CSE 601 : DATA MINING

CLUSTERING ALGORITHMS

TEAM 22

Name	Email	UB ID
Akash Yeleswarapu	akashyel@buffalo.edu	50207826
Maheedhara Achalla	maheedha@buffalo.edu	50207395

1. kNN CLASSIFICATION

OBJECTIVE:

Implementing Nearest Neighbor classification algorithm on the two given datasets and evaluate the results using 10fold cross validation using the performance metrics Accuracy, Precision, Recall and F-1 Measure

DEFINITION:

k-nearest neighbor algorithm is a non-parametric, instance based learning or lazy learning algorithm that can be used for both classification and regression.

ALGORITHM:

1. Store the training records and extract the target labels of each record.
2. Normalize the values of each attribute of the training records.
3. Compute the Euclidean distances between the test sample and all of the training records.
4. If there is a categorical attribute, add 1 to the distance, if categorical attribute of training and Test record are different. If they are same do not add anything.
5. Sort the distances and choose the "K" nearest training records to the test record.
6. Apply the voting scheme and the choose the majority output of the target labels of training Records and assign it to the test record.

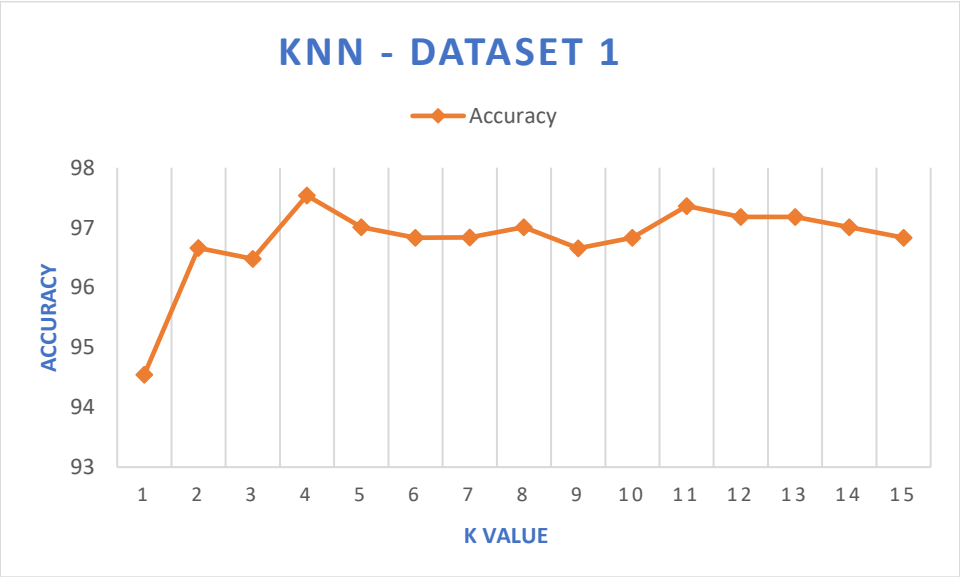
RESULTS:

We have implemented kNN algorithm on the two given data sets. Results given below are for the optimal value of K.

Project3_dataset1 :
K-Value 5
Accuracy: 97.53759398496238 %
Precision: 0.9916666666666666
Recall: 0.9410430757935089
F-1 Measure: 0.964477407807227

We played around with different values of K and tabulated the results. Table is given below :

K	Accuracy
2	94.545
3	96.66
4	96.481
5	97.537
6	97.008
7	96.832
8	96.835
9	97.008
10	96.657
11	96.829
12	97.359
13	97.18
14	97.18
15	97.008
16	96.832



Project3 dataset2 :

K-Value 30

Accuracy: 73.39037927844589

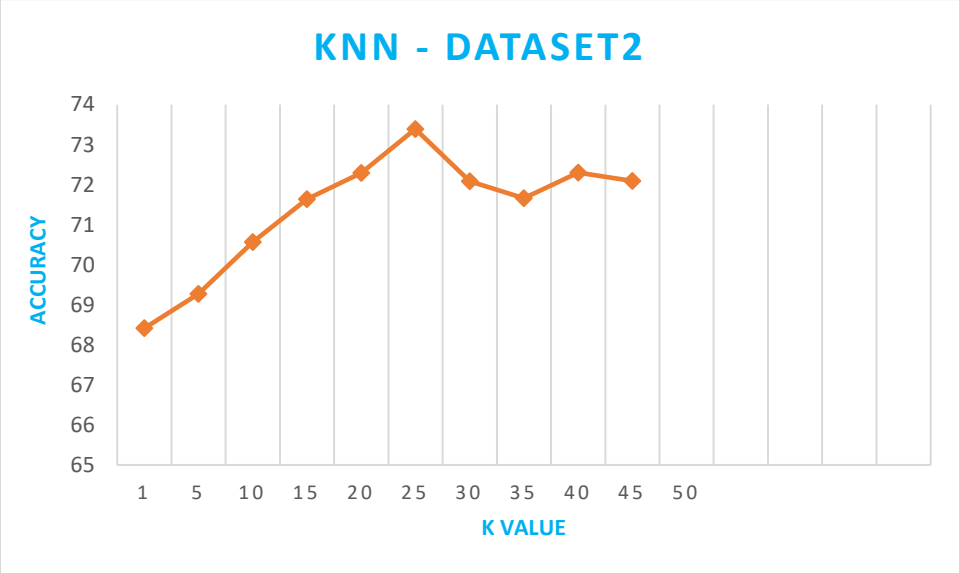
Precision: 0.7151839826839826

Recall: 0.42039830783251836

F-1 Measure: 0.5180310756647062

We played around with different values of K and tabulated the results. Table is given below:

K	Accuracy
1	64.292
5	68.413
10	69.269
15	70.568
20	71.637
25	72.289
30	73.39
35	72.086
40	71.66
45	72.298
50	72.09



OUR OBSERVATIONS:

We observe that boundaries between the two classes becomes smoother with increase in value of K. If we increase the value of K to the infinity, then algorithm either classify the test data as either 1 or 0 based on the majority population of 1 and 0. kNN algorithm performed well with the dataset1 than with dataset2, it may be due to the fact that data is nicely distributed i.e similar records are closely packed.

Parameter Tuning(K):

If very small value of K chosen, model will be sensitive to noise and if very high value of K is chosen , outliers will impact the decision. It is very important to choose the optimal value of K. Best value of "K" can be obtained from the validation error VS K value curve, which is the exact opposite of the plots given above (i.e. Accuracy vs K value) curve. Choose the value of K where there is a minimum in validation error curve or maxima in the accuracy curve.

Cross Validation:

As we are using test set for hyperparameter tuning, there are high chances of overfitting. To avoid overfitting, we used K-fold cross validation. In an k-fold validation, data is split k times and test data of each split is assessed according to the training data of that split. Accuracy, Precision and other performance measures are calculated for each split and then the average accuracy is considered as one of the model performance measures.

Preprocessing :

Normalization:

We have normalized the entire data using Min-Max normalization to make sure that different type of features (categorical and continuous) have similar impact in determining the final class.

PRO :

- Very simple to implement.
- Handles multi class classification.
- It has zero to little training time.
- Effective for large data if the similar records are closely packed.

CONS :

- High computational cost.
- High memory cost.
- Algorithm is sensitive to K parameter.
- Algorithm can suffer from skewed class distributions.
- Choosing k value is expensive
- Different distance metrics leads to different results

2. DECISION TREE

OBJECTIVE:

Implementing Decision Tree algorithm on the two given datasets and evaluate the results using 10fold cross validation using the performance metrics Accuracy, Precision, Recall and F-1 Measure

DEFINITION:

Decision Trees are eager learners (They build the model early) and one of the widely used classification algorithms. In these tree structures, leaves represent class labels and branches represent conjunction of features that lead to those class labels. We usually work top-down, by choosing a variable at each step that best splits the set of items. There are several algorithms to choose the best split at each step. They are:

1) Gini Index
2) Entropy
3) Error calculation

We chose Gini Index to split the items at each node.

ALGORITHM:

1. In the training data set, compute the GINI values of all the attributes. Find the attribute value with least GINI value. Make this attribute value as a root node.
2. Now split the all items in the training set based on the root node.
3. Different decisions are taken for different types of attributes.
4. If the attribute is continuous, put all the data records that have an attribute value less than or equal to root node value to the left of the parent node. Similarly put all the records that have an attribute value greater than root node value to the right of parent node.
5. If the attribute is categorical, put all the data records that have a same attribute value as root node value to the left of the root node. Otherwise put those data records to the right of the root node.
6. Among the data records in the left and right nodes, choose the best attribute values based on GINI values and split them again on these attribute values.
7. Grow the tree until it reaches the terminal nodes.
8. There are three conditions to find out the terminal nodes.
    i) If all the records corresponding to the node has same target class label.
    ii) If values across all the attributes are same in the dataset of that node.

iii) If there is a best split such that one of the child node is empty

9. Label of leaf node corresponds to maximum of all the possible target labels at that node.
10. To predict the class for the test data, traverse along the path based on the attribute values of the test data till it reached the leaf node. Value at the leaf node will be the predicted value of the test record.

Choosing Best Features:

At every possible node split, consider all the possible attribute values that provides the maximum purity i.e the minimum GINI value. This attribute value is chosen as a best value for the node split.

RESULTS:

We have implemented Dataset algorithm on the two given data sets. Results are given below :

Project3_dataset1:

Accuracy for iteration 1: 96.61016949152543
Precision for iteration 1: 95.83333333333334
Recall for iteration 1: 95.83333333333334
F-1 Measure for iteration 1: 95.83333333333334

Accuracy for iteration 2: 90.47619047619048
Precision for iteration 2: 85.0
Recall for iteration 2: 85.0
F-1 Measure for iteration 2: 85.0

Accuracy for iteration 3: 93.44262295081968
Precision for iteration 3: 82.35294117647058
Recall for iteration 3: 93.33333333333333
F-1 Measure for iteration 3: 87.5

Accuracy for iteration 4: 93.44262295081968
Precision for iteration 4: 95.45454545454545
Recall for iteration 4: 87.5
F-1 Measure for iteration 4: 91.30434782608695

Accuracy for iteration 5: 96.61016949152543
Precision for iteration 5: 95.23809523809523

Recall for iteration 5: 95.23809523809523
F-1 Measure for iteration 5: 95.23809523809523
Accuracy for iteration 6: 96.61016949152543
Precision for iteration 6: 96.42857142857143
Recall for iteration 6: 96.42857142857143
F-1 Measure for iteration 6: 96.42857142857143

Accuracy for iteration 7: 96.61016949152543
Precision for iteration 7: 95.65217391304348
Recall for iteration 7: 95.65217391304348
F-1 Measure for iteration 7: 95.65217391304348

Accuracy for iteration 8: 90.47619047619048
Precision for iteration 8: 83.33333333333334
Recall for iteration 8: 83.33333333333334
F-1 Measure for iteration 8: 83.33333333333334
Accuracy for iteration 9: 94.91525423728814
Precision for iteration 9: 96.7741935483871
Recall for iteration 9: 93.75
F-1 Measure for iteration 9: 95.23809523809523

Accuracy for iteration 10: 87.93103448275862
Precision for iteration 10: 83.33333333333334
Recall for iteration 10: 86.95652173913044
F-1 Measure for iteration 10: 85.1063829787234

Total Accuracy: 93.71245935401686
Total Precision: 90.94005207591132
Total Recall: 91.30253623188406
Total F-1 Measure: 91.06343332892823

Project_dataset2:
Accuracy for iteration 1: 66.19718309859155
Precision for iteration 1: 63.63636363636363
Recall for iteration 1: 63.63636363636363
F-1 Measure for iteration 1: 63.63636363636363

Accuracy for iteration 2: 67.14285714285714
Precision for iteration 2: 52.0
Recall for iteration 2: 54.166666666666664
F-1 Measure for iteration 2: 53.06122448979592

Accuracy for iteration 3: 65.71428571428571
Precision for iteration 3: 61.29032258064516
Recall for iteration 3: 61.29032258064516
F-1 Measure for iteration 3: 61.29032258064516

Accuracy for iteration 4: 67.64705882352942
Precision for iteration 4: 57.692307692307686
Recall for iteration 4: 57.692307692307686
F-1 Measure for iteration 4: 57.692307692307686

Accuracy for iteration 5: 66.66666666666666
Precision for iteration 5: 64.51612903225806
Recall for iteration 5: 62.5
F-1 Measure for iteration 5: 63.49206349206349

Accuracy for iteration 6: 66.66666666666666
Precision for iteration 6: 60.0
Recall for iteration 6: 62.06896551724138
F-1 Measure for iteration 6: 61.016949152542374

Accuracy for iteration 7: 65.71428571428571
Precision for iteration 7: 52.0
Recall for iteration 7: 52.0
F-1 Measure for iteration 7: 52.0

Accuracy for iteration 8: 65.71428571428571
Precision for iteration 8: 47.82608695652174
Recall for iteration 8: 47.82608695652174
F-1 Measure for iteration 8: 47.82608695652174

Accuracy for iteration 9: 67.64705882352942
Precision for iteration 9: 59.25925925925925
Recall for iteration 9: 59.25925925925925

F-1 Measure for iteration 9: 59.25925925925925

Accuracy for iteration 10: 46.3768115942029
Precision for iteration 10: 29.629629629629626
Recall for iteration 10: 30.76923076923077
F-1 Measure for iteration 10: 30.18867924528302

Total Accuracy: 64.54871599589009
Total Precision: 54.78500987869852
Total Recall: 55.12092030782363
Total F-1 Measure: 54.94632565047824

CROSS VALIDATION:
To handle the overfitting of the data set, we have used K-fold cross validation technique where entire data set is divided into K folds, one of the folds is used as test data and other K-1 folds are used as training data.

INTERPRETAION:
When compared with the data set1, dataset 2 has very less accuracy around 64%. This could be due to the reason the data distribution is more nicer in the dataset1 than in the dataset2. If tree is fully grown, it may lead to overfitting. To avoid this, we can do the following :

1) Limiting the size of each size
2) Limiting the depth of the tree
3) Limiting proportion of true labels in the node.

PROS

1. Decision trees are eager learners.
2. This algorithm analyses all the intricacies in the data set.
3. Results are very easy to interpret.
4. Very good classification technique for rectilinear classification.

CONS

1. Deeper the node, complex the tree is. It causes over fitting.
2. Changes in the training data set will lead change in tree structure and effect the decision rules.
3. Model will not generalize much for the new data.
4. Cannot handle non-rectilinear combination of data.

3 NAÏVE BAYES CLASSIFIER

OBJECTIVE:

Implementing Naive Bayes Classification on the given dataset and train the model by dividing the dataset into training and testing set by 10-fold cross validation.

DEFINITION:

Naive Bayes algorithm is a simple probabilistic classifier which can be built using Bayes theorem based on the assumption that all the features are independent of each other.

ALGORITHM:

1. Build a training and testing set based on the type of cross-validation algorithm we choose.
2. Find the mean and standard deviation on the training set for each attribute
3. Extract the class labels and keep a count of the number of occurrences of each label.
4. Find probabilities of each attribute in the test set and use Bayes Theorem to calculate the class posterior probability. The formula is

$$P(h/x) = P(x/h)\ P\ (h)\ /p(x)$$

Where

- $P(h|x)$ is the posterior probability of class
- $P(h)$ is class prior probability
- $P(x|c)$ is the descriptor posterior probability
- $P(x)$ is the descriptor prior probability

5. Predict the class label for each test set based on this probability (by taking the maximum probability of that set for a class label)
6. Calculate accuracy and other values based on the predicted values by comparing them with the true labels.

HANDLING DATA:

We have datasets with mixed variable types and naïve bayes' approach can't be generalized on all the types. We'll have to handle them separately. Following shows our approach in handing continuous and categorical variables.

CONTINUOUS VARIABLES:

For continuous variables, we need to calculate mean and standard deviation for the continuous attributes, then calculating the gaussian distribution for retrieving the probability for that particular attribute.

CATEGORICAL VARIABLES:

For categorical variables, we need to calculate the descriptor posterior probability and class prior probability and dividing it by descriptor prior probability. We can simply ignore the descriptor prior probability and just compare the numerators since the denominator is going to be the same.

ZERO FREQUENCY PROBLEM:

There is a very likely chance of getting zero probability when a particular attribute value doesn't occur with every class value. This would result in zero probability for the entire attribute since we multiply the independent probabilities. One way to avoid this is by adding a count to all the frequencies in the frequency tables of attributes. There are quite a few well known methods to handle this like Laplace-estimate and M-estimate which would take care of this scenario.

RESULTS:

We have implemented naïve bayes algorithm on the two given data sets by running them on 10-fold cross validation.

Project3_dataset1 :

Accuracy: 93.48997493734335
Precision: 0.9178709362532891
Recall: 0.9044426356826325
F-1 Measure: 0.9100318381263209

Project3_dataset2 :

Accuracy: 70.34690101757631
Precision: 0.5709275640080593
Recall: 0.6159396451501715
F-1 Measure: 0.5867396458138145

PROS :

- Very simple to implement and understand.
- Can be used on both huge and small datasets with any number of attributes
- It's insensitive to irrelevant parameters
- It's quite fast when compared to other classifiers.

CONS :

- This algorithm assumes class conditional independence between all the attributes which will not be the case for all the datasets. So it can't be used on datasets with dependent features
- There's a chance for zero-frequency problem if not handled properly.

4.RANDOM FORESTS

OBJECTIVE:

Implementing Random forest classification algorithm on the two given datasets and evaluate the results using 10 fold cross validation using the performance metrics Accuracy, Precision, Recall and F-1 Measure

DEFINITION:

Random forests are ensemble learning methods for classification. They grow many decision trees. To classify a new test data, put the input vector down each of the trees in the forest. Each tree gives a classification, and we take the majority of the tree outputs and return it as label for the test data.

ALGORITHM :

1. Apply K- fold cross validation on the entire data set.
2. Instead of growing a single decision tree, we grow a forest of trees. Select "T" (10 for our case) number of trees per each fold of training data. Training data is chosen using the sampling with replacement (Boot strapping) so that training data for each decision tree is different in every fold.
3. Instead of taking all the 'M' features for splitting the items at each node, choose 'm' features (m<M) random and distinct features at each node split. In our case we chose m as 3
4. Test data is run against each of the decision trees of the forests, output is taken by taking majority of the labels returned by the each forest.

RESULTS :

Project_Dataset1 :

T=10

m=3

Accuracy for iteration 1 : 95.0
Precision for iteration 1 : 95.83333333333334
Recall for iteration 1 : 92.0
F-1 Measure for iteration 1 : 93.87755102040816

Accuracy for iteration 2 : 96.61016949152543
Precision for iteration 2 : 94.44444444444444

Recall for iteration 2 : 94.44444444444444
F-1 Measure for iteration 2 : 94.44444444444444
Accuracy for iteration 3 : 95.0
Precision for iteration 3 : 93.33333333333333
Recall for iteration 3 : 87.5
F-1 Measure for iteration 3 : 90.32258064516128

Accuracy for iteration 4 : 95.0
Precision for iteration 4 : 91.30434782608695
Recall for iteration 4 : 95.45454545454545
F-1 Measure for iteration 4 : 93.33333333333333

Accuracy for iteration 5 : 95.0
Precision for iteration 5 : 90.9090909090909
Recall for iteration 5 : 95.23809523809523
F-1 Measure for iteration 5 : 93.02325581395348

Accuracy for iteration 6 : 95.0
Precision for iteration 6 : 93.10344827586206
Recall for iteration 6 : 96.42857142857143
F-1 Measure for iteration 6 : 94.73684210526315

Accuracy for iteration 7 : 93.44262295081968
Precision for iteration 7 : 91.66666666666666
Recall for iteration 7 : 91.66666666666666
F-1 Measure for iteration 7 : 91.66666666666666

Accuracy for iteration 8 : 95.0
Precision for iteration 8 : 88.23529411764706
Recall for iteration 8 : 93.75
F-1 Measure for iteration 8 : 90.9090909090909

Accuracy for iteration 9 : 95.0
Precision for iteration 9 : 96.875
Recall for iteration 9 : 93.93939393939394
F-1 Measure for iteration 9 : 95.38461538461539

Accuracy for iteration 10 : 94.91525423728814

Precision for iteration 10 : 91.66666666666666
Recall for iteration 10 : 95.65217391304348
F-1 Measure for iteration 10 : 93.61702127659575

Total Accuracy: 94.3968046679633
Total Precision: 92.73716255731314
Total Recall: 93.60738910847607
Total F-1 Measure: 93.13154015995326

Project_DataSet2.txt
T=10
m=3
Accuracy for iteration 1 : 65.27777777777779
Precision for iteration 1 : 61.76470588235294
Recall for iteration 1 : 63.63636363636363
F-1 Measure for iteration 1 : 62.68656716417911

Accuracy for iteration 2 : 65.27777777777779
Precision for iteration 2 : 52.0
Recall for iteration 2 : 50.0
F-1 Measure for iteration 2 : 50.98039215686274

Accuracy for iteration 3 : 64.7887323943662
Precision for iteration 3 : 61.29032258064516
Recall for iteration 3 : 59.375
F-1 Measure for iteration 3 : 60.317460317460316

Accuracy for iteration 4 : 65.71428571428571
Precision for iteration 4 : 55.55555555555556
Recall for iteration 4 : 55.55555555555556
F-1 Measure for iteration 4 : 55.55555555555556

Accuracy for iteration 5 : 65.71428571428571
Precision for iteration 5 : 62.5
Recall for iteration 5 : 62.5
F-1 Measure for iteration 5 : 62.5

Accuracy for iteration 6 : 65.71428571428571

Precision for iteration 6 : 60.0
Recall for iteration 6 : 60.0
F-1 Measure for iteration 6 : 60.0

Accuracy for iteration 7 : 64.7887323943662
Precision for iteration 7 : 52.0
Recall for iteration 7 : 50.0
F-1 Measure for iteration 7 : 50.98039215686274

Accuracy for iteration 8 : 64.7887323943662
Precision for iteration 8 : 45.83333333333333
Recall for iteration 8 : 47.82608695652174
F-1 Measure for iteration 8 : 46.808510638297875

Accuracy for iteration 9 : 65.71428571428571
Precision for iteration 9 : 57.14285714285714
Recall for iteration 9 : 57.14285714285714
F-1 Measure for iteration 9 : 57.14285714285714

Accuracy for iteration 10 : 65.71428571428571
Precision for iteration 10 : 53.84615384615385
Recall for iteration 10 : 53.84615384615385
F-1 Measure for iteration 10 : 53.84615384615385

Total Accuracy: 65.34931813100829
Total Precision: 56.19329283408979
Total Recall: 55.98820171374519
Total F-1 Measure: 56.081788897822925

INTERPRETATION :

Random forests produce the better predictions when compared to single decision tree. This is evident from the performance metrics of both the single decision tree and random forests. As randomness generally helps in better generalization of the model. Results for each run will vary as Random forests generate random trees on random set of data.

BAGGING :

Bagging ensures that the N number of trees constructed for every split are not constructed on the same set of data and thus results in a better assumption for predicting labels for test data**.**

CHOICE OF HYPER-PARAMETER :

Reducing the value of 'm' reduces the correlation between two trees and strength of each individual tree while increase in value of increases both. Optimal value of "m" should be taken carefully using out of bag(oob) error rate.

PROS :

1. Free of overfitting issue which happens in the single decision tree.
2. Handles all types of attributes (continuous and categorical)
3. Runs efficiently on all data sets.
4. Generated forests can be saved for future use on other data.

CONS :

1. Computation is expensive when compared to single decision tree.
2. Not easy to visually interpret.

5. BOOSTING

OBJECTIVE:

Implementing Ada boosting classification algorithm on the two given datasets and evaluate the results using 10 fold cross validation using the performance metrics Accuracy, Precision, Recall and F-1 Measure

DEFINITION:

Boosting is an ensemble method for the classification. K number base learners are built, and target label is concluded based on all the learners. Decision trees are used as a base learner. This algorithm converts the weak learners into stronger ones.

ALGORITHM :

K fold cross validation is performed.

1. In each fold, initialize equal weight to each training record.
2. For training data, perform the bootstrap sampling with the weights of each record.
3. Build decision tree on sampled training data and predict the outputs.
4. Calculate the error and importance of each classifier using predicted outputs and target labels.
5. Update the weights of each record based on the error. (misclassified records weight will increase, and correctly classified records weight will decrease)
6. If there is high error, initialize the weights of each record to [1/N,1/N,1/N……..1/N]
7. Target class is the one which has most aggregated importance.

RESULTS :

Project_dataset1.txt

Boosting layers =10

Accuracy for iteration 1 : 96.61016949152543
Precision for iteration 1 : 95.83333333333334
Recall for iteration 1 : 95.83333333333334
F-1 Measure for iteration 1 : 95.83333333333334
Accuracy for iteration 2 : 93.44262295081968
Precision for iteration 2 : 89.47368421052632
Recall for iteration 2 : 89.47368421052632
F-1 Measure for iteration 2 : 89.47368421052632
Accuracy for iteration 3 : 95.0

Precision for iteration 3 : 87.5
Recall for iteration 3 : 93.33333333333333
F-1 Measure for iteration 3 : 90.32258064516128
Accuracy for iteration 4 : 93.44262295081968
Precision for iteration 4 : 91.30434782608695
Recall for iteration 4 : 91.30434782608695
F-1 Measure for iteration 4 : 91.30434782608695
Accuracy for iteration 5 : 96.61016949152543
Precision for iteration 5 : 95.23809523809523
Recall for iteration 5 : 95.23809523809523
F-1 Measure for iteration 5 : 95.23809523809523
Accuracy for iteration 6 : 93.44262295081968
Precision for iteration 6 : 93.10344827586206
Recall for iteration 6 : 93.10344827586206
F-1 Measure for iteration 6 : 93.10344827586206
Accuracy for iteration 7 : 95.0
Precision for iteration 7 : 95.65217391304348
Recall for iteration 7 : 91.66666666666666
F-1 Measure for iteration 7 : 93.61702127659575
Accuracy for iteration 8 : 95.0
Precision for iteration 8 : 93.75
Recall for iteration 8 : 88.23529411764706
F-1 Measure for iteration 8 : 90.9090909090909
Accuracy for iteration 9 : 95.0
Precision for iteration 9 : 96.875
Recall for iteration 9 : 93.93939393939394
F-1 Measure for iteration 9 : 95.38461538461539
Accuracy for iteration 10 : 94.91525423728814
Precision for iteration 10 : 95.65217391304348
Recall for iteration 10 : 91.66666666666666
F-1 Measure for iteration 10 : 93.61702127659575
Total Accuracy: 94.84634620727978
Total Precision: 93.4382256709991
Total Recall: 92.37942636076116
Total F-1 Measure: 92.88032383759631

Project_dataset2.txt
Boosting layers =10
Accuracy for iteration 1 : 66.19718309859155
Precision for iteration 1 : 63.63636363636363
Recall for iteration 1 : 63.63636363636363
F-1 Measure for iteration 1 : 63.63636363636363

Accuracy for iteration 2 : 66.19718309859155
Precision for iteration 2 : 52.0
Recall for iteration 2 : 52.0
F-1 Measure for iteration 2 : 52.0

Accuracy for iteration 3 : 65.71428571428571
Precision for iteration 3 : 61.29032258064516
Recall for iteration 3 : 61.29032258064516
F-1 Measure for iteration 3 : 61.29032258064516

Accuracy for iteration 4 : 66.66666666666666
Precision for iteration 4 : 55.55555555555556
Recall for iteration 4 : 57.692307692307686
F-1 Measure for iteration 4 : 56.60377358490566

Accuracy for iteration 5 : 66.66666666666666
Precision for iteration 5 : 62.5
Recall for iteration 5 : 64.51612903225806
F-1 Measure for iteration 5 : 63.49206349206349

Accuracy for iteration 6 : 67.64705882352942
Precision for iteration 6 : 62.06896551724138
Recall for iteration 6 : 62.06896551724138
F-1 Measure for iteration 6 : 62.06896551724138

Accuracy for iteration 7 : 66.66666666666666
Precision for iteration 7 : 54.166666666666664
Recall for iteration 7 : 52.0
F-1 Measure for iteration 7 : 53.06122448979592

Accuracy for iteration 8 : 65.71428571428571

Precision for iteration 8 : 47.82608695652174
Recall for iteration 8 : 47.82608695652174
F-1 Measure for iteration 8 : 47.82608695652174

Accuracy for iteration 9 : 67.64705882352942
Precision for iteration 9 : 59.25925925925925
Recall for iteration 9 : 59.25925925925925
F-1 Measure for iteration 9 : 59.25925925925925

Accuracy for iteration 10 : 66.66666666666666
Precision for iteration 10 : 56.00000000000001
Recall for iteration 10 : 53.84615384615385
F-1 Measure for iteration 10 : 54.90196078431373

Total Accuracy: 66.57837219394801
Total Precision: 57.43032201722536
Total Recall: 57.41355885207508
Total F-1 Measure: 57.414002030110986

Overfitting Issues :
To avoid overfitting, depth of the tree and size of the node need to be controlled. Model must be tried with different number of boosting layers to pick the best one to generalize the model.

PROS

1) Complex data can be easily classified.
2) Difficult records can be predicted in successive runs.
3) Weak learners can be used to predict the target.

CONS

1) Difficult to tune the model due to hyper parameters
2) Takes significant amount of time for training.
3) Slow to predict the target label.
4) Results are difficult to interpret.