HOMEWORK 4

CLASSIFICATION ALGORITHMS

CSE 601 – Data Mining and Bio-Informatics

UNIVERSITY AT BUFFALO

Authors:

HARISH MANGALAMPALLI

hmangala@buffalo.edu

KEYUR SANJEEV JOSHI

keyurjos@buffalo.edu

VIBHAV GUPTA

vibhavgu@buffalo.edu

ABSTRACT

OBJECTIVE

1. Implement the Classification algorithm to identify clusters in dataset. Report the accuracy of classes obtained for the given dataset.
2. Display the accuracy of results obtained:
    - Identify useful attributes from dataset.
    - Divide data into train and test set.
    - Perform classification using Decision-Tree and Naïve Bayes algorithms.

INTRODUCTION

What is classification?

Classification is the problem of identifying to which of a set of categories a new observation belongs. It is a type of supervised learning.

It basically involves training on a set of data and apply the learning to a test set.

The types of classification methods:

Naïve Bayes

Logistic Regression

Neural Networks

Support Vector Machines

Decision Tree

What is Naïve Byes Classification?

The Naïve Bayes classifier is probabilistic approach for classification based on Bayes theorem assuming independence of features.

The algorithm relies on simple logic that posterior probability of event occurring is proportional to the product of likelihood and prior. It is suited for categorical and independent features.

$$p(C|F_1,\ldots,F_n) = \frac{p(C)\ p(F_1,\ldots,F_n|C)}{p(F_1,\ldots,F_n)}.$$

The classifier calculates the posterior probability of the input vector containing features F1 to Fn belongs to class C1, C2 etc. The classifier labels the input as the class which has the maximum probability this is known as the MAP (Maximum Aposteriori Probability) approach.

$$\text{classify}(f_1,\ldots,f_n) = \operatorname*{argmax}_c\ p(C=c)\prod_{i=1}^{n} p(F_i=f_i|C=c).$$

Decision Tree

Decision Tree represents M-ary node tree of decision or features and their values along with the corresponding resultant class. It is a graphical form of multiple and/or nested if else statements with the conditions being the attribute values.

It is useful for classifying both numeric and categorical data. Training data is used to build the tree by selecting the most useful (maximum variance) features at each step. Different measure are used to select these features. The testing step involve traversing the tree along matching attribute values until leaf node is reached. The leaf node contains the actual result of classification.

Splitting on attributes is done either as binary or multi-way split. Algorithms used for choosing attribute

GINI INDEX

$$GINI(t) = 1 - \sum_j [p(j|t)]^2 \text{ , where} GINI(t) = \sum_{i=1}^{k} \frac{ni}{n} GINI(i)$$

INFORMATION GAIN

$$GAIN = Entropy(p) - (\sum_{i=1}^{k} \frac{n_i}{n} Entropy(i)\ ) \text{ ,where } Entropy(p) = -\sum_{i=1}^{N} p_i * \log p_i$$

Cross Validation

Cross-validation or rotation estimation, is a model validation technique for assessing how the results of a statistical analysis will generalize to an independent data set. This involves dividing data into k parts and using k-1 for training and one for testing and averaging the output across each of the k-1 combinations. By supplying different combination of test and train data we understand how our model responds to variations input data and its response

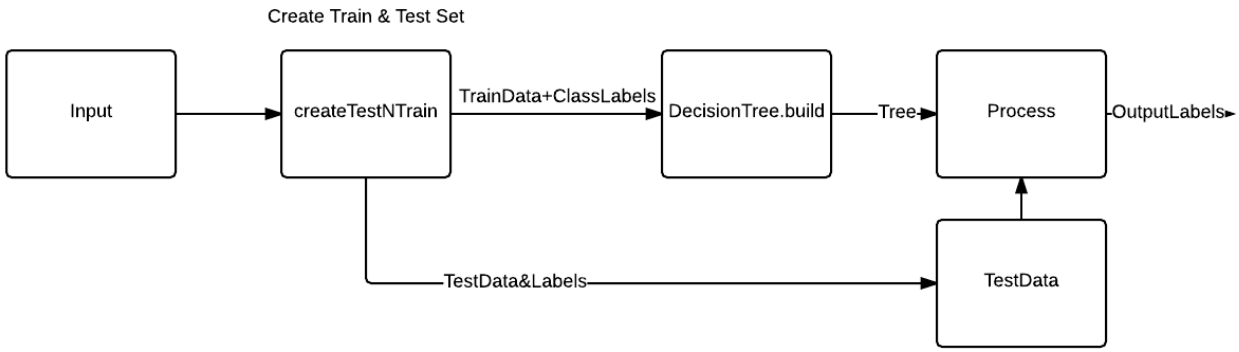
APPROACH

1. Decision tree implemented here uses multi-way split to reduce number of branches and hence computation and storage. Entropy based information gain was used to split data. The entire data set is divided into train and test set in 80:20 ratio. The DataSplitter class randomly splits the data input row as train or test using a random number generator .It returns the corresponding datasets and class Labels.

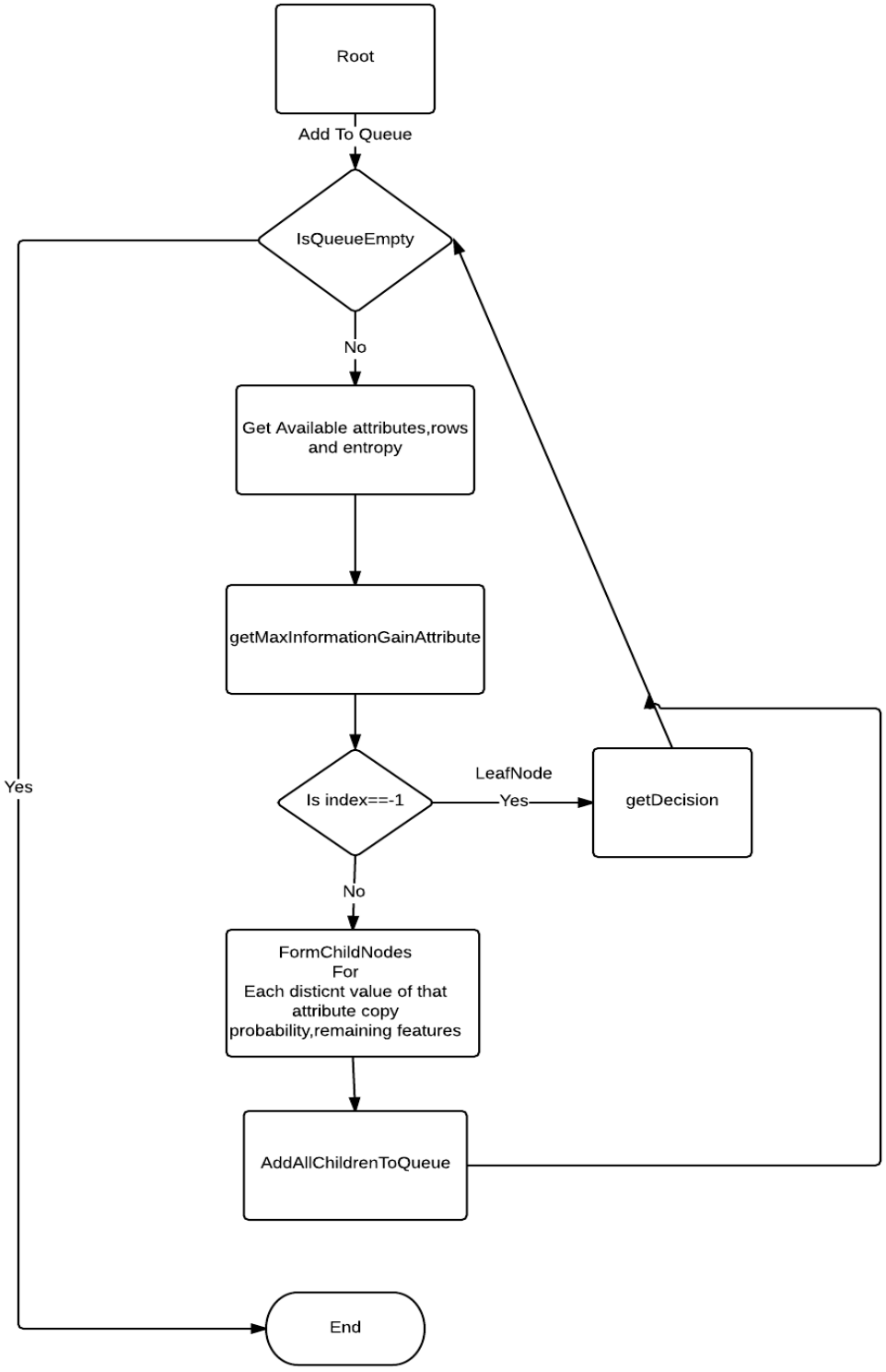
   The train data and labels are fed to DecisionTree class and buildTree method is called. DecisionTreeNode represents a node of tree containing list of children, remaining attributes decision if any and rows in that sub tree .A root node is first created and prior entropy is calculated for entire train set (before split). Information gain is then computed for each attribute and one giving highest gain is chosen for split. Next this is repeated for each distinct values of chosen attribute and split performed on one giving highest gain from remaining attributes. The process is continued till a leaf node is reached. To avoid stack overflow in case of a very large set level order traversal similar to BFS (breadth first search) is followed instead of recursion.

   A leaf node either has all values falling in the same class or has no more attributes remaining. If leaf node is such that no unused attributes remaining and all inputs do not fall in one class then the one which has the maximum probability is assigned as the decision for that class. Threshold can be specified for information gain. If gain is below threshold that node is treated as leaf. Pruning in this way avoids unnecessary computations.

   In testing the tree is traversed from root towards leaf along corresponding attribute values and class is assigned. This can be done using the traverseTree method which returns the decision.



Flow Chart-Decision Tree



2. Naïve Bayes implementation also divides the data into train and test sets. First the class priors are calculated for each class by the following formula

$$P(C) = \frac{N_c}{N}$$

This is done by iterating over the class column and finding the number of elements belonging to each class followed by taking ratios. This information is stored as HashMap with class label as key and probability as value.

Assuming independence of attributes we calculate the occurrence of each attribute value corresponding to each class i.e. frequency based probabities are obtained. This is also stored in map which records each value of each of the features in the training set and their probabilities.

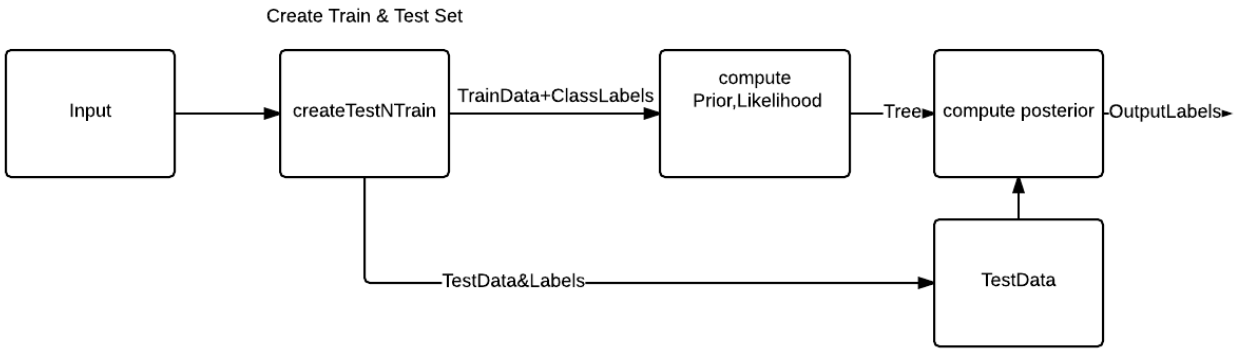
Each test input is used to calculate posterior probability of data belonging to a particular class. The class whose posterior is maximum is chosen as the class label.

To calculate posterior each of the features' value is read from input tuple and corresponding probability is obtained from the map. Multiplying features we get the likelihood of input which is multiplied prior to get posterior. In case of missing values use 1/ (feature-size +1) for that feature and also update the other probabilities accordingly.

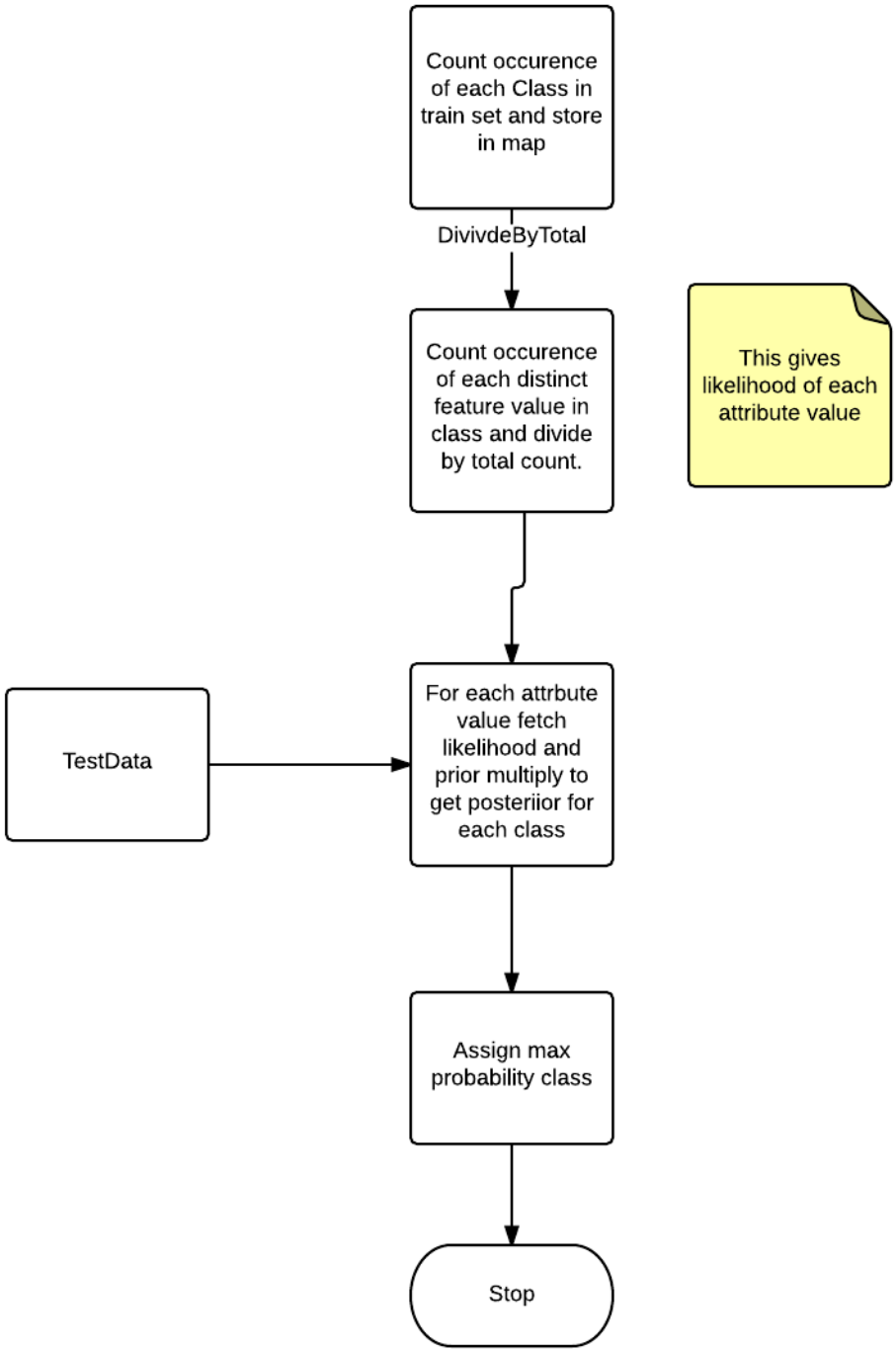
Likelihood

$$P(x|C_i) = P(x_1|C_i) * P(x_2|C_i) \dots P(x_d|C_i)$$

$where$
$d = no\ of\ attributes$
$P(xj|Ci) = \frac{n_{i,j}}{n_i}$ … relative frequency



Flow Chart-Naïve Bayes



3. Choice of features

   The given dataset has large number of features out of which only certain feature give better results than others. It is expensive to use all features 5 useful features were selected from among available. The type of method used for classification also affects choice of features.

   Encounter_Id and Patient_nbr are mostly unique and hence have been omitted.

   Methods like Naïve Bayes rely on counts to obtain probabilities which are sued for classification through the MAP (Maximum Aposteriori Probability) approach .Here categorical features become the default choice over numeric ones. However some numeric attributes that showed variation within range were also selected.

   Except for diabetes parameters and medicines almost all others were used.
   Medical-Specialty can be an attribute that can help in classification but was omitted due to large number of missing values.

   Decision Tree can deal with both categorical and numeric data. A multi-way split approach was followed. Experimentally, it was found that categorical data was a better feature than the numeric ones. Many numeric features are largely static with little variation/less information (not useful for classifying) hence were omitted from use. Remaining-Attributes is a Set that denotes available attributes at each node in tree.

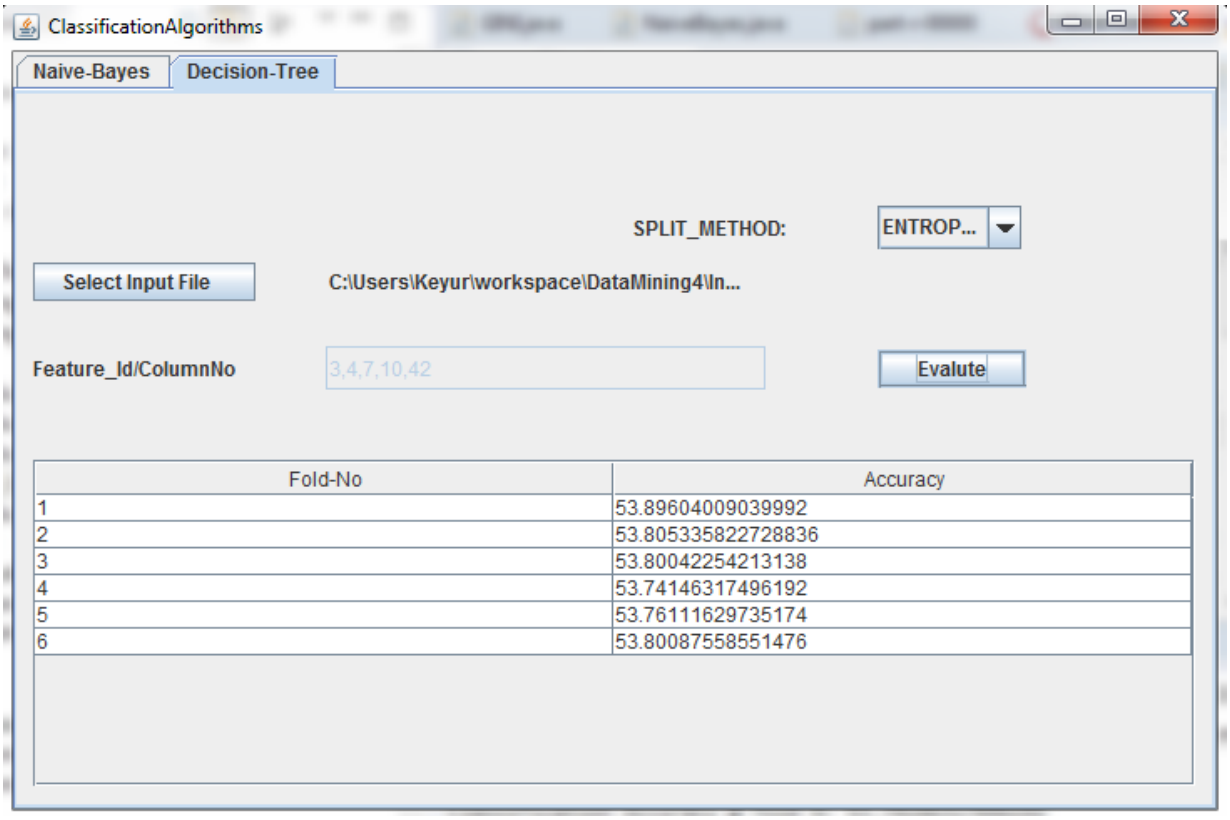
   Features used –Race, Gender, Insulin, num_procedures, time_in_hospital

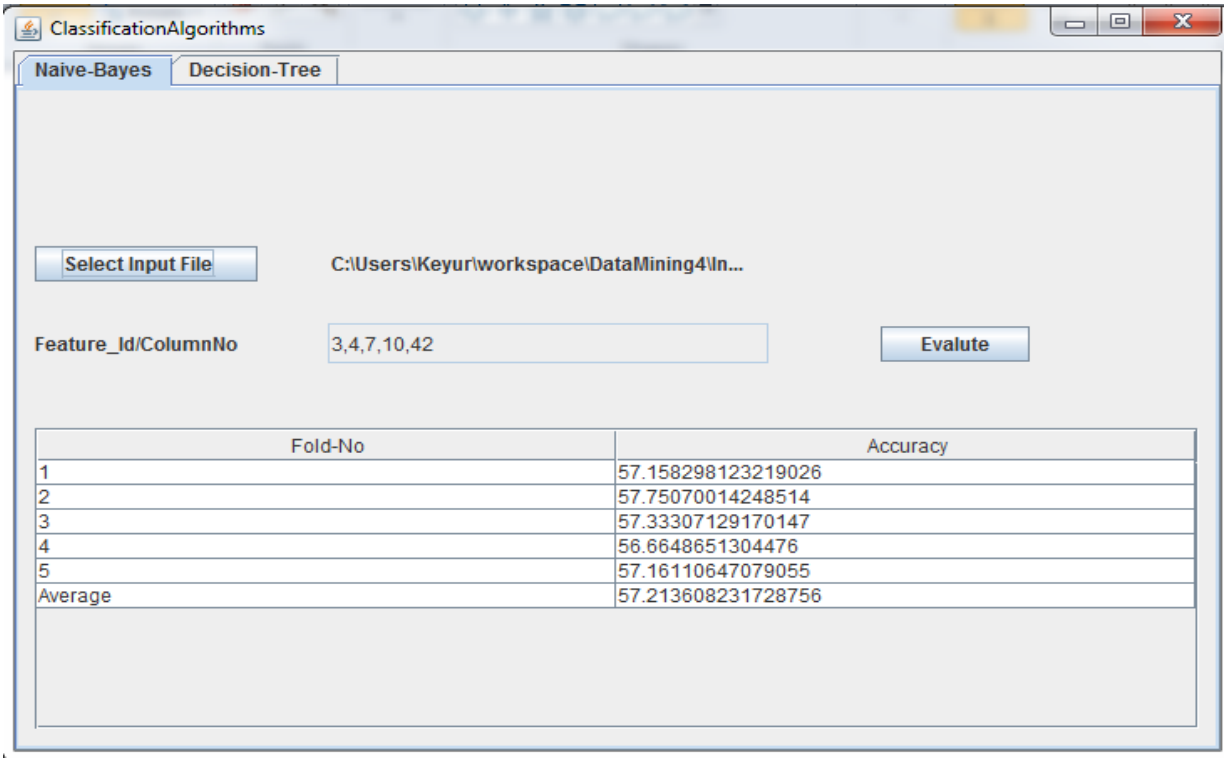
   Decision Tree can deal with both categorical and numeric data. A multi-way split approach was followed. Experimentally, it was found that categorical data was a better feature than the numeric ones. Many numeric features are largely static with little variation/less information (not useful for classifying) hence were omitted from use. Remaining-Attributes is a Set that denotes available attributes at each node in tree.

4. Cross-Validation
   Five-fold cross validation was performed by using random splits. The function is generic and supports up to k splits. The accuracy for each was calculated through which average accuracy was found.

5. Accuracy
   Accuracy is calculated by measuring the number of correctly classified outputs. Thus ratio of number of correctly classified outputs to size of dataset is used.

GUI


ClassificationAlgorithms
Naive-Bayes
Decision-Tree
SPLIT_METHOD:
ENTROP...
Select Input File
C:\Users\Keyur\workspace\DataMining4\In...
Feature_Id/ColumnNo
3,4,7,10,42
Evalute
Fold-No
Accuracy
1
53.89604009039992
2
53.805335822728836
3
53.80042254213138
4
53.74146317496192
5
53.76111629735174
6
53.80087558551476


ClassificationAlgorithms
Naive-Bayes
Decision-Tree
Select Input File
C:\Users\Keyur\workspace\DataMining4\In...
Feature_Id/ColumnNo
3,4,7,10,42
Evalute
Fold-No
Accuracy
1
57.158298123219026
2
57.75070014248514
3
57.33307129170147
4
56.6648651304476
5
57.16110647079055
Average
57.213608231728756

RESULTS

Decision Tree

Information Gain
Results (Java console):

BuildTime-354 ms avg
Classification Accuracy @ fold 0: 54.14660508990862
Classification Accuracy @ fold 1: 53.95273424065248
Classification Accuracy @ fold 2: 53.967474082444845
Classification Accuracy @ fold 3: 53.95764752124994
Classification Accuracy @ fold 4: 53.9723873630423
Average Classification Accuracy: 53.99936965945964

No of Attributes	Attributes	Accuracy
5	race,gender,insulin,age,num_procedures,time_in_hospital	53.99936965945964
3	race,gender,	53.99090695876679
6	race,gender,insulin,age,num_procedures,time_in_hospital,age	53.96202877519727
20	-	49.83458761830916

ID3 using GINI

Tree Build Time 600 ms avg
Classification Accuracy @ fold 1: 53.89604009039992
Classification Accuracy @ fold 2: 53.7660295779492
Classification Accuracy @ fold 3: 53.82990222571611
Classification Accuracy @ fold 4: 53.756203016754284
Classification Accuracy @ fold 5: 53.76111629735174
Average Classification Accuracy: 53.80185824163425

Naïve Bayes

Results (Java console):

Learn time 618 ms avg
Classification Accuracy @ fold 0: 57.114080770364545
Classification Accuracy @ fold 1: 57.62295484695131
Classification Accuracy @ fold 2: 57.224979118557464
Classification Accuracy @ fold 3: 56.57642607969341
Classification Accuracy @ fold 4: 57.05301429764654
Average Classification Accuracy: 57.11829102264265

ANALYSIS

In this section, we discuss the results obtained, observations drawn

Results

1) Naïve Bayes took more time for training due to more features being used.
2) Naïve Bayes shows better accuracy than decision tree under given conditions.
3) Reducing the number of attributes actually increased accuracy for decision tree meaning not all features are useful for classification. In fact features that increase overlap of data in feature space could result in poor classification.

4) Similarly in case of numeric features splitting multiple times leaves very few train input in node and this insufficient data can result in erroneous classification.
5) Changing the number or set of attributes has almost no contribution to accuracy in decision tree.
6) Prior probabilities could be used to decide features.

DISCUSSION

In this section, we discuss the advantages, challenges and possible improvements of the MCL algorithm.

ADVANTAGES

1) Naïve Bayes is simple and easy to implement.
2) It easily supports implementation on computing grid/distributed approach useful for large data set. Basically it needs a key/feature and its count so Map Reduce can be used.
3) Decision Tree is a hard classifier unlike Naïve Bayes.

CHALLENGES OF CLASSIFICATION ALGORITHMS

Choosing the correct features in decision tree.

1) Deciding the prune threshold on information gain.
2) Use of multi-way against binary split.

IMPROVING

1) Reduce the number iterations through proper pruning threshold.
2) Using Map-Reduce framework for Naïve Bayes.
3) Using file backed in memory database like HSQLDB on standalone systems or H-Base for distributed systems for decision tree.
4) Better performance evaluation measures like f-measure, kappa etc. should be used to describe model performance.

CONCLUSION

REFERENCES

[1] https://piazza.com/class_profile/get_resource/hz34ir6d54t4j7/i2zl1qm28m74mn

(Naïve Bayes)

[2] http://datamining-ttmk52.googlecode.com/svn/trunk/dangbk/DecisionTree2.pdf