SMAI (CSE 471)
Spring-2019
Assignment-1 (100 points)
Posted on: 11/1/2019
Due on: 20/1/2019, 11:55 PM

Part 1:-

1- imported necessary modules to be used like numpy, pandas

2- read data from input file(train.csv) into a dataframe via pandas.

3- dropped the target column and added it as the last coulmn in the dataframe.

4- Splitted the data into training data(80% for training the model) and test data(20% for validation).

5- Functions used:-

- **find_entropy**- finds entropy of the target column
- **find_entropy_attribute**- finds the entropy for the attribute provided as the parameter
- **find_winner**- returns the attribute with largest information gain
- **get_subtable**- return the rows where the particular column(attribute) has a value provided as the paramter
- **buildTree**- trains the model and the return the tree(I have used dictionary as an n-ary tree)
- **predict**- For predicting the target value for the input row
- **testing**- Provided the dataframe predicts the value for each row and prints the accuracy score, confusion matrix and classification report(containing the precision, recall, f1 score)
- **predictleft(tree,'sample_test.csv')**- predicts the value for the sample data

6- Algorithm used:-

I have followed ID3 algorithm and entropy as the criteria for building of the model(decision tree classifier).

Steps:-

- Calculated the entropy of every attribute using function find_entropy_attribuute
- Splitted the dataframe into subsets using the attribute for which information gain is maximum.
- Made the decision tree node contaning that attribute
- Recursed on subsets using the remaining attributes till the subtable dataset is pure or there is just one coulmn left(i.e on;y target column is left as the attribute in the subtable)

7- On testing the train model over the test data(20% of the actual data), below are stats encountered:-
accuracy score:- 0.7655393053016454
confusion matrix:-

1673	1

512	2

classification report:-

	precision	recall	f1-score	support
0	0.77	1.00	0.87	1674
1	0.67	0.00	0.01	514
Micro avg	0.77	0.77	0.77	2188
Macro avg	0.72	0.50	0.44	2188
Weighted avg	0.74	0.77	0.67	2188

8- Compared the result with in-built(scikit-learn) decision tree classifier to check correctness of algorithm used over the test data. Got the below stats as the output:-

accuracy score:- 0.7671173135630179
confusion matrix:-

1758	0
534	1

classification report:-

	precision	recall	f1-score	support
0	0.77	1.00	0.87	1758
1	1.00	0.00	0.00	535
Micro avg	0.77	0.77	0.77	2293
Macro avg	0.88	0.50	0.44	2293
Weighted avg	0.82	0.77	0.67	2293

Part 2:-

1- Converted all the numerical data to binary form .
Steps followed are mentioned below:-
- For each numerical attribute founded the midpoint from the values of that attribute which has high entropy.
- Assigned 0 as the attribute value whose value is less than midpoint of that attribute and 1 if value is greater than mid point.

2-Followed the steps mentioned in part1 for building tree taking all the attributes into account.

3- On testing the train model over the test data(20% of the actual data), below are stats encountered:-
accuracy score:- 0.8785495849716033

confusion matrix:-

1696	42
236	315

classification report:-

	precision	recall	f1-score	support
0	0.88	0.98	0.92	1758
1	0.88	0.57	0.69	551
Micro avg	0.88	0.88	0.88	2289
Macro avg	0.88	0.77	0.81	2289
Weighted avg	0.88	0.88	0.87	2289

4- Compared the result with in-built(scikit-learn) decision tree classifier to check correctness of algorithm used over the test data. Got the below stats as the output:-
accuracy score:- 0.7671173135630179
confusion matrix:-

1666	45
25	512

classification report:-

	precision	recall	f1-score	support
0	0.99	0.97	0.98	1711
1	0.92	0.95	0.94	537
Micro avg	0.97	0.97	0.97	2248
Macro avg	0.95	0.96	0.96	2248
Weighted avg	0.97	0.97	0.97	2248

Part 3:-

1- Followed same steps mentioned in part 2 for training of the model based on different criterias (gini indes, entropy, misclassification error).

2- Functions added:-
predictmisclassification(model,mid_point3,testdata)
model – decision tree classifier trained
mid_point – dictionary containing mid points for each numerical data attributes
testdata- sample test data
3- On testing the train model over the test data(20% of the actual data), below are stats encountered misclassification criteria:-
accuracy score:- 0.7814183123877917

confusion matrix:-

1740	1
486	1

classification report:-

	precision	recall	f1-score	support
0	0.78	1.00	0.88	1741
1	0.50	0.00	0.00	487
Micro avg	0.78	0.78	0.78	2228
Macro avg	0.64	0.50	0.44	2228
Weighted avg	0.72	0.78	0.69	2228

4- On testing the train model over the test data(20% of the actual data), below are stats encountered for entropy citeria:-
accuracy score:- 0.8785495849716033

confusion matrix:-

1678	63
178	309

classification report:-

	precision	recall	f1-score	support
0	0.90	0.96	0.93	1741
1	0.83	0.63	0.72	487
Micro avg	0.89	0.89	0.89	2228
Macro avg	0.87	0.80	0.83	2228
Weighted avg	0.89	0.89	0.89	2228

5- On testing the train model over the test data(20% of the actual data), below are stats encountered for gini index criteria:-
accuracy score:- 0.7814183123877917

confusion matrix:-

1740	1
486	1

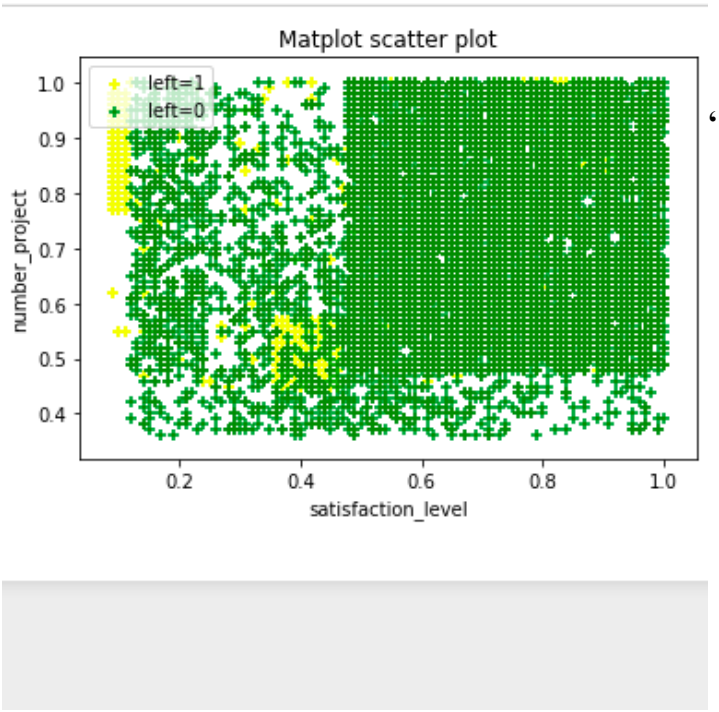
classification report:-

	precision	recall	f1-score	support

0	0.78	1.00	0.88	1741
1	0.50	0.00	0.00	487
Micro avg	0.78	0.78	0.78	2228
Macro avg	0.64	0.50	0.44	2228
Weighted avg	0.72	0.78	0.69	2228

Part 4:-

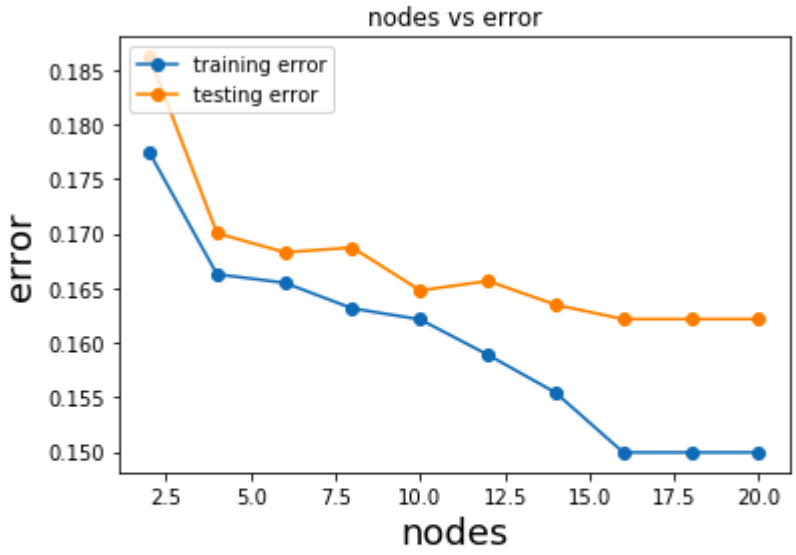
Plotted the scatter plot for attributes last_evaluation vs satisfaction_level.



Part 5:

1- Build the tree contaning 2,4,6 etc. Nodes, and calculated training and validation error for each tree and stored it in a list.

2- Plotted the graph for values stored in list calculated above.



Part 6:-

Handling missing data -

A. Discard Testing Instance-This is the simplest approach of discarding test cases with unknown attribute values. However, in practice, it may not be acceptable to reject few cases for prediction.

B. Imputation - Imputation a class of methods by which an estimation of missing value or of its distribution is used to generate prediction. For eg. The missing value might be imputed with the maximum, minimum, median,mean or commonly occuring value of the attribute.

C. C4.5 does not replace the missing values. At the time of selection an attribute at splitting node, all instances with known value of that attribute are used for information gain calculation. After selection of an attribute, instance with known attribute values are split as per actual values and instances with unknown attribute value are split in proportionate to the split off known values. At the time of testing, a test instance with missing value is split into branches according to the portions of training examples falling into those branches and goes down to leaves. This has an advantage over the methods of discarding all incomplete instances in that fewer instances are being discarded when computing the best attribute.

D. In ‘Null value’ method missing values are treated as a regular value, ‘Null’ in tree construction and testing process. As values might be missing for certain reason, it might be good idea to assign a special value. There are two drawbacks of this method. Firstly, it does not try to identify the original known values as missing values are considered as equally as a original value. Another disadvantage is, in case of more than one actual missing values and replacing all of them by one value ‘null’ may not be correct. Also, subtrees can be built under the null branch, oddly suggesting that the missing value is more discriminating then the known value. The advantage of this method is that all the records can be used for tree building and it is very simple to implement. This approach handles the missing at the training time as well.

E. Lazy Decision Tree (Reduced Feature Models/Known Value Strategy):- here the prediction model is constructed at testing time based on the available test instance values. This is also known as ‘Known values strategy’. During tree construction it uses only attributes whose values are known at testing. Hence it naturally handles the missing values at testing. The main drawback of this approach is its high computational cost.