
| ***Name*** | ***Description*** |
| age | Age of Patient |
| sex | 0-Female/ 1-Male |
| cp | 1-typical angina/ 2-atypical angina/ 3-non-anginal pain/ 4- asymptomatic |
| trestbps | Resting blood pressure |
| chol | Cholesterol |
| fbs | Fasting blood sugar |
| restecg | Resting electrocardiographic results |
| thalach | Maximum heart rate achieved |
| exang | Exercise induced angina |
| Oldpeak | Depression induced by exercise |
| Slope | 1-unsloping/ 2-flat/ 3-downsloping |
| Ca | No. of majorvessels color by fluroscopy |
| Thal | Thalium heart scan |
| num | 1-Heart Disease / 0 -No Heart Disease |

Experiment Result:

The experiment results of datasets are summarized in following tables.

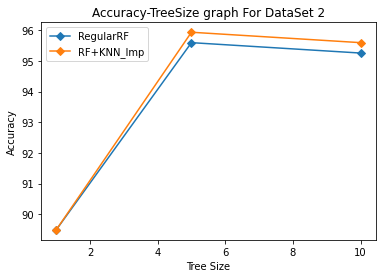
Following Diagrams shows the accuracy in prediction of predicting variable by the improved and the regual algorithm respect to number of tree for the Car evaluation and heart disease Data set.

Diagram 1: For Car Evaluation

Chart, line chart

Description automatically generated

Diagram 2: For Heart Disease



From the diagrams above we observed a inclination in accuracy respect to tree size.

A detailed version of table was drawn from results to show accuracies between both datasets in actual percentages.

|  |  |  |
| --- | --- | --- |
| Algorithm  ------------------  Number of trees used | Random Forests (improved with KNN) | Random Forests |
| One tree (Dataset-1) | 89.45 % | 87.594 % |
| One tree (Dataset-2) | 89.492% | 89.492% |
| Five trees (Dataset-1) | 95.48 % | 94.26 % |
| Five (Dataset-2) | 95.932% | 95.593% |
| Ten trees (Dataset-1) | 94.203 % | 93.39 % |
| Ten trees (Dataset-2) | 85.592% | 85.254% |

From the above table, we saw a big change in percentage of accuracy in Dataset-1 comparing to Dataset-2. On average, there was 1.29% and 0.677% increase of accuracy in Dataset-1 and Dataset-2. The percentage of Dataset-1 is higher than Dataset-2 because the ratio of feature importance was the main influencer here. The variation in feature importance for each feature in Dataset-1 was high as compared to Dataset-2.

Following table show the feature importance for each feature in both datasets:

**For Dataset-1: (size:6)**

[ 80.59031849, 60.0199957, 4.08142355, 88.80713295, 17.06154878, 129.89756176]

**For Dataset-2: (size:22)**

[4.78804980e+00, 1.03073322e+01, 8.52242936e+00, 1.78669763e-01, 4.76711002e+01, 1.81542998e+01, 3.78629445e+01, 1.90874393e+01,

3.75827527e+01, 1.50316791e-01, 1.03562711e+01, 5.21884250e+00,

1.43868297e+01, 1.87212964e-04, 3.17355471e-01, 6.32277193e-03,

8.57947733e+00, 6.87815171e+00, 7.00805406e-01, 1.03626374e+02,

1.13736264e+01]

Above Statical data show that the difference between the minimum and maximum important feature was higher in Dataset-1 as compared to Dataset-2.

In general, improved in algorithm shows a notable increase in accuracy with respect to number of trees.

From the discussion above we conclude that our algorithm is quite fit for use over the regular random forests. Big changes can be observed respect to feature importance.

CONCLUSION

This project was a trial experiment on improving [an attempt to improve] the accuracy of an already well-established machine learning algorithm. We built a table showing the trade-off between accuracies and latencies which shows us that using too many trees leads to accuracy not getting improved to a significant extent, which could be explained by the fact that when random forests use a lot of trees, we get such a huge effect of majority voting provided by the variance in number of trees that our small addition of a KNN like algorithm in the leaf nodes is offset by it.

Therefore, the limitation of our KNN like algorithm inserted in the leaf nodes is that the improvement provided by it is easily offset by the large number of trees when random forests start using a huge number of trees. This basically means that when in random forests the number of trees grows beyond 20, we start getting very huge increases in accuracy and at that point, the improvement provided by our algorithm starts becoming marginal. One possible way of overcoming this problem would be to use a more advanced algorithm in the leaf nodes or to have two or three different types of algorithms in the leaf nodes and use variations of them in the different number of trees. This way we can make sure not all trees use the same algorithm, which might start giving us

significant increases even in the case of a large number of trees.

REFERENCES

[1] S. del Río, V. López, J. M. Benítez, and F. Herrera, “On the use of mapreduce for imbalanced big data using Random Forest,” Information Sciences, vol. 285, pp. 112–137, 2014.

[2] K. E. Dierckens, A. B. Harrison, C. K. Leung and A. V. Pind, "A Data Science and Engineering Solution for Fast K-Means Clustering of Big Data," 2017 IEEE Trustcom/BigDataSE/ICESS, 2017, pp. 925-932, doi: 10.1109/Trustcom/BigDataSE/ICESS.2017.332.

[3] Z. Bingzhen, Q. Xiaoming, Y. Hemeng and Z. Zhubo, "A Random Forest Classification Model for Transmission Line Image Processing," 2020 15th International Conference on Computer Science & Education (ICCSE), 2020, pp. 613-617, doi: 10.1109/ICCSE49874.2020.9201900.

[4] Zhang Z. (2016). Introduction to machine learning: k-nearest neighbors. Annals of translational medicine, 4(11), 218. https://doi.org/10.21037/atm.2016.03.37

[5] S. Sun and R. Huang, "An adaptive k-nearest neighbor algorithm," 2010 Seventh International Conference on Fuzzy Systems and Knowledge Discovery, 2010, pp. 91-94, doi: 10.1109/FSKD.2010.5569740

[6] Xiong, L. and Yao, Y. (2021). Study on an adaptive thermal comfort model with K-nearest-neighbors (KNN) algorithm. Building and Environment, 202, p.108026.

[7] O. Borgohain, M. Dasgupta, P. Kumar, and G. Talukdar, “Performance Analysis of Nearest Neighbor, K-Nearest Neighbor and Weighted K-Nearest Neighbor for the Classification of Alzheimer Disease,” Springer Link, 28-Nov-2020. [Online]. Available: https://link-springer-com.uml.idm.oclc.org/chapter/10.1007/978-981-15-7394-1\_28. [Accessed: 08-Dec-2021]

[8] Larijani, M. R. et al. (2019) Evaluation of image processing technique in identifying rice blast disease in field conditions based on KNN algorithm improvement by K‐means. Food science & nutrition. [Online] 7 (12), 3922–3930.

[9] Manning C. D. and Schutze H., 1999. “Foundations of Statistical Natural Language Processing” [M]. Cambridge: MIT Press.

[10] Chen, RC., Dewi, C., Huang, SW. et al. Selecting critical features for data classification based on machine learning methods. J Big Data 7, 52 (2020). https://doi.org/10.1186/s40537-020-00327-4

[11] M. L. Mchugh, “The Chi-square test of independence,” Biochemia Medica, pp. 143–149, 2013.

[12] Lavrakas, Paul. “Encyclopedia of Survey Research Methods.” 2008, https://doi.org/10.4135/9781412963947.

[13] J. Hidalgo and P. Zaffaroni, “A goodness-of-fit test for ARCH(∞) models,” Journal of Econometrics, vol. 141, no. 2, pp. 835–875, 2007.