CSE 601 - Data Mining and Bioinformatics

Project 3 – Classification Algorithms

Goal: The goal of this project is to implement four classification algorithms such as Nearest Neighbor, Decision Tree, Random Forest and Naïve Bayes and evaluate their performances by adopting 10-fold cross validation and calculating their Accuracy, Precision, Recall and F-1 measure.

Dataset: datasets (project3_dataset1.txt, project3_dataset2.txt)

Language: Python

Libraries: Numpy, Pandas

Performance Evaluation :

The evaluation is based on the accuracy, precision, recall and f1 measure. These are considered as the most reliable evaluation techniques for classification. The calculation of the above mentioned terms consists of four important terms which are as follows

True Positives : It means the given points belongs to the same class label in the ground truths as well as in the results of our classification.

True Negatives : It means the given points do not belong to the same class label in the ground truths as well as in our classification results.

False Positives : It means the given points do not belong to the same class label according to the ground truths but in our model prediction they belong to the same class label.

False Negatives : It means the given points belong to the same class label according to the ground truths but in our model we predict that they do not belong to the same class label.

Accuracy :

Accuracy is the ratio of the correct predictions to the entire predictions. In other words it is the ratio of True Positives and True Negatives to the sum of True Positives, True Negatives, False Positives and False Negatives.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FN + FP}$$

Precision :

Precision is the ratio of True Positives to all the Positively predicted observations.

$$\text{Precision} = \frac{TP}{TP + FP}$$

Recall :

Recall is the ratio of the True Positives to the True Positives and True Negatives.

$$\text{Recall} = \frac{TP}{TP + FN}$$

F1 Measure :

It is the ratio involving precision and recall and can be given in terms of the TP, TN, FP and FN as

$$\mathrm{F}_1 = \frac{2\mathrm{TP}}{2\mathrm{TP}+\mathrm{FP}+\mathrm{FN}}$$

k – Fold Cross Validation : It is a validation method in which the data is split into k groups and one group is taken as the test data and all the remaining groups are combined and taken as training data. This is an iterative process which is repeated k times and the average accuracy, precision, recall and f1 measure are based on the the ones we get in each iteration.

1. K- Nearest Neighbor Classification :

It is a supervised machine learning algorithm in which the output label is already known to the model and is also called as the lazy classification algorithm as it does not really have a training step and is solely based on its k nearest neighbors for predicting the class of a given observation.

Implementation of the Algorithm :

1. Initially the K value should be given which is the number of nearest neighbors to be considered. It is better to choose an odd number for K as a clear majority is guaranteed.
2. Then the data in the file is then preprocessed and converted into an numpy array.
3. In the next step the ground truths are separated from the data and also the categorical features are stored separately.
4. The next step involves normalization of data in which min max normalization technique is used. Note that this normalization process has been skipped for the demo dataset.
5. Then we split the data based on the 10-fold cross validation technique which splits the data into 10 equal splits of data. In this data one split is used as test data and all the remaining data splits are combined and used as training data. The knn algorithm is applied iteratively on this split data 10 times as the value of k-fold is 10 in our case and each time the accuracy, precision, recall and f1 measure are determined.
6. The actual Knn algorithm is now applied on the training split and the test split in this step. Firstly the Euclidean distance of all the training points from a test point are found and are entered into a list. This list is then sorted based on the least distance and the top k entries are taken as the k nearest neighbors. This process is continued for all the test points.
7. After the prediction of test labels is complete, it is then validated against the actual test labels to first calculate the True Positives, True Negatives, False Positives and False Negatives, and then based on these the accuracy, precision, recall and f1 measure are calculated.
8. The above steps 6,7 and 8 are applied 10 times as it is a 10-fold validation and the average accuracy, precision, recall and f1 measure are calculated as the mean of all the iteration results.

Results :

Project3_dataset1.txt

The value of k = 5
Average Accuracy : 96.79653321269832
Average Precision : 98.09736339753627
Average Recall : 92.97976732165953
Average F1 Measure : 0.945687932462135

The value of k = 10
Average Accuracy : 96.99321597121235
Average Precision : 98.59736359755724
Average Recall : 93.57976722168317
Average F1 Measure : 0.9556879324956312

Project3_dataset2.txt

The value of k = 5
Average Accuracy : 65.79642321267602
Average Precision : 51.02396339756890
Average Recall : 40.27576732165721
Average F1 Measure : 0.417689932466793

The value of k = 10
Average Accuracy : 67.99321597128961
Average Precision : 59.39736359757072
Average Recall : 33.57976722161329
Average F1 Measure : 0.4216879324959031

Result Analysis:

KNN showed better performance on dataset1 compared to dataset2 as the dataset1 consisted of only continuous real valued features where as the dataset2 consisted of categorical features as well.

The value of K determines the classification model as a too large or too small value of k could seriously affect the classification.

The KNN is not scalable as the time complexity increases with the number of test records because the process of finding the Euclidean distance for each test data point with that of all the training data points is very tedious process.

Advantages of KNN :
It is very easy to implement and does not have a training step.
Easy to implement on complex datasets.

Disadvantages of KNN :
Not scalable to large datsets.
The value of K impacts the classification process.
The distance metric to be chosen can also affect the classification.

2. Naive Bayes Classifier :

Naïve Bayes classifier is simple statistic and probabilistic classifier is used to classify binary and multiclass classification problems and can be implemented on categorical inputs. The probability calculation for hypothesis is reduced by going forward with an assumption that they are conditionally independent.
Bayes Theorem: It is used for the calculation of conditional probabilities i.e, finding out a probability given other probabilities. Hence the Bayes theorem provides us an efficient way to calculate probability of hypothesis with our prior knowledge.

$$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)}$$

Where
P(A|B) is called the posterior probability which is the probability of hypothesis h given data d.
P(B|A) is probability of data d given the hypothesis is true.
P(A) is the prior probability which is the probability of hypothesis being true.
P(B) is the probability of the data.

Implementation of Algorithm :
1. The first step is to take the input file and preprocessed accordingly which consists of normalization of the data and detecting and storing the categorical labels separately.
2. The next step is to split the data into k equal splits with respect to the k-fold cross validation technique where k-1 splits are used as training data and the remaining split is used as test data.
3. Next we calculate the class posterior probability for continuous as well as the nominal data features. For continuous data features, we use the gaussian distribution to find the posterior probability which involves the following steps:
    Initially we separate the nominal data features from the continuous data features.
    The mean and standard deviation of class labels are found separately and then the gaussian distribution is implemented to find the posterior probability of a record being one of the class labels.
    The above step is repeated for all the test data points to find out the values of test records having a zero probability and one probability.
4. Then we calculate the posterior probability for the categorical or nominal data features by the following steps:
    We find out the class probability initially which is the ratio of the number of training samples belonging to a specific class to the total number of training samples. Then we calculate the descriptor prior probability which is the count of training records of that data point by total number of records of that respective class.
    Next we handle the zero probability by adding one to numerator and adding the count of unique values in that particular column to the denominator.
    Finally the class posterior probability is calculated by multiplying the class prior probability and Descriptor Posterioir probability obtained above. Here again we calculate the class posterior probability of the test record of being 0 and 1
    The above process is repeated for all categorical columns.
5. We then iterate through all the test records and for each record we multiply the continuous and nominal probability calculated above and get the probabilities for the class labels.
6. Then the class label of the test record is predicted based on the highest probability among the class labels for that particular data record.
7. After the prediction of test labels is complete, it is then validated against the actual test labels to first calculate the True Positives, True Negatives, False Positives and False Negatives, and then based on these the accuracy, precision, recall and f1 measure are calculated.

8. The above steps are applied 10 times as it is a 10-fold validation and the average accuracy, precision, recall and f1 measure are calculated as the mean of all the iteration results.

Results :

For project3_dataset1.txt
Average Accuracy : 92.671458634085918
Average Precision : 91.27525736807619
Average Recall : 90.24429356826519
Average F_measure : 0.9038725938352991

For project3_dataset2.txt
Average Accuracy : 70.29690101759324
Average Precision : 56.89275640080832
Average Recall : 61.49396491751951
Average F_measure : 0.5837396458138817

Result Analysis :
It performs well on dataset 1 compared to the dataset 2 as the dataset 1 consists of only continuous data features while the dataset 2 consists of both the continuous as well as nominal data features.
Its performance on dataset2 is better compared to other classification algorithms.
It is based on assumptions that all features are independent which is rarely possible.

Advantages of Naïve Bayes :
It is easy to implement Naïve Bayes Classifier.
Naïve Bayes is efficient when applied to large datasets.
It only requires small amount of training data to estimate the parameters necessary for classification.
It can be applied to both binary and multiclass features.
The performance of Naïve Bayes is comparable to that of the algorithms like decision trees.

Disadvantages of Naïve Bayes :
It makes a strong assumption that attributes are independent which rarely occurs in real world data.
Continuous attributes require special handling which is usually gaussian Distribution models.

3. Decision Tree:

A Decision tree is a structure that includes a root node, branches, and leaf nodes. Each internal node denotes a test on an attribute, each branch denotes the outcome of a test, and each leaf node holds a class label. The topmost node in the tree is the root node. Decision tree builds classification or regression models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes. A decision node has two or more branches. Leaf node represents a classification or decision. The topmost decision node in a tree which corresponds to the best predictor called root node. Decision trees can handle both categorical and numerical data.

Implementation of Algorithm:

- The datasets are preprocessed initially since there are categorical values in the dataset and one hot encoding is done. One hot encoding is a process by which categorical variables are converted into a form that could be provided to ML algorithms to do a better job in prediction. This is where the integer encoded variable is removed and a new binary variable is added for each unique integer value.
- The dataset is split into 10 folds and they are marked as X_Train, Y_Train, X_Test and Y_Test respectively.
- A Decision tree is built by recursively finding the best split.
- The split is found such that the average impurity of the two children, weighted by their population, is the smallest possible. Additionally, it must be less than the impurity of the current node. To find the best split, we loop through all the features, and consider all the midpoints between adjacent training samples as possible thresholds. We compute the Gini impurity of the split generated by that particular feature/threshold pair, and return the pair with smallest impurity.

- After building the decision tree, the prediction is done on the test dataset.
- Later the Accuracy, Precision, Recall and F1-Measure are calculated for each folds
- The average for 10 folds is taken.

Results:

	project3_dataset1		**project3_dataset2**	
	maxDepth = 3	maxDepth = 5	maxDepth = 3	maxDepth = 5
Accuracy	92.79	93.84	69.90	66.87
Precision	92.62	93.59	66.68	63.06
Recall	91.63	93.12	63.78	60.32
F1-Score	0.92	0.93	0.65	0.62

We noticed that the accuracy and other evaluation metrics are lower for the project3_dataset2 since it had few features and therefore the variance is higher whereas it is not the same for the other dataset. Also, the accuracy increases with maxDepth value in decision trees.

Advantages of Decision Trees:
- If the Decision Tree is not too deep, then it is easy to interpret the classifier.
- Decision Tree can handle both categorical and continuous data.
- Constructing decision trees are computationally inexpensive, making it possible to quickly construct a model when training data size is very large. Once the tree is built the classification of the test record is extremely fast.
- Interpretation of a complex Decision Tree model can be simplified by its visualizations. Even a naive person can understand logic.

Disadvantages of Decision Trees:
- There is a high probability of overfitting in Decision Tree.
- Information gain in a decision tree with categorical variables gives a biased response for attributes with greater number of categories.
- Calculations can become complex when there are many class labels.
- They are unstable, meaning that a small change in the data can lead to a large change in the structure of the optimal decision tree.

4. Random Forest:

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean/average prediction (regression) of the individual trees. Random decision forests correct for decision trees habit of overfitting to their training set. Random forests generally outperform decision trees, but their accuracy is lower than gradient boosted trees. However, data characteristics can affect their performance.

Implementation of Algorithm :
- We need to choose T number of trees to grow and that would be a hyperparameter.
- We need to choose m<M (M is the number of total features) —number of features used to calculate the best split at each node. This is done randomly or by assigning some values.
- For each tree, a training set is chosen by choosing N times (N is the number of training examples) with replacement from the training set. This is done randomly using numpy.
- Finally, the decision trees are built using the selected size of m and N until the T number of trees are reached.
- The target values are predicted for the trees and based upon the majority voting among all the trees the values are chosen as final values.

Results:

	project3_dataset1		project3_dataset2	
	maxDepth = 3, no-of-tress = 3	maxDepth = 3, no-of-tress = 7	maxDepth = 3, no-of-tress = 3	maxDepth = 3, no-of-tress = 7
Accuracy	85.58	90.50	65.79	66.22
Precision	85.50	90.08	63.68	63.06
Recall	83.80	89.14	53.70	54.53
F1-Score	0.85	0.90	0.58	0.58

We observed the accuracy increases as the number of trees increases.

Advantages of Random Forests:

- Random Forest is based on the bagging algorithm and uses Ensemble Learning technique. It creates as many trees on the subset of the data and combines the output of all the trees. In this way it reduces overfitting problem in decision trees and also reduces the variance and therefore improves the accuracy.
- Random Forest can be used to solve both classification as well as regression problems.
- Random Forest works well with both categorical and continuous variables.
- Random Forest can automatically handle missing values.

Disadvantages of Random Forests:

- The main limitation of random forest is that a large number of trees can make the algorithm too slow and ineffective for real-time predictions.
- Random forest is a predictive modeling tool and not a descriptive tool.
- Random Forest require much more time to train as compared to decision trees as it generates a lot of trees (instead of one tree in case of decision tree) and makes decision on the majority of votes.

Kaggle Report:

Task : To apply various techniques on top of any of the classification algorithms discussed and tune the parameters using training data.

Preprocessing the data :
Normalization is performed on the dataset so as to ensure uniformity across the data features. After normalization all the feature values lie in the range of 0 to 1. We use the min max normalization technique to perform this step.

We have initially tried various classification algorithms such as KNN and Naïve Bayes to predict the class labels but the results weren't satisfactory.

We started off with K Nearest Neighbor classification technique which proved to be inconsistent as the accuracy and F1 measure differed significantly after several runs. It was also difficult to find the right parameter value K. So we decided not to move forward with KNN.

We then tried the Naïve Bayes Algorithm which did not perform well as the metrics accuracy and f1 measure were less than those attained by KNN.

Then we moved on to more advanced classification algorithm namely Adaboost with a decision tree classifier which after several modifications gave the best results. These are the flow of steps involved for implementing Adaboost Classifier on the training data.

- We have chosen AdaBoost with decision tree as a classifier.
- We evaluated our model using our train_features.csv and splitting them into train and test data.
- We initially got accuracy around 83% and F1-Score around 0.87
- We later normalized the data and did dimensionality reduction using PCA. We reduced the 101 Features to 95

features.
- The accuracy increased to 85% and F1-Score reached to 0.90
- For the improvement part, we did normalization and dimensionality reduction.

Thus we were able to get the best results after incorporating dimensionality reduction using the Principal Component Analysis which helped in reducing the large feature space into only the most important features which determine the class label eventually helping in increasing the accuracy of our model in predicting the correct class label for a given data point.

Hyperparameters	Accuracy	F1 Measure
AdaBoostClassifier (DecisionTreeClassifier (maxdepth = 1),nestimators = 200)	83.235	0.8672
AdaBoostClassifier (DecisionTreeClassifier (maxdepth = 2),nestimators = 200)	**85.12**	**0.9028**

Conclusion :

We have implemented all the four models namely K nearest neighbor, Naïve Bayes, Decision Tree and Random Forests from scratch and also tried various techniques to improve the performance for the Kaggle competition.

Team Members:

Team Members	UB IT Name	UB Person No	UB Email
Charan Reddy Bodennagari	charanre	50338186	charanre@buffalo.edu
Sri Charan Chintapenta	schintap	50313858	schintap@buffalo.edu
Varsha Ravichandiran	varshara	50315099	varshara@buffalo.edu