An Ensemble of Classifiers using Dynamic Method on Ambiguous Data

Dnyaneshwar Kudande
D.Y. Patil College of Engineering, Pune, Maharashtra, India

***Abstract-* The aim of proposed work is to analyze the Instance Selection Algorithm first. There are Weighted Instance Selection algorithms are available such as wDROP3 (weighted Decremental Reduction Optimization Procedure 3), wRNN (weighted Reduced Nearest Neighbor), which reduces the Sample set applied. Then the multiclass Instance Selection is useful technique for reducing space and time complexity. This removes irrelevant, noisy, superfluous instances from Training Set. Then the multiclass problem is solved by considering number of two class problem thus designing multiple two class classifiers and its combined output produces the result for it. The Boosting is use for providing weight for each instance of training set. The Designing of ensemble of classifiers is to combine all classifiers and learn by reduced training set. There are different techniques are available for designing an ensemble such as Bagging (Bootstrap Aggregating), Boosting (ADABOOST) and Error Correcting Output Code (ECOC) etc. The output of ensemble is better than the individual classifiers. The approach is tested with few benchmark data sets. It is found that Classification accuracy in the case of wDROP3 algorithm lies between 70% to 87%, but in case of wRNN algorithm lies between 61% to 89% and the Generalization accuracy in the case of wDROP3 algorithm lies between 79% to 96%, but in wRNN algorithm it lies between 75% to 94%. Another observation, when increases number of Classifiers per Ensemble then accuracy improves by 0.5 to 1.5%.**

***Index Terms-* Data mining, data reduction, instance selection, Classification, weighted instance selection, Reduced Nearest Neighbor.**

I. INTRODUCTION

In recent years data are rapidly growing in size, there is difficult task for storing these data in well mannered. Now a day this huge data is used for discovering the hidden knowledge that is termed as Data mining. Data mining involves different domains. There are four major domains such as Association, Classification, Clustering and Regression. Association is a method for discovering interesting relations between variables in large databases. Clustering is a method of unsupervised learning, and a common technique for statistical data analysis used in many fields, including machine learning, data mining, pattern recognition, image analysis, information retrieval, and bioinformatics.

Regression is widely used for prediction and forecasting, where its use has substantial overlap with the field of machine learning. Regression analysis is also used to understand which among the independent variables are related to the dependent variable, and to explore the forms of these relationships. In restricted circumstances, regression can be used to infer causal relationships between the independent and dependent variables. Classification is the process of extracting patterns from large data sets by combining methods from statistics and artificial intelligence with database management. There are many Classification techniques are available such as Statistical approaches, soft Computing based approaches, Support Vector based approaches. An Ensemble of Classifiers is another approach in data mining. Ensemble of Classifiers is powerful technique in classification area. This is the more expressive concept than single classifier, because it is combination of individual predictions. Ensembles of classifiers offer promise in increasing overall classification accuracy. The availability of extremely large datasets has opened avenues for application of distributed and/or parallel learning to efficiently learn models of them.

Data mining techniques can be implemented rapidly on existing software and hardware platforms to enhance the value of existing information resources, and can be integrated with new products and systems as they are brought on-line. Commercial databases are growing at unprecedented rates. The accompanying need for improved computational engines can now be met in a cost-effective manner with parallel multiprocessor computer technology. Data mining algorithms embody techniques that have existed for at least 10 years, but have only recently been implemented as mature, reliable, understandable tools that consistently outperform older statistical methods.

Ensemble is useful for several data mining applications like Intrusion Detection, Customer Relation Management (CRM), and Medical Image Classification etc. In many practical learning domains, there is large supply of instances which is useful to solve the problem. Then the Classifier needs Instance Selection (IS) which try to pick a subset of instances that are relevant to the target concept. The instance selection is use for reducing unwanted, redundant instances from training set.

There are different instance selection algorithms are categories like ordered removal, nearest neighbor rule and random sampling. For uniformly distribution of instances into feature space, calculate weights of each instance of dataset. By applying weighted Instance Selection on training set, it reduces the size of training set and also reduces training time for classifier. Hence, Weighted Instance selection has become recent interest. Classification is a machine learning technique used to predict group membership for data instances. A classifier is function or an algorithm that maps all inputs to possible classes. This is called the classification of inputs. One of the data classification approach is (k- NN) k-nearest neighbor classifier. Classification using an instance-based classifier can be a simple matter of locating the nearest neighbor in feature space and labeling the unknown instance with the same class label as that of the located (known) neighbor.

Instance selection is a particular focusing task where the input is a set of instances and the outputs is a subset of input. Instance selection is to choose a subset of data to achieve the original purpose of a data mining application as if the whole data is used. Clearly, instance selection cleans the dataset that is in use: it removes irrelevant instances, as well noisy and redundant ones. For uniformly distribution of instances in feature space weights are use, so it a version of instance selection is weighted instance selection (wIS).

An ensemble of classifiers [3], [6] is set of classifiers by integrating multiple individual classifiers. An Ensemble of classifiers offers promise increasing overall classification accuracy. The availability of extremely large datasets has opened avenues for application of distributed and/or parallel learning to efficiently learn models of them. It is well known that an ensemble is able to outperform its best performing member if ensemble members make mistakes on different cases so that their predictions are uncorrelated and diverse as much as possible. The objectives of proposed work are containing an Analysis and Design of Weighted Instance Selection technique for k-NN (K-Nearest Neighbor) Classifiers. It is an approach to construct Ensemble of Classifiers more accurate, simpler, and efficient. It is Designing Ensemble of k-NN Classifiers using Weighted Instance Selection to reduce training error of an Ensemble. This approach is tested with few benchmark data sets and evaluates the performance of Ensemble of Classifiers in terms of Generalization and Classification Error. It contains comparison between Constructing Ensemble of Classifiers with Existing System.

II. WORKING OF THE SYSTEM

Working system of the project is visualized as follows. In the system given below the input data is labeled and numeric. Labeled data are those in which label is defined for each sample. The datasets from the different domain like heart disease, monk, a smoker and from marketing, manufacturing field like marketing, car, and vehicle database etc. are taken for the purpose of experimenting our system.

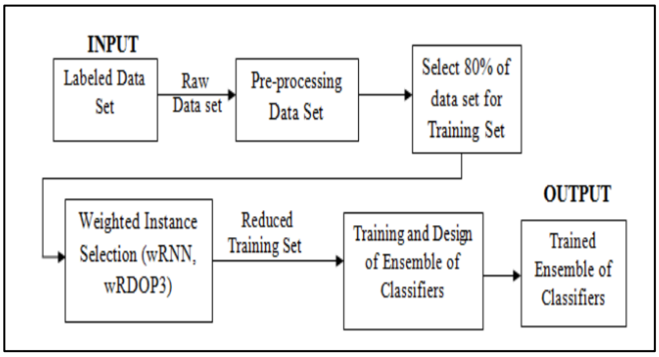

INPUT
Labeled Data Set
Raw Data set
Pre-processing Data Set
Select 80% of data set for Training Set
Weighted Instance Selection (wRNN, wRDOP3)
Reduced Training Set
Training and Design of Ensemble of Classifiers
OUTPUT
Trained Ensemble of Classifiers

Figure 1: Overview of the System

The design of Ensemble of Classifiers by using weighted instance selection System classify data in two phases, first is training and second is testing phase. The Design of system shows in figure 2. First of all, the dataset is taking from real world problem. Before doing instance selection of any dataset, do pre-process dataset. One part of pre- processed dataset use as training set, and another portion of dataset use as testing set. After doing instance selection output of it i.e. reduced training set, applying for constructing ensemble of classifiers. By referring figure, the labeled data set is the input to our System. The pre-processed data set is partitioned into training set and testing. Here 80% of dataset use for training set and remaining 20% for testing set.

Now pre-processed this training set, and input to Weighted Instance Selection Algorithm. There are two weighted instance selection algorithms are available such as wRNN(weighted Reduced Nearest Neighbor) and wDROP3(weighted Decremental Reduction Optimization Procedure 3). The output of, it is a reduced training set. It is more effective than training set. This reduced training set, as input for Training and Designing of ensemble of Classifiers.

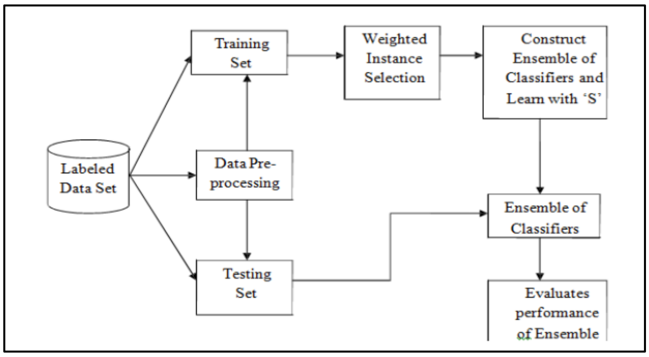

Training Set
Weighted Instance Selection
Construct Ensemble of Classifiers and Learn with 'S'
Labeled Data Set
Data Pre-processing
Ensemble of Classifiers
Testing Set
Evaluates performance of Ensemble

Figure 2: Architecture of the System

For the Designing System following algorithms are used:

Algorithm 1: Data pre-processing Algorithm.
Algorithm 2: Weighted Instance Selection Algorithm.
Algorithm 3: k-Nearest Neighbor Algorithm.
Algorithm 4: Design and Training of Ensemble of Classifiers.
Algorithm 5: Testing of Ensemble of Classifiers.

The Data pre-processing algorithm is major role play for constructing each and every System. The raw data is used as input to this algorithm.

Algorithm 1: Data Pre-processing Algorithm:

The input for system is numeric and with class label of each instance. A particular algorithm works on a specified format of data and thus it becomes mandatory to change the existing format of data. This is called Data pre- processing.

Input:

Data Set D={(x_1, y_1), (x_2, y_2), (x_n,y_n)}

Output:

(D”) Pre-processed Data Set.

```
Step 1: load D and find the size of data set in the terms of number of rows and columns, and initialize row and col.
for i=1 to col
    for j=1 to row
        Initialize c=i and d=i+1;
        $D[j,i] = \left(\frac{D[j,i] - minimum\ value\ of\ D[j]}{maximum\ value\ of\ D[j] - minimum\ value\ of\ D[j]}\right) * (d-c) + c$
    for loop end for loop end
Step 2: Return D”.
```

The pre-processed data set(D“) is output of Data pre-processing algorithm, which is now input to the Weighted Instance Selection Algorithm. As discussed in Literature Survey Chapter, there are several Instance Selection algorithms (2.2) are available. The weighted instance selection is a flavor of this. Here wDROP3 [1] or wRNN[1] is use for weighted instance selection of training set.

Algorithm 2: Weighted Instance Selection
(a)wDROP3(weighted Drecemental Reduction Optimization Procedure 3):

Input:

1. Training Set (T) = {(x_1, y_1), (x_2, y_2) … (x_n,y_n)}.
2. *k* value: number of nearest neighbors.

Output:

Reduced Training Set (S).

```
Step 1: S=T. // reduced training set initialize with T.
Step 2: wght=1/n.
Step 3: for i =1:1: n
Step 3.1: call k-NN (T(i),T, wght, k).
/* it returns k-nearest neighbor instances for each base instance T(i) */
Step 3.2: Add these k-nearest neighbor instances to associated list (AL).
Step 3.3 Initialize sum_with=0, sum_without=0;
/*sum_with= if base instance T(i) is correctly classified with its neighbor instances i.e. base instance class matches with its nearest neighbor instances class and it is sum of weight of instances which are correctly classified with base instance.*/
/*Sum_without= if base instance T(i) is temporarily remove and its nearest neighbors are correctly classified and it is sum of weight of instances which are correctly classified without base instance.*/
Step 3.4: find value of sum_with and sum_without for each base instance (T(i)).
if(sum_without>=sum_with) Remove base instance (T(i)) from S.
Step 3.5: m=size of AL. Step 3.6: for j=1:1:m
Step 3.6.1: call k-NN (AL(j), T, wght, k).
/*this returns k-nearest neighbors instance for AL(j) as base instance in T and add these neighbors to associated list(AL). Here wght is same as calculated in Step 2.*/
```

for loop end.
Step 3.7: Remove T (i) from AL. for loop end.
Step 4: Return S.

(b) wRNN (weighted Reduced Nearest Neighbor)
algorithm:

Input:
1. Training Set (T) = {(x_1, y_1), (x_2, y_2) … (x_n,y_n)}..
2. *k* value: number of nearest neighbors.

Output:
Reduced Training Set(S).
Step 1: S=T. *// initialize reduced training set with training set (T).*
Step 2: wght=1/n; *// where n is number of instances in trainings set(T), wght is weight for each instance of training set.*
Step 3: for i=1 to n
Step 3.1: Initialize sum_with=0, sum_without=0;
Step 3.2: call k-NN (T(i),T, wght, k)
// this returns k nearest neighbor instances for every base instance (T(i)).
Step 3.3: Evaluate sum_with, sum_without with each base instance (T(i)) in T.
/ sum_with: it is sum of weights of those instance which*
are misclassified with base instance i.e. base instance's class is not match with its nearest neighbor instances class then these are misclassified instances./*

*/*sum_without: if base instance is temporarily remove. it is sum of weights of those instance which are misclassified.*/*
if (sum_with>=sum_without) Remove base instance (T(i)) from S. end if
for loop end
Step 4: Return S.

Find out *k*-Nearest Neighbors for each Base Instance (T(i)) in the training set. The input for this algorithm is Training Set(T), Base Instance(T(i)), weight(wght) for each instance and k-Nearest Neighbors. The output of this algorithm is k-nearest neighbor for each base instance. In Algorithm 3 will discuss how k-NN algorithm is work.

Algorithm 3: k-Nearest Neighbor (k-NN) Algorithm

Input:
1. Training set (T) = {(x_1, y_1), (x_2, y_2) … (x_n,y_n)}.
2. K value: number of nearest neighbors.

Output: k-nearest neighbors instances for base instance. for i=1 to n *//n is number of instances in training set.*
find Euclidean distance between base instance (T(i)) to T.

/ Euclidean distance (dist(x,y)=* $\sqrt{\sum_{i=1}^{n}(X_{1i}-X_{2i})^2}$)*/

Arrange instances in ascending order of distance among them.
Select *k* nearest neighbor instances and return these instances.
for loop end

Algorithm 4: Design and Training of Ensemble of
Classifiers Algorithm:

The classifiers are group together to construct ensemble of classifiers (2.7). Here input for this algorithm is reduced training set (S). It is output of Weighted Instance Selection apply on pre-processed Training Set.

Input:
1. Reduced training set (S) = {(x''_1, y''_1), (x''_2, y''_2) …(x''_n,y''_n)}.
2. *k* value: number of nearest neighbors.
3. *M* value: number of classifiers per ensemble, it is user defined.

Output:
Misclassification
Error(misclasserr).
Step 1: load S.
*/*S is reduced training set after applying weighted instance selection algorithm (wDROP or wRNN). */*
Step 2: wght=1/n;
/*wght is weight for each
instances.*/ Step 3: for t=1 to
M
*/*where M is number of classifiers per ensemble.*/*
Step 3.1: with=0, without=0, □t=0;
*/*with= if base instance's class match with its neighbor*
instance class then increment with by one./*
*/*without= if base instance's class does not match with its neighbor instance's class then increment with by one.*/*
Step 3.2: for i=1 to n
Step 3.2.1: call k-NN (S (i), S, wght, k);
/ here S(i) is base instance.*/*

```
Step 3.2.2: find value of with and without variables
for all neighbor instances of every base instance
(train set (i)).
for loop end
Step 3.3: □t=without, sum_with=with*wght;
      if(□t>0.5)
      □(t)=0; /*weight on classifier*/
      new_wght=wght; /*Reinitialize weight
      value*/ S= Select Bootstrap Sample of a
      same size as S; else
      □=(□t /(1-□t));
new_wght=wght*□^(1-sum_with); /*Calculate new weight value*/
     endif
Step 3.4 □(t)= (log(1/□))/2.303;
mis(t)=without;
total=mis+with;
wght(t)=new_wght;
save this weights (wght) into file.
misclasserr=mis/total;
for loop end
Step 4: Return misclasserr;
```

Algorithm 5: Testing Ensemble of Classifiers:

After training of ensemble of Classifiers with reduced training set (3.3.4), system will be tested with applying testing set on this. The testing set is a partitioned of pre- processed dataset.

Input:

1. Reduced training set (S)={(x1,y1), (x2,y2),…… (xn,yn)}.
2. Testing set (Ts)={(x‟1,y‟1), (x‟2,y‟2),…… (x‟m,y‟m)}.
3. k value: number of nearest neighbor.
4. M value: number of classifiers per ensemble.

Output:

Generalization Error (generror).

```
Step 1: load S;
Step 2: load wght file. / this file stores weight for instance of every classifiers.
Step 3: for i=1:1: M
wght=wght(i).// weight value (wght(i)) for each classifier
clas=0, mis=0; Step 3.1: for j=1:1:m
// m is number of rows in testing set (Ts).
Step 3.1.1: call k-NN(Ts(j), S, wght, k); Step 3.1.2: class1=0, class2=0, class3=0;
Step 3.1.3: find k-nearest neighbor instance belong to which class, if it belong from class1 then increment class1 by one or it belong to class2 or it belong to class3.
Step 3.1.4: Do majority voting for nearest neighbor instance class and allocate class to base instance (Ts (j)).
Step 3.1.5: check base instance class with its original class. If it is match then increment clas by one else increment mis by one.
Step 3.1.6: total=mis+clas;
  for loop end
Step 3.2: generror=mis/total;
for loop end
Step 4: Return generror
```

III. TESTING AND RESULTS

Various benchmarks dataset used for experimentation. With reference to chapter 3, these dataset have taken from different domain like Iris, Satimage, Shuttle, Pima, Zoo, and Wine etc. With reference to chapter 4 there is one factor i.e. weight for instances for every classifiers. The number of classifiers per ensemble is decided at the time of training.

There are following parameters used for constructing an

Ensemble of classifiers:

*Value of „k‟: Number of Nearest Neighbor for every base instance. It is user defined. Always value of

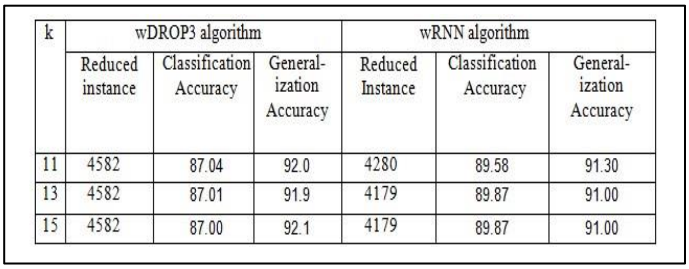
„k‟ is selected odd, so that this is automatically removes problem of overlapping.

*Value of „M‟: Number of Classifiers per Ensemble. It is also user defined. If require more accuracy of ensemble then set large value, but not greater than 30. Here use the value of M is 13.

Experimenting With Various Datasets

For experimentation of the dataset, 80% of the data considered for training set. Perform Instance selection on this training set and output of instance selection algorithm use for learning of ensemble of Classifiers and rest 20% used for testing. Brief description of all the datasets used, is given in table 1.

Table 1.
Details of Datasets used for Training and Testing of An Ensemble.



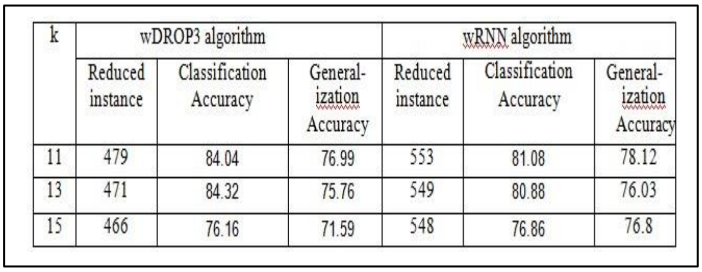
k	wDROP3 algorithm			wRNN algorithm		
	Reduced instance	Classification Accuracy	General-ization Accuracy	Reduced Instance	Classification Accuracy	General-ization Accuracy
11	4582	87.04	92.0	4280	89.58	91.30
13	4582	87.01	91.9	4179	89.87	91.00
15	4582	87.00	92.1	4179	89.87	91.00

Experimentation With Iris Dataset
Obtaining Iris training set and testing set for an Ensemble of Classifiers. Instance selection algorithm applies on 80% of Iris dataset and rest of 20% is kept for testing set. An Ensemble tested in terms of Generalization and Classification Accuracy.

The following inferences after experimenting with Iris Dataset:
1. Whenever increases value of „k": number of nearest neighbors than, In case of wDROP3 algorithm. There is slightly change in reduction of training set instances. But in case of wRNN there are major changes in reduction of training set instances. Corresponding generalization and Classification accuracy are also increasing.
2. After instance selection classification accuracy of classifier also increases.

Experimentation With Pima Indians Diabetes Dataset
Obtaining Pima Indians Diabetes Dataset training set and testing set for an Ensemble of Classifiers. Instance selection algorithm applies on 80% of Iris dataset and rest of 20% is kept for testing set. An Ensemble tested in terms of Generalization and Classification accuracy.

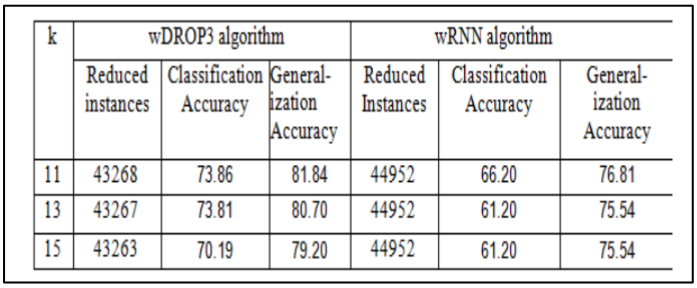
Table 2.
Show results of IRIS dataset in terms of reduced instances, Classification accuracy and Generalization accuracy by applying wDROP3 and wRNN



k	wDROP3 algorithm			wRNN algorithm		
	Reduced instance	Classification Accuracy	General-ization Accuracy	Reduced instance	Classification Accuracy	General-ization Accuracy
11	479	84.04	76.99	553	81.08	78.12
13	471	84.32	75.76	549	80.88	76.03
15	466	76.16	71.59	548	76.86	76.8

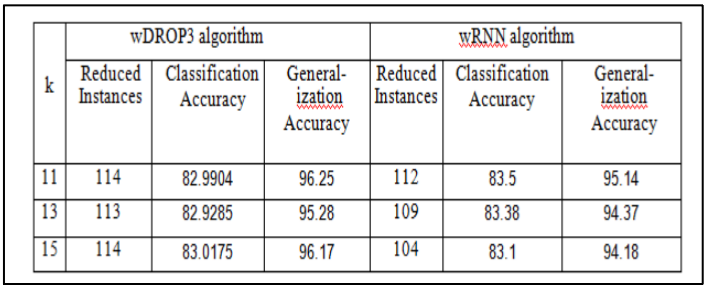
Experimentation With Zoo Dataset
Obtaining Zoo Dataset training set and testing set for an Ensemble of Classifiers. Instance selection algorithm applies on 80% of Iris dataset and rest of 20% is kept for testing set. An Ensemble tested in terms of Generalization and Classification Accuracy.

Table 3
Show results of Pima Indians Diabetes dataset



k	wDROP3 algorithm			wRNN algorithm		
	Reduced instances	Classification Accuracy	General-ization Accuracy	Reduced Instances	Classification Accuracy	General-ization Accuracy
11	43268	73.86	81.84	44952	66.20	76.81
13	43267	73.81	80.70	44952	61.20	75.54
15	43263	70.19	79.20	44952	61.20	75.54

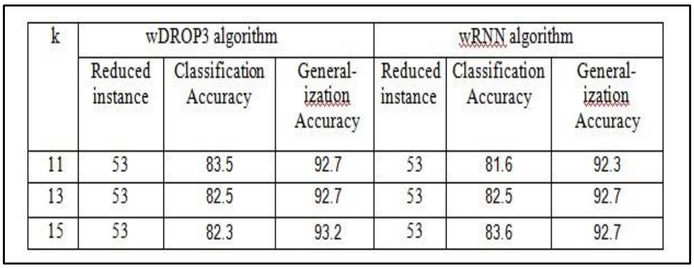
Table 4.
Show results of Zoo dataset



k	wDROP3 algorithm			wRNN algorithm		
	Reduced Instances	Classification Accuracy	General-ization Accuracy	Reduced Instances	Classification Accuracy	General-ization Accuracy
11	114	82.9904	96.25	112	83.5	95.14
13	113	82.9285	95.28	109	83.38	94.37
15	114	83.0175	96.17	104	83.1	94.18

Experimentation With Satimage Dataset
Obtaining Satimage Dataset training set and testing set for an Ensemble of Classifiers. Instance selection algorithm applies on 80% of Iris dataset and rest of 20% is kept for testing set.

Table 5.
Show results of Sat image dataset

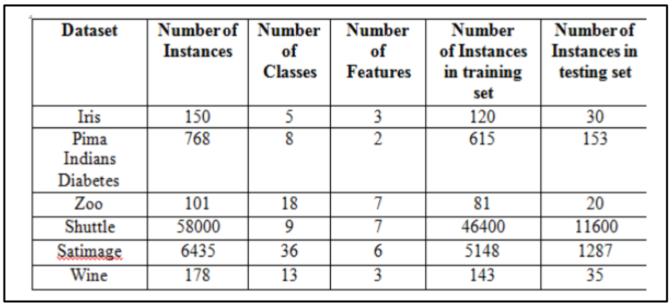


k	wDROP3 algorithm			wRNN algorithm		
	Reduced instance	Classification Accuracy	General-ization Accuracy	Reduced instance	Classification Accuracy	General-ization Accuracy
11	53	83.5	92.7	53	81.6	92.3
13	53	82.5	92.7	53	82.5	92.7
15	53	82.3	93.2	53	83.6	92.7

Experimentation with Shuttle Dataset

Obtaining Shuttle Dataset training set and testing set for an Ensemble of Classifiers.

Table 6.
Show results of Shuttle dataset

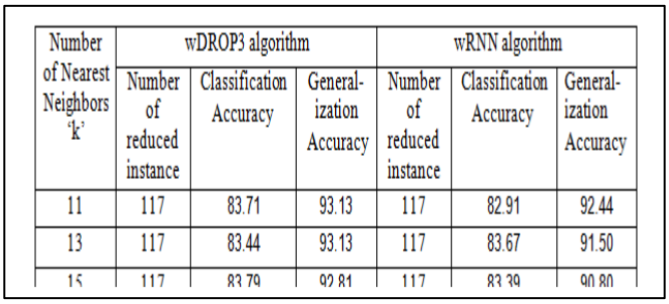


Dataset	Number of Instances	Number of Classes	Number of Features	Number of Instances in training set	Number of Instances in testing set
Iris	150	5	3	120	30
Pima Indians Diabetes	768	8	2	615	153
Zoo	101	18	7	81	20
Shuttle	58000	9	7	46400	11600
Satimage	6435	36	6	5148	1287
Wine	178	13	3	143	35

Experimentation with Wine Dataset

Obtaining Wine Dataset training set and testing set for an Ensemble of Classifiers. Instance selection algorithm applies on 80% of Iris dataset and rest of 20% is kept for testing set. An Ensemble tested in terms of Generalization and Classification Accuracy. Here „M‟: Number of Classifiers per Ensemble is 13.

Table 7
Show results of Wine dataset



Number of Nearest Neighbors 'k'	wDROP3 algorithm			wRNN algorithm		
	Number of reduced instance	Classification Accuracy	General-ization Accuracy	Number of reduced instance	Classification Accuracy	General-ization Accuracy
11	117	83.71	93.13	117	82.91	92.44
13	117	83.44	93.13	117	83.67	91.50
15	117	83.79	92.81	117	83.39	90.80

IV. CONCLUSION

The aim of the project is to design and implement an Ensemble of Classifiers using weighted Instance Selection. System developed for analyzed and design Ensemble of Classifiers using weighted instance Selection. The analysis of the problem is performed with help of system architecture and each module has been described. Design phase proceeds with various algorithms that would be implemented to build an Ensemble. The system is implemented using MATLAB. Experimentation is carried out on different benchmark datasets like Iris, Zoo, Satimage, Pima, Wine datasets has been used for training, testing and demonstrating this approach.

The following is the inferences drawn from the observation during development process:

1. Applying Weighted Instance Selection, it reduces learning time for classifier and improve classification rate of classifier.
2. By changing number of Classifiers per Ensemble (M) 0.5 to 1.5% accuracy of ensemble is change.
3. An Ensemble of Classifiers is more effective than the individual classifiers.
4. □t is observed that the range of Classification accuracy in the case of wDROP3 Algorithm lies between 70% to 87%, but in case of wRNN Algorithm lies between 61% to 89% and the Generalization accuracy in the case of wDROP3 Algorithm lies between 79% to 96%, but in wRNN Algorithm it lies between 75% to 94%.

Scope for Future Enhancements:

Following enhancements can be implementing in future to improve the accuracy of Ensemble of classifiers system:

Simultaneously Feature Selection technique with Instance Selection is useful for increasing the accuracy of system. Design Ensemble of Classifiers for Semi Supervised datasets.

REFERENCES

[1] Garcia-Pedrajas N., "Constructing Ensembles of Classifiers by Means of Weighted Instance Selection," IEEE Tran. On Neural Networks, vol. 20, pp. 258-277, Feb 2009.

[2] J. Merz C., "Using correspondence analysis to combine classifiers,"
Mach. Learn., vol. 36, no. 1, pp. 35-58, Jul 1999.

[3] G. Dietterich T., "Ensemble methods in machine learning," in
Lecture Notes in Computer Science, Springer-Verlag, vol. 1857, pp.
1-15, 2000.

[4] Freund Y. and Schapire R., "Experiments with new boosting algorithm," in Proc. 13th

International Conference Machine Learning, pp. 148-156, 1996.

[5] Liu H. and Motoda H., “On issues of instance selection,” Data Mining Knowledge Discovery, vol. 6, pp. 115-130, 2002.

[6] Kuncheva L. and J. Whitaker C., “Measures of Diversity in Classifier Ensembles,” Machine Learning, vol. 51, pp. 181–207, 2003.

[7] Cover T. and Hart P., “Nearest neighbor pattern classification,” IEEE Transaction on Information Theory, vol. 13, No. 1, pp. 21–27. Jan 1967.

[8] Jose Ramon Cano and Francisco Herrera, “Using Evolutionary Algorithms as Instance Selection for Data Reduction,” in KDD: An Experimental Study. IEEE Transactions on Evolutionary Computation, vol. 7, No. 6, Dec 2003.

[9] Jose-Federico and Olac Fuentes, “Instance Selection and Feature Weighting Using Evolutionary Algorithms,” in IEEE Proceedings of the 15th International Conference on Computing, pp. 73-79, Nov 2006.

[10] E Hart P., “The Condensed Nearest Neighbor Rule,” in IEEE Transactions on Information Theory, vol. 14, No. 3, pp. 515-516, 1968.

[11] Gates G., “The reduced nearest neighbor rule,” in IEEE Transactions on Information Theory, vol. 18, No. 3, pp. 431-433, May 1972.

[12] Jankowski, N., and Grochowski, M., “Comparison of Instances Selection Algorithms,” in ICAISC, pp. 598-603, 2004.

[13] Zhouyu Fu; Robles-Kelly, A., “An instance selection approach to Multiple Instance Learning,” in IEEE Conference on CVPR, pp. 911-918, 2009.

[14] Bhardwaj, M.; Gupta, T.; Grover, T.; Bhatnagar, V., “An efficient classifier ensemble using SVM,“ in Proceeding of International Conference on Methods and Models in Computer Science, ICM2CS, pp. 240-246, 2009.

[15] Breiman L., “Bagging Predictors” in Machine Learning, pp.123-140, 1996.

[16] Freund Y. and Schapire R., “A Short Introduction to Boosting,” in Journal of Japanese Society for Artificial Intelligence, pp. 771-780, Sep 1999.

[17] Yan R., Liu Y., Jini R. and Hauptmann A., “On Predicting Rare Classes with SVM Ensembles in Scene Classification” in IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP‟03), April 2003.

[18] Shi-Jin Wang, Avin Mathew, Yan Chen, Li-Feng Xi, Lin Ma and Jay Lee., “Empirical Analysis of Support Vector Machine Ensemble Classifiers,” in Expert Systems with Applications, pp. 6466-6476, 2009.

[19] Jiawei Han and Micheline Kamber,” Data Mining: Concepts and Techniques,” Morgan Kauffman publishers, 2006.

[20] G. Dietterich T. and Bakiri G., “Solving multiclass learning problems via error-correcting output codes,” J. Artificial Intelligence Research, vol. 2, pp. 263–286, 1995.