A Decision Tree Machine Learning Approach for Optimizing 1P–3P Fault Detection Accuracy

Ashish Lonare*[1], Anjana Tripathi[2], Balram Yadav[3]

*[*1] M.Tech Student, School of Electrical & Electronics Engineering, Scope Global Skills University, Bhopal, M.P.,*

ashishlonare101@gmail.com[1]

[2]Assistant Professor , School of Electrical & Electronics Engineering, Scope Global Skills University, Bhopal, M.P.

anjana.tripathi2210@gmail.com[2]

[3] HOS, School of Electrical & Electronics Engineering, Scope Global Skills University, Bhopal, M.P.

balram@sgsuuniversity.ac.in[3]

ABSTRACT ***Protective relaying system is a versatile tool for protection of electric power systems. Because of the drastic changes occurring in power systems, the necessity for providing better protective relaying systems for transmission lines is essential. This work gives brief overview on different existing techniques for fault analysis and apart from the existing methodologies a novel fault detection scheme is presented in this work. It uses post fault current samples of all the phases.***

Key Words: DT-ML Algorithm, power system, protection system, faults

1. INTRODUCTION:

A decision tree classifier is a simple but powerful classification model. It is a supervised learning model having a flowchart like structure that creates a training model from the dataset. A decision tree is a tool for the detection of patterns, relationships, and knowledge from data. Decision trees give a straightforward visualization of data. These are easy to interpret for decision-making in real-time applications. To diagnose the disease of a patient, other classifiers like Support Vector Machine (SVM), Naïve Bays classifier, Artificial Neural Network (ANN) works as a black box where they only show the output but decision trees show decision using a simple flowchart like structure. Due to this expressive nature, decision trees are a popular decision-making tool in various applications. A decision tree is the best choice if the volume of data is increasing to large sizes, both in the number of instances and attributes. As data increases, decision tree learning becomes a complex and slow task.

There are different issues related to training with large data sets and the construction of decision trees. To handle these problems different methods have been proposed in the literature by different researchers. These approaches related to decision trees are summarized in this chapter. The emphasis is given on the approaches to create accurate and optimized decision trees. A decision tree is a supervised classifier having a tree structure made up of a set of nodes. These nodes are categorized into internal nodes and leaves. Internal nodes are also called decision nodes because attributes are tested at these nodes to take a decision and partition the data set. Leaves are used to represent predicted class or decision class. In a decision tree construction, splitting criteria is used for partitioning attributes.

2. DECISION TREE CONSTRUCTION ALGORITHM

Decision tree construction is a top-down recursive supervised classifier. It uses the divide and conquer approach to create a decision tree. It constructs a decision tree for a given training set $Ttrain$ consisting of a set of n instances. Each instance consists of values for a set of attributes and a class label. Let the k number of classes are denoted by $C1$, $C2$, $C3$, … … Ck . Initially, the frequency of class for available instances is calculated in the training set . If all instances are of the same class, then node n d with that class is created. However, if the training set $Ttrain$ contains instances of more than one class, evaluation of splits for each attribute is computed and the attribute satisfying splitting criteria is selected for the test at the decision node. The training set $Ttrain$ is then partitioned into s exclusive subsets $Ttrain$ 1 , $Ttrain$ 2 , $Ttrain$ 3 , … … $Ttrains$ depending on the tested attribute and then the same algorithm is recursively applied on every non-empty partition of data. The algorithm for decision tree construction is given below.

Construct Tree (Training Data $\boldsymbol{Ttrain}$) Partition($\boldsymbol{Ttrain}$) Partition(Data$\boldsymbol{Ttrain}$) If (all instances in $\boldsymbol{Ttrain}$ are of the same class Ci) then create a node nd with class Ci and return; else Evaluate Splits for each attribute A ; Use best split to partition $\boldsymbol{Ttrain}$ into $Ttrain$ 1 , $Ttrain$ 2 , $Ttrain$ 3 , … … $Ttrains$ for each $\boldsymbol{Ttrain\_i}$ Partition ($\boldsymbol{Ttrain\_i}$);

The major part in the construction of a decision tree is selecting the best splitting attribute. Hunt's Concept Learning System (CLS) [1] is a top-down non backtracking (i.e. greedy) and recursive partitioning strategy, which finds purer class value distribution in the subsequent partitions. Different decision tree learning algorithms follow the same composition of CLS but only differ in using various types of measures to select the best splitting attribute like distance-based measures, impurity based measures, and statistics based measures. These different types of measures are used to calculate the purity of class 0distribution [2]. Among these various types of measures, impurity measures are commonly used for constructing decision trees. For example, ID3 [3] uses information gain, CART [4] uses Gini Index and C4.5 [5, 6] uses Gain Ratio. In the next section, some most popular impurity measures are described.

3. BEST ATTRIBUTE SELECTION MEASURES

In a decision tree induction, the best attribute for splitting is selected using commonly used information theory or distance measures [6,7,8]. Following are some widely used attribute selection methods. Gain Ratio: The main objective of classification is to maximize the information gain for increasing the prediction accuracy of the classifier [9,10]. The advanced version of "Information Gain" introduced in ID3 is Gain Ratio [8,11]. Information Gain measures information given by attributes about the class. An attribute with the highest information gain is selected as the best attribute for splitting in a decision tree [12-13]. The information gain is based on the decrease in entropy after a dataset is split on a selected attribute. ID3 [9, 14,15] decision tree algorithm uses entropy to calculate the homogeneity of a sample. Entropy is a measure of impurity or randomness in data. If the instances are homogeneous then entropy is zero. According to information theory, higher entropy leads to higher diversion [16]. The reason for splitting is to create a lower diversion (purer distribution) of class values in the subsequent partitions. The nonclass attribute that produces the lowest entropy (lowest diversion) in subsequent partitions attains the highest gain in entropy reduction (called Information Gain). Information gain is calculated using a probability distribution pi of each nonclass attribute. The training

set$T\,t\,r\,a\,i\,n$ consist of n different classes C = C 1 , C 2 , C 3 , … . $C\,n$. The probability $p\,i$ that an instance belongs to a class $C\,i$ is calculated using Eq. 1

$$p_i = \frac{freq(C_i, T_{train})}{|T_{train}|}$$

(1)

Where, $T\,t\,r\,a\,i\,n$ is the total number of instances in. The number of instances that belongs to a class $C\,i$ is given by , $T\,t$. Information gain of $T\,t\,r\,a\,i\,n$ is calculated using Eq. 2

$$info(T_{train}) = -\sum_{i=1}^{n} p_i \times \log_2(p_i)$$

(2)

4. CONFUSION MATRIX

A disarray grid is a table that is oftentimes used to depict the execution of a portrayal model (or "classifier") on a course of action of test data for which the certifiable regards are known. The disorder grid itself is by and large simple to see, anyway the connected phrasing can be dumbfounding.

I expected to make a "quick reference control" for perplexity grid phrasing since I was unable to find a current resource that fit my requirements: more modest in presentation, using numbers instead of optional factors, and explained both to the extent plans and sentences.

- There are two potential anticipated classes: "yes" and "no". In case we were anticipating the presence of a sickness, for instance, "yes" would mean they have the infection, and "no" would mean they don't have the illness.
- The classifier made a sum of 165 expectations (e.g., 165 patients were being tried for the presence of that infection).
- Out of those 165 cases, the classifier anticipated "yes" multiple times, and "no" multiple times.
- In reality, 105 patients in the example have the infection, and 60 patients don't.

Precision provides a measure of how accurate your model is in predicting the actual positives out of the total positives predicted by your system. Recall provides the number of actual positives captured by our model by classifying these as true positive. F-measure can provide a balance between precision and recall, and it is preferred over accuracy where data is unbalanced.

5. CONFUSION MATRIX

A disarray grid is a table that is oftentimes used to depict the execution of a portrayal model (or "classifier") on a course of action of test data for which the certifiable regards are known. The disorder grid itself is by and large simple to see, anyway the connected phrasing can be dumbfounding.

I expected to make a "quick reference control" for perplexity grid phrasing since I was unable to find a current resource that fit my requirements: more modest in presentation, using numbers instead of optional factors, and explained both to the extent plans and sentences.

- There are two potential anticipated classes: "yes" and "no". In case we were anticipating the presence of a sickness, for instance, "yes" would mean they have the infection, and "no" would mean they don't have the illness.

- The classifier made a sum of 165 expectations (e.g., 165 patients were being tried for the presence of that infection).
- Out of those 165 cases, the classifier anticipated "yes" multiple times, and "no" multiple times.
- In reality, 105 patients in the example have the infection, and 60 patients don't.

Precision provides a measure of how accurate your model is in predicting the actual positives out of the total positives predicted by your system. Recall provides the number of actual positives captured by our model by classifying these as true positive. F-measure can provide a balance between precision and recall, and it is preferred over accuracy where data is unbalanced.

6. SIMULATION TOOL

Different tools are used for this study. All of them are free and open source.

1. Python 3.5
2. NumPy 1.11.3
3. Pandas 0.19.1
4. Seaborn 0.7.1
5. SciPy and Scikit-learn 0.18.1

Python is a general programming language and is broadly utilized in a wide range of disciplines like general programming, web improvement, programming advancement, information investigation, M and so on Python is utilized for this venture since it is entirely adaptable and simple to utilize and furthermore documentation and local area support is exceptionally huge.

NumPy is amazingly fantastic group which enables us for coherent enlisting. It goes with refined limits and can perform N-layered display, variable based math, Fourier change, etc NumPy is used where in data examination, picture getting ready and besides exceptional various libraries are worked above NumPy and NumPy goes probably as a base stack for those libraries.

Pandas is open source BSD approved programming exceptionally created for python programming language. It gives complete plan of data examination gadgets for python and is best competitor for R programming language. Assignments like scrutinizing data layout, examining csv and dominate records, cutting, ordering, combining, taking care of missing information and so on, can be effortlessly performed with Pandas. Most significant element of Pandas is, it can perform time series investigation.

Seaborn is library utilized for information perception and is made by utilizing python programming language. It is obvious level library stacked on top of matplotlib. Seaborn is more charming and instructive than matplotlib and incredibly easy to use and is solidly joined with NumPy and Pandas. Seaborn and matplotlib can be used essentially one close to the next to get closes from the datasets.

SciPy is a variety of key mathematical limits and depends on the most elevated place of Numpy while Scikit-learn is extensively used renowned library for ML it is untouchable development to SciPy. Scikit-learn consolidate all of the gadgets and computations needed for most of ML errands. Scikit-learn maintains backslide, plan, gathering, dimensionality reduction and data pre-getting ready.

For this examination, scikit-learn is used in light of the fact that it relies upon python and can interoperate to NumPy library. It is also amazingly easy to use.

5.1 SIMULATION RESULTS

```
df = pd.DataFrame()
for i in ['/content/drive/MyDrive/DATASET_FAULT_EXCEL/svmBGFAULT100.xlsx',
'/content/drive/MyDrive/DATASET_FAULT_EXCEL/svmBGFAULT200.xlsx',
'/content/drive/MyDrive/DATASET_FAULT_EXCEL/svmBGFAULT50.xlsx',
'/content/drive/MyDrive/DATASET_FAULT_EXCEL/svmBGFAULT75.xlsx',
'/content/drive/MyDrive/DATASET_FAULT_EXCEL/svmCAFAULT100.xlsx',
'/content/drive/MyDrive/DATASET_FAULT_EXCEL/svmCAFAULT150.xlsx',
'/content/drive/MyDrive/DATASET_FAULT_EXCEL/svmCAFAULT200.xlsx',
'/content/drive/MyDrive/DATASET_FAULT_EXCEL/svmCAGFAULT100.xlsx',
'/content/drive/MyDrive/DATASET_FAULT_EXCEL/svmCAGFAULT150.xlsx',
'/content/drive/MyDrive/DATASET_FAULT_EXCEL/svmCAGFAULT200.xlsx',
'/content/drive/MyDrive/DATASET_FAULT_EXCEL/svmCGFAULT100.xlsx',
'/content/drive/MyDrive/DATASET_FAULT_EXCEL/svmCGFAULT150.xlsx',
'/content/drive/MyDrive/DATASET_FAULT_EXCEL/svmCGFAULT200.xlsx',
'/content/drive/MyDrive/DATASET_FAULT_EXCEL/svmCGFAULT50.xlsx',
'/content/drive/MyDrive/DATASET_FAULT_EXCEL/svmCGFAULT75.xlsx',
'/content/drive/MyDrive/DATASET_FAULT_EXCEL/svmNOFAULT.xlsx'
]:
```

```
dictt_3 = dict()
dictt_3['BGFAULT50'] = 'BGFAULT'
dictt_3['BGFAULT75'] = 'BGFAULT'
dictt_3['BGFAULT100'] = 'BGFAULT'
dictt_3['BGFAULT200'] = 'BGFAULT'
dictt_3['CAFAULT100'] = 'CAFAULT'
dictt_3['CAFAULT150'] = 'CAFAULT'
dictt_3['CAFAULT200'] = 'CAFAULT'
dictt_3['CAGFAULT100'] = 'CAGFAULT'
dictt_3['CAGFAULT150'] = 'CAGFAULT'
dictt_3['CAGFAULT200'] = 'CAGFAULT'
dictt_3['CGFAULT50'] = 'CGFAULT'
dictt_3['CGFAULT75'] = 'CGFAULT'
dictt_3['CGFAULT100'] = 'CGFAULT'
dictt_3['CGFAULT150'] = 'CGFAULT'
dictt_3['CGFAULT200'] = 'CGFAULT'
dictt_3['NOFAULT'] = 'NOFAULT'
```

```
accuracy_score(Y_Test, Y_RF_pred)

    0.8664169787765293
```

Fault Type	Resistance	Accuracy
AG ,BG,CG	25Ω	47.17
AG ,BG,CG	25Ω,50 Ω	82.89
AG ,BG,CG	25Ω,50Ω ,75Ω	89.03
AG ,BG,CG	25Ω,50 Ω ,75Ω,100 Ω	92.93
AG ,BG,CG	25Ω,50 Ω ,75Ω,100 Ω ,150Ω	93.88
AG ,BG,CG	25Ω,50 Ω, 75Ω,100 Ω ,150Ω 200Ω	95.15
AG ,BG,CG	25Ω,50 Ω, 75Ω,100 Ω ,150Ω 200Ω,300Ω	94.44

Table 1: Fault Classification for LG Fault

Table 2: Fault Classification For L l Fault

Fault Type	Resistance	Accuracy
AC ,BC,CA	25Ω	49.50
AC ,BC,CA	25Ω,50 Ω	83.55
AC ,BC,CA	25Ω,50Ω ,75Ω	90.25
AC ,BC,CA	25Ω,50 Ω ,75Ω,100 Ω	92.76
AC ,BC,CA	25Ω,50 Ω ,75Ω,100 Ω ,150Ω	93.94
AC ,BC,CA	25Ω,50 Ω, 75Ω,100 Ω ,150Ω 200Ω	94.68
AC ,BC,CA	25Ω,50 Ω, 75Ω,100 Ω ,150Ω 200Ω,300Ω	94.15

Fault Type	Resistance	Accuracy
ABCG ,ABC	25Ω	38.40
ABCG ,ABC	25Ω,50 Ω	74.06
ABCG ,ABC	25Ω,50Ω ,75Ω	71.07
ABCG ,ABC	25Ω,50 Ω ,75Ω,100 Ω	78.55
ABCG ,ABC	25Ω,50 Ω ,75Ω,100 Ω ,150Ω	80.09
ABCG ,ABC	25Ω,50 Ω, 75Ω,100 Ω ,150Ω 200Ω	81.96
ABCG ,ABC	25Ω,50 Ω, 75Ω,100 Ω ,150Ω 200Ω,300Ω	81.36

Table 3: Fault Classification for LIG Fault

Fault Type	Resistance	Accuracy
ACG ,BCG,CAG	25Ω	51.16
ACG ,BCG,CAG	25Ω,50 Ω	85.38
ACG ,BCG,CAG	25Ω,50Ω ,75Ω	90.47
ACG ,BCG,CAG	25Ω,50 Ω ,75Ω,100 Ω	92.43
ACG ,BCG,CAG	25Ω,50 Ω ,75Ω,100 Ω ,150Ω	93.35
ACG ,BCG,CAG	25Ω,50 Ω, 75Ω,100 Ω ,150Ω 200Ω	93.40
ACG ,BCG,CAG	25Ω,50 Ω, 75Ω,100 Ω ,150Ω 200Ω,300Ω	93.44

Table 4: Fault Classification for LLG and LLLG Fault

CONCLUSION:

But with the advent in field of engineering particularly in the power system, some efficient algorithms recently have come out for the purpose of fault detection and recognizing. It has already been discussed in the previous chapters what are the some current and future trends towards the given purpose with precise and fast response.

A fuzzy logic based technique has been presented for classification of faults. The proposed technique requires post fault currents of all three phases at one end of the transmission system. In this presented method, separate rules have been framed for both ground and phase faults. Simulation has been performed by considering various conditions to test the efficiency of the presented technique. The simulation results have led to a conclusion that the technique is quite robust for medium and long transmission lines but not very opt for short transmission lines.

REFERENCES:

[1] T. Lobos, P. Kostyla, J. Pospieszna, MJaroszewski, Location of FaultsOn Transmission Lines Using Wavelet Transforms, International Conference on High Voltage Engineering and Application, November9-13, 2008, pp: 633-636.

[2] P K Murthy, J Amarnath, S Kamakshiah et al., "Wavelet transform approach for detection and location of fault HVDC system[C]", Proceedings of 2008 IEEE Region 10 and the Third International Conference on Industrial and Information Systems, pp. 1-6, December 8-10, 2008.

[3] B. R. Reddy, M. V. Kumar, M. Suryakalavathi et al., "Fault detection classification and location on transmission lines using wavelet transform", the proceedings of the 2009 Annual Report Conference on Electrical Insulation and Dielectric Phenomena, pp. 409-411, 2009.

[4] Zhengyou He, Ling Fu, Sheng Lin, and Zhiqian Bo, "Fault detection and classification in EHV transmission line based on wavelet singular entropy, "IEEE Trans. Power Del, vol. 25, no. 4, 2010, pp. 2156-2163.

[5] A. Yadav, A. Swetapadma, "A novel transmission line relaying scheme for fault detection and classification using wavelet transform and linear discriminant analysis", Ain Shams Engineering Journal, 2014.

[6] R. C. Mishra, P. M. Deoghare, C. Bhale, S. Lanjewar, "Wavelet Based Transmission Line Fault Classification And Location", IEEE International Conference on Smart Electric Grid (ISEG), pp. 1-5.

[7] Majid Jamil, Rajveer Singh, Sanjeev Kumar Sharma, "Fault identification in electrical power distribution system using combined discrete wavelet transform and fuzzy logic", Journal of Electrical Systems and Information Technology, vol. 2, pp. 257-267, 2015.

[8] N. A. Sundaravaradan , Rounak Meyur , P. Rajaraman , "A wavelet based novel technique for detection and classification of parallel transmission line faults" , International Conference on Power and Embedded System (SCOPES) – 2016, Vol. 2, pp. 1951 - 1955 , 2016.

[9] S. Kirubadevi, S. Sutha, "Wavelet based transmission line fault identification and classification", International Conference on Computation of Power Energy Information and Communication (ICCPEIC) – 2017, pp. 737 – 741, 2017.

[10] J.R. Quinlan, "Induction of Decision Trees," Machine Learning, vol. 1, no. 1, pp. 81-106, 1986.

[11] G. M.Weiss, "Mining with rarity: A unifying framework," ACM SIGKDD Explor. Newslett., vol. 6, no. 1, pp. 7–19, Jun. 2004.

[12] N. V. Chawla, N. Japkowicz, and A. Kolcz, Eds., Special Issue Learning Imbalanced Datasets, SIGKDD Explor. Newsl., vol. 6, no. 1, 2004.

[13] W.-Z. Lu and D.Wang, "Ground-level ozone prediction by support vector machine approach with a cost-sensitive classification scheme," Sci. Total. Enviro., vol. 395, no. 2-3, pp. 109–116, 2008.

[14] Y.-M. Huang, C.-M. Hung, and H. C. Jiau, "Evaluation of neural networks and data mining methods on a credit assessment task for class imbalance problem" Nonlinear Anal. R. World Appl., vol. 7, no. 4, pp. 720–747, 2006.

[15] D. Cieslak, N. Chawla, and A. Striegel, "Combating imbalance in network intrusion datasets" in IEEE Int. Conf. Granular Comput., 2006, pp. 732–737.

[16] M. A. Mazurowski, P. A. Habas, J. M. Zurada, J. Y. Lo, J. A. Baker, and G. D. Tourassi, "Training neural network classifiers for medical decision making: The effects of imbalanced datasets on classification performance," Neural Netw., vol. 21, no. 2–3, pp. 427–436, 2008.