AYUSH
KUMAR
DWIVEDI
2018802002

Statistical Methods in Artificial Intelligence

ECE 471

SPRING
2019

Assignment-1 Report

DECISION TREE

Decision Tree

Decision tree is a type of supervised learning algorithm where the data is split into two or more homogeneous sets based on most significant input variables. Each decision outcome at a node is called split. The root node splits the full training data. Each decision results into a proper subset of data. To decide which feature of the data is to be used for performing the split, we make use of information gain(IG). To find IG, Entropy, Gini Index or Misclassification Rate can be used as the information measure.

INFORMATION MEASURE

1. **Entropy:**

$$E(s) = \sum_{i \in X} -p_i \, log(p_i)$$

2. **Gini Index:**

$$G(X) = 1 - \sum_{c \in X} p^2(i)$$

3. **Misclassification Rate:**

$$M(X) = 1 - max(p)$$

IMPLEMENTATION

- Both the data files decision_tree_train and decision_tree _test is imported using Pandas.
- Made a class decision node to create decision tree.
- This is a recursive algorithm which calls itself to create new node.

DEFINATIONS

Definitions:

1. **Split_set(dataset, column, value)** : splits the data set into true branch and false branch based on the column and value given.

2. **count_class(dataset):** count the number of positive and negative labels in the given dataset.

3. **entropy(dataset):** Calculates the entropy of the dataset.

4. **create_tree(dataset):** Creates decision tree by using the above definitions.

5. **classify(test_row, decision_tree):** Classifies a test row to negative or positive label using the learnt decision tree.

PART – 1, 2 & 3

Command to run the script:

```
python q-1.py --data_type <0/1/2> --measure_type <0/1/2>
```

data_type 0: Numerical Data	measure_type 0: Entropy
data_type 1: Categorical Data	measure_type 1: Gini
data_type 2: Complete Data	measure_type 2: Misclassification

Output is presented on the next page in the following manner:

1. **Red Box**: Calculation of performance (Accuracy, Recall, Precision and F1 Score) for **both categorical and complete** data one by one **using Entropy** as information measure.
2. **Yellow Box**: Calculation of performance (Accuracy, Recall, Precision and F1 Score) for **both categorical and complete** data one by one **using Gini Index** as information measure.
3. **Green**: Calculation of performance (Accuracy, Recall, Precision and F1 Score) for **both categorical and complete** data one by one **using Misclassifications rate** as information measure.

C:\Windows\System32\cmd.exe

```
C:\Users\minak\Desktop\DT>python q-1.py --data_type 1 --measure_type 0
Data Type:  Categorical
Information Measure:  Entropy
Training Accuracy: 0.7633333333333333, Recall: 0.7633333333333333, Precision: 0.6617239079693232, f1Score: 0.7089061432435553
Training Accuracy: 0.2822222222222222, Recall: 0.2822222222222222, Precision: 0.6506427076771201, f1Score: 0.3936814965337674
Predicted Labels: [1. 0.]

C:\Users\minak\Desktop\DT>python q-1.py --data_type 1 --measure_type 1
Data Type:  Categorical
Information Measure:  Gini
Training Accuracy: 0.7633333333333333, Recall: 0.7633333333333333, Precision: 0.6617239079693232, f1Score: 0.7089061432435553
Training Accuracy: 0.2822222222222222, Recall: 0.2822222222222222, Precision: 0.6506427076771201, f1Score: 0.3936814965337674
Predicted Labels: [1. 0.]

C:\Users\minak\Desktop\DT>python q-1.py --data_type 1 --measure_type 2
Data Type:  Categorical
Information Measure:  Miss Classification
Training Accuracy: 0.7633333333333333, Recall: 0.7633333333333333, Precision: 0.6617239079693232, f1Score: 0.7089061432435553
Training Accuracy: 0.244, Recall: 0.244, Precision: 0.059670964873277006, f1Score: 0.09589138978206502
Predicted Labels: [1. 0.]

C:\Users\minak\Desktop\DT>python q-1.py --data_type 2 --measure_type 0
Data Type:  Both(Numerical and Categorical)
Information Measure:  Entropy
Training Accuracy: 0.7634444444444445, Recall: 0.7634444444444445, Precision: 0.7634444444444446, f1Score: 0.7634444444444444
Training Accuracy: 0.6613333333333333, Recall: 0.6613333333333333, Precision: 0.6349356231919489, f1Score: 0.6478656917977641
Predicted Labels: [1. 0.]

C:\Users\minak\Desktop\DT>python q-1.py --data_type 2 --measure_type 1
Data Type:  Both(Numerical and Categorical)
Information Measure:  Gini
Training Accuracy: 0.7634444444444445, Recall: 0.7634444444444445, Precision: 0.7634444444444446, f1Score: 0.7634444444444444
Training Accuracy: 0.6608888888888889, Recall: 0.6608888888888889, Precision: 0.63270552421882, f1Score: 0.6464901929968703
Predicted Labels: [1. 0.]

C:\Users\minak\Desktop\DT>python q-1.py --data_type 2 --measure_type 2
Data Type:  Both(Numerical and Categorical)
Information Measure:  Miss Classification
Training Accuracy: 0.7628888888888888, Recall: 0.7628888888888888, Precision: 0.6354043673302933, f1Score: 0.6933351493068654
Training Accuracy: 0.2986666666666667, Recall: 0.2986666666666667, Precision: 0.6230828311473473, f1Score: 0.4037844830452121
Predicted Labels: [1. 0.]

C:\Users\minak\Desktop\DT>
```

Observations:

Part 1 & 2: Performance is observed to better while using complete data when compared to using only numerical or only categorical data. Hence both numerical as well as categorical features seems to be important for accurate prediction. This is so because the number of attributes in alone categorical or alone numerical data are insufficient.

Part 3: The following seems to be the order of information measure in decreasing order of performance for this particular data set:

Entropy > Gini Index > Misclassification Ratio

Part 6: There are several ways in which decision tree provides flexibility in handling missing values in data set. Few of them are as follows:

1. Ignoring the missing values: If missing values are not spread across multiple features and data is rich in terms of number of features, then the features with missing values can be ignored.

2. Predefining the missing value: Missing values can be set with predefined values i.e. mean/median/mode of all the values or maximum of all the values or minimum of all the values etc.

3. Assign a unique category: The missing values can also be replaced with a 'NaN' and while gain and entropy calculations, these values can be excluded.

4. Method of assigning all possible values: In such case, multiple data sets are created by assigning the missing value with each of the possible values that it can take.

5. Predicting the missing values: By applying suitable machine learning algorithms, missing values can be predicted with the help of features which do not have any missing values.