RAJSHAHI UNIVERSITY OF ENGINEERING AND TECHNOLOGY

CSE 3210

Lab Report

Submitted to:
Mahit Kumar Paul
Lecturer,
Department of Computer Science and Engineering
Rajshahi University of Engineering and Technology

Submitted by:
Riyad Morshed Shoeb
Roll No: 1603013
Section: A
Department of Computer Science and Engineering
Rajshahi University of Engineering and Technology

Decision Tree

Introduction: A decision tree is one of the supervised machine learning algorithms. It is a flowchart-like structure in which each internal node represents a test on an attribute, each branch represents the outcome of the test, and each leaf node represents a class label i.e. decision taken after computing all attributes. A decision tree follows a set of if-else conditions to visualize the data and classify it according to the conditions. The paths from root to leaf represent classification rules.

Description: A decision tree represents a function that takes as input a vector of attribute values and returns a decision, a single output value. The input and output values can be discrete or continuous. It is a predictive model expressed as a recursive partition of the feature space to subspaces that constitute a basis for prediction. It is a rooted directed tree.

In Decision Trees, nodes with outgoing edges are the internal nodes. All other nodes are terminal nodes or leaves of the tree. Decision Trees classify using a set of hierarchical decisions on the features. The decisions made at internal nodes are the split criterion. In Decision Trees, each leaf is assigned to one class or its probability. Each internal node in the tree corresponds to a test of the value of one of the input attributes and the branches from the node are labeled with possible values of the attribute.

Terms related to Decision Trees are:

1. Root Node: This attribute is used for dividing the data into two or more sets. The feature attribute in this node is selected based on Attribute Selection Techniques.
2. Branch or Sub-Tree: A part of the entire decision tree is called a branch or sub-tree.
3. Splitting: Dividing a node into two or more sub-nodes based on if-else conditions.
4. Decision Node: After splitting the sub-nodes into further sub-nodes, then it is called the decision node.
5. Leaf or Terminal Node: This is the end of the decision tree where it cannot be split into further sub-nodes.
6. Pruning: Removing a sub-node from the tree is called pruning.
7. Information Gain: Entropy is the main concept of this algorithm, which helps to determine a feature or attribute that gives maximum information about a class is called Information gain. By using this method, we can reduce the level of entropy from the root node to the leaf node. The feature or attribute with the highest information gain is used as the root for the splitting.

A decision tree is made up of two major procedures - building the tree and classification.

1. Building the tree: Constructing a decision tree is a top–down building procedure. It starts at the root with the whole training set. The objective is to find in each decision node of the tree, the best test attribute allowing to diminish as much as possible the mixture of classes between each subset created by the test. This process will continue for each sub decision tree until reaching leaves and fixing their corresponding classes.
2. Classification: Classifying new objects is based on the induced tree. So, to classify an object, we start from the root, we evaluate the relative test attribute and we take the branch corresponding to the test's outcome. This process is repeated until a leaf is encountered. The new object is then classified to the class labeling the leaf.

The decision tree method is a powerful statistical tool for classification, prediction, interpretation, and data manipulation. Building a decision tree that is consistent with a given data set is easy. The challenge lies in building good decision trees, which typically means the smallest decision trees.

Fields:

1. Financial institutions use decision trees. One of the fundamental use cases is in option pricing, where a binary-like decision tree is used to predict the price of an option in either a bull or bear market.
2. Marketers use decision trees to establish customers by type and predict whether a customer will buy a specific type of product.
3. In the medical field, decision tree models have been designed to diagnose blood infections or even predict heart attack outcomes in chest pain patients. Variables in the decision tree include diagnosis, treatment, and patient data.
4. The gaming industry now uses multiple decision trees in movement recognition and facial recognition. The Microsoft Kinect platform uses this method to track body movement. The Kinect team used one million images and trained three trees. Within one day, and using a 1,000-core cluster, the decision trees were classifying specific body parts across the screen.

Uses and Applications:

1. <u>Assessing prospective growth opportunities:</u> One of the applications of decision trees involves evaluating prospective growth opportunities for businesses based on historical data. Historical data on sales can be used in decision trees that may lead to making radical changes in the strategy of a business to help aid expansion and growth.
2. <u>Using demographic data to find prospective clients:</u> Another application of decision trees is in the use of demographic data to find prospective clients. They can help in streamlining a marketing budget and in making informed

decisions on the target market that the business is focused on. In the absence of decision trees, the business may spend its marketing market without a specific demographic in mind, which will affect its overall revenues.

3. Serving as a support tool in several fields: Lenders also use decision trees to predict the probability of a customer defaulting on a loan, by applying predictive model generation using the client's past data. The use of a decision tree support tool can help lenders in evaluating the creditworthiness of a customer to prevent losses.
4. Decision trees can also be used in operations research in planning logistics and strategic management. They can help in determining appropriate strategies that will help a company achieve its intended goals. Other fields where decision trees can be applied include engineering, education, law, business, healthcare, and finance.

Drawbacks: For a wide variety of problems, the decision tree format yields a nice, concise result. But some functions cannot be represented concisely. For example, the majority function, which returns true if and only if more than half of the inputs are true, requires an exponentially large decision tree. In other words, decision trees are good for some kinds of functions and bad for others.

There is a danger of over-interpreting the tree that the algorithm selects. When there are several variables of similar importance, the choice between them is somewhat arbitrary: with slightly different input examples, a different variable would be chosen to split on first, and the whole tree would look completely different. The function computed by the tree would still be similar, but the structure of the tree can vary widely.

The small variation in the input data can result in a different decision tree.

Decision Trees are unstable, meaning that a small change in the data can lead to a large change in the structure of the optimal decision tree.

Decision Trees are often relatively inaccurate. Many other predictors perform better with similar data.

Calculations can get very complex, particularly if many values are uncertain and/or if many outcomes are linked.

Conclusion: While Decision Trees are easy to work with and work well with many data, they can also create faulty trees. Carefully prepared data can lead to creating a better decision tree and more accurate result.

Resources:

1. Artificial Intelligence: A Modern Approach - Stuart Russel and Peter Norvig
2. Machine Learning: Hands-on for developers and technical professionals - Jason Bell
3. https://en.wikipedia.org/wiki/Decision_tree

4. Decision Trees Explained With a Practical Example – Towards AI — The Best of Tech, Science, and Engineering
5. https://www.sciencedirect.com/topics/computer-science/decision-trees
6. https://corporatefinanceinstitute.com/resources/knowledge/other/decision-tree/

K-Means Clustering

Introduction: K-means clustering is a method of vector quantization, originally from signal processing, that aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean (cluster centers or cluster centroid), serving as a prototype of the cluster. The problem is computationally difficult (NP-hard); however, efficient heuristic algorithms converge quickly to a local optimum.

Description: K-means clustering is one of the simplest and popular unsupervised machine learning algorithms. Typically, unsupervised algorithms make inferences from datasets using only input vectors without referring to known, or labelled outcomes.

A cluster refers to a collection of data points aggregated together because of certain similarities. A centroid is the imaginary or real location representing the center of the cluster. Every data point is allocated to each of the clusters through reducing the in-cluster sum of squares.

In other words, the K-means algorithm identifies *k* number of centroids, and then allocates every data point to the nearest cluster, while keeping the centroids as small as possible. The '*means*' in the K-means refers to averaging of the data i.e. finding the centroid.

While working with K-means algorithm we need to take care of the following things:

1. It is recommended to standardize the data because such algorithms use distance-based measurement to determine the similarity between data points.
2. Due to the iterative nature of K-Means and random initialization of centroids, K-Means may stick in a local optimum and may not converge to global optimum. That is why it is recommended to use different initializations of centroids.

To process the learning data, the K-means algorithm starts with a first group of randomly selected centroids, which are used as the beginning points for every cluster, and then performs iterative calculations to optimize the positions of the centroids. It halts creating and optimizing clusters when either:

1. The centroids have stabilized i.e. there is no change in their values because the clustering has been successful.
2. The defined number of iterations has been achieved.

Fields:

1. Market segmentation: Companies gather all sorts of data from their customer base. Performing k-means clustering analysis on the customer base of a company reveals market segments that have defined characteristics.

Customers belonging to the same segment can be treated similarly. Different segments receive different treatment.

2. Classification of books, movies, or other documents: When influencers build their personal brand, authors write books and create books, or a company manages its social media accounts, content is king. Content is often described by hashtags and other data. This data can be used as a basis for clustering to locate groups of documents that are similar in nature.
3. Detection of fraud and criminal activities: Fraudsters often leave clues in the form of unusual customer or visitor behavior. For instance, car insurance protects drivers from theft and damage arising from accidents. Real theft and fake theft are characterized by different feature values. Similarly, wrecking a car on purpose leaves different traces than wrecking a car by accident. Clustering can often detect fraud, helping industry professionals understand the behavior of their worst customers better.

Uses and Applications:

1. Behavioral segmentation:
    a. Segment by purchase history
    b. Segment by activities on application, website, or platform
    c. Define personas based on interests
    d. Create profiles based on activity monitoring
2. Inventory categorization:
    a. Group inventory by sales activity
    b. Group inventory by manufacturing metrics
3. Sorting sensor measurements:
    a. Detect activity types in motion sensors
    b. Group images
    c. Separate audio
    d. Identify groups in health monitoring
4. Detecting bots or anomalies:
    a. Separate valid activity groups from bots
    b. Group valid activity to clean up outlier detection

Drawbacks: The following are some disadvantages of K-Means clustering algorithms:

1. It is a bit difficult to predict the number of clusters i.e. the value of k.
2. Output is strongly impacted by initial inputs like number of clusters (value of k)
3. Order of data will have a strong impact on the final output.
4. It is very sensitive to rescaling. If we will rescale our data by means of normalization or standardization, then the output will completely change.
5. It is not good to do clustering jobs if the clusters have a complicated geometric shape.

Conclusion: The K-means algorithm assigns the points to clusters very similarly to how we might assign them by eye. But it is challenging when the size of the clusters are different. It is also problematic when the density of the clusters are different. Resultant clusters can be quite different from expectation. Even after its drawbacks and challenges, K-means is still useful in many areas.

Resources:

1. Artificial Intelligence and Machine Learning Fundamentals - Zsolt Nagy
2. https://en.wikipedia.org/wiki/K-means_clustering
3. https://towardsdatascience.com/understanding-k-means-clustering-in-machine-learning-6a6e67336aa1
4. https://blogs.oracle.com/datascience/introduction-to-k-means-clustering
5. https://www.tutorialspoint.com/machine_learning_with_python/machine_learning_with_python_clustering_algorithms_k_means.htm

Performance of a Classifier

Introduction: After implementing a machine learning algorithm, we need to find out how effective the model is. The criteria for measuring the effectiveness may be based upon datasets and metrics. For evaluating different machine learning algorithms, we can use different performance metrics. In one or other sense, the metric we choose to evaluate our machine learning model is very important because the choice of metrics influences how the performance of a machine learning algorithm is measured and compared.

Description: Evaluating model performance can tell us if our approach is working. This turns out to be helpful. We can continue to explore and see how far we can push our existing concept of the problem we are working on. It can also tell us if our approach is not working. This turns out to be even more helpful because if our adjustments are making the model worse at what it is supposed to be doing, then it indicates that we may have misunderstood our data or the situation being modelled.

We use the terms true/false positive and true/false negative to describe each of the four possible outcomes. The true/false part refers to whether the model was correct or not. The positive/negative part refers to whether the instance being classified actually was or was not the instance we wanted to identify. A good model will have a high level of true positive and true negatives, because these results indicate where the model has got the right answer. A good model will also have a low level of false positives and false negatives, which indicate where the model has made mistakes. These four numbers can tell us a lot about how the model is doing and what we can do to help. Often, it's helpful to represent them as a confusion matrix.

Confusion Matrix: A confusion matrix is a table that is often used to describe the performance of a classification model on a set of test data for which the true values are known. Basically it is used for classification problems where the output can be of two or more types of classes. It is the easiest way to measure the performance of a classifier.

A confusion matrix is basically a table with two dimensions namely "Actual" and "Predicted". Both the dimensions have "True Positives (TP)", "True Negatives (TN)", "False Positives (FP)", "False Negatives (FN)". A true positive means we predicted a sample belonged to our target class ("positive"), and it actually did ("true"). Similarly, a false negative means we predicted that a sample did not belong to our target class ("negative"), we were wrong ("false").

Accuracy: This term refers to the percentage of how much data the classifier predicted correctly from all the data in the dataset. It can be calculated from the confusion matrix as the total number of correct predictions divided by the number of all predictions.

Precision: Precision is interested in the number of genuinely positive examples the model identified against all the examples it labeled positive.

Mathematically, it is the number of true positives divided by the true positives plus the false positives.

Recall or Sensitivity: The term recall refers to the proportion of genuine positive examples that a predictive model has identified. To put that another way, it is the number of true positive examples divided by the total number of positive examples and false negatives.

Specificity: Specificity is the number of genuinely negative examples the model identified divided by total negative examples in the dataset. It is mathematically defined by the proportion of true negative examples to true negative and false positive examples.

Conclusion: Understanding classifier performance is crucial for developing effective machine learning models that generate value. The performance metrics we choose should not only make sense given the problem but should also align with our goals.

Resources:

1. Artificial Intelligence with Python - Tutorialspoint
2. https://towardsdatascience.com/evaluating-classifier-model-performance-6403577c1010
3. https://kambria.io/blog/confused-about-the-confusion-matrix-learn-all-about-it/