Assignment Title : Decision Tree And K-Nearest Neighbor

CSE-0408 Summer 2021

Tahsin Shovon(ID:UG02-44-17-023)
Department of Computer Science and Engineering
State University of Bangladesh (SUB)
Dhaka, Bangladesh

***Abstract*—A decision tree is a decision support tool that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements.**

***Index Terms*—Explaining About decision Tree,decision tree classifiers And I will Try An Problem Solving With Decision Tree Using Python language.**

I. Introduction

Decision tree as the name suggests it is a flow like a tree structure that works on the principle of conditions. It is efficient and has strong algorithms used for predictive analysis. It has mainly attributes that include internal nodes, branches and a terminal node.

Every internal node holds a "test" on an attribute, branches hold the conclusion of the test and every leaf node means the class label. This is the most used algorithm when it comes to supervised learning techniques.

It is used for both classifications as well as regression. It is often termed as "CART" that means Classification and Regression Tree. Tree algorithms are always preferred due to stability and reliability.

II. THE PURPOSE OF CHOOSING THIS TOPIC:

A decision tree is a graphical representation of all possible solutions to a decision based on certain conditions. On each step or node of a decision tree, used for classification, we try to form a condition on the features to separate all the labels or classes contained in the dataset to the fullest purity

III. Literature Review

Machine Learning, Tom Mitchell, McGraw Hill, 1997. In the next post we will be discussing about ID3 algorithm for the construction of Decision tree given by J. R. Quinlan

IV. PROPOSED METHODOLOGY

How to Construction of An Decision Tree : A tree can be "learned" by splitting the source set into subsets based on an attribute value test. This process is repeated on each derived subset in a recursive manner called recursive partitioning. The recursion is completed when the subset at a node all has the same value of the target variable,or when splitting no longer adds value to the predictions. The construction of decision tree classifier does not require any domain knowledge or parameter setting, and therefore is appropriate for exploratory knowledge discovery. Decision trees can handle high dimensional data. In general decision tree classifier has good accuracy. Decision tree induction is a typical inductive approach to learn knowledge on classification. How to An Decision Tree Representation : Decision trees classify instances by sorting them down the tree from the root to some leaf node, which provides the classification of the instance. An instance is classified by starting at the root node of the tree, testing the attribute specified by this node, then moving down the tree branch corresponding to the value of the attribute as shown in the above figure. This process is then repeated for the subtree rooted at the new node. The decision tree in above figure classifies a particular morning according to whether it is suitable for playing tennis and returning the classification associated with the particular leaf.(in this case Yes or No). For example, the instance (Outlook = Rain, Temperature = Hot, Humidity = High, Wind = Strong)

V. TYPES OF DECISION TREES

Types of decision trees are based on the type of target variable we have. It can be of two types: 1.Categorical Variable Decision Tree: Decision Tree which has a categorical target variable then it called a Categorical variable decision tree. 2. Continuous Variable Decision Tree: Decision Tree has a continuous target variable then it is called Continuous Variable Decision Tree.

VI. IMPORTANT TERMINOLOGY RELATED TO DECISION TREES

1.Root Node: It represents the entire population or sample and this further gets divided into two or more homogeneous sets. 2.Splitting: It is a process of dividing a node into two or more sub-nodes. 3. Decision Node: When a sub-node splits into further subnodes, then it is called the decision node 4.Leaf / Terminal Node: Nodes do not split is called Leaf or Terminal node. 5. Pruning: When we remove sub-nodes of a decision node, this process is called pruning. You can say the opposite process of splitting. 6.Branch / Sub-Tree: A subsection of the entire tree is called branch or sub-tree 7.Parent and Child

Node: A node, which is divided into subnodes is called a parent node of sub-nodes whereas sub-nodes are the child of a parent node.

VII. ADVANTAGE OF DECISION TREE

1. Clear Visualization: The algorithm is simple to understand, interpret and visualize as the idea is mostly used in our daily lives. Output of a Decision Tree can be easily interpreted by humans. 2. Simple and easy to understand: Decision Tree looks like simple if-else statements which are very easy to understand. 3. Decision Tree can be used for both classification and regression problems. 4. Decision Tree can handle both continuous and categorical variables. 5. No feature scaling required: No feature scaling (standardization and normalization) required in case of Decision Tree as it uses rule based approach instead of distance calculation. 6. Handles non-linear parameters efficiently: Non linear parameters don't affect the performance of a Decision Tree unlike curve based algorithms. So, if there is high non-linearity between the independent variables, Decision Trees may outperform as compared to other curve based algorithms. 7. Decision Tree can automatically handle missing values. 8. Decision Tree is usually robust to outliers and can handle them automatically. 9. Less Training Period: Training period is less as compared to Random Forest because it generates only one tree unlike forest of trees in the Random Forest.

VIII. DISADVANTAGE OF DECISION TREE

1. Overfitting: This is the main problem of the Decision Tree. It generally leads to overfitting of the data which ultimately leads to wrong predictions. In order to fit the data (even noisy data), it keeps generating new nodes and ultimately the tree becomes too complex to interpret. In this way, it loses its generalization capabilities. It performs very well on the trained data but starts making a lot of mistakes on the unseen data.

I have written a detailed article on Overfitting here.

2. High variance: As mentioned in point 1, Decision Tree generally leads to the overfitting of data. Due to the overfitting, there are very high chances of high variance in the output which leads to many errors in the final estimation and shows high inaccuracy in the results. In order to achieve zero bias (overfitting), it leads to high variance.

3. Unstable: Adding a new data point can lead to re-generation of the overall tree and all nodes need to be recalculated and recreated.

4. Affected by noise: Little bit of noise can make it unstable which leads to wrong predictions.

5. Not suitable for large datasets: If data size is large, then one single tree may grow complex and lead to overfitting. So in this case, we should use Random Forest instead of a single Decision Tree

IX. ALGORITHM OF DECISION TREE

import numpy as np import matplotlib.pyplot as plt import pandas as pd import math import copy dataset = pd.readcsv(0Book1.csv0) X = dataset.iloc[:, 1:].values print(X) attribute = ['outlook', 'temp', 'humidity', 'wind'] class Node(object): def init(self): self.value = None self.decision = None self.childs = None def findEntropy(data, rows): yes = 0 no = 0 ans = -1 idx = len(data[0]) - 1 entropy = 0 for i in rows: if data[i][idx] == 'Yes': yes = yes + 1 else: no = no + 1 x = yes/(yes+no) y = no/(yes+no) if x != 0 and y != 0: entropy = -1 * (x*math.log2(x) + y*math.log2(y)) if x == 1: ans = 1 if y == 1: ans = 0 return entropy, ans def findMaxGain(data, rows, columns): maxGain = 0 retidx = -1 entropy, ans = findEntropy(data, rows) if entropy == 0: """"if ans == 1: print("Yes") else: print("No")""" return maxGain, retidx, ans for j in columns: mydict = idx = j for i in rows: key = data[i][idx] if key not in mydict: mydict[key] = 1 else: mydict[key] = mydict[key] + 1 gain = entropy print(mydict) for key in mydict: yes = 0 no = 0 for k in rows: if data[k][j] == key: if data[k][-1] == 'Yes': yes = yes + 1 else: no = no + 1 print(yes, no) x = yes/(yes+no) y = no/(yes+no) print(x, y) if x != 0 and y != 0: gain += (mydict[key] * (x*math.log2(x) + y*math.log2(y)))/14 print(gain) if gain ¿ maxGain: print("hello") maxGain = gain retidx = j return maxGain, retidx, ans def buildTree(data, rows, columns): maxGain, idx, ans = findMaxGain(X, rows, columns) root = Node() root.childs = [] print(maxGain) if maxGain == 0: if ans == 1: root.value = 'Yes' else: root.value = 'No' return root root.value = attribute[idx] mydict = for i in rows: key = data[i][idx] if key not in mydict: mydict[key] = 1 else: mydict[key] += 1 newcolumns = copy.deepcopy(columns) newcolumns.remove(idx) for key in mydict: newrows = [] for i in rows: if data[i][idx] == key: newrows.append(i) print(newrows) temp = buildTree(data, newrows, newcolumns) temp.decision = key root.childs.append(temp) return root def traverse(root): print(root.decision) print(root.value) n = len(root.childs) if n ¿ 0: for i in range(0, n): traverse(root.childs[i]) def calculate(): rows = [i for i in range(0, 14)] columns = [i for i in range(0, 4)] root = buildTree(X, rows, columns) root.decision = 'Start' traverse(root) calculate()

X. CONCLUSION

Conclusion. Decision trees assist analysts in evaluating upcoming choices. The tree creates a visual representation of all possible outcomes, rewards and follow-up decisions in one document.

***Abstract*—A decision tree is a decision support tool that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements.**

—KNN (k-nearest neighbor) is an extensively used classification algorithm owing to its simplicity, ease of implementation and effectiveness. It is one of the top ten data mining algorithms, has been widely applied in various fields. KNN has few shortcomings affecting its accuracy of classification. It has large memory requirements as well as high time complexity. Several techniques have been proposed to improve these shortcomings in literature. In this paper, we have first reviewed some improvements made in KNN algorithm.

***Index Terms*—Explaining About decision Tree,decision tree classifiers And I will Try An Problem Solving With Decision Tree Using Python language.**

K-Nearest Neighbors is one of the most basic yet essential classification algorithms in Machine Learning. It belongs to the supervised learning domain and finds intense application in pattern recognition, data mining and intrusion detection. And I will Try An Problem Solving With Decision Tree Using Python language-matplotlib.pyplot,pandas as pd, numpy as np.

XI. INTRODUCTION

K-Nearest Neighbors algorithm (or KNN) is one of the most used learning algorithms due to its simplicity. So what is it? KNN is a lazy learning, non-parametric algorithm. It uses data with several classes to predict the classification of the new sample point. KNN is non-parametric since it doesn't make any assumptions on the data being studied, i.e., the model is distributed from the data. What does it mean to say KNN is a lazy algorithm? It means it doesn't use the training data points to make any generalisation. Which implies: You expect little to no explicit training phase, The training phase is pretty fast, KNN keeps all the training data since they are needed during the testing phase. Most data does not obey the typical theoretical assumptions, like when we consider a model like linear regression, which makes KNN crucial when studying data with little or no prior knowledge.

XII. PURPOSE OF CHOOSING THIS TOPIC

KNN works by finding the distances between a query and all the examples in the data, selecting the specified number examples (K) closest to the query, then votes for the most frequent label (in the case of classification) or averages the labels (in the case of regression).

XIII. LITERATURE REVIEW

KNN was born out of research done for the armed forces. Fix and Hodge – two officers of USAF School of Aviation Medicine – wrote a technical report in 1951 introducing the KNN algorithm.

XIV. USES OF KNN

KNN can be used in both regression and classification predictive problems. However, when it comes to industrial problems, it's mostly used in classification since it fairs across all parameters evaluated when determining the usability of a technique 1. Prediction Power 2.Calculation Time 3. Ease to Interpret the Output KNN algorithm fairs across all parameters of considerations. But mostly, it is used due to its ease of interpretation and low calculation time.

XV. WORKING OF KNN

The k-nearest neighbor algorithm stores all the available data and classifies a new data point based on the similarity measure (e.g., distance functions). This means when new data appears. Then it can be easily classified into a well-suited category by using K- NN algorithm. Suppose there are two classes, i.e., Class A and Class B, and we have a new unknown data point "?", so this data point will lie in which of these classes. To solve this problem, we need a K-NN algorithm. With the help of K-NN, we can easily identify the class of a particular dataset. The data point is classified by a majority vote of its neighbors, with the data point being assigned to the class most common amongst its K nearest neighbors measured by a distance function. Consider the below diagram: Here, we can see that if k = 3, then based on the distance function used, the nearest three neighbors of the data point is found and based on the majority votes of its neighbors, the data point is classified into a class. In the case of k = 3, for the above diagram, it's Class B. Similarly, when k = 7, for the above diagram, based on the majority votes of its neighbors, the data point is classified to Class A.

XVI. ADVANTAGE OF KNN

Some Advantages of KNN: 1.Quick calculation time 2.Simple algorithm – to interpret 3.Versatile – useful for regression and classification 4.High accuracy – you do not need to compare with bettersupervised learning models 5.No assumptions about data – no need to make additional assumptions, tune several parameters, or build a model. This makes it crucial in nonlinear data case.

XVII. DISADVANTAGE OF KNN

Some Disadvantages of KNN 1.Accuracy depends on the quality of the data 2.With large data, the prediction stage might be slow 3.Sensitive to the scale of the data and irrelevant features 4. Require high memory – need to store all of the training data 5.Given that it stores all of the training, it can be computationally expensive

XVIII. ALGORITHM AND SOURCE CODE

The Crisp K-NN Algorithm Let W = xx, x2, · · ·, xn be a set of n labeled samples.Thealgorithm is as follows: BEGIN Input y9 of unknown classification. Set , K n. Initialize / = 1. DO UNTIL (AT-nearest neighbors found) Compute distance from y to xt. IF (/ ¡ K) THEN Include xt in the set of Af-nearest neighbors ELSE IF (x; is closer to y than any previous nearest neighbor) THEN Delete farthest in the set of nearestneighborsIncludex,inthesetofAfnearestneighbors. END IF Increment /. BEGIN Input x, of unknown classification. Set K, 1 ¡ K ¡ n. Initialize / = 1. DO UNTIL (AT-nearest neighbors to x found) Compute distance from x to x. I F (i ¡ K) THEN Include JC,· in the set of AT-nearest neighbors ELSE I F (xt closer to x than any previous nearest neigh-bor) T H E N Delete the farthest of the -nearest neighbors Include xt in the set of -nearest neighbors. E N D I F E N D DO UNTIL Initialize i = 1. DO UNTIL (x assigned membership in all classes) Compute W,(JC) using (1). Increment i. END DO UNTIL.

CODE :-

Import necessary modules from sklearn.neighbors import KNeighborsClassifier from sklearn.modelselectionimporttraintestsplit from sklearn.datasets import loadiris import numpy as np

import matplotlib.pyplot as plt irisData = loadiris() Create feature and target arrays X = irisData.data y = irisData.target Split into training and test set Xtrain, Xtest, ytrain, ytest = traintestsplit(X, y, testsize = 0.2, randomstate = 42) neighbors = np.arange(1, 9) trainaccuracy = np.empty(len(neighbors)) testaccuracy = np.empty(len(neighbors)) Loop over K values for i, k in enumerate(neighbors): knn = KNeighborsClassifier(nneighbors = k) knn.fit(Xtrain, ytrain) Compute training and test data accuracy trainaccuracy[i] = knn.score(Xtrain, ytrain) testaccuracy[i] = knn.score(Xtest, ytest) Generate plot plt.plot(neighbors, testaccuracy, label =0 T estingdatasetAccuracy0) plt.plot(neighbors, trainaccuracy, label =0 T rainingdatasetAccuracy0) plt.legend() plt.xlabel('nneighbors0) plt.ylabel('Accuracy') plt.show()

XIX. IMPLEMENTATION OF KNN

In the example shown above following steps are performed: 1.The k-nearest neighbor algorithm is imported from the scikit-learn package. 2.Create feature and target variables. 3.Split data into training and test data. 4.Generate a k-NN model using neighbors value. 5.Train or fit the data into the model. 6.Predict the future.

XX. CONCLUSION

A fuzzy AT-NN decision rule and a fuzzy prototype deci- ' sion rule have been developed along with three methods for assigning membership values to the sample sets. The fuzzy /iT-nearest neighbor and fuzzy nearest prototype algorithms developed and investigated in this report show useful results. In particular, concerning the fuzzy -nearest neighbor algorithm with fuzzy X-nearest neighbor initialization, the mem- bership assignments produced for classified samples tend to possess desirable qualities. That is, an incorrectly classified sam- ple will not have a membership in any class close to one while a correctly classified sample does possess a membership in the correct class close to one. The fuzzy nearest prototype classifier, while not producing error rates as low as the fuzzy nearest neighbor classifier, is computationally attractive and also pro- duces membership assignments that are desirable.

ACKNOWLEDGMENT

I would like to thank my honourable**Khan Md. Hasib Sir** for his time, generosity and critical insights into this project.

REFERENCES

[1] Y. Chen, S. Fay, and Q. Wang. Marketing implications of online consumer product reviews. Business Week, 7150:1–36, 2003

[2] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen. Classification and regression trees. CRC press, 1984.

[3] Cormen, Thomas H. "22.2 Breadth-first search". Introduction to algorithms. ISBN 978-81-203-4007-7. OCLC 1006880283. International Conference on Robotics and Automation, pp. 3310–3317 (1994)

[4] I. Vlachos and Z. Lin. Drivers of airline loyalty: Evidence from the business travelers in china. Transportation Research Part E: Logistics and Transportation Review, 71:1–17, 2014.

[5] Chakrabarti, P., Ghosh, S., Acharya, A., DeSarkar, S.: Heuristic search in restricted memory. Artificial Intelligence 47, 197–221 (1989)

[6] J. J. Liou and G.-H. Tzeng. A dominance-based rough set approach to customer behavior in the airline market. Information Sciences, 180(11):2230–2238, 2010

[7] Y. Zhao and Y. Zhang. Comparison of decision tree methods for finding active objects. Advances in Space Research, 41(12):1955–1959, 2008.

[8] Park, J.W.; Robertson, R.; Wu, C.L. The effect of airline service quality on passengers' behavioural intentions: A Korean case study. J. Air Transp. Manag. 2004, 10, 435–439. [CrossRef]

[9] Gilbert, D.; Wong, R.K. Passenger expectations and airline services: A Hong Kong based study. Tour. Manag. 2003, 24, 519–532. [

[10] Siering, M.; Deokar, A.V.; Janze, C. Disentangling consumer recommendations: Explaining and predicting airline recommendations based on online reviews. Decis. Support Syst. 2018, 107, 52–63

[11] Yayla-Kullu, H.M.; Tansitpong, P. A critical evaluation of US airlines' service quality performance: Lower costs vs. satisfied customers. J. Manag. Strategy 2013, 4, 1

[12] Yayla-Kullu, H.M.; Tansitpong, P. A critical evaluation of US airlines' service quality performance: Lower costs vs. satisfied customers. J. Manag. Strategy 2013, 4, 1

[13] Park, J.W.; Robertson, R.; Wu, C.L. The effects of individual dimensions of airline service quality: Finding

[14] Park, J.W.; Robertson, R.; Wu, C.L. The effects of individual dimensions of airline service quality: Finding

[15] Shuting, T.; Kang, B.; Kim, H.S. Understanding the food hygiene of cruise through the big data analytics using the web crawling and text mining. Culin. Sci. Hosp. Res. 2018, 24, 34–43. 41. Kim, H.S.; Yim, H.R. An exploratory study on the semantic network analysis of 'Culinary Science Hospitality Research' through the Google Scholar. Culin. Sci. Hosp. Res. 2018, 24, 1–10.

[16] Kim, J.J.; Kim, K.; Hwang, J. Self-enhancement driven first-class airline travelers' behavior: The moderating role of third-party certification. Sustainability 2019, 11, 3285. [CrossRef] 50. Lim, S.S.; Tkaczynski, A. Origin and money matter: The airline service quality expectations of international students. J. Hosp. Tour. Manag. 2017, 31, 244–252. [CrossRef] 51. Choi, H.Y.; Kwak, G.H.; Kim, H.S. A positioning study of national food: In perspective of Korean, American, Chinese food tourists. Culin

[17] Yayla-Kullu, H.M.; Tansitpong, P. A critical evaluation of US airlines' service quality performance: Lower costs vs. satisfied customers. J. Manag. Strategy 2013, 4, 1