Final Examination Report On Decision Tree, K-Nearest Neighbors

CSE-0408 Summer 2021

Rebecca Sultana
Dept. of CSE
State University of Bangladesh (SUB)
Dhaka, Bangladesh
rebecca.sultana.riya@gmail.com

***Abstract*—This is my academic projects.**

Assignment 1. Classification and Learning [Decision Tree]

I. Introduction

Decision Tree algorithm belongs to the family of supervised learning algorithms. Unlike other supervised learning algorithms, the decision tree algorithm can be used for solving regression and classification problems too.

The goal of using a Decision Tree is to create a training model that can use to predict the class or value of the target variable by learning simple decision rules inferred from prior data(training data).

In Decision Trees, for predicting a class label for a record we start from the root of the tree. We compare the values of the root attribute with the record's attribute. On the basis of comparison, we follow the branch corresponding to that value and jump to the next node.

II. Types of Decision Trees

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.

III. Advantages

1. Compared to other algorithms decision trees requires less effort for data preparation during pre-processing.
2. A decision tree does not require normalization and scaling of data.
3. Missing values in the data also do NOT affect the process of building a decision tree to any considerable extent.

IV. Drawbacks

1. A small change in the data can cause a large change in the structure of the decision tree causing instability.
2. For a Decision tree sometimes calculation can go far more complex compared to other algorithms.
3. Decision tree training is relatively expensive as the complexity and time has taken are more.

Assignment 2. K-Nearest Neighbor(K-NN) Algorithm

V. Introduction

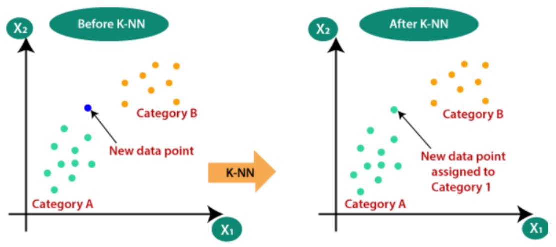
The k-nearest neighbors (K-NN) algorithm is a simple, supervised machine learning algorithm that can be used to solve both classification and regression problems. It's easy to implement and understand, but has a major drawback of becoming significantly slows as the size of that data in use grows.

VI. Why do we need a K-NN Algorithm?

Suppose there are two categories, i.e., Category A and Category B, and we have a new data point x1, so this data point will lie in which of these categories. To solve this type of problem, we need a K-NN algorithm. With the help of K-NN, we can easily identify the category or class of a particular dataset.

VII. How does K-NN work?

Step-1: Select the number K of the neighbors.
Step-2: Calculate the Euclidean distance of K number of neighbors.
Step-3: Take the K nearest neighbors as per the calculated Euclidean distance.
Step-4: Among these k neighbors, count the number of the data points in each category.
Step-5: Assign the new data points to that category for which the number of the neighbor is maximum.



VIII. Advantages of K-NN Algorithm:

1. It is simple to implement.
2. It is robust to the noisy training data.
3. It can be more effective if the training data is large.

IX. Drawbacks of K-NN Algorithm:

1. Always needs to determine the value of K which may be complex some time.
2. The computation cost is high because of calculating the distance between the data points for all the training samples.

X. K-NN Code

Code figure in below:
importing libraries
import numpy as nm
import matplotlib.pyplot as mtp
import pandas as pd

importing datasets
data$_set = pd.read_csv('user_data.csv')$

Extracting Independent and dependent Variable
x= data$_set.iloc[:,[2,3]].values$
$y = data_set.iloc[:,4].values$

Splitting the dataset into training and test set.
from sklearn.model$_selectionimporttrain_test_split$
$x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.25, random_state = 0)$

feature Scaling
from sklearn.preprocessing import StandardScaler
st$_x = StandardScaler()$
$x_train = st_x.fit_transform(x_train)$
$x_test = st_x.transform(x_test)$

Fitting K-NN classifier to the training set
from sklearn.neighbors import KNeighborsClassifier
classifier= KNeighborsClassifier(n$_neighbors = 5, metric =' minkowski', p = 2)$
$classifier.fit(x_train, y_train)$

Predicting the test set result
y$_pred = classifier.predict(x_test)$